Oracle® Communications WebRTC Session Controller

Media Engine Object Reference

Release 7.2

E69514-02

June 2016

ORACLE®

Oracle Communications WebRTC Session Controller Media Engine Object Reference, Release 7.2

E69514-02

Contents

2 Global Commands

3 Actions A Through G

4 Actions H Through Z

5 Status Provider Show Commands A Through M

6 Status Provider Show Commands N Through Z

7 Configuring Access Objects

8 Configuring Autonomous IP Objects

9 Configuring BOOTP Client and Server Object

10 Configuring CLI Objects

11 Configuring Cluster, Box, and Interface Objects

22 Configuring IP Objects

23 Configuring Kernel Filter Rule Objects

24 Configuring Master Services Objects

29 Configuring Policy Objects

30 Configuring Preferences Objects

31 Configuring Pre-Session Configuration Objects

32 Configuring Processes Objects

33 Configuring RADIUS-Group Objects

34 Configuring Routing Objects

35 Configuring Secure Shell Objects

36 Configuring Server Objects

37 Configuring Services Routing Objects

38 Configuring Services Objects

39 Configuring Session Configuration Objects A Through M

40 Configuring Session Configuration Objects N Through Z

41 Configuring Session Configuration Pool Objects

42 Configuring Settings Objects

43 Configuring Session Initiation Protocol Objects

44 Configuring SNMP Objects

45 Configuring Static Stack Settings Objects

46 Configuring STUN Server Objects

47 Configuring Telnet Objects

48 Configuring TFTP Server Objects

49 Configuring Third-Party Call Control Server Objects

50 Configuring TLS Objects

51 Configuring User Objects

52 Configuring Virtual System Partition Objects

53 Configuring Virtual Threads Objects

54 Configuring VLAN Objects

55 Configuring VRRP Objects

56 Configuring Web Objects

57 Configuring Web-Service Objects

Preface

This reference guide describes all of the actions, show status commands, and configuration objects and properties available on the Oracle Communications WebRTC Session Controller Media Engine.

Audience

This document is for system administrators who install and configure the WebRTC Session Controller. The person installing the software should be familiar with the following topics:

- Operating system commands
- Network Management

Before reading this guide, you should have a familiarity with WebRTC Session Controller. See *Oracle Communications WebRTC Session Controller Concepts*.

Related Documents

Refer to these additional documents for related information on WebRTC Session Controller:

- *Oracle Communications WebRTC Session Controller Concepts*
- *Oracle Communications WebRTC Session Controller Installation Guide*
- *Oracle Communications WebRTC Session Controller System Administrator's Guide*

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Accessing Oracle Communications Documentation

WebRTC Session Controller documentation is available from the Oracle Documentation Web site: http://docs.oracle.com.

1

How to Use the ACLI

This chapter describes using the command line interface (CLI) to manage and monitor your WebRTC Session Controller Media Engine (ME) system.

Selecting a Management Tool

You can configure, manage, and/or monitor ME using any one of the following interfaces:

- Command line interface (CLI)
- ME Web-based management system
- SNMP (Simple Network Management Protocol)
- XML (Extensible Markup Language)
- WSDL (Web Services Description Language)

Getting Started With the CLI

The CLI is a text-based interface that allows you to manage and configure all aspects of ME. Use the CLI to create, edit, and display the ME configuration file. In addition, you can display various components of system status and data. You can access the CLI using a PC or system console, Telnet application, or Secure Shell protocols (SSH1 and SSH2).

CLI Quick Start

The CLI features a quick-start script that allows you to complete basic configuration through prompting. See the *Oracle Communications OS-E System Installation and Commissioning Guide* for more information.

Accessing the CLI

You access the CLI from a system console (serial connection), Telnet, or SSH.

Using a System Console

The following procedure lists the steps for accessing the CLI using a system console attached to ME:

1. Connect a PC to the ME device by connecting an EIA-232 (RS-232) straight-through serial cable between the console port on the system and the PC COM1 or serial port.

2. Start a terminal emulator on the PC. Tera Term Pro and HyperTerminal are popular terminal emulation programs.
3. Configure the terminal emulator for: T100 emulation, or let it autodetect the terminal type. 115200 baud, 8 data bits, no parity bit, 1 stop bit (8/N/1), no flow control.
4. Press [*Enter*] on the PC keyboard until the CLI prompt appears: NNOS-E> _

Using Telnet

The following procedure lists the steps for accessing the CLI using Telnet, the TCP/IP terminal emulation protocol:

1. Connect the ME device to a network that the Telnet-client system can reach by connecting a network cable from the Ethernet 0 port on the device to a network patch panel, Ethernet switch, or device.
2. Configure an IP address for the Ethernet port according to instructions in the *Oracle Communications WebRTC Session Controller System Administration Guide*.
3. Start the Telnet client. The default Telnet port is 23 (although you may configure the ME and client for a different port number). At the Telnet prompt, enter **open** *ipaddress*, where *ipaddress* is the IP address of the ME.
4. Log in using the user name and password that the system administrator assigned to you. username: **user1,** password: **mypassword,** NNOS-E>

Using SSH1 or SSH2 (Secure Shell)

The following procedure describes how to access the CLI using SSH1 or SSH2 (Secure Shell, a secure Telnet-like terminal emulation protocol). See Supported SSH clients for a list of supported SSH clients.

1. Connect the system to a network that the SSH client system can reach by connecting a UTP/STP Category 5 network cable from the Management port to a network patch panel or device.
2. Configure an IP address for the Management port according to the instructions in the *Oracle Communications WebRTC Session Controller System Administration Guide.*
3. Start the SSH client and specify the host name or address for the system. If using the password authentication method, the system prompts you for a password. Otherwise, if using a valid public key or no authentication is required, you connect to the system.
4. Log in using the user name and password that the system administrator assigned to you: username: **secureuser1,** password: **mypassword,** NNOS-E>

Supported SSH Clients

ME supports the following SSH clients:

- SecureCRT®, for Microsoft Windows platforms.
- PuTTY, for Microsoft Windows platforms.
- OpenSSH, for UNIX and Linux platforms.

Other SSH clients may work, however Oracle has not tested other clients with the ME at this time.

CLI Basic Concepts

The ME CLI structure consists of a command hierarchy of configurable objects and properties. When you use the CLI to create, edit, or modify the configuration, the software writes the new configuration to the default file named **cxc.cfg**.

It is important to understand the states of the ME configuration file: saved, running, and working:

1. the working config, which keeps a record of configuration edits
2. the running config, which is used by the system
3. the saved config, which the system boots from.

Before using the CLI, refer to the *Oracle Communications OS-E Management Tools Guide* for a description of each state.

CLI Structure Summary

The ME configuration file is a hierarchy of objects and properties; objects and properties describe the configuration. You use actions and commands to manipulate the data. Specifically, you open objects with the config command; you configure properties with the set command. Use the **delete** command to remove object configured settings from the configuration. The following table describes each of these elements.

Table 1–1 CLI Elements

Element	Description
Object	An *object* is a configuration container that contains properties of a specific configuration class. Objects are available at all configuration levels of the CLI; use the **config** command to open an object. Some examples of objects are: vsp dial-plan enterprise
Property	A *property* is a value for a characteristic of an object. Properties are available at all configuration levels of the CLI; use the **set** command within an object to change properties. Some examples of properties are: admin apply-to-methods domain-name
Command	A *command* is a tool used to change the configuration file. The changes do not affect ME until you save or update the configuration. Commands are available throughout all levels of the CLI. Some examples of commands are: config set reset move

Table 1–1 (Cont.) CLI Elements

Element	Description
Action	An *action* immediately acts on ME and effects one of the components. Actions are only available at the top-level prompt of the CLI. Some examples of actions are: config save dns-lookup

Editing Objects and Properties

The following table describes the commands used to edit the configuration. To enter configuration mode, enter **config** at the top-level command prompt Once in configuration mode, there are some commands (and all actions) that are no longer available to you.

Table 1–2 Configuration-Editing Commands

Command	Function
config> **config** *object*	Opens the specified object. config> **config box** config box> **config cli** config cli>
config *object*> **set** *property*	Sets the specified property, either setting the value, overwriting a previous or default value, or adding an additional value. config cli> **set display paged 24**
config *object*> **delete** *object*	Deletes your settings for the specified object from the current configuration as well as all objects (and their properties) contained within the deleted object. Anything that you can **config**, you can delete. For some services (e.g., NTP, SSH, web, etc.), **delete** kills the service. For some, **delete** returns the object to its default settings. The following example not only deletes the setting of the default-server (a property of the servers object), but also deletes all configured servers. config servers> **delete sip-gateway NNOS-E-1**
config *object*> **remove** *property*	Removes the specified property from the configuration. You can remove properties that are references to other properties (see Referencing Previously Configured Objects) and properties in a vector. (For properties that fit neither of these descriptions, you simply reset the value.) config lcs test1> **remove domain-alias[2]**
config *object*> **reset** *object*	From the specified object, resets all of the properties to their default values. Note that an objects properties also include the properties of all subobjects. For example, resetting the VSP object resets the defaults of the enterprise, servers, and directories properties, as well as the accounting and location service properties and many more. config vsp> **reset**

Saving Changes to the Configuration File

The method you use to save a configuration file effects which configuration file is modified. Saving also may move your position in the hierarchy. The following table describes each method of saving.

Table 1–3 Methods to Save Configuration File

Command	Description
config save	Executed at the top-level prompt (**NNOS-E>**), saves the running config to either the saved config, or if a pathname is supplied, to that file name. You can choose standard, verbose, or XML formats. Standard format only outputs properties with a value different from the default; verbose outputs every property. The files are functionally equivalent.
save	Executed at the config-level prompt (**config>**), works the same as config save, above.
`commit (top)`	Executed from within an object (e.g., from **config vsp>** but not **config>**), saves changes from the working config to the running config and moves you to the top-level config prompt. You must still save changes to the saved config for them to be available at the next boot.
return	Executed from within an object, saves changes made in the current object to the working config and moves you up one level in the hierarchy. Changes do not get committed to the running config until you move to the top of the hierarchy.
update	Not directly accessible, writes changes from the running config to the saved config. The command appears when you try to exit config mode, and is executed by ME when you answer yes to the question: Do you want to update the startup configuration (y or n)? This prompt only appears if you have changed the running config.

Importing and Exporting Files in XML

To save the configuration to XML, select the XML format option when using the **save** or **config save** commands. You can then import this XML file to other ME devices to create a saved configuration. This will save you time if you have identical configuration settings across ME systems in the cluster. With XML, you can also work on the configuration file offline. In the CLI, XML and "standard CLI" config files are interchangeable, and the default save location, cxc.cfg (i.e. the startup config), is the same.

The following example saves config file cxc.cfg as XML:

```
config> save xml
```

Optionally, you can supply a file name after the format to save the configuration to a named file elsewhere.

Location in the CLI Hierarchy

The CLI prompt always indicates where you are located in the CLI hierarchy. It does not show a complete object path hierarchy; instead it shows the object (and instance, if applicable) in which you are located. The following table describes the prompts that you can see.

Table 1–4 CLI Prompts

Description	Prompt	Examples
Top-level prompt (default)	**NNOS-E >**	NNOS-E> **show sessions**

Table 1–4 (Cont.) CLI Prompts

Description	Prompt	Examples
Config-level prompt: object	**config>**	config> **config vsp** config enterprise> **config servers**
Config-level prompt: object and instance	**config>**	config servers> **config lcs 1** config lcs 1>

To see a complete object path hierarchy (with property settings) from your current location, use the **show** command:

```
config enterprise> show
vsp
 enterprise
  directories
  servers
  federations
  user-group-policy[1] grpEast "vsp\policies\session-policies\policy default"
  3pcc-servers
config directories> show
vsp
 enterprise
  directories
   admin enabled
   notes-directory abc
   phantom abc1
   on-failure ignore
   resolve-on-update false

config ldap XYZinc> show
vsp
 enterprise
  directories
   ldap XYZinc
    admin enabled
    tag test
    group East
    domain xyz.com
    host 0.0.0.0
    port 389
    transport TCP
    timeout 15000 ms
    username
    password-tag
    user-settings
    group-settings
    ignore-unresolved true
    ignore-domain true
```

Navigating the CLI

You can move through the object path hierarchy in a variety of ways. In addition, the CLI returns error messages to indicate the type of “transgression” it encountered at the command line.

Moving Down Through the Hierarchy

You can move down through the CLI in two ways:

- By entering config commands individually, each on a new command line
- By entering the object path hierarchy on a single command line.

  For example:

```
NNOS-E> config
config> vsp
config vsp> config enterprise
config enterprise> config servers
config servers> config sametime company1
config sametime company1>
```

Results in the same position as:

```
NNOS-E> config vsp enterprise servers sametime company1
config sametime company1>
```

Note that if you make a mistake in your object path entry, the system moves you to the last correctly completed object. For example:

```
config> config vsp enterprise goof
Invalid class
config enterprise>
```

Moving Up the Hierarchy

There are several commands that move closer to the top-level prompt. The following table describes the commands that allow you to navigate up though the CLI.

Table 1–5 CLI Navigation Commands

Command	Function
return	Moves to the previous level in the object hierarchy. If you are at the config> prompt, **return** is not available. You must use **exit** to back out to the NNOS-E> prompt.
commit or **top**	Moves to the top level configuration prompt (config>).
exit	Leaves configuration mode and returns to the NNOS-E> prompt. If you have made any changes to the configuration, you are prompted to commit changes before you exit.

Understanding CLI Error Messages

The following table explains the some of the more common messages that the CLI returns in response to navigation problems:

Table 1–6 Common CLI Error Messages

Message	Meaning
Invalid class	The object name you supplied either does not exist or is not available at this place in the CLI hierarchy.

Table 1–6 (Cont.) Common CLI Error Messages

Message	Meaning
Invalid command	You have entered a command name that does not exist at that point in the hierarchy. This could mean that you have attempted to issue: A **set** when there are no properties to set A **config** when there are no objects beneath the current container A command that does not exist.
Invalid object	The instance of the object you tried to delete or display does not exist.
Invalid property	The property name you supplied either does not exist or is not available at this place in the CLI hierarchy.
Illegal value	The value you supplied for a property is not correct. This could occur if you entered the wrong type of value (e.g., entered "15" instead of "enabled").
Required Property is Missing	You did not supply any value for an object or property that required one.
Too many arguments	You have tried to enter more values than the current property accepts.
Value is out of range	The value you supplied for a property is not within the allowable range.

Using the CLI

The following sections describe usage techniques for working with configurations at the command line.

Entering Properties

The CLI properties, both required and optional, can be either a variable or one of multiple predefined values. The following example takes a *variable*, a value that you supply, such as 192.168.100.10:

```
set ip-address ipAddress
```

The following example takes a predefined value. Enter one:

```
set admin {enabled | disabled}
```

The following example takes both. If you select TLS for your transport protocol, you must enter the path to a certificate on the system:

```
set transport {TCP | TLS certificateReference}
```

Some objects have multiple, or compound, properties. For these, you can set more than one property for the object from the same command line configuration. In the following example, you must supply an IP address for the server, but you need not supply a transport protocol or a port, as these have default values:

```
set server ipAddress [UDP | TCP | TLS] [port]
```

It is important to note that properties are positional. If you want to set or change a property, you must supply any previous properties, even if they have default values. In the example above, if you wanted to change the port, you must first enter an IP address and transport protocol, even if you are not changing those values.

Displaying Help Text

There are several mechanisms for displaying help in the CLI. You can display a brief summary of the object or property you are setting. You can also display the list of available options from your current position.

Using the ? Character with the Config Command

Use the question mark character (?) to display a brief summary of the objects or properties available to you. For example, to determine the type of servers you can configure, with a brief description of each, enter the following:

```
config servers> config ?
configure an object

 sametime       IBM Lotus Sametime Server
 lcs            Microsoft Live Communications Server 2005
 mcs            Nortel MCS
 avaya          Avaya PBX configuration
 sip-gateway    SIP Application Server or PSTN Gateway
 h323-gateway   SIP Application Server or PSTN Gateway
 sip-host       generic SIP source/destination
 dns-group      DNS resolved server group
 sip-connection SIP connection
```

Note that you receive the same result by simply typing **config** [Enter] at the prompt.

Using the ? Character with the Set Command

When you use the question mark with the set command alone, you display abbreviated help text associated with each property within the current configuration object:

```
config transport-policy 1> set ?

 Configures a transport layer DOS policy

 description
 admin              Sets whether resource is enabled or disabled
 select             Sets the properties in addition to remoteIP to observe
 condition-list     Specifies conditional criteria on the database search
 threshold          Sets the number of unique instances to be a DOS attack
 period             Specifies how many seconds between database scans,and how many
seconds of data to analyze
```

If there are no properties to set, you receive an error:

```
config tls> set ?
Invalid command
```

Note that you receive the same result by simply typing **set** [*Enter*] at the prompt.

When you use the question mark with a specific property name, and there are predefined values for the property (an enumeration), you display the values allowed for that property:

```
config transport-policy 1> set admin ?

 Sets whether resource is enabled or disabled

 enabled  Resource is active
 disabled Resource is inactive
```

If it is not an enumeration, you receive the simple help summary:

```
config transport-policy 1> set remote-ip-netmask ?

 Sets the mask of the remote-ip
config transport-policy 1>
```

Displaying Available Commands and Properties

At any point in the CLI you can enter the question mark character to display available commands or options. Note that you can also type just the command and [*return*] in some cases (as noted) to achieve the same result. When you use the verbose option (**-v**), the system displays the properties related to the objects being shown.

Note that when you display a list of actions, and in some cases show commands, the output is limited to the services registered with ME. Actions and status providers are only available to you if the service is registered (running). For example, if you do not have the authentication master service enabled, the RADIUS actions do not display. Therefore, an action that would have no effect does not appear as available.

The following table lists the commands available from each.

Table 1–7 Displaying Available Options

Command	Function
NNOS-E> ?	Displays the list of actions and global commands available.
NNOS-E> show NNOS-E> show ?	Displays the set of valid show commands.
config> ?	Displays the list of CLI commands available at the top level of configuration mode.
config> config config> config ?	Displays the objects available for configuration from the **config>** prompt.
config *object*> ?	Displays the list of CLI commands available for the specific configuration object.
config *object*> **config** config *object*> **config** ?	Displays the objects available for configuration from the specific configuration object.
config *object*> set config *object*> set	Displays the list of properties available for configuration within the specific configuration object.

Displaying Secondary Commands

The CLI uses the concept of secondary objects and properties to filter out those items that are rarely used. These would be properties for fine-tuning a configuration, and would never be necessary for normal operations.

You cannot view secondary properties through the channels described in Displaying Available Commands and Properties. When you list the available objects or properties, those that are secondary do not display, nor do their settings display in the standard help output. Instead, you must use the **help** or verbose **show** command to see the availability. You enter these properties as you would any other. However, command completion is not implemented for them.

The following example displays the standard properties of the DOS transport policy object (help descriptions removed for clarity):

```
config transport-policy test> set ?
 Configures a transport layer DOS policy
 description
```

```
 admin
 select
 condition-list
 threshold
 period
```

Compare the standard list to list available with the **help** command:

```
config transport-policy test> help
transport-policy
 description
 admin
 select
 remote-ip-netmask
 condition-list
 threshold
 period
 inactivity-timeout
```

The additional properties of **remote-ip-netmask** and **inactivity-timeout** are now viewable. To set these, use the standard procedure:

```
config transport-policy test> set inactivity-timeout 600
```

The secondary property that was manually set now appears in the regular **show** output:

```
config transport-policy test> show
vsp
 policies
  dos-policies
   transport-policy test
    description
    admin enabled
    select
    threshold 1000 instances
    period 30 seconds
    inactivity-timeout 0 days 00:10:00
```

Using the Show Command

When you use the **show** command from the config prompt, the system displays a list of configured objects in the running configuration If you specify the verbose option (-v), the system displays the properties related to the objects being shown. Note that this does not apply to show commands available from the top-level prompt.

At the top-level prompt or the config-level prompt, the show output includes all configured objects. The following example displays all configured objects in the running configuration:

```
NNOS-E> config show
cluster
box
services
master-services
vsp
external-services
preferences
access
features
NNOS-E>
```

The following example displays all configured objects and their associated properties:

```
NNOS-E> config show -v
```

```
cluster
 name NNOS-E-1
 box 1
  admin enabled
  hostname master
  timezone eastern
  name
  description Acme Packet Net-Net OS-E
  contact Jane Doe
  location Boston, MA
  identifier 00:55:66:00:11:22
  interface eth0
   admin enabled
   mtu 1500
   arp enabled
   speed 1Gb
   duplex half
   autoneg enabled
   ip a
    admin enabled
    ip-address static 192.168.100.100/24
---More---
```

You can also display just a portion of the running config, relevant to the object in which you are currently located. (This also shows you the path to your location.) For example:

```
config active-directory company1> show
vsp
 enterprise
  directories
   active-directory company1
    admin enabled
    tag east
    domain abcCo.com
    .
    .
    .
```

The following table summarizes the configuration display commands.

Table 1–8 Displaying the Configuration

Command	Function
NNOS-E> **show** NNOS-E> **show ?**	Displays the set of valid status (show) commands.
NNOS-E> **config show** config> **show**	Displays a list of all configured objects in the running configuration.
NNOS-E> **config show -v** config> **show -v**	Displays, from the running configuration, a list of all configured objects as well as their associated properties.
config> **show object** config *object*> **show**	Displays, from the running configuration, the settings of the specified object, which includes immediate subobjects, and parentage, if applicable.
config> **show object -v** config *object*> **show -v**	Displays, from the running configuration, the contents of the specified object, all its properties (including the properties of subobjects), and parentage, if applicable.

Using Command Auto-Completion

The CLI uses a command completion feature that automatically finishes typing an object or property name for you. Pressing the keyboard [*Tab*] or [*Spacebar*] executes the completion.

Note the following requirements for using command completion:

- You must type an entry until it is minimally unique on the command line before pressing [Tab] or [*Spacebar*]. If there are two commands that begin with the same spelling, the CLI cannot differentiate between the two until you type enough letters to distinguish one from the other.

  Note: You must press [Tab] or [Spacebar] to complete the object or property name. It is not sufficient for the name to be minimally unique for execution.

- The entry must be a valid object or property in the hierarchy.
- The CLI does not auto-complete on user-configured instances or values.

The following example shows use of the auto-completion feature:

```
config box> con[Spacebar]nfig c[Spacebar]

possible completions:
 cli            CLI settings
 console        Console settings

config box> config co[Spacebar]nsole [Enter]
config console>
```

In this example, pressing the [*Spacebar*] after typing con completes the config command line. However, pressing [*Spacebar*] after typing just **c**, which was not minimally unique, resulted in prompting for further characters. Entering the minimum unique characters, and then pressing [*Spacebar*] to complete the string, allows you to press [*Enter*] to move to into console configuration mode.

If you are entering an object path, you can use the auto-complete feature for each component:

```
config> con[Spacebar]nfig v[Spacebar]sp en[Spacebar]

possible completions:
 enterprise Enterprise services
 enum       ENUM settings for phone number to URL conversion

config> config vsp en_
config> config vsp ent[Spacebar]erprise se[Spacebar]rvers
config servers>
```

Referencing Previously Configured Objects

References allow you to re-use objects in the system. Therefore, you can define an object once, and then reference it later for other uses. For example, when configuring a DOS transport policy object, you need to include a reference to a condition list. Assuming you had configured a list named "remoteIP," you would include it as follows:

```
config transport-policy> set condition-list vsp policies dos-policies
transport-condition-list remoteIP
```

Entering References in the CLI

When you reference an object, you must use the full path name to the object. You can separate objects with backslashes or spaces. For example, either of the following is acceptable:

```
config transport-policy test> set condition-list
vsp\policies\dos-policies\transport-condition-list 2

config transport-policy test> set condition-list vsp policies dos-policies
transport-condition-list 1
```

However, in some cases, quotation marks are required if you are using spaces. This would be true when the configuration is a compound: it includes a reference and other values in a single property. In the example below, you must first specify the peer type (server) and then a reference to that server:

```
config source-route 192.168.100.100> set peer server "vsp enterprise servers
sip-gateway pstn"
```

Note the following:

- When you are entering a reference, tab completion is available (unless the reference is within quotation marks).
- If you enter a path name to a reference that does not exist, the system creates an object of that name and supplies it with default values.

String Requirements for the CLI

When a property or object requires a string (e.g., user name, directory service instance, etc.) the following rules apply:

- Any printable character is acceptable.
- If the string contains delimiters (white space or \ character), it must be enclosed in double quotes "").
- If the string contains backslash character (\ ) and is therefore in quotes, you must use double backslash (\\ ) to get a single backslash in the result.
- Strings are case sensitive (i.e., admin is not the same as Admin).

The following string length limits are advised:

- Object names up to 16 characters
- Descriptions, usually in quotes, up to 32 characters
- Regular expressions up to 128 characters.

Using Regular Expressions

A regular expression is a formula for matching strings that follow some pattern. Many of the conditions and predicates require a regular expression entry. ME uses PERL-compliant regular expressions.

You can configure replacement strings in several places throughout the ME configuration. The replacement string can include references to substrings from the source string. A substring in the source string is specified by enclosing it in parenthesis ( ). It is referenced in the destination string via \1 for the first substring, \2 for the second substring, and so on.

For example, if the source string is "The Quick Brown Fox and your expression is (.*)Quick(.*), and your replacement string is \Fuzzy\2, the result is "The Fuzzy Brown Fox".

The following is a list of replacement string tokens along with examples. For each example, assume the input string is “The Quick Brown Fox”.

Table 1–9 Replacement Strings and Examples

Replacement string token	Expression	Replacement	Output
\n The nth substring in the match.	*“(.*)Quick(.*)”*	“\1Fuzzy\2”	“The Fuzzy Brown Fox”
\n\d The nth substring in the match followed by the digit d.	*“(.*)Quick(.*)”*	“\Fuzzy\2\\7”	“The Fuzzy Brown Fox7”
\n\\d The nth substring in the match followed by the dth substring in the match.	*“(.*)Quick(.*)”*	“\Fuzzy\2\\1”	“The Fuzzy Brown FoxThe”
\n\\\ The nth substring in the match followed by a single backslash character.	*“(.*)Quick(.*)”*	“\1Fuzzy\2\\\”	“The Fuzzy Brown Fox\”
\c An incrementing counter.	*“(.*)Quick(.*)”*	“\c\1\Fuzzy\2\c”	“1The 2Fuzzy Brown Fox3”
\\ A single backslash character.	*“(.*)Quick(.*)”*	“\\”	“\”

Use the **expression** action to develop and test regular expression match and replacement strings. For more information on this action see the **expression** description in the Actions chapter.

Refer to one of the following sites for more complete instructions on writing regular expressions:

- http://www.perl.com/doc/manual/html/pod/perlre.html
- http://www.oreilly.com/catalog/regex/

Using Relational Operators

In policy building, ME uses some predefined relational operators for building conditions lists and predicate statements with elements of the same type. For example, use these operators to define ranges or compare values for equality or inequality. With them, your statements form logical expressions to determine choice, such as inclusion or exclusion, and sometimes action. (For enumerated lists, IP addresses, ports, and regular expressions, you use match and exclude statements.) The operators are as follows:

- eq=equal to
- ne=not equal to
- gt=greater than

- lt=less than
- ge=greater than or equal to
- le=less than or equal to

In addition, you can use match and exclude statements to define the use of the string. A match statement includes values that match the specified string; an exclude statement ignores them.

Named Variable Support

The ME supports a generic database used to hold named variables. A named variable is a variable paired with a value through the reg-exp header code. This allows you to modify SIP message fields and CDR fields more generically.

When a session is created, a named variable list is automatically created. If any named variables are configured in the **default-session-config**, they populate this list. All variable names in the named variable list must be unique.

The ME updates the named variable list when any of the following happens.

- When the session configuration **merge-object** is set to **merge**, the named variables configured in the new session config are appended to the existing named variable list.
- When the session configuration **merge-named-variables** is set to **replace**, the existing session configuration named variables are replaced by the newly configured named variables.
- When the **header-settings > named-variable-collector** collects new named variables via the reg-exp code.
- When you specify or create a named variable list and a named variable of the same name already exists, the ME overwrites the value of the existing variable name.

Note: Variable names cannot start with a "$" and you should not use special characters such as "\", "%", "#", "!", "?", "[", "]", :&", "{", "}", "@" when naming variables.

Available accounting variables are:

- **$acct.box-id**: The box ID.
- **$acct.digest-realm**: The digest realm.
- **$acct.source-lnp**: The source LNP.
- **$acct.destination-lnp**: The destination LNP.
- **$acct.diversion-header**: The diversion header.
- **$acct.cluster-name**: The cluster name.
- **$acct.radius-caller-id**: The radius caller ID.
- **$acct.request-id**: The request ID.
- **$acct.connected**: The connected boolean.
- **$acct.scan-time**: The file scan time.
- **$acct.file-time**: The file time.

- **$acct.play-time**: The file play time.
- **$acct.disconnect-reason**: The disconnect reason.
- **$acct.final-reason-code**: The last response code.
- **$acct.post-dial-digits**: The post-dial digits.
- **$acct.source-leg-current-jitter**: The source leg's current jitter.
- **$acct.source-leg-max-jitter**: The source leg's maximum jitter.
- **$acct.destination-leg-current-jitter**: The destination leg's current jitter.
- **$acct.destination-leg-max-jitter**: The destination leg's maximum jitter.
- **$acct.source-leg-rtcp-max-jitter**: The source leg's RTCP maximum jitter.
- **$acct.destination-leg-rtcp-max-jitter**: The destination leg's RTCP maximum jitter.
- **$acct.source-leg-avg-jitter**: The source leg's RTCP average jitter.
- **$acct.destination-leg-rtcp-avg-jitter**: The destination leg's RTCP average jitter.
- **$acct.source-leg-rtcp-packets-lost**: The source leg's RTCP packets lost.
- **$acct.desination-leg-rtcp-packets-lost**: The destination leg's RTCP packets lost.
- **$acct.source-leg-rfactor**: The source leg's RFactor based on RTCP statistics.
- **$acct.destination-leg-rfactor**: The destination leg's RFactor based on RTCP statistics.
- **$acct.source-leg-mos**: The source MOS based on RTCP statistics.
- **$acct.dest-leg-mos**: The destination MOS based on RTCP statistics.

Available CDR variables are:

- **$cdr.session-id**: The unique internal session-ID.
- **$cdr.recorded**: An indicator as to whether the call was recorded or not.
- **$cdr.call-id**: The unique call ID from the user agent.
- **$cdr.to**: The To: URI.
- **$cdr.from**: The From: URI.
- **$cdr.method**: The SIP method that initiated the session.
- **$cdr.incoming-request-uri**: The request URI for the incoming leg.
- **$cdr.previous-hop-ip**: The IP address of the previous hop.
- **$cdr.previous-hop-via**: The Via: header for the previous hop.
- **$cdr.outgoing-request-uri**: The request URI for the outgoing leg.
- **$cdr.next-hop-ip**: The IP address of the next hop.
- **$cdr.next-hop-dn**: The domain name of the next hop.
- **$cdr.header**: An arbitrary header from the call.
- **$cdr.origin**: The origin header from the call.
- **$cdr.setup-time**: The time at which the call was set up.
- **$cdr.connect-time**: The time at which the call was connected.
- **$cdr.disconnect-time**: The time at which the call was disconnected.

- **$cdr.disconnect-cause**: The reason for the disconnection.
- **$cdr.duration**: Duration of the call in seconds.
- **$cdr.scp-name**: The VSP that handled the call.
- **$cdr.call-id-2**: The secondary call ID for the outgoing call.
- **$cdr.originating-gateway**: The origin Gateway.
- **$cdr.terminating-gateway**: The terminating Gateway.
- **$cdr.packets-received-on-src-leg**: The number of packets received on the source leg.
- **$cdr.packets-lost-on-src-leg**: The number of packets lost on the source leg.
- **$cdr.packets-discarded-on-src-leg**: The number of packets discarded on the source leg.
- **$cdr.pdv-on-src-leg**: The average jitter on the source leg.
- **$cdr.max-jitter-on-src-leg**: The maximum jitter on the source leg.
- **$cdr.codec-on-src-leg**: The codec on the source leg.
- **$cdr.mimetype-on-src-leg**: The mimetype on the source leg.
- **$cdr.latency-on-src-leg**: Average latency on the source leg.
- **$cdr.max-latency-on-src-leg**: Maximum latency on the source leg.
- **$cdr.packets-received-on-dest-leg**: The number of packets received on the destination leg.
- **$cdr.packets-lost-on-dest-leg**: The number of packets lost on the destination leg.
- **$cdr.packets-discarded-on-dest-leg**: The number of packets discarded on the destination leg.
- **$cdr.pdv-on-dest-leg**: The average jitter on the destination leg.
- **$cdr.max-jitter-on-dest-leg**: The maximum jitter on the destination leg.
- **$cdr.codec-on-dest-leg**: The codec on the destination leg.
- **$cdr.mimetype-on-dest-leg**: The mimetype on the destination leg.
- **$cdr.latency-on-dest-leg**: Average latency on the destination leg.
- **$cdr.max-latency-on-dest-leg**: Maximum latency on the destination leg.
- **$cdr.rfactor-on-dest-leg-times-1000**: The Rfactor on the destination leg.
- **$cdr.rfactor-on-src-leg-times-1000**: The Rfactor on the source leg.
- **$cdr.mos-fmt-on-dest-leg**: The MOS formatted on the destination leg.
- **$cdr.mos-fmt-on-src-leg**: The MOS formatted on the source leg.
- **$cdr.mos-on-dest-leg**: The MOS on the destination leg.
- $**cdr.mos-on-src-leg**: The MOS on the source leg.
- **$cdr.call-type**: The call type.
- **$cdr.disconnect-error-type**: The disconnect error type.
- **$cdr.ani**: The ANI.
- **$cdr.call-source-regid**: The source registration ID.

- **$cdr.call-dest-regid**: The destination registration ID.
- **$cdr.new-ani**: The new ANI.
- **$cdr.cdr-type**: The CDR type.
- **$cdr.hunting-attempts**: The number of hunting attempts.
- **$cdr.call-pdd**: The call PDD.
- **$cdr.call-source-realm-name**: The source realm name.
- **$cdr.call-dest-realm-name**: The destination realm name.
- **$cdr.call-dest-cr-name**: The destination CR name.
- **$cdr.inleg-peer-dest**: The in-leg peer destination.
- **$cdr.inleg-anchor-source**: The in-leg anchor source.
- **$cdr.inleg-anchor-dest**: The in-leg anchor destination.
- **$cdr.inleg-peer-sourcet**: The in-leg peer source.
- **$cdr.outleg-peer-dest**: The out-leg peer destination.
- **$cdr.outleg-anchor-source**: The out-leg anchor source.
- **$cdr.outleg-anchor-dest**: The out-leg anchor destination.
- **$cdr.outleg-peer-source**: The out-leg peer source.
- **$cdr.called-party-after-src-calling-plan**: The called party after source calling plan.
- **$cdr.last-status-message**: The last status message.
- **$cdr.last-pkt-timestamp-on-dest-leg**: The time of the last media packet on the destination leg.
- **$cdr.last-pkt-timestamp-on-src-leg**: The time of the last media packet on the source leg.
- **$cdr.setup-time-integer**: The setup-time as an integer.
- **$cdr.incoming-uri-stripped**: The stripped down version of the incoming request URI.
- **$cdr.dnis**: The DNIS.
- **$cdr.new-dnis**: The new DNIS.
- **$cdr.custom-data**: The custom data.
- **$cdr.creation-timestamp**: The time the accounting record was written to the target.

Available session variables are:

- **$session.session-id**: The session-ID for this session.
- **$session.request-id**: The request ID for this session.
- **$session.caller-id**: The caller ID for this session.
- **$session.diversion-header**: The diversion-header for this session.
- **$session.pcharging-vector**: The p-charging-vector for this session.
- **$session.digest-realm**: The digest realm for this session.
- **$session.source-lnp**: The source LNP for this session.
- **$session.destination-lnp**: The destination LNP for this session.

Available media steering variables are:

- **$inleg.source.ip:** The IP address to use for allocating media resources on the in-leg.
- **$inleg.source.port:** The port to use for allocating media resources on the in-leg.
- **$outleg.source.ip:** The IP address to use for allocating media resources on the out-leg.
- **$outleg.source.port:** The port to use for allocating media resources on the out-leg.

For more information on configuring named variables, see Configuring Session Configuration Objects.

CDR Custom Data Fields and Reserved Keywords

Using the named variable table, the ME is able to write out any information in the customData field of CDRs.

Leverage the **accounting-data** configuration object to write out any information you want in the CDR customData field. Use the **entry > value** property to reference the named variables table collected earlier via the **session-config** or **reg-exp** code. Or use it to reference a reserved keyword (reserved keywords are described later in this section).

Via the Radius Access message, VSA can also be stored and referenced in the named variable table. In the **accounting-data > entry** property, reference this information in the CDR customData field.

The ME has a list of reserved keywords you can use in CDRs or reg-exp code to access information you want. This avoids any confusion with named variables.

Reserved keywords must be referenced using the \!<*reserved-keyword*>! syntax.

The following table shows a list of keywords that extract information from either a SIP message or a session.

Table 1–10 Keywords to Extract SIP Message or Session Information

Current Abbreviation	New Custom Keyword	Information Source
\r	$MSG_REMOTE_IP	SIP Message - Remote IP
\R	$MSG_REMOTE_PORT	SIP Message - Remote Port
\p	$MSG_PRIVATE_REMOTE_IP	SIP Message - Private Remote IP
\P	$MSG_PRIVATE_REMOTE_PORT	SIP Message - Private Remote Port
\l	$MSG_LOCAL_IP	SIP Message - Local IP
\L	$MSG_LOCAL_PORT	SIP Message - Local Port
\n	$SESS_LOCAL_IN_IP	SIP Session - IP Address of the Interface For the In-Leg
\N	$SESS_LOCAL_IN_PORT	SIP Session - IP Port of the Interface For the In-Leg
\a	$SESS_LOCAL_OUT_IP	SIP Session - IP Address of the Interface for the Out-Leg
\A	$SESS_LOCAL_OUT_PORT	SIP Session - IP Port of the Interface For the Out-Leg

Table 1–10 (Cont.) Keywords to Extract SIP Message or Session Information

Current Abbreviation	New Custom Keyword	Information Source
\g	$SESS_REMOTE_IN_IP	SIP Session - IP Address of the Remote End of the In-Leg
\G	$SESS_REMOTE_IN_PORT	SIP Session - IP Port of the Remote End of the In-Leg
\d	$SESS_DEST_OUT_IP	SIP Session - Out-Leg Destination IP Address
\D	$SESS_DEST_OUT_PORT	SIP Session - Out-Leg Destination Port
\e	$SESS_PEER_IP	SIP Session - ME Peer IP Address
\E	$SESS_PEER_PORT	SIP Session - ME Peer Port
\z	$SESS_LOCAL_PEER_IP	SIP Session - ME Local Peer IP Address
\Z	$SESS_LOCAL_PEER_PORT	SIP Session - ME Local Peer Port
\t	$SESS_IN_CONTACT_HDR	SIP Session - ME's In-Leg Contact HDR
\o	$SESS_OUT_CONTACT_HDR	SIP Session - ME's Out-Leg Contact HDR
\h	$SESS_ORIG_IN_REQUEST_URI	SIP Session - Origin In Request URI
\i	$SESS_ORIG_IN_TO_URI	SIP Session - Origin In To URI
\j	$SESS_ORIG_IN_FROM_URI	SIP Session - Origin In From URI

The following table shows a list of keywords used in CDRs.

Table 1–11 Keywords Used In CDRs

Current Abbreviation	New Custom Keyword	Information Source
\b	$BOX_ID	Box Identifier
\d	$CDR_DIGEST_REALM	SIP Session - Digest Realm
\s	$CDR_SRC_LNP	SIP Session - Source LNP
\e	$CDR_DEST_LNP	SIP Session - Destination LNP
\v	$CDR_DIV_HDR	SIP Session - Diversion Header
\c	$CLUSTER_NAME	SIP Meta - Cluster Name
\r	$CDR_RADIUS_CALLER_ID	SIP Session - RADIUS Caller ID
\o	$CDR_REQUEST_ID	SIP Session - Request ID
\z	$CDR_CONNECTED	Call Data - Connected Bool
\y	$CDR_SCAN_TIME	Call Data - Scan Time
\x	$CDR_FILE_TIME	Call Data - File Time
\w	$CDR_PLAY_TIME	Call Data - Play Time
\u	$CDR_DISCONNECT_REASON	Call Data - Disconnect Reason

Table 1–11 (Cont.) Keywords Used In CDRs

Current Abbreviation	New Custom Keyword	Information Source
\t	$CDR_FINAL_REASON_CODE	SIP Session - Last Response Code
\q	$CDR_POST_DIAL_DIGITS	Call Data - Post Dial Digits
\j	$CDR_SRCLEG_CURR_JITTER	Normalized Source Leg Current Jitter
\m	$CDR_SRCLEG_MAX_JITTER	Normalized Source Leg Max Jitter
\i	$CDR_DESTLEG_CURR_JITTER	Normalized Destination Leg Current Jitter
\k	$CDR_DESTLEG_MAX_JITTER	Normalize Destination Leg Max Jitter

Custom Data Fields in ME Events

You can add custom fields into ME-generated events. By using the named variables table, you can extract information from any SIP message header and reference it in the events to add the custom information. There are three events which allow you to include this information: callCreatedEventCustom, callConnectedEventCustom, and callTerminatedEventCustom.

Add a custom data field to the callCreated, callConnected, and callTerminated events via the **third-party-call-control > custom-event-fields** object.. Within this object, define the content of that field via the **named-variable-entry** object.

There are two advanced properties under the **custom-event-fields** object, **custom-events-grouping-string** and **custom-event-delimiter**which change the characters used to associate an event's variable with its value (default is **=**) as well as the character used to separate a group's custom event entries (default is **;**). For more information on configuring custom event fields, see Configuring Session Configuration Objects.

Setting Time and Time Intervals

Several configuration objects and actions require that you set a time or time interval. The time specifies a date and time, for example, a start date. The interval reflects a number of days, hours, minutes, and seconds, for example, a refresh timer. The CLI accepts multiple entry formats for setting these intervals and displays them in the following formats:

```
master-services
 file-mirror
  external-backup
   admin enabled
   url
   refresh 0 days 00:30:00
```

You can enter a number of seconds; anything greater than 60 will be converted to *hh:mm:ss*. For example:

```
config external-backup> set refresh 120
config external-backup> show
master-services
 file-mirror
  external-backup
   admin enabled
```

```
   url
   refresh 0 days 00:02:00
```

You can enter minutes or hours explicitly. For example:

```
config external-backup> set refresh 10:30:00
config external-backup> show
master-services
 file-mirror
  external-backup
   admin enabled
   url
   refresh 0 days 10:30:00
```

To enter a number of days, enclose the string in quotation marks. You must enter the complete string for valid entry. For example:

```
config external-backup> set refresh “2 days”
Illegal value
config external-backup> set refresh “2 days 00:00:00”
config external-backup> show
master-services
 file-mirror
  external-backup
   admin enabled
   url
   refresh 2 days 00:00:00
```

In addition, the CLI accepts lexical representation for duration from the ISO 8601 extended format.

Using Automatic Values

Several properties within this object can be configured to allow ME to determine the appropriate value (a setting of **automatic**). The default value for these properties is automatically determined by ME-based on the system hardware (processor, platform, memory, etc.). Although you can do so manually, do not change the value of these properties unless instructed to do so by Technical Support. Use the **show automatic-values** command to see the actual setting on your system.

Understanding Passwords and Tags

For increased security, ME uses a two-part password mechanism for passwords shared with other devices (also known as shared secrets). You must configure both a password and a tag. An enterprise or RADIUS server, for example, probably has a configured password that ME must use to access the server. This shared secret is the password. The tag is not the password itself, but rather a user-configurable name used to access the real password. By managing shared secrets, you can maintain the secrecy of the other passwords on other devices. An administrator can set up the tags and passwords; end users can work with the configuration files and use the password tag, without having access to the password itself.

For example, if the secret for your RADIUS server is **RadPswd**, you can create a secret-tag of **myTag**. When administrators configure ME to communicate with the RADIUS server, they supply the tag, **myTag**. The real password for server authentication, **RadPswd,** remains hidden to the user. The tag can be reused when creating other configurations that use the same real password. Or, if the password is compromised, it can be changed without changing the configuration on ME.

ME uses a password store to maintain the actual password known to the other device. Using a password store allows the shared passwords to be stored outside of, and not displayed in, the configuration file. Password tags are stored in the ME configuration.

> **Note:** You can create a blank password by creating a tag without a corresponding password. This may cause problems when the external system, however, when it tries to authenticate the ME.

This password mechanism applies only to cases of ME using a shared secret. It does not apply to passwords created for users under the **access** object. (These are stored as hashed data, never as plaintext.)

Using Passwords and Tags

There are several tag properties throughout the ME configuration. These include the various external databases, enterprise servers and directories, and phone configurations, among others. The minimum password length for users is set within the **password-policy** object. When setting the Linux root password, with the **secret** action, the default minimum-length of four characters is applied. No password length minimum is enforced for other secrets that live on other machines (RADIUS servers, etc.).

There are two ways to set up a password and tag correspondence: from within the object configuration and by executing an action.

In the example below, ME creates a password tag, **blue**, for an LCS server:

```
NNOS-E> config vsp accounting database group Boston server 1
config server 1> set password-tag blue
password: ***********
 confirm: ***********
config server 1>
```

Because the tag did not already exist, the system prompted for the real password. If the tag had previously been created, the system would have simply accepted the password tag as part of the configuration.

```
config server 1> set password-tag blue
config server 1>
```

You can also create a password and tag correspondence outside of the object configuration, using the **secret** action.

```
NNOS-E> secret set red
password: *****
 confirm: *****
Success!
```

If you re-execute the **secret set** action, and supply a different password, ME overwrites the password that was associated with the tag with the new password.

Use the **show secrets** command to display configured password tags:

```
NNOS-E> show secrets

tag
---
blue
red
```

> **Note:** Passwords are maintained in a separate store; simply copying the configuration file between devices does not copy the password store. You can manually enter your passwords on each ME device. Or, you can use the secret synchronize action on the master device to copy your passwords on to other devices in the cluster.

Avoiding Configuration Conflicts with Other Users

To support two or more users editing the same copy of the configuration, ME implements a configuration conflict feature. This applies to all changes to the configuration, regardless of the tool used to make the changes (ME Management System, CLI, etc.). ME warns any user who attempts to update and save a configuration if the configuration has been saved elsewhere since it was loaded or last saved (indicating that the user does not have the most current version).

> **Note:** When using web services to update the configuration, the ME does not check for revision numbers, and therefore does not implement configuration conflict detection. When the ME receives a SetConfig message, it overwrites the running configuration regardless of whether unsaved changes have been made by other users or tools.

Revision management is the mechanism ME uses for avoiding these conflicts. Each top-level configuration object has an associated revision number. The top-level objects are: access, box, cluster, external-services, features, master-services, preferences, services, and vsp. ME increments the object revision number each time there is a saved change to that object (including one of its "children"). You can use the **show config-details** command to display the current revision number for each top-level object. Each time the box is restarted, the revision numbers revert to 1.

The example below shows sample output for a box that was newly updated, with changes made to the access and source-route objects. Because the vsp object is the top-level parent of the source-route object, ME increments the vsp count. A change that increments the revision can be an addition, modification, or deletion: anything that is then saved.

```
NNOS-E> config access
config access> delete permissions allowAll
config access> return
config> config vsp dial-plan
config dial-plan> config source-route src1
config source-route src1> set priority 500
config source-route src1> exit
Do you want to commit your changes before you exit (y or n)? y
Do you want to update the startup configuration (y or n)? y
NNOS-E> show config-details

object               revision   size       changed
------               --------   ----       -------
access               2          572        08:31:08 Mon 2008-08-28
box                  1          6376       08:11:36 Mon 2008-08-28
cluster              1          9572       08:11:36 Mon 2008-08-28
external-services    1          151        08:11:36 Mon 2008-08-28
features             1          208        08:11:36 Mon 2008-08-28
master-services      1          558        08:11:36 Mon 2008-08-28
preferences          1          120        08:11:36 Mon 2008-08-28
services             1          925        08:11:36 Mon 2008-08-28
vsp                  2          40906      08:17:49 Mon 2008-08-28
```

If another user had made changes to any part of the configuration before the sample user had saved changes, the CLI presents the following message:

```
Do you want to commit your changes before you exit (y or n)? y
Your box changes will overwrite changes made by somebody else.
Are you sure that you want to commit your changes (y or n)? n
```

All configuration changes are recorded in the event log. Use the **show event-log** command to display the contents:

```
08:16:16 Mon 2008-08-28[notice] 1:manager[system] 'vsp' configuration changed by
userA via console
```

Customizing the CLI Display

You can use CLI commands to control the display of output to your screen and set your prompt.

Customizing the Output Display

You can configure the CLI to either output text a page at a time or scroll text continuously. You do this from CLI config mode (see the **cli** command description for more information). To scroll text:

```
NNOS-E> config
config> config box
config> config cli
config cli> set display scrolled
```

To pause the display with the `--More--` prompt, enter the following command, specifying the number of lines in your display:

```
config cli> set display paged 24
```

To temporarily change the display output without changing your configuration, you can execute the following command from the top level, using the **display** action:

```
NNOS-E> display paged 24
```

When you specify paged output, the `--More--` prompt accepts the following keystrokes:

Table 1–12 Keystrokes For Display Customization

Keystroke	Result
Enter	Outputs the next line of text.
Tab	Outputs the remainder of the text.
Esc or *Q* or q	Cancels the display, outputs no more text, and returns to the prompt.
any other keystroke	Outputs the next page of text.

Resetting Your Prompt

By default, the system uses the prompt **NNOS-E>** as the top level prompt. If you'd like to change the prompt, use the following command:

```
NNOS-E> config box cli
config cli> set prompt "Think Big>"
config cli> exit
Do you want to commit your changes before you exit (y or n)? y
Do you want to update the startup configuration (y or n)? y
Think Big>
```

The prompt that you enter can be up to 64 alphanumeric characters. If you want the prompt to contain spaces, be certain to enclose it in quotation marks.

Exiting the CLI

The CLI supports several mechanisms for exiting hierarchy levels and the CLI itself. As you exit certain situations, you are prompted by the system as to whether you wish to commit changes. The following table describes the prompts and their implications:

Table 1–13 CLI Prompts and Implications

Prompt...	Occurs When...	Responses Result In....
Do you want to commit your changes before you exit (y or n)?	You have made changes to the working config and you are leaving config mode.	If you answer yes to the prompt, your changes are written to the running config (but not the saved config). They are used for your current session and until you next boot the system. If you answer no, your changes are discarded.
Do you want to update the startup configuration (y or n)?	You have committed changes from the working to running config, but have not yet saved them to the startup config for use when the system next boots.	If you answer yes, the changes are written to the startup config. If you enter no, the changes are not written, but are still in the running config. If you later answer yes to this question, without having rebooted the system, ME writes those changes (if they still exist in the running config) to the startup config.

The following table describes the commands used for exiting configuration mode levels and/or the CLI.

Table 1–14 Exit Configuration Commands

Command	Function
NNOS-E> **exit**	From the top-level prompt, **exit** exits the CLI.
NNOS-E> **quit**	From the top-level prompt, **quit** exits the CLI.
config> **exit**	From the config prompt, **exit** exits configuration mode and returns to the top level CLI prompt. If you made changes to the configuration, the CLI issues a prompt asking you if you want to commit your changes.
config *object*> **cancel**	From within object configuration mode, **cancel** discards any changes to that object during the current running configuration session and moves you up one level in the configuration hierarchy.
config *object*> **exit**	From within object configuration mode, **exit** exits configuration mode and returns to the top level CLI prompt. If you made changes to the configuration, the CLI issues a prompt asking you if you want to commit your changes.

2

Global Commands

This chapter covers the global commands. A global command is a tool used to change the configuration file. The changes do not affect the ME until you save or update the configuration. While global commands in general are available throughout all levels of the CLI, specific commands may only be available from certain prompts.

> **Note:** Although it does not affect the configuration file, the help command is also described in this chapter.

Displaying Global Commands

At any level of the CLI, typing a question mark displays the options available to you from that point in the hierarchy. When at the top-level prompt, the global commands are mixed with the actions available. For example, in the list below, only the bolded **config** command is a global commands:

```
NNOS-E> ?
--More--
 clock                    set the system time
 cls                      clear terminal screen
 cluster                  cluster debug commands
 config                   configuration commands
 cpu-monitor              Monitor CPU usage; press Esc to cancel
 csta-moc-commands        Various commands related for MOC clients
 csta-uri-normalization   Perform CSTA URI normalization operations
--More--
```

Once you are in config mode, the actions are no longer available and ME only displays the relevant global commands:

```
config> ?

config        configure an object
delete        delete an object
dump dump     the configuration database
exit          exit configuration mode
help          display all configuration settings
save          save the running configuration
show          display configuration data
?             -v to show verbose help
```

cancel

Cancels all changes to the open configuration object, restores the prior or default settings, and exits that configuration object.

Prompt

```
NNOS-E>
config>
config object>
```

Syntax

```
cancel
```

Example

ACMEPACKET#

notify all trace all aug.gz

The following CLI session configures Ethernet interface settings in the **box** configuration object; the **cancel** command ignores the new settings and reverts back to the prior or default settings.

```
config> config box
config box> config interface eth0
config interface eth0> set admin disabled
config interface eth0> set arp disabled
config interface eth0> cancel
config box>
```

commit

Executed from within an object (e.g., from **config vsp>** but not **config>**), saves changes from the working config to the running config and moves you to the top-level config prompt. You must still save changes to the saved config for them to be available at the next boot.

This command is the same as the **top (config>** command.

Prompt

```
NNOS-E>
config>
config object>
```

Syntax

```
commit
```

Example

The following example creates an IP interface, commits the change to the running config, and displays the new interface.

```
config> config box interface eth1 ip z
Creating 'ip z'
config ip z> commit
config> config box interface eth1
config interface eth1> show

box
 interface eth1
  admin enabled
```

```
mtu 1500
arp enabled
speed 1Gb
duplex full
autoneg enabled
ip d
ip z
```

config

Enters or moves deeper within ME configuration mode. The configuration mode provides access to all objects that configure and manage ME. You must have permissions for CLI access set to **normal** (standard CLI access) to use **config** and therefore change the configuration file. When you log into the ME Management System, the system automatically places you in the configuration mode. Note that there are several configuration-related actions as well.

Prompt

```
NNOS-E>
config>
config object>
```

Syntax

```
config
config objectName
```

Example

The following example illustrates entering configuration mode from the top-level prompt and then traversing the hierarchy two levels deeper.

```
NNOS-E> config
config> config vsp
config vsp> config enterprise
config enterprise> config servers
config servers>
```

delete

Deletes your settings for the specified object from the running configuration as well as all objects (and their properties) contained within the deleted object. (Use the **remove** command to remove individual properties from an object in the configuration.) You delete an object from within the parent object. All setting associated with the deleted configuration object return to the system default settings. However, to restore the object and default settings to the configuration, you must enter configuration mode for that object.

Anything that you can **config**, you can delete. For some services (e.g., NTP, SSH, web, etc.), **delete** kills the service. For some, **delete** returns the object to its default settings.

Prompt

```
NNOS-E>
config>
```
config *object*>

Syntax

```
delete objectName
```

Example

The following example deletes all configured servers.

```
config enterprise> delete servers
```

The following CLI session displays the current Telnet settings followed by the **delete** command. The **config telnet** command restores the default Telnet settings to the running configuration.

```
config> config box interface eth0 ip x
config ip x> show
box
 interface eth0
  ip x
   admin enabled
   ip-address dhcp
   geolocation 0
   metric 1
   classification-tag
   security-domain trusted
   address-scope private
   filter-intf disabled
   telnet
config ip x> config telnet
config telnet> show
telnet
 admin disabled
 max-sessions 12
 idle-timeout 60 seconds
 port 22
config ip x> delete telnet
config ip x> show
box
 interface eth0
  ip x
   admin enabled
   ip-address dhcp
   geolocation 0
   security-domain trusted
   metric 1
   classification-tag
   address-scope private
   filter-intf disabled
config ip x> config telnet
config telnet> show
box
 interface eth0
  ip x
   telnet
    admin enabled
    max-sessions 8
    idle-timeout 600 seconds
    port 23
```

dump

Displays a detailed summary of all the objects in the configuration database. This output is primarily for use by Technical Support personnel.

Prompt

```
NNOS-E>
config>
```

Syntax

```
dump
```

Example

The following is an example of the **dump** command output.

```
config> dump

object         class     id     obsrvrs    hldrs   weight
------         -----     --     -------    -----   ------
interface eth0   6       10        0        0     2601
interface eth1   6       33        0        0     1288
interface eth0   6       43        0        0      866
interface eth1   6       50        0        0      574
ip a             7       11        0        1     1149
ip b             7       27        0        1      708
ip c             7       30        0        1      628
ip d             7       34        0        1      750
ip a             7       44        0        1      798
ip b             7       51        0        1      506
ip a             7       55        0        0     1149
ip b             7       71        0        0      708
ip c             7       74        0        0      628
cluster         39       53        5        0     6214
messaging       42       52        0        0       58
messaging       42       70        0        0       58

--More--
```

exit

Leaves the current configuration mode and/or quits the CLI.

The **exit** command does the following, depending on where you are in the CLI hierarchy. If entered:

- At levels within the configuration hierarchy (**config** *object***>**), the **exit** command prompts you to commit changes to the current configuration. Type **n** (no) to discard all changes not previously saved and return to the top-level prompt. Type **y** (yes) to commit changes to the running configuration. You are then prompted to save these changes to the startup configuration. If you enter:
  - No, ME exits configuration mode and returns you to the top-level prompt. Changes are in the running config but not the startup config, and will be lost at the next system boot.
  - Yes, ME saves changes to the startup config, exits configuration mode, and returns you to the top-level prompt.

- At the top level configuration mode (**config>**), returns you to the NNOS-E> prompt.
- At the NNOS-E> prompt, exits the CLI. (This command functions the same as **quit**).

Prompt

```
NNOS-E>
config>
config object>
```

Syntax

```
exit
```

Example

```
NNOS-E> config vsp
config vsp> set local-identity abcCo.com
config vsp> exit
Do you want to commit your changes before you exit (y or n)? y
Do you want to update the startup configuration (y or n)? y
NNOS-E>

NNOS-E> config vsp
config vsp> set local-identity abcCo.com
config vsp> exit
Do you want to commit your changes before you exit (y or n)? n
NNOS-E>

config> exit
NNOS-E>

NNOS-E> exit
```

help

Displays help that is dependent on your position in the hierarchy. From the top-level prompt (**NNOS-E>** by default):

- **help** provides a list of all available commands and actions, with brief text summaries.
- **help -v** (verbose help) provides the list of commands and actions with their possible settings (as well as text summaries).

From within configuration mode (either **config>** or **config** *object>*):

- **help** lists all possible objects for configuration, and their associated properties, from your current position in the hierarchy.
- **help -v** lists objects and properties from your current position in the hierarchy with brief text summaries.

Prompt

```
NNOS-E>
config>
config object>
```

Syntax

```
help [-v]
```

Example

The following example shows a sample of each form of help.

```
NNOS-E> help
accounting         accounting commands
announce           Insert announcement on active call from a WAV file
archive            Run the archiving task for a given vsp
arena              Various arena debugging commands
arp                manage the ARP cache
assign-uri         Assign a sip URI to a user
--More--
NNOS-E> help -v

accounting                accounting commands
  copy     copy entries from one database to another
  database Examine an accounting database
   contact Contact a server
   create  Create the accounting table on a server
   count   Count the accounting records on a server
   query   Perform a query on a server
   clear   Clear a range of accounting records on a server
   reset   Reset the connections to a server
 announce                 Insert announcement on active call from a WAV file
 archive                  Run the archiving task for a given vsp
  specific Archive a specific set of sessions
   session Archive a single session
   between Archive a group of sessions between two times
--More--
```

move

Re-orders an item to a different position in the hierarchy. For some properties, the order determines the sequence in which ME processes the properties. For example, you may want to control the order in which ME checks user access. Initially, the order is determined by the order in which you configured the directories. Use the **show** command to verify the current order; the output displays an index inside a bracket. This is the number that you use in the move command.

```
config access> show

vsp
 access
  users[1]
  radius[2]
  enterprise[3]
```

The move command only appears when you have a list of items in which the order matters.

Prompt

```
config object>
```

Syntax

```
move item[originalPosition] destinationPosition
```

Example

The following example sets enterprise directory to be the first thing checked by the ME.

```
config access> show

vsp
 access
  users[1]
  radius[2]
  enterprise[3]

config access> move enterprise[3] 1
config access> show

vsp
 access
  enterprise[1]
  users[2]
  radius[3]

config access>
```

quit

Exits the CLI. The **quit** command only operates from the top-level of the CLI and has the same functionality as the **exit** command.

Prompt

```
NNOS-E>
```

Syntax

```
quit
```

Example

```
NNOS-E> config box
config box> set admin enabled
config box> top
config> exit
Do you want to update the startup configuration (y or n)? y
NNOS-E> quit
```

remove

Removes the specified property from the configuration. (Use the **delete** command to remove objects from the configuration.) You can remove properties that are references to other properties (see Referencing Previously Configured Objects) and properties in a vector. (For properties that fit neither of these descriptions, you simply reset the value.)

For properties that accept multiple values, the system lists each configured value and assigns an index (inside a bracket) to that value. Supply the property name and the index value of the instance you want to remove.

Prompt

```
NNOS-E
config>
config object>
```

Syntax

```
remove propertyName [index]
```

Example

The following example shows an LCS server configuration with three domain aliases specified. (Multiple properties have been left out of the display for clarity). The **remove** command deletes one of the aliases and changes the index of the following alias.

```
config lcs test> show

vsp
 enterprise
  servers
   lcs test
    [PROPERTIES]
    domain-alias[1] abc.com
    domain-alias[2] lmn.com
    domain-alias[3] xyz.com
--More--

config lcs test> remove domain-alias[2]
config lcs test> show

vsp
 enterprise
  servers
   lcs test
    [PROPERTIES]
    domain-alias[1] abc.com
    domain-alias[2] xyz.com
--More--
```

reset

Resets all object properties to the original default values. Note that properties also include the properties of all subobjects. For example, resetting the VSP object resets the defaults of the enterprise, servers, and directories properties, as well as the accounting and location service properties and many more. For those objects without default values (those that were created by their configuration), the **reset** command deletes the object and its subobjects.

Prompt

```
NNOS-E>
config>
```

```
config object>
```

Syntax

```
reset
```

Example

The following example resets interface eth4 to its defaults.

```
config interface eth4> show
box
 interface eth4
  admin enabled
  mtu 1200
  arp enabled
  speed 1Gb
  duplex half
  autoneg enabled
config interface eth4> reset
config interface eth4> show
box
 interface eth4
  admin enabled
  mtu 1500
  arp enabled
  speed 1Gb
  duplex full
  autoneg enabled
```

return

Moves you up one level in the configuration hierarchy and saves changes from that level to the running config.

Prompt

```
NNOS-E>
config>
```
config *object*>

Syntax

```
return
```

Example

The following CLI session configures the administrative status of Ethernet interface eth0. The **return** command (executed twice) saves the change to the running config and moves back up the hierarchy to the top level configuration mode.

```
config> config box
config box> config interface eth0
config interface eth0> set admin enabled
config interface eth0> return
config box> return
config>
```

save

Writes the running configuration to the default configuration file (/cxc/cxc.cfg), or if a path name is supplied, to that file name. You can choose standard, verbose, or XML formats. Standard format only outputs properties with a value different from the default; verbose outputs every property. By default, the configuration is saved in standard format.

You can also save your configuration file in XML format. You can then import this XML file to other ME devices to create a saved configuration. This will save you time if you have identical configuration settings across multiple devices in the cluster. With XML, you can also work on the configuration file offline. In the CLI, XML and "standard CLI" configuration files are interchangeable, and the default save location, cxc.cfg (i.e., the startup config), is the same. ME creates a numbered backup (cxc.cfg.#) with each execution, creating up to 100 backups. Files are saved in the /cxc/backup directory.

This command works the same as the **config save** action.

Prompt

```
NNOS-E>
```

Syntax

```
save [standard | verbose | [xml] [fileName]
```

Example

The following example saves the running configuration to standard CLI script:

```
NNOS-E> config
config> config box cli
config cli> set display scrolled
config cli> top
config> save
```

set

Configures properties of an object. It either sets the value, overwrites a previous or default value, or adds an additional value.

Prompt

```
NNOS-E>
config>
```
config *object*>

Syntax

```
set propertyName
```

Example

The following example sets properties of a box.

```
NNOS-E> config box
config box> set admin enabled
```

```
config box> set timezone eastern
config box> set name NNOS-E-1
config box> set contact "Jack Spratt"
```

show

Displays configuration entries and status reports for each system provider. To view a listing of the **show** commands that are available from a particular point in the command hierarchy, navigate to the command mode and enter:

show ?

To view the filter fields available for specifying an entry to display, enter a question mark after the **show** *argumentName* entry. For example:

show dial-plan ?

Using the show command also indicates your command path, because it displays the path from the top of the hierarchy to your position.

See Status Provider Show Commands for a detailed description of use of the **show** command and a description of each status provider report. See Displaying Help Text for more details on using the **help** command.

Prompt

```
NNOS-E>
config>
```
config *object*>

Syntax

```
show [objectName]
```

Example

The following examples shows different points from which you can enter the **show** command and different types of output:

```
NNOS-E> show ?
show commands
accounting-database       request information for accounting database connections
accounting-process       General statistics from the accounting process
accounting-recent        calls recently accounted
accounting-server        request information for accounting servers
accounting-status          accounting activity information
actions                    action provider statistics

--More--

NNOS-E> show interfaces

interface     name     ip-address           op-state      type
---------     ----     ----------           ---------     -----
eth0          a        192.168.215.100/24    up           public
eth0:1        b        192.168.215.110/24    down         public
eth0:2        c        192.168.215.120/24    up           public
eth1          d        192.168.216.100/24    up           public

NNOS-E> config
```

```
config> config box
config box> config cli
config cli> show

box
 cli
  prompt NNOS-E>
  banner Shutdown at 12:00 midnight
  display paged 24
```

top (config>)

Saves changes from the working config to the running config and moves you to the top-level config (**config>**) prompt. You must still save changes to the saved config for them to be available at the next boot. This command is only available from within an object.

This command is the same as the **commit** command.

Prompt

```
NNOS-E>
config>
```
config *object*>

Syntax

```
top
```

Example

The following example creates an IP interface, commits the change to the running config, and displays the new interface.

```
config> config box interface eth1 ip z
Creating 'ip z'
config ip z> commit
config> config box interface eth1
config interface eth1> show

box
 interface eth1
  admin enabled
  mtu 1500
  arp enabled
  speed 1Gb
  duplex full
  autoneg enabled
  ip d
  ip z
```

top (NNOS-E>)

Displays the processes that are top users of a category of system resources. You can select the category to sort on by typing the first letter of the name (except for Process Name, type N). The system resorts the list of processes, in descending order and at

two-second intervals, based on their CPU use. Press ESC to exit the display. This command is only available for users with debug permissions.

Prompt

```
NNOS-E>
```

Syntax

```
top
```

Example

```
NNOS-E> top
Procs:  75  Interval: 2  CPUs: 4  Mem:  4053M Swap:   0M  Kern: 12%   CPU:  0%
-------------------------------------------------------------------------------
  PID  Process Name      CPU %  Memory  Resident  Locked  Threads   Files
-------------------------------------------------------------------------------
 2976  managerl.elf       0.1%    180M       38M      0K       41      69
 4232  login              0.0%      1M      352K      0K        1       4
 4230  sshd               0.0%      4M        1M      0K        1       9
 3928  java               0.0%    581M      108M      0K       35      12
 3927  java               0.0%    456M      130M      0K       35      14
 3926  authl.elf          0.0%     42M       27M      0K       24      38
 3924  java               0.0%    450M      120M      0K       48      13
 3864  postmaster         0.0%     15M        4M      0K        1       5
 3857  postmaster         0.0%     15M        4M      0K        1       5
-------------------------------------------------------------------------------
       Totals:            0.1%   3971M     1745M      0K      494     54
```

3

Actions A Through G

This chapter covers Media Engine (ME) actions from A through G in alphabetical order. For actions from H to Z, see "Actions H Through Z." An action immediately acts on ME and effects one of the components (manipulates data), whereas objects and properties describe the configuration. Actions are only available at the top-level prompt of the CLI (or through the ME Management System **Actions** tab).

Most actions become available when ME starts, but some may only become available when the corresponding service or provider registers with ME. The registration is dependent on the configuration. For example, the **directory-reset** action cannot register if the directory service is not configured.

accounting

Sets up and debugs a remote accounting database. These are the databases identified with the **accounting database** object.

- **contact**: Contacts the specified server to verify connectivity.
- **create**: Creates the accounting table on the specified server. You can select, also, whether to execute the command. If true, ME executes the command and creates the table. If false, ME returns the SQL statement to create the table, but it does not actually create it. Use this, for example, to create the table using another tool.
- **count**: Returns a count of the number of accounting records on the specified server.
- **query**: Queries the server using any SQL query you enter. Optionally, define the number of rows you'd like to query.
- **clear**: Deletes a range of accounting records from the specified server. Define the range using the format *hh:mm:ss yyyy-mm-dd*. If you do not enter a range, ME deletes all records.
- **reset**: Resets the connection to the specified server.
- **flush**: Flushes an accounting target
- **purge**: Forces an immediate run of the purge process and cleans up all CDRs on the file system that are eligible for deletion. See the **purge-criteria** property of the **accounting** object for information on eligibility.

Syntax

```
accounting database contact accountingServerReference
accounting database create accountingServerReference [true | false]
accounting database count accountingServerReference
```

```
accounting database query accountingServerReference query [rows]
accounting database clear accountingServerReference [from] [to]
accounting database purge
```

Example

In the following examples, the actions first check connectivity to the server and then try to create a table in the database. The error indicates that the table already exists. The final example illustrates a query.

```
NNOS-E> accounting-database contact vsp accounting database group Boston server
Boston
Success!
NNOS-E> accounting database create "vsp accounting database group Boston server
Boston"
ERROR: relation "acctcallstruct" already exists
Failed to create table on server
NNOS-E> accounting database query "vsp accounting database group Boston server
Boston" "select count (*) from acctcallstruct"
<ResultSet><header>count</header><row><ResultSetRow><column>0</column></ResultSetR
ow></row></ResultSet>
```

accounting flush

The accounting flush action allows you to override existing configuration settings to manually purge accounting files. The results of this operation are logged and you can view the logs to obtain more information on files that have been flushed. These are the targets identified with the accounting object that can be specified:

- file-system: A rollover is performed when the flush is executed. This means the current file is closed and a new file has automatically been created. In the case of a postgresql file, the trailer is written to the temp file and renamed in the proper format.
- external-file-system: A file whose send has failed previously retries the flush immediately when this action is executed before waiting for the retry interval to try again.

Syntax

```
accounting flush file-system url
accounting flush external-file-system path
```

Example

```
NNOS-E> accounting flush file-system "vsp\accounting\file-system\path a"
Accounting flush has been initiated...check the event log for results.
NNOS-E>
```

accounting reapply

Re-exports accounting entries from one target to another. Use this, for example, if the connection is broken or in any event that prevented accounting records from being written to the external target. When you execute this action, ME returns the qualifying records on the file system to an unprocessed state. As a result, the records are resubmitted to the selected accounting targets. This action is limited to data that falls within the time frame set with the **accounting** object **retention-period** property.

Enter the begin and end time of the period for which you want to copy records. Enter the times in the format *hh*:*mm*:*ss yyyy-mm-dd*. If you do not enter the full format, the system completes the entry with the current date. For example, if on April 16 of 2007 you enter simply "1:00," the system uses 01:00:00 2007-04-16. Also enter a reference to one or more previously configured targets.

Syntax

```
accounting copy startTime endTime [databaseReference] [syslogReference]
[radiusReference] [filesystemReference]
```

Example

```
NNOS-E> accounting reapply 1:00:00 2008-09-01 00:00:00 2009-09-02 "vsp accounting
database group myDB"
Success!
NNOS-E>
```

add-device

Mounts or remounts data drives on to the system. You may want to use this in cases when you do a system upgrade or if you are moving a hard drive from one ME device to another. When upgrading software, data drives will no longer be mounted after the upgrade. Run this action after the restart to restore the data drives to operate with the new software. Specify the drive to add and the file system in use by that drive.

Note that the **format** action automatically performs the equivalent of the **add-device** and **mount** actions. If you have data on a drive that you want to maintain, be sure to manually execute these two actions. Do not use **format** as it will remove all data from the drive. See the *Net-Net OS-E – USB Creation and Commissioning Instructions* for a complete description of system drives and partitioning.

Syntax

```
add-device {data-1 | data-2} {reiser-3 | xfs | ext4 | ext3}
```

Example

```
NNOS-E> add-device data-2 xfs
Success!
```

announce

Plays the specified .WAV file on a connected/anchored call. Use **show active-calls** (the **sessH** field) to retrieve the session handle. You can specify in which direction to insert file, either in the inbound direction (for the caller), outbound (for the callee), or both. Additionally, you can specify when the call should terminate. If termination is set to **true**, ME hangs up the call after playing it. By default, ME does not hang up the call (terminate set to **false**). (By contrast, the **file-play** action establishes a call and hangs up after the .WAV file is finished playing.)

Syntax

```
announce sessionHandlefile filename [in | out | both] [true | false]
```

Example

```
NNOS-E> show active-calls

Active Calls:
-------------------------------------------------------------------------
SessionID: 04c298ee0bdbe1f6
From: "rick" <sip:3933@barry.acmepacket.com>;tag=102949606215911
To: <sip:1234@barry.acmepacket.com>
CallID: 34990D8B-7B2F-4E47-BFD3-BA1808A6835E@172.30.1.6
State: B2B_CONNECTED
sessH: E33E6C12
Connect:
Duration: 12 seconds
In Conn:
Out Conn:
-------------------------------------------------------------------------
Total Active Calls: 1

NNOS-E> announce 0xE33E6C12 /cxc_common/media/greeting.wav both
Success!
```

archive

Saves stored sessions for the specified VSP. The archiving action archives all data that has not been successfully archived previously. You can archive a specific session or sessions that occur between specified times using the **archive specific** action.

Use this action to initiate the backup immediately; use the **task** object to schedule automated backups. You must enable archiving with the **archiving** object for this action to succeed. Use the **show archive-result** command to view the outcome of archiving operations.

> **Note:** If you have record-based archiving configured, manual archive operations will fail. You must set the record-count property of the archiving object to 0 for this archive action to work.

If you do not enter a VSP name, the ME archives the default VSP.

Syntax

```
archive [VSPname]
```

Example

```
NNOS-E> archive
The specified vsp is not configured for archiving
NNOS-E> config vsp accounting
config accounting> config archiving
config archiving> set admin enabled
config archiving> top
config> exit
Do you want to update the startup configuration (y or n)? y
NNOS-E> archive
Success!
NNOS-E>
```

archive specific

Saves specific sessions for later retrieval. You can archive either a specific session, or a range sessions that occur between specified times. (Use the **archive** action to archive all sessions.) You must enable archiving with the **archiving** object for this action to succeed.

When you execute the action, ME creates temporary files based on the session ID, and writes them to either the file displayed in the response message (**session**) or the specified directory (**between**). Once the files are archived to ME, you can move the file(s) off the device (using TFTP, PSCP, etc.).

Select one of the following operations:

- **session**: saves a specific session, identified by its session ID number, to a system-assigned temporary file name. Or, you can specify a file name. Use the ME Management System **Call Logs** page to determine the session ID. In the CLI, there are various status providers that display the session ID as part of their output.
- **between**: saves all sessions in the specified time range to a file. Enter a time in the format *hh*:*mm*:*ss yyyy-mm-dd*. If you do not enter the full format, the system completes the entry with the current date. For example, if on April 16 of 2006 you enter simply "1:00," the system uses 01:00:00 2006-04-16. In addition, you must enter a directory to which the files can be written.

Syntax

```
archive specific {session sessionID [fileName] | between startTime endTime
directory}
```

Example

```
NNOS-E> archive specific between 8:00 22:00 /nightly_archives
Success!
NNOS-E> archive specific session 0x04C20B07E06BD88B
/tmp/arc41254.zip
NNOS-E>
```

arp

Manages the ARP cache.

- **delete**: Removes either a specific IP address or, if no address is specified, all entries from the ARP cache. Use the **show arp** command to view the contents of the cache.
- **request**: Generates an ARP request directed to the specified IP address. This option is analogous to the **ping** action; both verify connectivity to an address.
- **reply**: Generates an ARP reply for an interface (also known as a gratuitous ARP). The **arp reply** action broadcasts the specified interface to all hosts in the network. Use this to force a broadcast notifying of a change to the interface.

Syntax

```
arp {delete [ipAddress] | request ipAddress [ethX] | reply ethX}
```

Example

```
NNOS-E> show arp
ip-address      type     flags               mac-address       interface
```

```
----------      ----     -----               -----------       ---------
172.26.0.1      ETHER    COM                 00:13:1a:73:d1:c2 eth0
172.26.0.189    ETHER    COM                 00:04:23:b2:fa:8a eth0
172.26.0.252    ETHER    COM                 00:04:23:b2:f6:c6 eth0

NNOS-E> arp request 172.26.0.252
172.26.0.252 is 00:04:23:b2:f6:c6

NNOS-E> arp delete 172.26.0.252
Success!

NNOS-E> arp request 172.26.0.252
Did not receive an ARP response

NNOS-E> show interfaces
interface  name           ip-address         op-state         type
---------  ----           ----------         --------         ----
eth1       b              192.168.216.210/24 up               public
eth1.5     one            192.168.216.200/24 up               public

NNOS-E> arp reply eth1.5
Success!
```

auth request

Tests validity of a variety of different authentication types. Note that this command tests a RADIUS group; to test credentials on an individual server, use the **radius test** action.

> **Note:** This command is available for the CLI only.

Enter the following:

- -t: Specifies authentication type to test, which can be one of the following:

  Local: Perform local authentication

  RADIUS: Perform RADIUS authentication

  DIAMETER: Perform DIAMETER authentication

  Directory: Perform Directory authentication

  Accept: Accept all authentication attempts

  Reject: Reject all authentication attempts
- -g Configuration reference (“vsp\radius-group Boston”, etc. Overrides -t.

  For RADIUS and DIAMETER, must specify a group, not a server.
- -n: User name
- -p: Password
- -c: Request count. Default is 1 request.
- -r Rate, in requests/second. Default is no delay between requests.
- -u User name/password authentication. This is the default.
- -d Digest authentication. Default is user name/password authentication.

- -dr Digest realm. Default is 'testrealm'; implies '-d'.

  -dm Digest method. Default is 'INVITE'; implies '-d'.

  -du Digest URI. Default is 'sip:5555551212@example.com'; implies '-d'.

  -q Quiet mode.

Syntax

```
auth request -t [type] -g [config reference] -n [name] -p [password] -c [count] -r
[rate] -u -d -dr [realm] -dm [method] -du [URI] -q
```

Example

```
NNOS-E> auth request -g "vsp\radius-group East" -n user1 -p pswd1

Provider type:     RADIUS
Config reference:  vsp\radius-group East
User name:         user1
Password:          pswd1
Request type:      Password
Request count:     1
Rate:              0/second
Period:            0 seconds
Initiated          1 requests
Received           1 successes in 0.002 seconds (500.0/second).
Received           0 failures.
```

authentication-cache-flush

Removes all entries from the ME authentication cache, used for re-authenticating REGISTER requests.

Syntax

```
authentication-cache-flush
```

Example

```
NNOS-E> authentication-cache-flush
Success!
```

autonomous-ip

Reports whether two endpoints are within the same autonomous-ip-group. Use this action to test your autonomous IP configuration and connectivity. Note that when supplying fields for this action, you must supply all values that are configured for the endpoint. Otherwise, evaluation results will be inaccurate. For example, if the source endpoint has an associated routing-tag, you must supply that value for the *srcTag* field. Enter the following values, as applicable:

- **address**(required): Specifies the public IP address for the endpoints. This is the address of the firewall (e.g., router) or, if directly connected, the endpoint itself.
- **public-ip**: Specifies the private IP address for the endpoints. If **sip nat-translation** is enabled, this is the IP address of the phone in the private network. If **nat-translation** is disabled, this value should be 0.0.0.0 (the default).

- **srcTag** and **destTag**: Specifies the routing tag associated with the endpoint. This is a tag derived from either the **ip routing-tag** property for an interface or the **routing-settings ingress-** and/or **egress-classification-tag** in the session configuration.

The results of this action are either release or anchor. Release indicates that the endpoints are both part of a group and therefore need not be anchored. Anchor indicates that they are not part of a group so ME must anchor the call media.

Syntax

```
autonomous-ip-evaluate srcPublicIP destPublicIP [srcPrivateIP] [destPrivateIP]
[srcTag] [destTag]
```

Example

```
NNOS-E> show autonomous-ip-group

group-name   gateway   connected selfConnected
----------  -------   --------- -------------
group-1                 true      true
group-2                 true      true

NNOS-E> show autonomous-ip-route

name     match         hits
----     -----         ----
group-1  10.10.10.0/24  4
group-2  192.168.1.0/24 4
NNOS-E> autonomous-ip-evaluate 10.10.10.5 192.168.1.5
Result is anchor

NNOS-E> autonomous-ip-evaluate 10.10.10.5 10.10.10.6
Result is release
```

bandwidth-calculate

Calculates the amount of bandwidth required for a single RTP stream. You can change the overhead of the optional fields to more closely match your network scenario, and use the output for network planning. Use the s**how codec-info** status provider to determine the ptime and payload for the CODEC in use. Enter the following fields:

- **packetInterval**: Enter the ptime for the CODEC in use.
- **rtpPayload**: Enter the bytes of RTP payload per packet.
- **rtpOverhead**: Enter the bytes of RTP overhead per packet. Additional overhead might be the result of SRTP authentication or MKI.
- **ipOverhead**: Enter the bytes of IP overhead per packet. Additional overhead might be the result of running IPsec or other IP options.
- **ethOverhead**: Enter the bytes of IP overhead per packet. Additional overhead might be the result of using VLAN tags.

Syntax

```
bandwidth-calculate packetInterval rtpPayload [rtpOverhead] [ipOverhead]
[ethOverhead]
```

Example

```
NNOS-E> bandwidth-calculate1
Location at public:192.168.10.1 and private:0.0.0.0 is associated with group test
NNOS-E> autonomous-ip-lookup uri sip:00004e20@acmepacket.com
URI sip:00004e20@acmepacket.com is associated with group test
```

base-64

Encodes a data string into base-64 or decodes a data string from base-64. You can select to encode a hexadecimal or text string or decode a base-64 string into hex or text. In other words, the encode option indicates how to treat the input (as hex or a string of characters). The decode option specifies the output type.

Syntax

```
base-64 {encode-hex | encode-text | decode-hex | decode-text} data
```

Example

```
NNOS-E> base-64 encode-hex 0x1234567890
EjRWeJA=
NNOS-E> base-64 decode-hex EjRWeJA=
0x1234567890
```

call-accept

This action may be initiated in response to either a call-connecting-event or after a call-create has completed. The ME injects the initial-answer-SDP media specification into the media session and the provides the resulting actual-answer-SDP to the peer endpoint.

Syntax

```
call-accept <out-leg-handle> <media-type> <media-specification>
```

Arguments

- <*out-leg-handle*>—The handle identifying the call-leg accepting the call invitation.
- <*media-type*>—The mime type of media specification (initial-answer-SDP).
- <*media-specification*>—The answer media specification (initial-answer-SDP).

call-alerting

This action may be initiated after call-create has completed. In the case of a multi-protocol session, this action is invoked to alert the destination endpoint that an incoming call is being received.

Syntax

```
call-alerting <out-leg-handle> <media-type> [media-specification]
```

Arguments

- <*out-leg-handle*>—The handle identifying the call-leg that is alerting.

- <*media-type*>—The mime type of the media specification (initial-answer-SDP).
- [*media-specification*]—The media specification (initial-provisional-SDP).

call-control

Sets up and manages calls that are originated via ME. To do this, you must first create the call using the **call-control call** action. That action results in ME creating a handle for the call. That handle is then used to identify the call in all further call control actions.

The call-control action has been deprecated and exists for backwards compatibility only. Use the following actions to control a call:

- **call**: Creates a call. Specify the destination (to) and originating (from) SIP or TEL URI. This action results in a call handle. Optionally you can enter any of the following fields:
  - Optionally enter a request ID. This value is returned in any generated events for the call.
  - Specify whether to ring originator first (**enabled**, the default). If **disabled**, ME rings the terminator first.
  - Specify whether to place the call asynchronously. If **enabled**, ME returns an action response before the call is connected. If **disabled**, the default, the action does not return until the call connects.
  - Specify a transport protocol or use the default of **any**.
  - Reference a saved session configuration to apply to the call.
- **hold**: Places the call on hold. Specify the ME
- -created handle.
- **retrieve**: Reactivates a call that was placed on hold. Specify the ME
- -created handle.
- **transfer**: Transfers a call to a new destination. Enter the handle to identify the call and a SIP or TEL URI to identify the destination.
- **disconnect**: Ends an active call. Specify the ME
- -created handle of the call.
- **join**: Performs an "announced transfer" of a call to another call. For example, if you have a phone that has placed a call to two different destinations, ME assigns each its own handle. You can then join the two calls, letting the original originating phone drop out.
- **loop**: Places a loopback call, which loops back RTP media, allowing ME to gather call quality statistics.
- **annotate**: Attaches an annotation (text description) to the call identified by the specified call handle. Use the **get-annotation** option to retrieve the text on ME. This data is also available to other systems.
- **get-annotation**: Retrieves and displays an annotation (text description) attached to the call identified by the specified call handle. Use the **annotate** option to attach the text.
- **park**: Places a call to the specified endpoint (a single-sided phone call). Using that call handle at a later time, you can use the **connect** option to connect it to another

endpoint. Optionally, specify whether to perform this action asynchronously (see the **call** option for a description of asynchronous). You can also reference a saved session configuration to apply to the call.

- **connect**: Using the call handle of a previously parked call, connect that call to the specified endpoint. Optionally, specify whether to perform this action asynchronously (see the **call** option for a description of asynchronous). You can also attach a request ID to be included in related events.
- **terminate**: Disconnects the specified call. The other end of the call is left on hold.
- **memo-begin**: Begins recording a voice memo, which is saved as a WAV file to the specified file name. Use this with the **play** option. The system records anything spoken into the phone until the memo-end action occurs or the phone hangs up.
- **memo-end**: Ends recording of a voice memo that was started with the **memo-begin** option. Until you end the memo, the WAV file created cannot be used.
- **play**: Plays the indicated WAV file on the call specified by the call handle.
- **drop-file**: Plays the indicated WAV file on the call specified by the call handle, and parks the originator.
- **notify**: Sends the specified SIP notify message to the endpoint.
- **message**: Connect to an endpoint, play a file, and terminate the call.

Syntax

```
call-control call to from [requestId] [enabled | disabled] [enabled | disabled]
[any | UDP |TCP |TLS] [SessionConfigReference]
call-control hold handle
call-control retrieve handle
call-control transfer handleto
call-control disconnect handle
call-control join handle1 handle2
call-control loop handle
call-control annotate handletext
call-control get-annotation handle
call-control park endpoint [enabled | disabled] [SessionConfigReference]
call-control connect handle endpoint [enabled | disabled] [requestId]
call-control terminate handle
call-control memo-begin handle filename
call-control memo-end handle
call-control play handle filename
call-control drop-file handle filename
call-control notify handle event
call-control message filename endpoint [from] [requestID] [enabled | disabled]
[sessionConfigReference]
```

call-control-accept

Accepts an incoming call from an offering endpoint.

> **Note:** You must specify content-type as application/sdp and body as the SDP for the call.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- [*content-type*]: Specifies the Content-Type: for the indication.
- [*body*]: Specifies the body for the indication.

Syntax

```
call-control-accept <handle> [content-type] [body]
```

call-control-annotate

Annotates the text you specify to a call leg.

Enter the following arguments:

- <*handle*>: Identifies the handle of the call to which you want to add annotated information.
- <*text*>: The annotated text you are providing to the call leg.

Syntax

```
call-control-annotate <handle> <text>
```

call-control-attach

Attaches a call leg to an existing SIP session.

Enter the following arguments:

- <*handle*>: The handle of the endpoint to be attached.
- <*session-id*>: The session to which the endpoint is being attached.

Syntax

```
call-control-attach <handle> <session-id>
```

call-control-call

Initiates a call using To and From SIP URIs you provide.

You can set the ME to add post-dial digits to a **call-control call** action. Append the string *postd=digits* to the user portion of the **to** parameter. The following example shows the ME adding post-dial digits **12345@acmepacket.com** to a call.

Enter the following arguments:

- <*to*>: The destination SIP URI of the call.
- <*from*>: The originating SIP URI of the call.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*originatorFirst*]: When **enabled** (the default), the originating party is connected first. When **disabled**, the called party is connected first.

- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled** (the default), the ME waits for the action to complete before returning a response.
- [*transport*]: The transport method to use for the call. This can be set to **any**, **TCP**, **UDP**, or **TLS**.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

  `vsp\session-config-pool\entry MyConfig`
  Enclose the value in quotation marks when using the CLI.
- [*session-id*]: The optional ID for the session.
- [*content-type*]: The content type of the message body of the initial call.
- [*body*]: The message body of the initial call.
- [*additional-session-config*]: Any additional session-configs the ME can use to support different endpoint types. Separate session-configs with a semi-colon (;).

Syntax

```
call-control-call <to> <from> [requestId] [originatorFirst] [async] [ transport]
[config] [session-id] [content-type] [body] [additional-session-config]
```

call-control-call-to-session

Initiates a call to an existing session.

Enter the following arguments:

- <*to*>: The destination SIP URI of the session.
- <*from*>: The originating SIP URI of the session.
- <*session-id*>: The optional session ID for the session.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*originatorFirst*]: When **enabled** (the default), the originating party is connected first. When **disabled**, the called party is connected first.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled**, (the default) the ME waits for the action to complete before returning a response.
- [*transport*]: The transport method to use for the call. This can be set to **any**, **TCP**, **UDP**, or **TLS**.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

`vsp\session-config-pool\entry MyConfig`

- [*content-type*]: Specifies the Content-Type: for the indication.
- [*body*]: Specifies the body for the indication.

Syntax

```
call-control-call-to-session <to> <from> <session-id> [requestId]
[originatorFirst] [async] [transport] [config] [content-type] [body]
```

call-control-create-session

Creates a rendezvous session to which you can then add call-legs, add named-variables, or destroy the session. The ME automatically assigns the session a unique 64-bit session ID.

Enter the following arguments:

- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*to*]: The To URI for the rendezvous session.
- [*from*]: The From URI for the rendezvous session.

Syntax

```
call-control-create-session [requestId] [to] [from]
```

call-control-connect

Connects an existing parked call leg to a given endpoint. If the called party ends the call, the original call reverts back to a parked state.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*endpoint*>: The URI of the call's destination.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled**, (the default) the ME waits for the action to complete before returning a response.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*park*]: When enabled, the outgoing call leg persists and reverts to a parked state when its peer is terminated.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Syntax

```
call-control-connect <handle> <endpoint> [async] [requestId] [park] [config]
```

call-control-custom

Creates and controls established calls and overrides specific session configuration settings.

Enter the following arguments:

- <*call*>: Initiates a call using provided To and From SIP URIs.
- <*to*>: The To URI for the session.
- <*from*>: The From URI from the session.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*originatorFirst*]: When **enabled** (the default), the originating party is connected first. When **disabled**, the called party is connected first.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled**, (the default) the ME waits for the action to complete before returning a response.
- [*transport*]: The transport method to use for the call. This can be set to **any**, **TCP**, **UDP**, or **TLS**.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Enclose the value in quotation marks when using the CLI.

- [*session-id*]: The optional session ID for the session.

Syntax

```
call-control-custom <call> <to> <from> [requestId] [originatorFirst] [async]
[transport] [config] [session-id]
```

call-control-destroy-session

Destroys a rendezvous session.

Enter the following arguments:

- <*session-id*>: The session-id for the rendezvous session you are destroying. This is the unique 64 bit session ID given to the session by the ME when it was created.

Syntax

```
call-control-destroy-session <session-id>
```

call-control-detach

Detaches a call leg from an existing SIP session. If you do not specify a session ID, the ME creates a new parked session with that call leg. If you specify a session ID, the ME parks the call leg to that existing session.

Enter the following arguments:

- <*handle*>: The handle of the endpoint to be detached.

- <*session-id*>: The optional session ID to which you are parking this call leg. If you do not specify a session ID, the ME creates a new session.

Syntax

```
call-control-detach <handle> [session-id]
```

call-control-detach-to-session

Detaches a call leg and parks it to an existing specified session.

Enter the following arguments:

- <*handle*>: The handle of the endpoint from which you are detaching.
- <*session-id*>: The session ID to which you are parking this call leg.

Syntax

```
call-control-detach-to-session <handle> <session-id>
```

call-control-disconnect

Disconnects all legs of a call. The **handle** parameter can be the handle of either call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-disconnect <handle>
```

call-control-drop-file

Plays the specified audio file to the party connected to the call leg. When finished, the ME terminates the call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*filename*>: The name of the audio file where a message is recorded or from where a message is played. Audio files must be .wav files in 44.1 kHz, 16-bit mono PCM format. If you give an invalid filename, it is placed in or taken from the /cxc directory.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled** (the default), the ME waits for the action to complete before returning a response.

Syntax

```
call-control-drop-file <handle> <filename> [async]
```

call-control-fork

Adds a new endpoint's SIP URI to the parked call. The endpoint can receive media but cannot send it. Multiple endpoints can be added using this action.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*endpoint*>: The URI of the call's destination.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled** (the default), the ME waits for the action to complete before returning a response.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Enclose the value in quotation marks when using the CLI.

Syntax

```
call-control-fork <handle> <endpoint> [async] [requestId] [config]
```

call-control-get-annotation

Retrieves the annotated text given to the call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-get-annotation <handle>
```

call-control-hold

Places the specified call leg on hold. This puts the media of that call leg into send-only mode. The media of the other call leg, if present, is put into receive-only mode.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-hold <handle>
```

call-control-info-request

Sends an INFO on an existing call.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- [*info-package*]: The INFO message to send to the existing call.
- [*content-type*]: Specifies the Content-Type: for the indication.
- [*body*]: Specifies the body for the indication.

Syntax

```
call-control-info-request <handle> [info-package] [content-type] [body]
```

call-control-insert-dtmf

Inserts DTMF digits into the call leg. DTMF is inserted only into the call leg specified; the other party does not hear it.

Note also that DTMF insertion is currently only supported for two-legged calls, not parked calls.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*digits*>: Specifies the digits inserted into the call leg.
- [*volume*]: The volume of the DTMF digits, in decimals from -36 to 0. The value **1** is the default.
- [*duration*]: The duration of each digit in milliseconds, from 100 to 10000. The value **0** is the default.

Syntax

```
call-control-insert-dtmf <handle> <digits> [volume] [duration]
```

call-control-intercept

Connects an incoming call to an existing parked call.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*target*>: The handle of the parked call.

Syntax

```
call-control-intercept <handle> <target>
```

call-control-join

Connects the parties of two separate calls together. The original call legs, identified by handle1 and handle2, are disconnected.

Enter the following arguments:

- <*handle1*>: Identifies the leg of the first call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*handle2*>: Identifies the leg of the second call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-join <handle1> <handle2>
```

call-control-media-loss-start

Initiates on-demand media loss detection for the call-leg of a specified session.

Enter the following arguments:

- <*handle*>: Specify the call-leg handle on which to start monitoring for a loss of media.
- [*interval*]: Specify the interval, in seconds, to monitor for a loss of media. This value indicates how quickly the loss of media or resumption of media is detected and an event generated. The default value is **5** seconds.

Syntax

```
call-control-media-loss-start <handle> [interval]
```

call-control-media-loss-stop

Terminates the on-demand monitoring of a specified call-leg for the loss of media.

Enter the following arguments:

- <*handle*>: Specify the call-leg handle on which to stop monitoring for a loss of media.

Syntax

```
call-control-media-loss-stop <handle>
```

call-control-media-resume

Resumes the playing of an audio file on an active call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-media-resume <handle>
```

call-control-media-scanner-start

Attaches a media scanner to a call-leg (in-leg or out-leg) and begins analyzing the signal strength of the received audio. The media scanner reports events when the signal strengths detected cross from the low-threshold property settings to the high-threshold property settings, or vice-versa, and based on the configuration for the media-scanner settings. The media-scanner configuration is retrieved from the session-config associated with the target call. You can specify a named session config, which overrides the session config. The media-scanner settings configuration applied is based on the following precedence:

- **in-media-scanner-settings**: media scanner settings per in-leg call
- **out-media-scanner-settings**: out media scanner settings per out-leg call
- **session-config-media-scanner-settings**: media scanner settings per session-config
- **default-media-scanner-settings**: default property settings

The media scanner will report one of the following events when a transition has occurred:

- **Short-pause**: When a transition (for example, from stable tone to quiet) takes less time than the low-long-duration property setting, such as less than 200 milliseconds
- **Long-pause**: When a transition (for example, from stable tone to quiet) takes more time than the low-long-duration property setting, such as more than 200 milliseconds
- **Short-talk**: When the media-scanner detects talk less than the high-long-duration property setting, such as less than 900 milliseconds
- **Long-talk**: When the media-scanner detects talk longer than the high-long-duration property setting, such as longer than 900 milliseconds
- **Stable-tone**: When the media-scanner detects a constant signal over a sample interval as determined by the averaging window

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Syntax

```
call-control-media-scanner-start <handle> [config]
```

call-control-media-scanner-stop

Detaches a media scanner from a call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-media-scanner-stop <handle>
```

call-control-media-seek

Seeks a specific point in a monitored recording file.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*seek-offset*>: The offset, in milliseconds, to begin seeking. A negative value seeks backwards. Seeking starts at the spot specified by the **position** parameter.
- [*position*]: Indicates the position to begin seeking:
  - start: Seek from the start of the file. This is the default behavior.
  - current: Seek from the current position of the file.
  - end: Seek from the end of the file.

Syntax

```
call-control-media-seek <handle> <seek-offset> [position]
```

call-control-media-stop

Stops the playing of an audio file on an active call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-media-stop <handle>
```

call-control-memo-begin

Records a message from the parked party, identified by a call leg handle, and stores it in a file you specify.

> **Note:** When **cluster** is **enabled**, **master-service > file-mirror** must be enabled for it to work properly.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*filename*>: The name of the audio file where a message is recorded or from where a message is played. Audio files must be .wav files in 44.1 kHz, 16-bit mono PCM format. If you give an invalid filename, it is placed in or taken from the /cxc directory.
- [*greeting*]: A greeting file that may be applied first as a prompt.
- [*cluster*]: When **enabled**, the file is available to all MEs in the cluster. When **disabled** (the default), the file is only available on the local ME.

Syntax

```
call-control-memo-begin <handle> <filename> [greeting] [cluster]
```

call-control-memo-end

Ends a recording on the specified call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-memo-end <handle>
```

call-control-message

Connects to a given endpoint, plays the file you specify, then disconnects the call. If you specify a From URI, that appears in the From header as the calling party; if no URI is specified, the To URI is used as the From header.

Enter the following arguments:

- <*filename*>: The name of the audio file where a message is recorded or from where a message is played. Audio files must be .wav files in 44.1 kHz, 16-bit mono PCM format. If you give an invalid filename, it is placed in or taken from the /cxc directory.
- <*endpoint*>: The URI of the call's destination.
- [*from*]: The originating SIP URI of the call.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled** (the default), the ME waits for the action to complete before returning a response.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```
Enclose the value in quotation marks when using the CLI.

Syntax

```
call-control-message <filename> <endpoint> [from] [requestId] [async] [config]
```

call-control-media-pause

Pauses the playing of an audio file on an active call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-media-pause <handle>
```

call-control-message-request

Sends a MESSAGE on an existing call.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- [*content-type*]: Specifies the Content-Type: for the indication.
- [*body*]: Specifies the body for the indication.

Syntax

```
call-control-message-request <handle> [content-type] [body]
```

call-control-modify

Sends a re-INVITE on an existing call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- [*content-type*]: Specifies the Content-Type: for the indication.
- [*body*]: Specifies the body for the indication.

Syntax

```
call-control-modify <handle> [content-type] [body]
```

call-control-monitor-file

Attaches a monitor session to a recording file. A recording file can be a live session currently being recorded, an old session that was recorded, an on-demand recording of a session, or a memo actively being recorded.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*session-id*>: The optional session ID for the session.
- <*monitor-target*>: The filename of the file to be played. This can be:
  - session: A session recording file is going to be monitored.
  - memo: A memo actively being recorded is going to be monitored.
  - name: The on-demand filename specified in the call-control-record-start action is being monitored.
- [*seek-offset*]: The offset, in milliseconds, to begin seeking. A negative value seeks backwards. Seeking starts at the spot specified by the **position** parameter.
- [*position*]: Indicates the position to begin seeking.

Syntax

```
call-control-monitor-file <handle> <session-id> <monitor-target> [seek-offset]
[position]
```

call-control-monitor-session

Attaches a monitor session to a live target session. The monitor session must join the target session in-progress as it has no ability to seek forward or backward during a live recording.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*session-id*>: The optional session ID for the session.

Syntax

```
call-control-monitor-session <handle> <session-id>
```

call-control-mute-off

Turns off the mute functionality for a call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-mute-off <handle>
```

call-control-mute-on

Turns on the mute functionality for a call leg.

Enter the following arguments:

- *<handle>*: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-mute-on <handle>
```

call-control-notify

Causes a SIP NOTIFY message to be sent to the party you specify in the **handle** parameter, with the value of the Event header set by the **event** parameter.

Enter the following arguments:

- *<handle>*: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- *<event>*: The content of the Event header.

Syntax

```
call-control-notify <handle> <event>
```

call-control-notify-request

Sends a NOTIFY on an existing call.

Enter the following arguments:

- *<handle>*: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- *<event>*: The content of the event header.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled**, (the default) the ME waits for the action to complete before returning a response.
- [*content-type*]: Specifies the Content-Type: for the indication.
- [*body*]: Specifies the body for the indication.

Syntax

```
call-control-notify-request <handle> <event> [async] [content-type] [body]
```

call-control-options-request

Sends an OPTIONS on an existing call.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- [*content-type*]: Specifies the Content-Type: for the indication.
- [*body*]: Specifies the body for the indication.

Syntax

```
call-control-options-request <handle> [content-type] [body]
```

call-control-park

Creates a call to an endpoint from a given SIP URI. If you specify a From URI, it is used as the From URI in the SIP message; if you specify no From URI, the From URI is that of the given endpoint.

Enter the following arguments:

- <*endpoint*>: The URI of the call's destination.
- [*from*]: The originating SIP URI of the call.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled** (the default), the ME waits for the action to complete before returning a response.
- [*sessionId*]: The optional session ID for a rendezvous session.
- [*persist*]: When **enabled**, a connected session remains parked even when the remote endpoint disconnects.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Enclose the value in quotation marks when using the CLI.

Syntax

```
call-control-park <endpoint> [from] [requestId] [async] [sessionID] [persist]
[config]
```

call-control-park-to-session

Parks a call to an existing session.

Enter the following arguments:

- <*endpoint*>: The handle of call leg on the existing session.

- <*session-id*>: The optional session ID for the session.
- [*from*]: The From URL of the parked call.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled** (the default), the ME waits for the action to complete before returning a response.
- [*persist*]: When **enabled**, a connected session remains parked even when the remote endpoint disconnects.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Syntax

```
call-control-park-to-session <endpoint> <sessionID> [from] [requestId] [async]
[persist] [config]
```

call-control-persistence

Makes a call-leg persist in a parked state even when its peer is terminated.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*persist*>: When **enabled**, a connected session remains parked even when the remote endpoint disconnects.

Syntax

```
call-control-persistence <handle> <persist>
```

call-control-play

Plays a given audio file to the specified call leg. If two call legs are connected, the file is played to both parties.

If the **session-config > media-scanner-settings** is configured, the ME waits until the recipient (or an answering machine) has finished speaking before delivering the message. If the media scanner times out waiting for the recipient to finish speaking, the file is not played.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*filename*>: The name of the audio file where a message is recorded or from where a message is played. Audio files must be .wav files in 44.1 kHz, 16-bit mono PCM

format. If you give an invalid filename, it is placed in or taken from the /cxc directory.

- [*startTime*]: The number of milliseconds the ME waits before playing the file.
- [*async*]: When **enabled**, causes the ME to return a response immediately without waiting for the action to complete. When **disabled** (the default), the ME waits for the action to complete before returning a response.

Syntax

```
call-control-play <handle> <filename> [startTime] [async]
```

call-control-record-start

Starts the on-demand recording or a target session to a specific <*filename*> file. This recording can then be monitored via the **call-control-monitor-file** action. You can execute this command one or more times for a given target session, provided you give it a different <*filename*> each time. If a <*filename*> already exists for a given target session, the existing <*filename*> is preserved and the action fails.

Enter the following arguments:

- <*session-id*>: The optional session ID for the session.
- <*filename*>: The name of the recording for this particular target session.

Syntax

```
call-control-record-start <session-id> <filename>
```

call-control-record-stop

Stops the on-demand recording or a target session to a specific <*filename*>.

Enter the following arguments:

- <*session-id*>: The optional session ID for the session.
- <*filename*>: The name of the recording for this particular target session.

Syntax

```
call-control-record-stop <session-id> <filename>
```

call-control-redirect

Redirects an initiated call to a new endpoint, prior to the call being answered. This creates a new call leg and cancels the original one.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*endpoint*>: The URI of the call's destination.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Enclose the value in quotation marks when using the CLI.

Syntax

```
call-control-redirect <handle> <endpoint> [config]
```

call-control-reject

Rejects an incoming call from an offering endpoint.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- [*response-code*]: The SIP response code to return in response.
- [*responseText*]: Text to return in the response.

Syntax

```
call-control-reject <handle> [response-code] [responseText]
```

call-control-retrieve

Retrieves the held call leg you specify by call handle. This reconnects the call's media for that call leg and, if present, the other call leg.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-retrieve <handle>
```

call-control-ringing

Redirects an initiated call to a new endpoint, prior to the call being answered. This creates a new call leg and cancels the original one.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*endpoint*>: The URI of the call's destination.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Enclose the value in quotation marks when using the CLI.

Syntax

```
call-control-redirect <handle> <endpoint> [config]
```

call-control-send-message

Sends a message to the endpoint specified by the To URI. If you specify a From URI, it is used for the From URI. If a From URI is not specified, the From URI is the same as the To URI.

Enter the following arguments:

- <*to*>: The destination SIP URI of the call.
- <*from*>: The originating SIP URI of the call.
- [*requestId*]: A unique identifier provided by an external application. This value can be used to identify the call in subsequent events and actions. If a requestId is specified, there is a corresponding XML element in the event messages generated for the session.
- [*content-type*]: Should be set to **text/plain.**
- [*body*]: The content of the message.
- [*config*]: The **session-config** on the ME to use to process a call. Use the full path to the **session-config**. For example:

```
vsp\session-config-pool\entry MyConfig
```

Enclose the value in quotation marks when using the CLI.

Syntax

```
call-control-send-message <to> <from> [requestId] [content-type] [body] [config]
```

call-control-send-info

The **call-control-send-info** action sends an in-dialog SIP INFO request.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the registration binding (returned in SubscribeEvent).
- [*session-config*]: The session-config used for formatting INFO headers and bodies.

Syntax

```
call-control-send-info <AOR> [id] [session-config]
```

Example

```
NNOS-E>call-control-send-info http://abc.com 654321 config1
```

call-control-send-notify

The **call-control-send-notify** action sends an in-dialog SIP NOTIFY request.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the subscription binding (returned in SubscribeEvent).
- [*session-config*]: The session-config used for formatting NOTIFY headers and bodies.

Syntax

```
call-control-send-notify <AOR> [id] [session-config]
```

Example

```
NNOS-E>call-control-notify-send http://abc.com 654321 config1
```

call-control-send-options

The **call-control-send-options** action sends in-dialog SIP OPTIONS requests.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the registration binding (returned in SubscribeEvent).
- [*session-config*]: The session-config used for formatting OPTIONS headers and bodies.

Syntax

```
call-control-send-options <AOR> [id] [session-config]
```

Example

```
NNOS-E>call-control-send-options http://abc.com 654321 config1
```

call-control-send-other

The **call-control-send-other** action sends in-dialog requests for all other SIP methods which do not have a specific action.

Valid arguments for this action are:

- <*method*>: The SIP method to use in the request.
- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the registration binding (returned in RegisterEvent).
- [*session-config*]: The session-config used for formatting the request headers and bodies.

Syntax

```
call-control-send-other <method> <AOR> [id] [session-config]
```

Example

```
NNOS-E>call-control-send-other INVITE http://abc.com 654321 config1
```

call-control-send-subscribe

The **call-control-send-subscribe** action sends an in-dialog SIP SUBSCRIBE request.

When an application either sends a SUBSCRIBE request and receives a 200 OK or accepts a SUBSCRIBE request by responding with a 200 OK, the ME creates a subscription binding. This binding contains the supported events and a unique ID. The application can use this ID to distinguish between subscriptions that it created and other subscriptions created for the same AOR but a different application.

Subscription bindings are arranged in a vector under the AOR and have an expiration time that can be specified in the **sip-send-subscribe** and **call-control-send-subscribe** actions. If you do not specify an expiration time, the default is 3600 seconds.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*event*]: The name of the event type to subscribe to.
- [*accept*]: The acceptable event data to include.
- [*id*]: The unique ID that identifies the subscription binding (returned in SubscribeEvent). This value is optional on a first call but required on subsequent re-subscribes.
- [*session-config*]: The session-config used for formatting SUBSCRIBE headers and bodies.
- [*expiration*]: The expiration time of the binding, in seconds. If this value is not specified, the default is **3600** seconds.

Syntax

```
call-control-send-subscribe <AOR> [event] [accept] [id] [session-config]
[expiration]
```

Example

```
NNOS-E>call-control-send-subscribe http://abc.com INVITE 654321 config1 30
```

call-control-subscribe-request

Sends a SUBSCRIBE on an existing call.

Enter the following arguments:

- <*handle*>: Identifies the leg of a session. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- [*pkg*]: Specifies the package for the SUBSCRIBE.
- [*expires*]: The expiration value for the SUBSCRIBE.
- [*content-type*]: Specifies the Content-Type: for the indication.
- [*body*]: Specifies the body for the indication.

Syntax

```
call-control-subscribe-request <handle> [pkg] [expires] [content-type] [body]
```

call-control-terminate

Terminates the call leg indicated by the handle you specify. This parameter is only available for calls with a parked status.

Enter the following argument:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.

Syntax

```
call-control-terminate <handle>
```

call-control-transfer

Transfers the specified call leg to the specified To SIP URI. The original call leg, referred to by its handle, is disconnected. Handle can be thought of as belonging to the party doing the transfer, even though the transfer is done via a third-party action.

Enter the following arguments:

- <*handle*>: Identifies the leg of a call. Handles are returned as part of the <info> element of **call-control** results and can be used to manipulate each leg of a call independently.
- <*to*>: The destination SIP URI of the call.

Syntax

```
call-control-transfer <handle> <to>
```

call-create

Creates a media session that includes the underlying media subsystem. The ME performs a route lookup and if it determines the destination endpoint is a media endpoint, no signaling is generated. However, if the ME determines the destination endpoint is SIP, H.323, or MSS, it uses the existing infrastructure for establishing a call to an endpoint of that type to establish the out-leg for the session.

Enter the following arguments:

- <*to-uri*>—The destination endpoint.
- <*from-uri*>—The source endpoint.
- [*request-id*]—The string returned in all events associated with this session.
- <*prev-hop*>—The address of the media source.
- <*next-hop*>—The address of the media destination.
- <*media-specification*>—The media specification (initial-offer-SDP).
- [*named-configs*]—A list of one or more associated named session-configs.

Syntax

```
call-create <to-uri> <from-uri> [request-id] <prev-hop> <next-hop>
<media-specification> [named-configs] [session-config]
```

call-destroy

Terminates a media session resulting in all endpoints being disconnected.

Enter the following arguments:

- <*call-leg-handle*>—The handle of the call-leg indicating a desire to terminate the session.
- <*response-code*>—An RFC 3261 code indicating why the call is being terminated.
- [*response-phrase*]—A text string summarizing the reason the call is being terminated.

Syntax

```
call-destroy <call-leg-handle> <response-code> [response-phrase]
```

call-failover

Flushes the call-failover database of any signaling and media-session records used to maintain call state between redundant ME devices. Typically this action is used for troubleshooting purposes only, when advised to do so from customer support. This action is only available in the CLI if you have enabled the **call-failover** master service. You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
call-failover flush
```

Example

```
NNOS-E>call-failover flush
Success!
```

call-hold

This action may be initiated following a call-connected-event or after completion of a media-session-accept action. When executed, this action notifies the peer endpoint that a hold has been requested and completes immediately. No structured results are returned.

Enter the following argument:

- <*call-leg-handle*>—The handle of the call-leg indicating it has entered the held state.

Syntax

```
call-hold <call-leg-handle>
```

call-lookup

Displays, for a specified Request URI, the dial-plan settings that ME assigned, the routing arbitration process, and the selected server. This action simulates the call routing path, but does not actually trigger an outbound call. It exercises both dial plan and location database lookups. Its output indicates which dial plan entry (or location cache entry) the call would use and the next hop.

Enter a Request URI. The action returns results for any AOR that matches a configured dial plan, or you can find the URI of interest in the AOR field displayed with the **show location-cache** command.

Syntax

```
call-lookup requestUri [wildcard | named tag | anonymous] [soureceIP] [localPort]
[fromURI] [toURI] [contactHeader]
```

Example

```
NNOS-E> call-lookup sip:2078548355@elmaple.com
Resulting priority 100 sequential hunting total 1 next 0
option 0: preference 10000 bandwidth 0 cap 0 rate 0 QOS null selected
192.168.77.179
All matching routes:
route domain elmaple.com priority 100 best yes
This call will be forwarded to 192.168.77.179 transport UDP port 5060
```

call-lookup-detail

Displays, for a specified Request URI, the content of the session configuration associated with the selected **dial-plan**. If the SIP URI matches a dial-plan (source-) **route** or **arbiter**, the action returns the session configuration for that entry (or the default session configuration if the entry does not have one assigned). If the dial-plan **route** has the **peer** set to server, and the server has a session configuration, this action displays the merged session configuration. (The server session configuration takes precedence over the route session configuration.)

Syntax

```
call-lookup-detail requestUri [wildcard | named tag | anonymous] [soureceIP]
[localPort] [fromURI] [toURI] [contactHeader]
```

Example

```
NNOS-E> call-lookup-detail sip:2078548355@elmaple.com
Resulting session config merged from server/group egress:
sip-settings
mode auto-determine
transport any
port auto-determine
route-hdr none
route-hdr-use-fqdn enabled
route-hdr-uri-host
route-hdr-add-register-msg disabled
route-hdr-preprocess-strip disabled
lcs-compatibility disabled
in-server unknown
out-server unknown
```

```
utilize-contact enabled
add-contact-nat disabled
compress-signaling disabled
preserve-call-id disabled
preserve-cseq disabled
proxy-generate-100-trying
handle-3xx-locally enabled
handle-3xx-locally-lookup-original-invite disabled
session-timeout 300 seconds
session-duration-max 0 seconds
--more--
```

call-modify

This action may be initiated following either a call-connected-event or after completion of a call-accept or a call-create action. The ME injects the initial-offer-SDP media specification provided into the media session and the resulting actual-offer-SDP is forwarded to the appropriate peer in accordance with the endpoint type.

Enter the following arguments:

- <*call-leg-handle*>—The handle identifying the call-leg indicating a need for media re-negotiation.
- <*media-type*>—The mime type of media specification (initial-answer-SDP).
- <*media-specification*>—The offer media specification (initial-offer-SDP).

Syntax

```
call-modify <call-leg-handle> <media-type> <media-specification>
```

call-reject

This action may be initiated in response to a call-connecting-event and causes the media session to be terminated. The action completes immediately and does not provide structured results. The initiator of the action or signaling event creating the media session is notified appropriately.

Enter the following arguments:

- <*out-leg-handle*>—The handle identifying the call-leg that is rejecting the call invitation.
- <*response-code*>—The RFC 3261 code for why the call was rejected.
- [*response-phrase*]—A text string summarizing the reason the call is being rejected.

Syntax

```
call-reject <out-leg-handle> <response-code> [response-phrase]
```

call-retrieve

This action may be initiated following a call-held-event or after a call-hold action has completed. When executed, this action notifies the peer endpoint that a retrieve has been requested and completes immediately. No structured results are returned.

Enter the following argument:

- <*call-leg-handle*>—The handle of the holding call-leg indicating a desire to terminate the hold.

Syntax

```
call-retrieve <call-leg-handle>
```

call-retrieved

This action may be initiated following a call-retrieved-event or is invoked when the peer endpoint has indicated via signaling that the endpoint is no longer in the held state. This action completes immediately and does not return structured results.

Enter the following argument:

- <*call-leg-handle*>—The handle of the held call-leg indicating the hold has been terminated.

Syntax

```
call-retrieved <call-leg-handle>
```

cert-gen

Generates a 1024-bit key pair (public and private) using the RSA algorithm. In addition, the action generates an X.509 V3 self-signed certificate. You can subsequently use the keyfile you create to generate a Certificate Signing Request (CSR) to send to a CA. (Generate the request using the **cert-request** action and update the self-signed certificate with the validated certificate using the **cert-update** action.)

The following fields must or can be specified as part of the action to create a certificate's Distinguished Name (DN), which uniquely identifies the entity:

- **keyfile**: The name of the file to which the key and certificate will be saved.
- **password**: The password used to encrypt the private key.
- **alias**: The name given to this entry within the keyfile.
- **commonName**: The fully-qualified domain name for the site using the certificate. To include subdomain, use a wildcard (e.g., www*.companyABC.com).
- [*daysValid*]: The number of days that you are requesting from the CA that your certificate remains valid. The default is 365 days.
- [*country*]: The two-letter ISO country code for the country in which the business is registered. See the *International Organization for Standardization* for a listing of codes.
- [*alternateName*]: Any other name you want associated with the distinguished name (e.g., an email address).
- [*organization*]: The legally registered name of the business or holder of the domain name.
- [*organizationalUnit*]: A division within the organization (e.g., marketing, engineering).
- [*state*]: The name of the state, province, region, or territory in which the business is registered. Do not abbreviate this field.

- [*locality*]: The name of the city or locality in which the business is registered. Do not abbreviate this field.

When you invoke this action, you are required to enter and confirm your password. Enter the password that you used, in this action, to create the key pair.

See *RFC 1779, A String Representation of Distinguished Names*, for more information.

Syntax

```
cert-gen keyFile password alias commonName [daysValid] [country] [alternateName]
[organization] [organizationUnit] [state] [locality]
```

Example

```
NNOS-E>cert-gen keyfile1 pass1 alias1 www.test.com 730 US it@test.com TestCo MIS
Massachusetts Boston
password: ***************
confirm: ***************
Success!
```

cert-request

Generates a certification request for a private key and its certificate. You specify the keyfile and entry within it that you want to generate for, as well as a file name to write the data to. You then send that file to the CA authority. You can use the **cert-gen** action to generate an entry in the keyfile for which you can request certification, or you can use a keyfile and alias that exist on the system. Use the **cert-update** action to update the self-signed certificate with the validated certificate. Enter the following:

- **keyfile**: The name of the file containing the key and the self-signed certificate.
- **password**: The password used to encrypt the private key when it was created.
- **alias**: The entry within the certificate for which you are requesting a validation certificate.
- **csrfile**: The file to which the certification request should be saved. The data is output in PEM format. You must send the contents of this file, along with your CSR, to the CA. By default, the file is written to the /cxc/certs directory.

When you invoke this action, you are required to enter and confirm your password. Enter the password that you used, in the **cert-gen** action, to create the key pair.

Syntax

```
cert-request keyFile password alias csrFile
```

Example

```
NNOS-E>cert-request keyfile1 pass1 alias1 CSRfile
password: ***************
confirm: ***************
Success!
```

cert-update

Loads a signed certificate onto ME and associates it with the key specified by alias. Use the action to update your self-signed certificate when the CA has returned a

signed certificate. When you receive the file, you can put it anywhere on ME. However, if you store it in the directory /cxc/certs, it will be available to all boxes in the cluster. Enter the following:

- **keyfile**: The name of the file containing the key and the self-signed certificate.
- **password**: The password used to encrypt the private key when it was created.
- **alias**: The entry within the certificate for which you are creating a validated certificate.
- **certFile**: The signed certificate file returned from the CA.

You can use the **cert-gen** action to generate an entry in the keyfile for which you are requesting certification; generate the request using the **cert-request** action.

Syntax

```
cert-update keyFile [password] alias certFile
```

Example

```
NNOS-E>cert-update keyfile1 pass1 alias1 signed_1.cer
password: ***************
confirm: ***************
Success!
```

clock

Sets the ME system time. The internal clock uses a 24-hour clock, beginning at midnight starting at 00:00 hours.

Syntax

```
clock hour:minutes
```

Example

The following example sets the system clock to 2:00 p.m.

```
NNOS-E>clock 14:00
Success!
NNOS-E> show clock
time: 14:00:04 Mon 2006-01-30
uptime: 0 days 01:36:41
```

cls

Clears the window in which your CLI session is active by moving your prompt and cursor to the first line of the display. Your terminal emulation program defines at what point the screen erasure begins (and therefore, which previous data is still displayed if you scroll backward).

Syntax

```
cls
```

Example

The following example clears the terminal screen and moves your cursor and prompt to the top line:

```
NNOS-E>cls

possible completions:
clock
cls
cluster

NNOS-E>cls
```

config

Manipulates or saves the running configuration

- **merge**: Merges the specified file into the current running configuration. If any properties overlap (set in both configuration files), the values from the file being merged in take precedence.
- **replace**: Writes the specified file to the running configuration. All values are overwritten with the values of the new file.
- **save**: Writes the running configuration to the default configuration file (/cxc/cxc.cfg), or if a pathname is supplied, to that file name. You can choose standard, verbose, or XML formats. Standard format only outputs properties with a value different from the default; verbose outputs every property.

  You can also save your configuration file in XML format. You can then import this XML file to other ME systems to create a saved configuration. This will save you time if you have identical configuration settings across systems in the cluster. With XML, you can also work on the configuration file offline. In the CLI, XML and standard/verbose CLI configuration files are interchangeable (functionally equivalent), and the default save location is the same.ME creates a numbered backup (cxc.cfg.#) with each execution, creating up to 100 backups. Files are saved in the /cxc/backup directory.

  This command works the same as the **save** global command.
- **setup**: Creates a minimal operating configuration file from a setup script run from the top-level prompt. After answering a a series of questions at the command line, you can then PING the system to check connectivity. See one of the following for a complete description:
  - *Oracle Communications WebRTC Session Controller Installation Guide*
  - The appropriate Quick Installation card that ships with your hardware

Syntax

```
config merge fileName
config replace fileName
config save [standard | verbose | xml] [fileName]
config setup
```

Example

The following example saves the running configuration to XML format under a new file name.

```
NNOS-E> config merge /cxc/cxc.stock
Success!
NNOS-E> config replace /cxc/cxc.cfg
Success!
NNOS-E> config save xml cfg.xml
Success!
```

cpu-monitor

Displays the percentages of CPU usage over time. The first column of the output indicates the system time stamp of the reading. The second column indicates CPU usage as a percentage of total CPU processing power. The third column, if present, indicates the number of seconds that the measurement has been unchanged. Use the **show cpu-usage** command to display usage levels at preset intervals over time.

In the first example, below, the system is using 0% of the total CPU. The value has been unchanged for 34 seconds. In example 2, the value is continuously changing. Example 3 shows the MEII output of system under DOS attack with no DOS rules enabled to protect it. If you have a color ANSI terminal emulation program, the system will display CPU monitoring graphically, as shown in Example 4. Green bars show usage between 1% and 60%; yellow shows 61-80%, and red shows 81-100%.

Press EXC to cancel the display.

Syntax

```
cpu-monitor
```

Example

```
NNOS-E>cpu-monitor
Monitoring cpu usage; press Esc to cancel...
15:26:43 0% 34
```

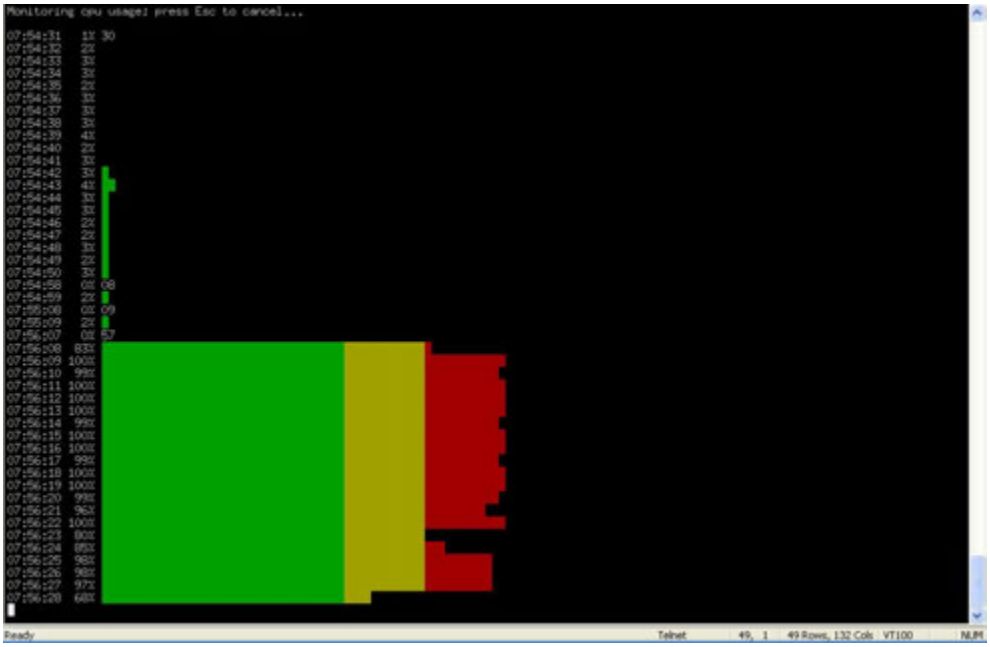
```
NNOS-E>cpu-monitor
Monitoring cpu usage; press Esc to cancel...
21:23:23 9%
21:23:24 7%
21:23:25 7%
21:23:26 6%
21:23:27 6%
21:23:28 9%
21:23:29 7%
```

The following example shows first the MEII output of a system under attack with no DOS rules enabled, and then the terminal emulation output of the same attack.

```
NNOS-E>cpu-monitor
Monitoring cpu usage; press Esc to cancel...
08:04:26 0% 13
08:04:27 34%
08:04:28 98%
08:04:29 99%
08:04:30 100%
08:04:32 100%
08:04:33 100%
08:04:34 100%
08:04:35 99%
08:04:36 100%
08:04:37 99%
```

```
08:04:38 99%
08:04:39 99%
08:04:40 77%
08:04:41 41%
08:04:42 43%
08:04:43 77%
08:04:44 55%
08:04:45 57%
08:04:46 33%
08:04:47 68%
08:04:48 39%
08:04:49 60%
08:04:50 54%
08:04:51 76%
08:04:52 56%
08:04:53 49%
08:04:54 32%
08:04:55 58%
08:04:56 51%
08:04:57 40%
08:04:58 29%
08:04:59 14%
08:05:00 12%
08:05:01 8%
08:05:02 6%
08:05:03 4%
08:05:04 2%
08:05:05 4%
08:05:06 3%
08:05:07 1% 01
08:05:08 3%
```

The following image is a terminal emulation output of a **cpu-monitor** action.



csta-moc-commands

Manages MOC clients, such as changing status for one, two, or all clients, or finding login status of clients.

- **update-moc**: Changes the status of a MOC client to the status you specify by issuing a CSTA Call Update. Enter the From URI, in the form of a telephone number (e.g., tel:+14135551212 or the normalized 5551212), to change the status of the caller. Optionally, you can simultaneously change the "callee" to the same status by entering the To URI.
- **find-moc**: Searches the list of MOC clients, listed in the CSTA and/or MOC caches, for one with the specified URI. Results of the action indicate whether or not the client is logged in.
- **reset-all-moc**: Resets the status of all the MOC clients that are currently connected to ME. Status for all is changed to Available.

Syntax

```
csta-moc-commands update-moc {do-not-disturb | call-forward | call-connected |
call-terminated | onhook | offhook} fromURI [toURI]
csta-moc-commands find-moc fromURI
csta-moc-commands reset-all-moc
```

Example

```
NNOS-E>csta-moc-commands find-moc tel:+9788235226
Device <tel:+9788235226> not found in CSTA cache.
Specified URI not found in MOC cache.
NNOS-E>csta-moc-commands find-moc 6474840
Device <6474840> found in CSTA cache.
```

csta-uri-normalization

Tests the regular expression rules for URI normalization that are configured for a 3PCC server. This is a way to reference a configured server and run intended URI normalization rules against it to verify the applicability of the normalization properties. When you execute this action, you reference a server and indicate which type of normalization rules to test. ME then applies the rules in that configuration. For a full description of using URI normalization in third-party call control, see the **3pcc-servers** object description. Note that issuing the action using the keyword **none** initiates no action.

To use this action, specify:

- A server type
- A reference to a server of that type
- The type of normalization for that server that you are testing
- The phone number to run the test against.

Syntax

```
csta-uri-normalization none
csta-uri-normalization {internal | broadworks | cisco | avaya | loopback}
3pccServerReference {incoming | outgoing | server} telNumber
```

Example

```
NNOS-E>csta-uri-normalization internal
"vsp\enterprise\3pcc-servers\internal-csta-server "Internal Server"" incoming
5551212
```

database

Deletes or cleans database records. This is for databases you configured with the master services' **database** object. You can clean or delete an entire database, or a specific table within the database. Use the **show database-tables** command to list the tables and their associated database.

Regular vacuuming is done automatically as part of the nightly maintenance and logs should show when a table is automatically vacuumed.

- **delete**: Purges the database of entries contained in the specified database, or entries in the table within the database. The **database delete** action (without qualifiers) deletes all rows in all tables in the database.
- **vacuum**: Based on the SQL VACUUM command, reclaims storage occupied by deleted entries and makes it available for re-use. The system can read and write to the table while the process is occurring.
- **vacuum-full**: Based on SQL VACUUM command, reclaims storage occupied by deleted entries and makes it available for re-use. It also does more extensive processing, however, and as a result the table is not available for read/write operations during the process. To do a periodic "global" vacuum, as Oracle recommends in the release notes, use the **database vacuum-full system** command. If you receive a message telling you a specific table needs to be vacuumed, execute the **database vacuum-full system <table>** command.
- **drop**: Deletes all data stored in the specified table and removes the table definition from the database schema.
- **repair**: Initiates database repair options. If you select the **data-recovery** option, the system recovers data that was removed by ME when it corrected a corrupted database. The **translate** option migrates earlier databases to a format compatible with release 3.2 and later.
- **initialize**: Deletes all data and reinitializes the database.
- **snapshot**: Breaks the database into smaller pieces, each starting from either the beginning of the database or the last snapshot, and ending when this action is executed (either manually or as a **task**). This results in fewer and faster disk accesses and improved performance. Use this action to manually take a snapshot, or schedule periodic snapshots with the **task** object. (See the following section Taking snapshots at regular intervals for more information.) You can access archived snapshots from the ME Management System **Call Logs** tab by clicking on the **Database Archives** link.

Select a minimum number of records in the snapshot, either an integer or the **force** or **automatic** options. If you enter a number, ME takes the snapshot if there is at least one table that has at least that many records. Otherwise, it does not execute the action for that interval. The default number of records is three million. The **force** option takes the snapshot regardless of how many records there are in the database tables. The **automatic** option skips the snapshot if no table has 3 million records or more.

Taking Snapshots at Regular Intervals

To take a snapshot at regular intervals, use the task object. You can, for example, schedule the ME to take a snapshot every four hours. The content of the snapshot will depend on the settings of the snapshot option, but will contain the data that was written to the database from the end of the previous snapshot to four hours forward. Each snapshot will contain only those four hours worth of data. Queries can then be

performed on snapshot segments instead of the whole database. When creating a database snapshot, ME:

1. Takes the current timestamp and uses it as the part of the snapshot data directory name;
2. Dumps the whole database up to the timestamp to a new database stored in the named data directory;
3. Deletes from the running database all records with a timestamp up to the above set timestamp;
4. Performs a vacuum analyze on the running database to reclaim disk space.

While a snapshot is in progress, all database reads and writes, as well as DOS queries, are performed as usual.

Syntax

```
database delete database [table]
database vacuum database [table]
database vacuum-full database [table]
database drop} database [table]
database repair {translate | data-recovery}
database initialize [database-path]
database snapshot database {integer | force | automatic}
```

Example

```
NNOS-E>database vacuum status cpuusage
Are you sure (y or n)? y
Success!
NNOS-E>
```

database-backup

Executes a database backup or restore operation. A backup saves the database to the path you specify. The restore actions loads the specified database file from the location you specify to ME. Any restore action adds entries from that file to the database. (If your goal is to overwrite the database, then you should first use the **database delete** action and then use the **database-backup restore** action.)

When you supply a path name, you are also giving a name to the database file. The ME saves the file to /cxc/pg_dump/*name*. Do not specify a path name unless it begins with /cxc/pg_dump/. For example, if you specify db1, the ME saves it to /cxc/pg_dump/db1. Or, you could specify, /cxc/pg_dump/db1. However, if you specify /cxc/db1, the operation will fail.

Note that by default the ME uses BZIP2 compression. This format is optimized for size, but can take longer to produce. If you would prefer to use GZIP compression, which is faster but results in a 30-40% larger archive, you can do so by supplying the **gz** suffix when you initiate the action. The following table provides an example of compression suffixes:

Table 3–1 Compression Suffixes

Enter this file name at the command line...	Get an archive of this type...
DBbackup	DBbackup.bz2
DBbackup.gz	DBbackup.gz

Syntax

```
database-backup {backup | restore} {log | system | status dos | directory |
accounting} databasePath
```

Example

```
NNOS-E>database-backup restore system /cxc/pg_dump/sysDB
Are you sure (y or n)? y
Starting database restore as a background operation.
 -- this may take a very long time --
Please check database-maintenance-status for notification when this operation is
complete.
NNOS-E>show database-maintenance-status
maintenance-status: idle
```

directory-reset

Resets the enterprise directory, causing the ME to reread the directory and update the user information. Use this action when you have added users and want the ME to retrieve the new entries.

Enter the name of the VSP that houses the directory. In addition, you can set a directory purge action of true or false:

- **true**: Clears out the contents of the database and then repopulates it.
- **false**: Updates the database but leaves users that are no longer in the directory itself in the database.

If you do not enter a VSP name, the system uses the VSP **default**. For the directory-reset action, the default purge action is **true**.

You can cancel this action when it is in progress using the **directory-reset-cancel** action. You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
directory-reset [vsp] [true | false]
```

Example

```
NNOS-E> directory-reset
Success!
NNOS-E> directory-reset vsp2 false
The specified vsp was not found
NNOS-E>
```

directory-reset-cancel

Cancels the execution of an in-progress **directory-reset** action. When the **directory-reset-cancel** action is invoked, it immediately stops the reset action: some entries are updated and some are not. Use this action with caution as it leaves the directory entries in a mixed state.

Syntax

```
directory-reset-cancel
```

Example

```
NNOS-E> directory-reset-cancel
directory-reset-cancel has been submitted.  Please check directory-status for
progress.
NNOS-E>
```

disconnect-call

Disconnects the specified call based on the session ID. If a call hasn't yet been answered, this action sends a "403 Forbidden" response to the UA that initiated the call, and a CANCEL request to the called UA. If the call has already been cancelled, the action sends BYE requests to both UAs (informing all involved parties that the call has been shut down). Use the **show active-call** command to retrieve the ID. You can also do this from the Call Logs **Session** pages in the ME Management System. From the GUI, you can list the active calls and then select and disconnect. Use the **terminate-call** action if the endpoints are no longer reachable.

Syntax

```
disconnect-call sessionID
```

Example

```
NNOS-E> disconnect-call 0x4c22932e1cf4ba6
Success!
NNOS-E>
```

display

Temporarily changes the display output without changing your configuration. After ending your CLI session, ME erases the changes and uses the settings in your configuration at the next session. Select either:

- **paged**: The CLI outputs text a page at a time, pausing the display with the **--More--** prompt. You set the number of lines displayed before the prompt. The following table shows the keystrokes the prompt accepts:

Table 3–2 Keystrokes

Keystroke	Result
Enter	Outputs the next line of text.
Tab	Outputs the remainder of the text.
Esc or *Q* or q	Cancels the display, outputs no more text, and returns to the prompt.
any other keystroke	Outputs the next page of text.

- **scrolled**: The CLI scrolls text continuously.

To set the output display in your configuration, use the **cli** object.

Syntax

```
display {paged numberOfLines | scrolled}
```

Example

```
NNOS-E> display paged 12
NNOS-E> ?
archive                     Run the archiving task for a given vsp
arp-delete                  Flush the ARP cache
autonomous-ip-lookup        Actions to determine Autonomous IP group
call-lookup                 Actions for Dial Plan operations
clock                       Set the system time
cls                         Clear terminal screen
config                      Configuration commands
database                    Delete database records or vacuum tables
--More--
```

dns

Executes a variety of DNS actions. Use the **dns** object to configure how the ME services DNS requests. Select one of the following options:

- **lookup**: Executes a DNS lookup on a named host. You can specify a server to query; otherwise the ME uses the server(s) configured through the resolver **server**. You can set a timeout for the request, either a number of milliseconds or that the request never timeout (forever). Optionally you can set the record type that the ME returns:
  - **A**: An IP address (the default).
  - **PTR**: Pointer record, mapping IP address to the canonical name.
  - **SRV**: Service location record.
  - **NAPTR**: Name authority pointers, mapping a domain name to general information such as a URI.
  - **CNAME**: Canonical name, mapping any name aliases.
  - **NS**: Name server record, mapping a domain name to authoritative DNS servers for that name.
- **flush-cache**: Flushes all dynamic entries from the DNS resolver cache of all processes. Optionally, you can specify an individual process cache to flush.
- **reset-servers**: Flushes all dynamic entries from the server DNS resolver cache and resets the DNS sockets. It is possible for a DNS socket has become nonfunctional but the ME continues sending DNS messages to it. This action closes and then reopens all DNS sockets. In addition it resets server counts, refreshes the server, and sets the administrative state to UP. By default, the action impacts all servers configured as resolver **servers**. Optionally, you can specify an individual server.
- **delete-entry**: Deletes the specified record from the DNS cache. Enter the name, and, optionally, the record type of the entry to be deleted. You can delete the entry from all configured servers (the default), or a specific server. Use the **show dns-cache** command to display names of the entries in the cache. a server.

Syntax

```
dns lookup hostName [A | PTR | SRV | NAPTR | CNAME | NS] [serverName] [forever |
milliseconds]
dns flush-cache [process]
dns reset-servers [serverName]
dns delete-entry name [A | PTR |SRV | NAPTR | CNAME | NS] [serverName]
```

Example

```
NNOS-E> dns lookup www.yahoo.com
www.yahoo.com IN
A tag-mine Resolved 60 69.147.76.15
```

dos-delete-rules

Deletes all or a specific configured denial-of-service (DOS) rules. Use the **show dos-rule** command to see the current rules. Rules can be deleted on the local system or for the cluster. By default the action deletes all rules across the cluster.The ME automatically creates DOS rules as a result of the configured policies. The policy selects which table rows to consider, the threshold for instances, and the frequency (period) of comparison. When the threshold is reached for a given period, the ME generates a rule in the kernel to block traffic meeting that selection criteria. The rule will persist as long as any traffic matching the rule is seen within the user-configurable inactivity-timeout period. The ME automatically deletes a rule when the inactivity timer expires,. You can delete rules manually using this action.

Syntax

```
dos-delete-rules [cluster | local] [all | number]
```

Example

```
NNOS-E> dos-delete-rules
Success!
```

dynamic-event-service

This action allows third party applications to subscribe to events without having to be configured directly on the ME.

- register: Allows a web application to register itself by using the web service REST and SOAP clients. Valid arguments are:
  - <*endpoint*>: The application endpoint that receives events.
  - [*channels*]: The channels for which the endpoint is getting events.
  - [*xml-format*]: The XML format used by this server. This can be either **simplified** (the default) or **legacy**.
  - [*time-to-live*]: The time to live, in minutes, for the keepalive on this registration. The default is untilRestart, meaning the registration stays alive until the system is restarted.
  - [*connect-timeout*]: The connect timeout, in milliseconds, for the endpoint. The default is **1000**.
  - [*read-timeout*]: The read timeout, in milliseconds, for the endpoint. The default is **1000**.
  - [*character-set*]: The character set to use when forming requests to this endpoint. This can be **utf-8** (the default) or **iso-8859-1**.
  - [*request-style*]: The style to use when sending events to this listener. This can be **SOAP** (the default), **XML**, or **JSON**.
  - [*include-channels-in-events*]: Whether channels are included in events. This is **enabled** by default.

- [*backupEndpoint*]: The dynamic events endpoint server to be used for redundancy.

- keepalive: Keeps current registrations alive.
- unregister: Allows the web application to unregister itself.

Syntax

```
dynamic-event-service register <endpoint> [channels] [xml-format] [time-to-live]
[connect-timeout] [read-timeout] [character-set] [request-style]
[include-channels-in-events] [backupEndpoint]
dynamic-event-service keepalive <registration-id>
dynamic-event-service unregister <registration-id>
```

Example

```
NNOS-E>dynamic-event-service unregister reg-150
```

enum-lookup

Enter a phone number to resolve. As a resolver, the ME obtains resource records from servers on behalf of resident or requesting applications. In addition, it maps a server to a domain name.

Syntax

```
enum-lookup phoneNumber [domainName] [server]
```

Example

```
NNOS-E>config vsp enum mapping 15085551212
Creating 'mapping 15085551212'
config mapping 15085551212> set url SIP:skd@skdPC:5060;transport=TLS
config mapping 15085551212> exit
Do you want to commit your changes before you exit (y or n)? y
Do you want to update the startup configuration (y or n)? y
NNOS-E> enum-lookup 15085551212
Phone 15085551212 has a mapping to URL SIP:skd@skdPC:5060;transport=TLS
```

ethernet-negotiate

Causes the specified Ethernet interface to renegotiate its parameters. To see the current settings, execute the **show ethernet** command. Enter the Ethernet interface you want to effect, eth0 through eth19.

Syntax

```
ethernet-negotiate ethX
```

Example

```
NNOS-E>ethernet-negotiate eth0
Success!
NNOS-E> show ethernet

name  link speed duplex autoneg
----  ---- ----- ------ -------
```

```
eth0  up   1Gb   full   enabled
eth1  down             enabled
eth2  down             enabled
eth3  down             enabled
```

expression

Provides actions to test match and replacement in regular expressions. A scan option provides line-by-line processing output to help with debugging. Remember that you must use quotation marks around special characters at the command line. Select one of the following:

- **match**: Tests a URI, allowing you to determine whether an expression you created matches a supplied string.
- **replace**: Tests both the regular expression and the replacement for a URI against the supplied string. Supply both and the system responds with the resulting output. The ME executes the replacement on the first occurrence of the expression. To replace all instances, use the **all** keyword option.
- **scan**: Debugs a URI and regular expression by providing pass-by-pass information on how the system is processing the string.

Syntax

```
expression match string regExp
expression replace string regExp replacement [all]
expression scan string regExp
```

Example

```
NNOS-E>expression match 5551212 ^([0-9]{7})$
Success!
NNOS-E>expression replace 5551212 ^([0-9]{7})$ "tel:+1413\1"
tel:+14135551212
NNOS-E>expression scan 5551212  ^([0-9]{7})$
match at offset 0, length 7
...substring 0: '5551212'
...substring 1: '5551212'
```

external-normalization

Manages the file used to maintain DNIS-to-ANI translation data.These mappings are checked prior to any dial-plan lookups. Typically, this action is initiated though the **task** object scheduler. Select one of the following operations:

- **replace-file**: Pulls data from the named file into the external normalization database of the SIP process. All previous mappings are removed. (The mappings known to the ME can be displayed using the **show external-normalization** status provider.) Enter a file path or browse the system for a file name. The default normalization file is **/cxc/normalization.xml**.
- **replace-url**: Identifies the external service that hosts the normalization (mapping) data. Specify a host name or IP address.
- **flush**: Removes all entries from the internal normalization cache. (It does not affect the normalization.xml file.)

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
external-normalization {replace-file fileName | replace-url source | flush}
```

Example

```
NNOS-E>external-normalization flush
Success!
NNOS-E>show external-normalization
match   pri   hits   ALT-REQ-USR   ALT-TO-USR   ALT-FROM-USR   DLG   tag
------  ----  -----  -----------   -----------  ------------   ---   ---

NNOS-E>external-normalization replace-file normalization.xml
Success!

NNOS-E>show external-normalization
match   pri   hits   ALT-REQ-USR   ALT-TO-USR   ALT-FROM-USR   DLG   tag
------  ----  -----  -----------   -----------  ------------   ---   ---
899101201 99   0    **12027423317   OUT
1202742331 99  0    **8991012017    IN                              abc
NNOS-E>external-normalization replace-url
ftp://myserver.foo.com/external-normalization.xml
Success!
```

external-presence

Clears all or a specified entry from the external presence cache. The external cache is the database running on the backup system in a cluster configuration. The primary or master appliance contains the presence cache from which the external presence cache is mirrored. If a failover happens, the external cache becomes the master cache. See the **presence** action for information on managing the master cache.

Select one of the following operations:

- **delete**: Deletes the specified entry from the external presence cache. Enter the URL for the entry, which can be found in the **show presence-cache-external** command.
- **flush**: Removes all entries from the external presence cache.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
external-presence {delete url | flush}
```

Example

```
NNOS-E>show presence-cache-external
url: 2078548355@elmaple.com
Box: 0.0.0.0
state: Online
prestype: Voice
LastRegisteredTime: 0
ExpireInterval: 4294967295
numCurrSubscribers: 1
url: 2078548357@elmaple.com
Box: 0.0.0.0
state: Online
prestype: Voice
LastRegisteredTime: 0
```

```
ExpireInterval: 4294967295
numCurrSubscribers: 3
NNOS-E>external-presence flush
Success!
```

external-session

Removes all entries from the external CSTA SIP session cache. If configured to do so, the cache contains state information for all active CSTA sessions on all boxes in the cluster. This action is typically used to clear the cache without rebooting in the event of a failover recovery. Use the **install examine** action to clear the cache on an individual box.

To ensure that boxes share this information, set **share-signaling-entries** to **true** in the **cluster** object.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
external-session flush
```

Example

```
NNOS-E>external-session flush
Success!
```

file

Performs file management operations. The file actions support a number of protocols (HTTP, HTTPS, FTP, etc.) and can be used to manage a variety of file types, for example, configuration files, certificates, patches, or licenses. All operations begin with \cxc as the root directory. Select one of the following operations:

- **erase**: Deletes the specified file.
- **purge**: Deletes one or more files. Use a wildcard to specify more than one (e.g., /cxc_common/announcements/*). You can select a minimum age for the file(s): the system deletes anything older. See Setting Time and Time Intervals for information on entry format requirements for minimum age.
- **fetch**: Moves a file from a specified location to the ME.
- **send**: Copies a file from the ME to the specified location.

Syntax

```
file erase fileName
file purge fileName [age]
file fetch sourceURL [destinationFile]
file send sourceFile [destinationURL]
```

Example

```
NNOS-E>file erase signature.txt
Success!
NNOS-E>file fetch https://192.168.10.10/cxc.cfg
Success!
```

file-based-word-lists-refresh

Rereads any saved **word-list** or **url-list file** entry into memory. Use this if you have made a change to the file and want the new words, expressions, or domains read into memory.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
file-based-word-lists-refresh
```

Example

```
NNOS-E>file-based-word-lists-refresh
Success!
```

file-mirror-service

Manages mirrored files and the file database. To enable file mirroring, use the **file-mirror** master service. File mirroring sets all participating ME devices to share particular files, such as media recordings, log files, etc., making them highly available. Using this action, you can execute the following operations:

- **make-available**: Moves the named file from its current location to the common directory of the first highly available directory. This is the first entry in the **file-mirror file-mirror-directory** property configuration. The master then distributes the file across the cluster. Use this option to mimic the file mirror function for files that are not automatically managed under the service.
- **fetch**: Validates whether the specified file is up-to-date on the disk. If it is not, the action retrieves the current file from the master.
- **delete**: Removes the specified file form both the local disk and the shared database. The action also sends a message to the master, instructing it to remove the file from each backup box.
- **find**: Returns a list of full path name matches to the relative path name that you enter.

All operations take a file name.

Syntax

```
file-mirror-service {make-available | fetch | delete | find} filePath
```

Example

```
NNOS-E> show file-mirror-db-record

canonical-file-name   timestamp   file-permission   last-update-from-box-number
-------------------   ---------   ---------------   ---------------------------
/cxc_common/mirror1/file1.txt 15:24:16 Tue 2007-10-16  33188            1
/cxc_common/mirror1/file3.txt 15:24:48 Tue 2007-10-16  33188            1
/cxc_common/mirror2/file2.txt 15:26:21 Tue 2007-10-16  33188            1
/cxc_common/mirror2/file3.txt 15:26:14 Tue 2007-10-16  33188            1

NNOS-E> file-mirror-service find file1.txt
Found file /cxc_common/mirror1/file1.txt
```

```
NNOS-E> file-mirror-service find file3.txt
Found file /cxc_common/mirror1/file3.txt
Found file /cxc_common/mirror2/file3.txt
```

file-play

Places a call to the specified SIP URI, plays the .WAV file, and then disconnects the call. This could be a .WAV file you recorded and moved to the ME, for instance with the **file fetch** action. Compare this to the **playback** action. The **playback** action plays recorded sessions only (the ME takes care of mixing the media for playing). This action plays any audio file. For example, if you made a file using the **mix-session** action, you can play it using **file-play**.

Enter the following information:

- **filename**: The location of the .WAV file you want played.
- **to**: The SIP URI that specifies where to place the call to.
- **from**: A SIP URI that appears as the caller ID.
- **transport**: The transport protocol to use, either any, UDP, TCP, or TLS.

Syntax

```
file-play fileName to [from] [transport]
```

Example

```
NNOS-E>file-play greeting.wav sip:users@cov.com sip:management@cov.com
Success!
```

file-play-verify

Verifies that a WAV file is of a format supported by the ME (for example, PCM16/PCMA/PCMU, 16000/8000 sample rate, single channel, and others). Use this action to validate a file used for playout or for insertion as an introduction (see the **media** object) or **periodic-announcement**.

Syntax

```
file-play-verify fileName
```

Example

```
NNOS-E>file-play-verify /cxc/recordings/intro1
Success!
```

file-transfer-delete

Deletes a file that was added to the ME via a file transfer (e.g., via IMs) and entered in the database. (The database maintains a record of all files transferred.) Deleting the file removes the entry from the database.

Enter the following:

- **sessionID**: To find the session ID, use the **Call Logs** tab in the ME Management System or the **show file-transfer-files** command in the CLI.

- **fileName**: The name of the file to delete.
- **identifier**: The unique identifier that distinguishes the file in the event that the same named file was transferred more than once during a single session.

Syntax

```
file-transfer-delete sessionID fileName identifier
```

Example

```
NNOS-E>file-transfer-delete ca3e3764f07c42f5b7b6eda269d2d0c4 greeting.wav 43ab
Success!
```

file-transfer-delete-old

Invokes the ME to delete all files added to the ME via a file transfer that are older than the specified number of days or seconds. Enter a number and a unit of measure. By default, the ME deletes transferred files that are older than seven days when you execute this command.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
file-transfer-delete-old age [days | seconds]
```

Example

```
NNOS-E>file-transfer-delete-old 30
Success!
```

file-transfer-retrieve

Creates a link to the specified file transfer so that the ME Management System can retrieve the file. This action is primarily for use with the GUI only.

Enter the following:

- **sessionID**: To find the session ID, use the **Call Logs** tab in the ME Management System or the **show file-transfer-files** command in the CLI.
- **fileName**: The name of the file to retrieve.
- **identifier**: The unique identifier that distinguishes the file in the event that the same named file was transferred more than once during a single session.
- **file**: The path to the link created for the file.

Syntax

```
file-transfer-delete sessionID fileName identifier file
```

Example

```
NNOS-E>file-transfer-retrieve ca3e3764f07c42f5b7b6eda269d2d0c4 greeting.wav 43ab
/cxc/web/tmp928421
Success!
```

format

Formats the specified system hard drive. Enter the drive you wish to format and the file system to use on that drive. Use this action with caution, as it formats or reformats the specified drive, removing all existing data. Note that the **format** action also automatically performs the equivalent of the **add-device** and **mount** actions. If you have data on a drive that you want to maintain, be sure to manually execute these two actions. See the *Net-Net OS-E – USB Creation and Commissioning Instructions* for a complete description of system drives and partitioning.

Syntax

```
format {usb | data-1 | data-2} [reiser-3 | xfs | ext4 | ext3 | vfat]
```

Example

```
NNOS-E>format data-1
Are you sure (y or n)?y
Success!
```

ftrace

Manages debug tracing results that are directed to a file. You must have an existing settings file to start writing trace results to a file. The settings file defines the trace class and severity to record. You can create a settings file with the ME CLI **trace** command or another program.

Select one of the following options

- **start**: Begins writing tracing results to a file. Specify the process that you want to collect debugging information on, and a file name to write the information to. Optionally, you can specify a settings file. The default settings file used by the ME, if you do not specify a settings file name, is **target.ini**. However, you must create that file before you can use this action.
- **stop**: Ends the tracing action for a specific process to a specific file.
- **remove**: Deletes the specified file for a given target.
- **import**: Imports a trace file into the log database. Specify the full path name of the file to import. The default table to import the file to is **TraceStruct**.

Syntax

```
ftrace start process target [settings]
ftrace stop process target
ftrace remove process target
ftrace import file [table]
```

Example

```
NNOS-E>ftrace start sip /trace/sip
Success!
NNOS-E>ftrace stop monitor /trace/mtr
Success!
```

group-down

Simulates a VRRP group becoming non-operational. You can use this action to test your configuration. If configured for failover, when you execute this action, the ME fails over interfaces and master services to the backup box. If you execute this action without such a configuration, the ME brings down and then restores the specified group.

Syntax

```
group-down vrrpGroup
```

Example

```
NNOS-E>group-down 1
Success!
```

4

Actions H Through Z

This chapter covers Media Engine (ME) actions, from H through Z in alphabetical order. For actions A through G, see "Actions A Through G." An action immediately acts on ME and effects one of the components (manipulates data), whereas objects and properties describe the configuration. Actions are only available at the top-level prompt of the CLI (or through the ME Management System **Actions** tab).

Most actions become available when ME starts, but some may only become available when the corresponding service or provider registers with ME. The registration is dependent on the configuration. For example, the **directory-reset** action cannot register if the directory service is not configured.

h323-issue-lrq

The ME supports an action that allows you to manually issue an H.225 RAS Location Request (LRQ) to a configured peer gatekeeper (GK). If a peer GK with this remote IP address is configured, the LRQ reports the results. The result can be either a route-server with an address, an LRJ with a reason, or a timeout.

You must enter the configured GK address, a string representing the destination to locate, and the type of destination to locate. Optionally, you can enter a string representing the source and the type of source.

A peer GK and LRQ timeout must be configured for you to execute this action.

Syntax

```
h323-issue-lrq <addr> <destinfo> <destinfotype> [srcinfo] [srcinfotype]
```

Example

```
NNOS-E>h323-issue-lrq 172.44.200.35:1719 8675309 dialedDigits CXC-h323-1 h323ID
LRQ sent to peerGK at 172.44.200.35:1719
LRQ for '8675309' was Confirmed, Call Address 10.1.20.126:1720
H.225.0  RAS: locationRequest
H.225.0 RAS
    0... .... Extension Bit: False
    Range = 25 Bitfield length 5, Choice Index: .100 10..
    Choice Index: 18
    RasMessage: locationRequest (18)
        locationRequest
            .... ..1. Extension Bit: True
            .... ...1 Optional Field Bit: True (endpointIdentifier is present)
            0... .... Optional Field Bit: False (nonStandardData is NOT present)
            requestSeqNum: 3
```

```
            Range = 128 Bitfield length 7, Octet String Length: 0001 001.
            Octet String Length: 10
            endpointIdentifier: CXC-H323-1
            Sequence-Of Length: 1
            destinationInfo: 1 item
                Item 0
                    0... .... Extension Bit: False
                    Range = 2 Bitfield length 1, Choice Index: .0.. ....
                    Choice Index: 0
                    DestinationInfo item: dialedDigits (0)
                        Range = 128 Bitfield length 7, Octet String Length: ..00
0011  0... ....
                        Octet String Length: 7
                        dialedDigits: 8675309
            .... 0... Extension Bit: False
            Range = 7 Bitfield length 3, Choice Index: .... .000
            Choice Index: 0
replyAddress: ipAddress (0)
                ipAddress
                    ip: 172.44.10.67 (172.44.10.67)
                    port: 1719
            0... .... Small Number Bit: False
            Number of Sequence Extensions: 15
            .... ...1 Extension Present Bit: True (sourceInfo is present)
            0... .... Extension Present Bit: False (canMapAlias is NOT present)
            .0.. .... Extension Present Bit: False (gatekeeperIdentifier is NOT
present)
            ..0. .... Extension Present Bit: False (tokens is NOT present)
            ...0 .... Extension Present Bit: False (cryptoTokens is NOT present)
            .... 0... Extension Present Bit: False (integrityCheckValue is NOT
present)
            .... .0.. Extension Present Bit: False (desiredProtocols is NOT
present)
            .... ..0. Extension Present Bit: False (desiredTunnelledProtocol is
NOT present)
            .... ...0 Extension Present Bit: False (featureSet is NOT present)
            0... .... Extension Present Bit: False (genericData is NOT present)
            .0.. .... Extension Present Bit: False (hopCount is NOT present)
            ..0. .... Extension Present Bit: False (circuitInfo is NOT present)
            ...0 .... Extension Present Bit: False (callIdentifier is NOT present)
            .... 0... Extension Present Bit: False (bandWidth is NOT present)
            .... .0.. Extension Present Bit: False (sourceEndpointInfo is NOT
present)
            .... ..0. Extension Present Bit: False (canMapSrcAlias is NOT present)
            Open Type Length: 23
            Sequence-Of Length: 1
            sourceInfo: 1 item
                Item 0
                    0... .... Extension Bit: False
                    Range = 2 Bitfield length 1, Choice Index: .1.. ....
                    Choice Index: 1
                    AliasAddress: h323-ID (1)
                        Octet String Length: 10
                        h323-ID: CXC-h323-1
```

h323-reregister-gatekeeper

Sends an UNREGISTER and then REGISTER request to an external gatekeeper, which renegotiates the registration. Use the s**how h323-external-gatekeeper** command to list gatekeepers and their IP addresses.

Syntax

```
h323-reregister-gatekeeper remoteIP
```

Example

```
NNOS-E>show h323-external-gatekeepers
       remote-address: 172.10.100.10:1719
        local-address: 172.30.0.143:1719
             regstate: registered
 last-registered-time: 06:51:26 Mon 2008-10-06
registration-interval: 3600
        gatekeeper-id: OpenH323GK
          endpoint-id: 1290_endp
          activecalls: 0
           default-gw: false

NNOS-E>h323-reregister-gatekeeper 172.10.100.10
Success!
```

h323-unregister-gatekeeper

Sends an UNREGISTER request to an external gatekeeper. Use the s**how h323-external-gatekeeper** command to list gatekeepers and their IP addresses.

Syntax

```
h323-unregister-gatekeeper remoteIP
```

Example

```
NNOS-E>h323-unregister-gatekeeper 172.10.100.10
Success!
```

install

Manages system software releases and network interface cards (NICs). Note that if you have a file on your PC, and want to install it, you can either use PSCP (if you have SSH configured on the ME) or TFTP (if you have TFTP configured on the ME). Or, you can use the ME Management System **Update Software** tool.

Select one of the following:

- **file**: Installs a file that exists on the ME in a format that is install-ready (e.g., decompressed). Enter the file name. Optionally, specify whether to install the file on the local box or the entire cluster. However, to upgrade members in a cluster without interrupting call flow, use the **controlled** option. By default, the ME installs the file on the device.
- **url**: First downloads the specified file, and then installs it onto the ME. Enter a URL for the software upgrade. Optionally, specify whether to install the file on the local box or the entire cluster. However, to upgrade members in a cluster without

interrupting call flow, use the **controlled** option. By default, the ME installs the file on the device.

- **nic**: Updates the stored MAC addresses and associates them to interface names, making the system aware of interfaces it can use. Execute this action if you purchased a new NIC after your system was up and running (i.e., you are adding a NIC). Optionally, for a faster installation, you can specify the hardware platform instead of waiting for the ME to read the IPMI.
- **nic-reinitialize**: Wipes the information from a NIC. Use this action if you selected the wrong system model with the **nic** action, but only if instructed to do so by technical support.
- **module**: Forces installation of an older module from a release marked as good. This action is used to help regain network connectivity should there be problems, but should only be used under direction of Oracle personnel. You can type a question mark at the command line to see the available modules (**install module ?**).
- **cancel**: Cancels the operations initiated by the install file controlled action.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
install file source [box | cluster | controlled]
install url source [box | cluster | controlled]
install nic [model]
install nic-reinitialize
install module kernelModule
install cancel
```

Example

```
NNOS-E>install file release.tar.gz controlled
Are you sure (y or n)?y
sending release.tar.gz to 172.66.0.10...
sending release.tar.gz to 172.66.0.11...
sending release.tar.gz to 172.66.0.12...
Restarting 172.66.0.10...
Restarting 172.66.0.11...
Restarting 172.66.0.12...
NNOS-E is restarting...
NNOS-E>
```

install examine

Displays information about files used for upgrade. Typically this information indicates build, platform, prerequisite, and component data.

Syntax

```
install examine sourcePath
```

Example

```
NNOS-E>install examine /releases/release-b3.4-32250.tar.gz
----------------------------------
/releases/release-b3.4-32250.tar.gz
```

```
---------------------------------
summary NNOS-E Application
description
type simple
restart warm
variety app
platform all
platformType none
version 3.4.99
build 32250
branch b3.4
prereqOSBranch
prereqOSBuild 0
prereqAppBranch
prereqAppBuild 0
name
within
untarDir /cxc
preInstallScript
installScript install/install.sh
InstallComponents
component[1] os 1.6 "" 32170
InstallComponents
component[1] postgres "07.02.0005 PostgreSQL 8.1.2"
component[2] postgres "07.02.0005 PostgreSQL 8.1.9"
InstallComponents
component[1] kernel 2.6.11-6-cov kernel-2.6.11-6
```

internal-session

Removes all entries from the internal CSTA SIP session cache. The cache contains state information for all active CSTA sessions being handled by the local ME devices. This action is typically used to clear the cache without rebooting in the event of a failover recovery. Use the **external-session** action to clear state data for all boxes in the cluster.

Syntax

```
internal-session flush
```

Example

```
NNOS-E>internal-session flush
Success!
```

jtapi-control

Manages terminals associated with 3PCC servers and tests PBX connections. See the **csta-settings** object for a general description, and Identifying the Active Device for a discussion of how the ME uses device ID information. See the Cisco online documentation, *Partitions and Calling Search Spaces*, for complete information on Cisco partitions.

- **get-terminals**: Returns the unique identifier associated with the terminal(s) residing at the specified phone address. Use the output of this action with the **set-active-terminal** action.

- **set-active-terminal**: Sets the active terminal for a phone address. Specify the terminal ID and the address to indicate which terminal is active for that address. Use the **get-terminals** option to returns a list of IDs.
- **clear-active-terminal**: Removes the active terminal designation from the currently active terminal for a specific phone address. You can use **set-active-terminal** to overwrite the selection; use this to remove without replacing the active terminal selection.
- **select-active-terminal**: Clears the current active terminal designation and then rings all terminals associated with the specified address. The terminal used to answer the query becomes the active terminal.
- **get-all-terminals**: Returns, listed by phone address, all terminals for all users currently logged in to MOC.
- **test-pbx-connection**: Used for debugging, tests to make sure that all terminals have a connection to the PBX. Specify a phone address, and the ME tests each configured 3PCC server to see if that number can reach the PBX through the server.
- **reload-terminals-from-termdb**: Refreshes the ME list of active terminals by overwriting the current list with the list stored in the database.
- **set-default-partition**: Sets the partition that the specified address uses when involved in call operations. Enter the partition name and the phone address. The default partition can also be set with the **default-partition** property of the session configuration **csta-settings** object, but the setting of this action overrides the configuration.
- **set-terminal-partition**: Sets the partition that a specific terminal within a specified address uses when involved in call operations. Enter the terminal ID (use the **get-terminals** option to returns a list of IDs) partition name and the phone address.
- **show-connections**: Lists all JTAPI connections and their status.

Syntax

```
jtapi-control get-terminals phoneAddress
jtapi-control set-active-terminal terminalID phoneAddress
jtapi-control clear-active-terminal phoneAddress
jtapi-control select-active-terminal phoneAddress
jtapi-control get-all-terminals
jtapi-control test-pbx-connection phoneAddress
jtapi-control reload-terminals-from-termdb
jtapi-control set-default-partition partition phoneAddress
jtapi-control set-terminal-partition terminal partition phoneAddress
jtapi-control show-connections
```

Example

```
NNOS-E>jtapi-control get-all-terminals

# Provider
# Monitored Addresses 3
Provider: 172.30.3.201 Address: 54815 Active Terminal:
Provider: 172.30.3.201 Address: 54804 Active Terminal: SEP003094C3D233
Provider: 172.30.3.201 Address: 54810 Active Terminal:
# Provider
# Monitored Addresses 1
```

```
Provider: 172.30.3.199 Address: 77704 Active Terminal:

NNOS-E>jtapi-control get-terminals 77702
SEP000E0C774E0B

NNOS-E>jtapi-control test-pbx-connection 77702
Cisco Call Manager 6.0
Provider: 172.30.3.199 16
Address=77702 Exists
Terminal=SEP000E0C774E0B

cisco 4.0
Provider: 172.30.3.201 16
Address=77702 Not found on Provider
```

license

Manages the Oracle-licensed software that contains the your specific product features. You must install a license to unlock your customer-specific features. You can fetch a license from the ME license server, or apply a license that is already on your local system. Use the **show licenses** command to see whether a license is installed or to determine the license name.

A license contains one or more features. You can have multiple licenses active at any time (displayed with **show licenses**). You can apply, refresh, and revoke licenses, but not the individual features within a license.

- **fetch**: Downloads a license from the ME license server and installs it on your system. Before executing this action be sure that you have a connection to the public Internet and that port 616 is available and is not blocked by a firewall. You must supply a key to access your file; copy and paste the key you received from your sales representative. Optionally, you can specify a different server. The default server path is https://license.covergence.com:616/.
- **apply**: Applies the specified license file, which must be stored on the system from which the action is executed. Use this if you have multiple devices as part of your license agreement and you have already saved the file locally. Optionally you can specify whether the license application is temporary (the life of the current session) or permanent. By default it is temporary.
- **revoke**: Disables the specified license. You might use this to disable a feature for testing, for example. Optionally you can specify whether the change is temporary (the life of the current session) or permanent. By default it is temporary.
- **refresh**: Refreshes a license expiration or feature change. Use this action if you have an existing license nearing expiration or an evaluation copy. When you execute the action, ME logs in to the license server with the key stored key and updates the license (if available) with a later expiration time. Optionally, you can specify a different server. The default server path is https://license.covergence.com:616/.

Syntax

```
license fetch key [server]
license apply file [temporary | permanent]
license revoke license [temporary | permanent]
license refresh license [server]
```

Example

```
NNOS-E>show licenses

       name: ABCco LICENSE
description: LICENSE for company ABC
        key: 87702f9a-be13-9974-83904-d00b7e4ab51f
    expires: 12.31.06
       file: license.xml

NNOS-E>license refresh ABCco License
Success!
```

linksys

Manages the Linksys certificate process. Linksys equipment supports a proprietary version of SRTP. It uses SIP INFO messages to exchange credentials (in mini-certificates) and securely distribute the key used to encrypt/decrypt the RTP packets. The RTP encryption is a variation of *RFC 3711, The Secure Real-time Transport Protocol (SRTP)*. Linksys uses the same encryption algorithm (AES-CM-128), but uses HMAC-MD5 instead of HMAC-SHA1 for authentication. The ME must have access to the key used to generate the mini-certificates for participation in encryption/decryption. This action can generate the mini-certificate and private key needed by each phone.

> **Note:** You must have a root certificate loaded on the system for this action to be successful. The default location for the root certificate is /cxc/certs/linksys_ca.pem.

The linksys action provides five tools.

- **mini-certificate**: Creates a mini-certificate, which will later be used by a Linksys phone to exchange an encrypted symmetric key. When both phones in a call support cryptographic exchange, use this action to create a mini-certificate that is sent in an INFO message to the other phone. (You must execute this action for both phones.) After exchanging mini-certificate, the phones can then exchange an encrypted symmetric key.

Enter the following fields to generate a mini-certificate:

- **userID**: A name that identifies this phone (subscriber) to the other party. The user ID can be up to 32 characters.
- **displayName**: A name used by the caller to verify that the callee is the intended call recipient. Enter the user ID field in the Request URI of the INVITE message sent to the proxy server by the caller UAC when making a call to this subscriber (UAS). The display name can be up to 16 characters.
- **expiration**: The date and time at which this mini-certificate expires. Enter the date in the format *hh*:*mm*:*ssyyyy-mm-dd.*
- **filename**: A name for an output file that will contain the mini-certificate and private key. If you do not specify a file name, the output is not written to a file.

Once you execute this option, the ME returns the content of the mini-certificate and the SRTP private key. You can copy and paste each of these fields into your Web GUI for your phone (or other software interface), as well as test the certificate using the **check-mini-cert** option.

- **generate-ca-key**: Generates a Linksys/Sipura CA key. This is the public/private key pair that acts as the Sipura certificate authority. It is needed to generate the mini-certificates for each phone and during the key exchange.
- The key is stored in **/cxc/certs/linksys_ca.pem**. When executing this action, you can specify whether to overwrite any previous CA key. The default setting, **false**, does not overwrite the key. Set the field to **true** to force an overwrite.
- **check-mini-cert**: Verifies the contents of a certificate created with the **mini-certificate** option. When executed, the ME checks the expiration date and signature of the certificate. Enter the content of the mini-certificate to invoke this option.
- **display-mini-cert**: Displays the values of the fields that you entered when you created the mini-certificate. Paste the encoded certificate into the system to display output in a readable format.
- **display-msg**: Displays the fields of the SIP INFO messages that were created during key exchange. After a call is established, endpoints send INFO messages to negotiate key exchange. The bodies of these messages are base-64 encoded. Enter a message (which can be taken from the call log or a trace) to display the message in a readable format.

Syntax

```
linksys mini-certificate userID displayName expiration [fileName]
linksys generate-ca-key [false | true]
linksys check-mini-cert certContent
linksys display-mini-cert certContent
linksys display-msg message
```

Example

```
NNOS-E>linksys mini-certificate 9577 9577-display "23:05:45 2004-11-25"

Mini Certificate:
OTU3Ny1kdWVyb2QAAAAAAAAAAAAAAAAAAAAAAAAAAAA5NTc3AAAAAAAAAAAAAAAAMDMwNTQ1MjYxMTA0yb
YgcwG8IeaYz225Grs7sDJflnfyJxARPehQ+CO6WisAZ77U2zBi8TCapIwqcDhNXwgYKZxljAET3dFnzAxs
2ze1/kEHCqvUmDIEjaYL+1WTySaI1TGKy15FbyZb6dQXtbPF+fXiRP//caFfKUBTuuwtjExxaAz0H3u8Tc
2YT/wH7a0+snpUTFeK/Sv9vd7aAUbufSxewlL2GeTdOu0v2i4R25/RH6iOHyChGpVt2EJ3BHAlLgXTfJib
iwwkrMSe1grSibsCy0D825ezAt66AVKTa/hOmSBvdZvdamJIsbP89vnAJPiOfWNet8T40/wOYyylAE5JDJ
/2+G/MDyc5ImzFTvifKvIQ55T7Jr5E0RUbacDZIlHy5oW+x4sfawCiQZunnb11qlAgYhvOeuo4f3JGUKJA
ld0GRjHfvjRhb3c=

SRTP Private Key:
Oxq38oJqjhe++yBTtTotoMndnZXulkgnnxFQPd0v96oc81IZ5dug9Szob9ZYQXsPkWAxSb
Oxq38oJqjhe++BVpyxz2P2qtZEg==
NNOS-E>linksys check-mini-cert
OTU3NwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA5NTc3AAAAAAAAAAAAAAAAMTU1ODQ4MTEyNTA0z1
TBkpXzjmR6PFX5K4S7G5SxdpozH460T14KpwOxZ8ly4KWpFlcC2rTTWEU6WnOufcj5Bfif7cdsAF/89kZu
83NFceK2ZBRGrJ4cbxREtuPwy1FqkXpBQcztTFXjeyFaq8K7OESebQayFetBEceIupuzxfedlJPRsMRhsH
N1uKpomc/tdJFHJhxSzn+fX+GTACrXQEHzI+ooDL+iQvzhJ1zk/gXTGuk76lkJG2XLvSvdjTp8RjQX/F5h
0GnBa02d3bQ51n7IBvJnTeaGKp/U/e5pQvW5u6vD/uHkqkTGkZDZzOyIISIdgWVxdjA9cpaSa2D5nPhr8G
/WhOadLZ08fmB0kPwEFjJ0h0dojjknjNJp/qVjR5NEEzuj5kH7Qlvxk25l0MThhydCYpbxShy2GSno7apn
yCA02YBQCRlGBOs=
Certificate has expired

NNOS-E>linksys generate-ca-key
Unable to overwrite Linksys CA key
```

```
NNOS-E>linksys display-message

AAAAAMFgaBarzavNNzc3NwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA3Nzc3AAAAAAAAAAAAAAAAMD
Q1OTU5MTEwMTA4qtnZJ1ER7t64CqEHeDYytppxkTkX0OWl6hR8RKRCLu5yruwXVM4c4QYJ/AjcHZ90vcdy
xbEjPXDTO9nZLAZkU8k1iBGUaBUkHZbiae8H3+xa8dIq3m0Ydv28dPqA3UkGg3oIepsfaxDJXi/ci3VSD+
J78hYlVEHwOu0JfI+Y0cuyk1wOxSLIJP7smLTb8oDCuN+wGdMUVNuphM+zY6SGJwkaae8sR7sikpun/fG3
aFx51aRxSt9AHyAplLZbzPnb
Message ID        : HELLO (0x00000000)
SSRC              : 3244320790 (0xc1606816)
Flags             : 0xabcdabcd
Mini-certificate  :
User name         : 7777
User ID           : 7777
Expiration        : 00:59:59 Sat 2008-11-01
Public key        :
0xaad9d9275111eedeb80aa107783632b69a71913917d0e5a5ea147c44a4422eee72aeec1754ce1ce1
0609fc08dc1d9f74bdc772c5b1233d70d33bd9d92c066453
Signature         :
0xc9358811946815241d96e269ef07dfec5af1d22ade6d1876fdbc74fa80dd4906837a087a9b1f6b10
c95e2fdc8b75520fe27bf216255441f03aed097c8f98d1cbb2935c0ec522c824feec98b4dbf280c2b8
dfb019d31454dba984cfb363a48627091a69ef2c47bb22929ba7fdf1b7685c79d5a4714adf401f2029
94b65bccf9db
```

load-balancing

Manipulates the load balancing database. Do not execute this action unless instructed to do so by Technical Support.

Syntax

```
load-balancing {rewrite-rules | refresh-interface-stats}
```

Example

```
NNOS-E>load-balancing rewrite-rules
Success!
```

location

Manages the location cache for the current box. Use the **location-database** action to manage the database that runs on the master. Select one of the following operations:

- **lookup**: Returns location information for the specified AOR.
- **flush**: Removes all entries from the location cache. Optionally you can specify whether to flush the cache immediately or as the entries time out. By default, the system flushes entries immediately.
- **delete**: Deletes the specified entry from the location cache. Enter the address of record for the entry, which can be found in the **show location-cache** command.
- **restate**: Changes the state of the location cache entry. Enter the AOR and the new state.
- **audit**: Verifies that location bindings are not corrupted. You can enter a specific AOR to verify or verify the entire location cache.
- **prune**: Removes all or the specified corrupted bindings from the location cache.

- **save**: Writes the specified AOR or the entire location cache to the location database on the cluster master.
- **reload**: Reloads the location database to the location cache on the local box, wiping out the current cache. If you enter an AOR, only that entry is rewritten.
- **cleanup**: Immediately removes all location bindings that are not in a registered state. You can automate this action by setting the **cache-cleanup-interval** property of the location service **settings** object.
- **expire**: Forces the location cache entry expiration time to 0, resulting in no throttling for the following applicable REGISTER request. You can specify the address of record that this action applies to, or apply it to all AORs.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
location lookup aor
location flush [now | gracefully]
location delete aor
location restate aor {unregistered | trying | in-service | redirect | registered |
out-of-service}
location audit [aor]
location prune [aor]
location save [aor]
location reload [aor]
location cleanup
location expire [aor]
```

Example

```
NNOS-E>location restate sip:6667778888@abc.com out-of-service
Success!

NNOS-E>location lookup sip:6667770001@best.com
sip:6667770001@best.com currently has the following locations:
Contact:
<sip:6667770001@192.168.215.95:5060;transport=UDP;line=g4ced3b1>;expires=59

NNOS-E>location audit
Success!>
```

location-database

Manages the location database across the cluster. The primary or master appliance contains the main location database. The external database, which is mirrored from the main database, is the database running on the backup system in a cluster configuration. If a failover happens, the external database becomes the master database. Use the **location** action to manage the cache that runs on an individual box. Select one of the following operations:

- **merge**: Merges the specified file into the existing master location cache. If the merge file has a new binding (i.e., one with an index not present in the cached AOR), it is added to the existing cache. If the merged copy has a binding that already exists in the AOR, the values from the merged copy take precedence, overwriting the values in the existing cache. By default, the ME uses the file /cxc/location.xml. Optionally, you can specify a different file path.

- **replace**: Writes the specified file to the location database, wiping out the current entries. By default, the ME uses the file /cxc/location.xml. Optionally, you can specify a different file path
- **save**: Writes the location database to the supplied file name. If you do not supply a name, it saves to the default location: /cxc/location.xml.
- **delete**: Deletes the specified entry from the location database. Enter the address of record for the entry, which can be found in the **show location-cache** command.
- **flush**: Removes all entries from the location database.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
location-database merge [fileName]
location-database replace [fileName]
location-database save [fileName]
location-database delete aor
location-database flush
```

Example

```
NNOS-E>location-database merge /cxc/backup/location.xml
Success!
```

log-target

Turns logging on and off for the current CLI session. You configure the CLI as a log target, including the desired class and severity filters, using the services **cli** object.

For any given CLI session, you can turn this output on or off using this **log-target** action. You may want to do this, for example, so that you can record logging in one CLI session, while working in another. In that way, your work will not be interrupted by log messages.

You must have enabled CLI logging and event logging on a global level for this action to take effect. Set admin to enabled for both the **cli** and **event-log** objects.

This action is not available from the ME Management System.

Syntax

```
log-target {enabled | disabled}
```

Example

```
NNOS-E>log-target enabled
NNOS-E>
```

login

Manages user logins. You can either terminate any active login session or unlock users that have been locked out.

- **kill**: To terminate a session, use the **show login-sessions -v** command to list active sessions with identifiers. Specify the session type and the ID to identify the session to end.

- **unlock**: Reinstates access to a user who has been locked out. (This may happen, for example, if the user violated the **password-policy** configuration.) Specify the name of the user to unlock. Use the **show local-users** command to verify that a lockout is the problem for the user.

Syntax

```
login kill {console | ssh | telnet | web | web-service | desktop | monitor} id
login unlock name
```

Example

```
NNOS-E>login kill web 2
Success!
```

loopback

Establishes an outgoing SIP loopback call. In this call type, the media is looped back to the ME to provide various performance metrics. The loopback process works by an endpoint encapsulating and reflecting RTP packets back to the ME. Data such as media path, round-trip time, and packet loss are written to the ME database, and can be displayed using the ME Management System **Call Logs Monitored Calls** link or the **call-loopback** Trend Graph (on the **Status** page). To execute this action, specify which type of loopback to perform and other parameters:

- **packet**: The ME initiates sending RTP packets. When the endpoint receives an RTP packet it responds by reflecting it back to the ME.
- **packet-init**: When
- The Session Controller Media Engine
- initiates the action, the endpoint begins responding immediately by sending packets of the type negotiated in the SDP. When the endpoint receives the RTP packets, it ceases sending SDP packets and reflects RTP back to the ME.
- **seconds**: Specify the duration of the call, between 2 and 600 seconds.
- **to**: Specify the endpoint in the format of a SIP URI.
- **from**: *Optionally*, specify the endpoint in the format of a SIP URI.
- **protocol**: *Optionally*, specify the content of the FROM header in the SIP INVITE.

You can also schedule this action as part of routine maintenance using the **task** object or from the **Monitored Calls** or **Monitored URIs** pages in the ME Management System.

Syntax

```
loopback {packet | packet-init} seconds to [from] [any | udp | tcp | tls]
```

Example

```
NNOS-E>loopback packet 10 sip: 5554443211@jane.cov.com
Success!
```

make-usb-bootable

Changes the state of a USB stick that was used for a system upgrade. This action is only necessary on those platforms that require removal of the USB stick in order for the ME device to boot from its hard drive. In those cases, you must remove the stick prior to the system rebooting. By doing so, you leave the stick in an unusable state. However, the stick still contains data required for the upgrade (license, configuration, etc.). To retrieve that information:

1. Start the upgrade and remove the stick from the ME device before the system reboots.
2. After the system reboots, reinsert the stick.
3. Execute this action with the **true** option.
4. Perform a warm **restart** to retrieve additional configuration from the stick.

Do not use the **false** option with this action unless instructed to do so by Technical Support.

Syntax

```
make-usb-bootable [true | false]
```

Example

```
NNOS-E>make-usb-bootable true
Success!
```

media-delete

Deletes all media files associated with a specific session ID. Use the **show media-files** command to list the files. The path field of the output is the complete path and file name. That name also contains the session ID.

Syntax

```
media-delete sessionID
```

Example

```
NNOS-E>show media-files
session-id        channel  date                        path
----------        -------  -----                       ----
0x4c2264e08bc7098  0       15:30:08 Thu 2006-02-23     /cxc_common/rtp_
recorded/04c2/264e/08bc/sess-04c2264e08bc7098-0-0.xml
0x4c2264e08bc7098  1       15:30:08 Thu 2006-02-23     /cxc_common/rtp_
recorded/04c2/264e/08bc/sess-04c2264e08bc7098-0-1.xml
0x4c2264e09522afa  0       15:30:17 Thu 2006-02-23     /cxc_common/rtp_
recorded/04c2/264e/0952/sess-04c2264e09522afa-0-0.xml
0x4c2264e09522afa  1       15:30:17 Thu 2006-02-23     /cxc_common/rtp_
recorded/04c2/264e/0952/sess-04c2264e09522afa-0-1.xml
NNOS-E>media-delete 0x4c2264e08bc7098
Success!
```

media-delete-old

Invokes the ME to delete all recorded media files that are older than the specified number of days or seconds. Enter a number and a unit of measure. By default, the ME deletes media files older than seven days when you execute this command.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
media-delete-old age [days | seconds]
```

Example

```
NNOS-E>media-delete-old 30 days
Success!
```

media-directory-clean

Removes empty recorded media directories. You may have an empty directory, for example, if the ME cleaned a directory as part of a scheduled maintenance operation. That action removes data but leaves the directories. You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
media-directory-clean
```

Example

```
NNOS-E>media-directory-clean
Success!
NNOS-E>
```

media-package

Creates a tarball of specified WAV files suitable for installation on other ME devices. Use a wildcard expression to enter multiple source files. The ME verifies that all files are playable WAV files. Enter the desired tar ball file name as the destination.

Syntax

```
media-package sourceFiles destinationFile
```

Example

```
NNOS-E>media-package /cxc_common/media/*.wav package.tar.gz
Success!
```

media-session-audit

Sets the minimum age for a media session before it is subject to a media session audit. When you execute this action, the ME checks to ensure that each media session has a matching signaling session. If the ME finds a media session without accompanying signaling, it deletes the orphaned media session. Because the media session is established before signaling is set up, the ME disregards sessions until they have

reached an age set by this action. See Setting Time and Time Intervals for information on entry format requirements for minimum age.

Syntax

```
media-session-audit minAge
```

Example

```
NNOS-E>media-session-audit 2:30
Success!
```

mikey

Provides utilities to work with the Multimedia Internet KEYing (MIKEY) key management scheme. MIKEY is intended for use with real-time applications, specifically to set up encryption keys for multimedia sessions that are secured using SRTP. (See *RFC 3830, MIKEY: Multimedia Internet KEYing*, for more information.) Select one of the following options:

- **display-message**: Displays the base-64 encoded MIKEY messages sent in SDP. For encrypted fields (mainly the KEMAC field), the system displays hexadecimal values within the MIKEY message. Both the field values and the derived SRTP Master keys are displayed for each direction (if the KEMAC is not encrypted). Enter the content of the a=key-mgmt:mikey field. The **derived-salt** Boolean indicates whether the SRTP Master Salt (typically the last 14-bytes of the key) is directly used (**false**) or derived from the MIKEY message (**true**). The default is **false**.
- **decrypt-message**: Decrypts the encrypted portions of the MIKEY message and then displays the base-64 encoded MIKEY messages sent in SDP. If the encryption algorithm in the KEMAC field is not enc-alg=None, use this option to decrypt the field. Once decrypted, the action then provides the same display as the output of the **display-message** option. For this action to work, the data-type field of the MIKEY message must be Pre-Shared Secret. Enter the content of the a=key-mgmt:mikey field and the pre-shared key (secret) associated with the message. The **derived-salt** Boolean indicates whether the SRTP Master Salt (typically the last 14-bytes of the key) is directly used (**false**) or derived from the MIKEY message (**true**). The default is **false**.

Syntax

```
mikey display-message message [true | false]
mikey decrypt-message message secret [true | false]
```

Example

```
NNOS-E>mikey display-message
AQAVgCSvwBQCAAAAAAAAAAAAAADmKwoBAAAAAAUBAAVtaWtleQsAw6WosRcKPAAKFBDQ7YekSuOfDzXIDe
pCWouYsdFsAQAAADYCAQEDBAAAAKAEBAAAAHALBAAAAFAAAQEBBAAAAIAJAQAGAQAFAQAIAQEKAQEHAQEM
BAAAAAAAAAAkABAAECLFFMcmhxxR1ssTIaW5Zw8ADrQBG2lw7G64M2W3PTLwAA==
MIKEY Message:
Version           : 1 (0x01)
Data-type         : Preshared - Init (0x00)
Response          : true
Pseudo-Random     : MIKEY-1 (0x00)
CSB-ID            : 615497748 (0x24afc014)
```

```
  CS Maps           : 2 SRTP-ID maps
  Policy            : #0, SSRC=0 (0x00000000), ROC=0
  Policy            : #0, SSRC=3861580289 (0xe62b0a01), ROC=0
  Attributes        : 5 attributes
  Extension (0x15)  : SDP (01), len=5, 0x6d696b6579
  Timestamp (0x05)  : NTP-UTC (00), 0xc3a5a8b1170a3c00
  Random (0x0b)     : 20-byte random=0x10d0ed87a44ae39f0f35c80dea425a8b98b1d16c
  Security Policy (0x0a) : #0: SRTP, total-length=54 bytes, Auth-alg (2)=0x01,
Auth-key-len (3)=0x000000a0, Salt-key-len (4)=0x00000070, Auth-tag-len
(11)=0x00000050, Enc-alg (0)=0x01, Enc-key-len (1)=0x00000080, FEC (9)=0x00, KDR
(6)=0x00, PRF (5)=0x00, Rtcp-enc (8)=0x01, Rtp-auth (10)=0x01, Rtp-enc (7)=0x01,
Prefix-len (12)=0x00000000
Keymac (0x01)        : enc-alg=None, 36-byte value={key-type=TGK-Salt,
key-validity=None, length=16, value=0x22c514c726871c51d6cb1321a5b9670f, 14-byte
salt value=0xb4011b6970ec6eb83365b73d32f0}, mac-alg=None
```

mix-session

Mixes audio files into a WAV file. You can use the **file-play** action once you have created the file.

When the ME records a call, each direction is stored in an XML format. The XML format contains information about packet timing etc. (SSRCs, timestamps, sequence numbers). The **mix-session** action takes the audio data out of those XML files (usually two files: one for each direction) and puts it into a single .WAV file. During that process, the ME decodes (if necessary) to a standard linear format, which you select.

Enter:

- **sessionID**: The session ID of the originating audio file. Use the **show media-files** command to list the files and find the ID.
- **fileName**: The WAV destination file. The ME creates a file with just the name you supply, so append the **.wav** suffix to the file name if you want it.
- **outputChannels**: The number of channels the file should be mixed for. Enter 1 for mono, 2 for stereo. If you specify stereo, you will hear the different sides of the conversation through different speakers. The default setting is 2.
- **WAV format**: The format you'd like the final WAV file in. The default format is **pcmu**.
- **recordedPath**: Specifies the location of the files to be mixed. Use this only if the files are not in the default location.

Syntax

```
mix-session sessionID fileName [outputChannels] [pcmu | pcma | pcm16]
[recordedPath]
```

Example

```
NNOS-E>mix-session 0x4c22760ab06a58a test1.wav
Success!
```

mix-session-threaded

The **mix-session-threaded** action has does the same thing as the **mix-session** action, but with improved performance from multi-threading. Valid wave formats are PCMU, PCMA, and PCM16.

Enter:

- **sessionID**: The session ID of the originating audio file. Use the **show media-files** command to list the files and find the ID.
- **fileName**: The WAV destination file. The ME creates a file with just the name you supply, so append the **.wav** suffix to the file name if you want it.
- **outputChannels**: The number of channels the file should be mixed for. Enter 1 for mono, 2 for stereo. If you specify stereo, you will hear the different sides of the conversation through different speakers. The default setting is 2.
- **WAV format**: The format you'd like the final WAV file in. The default format is **pcmu**.
- **recordedPath**: Specifies the location of the files to be mixed. Use this only if the files are not in the default location.

You can specify the number of threads that can be used in this action via the **rtp-mixing-action-threads** advanced configuration property.

Syntax

```
mix-session-threaded <session-id> <file> [output-channels] [wav-format]
[recorded-path]
```

Example

```
NNOS-E>mix-session-threaded 0x4c22760ab06a58a test1.wav
Success!
```

mos-calculate

Calculates a MOS score based on various network conditions. Mean Opinion Score (MOS) is a subjective measurement and an "opinion" of the audio quality heard by the listener on a phone. By plugging values into this action that represent your network conditions, you can determine the call quality. See the *Net-Net OS-E – Session Services Configuration Guide* for more information on formulating MOS results.

Enter the following parameters:

- **pktsReceived**: The number of RTP packets received.
- **pktsLost**: The number of packets lost during a call, detected by RTP header examination.
- **pktsDuplicated**: The number of duplicate RTP packets during a call, detected by RTP header examination.
- **jitter**: The average jitter, in milliseconds.
- **latency**: The average transmission delay for each packet, in milliseconds.
- **codec**: The CODEC used which to base default values for other fields.
- **Rfactor**: The R-factor, a rating on the overall conversational quality of a call, expressed on a 0-to-100 scale. A value of 0 uses the CODEC defaults, otherwise enter a number up to 100, where 0 is extremely bad quality, and 100 is very high quality.
- **pktInterval**: The ptime (nominal time between adjacent RTP packets) for the CODEC, in milliseconds. Enter a value between 0 and 100. A value of 0 indicates the default ptime for the CODEC.

Syntax

```
mos-calculate pktsReceived pktsLost pktsDuplicated jitter latency codec [Rfactor]
[pktInterval]]
```

Example

```
NNOS-E>mos-calculate 10 0 3 6

Codec                : pcmu
Packet-interval      : 20 msecs
R-factor             : 93.2
Received             : 7 packets
Lost                 : 0 packets
Average Jitter       : 6 msecs
Average Latency      : 0 msecs
MOS                  : 4.44

NNOS-E>mos-calculate 10 1
Codec                : pcmu
Packet-interval      : 20 msecs
R-factor             : 93.2
Received             : 10 packets
Lost                 : 1 packets
Average Jitter       : 0 msecs
Average Latency      : 0 msecs
MOS                  : 4.06
```

mount

Mounts a data partition on the selected drive so that the ME can access the data on the device. You can also mount a CD-ROM or USB device. Use this action to insert a device into a live system. (If the device is present at boot, the ME automatically mounts it.) Use the **show mounts** command to display mount status; the Drive Name field indicates whether a partition is mounted.

Syntax

```
mount {data-1 | data-2 | usb | cdrom | system-1 | system-2}
```

Example

```
NNOS-E>mount hard-drive-1
Device is mounted
```

named-variable-add

Adds a variable and its value to a specific session.

Enter the following parameters:

- <*session-id*>: The session-id for the session you are applying this named variable.
- <*variable*>: The name of the named variable to be added.
- <*value*>: The value of the named variable to be added.

Syntax

```
named-variable-add <session-id> <variable> <value>
```

Example

```
NNOS-E>named-variable-add session1 var1 /*call-id
```

named-variable-delete

Deletes a named variable for a specified session.

Enter the following parameters:

- <*session-id*>: The session-id for the session whose named variable you are deleting.
- <*variable*>: The name of the named variable to be deleted.

Syntax

```
named-variable-delete <session-id> <variable>
```

Example

```
NNOS-E>named-variable-delete session1 var1
```

named-variable-modify

Modifies a specific named variable for a specified session.

Enter the following parameters;

- <*session-id*>: The session-id for the session whose named variable you are modifying.
- <*variable*>: The name of the named variable you are modifying.
- <*value*>: The new value of the named variable.

Syntax

```
named-variable-modify <session-id> <variable> <value>
```

Example

```
NNOS-E>named-variable-modify session1 var1 /*call-start
```

orderly-restart

Causes a restart of the type specified after gracefully terminating any existing connections. By default, the **orderly-restart** action causes the box to restart at the first point in time when there are no active calls. This is useful for code upgrades on a unit which is currently in service. To immediately restart the box, use the **restart** action.

Select one of the following restart types:

- **warm**: Exits and restarts the ME application when there are no active calls. This is the default.
- **cold**: Exits the operating system and then restarts.

- **halt**: Stops the ME and does not restart the system.
- **cnx0**, **cnx1**: Stops and restarts the specified CNX card.
- **cluster**: Performs a warm restart of all boxes within the cluster.
- **controlled**: Performs a warm restart of all boxes within the cluster without interrupting call flow.
- **cancel**: Cancels the operations initiated by the **orderly-restart controlled** action.

You can also schedule this action as part of routine maintenance using the **task** object. Use the **show orderly-restart** command to report on the current status of an invoked orderly-restart action (i.e., to display a report of the number of currently active calls that this system is awaiting termination on.)

Syntax

```
orderly-restart {warm | cold | halt | cnx0 | cnx1 | cluster | controlled | cancel}
```

Example

```
NNOS-E>orderly-restart cnx0
Are you sure (y or n)?y
Success!
```

performance-tracking

For Technical Support use only.

Syntax

```
performance-tracking start [filename] [samplingInterval] [collectionDuration]
[enabled | disabled]
performance-tracking info
performance-tracking stop
```

ping

Tests whether a specific IP address can accept requests, verifying the existence and connectivity of a host on the Internet. Enter a host name or IP address. Optionally, you can set the number of attempts (packets sent) and the outgoing interface used. The default number of packets sent is 3. The ME does a route lookup to select an outgoing interface unless you specify otherwise. You can also verify connectivity using the **arp request** action.

Syntax

```
ping host [count] [interface]
```

Example

```
NNOS-E>ping 196.84.32.1
no response from 196.84.32.1
no response from 196.84.32.1
no response from 196.84.32.1
3 packets sent, 0 packets received, 3 packets lost (100%)

NNOS-E>ping 172.26.0.49
```

```
28 bytes from 172.26.0.49: 0.272 ms
28 bytes from 172.26.0.49: 0.236 ms
28 bytes from 172.26.0.49: 0.201 ms
3 packets sent, 3 packets received, 0 packets lost (0%)
roundtrip minimum/average/maximum: 0.201/0.236/0.272 ms
```

playback

Places a call to the specified SIP URI, plays the recorded media specified by the session ID, and then disconnects the call. Compare this to the **file-play** action. The **playback** action plays recorded sessions only (the ME takes care of mixing the media for playing). The **file-play** action plays any file. For example, if you made a file using the **mix-session** action, you can play it using **file-play**.

Enter the following information:

- **sessionID**: The session ID of the recorded media. Use the **show media-files** command to list the files and find the ID.
- **to**: The SIP URI that specifies where to place the call to.
- **from**: *Optional*. A SIP URI that appears as the caller ID.
- **transport**: *Optional*. The transport protocol to use, either any, UDP, TCP, or TLS.

Syntax

```
playback sessionID to [from] [transport]
```

Example

```
NNOS-E>playback 0x4c22760ab06a58a sip:management@cov.com sip:hr@cov.com
Success!
```

presence

Manages the presence cache. The primary or master appliance contains the main presence cache. The external cache, which is mirrored from the main cache, is the database running on the backup system in a cluster configuration. If a failover happens, the external cache becomes the master cache. See the **external-presence** action for information on managing the external cache.

Select one of the following operations:

- **merge**: Merges the specified file into the existing master presence cache. If the merge file has a new URL entry, it is added to the existing cache. If the merged copy has a URL that already exists, the values from the merged copy take precedence, overwriting the values in the existing cache.
- **replace**: Writes the specified file to the presence cache, wiping out the current cache.
- **save**: Writes the presence cache to the supplied file name. If you do not supply a name, it saves to the default location: /cxc/presence.xml.
- **delete**: Deletes the specified entry from the presence cache. Enter the URL for the entry, which can be found in the **show presence-cache** command.
- **flush**: Removes all entries from the presence cache.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
presence {merge fileName | replace fileName | save fileName | delete URL | flush}
```

Example

```
NNOS-E>presence delete 5085551212@abc.com
Success!
```

proxy-accept

The **proxy-accept** action tells the proxy state machine that the request referenced by the handle (returned in the SIP message event) can proceed. You can also specify the response code, text string, and session-config reference if additional header or body manipulation is required on the response.

Syntax

```
proxy-accept <handle> [response-code] [response-string] [session-config]
```

Valid arguments for this action are:

- <*handle*>: References the proxy session handling the request (this value is returned in the SIP message event).
- [*response-code*]: The SIP response code to return in response. By default the ME uses the response-code configured in the registration-plan.
- [*response-string*]: The SIP response string to return in the response. By default the ME uses the response-string configured in the registration-plan.
- [*session-config*]: The session-config to use for formatting the response headers and bodies.

Example

```
NNOS-E>proxy-accept abc123 300 Accept config1
```

proxy-discard

The **proxy-discard** action tells the proxy state machine that the request referenced by the handle (returned in the SIP message event) has been rejected and to silently discard the message.

Syntax

```
proxy-discard <handle>
```

Valid arguments for this action are:

- <*handle*>: References the proxy session handling the request (this value is returned in the SIP message).

Example

```
NNOS-E>proxy-discard abc123
```

proxy-reject

The **proxy-reject** action tells the proxy state machine that the request referenced by the handle (returned in the SIP message event) has been rejected. You can also specify the response code, text string, and session-config reference if additional header or body manipulation is required on the response.

Syntax

```
proxy-reject <handle> [response-code] [response-string] [session-config]
```

Valid arguments for this action are:

- <*handle*>: References the proxy session handling the request (this value is returned in the SIP message event).
- [*response-code*]: The SIP response code to return in response. By default the ME uses the response-code configured in the registration-plan.
- [*response-string*]: The SIP response string to return in the response. By default the ME uses the response-string configured in the registration-plan.
- [*session-config*]: The session-config to use for formatting the response headers and bodies.

Example

```
NNOS-E>proxy-reject abc123 404 Forbidden config1
```

prune-assoc

Immediately removes inactive associations from the location database to reclaim memory. You can also configure this to happen at a regular interval by enabling the prune-association property and defining the frequency with the **pruning-interval** property, both in the **settings** object.

Syntax

```
prune-assoc
```

Example

```
NNOS-E>prune-assoc
Success!
```

radius

Enables, disables, or tests a previously configured server that is part of a RADIUS group. Enter a reference to the configured server (configured with the RADIUS group **server** object). Enclose the reference path in quotation marks, and keep in mind that the server name is case-sensitive.

When using the **test** action, you can validate user credentials on the server via the ME. When invoked, the ME sends a test authentication message to the server to ensure that the RADIUS server is configured properly. The RADIUS server has a list of users and their associated passwords. This action verifies the name and, optionally, password, as well as the Digest settings for the user. Enter the following:

- **radiusServerReference**: A reference to the configured server and group. Enter this in quotation marks in the format "*groupNamePath*\server *ipAddress*"
- **userName**: The user name to test. This is the name configured on the RADIUS server.
- **password**: *Optional*. The password associated with the specified user name, as configured on the RADIUS server.
- **digest**: *Optional*. The setting for Digest use for the specified user name. Enter **true** (user sends Digest requests) or **false**. The default is true.
- **digestRealm**: *Optional*. The realm the Digest user is associated with. The default setting is **testrealm**, which is recognized and accepted by most RADIUS servers.

Syntax

```
radius deactivate radiusServerReference
radius reactivate radiusServerReference
radius test radiusServerReference userName [password] [true | false] [digestRealm]
```

Example

```
NNOS-E>radius reactivate "vsp radius-group East server Boston"
Invalid object
NNOS-E>radius reactivate "vsp radius-group East server boston"
Success!
NNOS-E>
```

The following examples illustrate the **radius test** action. In the first example, the test succeeds because the Digest

```
NNOS-E>radius test "vsp\radius-group East\server 172.26.0.147" user1 password1
false
RADIUS test authentication:
User name: user1
Password: password1
Type: Normal
Result: Accept
```

The following RADIUS test fails because it is a Digest request (Type=Digest) and the user is not configured on the RADIUS server for Digest:

```
NNOS-E>radius test "vsp\radius-group East\server 172.26.0.147" user1 password1

RADIUS test authentication:
User name: user1
Password: password1
Type: Digest
Realm: testrealm
Result: Reject
Authentication attempt failed
```

raid-check-consistency

This action only applies to the NN 2620 with RAID controller. Starts or stops a consistency check on the specified RAID logical volume. The consistency check compares images on mirrored or mirrored/striped drives, and reports inconsistencies to the event log. (The RAID controller automatically manages resynchronization of inconsistent logical volumes.)

Syntax

```
raid-check-consistency {start | abort} {L0 | L1}
```

Example

```
NNOS-E>raid-check-consistency start L0
Invalid provider
```

raid-set-adapter

These actions only apply to the NN 2620 with RAID controller. Sets and manages thresholds, rebuild rates, and refresh intervals for the RAID controller. In addition, you can control (silence) the audible alarm.

Select one of the following operations:

- **alarm**: Enables or disables the audible alarm that indicates that a logical volume is not optimal (e.g., physical drive dead, logical drive out of sync). Set to **enabled** to activate the alarm; **disabled** deactivates it. The **silence** option stops the sound of the current alarm, leaving the feature enabled.
- **cache-flush-interval**: Sets the number of seconds between flushes of the RAID controllers battery cache. The option sends the contents of the battery cache memory to the logical drives. Enter a value between 0 and 255.
- **rebuild-rate**: Sets the percentage of the compute cycles that are dedicated to rebuilding data onto a new physical disk after a drive has failed.
- **patrol-read-rate**: Sets the percentage of the compute cycles that are dedicated to preventative scanning. A patrol read scans the system for possible physical disk drive errors that could lead to drive failure. It helps protect data integrity by taking corrective action on the error before failure occurs.
- **cc-rate**: Sets the percentage of the compute cycles that are dedicated to a consistency check of data across logical volumes.
- **recon-rate**: Sets the percentage of the compute cycles that are dedicated to reconstruction (resynchronization and copy) operations. This activity is undertaken automatically by the RAID controller if the logical volumes are not synchronized.
- **pred-fail-poll-interval**: Sets the number of seconds between polls of the hard drives for reliability status. Enter a value between 0 and 65535.
- **battery-warn**: Enables or disables the battery warning message displayed in BIOS. This is the battery backup capability for the RAID array. If **enabled**, when BIOS boots, if it detects low or no battery, it displays a message to the screen.

Syntax

```
raid-set-adapter alarm {enabled | disabled | silence}
raid-set-adapter cache-flush-interval seconds
raid-set-adapter rebuild-rate percentage
raid-set-adapter patrol-read-rate percentage
raid-set-adapter cc-rate percentage
raid-set-adapter recon-rate percentage
raid-set-adapter pred-fail-poll-interval seconds
raid-set-adapter battery-warn {enabled | disabled}
```

Example

```
NNOS-E>raid-set-adapter alarm silence
Success!
```

reg-lookup

Displays, for a specified URI, the **registration-plan** settings that the ME assigned, the routing arbitration process, and the selected server. This action simulates the registration routing path, but does not actually trigger an outbound call. It exercises both registration plan and location database lookups. Its output indicates which registration plan entry (or location cache entry) the registration would use and the next hop.

Enter a To URI. The action returns results for any AOR that matches a configured registration plan, or you can find the URI of interest in the AOR field displayed with the **show location-cache** command.

Syntax

```
reg-lookup toUri [soureceIP] [localPort]
```

Example

```
NNOS-E>reg-lookup sip:4135555555@company.com
Resulting priority 200 sequential hunting total 2 next 0

All matching routes:
route phone 413555!* priority 200 best yes
This call will be forwarded to 12.39.208.251 transport UDP port 5060
```

reg-lookup-detail

Displays, for a specified To URI, the content of the session configuration associated with the selected **registration-plan**. If the SIP URI matches a registration-plan (source-) **route** or **arbiter**, the action returns the session configuration for that entry (or the default session configuration if the entry does not have one assigned). If the registration-plan **route** has the **peer** set to server, and the server has a session configuration, this action displays the merged session configuration. (The server session configuration takes precedence over the **route** session configuration.)

Syntax

```
reg-lookup-detail toUri [soureceIP] [localPort]
```

Example

```
NNOS-E>reg-lookup-detail sip:2078548355@elmaple.com
Resulting session config merged from server/group egress:
sip-settings
mode auto-determine
transport any
port auto-determine
route-hdr none
route-hdr-use-fqdn enabled
route-hdr-uri-host
route-hdr-add-register-msg disabled
```

```
route-hdr-preprocess-strip disabled
lcs-compatibility disabled
in-server unknown
out-server unknown
utilize-contact enabled
add-contact-nat disabled
compress-signaling disabled
preserve-call-id disabled
preserve-cseq disabled
proxy-generate-100-trying
handle-3xx-locally enabled
handle-3xx-locally-lookup-original-invite disabled
session-timeout 300 seconds
session-duration-max 0 seconds
--more--
```

register

Executes an ME call using the ME's. This action allows you to bind a web endpoint to a particular URI. It creates a location cache entry and a unique binding that ties the remote application to the specified URI, allowing remote applications to start receiving calls for the URI without the need to statically configure a dial-plan that routes the calls to a web endpoint.

The URI is a SIP URI in the following formats:

```
sip:user@domain:port
```

When the **register** action is executed, the ME first verifies that the user has permission to register that URI. If not, the ME returns an "unauthorized" error message.

If the URI is valid, the ME performs a registration-plan lookup. If no matches are found, the ME returns a "no routes" error message. If a match is found, the ME creates a binding that ties the specified URI with the server returned by the registration-plan lookup. Along with the binding, the ME also creates an identifier that uniquely identifies the binding. This identifier persists throughout the lifetime of the binding. At the completion of a successful binding, the ME returns this identifier along with a "success" message.

When the URI sent in the register action is linked to an existing SIP server, the registration is executed asynchronously. The ME returns a "pending" message indicating that the application must monitor for a register event containing the result of the asynchronous registration.

Enter the following arguments:

- <*URI*>: The URI tied to this binding.
- [*expiration*]: The expiration time of the binding in seconds.

Syntax

```
register <URI> [expiration]
```

Example

```
NNOS-E>register sip:16175551237 30
```

registration

Manages the registration information for the ME, specifically the registration routing and client tables. Use the **show registration-clients** and **show registration-routing** commands to view bindings and statistics for the tables.

Enter one of the following:

- **refresh**: Resends a REGISTER for each entry to reset the client binding. The ME sends the register to a specific peer, if specified, or all peers if not. To send only to a specific peer, enter the SIP URI set in the **server peer-identity** property.
- **purge**: Clears all entries from the registration client table.
- **query**: Sends a REGISTER to the specified peer with a query regarding a specified peer. The peer returns its current binding. Enter the address of record you are interested in and the server to which the ME should send the request.
- **proxy**: Simulates the registration proxy process. Instead of sending a registration on behalf of AORs in the location database, the registration proxy sends a registration manually for the specified AOR.
- **client-save**: Restarts collection of client state information if collection was stopped using the **client-clear** option.
- **client-clear**: Purges the state information from the client table and stops the future collection of client state data. Use **client-save** to restart client state collection.
- **status-clear**: Clears all registration status counters. These are the entries that can be viewed with the **show registration-status**, **show registration-delegate-status**, and **show registration-proxy-status** commands.
- **provider-enabled**: Enables the location information services provider.
- **provider-disabled**: Disables the location information services provider.

```
registration refresh [peerURI] |
registration purge [peerURI]
registration query AORpeer peerURI
registration proxy AORpeer peerURI fromURI contactHeader
registration client-save
registration client-clear
registration status-clear
registration provider-enabled
registration provider-disabled
```

Example

```
NNOS-E>registration client-save
Success!
NNOS-E>registration refresh
Success!
```

remove-device

Removes the specified device from the list of devices that automatically mount at boot-up. Use this action before physically removing a drive from the system. Before executing this action, ensure that you do not have any configuration pointing to mounts that are on the removed device (e.g., recorded files, logs), as writes to that device will fail.

Syntax

```
remove-device {data-1 | data-2}
```

Example

```
NNOS-E>remove-device data-2
Are you sure (y or n)?y
Changes will take effect at the next restart
```

restart

Causes an immediate restart of the type specified. To restart after gracefully terminating any existing connections, use the **orderly-restart** action. Select one of the following restart types:

- **warm**: Exits and restarts the application.
- **cold**: Exits the operating system and then restarts.
- **halt**: Stops the ME and does not restart the system.
- **cnx0**, **cnx1**: Stops and restarts the specified CNX card.
- **cluster**: Performs a warm restart of all boxes within the cluster.
- **controlled**: Restarts members in a cluster without interrupting call flow.
- **cancel**: Cancels the operations initiated by the **restart controlled** action.

The default type is **warm**.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
restart {warm | cold | halt | cnx0 | cnx1 | cluster | controlled | cancel}
```

Example

```
NNOS-E>restart warm
Are you sure (y or n)? y
NNOS-E is restarting...
Acme Packet Net-Net OS-E
Copyright (c) 2004-2006 Acme Packet, Inc.
NNOS-E>
NNOS-E> restart controlled
Are you sure (y or n)? y
restarting 172.66.0.10...
restarting 172.66.0.11...
restarting 172.66.0.12...
NNOS-E is restarting...
```

restore-defaults

Resets the ME configuration settings to the factory defaults and executes a cold restart of the system. Use this with care, as your startup configuration is deleted.

Syntax

```
restore-defaults
```

Example

```
NNOS-E>restore-defaults
Are you sure (y or n)? y
NNOS-E is restarting...
```

restore-stick-create

Creates a bootable USB recovery stick by copying system images to a USB stick plugged into the USB port of the system. When you select the **full-backup** option, the default, the current image on the ME, including application and configuration files, and all associated software, is written to the stick. The recovery image does not include copies of the ME database, system tar files (.gz), or of media files on the system at the time of creation. When you select the **config-backup** option, just the current configuration file is written to the stick.

Determine how often to create/update the recovery stick based on the frequency of configuration changes to your system. capture the current software, certificates, and operating system image to the USB stick. Oracle recommends that you use restore-stick-create to preserve the image prior to performing a system software upgrade, or whenever you have made significant and reliable changes to the system configuration.

Refer to the *Oracle Communications WSC Installation Guide* for more information on use of the recovery stick.

Syntax

```
restore-stick-create {full-backup | config-backup}
```

Example

```
NNOS-E>restore-stick-create full-backup
Starting rescue-stick-create as a background operation.
 -- this could 10 minutes or longer --

Please use the USB stick's activity light as an indication when this operation is
complete.
```

route-server

Executes actions for the route server. The following are operations you can execute with this action.

- **replace-file**: Replaces the route-server routing table with the specified file.
- **commit**: Commits replaced route-server routing tables.
- **replace-url**: Replaces the route-server routing table with the specified file URL.
- **flush**: Flushes the route-server routing table.
- **revert**: Revers a previous route-server routing table change.
- **lookup**: Looks up least cost routes.
- **load**: Loads a route.xml file into a temporary non-active routing-table that can be referenced with the *table* name.
- **drop**: Removes a previously loaded routing table from memory.

Syntax

```
route-server replace-file <file>
route-server commit
route-server replace-url <source>
route-server flush
route-server revert
route-server lookup <to-url> [from-url] [table] [time] [display-mode]
route-server load <file> <table>
route-server drop <table>
```

Example

```
NNOS-E>route-server replace-file 06222011.xml
Route-Server action in progress, check route-server-action-status for result.

NNOS-E>route-server commit
Route-Server action in progress, check route-server-action-status for result.

NNOS-E>route-server replace-url http://www.acme.com
Route-Server action in progress, check route-server-action-status for result.

NNOS-E>route-server flush
Route-Server action in progress, check route-server-action-status for result.

NNOS-E>route-server revert
Route-Server action in progress, check route-server-action-status for result.

NNOS-E>route-server lookup http://www.acme.com http://www.abc.com table1
Route-Server action in progress, check route-server-action-status for result.

NNOS-E>route-server load 06222011.xml table1
Route-Server action in progress, check route-server-action-status for result.

NNOS-E>route-server drop table1
Route-Server action in progress, check route-server-action-status for result.
```

route-server

Provides utilities for route-server. This action is only available from the CLI if the **route-server** master service is enabled. Select one of the following actions:

- **replace-file**: Replaces the existing route server routing table with the contents of the specified file. Use this to update the routing definition database on the route server. Enter the name of a properly formatted XML file; the default file name is /cxc/carrier_routing.xml. You cannot specify either the current or the most recent backup routing file. (To replace the routing table with the most recent backup, use the **revert** option.)
- **commit**: Writes appended route server routing table entries from memory to the routing table, clearing the memory. (The memory provides a temporary backup holding area.)
- **replace-url**: Replaces the existing route server routing table with the contents of the file found at the specified URL. Use this to update the routing definition database on the route server. Enter the name of a properly formatted XML file; the default file name is /cxc/carrier_routing.xml.
- **flush**: Removes all entries from the route server routing table on the master.

- **revert**: Reverts the existing route server routing table to the table in use prior to the last update.
- **lookup**: Tests retrieval from the route server routing table. Results display all routes that match the specified To, and optionally the From, URI(s).

Syntax

```
route-server replace-file fileName
lroute-servercr commit
route-server replace-url urlSource
route-server flush
route-server revert
route-server lookup toURL [fromURL]
```

Example

```
NNOS-E>route-server lookup 9788972990@acmepacket.com 7818972990@company.com
-----------------------------------------------------------------
Carrier                      Endpoint                   Mapping
-----------------------------------------------------------------
S - 10pct Mup Customer,    gateway1        ANI:7819376550
S - 10pct Mup Customer,    gateway10       ANI:7819376550
N - 10pct Mup Customer,    gateway4
-----------------------------------------------------------------
Total routes: 3
```

route-server-controlled

This action allows you to manually verify that route servers on each ME in a cluster are synced up properly during a route-set update, activating a new route-set, deleting a backup, or cancelling a controlled update or activation. When executed, the master ME controls and checks the success of each operation on each of the ME slaves.

Any error that occurs during the upgrade and activation processes, either on the master or any slave, results in the master initiating a rollback. The entire operation is retried and all failures that occur are logged and traced.

Before executing the **route-server-controlled** action, both NTP and logging must be configured on all MEs. This action must always be executed by the master. Any attempt to execute this action on a slave results in the error, "Execute action on master."

The following are operations you can execute with this action:

- **route-server-controlled update <file> [activate-time] [peer-wait-seconds]**: Allows you to replace the route-set used by the cluster while ensuring the route server databases on each ME are properly synced. You can optionally configure the specific time for this action to be executed, as well as how many seconds the master will wait for a peer to finish each step in the action. The master allows each peer three attempts at a step. The first attempt, the master waits the configured number of seconds. The second try, the master waits twice the configured number of seconds, and the third time three times the number of seconds before the master will halt the entire action.
- **route-server-controlled activation [activate-time] [peer-wait-seconds]**: Activate a new route-set used by the cluster while ensuring the route server databases on each ME are properly synced. You can optionally configure the specific time for this action to be executed, as well as how many seconds the master will wait for a

peer to finish each step in the action. The master allows each peer three attempts at a step. The first attempt, the master waits the configured number of seconds. The second try, the master waits twice the configured number of seconds, and the third time three times the number of seconds before the master will halt the entire action.

- **route-server-controlled delete-backup <backup-name>**: Delete a backup route-set that the cluster does not use anymore while ensuring the route server databases on each ME are properly synced.
- **route-server-controlled cancel [peer-wait-seconds]**: Cancel a controlled update or activation that is currently in progress. You can optionally configure the number of seconds the master will wait for a peer to finish each step in the action. The master allows each peer three attempts at a step. The first attempt, the master waits the configured number of seconds. The second try, the master waits twice the configured number of seconds, and the third time three times the number of seconds before the master will halt the entire action.

Both the master and slave ME cycle through a set of states during a controlled update. The following table shows the states a master ME goes through.

Table 4–1 Master ME States

State	Description
Ready	The master is ready to receive a new action request.
Loading	The master is loading.
Peers_Fetching	Waiting for all slaves to get the file from the master.
Peers_Loading	Waiting for all slaves to load.
Peers_Activating	Waiting for all slaves to activate before the master activates itself.
Peers_Cancelling	Cancel the current operation.
Initializing	The master ME is initializing its state.
Activate_Scheduled	A controlled activation is scheduled.

The following table shows the states a slave ME goes through.

Table 4–2 Slave ME States

State	Description
Ready	The slave is ready to receive a new action request.
Fetching file	Received route-set .xml file from master.
Loading	Slave is loading.
Activating	Slave is activating.
Cancelling	Master requested a cancel.

Syntax

```
route-server-controlled update <file> [activate-time] [peer-wait-seconds]
route-server-controlled activation [activate-time] [peer-wait-seconds]
route-server-controlled delete-backup <backup-name>
route-server-controlled cancel [peer-wait-seconds]
```

Example

```
Cluster2>route-server-controlled update /cxc_common/rs/rsdid_201004131550.xml
Route-Server action in progress, check route-server-controlled-action-status for
result.
```

route-server-test

You can test imported DID ranges and prefix changes you have made in the route-server import tool before activating them in a live environment. Via the **route-server-test** action you can test routes, CDRs, and queries, and analyze, compare, and validate results of the routes.

Enter one of the following:

- **config**: Generates a series of test vectors derived from the routes.xml file and outputs them to a specified test.xml file. If you do not specify a text.xml file, the ME writes the resulting output to the screen. The test.sml file has the following format.

```
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mgmt_data.xsd">
<RouteServerTestSuite suite="1">
 <description>DID 1000-1002</description>
 <tests>
  <RouteServerTestCase>
   <query>1000-1002</query>
   <from/>
   <time></time>
  </RouteServerTestCase>
 </tests>
</RouteServerTestSuite>
</config>
```

- **cdr**: Generates a series of test vectors derived from accounting records in the CSV format and outputs them to a specified test.xml file. If you do not specify a test.xml file, the ME writes the resulting output to the screen. The test.xml file has the following format.

```
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mgmt_data.xsd">
<RouteServerTestSuite suite="1">
 <description>DID 1000-1002</description>
 <tests>
  <RouteServerTestCase>
   <query>1000-1002</query>
   <from/>
   <time></time>
  </RouteServerTestCase>
 </tests>
</RouteServerTestSuite>
</config>
```

The CSV file has the following format.

```
"SessionID","Recorded","CallID","To","From","Method","IncomingRequestURI","Previou
sHopIp","PreviousHopVia","OutgoingRequestURI","NextHopIp","NextHopDn","Header","Or
igin","SetupTime","ConnectTime","DisconnectTime","DisconnectCause","Duration","scp
Name","CallID2","OrigGW","TermGW","PacketsReceivedOnSrcLeg","PacketsLostOnSrcLeg",
"PacketsDiscardedOnSrcLeg","PdvOnSrcLeg","MaxJitterOnSrcLeg","CodecOnSrc
Leg","MimeTypeOnSrcLeg","LatencyOnSrcLeg","MaxLatencyOnSrcLeg","RFactorOnSrcLeg","
PacketsReceivedOnDestLeg","PacketsLostOnDestLeg","PacketsDiscardedOnDestLeg","PdvO
nDestLeg","MaxJitterOnDestLeg","CodecOnDestLeg","MimeTypeOnDestLeg","LatencyOnDest
```

```
Leg","MaxLatencyOnDestLeg","RFactorOnDestLeg","Rx1000FactorOnDestLeg","Rx1000Facto
rOnSrcLeg","MOSFmtOnDestLeg","MOSFmtOnSrcLeg","callType","disconnectErrorType","an
i","callSourceRegid","callDestRegid","newAni","cdrType","huntingAttempts","callPDD
","callSourceRealmName","callDestRealmName","callDestCRName","in_peer_dst","in_
anchor_src","in_anchor_dst","in_peer_src","out_peer_dst","out_anchor_src","out_
anchor_dst","out_peer_
src","calledPartyAfterSrcCallingPlan","lastStatusMessage","LastMediaPktTimestampOn
DestLeg","LastMediaPktTimestampOnSrcLeg","SetupTimeInt","IncomingURIStripped","dni
s","newDnis","customData","CreationTimestamp"
```

- lookup: Uses the test vectors generated from the **route-server-test config** and **route-server-test cdr** actions and queries the route-server. The results of the queries are outputted to a specified results.xml file. If you do not specify a result.xml file, the ME writes the resulting output to the screen. The result.xml file has the following format.

```
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mgmt_data.xsd">
<RouteServerTestSuiteResults suite="1">
 <description>DID 1000-1002</description>
 <results>
  <RouteServerTestResult>
   <query>1000</query>
   <from/>
   <time></time>
   <routes>
    <RouteServerTestRouteResult>
     <match>route-plan:2</match>
     <carrier>default</carrier>
     <endpoint>a.example.net</endpoint>
    </RouteServerTestRouteResult>
   </routes>
  </RouteServerTestResult>
 </results>
 <results>
  <RouteServerTestResult>
   <query>1001</query>
   <from/>
   <time></time>
   <routes>
    <RouteServerTestRouteResult>
     <match>route-plan:2</match>
     <carrier>default</carrier>
     <endpoint>a.example.net</endpoint>
    </RouteServerTestRouteResult>
   </routes>
  </RouteServerTestResult>
 </results>
 <results>
  <RouteServerTestResult>
   <query>1002</query>
   <from/>
   <time></time>
   <routes>
    <RouteServerTestRouteResult>
     <match>route-plan:2</match>
     <carrier>default</carrier>
     <endpoint>a.example.net</endpoint>
    </RouteServerTestRouteResult>
   </routes>
  </RouteServerTestResult>
```

```
 </results>
</RouteServerTestSuiteResults>
</config>
```

You can also execute this action with the optional **table** parameter. This allows you to execute the lookup in a different routing table other than the currently active one.

- **analyze**: Analyzes the results file generated by the **route-server-test lookup** action and summarizes the results. The results of the analysis are outputted to a specified analysis.xml file. If you do not specify an analysis.xml file, the ME writes the resulting output to the screen. This action allows you to view how various resources are utilized with the current routing configuration being tested. The output of the analysis has the following format.

```
Analysis of results file : /tmp/results.xml
           Analysis created on : 13:09:56.729762 Mon 2010-11-01
             Total test suites : 1
              Total test cases : 3
     Total results with routes : 3
  Total results without routes : 0
          Smallest hunt result : 1
           Largest hunt result : 1

 Route position : 1
         Total : 3
-----------------------------------------------------
        Carrier "default" referenced 3 times
           |--- Endpoint "a.example.net" referenced 3 times


        Route "route-plan:2" referenced 3 times

Queries with no route results:
-----------------------------------------------------
None
```

- **compare**: Compares two results files and outputs the differences to a specified diff-results.xml file. If you do not specify a diff-results.xml file, the ME writes the output to the screen. The output of the comparison has the following format.

```
Comparing results from file /tmp/results.xml with /tmp/results2.xml
-----------------------------------------------------------------------
Test suite 1 "DID 1000-1002"
   |--- Query "1001"
          |--- Route 0
                 |--- name "route-plan:2" not equal "route-plan:3"
```

- **validate**: Compares the results file with the active routes and outputs the differences to a specified validation-results.xml file. If you do not specify a validation-results.xml file, the ME writes the output to the screen. Any differences between the results.xml file and the active routing tables has the following format.

```
Comparing results from file /tmp/results.xml with /tmp/results2.xml
-----------------------------------------------------------------------
Test suite 1 "DID 1000-1002"
   |--- Query "1001"
          |--- Route 0
                 |--- name "route-plan:2" not equal "route-plan:3"
```

You can also execute this action with the optional **table** parameter. This allows you to execute the validation in a different routing table other than the currently active one.

Syntax

```
route-server-test config [file] [test-vector-file]
route-server-test cdr filename [test-vector-file]
route-server-test lookup test-vector-file [test-results-file] [table]
route-server-test analyze test-results-file [analysis-results-file]
route-server-test compare test-results-file1 test-results-file2
[diff-results-file]
route-server-test validate test-results-file [validation-results-file] [table]
```

Example

```
NNOS-E>route-server-test config /cxc_common/rs/files/routes.xml
<RouteServerTestSuite suite="293">
 <description>DID 3612888825-3612888828</description>
 <tests>
  <RouteServerTestCase>
   <query>3612888825-3612888828</query>
   <from/>
   <time></time>
  </RouteServerTestCase>
 </tests>
</RouteServerTestSuite>
<RouteServerTestSuite suite="294">
 <description>DID 3612888800-3612888899</description>
 <tests>
  <RouteServerTestCase>
   <query>3612888800-3612888899</query>
   <from/>
   <time></time>
  </RouteServerTestCase>
 </tests>
</RouteServerTestSuite>

NNOS-E>route-server-test lookup /tmp/testcases.xml /tmp/results.xml

<RouteServerTestSuiteResults suite="293">
 <description>DID 3612888825-3612888828</description>
 <results>
  <RouteServerTestResult>
   <query>3612888825</query>
   <from/>
   <time></time>
  </RouteServerTestResult>
 </results>
 ....
</RouteServerTestSuiteResults>

NNOS-E>route-server-test analyze /tmp/results.xml

      Analysis of results file : /tmp/results.xml
           Analysis created on : 10:03:10.492217 Thu 2011-06-23
             Total test suites : 2
              Total test cases : 104
     Total results with routes : 103
  Total results without routes : 1
         Smallest route result : 1
          Largest route result : 0

Queries with no route results:
```

```
--------------------------------------------------------------------------------
        Suite 0 test 1 query "3612888825" from (null) time ""

NNOS-E>route-server-test compare /tmp/results.xml /tmp/results2.xml

Comparing results from file /tmp/results.xml with /tmp/results.xml
--------------------------------------------------------------------------------
Results are identical.

NNOS-E>route-server-test validate /tmp/results.xml

Comparing results from file /tmp/results.xml with active routes
--------------------------------------------------------------------------------
Results are identical.
```

rtp-cache-delete

Deletes the specified WAV file or DTMF event from the RTP cache, if the entry is inactive. The ME caches copies of encoded media stream data for music-on-hold and periodic/introduction announcements, instead of re-encoding them for each call. However, if the configuration no longer uses an announcement or if use of a particular CODEC is discontinued, you can free up memory resources occupied by unused announcements by cleaning the cache of these unused entries. The ME checks to make sure the entry is not currently streaming to an on-hold call.

Use the **show rtp-cache** command output to find the required file name or event ID. Delete based on either:

- **file**: Enter the file name of the WAV file to delete.
- **event**: Enter the event ID for the entry, and optionally, the volume. DTMF events display as "Event=event_id/volume" in the **show rtp-cache** output.

Syntax

```
rtp-cache-delete file filename [codec] [packetTime]
rtp-cache-delete event eventID [volume] [codec] [packetTime]
```

Example

```
NNOS-E>rtp-cache-delete file announce1.wav
Success!

NNOS-E>show rtp-cache
Type  Cache  Codec  PacketTime  Current-Streams  Pkts-Cached  Pkts-Sent
----  -----  -----  ---------   ---------------  -----------  ---------
event  0/10   g723   30           0               33           0
              g728   20           0               50           0
              g729   20           0               50           0
              iLBC   30           0               33           0
              pcma   20           0               50           0

NNOS-E>rtp-cache-delete event 0 10
Success!

NNOS-E> show rtp-cache
Type  Cache  Codec  PacketTime  Current-Streams  Pkts-Cached  Pkts-Sent
----  -----  -----  ----------  ---------------  -----------  ---------
event  1/10  gsm     20          0               50           0
```

```
g723        30          0                33          0
g728        20          0                50          0
```

rtp-header

Decodes binary RTP packets (the output of the **srtp** action). The action results in output that prints out the fields of the header, as defined in RFC 1889, RTP: A Transport Protocol for Real-Time Applications. To execute, enter the hexadecimal value of the binary RTP header. The action returns

- the version and any additional flags
- RFC-defined payload type
- The sequence number, used to detect missing or out-of-order packets
- The timestamp, a sample count used to synchronize RTP sequences
- The source synchronization ID.

Syntax

```
rtp-header packet-header
```

Example

```
NNOS-E>rtp-header 0x8070c351ed6508afff311f7b
Version       : RFC-1889
Payload type  : 112 (0x70) Sequence      : 50001 (0xc351) Timestamp     :
3982821551 (0xed6508af) SSRC          : 4281409403 (0xff311f7b)
```

rtp-stream

Displays the packets that make up a recording. The options allow you to control the detail level of the display and also to create an XML file from a WAV file.

Select either:

- **details**: Displays a full or compressed list of all packets in the RTP stream that made up the recording. Enter a file name that contains the recording. You can also select to display timestamps. Select **delta** to display the time passed since the last packet. Select **relative** to display timestamps relative to the beginning of the call. Select **absolute** to display the complete date/time string. By default (**none**) the output displays no time stamps.

The first true/false boolean controls display of an interpretive line highlighting potential changes. If **true**, the default, the output includes the line. The second true/false boolean controls compression. If **true** (the default), the output only displays a summary of packets as long as the sequence is as expected (no change to SSRC). Each unexpected line is displayed. If set to **false**, the output lists all packets.

- **summary**: Displays high-level (header) information that summarizes the packets in the RTP stream.
- **stats**: Displays MOS and other information for recorded files, including the CODEC in use, packet counts, jitter and others.
- **create**: Creates and encodes an XML file from a WAV file. Optionally you can set the packetization rate.

Syntax

```
rtp-stream details xmlSource {none | delta | relative | absolute} [true | false]
[true | false]
rtp-stream summary xmlSource
rtp-stream stats xmlSource
rtp-stream create wavSource xmlDest codec [packetTime]
```

Example

```
NNOS-E> rtp-stream summary /cxc_common/recorded/sess-04c2a2312f70ca7a-0-1.xml
Filename: /cxc_common/recorded/sess-04c2a2312f70ca7a-0-1.xml
Start Time: 15:01:56.458930 Fri 2007-07-13
Packet-time: 20
2 rtpmaps:
PCMU payload-type=0, sample-rate=8000, channels=1   telephone-event
payload-type=101, sample-rate=8000, channels=1
Success!

NNOS-E> rtp-stream details /cxc_common/recorded/sess-04c2a2312f70ca7a-0-1.xml
relative
1     34 RTP Invalid: RTP version is invalid
***** SSRC changed from 0 to 1067843479 ******
2    134 Payload type=PCMU, SSRC=1067843479, Seq=43432, Time=1373220824, Mark,
payload bytes=160
3-162  Payload type=PCMU, SSRC=1067843479
163   3352 Payload type=PCMU, SSRC=1067843479, Seq=43593, Time=1373246584, payload
bytes=160
***** Missed 1 sequence numbers ******
164   6679 Payload type=PCMU, SSRC=1067843479, Seq=43595, Time=1373273064, payload
bytes=160
165   6700 Payload type=PCMU, SSRC=1067843479, Seq=43596, Time=1373273380, payload
bytes=160
***** Timestamp discontinuity (possible silence): old(1373273380) + expected(316)
!= new(1373273540) ******
166   6721 Payload type=PCMU, SSRC=1067843479, Seq=43597, Time=1373273540, payload
bytes=160
167-389  Payload type=PCMU, SSRC=1067843479
390  11197 Payload type=PCMU, SSRC=1067843479, Seq=43821, Time=1373309380, payload
bytes=160

NNOS-E>rtp-stream stats /cxc_common/rtp_
recorded/04c30cd29/sess-04c30cd2909dcc70-0-1.xml
Codec           : PCMU (pt=0)
SSRC            : 2691206733 (0x4d8e68a0)
Duration        : 8.74 seconds
Packet interval : 20 msecs
Jitter (sum)    : 116.0 msecs
Received        : 437
Lost            : 0
MOS             : 4.39
```

rule-failover

Deletes an individual entry from or flushes the rule failover database. This database contains rules that are internally created by the third-party call control process to do advanced call control operations. You can list current rules with the **show automatic-rules** command. This action is for Technical Support use only.

Syntax

```
rule-failover {delete entry | flush}
```

Example

```
NNOS-E>rule-failover delete rule-24
Success!
```

script

For Technical Support use only.

Syntax

```
script filename [variable1] [variable2] [variable3] [variable4] [variable5]
[variable6] [variable7] [variable8] [variable9]
```

secret

Manages the ME passwords and tags. The ME uses this two-part password mechanism for passwords shared with other devices (also known as shared secrets). See Understanding Passwords and Tags for a complete explanation of this mechanism.

You can also set password/tag associations from various points within the configuration. This password mechanism applies does not apply to passwords created for users under the **access** object. Use the **show secrets** command to view configured password tags.

Enter one of the following:

- **set**: Creates a password/tag association. If you re-execute this action, and supply a different password, the ME overwrites the password that was associated with the tag with the new password. When you set an association, you supply a tag. The system then prompts you for the secret (password). The tag is what users enter, the secret is the password known to the other device. Tags cannot contain the pound symbol (#). If you do not specify a tag, the system saves the password without an associated tag.

  Note: You must manually enter passwords on each ME device. Because passwords are maintained in a separate store, simply copying the configuration file between devices does not copy the password store.

- **delete**: Removes a secret so that the association between tag and secret no longer exists.
- **root**: Resets the Linux root password. When prompted, specify and confirm the new root password. The root secret must be at least four characters long.
- **ssh**: Sets the SSH account password. When prompted, specify and confirm the new password.
- **synchronize**: Copies passwords to other devices in the cluster. Passwords are maintained in a separate store; simply copying the configuration file between devices does not copy the password store. Use this action on the master device to copy your passwords the other devices in the cluster.

- **verify**: Confirms a secret that is associated with the specified tag. Enter the tag, and the system prompts you for the secret. If you did not associate a secret with a tag, you get a secret mismatch message.

Syntax

```
secret set tag secret
secret delete tag
secret root password
secret ssh password
secret synchronize
secret verify tag secret
```

Example

```
NNOS-E>secret set red
password: ***********
confirm: ***********
Success!

NNOS-E>secret verify red
secret: ***********
Success!

NNOS-E>show secrets

tag
---
red
```

send-notify-event

Sends an event within the body of a NOTIFY message to the phone at the specified URL. If you select one of the preconfigured events (reboot, resync, restart, or report), the ME sends the event expected by a Sipura phone. To send an event to any other type of phone, enter the appropriate string. Note that some phones and event types require credentials, so username and password may be required. Also, optionally you can specify which interface the NOTIFY message goes out. In the **tag** field, enter a configured **classification-tag** from the **ip** object.

Use this action, for example, to send an event that will reboot the phone, check for configuration changes, and/or download a configuration.

Syntax

```
send-notify-event {string | reboot | resync | restart | report} URL [from] [tag]
[any | UDP | TCP | TLS] [username] [password]
```

Example

```
NNOS-E>send-notify-event check-sync sip:2125551212@voip.acmepacket.com
--- End of Data ---
SIP Tx: [09:58:23.193602]   457 bytes to 66.10.143.110:22324 on vx1 (UDP socket 69
- 82.134.77.12:5060):
--- Start of Data ---
NOTIFY sip:2125551212@66.10.143.110;transport=UDP SIP/2.0
From:
<sip:2125551212@66.10.143.110:22324>;tag=af4da8c0-13c4-469f6dfd-3f1eefb-2ac0a04f;r
```

```
instance=b6620d67f3884ca8
To: <sip:2125551212@66.10.143.110:22324>;rinstance=b6620d67f3884c
Call-ID: NNOS-E-1-af4da8c0-13c4-469f6dfd-3f1eefb-5f0726f5
CSeq: 1 NOTIFY
Via: SIP/2.0/UDP 82.134.77.12:5060;branch=z9hG4bK-678d-469f6dfd-3f1eefb-d68961
Event: check-sync
Subscription State: Active
Content-Length: 0
```

sensor

Manipulates elements of the sensor management system. Enter one of the following:

- **delete-events**: Clears the sensor event log. Use the show sensor-events command to view the contents of the log prior to deleting. The sensor event log maintains information pertaining to such events as temperature, voltage, and others.
- **identify**: Activates a blue light on the front of the system that can be used to identify the system you are working with. By executing the sensor identify action, the light will begin to blink and will continue for the number of seconds specified. Enter a value from 1 to 127. The default is 60 seconds.
- **reset-processors**: Clears any error or disabled sensor states from all processors in the system. At boot time, the system performs a diagnostic test of its processors. If a processor fails, it is automatically disabled. (The system generates a log event if this happens; also **show sensors** and **show sensor-events** indicates the problem. use this action to begin recovery, followed by the **restart cold** action, causing the processor to be retested during the cold start.

Syntax

```
sensor delete-events
sensor identify [timeout]
sensor reset-processors
```

Example

```
NNOS-E>sensor identify
Success!
NNOS-E>sensor delete-events
Success!
NNOS-E>
```

server

Clears counters and configurations related to the configured servers. Enter one of the following:

- **routing-clear**: Clears dial plan and registration plan statistics counters. These are the counters that are visible using the **show dial-plan** and **show dial-plan** commands.
- **purge-dynamic**: For use with configured dns-group **servers**, purges the server-pool configuration. A dns-group server learns its server-pool configuration dynamically through DNS. If the data is not synchronized, for example the DNS server may have gone down and then come back up, you can execute this action to delete the server-pool configuration and rebuild it through incoming SIP traffic.

- **age-dynamic**: For use with configured dns-group **servers**. When initiated, the ME purges the server IP address, port, and transport protocol information (but maintains the rest of the server configuration). This forces a DNS query the next time the server is used.

Syntax

```
server {routing-clear | purge-dynamic | age-dynamic}
```

Example

```
NNOS-E>server routing-clear
Success!
NNOS-E>server purge-dynamic
Success!
```

service-route-lookup

Performs a lookup in the specified service route table. The output of the action returns the best route to the destination, given the filtering parameters applied. It also displays other aspects of the route such as the gateway, physical interface, geolocation, and metrics. The metrics are the resulting values of the **services-routing** metric assignments. For example, if you had assigned user-metric, the ME displays the cost of the route (configured with the **ip** metric property). If you had assigned intf-throughput, the ME displays the most recent calculation of interface throughput for the route.

You must enter the following arguments:

- **service table name**: Selects the specific routing table to search for the best route to the destination. Enter either sip, media, or stun.
- **destination**: Specifies the host address of the destination. Enter in IP address format.
- **transport protocol**: For a STUN service routing table lookup, specifies the transport protocol that the route must use to reach the specified destination. Each **stun-server** in the cluster is configured to support a particular protocol (UDP, TCP, or TLS) with the **port** property. The ME then returns the best route using that protocol.

Optionally you can enter:

- **partner IP address**: Specifies the address of a peer in a cluster network. If you use this argument, the output will return the best route from the specified peer. By default, the ME finds the best route from any box (255.255.255.255). Use 0.0.0.0 as the partnerIP address to return results from the local box.
- **load-balance**: Selects whether to load balance the results (choose true or false). If set to **true**, the ME uses a round robin algorithm to return results. Each time you execute the command, the next entry is returned. If set to **false**, the ME returns the first entry in the table (the default behavior).
- **geolocation**: Filters results based on the geolocation. This value is assigned to an interface with the **ip** object, and is stored with the route. When you specify a geolocation, the ME returns the best route to the destination that has the specified geolocation.

Syntax

```
service-route-lookup media destination [partnerIP] [true | false] [geolocation]
service-route-lookup sip destination [true | false] [geolocation]
service-route-lookup stun destination {udp | tcp | tls} [true | false]
```

Example

```
NNOS-E>service-route-lookup media 192.168.55.55

service-name:       media
destination:        192.168.55.55/32
gateway:            192.168.215.1
source-ip:          192.168.215.100
interface:          eth0
origin:             local
geo-location:       7654
metric1:            1
metric2:            0
metric3:            0
metric4:            0
metric5:            0
partner-ip-address: 0.0.0.0
```

set-call-forwarding

Sets the ME to forward any calls intended for the specified address-of-record to a specified URI. Once configured, you can then use the enable and disable arguments to activate the forwarding.

Syntax

```
set-call-forwarding aor {enabled | disabled} callForwardURI [cookie]
```

Example

```
NNOS-E>set-call-forwarding sip:jdoe@cov.com enabled sip:confRm1@cov.com
Success!
```

set-chassis-config-boot

Sets the system partition from which the ME boots. Use this, for example, to revert to an old system image after having installed and run off of a new one. In that case, set this to the alternate partition and reboot the system. You can use the **on-board-rescue** option to reboot using a limited functionality that provides access to USB stick rescue utilities without using the stick itself. Use the **show chassis-config** command to display the current partition assignment.

Syntax

```
set-chassis-config-boot {system-1 | system-2 | on-board-rescue}
```

Example

```
NNOS-E>set-chassis-config-boot system-2
Changes will take effect at the next cold restart
```

set-chassis-config-console

Configures the endpoint to which the ME directs console output the next time that it boots. To display output to the screen, set this to the port used for your management console. Use the **show chassis-config** command to display the current management console assignment.

Syntax

```
set-chassis-config-console {serial-0 | serial-1 | vga}
```

Example

```
NNOS-E>set-chassis-config-console system-2
Changes will take effect at the next cold restart
```

set-chassis-config-ipmi

Enables and disables IPMI functionality. By default, IPMI functionality is enabled, and allows system reports such as those returned by the **show sensors** command. Disable this only in cases where the IPMI functionality causes a problem with your specific platform. Use the **show chassis-config** command to view the current IPMI setting.

Syntax

```
set-chassis-config-ipmi {enabled | disabled}
```

Example

```
NNOS-E>set-chassis-config-ipmi disabled
Changes will take effect at the next cold restart
```

set-do-not-disturb

Sets the ME to return a busy response to any call directed to the specified address of record. The phone registered to that AOR will respond according to its configuration (busy, voice mail, etc.). Once configured, you can then use the enable and disable arguments to activate and deactivate the setting.

Syntax

```
set-do-not-disturb aor {enabled | disabled}
```

Example

```
NNOS-E>set-do-not-disturb sip:jdoe@cov.com
Success!
```

sip

Performs actions on SIP transport connections. Each action, and its specific syntax, is described below.

- **ping**: Pings the specified server, using the SIP OPTION message (instead of ICMP), to validate a SIP node. Enter a transport protocol and a port, if desired. The default protocol is UDP; the default port is 5060.

- **ping-aor**: Pings the specified address of record, using the SIP OPTION message (instead of ICMP), to validate a SIP node.
- **traceroute**: Sends SIP OPTION message trace packets to determine a routing path. The default protocol is UDP; the default port is 5060.
- **server-monitor**: Adds the specified server to the monitor pool. The ME periodically pings the servers in this pool to check availability. Use the **show sip-server-availability** command to view the results for all monitor pool entries. Enter a server hostname or IP address, and optionally, a transport protocol and port. The default protocol is UDP; the default port is 5060.
- **server-unload**: Removes the specified server from the monitor pool (see **server-monitor**, above). Enter a server hostname or IP address, and optionally, a transport protocol and port. The default protocol is UDP; the default port is 5060.
- **lookup-connection**: Does a lookup in the SIP connection table. Specify the endpoint IP address. The ME returns an entry if there is a connection between that endpoint and the ME. Optionally, you can set the transport protocol and/or a local IP address to perform the lookup from, rather than from the ME.
- **delete-connection**: Deletes the connection between the ME and the specified endpoint. Optionally, you can set the transport protocol and/or a local IP address to perform the lookup from, rather than from the ME.
- **purge-connection**: Disconnects and deletes all entries in the connection table. (This is all active calls on the ME.)
- **reset-connection**: Tears down and then resets all entries in the connection table.
- **clear-statistics**: Clears all counters associated with the connection table.
- **ping-resume**: Resumes sending SIP OPTION message ping packets to entries in the monitor pool. These packets would have been stopped with the **ping-suspend** option, below.
- **ping-suspend**: Temporarily halts sending SIP OPTION message ping packets to entries in the monitor pool. The ME uses the last known state for each server as its current state data. Use the **ping-resume** option, above, to restart sending packets.
- **udp-log-on**: Starts logging a partial header from each SIP UDP message to the UDP buffer. The UDP buffer is a FIFO buffer, the size of which is set with the **vsp** object **max-udp-outbound-log** property.
- **udp-log-off**: Turns off logging of SIP UDP messages to the UDP buffer. See the **udp-log-on** option, above.

Syntax

```
sip ping server [any | udp | tcp | tls] [port]
sip ping-aor aor
sip traceroute server [any | udp | tcp | tls] [port]
sip server-monitor server [any | udp | tcp | tls] [port]
sip server-unload server [any | udp | tcp | tls] [port]
sip lookup-connection remoteIP [any | udp | tcp | tls] [localIP]
sip delete-connection remoteIP [any | udp | tcp | tls] [localIP]
sip purge-connection
sip reset-connection
sip clear-statistics
sip ping-resume
sip ping-suspend
sip udp-log-on
```

```
sip udp-log-off
```

Example

```
NNOS-E>sip ping 172.26.0.143
Sending OPTIONS to 172.26.0.143:5060 UDP
Success! Received OPTIONS Response 200:
From: sip:172.26.0.153
To: sip:172.26.0.143
```

sip-send-info

The **sip-send-info** action sends out-of-dialog SIP INFO requests.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the registration binding (returned in RegisterEvent).
- [*session-config*]: The session-config used for formatting INFO headers and bodies.

Syntax

```
sip-send-info <AOR> [id] [session-config]
```

Example

```
NNOS-E>sip-send-info http://abc.com 654321 config1
```

sip-send-message

The **sip-send-message** action sends out-of-dialog SIP MESSAGE requests.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the registration binding (returned in RegisterEvent).
- [*session-config*]: The session-config used for formatting MESSAGE headers and bodies.

Syntax

```
sip-send-options <AOR> [id] [session-config]
```

Example

```
NNOS-E>sip-send-options http://abc.com 654321 config1
```

sip-send-notify

The **sip-send-notify** command sends out-of-dialog SIP NOTIFY requests.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.

- [*id*]: The Unique ID that identifies the subscription binding (returned in SubscribeEvent).
- [*session-config*]: The session-config used for formatting NOTIFY headers and bodies.

Syntax

```
sip-send-notify <AOR> [id] [session-config]
```

Example

```
NNOS-E>sip-send-notify http://abc.com 654321 config1
```

sip-send-options

The **sip-send-options** action sends an out-of-dialog SIP OPTIONS request.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the registration binding (returned in RegisterEvent).
- [*session-config*]: The session-config used for formatting OPTIONS headers and bodies.

Syntax

```
sip-send-options <AOR> [id] [session-config]
```

Example

```
NNOS-E>sip-send-options http://abc.com 654321 config1
```

sip-send-other

The **sip-send-other** action sends out-of-dialog requests for all other SIP methods which do not have a specific action.

Valid arguments for this action are:

- <*method*>: The SIP method to use in the request.
- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the registration binding (returned in RegisterEvent).
- [*session-config*]: The session-config used for formatting the request headers and bodies.

Syntax

```
sip-send-other <method> <AOR> [id] [session-config]
```

Example

```
NNOS-E>sip-send-other INVITE http://abc.com 654321 config1
```

sip-send-subscribe

The **sip-send-subscribe** action sends an out-of-dialog SIP SUBSCRIBE request.

When an application either sends a SUBSCRIBE request and receives a 200 OK or accepts a SUBSCRIBE request by responding with a 200 OK, the ME creates a subscription binding. This binding contains the supported events and a unique ID. The application can use this ID to distinguish between subscriptions that it created and other subscriptions created for the same AOR but a different application.

Subscription bindings are arranged in a vector under the AOR and have an expiration time that can be specified in the **sip-send-subscribe** and **call-control-send-subscribe** actions. If you do not specify an expiration time, the default is 3600 seconds.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*event*]: The name of the event type to subscribe to.
- [*accept*]: The acceptable event data to include.
- [*id*]: The unique ID that identifies the subscription binding (returned in SubscribeEvent).
- [*session-config*]: The session-config used for formatting SUBSCRIBE headers and bodies.
- [*expiration*]: The expiration time of the binding, in seconds. If this value is not specified, the registration-plan expiration is used.

Syntax

```
sip-send-subscribe <AOR> [event] [accept] [id] [session-config] [expiration]
```

Example

```
NNOS-E>sip-send-subscribe http://abc.com subscribeEvent 654321 config1 30
```

snmp-trap-test

The **snmp-trap-test** command allows you to generate any trap on the ME manually in order to test it without having to set up conditions to actually trigger the trap. After configuring your traps, filters, and logs, use this command to ensure they show up when triggered.

Syntax

```
snmp-trap-test <trap name>
```

Example

```
NNOS-E>snmp-trap-test sipcallfail
sessionID 0
from SNMPTestData
to SNMPTestData
reason SNMPTestData
action SNMPTestData
Result code 1: false, Result code 2: Success!
Generic system error
```

srtp

Provides a diagnostic tool for testing SRTP encryption/decryption of packets. When you select to decrypt a packet, use the same key and encryption suite that was used to originally encrypt the packet. When viewing the decrypt action output, if input and output are the same, the encryption is likely broken.

Select an action. The following fields are available for the **start** action.

- **action**: Specify whether to start, send, or stop the testing.
  - start: Sets up the context for the testing session.
  - send: Begins sending packets for testing.
  - stop: Tears down the testing session.
- **operation**: Specify whether to encrypt or decrypt the specified packet.
- **packetType**: Select whether to send RTP or RTCP packets, as defined in RFC 3550, "RTP: A Transport Protocol for Real-Time Applications."
- **suite**: Enter the SRTP protection suite consisting of encryption and authentication algorithms. Enter either:
  - None (no encryption or authentication)
  - AES-128 Countermode encryption, SHA-1 authentication (80 bit)
  - AES-128 Countermode encryption, SHA-1 authentication (32 bit)
  - F8-128 encryption, SHA-1 authentication
  - AES-128 Countermode encryption, MD5 authentication (Sipura)
  - DES Cipher Block Chaining per RFC-1889
- **key**: Enter the hexadecimal value of the master key/salt (it must include the "0x") that is derived from the client or through the ME tracing. This is the value typically passed in the SDP message, or the decrypted value of the key passed in Linksys INFO messages.
- **mkiLen**: Enter a value, in bytes, that sets the number of bytes in the master key identifier (MKI). Enter a value between 0 and 4. A value of 0 disables the MKI.
- **mkiID**: Enter the MKI identifier that should be included in each packet.
- **roc**: Enter value for the rollover counter. If it is a non-zero value, set the ssrc and sequence fields. This is state information used for testing.
- **ssrc**: Enter a value for the synchronization source. This is state information used for testing.
- **sequence**: Enter a sequence number. This is state information used for testing.
- **flags**: enter a number, which will be used by the ME, to complete the encryption/decryption testing.
- The **send** action requires the packet field:
- **packet**: Enter the hexadecimal representation of the UDP payload of the packets (it must include the "0x"). You would typically obtain this with an Ethereal capture.

The **stop** action takes no parameters.

Syntax

```
srtp start {encrypt | decrypt} {rtp | rtcp} suite key [mki-len] [mki-id] [roc]
```

```
[ssrc] [sequence] [flags]
srtp send packet
srtp stop
```

Example

The following example shows a successful SRTP decrypt:

```
NNOS-E>srtp start encrypt rtp AES_CM_128_HMAC_SHA1_80
0x4b8730843d222bdc3ab023fb49b43b987e5611ccf4f4320db6bfa16eafc
Success!
NNOS-E>srtp send
0x8008000000123bad0000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000
SRTP encrypted packet (182 bytes):
0x8008a60c00123bad000000000e183fbfb896e28b1848302c265db177a376deaacade96d56dab52bc
370f0ddff7bf9ec23d61d5bc9ecda0d781bafd62bcc61bab070bf954829b8d02bddad2d75970d906b6
9cf764a98275fcd12968af8a4c43a1db3d985c3c5ba4b772079922315eac75efbf586a75e4f0e7eed6
15d549a034c6890dae3b5dbdf1fd3cef518891a9c7cdb5f0ed8cf0f119e137406e337838dfe17ce3c2
cdeb340b4fb4a9764493b75e3056754e706d97
NNOS-E>srtp stop
AES_CM_128_HMAC_SHA1_80 Stats:   Pass              : 1   Encrypt           : 1
Decrypt            : 0   Drop: Crypto        : 0   Drop: Auth          : 0
Drop: Replay       : 0   Drop: Force         : 0   Drop: Internal      : 0
```

ssh

Provides tools to manage SSH public keys. When installing a public key on to the system, make sure to select the OpenSSH format. You can make sure

- **regenerate-host-key**: Regenerates the host keys used by SSH. Use this if you feel your SSH authentication system has been compromised. Note that the ME shuts down and restarts when you execute this action.
- **import-public-key**: Imports a public key into the ME key store. Use this option if the key already resides on the system as file.
- **add-public-key**: Adds a public key to the key store. Use this option when you can paste or enter the key content in to the command line.
- **remove-public-key**: Removes a public key from the key store. Use the **show ssh-public-keys** command to determine the number that corresponds with the key.
- **password**: Assigns the SSH account password. use this option if you have set the **account** property of the **ssh** object to **ssh**.

Syntax

```
ssh regenerate-host-keys
ssh import-public-key fileName
ssh add-public-key key
ssh remove-public-key integer
ssh password password
```

Example

```
NNOS-E>ssh regenerate-host-keys
Are you sure (y or n)?y
Acme Packet Net-Net OS-E is restarting...

NNOS-E>ssh add-public-key "1023 37
892894402808181734240272878525136869147515112964576516251364882012989536815157860 1
190400849935026306278579866563326902359915835310778566590770093344392441035802734 9
765766969317515908220381678683658869849422920314420373480212058574602902298344355 6
562727072942284125819207404381013102478430813101903266903760 69 rsa-1"
Are you sure (y or n)?y
Success!
```

stest

For Technical Support use only.

Syntax

```
stest options
```

stream

This action tests the connectivity of a newly configured **multimedia-streaming-server** object. You can test either:

- stream publish: Tests audio or video stream published to a specified server.
- stream subscribe: Tests a subscription to an audio or video stream on a specified server.

Syntax

```
stream publish [audio-file] [video-file] [server] [stream-name]
stream subscribe [audio-file] [video-file] [server] [stream-name]
```

Example

```
NNOS-E>stream publish file.wav server1 stream1
```

subscribe-delete

The **subscribe-delete** action removes either a single subscription binding or all of the bindings for a single AOR.

Valid arguments for this action are:

- <*AOR*>: The Address of Record in the form of a URI.
- [*id*]: The Unique ID that identifies the subscription binding (if not specified, all bindings are removed).

Syntax

```
subscribe-delete <AOR> [id]
```

Example

```
NNOS-E>subscribe-delete http://abc.com 654321
```

subscribe-flush

The **subscribe-flush** action removes all subscription bindings from the entire cluster.

Syntax

```
subscribe-flush
```

Example

```
NNOS-E>subscribe-flush
```

terminal-failover

Manages the database containing the active terminals known to the ME. The database contains phone numbers and their associated devices, and is persistent across reboots. Select either:

- **merge**: Merges the specified file into the active terminal list. Specify a file path; by default the system merges the file /cxc/terminal.xml.
- **replace**: Replaces the active terminal list with the specified file. By default, the system replaces the list with the file /cxc/terminal.xml.
- **save**: Saves the active terminal list cache to the database. By default, the system saves the cache to the file /cxc/terminal.xml. Optionally, you can specify a different file name.
- **delete**: Deletes a specific entry from the from the active terminal list. Use the show active-terminals command to list entries.
- **flush**: Removes all entries from the active terminal cache.

Syntax

```
terminal-failover merge [filePath]
terminal-failover replace [filePath]
terminal-failover save [filePath]
terminal-failover delete entry
terminal-failover flush
```

Example

```
NNOS-E>show active-terminals

address                    terminal                  type
-------                    --------                  ----
tel:+15086474840           SEP0013722AC21F           cisco
tel:+15086474850           SEP000E0C774E0B           cisco

NNOS-E>terminal-failover delete SEP000E0C774E0B
Success!
```

terminate-call

Immediately disconnects the call associated with the specified session ID. This action is intended to be used only as a last resort when the endpoints are no longer reachable. It does not make any attempt to cancel the call at the endpoints (the UAs still believe that the call is in progress). Instead, it does an internal cleanup to remove the session from the ME and free up any used resources. Use the **show active-call** command to view a list of currently active calls and their associated session IDs. Use the **disconnect-call** action to immediately terminate a call that has not been answered or that has been cancelled.

Syntax

```
terminate-call sessionID
```

Example

```
NNOS-E>terminate-call ca3e3764f07c42f5b7b6eda269d2d0c4
Success!
```

third-party-call-control

Manages the database containing the active terminals known to the ME. The database contains phone numbers and their associated devices, and is persistent across reboots. Select one of the following:

- **call**: Places a call
- **hold**: Places an existing call on hold
- **retrieve**: Retrieve an existing call from hold
- **transfer**: Transfer an existing call to another endpoint
- **disconnect**: Disconnect an existing call
- **join**: Given two calls, remove the originator from both calls and join the two terminators into a new call
- **loop**: Place a loopback call
- **annotate**: Attach an annotation on an existing call
- **get-annotation**: Display any annotation on an existing call
- **park**: Place a call to an endpoint, immediately placing it on hold
- **connect**: Place a call to an endpoint, and connect it to a parked call
- **terminate**: Disconnect one end of a call, leaving the other end on hold
- **memo-begin**: Start recording voice memo to a .wav file
- **memo-end**: Stop recording voice memo to a .wav file
- **play**: Play a .wav file on an existing call
- **drop-file**: Play a .wav file on an existing call, parking the originator
- **notify**: Send a NOTIFY event
- **message**: Connect to an endpoint, play a file, and terminate the call

Syntax

```
third-party-call-control call to from [requestId] [enabled | disabled] [enabled |
disabled] [any | UDP | TCP | TLS] [sessionConfigReference]
third-party-call-control hold handle [serverConfigReference]
third-party-call-control retrieve handle [serverConfigReference]
third-party-call-control transfer handle to [serverConfigReference]
third-party-call-control disconnect handle [serverConfigReference]
third-party-call-control join handle1 handle2 [serverConfigReference]
third-party-call-control loop handle [sessionConfigReference]
third-party-call-control annotate handle text [sessionConfigReference]
third-party-call-control get-annotation handle [sessionConfigReference]
third-party-call-control park endpoint [from] [requestId] [enabled | disabled]
[sessionConfigReference] [serverConfigReference]
third-party-call-control connect handle endpoint [enabled | disabled] [requestId]
[sessionConfigReference] [serverConfigReference]
third-party-call-control terminate handle [serverConfigReference]
third-party-call-control memo-begin handle filename [greeting] [enabled |
disabled]
third-party-call-control memo-end handle [serverConfigReference]
third-party-call-control play handle filename [enabled | disabled]
[serverConfigReference]
third-party-call-control drop-file handle filename [serverConfigReference]
third-party-call-control notify handle event [serverConfigReference]
third-party-call-control message filename endpoint [from] [requestId] [enabled |
disabled] [sessionConfigReference] [serverConfigReference]
```

tls test

Tests outgoing TLS connectivity to a remote device. Specify the destination IP address and port of the target device. Optionally, you can specify a reference to a configured certificate. If you do not specify the certificate, the ME uses the default outgoing TLS certificate entry.

Syntax

```
tls test address:port [certificateReference]
```

Example

```
NNOS-E>tls test 10.1.80.2:5061 "vsp\tls\certificate Client"
Attempting TLS test connection.
Destination: 10.1.80.2:5061
Certificate: Record 129
TLS connection established; waiting for validation by remote TLS server...
TLS connection still up; connection must have passed validation. Success.!
If you do not specify a certificate and the remote peer required one, you would
see this type of output:

NNOS-E>tls test 10.1.80.2:5061
Attempting TLS test connection.
Destination: 10.1.80.2:5061
Certificate: <Default Outgoing>
TLS connection lost: TLS handshake failure.
Connection attempt failed
If you specify the wrong certificate, you would see this type of output:

NNOS-E>tls test 10.1.80.2:5061 "vsp\tls\certificate Client"
Attempting TLS test connection.
```

```
Destination: 10.1.80.2:5061
Certificate: Record 129
TLS connection established; waiting for validation by remote TLS server...
TLS connection lost: Received TLS shutdown from peer.
Connection attempt failed
```

trap-reset

Sends an acknowledgement to the SNMP agent to discontinue retransmission of SNMP traps. Use this action if you have enabled this feature with the **trap-retransmit** property of the **snmp** object. When you execute this action, all traps being retransmitted will be stopped. However, any subsequent traps will be retransmitted until you either re-execute this action or disable the **trap-retransmit** feature.

Syntax

```
trap-reset
```

Example

```
NNOS-E>trap-reset
Success!
```

trickle-ice-update

This action behaves as if the trickle-ice-media-specification has been received via a SIP INFO request.

Enter the following arguments:

- <*call-leg-handle*>—The handle identifying the holding call-leg providing the trickle-ice update.
- <*media-type*>—The mime type of media specification (initial-answer-SDP).
- <t*rickle-ice-media-specification*>—The offer media specification forwarded to the peer call-leg (incremental- SDP).

Syntax

```
trickle-ice-update <call-leg-handle> <media-type> <trickle-ice-mediaspecification>
```

turn-allocation-purge

This action allows you to manually remove TURN Allocations. Per RFC5766, TURN clients that no longer want to use an Allocation are encouraged to delete the Allocation via a TURN Refresh request with a requested lifetime of 0. However, some TURN clients currently do not remove Allocations and these remain in the ME until they expire.

Note: Ensure you remove only unused Allocations. Removing valid and in-use Allocations disrupts a WebRTC call using the ME's TURN server.

Syntax

```
turn-allocation-purge [turn-client]
```

Note: By default, the **turn-allocation-purge** action purges all TURN Allocations, unless otherwise specified.

umount

Removes the specified device from the ME usable devices. In essence, this destroys a logical disconnection of the device, and you can no longer read or write to it. The effect of this is that any directories contained on the device become unavailable.

Syntax

```
unmount {data-1 | data-2 | usb | cdrom | system-1 | system-2}
```

Example

```
NNOS-E>umount data-2
Are you sure (y or n)?y
Success!
NNOS-E>
```

unregister

Disconnects an ME call using ME's REST APIs.

Syntax

Enter the following arguments:

- <*URI*>: The URI tied to this binding.
- <*binding-identifier*>: The identifier that uniquely identifies this binding.

Syntax

```
unregister <URI> <binding-identifier>
```

Example

```
NNOS-E>unregister sip:16175551234 987654
```

uri-alias

Manages the alias table, an indexed table into the location database. This table maintains all the alias information for any known address of record. Select one of the following:

- **lookup**: Returns all known aliases found in the alias table for the specified address of record.
- **reset**: Deletes all alias mappings from the alias table and then recreates them by synchronizing with the enterprise directory. The ME repopulates the table with any alias associated with a known AOR.

- **flush**: Deletes all alias mappings from the alias table and repopulates the alias table as it relearns them. Optionally you can specify whether to flush the cache immediately or as the entries time out. By default, the system flushes entries immediately.
- **seek**: Returns all known aliases found in the directory service for the specified address of record.
- **delete**: Deletes the specified entry from the location alias table.
- **change-state**: Changes the state of the location cache entry in the alias table. Enter the AOR and the new state.

You can also schedule this action as part of routine maintenance using the **task** object.

Syntax

```
uri-alias lookup aor
uri-alias reset
uri-alias flush [now | gracefully]
uri-alias seek aor
uri-alias delete aor
uri-alias change-state aor {unregistered | trying | in-service | redirect |
registered | out-of-service}
```

Example

```
NNOS-E>uri-alias lookup sip:3000010004@tom.com
URL sip:3000010004@tom.com has the following aliases:
sip:3000010004@tom.com tag virtual
```

uri-resolve

Returns the normalized URI and the address of record and binding for the specified URI. The URI can be a SIP URI (e.g., sip:joe@abc.com) or a TEL URI (e.g., tel:19788236666).

Syntax

```
uri-resolve uri
```

Example

```
NNOS-E>uri-resolve tel:+16667770001
URI tel:+16667770001 normalized to sip:6667770001@best.com
AOR sip:6667770001@best.com has a binding at 192.168.215.95 at 5060 via UDP

NNOS-E>uri-resolve sip:6667770001@best.com
URI sip:6667770001@best.com normalized to sip:6667770001@best.com
AOR sip:6667770001@best.com has a binding at 192.168.215.95 at 5060 via UDP
```

user-cache-lookup

Performs a lookup in the ME user database for the requested AOR. The result returns the AOR with the associated UID and tag. You can use the **show user-cache** command to display all users in the cache. The ME directory process stores all entries for all configured enterprise directories in a database file. That file, which is read in to the SIP processing side or operations, creates a cache of all directory (user) information.

Syntax

```
user-cache-lookup aor
```

Example

```
NNOS-E>user-cache-lookup sip:jdoe@cov.com
Found - sip:jdoe@cov.com:4:test
```

vsp-reset

Resets all sessions on the VSP, disconnecting any active sessions. This action is primarily a debugging tool and should only be used if Technical Supports instructs you to do so. However, it is also required to activate any changes to the properties in the **static-stack-settings** object.

Syntax

```
vsp-reset [vspName]
```

Example

```
NNOS-E>vsp-reset
Success!
NNOS-E>
```

web-services

Sets the status/availability of a previously configured external location, policy, or event service server. Configure the services using the **external-services group** and **service** objects. Use the **show web-services-callout-details** command with the **availability** field to verify the status. Set the server status to one of the following:

- **disabled**: Puts the server out of service until it is manually set to **available** using this action.
- **available**: Returns a server to service with the ME, whether it was manually set or detected as unavailable.
- **unavailable**: Marks a server as temporarily unavailable. If a **heartbeat-url** is configured for the server, the ME will attempt to bring it back into service.

Note that when the ME detects that a server is unavailable, it automatically changes that web service server status to unavailable.If the status is unavailable, you must set it to available (once it is) with this action before the ME can use that server again.

Syntax

```
web-services set-location server {disabled | available | unavailable}
web-services set-policy server {disabled | available | unavailable}
web-services set-event server {disabled | available | unavailable}
```

Example

```
NNOS-E>web-services set-policy "external-services policy-group pol1 policy-service
polSrvc1" available
No service for server reference
```

```
NNOS-E>config external-services policy-group pol1 policy-service polSrvc1
Creating 'policy-group pol1'
Creating 'policy-service polSrvc1'
config policy-service polSrvc1>exit
Do you want to commit your changes before you exit (y or n)?y

NNOS-E>web-services set-policy "external-services policy-group pol1 policy-service
polSrvc1" available
Success!>
```

xml

Provides tools to manage XML files on the ME. You must be running web services (enabled with the **web** object) on the system for this action to be available.

- **parse**: Checks the specified file for well formedness.
- **transform**: Translates the source file into the named destination file, using the specified style sheet. Enter a style sheet (.xsl file) to be used for the transformation. This can be a Oracle-supplied style sheet, for example, to convert configuration files for upgrade. Or, you can create your own to make modifications that you can then apply on each system. Optionally, you can use the **parameters** argument to alter the output transformed with the specified xsl style sheet.
- **validate**: Verifies that the content of the of the source file conforms to the default or specified schema. If you do not enter a schema, the system uses the configuration file as the default.

Syntax

```
xml parse file
xml transform stylesheet source destination [parameters]
xml validate fileName [schema]
```

Example

```
NNOS-E>xml parse /cxc/backup/cfg1106.xml
The XML file is not valid
NNOS-E>xml parse /cxc/install/install.xml
Success!
NNOS-E>xml transform serverUpdate.xsl /cxc/cxc.cfg /cxc/server.cfg
The XSL template appears invalid
NNOS-E>xml transform 2.1-to-3.0.xsl /cxc/backup/cxc.cfg /cxc/cxc.cfg red=blue
Success!

NNOS-E>xml validate /cxc/install.xml
Success!
```

5

Status Provider Show Commands A Through M

This chapter describes the status show commands from A through M alphabetically. for status show commands from N to Z, see "Status Provider Show Commands N Through Z."

Global Show Command Characteristics

All of the status show commands contained in this chapter share the same MIB and require the same user access level. In addition, they are available through a single place in the GUI.

Associated MIB

All status display commands reference the Media Engine (ME), Inc. enterprise MIB:

`CXC.MIB`

ME Management System Path Information

All status display commands are available in the ME Management System.

You can access the commands from the **Status** tab at the top of the window or from the **Status** link on the Home page. Expand the list on the left and click on a status report listed below it.

Filtering Command Output

The CLI allows you to filter output of show commands so that your display only includes the specific properties you requested. With no properties, **show** *object-name* displays all instances of the specified object. For example, if you execute the **actions** command, the CLI displays a list of all actions that have occurred in the current CLI session:

```
NNOS-E> show actions

action               process timeout    requests   errors     timeouts
------               ------- -------    --------   ------     --
archive                acct    10000      0          0          0
arp-delete             manager 10000      0          0          0
clock                  manager 10000      0          0          0
config merge           manager 30000      0          0          0
config replace         manager 30000      0          0          0
config save            manager 30000      2          0          0
database               manager 300000     0          0          0
database-maintenance   manager 10000      0          0          0
```

```
diameter                auth    10000      0          0          0
```

The output includes indices to the action (e.g., arp-delete, clock, config merge, config save, etc.) and properties of the indices (in this case, process, requests, errors, timeouts).

However, if you are only interested in seeing a specific index or property, you can filter on those fields. You can specify an index name to display only the instances with those values. Or, you can specify one or more property values to display only the instances with those property values.

Note: Index and property names are case insensitive.

To display a list of the properties you can filter on, enter the command with a question mark:

```
NNOS-E> show actions ?
 action provider statistics
 action
 process
 timeout
 requests
 errors
 timeouts
 -c       display the total number of instances
 -n       display a specified number of instances
 -v       verbose display
```

Filtering On an Index

To filter on the index, enter the object name with the command (in quotation marks if the name includes white space). For example, you can display only the number of saves to the configuration file that have occurred:

```
NNOS-E> show actions "config save"
action             process timeout    requests   errors     timeouts
------             ------- -------    --------   ------     -----
config save        manager 30000      2          0          0
```

To filter on a property, enter the property name followed by an equal sign (or let the system enter the correct format with a TAB complete). To see only directory processes:

```
NNOS-E> show actions process=dir
action             process timeout    requests   errors     timeouts
------             ------- -------    --------   ------     -----
directory-reset        dir     120000     0          0          0
```

Note that you can enter multiple properties to further refine your output. You cannot, however, enter multiple instances of the same property. In that case, the last property entered is acted on:

```
NNOS-E> show actions process=SIP process=auth
action             process timeout    requests   errors     timeouts
------             ------- -------    --------   ------     -----
diameter           auth    10000      0          0          0
radius             auth    10000      0          0          0
```

Displaying Total, Count, and Verbose Reports

You can display summary reports on a status using one of the options defined in the following table. The following table displays examples of show command report types.

Table 5–1 Show Command Report Types

Option	Description
-c	Displays a count of the total number of entries in a status report. Enter in the form: show *object-name* -c
-n	Displays the specified number of entries from the status report, counting from the first entry. Enter in the form: show *object-name* -n=*x* **-n=x** displays the first *x* instances.
-v	Displays a more detailed report of the object. This option does not change all output, only for those reports where summary and detailed reports are both available. Enter in the form: show *object-name* -v

Examples of -c, -n, and -v use

Without using display options, the output of **show dial-plan** looks like this:

```
NNOS-E> show dial-plan
plan-name     type     destination-url    from     peer-name   fwd
---------     ----     ---------------    ----     ---------   --
abc.com       tag      aster              .*       abc         0
company.net   tag      bw                 .*       company     0
xyz.com       tag      aaa                .*       xyz         0
123.com       domain   sip:.*@123\.com    .*       123         0
server.com    domain   sip:.*@server\.com .*       server      0
Using the count option, the output of show dial-plan -c looks like this:
NNOS-E> show dial-plan -c
dial-plan returned 5 instances
```

Specifying that the system display the first two entries using the number of entries option, the output of **show dial-plan -n=2** looks like this:

```
NNOS-E> show dial-plan -n=2
plan-name     type     destination-url    from     peer-name   fwd
---------     ----     ---------------    ----     ---------   --
abc.com       tag      aster              .*       abc         0
company.net   tag      bw                 .*       company     0
```

Using the verbose option, the output of **show dial-plan -v** looks like this:

```
NNOS-E> show dial-plan -v

                          plan-name: abc.com
                               type: tag
                                url: aster
```

```
                        destination-url: aster
                                 source: plan
                                  level: 0
                                   from: .*
                              peer-name: abc
                          peer-identity: sip:sametime@abc.com
                              peer-mode: provider
                                 action: redirect
                                    fwd: 0
                                   hits: 0
            incoming-host-normalizations:
                        in-request-user:
                    in-request-user-template:
                                in-to-user:
                           in-to-user-template:
                                  in-from-user:
                       in-from-user-template:
            outgoing-host-normalizations:
--More--
```

Note: All show commands include the -c, -n, and -v options (although in some commands the options do not change the output). Because they are universal, these options are not included in the command description syntax statement in this chapter.

show accounting-targets

Displays information from all accounting targets configured on the ME. The settings are configured using the file-system object. See the file-client config object for information on the proper configuration when the external file-system is configured for SCP or SFTP.

Sample Output

```
NNOS-E> show accounting-targets
type: file-system
name: path 1
received: 0 CDRs
processed: 0 CDRs
failures: 0
missing-records: 0
average-processing-time: 0 milliseconds/CDR
```

Properties

The following table describes the properties for the show accounting-targets command.

Table 5–2 Show Accounting-Targets Properties

Field	Description
type	The type of file-system target this command is displaying.
name	The name of the accounting target whose status is displayed.
received	The number of raw CDRs received.

Table 5–2 (Cont.) Show Accounting-Targets Properties

Field	Description
processed	The number of CDRs processed.
failures	The number of failures.
missing-records	The number of raw CDRs the target found missing and could not write to the output. These messages may be missing or corrupt. A purge can cause this. Check logs for details.
average-processing-time	The average processing time per CDR of this accounting target.

show accounting-targets-archive

Displays detailed information regarding archive targets.

Sample Output

```
NNOS-E>show accounting-targets-archive

 name: archive-day1
                 url: http://172.40.100.1/cgi-bin/archive_http_upload_example_
null.pl/dev/null
  tasks-in-progress: 0
           received: 641 CDRs
        in-progress: 0
               sent: 0 archives
              saved: 0
      archive-fails: 0
      create-errors: 0
    transmit-errors: 0
              state: clear
      current-saved: 0
last-commit-success: 16:12:35 Tue 2011-03-22
 average-time-taken: 228 msec/archive
```

Properties

The following table shows the properties for the show accounting-targets-archive command.

Table 5–3 Show Accounting-Targets-Archive Properties

Field	Description
name	The name of the displayed archiving target.
url	The URL of this external archiving target.
tasks-in-progress	The number of archiving tasks currently in progress.
received	The number of raw CDRs received.
processed	The number of raw CDRs processed.
sent	The number of archive files sent to the archiving target.
archive-fails	The number of failures that occurred during the archiving process

Table 5–3 (Cont.) Show Accounting-Targets-Archive Properties

Field	Description
create-errors	The number of errors that occurred during the creation of the archive fail.
transmit-errors	The number of errors that occurred during the transmission of the archive file.
saved	The number of saved CDRs.
state	The state of the archiving target.
last-commit-success	The last time the ME successfully sent a file to the archiving target.
average-time-taken	The average processing time per CDR of this archiving target.

show accounting-targets-archive-tasks

Displays information about currently running archiving tasks on the ME.

Sample Output

```
gregg-b5-74>show accounting-targets-archive-tasks

name           record   errors     in-progress
------         -------  ------     -----------
nnose-backup   1170995    2          (send)
nnose-backup   1171000    2          (send)
nnose-backup   1171001    2          (send)
```

Properties

The following table shows the properties for the show accounting-targets-archive-tasks command.

Table 5–4 Show Accounting-Targets-Archive-Tasks Properties

Field	Description
name	The name of the archiving target.
record	The CDR ID that the archive target is processing.
errors	The number of errors encountered while this CDR is being processed. These are errors that do not stop processing. For example, if a send has failed but there are retries left.
in-progress	A task is the processing for a CDR. A task has several operations it performs. This property shows the operations within the task currently in progress.

show accounting-targets-file-system

Displays information for each accounting target configured on the ME. This shows information for both file-system and external-file-system targets. See the file-client config object for information on the proper configuration when the external file-system is configured for SCP or SFTP.

There are four states that the external target cycles through as it processes raw CDRs, writes to the output file, and sends it to the remote system.

- Clear: The target is ready to write.
- Writing: The target is currently writing to the temporary file.
- Sending: The target is sending a file. At this time, the file can also be writing to a temporary file that will become the next file to send once the current file is successfully sent.
- Blocked: The target has one file in the middle of sending and another one ready to send. The target will not process anymore requests from the accounting server, but will send retries to the server giving retry interval based on its best estimate of when the retry can work.

Sample Output

```
NNOS-E> show accounting-targets-file-system
type: file-system
name: path 1
url:
master: enabled
state: clear
received: 0 CDRs
saved: 0 CDRs
files-sent: 0
current-file: /cxc_common/acct/test.2009.09.17.03.53.50.csv
cdrs-in-current-file: 0 CDRs
save-fails: 0
transmit-fails: 0
missing-records: 0
```

Properties

The following table shows the properties for the show accounting-targets file system command.

Table 5–5 Show Accounting-Targets-File-System Properties

Field	Description
type	The type of file-system this command is displaying.
name	The name of the accounting target whose status is displayed.
url	The URL of this external accounting target.
master	Displays whether this box is a master for accounting master-service or not.
state	The state of the file system.
received	The number of raw CDRs received.
saved	The number of saved CDRs.
files-sent	The number of files sent to the target. This is applicable only to external file systems.
current-file	The file to which raw CDRs are currently being written. At the next rollover, this file is closed and a new one is opened.

Table 5–5 (Cont.) Show Accounting-Targets-File-System Properties

Field	Description
cdrs-in-current-file	The number of CDRs in the current file.
save-fails	The number of failures that occurred during saving.
transmit-fails	The number of failures that occurred during transmission. This is applicable only to external file systems.
missing-records	The number of raw CDRs the target found missing and could not write to the output. These messages may be missing or corrupt. A purge can cause this. Check logs for details.

show active-msrp-sessions

Displays information regarding active MSRP session statistics.

Sample Output

```
SIP>show active-msrp-sessions
Active MSRP Sessions:
------------------------------------------------------------------------
session-handle: 0xC7C634F3
inleg-type: Msrp
inleg-state: CONNECTED
outleg-type: Msrp
outleg-state: CONNECTED
caller-session-id: mhnb1ad02f
caller-path: msrp://wscAddress.invalid:2855/mhnb1ad02f;ws
called-session-id: 2511644601
called-path: msrp://10.138.238.49:53847/2511644601;tcp
create-time: 12:09:59.163681 Thu 2014-10-30
duration: 24 seconds
------------------------------------------------------------------------
Total Active MSRP Sessions: 1
```

Properties

The following table shows the properties for the **show active-msrp-sessions** command.

Table 5–6 Show Active-MSRP-Sessions Properties

Field	Description
session-handle	The handle for this session.
inleg-type	The type of endpoint of the in-leg session.
inleg-state	The state of the in-leg session endpoint.
outleg-type	The type of endpoint of the out-leg session.
outleg-state	The state of the out-leg session endpoint.
caller-session-id	The session ID of the calling endpoint.
caller-path	The path of the calling endpoint.
called-session-id	The session ID of the called endpoint.

Table 5–6 (Cont.) Show Active-MSRP-Sessions Properties

Field	Description
called-path	The path of the called endpoint.
create-time	The time this session was created.
duration	The length, in seconds, of this session.

show active-session

Displays message flow for a session. The message-log field indicates all messages for the session. In the first example, the session was a simple registration. In the second example, the session consisted of a call with multiple messages.

Sample Output

```
NNOS-E> show active-session
                    index: 1
                session-id: 0x4c2b67ad57e1cee
            association-id: 0x98000000006
             creation-time: 11:10:37.814301 Thu 2007-10-04
              session-type: proxy
            in-leg-call-id: bacd6b4a56394d5ea0f1fe0f6ae39e58
           out-leg-call-id: bacd6b4a56394d5ea0f1fe0f6ae39e58
         association-index: 3
               message-log: |-->INVITE|INVITE-->|INVITE 100<--|INVITE 200<--|<--
INVITE 200|-->ACK|ACK-->|-->MESSAGE|MESSAGE-->|MESSAGE 200<--|<--MESSAGE 200|-->
BYE|BYE-->|
ingress-classification-tag: qik-finemode
 egress-classification-tag: qik-finemode
```

Properties

The following table shows the properties for the show active-session command.

Table 5–7 Show-Active-Session Properties

Field	Description
index	A system-assigned internal identifier that indicates the current position of the session within the list of active sessions.
session-id	The internal identifier for the session. A session is a particular “conversation” between two endpoints.
association-id	The internal identifier for the association. An association is a pair of endpoints that might have had, be having, or in the future have, a “conversation.”
creation-time	A time-stamp indicating when the session was created.

Table 5–7 (Cont.) Show-Active-Session Properties

Field	Description
session-type	The type of session being reported on, either: **proxy**: Stateful proxy **stateless proxy**: Stateless proxy **b2bua**: B2B user agent **outbound**: Outbound call **regServer**: Registration server **regClient**: Registration client
in-leg-call-id	The call ID used for the call as it came into the system.
out-leg-call-id	The call ID the system used when forwarding the call out.
association-index	The index of the association that matches the From/To pair of this particular session.
message-log	A very short description of each message type that came through on the session.
ingress-classification-tag	The tag used to associate incoming traffic with the configured tag. The configured ingress-tag must match a configured ip routing-tag. You can also configure a classification-tag through the ip interface object. If this property is configured in both places, the session-config setting takes precedence.
egress-classification-tag	The tag used to select the outgoing interface. That tag must then be associated with an ip **routing-tag**, which controls the available egress interfaces and routes. You can also configure a **classification-tag** through the **ip interface** object. If this property is configured in both places, the **session-config** setting takes precedence.

show authentication-details

Displays authentication error details on the ME.

Sample Output

```
NNOS-E>show authentication-details
------------------------------------------------------------------
Provider      Requests   Replies  Accepts  Rejects  Timeouts  QClipped   Others
------------------------------------------------------------------
Local                0         0        0        0         0         0        0
RADIUS               3         3        0        0         3         0        0
Diameter             0         0        0        0         0         0        0
Directory            0         0        0        0         0         0        0
Accept               0         0        0        0         0         0        0
Reject               0         0        0        0         0         0        0
------------------------------------------------------------------
```

Properties

The following table shows the properties for the show authentication-details command.

Table 5–8 Show Authentication-Details Properties

Field	Description
Provider	The protocol to be used for errors.
Requests	The number of requests submitted to each provider.
Replies	The number of replies to errors.
Accepts	The number of positive replies received from the remote server.
Rejects	The number of rejects received from the remote server.
Timeouts	The number of timeouts that have caused errors.
QClipped	The number of errors that have failed locally, without ever being sent to the remote server because the queue of requests outstanding to the server(s) has grown too long.
Others	Sum of other errors. These can be seen individually by adding **-v** to the end of the action.

show authorized-user-attributes

The **show authorized-user-attributes** action displays information about configured ME users and their attributes and values.

```
NNOS-E>show authorized-user-attributes

username   attribute                    value
--------   ---------                    -----
sjones     mail                         sjones@acmepacket.com
sjones     msrtcsip-primaryuseraddress  sip:sjones@acmepacket.com
sjones     cn                           Sam Jones
sjones     samaccountname               sjones
sjones     msrtcsip-line                tel:+17815557256
sjones     st                           MA
sjones     telephonenumber              +1 (781) 555-4839
```

Properties

The following table shows the properties for the show authorized-user-attributes command.

Table 5–9 Show Authorized-User-Attributes Properties

Field	Description
username	The configured ME user.
attribute	The attribute name.
value	The value of the attribute for that user.

show authorized-user-groups

The **show authorized-user-groups** action displays the configured users and the groups to which they belong from the user cache.

```
NNOS-E>show authorized-user-groups
```

```
username     group
--------     -----
sjones       eng
sjones       software
sjones       dev
sjones       ct
sjones       engineering
sjones       deliveries
sjones       funcspec
```

Properties

The following table shows the properties for the show authorized-user-groups command.

Table 5–10 Show Authorized-User-Groups Properties

Field	Description
username	The configured ME user.
group	The group to which the user belongs.

show authorized-user-privileges

The **show authorized-user-privileges** action displays information about users' authorization privileges from the user cache.

> **Note:** If a user has never logged into the ME, their name does not appear in the cache and, therefore, is not displayed in the **show authorized-user-privileges** command output.

```
NNOS-E>show authorized-user-privileges

username   resource-type privilege identity-type resource-identity
--------   ------------- --------- ------------- -----------------
admin      event-channel C+R+U+D   equals        /system/*
```

Properties

The following table shows the properties for the show authorized-user-privileges command.

Table 5–11 Show Authorized-User-Privileges Properties

Field	Description
username	The name of the configured ME user.
resource-type	The resource-type of the grant configured for this user.
privilege	The CRUD privileges of the of the resource-type configured for this user.
identity-type	The method in which the ME matches the users' resource-identity.

Table 5–11 (Cont.) Show Authorized-User-Privileges Properties

Field	Description
resource-identity	The value or regular expression the ME uses to check users' authorization privileges.

show authorized-user-summary

The **show authorized-user-summary** action displays an abbreviated version of users' authorization privileges from the user cache.

```
NNOS-E>show authorized-user-summary

username    resource-types
--------    --------------
admin       event-channel
test_user   event-channel
```

Properties

The following table shows the properties for the show authorized-user-summary command.

Table 5–12 Show Authorized-User-Summary Properties

Field	Description
username	The name of the configured ME user.
resource-type	The resource-type of the grant configured for this user.

show automatic-settings

Displays values that the ME has generated for each property that supports the automatic settings feature. For these properties, the ME determines an appropriate value based on various aspects of the system hardware, such as the platform, CPU speed, and available memory.

Note: Do not change the values of properties configured with automatic-settings unless instructed to do so by Technical Support.

Sample Output

```
NNOS-E> show automatic-settings
name                          value
----                          -----
cac-max-calls                 7500
cac-max-calls-in-setup        1500
cac-max-number-of-tls         3000
cac-max-tls-in-setup          423
cac-min-calls-in-setup        10
max-number-of-sessions        7500
stack-socket-event-threads-max 4
stack-socket-threads-max      4
stack-worker-threads          4
```

Properties

The following table shows the properties for the show automatic settings command.

Table 5–13 Show Automatic-Settings Properties

Field	Description
name	The name of the property whose default value is automatically determined by the system. For example, **cac-max-calls** sets the maximum number of concurrent calls allowed on the VSP.
value	The default value that the system assigns to the property.

show boxes

Saves stored sessions for the specified VSP. The archiving action archives all data that has not been successfully archived previously. You can archive a specific session or sessions that occur between specified times using the **archive specific** action.

Use this action to initiate the backup immediately; use the **task** object to schedule automated backups. You must enable archiving with the **archiving** object for this action to succeed. Use the **show archive-result** command to view the outcome of archiving operations.

Sample Output

```
NNOS-E> show boxes
-------------------------------------------------------------------
Box Address      ? Prot  State       Up Time   Connects  Errors  Last Error
-------------------------------------------------------------------
Local            O None  Connected   01:09:20         1       0  Unknown
192.168.0.2      A TCP   Connected   01:09:10         1       0  None
-------------------------------------------------------------------
```

Example

The following table shows the properties for the show boxes command.

Table 5–14 Show Boxes Properties

Field	Description
Box Address	The IP address for all boxes in the cluster (or Local to indicate the local box).
?	The role of the box in its connection to the local box, either originator (O) or accepter (A).
Prot	The messaging protocol in use between boxes, either TCP or TLS.
State	The state of the connection between boxes, either: Idle Connecting Helloing Connected Waiting (the box is not connected and the ME is waiting a short period before attempting to reconnect).

Table 5–14 (Cont.) Show Boxes Properties

Field	Description
Up Time	If **State** is **Connected**, the time of connection between boxes.
Connects	The number of times the system has successfully connected to the local box since 1) this ME booted, and 2) the device was added to the cluster.
Errors	The number of attempts to reconnect that were unsuccessful.
Last Error	The type of the last error. Error types are: **None**: No error **No Route**: No route can be found to remote box **No Socket**: Failed to create socket to connect to this box **No Connect**: Connection failed (e.g., due to box is down, or network or configuration error) **Connect Timeout**: Connection timed out **Disconnect**: Boxes were disconnected **Loopback**: Configuration error, i.e., the "remote" box is the local box **Duplicate MAC**: Duplicate MAC address detected (probably loopback) **Hello Timeout**: Connected, but failed to communicate **Version Mismatch**: Communicated, but discovered incompatible versions **Keepalive Failed**: Connected, but box didn't respond to keepalive messages **Other**: Other type of error

show call-admission-control

Displays settings and statistics for call admission control on this VSP (INVITE requests). The name field identifies the VSP being reported on. The settings are configured using the **admission-control** object.

Sample Output

```
NNOS-E> show call-admission-control
                                    name: default
                  call-admission-control: enabled
                               max-calls: 7500
                      max-calls-in-setup: 1500
                      min-calls-in-setup: 10
        calls-in-setup-dynamic-threshold: 1500
                         cpu-monitor-span: 20 seconds
                    cpu-monitor-interval: 10 seconds
                          average-sip-cpu: 0 %
                 calls-high-cpu-threshold: 90 %
                  calls-low-cpu-threshold: 50 %
                            current-calls: 0
                  current-calls-in-setup: 0
                               most-calls: 0
                      most-calls-in-setup: 0
                        max-calls-dropped: 0
          max-calls-dropped-last-logging:
max-calls-in-setup-dropped-this-interval: 0
```

```
max-calls-in-setup-dropped-last-interval: 0
             max-calls-in-setup-dropped: 0
```

Properties

The following table shows the properties for the show call-admission-control command.

Table 5–15 Show Call-Admission-Control Properties

Field	Description
name	The name of the VSP whose status is displayed.
call-admission-control	The state of CAC for this VSP: whether it is enabled or disabled.
max-calls	The maximum number of calls allowed on this VSP. This is the overall simultaneous call limit.
max-calls-in-setup	The maximum number of simultaneous inbound and outbound call legs in the setup stage allowed by the CAC.
min-calls-in-setup	The minimum number of simultaneous inbound and outbound call legs in the setup stage allowed by the CAC.
calls-in-setup-dynamic-threshold	The limit for the number of in-progress calls allowed before the system suppresses all calls.
cpu-monitor-span	The number of seconds over which the system calculates the total system CPU average. At the conclusion of the span, the average value is compared to the call and registration CPU thresholds to determine whether to modify the dynamic threshold. The longer the span, the fewer the changes to the thresholds. A shorter span will result in reactions to brief CPU activity spikes.
cpu-monitor-interval	The frequency, in seconds, with which the system calculates the total system CPU average for the last span.
average-sip-cpu	The current average CPU usage.
calls-high-cpu-threshold	The percentage value of CPU usage that determines whether the system modifies the call dynamic threshold.
calls-low-cpu-threshold	The lowest percentage value of CPU usage that the system can drop to when decreasing the dynamic threshold. The system starts decreasing the dynamic threshold when the average CPU usage exceeds the value for calls-cpu-threshold.
current-calls	The number of calls currently being processed by the system.
current-calls-in-setup	The number of calls currently in the setup stage on the system.
most-calls	The highest number of calls processed at any one time (since last system boot).
most-calls-in-setup	The highest number of calls in setup stage at any one time (since last system boot).
max-calls-dropped	The total number of active calls that were dropped since the last system boot.

Table 5–15 (Cont.) Show Call-Admission-Control Properties

Field	Description
max-calls-in-setup-dropped-this-interval	The number of calls that were in setup stage but dropped during the current interval. The interval is defined with the **cpu-monitor-interval** property.
max-calls-in-setup-dropped-last-interval	The number of calls that were in setup stage but dropped during the previous interval. The interval is defined with the **cpu-monitor-interval** property.
max-calls-in-setup-dropped	The maximum number of calls in the setup stage that were dropped since the last system boot.

show call-routing

Displays the call routing table, which defines how the ME forwards an outgoing call. The output displays a summary of each *active* dial-plan entry, its match criteria and peer (and other configuration elements), and the number of times the ME has applied it to forward a call. Use the **show dial-plan** command to see all *configured* dial plans.

Sample Output

```
NNOS-E> show call-routing
-----------------------------------------------------------------
Forwards  Pri  Type    Data
-----------------------------------------------------------------
       5  100  domain  Plan name:  companyXYZ.com
                       Match:      companyXYZ.com
                       Peer name: Company ST
```

Properties

The following table shows the properties for the show call-routing command.

Table 5–16 Show Call-Routing Properties

Field	Description
forwards	The number of times this plan has matched an INVITE request, and the system forwarded the request. This is a counter internal to the system.
pri	The priority (order of preference) setting for the **dial-plan** entry. This property overrides the default behavior (most specific match) and sets a preference based on the **request-uri-match** (route) or **source-match** (source-route) property.
type	The portion of the request to match on. If the INVITE matches the portion identified by the type, the system forwards the request to that server. The type can be contributed from the dial-plan configuration. Types of *tag* or *domain* can be contributed from the **auto-tag-match** and **auto-domain-match** options of the **server routing-setting** property.
Data	The data field is made up of Plan name, Match, an Peer name, all of which are described below.

Table 5–16 (Cont.) Show Call-Routing Properties

Field	Description
Plan name	The name of the active dial plan, created with the **dial-plan** or **dial-prefix** configuration.
Match	A derivative of the regular expression or tag (for faster matching) that identifies the "to" or "from" mapping. This string is configured in the **dial-plan** configuration. If contributed through a **route** object entry, the string to match in the SIP header fields or transport information in order for the system to apply the plan to calls containing the prefix ("to" mapping). If contributed through a **source-route** object entry, the match criteria for the source of the SIP message ("from" mapping). The **match** field is derived from either the **to-uri-match** (route) or **source-match** (source-route) property. If type is condition-list, the match is derived from the priority plus plan name.
Peer name	A statically entered peer. This is a configured **server** of type **sip-registrar**.

show chassis-info

Displays hardware and firmware information (e.g. serial, part, and version numbers) for the chassis. The three fields described are useful information for Technical Support. All other fields are Intel-specific, and are not used at this time.

Sample Output

```
NNOS-E> show chassis-info
             BIOS-version: 6.7.1.1
          chassis-version: 1
             chassis-type: 17
      chassis-part-number: NN 2610
    chassis-serial-number: 201-01060
           chassis-custom: SR1500
            board-version: 1
          board-lang-code: 25
           board-mfg-time: 16:30:00 Sun 2026-09-27
       board-manufacturer: Intel
       board-product-name: SE7520JR22
      board-serial-number: BZJR44325226
        board-part-number: C53660-403
        board-fru-file-id: FRU Ver 0.01
             board-custom:
          product-version: 1
        product-lang-code: 25
     product-manufacturer: Intel
             product-name:
product-part-model-number:
         product-revision:
    product-serial-number:
        product-asset-tag: 1234
      product-fru-file-id:
           product-custom:
```

Properties

The following table shows the properties for the show chassis-info command.

Table 5–17 Show Chassis-Info Properties

Field	Description
BIOS-version	The current software update package revision; the revision of firmware that the ME hardware is running. Also sometimes known as the System Update Package (SUP), it controls fans, power, IMM (Management Module), BIOS, and more.
chassis-part-number	The model number of the current device.
chassis-serial-number	The associated part number of the current ME device.

show clock

Displays the current date and time, and the amount of uptime in days, hours, and minutes since the ME was started. You can set the time with the clock action.

Sample Output

```
NNOS-E> show clock
  time: 12:05:32 Thu 2006-12-21
uptime: 0 days 01:09:20
```

Properties

The following table shows the properties for the show clock command.

Table 5–18 Show Clock Properties

Field	Description
time	The current time as configured on the box. You can set (or reset) the system time with the **clock** action
uptime	The amount of time since the last system boot.

show cluster

Displays each box, by IP address, that is part of the cluster. Additionally, the output indicates whether the box is receiving, or due to receive, configuration from the cluster master. The output displays the configuration of the **cluster** and **box** objects.

Syntax

```
NNOS-E> show cluster
ip-address      box-id
----------      ------
0.0.0.0         1
192.168.0.2     2
```

Properties

The following table shows the properties for the show cluster command.

Table 5–19 Show Cluster Properties

Field	Description
ip-address	The IP address of each box connected to this local box. The local box displays as 0.0.0.0.
box-id	The ID assigned to the box. This is the identifier assigned through the **box** object.

show codec-info

Displays CODECs recognized by the ME, and whether the ME supports encoding/decoding of that type. Additionally, the output displays the expected RTP payload sizes for the default packetization time/rate.

Sample Output

```
NNOS-E>show codec-info

Name         PacketTime MinPayload MaxPayload Bandwidth  R-factor     Playable
----         ---------- ---------- ---------- ---------  --------     --------
L16          20         0          0          0          93.2         true
NSE          20         4          4          0          0.0          true
gsm          20         31         31         35         100.0        true
t38          20         0          0          0          0.0          false
g722         20         80         80         55         100.0        false
g723         30         4          24         21         74.2         true
g728         20         40         40         39         100.0        true
g729         20         4          20         31         82.2         true
h263         20         0          0          90         0.0          true
h264         20         0          0          90         0.0          false
iLBC         30         50         50         28         100.0        true
```

Properties

The following table shows the properties for the show codec-info command.

Table 5–20 Show Codec-Info Properties

Field	Description
Name	The name of the codec that the ME recognizes.
PacketTime	The default length of time in milliseconds represented by the media in a packet. For example, a codec with a sample rate of 8000 samples/second (8 samples/ms) sends 160 samples in a media packet with a 20 ms packet time.
MinPayload	The minimum acceptable RTP payload size that is verified when **media-verify-config** is enabled.
MaxPayload	The maximum acceptable RTP payload size that is verified when **media-verify-config** is enabled.
Bandwidth	Note that when this value is 0, it is not typically a voice-carrying CODEC type, but rather it is more likely it is used to capture keypad input.
R-factor	The R-factor used for MOS calculations.
Playable	Whether a codec can be mixed by the ME.

show collect-status-classes

The **show collect-status-classes** action displays which status classes are being collected. When entered with the **default** parameter, the ME default status classes are listed.

```
NNOS-E>show collect-status-classes default
```

You can also use the **show collect-status-classes** status provider to display status classes defined in custom configurations. The following shows accounting as an example.

Sample Output

```
NNOS-E>show collect-status-classes accounting

Status classes to be collected for 'Accounting':
-----------------------------------------------------------------
Source   Status class               Description
-----------------------------------------------------------------
config   accounting-recent          calls recently accounted
config   accounting-database        request information for accounting database
connections
config   accounting-files           accounting file information
config   accounting-store           accounting disk storage information
config   accounting-cdr-summary     accounting CDR summary
config   accounting-targets-file-system accounting file-system and
external-file-system targets
config   accounting-targets         accounting targets
```

Properties

The following table shows the properties for the show collect-status-classes command.

Table 5–21 Show Collect-Status-Classes Properties

Field	Description
Source	The source of the status classes being collected.
Status class	The status classes the ME is configured to be collecting.
Description	A description of the status classes being collected.

show cometd-channel-detail

provides more detailed channel information, specifically, on the subscribers to each of the channels.

Note that if a channel appears in the **show cometd-channels-summary** status provider, but not in the details, it means that the channel exists without any active cometd client subscriptions.

Sample Output

```
NNOS-E>show cometd-channel-details

name             remote-address  remote-port id               user-agent
----             --------------  ----------- --              ----------
/**       10.1.21.57   49804    372tj5ikmvga8ant2b6m2wcjs Mozilla/5
.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11 (KHTML, like Gecko) Chrome/17.0.96
```

```
3.79 Safari/535.11
/call/to/019785551212 10.1.21.57   49728   21sxpszu2lkikc1pnadt0mdfzvg Mo
zilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11 (KHTML, like Gecko) Chrome/
17.0.963.79 Safari/535.11
/cometd/meta   10.1.21.57   49804   372tj5ikmvga8ant2b6m2wcjs Mozilla/5
.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11 (KHTML, like Gecko) Chrome/17.0.96
3.79 Safari/535.11
```

Properties

The following table shows the properties for the show cometd-channel-detail command.

Table 5–22 Show Cometd-Channel-Detail Properties

Field	Description
name	The name of the channel.
remote-address	The remote address for this subscriber.
remote-port	The remote port for this subscriber.
id	The identifier assigned internally by the ME for this publisher.
user-agent	The user agent the subscriber used to establish the session.

show cometd-channel-summary

Provides a summary of channel information for the cometd server.

Sample Output

```
NNOS-E>show cometd-channel-summary

name                        subscriber-count
----                        ----------------
/**                         1
/call                       0
/call/to                    0
/call/to/019785551212       1
/cometd                     0
/cometd/meta                1
/meta                       0
/meta/connect               0
/meta/disconnect            0
/meta/handshake             0
/meta/subscribe             0
/meta/unsubscribe           0
```

Properties

The following table shows the properties for the show cometd-channel-summary command.

Table 5–23 Show Cometd-Channel-Summary Properties

Field	Description
name	The name of the channel.

Table 5–23 (Cont.) Show Cometd-Channel-Summary Properties

Field	Description
subscriber-count	The number of subscribers on this channel.

show cometd-status

The **show cometd-status** command displays information about the **eventpush-service** configuration and activity on the ME.

```
NNOS-E>show cometd-status

                          ip: 100.40.10.7
                        port: 8081
                    sessions: 0
                max-sessions: 10000
         max-sessions-reached: 0
         session-idle-timeout: 60 seconds
                 pool-threads: 1
                  max-threads: 10
      idle-connection-timeout: 20 seconds
                  connections: 0
 cross-origin-requests-denied: 0
cross-origin-requests-allowed: 0
```

Table 5–24 Show Cometd-Status Properties

Field	Description
ip	The eventpush-service IP address.
port	The eventpush-service port number.
sessions	The number of active sessions.
max-sessions	The configured maximum number of sessions allowed.
max-sessions-reached	The number of times a session was not created because the **max-sessions** value was reached.
session-idle-timeout	The configured session idle timeout.
pool-threads	The current number of request processing threads in the thread pool.
max-threads	The configured maximum number of request processing threads.
idle-connection-timeout	The configured connection idle timeout.
connections	The current number of connections.
cross-origin-requests-denied	The number of CORS requests allowed.
cross-origin-requests-allowed	The number of CORS requests denied.

show cometd-subscriber-details

Provides more detailed information, specifically on the channels subscribed to by each subscriber.

Note that if a subscriber appears in the **show cometd-subscriber-summary** status provider, but not the details, it means that the subscriber exists without any active cometd channel subscriptions.

Sample Output

```
NNOS-E>show cometd-subscriber-details

remote-address  remote-port channel
--------------  ----------- -------
10.1.21.57      49728       /call/to/019785551212
10.1.21.57      49804       /**
10.1.21.57      49804       /cometd/meta
```

Properties

The following table shows the properties for the command.

Table 5–25 Show Cometd-Subscriber-Details

Field	Description
remote-address	The remote address for the subscriber.
remote-port	The remote port for the subscriber.
channel	The name of the channel.

show cometd-subscriber-summary

The show cometd-subscriber-summary command provides high-level information about the subscribers.

Sample Output

```
NNOS-E>show cometd-subscriber-summary

remote-address  remote-port id          channel-count message-count user-agent
--------------  ----------- --          ------------- ------------- ----------
10.1.21.57      49728       21sxpszu2lkikc1pnadt0mdfzvg 1             0
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11 (KHTML, like Gecko)
Chrome/17.0.963.79 Safari/535.11
10.1.21.57      49804       372tj5ikmvga8ant2b6m2wcjs 2             0
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11 (KHTML, like Gecko)
Chrome/17.0.963.79 Safari/535.11
```

Properties

The following table shows the properties for the command.

Table 5–26 Show Cometd-Subscriber-Summary

Field	Description
remote-address	The remote address for the subscriber.
remote-port	The remote port for the subscriber.
id	The identifier assigned internally by the ME for this publisher.
channel-count	The number of channels to which the subscriber is currently subscribed.
message-count	The number of messages a subscriber has currently been sent.

Table 5–26 (Cont.) Show Cometd-Subscriber-Summary

Field	Description
user-agent	The user agent the subscriber used to establish the session.

show core-dumps

Displays all core dumps. By default, the ME does not generate core dumps. To generate them, you must enable the **services > instrument > set core_dump** property.

Sample Output

```
NNOS-E>show core-dumps

time                     file                       size       archived
----                     ----                       ----       --------
14:26:42 Wed 2013-02-13  core.sipl.elf.1360783.9012 919777280  true
14:29:24 Wed 2013-02-13  core.sipl.elf.1360783.1585 780656640  true
14:33:31 Wed 2013-02-13  core.sipl.elf.1360784.1662 780656640  false
```

Properties

The following table shows the properties for the command.

Table 5–27 Show Core-Dumps Properties

Field	Description
time	System timestamp indicating the time at which the core was dumped.
file	The name of the file that contains the core dump. These files are located in either /cxc_common/cores or /cxc_common/cores.archived depending on the value of the archived property.
size	The size of the core dump file.
archived	Indicates whether the core dump file is archived or not. When true, the file has been archived and is stored in the /cxc_common/cores.archived directory. When false, the dump files have not been archived and are stored in the /cxc_common/cores directory.

show cpu-usage

Displays CPU usage over various preset time intervals. Use the **cpu-monitor** action to do live monitoring of system use.

Sample Output

```
NNOS-E> show cpu-usage
 1 second: 0 %
10 second: 0 %
 1 minute: 1 %
10 minute: 2 %
   1 hour: 7 %
```

Properties

The following table shows the properties for the command.

Table 5–28 Show CPU-Usage Properties

Field	Description
1 second	The last reading of CPU usage on the system.
10 seconds	The average CPU usage on the system for the last 10 seconds.
1 minute	The average CPU usage on the system for the last one minute.
10 minutes	The average CPU usage on the system for the last 10 minutes.
1 hour	The average CPU usage on the system for the last one hour.

show database-maintenance-status

Displays the current maintenance status of database operations. Use this to determine whether an operation (e.g., a backup or restore) has finished. Or, if you receive an error that the ME could not execute a database operation, check this status to verify the state of the database. All previous operations must be complete (indicated by a status of idle) before a new operation can begin.

Sample Output

```
NNOS-E> show database-maintenance-status
  status: backup
   table:
 started: 09:51:23 Fri 2007-10-05
finished: 09:51:25 Fri 2007-10-05
  result: Success!
```

Properties

The following table shows the properties for the show database-maintenance-status command.

Table 5–29 Show Database-Maintenance-Status Properties

Field	Description
status	The current progress of database maintenance operations (initiated by either the **database-backup** or **database-maintenance** action or task). You can see the list of database tables using the **show database-tables** command. Table names are listed in parenthesis next to the descriptions below. The state reported indicates that the system is: **idle**: There are no current operations; you can initiate an action. **upgrading**: Executing a set of database upgrade commands to upgrade the database to a new version. The upgrade process compares the existing database version on the box to the upgrade package, and upgrades if the package version is newer. **initializing**: Initializing the database and loading the stored procedure calls. **reindexing**: Re-indexing the database tables. **analyzing**: Collecting statistics about the contents of tables in the database. **purging-sip**: Deleting SIP message table (sipmessage) entries. **purging-transport**: Deleting transport table (spotlitetransportmsg) entries. **purging-RTCP-Tx**: Deleting RTCP transmit table (spotlitertcptxmsg) entries. **purging-RTCP-Rx**: Deleting RTCP receive table (spotlitertcprxmsg) entries. **purging-monitored-calls**: Deleting monitored calls (monitoredcalls) table. **purging-URL**: Deleting URL table (urlmsg) entries. **purging-acct**: Deleting accounting table (acctcallstruct) entries. **purging-media**: Purging media message (mediamsg) table entries. **purging-file-transfer**: Purging file transfer (filetransmsg) message table entries. **purging-archive-IM**: Purging archived IM messages (archiveimmsg) table entries. **purging-calllegstart**: Purging call-leg-start (calllegstart) table entries. Continued

Table 5–29 (Cont.) Show Database-Maintenance-Status Properties

Field	Description
maintenance-status *continued*	**purging-calllegstop**: Purging call-leg-stop (calllegstop) table entries. **vacuuming**: Reclaiming storage occupied by deleted entries. **backup**: Performing a pg_dump operation. **restore**: Restoring the database from a previously backed up version. **failed**: Operation failed. Retry the operation or reload the database before calling Technical Support. **translating-tables**: Translating data into the current database format to allow for better database query performance. This state only appears when upgrading a system from 3.2.0 or earlier to a later release.
table	The name of the database table.
started	The time at which the database maintenance operation was started.
finished	The time at which the database maintenance operation was completed.
result	An indication of whether the database maintenance operation was successful or not.

show dial-plan

Displays the dial-plan table, which handles call forwarding. The output displays a summary of each *configured* (but not necessarily active) **dial-plan** entry, its match criteria and peer (and other configuration elements), and the number of times the ME has applied it to forward a call. Use the **show call-routing** command to see all *active* dial plans.

Sample Output

```
NNOS-E> show dial-plan
plan-name      type      match       min     pri     peer-name     fwd
---------      ----      -----       ---     ---     ---------
E911           default   !*          2      99                    0
default        phone     !*          2      100     Verizon       0
New York       phone     212!*       3      100     NNOS-E@NewYork   0
San Jose       phone     506!*       3      100     NNOS-E@SanJose   0
Boston         phone     617!*       1      100                   0
Maynard        phone     978!*       1      100                   0
```

Properties

The following table shows the properties for the command.

Table 5–30 Show Dial-Plan Properties

Field	Description
plan-name	The name of the active dial plan, created with the dial-plan or dial-prefix configuration.

Table 5–30 (Cont.) Show Dial-Plan Properties

Field	Description
type	The portion of the request to match on. If the INVITE matches the portion identified by the type, the system forwards the request to that server. The type can be contributed from the dial-plan configuration. Types of *tag* or *domain* can be contributed from the **auto-tag-match** and **auto-domain-match** options of the **server routing-setting** property.
match	A derivative of the regular expression or tag (for faster matching) that identifies the "to" or "from" mapping. This string is configured in the dial-plan configuration. If contributed through a **route** object entry, the string to match in the SIP header fields or transport information in order for the system to apply the plan to calls containing the prefix ("to" mapping). If contributed through a **source-route** object entry, the match criteria for the source of the SIP message ("from" mapping). The **match** field is derived from either the **to-uri-match** (route) or **source-match** (source-route) property. If type is condition-list, the match is derived from the priority plus plan name.
min	The minimum number of digits required for a match on a phone prefix, if configured. In some cases, the system calculates a value for other types of matches based on the number of characters (including wild cards). In some cases it displays as-is. The value is only meaningful to a phone-prefix match, however.
pri	The priority (order of preference) setting for the dial-plan entry. This property overrides the default behavior (most specific match) and sets a preference based on the **request-uri-match** (route) or **source-match** (source-route) property.
peer-name	A statically entered peer. This is a configured server of type **sip-registrar**.
fwd	The number of times this plan has matched an INVITE request, and the system forwarded the request. This is a counter internal to the ME.

show dns-cache

Displays the DNS cache, organized by process (monitor, manager, SIP, media, auth, etc.), on the ME. The cache displays host information, including type, state, and references. Configure DNS using the **dns** object.

Sample Output

```
NNOS-E> show dns-cache
process   name            type    ttl         state       references
-------   ----            ----    ---         -----       -----
manager   127.0.0.1       PTR     static      Resolved    0
manager   172.26.0.155    A       static      Resolved    0
manager   192.168.0.1     PTR     static      Resolved    0
SIP       10.1.34.160     PTR     static      Resolved    0
```

```
SIP       172.26.0.155    A       static      Resolved    0
SIP       192.168.0.1     PTR     static      Resolved    0
SIP       localhost       A       static      Resolved    0
SIP       vfn.com         NS      169023      Resolved    0
media     10.1.34.160     PTR     static      Resolved    0
media     127.0.0.1       PTR     static      Resolved    0
media     localhost       A       static      Resolved    0
reg       10.1.34.160     PTR     static      Resolved    0
reg       127.0.0.1       PTR     static      Resolved    0
reg       172.26.0.155    A       static      Resolved    0
reg       localhost       A       static      Resolved    0
```

Properties

The following table shows the properties for the command.

Table 5–31 Show DNS-Cache Properties

Field	Description
process	The name of the system process that did the DNS cache lookup for an entry. That entry is then installed in the process cache (each process has its own cache). A static entry is installed in every process cache.
name	The identifier for the entry (e.g., IP address, host name, etc.). The name format is determined by the record type.
type	The record type for the entry, either: **A**: Host name is a IPv4 address **AAAA**: Host name is a IPv6 address **PTR**: IP address is an address-to name-mapping pointer record (reverse lookup) **SRV**: Service name (server resource rule) **NAPTR**: Domain name (Naming Authority Pointer rule) **CNAME**: Canonical name record (makes one domain name an alias of another) **NS**: Name server record **SOA**: Server of authority record
ttl	The time to live for the entry, either a number of milliseconds or static. A value of static indicates that the entry was manually entered and will not time out of the cache.
state	The state of the entry in the cache, either Pending (resolution in progress), Resolved, or Not Available.
references	The number of accesses to that cache entry.

show dns-cache-detail

Displays DNS cache entries, organized by process. An entry is installed in a process cache (each process has its own cache) when the process does a DNS lookup for that entry. A static entry, configured with the **dns** object, is installed in every process cache.

Sample Output

```
NNOS-E> show dns-cache-detail
Process: manager
-----------------------------------------------------------------
DNS name            type    ttl      data
-----------------------------------------------------------------
10.1.34.160         PTR     static   lingo.com
127.0.0.1           PTR     static   localhost
172.26.0.155        A       static   192.168.0.1
192.168.0.1         PTR     static   172.26.0.155
localhost           A       static   127.0.0.1

Process: SIP
-----------------------------------------------------------------
DNS name            type    ttl      data
-----------------------------------------------------------------
10.1.34.160         PTR     static   lingo.com
127.0.0.1           PTR     static   localhost
172.26.0.155        A       static   192.168.0.1
192.168.0.1         PTR     static   172.26.0.155
lingo.com           A       static   10.1.34.160
localhost           A       static   127.0.0.1
```

Properties

The following table shows the properties for the command.

Table 5–32 Show DNS-Cache-Detail Properties

Field	Description
DNS name	The main name used by the ME for the DNS entry. This is the name configured as the **host**, **service**, or **naptr** in the **dns** configuration object (static record) or learned by the ME.
type	The record type for the entry, either: **A**: Host name is a IPv4 address **AAAA**: Host name is a IPv6 address **PTR**: IP address is an address-to name-mapping pointer record **SRV**: Service name (server resource rule) **NAPTR**: Domain name (Naming Authority Pointer rule) **CNAME**: Canonical name record (makes one domain name an alias of another) **NS**: Name server record **SOA**: Server of authority record
ttl	The time to live for the entry, either a number of milliseconds or static. A value of static indicates that the entry was manually entered and will not time out of the cache.
Data	The resolution of the DNS lookup, depending on the record type. For PTR and A records, this field contains an IP address or domain name. For SRV and NAPTR records, the data field displays the configured rules that describe lookup procedures and precedence.

show dns-resolver

Displays information identifying the servers that receive the ME resolver requests and various counters. As a resolver, the ME obtains resource records from servers on behalf of resident or requesting applications. The load-balancing algorithms determine which server receives the request. Note that if you change the DNS server configuration (with the **server** property of the **resolver** object), the ME resets the DNS resolver statistics.

Sample Output

```
NNOS-E> show dns-resolver
        process: manager
     preference: 100
         server: 172.31.4.50:53
     domainName: vfn.com
       protocol: UDP
          state: up
   echoRequests: 0
    echoReplies: 0
       requests: 0
      responses: 0
  request-fails: 1
       discards: 0
        retries: 1
pending-queries: 1
   sip-location: ALL

        process: manager
     preference: 100
         server: 172.31.4.51:53
     domainName: vfn.com
       protocol: UDP
          state: up
   echoRequests: 0
    echoReplies: 0
       requests: 0
      responses: 0
  request-fails: 0
       discards: 0
        retries: 0
pending-queries: 4294967295
   sip-location: ALL

--more--
```

Properties

The following table shows the properties for the show dns-resolver command.

Table 5–33 Show Dns-Resolver Properties

Fields	Description
Field	Description
time	The current time as configured on the box. You can set (or reset) the system time with the **clock** action
uptime	The amount of time since the last system boot.
Field	Description

Table 5–33 (Cont.) Show Dns-Resolver Properties

Fields	Description
time	The current time as configured on the box. You can set (or reset) the system time with the **clock** action
uptime	The amount of time since the last system boot.
Field	Description
echoReplies	The number of ICMP echo responses sent from the server.
requests	The number of DNS requests sent to the server.
responses	The number of responses to DNS requests sent from the server.
request-fails	The number of transaction-layer failures (e.g., timeout or socket failure).
discards	The number of discarded resource record requests.
retries	The number of retries on request failures.
pending-queries	The number of as-yet unanswered queries awaiting resource record responses from the DNS server.
sip-location	The DNS lookup behavior, which determines how the system should attempt to locate a SIP server using DNS when it receives a SIP message that is not destined to any dial plan or locally registered user. If configured as a resolver, the system attempts to obtain the SIP NAPTR or SRV record for the domain. This parameter sets the behavior if the system does not find that NAPTR or SRV record. Method could be: ALL: Uses NAPTR, SRV or Address records (the default) NONE: Prevents the use of DNS to locate a SIP server SRV: Uses the service record only (no NAPTR lookup) AAA: Uses the Address record only (no NAPTR/SRV lookup) RFC3263: Uses NAPTR and SRV records as described in *RFC 3263, Session Initiation Protocol (SIP): Locating SIP Servers* (no AAA lookup)

show dos-rules

Displays a summary and statistics for each DOS rule (transport and SIP) created by your DOS policies. It displays the results of each rule generated by the DOS engine (what got locked out). The command only displays output if:

- The ME is currently under DOS attack.
- You have configured DOS policies that are thwarting that attack.

Sample Output

```
NNOS-E> show dos-rule
rule-number: 1
last-timestamp: 13:22:17 Wed 2007-04-11
packet-count: 610825
action: filter
missive: sip policy: s1, filter localIP: 10.1.208.140 requestMethod: INVITE
requestUriUserHost: proxy.companyXYZ.com
```

Properties

The following table shows the properties for the show dos-rules command.

Table 5–34 Show DOS-Rules Properties

Field	Description
rule-number	Number assigned to rule in the index of rules.
last-timestamp	The last time a packet tripped the rule.
packet-count	Total number of packets to hit this rule.
action	The action to take. Either: Filter these SIP packets (throw away). Alert (allow these SIP packets to go through, but send a log message and SNMP traps).
missive	A statement for the given policy rule specifying what kind of packets to look for and which ones to drop.

show dynamic-event-services

Once an application has registered itself to receive events, you can view information about the registration via this status provider.

Sample Output

```
NNOS-E>show dynamic-event-services

       endpoint: 10.0.0.10
registration-id: d710c03c-70b3-454d-9ee2-c1b6f60dd5b7
        created: 12:10:20.857000 Thu 2012-03-01
   time-to-live: untilRestart seconds
 last-keepalive: 12:10:20.857000 Thu 2012-03-01
       channels:
connect-timeout: 1000 ms
   read-timeout: 1000 ms
  character-set: utf-8
  request-style: soap
       requests: 0
       failures: 0
```

Properties

The following table shows the properties for the show dynamic-event-services command.

Table 5–35 Show Dynamic-Event-Services Properties

Field	Description
endpoint	The application endpoint being called out.
registration-id	The registration identifier.
created	The date and time this registration was created.
time-to-live	The configured time to live, in minutes, on this registration.

Table 5–35 (Cont.) Show Dynamic-Event-Services Properties

Field	Description
last-keepalive	The date and time that the last keep alive was received.
channels	The channels for which the endpoint is getting events.
connect-timeout	The configured connect timeout, in milliseconds, for the endpoint.
read-timeout	The configured read timeout, in milliseconds, for the endpoint.
character-set	The character set used when forming requests to this endpoint. This can be either utf-8 or iso-8859-1.
request-style	The style used when sending events to this listener. This could be either XML, JSON, or SOAP.
requests	The number of requests that have been made to the endpoint.
failures	The number of requests that have failed to reach the endpoint.

show ethernet

Displays configuration and status information for each configured Ethernet interface.

Sample Output

```
NNOS-E>show ethernet -v
name: eth0
link: up
speed: 1Gb
duplex: full
autoneg: enabled
rx-ring: 0
rx-ring-max: 0
tx-ring: 0
tx-ring-max: 0
```

Properties

The following table shows the properties for the show ethernet command.

Table 5–36 Show Ethernet

Field	Description
name	The name of the Ethernet interface. You can configure up to 20 gigabit Ethernet, full-duplex interfaces. The actual number available depends on your hardware configuration
link	The operational state of the Ethernet interface, either up or down. For the link to be up, the interface must be administratively enabled with a link detected on the line.

Table 5–36 (Cont.) Show Ethernet

Field	Description
speed	The speed of the Ethernet connection between the ME and the piece of equipment to which it is connected. The value displayed is the speed configured with the **speed** property of the **interface** object. However, the ME ignores this value if **autoneg** is set to **enabled**.
duplex	The acceptable duplex method for the interface, either half (asynchronous) or full (simultaneous) transmission. The value displayed is the setting configured with the **duplex** property of the **interface** object. However, the ME ignores this value if **autoneg** is set to **enabled**.
autoneg	The autonegotiation setting for the interface. If **enabled**, the system negotiates with the piece of equipment to which it is connected to achieve optimal agreed upon settings. If **disabled**, the system uses the configured (or default) settings.
rx-ring	The current Ethernet Rx ring size.
rx-ring-max	The maximum allowed Ethernet Rx ring size.
tx-ring	The current Ethernet Tx ring size.
tx-ring-max	The maximum allowed Ethernet Tx ring size.

show eventcore-channelevt <*channel*>

Displays all events currently associated with the specified channel.

Sample Output

```
NNOS-E>show eventcore-channelevt

channelName         status        arrivalTime                    eventType
-----------         ------        -----------                    ---------
webphone_channel    Available     09:34:26.965313 Tue 2012-03-13 CallCreated
```

Properties

The following table shows the properties for the show eventcore-channelevt <*channel*> command.

Table 5–37 Show Eventcore-channelevt <channel> Properties

Field	Description
channelName	The name of the channel.
status	Indicates whether an event has been forwarded or not.
arrivalTime	The date and time the event was received by the eventcore.
eventType	The event class name.

show eventcore-channels

Provides a list of all existing event core channels, indicating the total number of events currently stored in each channel queue, as well as the number of events that have been forwarded to publishers.

Sample Output

```
NNOS-E>show eventcore-channels

channelName          totalItems unforwardedItems TimeCreated
-----------          ---------- ---------------- -----------
webphone_channel     100        27               09:34:26.965313 Tue 2012-03-13
```

Properties

The following table shows the properties for the show eventcore-channels command.

Table 5–38 Show Eventcore-Channels Properties

Field	Description
channelName	The name of the channel.
totalItems	The number of events currently stored in the channel's event queue.
unforwardedItems	The number of events currently stored in the channel's event queue waiting to be forwarded.
TimeCreated	The date and time the channel was created.

show eventcore-publishers

Provides a list of all publishers requesting event delivery from the event core.

Sample Output

```
NNOS-E>show eventcore-publishers

num  id         cookie     createTime               location
fwdcount
---  --         ------     ----------               --------
--------
1    0xc0f5c7b3 0x00000000 10:46:30.362959 Mon 2012-03-19 0.0.0.0 19 1 162   13
2    0xc0f5c7fb 0x00000000 10:46:30.398149 Mon 2012-03-19 0.0.0.0 10 1 89    13
1    0xc0f5b4d3 0x00000000 09:34:26.96 Tue 2012-03-13 0.0.0.0 19 1 162  26
2    0xc0f5b51b 0x00000000 09:34:26.97 Tue 2012-03-13 0.0.0.0 10 1 89   26
```

Properties

The following table shows the properties for the show eventcore-publishers command.

Table 5–39 Show Eventcore-Publishers Properties

Field	Description
num	The publisher's index number.
id	The identifier assigned internally by the ME for this publisher.

Table 5–39 (Cont.) Show Eventcore-Publishers Properties

Field	Description
cookie	The identifier sent to the ME by the publisher. This ID is returned to the publisher within each forwarded event.
createTime	The date and time that the publisher requested event notifications from the ME.
location	The publisher's cluster endpoint identifier.
fwdcount	The total number of events that have been forwarded to the publisher.

show event-log

Displays the time, severity, box, process, and message for each event in the ME local database. You must enable global event-log administration (through the **event-log** object) for data to be written to the event log. Set the severity level, log class, and history properties using the event-log **local-database** object. See the appendix for a list of events and their corresponding severities.

Sample Output

```
NNOS-E> show event-log
timestamp               severity   box      process   class      message
---------               --------   ---      -------
10:57:27 Tue 2006-11-07  error      1        manager   sm         Cert entry 'vfn'
could not parse certificate file '/cxc/certs/vfn.p12'; could be wrong passphrase
(secret tag is pass), or unsupported format (PEM or PKCS#12)
10:57:28 Tue 2006-11-07  error      1        manager   sm         OpenSSL error
from PKCS12_parse()() (returned 0):
10:57:33 Tue 2006-11-07  error      1        SIP       sm         OpenSSL error
23076071: error:23076071:PKCS12 routines:PKCS12_parse:mac verify failure (p12_
kiss.c:117)
10:57:33 Tue 2006-11-07  alert      1        SIP       sipTLS     Reinitializing
table AssociationSQL in database spotlite
10:57:40 Tue 2006-11-07  alert      1        SIP       snmp       VRRP Group 2
failover
10:57:46 Tue 2006-11-07  alert      1        manager   snmp       VRRP vinterface
vx112 failover: went to master on interface eth2
```

Properties

The following table shows the properties for the show event-log command.

Table 5–40 Show Event-Log Properties

Field	Description
timestamp	The time at which the event occurred, according to the system clock.
severity	The severity of the event. The severity indicates the lowest level message to display. You receive messages of that class and below, with Emergency being the lowest and Debug the highest. Set the severity level you wish to display using the **local-database** object **filter** property.
box	The box in the cluster on which the event occurred. If the box is not in a cluster, the system reports the box number as 1.

Table 5–40 (Cont.) Show Event-Log Properties

Field	Description
process	The process that sent the event to the local database. You can display a list of eligible processes using the **show processes** command.
class	The log class related to the event, which indicates the subsystem that generated the message. Set the severity level you wish to display using the **local-database** object **filter** property.
message	The text of the event message.

show exceptions

Provides information about exceptional conditions that caused the ME to purposely restart the process.

Sample Output

```
NNOS-E>show exceptions

   time: 17:01:46 Mon 2013-03-11
   file: media1-1363035706.txt
address:
 reason: Aborted
 uptime: 0 days 00:00:49
version: E3.7.0
  build: 56481
 branch: crux
```

Properties

The following table shows the properties for the show exceptions command.

Table 5–41 Show Exceptions Properties

Field	Description
time	System timestamp indicating the time of the crash.
file	The name of the file to which the system wrote the crash data and analysis. Files are stored in the directory cxc_common/crash.
address	The ME address at which the crash occurred.
reason	A brief description of the cause of the crash.
uptime	The length of time the system was up before the crash occurred.
version	The software version installed on the system at the time of the crash.
build	The specific build of the software version that was installed on the system at the time of the crash.
branch	The internal development tracking ID for the build that was installed on the system at the time of the crash.

show faults

Displays information about unexpected process crashes that have occurred.

Sample Output

```
NNOS-E> show faults
   time: 16:32:53 Wed 2006-12-20
   file: SIP1-000.txt
address:
 reason: Aborted
 uptime: 1 days 00:10:15
version: 3.2.0
  build: 22831
 branch: b3.2.0

   time: 16:41:48 Wed 2006-12-20
   file: SIP1-001.txt
address:
 reason: Aborted
 uptime: 0 days 00:07:52
version: 3.2.0
  build: 22831
 branch: b3.2.0

   time: 17:37:15 Wed 2006-12-20
   file: SIP1-002.txt
address: b7cfbc04 __vsnprintf + 0x59 /lib/libc-2.3.4.so
 reason: Segmentation fault
 uptime: 0 days 00:55:20
version: 3.2.0
  build: 22831
 branch: b3.2.0
```

Properties

The following table shows the properties for the show faults command.

Table 5–42 Show Faults Properties

Field	Description
time	System timestamp indicating the time of the crash.
file	The name of the file to which the system wrote the crash data and analysis. Files are stored in the directory cxc_common/crash.
address	The ME address at which the crash occurred.
reason	A brief description of the cause of the crash, for example, OOM (out of memory) or segmentation fault.
uptime	The length of time the system was up before the crash occurred.
version	The software version installed on the system at the time of the crash.
build	The specific build of the software version that was installed on the system at the time of the crash.
branch	The internal development tracking ID for the build that was installed on the system at the time of the crash.

show features

Displays each licensed feature for the ME and its capacity. The output also displays any changes to capacity (either number of sessions or endpoints), current use, and a running total of use. Capacity changes can be implemented using the **features** object.

The example below shows only a small portion of the output. Fields displayed are the same for each feature type. See the **features** object description in Configuring Feature Licensing Objects for a description of each licensable feature.

Sample Output

```
NNOS-E> show features
 feature: signaling-sessions
licensed: 200000
 current: 1
 maximum: 6
   total: 88
failures: 0

 feature: media-sessions
licensed: 200000
 current: 1
 maximum: 0
   total: 9
failures: 0

 feature: instant-message-and-presence-sessions
licensed: 200000
 current: 0
 maximum: 0
   total: 0
failures: 0
.
.
.
```

Properties

The following table shows the properties for the show features command.

Table 5–43 Show Features Properties

Fields	Description
feature	The name of the feature that is licensed. If a feature is not licensed, it does not display in this list.
licensed	The number of sessions or endpoints allowed concurrently, as specified in the license. Typically this is the value from the license, but if you reset the value with the features object, that setting displays here.
current	The number of sessions or endpoints using the license at the moment of command execution.
maximum	The maximum number of licenses ever used on the box.
total	The total number of accesses to the feature.
failures	The number of times, after a license maximum was reached, that a user or application tried to use the feature.

show ice-dtls-status

Displays information per call leg for sessions using DTLS encryption.

Sample Output

```
OS-E>show ice-dtls-status
session-id: 0x4c40106b423123b
leg: 1
stream: 0
address: 172.30.12.82:24472
remote: 172.30.12.82:24352
type: 1-RTP
role: Passive
state: Succeed
```

Properties

The following table shows the properties for the show features command.

Table 5–44 Show Ice-Dtls-Status Properties

Fields	Description
session-id	The unique ID of the ME session.
leg	Specifies in-leg (0) or out-leg (1).
stream	The media stream index, audio (0) or video (1).
address	The local ME IP and port for this DTLS socket.
remote	The remote peer IP and port for this DTLS socket.
type	Specifies the type of ICE port, either RTP (1) or RTCP (2).
role	Specifies the DTLS role, either Active or Passive.
state	The state of the DTLS socket, either Connected, Listening, Succeeded, or Closed.

show ice-candidate-pair-status

Displays information and state for each ICE candidate pair.

Sample Output

```
NNOS-E>show ice-pair-candidates
     session-id: 0x8c4ef6081de8c26
     leg: 1
     checklist: 0
     local:172.44.10.55:20676 UDP
     remote: 172.44.10.57:22656 UDP
     state: Succeeded
     componentID: 2130706430
     nominated: true
```

Properties

The following table shows the properties for the **show ice-candidate-pair-status** command.

Table 5–45 Show Ice-Candidate-Pair-Status Properties

Field	Description
session-id	The session ID on which ICE is running.

Table 5–45 (Cont.) Show Ice-Candidate-Pair-Status Properties

Field	Description
leg	The call leg on which ICE is running.
checklist	The checklist that owns the candidate pair. This is also known as the media description index.
local	The local candidate in the pair.
remote	The remote candidate in the pair.
state	The pair state. This can be either Frozen, Waiting, Succeeded, or Failed.
componentID	The componentID of the pair. This value is an integer.
nominated	Specifies whether or not this pair has been nominated for media transmission.

show ice-local-candidates

Displays ICE information for the local candidates used by each state machine.

Sample Output

```
NNOS-E>show ice-local-candidates
     session-id: 0x8c4ef6081de8c26
     leg: 1
     checklist: 0
     transport: 172.44.10.60:20927
     componentID: UDP
     type: host
     priority: 2130706430
     foundation: 1
```

Properties

The following table shows the properties for the **show ice-local-candidate** command.

Table 5–46 Show Ice-Local-Candidate Properties

Field	Description
session-id	The ID of the session that owns the ICE state machine.
leg	The call-leg on which the ICE state machine is running.
checklist	The checklist number that owns the candidate. This is also known as the media description index.
transport	The IP, port, and transport protocol of the candidate.
componentID	.The ICE component ID. This value is an integer.
type	The ICE candidate type. This can be either host, srflx, prflx, or relay.
priority	The candidate priority.
foundation	The foundation string.

show ice-remote-candidates

The following table shows the properties for the **show ice-local-candidate**.

Sample Output

```
NNOS-E>show ice-remote-candidates
     session-id: 0x8c4ef7581fe8c27
     leg: 1
     checklist: 0
     transport: 172.66.20.40:20927
     componentID: UDP
     type: host
     priority: 2130706430
     foundation: 1
```

Properties

The following The following table shows the properties for the **show ice-remote-candidate**.

Table 5–47 Show Ice-Remote-Candidates Properties

Field	Description
checklist	The checklist number that owns the candidate. This is also known as the media description index.
transport	The IP, port, and transport protocol of the candidate.
componentID	.The ICE component ID. This value is an integer.
type	The ICE candidate type. This can be either host, srflx, prflx, or relay.
priority	The candidate priority.
foundation	The foundation string.
checklist	The checklist number that owns the candidate. This is also known as the media description index.
transport	The IP, port, and transport protocol of the candidate.

show interface-details

Displays statistics and the MAC address for each configured Ethernet interface. Interfaces are configured using the **interface** object.

Sample Output

```
NNOS-E> show interface-details
          name: eth0
   mac-address: 00:04:23:c3:22:04
      op-state: up
     rcv-bytes: 27723136
   rcv-packets: 195105
      rcv-errs: 0
      rcv-drop: 0
      rcv-fifo: 0
     rcv-frame: 0
rcv-compressed: 0
 rcv-multicast: 0
      tx-bytes: 220274389
    tx-packets: 3350328
       tx-errs: 0
       tx-drop: 0
       tx-fifo: 0
```

```
      tx-colls: 0
    tx-carrier: 0
 tx-compressed: 0

          name: eth1
   mac-address: 00:04:23:c3:22:05
      op-state: up
     rcv-bytes: 1978425176
   rcv-packets: 6394797
      rcv-errs: 0
      rcv-drop: 0
      rcv-fifo: 0
     rcv-frame: 0
rcv-compressed: 0
 rcv-multicast: 0
      tx-bytes: 1335807214
    tx-packets: 5548521
       tx-errs: 0
       tx-drop: 0
       tx-fifo: 0
      tx-colls: 0
    tx-carrier: 0
 tx-compressed: 0
```

Properties

The following table shows the properties for the show interface-details command.

Table 5–48 Show Interface-Details Properties

Field	Description
name	The name of the Ethernet interface.
mac-address	The MAC address assigned to the Ethernet interface.
op-state	The operational state of the interface, either up or down.
rcv-bytes tx-bytes	The number of bytes transmitted or received without error on the interface.
rcv-packets tx-packets	The number of packets transmitted or received without error on the interface.
rcv-errs tx-errs	The number of errored packets transmitted or received on the interface.
rcv-drop tx-drop	The number of packets dropped during the transmit or receive process because the queue was full.
rcv-fifo tx-fifo	The number of times a FIFO underrun occurred during transmit or receive.
rcv-frame	The number of receive packets with bad framing bytes.
rcv-compressed tx-compressed	Total number of compressed packets sent or received by the interface.
rcv-multicast	The number of multicast packets received on the interface.

Table 5–48 (Cont.) Show Interface-Details Properties

Field	Description
tx-colls	The number of transmit collisions detected on the interface.
tx-carrier	The number of times the carrier sense was lost during transmit.

show interface-throughput

Displays, for each Ethernet **interface**, the throughput across the interface measured in packets per second (pps) and kilobits per second (kbps), for both transmit and receive. (Each interface has four entries.)

Sample Output

```
NNOS-E> show interface-throughput
name  value   10 second  1 minute   10 minute  1 hour     maximum
----  -----   ---------  --------   ---------  ------     -------
eth0  rx-kbps 0          0          0          0          16
eth0  tx-kbps 10         10         10         10         31
eth0  rx-pps  0          0          0          0          15
eth0  tx-pps  20         20         20         20         28
eth1  rx-kbps 779        388        249        298        22376
eth1  tx-kbps 12309      3924       1004       316        13663
eth1  rx-pps  1163       436        180        134        4051
eth1  tx-pps  1627       574        206        126        1797
eth2  rx-kbps 0          0          0          0          0
eth2  tx-kbps 0          0          0          0          0
eth2  rx-pps  0          0          0          0          0
eth2  tx-pps  0          0          0          0          1
eth3  rx-kbps 0          0          0          0          0
eth3  tx-kbps 0          0          0          0          0
eth3  rx-pps  0          0          0          0          0
eth3  tx-pps  0          0          0          0          0
```

Properties

The following table shows the properties for the show interface throughput command.

Table 5–49 Show Interface-Throughput Properties

Field	Description
name	The name of the Ethernet interface.
value	The measurement for the corresponding interface, either receive (rx) or transmit (tx) packet and speed rates.
10 second	The average rate in the last ten seconds.
1 minute	The average rate in the last one minute.
10 minutes	The average rate in the last ten minutes.
1 hour	The average rate in the last one hour.
maximum	The maximum value recorded since last system boot.

show interfaces

Displays the name, IP address, MAC address, and the current operational state of each displays administratively enabled IP interfaces on the system. Interfaces are configured using the **interface** object.

Sample Output

```
NNOS-E> show interfaces
interface   name          ip-address         op-state      type
---------   ----          ----------         --------      ----
eth0        heartbeat      192.168.0.1/30     up            public
eth1:1      public         215.2.3.0/24       down          public
vx111       Management     172.100.0.10/24    up            public
vx112       Public         10.10.10.10/24     up            public
```

Properties

The following table shows the properties for the show interfaces command.

Table 5–50 Show Interfaces Properties

Fields	Descriptions
interface	The interface on which the system is reporting status, either an Ethernet (ethX) or virtual (vx) interface. The name displayed is the unique OS interface name.
name	The name given to the IP interface when it was created (the IP configuration object name).
ip-address	The IP address assigned to the IP interface.
op-state	The operational state of the interface.
type	The type of interface, either public or private. A standard interface is public. IP interfaces configured for virtual firewalls are private.

show ip-counters

Displays IP statistics for all interfaces configured for IP routing.

Sample Output

```
NNOS-E> show ip-counters
       forwarding: 1
      default-ttl: 64
      in-receives: 5973190
    in-hdr-errors: 0
   in-addr-errors: 0
   forw-datagrams: 0
in-unknown-protos: 0
      in-discards: 0
      in-delivers: 5636163
     out-requests: 8823298
     out-discards: 0
    out-no-routes: 0
    reasm-timeout: 0
      reasm-reqds: 0
        reasm-oks: 0
      reasm-fails: 0
```

```
    frag-oks: 0
  frag-fails: 0
frag-creates: 0
```

Properties

The following table shows the properties for the show ip-counters command.

Table 5–51 Show IP-Counters Properties

Field	Description
forwarding	The state of the interface, either forwarding (1) or not forwarding (2).
default-ttl	The default time-to-live value inserted into IP headers.
in-receives	The number of incoming datagrams received, including those received with errors.
in-hdr-errors	The number of incoming datagrams discarded due to errors in the IP header (e.g., bad checksum, ttl expire, version mismatch, etc.).
in-addr-errors	The number of incoming datagrams discarded because the IP address in the destination field was in error.
forw-datagrams	The number of datagrams forwarded.
in-unknown-protos	The number of datagrams discarded due to an unknown or unsupported protocol.
in-discards	The number of non-errored incoming datagrams that were discarded (e.g., due to congestion).
in-delivers	The number of ICMP, UDP, BOOTP, and TCP incoming datagrams that were successfully delivered to higher-layer protocols.
out-requests	The number of datagrams that local IP protocols requested be transmitted transmission.
out-discards	The number of outgoing error-free datagrams discarded, for example due to buffer overload.
out-no-routes	The number of outgoing datagrams discarded because a valid route could not be found.
reasm-timeout	The maximum number of seconds that the pieces of a fragmented datagram can be held awaiting reassembly before they are discarded.
reasm-reqds	The number of fragments received that required reassembly.
reasm-oks	The number of datagrams successfully reassembled.
reasm-fails	The number of datagrams that could not be reassembled.
frag-oks	The number of datagrams successfully fragmented.
frag-fails	The number of datagrams that were discarded because they could not be fragmented (e.g., because the DoNot Frag flag was set).
frag-creates	The number of datagram fragments created.

show kernel-rule

Displays the rules, derived from **dos-policies**, that reside in the kernel and how the ME acts on them (used primarily for debugging). The output could potentially contain several thousand rules. For simple debugging, use the command to verify that the hit count (Pass) is increasing and that the Drop count is not. An increase in **Forced** is not problematic however. **Forced** indicates the result of rules being set to drop specific types of packets. For example, a **Forced** drop could indicate that a rule was set to drop DTMF packets. In another example, the kernel could drop a packet from further processing until the criteria are met because RTCP has not seen the minimum number of consecutive packets specified in the rule.

Sample Output

```
NNOS-E> show kernel-rule

source              dest                Prot intf         info
------              ----                ---- ----         ----
0.0.0.0:0           255.255.255.255:67  udp  eth2         Pass: 0  Drop: 0
                                                          End
0.0.0.0:0           224.0.0.18:0        vrrp eth2         Pass: 612673  Drop: 0
                                                          End
0.0.0.0:0           172.26.0.56:0       all               Pass: 12954  Drop: 0
                                                          End
0.0.0.0:0           192.168.217.1:0     all               Pass: 48321  Drop: 0
                                                          End
0.0.0.0:0           192.168.215.101:0   all               Pass: 0  Drop: 0
                                                          End
0.0.0.0:0           172.26.0.56:0       udp  eth0         Pass: 50979  Drop: 0
                                                          PortTracker
                                                          End
0.0.0.0:0           192.168.215.101:0   udp  vx0          Pass: 0  Drop: 0
                                                          PortTracker
                                                          End
```

Properties

The following table shows the properties for the show kernel-rule command.

Table 5–52 Show Kernel-Rule Properties

Field	Description
source	The source IP address of packets dropped. An address of 0.0.0.0 indicates that the system will apply the rule to any IP address. The suffix :0 indicates that the system will apply the rule to any port (TCP or UDP).
destination	The destination IP address of packets dropped.
protocol	The protocol that the incoming traffic is using. Any protocol is valid.
interface	The specific physical interface.
information	A description of how the rule was applied to each source and destination in terms of hits and drops.

show kernel-sip-rules

Displays, for all traffic except TLS, filtering at the kernel level instead of at the application layer (use **show dos-rules** for the application layer). If a DOS attack is in progress, these are the rules that block the attack. The application layer detects the attack and creates the rule; whereas, at the kernel level the attack is actually blocked. The kernel level does not perform decryption. Since TLS is encrypted, handling of the DOS attack is elevated to the **dos-rules** level, where decryption can occur and where the attack is blocked only if all criteria for the block match.

Sample Output

```
NNOS-E> show kernel-sip-rules
rule: 1
packet-count: 624872
remote: any
local: 10.1.208.140
protocol: any
action: filter
match: method: "invite" request-uri: ""
```

Properties

The following table shows the properties for the kernel sip-sip-rules command.

Table 5–53 Show Kernel-SIP-Rules Properties

Field	Description
rule	The internal number assigned to the rule in the index of rules.
packet-count	The number of packets received that encountered this rule in this attack.
remote	The remote IP address that is the source of the packet, as specified by the DOS rule. This may be a specific address or "any" to indicate that any sending endpoint that matches the other criteria qualifies for filtering.
local	The destination address of the incoming packets.
protocol	The protocol specified in the policy. If not specified, the value is any.
action	The action taken by the system on matching packets, either: Filter these SIP packets (throw away) Alert (allow these SIP packets to go through, but send a log message and SNMP traps)
match	A description of the packet type and fields that are being examined. The method field indicates the message type. If the DOS rule specified header and match strings, the output displays those as well.

show kernel-version

Displays detailed information about the kernel currently running.

Sample Output

```
NNOS-E> show kernel-version

version       build        branch        time                     computer
-------       -----        ------        ----                     -
2.6.11-4-cov  25500:25501S kernel-2.6.11-4 15:42:49 Tue 2007-04-03  dubuc
```

Properties

The following table shows the properties for the show kernel-version command.

Table 5–54 Show Kernel-Version Properties

Field	Description
version	The version number of the kernel currently running.
build	The build number of the kernel currently running.
branch	The branch of the kernel currently running.
time	The time when the kernel was built.
computer	The name of the system on which the kernel was built.

show license-details

Displays configurable features based on the license allowances. Features can be managed using the **features** object, but ultimately are determined by your purchased licensed. Output indicates the objects (class) and property that the license controls. The sample output shows only a piece of representational command output.

Sample Output

```
NNOS-E> show license-details
name                      class                  property
----                      -----                  --------
INTERNAL BULK LICENSE     accounting
INTERNAL BULK LICENSE     box                    number
INTERNAL BULK LICENSE     cluster                vrrp
INTERNAL BULK LICENSE     database
INTERNAL BULK LICENSE     enterprise             directories
INTERNAL BULK LICENSE     entry                  authentication
INTERNAL BULK LICENSE     entry                  media-type
INTERNAL BULK LICENSE     features               audio-recording-entities
INTERNAL BULK LICENSE     features               audio-recording-sessions
INTERNAL BULK LICENSE     features               cpus
INTERNAL BULK LICENSE     features               crypto
INTERNAL BULK LICENSE     features               dos-protection-entities
INTERNAL BULK LICENSE     log-alert              message-logging
INTERNAL BULK LICENSE     mcafee
INTERNAL BULK LICENSE     mcafee                 update-url
INTERNAL BULK LICENSE     media
INTERNAL BULK LICENSE     media                  anchor
INTERNAL BULK LICENSE     media                  nat-traversal
INTERNAL BULK LICENSE     media                  packet-marking
INTERNAL BULK LICENSE     media                  recording-policy
INTERNAL BULK LICENSE     media                  verify
INTERNAL BULK LICENSE     permissions            debug
```

```
INTERNAL BULK LICENSE     policies                dos-policies
INTERNAL BULK LICENSE     preferences             dos-queries
INTERNAL BULK LICENSE     presence                presence-translation
INTERNAL BULK LICENSE     session-config          authentication
INTERNAL BULK LICENSE     session-config          csta-settings
INTERNAL BULK LICENSE     session-config          file-transfer
INTERNAL BULK LICENSE     session-config          forking-settings
INTERNAL BULK LICENSE     session-config          header-settings
INTERNAL BULK LICENSE     session-config          media-type
INTERNAL BULK LICENSE     vsp
INTERNAL BULK LICENSE     vsp                     call-admission-control
INTERNAL BULK LICENSE     vsp                     max-calls-in-setup
```

Properties

The following table shows the properties for the show license-details command.

Table 5–55 Show License-Details Properties

Field	Description
name	The name of the license that controls the class and property.
class	The configuration object that the license controls.
property	The property of the object (class) that the license controls.

show licenses

Displays summary information for the active license.

Sample Output

```
NNOS-E> show licenses
       name: INTERNAL BULK LICENSE
description: INTERNAL BULK LICENSE
        key: 84420f9a-da13-4107-8833-d00b7d4d751d
    expires:
       file: 84420f9a-da13-4107-8833-d00b7d4d751d.xml
```

Properties

The following table shows the properties for the show licenses command.

Table 5–56 Show Licenses Properties

Field	Description
name	The name of the license, as provided.
description	A text field, provided, to help identify the contents of the license.
key	The private key provided to you by Oracle. This key is used for authentication when you contact the Oracle licensing server. You need to supply this value to retrieve a modified license, for example, for an extension on the expiration date.

Table 5–56 (Cont.) Show Licenses Properties

Field	Description
expires	The date at which the license expires. The ME generates an event when it nears that expiration date. You can renew your license by re-executing the license fetch command. The Oracle license server verifies that there is a valid license renewal associated with your system ID, and then resets the license expiration to a new date.
file	The name of the actual file on the system that contains the license. The name will be the same as the key in most circumstances.

show location-bindings

Displays registration status and location information for each binding of each AOR in the location cache. This command provides information on how to contact an endpoint. The cache is the location information for the local box. Use the **show location-database-bindings** command to see bindings for all entries shared throughout the cluster.

Sample Output

```
NNOS-E> show location-bindings
AOR                          STATE        HOST            PORT   TPT   EXP
---                          -----        ----            ----   -
sip:2125551111@vfn.com       registered   172.30.0.208    2057   UDP   225
sip:2125552222@vfn.com       registered   172.30.0.208    2057   UDP   220
```

Properties

The following table shows the properties for the show location-bindings command.

Table 5–57 Show Location-Bindings Properties

Field	Description
AOR	The entry in the location cache. Typically, this is the URI associated with the SIP user.
STATE	The state of the AOR.
HOST	The location the ME should use to reach this AOR. Displays as a host name or IP address.
PORT	The contact port on the ME used to reach this AOR.
TPT	The protocol used with this AOR. Note that if the AOR display name begins with “sips,” the protocol must be TLS.
EXP	The number of seconds, as reported in the 200 OK, until the binding expires if it is not renewed.

A binding in the location cache can be in one of 13 states:

Table 5–58 Binding States

State	Description
requested	REGISTER request received.
trying	REGISTER forwarded and waiting.
responded	REGISTER response received.
aborted	REGISTER aborted from trying.
waiting	Waiting on server busy and will re-register in brief interval.
challenged	SIP 401/407 “Auth Required” response has been sent to the endpoint.
unauthenticated	Client did not responded to challenge in challenge-timeout period.
declined	REGISTER declined with proper code; the ME continues to process subsequent REGISTERs.
rejected	All REGISTERs for this binding were rejected with proper code before session was created.
discarded	All REGISTERs for this binding were discarded silently before session was created.
registered	This binding is valid and registered.
aged	This binding is aged but not deleted.
disconnected	The TCP/TLS connection for this binding is broken.

show location-cache

Displays the location database known to the local box. The location cache is the local listing of AORs. Use **show location-database** to view the shared location service across a cluster.

The output displays state and registry information for each static and learned address of record in the local ME database of AORs. All location record types are stored in the location cache, a binary tree-based table that contains all location bindings.

This command displays only the AORs; see **show location-bindings** to display each binding that is associated with the AOR.

Sample Output

```
NNOS-E> show location-cache
AOR                                BOX   STATE        BD   SERVER     HITS   CL
                                  ---   -----        --   ------     ----   --
sip:15554443333@test.babytel.ca    1     registered   2    btel-jim   109    2
sip:15554442222@test.babytel.ca    1     registered   4    btel-jim   141    4
sip:2125551111@voip2.cov.com       1     registered   2    bsoft      74     1
sip:2125552222@voip2.cov.com       1     in-service   2    bsoft      20     0
sip:2125553333@voip2.cov.com       1     registered   1    bsoft      1393   0
sip:2125554444@voip2.cov.com       1     registered   1    bsoft      2673   0
sip:jdoe@lcs.companyXYZ.com        1     unregistered 1    Eclipse    20     0
```

Properties

The following table shows the properties for the show location-cache command.

Table 5–59 Show Location-Cache Properties

Fields	Description
AOR	The entry in the location cache. Typically, this is the URI associated with the SIP user.
BOX	The number of the box that the AOR was registered on (learned from). The output displays “1” for the local box, either standalone or cluster.
STATE	The actual current state of the AOR, including any intermediate states (e.g., WAITING, TRYING). Contrast this to the **show location-database** command, where the system only displays the final state (e.g. REGISTERD, OUT-OF-SERVICE).
BD	The number of bindings associated with the AOR.
SERVER	The server that the AOR is registered to. If the AOR is registered to the local box, the output displays “Eclipse.” Otherwise, it displays the name of the enterprise server that handled the AOR.
HITS	The number of times the AOR was accessed to forward a call, via the dial- or registration-plan (lookups on the AOR).
CL	The number of calls this AOR has participated in: either originated or received.

show location-database

Displays the location database, the shared location service across a cluster. This database is used to maintain synchronization of boxes.

The database stores (and therefore this command shows) all learned location bindings; static records are not maintained in the location database as they are managed by configurations.

Sample Output

```
NNOS-E> show location-database
AOR                          BOX    STATE        BD    SERVER    HITS    CL
---                          ---    -----        --    ------    -
sip:2125551111@vfn.com        1     registered   0     6         0
sip:2125552222@vfn.com        1     registered   0     6         0
```

Properties

The following table shows the properties for the show location-database command.

Table 5–60 Show Location-Database Properties

Field	Description
AOR	The learned entry in the location database. Typically, this is the URI associated with the SIP user.
BOX	The number of the box that the learned AOR was registered on (learned from). The output displays “1” for the local box, either standalone or cluster.

Table 5–60 (Cont.) Show Location-Database Properties

Field	Description
STATE	The state of the AOR in the database. This command only displays the final state for the entry (e.g., REGISTERD, OUT-OF-SERVICE), not any intermediate states.
BD	The number of bindings associated with the AOR.
SERVER	The server that the AOR is registered to. If the AOR is registered to the local box, the output displays "Eclipse." Otherwise, it displays the name of the enterprise server that handled the AOR.
HITS	The number of times the AOR was accessed to forward a call, via the dial- or registration-plan (lookups on the AOR).
CL	The number of calls this AOR has participated in: either originated or received.

show location-database-bindings

Displays registration status and location information for each binding of each AOR in the location database. This command provides information on how to contact an endpoint. The database stores location information for all boxes across a cluster. Use the **show location-bindings** command to see bindings for the local box.

Sample Output

```
NNOS-E> show location-database-bindings
AOR                              STATE      HOST             PORT  TPT EXP
-----      ----             ----  --- ---
sip:2125551111@vfn.com           requested  0.0.0.0          2057  UDP 60
sip:2125552222@vfn.com           requested  0.0.0.0          2057  UDP 60
```

Properties

The following table shows the properties for the show location-database-bindings command.

Table 5–61 Show Location-Database-Bindings Properties

Field	Description
AOR	The entry in the location database. Typically, this is the URI associated with the SIP user.
STATE	The state of the AOR.
HOST	The location the ME should use to reach this AOR. Displays as a host name or IP address.
PORT	The contact port on the ME used to reach this AOR.
TPT	The protocol used with this AOR. Note that if the AOR display name begins with "sips," the protocol must be TLS.
EXP	The number of seconds, as reported in the 200 OK, until the binding expires if it is not renewed.

show log-targets

Displays logging statistics (messages, bytes, and errors) for each configured log target. Use the **services** object to enable and disable logging for each service.

Sample Output

```
NNOS-E> show log-targets

name                        messages   bytes              errors
----                        --------   -----              ------
file kernel                 60         5913               0
file messages               72959      9853391            0
local-database              72982      6570117            0
syslog 192.168.215.1        73         8213               0
```

Properties

The following table shows the properties for the show log-targets. command.

Table 5–62 Show Log-Targets Properties

Field	Description
name	The name of the log target.
messages	The total number of messages logged.
bytes	The total number of bytes logged.
errors	The number of errors that were reported for that log target.

show login-sessions

Displays all active login sessions, the type of connection, and the associated user name and permissions.

Sample Output

```
NNOS-E> show login-sessions

started                 type        username            permissions
-------                 ----        --------            -------
08:18:49 Wed 2006-12-20 console     guest               guest
08:21:46 Wed 2006-12-20 ssh         guest               guest
11:07:14 Wed 2006-12-20 web         guest               guest
```

Properties

The following table shows the properties for the show login-session command.

Table 5–63 Show Login-Sessions

Field	Description
started	The time that the login session was initiated.

Table 5–63 (Cont.) Show Login-Sessions

Field	Description
type	The type of connection to the ME, either: console: Serial console client ssh: SSH client telnet: Telnet client web: ME Management client web-service: Web services client monitor: Monitor console client
username	The name of the locally configured user logged in to the system. This user was created with the **users** object.
permissions	The named permissions profile associated with the user. This profile was created with the **permissions** object.

show master-services

Displays each master service and its current configuration. The output includes status of each service regarding mastership and any associated hosts. Master services are configured using the **master-services** object

Sample Output

```
NNOS-E> show master-services
name           hosted position waiting group host            host-position
----           ------ -------- ------- ----- ----            ----
accounting     false  2        false   0     192.168.0.2     1
authentication false  2        false   0     192.168.0.2     1
call-failover  false  0        false   0     0.0.0.0         0
cluster-master true   2        false   2     0.0.0.0         2
database       false  2        false   0     192.168.0.2     1
directory      false  2        false   0     192.168.0.2     1
registration   false  2        false   0     192.168.0.2     1
server-load    false  2        false   0     192.168.0.2     1
```

Properties

The following table shows the properties for the show master-services command.

Table 5–64 Show Master-Services Properties

Fields	Description
name	The name of the master service.
hosted	Whether the service is hosted on this box. True indicates that the service is currently hosted on this box, false that it is not hosted on this box (or not configured). The **host-box** property within each master-services object defines the primary box for that service. If you configured backup boxes, the master service would be hosted on the backup in the event of primary failure.

Table 5–64 (Cont.) Show Master-Services Properties

Fields	Description
position	The position this box is in for the hosting responsibility of the master service. A value of 0 indicates that the box is not in the list of eligible boxes (or the service is not configured). If the position for a hosted service is not in position 1, it indicates that the service has failed over to a backup box. The positions of the boxes can be displayed at the command line by typing **show -v** from within the master-service object. The number in brackets next to the **host-box** property lists the position of the device in the configuration.
waiting	The state of any host take over process. A value of true indicates that the listed host is currently attempting to take over the service.
group	The VRRP group that the master service is associated with.
host	The IP address of the box currently hosting the service. A value of 0.0.0.0 indicates the local box.
host-position	The position of the host box in the list of eligible boxes. See the description of the position field for more information.

show media-ports-held

Displays any media ports being held by the ME. A port is held if the ME suspects the port or knows it to be bad. The system clears the port status as appropriate.

Sample Output

```
NNOS-E> show media-ports-held
     port: 21448
remaining: 0 days 00:00:15
```

Properties

The following table shows the properties for the show media-ports-held command.

Table 5–65 Show Media-Ports-Held

Fields	Description
port	The number of the port.
remaining	The amount of time remaining until the system stops holding the port.

show media-ports-summary

Displays a summary of media port use and configuration for each IP interface on which media ports are configured. The base-port and count are set with the IP interface **media-ports** object. You set the per-processor limits on the number of ports (and thus, the number of active calls) available for media anchoring at any one time using the object **media-anchor-limits port-limits** properties. The output also displays the port status and availability.

Sample Output

```
NNOS-E> show media-ports-summary
ip-address      base-port count busy  held  free
```

```
----------      --------- ----- ----  ----  ----
10.1.34.160     20000     5000  0     0     5000
127.0.0.1       10000     55000 8     0     54992
172.26.0.155    20000     5000  4     0     4996
```

Properties

The following table shows the properties for the show media-ports-summary command.

Table 5–66 Show Media-Ports-Summary Properties

Field	Description
ip-address	The configured IP address for the interface.
base-port	The base or starting port number to use for the port pool on the corresponding interface.
count	The total number of ports available for the media port pool on the corresponding interface.
busy	The number of ports in use within the IP interface port pool.
held	The number of ports being held within the IP interface port pool. A port is held if the ME suspects the port or knows it to be bad. The system clears the port status as appropriate.
free	The number of ports available within the IP interface port pool.

show media-scanner-interval

Displays media scanner intervals. The media-scanner monitors the signal strength and duration of the received audio to divide it into intervals.

Sample Output

```
NNOS-E> show media-scanner-interval


session-id        streamIndex  call-leg   start-time   duration  level  flags
-------------------------------------------------------------
0x4c2d14240838d8f    0            1           9         1600      -54
short-pause
0x4c2d14240838d8f    0            1           1629      200       -18   short-talk
0x4c2d14240838d8f    0            1           1829      100       -40
short-pause
0x4c2d14240838d8f    0            1           1929      200       -20   short-talk
0x4c2d14240838d8f    0            1           2129      100       -46
short-pause
0x4c2d14240838d8f    0            1           2229      600       -26   short-talk
0x4c2d14240838d8f    0            1           2829      2000      -44   long-pause
0x4c2d14249b10e50    0            1           12        2000      -52   long-pause
0x4c2d1424de7598e    0            1           12        1400      -52
short-pause
0x4c2d1424de7598e    0            1           1432      500       -32   short-talk
0x4c2d1424de7598e    0            1           1932      100       -44
short-pause
0x4c2d1424de7598e    0            1           2032      200       -24
short-talk
```

Properties

The following table shows the properties for the show media-scanner-interval command.

Table 5–67 Show Media-Scanner-Interval Properties

Field	Description
session-id	The unique ID of the media scanner session.
streamIndex	An index number indicating a particular stream of media for the session. For example, a normal phone call has one audio stream. So stream is one. And a video call has one audio stream (stream=1) and one video stream (stream=2).
call-leg	An index for a call leg indicating the direction of the call.
total-duration	The duration of the call.
current-flags	The current interval a call is in. The following are possible interval values: short-pause: Small gaps between spoken words short-talk: Short talk spurts long-talk: Long, uninterrupted speech long-pause: Signal to the ME that the media scanning is complete stable-tone: Signal to the ME that the media scanning is complete
total-flags	The total number of intervals that have occurred.

show media-scanner-summary

Displays media scanner settings. The media-scanner monitors the signal strength and duration of the received audio to divide it into intervals.

Sample Output

```
NNOS-E> show media-scanner-summary
session-id       streamIndex  call-leg  total-duration  current-flags
total-flags
-------------------------------------------------------------
0x4c2d14240838d8f    0           1          4800          long-pause
short-pause+short-tal
0x4c2d14249b10e50    0           1          2000          long-paus
x4c2d1424de7598e    0           1          5800          long-pause
short-pause+short-talk
```

Properties

The following table shows the properties for the show media-scanner-summary command.

Table 5–68 Show Media-Scanner-Summary Properties

Field	Description
session-id	The unique ID of the media scanner session.

Table 5–68 (Cont.) Show Media-Scanner-Summary Properties

Field	Description
streamIndex	An index number indicating a particular stream of media for the session. For example, a normal phone call has one audio stream. So stream is one. And a video call has one audio stream (stream=1) and one video stream (stream=2).
call-leg	An index for a call leg indicating the direction of the call.
start-time	The time when the call started.
duration	The duration of the call.
level	The measured dbM level of the sound we received during a given time interval.
flags	The intervals that have occurred.

show media-stream-addresses

Displays the session ID, IP address, and port for each leg of a call in a media stream. The output also displays details on each segment of the call leg.

The rows of the display are ordered from top to bottom, approximating the stages a media packets flows through. For basic calls, the rows marked with call-leg=1 describe the flow of media packets from the calling phone towards the answering phone (the forward path). The rows marked with call-leg=2 describe the flow of media from the answering phone towards the calling phone (the reverse path).

The following are some troubleshooting tips:

- Command output displaying all zeroes in the **address** fields:
- Check that media anchoring is not disabled for the session. Verify that all **media** objects that apply (under policy, dial-plan, and default-session-config) have the anchoring property set to **enable**.
- One or more **address** entries for **type peer-source** displays zero:
- This indicates that the ME did not receive RTP packets on the problematic call-leg. Routing or other network issues may be preventing the phone from reaching the ME. Or, if the phone is behind a NAT, verify that the **symmetricRTP** property, in the **nat-traversal** object, is set to **true**.
- Output appears fine.
- If neither **peer-source** lines are zero, the ME is anchoring media in both directions. In this case, the packets sent by the ME towards one or both of the phones are not being received by the respective phones. Routing or other network issues may be the problem.

Sample Output

```
NNOS-E> show media-stream-addresses
session-id         stream  call-leg  type          origin      address
----------         ------  --------  ----          ------      --
0x4c2702a100005c6  1       1         peer-source   rtp         172.30.0.18:7350
                                     anchor-dest   media-port  172.26.0.15:22264
                                     anchor-source media-port  172.26.0.15:22194
                                     peer-dest     sdp         172.30.0.20:37520
                           2         peer-source   rtp         172.30.0.20:37520
```

```
anchor-dest   media-port  172.26.0.15:22194
anchor-source media-port  172.26.0.15:22264
peer-dest     sdp         172.30.0.18:7350
```

Properties

The following table shows the properties for the show media-stream-addresses command.

Table 5–69 Show Media-Stream-Addresses

Field	Description
session-id	The unique internal identifier assigned to the session.
stream	The index number of the media-stream. Voice calls have just one stream, with index equal to 1. Video calls have two streams: the audio stream is typically index equal to 1 and the video stream is typically index equal to 2.
call-leg	The index number of the call leg. For basic calls call-leg 1 describes the media flowing in the direction from the calling party (the SIP From: URI) towards the answering party (the SIP To: URI). Call-leg 2 describes the reverse direction: from the answering party towards the calling party.
type	The role the address plays in the call. The **type** column can have the following values: **peer-source**: The address of an RTP peer (either calling or answering party) that is the source of RTP packets received by the system. This type of address is always learned from the IP and UDP headers of received RTP packets and has an **origin** value (see below) of **rtp**. **peer-dest**: The address of an RTP peer (either calling or answering party) that is the destination of RTP packets sent by the system. This type of address is learned either from SDP (**origin** value of **sdp**) or from symmetric RTP (**origin** value of **symmetric-rtp**). **anchor-dest**: The system address that receives packets from an RTP peer. This type of address has an **origin** value of **media-ports** or **near-end-nat**. anchor-source: The system address that sends packets to an RTP peer. This type of address has an **origin** value of **media-ports**.
origin	The protocol source of the address (how the address was determined). The **origin** column can have the following values: **sdp** (Session Description Protocol): These addresses are learned from the "c=" lines in received SDP message bodies. **media-port**: These addresses are allocated from media-ports configured under the IP interface. **rtp** (Real Time Protocol): these addresses are learned from the IP/UDP headers of received RTP packets. **symmetric-rtp**: These addresses are learned from the IP/UDP headers of received RTP packets from the reverse media direction. **near-end-nat**: These addresses are determined by the near-side-nat configuration under the IP interface.
address	The IP address and UDP port related to media anchoring on the session.

show media-stream-client-sessions

Displays each signaling session that has media resources allocated on a media-proxy (a media stream server). The output is primarily used for debugging and troubleshooting.

Sample Output

```
NNOS-E> show media-stream-client-sessions

client-session-id server-id server-session-id
----------------- --------- -----------------
0x4c105504fc789e0 0.0.0.0   0x4c105504fc94123
```

Properties

The following table shows the properties for the show media-stream-client-sessions command.

Table 5–70 Show Media-Stream-Client-Sessions Properties

Field	Description
client-session-id	The unique internal ID of the media stream client session.
server-id	The IP address of the media stream server.
server-session-id	The ID of the media stream server session.

show media-stream-counts

Displays how many voice sessions are up on the ME at a given time.

Sample Output

```
NNOS-E> show media-stream-counts

client-id   server-id   sessions
---------   ---------   --------
0.0.0.0     0.0.0.0     2
```

Properties

The following table shows the properties for the show media-stream-counts command.

Table 5–71 Show Media-Stream-Counts Properties

Fields	Description
client-id	The IP address of the client.
server-id	The IP address of the media stream server.
sessions	The number of sessions involved in the call. A call can be made up of multiple sessions. For example, in a B2B configuration, the system receives the call, terminates the session, and then creates a new session and sends it on to the recipient. In this case, there are multiple sessions associated with the call.

show media-stream-server-sessions

Displays each media stream server session created for a signaling session on a signaling node (a media stream client). The output is primarily used for debugging and troubleshooting.

Sample Output

```
NNOS-E> show media-stream-server-sessions

server-session-id client-id client-session-id
----------------- --------- -----------------
0x4c1055063573eb0 0.0.0.0   0x4c105506355a06d
```

Properties

The following table shows the properties for the show media-stream-server-sessions command.

Table 5–72 Show Media-Stream-Server-Sessions Properties

Field	Description
server-session-id	The unique internal ID of the media stream server session.
client-id	The IP address of the media stream client (the signaling node).
client-session-id	The ID of the media stream client session.

show media-stream-stats

Displays a count of transmit and receive packets for each call leg in a media stream.

Sample Output

```
NNOS-E> show media-stream-stats

session-id        stream call-leg address               receive-packets
transmit-packets
----------        ------ -------- -------               ----------
0x4c105506355a06d 1      1        192.168.215.103:22566 21407           21437
                         2        192.168.215.103:22394 21437           21407
```

Properties

The following table shows the properties for the show media-stream-stats command.

Table 5–73 Show Media-Stream-Stats Properties

Field	Description
session-id	The ID of the media stream session.
stream	An index number indicating a particular stream of media for the session. For example, a normal phone call has one audio stream. So stream is one. And a video call has one audio stream (stream=1) and one video stream (stream=2).

Table 5–73 (Cont.) Show Media-Stream-Stats Properties

Field	Description
call-leg	An index for a call leg indicating the direction of the call.
address	The IP address and media port of the media stream server where it receives and transmits packets.
receive-packets	The number of packets the media stream server has received.
transmit-packets	The number of packets the media stream server has transmitted.

show memory-failures

Indicates whether any memory allocation failures have occurred on the box. If the output indicates a failure, you can troubleshoot possible reasons the box is out of memory (e.g., a configuration problem, a memory leak, etc.).

Sample Output

```
NNOS-E> show memory-failures
Memory allocation failures:
-----------------------------------------------------------------
Process  Address  Failures  OldestFail  NewestFail  Smallest   Largest
-----------------------------------------------------------
manager  08271b60     1     00:00:03     00:00:03   1048612  1048612
manager  08272834     1     00:00:03     00:00:03     65536    65536
manager  08272d0b     1     00:00:03     00:00:03        32       32
manager  08272efb     1     00:00:03     00:00:03        32       32
manager  08277353     1     00:00:03     00:00:03      1024     1024
manager  082774dc     1     00:00:03     00:00:03         1        1
manager  082760fb     1     00:00:03     00:00:03       256      256
manager  08276b1f     1     00:00:03     00:00:03       699      699
manager  08276b9f     1     00:00:03     00:00:03       699      699
-------------------------------------------------------------
```

Properties

The following table shows the properties for the show memory-failures command.

Table 5–74 Show Memory-Failures Properties

Field	Description
Process	The system process that generated the memory failure.
Address	The address of the failure.
Failures	The number of failures at that address.
Oldest Fail	The amount of time that has passed since the first failure at this address occurred. Use the verbose form of the command to see a timestamp for the failure.

Table 5–74 (Cont.) Show Memory-Failures Properties

Field	Description
Newest Fail	The amount of time that has passed since the most recent failure at this address occurred. Use the verbose form of the command to see a timestamp for the failure.
Smallest	The smallest memory allocation size that failed at that address.
Largest	The largest memory allocation size that failed at that address.

show msrp-connections

Displays statistics regarding all of the connections used by the current MSRP sessions.

Sample Output

```
SIP>show msrp-connections
-------------------------------------------------------------------------------
Process Proto LocalAddress RemoteAddress State Direction RefCount
-------------------------------------------------------------------------------
SIP TCP 10.138.236.35:23365 10.138.238.49:53847 Connected Answer 1
SIP WS 10.138.236.35:23385 10.138.238.49:53848 Connected Originate 1
```

Properties

The following table shows properties for the **show msrp-connections** command.

Table 5–75 Show MSRP-Connections Properties

Fields	Description
Process	The signaling process being used for this connection.
Proto	The media transport protocol being used for this connection.
LocalAddress	The local IP address and port.
RemoteAddress	The remote IP address and port.
State	The state of the connection.
Direction	The current direction of media transfer.
RefCount	Not currently supported. This value should always be 1.

show msrp-listeners

Displays information listing all ports on the ME interface that are waiting for MSRP connections.

Sample Output

```
SIP>show msrp-listeners
-------------------------------------------------------------------------
Process Proto Address Connections Rejected Current Timeouts
-------------------------------------------------------------------------
SIP WS 10.138.236.35:23385 0 0 1 0
```

Properties

Table 5–76 Show MSRP-Listeners Properties

Fields	Description
Process	The signaling process being used for this port.
Proto	The media transport protocol being used for this port.
Address	The IP address for this port.
Connections	The number of connections available on this port.
Rejected	The number of connections rejected by this port.
Current	The number of current connections on this port.
Timeouts	The number of timeouts that have occurred on this port.

show msrp-stats

Displays information regarding MSRP interworking statistics.

Sample Output

```
SIP>show msrp-stats
totalSessions: 4
totalConnections: 2
totalActiveConnections: 1
totalPassiveConnections: 1
RxRequests: 4
RxResponses: 4
TxRequests: 4
TxResponses: 4
RxMessagesDiscarded: 0
RxMessagesPartialRead: 0
RxMessagesFailed: 0
TxMessageRetries: 0
TxTcpWriteErrors: 0
TxMessagesFailed: 0
ListenerErrors: 0
SessionEstTimeouts: 0
UserMsgsExpired: 0
```

Properties

Table 5–77 Show MSRP-Stats Properties

Field	Description
totalSessions	The total number of MSRP sessions since the system was last started.
totalConnections	The total number of connections since the system was last started.
totalActiveConnections	The total number of connections created by the ME.
totalPassiveConnections	The total number of connections initiated by MSRP.
RxRequests	The total number of MSRP request messages received by the ME.

Table 5–77 (Cont.) Show MSRP-Stats Properties

Field	Description
RxResponses	The total number of MSRP response messages received by the ME.
TxRequests	The total number of MSRP request messages forwarded by the ME.
TxResponses	The total number of MSRP response messages forwarded by the ME.
RxMessagesDiscarded	The total number of MSRP messages discarded by the ME regardless of reason.
RxMessagesPartialRead	The total number of partial MSRP messages read. If this value is anything but zero, the ME is using partial-forwarding.
RxMessagesFailed	The total number of MSRP messages the ME has been unable to be read.
TxMessageRetries	The total number of attempts to forward MSRP messages (usually due to slow connection establishment).
TxTcpWriteErrors	The total number of times the ME encountered an error while attempting to forward an MSRP message.
TxMessagesFailed	The total number of MSRP messages not forwarded by the ME due to an error condition.
ListenerErrors	The total number of MSRP listener-related errors.
SessionEstTimeouts	The total number of times an MSRP session failed to be established.
UserMsgsExpired	Not currently supported.

show multimedia-streaming-pool

Provides information about the configured multimedia streaming pool that has been derived from sip-server-pool.

Sample Output

```
NNOS-E>show multimedia-streaming-pool

peer-name: internal1
   server: internal1
     host: 156.40.1.11
      TPT: any
     port: 1935
      box: local
    state: up
       in: 0
      out: 0
```

The following table shows the properties for the show multimedia-streaming-pool command.

Table 5–78 Show Multimedia-Streaming-Pool

Field	Description
peer-name	The name of this pool's peer.
server	The server associated with this pool.

Table 5–78 (Cont.) Show Multimedia-Streaming-Pool

Field	Description
host	The IP address of this server.
TPT	The preferred transport method of this server. Currently, this is always set to any.
port	The port over which this server is listening.
box	The ME where this server is configured. This value is currently always the local ME.
state	Whether the server is available (up) or not (down).
in	The number of packets received from this server. This value is currently not applicable.
our	The number of packets sent to this server. This value is currently not applicable.

show multimedia-streaming-server

Provides information about the multimedia streaming servers configured on the ME.

Sample Output

```
NNOS-E>show multimedia-streaming-server

name       protocol  host          port  hits
----       --------  ----          ----  ----
internal1  RTMP      156.40.1.11   1935  0
```

The following table shows the properties for the show multimedia-streaming-server command.

Table 5–79 Show Multimedia-Streaming-Server Properties

Field	Description
name	The name of the multimedia streaming server.
protocol	The protocol over which this port is listening.
host	The IP address of this server.
port	The port over which this server is listening.
hits	The number of requests currently sent to this server.

6

Status Provider Show Commands N Through Z

This chapter describes the status show commands from N through Z alphabetically. For status show commands from A through M, see "Status Provider Show Commands A Through M."

Global Show Command Characteristics

All of the status show commands contained in this chapter share the same MIB and require the same user access level. In addition, they are available through a single place in the GUI.

Associated MIB

All status display commands reference the Media Engine (ME), Inc. enterprise MIB:

```
CXC.MIB
```

ME Management System Path Information

All status display commands are available in the ME Management System.

You can access the commands from the **Status** tab at the top of the window or from the **Status** link on the Home page. Expand the list on the left and click on a status report listed below it.

Filtering Command Output

The CLI allows you to filter output of show commands so that your display only includes the specific properties you requested. With no properties, **show** *object-name* displays all instances of the specified object. For example, if you execute the **actions** command, the CLI displays a list of all actions that have occurred in the current CLI session:

```
NNOS-E> show actions

action               process timeout    requests   errors     timeouts
------               ------- -------    --------   ------     --
archive                acct    10000      0          0          0
arp-delete             manager 10000      0          0          0
clock                  manager 10000      0          0          0
config merge           manager 30000      0          0          0
config replace         manager 30000      0          0          0
config save            manager 30000      2          0          0
database               manager 300000     0          0          0
database-maintenance   manager 10000      0          0          0
```

```
diameter                auth    10000      0          0          0
```

The output includes indices to the action (e.g., arp-delete, clock, config merge, config save, etc.) and properties of the indices (in this case, process, requests, errors, timeouts).

However, if you are only interested in seeing a specific index or property, you can filter on those fields. You can specify an index name to display only the instances with those values. Or, you can specify one or more property values to display only the instances with those property values.

Note: Index and property names are case insensitive.

To display a list of the properties you can filter on, enter the command with a question mark:

```
NNOS-E> show actions ?
 action provider statistics
 action
 process
 timeout
 requests
 errors
 timeouts
 -c       display the total number of instances
 -n       display a specified number of instances
 -v       verbose display
```

Filtering On an Index

To filter on the index, enter the object name with the command (in quotation marks if the name includes white space). For example, you can display only the number of saves to the configuration file that have occurred:

```
NNOS-E> show actions "config save"
action             process timeout    requests   errors     timeouts
------             ------- -------    --------   ------     -----
config save        manager 30000      2          0          0
```

To filter on a property, enter the property name followed by an equal sign (or let the system enter the correct format with a TAB complete). To see only directory processes:

```
NNOS-E> show actions process=dir
action             process timeout    requests   errors     timeouts
------             ------- -------    --------   ------     -----
directory-reset        dir     120000     0          0          0
```

Note that you can enter multiple properties to further refine your output. You cannot, however, enter multiple instances of the same property. In that case, the last property entered is acted on:

```
NNOS-E> show actions process=SIP process=auth
action             process timeout    requests   errors     timeouts
------             ------- -------    --------   ------     -----
diameter           auth    10000      0          0          0
radius             auth    10000      0          0          0
```

Displaying Total, Count, and Verbose Reports

You can display summary reports on a status using one of the options defined in the following table. The following table displays examples of show command report types.

Table 6–1 Show Command Report Types

Option	Description
-c	Displays a count of the total number of entries in a status report. Enter in the form: show *object-name* -c
-n	Displays the specified number of entries from the status report, counting from the first entry. Enter in the form: show *object-name* -n=*x* **-n=x** displays the first *x* instances.
-v	Displays a more detailed report of the object. This option does not change all output, only for those reports where summary and detailed reports are both available. Enter in the form: show *object-name* -v

Examples of -c, -n, and -v use

Without using display options, the output of **show dial-plan** looks like this:

```
NNOS-E> show dial-plan
plan-name     type     destination-url    from     peer-name   fwd
---------     ----     ---------------    ----     ---------   --
abc.com       tag      aster              .*       abc         0
company.net   tag      bw                 .*       company     0
xyz.com       tag      aaa                .*       xyz         0
123.com       domain   sip:.*@123\.com    .*       123         0
server.com    domain   sip:.*@server\.com .*       server      0
Using the count option, the output of show dial-plan -c looks like this:
NNOS-E> show dial-plan -c
dial-plan returned 5 instances
```

Specifying that the system display the first two entries using the number of entries option, the output of **show dial-plan -n=2** looks like this:

```
NNOS-E> show dial-plan -n=2
plan-name     type     destination-url    from     peer-name   fwd
---------     ----     ---------------    ----     ---------   --
abc.com       tag      aster              .*       abc         0
company.net   tag      bw                 .*       company     0
```

Using the verbose option, the output of **show dial-plan -v** looks like this:

```
NNOS-E> show dial-plan -v

                          plan-name: abc.com
                               type: tag
                                url: aster
```

```
                        destination-url: aster
                                 source: plan
                                  level: 0
                                   from: .*
                              peer-name: abc
                          peer-identity: sip:sametime@abc.com
                              peer-mode: provider
                                 action: redirect
                                    fwd: 0
                                   hits: 0
            incoming-host-normalizations:
                         in-request-user:
                    in-request-user-template:
                                in-to-user:
                           in-to-user-template:
                                  in-from-user:
                      in-from-user-template:
            outgoing-host-normalizations:
--More--
```

Note: All show commands include the -c, -n, and -v options (although in some commands the options do not change the output). Because they are universal, these options are not included in the command description syntax statement in this chapter.

show named-variables-by-session

This action displays configured named-variables for each active session.

Sample Output

```
NNOS-E>show named-variables-by-session

------------------------------------------------
Named Variables for session 0x04c3e0841669a1b7
------------------------------------------------
variable1 = value1
variable2 = value1
variable3 = value3
variable4 = value4
variable5 = value3
------------------------------------------------
```

When you add a **-v** to the end of this action, the ME displays a verbose output which also shows the association ID of the session, the session type, creation time, and the value and source for each variable.

show netfilter

Displays the state of the Linux firewall, which is effected when a service (e.g., SSH, Telnet) is configured. The output displays entries through the firewall that the ME created to allow service traffic to be received. Netfilter rules are automatically generated by the services as they are configured on an interface.

Sample Output

```
NNOS-E> show netfilter
```

```
index  packets  policy  intf   proto source     destination        match
-----  -------  ------  ----   ----- ------     -----------
1      244161   permit         tcp   0.0.0.0:0  0.0.0.0:0          state:EST,REL
2      20846    permit         udp   0.0.0.0:0  0.0.0.0:0          state:EST,REL
3      437      permit         all   0.0.0.0:0  127.0.0.1:0
4      0        permit  eth0   tcp   0.0.0.0:0  172.26.0.153:23    flags:SYN
state:NEW
5      1        permit  eth0   tcp   0.0.0.0:0  172.26.0.153:22    flags:SYN
state:NEW
6      0        permit  eth0   udp   0.0.0.0:0  172.26.0.153:161   state:NEW
7      0        permit  eth0   udp   0.0.0.0:0  172.26.0.153:123   state:NEW
8      2077     permit  eth0   udp   0.0.0.0:0  255.255.255.255:67 state:NEW
--more--
```

Properties

The following table shows the properties for the show netfilter command.

Table 6–2 Show Netfilter Properties

Field	Description
index	The system-generated identifier for the netfilter rule. Each rule has a unique index, and index numbers are assigned in the order in which the services were configured on the system. Rules are processed in the order of the index numbers.
packets	The number of packets effected by the rule.
policy	The action that matching the rule results in, either permit or deny. The final rule entry is always to deny all. (If a packet did not match any other rule that determines its outcome, the system denies the packet.) All rules automatically created by the system in response to a service configuration are given a policy of **permit**.
intf	The interface on which the rule applies.
proto	The protocol used with the service that the rule represents. The protocol is determined by the service configuration.
source	The source address and port of the packet to match against. An entry of 0.0.0.0:0 represents a wildcard (match any).
destination	The destination address and port of the packet to match against. An entry of 0.0.0.0:0 represents a wildcard (match any).
match	The string that describes what the rule is matching against.

show network-settings

Displays the current system network settings for TCP properties. These properties are set with the **network** object.

Sample Output

```
NNOS-E> show network-settings
-----------------------------------------------------------------
```

```
Configuration Item        Current Value
------------------------------------------------------------------
tcp-keepalive-time        600 seconds
tcp-keepalive-probes      5
tcp-keepalive-interval    6 seconds
tcp-max-syn-backlog       1024
tcp-synack-retries        5
tcp-syncookies            enabled
tcp-fin-timeout           60 seconds
```

Properties

The following table shows the properties for the show network-settings command.

Table 6–3 Show Network-Settings Properties

Fields	Description
tcp-keepalive-time	The time, in seconds, that an established TCP connection can remain idle before the system sends a keepalive to the client. The idle time expiration initiates the keepalive process.
tcp-keepalive-probes	The number of unanswered TCP keepalive probes that are allowed before the system disconnects an idle session.
tcp-keepalive-interval	The time, in seconds, that the system waits for a response from a keepalive probe before ending the next one. The system continues to send probes until it has sent the number specified in the **tcp-keepalive-probes** property.
tcp-max-syn-backlog	The maximum number of queued (unacknowledged) connection requests allowed before the system begins dropping requests. This value is set to help prevent a TCP SYN flood attack.
tcp-synack-retries	The number of times the system retransmits a SYN-ACK in response to a SYN. If the number of retries is reached without a successful response, the system deletes the new connection from the table.
tcp-syncookies	Specifies whether SYN cookie support in the kernel is enabled or disabled. See the tcp-syncookies property in the **network** object for more information.
tcp-fin-timeout	The number of seconds the system waits for a final FIN packet before forcibly closing the socket. The system uses the FIN packet to disconnect a TCP connection, whether it's idle or not.

show ntp

Displays statistics relating to the Network Timing Protocol and the ME. You can configure the ME as both an **ntp-client** and an **ntp-server**.

Sample Output

```
NNOS-E> show ntp
requests: 174
responses: 173
discards: 0
adjustments: 71
last-adjustment: 10:38:27 Wed 2007-04-11
maximum-adjustment: 2180 milliseconds
average-adjustment: 733 milliseconds
server-requests: 411
```

```
server-responses: 411
server-discards: 0
```

Properties

The following table shows the properties for the show ntp command.

Table 6–4 Shot NTP Properties

Field	Description
requests	The number of NTP requests the system made when acting as an NTP client.
responses	The number of responses the system NTP client received.
discards	The number of responses the system NTP client received but did not accept.
adjustments	The number of changes the system made to its internal time based on an NTP response.
last-adjustment	The time and date of the last adjustment.
maximum-adjustment	The largest adjustment to the box time the system made based on an NTP response.
average-adjustment	The average adjustment time to the box the system made based on all NTP responses.
server-requests	The number of NTP requests the system received when acting as an NTP server.
server-responses	The number of responses the system NTP server sent out.
server-discards	The number of requests that the system NTP server discarded.

show policies

Displays, for each active policy, the number of rules it contains and the number of the ME elements that use that policy. You can configure policy from a variety of places. See Configuring Session Configuration Objects for more information.

Sample Output

```
NNOS-E> show policies

           name: default
number-of-rules: 10
     inclusions: 0

           name: to
number-of-rules: 0
     inclusions: 0

           name: from
number-of-rules: 0
     inclusions: 0
```

Properties

The following table shows the properties for the show policies command.

Table 6–5 Show Policies Properties

Field	Description
name	The name of the policy.
number-of-rules	The number of rules that comprise the policy.
inclusions	The number of times the policy was triggered (from the piece of the configuration in which it is referenced). For example, you could reference a policy in the vsp\enterprise\unknown-server-policy object. This value would indicate the number of times there was a user in an incoming message that was defined in a configured directory. Otherwise, if there is no such user, the policy is not used and the inclusions indicate zero.

show processes

Displays status for each process that is part of the ME operations. If you have debug permissions, you can use the top **(NNOS-E>)** command to display all processes running on the box.

Sample Output

```
NNOS-E> show processes
process   id        condition  run-level  starts  uptime           fds
-------   --        ---------  ---------  ------  ------
monitor   6538      running    7           1      0 days 01:18:11  26
manager   6686      running    7           1      0 days 01:18:11  58
SIP       6763      running    7           1      0 days 01:18:09  66
media     6764      running    7           1      0 days 01:18:09  26
auth      0         idle       init        0      0 days 00:00:00  0
reg       6765      running    7           1      0 days 01:18:09  26
dir       0         idle       init        0      0 days 00:00:00  0
web       6767      running    7           1      0 days 01:18:09  13
WS        6768      running    7           1      0 days 01:18:09  15
acct      0         idle       init        0      0 days 00:00:00  0
dos       0         idle       init        0      0 days 00:00:00  0
SSH       6769      running    none        1      0 days 01:18:09  6
```

Properties

The following table shows the properties for the show processes command.

Table 6–6 Show Processes Properties

Field	Description
process	The name of the system-specific process.
id	An internal ID number, assigned at startup. This ID is analogous to the Linux process ID.

Table 6–6 (Cont.) Show Processes Properties

Field	Description
condition	Possible process conditions are: **idle**: The process did not start or is administratively disabled. Check the master-services object to determine if the process is configured. **running**: The process is running. **dead**: the process cannot be restarted. To determine the reason for failure, use the **show faults** command and check the event logs. **disabled**: The process will not be started. Check to see whether it is unavailable due to licensing restrictions.
run-level	The process run level, which indicates its state. A process starts off at **init**, and then proceeds from 0 up through 7, at which point it is fully operational. If the process displays **none**, it does not participate in the run level mechanism.
starts	The number of times that the process started (or failed and restarted). For any value other than 0 or 1, use the **show faults** command and check the event logs for information relating to that process. This counter clears on system restart.
uptime	The length of time the process has been running. The timer restarts when the process boots.
fds	The file descriptor for the process. There is a file descriptor for each open file and socket.

show radius-auth

Display configuration information, status, and count and speed statistics for each configured **radius-group** object. The ME resets all statistics reboot.

Sample Output

```
NNOS-E> show radius-auth

Status for RADIUS group '172.26.0.147' (round-robin):
-------------------------------------------------------------------
Server Name        State     | Out  Pending  Requests  Accepts  Rejects  Errors
-------------------------------------------------------------------
172.26.0.147       Healthy   |   0        0        19        3        0      16
-------------------------------------------------------------------
Totals:                      |   0        0        19

Status for RADIUS group 'Boston' (fail-over):
-------------------------------------------------------------------
Server Name        State     | Out  Pending  Requests  Accepts  Rejects  Errors
-------------------------------------------------------------------
boston             Idle      |   0        0         0        0        0       0
127.0.0.1          Idle      |   0        0         0        0        0       0
-------------------------------------------------------------------
Totals:                      |   0        0         0        0        0       0
```

Properties

The following table shows the properties for the show radius-auth command.

Table 6–7 Show Radius-Auth Properties

Field	Description
Status for RADIUS group...	The name of the group reported on as well as the RADIUS group authentication operational algorithm.
Server Name	The name or IP address that identifies the server that is part of this RADIUS group.
State	The state of the RADIUS server, either: **Idle**: Server is enabled, but has not received traffic. **Disabled**: The server is disabled in the configuration. **Healthy**: The server is responding normally to system requests. **Failing**: The server has not responded to some system requests, but not enough to trigger a fail-over (if configured). **Failed**: The server has failed to respond to too many requests and the system has determined that it is down. If the RADIUS group is configured with fail-over mode, and a backup server is configured, then the system stops sending requests to this server and begins forwarding requests to the next. **BadSecret**: There is an error with the shared secret configured for this server is incorrect.
Out	The number of outstanding requests; the requests that the system has sent to the RADIUS server without a response.
Pending	The number of requests that have been generated but have not yet been sent to the server. The server's **window** setting defines the number of allowed requests. Requests generated once that threshold has been reached are counted as pending.
Requests	The total number of authentication requests generated.
Accepts	The number of times the RADIUS server has accepted a request, indicating that the user has the correct password.
Rejects	The number of times the RADIUS server has rejected a request, indicating that either the user has an incorrect password or the shared secret isn't right.
Errors	The number of request errors.

show registration-admission-control

Displays settings and statistics for registration admission control on this VSP (REGISTER requests). See the **admission-control** object for more complete descriptions of all configurable settings.

Sample Output

```
NNOS-E> show registration-admission-control
                                      name: default
          registration-admission-control: enabled
                        max-registrations: 30000
    pending-registrations-high-watermark: 500
     pending-registrations-low-watermark: 10
 pending-registrations-dynamic-threshold: 500
                         cpu-monitor-span: 20 seconds
                     cpu-monitor-interval: 10 seconds
                          average-sip-cpu: 0 %
        registrations-high-cpu-threshold: 90 %
         registrations-low-cpu-threshold: 50 %
                   total-client-bindings: 0
                registrations-in-progress: 0
          registrations-most-in-progress: 0
                   registrations-sessions: 0
              processed-new-registrations: 0
           processed-waiting-registrations: 0
       processed-challenged-registrations: 0
            processed-other-registrations: 0
  suppressed-registrations-this-interval: 0
  suppressed-registrations-last-interval: 0
             suppressed-new-registrations: 0
         suppressed-waiting-registrations: 0
      suppressed-challenged-registrations: 0
              last-register-suppressed-at:
            discarded-other-registrations: 0
               last-register-discarded-at:
             edp-transactions-in-progress: 0
```

Properties

The following table shows the properties for the show registration-admission-control command.

Table 6–8 Show Registration-Admission-Control Properties

Field	Description
name	The name of the VSP whose status is displayed.
registration-admission-control	The state of registration admission control for this VSP: whether it is enabled or disabled.
max-registrations	The setting for the maximum number of registrations allowed in the location cache. When the system reaches the maximum registration count, registrations are denied until the number falls below this threshold. See the **admission-control max-registrations** property.

Table 6–8 (Cont.) Show Registration-Admission-Control Properties

Field	Description
pending-registrations-high-watermark	The hard limit set for the number of in-progress registrations allowed before the system suppresses all registrations. See the **admission-control pending-registrations-high-watermark** property.
pending-registrations-low-watermark	Sets a hard limit for the number of in-progress registrations allowed before the system begins registration suppression. See the **admission-control pending-registrations-low-watermark** property
pending-registrations-dynamic-threshold	The current dynamic limit derived for the number of in-progress registrations allowed. The registration dynamic threshold is calculated based on the **admission-control pending-registrations-high-watermark** property.
cpu-monitor-span	The number of seconds over which the system calculates the total system CPU average. See the **admission-control cpu-monitor-span** property.
cpu-monitor-interval	The frequency, in seconds, with which the system calculates the total system CPU average for the last span. See the **admission-control cpu-monitor-interval** property
average-sip-cpu	The current average CPU usage.
registrations-high-cpu-threshold	The upper threshold, as a percentage, for registration processing average CPU usage. See the **admission-control registrations-high-cpu-threshold** property.
registrations-low-cpu-threshold	Sets the low-end threshold, as a percentage, for registration processing average CPU usage. See the **admission-control registrations-low-cpu-threshold** property.
total-client-bindings	The total number of bindings in the location cache.
registrations-in-progress	The number of registrations that the system is currently processing.
registrations-most-in-progress	The highest number of registrations that the system has had in process at any one time.
processed-new-registrations	The number of new registrations that the system successfully processed since the last system boot.
processed-waiting-registrations	The number of registrations for which the system received a REGISTER but the server was busy. As a result, the system responded locally, inserting the registration plan **route min-client-expiration** time.
processed-challenged-registrations	The number of challenge requests that the system processed since the last system boot.

Table 6–8 (Cont.) Show Registration-Admission-Control Properties

Field	Description
processed-other-registrations	The number of other types of requests that the system processed since the last system boot.
suppressed-registrations-this-interval	The number of REGISTER requests to which the system responded locally instead of delegating in the current interval. The interval is set with the **admission-control cpu-monitor-interval** property
suppressed-registrations-last-interval	The number of REGISTER requests to which the system responded locally instead of delegating in the previous interval. The interval is set with the **admission-control cpu-monitor-interval** property
suppressed-new-registrations	The number of new registrations that the system successfully processed locally (instead of delegating) since the last system boot.
suppressed-waiting-registrations	The number of registrations for which the system received a REGISTER but the server was busy. As a result, the system responded locally, inserting the registration plan **route min-client-expiration** time.
last-register-suppressed-at	The time at which the last REGISTER request was suppressed.
discarded-other-registrations	The number of registrations discarded that were not of type new, waiting, or challenge.
last-register-discarded-at	The time at which the last REGISTER request was discarded.
edp-transactions-in-progress	The number of Expiration Discovery Process (EDP) transactions that the system is currently processing. See the **admission-control pending-edp-transaction** properties for information on setting transaction thresholds; see the registration-plan **route edp** property for more information on EDP.

show registration-arbitration

Displays a summary of each configured registration-plan **arbiter** entry, its match criteria and rules (and other configuration elements), and the number of times the ME has applied the plan. The arbiter contains an ordered set of rules that configure different cost-based routing algorithms, which the ME uses to select where to forward REGISTER, SUBSCRIBE, and NOTIFY requests.

Sample Output

```
NNOS-E> show registration-arbitration

plan-name: abc
     type: default
    match: !*
      pri: 100
    rule1:
    rule2:
     hits: 0
```

Properties

The following table shows the properties for the show registration-arbitration command.

Table 6–9 Show Registration-Arbitration Properties

Field	Description
plan-name	The name of the arbiter plan entry.
type	The criteria for matching entries in the arbitration table. The matching arbiter is then applied, determining the calculation the system performs. The type field is derived from the **subscriber-match** property setting in the **arbiter** object.
match	The string to match in the USER and/or HOST fields of the FROM URI in order for the system to apply the plan configuration to requests containing the prefix. The **match** field is derived from the **subscriber-match** property specific string match setting in the **arbiter** object.
pri	The priority (order of preference) for this registration-plan **arbiter** entry. This property overrides the default behavior (most specific match) and sets a preference based on the **subscriber-match** property.
rule1	The first rule applied to determine server selection, set with the **rule** property of the **arbiter** object.
rule2	The second rule applied to determine server selection, set with the **rule** property of the **arbiter** object.
hits	The number of times this **arbiter** entry was matched on and applied.

show registration-plan

Displays a summary of each *configured* (but not necessarily active) **registration-plan** entry, its match criteria and peer (and other configuration elements), and the number of times the ME has applied it to forward a call. Use the **show registration-routing** command to see all *active* registration plans. The output is from the registration routing table, which defines how the ME proxies registrations (handles incoming requests).

Note that the command output includes an entry that is automatically generated by the system. The final entry, a **type** of **domain** and a **match** on the configured VSP domain name is the ME conversion of the registration service into a registration plan.

Sample Output

```
NNOS-E> show registration-plan
plan-name      type      match           min   pri  peer-name      action   hits
---------      ----      -----           ---   ---  ---------      ------   ----
vfn             default   !*              2     100                 accept   144
vfn.com         domain    vfn.com         7     100  fn.com         accept   12
978             phone     sip:1978.*@.*         50   abc.com        delegate 0
companyXYZ.com  domain    companyXYZ.com  14    100  companyXYZ.com accept   0
```

Properties

The following table shows the properties for the show registration command.

Table 6–10 Show Registration Properties

Field	Description
plan-name	The name of the **route** or **source-route** registration plan entry.
type	The criteria for matching entries in the routing table. The system then applies the appropriate normalization plan to matching entries. The **type** field is derived from either the **to-uri-match** (**route**) or **source-match** (**source-route**) property.
match	If contributed through a route object entry, the string to match in the USER and/or HOST fields in the SIP header in order for the system to apply the entry normalization plan to calls containing the prefix. If contributed through a source-route object entry, the match criteria for the source of the SIP message. The system then sets the next-hop server (defined with the **peer** property) for all traffic that matches this configured source. The **match** field is derived from either the **to-uri-match** (**route**) or **source-match** (**source-route**) property. If type is condition-list, the match is derived from the priority plus plan name.
min	The minimum number of digits required for a match on a phone prefix, if configured. In some cases, the system calculates a value for other types of matches based on the number of characters (including wild cards). In some cases it displays as-is. The value is only meaningful to a phone-prefix match, however.
pri	The priority (order of preference) for this registration-plan entry. This property overrides the default behavior (most specific match) and sets a preference based on the **subscriber-match** property.
peer-name	A statically entered peer referenced through the **peer** property.

Table 6–10 (Cont.) Show Registration Properties

Field	Description
action	The action that the system is configured to take when a match occurs. The following describes each of the possible actions: **accept**: Accepts the registration locally, functioning as a registrar. **delegate**: Forwards the REGISTER to an upstream SIP proxy (provider) and resets the contact to itself. **forward**: Forwards the REGISTER, unchanged, to the server specified in the header. **redirect**: Sends a response to the client with instructions to resend the REGISTER to a different server. **tunnel**: Creates an OC client-to-LCS server tunnel, via the registration plan, that you can then load balance across. **discard**: Silently discards REGISTER requests matching this registration plan or AOR. **block**: Rejects calls matching this registration plan or AOR with either a "603 Declined" message or with configured text.
hits	The number of times this registration-plan entry was matched on and applied.

show registration-routing

Displays all active entries in the registration routing table, which defines how the ME handles incoming REGISTER requests. The output displays a summary of each *active* **registration-plan** entry, its match criteria and peer (and other configuration elements), and the number of times the ME has applied the plan to forward a call. Use the **show registration-plan** command to see all *configured* registration plans.

Note that the command output includes an entry that is automatically generated by the system. The final entry, a **type** of **domain** and a **match** on the configured VSP domain name is the ME conversion of the registration service into a registration plan.

Sample Output

```
NNOS-E> show registration-routing
plan-name       type      match           min   pri  peer-name      action  hits
---------       ----      -----           ---   ---  ---------
vfn             default   !*              2     100                 accept  144
vfn.com         domain    vfn.com         7     100  vfn.com        accept  12
companyXYZ.com  domain    companyXYZ.com  14    100  companyXYZ.com accept  0
```

Properties

The following table shows the properties for the show registration-routing command.

Table 6–11 Show Registration-Routing Properties

Field	Description
plan-name	The name of the **route** or **source-route** registration plan entry.
type	The criteria for matching entries in the routing table. The system then applies the appropriate normalization plan to matching entries. The **type** field is derived from either the **to-uri-match** (**route**) or **source-match** (**source-route**) property.
match	If contributed through a route object entry, the string to match in the USER and/or HOST fields in the SIP header in order for the system to apply the normalization plan to calls containing the prefix. If contributed through a source-route object entry, the match criteria for the source of the SIP message. The system then sets the next-hop server (defined with the **peer** property) for all traffic that matches this configured source. The **match** field is derived from either the **to-uri-match** (**route**) or **source-match** (**source-route**) property. If type is condition-list, the match is derived from the priority plus plan name.
min	The minimum number of digits required for a match on a phone prefix, if configured. In some cases, the system calculates a value for other types of matches based on the number of characters (including wild cards). In some cases it displays as-is. The value is only meaningful to a phone-prefix match, however.
pri	The priority (order of preference) for this registration-plan entry. This property overrides the default behavior (most specific match) and sets a preference based on the **to-uri-match** (**route**) or **source-match** (**source-route**) property.
peer-name	A statically entered peer referenced through the **peer** property of the **route** or **source-route** object.

Table 6–11 (Cont.) Show Registration-Routing Properties

Field	Description
action	The action that the system is configured to take when a match occurs. The following describes each of the possible actions: **accept**: Accepts the registration locally, functioning as a registrar. **delegate**: Forwards the REGISTER to an upstream SIP proxy (provider) and resets the contact to itself. **forward**: Forwards the REGISTER, unchanged, to the server specified in the header. **redirect**: Sends a response to the client with instructions to resend the REGISTER to a different server. **tunnel**: Creates an OC client-to-LCS server tunnel, via the registration plan, that you can then load balance across. **discard**: Silently discards REGISTER requests matching this registration plan or AOR. **block**: Rejects calls matching this registration plan or AOR with either a "603 Declined" message or with configured text.
hits	The number of times this registration-plan entry was matched on and applied.

show registration-service

Displays status and statistics for each configured registration service (a registrar that can process REGISTER requests and add AORs updates to the location services database). Enable a domain to act as a registration service using the **registration-service** object.

Sample Output

```
NNOS-E> show registration-service
domain-name       admin     max-expiration     min-expiration
-----------       -----     --------------     --------
vfn.com           enabled   90                 30
```

Properties

The following table shows the properties for the show registration-service command.

Table 6–12 Show Registration-Service Properties

Field	Description
domain-name	The name of the domain that is enabled to act as the registration service. This is the configured VSP domain name.
admin	The administrative state of the registration service. If disabled in either the **registration-service** or session-config **registration** object, the system rejects any REGISTER request sent to the registration service.

Table 6–12 (Cont.) Show Registration-Service Properties

Field	Description
max-expiration	The maximum time (in seconds) to elapse before a client REGISTER request becomes invalid and the registration information is removed from the location cache. A value indicates that the registration-service configuration overwrites the maximum value; **as-requested** indicates that the system maintains the client value.
min-expiration	The minimum time (in seconds) to elapse before a client REGISTER request becomes invalid and the registration information is removed from the location cache. A value indicates that the registration-service configuration overwrites the maximum value; **as-requested** indicates that the system maintains the client value.

show restart-status

Displays the state of a controlled install or controlled restart of the system. Use this to check on the progress of these controlled install or restart actions. You can also use this command when you have used the **install file** or **url controlled** action to upgrade boxes in a cluster.

Sample Output

```
NNOS-E> show restart-status

started:
  state: idle
```

Properties

The following table shows the properties for the show restart-status command.

Table 6–13 Show Restart-Status Properties

Fields	Description
started	The time that the system install or restart was started.
state	The system state with regard to the install or restart action. For example, idle, loading, or restarting. The state may also indicate exactly what is loading or restarting, such as “loading 172.26.0.48” or “restarting sip stack.”

show route-server-box

To view the state of the route-server in the cluster use the show route-server-box action:

Sample Output

```
Cluster1> show route-server-box

box        master activated-at             routes     active-set
---        ------ ------------             ------     ----------
2          true   19:13:22 Wed 2010-05-26  3          five.xml
3          false  19:13:31 Wed 2010-05-26  3          five.xml
```

```
4          false  19:13:22 Wed 2010-05-26  3          five.xml
5          false  19:13:22 Wed 2010-05-26  3          five.xml
6          false  19:13:21 Wed 2010-05-26  3          five.xml
```

Properties

The following table shows the properties for the show route-server-box command.

Table 6–14 Show Route-Server-Box Properties

Fields	Description
box	The ME ID.
master	Whether the ME is the master.
activated-at	The currently active route-set was activated at this time.
routes	The number of routes in the currently active route-set.
active-set	The route-set file that is currently active.

show route-server-config

Displays information regarding tables configured under the **route-server > table-config** object.

Sample Output

```
NNOS-E>show route-server-config

name       filename        description             routes
-------    ------------    --------------------    ---------
default    routes.xml      My default routes       1000
lerg6      lerg6.xml       Telecordia exchanges    300000
```

Properties

The following table shows the properties for the show route-server-config command.

Table 6–15 Show Route-Server-Config

Field	Description
name	Name of the tagged table.
filename	Name of the table's associated route file.
description	Description of the table.
routes	Number of routes associated with the table.

show route-server-controlled-action-status

To display the progress of a currently executing update, activation, or deletion, use the show route-server-controlled-action-status action.

Sample Output

```
Cluster1> show route-server-controlled-action-status

box        master state                   routes     load-set
```

```
---        ------ -----                    ------     --------
2          true   Peer_Fetch_InProgress    80993      rsdid_201004131550.xml
3          false  Fetch_Success            0          rsdid_201004131550.xml
4          false  Active                     3
5          false  Fetch_Success            0          rsdid_201004131550.xml
6          false  Fetch_Success            0          rsdid_201004131550.xml
```

Properties

The following table shows the properties for the show route-server-controlled-action-status command.

Table 6–16 Show Route-Server-Controlled-Action-Status

Field	Description
box	The ME ID.
master	Whether the ME is the master.
state	The state of the ME. This field captures the information in both the box-state and result.
routes	The number of routes currently loaded.
load-set	The cluster is updating to this route-set.

show route-server-did

Displays DID routes in a DID start to DID end range. This can be helpful since, internally, DID routes are converted into prefix routes, making it harder for you to get a summary of active DID routes.

Sample Output

```
table range-start   range-end     carrier    endpoint  description
------  ----------   ------------   --------   ---------       ---
active    78153000    78253999     default   a.example.net   Peru
active    97876000    97876999     default   a.example.net   Brazil
```

Properties

The following table shows the properties for the show route-server-did command.

Table 6–17 Show Route-Server-DID Properties

Field	Description
table	The state of the DID route table.
range-start	The starting range of this DID prefix route.
range-end	The ending range of this DID prefix route.
carrier	The carrier for this DID prefix route.
endpoint	The endpoint for this DID prefix route.
description	A description of this DID prefix route.

show route-server-query

Displays information regarding queries configured under the **route-server-sequence** object.

Sample Output

```
NNOS-E>show route-server-queries

sequence    query        type       description                       hits
-------     ----------   ---------  --------------------------------  ----
Covergence  cgnLERG      variable   query the calling number OCN/LATA  10
Covergence  cdnLERG      variable   query the called number OCN/LATA   10
Covergence  interLATA    route      query inter-LATA routes            6
Covergence  intraLATA    route      query intra-LATA routes            4
```

Properties

The following table shows the properties for the show route-server-query command.

Table 6–18 Show Route-Server-Query Properties

Field	Description
sequence	Name of the route server sequence.
query	Name of the route server query.
type	Type of query. This can be either route or variable.
description	Description of the route server query.
hits	Number of times the query was referenced.

show route-server-sequence

Displays information regarding sequences configured under the **route-server-sequence** object.

Sample Output

```
NNOS-E>show route-server-sequences

name           description                     hits
-------        --------------------            ---------
Covergence     Use Covergence specific queries     888
Acme       Use Acme specific queries           1
```

Properties

The following table shows the properties for the show route-server-sequence command.

Table 6–19 Show Route-Server-Sequence Properties

Fields	Description
name	Name of the route server sequence.
description	Description of the route server sequence.
routes	Number of times the sequence was referenced.

show route-server-table

Displays DID route server tables configured on the ME.

Sample Output

```
tag       to-match      carrier     endpoint      description
------   ----------     --------    --------        -------
active    612864*        default    a.example.net    Peru
active    8621289*       default    a.example.net    Brazil
```

Properties

The following table shows the properties for the show route-server-table command.

Table 6–20 Show Route-Server-Table Properties

Field	Description
tag	The table tag.
to-match	The to-match value for this route server table.
carrier	The carrier for this DID route server table.
endpoint	The endpoint for this DID route server table.
description	A description of this DID route server table.

show route-server-table-config

Displays configured tables in the route server.

Sample Output

```
NNOS-E>show route-server-table-config

table                    filename                description
-----                    --------                -----------
Table1                   file1                   for customer1
Table2                   file2                   for customer 2
Table3                   did-file3               for customer 3
```

Properties

The following table shows the properties for the show route-server-table-config command.

Table 6–21 Show Route-Server-Table-Config

Field	Description
table	Name of the table.
filename	Name of the table's associated route file.
description	Description of the table.

show routing

Displays the generic routing table, which contains destination network and host addresses. These addresses are either configured IP interfaces (added through the **ip** object) or static routes (added with the **routing** object). If a route becomes unavailable, the ME removes it from the table. For example, if an interface becomes unavailable, the ME removes all associated routes.

The ME automatically installs (and the routing table displays) the loopback address. The ME uses this address, with a metric of 0 and a name of <lo>, for its internal connections.

This route table is for management traffic, not SIP or media traffic. As a result, no load balancing occurs. If the table should have two equal cost routes to reach a destination, the ME always uses the first listed route. Routes are ordered in the table by the length of the mask, since more specific routes are preferred.

Sample Output

```
NNOS-E> show routing
destination        type      gateway         source-ip       metric     name
-----------        ----      -------         ---------       ----
192.168.0.1/32     host      0.0.0.0         192.168.0.1     1          <eth0>
10.1.34.160/32     host      0.0.0.0         10.1.34.160     1          <vx112>
172.26.0.155/32    host      0.0.0.0         172.26.0.155    1          <vx111>
192.168.0.0/30     interface 0.0.0.0         192.168.0.1     1          <eth0>
10.1.34.0/24       interface 0.0.0.0         10.1.34.160     1          <vx112>
172.26.0.0/24      interface 0.0.0.0         172.26.0.155    1          <vx111>
127.0.0.1/8        interface 0.0.0.0         127.0.0.1       0          <lo>
172.0.0.0/8        network   172.26.0.254    172.26.0.155    1          172 nets
0.0.0.0/0          default   10.1.34.254     10.1.34.160     1          default
```

Properties

The following table shows the properties for the show routing command.

Table 6–22 Show Routing Properties

Fields	Description
destination	The destination address, which the system resolves to the corresponding next-hop for the route. If the next-hop is on the same local network, the destination itself is the next hop.
type	The type of device the route represents, either: **host**: A host IP address. This could come from either static or IP interface configuration. **interface**: A network route that is directly connected via an interface. **network**: An IP address and mask to match the destination network. Configured statically, this represents a network of hosts. **default**: Always at represented as 0.0.0.0/0, this route is applied for anything that has no other match in the table. Statically configure the gateway.
gateway	The next hop address for the route. A value of 0.0.0.0 indicates that the next hop is the destination itself.
source-ip	The IP address of the local interface on which the route is configured (the local interface address assigned with the route). When the route is selected this is the address the system uses to source the packet.
metric	The cost associated with the route, assigned when the route was configured (via **ip** or **routing** objects). The lower the metric the more preferred the route. The system chooses the more preferred route when there are multiple interfaces available on the same network.

Table 6–22 (Cont.) Show Routing Properties

Fields	Description
name	The name assigned to the route. If the name indicates an ethernet (eth*X*) or virtual (vx*X*) interface, it is the route associated with the local interface on the box. Otherwise, it displays the name assigned when you created a static route with the routing object.

show rtp-transcode-info

Displays session information relative to the transcoding of media types. Transcoding is the process of converting media from one CODEC into a different CODEC. This allows, in some cases, endpoints supporting different media types to communicate. See the **media** object for more information.

This command provides a quick snapshot of the current volume of transcoding activity, in terms of the number of streams, by state (active, DTMF-only, provisional, or unused state). It can be useful when troubleshooting log messages which indicate that transcoding limits have been exceeded.

Sample Output

```
NNOS-E> show rtp-transcode-info

          active: 2
       dtmf-only: 0
     provisional: 0
          unused: 0
   total-current: 2
         maximum: 2
exceeded-current: 0
  exceeded-count: 0
exceeded-seconds: 0
  exceeded-start:
    exceeded-end:
```

Properties

The following table shows the properties for the show rtp-transcode-info command.

Table 6–23 Show RTP-Transcode-Info Properties

Field	Description
active	The number of rtp-transcode streams passing RTP packets.
dtmf-only	The number of rtp-transcode streams waiting only to decode DTMF.
provisional	The number of rtp-transcode streams waiting for an SDP answer.
unused	The number of rtp-transcode streams with not encoders or decoders, typically the other side of DTMF-only.
total-current	The current number of rtp-transcode streams.
maximum	The maximum number of concurrent rtp-transcode streams.

Table 6–23 (Cont.) Show RTP-Transcode-Info Properties

Field	Description
exceeded-current	The number of rtp-transcode streams currently above the threshold.
exceeded-count	The number of times the system has exceeded the threshold.
exceeded-seconds	The number of seconds the system has operated above the threshold.
exceeded-start	The time at which the system last went above the threshold.
exceeded-end	The time at which the system last went below the threshold.

show rtp-transcode-stats

Displays statistics for active RTP transcoding sessions. It displays for each session ID the associated call leg, CODEC, and transcoding action taken by the ME, as well as packet counts. Use this status provider to view active and summary statistics for RTP transcoding. See Transcoding Media Types for more information on the action taken by the ME.

These transcoding statistics allow you to examine the RTP stream to determine which CODECs are being used for a given session. This command is most useful when trying to diagnose audio problems when transcoding is involved. Note that for basic calls, the rows marked with call-leg=1 describe the flow of media packets from the calling phone towards the answering phone (the forward path). The rows marked with call-leg=2 describe the flow of media from the answering phone towards the calling phone (the reverse path)

Sample Output

```
NNOS-E> show rtp-transcode-stats

session-id         call-leg  Action   Codec   payload-type  Packets   Dropped

----------         --------  ------   -----   ------------  -----

0x4c2b7ee991990f5  1         Decode   iLBC    99            481       0

                             Encode   pcma     8            721       0

                   2         Decode   pcma     8            720       0

                             Encode   iLBC    97            480       0
```

Properties

The following table shows the properties for the show RTP-Transcode-Stats command.

Table 6–24 Show RTP-Transcode-Stats Properties

Field	Description
session-id	The ID of the active RTP transcoding session.
call-leg	An index for a call leg indicating the direction of the call.

Table 6–24 (Cont.) Show RTP-Transcode-Stats Properties

Field	Description
Action	The action the system takes on RTP packets, either Encode or Decode, on a portion of a call leg. In the first line of the sample output, the system decoded 481 iLBC packets to linear samples. It encoded those samples to 721 PCMA packets. The difference in packet count is a result of the difference in the PTime defaults of the CODECs. There may be multiple decode CODECs per leg (e.g., if the phone negotiates more than one), but there will always be only one encode CODEC per leg.
Codec	The name of the CODEC.
payload-type	A value negotiated in the SDP, indicating the CODEC type for each RTP packet. There are certain well-known values (values below 96), but for dynamic payload types (above 96), there must be an "rtpmap" in the SDP. When you add a CODEC to the SDP (using the **media** transcode-media-types property), the system adds the corresponding rtpmap.
Packets	The total number of packets processed for this CODEC type in this media stream.
Dropped	The total number of packets dropped for this CODEC type in this media stream.

show rtp-transcode-summary

Displays summary statistics for RTP transcoding by CODEC. It includes actions and running packet counts since the last reboot. Use this status provider to view summary statistics for RTP transcoding. See Transcoding Media Types for more information on the action taken by the ME.

This command is most useful when trying to determine if transcode policy elements are set up correctly, for example, whether the right number of CODEC encoders have been allocated.

Sample Output

```
NNOS-E> show rtp-transcode-summary

Codec    Action    Alloc    Used     Active    Packets     Dropped
-----    ------    -----    ----     ------    -------     -------
iLBC     Decode    1        1        1         884         0
         Encode    1        1        1         882         0
pcma     Decode    1        1        1         1323        0
         Encode    1        1        1         1326        0
```

Properties

The following table shows the properties for the show rtp-transcode-summary command.

Table 6–25 Show RTP-Transcode-Summary

Field	Description
Codec	The name of the CODEC for which statistics are being reported.
Action	The transcoding action being reported on, either encode or decode.
Alloc	The number of times the system added a CODEC to the SDP for potential encoding or decoding use.
Used	The total number times the system used the CODEC for encoding or decoding (a CODEC must have processed at least one packet to be counted).
Active	The total number of this CODEC type currently in use, for encoding and for decoding, although it may not have yet been used. It may be allocated and listed as available, but may not have yet processed and RTP packets.
Packets	The total number of packets received and sent, encoded and decoded, for this CODEC type.
Dropped	The total number of packets dropped, encoded and decoded, for this CODEC type.

show rules

Displays the status and activity level of each rule configured in each policy. Rules are configured using the **rule** object.

Sample Output

```
NNOS-E> show rules

       name: policy default/rule regauth
      admin: enabled
evaluations: 0
  successes: 0
       name: policy default/rule abcTOxyz
      admin: enabled
evaluations: 0
  successes: 0
       name: policy default/rule abcTOdef
      admin: enabled
evaluations: 0
  successes: 0
       name: policy default/rule abc
      admin: enabled
evaluations: 0
  successes: 0
```

Properties

The following table shows the properties for the show rules command.

Table 6–26 Show Rules Properties

Field	Description
name	The name of the policy and the rule to which the policy applies.
admin	The administrative state of the rule associated with the named policy. If disabled, all other enabled rules under the named policy will still be checked against incoming SIP messages.
evaluations	The number of times the rule was applied against a SIP REQUEST message.
successes	The number of times a SIP REQUEST message matched the policy rule.

show sensor-events

Displays events sent to the Intelligent Platform Management Interface (IPMI) event log. Output indicates the effected sensor and an event description and time stamp.

Sample Output

```
NNOS-E> show sensor-events
timestamp                 sensor             type                     description
---------                 ------             ----
11:48:31 Mon 2006-10-09   Logging Disabled0  event-logging-disabled   log area
reset/cleared
11:48:38 Mon 2006-10-09   System Event0      system-event             OEM system
boot event
111:48:50 Mon 2006-10-09  SATA  Drv 3 Pres0  drive-slot               device
removed/absent
11:48:55 Mon 2006-10-09   SATA  Drv 4 Pres0  drive-slot               device
removed/absent
11:48:55 Mon 2006-10-09   Power Redundancy0  power-unit               fully
redundant
11:48:58 Mon 2006-10-09   POST Error0        system-firmware-progress error
11:48:58 Mon 2006-10-09   POST Error0        system-firmware-progress error
11:49:24 Mon 2006-10-09   System Event0      system-event             OEM system
boot event
11:49:24 Mon 2006-10-09   System Event0      system-event             timestamp
clock sync
11:02:10 Wed 2006-10-18   SATA  Drv 6 Pres0  drive-slot               device
removed/absent
11:02:13 Wed 2006-10-18   Power Redundancy0  power-unit               fully
redundant
```

Properties

The following table shows the properties for the show sensor-events command.

Table 6–27 Show Sensor-Events Properties

Field	Description
timestamp	The time the sensor event occurred.
sensor	The name of the effected sensor.
type	The category into which the sensor falls.

Table 6–27 (Cont.) Show Sensor-Events Properties

Field	Description
description	Descriptive text that indicates the specific action that triggered the event.

show sensor-info

Displays version and state information for the Intelligent Platform Management Interface (IPMI).

Sample Output

```
NNOS-E> show sensor-info
  version: 2.0
    state: up
   faults:
self-test: Success!
```

Properties

The following table shows the properties for the show sensor-info command.

Table 6–28 Show Sensor-Info Properties

Field	Description
version	The version of firmware running on the interface.
state	The state of the IPMI, either: initializing: Initializing IPMI unsupported: IPMI is not supported on this platform up: IPMI is active
faults	A reading of any detected hardware faults. The field is left blank if there are no faults, or displays one of the following: controller: The controller failed to power the system on or off. power: There was a main power subsystem fault. interlock: The chassis power interlock switch is active. overload: There is a power overload. fan: A cooling fan fault has been detected drive: There is a drive fault. intrusion: There has been a chassis intrusion.
self-test	The results of the IPMI power-on self-test. If you receive anything other than Success!, contact Technical Support.

show sensors

Displays status information for the various system hardware sensors (temperature, fan speed, etc.)

Sample Output

```
NNOS-E> show sensors
sensor                  type             value
------                  ----             -----
+1.5V NIC Core          voltage          1.5288 volts
AUX +3.3V               voltage          3.264 volts
BB +1.2V Vtt            voltage          1.2096 volts
BB +1.5v                voltage          1.5132 volts
BB +12V                 voltage          12.152 volts
BB +3.3V                voltage          3.354 volts
BB +5V                  voltage          5.07 volts
BB -12V                 voltage          -11.758 volts
BB Vbat0                battery          001------------
BMC Watchdog0           watchdog-2       0000----0------
Baseboard Fan 1         fan              5254 RPM
--more--
```

Properties

The following table shows the properties for the show sensors command.

Table 6–29 Show Sensors Properties

Field	Description
sensor	The name of the effected sensor.
type	The category into which the sensor falls.
value	The current value or reading of the associated sensor.

show server-conn-lookup

Displays entries of the SIP host lookup table that are using either TCP or TLS for transport and have a **host** configured. These entries are the servers configured using the **server-pool server** object or the gateways configured with the **switch** object. Use the **show server-host-lookup** command to display entries with hosts using any transport protocol.

Sample Output

```
NNOS-E> show server-conn-lookup
peer-name   server-name   trunk-name   connection  host        TPT   local  hits
---------   -----------   ----------   ----------  ----        ---   -----  ----
Cluster     10.1.34.13    10.1.34.13   0           10.1.34.13  TLS   5061   294
```

Properties

The following table shows the properties for the show server-conn-lookup command.

Table 6–30 Show Server-Conn-Lookup Properties

Field	Description
peer-name	SIP gateway server or carrier name.
server-name	The name of the server-pool server or gateway.

Table 6–30 (Cont.) Show Server-Conn-Lookup Properties

Field	Description
trunk-name	The name of a trunk group associated with the corresponding carrier gateway. If no gateway is configured, or the peer is a server and not a carrier, the field is blank.
connection	The memory address of the socket hosting the TCP or TLS connection.
host	The host name or IP address of the server or gateway involved in the connection. This is derived from the **host** field of the server or carrier configuration.
TPT	The transport protocol in use for the connection with this server, either TCP or TLS.
local	The port number the system is configured to use (with the server **local-port** property) in the Contact header, Via header, and source port when it sends a Register request (and subsequent SIP messages) to an upstream server. The server caches the binding and includes the local-port when contacting the system. Additionally, the server can be configured to send SIP messages to this particular local-port without prior registration from the system. With local-port configured, the system can tell: To which connection in the server pool to forward a call Which connection in the server pool it received the call from, when the connection sends SIP message to this local port
hits	The number of times the system has forwarded traffic to the server.

show server-host-lookup

Displays entries of the SIP host lookup table that are using any transport protocol and have a **host** configured. These entries are the servers configured using the server-pool **server** object or the gateways configured with the **switch** object. Use the **show server-conn-lookup** command to display only those entries using TCP or TLS for transport.

Sample Output

```
NNOS-E> show server-host-lookup
peer-name   server-name   trunk-name   connection  host        TPT   local  hits
---------   -----------   ----------   ----------  ----        --
Cluster     10.1.34.13    1.1.1.1      0           1.1.1.1     UDP   0      294
NNOS-E@NY      NY1                        0           100.0.0.1   TLS   15061  569
3
NNOS-E@SSJ     SJ1                        0           200.0.0.1   TLS   25061  692
3
```

Properties

The following table shows the properties for the show server-host-lookup command.

Table 6–31 Show Server-Host-Lookup Properties

Field	Description
peer-name	The SIP gateway server or carrier name.
server-name	The name of the server-pool server or gateway.
trunk-name	The name of a trunk group associated with the corresponding carrier gateway. If no gateway is configured, or the peer is a server and not a carrier, the field is blank.
connection	The memory address of the socket hosting the TCP or TLS connection.
host	The host name or IP address of the server or gateway involved in the connection. This is derived from the **host** field of the server or carrier configuration.
TPT	The transport protocol in use for the connection with this server, either TCP, TLS, or UDP.
local	The port number the system is configured to use (with the server **local-port** property) in the Contact header, Via header, and source port when it sends a Register request (and subsequent SIP messages) to an upstream server. The server caches the binding and includes the local-port when contacting the system. Additionally, the server can be configured to send SIP messages to this particular local-port without prior registration from the system. With local-port configured, the system can tell: to which connection in the server pool to forward a call which connection in the server pool it received the call from, when the connection sends SIP message to this local port
hits	The number of times the system has forwarded traffic to the server.

show services-routing

Displays a summary of the routes in all service routing tables. Information displayed includes the table type, destination and gateway, source IP address and the origin.

The verbose form displays **services-routing** metrics and cluster **media-partners** information (for media). In the CLI, you can filter the output can to display a specific table by entering the name of the service route table to display in the command line (e.g., show services-routing media).

The origin is either local or cluster, where local is a local route on the box itself and cluster means the route was learned from another box in the cluster. Source IP is the local interface to reach that route destination. If the route is a local route, then the source-ip is the local IP interface on which the route was configured. If the route is a cluster route, the source-ip is the local interface used to communicate with other cluster box that advertised the route (i.e., the local interface in which the route was learned).

Sample Output

```
NNOS-E> show services-routing
service              destination        gateway         source-ip       origin
-------              -----------        -------         ---------
sip                  10.1.34.0/24       0.0.0.0         10.1.34.160     local
sip                  172.26.0.0/24      0.0.0.0         172.26.0.155    local
sip                  172.0.0.0/8        172.26.0.254    172.26.0.155    local
sip                  0.0.0.0/0          10.1.34.254     10.1.34.160     local
media                10.1.34.0/24       0.0.0.0         10.1.34.160     local
media                172.26.0.0/24      0.0.0.0         172.26.0.155    local
media                172.0.0.0/8        172.26.0.254    172.26.0.155    local
media                0.0.0.0/0          10.1.34.254     10.1.34.160     local
```

Properties

The following table shows the properties for the show services-routing command.

Table 6–32 Show Services-Routing Properties

Field	Description
service	The name of the service, either media, stun, or sip.
destination	The destination address, which the system resolves to the corresponding next-hop for the route. If the next-hop is on the same local network, the destination itself is the next hop.
gateway	The next hop address for the route. A value of 0.0.0.0 indicates that the next hop is the destination itself.
source-ip	The local interface to reach the route destination. If the route is a local route, then the source-ip is the local IP interface on which the route was configured. If the route is a cluster route, the source-ip is the local interface used to communicate with the other system cluster box that advertised the route (i.e., the local interface from which the route was learned).
origin	The origin of the route, either local or cluster. Local means a local route on the box itself and cluster means the route was learned from another box in the cluster.

show services-routing-config

Displays current configuration settings for the **services-routing** and **load-balancing** objects.

Sample Output

```
NNOS-E>show services-routing-config

Common Settings
---------------
metric-timer:        0
cpu-sample-interval: 60
cpu-sample-divisor:  10

h323
```

```
---------------
metric1: user-metric     round-robin
metric2: none            round-robin
metric3: none            round-robin
metric4: none            round-robin
metric5: none            round-robin

media
---------------
metric1: user-metric     round-robin
metric2: none            round-robin
metric3: none            round-robin
metric4: none            round-robin
metric5: none            round-robin

sip
---------------
metric1: user-metric     round-robin
metric2: none            round-robin
metric3: none            round-robin
metric4: none            round-robin
metric5: none            round-robin

stun
---------------
metric1: user-metric     round-robin
metric2: none            round-robin
metric3: none            round-robin
metric4: none            round-robin
metric5: none            round-robin
```

Properties

The following table shows the properties for the show services-routing-config command.

Table 6–33 Show Services-Routing-Config

Field	Description
metric-timer	The interval, in seconds, that the ME updates services routing metrics.
cpu-sample-interval	The sampling interval, in seconds, at which the ME calculates the system CPU usage. The result of this calculation is a CPU usage percentage (between 0 and 100), used as the box-cpu-load metric.
cpu-sample-divisor	The value used to divide the raw CPU usage percentage used to normalize the box-cpu-load-metric.
h323	Displays the H.323 services-routing load balancing configuration.
media	Displays the media services-routing load balancing configuration.
sip	Displays the SIP services-routing load balancing configuration.
stun	Displays the STUN services-routing load balancing configuration.

show services-routing-load-share

Displays the calculated load-share, weight, and metric values for the services route tables.

Sample Output

```
NNOS-E>show services-routing-load-share

service  destination  source-ip  load-share weight met1 met2 met3 met4 met5
-------  -----------  ---------  ---------- ------ ---- ---- ---- ---- ----
sip     10.1.67.1/32  10.1.67.1   1         0      1    0    0    0    0
sip     10.1.69.1/32  10.1.69.1   1         0      1    0    0    0    0
sip     10.1.67.0/24  10.1.67.1   1         0      1    0    0    0    0
```

Properties

The following table shows the properties for the show services-routing-load-share command.

Table 6–34 Show Services-Routing-Load-Share Properties

Field	Description
service	The name of the service, either media, stun, or sip.
destination	The destination address, which the system resolves to the corresponding next-hop for the route. If the next-hop is on the same local network, the destination itself is the next hop.
source-ip	The local interface to reach the route destination. If the route is a local route, then the source-ip is the local IP interface on which the route was configured. If the route is a cluster route, the source-ip is the local interface used to communicate with the other system cluster box that advertised the route (i.e., the local interface from which the route was learned).
load-share	When one or more metrics are configured as weighted-round-robin (WRR), those metrics are used to calculate this value for each route in an equal cost set.
weight	When one or more metrics are configured as WRR, this value displays each WRR metric's dynamic weight that is calculated based on how close the metric is to its upper bound. This value is then used to calculate its load share across an equal cost route set.
met1-5	Displays the five services routing metric fields for each service (SIP, H.323, media, and STUN).

show services-routing-tables

Displays all the services routing tables (the default tables and those created as a result of tag-based route selection configured on an **ip** interface). The display also includes a total route count, which includes both active and inactive routes.

Sample Output

```
NNOS-E> show services-routing-tables

service                     route-count
-------                     -----------
media                       3
media.qik-eastmode          3
media.tomp                  3
sip                         3
sip.lcs1-eastmode           3
sip.qik-eastmode            3
sip.tomp                    3
stun                        0
```

Properties

The following table shows the properties for the show services-routing-tables command.

Table 6–35 Show Services-Routing-Tables Properties

Field	Description
service	The name of the services routing table.
route-count	The number of routes in the table.

show sip-authentication

Displays information about SIP authentication messages handled by the ME.

Sample Output

```
NNOS-E>show sip-authentication

                                    name: default
                sip-stack-pre-auth-timeout: 30 seconds
           sip-stack-pre-auth-max-pendings: 1024 seconds
    total-blocking-authentication-messages: 0
         total-sip-stack-pre-auth-messages: 0
            total-auth-suppressed-messages: 0
total-sip-stack-pre-auth-api-timeouts: 0 seconds
         total-sip-stack-pre-auth-timeouts: 0
total-sip-stack-pre-auth-unmatched-replies: 0
           total-sip-stack-pre-auth-queued: 0
            most-sip-stack-pre-auth-queued: 0
```

Properties

The following table shows the properties for the show sip-authentication command.

Table 6–36 Show SIP-Authentication Properties

Field	Description
name	The name of the VSP instance.
sip-stack-pre-auth-timeout	The number of seconds before the server times out and the ME discards the message.

Table 6–36 (Cont.) Show SIP-Authentication Properties

Field	Description
sip-stack-pre-auth-max-pendings	The number of seconds the ME allows a message to stay in its queue pending authentication.
total-blocking-authentication-messages	The number of authentication messages that will cause the ME to block message processing.
total-sip-stack-pre-auth-messages	The number of messages the ME has authorized.
total-auth-suppressed-messages	The number of authentication messages that have been suppressed. An authentication message is suppressed if the ME receives a REGISTER for an AOR that has not yet expired. Rather than authenticating, the ME sends back a 200 OK.
total-sip-stack-pre-auth-api-timeouts	The number of times the ME sends a message to the internal Authentication process and does not receive a response.
total-sip-stack-pre-auth-timeouts	The number of times an authentication message expired while waiting in the queue.
total-sip-stack-pre-auth-unmatched-replies	The number of authentication messages that failed because the ME could not find a match for authorization credentials.
total-sip-stack-pre-auth-queued	The total number of queued authorization messages.
most-sip-tack-pre-auth-queued	The Maximum number of queued authentication messages allowed.

show sip-authorization-details

Displays detailed information about SIP authorization on the ME.

Sample Output

```
NNOS-E>show sip-authorization-details
-----------------------------------------------------------------
Provider      Requests   Accepts  Average   Errors  Average  QClipped    Others
-----------------------------------------------------------------
Local                0         0    0.000        0    0.000         0         0
WSDL                 0         0    0.000        0    0.000         0         0
Diameter             0         0    0.000        0    0.000         0         0
RADIUS               0         0    0.000        0    0.000         0         0
-----------------------------------------------------------------
```

Properties

The following table shows the properties for the show sip-authorization-details command.

Table 6–37 Show SIP-Authorization-Details Properties

Field	Description
Provider	The protocol to be used for authorization.

Table 6–37 (Cont.) Show SIP-Authorization-Details Properties

Field	Description
Requests	The number of requests submitted to each provider.
Accepts	The number of positive replies received from the remote server.
Average	The average response time, in seconds, for each Accept.
Errors	The number of errors (rejected by the server, timed out, etc.).
Average	The average response time, in seconds, for each Error response. Note that this value is maintained separately because a server often delays a negative response for 1-2 seconds in an attempt to avoid attacks.
QClipped	The number of requests that failed locally, without ever being sent to the remote server, because the queue of requests outstanding to the server(s) has grown too long. This is often indicative of problems.
Others	The sum of any other types of errors. These can be seen individually by adding **-v** to the end of this action.

show sip-peers

Displays the name, domain, type and failover detection method of all configured SIP registration peers. Peers are identified with the **server** object.

Sample Output

```
NNOS-E> show sip-peers
name                   domain          type       mode        detect
----                   ------          ----       ----        --
Cluster 10.1.34.13     vfn.com         provider   provider    none
```

Properties

The following table shows the properties for the show sip-peers command.

Table 6–38 Show SIP-Peers Properties

Field	Description
name	The name of the configured enterprise SIP server.
domain	The domain name associated with the server.
type	The service-type setting for the server, either provider, internal, or external. Service type specifies the way in which the system handles INVITE and REGISTER requests and database exchanges.

Table 6–38 (Cont.) Show SIP-Peers Properties

Field	Description
mode	The status of the peer. When a server is down (not reachable), if the **routing-setting** property of the pstn-backup attribute is not selected, the system changes the state of the server to "not available." If it is selected, an unavailable server's state changes to "local mode." In its normal state, the system operates in provider mode, forwarding calls to a provider's application server. If the server fails, and the system has location information for the provider, it forwards calls locally. Otherwise, the system forwards calls to a PSTN gateway. You configure the gateway using the pstn-gateway server object. This is called local mode.
detect	The **failover-detection** setting for the server. Failover detection determines the method to use to detect a when a upstream server peer is unavailable. This field could display none, auto, ping, or register.

show sip-server-availability

Displays **server-pool server-pool-admission-control** or carrier **switch** configuration and status information for each configured SIP server. Configuration data includes transport protocol, port, failover detection method, counts, and thresholds.

Sample Output

```
NNOS-E>show sip-server-availability

host    transport port  detect waiting time count threshold fallback status reason
----    --------- ----  ------ ------- ---- ----- --------- -------- ------ ------
1.3.6.6  UDP      5060  none   false   0    0     4         300       up
internal-error
9.1.7.9  UDP      5060  none   false   0    0     4         300       up
detection-disabled
2.6.7.8  UDP      5060  none   false   0    0     4         300       up
detection-disabled
1.3.1.5  UDP      5060  none   false   0    0     4         300       up
detection-disabled
```

Properties

The following table shows the properties for the show sip-server-availability command.

Table 6–39 Show SIP-Server-Availability

Field	Description
host	The name of IP address of the **server-pool-admission-control** or **switch**.
transport	The protocol used by the **server-pool-admission-control** or **switch**, either any, TCP, UDP, or TLS.

Table 6–39 (Cont.) Show SIP-Server-Availability

Field	Description
port	The port used by the **server-pool-admission-control** or **switch** for SIP traffic.
detect	The **failover-detection** setting for the hosting **server** or **carrier**, either none, auto, ping, or register. Failover detection determines the method to use to detect a when a upstream server peer is unavailable.
waiting	The state of the system pinging the host. If true, the system has sent a ping and is awaiting a response. If false, it has not pinged the host.
time	The amount of time (number of seconds) the system has waited for a response to a ping to the host.
count	The number of failed pings to the host (the dead count).
threshold	The **dead-threshold** setting for the hosting **server** or **carrier**. This threshold specifies the number of transaction failures (and resulting retransmissions) a server can experience before the server state is changed to DOWN.
fallback	The **dead-fallback-interval** setting for the hosting **server** or **carrier**. During this period, the system does not send REGISTER or INVITES to the down server.
status	The current status of the server-pool **server-pool-admission-control** or carrier **switch**.
reason	The reason for the SIP server's availability status.

show sip-server-redirect

Displays a set of call counters used when the 302 redirect feature is enabled on the ME.

Sample Output

```
NNOS-E>
show sip-server-redirect


gateway               peer                  current    maximum    total
-------               ----                  -------    -------
RedirectClusters      cluster-Moog/Neely    0          1000
RedirectClusters      cluster-Butler/Thacker 0         50
SIPp-A                SIPp1                 0          50   ` 0
```

Properties

The following table shows the properties for the show sip-server-redirect command.

Table 6–40 Show SIP-Server-Redirect Properties

Field	Description
gateway	The name of each configured gateway.
peer	The name of each gateway's configured peer.
current	Indicates the current weight relative to the maximum. This value resets by either all gateways reaching their maximum weights or the cross cluster statistics being reported.
maximum	The maximum session threshold for each gateway.
total	A count of the number of redirects sent to a server.

show sip-server-pool

Displays all hosts available for each **server-pool server-pool-admission-control** peer, and the status of each. In addition, the output indicates configuration settings, and the number of requests sent to the peer.

Sample Output

```
NNOS-E> show sip-server-pool
peer-name     server        host         TRPT   port   box    state  hits  pref
---------     ------        ----         ----   ----   ---    ---
PBX Maynard   PBX Maynard   0.0.0.0      UDP    5060   local  up     0     none
PBX Boston    PBX Boston    0.0.0.0      UDP    5060   local  up     0     none
NNOS-E@SanJose   SJ1           200.0.0.1    TLS    5061   1      up     0     10
NNOS-E@SanJose   SJ2           200.0.0.2    TLS    5061   1      up     0     20
```

Properties

The following table shows the properties for the show sip-server-pool command.

Table 6–41 Show SIP-Server-Pool

Field	Description
peer-name	The name of the server hosting the connection. This is the enterprise **server**.
server	The name of the server entry in the server pool, configured with the **server-pool server-pool-admission-control** object. If the enterprise server is of type **sip-connection**, the server is the same as the peer-name.
host	The host address of the **server-pool server-pool-admission-control**. This value is configured with the **host** property. A host address of 0.0.0.0 indicates that the host is not configured. For a server of type **sip-connection**, a host of 0.0.0.0 indicates that the host will be learned dynamically through registrations. To do so, however, the local-port property must be set (non-zero).
TRPT	The protocol used by the connection.
port	The port used by the connection for SIP traffic.

Table 6–41 (Cont.) Show SIP-Server-Pool

Field	Description
box	The box on which the server is configured. A value of 0 indicates the local box.
state	The operational state of the hosting server. This state is determined by failover checks, configured in the enterprise **server failover-detection** property. If that property is set to **none**, the state always appears as up.
hits	The number of requests sent to the peer.
pref	The preference assigned to the server-pool server, which specifies the preference for the connection. The lower the value the higher the preference. If you use the value of **none**, the system uses the preference set in a different part of the configuration, such as the ordered set of arbitration rules in the **dial-plan** object.

show sip-stack

Displays general statistics about the ME SIP process. (Other SIP display commands provide more detailed counters specific to an aspect or process.)

Sample Output

```
NNOS-E> show sip-stack
                                  name: default
                                  mode: auto-determine
                            interfaces: 0
                          active-calls: 1
                       connected-calls: 1
                           total-calls: 10
                     total-failed-calls: 7
                    active-associations: 5
                       active-sessions: 1
                 total-message-received: 455
                     total-message-sent: 507
            total-dns-pending-messages: 0
           total-enum-pending-messages: 0
      total-location-pending-messages: 0
total-authentication-pending-messages: 0
                        worker-threads: 20
                        socket-threads: 1
                                status: Running
                          policy-epoch: 0
                call-admission-control: enabled
                             max-calls: 1000
                     max-registrations: 30000
                    max-calls-in-setup: 200
                         current-calls: 1
                current-calls-in-setup: 0
                     max-calls-dropped: 0
              max-calls-in-setup-dropped: 0
                         max-tls-calls: 10000
                max-tls-calls-in-setup: 500
                     current-tls-calls: 0
              current-tls-calls-in-setup: 0
```

```
                max-tls-calls-dropped: 0
          max-tls-calls-in-setup-dropped: 0
```

Properties

The following table shows the properties for the show sip-stack command.

Table 6–42 Show SIP-Stack Properties

Field	Description
name	The name of the VSP that this command is reporting on.
mode	The SIP operating mode to use with this server, as configured with the sip-settings object of the session config. Settings are either: auto-determine: The system determines the mode. proxy: The system is the SIP proxy that provides SIP registration, location, policy, and other services that determine the outcome of the SIP call
interfaces	The number of active SIP interfaces on the local box.
active-calls	The total number of calls, both in progress and connected, that the system is currently handling.
connected-calls	The number of connected calls currently being managed by the system.
total-calls	The total number of calls processed by the system since the last boot.
total-failed-calls	The total number of calls that did not make it to the connect state since the last boot.
active-associations	The number of “address pairs” that were used for communications through the system since it last boot. When a session is setup, there is a TO and FROM user (e.g., joe@123.com calls bob@456.org). The association is a unique identifier assigned to that combination.
active-sessions	The number of currently active SIP sessions. Active sessions include, calls, registrations, and presence sessions.
total-message-received	The total number of SIP messages received since last boot.
total-message-sent	The total number of SIP messages sent since last boot.
total-dns-pending-messages	The total number of SIP messages awaiting DNS lookup.
total-enum-pending-message	The total number of SIP messages awaiting ENUM lookup.
total-location-pending-messages	The total number of SIP messages awaiting master location database lookup.
total-authentication-pending-messages	The total number of SIP messages awaiting authentication verification.

Table 6–42 (Cont.) Show SIP-Stack Properties

Field	Description
worker-threads	The value of the **stack-worker-threads-max** property set in the **settings** object. This value is the number of SIP stack processing threads to create for this VSP. If the value displays as 0, the system executes a single thread.
socket-threads	The value of the **stack-socket-threads-max** property set in the **settings** object. This value is the number of SIP stack processing threads that should be used for TLS processing. If you are not using TLS, the value should be 1. If you are using TLS, value should be 4.
status	The state of the SIP stack, either running or disabled. This can be administratively altered using the **admin** property of the **vsp** object.
policy-epoch	A general counter that indicates the number of times there has been a change to a session-config, default-session-config, rule, enterprise server, or server-pool server configuration. This value is used internally for policy update decisions.
call-admission-control	The call-admission control setting. This setting must be enabled for the following to be applicable: max-calls-in-setup max-number-of-tls max-tls-in-setup If disabled, only the more general **max-number-of-sessions** property controls setup and connection limits.
max-calls	The maximum number of concurrent SIP sessions that the VSP can support. This value includes all REGISTER, SUBSCRIBE, INVITE, and other sessions. This value is set with the **max-number-of-sessions** property in the vsp>**settings** object
max-registrations	The maximum number of registrations allowed in the location cache. When the system reaches the maximum registration count, registrations are denied until the number falls below this threshold. This value is set with the **max-number-of-registrations** property in the vsp>**settings** object
max-calls-in-setup	The value of the **max-calls-in-setup** property set in the **vsp** object. This setting controls the maximum number of inbound call legs in setup stage allowed by the CAC.
current-calls	The current number of SIP sessions.
current-calls-in-setup	The current number of SIP sessions in setup stage.
max-calls-dropped	The number of SIP sessions that were dropped because you hit the **max-calls** threshold.
max-calls-in-setup-dropped	The number of in-progress SIP sessions that were dropped because you hit the **max-calls-in-setup** threshold.

Table 6–42 (Cont.) Show SIP-Stack Properties

Field	Description
max-tls-calls	The value of the **max-number-of-tls** property set in the **vsp** object. This setting controls the maximum number of TLS connections allowed on the VSP. This would include TLS connections for any type of SIP traffic, and includes TLS calls in setup and those that are established.
max-tls-calls-in-setup	The value of the **max-tls-in-setup** property set in the **vsp** object. This setting controls the maximum number of TLS connections allowed to be in the setup stage at one time. Establishing a TLS connection is very compute-intensive, so this value helps protect the system from being over-burdened by TLS connections.
current-tls-calls	The total number of TLS connections currently handled by the system, both established and in setup.
current-tls-calls-in-setup	The number of TLS connections currently in setup.
max-tls-calls-dropped	The number of TLS connections that were dropped because you hit the **max-tls-calls** threshold.
max-tls-calls-in-setup-dropped	The number of in-progress TLS connections that were dropped because you hit the **max-tls-calls-in-setup** threshold.

show sip-trunk-ports

Displays the bindings of AORs to trunk ports. The configuration for this is a result of the registration-plan **route alter-contact** property set to **trunk-port-per-binding**.

Trunk ports are allocated based on AORs, and the AORs are sent in the phone registrations. This command is most useful when registrations are failing because it allows you to observe the trunk ports. For example, if a registration fails, you can compare the allocated ports to the ports specified in the SIP message to determine whether there is a problem. In other cases, if SIP messages (like INVITES or NOTIFYs) are not being responded to correctly, it could be the result of a SIP message specifying an AOR/trunk port combination that has not been properly allocated.

The output of this command is most useful when combined with other information, such as SIP message logs and code traces.

Sample Output

```
NNOS-E> show sip-trunk-ports

outbound-port        source-port        owner                 hits
-------------        -----------        -----                 ---
172.26.0.109:24477   172.30.0.177:7584  sip:7812454444@rk.com 3
```

Properties

The following table shows the properties for the show sip-trunk-ports command.

Table 6–43 Show SPI-Trunk-Ports Properties

Field	Description
outbound-port	The IP address and port number of the system.
source-port	The IP address and port number of the client server.
owner	The AOR of the source.
hits	The number of times the system was able to match the call to the port allocated for the caller (source).

show stun-server

Provides information regarding the STUN server and, if configured, the TURN server. When you enter this action with a **-v** to display verbose information, the ASC displays information regarding the TURN server associated with the STUN server.

Sample Output

```
NNOS-E> show stun-server -v
index: 0
ifindex: 1
transport: UDP
ip-address: 172.44.10.60
port: 3478
turn-redirector: disabled
secondary-ifindex: 0
relay-ifindex: 1
relay-allocation-count: 2
connects: 0
disconnects: 0
rx-requests: 6065
tx-responses: 5394
tx-error-responses: 671
discards: 12
rx-binding-requests: 553
tx-binding-responses: 451
tx-binding-error-responses: 102
rx-allocate-requests: 244
tx-allocate-responses: 122
tx-allocate-error-responses: 122
rx-send-indications: 334
tx-app-relayed-data: 146669093
rx-app-relayed-data: 146141259
tx-data-indications: 423
tx-lite-framed-data: 146138511
message-integrity-failures: 0
fingerprint-failures: 0
rx-refresh-requests: 2614
tx-refresh-responses: 2321
tx-refresh-error-responses: 293
rx-create-permission-requests: 33
tx-create-permission-responses: 33
tx-create-permission-error-responses: 0
rx-channel-bind-requests: 2621
tx-channel-bind-responses: 2467
tx-channel-bind-error-responses: 154
```

```
rx-channel-data-indications: 146668771
tx-channel-data-indications: 146138511
```

Properties

The following table shows the properties for the **show stun-server** command.

Table 6–44 Show Stun-Server Properties

Field	Description
index	The index of this server
ifindex	The interface index used for this STUN server.
transport	The transport method used for this STUN server.
ip-address	The IP address used for this STUN server listener.
port	The port used for this STUN server listener.
turn-redirector	Indicates if the TURN redirector is enabled (not currently supported)
secondary-ifindex	The secondary interface index used for STUN server change-address.
relay-ifindex	The interface index used for the TURN server relay.
relay-allocation-count	The number of TURN Allocations in use by the TURN server.
connects	Not currently supported.
disconnects	Not currently supported.
rx-requests	The number of STUN/TURN requests received.
tx-responses	The number of STUN/TURN success responses sent.
tx-error-responses	The number of STUN/TURN error responses.
discards	The number of STUN/TURN messages discarded.
rx-binding-requests	The number of TURN binding requests received.
tx-binding-responses	The number of TURN binding success responses sent.
tx-binding-error-responses	The number of TURN binding error responses sent.
rx-allocate-responses	The number of TURN Allocate requests received.
tx-allocate-responses	The number of TURN Allocate success responses sent.
tx-allocate-error-responses	The number of TURN Allocate error responses sent.
rx-send-indications	The number of TURN send indications received from a TURN client.
tx-app-relayed-data	The total number of TURN encapsulated relay messages data sent.
rx-app-relayed-data	The total number of TURN encapsulated relay messages data received.
tx-data-indications	The number of TURN data indications sent to a TURN client.
tx-lite-framed-data	The number of TURN channel data indications transmitted.
message-integrity-failures	The number of STUN/TURN messages with improper authentication received.
fingerprint-failures	Not currently supported.
rx-refresh-requests	The number of TURN refresh requests received.

Table 6–44 (Cont.) Show Stun-Server Properties

Field	Description
tx-refresh-responses	The number of TURN refresh success responses sent.
tx-refresh-error-responses	The number of TURN refresh error responses sent.
rx-create-permission-requests	The number of TURN permission requests received.
tx-create-permission-responses	The number of TURN permission success responses sent.
tx-create-permission-error-responses	The number of TURN permission error responses sent.
rx-channel-bind-requests	The number of TURN binding requests received.
tx-channel-bind-responses	The number of TURN binding success responses sent.
tx-channel-bind-error-responses	The number of TURN binding success responses sent.
rx-channel-data-indications	The number of TURN data channel messages received from a TURN client.
tx-channel-data-indications	The number of TURN data channel messages sent to a TURN client.

show system-info

Displays the ME system information, including operating system release and version information and other integrated software information.

Sample Output

```
NNOS-E> show system-info
Machine type:      i686
System name:       Linux
Node name:         172.26.0.155
Box identifier:    01a1-6f9e-dcfa-9b9e
Kernel version:    2.6.11-4-cov
OS version:        1.4-22573-b3.2.0
OS uptime:         1 days 19:52:58
LIBC version:      glibc 2.3.4
Pthread version:   NPTL 2.3.4
OpenSSL version:   OpenSSL 0.9.8c 26 Sep 2006 (companyXYZ-20291)
  Export status:   Full version
Database version:  07.02.0005 PostgreSQL 8.1.2
```

Properties

The following table shows the properties for the show system-info command.

Table 6–45 Show System-Info Properties

Field	Description
Machine type	The Intel processor type.
System name	The system's operating system.
Node name	The host name configured under the **box** object. If no host name is configured, the system displays the default host name, micro4.

Table 6–45 (Cont.) Show System-Info Properties

Field	Description
Box identifier	This is a box identifier, used for licensing purposes. It is assigned during the manufacturing process.
Kernel version	The version of the kernel that is currently running on the system. Use the **show kernel-version** command for more details about the kernel build.
OS version	The version of the operating system running on the system.
OS uptime	The length of time since the last cold start.
LIBC version	The version of the GNU C Library currently running on the system.
Pthread version	The version of the POSIX Threads API, the standard for creating and manipulating threads, currently running on the system.
OpenSSL version	The version of OpenSSL, used for SSL and TLS connections, currently running on the system.
Export status	The cipher shipping status, either full version or export version. The export version does not contain the complete suite of TLS ciphers.
Database version	The version of the software used for managing all system databases currently running on the system.

show tcp

Displays local and remote TCP session state information, such as ESTABLISHED and LISTEN. The ME supports two types of ports: listeners and connections.

Sample Output

```
NNOS-E> show tcp
local                remote               state
-----                ------               -----
10.1.34.160:5060     0.0.0.0:0            LISTEN
10.1.34.160:5061     0.0.0.0:0            LISTEN
10.1.34.160:33617    10.1.34.13:5061      SYN-SENT
127.0.0.1:5432       0.0.0.0:0            LISTEN
172.26.0.155:22      0.0.0.0:0            LISTEN
172.26.0.155:22      172.26.3.63:3296     ESTABLISHED
172.26.0.155:22      172.30.1.2:4474      ESTABLISHED
172.26.0.155:80      0.0.0.0:0            LISTEN
172.26.0.155:5060    0.0.0.0:0            LISTEN
192.168.0.1:5132     192.168.0.2:33057    ESTABLISHED
```

Properties

The following table shows the properties for the show tcp command.

Table 6–46 Show TCP Properties

Field	Description
local	If the system is: **listener**: The IP address on the system used for the TCP connection. **initiator**: The system interface from which the connection originated.
remote	If the system is: **listener**: The remote IP address is 0.0.0.0 until a system connects to the listener. At that point, the remote IP becomes the address of the endstation connecting. **initiator**: The IP address of the endstation the system is connecting to.
state	The state of the TCP connection. See the following table for a description of each TCP state.

The following table describes each of the TCP states, as defined in *RFC 793, Transmission Control Protocol Specification*:

Table 6–47 TCP States

Fields	Description
LISTEN	waiting for a connection request from any remote TCP and port.
SYN-SENT	waiting for a matching connection request after having sent a connection request.
SYN-RECEIVED	waiting for a confirming connection request acknowledgment after having both received and sent a connection request.
ESTABLISHED	an open connection, data received can be delivered to the user. The normal state for the data transfer phase of the connection.
FIN-WAIT-1	waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.
FIN-WAIT-2	waiting for a connection termination request from the remote TCP.
CLOSE-WAIT	waiting for a connection termination request from the local user.
CLOSING	waiting for a connection termination request acknowledgment from the remote TCP.
LAST-ACK	waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request).
TIME-WAIT	waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request.
CLOSED	no connection state at all (and therefore never seen).

show tcp-skb-congestion-control

Displays the status of the TCP kernel buffer congestion control feature, including the admin state, current threshold, as well as some kernel buffer usage counters.

Sample Output

```
NNOS-E> show tcp-skb-congestion-control
admin:  enabled
threshold:  5000
skbs-in-use:  2096
max-skbs-in-use: 2506
tcp-dropped-pkts: 378
```

Properties

The following table shows the properties for the show tcp-skb-congestion-control command.

Table 6–48 Show TCP-SKB-Congestion-Control Properties

Field	Description
admin	Displays whether or not TCP kernel buffer congestion control is enabled.
threshold	Displays the configured threshold of the TCP kernel buffer congestion control.
skbs-in-use	Displays the current number of system-wide kernel buffers currently in use.
max-skbs-in-use	Displays the maximum number of kernel buffers that the ME has had in use at any given time.
tcp-dropped-pkts	Displays the number of TCP packets that have been dropped because the kernel buffer usage has exceeded the configured threshold.

show timezones

Displays a list of all preconfigured time zones recognized by the ME and their associated codes. (There are in excess of 1700 definitions known by the ME.) A system must be set to the time zone in which it is located so that time stamps and settings are correct in the software. If you type **set timezone ?** from the **box** object, the system displays about 30 of the most popular time zones. To display the code for a time zone other than one of the more common ones, use this command.

Sample Output

```
NNOS-E> show timezones

name                        codes
----                        -----
Africa/Abidjan              LMT GMT
Africa/Accra                LMT GHST GMT
Africa/Addis_Ababa          ADMT EAT
Africa/Algiers              PMT WEST WET CEST CET
Africa/Asmara               ADMT EAT
Africa/Asmera               ADMT EAT
Africa/Bamako               LMT GMT WAT
--More--
```

Properties

The following table shows the properties for the show timezone command.

Table 6–49 Show Timezone Properties

Field	Description
name	The geographic location of the time zone.
codes	The time zone abbreviation string.

show trap-categories

Displays the reported SNMP traps and their associated categories. You can then filter the SNMP traps by categories. The filter determines which categories of SNMP traps the ME sends out the WSDL interface to the external event server. Set the filter with the **event-group** object.

Sample Output

```
NNOS-E> show trap-categories

category   trap
--------   ----
csta       CallConnected
csta       CallCreated
csta       CallHeld
csta       CallRetrieved
csta       CallTerminated
csta       PlayComplete
csta       RecordComplete
dos       DosSIPPolicyTrap
dos       DosTransportPolicyTrap
dos       DosUrlPolicyTrap
h323      H323CallAlerting
h323      H323CallConnected
h323      H323CallCreated
h323      H323CallDisconnected
--More--
```

Properties

The following table shows the properties for the show trap-categories command.

Table 6–50 Show Trap-Categories Properties

Fields	Description
category	The SNMP trap category used to filter the traps going out the system to the external event server.
trap	The name of the trap. For example, DosTransportPolicyTrap indicates that a dynamic policy rule was instituted in response to a Transport Policy threshold being crossed.

show turn-allocations

Provides information for each TURN client allocated server relay port. WebRTC endpoints typically allocate a relay port for each media stream.

Sample Output

```
NNOS-E>show turn-allocations
```

```
server-port: 172.44.10.60:3478
user: TurnMike@TurnRealm
client: 10.1.26.32:56863
client-transport: UDP
relay-port: 172.44.10.60:20975
relay-transport: UDP
destination-count: 1
client-to-peer-packets: 61489
client-to-peer-bytes: 5822176
peer-to-client-packets: 61585
peer-to-client-bytes: 5512977
bandwidth-max: 150 kbits-per-second
allocation-time: 07:12:02.147705 Tue 2014-01-21
duration: 1177 seconds
remaining: 503 seconds
```

Properties

The following table shows the properties for the **show turn-allocations** command.

Table 6–51 Show Turn-Allocations Properties

Field	Description
server-port	The IP and port of the TURN server listener.
user	The user and realm of TURN LTC.
client	The IP and port of the TURN client.
client-transport	The transport method used for client/server communication.
relay-port	The IP and port of the TURN relay for this Allocation.
relay-transport	The transport method used for server/peer communication.
destination-count	The number of TURN destinations for this Allocation.
client-to-peer-packets	The number of packets relayed from client to peer for this Allocation.
client-to-peer-bytes	The number of bytes relayed from client to peer for this Allocation.
peer-to-client-packets	The number of packets relayed from peer to client for this Allocation.
peer-to-client-bytes	The number of bytes relayed from peer to client for this Allocation.
bandwidth-max	Currently not supported.
allocation-time	The time the Allocation was created and/or refreshed.
duration	The duration of the TURN Allocation.
remaining	The time remaining for the TURN Allocation.

show turn-destinations

Provides information for each TURN peer associated with a TURN client.

Sample Output

```
NNOS-E>show turn-destinations
index: 1
turn-client: 10.1.26.32:56864
```

```
turn-allocation: 0xd1c27f75
turn-relay: 172.44.10.60:20927
relay-transport: UDP
turn-peer: 172.44.10.60:20972
channel-number: 16384
chan-expire-time: 07:32:02.972915 Tue 2014-01-21
chanRemaining: 561 seconds
dest-permissions: Allowed
perm-expire-time: 07:32:02.972915 Tue 2014-01-21
permRemaining: 261 seconds
dest-anchored: true
```

Properties

The following table shows the properties for the **show turn-destinations** command.

Table 6–52 Show Turn-Destinations Properties

Field	Description
index	The index of this Destination. Note that an Allocation can have multiple destinations.
turn-client	The IP and port of the TURN client.
turn-allocation	The handle of the Allocation owning this Destination.
turn-relay	The IP and port of the TURN relay for this Destination.
relay-transport	The transport used for server/peer communication.
turn-peer	The IP and port of the TURN relay for this Destination.
channel-number	The TURN channel number for this Destination (a value of 0 means unused).
chan-expire-time	The time the TURN channel expires.
chanRemaining	The time remaining before the TURN channel expires.
dest-permissions	Permissions installed by the TURN client for this Destination.
perm-expire-time	The time Permissions expire.
permRemaining	The time remaining before Permissions expire.
dest-anchored	Indicates if media is anchored for this TURN relay.

show version

Displays version information for each application that makes up the software suite. This command is similar to the **show module-version** command, which displays for the kernel modules (the modules that are part of the operating system).

Sample Output

```
NNOS-E> show version
image    version    build        branch     time                      computer
-----    -------    -----        ------     ----
monitor  3.2.0      22831        b3.2.0     00:42:15 Sun 2006-12-17   AUTOBUILD
manager  3.2.0      22831        b3.2.0     00:43:10 Sun 2006-12-17   AUTOBUILD
SIP      3.2.0      22831        b3.2.0     00:58:29 Sun 2006-12-17   AUTOBUILD
media    3.2.0      22831        b3.2.0     00:59:40 Sun 2006-12-17   AUTOBUILD
reg      3.2.0      22831        b3.2.0     00:43:36 Sun 2006-12-17   AUTOBUILD
web      3.2.0      22831        b3.2.0     01:09:00 Sun 2006-12-17   autobuild
```

```
WS       3.2.0      22831        b3.2.0     01:12:00 Sun 2006-12-17  autobuild
```

Properties

The following table shows the properties for the show version command.

Table 6–53 Show Version Properties

Field	Description
image	The application name.
version	The software version installed on the system for the corresponding application.
build	The specific build of the software version for the corresponding application.
branch	The internal development tracking ID for the build.
time	The date and time of the build.
computer	The name of the computer system that built the image, typically AUTOBUILD.

show vrrp

Displays state and configuration information for each configured VRRP **vinterface**.

Sample Output

```
NNOS-E> show vrrp
vrrp-interface: vx112
  vinterface-admin-status: enabled
    active-host-interface: eth2
                vrrp-state: Master
                vrouter-id: 113
          hosted-priority: 255
active-host-intf-op-state: up
                vrrp-group: 2
                     bound: true
                host-index: 0
     activation-timestamp: 10:56:29 Thu 2006-12-21
                  duration: 0 days 01:17:57
```

Properties

The following table shows the properties for the show vrrp command.

Table 6–54 Show VRRP Properties

Field	Description
vrrp-interface	The name of the VRRP interface (the VX ID), configured with the **vinterface** object.
vinterface-admin-status	The administrative status of the virtual interface. When **enabled**, the referenced interfaces participate in the failover features of VRRP. When **disabled**, the interfaces do not serve as link backup.

Table 6–54 (Cont.) Show VRRP Properties

Field	Description
active-host-interface	A referenced Ethernet interface used as the host for this VRRP interface. Only the current Master VRRP interface lists as the active host; this field is blank for all Backups.
vrrp-state	The role or state of the VRRP interface, either Init, Master or Backup. The role is established by your order of entry. The first host is Master, and backups are ordered according to their position. Use the **move** command to change the order. A state of Init indicates that the host interface is not configured.
vrouter-id	The vrouter ID, as required by the VRRP specification. This identifier is calculated by the system by adding one to the vinterface value. For example, if the vinterface is vx2, the vrouter-id is 3.
hosted-priority	The priority of the VRRP interface, as defined by the *RFC 2338, Virtual Router Redundancy Protocol*. The Master interface must have a priority of 255. Backup interfaces have a value of one less than the interface above them in the configuration order.
active-host-intf-op-state	The operational state of the Master.
vrrp-group	The VRRP group of which the interface is a member. Grouping interfaces is a configuration technique to apply failover. A vinterface with a group number of 0 does not participate in grouping.
bound	A indication of whether the interface is Master. The output should be **true** for a Master interface and **false** for Backups.
host-index	An internal identifier.
activation-timestamp	The time at which the VRRP interface became the active Master.
duration	The length of time that the VRRP interface has been active Master.

show vrrp-hosts

Displays the status for each host configured for a VRRP interface. The information displayed includes active state, link status, priority, and failover timer setting for the **vinterface host-interface** property. You can have multiple local hosts for a vrrp interface (eth0, eth1, eth2, etc.); this status provider indicates the status of each host.

Sample Output

```
NNOS-E> show vrrp-hosts
         vrrp-interface: vx111
         host-interface: eth1
host-interface-op-state: up
                  bound: true
               priority: 255
   master-down-interval: 600
```

```
   activation-timestamp: 10:56:29 Thu 2006-12-21
               duration: 0 days 01:17:57
         vrrp-interface: vx112
         host-interface: eth2
host-interface-op-state: up
                  bound: true
               priority: 255
   master-down-interval: 600
   activation-timestamp: 10:56:29 Thu 2006-12-21
               duration: 0 days 01:17:57
```

Properties

The following table shows the properties for the show vrrp-hosts command.

Table 6–55 Show VRRP-Hosts Properties

Field	Description
vrrp-interface	The name of the VRRP interface (the VX ID), configured with the cluster>vrrp>**vinterface** object.
host-interface	A referenced Ethernet interface used as the host for this VRRP interface.
active-host-intf-op-state	The operational state of the Master.
bound	A indication of whether the interface is Master. The output should be **true** for a Master interface and **false** for Backups.
priority	The priority of the VRRP interface, as defined by the *RFC 2338, Virtual Router Redundancy Protocol*. The Master interface must have a priority of 255. Backup interfaces have a value of one less than the interface above them in the configuration order.
master-down-interval	A value calculated by the system and used to determine when the Master is determined to be down and the system should failover to the next configured interface.
activation-timestamp	The time at which the VRRP interface became the active Master.
duration	The length of time that the VRRP interface has been active Master.

show vx-bindings

Displays the binding between a VX interface and its associated Ethernet interface. The binding is configured using the **vinterface host-interface** property.

Sample Output

```
NNOS-E> show vx-bindings
vx-interface    ethernet
------------    --------

vx0             eth4
vx1             eth5
vx2             eth4
```

```
NNOS-E> show vx-bindings
vx-interface    ethernet
------------    --------

vx0             unbound
vx1             unbound
vx2             unbound
```

Properties

The following table shows the properties for the show vx-bindings command.

Table 6–56 Show VX-Bindings Properties

Field	Description
vx-interface	The name of the VX interface (the VX ID), configured with the VRRP **vinterface** object.
ethernet	A referenced Ethernet interface used as the host for this VX interface. A value of **unbound** indicates that the vinterface is not associated with an Ethernet interface.

show web-ext-status

The **show web-ext-status** command displays information about the web management interface configuration and activity.

```
NNOS-E>show web-ext-status

                    ip: 100.40.10.7
                  port: 80
              sessions: 0
          max-sessions: 30
   max-sessions-reached: 0
   session-idle-timeout: 30 minutes
           pool-threads: 1
            max-threads: 10
            connections: 0
       max-connections: 10000
idle-connection-timeout: 20 seconds
keep-alive-requests-max: -1
```

The following table shows the **show web-ext-status** properties.

Table 6–57 Show Web-Ext-Status Properties

Field	Description
ip	The web management interface IP address.
port	The web management interface port number.
sessions	The number of active sessions.
max-sessions	The configured maximum number of sessions allowed.
max-sessions-reached	The number of times a session was not created because the **max-sessions** value was reached.
session-idle-timeout	The configured session idle timeout.

Table 6–57 (Cont.) Show Web-Ext-Status Properties

Field	Description
pool-threads	The current number of request processing threads in the thread pool.
max-threads	The configured maximum number of request processing threads.
connections	The current number of connections.
max-connections	The configured maximum number of connections allowed.
idle-connection-timeout	The configured connection idle timeout.
keep-alive-requests-max	The configured maximum number of HTTP requests which can be queued before the connection is closed by the server.

show web-services-callout-details

Displays detailed information about web service process callouts. Callouts are external applications to which the ME makes requests for processing instructions. This information includes the location of these applications, information about failures, and requests received by the application.

Sample Output

```
SIP>show web-services-callout-details -v

                  endpoint: http://www.example.com:1001/events
                    server: external-services\event-group
application\event-service callprocessor
                   enabled: true
        failover-detection: none
              availability: available
           connect-timeout: 500 ms
              read-timeout: 2000 ms
             heartbeat-url:
        heartbeat-requests: 0
        heartbeat-failures: 0
heartbeat-response-minimum: 0 ms
heartbeat-response-average: 0 ms
heartbeat-response-maximum: 0 ms
    heartbeat-last-success:
     heartbeat-last-failed:
                  requests: 10
                  failures: 0
               io-failures: 0
         response-minimum: 4 ms
         response-average: 6 ms
         response-maximum: 7 ms
             last-success:
              last-failed:
           request-format: legacy
```

Properties

The following table shows the **show web-services-callout-details** properties.

Table 6–58 Show Web-Services-Callout-Details Properties

Field	Description
endpoint	The endpoint being called out to.
server	The name of the configured server.
enabled	Specifies whether or not the server is enabled to receive requests.
failover-detection	The type of failover detection being used for this endpoint.
availability	Specifies whether the endpoint is in a state to receive requests.
heartbeat-requests	The number of heartbeat requests made to the endpoint.
heartbeat-failures	The number of heartbeat requests that failed to reach the endpoint.
heartbeat-response-minimum	The fastest heartbeat response made by the endpoint.
heartbeat-response-average	The average heartbeat response made by the endpoint.
heartbeat-response-maximum	The slowest heartbeat response made by the endpoint.
heartbeat-last-success	The time at which the last successful heartbeat took place.
heartbeat-last-failed	The time at which the last failed heartbeat took place.
requests	The number of requests made to the endpoint.
failures	The number of requests that failed to reach the endpoint.
io-failures	The number of requests that failed to reach the endpoint due to I/O problems.
response-minimum	The fastest response made by the endpoint.
response-average	The average response made by the endpoint.
response-maximum	The slowest response made by the endpoint.
last-success	The time at which the last successful request took place.
last-failed	The time at which the last failed request took place.
request-format	The XML format used with this server.

show web-services-callout-status

Displays data for Web service requests made from the ME to a PC, including the endpoint involved, the specific request, and the number of times the request occurred. The output also displays response time statistics. This command only appears as available when the **web-service** server is configured and enabled.

Sample Output

```
NNOS-E> show web-services-callout-status

     type: event
 endpoint: http://172.10.10.10:80/covws/callouts
     sent: 0
 min-sent: 0 ms
 avg-sent: 0 ms
 max-sent: 0 ms
   failed: 10
```

```
min-failed: 507 ms
avg-failed: 1045 ms
max-failed: 5509 ms
```

Properties

The following table shows the properties for the show web-services-callout-status command.

Table 6–59 Show Web-Services-Callout-Status Properties

Field	Description
type	The type of callout the system made to the remote system, any of the following: event: The system sent an event. policy: The system requested a policy. location: The system requested a location.
endpoint	The URL of the remote web service that we sent the event to or requested information from.
sent	The number of successful transactions between the system and the web service.
min-sent	The fastest single transaction time between the system and a web service.
avg-sent	The average transaction time between the system and web services.
max-sent	The longest single transaction time between the system and a web service.
failed	The number of failed transactions between the system and the web service.
min-failed	The quickest time awaiting a failure response between the system and a web service.
avg-failed	The average failure response time between the system and a web service.
max-failed	The longest time awaiting a failure response between the system and a web service.

show web-services-client-status

Displays configuration information for clients that have contacted a web service endpoint implemented by the ME. This command only appears as available when the **web-service** server is configured and enabled. The output of this command displays one entry per client/endpoint combination. In the example below, client 172.30.0.210 contacted the templates service on the ME.

Sample Output

```
NNOS-E> show web-services-client-status
endpoint: /templates
  client: 172.30.0.210
   count: 9
```

Properties

The following table shows the properties for the **show web-services-client-status** command.

Table 6–60 Show Web-Services-Client-Status Properties

Field	Description
endpoint	The web services endpoint on the system that was contacted.
client	The IP address of the client that contacted the system.
count	The number of requests that the system received from web service clients.

show web-services-fault-status

Provides information about faults and exceptions that have occurred during the processing of web service requests. The faults are tracked for the endpoint on the ME that was invoked and the faults that have occurred on that endpoint.

Sample Output

```
SIP>show web-services-fault-status

endpoint             fault              count
--------             -----              -----
/mgmt                Connection reset       4
```

Properties

The following table shows the properties for the **show web-services-fault-status** command.

Table 6–61 Show Web-Services-Fault-Status Properties

Field	Description
endpoint	The endpoint that was invoked to process the request.
fault	The fault that occurred while processing the request.
count	The number of requests that faulted in this manner.

show web-services-ports

Provides information about the TCP ports that the web services server is listening on. These ports can be used by applications to make requests to the ME.

Sample Output

```
SIP>show web-services-ports

scheme address         port
------ -------         ----
http   10.138.236.36   8080
```

Properties

The following table shows the properties for the **show web-services-ports** command.

Table 6–62 Show Web-Services-Ports Properties

Field	Description
index	The index number of the port.
scheme	The protocol scheme supported.
address	The IP address where the TCP listener is bound.
port	The TCP port that the listener is bound to.

show web-services-request-status

Displays data for Web service requests made from a remote computer to the ME, including the endpoint involved, the endpoint service URL, and the number of times the request occurred. The output also displays response time statistics. This command only appears as available when the **web-service** server is configured and enabled. Note that the output displays an entry for each endpoint/function combination.

Sample Output

```
NNOS-E> show web-services-request-status

endpoint: /templates
 request: GetTemplate
   count: 8
     avg: 0 ms
     min: 0 ms
     max: 1 ms
endpoint: /templates
 request: GetTemplateFiles
   count: 1
     avg: 2 ms
     min: 2 ms
     max: 2 ms
```

Properties

The following table shows the properties for the show web-services-request-status command.

Table 6–63 Show Web-Services-Request-Status Properties

Field	Description
endpoint	The web services endpoint on the system that was contacted.
request	The type of request from the remote computer
count	The number of requests, of the type identified in the **request** field, the endpoint has received.
avg	The average response time for all requests of this type to this endpoint. A response time under one millisecond reports as zero.
min	The fastest response to a request of this type to this endpoint.

Table 6–63 (Cont.) Show Web-Services-Request-Status Properties

Field	Description
max	The slowest response to a request of this type to this endpoint.

show web-services-status

The **show web-services-status** command displays information about the **web-services** configuration and activity on the ME.

```
NNOS-E>show web-services-status

                          ip: 100.40.10.7
                        port: 8082
                    sessions: 0
                max-sessions: 10000
         max-sessions-reached: 0
         session-idle-timeout: 30 minutes
                 pool-threads: 0
                  max-threads: 10
                  connections: 0
              max-connections: 10000
      idle-connection-timeout: 20 seconds
      keep-alive-requests-max: -1
 cross-origin-requests-denied: 0
cross-origin-requests-allowed: 0
```

The following table shows the show web-services-status command properties.

Table 6–64 Show Web-Services-Status Properties

Field	Description
ip	The web-service IP address.
port	The web-service port number.
sessions	The number of active sessions on this web-service.
max-sessions	The configured maximum number of sessions allowed.
max-sessions-reached	The number of times a session was not created because the **max-sessions** value was reached.
session-idle-timeout	The configured session idle timeout.
pool-threads	The current number of request processing threads in the thread pool.
max-threads	The configured maximum number of request processing threads.
connections	The current number of connections.
max-connections	The configured maximum number of connections allowed.
idle-connection-timeout	The configured connection idle timeout.
keep-alive-requests-max	The configured maximum number of HTTP requests which can be queued before the connection is closed by the server.

Table 6–64 (Cont.) Show Web-Services-Status Properties

Field	Description
cross-origin-requests-denied	The number of CORS requests allowed.
cross-origin-requests-allowed	The number of CORS requests denied.

show web-services-virtual-hosts

Provides information about all virtual hosts configured on the ME.

Sample Output

```
NNOS-E>show web-services-virtual-hosts

name      state    applications-directory
----      -----    ----------------------
host1     STARTED  webapps
```

Properties

The following table shows the properties for the show web-services-virtual-hosts command.

Table 6–65 Show Web-Services-Virtual-Hosts

Field	Description
name	The name of the virtual host.
state	The state of the virtual host.
applications-directory	The directory where this virtual host's WAR files are located.

show web-services-virtual-host-application-parameters

Provides information about application context parameters configured on virtual hosts.

Sample Output

```
NNOS-E>show web-services-virtual-host-application-parameters

name                path         context-parameter-name context-parameter-value
----                ----         ---------------------- --------
m5apps.acmeasc.com  /citigroup   codec                pcmu
m5apps.acmeasc.com  /citigroup   flashVersion         1100
m5apps.acmeasc.com  /simring     contextConfigLocation
/WEB-INF/applicationContext.xml
m5apps.acmeasc.com  /simring     default.operator.sipUrl
sip:*17813284400@foo;postd=0
m5apps.acmeasc.com  /simring     default.timeOut.response 45
m5apps.acmeasc.com  /simring     default.voiceMail.sipUrl sip:*17813284444@foo
m5apps.acmeasc.com  /simring     Extract.sipUrl.replacement sip:*$0@foo
m5apps.acmeasc.com  /simring     Extract.user.group  2
```

Properties

The following table shows the properties for the show web-services-virtual-host-application-parameters command.

Table 6–66 Show Web-Services-Virtual-Host-Application-Parameters Properties

Field	Description
name	The name of the virtual host.
path	The path where this web application is deployed.
context-parameter-name	The name of the context-parameter for this servlet.
context-parameter-value	The value of the context-parameter for this servlet.

show web-services-virtual-host-applications

Provides information about all web applications configured on each virtual host on the ME.

Sample Output

```
name          path          display-name  state         applications-directory
available
----          ----          ------------  -----         ----------------------
davisapps.acmepacket.com                              STARTED       /cxc_
common/webapps true
davisapps.acmepacket.com /acme-packet  Acme Packet AXA-Technologies Web Phone
STARTED       /cxc_common/webapps true
davisapps.acmepacket.com /axa-tech     Acme Packet AXA-Technologies Web Phone
STARTED       /cxc_common/webapps true
davisapps.acmepacket.com /groupblast                  STARTED       /cxc_
common/webapps true
davisapps.acmepacket.com /rtpstats     Acme Packet RTP Stats STARTED       /cxc_
common/webapps true
davisapps.acmepacket.com /webphone     Acme Packet Web Phone STARTED       /cxc_
common/webapps true
```

Properties

The following table shows the properties for the show web-services-virtual-host-applications command.

Table 6–67 Show Web-Services-Virtual-Host-Application Properties

Field	Description
name	The name of the virtual host.
path	The path where this web application is deployed.
display-name	The display name used within the web application.
state	The state of the web application.
applications-directory	The directory where this virtual host's WAR files are located.

Table 6–67 (Cont.) Show Web-Services-Virtual-Host-Application Properties

Field	Description
available	The availability of the web application.

show web-services-virtual-host-application-servlets

Provides information on servlets configured for all web applications on each virtual hosts on the ME.

Sample Output

```
name             path             servlet          state            available
----             ----             -------          -----            ---------
davisapps.acmepacket.com /acme-packet   ConfigServlet   STARTED      true
davisapps.acmepacket.com /acme-packet    default       STARTED      true
davisapps.acmepacket.com /acme-packet    jsp          STARTED      true
davisapps.acmepacket.com /axa-tech     ConfigServlet   STARTED      true
davisapps.acmepacket.com /axa-tech      default       STARTED      true
davisapps.acmepacket.com /rtpstats     Acme Packet RTP Stats STARTED       /cxc_
common/webapps true
davisapps.acmepacket.com /webphone     Acme Packet Web Phone STARTED       /cxc_
common/webapps true
```

Properties

The following table shows the properties for the show web-services-virtual-host-application-servlets command.

Table 6–68 Show Web-Services-Virtual-Host-Application-Servlets

Field	Description
name	The name of the virtual host.
path	The path where the web application is deployed.
servlet	The name of the servlet.
state	The state of the servlet.
available	The availability of the servlet.

show web-services-virtual-host-application-servlets-parameters

Provides information on servlet parameter settings for each web applications on each virtual host on the ME.

Sample Output

```
NNOS-E>show web-services-virtual-host-application-servlet-parameters

name             path             servlet         init-parameter-name
init-parameter-value
----             ----             -------         -------------------
--------------------
davisapps.acmepacket.com /axa-tech       ConfigServlet   ascBaseUrl-disabled
https://davis:8443
davisapps.acmepacket.com /axa-tech       ConfigServlet   rtmpBaseUrl
rtmp://davis.acmepacket.com/live
```

```
davisapps.acmepacket.com /axa-tech       ConfigServlet   uriFormat
sip:{0}@davis.acmepacket.com
davisapps.acmepacket.com /groupblast     IGroupBlastService adminRole       admin
davisapps.acmepacket.com /webphone       ConfigServlet   ascBaseUrl-disabled
https://davis:8443
davisapps.acmepacket.com /webphone       ConfigServlet   rtmpBaseUrl
rtmp://davis.acmepacket.com/live
davisapps.acmepacket.com /webphone       ConfigServlet   uriFormat
sip:{0}@davis.acmepacket.com
```

Properties

The following table shows the properties for the show web-services-virtual-host-application-servlet-parameters command.

Table 6–69 Show Web-Services-Virtual-Host-Application-Servlet-Parameters Properties

Field	Description
name	The name of the virtual host.
path	The path where the web application is deployed.
servlet	The name of the servlet.
init-parameter-name	The servlet's init-parameter name.
init-parameter-value	The servlet's init-parameter value.

show web-services-virtual-host-deployable-applications

Provides information about deployable applications. deployable applications are web applications that the web services server has internally that it can deploy to a configured virtual host.

Properties

Table 6–70 Show Web-Services-Virtual-Host-Deployable-applications

Fields	Description
name	The name of the deployable application.
description	The description of the deployable application.
sample	The sample that is hosting the deployable application.

show ws-listener

The **show ws-listener** command displays information about the configured SIP over WebSocket listener port.

```
NNOS-E>show ws-listener
-------------------------------------------------------------------------
Process Intf Port Attempts Success Rejects Failed In-Flight Current
-------------------------------------------------------------------------
SIP     eth0 9080      5       5       0      0       0       0
-------------------------------------------------------------------------
```

The following table shows the show ws-listener command properties.

Table 6–71 Show WS-Listener Properties

Field	Description
Process	The process using this WS listener port.
Intf	The interface on which this WS listener port is configured.
Port	The port number on which this WS listener is configured.
Attempts	The number of times a remote client has attempted to reach this WS listener port.
Success	The number of times a remote client was successful in their attempt to reach this WS listener port.
Rejects	The number of times the ME rejected a remote client trying to access this WS listener port.
Failed	The number of times a remote client failed to access this WS listener port.
In-Flight	The number of in-flight sessions currently trying to access this WS listener port.
Current	The current number of sessions on this WS listener port.

show wss-listener

The **show wss-listener** command displays information about the configured SIP over WebSocket Secure (WSS) listener port.

```
NNOS-E>show wss-listener
------------------------------------------------------------------------
Process Intf Port Attempts Success Rejects Failed In-Flight Current
------------------------------------------------------------------------
SIP     eth0 9090      5       5       0      0       0       0
------------------------------------------------------------------------
```

The following table shows the show wss-listener command properties.

Table 6–72 Show WSS-Listener Properties

Field	Description
Process	The process using this WSS listener port.
Intf	The interface on which this WSS listener port is configured.
Port	The port number on which this WSS listener is configured.
Attempts	The number of times a remote client has attempted to reach this WS listener port.
Success	The number of times a remote client was successful in their attempt to reach this WSS listener port.
Rejects	The number of times the ME rejected a remote client trying to access this WS listener port.
Failed	The number of times a remote client failed to access this WSS listener port.

Table 6–72 (Cont.) Show WSS-Listener Properties

Field	Description
In-Flight	The number of in-flight sessions currently trying to access this WSS listener port.
Current	The current number of sessions on this WSS listener port.

7

Configuring Access Objects

The user access control defines the users that are allowed access to the ME device and the specific privileges that they are granted. There are two access points within the system for granting privileges. You can assign access system-wide, providing access to the entire box. This is done from the top level of the configuration hierarchy. Or, you can configure access to a specific VSP. This is done through the VSP configuration object.

System-wide users log in with their user name:

```
username: jdoe
```

VSP users log in with the VSP name followed by their user name:

```
username: cxc1\jdoe
```

Whether from the top level or within a VSP, the configuration is basically the same. The one major difference is in the RADIUS configuration. Because a VSP can already have a RADIUS server configured, you can simply reference that server from the VSP-level. When setting up RADIUS-based access from the top-level, you must also configure the server properties.

Directories

Each access point includes a set of user directories and a set of permission definitions. The user directory contains an authentication database of locally configured users. In addition, you can configure other authentication directory types, such as RADIUS.

> **Note:** The order in which you configure the directories establishes the order in which the ME checks directories for authentication. For example, if you want to override a users privileges as they are set in the RADIUS directory, configure the static users directory first.

If authentication succeeds, the permissions associated with that user are applied to all the subsequent operations.

access

Opens the access configuration object for editing. You can set access privileges from two points in the hierarchy, either:

- system-wide, from the top level of the CLI hierarchy
- per-vsp, from within each VSP object

Syntax

```
config access
config vsp access
```

Parameters

None

permissions

Opens or creates a set of permissions. From this object you can set access to a variety of box-level services.When a user successfully logs in, the ME applies the permissions associated with that user to all subsequent operations.

Note that enabling permissions is not the same as enabling the service. For more information on enabling services, see the following chapters:

Table 7–1 Services Chapters

Property	Chapter reference
CLI	Configuring CLI objects
ME Management System	Configuring Web objects
User portal	*Oracle Communications OS-E Management Tools*
Config	Throughout this manual
Status	Status provider show commands
Actions	Actions
Call logs	Configuring Master services objects
Templates	Configuring Web service objects
Web services	Configuring Web service objects
Debug	N/A

Enter a previously configured permissions set name to edit it or enter a new text string to create the permission set.

Syntax

```
config access permissions name
config vsp access permissions name
```

Parameters

cli: Sets permissions for users to access the CLI.

`Default: normal`

- Values: advanced: Allows full access to CLI commands.
- normal: Allows partial CLI access. When restricted, users do not have access to advanced functionality such as debug tools, shell, etc. Further access is dependent on the other properties set in this object (**config**, **status**, **actions**).
- disabled: Prohibits access to the cli.

Example: **set cli advanced**

cms: Sets access to the ME Management System.

Default: enabled

- **Values: enabled**: Allows access to the ME Management System
- **enabled-web-only**: Allows access to the ME Management System only.
- **disabled**: Prohibits access to the ME Management System.

Example: **set cms enabled-web-only**

user-portal: Sets the user access to the portal feature of the ME Management System. See the *Oracle Communications OS-E Management Tools* guide for complete information on this feature.

Default: disabled

- **Values: enabled**: Sets the user portal to display call and IM data. When **enabled**, and all other permission properties are disabled, the user is taken directly to the portal page when logging into the ME Management System. If other properties are enabled as well, the user is taken to the ME Management System home page, and the portal tab is available for selection.
- **enabled-advanced**: Sets the user portal to display session data, in addition to the standard call and IM data.
- **disabled**: Disables the user portal.

Example: **set user-portal enabled**

config: Sets access to system configuration commands. These commands are used to change the running configuration.

Default: enabled

- **Values: enable**: Allows full access to config commands.
- **view**: Allows users to view the system configuration, but prevents them from executing config commands.
- **disabled**: Prohibits access to config commands.

Example: **set config view**

status: Enables or disables the ability to execute system **show** status commands. These commands display various components of system status and data.

Default: enabled
`Values: enabled | disabled`
Example: **set status disabled**

actions: Enables or disables the ability to execute system actions. An action is a command that immediately acts on the ME and one of its components.

Default: enabled
`Values: enabled | disabled`
Example: **set actions disabled**

call-logs: Enables or disables access to the system accounting functions and call-log data. Accounting functions include RADIUS and Diameter accounting services, system logging (syslog), the accounting database, and the accounting file system. Call logs include user-specific sessions, whole sessions, and SIP message logs.

Default: enabled
`Values: enabled | disabled`
Example: **set call-logs disabled**

templates: Sets the ability to use the web services template API. Templates provide access to a bundled configuration process that simplifies the use of web services by automating aspects of the configuration. For example, you could create a template to automate provision of the ME devices. When **enabled**, the user can access the template interface; when **disabled**, the user cannot.

You must enable web-services permissions for access to the template API. Additionally, this permission provides read-only access. You must also enable other permissions (e.g., **config**, **status**, and **actions**) for full web services capabilities.

Default: enabled
Values: enabled | disabled
Example: **set templates disabled**

troubleshooting: Sets the ability to use the troubleshooting web service. The ME provides a troubleshooting web service that accesses the call database and sends troubleshooting requests to the ME device for call binding information. When **enabled**, the user can access the troubleshooting web service; when **disabled**, the user cannot.

Default: enabled
Values: enabled | disabled
Example: **set troubleshooting disabled**

web-services: Sets the ability to initiate WSDL requests through the web services management API. When **enabled**, the user can access the web service interface; when **disabled**, the user cannot. Note that this permission provides read-only access. You must also enable other permissions (e.g., config, status, and actions) for full web services capabilities. Enable template permissions for access to the **template** API.

Default: enabled
Values: enabled | disabled
Example: **set web-services disabled**

debug: Enables or disables the ability to access debug commands. When enabled, the user has shell and debug access; when disabled, the user does not. Typically, these commands, which are a licensed feature, are not for end-user use. If not licensed, the **debug** property does not display.

Default: disabled
Values: enabled | disabled
Example: **set debug enabled**

login-attempts: Specifies the maximum number of failed login attempts allowed by the ME device. When this value is reached the user is locked out until an administrator either configures a new password or executes the "login unlock" action for that user.

Default: unlimited
Values: Min: 3 / Max: 12
Example: **set login-attempts 3**

permitted-view: Assign a permitted view you want a user to have. If no permitted-view is specified, the default permitted view is set to **all**.

Default: all
- Values: all
- minimal
- basic
- secureAccessProxy
- secureMediaProxy

- lcs
- sametime
- imFederation
- e911
- phoneServices
- pstn
- csta
- security-admin
- security-operator
- sip-admin

Example: **set permitted-view security-admin**

config-filter: Select the existing config-filter you want to use for this permission set.

Default: There is no default setting
Example: **set config-filter filter1**

action-filter: Select the existing action-filter you want to use for this permission set.

Default: There is no default setting
Example: **set action-filter actionfilter1**

gui-tools-update-software: When **enabled**, this privilege allows users to use the **Update software** action under the **Tools** tab.

Default: enabled
Values: enabled | disabled
Example: **set gui-tools-update-software disabled**

gui-tools-upload-files: When **enabled**, this privilege allows users to use the **Upload license file** and **Upload file** actions under the **Tools** tab.

Default: enabled
Values: enabled | disabled
Example: **set gui-tools-upload-files disabled**

gui-tools-download-files: When **enabled**, this privilege allows users to use the **Retrieve license**, **Download file**, and **Download saved configuration file** actions under the **Tools** tab.

Default: enabled
Values: enabled | disabled
Example: **set gui-tools-download-files disabled**

directory-white-list: The directories that are allowed to be read from or written to by various services.

Default: There is no default setting
Example: **set directory-white-list /directory**

users

Opens the users directory for configuration. When setting up authentication through this directory, you statically add users, and their privileges, to the system authentication database. Alternatively, you can configure the ME to perform authentication via a RADIUS server with the **radius** object.

Syntax

```
config access users
config vsp access users
```

Properties

admin: Enables or disables access for statistically configured users.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

password-policy

Specifies the password requirements for locally configured users. It is through this object that you define string requirements, reusability, and expiration times.

Syntax

```
config access users password-policy
config vsp access users password-policy
```

Properties

duration: Specifies the length of time, in days, for which a password is valid. When a password expires, the ME prompts you to change it on your next log in and sends a message to the event log.

Default: unlimited
Values: Min: 1 / Max: 365; unlimited (the password never expires)
Example: **set duration 7**

minimum-length: Specifies the minimum number of characters allowed for a password.

Default: 4
Values: Min: 2 / Max: 64
Example: **set minimum-length 5**

character-types: Specifies the number of different character types allowed in a password. The character type choices are uppercase, lowercase, numeric, and other (anything non-alphanumeric).

Default: 1
Values: Min: 1 / 4
Example: **set character-types 3**

allow-sequences: Specifies whether the password can contain sequences or repeated characters. If set to **true**, any string is acceptable (if it meets the other property constraints). If set to **false** you cannot include a sequence or repeated character in a password. A sequence is considered two or more consecutive numbers or letters (ab, 67, or MN, for example). Characters are considered repeated only if they are directly next to each other (skiing would be invalid, banana would be allowed).

Default: true
Values: true | false
Example: **set allow-sequences false**

recycle-check: Specifies whether and when a password can be reused. If set to **disabled**, any password can be reused. Specifying a number indicates the number of

previous passwords that cannot be reused. For example, specifying four requires that a new password not be the same as any of the last four passwords.

Default: disabled
Values: enabled | disabled
Example: **set recycle-check enabled**

user

Adds the named user to the system authentication database and assigns a previously configured set of permissions.

Enter a user name for the entry; enclose the name in quotation marks if it contains spaces.

Syntax

```
config access users user name
config vsp access users user name
```

Properties

password: Configures a password for the named user. A password string must be at least four characters long.

Default: There is no default setting
Example: **set password admin**

confirm: **********

permissions: Associates a set of permissions with the named user. These permissions include access to a variety of box-level services. See the **permissions** object for details.

Enter a previously configured set of permissions.

Default: There is no default setting
Example: **set permissions vsp access permissions grantAll**

radius

Configures the ME to use a RADIUS server to perform user authentication and sets basic RADIUS functionality. For system-wide access use the group and server objects to define and identify the server. For VSP access, use the group property within this object. Alternatively, you can statically configure users for authentication and privileges via the users object.

> **Note:** The radius subobject is applicable to the access object whether you configure it from the top level of the CLI hierarchy or from within a VSP. However, the group property, which references a previously configured RADIUS group, is only available from within a VSP. When configuring RADIUS from outside of the VSP, you must create a new group and server.

Syntax

```
config access radius
config vsp access radius
```

Properties

admin: Enables or disables the RADIUS server authentication configuration. When enabled, the ME device forwards authentication requests to the specified RADIUS server.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

group: Specifies the RADIUS group that the ME uses for user authentication. A RADIUS group defines the authentication and accounting services associated with a group of RADIUS servers, configured using the VSP **radius-group** object. Enter a reference to a previously configured group.

Default: There is no default setting
Example: **set group "vsp radius-group mgmtEmployees"**

default permissions: Associates a set of permissions to apply if there are no specifically configured permissions in place. These permissions include access to a variety of box-level services. See the **permissions** object for details.

Enter a previously configured set of permissions.

Default: There is no default setting
Example: **set default-permissions vsp access permissions grantAll**

default-sip-address <regExp> <replacement>: Specifies the SIP address to use when displaying calls via the portal. When the portal is configured for a user, they only see their own calls in the ME Management System. In order to filter for the user, the ME needs to know the SIP address. This can be set on the RADIUS server. If there is not a SIP address defined for the user in the RADIUS server, the ME uses this property to generate a SIP address from the access user name.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting

- **Values: regExp**: Enter a regular expression identifying the portion of the attribute to match. For example, the following expression identifies a subexpression (between the parenthesis) that matches all names:

  (.*)

- **replacement**: Enter a string that defines how to recompose the resulting regExp string. The replacement string is what the ME searches on when displaying calls in the portal for that user. In the following example, the first component from the regular expression is substituted in place of the "1" and appended to "@company.com." \1@company.com

Example: **set default-sip-address (.*) \1@company.com**

group

Configures a RADIUS group allowing the ME device (the RADIUS client) to perform user authentication for user access. (To setup authentication of SIP traffic, use the VSP **radius-group** object.) Associate servers with the group using the **server** object.

This object is only available when configuring user access outside of the VSP. Specify the RADIUS group name using up to 16 alphanumeric characters with no blank spaces.

Syntax

```
config access radius group name
```

Properties

admin: Enables or disables the RADIUS authentication and accounting server configuration. When enabled, authentication and SIP call accounting records are forwarded to the specified server IP address and port numbers.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

accounting-mode: Sets the RADIUS group accounting operational algorithm.

Default: duplicate

- **Values: round-robin**: If you configure multiple accounting servers in the accounting group, the round robin algorithm performs continued accounting requests to primary and secondary servers until a valid accounting response is received.
- **duplicate**: The duplicate algorithm issues multiple duplicate accounting requests to all servers in the RADIUS accounting group. A duplicate accounting request uses the same client source IP address and source UDP port.
- **fail-over [retryNumber]**: If you configure multiple accounting servers, the failover algorithm forwards accounting requests to secondary servers should the current accounting server fail. You can specify up to 256 failover servers.

Example: **set accounting-mode round-robin**

authentication-mode: Sets the RADIUS group authentication operational algorithm.

Default: fail-over 3

- **Values: round-robin**: If you configure multiple authentication servers in the RADIUS group, the round robin algorithm performs continued authentication requests to primary and secondary servers until a valid authentication response is received.
- **fail-over <***retryNumber***>**: If you configure multiple authentication servers in the RADIUS group, the failover algorithm forwards authentication requests to secondary servers should the current authentication server fail. You can specify up to 256 failover attempts to other servers.

Example: **set authentication-mode round-robin**

type: Sets the type of SIP accounting record to use. Currently, the only valid SIP accounting record type is Cisco.

Default: cisco
Example: **set type cisco**

included-in-default: Specifies if this RADIUS group is to be included in the default RADIUS authentication and accounting target group.

If set to **true**, authentication and accounting requests are forwarded to this group if there are no configured policies that govern or redirect RADIUS requests to other servers.

Default: true
Values: true | false
Example: **set included-in-default false**

send-digest-contents: Specifies whether to include the SDP contents in the RADIUS Auth-Request message. If set to **true**, the ME does include the contents.

Note that this feature is for customized RADIUS use. If you enable it for a RADIUS server that does not support this option, the RADIUS server will then reject every RADIUS request.

Default: false
Values: true | false
Example: **set send-digest-contents true**

server

Identifies and defines the operating parameters of the RADIUS server(s) for a specified group. This object is only available when configuring user access outside of the VSP.

Enter the host name or IP address for your RADIUS server.

Syntax

```
config access radius group name server host
```

Properties

admin: Enables or disables the RADIUS authentication and accounting server configuration. When enabled, authentication and SIP call accounting records are forwarded to the specified server IP address and port numbers.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

authentication-port: Sets the UDP port number that the RADIUS client (the ME device) uses to send authentication requests to the RADIUS server.

Default: 1812
Values: Min: 1 / Max: 65535
Example: **set authentication-port 998**

accounting-port: Sets the UDP port number that the RADIUS client (the ME device) uses to send accounting requests to the RADIUS server.

Default: 1813
Values: Min: 1 / Max: 65535
Example: **set accounting-port 999**

secret-tag: Specifies the shared secret used to authenticate transactions between the ME device and the RADIUS server. The specified shared secret is never sent over the network.

Note that the secret you enter here is a shared secret, and must match the secret configured on the RADIUS server. Enter the string in alphanumeric characters. See Understanding Passwords and Tags for a description of the ME password handling.

Default: There is no default setting
Values: Min: 1 / Max: 32
Example: **set secret abc123xyz**

timeout: Specifies the time in milliseconds to elapse before an accounting or authentication request to a RADIUS server times out. If the request times out, the request is retried for the specified number of attempts before the request is forwarded to the next RADIUS server in the configuration or dropped.

Default: 1000
Values: Min: 1 / Max: 65535
Example: **set timeout 1500**

retries: Sets the number of times the ME retransmits an accounting or authentication request if the RADIUS server does not respond.

Default: 3
Values: Min: 2 / Max: 5
Example: **set retries 5**

window: Configures the maximum number of simultaneous RADIUS client requests (authentication and accounting) sent to the RADIUS server.

Default: 32
Values: Min: 1 / Max: 127
Example: **set window 115**

call-field-filter

Configures the specific fields of the call detail record that the ME should send to the target RADIUS server(s). See the **accounting call-field-filter** object description for complete details.

Syntax

```
config access radius group name call-field-filter
```

Properties

None

enterprise

Applies access permissions to a group that is already defined in an enterprise directory server.This could be a group created in any number of ways: for example, as part of the directory setup and inherited by the ME device or through the directories **group** object.

You can configure permissions for any number of groups through this object, but can only map a group to one set of permissions.

Syntax

```
config access enterprise
config vsp access enterprise
```

Properties

admin: Enables or disables the application of the specified permissions to the identified group.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

directory: Specifies the directory server from which the ME derives its user information. Enter a reference to a previously configured enterprise directory.

Default: There is no default setting

Example: **set directory active-directory employees**

group-mapping: Maps previously defined directory groups to a set of previously configured permissions. Enter a group name that is recognized on the specified **directory**. Enter a reference to the permissions (configured using the **permissions** object), enclosing the reference in quotation marks.

Default: There is no default setting
Example: **set group-mapping marketing "vsp access permissions viewOnly"**

permission-filters

This object allows you to apply action and configuration filtering on a per-user basis.

Syntax

```
config access permission-filters
```

Properties

config-filter: Applies configuration filtering on a per-user basis. Enter a name for the filter.

Default: There is no default setting
Example: **set config-filter filter1**

action-filter: Applies action filtering on a per-user basis. Enter a name for the filter.

Default: There is no default setting
Example: **set action-filter actionfilter1**

config-filter

This object applies configuration filtering on a per-user basis.

Syntax

```
config access permission-filters config-filter
```

Properties

admin: Enable or disable this configuration filter.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

filter: Specify the filter. Enter this value in free form, separating the class, object, and properties with a backslash "\".

Default: There is no default setting
Example: **set filter filter1 cluster\box\interface\ip**

action-filter

This object applies action filtering on a per-user basis.

Syntax

```
config access permission-filters action-filter
```

Properties

admin: enable or disable this action filter.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

filter: Specify the filter. Enter the action without any arguments. If you enter the action with arguments, the filter is ignored.

Default: There is no default setting
Example: **set filter restart**

grant-pattern

Configures the pattern to use to extract a privilege to grant.

Syntax

```
config access authorization attribute-grant <name> grant-pattern
```

Properties

name: Enter a descriptive name to give this grant.

Default: There is no default setting
Example: **set name DeskPhoneEvent**

pattern: Enter the regular expression pattern to use to define the attribute.

Default: There is no default setting.
Example: **set pattern "\+1\((\d{3})\)(\d{3})-(\d{4})"**

resource-identity: Select the type of matching to use to identify a resource-type.

Default: The default setting is equals.

- Values: equals <*value*>: The value that a user provides during an authorization request must be exactly the same as the resulting resource-identity. This is the default setting.
- matches <*expression*>: The value that a user provides during an authorization request is matched against the resource-identity using a regular expression match.
- any: Any value a user provides during an authorization request matches.

Example: **set resource-identity any**

regex-type: *Advanced property*. Specify the type of regular expression.

Default: The default setting is PCRE.

- Values: custom: Custom Regular Expressions and Replacements
- PCRE: Perl Compatible Regular Expressions and Replacements

Example: **set regex-type custom**

resource-type: Select the resource type that this extracted value represents.

Default: There is no default setting.

- Values: call

- call-recording
- call-monitors
- call-media-insertion
- event-channel
- registration
- sip-request
- file

Example: **set resource-type sip-request**

privileges: Select the CRUD privileges to allow for this resource-type.

Default: all

- Values: create
- retrieve
- update
- delete

Example: **set privileges create+retrieve**

default-grant

Configures default grants, which apply to all the ME users matching the specified resource identity.

Syntax

```
access authorization default-grant
access authorization group-grant <name> default-grant
```

Properties

name: Enter a descriptive name to give this grant.

Default: There is no default setting.

Example: **set name grant5**

resource-identity: Select the type of matching to use to identify a resource-type.

Default: The default setting is equals.

- Values: equals <*value*>: The value that a user provides during an authorization request must be exactly the same as the resulting resource-identity. This is the default setting.
- matches <*expression*>: The value that a user provides during an authorization request is matched against the resource-identity using a regular expression match.
- any: Any value a user provides during an authorization request matches.

Example: **set resource-identity any**

resource-type: Select the resource type that this extracted value represents.

Default: There is no default setting.

- Values: call
- call-recording

- call-monitors
- call-media-insertion
- event-channel
- registration
- sip-request
- file

Example: **set resource-type call-recording**

privileges: Select the CRUD privileges to allow for this resource-type.

Default: all

- Values: create
- retrieve
- update
- delete

Example: **set privileges create+retrieve**

group-grant

Configures default and attribute grants for specific groups. Group grants apply to users belonging to these groups and matching the resource-identity.

Syntax

```
access authorization group-grant
```

Properties

name: Enter the name of the group for which you are configuring this grant.

Default: There is no default setting.

Example: **set name engineering**

default-grant: Configures a default grant for this group.

attribute-grant: Configures an attribute grant for this group.

application-accessible: *Secondary property*. Indicate whether or not to expose this group value externally.

Default: true

Values: true | false

Example: **set application-accessible false**

8

Configuring Autonomous IP Objects

The autonomous IP objects allows you to define “islands” of IP subnets. Traffic from these subnets can then be excluded from passing through the ME (the system does not anchor the interaction). This might be desirable, for example, for phones within the same office. By implementing this feature, known as “smart anchoring,” you can ensure more efficient processing of media streams within identified subnets.

Note that to implement this feature you must set the **anchoring** property to **auto** in the **media** object of your session configuration. The auto feature allows the ME to determine whether or not to anchor a call.

Autonomous IP subnets are defined by an IP address and mask using the **private-group** object.

When a SIP call matches a source and destination subnet in a single autonomous IP location **private-group**, the ME does not anchor the call unless an explicit policy to do so takes precedence. If both the source and destination locations are unassigned (not in a configured subnet), the media anchoring decision follows the behavior defined in the default session config or other session config.

When hosts are behind the same firewall, the ME, by default, does not anchor the calls. However, you can configure the system so that it does anchor those calls by using the **private-group** object. Each endpoint behind a firewall has a public and private address. Initially, the ME examines the public address to see whether it is comprised within the address pool defined for a **private-group**. If there is a match, the ME then examines any private groups configured under the private gateway, and applies the same logic as the non-firewall traffic.

ME Anchoring Decision Process

The ME first determines membership in location groups and then executes the decision algorithm to decide whether to perform media anchoring. The caller and callee addresses may be defined within a configured subnet (location) group, in which case the membership is determined. Otherwise, the ME determines location membership through the location database.

A location is identified as a pair consisting of a public and private address. The *public address* is the address of either a private location gateway, if present, or a phone client address. The *private address* is the phone client address, which comes from the Contact header in the REGISTER request. If the location is behind a private location gateway (SIP proxy, firewall, etc.), then the public address is different from the private address.

Determining Location Group Membership

When the ME receives a call from a private gateway, it performs the following processing to determine location group membership:

1. The ME obtains the source location (source public address and private address) by looking up the From URI in the location database.
2. If the public address and private address are the same, the ME checks to see if this address is in a configured location group.
3. If the public address and private address are *not* the same, the ME determines the private location gateway in use and checks to see if this address is in a configured location group.
4. The ME obtains the destination location (destination public address and private address) by looking up the Request URI or the To URI in the location database.
5. If the public address and private address are the same, the ME checks to see if this address is in a configured location group.
6. If the public address and private address are *not* the same, the ME determines the private location gateway in use and check to see if this address is in a configured location group.

Determining Whether to Anchor

Once the ME establishes the location group, it runs a decision algorithm to determine whether to anchor the call. Source and destination groups are configured with the **private-group** object. The following summarizes the decision process:

1. If the source group and/or destination group are configured, but are different, then the ME anchors the media;
2. If the source and destination groups are configured and if they are the same, then the ME does not anchor the media;
3. If the source firewall and/or the destination firewall are configured, but are different, then the ME anchors the media;
4. If the source and destination firewalls are configured and are the same, then the ME does not anchor the media
5. If the source and destination public address are the same, then the ME does not anchor the media;
6. In all other cases, the ME anchors the media.

> **Note:** When an autonomous IP group is created, the subnet of 0.0.0.0/0 is added by default. When this configuration is activated and used in conjunction with auto-anchoring, all calls using this session-config will feature released (not anchored) media.

This default subnet persists even after a **delete** action is executed from the web interface unless other subnets are also configured within the group > subnet configuration object.

To disable this default subnet via the web interface, set a mask value of **/32**. For example:

```
vsp
autonomous-ip
```

```
group test
subnet[1] 0.0.0.0/32
connected true
self-connected true
```

autonomous-ip

Opens the object form which you configure autonomous IP subnets in order to implement smart anchoring. For smart anchoring to take effect, you must set **anchoring** to **enabled** in the **media** object of your session configuration.

Syntax

```
config vsp autonomous-ip
```

Properties

treat-tag-as-group: Enables tag routing for autonomous IP groups. When set to **true**, the ME treats the final routing tag (the tag with any changes that resulted from a matching session configuration) as an autonomous-ip **private-group** name. In that way, even if an endpoint does not fall within the subnets defined for the group, if it has a routing-tag that matches the VLAN group name, it is treated as part of the group.

Default: false
Values: true | false
Example: **set treat-tag-as-group true**

private-group

Creates a gateway for autonomous private IP groups. Use this object if you are configuring a SIP proxy, a firewall, or something similar that serves as the last hop from a phone client to the ME when the phone client registers. If the last-hop device has multiple public addresses, this object sets the address pool that covers the range of addresses. When the ME receives a call from a public address that is defined within the pool, it does not anchor the call.

Also this object adds subnets to the named group. If both the source and destination addresses are contained in one of the subnet entries in the group, and the **connected** property is set to **true**, no media anchoring applies. See the ME media anchoring decision process for a complete description of the determination process.

Syntax

```
config vsp autonomous-ip private-group name
```

Properties

subnet: Specifies the subnet(s) that you want to exclude from system anchoring. Within a single group, when both the source and destination are found within one of the configured subnets, the connection is a candidate for smart anchoring. Assuming **media** anchoring is set to **auto**, the **connected** property (below) is set to **true**, and no policy takes precedence, the system does not anchor the call.

You can enter multiple subnet addresses. The system processes the subnets in the order that they appear in the configuration. Use the global **media** command to re-order the subnets for processing.

Default: 0.0.0.0/32
Example: **set subnet 192.168.0.0/16**

connected: Specifies whether all members of the group can reach each other. If set to **true**, and all members are connected, the system does not anchor the media.

Default: true
Values: true | false
Example: **set connected false**

selfConnected: Specifies whether the system should anchor calls if they appear to have the same public and private IP address. (This may happen, for example, if both phones are behind a device that rewrites SIP headers, such as an ALG, and are NAT'd to the same public IP address.) By default (**true**), the ME device does not anchor those calls. When two phones are in the same group and have the same IP address, if this property is set to **false**, the ME anchors the call.

Default: true
Values: true | false
Example: **set selfConnected false**

9

Configuring BOOTP Client and Server Object

The BOOTP objects allow you to configure the Bootstrap Protocol (BOOTP) client and server settings in an ME network cluster. *Bootstrap Protocol*, described in RFC 951, is the Internet protocol that allows a network client to learn its own IP address and boot information from a BOOTP server.

In an ME network cluster, a BOOTP client (drone) requests its own IP address from the BOOTP server (master), as well as the IP address of the BOOTP server itself using the hardware MAC address. The BOOTP server responds to BOOTP client requests over the configured server UDP port.

If a BOOTP session cannot be established between the ME client and server, BOOTP closes the session across the BOOTP interfaces after 60 seconds.

Note: The BOOTP client and server objects are located in different places in the CLI hierarchy. You configure the client within the box object and the server within the interface object.

bootp-client

Opens the BOOTP client configuration object on the locally attached ME, or opens the BOOTP client configuration object on the specified ME device in a cluster configuration. You configure BOOTP clients from the box and cluster configuration objects. The box configuration object allows you to configure the BOOTP client on the locally attached ME device; the cluster configuration object allows you to configure the BOOTP client on individually numbered (indexed) ME devices in a network cluster.

Syntax

```
config box bootp-client
config cluster box <integer> bootp-client
```

Parameters

admin: Enables or disables BOOTP client services on the ME.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled** *client-port <ethX> <port-number>*: Sets the ME interface over which BOOTP sends requests to the BOOTP server (master). The client (drone) receives the IP address for this interface from the BOOTP server.

Default: eth0 68
- Values: Interface 0-19

- Port 1-65535

Example: **set client-port eth1 68** *server-port <port-number>*: Sets the ME interface over which BOOTP sends responses to the BOOTP client (drone). The server (master) sends IP addresses over this interface to the BOOTP client interfaces.

Default: 67
Values: Min: 1 / Max: 65535
Example: **set server-port 67**

bootp-server

Opens the BOOTP server configuration object on the specified IP interface. You configure BOOTP servers on an IP interface.

Syntax

```
config cluster box <number> interface ethX ip <name> bootp-server
config cluster box <number> interface ethX vlan number ip <name> bootp-server
config box interface ethX ip <name> bootp-server
config box interface ethX vlan <number> ip <name> bootpserver
```

Parameters

admin: Enables or disables the BOOTP server on the current ME Ethernet or VLAN interface.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

port <

port-number>: Sets the UPD port over which BOOTP sends responses to the BOOTP client (drone). The server (master) sends IP addresses over this port to the BOOTP clients.

Default: 67
Values: Min: 1 / Max: 65535
Example: **set port 300**

10

Configuring CLI Objects

The CLI object allows you to set the ME command line interface display settings.

cli

Sets the CLI top-level prompt and the output display lines. Set either the maximum number of lines to display in a single screen page or set to continuous scrolling. When you use paged output, the --More-- prompt operates on the following keystrokes:

- [Enter]: Displays the next line of text
- [Tab]: Displays the remainder of the text
- [Esc], Q, or q: No more text
- Any keystroke: Displays the next page of text

To temporarily change the CLI display mode without changing the default configuration, use the **display** action at the top level of the CLI hierarchy.

Syntax

```
config box cli
config cluster box number cli
```

Properties

prompt: Sets the top-level CLI prompt. Specify an alphanumeric string up to 64 characters. Use quotation marks to enclose prompt strings that include spaces.

Default: NNOS-E
Example: **set prompt boston1**

banner: Configures a banner that is displayed on all connected consoles.

Default: There is no default setting
Example: **set banner "system shutdown at noon"**

display: Sets the maximum number of lines to display on a CLI screen page or enables continuous page scrolling.

Default: 24

- Values: paged *integer;* Min: 5 Max: 500
- scrolled

Example: **set display paged 20**

11

Configuring Cluster, Box, and Interface Objects

The **cluster** object allows you to configure the Media Engine (ME) clusters. A cluster is a group of ME systems that operate within a real-time collaboration network. Systems defined in the cluster can provide synchronization and support failover operations. An ME cluster automatically distributes load among the cluster entities, and routes SIP sessions around any failed elements.

The **box** object allows you to configure the local box and interface settings. A local ME device is the hardware that you are managing over a local, Telnet, or Secure Shell (SSH) connection.

The **interface** object defines the physical interfaces: up to 20 Ethernet 1000 Mbps auto-negotiation, full-duplex interfaces (eth0 through eth 19).

For detailed information on clusters, refer to the *Oracle Communications WebRTC Session Controller Installation Guide*.

cluster

The cluster object allows you to enable media distribution between nodes within the cluster, creating media partners, using the **share-media-ports** property of this object. The pool of ports that participate in the distribution is defined using the **media-ports** object. From this object, you can open subobjects to first enable and configure ME devices and then Ethernet interfaces and their properties within a network cluster.

To set up load balancing across a cluster, see Configuring Head-End and Backing Interfaces. You create backing interfaces using the **sip** object.

For media distribution on a system outside of the cluster as a media partner, define media partners using the **partner** object.

Syntax

```
config cluster
```

Properties

name: Sets an administrative name for this cluster. Enter up 32 alphanumeric characters. For text strings with blank spaces, enclose the strings within quotation marks (“”).

Default: There is no default setting
Example: **set name “WebRTC cluster1”**

share-media-ports: Specifies whether or not to enable media distribution among the nodes within a cluster. When set to **true**, media partnering is enabled on the cluster. Media ports are defined for each interface using the **media-ports** object.

Default: false
Values: true | false
Example: **set share-media-ports true**

share-signaling-entries: Specifies whether or not all boxes in a cluster exchange active SIP session information. When set to **true**, they do exchange data. If the primary link then goes down, a backup link can use SIP session information from the primary box to handle existing calls.

This property should be set to **true** if you have configured VRRP (to provide the redundancy support).

Default: false
Values: true | false
Example: **set share-signaling-entries true**

backup-session-on-demand: Specifies the manner in which failover is handled by the system. By default (disabled), if signaling failover is configured the system immediately creates a backup session on the failover box when a session is established. When this property is **enabled**, the system does not create and mirror the session on the backup box. Instead, it maintains a minimum amount of information on the backup, and establishes the full session upon failover.

Default: disabled
Values: enabled | disabled
Example: **set backup-session-on-demand enabled**

share-turn-ports: Specifies whether or not to enable distribution of TURN ALLOCATE requests (i.e., load balance) across systems within a cluster. When set to **true**, TURN partnering is enabled on the cluster (the STUN service table contains STUN/TURN routes from all systems in the cluster). The ME performs a cluster-wide STUN service route lookup to determine the best ME device to handle the ALLOCATE request. If set to **false**, the table contains only those routes for the local box.

TURN servers are defined for each interface using the **stun-server** object. To load balance ALLOCATE requests, you must **enable** the TURN-redirector option of the **stun-server > port** property for one or more STUN servers. To determine route preference for load balancing, use the **stun-service-routing** object.

Default: false
Values: true | false
Example: **set share-turn-ports true**

mirror-media-streams: Specifies whether the cluster participates in media mirroring. When set to true, all boxes in the cluster share media state information. The selection of calls that are mirrored is determined by policy; you must also enable the **media** object **mirror** property.

Default: false
Values: true | false
Example: **set mirror-media-streams true**

media-resource-failure-timer: *Secondary property.* Sets how long to wait after a media proxy failure (in media proxy configurations). When **enabled**, if no backup media proxy takes ownership before the timer set with this property expires, the signaling sessions for the failed media proxy are terminated. Enter the value in the format displayed in the example or in standard W3C XML format (P*n*Y*n*M*n*DT*n*H*n*M*n*S).

Default: disabled

Values: enabled *days hh:mm:ss* | disabled
Example: **set media-resource-failure-timer enabled "5 days 12:00:00**

media-proxy-timeout: *Secondary property*. Specifies how long to wait for responses from the media proxy.

Default: 10
Values: Min: 1 / Max: 300
Example: **set media-proxy-timeout 25**

action-timeout-addon: *Secondary property*. Specifies how many milliseconds to add to the manager action timeout value.

Default: 0
Values: Min: 0 | Max: 300000
Example: **set action-timeout-addon 5**

box

Configures the settings and interfaces on the specified physical device. Specify a box number (integer) in the range 1 to 16 that makes this box unique among other boxes in the network cluster.

Until you configure a box, the ME does not list it as an available selection when you use the help. In the following example, the ME does not list box 3 as available until after you have opened the box 3 object.

```
config cluster> config box ?
 specify an object of type box
 1
 2
config cluster> config box 3
config box 3> return
config cluster> config box ?
 specify an object of type box
 1
 2
 3
```

Syntax

```
config box
config cluster box integer
```

Properties

admin: Enables or disables the specified box (number) within the cluster. When disabled, you can configure box properties, but the box cannot pass traffic.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

hostname: Assigns a network hostname or IP address to the box. This name can be seen in the configuration display as well as in network traffic data.

Enter a string, following the *RFC 922* or *RFC 1123* hostname naming standards (a string or an IP address).

Default: There is no default setting
Example: **set hostname NNOS-E-Boston**

timezone: Sets the time zone for the system. Use the help function to list the common values (**set timezone ?**). Or, to see the full list of supported time zones, see the file /usr/share/zoneinfo. If you enter an unsupported time zone, the system sends an error to the event log.

When interpreting a time zone, the system uses the Linux standard. For time zones east of GMT, use GMT-minus-hours; for time zones west of GMT use GMT-plus-hours. For example, you would configure **timezone** as **GMT-minus-1** if GMT is one hour earlier than your local time.

Default: Eastern
Example: **set timezone GMT-plus-5**

> **Note:** After changing the timezone on the ME, you may experience a discrepancy between the system time and the call logs database time. Oracle recommends performing a restart warm after a timezone change.

name: Assigns an administrative name for this system. Enter the name using up to 32 alphanumeric characters. For strings containing blank spaces, enclose the string within quotation marks (“”).

Default: There is no default setting
Example: **set name bostonMaster**

description: Assigns a user-defined text description to this system. Enter the description using up to 3p2 alphanumeric characters. For strings containing blank spaces, enclose the string within quotation marks (“”).

Default: Eclipse
Example: **set description “NNOS-E network master for company HQ”**

contact: Identifies the name of a contact person for this system. Enter the contact description using up to 32 alphanumeric characters. For strings containing blank spaces, enclose the string within quotation marks (“”).

Default: There is no default setting
Example: **set contact “Bob at extension 123”**

location: Identifies the physical location of this system. Enter the location description using up to 32 alphanumeric characters. For strings containing blank spaces, enclose the string within quotation marks (“”).

Default: There is no default setting
Example: **set location “Data center, 2nd floor”**

identifier: Assigns a MAC address to the eth0 interface of this box. This address is used to identify the box and associate the configuration with the correct physical appliance. You only assign a MAC address to boxes in a cluster; a standalone does not require this setting.

You must change the value for the configuration to be recognized and the box to be operational.

Default: 00:00:00:00:00:00
Example: **set identifier 0f:1c:22:cd:d1:32**

transcoding-threads: Sets the number of SIP stack processing threads that should be used for transcoding. Typically, the number of threads should match the number of system processors.

Default: 4

Values: Min: 1 / Max: 8
Example: **set transcoding-threads 8**

recording-socket-threads: Sets the number of SIP stack processing threads that should be used for servicing the recording sockets. Typically, the number of threads should match the number of system processors. (The automatic setting is based on that value.) Changes to this setting only take place after a system restart. See Using Automatic Values for more information.

Default: automatic
Values: automatic | *threads*
Example: **set recording-socket-threads 4**

dos-rule-source-limit: Sets the number of rules that can be evaluated strictly on source IP address. These source-only rules are evaluated before other kernel rules, and can provide faster evaluation under a heavy DOS attack.

Default: 1000
Values: Min: 0 (none) | 1000
Example: **set dos-rule-source-limit 50**

rtp-mixing-action-threads: *Secondary property*. Configures the number of threads that the worker pool will contain when the **mix-session-threaded** action is executed.

Default: automatic (The ME uses platform-specific factory default value)
Values: automatic | <*integer*>
Example: **set rtp-mixing-threads automatic**

interface

Configures the ME Ethernet network interfaces. You can configure up to 20 gigabit Ethernet, full-duplex interfaces. The actual number available depends on your hardware configuration.

It is from the interface object that you configure VLAN and IP settings:

- Configuring VLAN objects
- Configuring IP objects

Syntax

```
config box interface ethX
config cluster box integer interface ethX
```

Properties

admin: Enables or disables the administrative state of this Ethernet interface.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

mtu: Set the maximum transmission unit (MTU) in bytes for Ethernet packets transmitted over this interface.

Default: 1500
Values: Min: 100 / Max: 1500
Example: **set mtu 1000**

arp: Enables the Address Resolution Protocol (ARP) on this interface. ARP is the Internet protocol that maps real IP addresses to corresponding Ethernet addresses.

Default: enabled
Values: enabled | disabled
Example: **set arp disabled**

speed: Sets the speed of the Ethernet connection between the system and the piece of equipment to which it is connected. The system ignores this value if **autoneg** is set to enabled.

Default: 1Gb
Values: 10Mb | 100Mb | 1Gb
Example: **set speed 100Mb**

duplex: Sets the acceptable duplex method for the interface, either half (asynchronous) or full (simultaneous) transmission. The system ignores this value if **autoneg** is set to enabled.

Default: full
Values: half | full
Example: **set duplex half**

autoneg: Sets whether autonegotiation is enabled or disabled. If enabled, the **speed** and **duplex** settings are ignored. The system negotiates with the piece of equipment to which it is connected to achieve optimal agreed upon settings.

Default: enabled
Values: enabled | disabled
Example: **set autoneg disabled**

file-client

Configures the ME to recognize an intermediary (proxy) for file fetch operations. By specifying a match criteria for the proxy configuration, you can set the ME to send different types of file requests to different proxies.

Syntax

```
config box file-client
config cluster box integer file-client
```

Properties

proxy<reg-exp>[host][port]: Configures the proxy that the system uses for file fetch operations. Enter a regular expression to match the target URL. Use this, for example, to send FTP requests through one proxy (match ftp://) and HTTP requests through another (match http://). To send all file requests through one proxy, match all (.*).

Optionally, you can set a host name or IP address, and a port. If you set no host for a proxy match, that type will use no proxy.

Default: There is no default setting
Example: **set proxy ftp://192.168.10.10**

os

Specifies whether the ME should offload incoming packets to a different CPU and the verbosity level of the information from a kernel “panic.” In addition, you can enable compression to allows at least 40% more storage for kernel dumps.

Note: Do not modify these values unless told to do so by Technical Support.

Syntax

```
config box os
config cluster box integer os
```

Properties

rx-queue-offload: Specifies whether incoming packets should be offloaded to a different CPU. Do not change this property.

Default: none
Values: none | hyperthread-pair | alternate-cpu
Example: **set rx-queue-offload alternate-cpu**

ip-frag-queue-control: Specify the number of milliseconds the AA-SBC holds onto an IP fragment when the first fragment has not been received.

Default: 5
Values: Min: 1 / Max: 20
Example: **set ip-frag-queue-control 15**

crash-dump-compression: Specifies whether to enable GZIP compression for the dump facility. When **enabled**, the system compresses the dump image, allowing a greater number of active pages into the raw partition. This provides at least 40% more storage for kernel dumps.

Default: enabled
Values: enabled | disabled
Example: **set crash-dump-compression disabled**

crash-dump-level: Specifies the level of information to capture when a kernel panic occurs. The dump facility captures kernel logs, kernel memory, task states, and trace information. This information is written to a file, which can later be used for debugging. The kernel header, written in all levels except disabled, contains the following information:

- build time, time of the crash, kernel version, etc.
- In-memory kernel logs that have not yet been written to disk
- Kernel call trace details
- System task states

Use the **show faults** command to display the name of the file the system created as a result of the kernel panic.

Default: header-kernel-page-mem

- **Values: disabled**: Kernel dumping is disabled. If the kernel crashes, no extra processing occurs.
- **header**: The ME prints out dump information to the dump header and then writes it to the dump file.
- **header-kernel-page-mem**: The ME writes out the dump header all kernel memory pages to the dump file.
- **header-all-mem-except-free-pages**: The ME writes out the dump header and all kernel and user memory pages to the dump file.

- **all-mem**: The ME writes out the dump header and all conventional/cached memory (RAM) pages in the system (kernel, user, and free).

Example: **set crash-dump-level header**

arp-thresholds: Specifies the number of directly connected hosts (ARP entries in the cache) the system supports. Each of the three thresholds initiates an increasingly aggressive ARP cache action.

Default: 128 512 1024

- *Values: threshold1*: Specifies the point at which the system attempts to purge outdated ARP entries. Enter a value from 64 to 1,024.
- *threshold2*: Specifies the point at which the system aggressively tries to purge outdated ARP entries, making room for new entries. Enter a value from 128 to 4,096.
- *threshold3*: Specifies the maximum number of ARP entries the system supports. Any additional learned ARP entries are silently discarded until the cache entries drop below this level. Enter a value from 512 to 8,192.

Example: **set arp-thresholds 256 512 2048**

tcp-skb-congestion-control: Sets a threshold of system-wide kernel buffer usage before the ME proactively prevents the depletion of the remaining system resources by dropping all received TCP packets. When enabled, TCP packets will be dropped until the kernel buffer usage falls below the configured threshold.

Default: disabled and automatic

Values: enabled | disabled <*threshold*>

Example: **set tcp-skb-congestion-control enabled 300000**

receive-packet-steering: Enables Receive Packet Steering (RPS), which distributes the processing of received packets across available CPUs in the system and provides greater performance during high call rates.

Default: enabled

Values: enabled | disabled

Example: **set receive-packet-steering disabled**

receive-flow-steering: Enables Receive Flow Steering (RFS), which associates receive packets with a particular flow. Unique flows are defined by a packet's IP addresses and ports and are assigned CPUs for processing. With RFS enabled, the processing of receive packets is distributed to the CPU that is handling that flow, providing greater performance during high call rates. The rx-flow-entries property controls the size of the Rx flow table. This value represents the maximum number of concurrent flows supported by the system. The default, automatic allows the system to automatically configure the number of entries for optimum performance. You have the option to override the automatic setting and configure a specific number of Rx flow entries.

Default: enabled automatic

Values enabled | disabled

Example: **set receive-flow-steering enabled automatic**

transmit-packet-steering: Enables Transmit Packet Steering (TPS), which distributes the processing of transmitted packets across the available CPUs in the system and provides greater performance during high call rates.

Default: enabled

Values: enabled | disabled

Example: **set transmit-packet-steering disabled**

ethernet-rx-ring-size: Specifies the Rx ring size.

Default: automatic
Values:

- default-value: The Rx ring size does not change from its default setting.
- automatic: Each Ethernet ring size is optimized for better performance.
- custom: You specify the Ethernet Rx ring size. If the custom

Example: **set ethernet-rx-ring size default-value**

media-partners

Opens the object through which you identify the partners, other appliances outside of the cluster, for media distribution.

Media distribution through the **media-partners** object allows you to configure an appliance outside of the cluster as a media partner. The media partner system does not perform any SIP signaling; it has only media interfaces and handles media traffic. It offloads media anchoring to another appliance; the cluster does load balancing across its specified list of media partners.

Each cluster can specify one or more media partners. Multiple clusters can use the same media partner system(s).

For media distribution within a cluster, use the **share-media-ports** property of the **cluster** object.

Syntax

```
config cluster media-partners
```

Properties

None

partner

Configures a partner, outside of the cluster, to take part in the media distribution system. See **media-partners** for a description of media distribution.

Enter the IP address of the partner to configure when opening this object.

> **Note:** The protocol, port, and certificate set within this object must match the values set for these properties in the IP messaging object. If they do not match, the systems will not be able to communicate.

Syntax

```
config cluster media-partners partner ipAddress
```

Properties

admin: Enables or disables the specified partner. When disabled, you can configure properties, but the partner cannot participate in shared anchoring.

Default: enabled
Values: enabled | disabled
Example: set admin disabled

protocol: Specifies the protocol that the partners use to communicate.

Default: tcp
Values: tcp | tls
Example: **set protocol tls**

port: Sets the port number to connect to on the other media partners.

Default: 5132
Values: Min: 1 / Max: 65535
Example: **set port 6550**

certificate: Assigns the certificate that these media partners present if the protocol is set to TLS. Enter a reference to a previously configured certificate, used by all members of this partnership.

Default: There is no default setting
Example: **set certificate vsp tls certificate nnos-e.companyA.com**

media-anchor-limits

Opens the object through which you set the number of active ports available for media anchoring.You can set limits on a per-box and per-card basis, preventing an individual card from becoming oversubscribed.

Syntax

```
config box media-anchor-limits
config cluster box integer media-anchor-limits
```

Properties

None

port-limit

Sets per-processor limits on the number of ports (and thus, the number of active calls) available for media anchoring at any one time. You set limits for each processor on each “card” in the system.

You can configure the ME to use ports on the **cxc** processor. Do not configure media ports on ME ports (typically eth0 through eth3).

The limits you set with this object apply to the total ports on all Ethernet interfaces for the given processor. Use the **show media-ports-process-units** command to view the limits and configuration for each processor. In the example below, across interfaces eth0 though eth3 there are a total of 10,000 media ports allocated.

```
NNOS-E> show media-ports-process-units
card process-unit busy-threshold full-limit total      active
---- ------------ -------------- ---------- -----      ------
CXC  CXC          0              16000      10000      0
```

Media ports are assigned to an IP interface via the **media-ports** object, which defines the starting port number and total ports available for media anchoring on a given interface. By setting port limits (this object), you are defining the total number of ports that can be simultaneously *active* for each processor on the appliance.

Note: For purposes of calculating port requirements, typically one anchored voice call requires four ports.

To open the object, enter the card followed by the applicable processor.

Syntax

```
config box media-anchor-limits port-limit {all | CXC} {all | CXC}
config cluster box integer media-anchor-limits port-limit {all | CXC} {all | CXC}
```

Properties

busy-threshold: Specifies the point at which the system considers the processor busy, and finds a less-congested processor, if available.This is a "soft" limit that serves to distribute anchoring across processors for best performance. For the **cxc** processor, the default is 0 so that all media anchoring will be offloaded if other processors are available.

Default: 0 for the cxc processor; 12000 for all others
Example: **set busy-threshold 14000**

full-limit: Specifies the maximum allowable active ports for a processor. When this value is reached, no new calls can be established on the effected processor. Note that the total number of ports allocated for a processor, as displayed with the **show media-ports-processor-units** command, may be higher than this value, as those ports are not all active. The total port count may also be lower than this limit, in which case the active ports will not exceed the total.

Default: 16000
Example: **set full-limit 18000**

packet-discard

The IP discard packet logging feature allows you to enable a discard to be logged for UDP, TCP, and "Other," with an option to log which specific ports were hit within the previous configured scanning interval for the UDP and TCP packets.Within a given scan interval, the header of the first eight discarded packets are logged. The total discarded packets are counted and logged at the end of the scanning interval.

This feature is configured via the packet-discard object. If an IP interface has media-ports configured, you must first disable the media-ports>idle-monitor property, before the packet-discard object can be enabled. See the media-ports object for information on how to do this.

Syntax

```
config cluster box interface ip packet-discard
```

Properties

admin: Enable or disable the packet discard feature.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

track-port: When enabled, this will log an additional type of message with the list of ports that were hit within the logging interval.

Default: disabled
Values: enabled | disabled
example: **set track-port enabled**

scan-interval: The interval in seconds between reading and logging the latest discarded packet information.

Default: 60
Values: Min: 10 / Max: 86400
Example: **set scan-interval 70000**

lcr-import-service

Configures route-server import services settings

Syntax

```
config cluster box interface ip lcr-import-service
```

Properties

admin: Enable or disable the use of route-server import service over the selected interface. If enabled, the ME functions as an route-server import services server.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

protocol: Sets the protocol to use for route-server import operations. After setting a protocol, you can select the route-server import server's listening port (or accept the default). This is the port the server listens on for HTTP(S) requests.

Enter the **protocol** and **port**.

Default: http 8082

- Values: HTTP: Use for unencrypted transmission
- HTTPS: Use for secure transmission of web pages by using HTTP over SSL

Example: **set protocol https 8787**

max-threads: Specifies the maximum number of total worker threads, both active and spare (idle), allocated to the route-server import service server.

Default: 10
Values: Min: 1 / Max: 50
Example: **set max-threads 25**

min-spare-threads: Specifies the minimum number of inactive threads that the ME must leave allocated to the route-server import service server. When the ME removes idle threads, it must leave this number available.

Default: 1
Values: Min: 0 / Max: 50
Example: **set min-spare-threads 5**

max-spare-threads: Specifies the maximum number of inactive threads the ME can leave allocated to the route-server import service server. When the ME detects idle threads, it will not use more than this number for the route-server.

Default: 5

Values: Min: 0 / Max: 50
Example: **set max-spare-threads 10**

idle-timeout: Specifies an inactivity timeout, in minutes, for the ME GUI. When a session has been inactive for this amount of time, the ME logs the user off the system.

Default: 30
Values: Min: 0 / Max: 4294967296
Example: **set idle-timeout 50**

ciphers: Specifies SSL cipher suite names separated by commas.

Default: There is no default setting
Example: **set ciphers SSL_RSA_WITH_RC4_128_SHA_TLS_RSA_WITH_AES**

use-https-for-file-copy: When enabled, the route-server import service sends the route-server request to the Web server and specifies using HTTPS to download call rates file. When disabled, HTTP is used for downloading.

Default: enabled
Values: enabled | disabled
Example: **set use-https-for-file-copy disabled**

authentication: Specifies the HTTP SSL certificate sent to the route server for route server import service. Set to **basic** to use the default certificate. Set to **certificate** to specify a custom certificate that has to be in PKCS12 format.

Default: basic
Values: basic | custom <*certificate*>
Example: **set authentication custom cert1**

arp-heartbeat

Allows each VX to be associated with another system on the network. In addition to sending periodic VRRP advertisements across the messaging interface, this object allows the ME to send ARP requests across the VX interface to the associated timeout.

If the ARP probe times out, the master ME knows that its connection to the network is broken. It then transitions into a "link-down" state and notifies the backup ME to take over. While in the link-down state, the ME periodically sends an ARP probe to continue checking the link. The ME stays in this state until it gets an ARP response. The ME is then able to take over the VX again if required.

Syntax

```
config cluster box interface ip arp-heartbeat
```

Properties

admin: Enables or disables the ARP heartbeat on a redundant VM system or in a complicated network.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

heartbeat-system: Enter the IP address of the system on the subnet to query with ARPs.

Default: There is no default setting
Example: **set heart-beat-system 10.10.10.10**

12

Configuring Condition List Objects

Condition lists help define the criteria by which decisions are made as to how to manipulate or forward calls. You can set conditions to determine policy application and to select applicable dial or registration plans (and their components).

The policy configuration object allows you to create policies that govern the routing of SIP phone calls and instant messages to recipients, and then back to the original caller or sender. A policy is identified by a unique name and can contain one or more rules. With each rule, you configure a condition list with a set of properties, (as well as the session configuration properties).

Both dial and registration plans use condition lists as a "first pass" filter when matching a plan entry. The ME compares an incoming request against the configured conditions, and returns a list of matching plan entries. The match statements within the plan components then determine the final selection and/or alteration.

The following **registration-plan** components use condition lists:

- normalization
- arbiter
- route

The following **dial-plan** components use condition lists:

- normalization
- arbiter
- route
- source-route

The following **calling-groups** components use condition lists:

- route
- source-route

The following objects in the carriers **switch** and **trunk-group** subobjects use condition lists:

- outbound-normalization
- inbound-normalization

The following server-pool **server** subobjects use condition lists:

- outbound-normalization
- inbound-normalization

In addition, the condition list selection is available as a match component for URI matching that defines to which calls the ME should apply the configured plan. See Using the Match Properties for more information.

condition-list

Defines the conditions for policy, dial plan, or registration plan matching. If conditions match, the settings of the session configuration apply. Otherwise, default session configuration settings apply. Match statements in a condition-list include matching messages; exclude statements omit them. In some cases, you can specify a substring of a regular expression, which if found, will be a match. Specify a property and a regular expression or value for the corresponding variables. You must re-execute the command for each field you want included.

See Using Relational Operators for definitions of the operators used in condition construction.

Syntax

```
config vsp carriers carrier name gateway name inbound-normalization name
condition-list
config vsp carriers carrier name gateway name outbound-normalization name
condition-list
config vsp carriers carrier name gateway name trunk-group name
inbound-normalization name condition-list
config vsp carriers carrier name gateway name trunk-group name
outbound-normalization name condition-list
config vsp dial-plan normalization name condition-list
config vsp dial-plan arbiter name condition-list
config vsp dial-plan route name condition-list
config vsp dial-plan fork name condition-list
config vsp dial-plan voice-mail name condition-list
config vsp calling-groups group name route name condition-list
config vsp calling-groups group name source-route name condition-list
config vsp enterprise servers serverType serverName server-pool server serverName
outbound-normalization name condition-list
config vsp enterprise servers serverType serverName server-pool server serverName
inbound-normalization name condition-list
config vsp policies session-policies policy name rule name condition-list
config vsp registration-plan normalization name condition-list
config vsp registration-plan arbiter name condition-list
config vsp registration-plan route name condition-list
config vsp route-server-config route-server-sequence name query name
condition-list
```

Properties

operation: Specifies the decision operation to use (AND/OR) should a condition match occur in the SIP call session.

If multiple condition matches occur, the AND operation applies the matching rule with the highest precedence. The OR operation can use any of the matched conditions.

Default: and
Values: and | or
Example: **set operation or**

mode: Sets how the system applies the condition list. If set to **evaluate**, the system runs the conditions to determine whether or not to apply the session configuration

settings. If set to **always-true**, the system applies the session configuration settings, no conditions need to be configured. You may want to use this setting, for example, to apply session settings for a configured group without making up conditions to match the group.

Default: evaluate
Values: evaluate | always-true
Example: **set mode always-true**

sip-message-condition: See SIP Message Condition Options.

from-uri-condition: See From, To, and Request URI Condition Options.

to-uri-condition: See From, To, and Request URI Condition Options.

request-uri-condition: See From, To, and Request URI Condition Options.

date-time-condition: See Date and Time Condition Options.

action-condition: Specifies whether the configured rule is applied to normal SIP traffic or to a specific action. Select **none** to apply the rule to SIP traffic or select one of the supported actions.

Default: none
Values: none | call-control | presence-subscribe | presence-end-subscription
Example: **set action-condition call-control**

SIP Message Condition Options

Matches on general fields within the received SIP message.

Properties

box: Includes or excludes messages based on the ME box number within the cluster on which this SIP message was received.

Values: <eq | ne | gt | lt | ge | le> *number*
Example: **set sip-message-condition box exclude 3**

vsp: Includes or excludes messages based on the VSP on which this SIP message was received.

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition vsp match 1**

to-uid: Includes or excludes messages based on the identity of the recipient of the packet. The UID is a system-generated number that assigned to all SIP users known to the system. You can view this value using the ME Management System accounting/users screen.

Values: <eq | ne | gt | lt | ge | le> *UID*
Example: **set sip-message-condition to-user gt 013**

from-uid: Includes or excludes messages based on the user identity of the sender of the packet. The UID is a system-generated number that assigned to all SIP users known to the system. You can view this value using the ME Management System accounting/users screen.

Values: <eq | ne | gt lt | ge | le> *UID*
Example: **set sip-message-condition from-user eq 025**

ip-interface: Includes or excludes messages based on the ME network interface on which the SIP message was received.

Values: <match | exclude | contains> regExp

Example: **set sip-message-condition ip-interface match 192.168***

direction: Includes or excludes messages based on the direction of the packet: either TX (outbound) or RX (inbound).

Values: <match | exclude > <RX | TX>
Example: **set sip-message-condition direction exclude outbound**

transport: Includes or excludes messages based on the protocol type of the packet.

Values: <match | exclude> <any | udp | tcp | tls>
Example: **set sip-message-condition protocol exclude tls**

remote-ip: Includes or excludes messages based on the originating IP address. Enter an address and mask using CIDR notation.

Values: <match | exclude> *ipAddress*
Example: **set sip-message-condition remote-ip match 10.10.0.0/24**

local-ip: Includes or excludes messages based on the destination IP address. Enter a port number between 1 an d65535.

Values: <match | exclude> *ipAddress*
Example: **set sip-message-condition local-ip match 20.20.0.0/24**

remote-port: Includes or excludes messages based on the originating port of the packet. Enter a port number between 1 and 65535.

Values: <eq | ne | gt lt | ge | le> *port*
Example: **set sip-message-condition**

local-port: Includes or excludes messages based on the ME port over which the packet is received. Enter a port number between 1 and 65535.

Values: <eq | ne | gt lt | ge | le> *port*
Example: **set sip-message-condition local-port gt 1010**

private-remote-ip: Includes or excludes messages based on the private IP address of the remote device when NAT is being used and detected by the system.

Values: <match | exclude> *ipAddressmask*
Example: **set sip-message-condition private-remote-ip match 172.168.10.10/24**

private-remote-port: Includes or excludes messages based on the private port of the remote device when NAT is being used and detected by the system.

<eq | ne | gt lt | ge | le> *port*

Example: **set sip-message-condition private-remote-port eq 3660**

uac-public-ip: Includes or excludes messages based on the public IP address of the UAC (the user agent initiating the call leg), which is present if the UAC is doing NAT.

Values: <match | exclude> *ipAddressmask*
Example: **set sip-message-condition exclude 10.10.0.0/32**

uac-public-port: Includes or excludes messages based on the public port of the UAC (the user agent initiating the call leg), which is present if the UAC is doing NAT.

Values: <eq | ne | gt lt | ge | le> *port*
Example: **set sip-message-condition eq 10.10.0.0/32**

call-leg: Includes or excludes messages based on the direction of the packet: inbound or outbound.

Values: <match | exclude> <inbound | outbound>
Example: **set sip-message-condition call-leg exclude outbound**

message-type: Includes or excludes table rows based on the type of message: request or response.

`Values: <match | exclude> <inbound | outbound>`
Example: **set sip-message-condition message-type match response**

request-method: Includes or excludes messages based on their request method. Select a standard method.

- INVITE
- ACK
- OPTIONS
- BYE
- CANCEL
- REGISTER
- MESSAGE
- INFO
- NOTIFY
- SUBSCRIBE
- REFER
- PRACK
- PUBLISH
- UPDATE

`Values: <match | exclude> method`
Example: **set sip-message-condition request-method exclude info**

request-uri: Includes or excludes table rows based on the string found in the request URI field.

`Values: <match | exclude | contains> regExp`
Example: **set sip-message-condition match *badguy***

response-code: Includes or excludes messages based on the value in the response code field. Enter a code number or select a standard code. (Enter a ? after the mathematical operator to see the list of standard codes. For example, **set sip-message-condition response-code eq ?**) You can use the mathematical operators to specify a range of codes.

`Values: <eq | ne | gt lt | ge | le> code`
Example: **set sip-message-condition response-code match ge 600**

response-string: Includes or excludes messages based on the string found in the response field of the SIP header.

`Values: <match | exclude | contains> regExp`
Example: **set sip-message-condition response-string exclude OK**

call-id: Includes or excludes messages based on the value in the call ID field.

`Values: <match | exclude | contains> regExp`
Example: **set sip-message-condition call-id match 3ab7d43aab0d43dbbec041a***

from: Includes or excludes messages based on the value in the From: field of the SIP header.

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition from exclude mis@companyABC.com**

to: Includes or excludes messages based on the value in the To: field of the SIP header.

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition to match *.eng.*@companyABC.com**

cseq: Includes or excludes messages based on the value in the command sequence field. Use quotation marks to enclose strings that include spaces.

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition cseq match "3 invite"**

content-type: Includes or excludes messages based on the string in the content type field of the SIP header.

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition content-type match application/***

user-agent: Includes or excludes messages based on the specified string being found in the SIP header field of the packet.

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition user-agent exclude *.win.***

header: Includes or excludes messages based on the specified string being found in the SIP header field of the packet.

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition header match .*boom.***

content: Includes or excludes messages based on the specified string found in the packet payload.

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition content match "What are you doing?"**

result-code: Includes or excludes messages based on the value of the result field.

Values: <match | exclude>

- **Values: success**: the packet went through.
- **bad-header**: the system could not parse the header and so discarded the packet.
- **policy-discard**: the system immediately discarded the packet due to a session policy firing with an action set to discard.
- **policy-refuse**: the system discarded the packet, due to a session policy firing with an action set to discard, but sent a response to indicate having done so.
- **socket-timeout**: the packet was dropped due to a socket timeout.

Example: **set sip-message-condition result exclude success**

result-description: Includes or excludes messages based on any further information they may have been included in the results (such as a descriptive string for the resultCode).

Values: <match | exclude | contains> *regExp*
Example: **set sip-message-condition result-string match "from badguy.policy"**

uac-public-transport: Includes or excludes messages based on the transport protocol used by the UAC (the user agent initiating the call leg), which is present if the UAC is doing NAT.

Values: <match | exclude> <any | udp | tcp | tls>
Example: **set sip-message-condition uac-public-transport match tls**

request-uri-user-host: Includes or excludes messages based on the "user@host" portion of the Request URI.

`Values: <match | exclude | contains> regExp`
Example: **set sip-message-condition request-uri-user-host match exec@company.com**

to-user-host: Includes or excludes messages based on the "user@host" portion of the TO URI

`Values: <match | exclude | contains> regExp`
Example: **set sip-message-condition to-user-host match admin@company.com**

from-user-host: Includes or excludes messages based on the "user@host" portion of the FROM URI.

`Values: <match | exclude | contains> regExp`
Example: **set sip-message-condition from-user-host exclude joe@company.com**

cseq-message-type: Includes or excludes messages based on the SIP method type from the CSea: header line. Select one of the following method types.

- INVITE
- ACK
- BYE
- REGISTER
- REFER
- NOTIFY
- OTHER
- PRACK
- CANCEL
- SUBSCRIBE
- OPTIONS
- MESSAGE
- INFO
- PUBLISH
- UPDATE
- SERVICE
- NONE

`Default: match and INVITE`
`Values: <match | exclude> methodType`
Example: **set sip-message-condition cseq-message-type exclude INFO**

media-types: Includes or excludes messages based on the media (CODEC) type. If you set this attribute, you must enable the **prescan-media-types** property of the VSP **settings** object. Note that this attribute is case-insensitive.

`Values: <match | exclude | contains> regExp`
Example: **set sip-message-condition media-types contains g726**

From, To, and Request URI Condition Options

Includes or excludes messages based on the content of the To, From, or Request URI field strings in the SIP message.

Properties

scheme: Includes or excludes messages based on the URI type: SIP, SIPS, or Tel.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition scheme match sips**

user: Includes or excludes messages based on the user portion of the To, From, or Request URI.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition user exclude *bob.roberts***

user-param: Includes or excludes messages based on a user parameter in the SIP header of the To, From, or Request URI.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition user-param exclude *=internal**

host: Includes or excludes messages based on the SIP server host portion of the To, From, or Request URI.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition-host exclude *cov.com***

port: Includes or excludes messages based on the port number recorded in the To, From, or Request URI.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition-port match 5060**

ttl: Includes or excludes messages based on the time-to-life value associated with the UDP multicast packet for the particular URI type.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition ttl exclude 15**

method: Includes or excludes messages based on the SIP request method: INVITE, REGISTER, NOTIFY, etc.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition method match invite**

url: Includes or excludes messages based on the SIP URL.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition url match *companyABC*.com**

other: Includes or excludes messages based on any parameters in the SIP header of the To, From, or Request URI *except* the user parameter.

`Values: <match | exclude> regExp`
Example: **set from-uri-condition other exclude match *=temp**

transport: Includes or excludes messages based on the transport protocol type of the packet: UDP, TCP, TLS, or any protocol.

`Values: <match | exclude> <any | udp | tcp | tls>`
Example: **set from-uri-condition protocol exclude tls**

named-variable-condition: Specifies the named variable to match to this policy rule and how you want to compare them.

- Values: named-variable-value<*variable-name*><*match-type*><*regexp-value*>: Compare the specified variable value.

Named match values for this argument are:

- match: Allow values which match the specified expression.
- exclude: Exclude values which match the specified expression.
- contains: Allow values which contain the specified expression.
- Values: compare-named-variables<*variable-name-1*><*match-type*><*variable-name-2*>: Compare the values of two different named variables. The default **match-type** and only option is **match**, which means the ME compares if the named variable values match.

Example: **set named-variable-condition named-variable-value var1 exclude \s**

Date and Time Condition Options

Creates a match condition based on the date and time of the SIP message. You can enter as many date time conditions as you wish. Use multiple conditions to further refine the date/time.

Properties

hour: Specifies the hour in the time string of the SIP message to match on for this rule to apply. Enter a whole number between 0 (midnight) and 23. To create times that do not fall on the hour boundary, for example, 1:30, use the hour and minute conditions

`Values: <eq | ne | gt | lt | ge | le>` *value*
Example: **set date-time-condition hour gt 5**

minute: Specifies the minute in the time string of the SIP message to match on for this rule to apply. Enter a whole number between 0 and 59.

`Values: <eq | ne | gt | lt | ge | le>` *value*
Example: **set date-time-condition minute eq 30**

day: specifies the day of the week in the time string of the SIP message to match on for this rule to apply.

`Values: <match | exclude> <Sunday | Monday | Tuesday | Wednesday | Thursday | Friday | Saturday>`
Example: **set date-time-condition exclude saturday**

date: Specifies the date in the time string of the SIP message to match on for this rule to apply. Enter a whole number between 1 and 31.

`Values: <eq | ne | gt | lt | ge | le>` *value*
Example: **set date-time-condition date gt 15**

month: Specifies the month in the time string of the SIP message to match on for this rule to apply. Enter the name of a month.

`Values: <match | exclude>` *month*
Example: **set date-time-condition month exclude january**

year: Specifies the year in the time string of the SIP message to match on for this rule to apply. Enter any value.

`Values: <eq | ne | gt | lt | ge | le>` *value*
Example: **set date-time-condition year eq 2011**

Route Server Query Condition Options

Defines the conditions required for the ME to execute a route server query.

from-server-condition: Specifies the criteria against which the SIP server that sent the SIP message is compared to match this policy rule.

Default: Not configured
Values: tag | address | server-type *<match>* *<server-type>*
Example: **set from-server-condition server-type match sip-proxy**

user-group-condition: Specifies the user group names in the To: and From: fields to match in this policy rule. The match indicates that the SIP message caller (From:) and the recipient (To:) are members of the specified group.

Default: Not configured
Values: to-user-group | from-user-group | *<match>* *<user-group>*
Example: **set user-group-condition to-user-group exclude group**

named-variable-condition: Specifies the named variable to match this policy rule.

Default: Not configured
Values: named-variable-value *<variablename>* *<match>* *<value>* |
compare-named-variables *<variable-name1>* *<match>* *<variable-name2>*
Example: **set named-variable-condition compare-named-variables var1 match var2**

abort-on-failure: Specifies the ME's behavior when a query fails. When **true**, the ME stops querying the route server.

Default: false
Values: true | false
Example: **set abort-on-failure true**

stop-on-success: Specifies the ME's behavior when a query is successful. When **true**, the ME stops querying the route server.

Default: false
Values: true | false
Example: **set stop-on-success true**

13

Configuring Console Objects

The **console** object configures the serial port and remote console settings for a box.

console

Configures the serial port settings on the ME device.

Syntax

```
config box console
config cluster box number console
```

Properties

rate: Sets the bad rate in bits per second. Enter **set rate ?** at the prompt to see available rates.

Default: 115200
Example: **set rate 460800**

data-bits: Sets the number of data bits used.

Default: 8
Values: 5 | 6 | 7 | 8
Example: **set data-bits 6**

parity: Sets the parity value used for parity checking.

Default: none
Values: none | even | odd
Example: **set parity even**

stop-bits: Sets the number of stop-bits used for data sent from the box.

Default: 1
Values: 1 | 2
Example: **set stop-bits 2**

flow-control: Sets the flow control mechanism to be used on the box.

Default: none
Values: none | xon-xoff
Example: **set flow-control xon-xoff**

remote

Configures remote console access for the system via the eth0 interface. When configured, a client using the Serial Over LAN protocol can connect to the remote console, providing access similar to a direct serial port connection.

Be cautious when configuring this object. When enabled, this object makes the serial console accessible. If eth0 is configured as part of a private network and is behind a firewall, remote console access is secure. However, if the interface is public, enabling this object makes the serial console publicly available. Do not configure this object without first contacting Oracle Technical Support.

Syntax

```
config box console remote
config cluster box number console remote
```

Properties

admin: Sets the administrative state of the remote console configuration.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

ip-address: Sets the IP address on the eth0 interface, for the remote console.

Default: There is no default setting
Example: **set ip-address 172.168.10.10/32**

default-gateway: Sets the IP address for the default gateway used to reach this console address.

Default: There is no default setting
Example: **set default-gateway 172.168.10.10**

username: Configures the username required for console access. The system will prompt users for this name when they attempt to access the remote console.

Default: There is no default setting
Example: **set username admin**

password-tag: Configures the password required for remote console access. The system will prompt users for this password when they attempt to access the remote console. See Understanding Passwords and Tags for information on the two-part password mechanism.

Default: There is no default setting
Example: **set password tag root**

14

Configuring Default Session Configuration Objects

The default session configuration defines the policy settings to apply to those SIP calls for which there are no specific configured policies. When a SIP call is received, the ME registers the call and checks all policies and rules to determine how the call should be processed, including those services (such as authentication and accounting services) that should be applied to the SIP call.

If there are no policies that specifically match the SIP call registration information, the call is either forwarded to the SIP call recipient or the call is dropped based on the settings in the default session configuration.

For a description of all default session configuration subobjects, see Configuring Session Configuration Objects.

For more information on the ME policies, refer to the *Oracle Communications OS-E Session Services Configuration Guide*.

default-session-config

Opens the default session configuration object for editing, which is where you specify the default configuration used for all sessions before policy is applied. For a description of all default session configuration subobjects, see Configuring Session Configuration Objects.

Syntax

```
config vsp default-session-config
```

Properties

None

dialog-control-settings

Allows you to configure the ME to reject a message sent within a dialog that contains specified release code and text.

Syntax

```
config vsp default-session-config dialog-control-settings
```

Properties

refused-methods<source><code><text>: Select the type of message you want the WebRTC Session Controller Media Engine to reject.

Default: 405 Method Not Allowed
Values: Min: 400 / Max: 499

- Values: INVITE
- ACK
- BYE
- REGISTER
- REFER
- NOTIFY
- OTHER
- PRACK
- CANCEL
- SUBSCRIBE
- OPTIONS
- MESSAGE
- INFO
- PUBLISH
- UPDATE
- SERVICE
- PING
- NONE

Example: **set refused-methods invite 450 Method Rejected**

15

Configuring DNS Service Resolver and Server Objects

Domain Name System (DNS) servers are responsible for translating Internet host names to IP addresses. For example, DNS converts the name entered on a Web browser address bar to the IP address of the Web server that hosts that particular Web site. DNS uses a distributed database to store this name and address information for all public hosts on the Internet.

When an Internet client issues a request that involves an Internet host name, a DNS server determines the host IP address. If the DNS server cannot service the request, it sends the request to other DNS servers until the IP address is resolved, completing the Internet client request.

The ME can service both DNS requests from internal client processes (resolver function) or act as a DNS server. As a resolver residing in the management process, it accepts requests for resolutions from client processes (e.g., SIP, route-server). The resolver maintains a cache of entries as does each client process. As a server, the ME accepts both internal and external queries. It may first try to use the resolver to respond to them or it may immediately forward (proxy) them to an external server. See the **cname** object description for request processing details.

> **Note:** The DNS resolver and server objects are located in different places in the CLI hierarchy. You configure the resolver within the VSP object and the server on an IP interface.

Within the **dns** object, you can create static configurations to map a host or a domain to a SIP service. These are maintained in DNS NAPTR and SRV records. The ME also maintains a cache of query responses which it consults for information before querying an external server.

Understanding FQDN and Single-Label Queries

The ME has features in the **resolver** to process and attempt resolution of single-label queries. A single-label query is just that: a label that is not a fully qualified domain name (FQDN). Depending on the setting of the **send-single-label-query** property, The ME attempts to resolve these queries internally and/or externally. If the name is an FQDN, the system does not append a domain or search name.

The following table illustrates resolution for different query types. For this example, assume that the domain-name is set to nnos-e.com and additional search domains are set to search1.com and search2.com.

Table 15–1 Query Types and Resolutions

Query type	Example	Resolutions
Single-label	abc	abc abc.nnos-e.com abc.search1.com abc.search2.com
Two label	abc.com	abc.com abc.com.nnos-e.com abc.com.search1.com abc.com.search2.com
FQDN	abc.com.	abc.com.

For more information on DNS, refer to:

- RFC 2782, A DNS RR for specifying the location of services (DNS SRV)
- FC 2915, The Naming Authority Pointer (NAPTR) DNS Resource Record

dns

Opens the DNS configuration object for editing. From within this object you configure DNS service and resolver characteristics and identify external servers that can respond to requests not satisfied through the internal DNS cache.

Syntax

```
config vsp dns
```

Properties

None

resolver

Sets the characteristics of the ME resolver function. As a resolver, the ME obtains resource records from servers on behalf of resident or requesting applications.

Syntax

```
config vsp dns resolver
```

Properties

admin: Enables and disables the DNS configuration.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

server-scheme: Specifies how the ME selects the server to which it forwards DNS queries.

Default: preference-order

- **Values: preference-order**: The preference assigned to the server with the **preference** property.
- **load-balance**: Criteria such as pending queries and number of previous requests.
- **least-cost**: The best performance, based on, in part, response time in previous queries.

Example: **set server-scheme load-balance**

query-timeout: Specifies the time, in seconds, that a lookup can go unanswered before it times out. Once it times out, the system retries for the configured number of times.

Default: 2
Values: Min: 1 / Max: 10
Example: **set query-timeout 5**

query-retries: Specifies the number of DNS query (lookup) retries to execute if a DNS query times out. Once the configured number of retries is attempted, the system tries the next configured server.

Default: 2
Values: Min: 0 (no retries) / Max: 10
Example: **set query-retries 5**

cache-poll-interval: Specifies the number of seconds that the system waits between refreshing the DNS cache and cleaning up aged entries.

Default: 60
Values: Min: 1 / Max: 65535
Example: **set cache-poll-interval 100**

dead-threshold: Specifies the number of unanswered queries the system can tolerate before changing the DNS server state to DOWN. The server remains in that state until expiration of the time set with the **dead-interval** property.

Default: 10
Values: Min: 1 / Max: 65535
Example: **set dead-threshold 50**

not-available-ttl: Specifies the number of seconds that the system caches DNS entries that are not found on the server. When the **cache-poll-interval** time expires, the system purges the entry.

Default: 3600
Example: **set not-available-ttl 4500**

use-nnos-domain-in-search: Specifies whether the resolver should append the configured domain name to queries it receives that are not FQDNs. If **enabled**, the ME appends the name set with the domain-name property of the **static-stack-settings** object to queries.

Default: enabled
Values: enabled | disabled
Example: **set use-nnos-domain-in-search disabled**

additional-search-domains: Specifies additional domains to append to single-label queries. When the ME receives a query which is not an FQDN, it appends the configured domain (if **use-nnos-domain-in-search** is enabled) and any additional domains specified with this property.

Default: There is no default setting
Example: **set additional-search-domains companyabc.com**

enum-domain: Specifies the domain name to append to a phone number when the system performs an ENUM lookup. The ENUM lookup converts a phone number to an IP address. By default, the system uses the standard ENUM DNS domain as specified in *RFC 2916, E.164 number and DNS*.

Default: e164.arpa
Example: **set enum-domain e164.arpa**

dead-interval: *Secondary property.* Specifies the number of seconds that a server is considered DOWN by the system. When unanswered responses exceed the threshold set with the **dead-threshold** property, the ME considers the server down. (The server state and the current dead count are reported using the **show dns-resolver -v** command.) The server stays in the down state (and therefore has no queries forwarded to it) for the number of seconds set with this property. When this timer expires, the dead count is reset to zero and server use resumes until the **dead-threshold** is once again reached.

Default: 10
Example: **set dead-interval 5**

send-single-label-queries: *Secondary property.* Specifies whether the ME processes queries that have only one label to the configured server. If a query comes in with only one label, the ME appends to the label the static domain (if **use-nnos-domain-in-search** is **enabled**) and any additional domains (if **additional-search-domains** is configured) while processing the query. If this property is **enabled**, the system forward the single-label query to external servers. If it is disabled, the system tries to resolve the query internally but, failing that, does not send it out to an external server.

Default: disabled
Values: enabled | disabled
Example: **set**

server

Configures the server(s) to use for DNS and ENUM queries. You can enter any number of servers, and specify, for each, its use (DNS, ENUM, or both). The ME selects which servers will handle a query based on the setting of the **resolver** > **server-scheme** property.

Syntax

```
config vsp dns resolver server ipAddress
```

Properties

protocol: Sets the protocol the DNS resolver service uses to communicate with the identified server(s). Currently, UDP is the only supported protocol.

Default: UDP
Values: any | UDP | TCP | TLS
Example: **set protocol tls**

port: Sets the UDP port number over which the resolver service communicates with the identified server(s).

Default: 53
Example: **set port 54**

preference: Specifies the preference assigned to this server. The lower the value the higher the preference. This value is used if the **resolver > server-scheme** property is set to **preference-order**.

Default: 100
Example: **set preference 50**

type: Specifies the type of queries this server will perform. If **both** is not selected, only queries of the specified type are sent to the server.

Default: both
Values: dns-only | enum-only | both
Example: **set type dns-only**

name: Associates a name string with the server configuration. You can then reference this server from other parts of the configuration using this name. For example, the session configuration **dns-client-settings** objects uses these names to reference servers for client use.

Default: There is no default setting
Example: **set name corp-server**

host

Statically maps an IP address to a host name. Use this object to more easily manage your DNS configuration by using names instead of addresses. By creating a static configuration for a host name, you prevent a DNS lookup from going out on the wire.

The **host** object requires that you supply a *name* variable. This is the name of an Internet node, for example, a server, a router, or a PC in your network.

Syntax

```
config vsp dns host name
```

Properties

address: Sets the IP address to map to the name supplied with the **host** object.

Default: There is no default setting
Example: **set address 192.168.10.10**

service

Creates a static SIP server-to-service configuration, adding a DNS server resource (SRV) record for each SIP service. (SRV records provide contacts for the specific domain services.) Within each service, you execute this object for each SIP server to establish the order in which to contact them.

The **service** object requires that you supply a *domainName* or *hostName*, a scheme of either SIP or H323, and a protocol. The ME derives the name of the service for which you are configuring server information from these entries.

Syntax

```
config vsp dns service name {sip | h323} {any | UDP | TCP | TLS}
```

Properties

rule <sipServer>[port][priority][weight]: Sets the priority of a SIP server when there are multiple servers configured for a service. This rule sets the criteria for selecting a SIP server for the service derived from the name and protocol entered for the **service** object.

Default: There is no default setting for the server; the default port is 5060; the default priority is 1; the default weight is 0.

- **Values: SIP-server**: The name of the SIP server for the named service.
- **port**: The port on the SIP server through which this service is accessed.
- **priority**: The priority of the SIP server. The lower the number, the higher the priority. If two servers have the same priority, the system tries the server with the higher weight first. Enter a value between 1 and 65535.
- **weight**: The preference weighting for use when priority settings are equal. The higher the weight, the higher the preference. Enter a value between 0 and 65535. Use a value of 0 when there is only one SIP server configured.

Example: **set rule sipServer.companyABC.com 5001 1 10**

naptr

Creates a static mapping of service information to a specific host or domain name. The ME uses this information to do a lookup for requests in which it cannot determine either the protocol or port of the destination.

Naming-authority pointer (NAPTR) records are used to set up different services in a domain. They contain rules for converting each request to the correct configured service. Because each transport service over SIP is viewed as a different service (SIP over TCP, UDP, or TLS are each different services), they establish three different NAPTR records. This object configures the preference for use of an appropriate service for each domain. Set one rule for each protocol: UDP, TCP, and TLS. Before a request can be forwarded on, the system must know both the protocol and port for the destination.

The following table describes the decision process for different types of received requests:

Table 15–2 Received Request Process

The System Knows	The System Does Not Know	Resolution
Protocol Port	None	No lookup is necessary.
Protocol	Port	The system matches on the protocol in the NAPTR records. It then uses that record to identify which service to use. From there, the system does an SRV lookup on the service name to establish the port number.

Table 15–2 (Cont.) Received Request Process

The System Knows	The System Does Not Know	Resolution
Port	Protocol or Port Protocol	The system does a NAPTR record lookup based on the port number (if known), starting with the protocol that has the highest priority. If the system cannot find a port match, or does not know the port number, it uses the default protocol (UDP) and the port provided in the original request.

Enter the domain name or host name that you are going to map to a service.

Syntax

```
config vsp dns naptr name
```

Properties

match: Sets the match criteria for the domain name supplied when you opened the NAPTR object. If set to **exact**, the system only maps to service names that contain an exact match of the domain name you entered. If set to **wildcard**, the system maps to any service name containing the full domain name, but the service name may also contain additional characters to the left of the domain name. For example, if the NAPTR object were opened with the domain name companyABC.com, the service name could match, for example, abc.companyABC.com and xyz.companyABC.com.

Default: exact
Values: exact | wildcard
Example: **set match wildcard**

rule <protocol>[order][preference]: Sets the lookup procedure for destinations with unknown protocols or ports. You can only have one entry for each protocol within the domain name specified to open the **naptr** object. Enter a rule for each protocol (a total of three rules): UDP, TCP, and TLS.

Default: The default protocol is udp; the default order is 10; the default preference is 50

- **Values: protocol**: The protocol that the rule applies to. Enter either any, UDP, TCP, or TLS.
- **order**: The priority of the rule. Use this parameter to set the order in which the system checks The lower the number, the higher priority. If two rules have the same priority, the system uses the rule with the higher weight. Enter a value between 1 and 65535.
- **preference**: The preference weighting for use when order settings are equal. The higher the weight, the higher the preference. Enter a value between 0 and 65535.

Example: **set rule tls 5 10**

enum-mapping

Creates a static configuration mapping between an E.164 number and a host name, providing a static mapping function for unresolvable addresses. The configuration is applied when **enum-operation** property of the dial plan **normalization** object is **enabled**.

ENUMs are mappings between E.164 (the public network addressing standard) number assignments and URLs. ENUM is a protocol that makes internet resources addressable via a phone number. The protocol uses the DNS cache to identify services available to an E.164 number. By converting E.164 numbers into URLs, the ME uses Enum Server and Naming Authority Pointer (NAPTR) records to look up the services available for a specific E.164 number (via its domain name) in the DNS cache.

The **mapping** object requires that you supply a *phoneNumber* variable. Enter a number for which you want to create a permanent listing in the DNS/ENUM cache. The phone number must be at least four characters long. It is stored in the cache as type NAPTR.

For more information on using DNS to store E.164 numbers, refer to:

- RFC 3761, The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM)

Syntax

```
config vsp dns enum-mapping phoneNumber
```

Properties

domain: Specifies the domain name to append to this phone number mapping. The system will use this domain when the performing an ENUM lookup. By default, the system uses the standard ENUM DNS domain as specified in *RFC 2916, E.164 number and DNS*.

Default: e164.arpa
Example: **set domaine164.arpa**

order: Sets the priority of the mapping when there are multiple entries for a single phone number. The lower the order number, the higher priority. If two mappings have the same priority, the system uses the entry with the higher preference (see below) first.

Default: 10
Values: Min: 1 / Max: 65535
Example: **set order 30**

preference: Sets the preference weighting for use when order settings are equal. The higher the weight, the higher the preference.

Default: 50
Values: Min: 1 / Max: 65535
Example: **set preference 100**

protocol: Sets the protocol that should be associated with the telephone number

Default: UDP
Values: UDP | TCP | TLS
Example: **set protocol tcp**

replacement: Specifies the name to associate with the phone number. This field is required. You must enter a SIP URL, in the format:

SIP: *urlAddress*
The URL address that you enter can be:

- a host name only
- an IP address only
- *username@hostName* or *username@ipAddress*.

See the following SIP specifications for entry format details:

- *RFC 3761, The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM)*
- RFC 3966, The tel URI for Telephone Numbers

Default: There is no default setting
Example: **set replacement url SIP:jane@company.com**

reject

Statically enters reject entries to the resource records in the cache. These entries are not cleared when the cache otherwise clears, and continually prevent access until the entry is manually removed. Because the record indicates that the entry is a failure, the system does attempt a lookup. (Dynamic reject entries work according to the DNS specification; they are cleared every 5 minutes.)

The **reject** object requires that you supply a *name* and a **type**. Enter a host name, service name, domain name, or IP address. Any request containing that name will be rejected. (See the property descriptions for information on entering wildcards.) Also enter the record type for the entry to be rejected, either:

- **A**: Reject the supplied host name (IPv4 address).
- **PTR**: Reject the supplied IP address (Address-to name-mapping pointer records).
- **SRV**: Reject the supplied service name (Server resource rule).
- **NAPTR**: Reject the supplied domain name (Naming Authority Pointer rule).
- **CNAME**: Reject the canonical name record (makes one domain name an alias of another).
- **NS**: Reject the name server record.

Syntax

```
config vsp dns reject name type
```

Properties

match: Sets the match criteria for the name supplied when you opened the reject object. If set to **exact**, the system only maps to names that contain an exact match of the name you entered. If set to **wildcard**, the system maps to any name containing the full name, plus any addition characters to the left of the name. Below, if the reject object were opened with the name companyABC.com, the service name could match, for example, tls.companyABC.com and udp.companyABC.com.

Default: exact
Values: exact | wildcard
Example: **set match wildcard**

cname

A CNAME record maps a configured alias to a known name. This value can be exact or wildcarded.

Syntax

```
config vsp dns cname <name>
```

Properties

match: Select whether the ME matches the CNAME exactly or if it is a wildcarded match.

Default: exact

- **Values: exact**: The ME only matches if the FQDN is exact (in this example abc.com).
- **wildcard**: The ME matches if the FQDN is a wildcard (in this example www.abc.com, but not if it's exactly abc.com).

Example: **set match wildcard**

alias: Enter the alias you want associated with CNAME records.

Default: There is no default setting
Example: **set alias internal.abc.com**

dns-server

Identifies the IP interface on which the DNS server resides. The DNS server function is defined by the setting of the **mode** property. In **proxy mode**, the DNS server acts as a proxy server, in that it accepts a request, but forwards it to the configured server for fulfillment. In **cache mode**, the ME forwards the request to the **resolver**. If the resolver has a cached or static entry, it does not forward the request. Instead, it responds to the **cname** with that entry and the server responds to the requestor.

Syntax

```
config cluster box number interface ethX ip name dns-server
config cluster box number interface ethX vlan number ip name dns-server
config box interface ethX ip name dns-server
config box interface ethX vlan number ip name dns-server
```

Properties

admin: Enables or disables the DNS server on the current IP interface.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

port: Sets the protocol and port used to send and receive DNS requests with the DNS server.

Default: The default protocol is udp; the default server port is 53
Values: udp | tcp [*portNumber* Min: 1 / Max: 65535]
Example: **set port tcp 54**

mode: Specifies where the system's DNS server functionality retrieves entries from. If set to **cache**, the default, the system accepts requests, but forwards them to the resolver for fulfillment. (If the resolver has a cached or static entry, it does not forward the request, but responds with the information.) If set to **proxy**, the system forwards the request directly to the specified server. Enter the IP address of the server, as well as the protocol and port for contact.

Default: cache
Values: cache | proxy *ipAddress* [udp | tcp] [*port*]
Example: **set mode proxy 10.10.10.10 tcp 222**

16

Configuring DTMF Generation Objects

The ME, if configured to do so with the **auto-conference** property of the session config **media** object, strips conference codes (DTMF strings) from an INVITE request and injects them into the RTP stream (the established call). This allows the ME to play the DTMF tone digits to a conference server on behalf of the client calling the server. Through this object, you can control the length of play and pause time and volume for the digits that the ME plays.

dtmf-generation

Sets parameters for the conference codes derived using the **auto-conference** property of the session config **media** object. The ME applies these parameter settings to the conference codes found in INVITEs, and injects them into the RTP stream.

Syntax

```
config vsp dtmf-generation
```

Properties

digit-volume: Specifies the volume setting for the conference code tones. The digit volume is measured in decibel (dB) of the measured power referenced to one milliwatt, measured at a zero transmission level point. The smaller the dBm0, the louder the volume.

Default: 20
Values: Min: 1 / Max: 63
Example: **set digit-volume 25**

digit-duration: Specifies the length of time, in milliseconds, that the ME plays each digit of the conference code.

Default: 750
Values: Min: 100 / Max: 10000
Example: **set digit-duration 500**

inter-digit-duration: Specifies the length of time, in milliseconds, that the ME pauses between playing each digit in the conference code.

Default: 250
Values: Min: 0 / Max: 1000
Example: **set inter-digit-duration 500**

pause-duration: Specifies the number of milliseconds that the system pauses when it encounters a comma character in the conference code. The comma is a special

character, written in to the conference code, that indicates the system must wait for the specified time before playing the next tone.

Default: 3000
Values: Min: 500 / Max: 10000
Example: **set pause-duration 4500**

as-audio: Specifies whether the system sends audio or DTMF packets, when representing conference code tones, to the conference server. When true, the system encodes the sound in the current CODEC (e.g., PCMU or IBC). When false, the system attempts to send DTMF packets.

Default: true
Values: true | false
Example: **set as-audio false**

min-digit-duration: Specifies the minimum length of time, in milliseconds, that the ME plays each DTMF digit. If a DTMF event has a duration less than this value, the **digit-duration** property overrides this value and is used to play the DTMF event.

Default: 60
Values: Min: 5 / Max: 100
Example: **set min-digit-duration 75**

max-digit-duration: Specifies the maximum length of time, in milliseconds, that the ME plays each DTMF digit. If a DTMF event has a duration greater than this value, the **digit-duration** property overrides this value and is used to play the DTMF event.

Default: 2000
Values: Min: 100 / fMax: 10000
Example: **set**

digit-duration-update: When the actual duration of a DTMF event is not known, the ME sends this value in the Call-Info header when Notify-based out-of-band DTMF is supported, or as the initial duration sent in a DTMF H.245 Signal message.

Default: 2000
Values: Min: 60 / Max: 10000
Example: **set digit-duration-update 1500**

17

Configuring Enterprise Objects

Enterprise services work by using an existing directory name service in conjunction with client programs that access that service to look up user entries. By configuring the ME to recognize a particular enterprise service, you are drawing that service under the security protection of the ME, preventing application-level attacks.

Enterprise services are SIP-enabled real-time communication systems and collaboration services. You are configuring the ME so that it can access the required databases to derive the recognized SIP addresses within your enterprise. These services allow an organization to support, among others:

- IP PBX hosted VoIP services
- Enterprise instant messaging systems
- Mobile devices
- Presence-based applications

> **Note:** While you can configure enterprise services at any time, you must enable the master-services directory object for the ME to use the service. See Configuring Master Services Objects for more information.

For detailed information on ME enterprise gateways, refer to the *Oracle Communications OS-E Session Services Configuration Guide*.

enterprise

Opens the enterprise configuration group object, which is the container for the directory services and application configurations.The enterprise object contains three objects that house these additional configurations: directories, servers, and federations. In addition, it is the parent object of the unknown server policy, which sets the policies to apply to users that are not identified as belonging to a specific server.

> **Note:** You must enable the directory object in master-services before the enterprise object can become active. See Configuring Master Services Objects for more information.

Syntax

```
config vsp enterprise
```

Properties

user-group-policy <group-name><policy-reference>: Specifies the policy to apply to users of any server who are members of the specified group. The group can be either a user group from the directory service schema or a virtual group constructed in the configuration for policy application purposes.

Enter a group name, and the system applies the specified policy to any user belonging to that group, regardless of the server for which they are intended. Also enter the complete path to a previously configured policy reference.

If this value is set both here and at the server level, through the **server** object, the system applies both settings.

Default: There is no default setting
Example: **set user-group-policy lcsAdmin "vsp policies session policies policy noIM"**

directories

Opens the directories configuration object for editing. The directories container includes those objects that represent the directory services available in your enterprise. See the following chapter for a description of each type of enterprise directory and its objects and properties:

- Configuring Directory Objects

Syntax

```
config vsp enterprise directories
```

Properties

admin: Specifies whether the directory configurations are available to the system. When **enabled**, the system uses the configuration; when **disabled**, directory services are not available to the system.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

on-failure: Specifies the length of time, in milliseconds, that the ME pauses between playing each digit in the conference code.

Default: ignore

- **Values: abort**: Cancels the current attempt at establishing a session with the communications server and disregards any data generated for other enterprise services. (If one service fails, they all fail.) The system cancels all attempts to bring up any of the configured name services directories. Without a directory loaded, most likely all configured policy will fail.
- **ignore**: Ignores the attempt at establishing a session with the failed communications server, but maintains data from other services.
- **retry**: Sets the parameters for retrying the name server. Specify:
  - The number of times, between 1 and 5, to attempt to establish a session with the communications server. The default is **3** attempts.
  - The interval, in seconds, between attempts. The default is **10** seconds.

– The next desired action if all retry attempts fail (either abort or ignore). The default is **ignore**.

Example: **set on-failure retry 5 15 abort**

resolve-on-update: Specifies whether to resolve SIP addresses after a directory update. When set to **true**, the system checks its SIP address database against the updated directory, and changes the address database accordingly. When set to **false**, the system does not change the address database. (You update the directories automatically at boot or by executing the **directory-reset** action.)

Default: false
Values: true | false
Example: **set resolve-on-update true**

servers

Opens the servers configuration object for editing. The servers container includes those objects that represent the servers that provide applications in your enterprise. See the following chapter for a description of each type of enterprise server and its objects and properties

- Configuring Server Objects

Syntax

```
config vsp enterprise servers
```

Properties

default-server: Sets the server to use for all situations in which a packet arrives and does not match any criteria for server selection. Enter the server name as a reference to a previously configured server.

Default: There is no default setting
Example: **set default-server vsp enterprise servers sametime abcCo**

federations

Opens the federations configuration object for editing. The federations container includes the objects that represent junction points between the servers in your enterprise. A federation is formed by including previously configured servers into the named object. See the following chapter for information on creating a federation:

- Configuring Federation Objects

Syntax

```
config vsp enterprise federations
```

Properties

None

unknown-server-policy

Sets the policy to apply to sessions going to or coming from unknown users. You can configure the ME to apply policy when it detects a sender or receiver of a packet that is not registered in the enterprise directory service.

The unknown-server-policy object allows you to specify separate “from” and “to” policies for unregistered users. When configuring this object, you reference previously created policies. See the following for more information on policy:

- Configuring Policy Objects

Syntax

```
config vsp enterprise unknown-server-policy
```

Properties

to-policy: Sets the policy to use for all situations in which a packet is destined for a user that is not registered in any enterprise directory service. Enter the policy name as a reference to a previously configured policy.

Default: There is no default setting
Example: **set to-policy vsp policies session-policies toPolicy**

from-policy: Sets the policy to use for all situations in which a packet arrives from a user that is not registered in any enterprise directory service. Enter the policy name as a reference to a previously configured policy.

Default: There is no default setting
Example: **set from-policy vsp policies session-policies fromPolicy**

18

Configuring Eventpush Service Objects

The Eventpush service object enables you to redirect ME logged events to an external web browser, providing pushlet functionality for use in web services applications. This functionality is only useful in environments where the application uses only Javascript or only PHP. When configured, the feature overcomes a limitation in these languages that prevent them from interacting with a Java-based pushlet. (For web services applications written in Java, configuration of this object is necessary.)

The pushlet included in the ME software allows users of a web services application to have the events applicable to their call returned to their web browser. When a sample application receives an event via WSDL, it passes the event to the pushlet. Because the pushlet sorts events by ID, the web browser can request events by pushlet ID to maintain current call status.

eventpush-service

Configures and redirects the ME logged events to an external web browser over an HTTP or HTTPS web service port. It enables events to be asynchronously sent to clients. The **eventpush-service** requires the **external-services > event-group** to declare the destination service URL of the external device.

Perform the following steps to enable and configure the **eventpush-service**.

1. Configure the IP **eventpush-service**. This creates the process in the ME that responds top client event requests.

```
CXC> config box
config box> config interface eth3
config interface eth3> config ip eventpush
Creating 'ip eventpush'
config ip eventpush> config eventpush-service
config eventpush-service> set admin enabled
config eventpush-service> set protocol http 8081
config eventpush-service> set page-domain companyABC.com
```

2. Edit the target web application to include an IFrame. The IFrame is comprised of the name of the ME device running the eventpush-service application, the eventpush service port, and the string **/cometapp/covergence.html**.

   For example, if the name of the ME device running the eventpush-service is *xyz.com* with the eventpush web service running on port 8081, and if the system is running over HTTP, then the reference is **http://xyz.com:8081/cometapp/covergence.html**.

3. Configure the **external-services\event-group** > **event-service > service-url** property, so that events are passed to the destination eventpush web service. Enter

the destination domain IP, the eventpush service port, and the context string **/cometapp/callouts**. For example, **http://127.0.0.0:8081/cometapp/callouts**.

```
CXC> config external-services
config external-services> config event-group a
config event-group a> config event-service cometd
Creating 'event-service cometd'
config event-service cometd> set service-url http://127.0.0.0:8081/
cometapp/callouts
```

Syntax

```
config cluster box number interface ethX ip name eventpush-service
config cluster box number interface ethX vlan number ip name eventpush-service
config box interface ethX ip name eventpush-service
config box interface ethX vlan number ip name eventpush-service
```

Properties

admin: Enables or disables the eventpush service. Enabling the service allows you to redirect the ME logged events to an external Web browser.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

protocol: Sets the protocol to use for pushlet operations. After setting a protocol, you can select the pushlet listen port (or accept the default). This is the port over which the server listens for HTTP(S) requests.

Default: The default protocol is http with a port setting of 8081. If you set the protocol to https, the default port is 8443.

- **Values: http**: Sets an insecure (unencrypted) protocol for use in web transmission.
- **https**: Provides secure transmission of web pages by using HTTP over SSL. Optionally, you can set:
  - A reference to a previously configured certificate (configured with the certificate object).
  - An alias for the key in the key store (named in the certificate configuration).

Example: **set protocol https 8444 "vspp tls certified cxc.company.com" certKey**

max-threads: Specifies the maximum number of total worker threads, both active and spare (idle), allocated to the web server (eventpush service).

Default: 10
Values: Min: 1 / Max: 500
Example: **set max-threads 15**

min-spare-threads: Specifies the minimum number of inactive threads that the system must leave allocated to the web server. When the system removes idle threads, it must leave this number of spares available.

Default: 1
Values: Min: 0 / Max: 50
Example: **set min-spare-threads 3**

max-spare-threads: Specifies the maximum number of inactive threads the system can leave allocated to the eventpush service. When the system detects idle threads, it cannot have more than this number.

Default: 5

Values: Min: 0 / Max: 50
Example: **set max-spare-threads**

page-domain: Specifies the common domain name of the MEand the system running the web application.

Default: There is no default setting
Example: **set page-domain www.acme.com**

legacy-events: Enables or disables support for legacy events where the channel set is statically generated.

Default: enabled
Values: enabled | disabled
Example: **set legacy-events disabled**

19

Configuring External Services Objects

The external-services object defines various external web services that provide, or act as receptacle for, information to and from the Media Engine (ME). A web service provides interoperability between platforms and operating systems. Because the public interface is described in the Web Services Description Language (WSDL), an XML-based service description language, disparate platforms can use the web services to exchange data.

You can configure the ME as both a WSDL client and server. Use the **external-services** object to configure it as a client; use the **web-service** object to enable the interface, allowing the ME to function as a server.

In the ME, external web services can be used to share location databases, apply session policy, and track system notifications. See the *Oracle Communications WebRTC Session Controller Installation Guide* for more information and sample configurations.

external-services

Opens the external-services object. Within the subobjects, you configure the URLs used to access the remote web services. Configuring external services configures the ME as a client, which allows it to make calls out to other web service endpoints that implement the call out interfaces. The call out interfaces can be used to provide location information and/or policy information.

Syntax

```
config external-services
```

Properties

policy-services-type: Specifies the method by which the ME sends requests and processes responses for WSDL policy call-outs.

Default: old

- Values: **old**: Select this option to use the ME's older Java-code method. This method does not support persistent TCP connections and has to connect and disconnect for each request.
- **new**: Select this option to use the ME's newer C-code method. This method is faster and more reliable than the old method and is able to handle persistent connections, sending multiple requests on a single TCP connection.

Example: **set policy-services-type new**

trap-event: Specifies whether the ME transmits traps as events without channels.

Default: enabled
Values: enabled | disabled
Example: **set trap-event disabled**

trap-channel-events: Specifies whether the ME transmits traps as events with the trap category as a channel.

Default: enabled
Values: enabled | disabled
Example: **set trap-channel-events disabled**

policy-group

Creates a group to which the individual external policy service servers belong. The ME applies the configured failover mechanism to all server configurations within this group. When a request is made, the ME searches the list of servers in the group sequentially for one that is available. If it does not find an available server, it does not send out the request.

When the ME detects that a server is unavailable, it changes that web service server status to unavailable. (You can verify the status of a server through the **show web-services-callout-details** command with the **availability** field.) If the status is unavailable, you must set it to available (once it is) with the **web-services** action before the ME can use that server again.

Syntax

```
config external-services policy-group name
```

Properties

failover-detection: Specifies the type of failure detection the ME should perform for all external policy servers in this group.

Default: The default protocol is http with a port setting of 8081. If you set the protocol to https, the default port is 8443.

- **Values: none**: The ME does no failover detection. If a request is not serviced, the system continues to send requests until a configured timeout value is reached or the request is manually withdrawn.
- **passive**: Sets a number of failures or a timeout period before that the system can experience before it determines that the server is unavailable. The passive method only detects failures in response to a service request. Set the following:
  - *response*: The number of milliseconds that the system waits for a response from the server. When the timer expires, the system classifies the server as unavailable. Enter a value from 10 to 60,000. The default is 60,000 seconds.
  - *failures*: The number of failures that the system allows before determining that the server is unavailable. Enter a value from 1 to 100,000. The default is 3 failures.
- **active**: Uses the heartbeat URL assigned with the policy-, location- or event-service objects to constantly test server availability. Additional parameters to this property allow detection in the event that the heartbeat location is available but the web service is not. Set the following:
  - *interval*: The number of seconds between requests sent to the **heartbeat-url**. Enter a value 1 to 600. The default is 10 seconds.

- *failureInterval*: The number of seconds to wait between request attempts to the **heartbeat-url** when the previous request failed. Enter a value 1 to 10. The default is 1 seconds.
- *retries*: The number of unsuccessful requests to the **heartbeat-url** the system allows before determining that the server is unavailable. Enter a value 0 to 10. The default is 1 retry.
- *response* and *failures*: See the **passive** description.

Example: **set failover-detection active 20 2 2**

max-queue-length: Sets the maximum number of WSDL requests that can be queued for a policy group (awaiting assignment to a server). If the queue grows to this number, subsequent requests are rejected, with the result "queue-clipped," until the queue drops below this level.

Default: 64
Example: **set max-queue-length 32**

connection-mode: Specifies the manner in which connections between the ME and WSDL server are established and maintained.

Default: persistent 10 /covws/callsouts?wsdl

- **Values: persistent** [*seconds*][*page*]: Connections are initiated at boot time, and maintained using periodic keepalives. Specify an inactivity timeout, between 2 and 120 seconds, and a keepalive page.
- **lingering**: Connections are made on demand, then linger until broken by the remote server.
- **transient**: Connections are made on demand, then broken when a response is received.

Example: **set connection-mode-lingering**

overall-request-timeout: Specifies the number of seconds a request can remain in the queue for a policy server before it is timed out by the ME.

Default: 5
Values: Min: 1 / Max: 30
Example: **set overall-request-timeout 10**

connection-count: Specifies the number of seconds a request can remain in the queue for a policy server before it is timed out by the ME.

Default: 5
Example: **set connection-count 10**

request-format: Specifies the XML SOAP message format to use when making external service calls.

Default: legacy

- Values: legacy: Use the legacy deeply nested format.
- simplified: Use the simplified flat format.

Example: **set request-format simplified**

connection-mode: Specifies the manner in which connections between the ME and WSDL server are established and maintained.

Default: persistent 10/covws/callouts?wsdl
Values:

- persistent <*inactivity-time*> <*keepalive-page*>: Connections are initiated at boot time and maintained using periodic keepalives. Specify an activity timeout (between 2 and 120 seconds) and a keepalive page.
- lingering: Connections are made on demand then linger until broken by the remote server.
- transient: Connections are made on demand then broken when a response is received.

Example: **set connection-mode lingering**

connection-mode: Specifies the manner in which connections between the ME and WSDL server are established and maintained.

Default: persistent 10 /covws/callouts?wsdl
Values:

- persistent <*inactivity-time*> <*keepalive-page*>: Connections are initiated at boot time and maintained using periodic keepalives. Specify an activity timeout (between 2 and 120 seconds) and a keepalive page.
- lingering: Connections are made on demand then linger until broken by the remote server.
- transient: Connections are made on demand then broken when a response is received.

Example: **set connection-mode lingering**

policy-service

Sets the URL of the external server that maintains policy configurations to apply to a session. When configured, the ME calls out to the specified server with session data, and the server returns the appropriate policy configuration, which the ME then applies to the session. To open this object, enter a name for the policy server configuration.

Syntax

```
config external-services policy-group name policy-service name
```

Properties

admin: Specifies whether this policy service server is enabled for use.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

service-url: Specifies the URL of the remote web service that maintains the session configuration.

Default: There is no default setting
Example: **set service-url http://myserver:8080/myPolicyServer**

heartbeat-url: Specifies a location on the external server that can be used to test server connectivity. Typically, you would specify a pointer to static content on the server (e.g., a small file). The system then requests the file from the external service to determine server availability. This property is used when the **failover-detection** property is set to **active**.

Default: There is no default setting
Example: **set heartbeat-url http://myserver:8080/jeartbeatTest.html**

connection-timeout: Specifies the length of time, in milliseconds, that the system allows to complete a connection to the external policy service before cancelling the request.

Default: 500
Values: min: 100 / Max: 30000
Example: **set connection-timeout 1000**

read-timeout: Specifies the length of time, in milliseconds, that the system waits for a response from the external policy service before cancelling the request.

Default: 2000
Values: Min: 100 / Max: 30000
Example: **set read-timeout 1500**

priority: Specifies the priority of this server within the policy group. The lower the number, the higher the priority.

Default: 1
Values: Min: 1 / Max: 99
Example: **set priority 5**

connection-count: Specifies the number of simultaneous connections allowed to this server. Multiple connections can improve performance.

Default: 1
Values: Min: 1 / Max: 16
Example: **set connection-count 8**

content-type-char-set: Indicates the Content-Type value in each outgoing processEvent request and formats the body of the message appropriately.

Default: iso-8859-1
Values: UTF-8 | iso-8859-1
Example: **set content-type-char-set UTF-8**

location-group

Creates a group to which the individual external location service servers belong. The ME applies the configured failover mechanism to all server configurations within this group. When a request is made, the ME searches the list of servers in the group sequentially for one that is available. If it does not find an available server, it does not send out the request.

When the ME detects that a server is unavailable, it changes that web service server status to unavailable. (You can verify the status of a server through the **show web-services-callout-details** command with the **availability** field.) If the status is unavailable, you must set it to available (once it is) with the **web-services** action before the ME can use that server again.

Syntax

```
config external-services location-group name
```

Properties

failover-detection: Specifies the type of failure detection the ME should perform for all external policy servers in this group.

Default: The default protocol is http with a port setting of 8081. If you set the protocol to https, the default port is 8443.

- **Values: none**: The ME does no failover detection. If a request is not serviced, the system continues to send requests until a configured timeout value is reached or the request is manually withdrawn.
- **passive**: Sets a number of failures or a timeout period before that the system can experience before it determines that the server is unavailable. The passive method only detects failures in response to a service request. Set the following:
  - *response*: The number of milliseconds that the system waits for a response from the server. When the timer expires, the system classifies the server as unavailable. Enter a value from 10 to 60,000. The default is 60,000 seconds.
  - *failures*: The number of failures that the system allows before determining that the server is unavailable. Enter a value from 1 to 100,000. The default is 3 failures.
- **active**: Uses the heartbeat URL assigned with the policy-, location- or event-service objects to constantly test server availability. Additional parameters to this property allow detection in the event that the heartbeat location is available but the web service is not. Set the following:
  - *interval*: The number of seconds between requests sent to the **heartbeat-url**. Enter a value 1 to 600. The default is 10 seconds.
  - *failureInterval*: The number of seconds to wait between request attempts to the **heartbeat-url** when the previous request failed. Enter a value 1 to 10. The default is 1 seconds.
  - *retries*: The number of unsuccessful requests to the **heartbeat-url** the system allows before determining that the server is unavailable. Enter a value 0 to 10. The default is 1 retry.
  - *response* and *failures*: See the **passive** description.

Example: **set failover-detection active 20 2 2**

request-format: Specifies the XML SOAP message format to use when making external service calls.

Default: legacy

- Values: legacy: Use the legacy deeply nested format.
- simplified: Use the simplified flat format.

Example: **set request-format simplified**

red-sky-location-service

Configures the URL of the VoIP Positioning Center (VPC) providing location services (caller location) for VoIP subscribers using Location Information Services (LIS) from RedSky Technologies, Inc. If the VPC returns a location record from the WSDL query, or a message that indicates that a location record exists, the call registration completes and the ME forwards the call. Otherwise, the call is declined.

Syntax

```
config external-services location-group name red-sky-location-service name
```

Properties

admin: Specifies whether this location service server is enabled for use.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

service-url: Specifies the URL of the location web services provided by RedSky Technologies.

Default: There is no default setting
Example: **set service-url http://e911_RedSky@providerOne.com**

heartbeat-url: Specifies a location on the external server that can be used to test server connectivity. Typically, you would specify a pointer to static content on the server (e.g., a small file). The system then requests the file from the external service to determine server availability. This property is used when the **failover-detection** property is set to **active**.

Default: There is no default setting
Example: **set heartbeat-url http://e911_ RedSky@providerOne.com/heartbeatTest.html**

connect-timeout: Specifies the length of time, in milliseconds, that the system allows to complete a connection to the external location service before canceling the request. If the connection times out, the user record is still added to the location cache, but as an unregistered user.

Default: 500
Values: Min: 100 / Max: 30000
Example: **set connection-timeout 1000**

read-timeout: Specifies the length of time, in milliseconds, that the system waits for a response from the external location service before canceling the request. If the read timer expires, the user record remains unverified in the location cache.

Default: 2000
Values: Min: 100 / Max: 30000
Example: **set read-timeout 1500**

content-type-char-set: Indicates the Content-Type value in each outgoing processEvent request and formats the body of the message appropriately.

Default: iso-8859-1
Values: UTF-8 | iso-8859-1
Example: **set content-type-char-set UTF-8**

tcs-location-service

Configures the URL of the VoIP Positioning Center (VPC) providing location services (caller location) for VoIP subscribers using location-based E911 services provided by TeleCommunications Systems, Inc. If the VPC returns a location record from the WSDL query, or a message that indicates that a location record exists, the call registration completes and the ME forwards the call. Otherwise, the call is declined.

Syntax

```
config external-services location-group name tcs-location-service name
```

Properties

admin: Specifies whether this location service server is enabled for use.

Default: enabled
Values: enabled | disabled

Example: **set admin disabled**

service-url: Specifies the URL of the location web services provided by TeleCommunications Systems.

Default: There is no default setting
Example: **set service-url http://e911_TCS@locationServer.com**

heartbeat-url: Specifies a location on the external server that can be used to test server connectivity. Typically, you would specify a pointer to static content on the server (e.g., a small file). The system then requests the file from the external service to determine server availability. This property is used when the **failover-detection** property is set to **active**.

Default: There is no default setting
Example: **set heartbeat-url service-url http://e911_ TCS@locationServer.com/heartbeatTest.html**

connect-timeout: Specifies the length of time, in milliseconds, that the system allows to complete a connection to the external location service before canceling the request. If the connection times out, the user record is still added to the location cache, but as an unregistered user.

Default: 500
Values: Min: 100 / Max: 30000
Example: **set connect-timeout 1000**

read-timeout: Specifies the length of time, in milliseconds, that the system waits for a response from the external location service before canceling the request. If the read timer expires, the user record remains unverified in the location cache.

Default: 2000
Values: Min: 100 / Max: 30000
Example: **set read-timeout 1500**

content-type-char-set: Indicates the Content-Type value in each outgoing processEvent request and formats the body of the message appropriately.

Default: iso-8859-1
Values: UTF-8 | iso-8859-1
Example: **set content-type-char-set UTF-8**

generic-service

Configures the URL of the VoIP Positioning Center providing location services for VoIP subscribers using services other than RedSky Technologies or TeleCommunications Systems. If the VPC returns a location record from the WSDL query, or a message that indicates that a location record exists, the call registration completes and the ME forwards the call. Otherwise, the call is declined.

Syntax

```
config external-services location-group name generic-service name
```

Properties

admin: Specifies whether this location service server is enabled for use.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

service-url: Specifies the URL of the remote web services providing location information.

Default: There is no default setting
Example: **set service-url http://loc_rcrds@otherLocations.com**

heartbeat-url: Specifies a location on the external server that can be used to test server connectivity. Typically, you would specify a pointer to static content on the server (e.g., a small file). The system then requests the file from the external service to determine server availability. This property is used when the **failover-detection** property is set to **active**.

Default: There is no default setting
Example: **set heartbeat-url service-url http://loc_ rcrds@otherLocations.com/heartbeatTest.html**

connect-timeout: Specifies the length of time, in milliseconds, that the system allows to complete a connection to the external location service before canceling the request. If the connection times out, the user record is still added to the location cache, but as an unregistered user.

Default: 500
Values: Min: 100 / Max: 30000
Example: **set connect-timeout 1000**

read-timeout: Specifies the length of time, in milliseconds, that the system waits for a response from the external location service before canceling the request. If the read timer expires, the user record remains unverified in the location cache.

Default: 2000
Values: Min: 100 / Max: 30000
Example: **set read-timeout 1500**

content-type-char-set: Indicates the Content-Type value in each outgoing processEvent request and formats the body of the message appropriately.

Default: iso-8859-1
Values: UTF-8 | iso-8859-1
Example: **set content-type-char-set UTF-8**

event-group

Creates a group to which the individual external event service servers belong. The ME applies the configured failover mechanism to all server configurations within this group. When a request is made, the ME searches the list of servers in the group sequentially for one that is available. If it does not find an available server, it does not send out the request.

When the ME detects that a server is unavailable, it changes that web service server status to unavailable. (You can verify the status of a server through the **show web-services-callout-details** command with the **availability** field.) If the status is unavailable, you must set it to available (once it is) with the **web-services** action before the ME can use that server again. When a request is made, the ME searches the list of servers in the group sequentially for one that is available. If it does not find an available server, it does not send out the request.

Syntax

```
config external-services event-group name
```

Properties

failover-detection: Specifies the type of failure detection the ME should perform for all external policy servers in this group.

Default: none

- **Values: none**: The ME does no failover detection. If a request is not serviced, the system continues to send requests until a configured timeout value is reached or the request is manually withdrawn.
- **passive**: Sets a number of failures or a timeout period before that the system can experience before it determines that the server is unavailable. The passive method only detects failures in response to a service request. Set the following:
  - *response*: The number of milliseconds that the system waits for a response from the server. When the timer expires, the system classifies the server as unavailable. Enter a value from 10 to 60,000. The default is 60,000 seconds.
  - *failures*: The number of failures that the system allows before determining that the server is unavailable. Enter a value from 1 to 100,000. The default is 3 failures.
- **active**: Uses the heartbeat URL assigned with the policy-, location- or event-service objects to constantly test server availability. Additional parameters to this property allow detection in the event that the heartbeat location is available but the web service is not. Set the following:
  - *interval*: The number of seconds between requests sent to the **heartbeat-url**. Enter a value 1 to 600. The default is 10 seconds.
  - *failureInterval*: The number of seconds to wait between request attempts to the **heartbeat-url** when the previous request failed. Enter a value 1 to 10. The default is 1 seconds.
  - *retries*: The number of unsuccessful requests to the **heartbeat-url** the system allows before determining that the server is unavailable. Enter a value 0 to 10. The default is 1 retry.
  - *response* and *failures*: See the **passive** description.

Example: **set failover-detection active 10 1 1 45000**

trap-filter: Specifies which categories of SNMP traps the system sends out the WSDL interface to the external event service server. You can set as many of the pre-configured trap categories as necessary. If you do not set any trap filters, the system sends all traps. Use the **show trap-categories** command to list the possible trap types in each category.

Default: There is no default setting

`Values: generic | csta | dos | sip | system | tls`

Example: **set trap-filter sip**

request-format: Specifies the XML SOAP message format to use when making external service calls.

Default: legacy

- Values: legacy: Use the legacy deeply nested format.
- simplified: Use the simplified flat format.

Example: **set request-format simplified**

channels: Specifies the event channels for which this event group should be notified.

Default: There is no default setting

Example: **set channels /call**

include-channels-in-events: Indicates whether or not event notifications should include the channel.

Default: disabled
Values: enabled | disabled
Example: **set include-channels-in-events enabled**

request-style: Specifies the style to use when sending events to this listener.

Default: soap
- Values: xml: A REST-based XML event listener.
- json: A REST-based JavaScript Object Notation event listener.
- soap: A SOAP-based XML event listener.

Example: **set request-style xml**

event-service

Configures the URL of a server used for tracking ME events. (These events are similar to SNMP traps.) To open this object, enter a name for the server configuration.

Syntax

```
config external-services event-group name event-service name
```

Properties

admin: Specifies whether this event service server is enabled for use.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

service-url: Specifies the URL of the remote server tracking system notifications.

Default: There is no default setting
Example: **set service-url http://myEventServer:8080/myNotifications**

heartbeat-url: Specifies a location on the external server that can be used to test server connectivity. Typically, you would specify a pointer to static content on the server (e.g., a small file). The system then requests the file from the external service to determine server availability. This property is used when the **failover-detection** property is set to **active**.

Default: There is no default setting
Example: **set heartbeat-url service-url http://myEventServer:8080/heartbeatTest.html**

connect-timeout: Specifies the length of time, in milliseconds, that the system allows to complete a connection to the external location service before canceling the request.

Default: 500
Values: Min: 100 / Max: 30000
Example: **set connect-timeout 1000**

read-timeout: Specifies the length of time, in milliseconds, that the system waits for a response from the external location service before canceling the request.

Default: 2000
Values: Min: 100 / Max: 30000
Example: **set read-timeout 1500**

content-type-char-set: Specify the charset the ME uses on the ContentType field of outgoing Web Service requests sent to this endpoint.

Default: iso-8859-1

- Values: iso-8859-1: The ME uses the ISO-8859-1 character encoding.
- utf-8: The ME uses the UTF-8 character encoding.

Example: **set content-type-char-set utf-8**

20

Configuring Features Licensing Objects

The ME system and management software uses a license to provide you with the specific features and capacity you requested. When you purchased the ME, you selected the features or bundles of features desired. You were then provided with a license file that enabled those features and set the permissible number of sessions or endpoints.

The **features** object allows you to modify the session/endpoint values. Note that you can not exceed the value specified in the license. To obtain more capacity, contact Technical Support or your sales representative.

features

Temporarily modifies your existing ME software license to support a different capacity (either number of sessions or endpoints). You can only set the capacity for a session/endpoint to a value lower than the number allowed by your license. For example, you may have a license that allows 5000 IM sessions. You can use this configuration object to temporarily allow only 3000 sessions. You can also reset the feature to full capacity through this object.

Note that the properties you see at the CLI depend on the software features that your license supports. There are additional "non-royalty" CODECs that the system supports but that are also not displayed in the CLI. (These field values appear greyed-out when displaying the feature list from the ME Management System). You must obtain a license update to make those feature values configurable. The **Properties** table provides a complete list of licensable software features in the ME.

The default value for all properties is the maximum capacity permitted by your license.

Syntax

```
config features
```

Properties

signaling-sessions: Enables the system to perform stateful, application-level inspection, processing, and routing of SIP signaling streams. To be operational, an ME cluster requires a signaling and/or media processing software license.

media-sessions: Enables the system to anchor and route SIP-associated media streams (audio, video, file transfer, etc.). To be operational, an ME cluster requires a signaling and/or media processing software license. Also, each system media proxy must be

controlled by a system signaling proxy, which may be co-resident in the same ME chassis as the signaling proxy or in a different system chassis.

instant-message-and-presence-sessions: Enables the system to perform stateful, application-level inspection, processing, and routing of SIP/SIMPLE-based instant messaging and presence traffic. The IM and presence proxy must be co-resident with a signaling proxy on the same system chassis.

high-availability-sessions: Enables the system to participate in an active-active or active-standby high-availability cluster.

authentication-access-sessions: Validates the identities of users and/or domains cryptographically, preventing unauthorized users from gaining access to network resources. Integrates with existing authentication and credentialing systems via standard protocols (RADIUS, PKI, DIAMETER, etc.).

signaling-encryption-sessions: Encrypts and decrypts SIP signaling message headers and bodies using TLS. This ensures the authenticity, confidentiality, and integrity of SIP signaling streams.

media-encryption-sessions: Encrypts and decrypts SIP-associated media sessions (audio, video, file transfer, etc.) using the Secure Real-time Transport Protocol (SRTP). This ensures the authenticity, confidentiality, and integrity of real-time media information.

media-validation-sessions: Ensures that the media sessions set up by a SIP user agent is the same as the session that was negotiated during the associated signaling dialogs and permitted by media control policies. This prevents both attacks that exploit the independence of SIP signaling and media channels and unauthorized consumption of bandwidth.

dos-protection-sessions: Detects and mitigates brute force (resource exhaustion) DOS attacks. Monitors short-term network, transport, and application-level connection activity, detects abnormal aggregate signaling patterns, and denies resources to sessions that match the attack profile.

session-admission-control-sessions: Limits calling activity based on administratively defined thresholds for session count, total bandwidth, and/or observed quality of service (QOS) metrics. Session admission control policies can be defined on logical and/or physical network interfaces.

media-control-sessions: Enables fine-grained control over SIP-associated media sessions. For example, an administrator could define a policy saying that only people in a particular group or department (as defined in the directory) can do video sessions.

qos-control-sessions: Enables control of the QoS of SIP-based applications by performing policy based L2/L3 QoS marking.

session-routing-control-sessions: Implements intelligent session routing policies such as application-aware load balancing and inbound call routing (e.g., parallel fork, sequential search, presence based routing, and others).

file-transfer-control-sessions: Enables policy-based control over SIP file transfers.

instant-message-content-control-sessions: Enforces acceptable use policies on the content of instant messages. Scans IM content for string patterns matching regular expressions and takes a policy-defined action when it finds a match.

url-control-sessions: Controls propagation of URLs embedded in SIP-based instant messages. This enables enforcement of acceptable use policies and prevents the propagation of blended threats via SIP applications.

session-detail-recording-sessions: Creates detailed records for signaling sessions that traverse the ME. From this you can track the usage of SIP-based services and applications.

qos-detail-recording-sessions: Creates QoS records for media sessions that traverse the system, providing data for network engineering, capacity planning, and troubleshooting.

audio-recording-sessions: Enables policy-based recording of audio session content, demonstrating compliance with electronic communications monitoring policy and regulation.

video-recording-sessions: Enables policy-based recording of video session content, demonstrating compliance with electronic communications monitoring policy and regulation.

file-recording-sessions: Enables policy-based recording of file transfer session content, demonstrating compliance with electronic communications monitoring policy and regulation.

file-mirror-db-size: Enables file mirroring and sets the number of files the system can concurrently mirror.

instant-message-recording-sessions: Enables policy-based recording of IM session content, demonstrating compliance with electronic communications monitoring policy and regulation.

nat-traversal-sessions: Enables SIP-based applications to traverse remote NAT/firewall functions that may not be under the organization's authority or control. This extends SIP-based applications and services to remote endpoints.

directory-integration-sessions: Enables the system to import information from directories with Lightweight Directory Access Protocol (LDAP) interfaces so that administrators can define and enforce directory-based policies.

lcs-sametime-gateway-sessions: Enables presence visibility and messaging connectivity between Microsoft Live Communication Server (LCS) and IBM Lotus Sametime communities.

transcode-sessions: Enables transcoding media types, which is the process of converting media from one CODEC into a different CODEC on output.

provisional-transcode-sessions: Enables provisional transcoding sessions. These sessions are temporary sessions used until the system has established whether a call needs transcoding. If it does require transcoding, the session then uses the **transcode-sessions** license.

g723 | g728 | g729 | g726-16 | g726-24 | g726-32 g726-40 | gms | gsm-amr | ilbc | g722-1: Sets the maximum number of concurrent CODEC sessions available for encoding/decoding. The limit does not apply to forwarding or recording RTP packets, only to:

- Making a transcoded call (see Transcoding Media Types for more information)
- Playing recorded announcements
- Mixing recorded calls for playback
- Playing back of recorded calls
- Archiving calls.

Note that if you have the **media** object **music-on-hold** property enabled, the ME holds open one license seat for playing the music until the rtp-cache is released (e.g., system

reboot or **rtp-header** action). This applies to each CODEC type used for playing music on hold. For other announcements and DTMF generation (as audio), the system requires one license until the entire announcement or tone is cached, and the seat is released.

The default setting for these properties is the maximum number of allowable licenses. For g723, g729, gsm-amr, and g722-1, the number of available license seats is more restrictive. Contact Technical Support for additional capacity.

monitoring-calls: Sets the maximum number of monitored calls allowed concurrently.

csta-sessions: Sets the maximum number of concurrent CSTA sessions allowed.

h264: Specifies the maximum concurrent number of H264 encoders and decoders for playout, announcements, mixing, or transcoding.

Default: 200000
Values: Min: 0 / Max: 200000
Example: **set h264 150000**

speex: Specifies the maximum concurrent number of Speex encoders and decoders for playout, announcements, mixing, or transcoding.

Default: 200000
Values: Min: 0 / Max: 200000
Example: **set speex 150000**

silk: Sets the maximum number of concurrent SILK sessions available for encoding and decoding. The limit does not apply to forwarding or recording RTP packets, only making transcoded calls, playing recorded announcements, mixing recorded calls for playback, playing back recorded calls, and archiving calls.

Default: 200000
Values: Min: 0 / Max: 200000
Example: **set silk 150000**

amr-wb: Sets the maximum number of concurrent AMR-WB sessions available for encoding and decoding. The limit does not apply to forwarding or recording RTP packets, only to making transcoded calls, playing recorded announcements, mixing recorded calls for playback, playing back recorded calls, and archiving calls.

Default: 200000
Values: Min: 0 / Max: 200000
Example: **set amr-wb 100000**

control-sessions: Sets the maximum number of control sessions for signaling or web-services.

file-services: Sets the maximum number of sessions for file playback or recording.

Default: 100
Values: Min: 0 / Max: 100
Example: **set file-services 75**

web-audio: Sets the maximum number of sessions for Flash audio.

web-video: Sets the maximum number of sessions for Flash video.

conferencing: Sets the maximum number of sessions for third-party conferencing.

21

Configuring ICMP Objects

The *Internet Control Message Protocol (ICMP)*, defined in RFC 792, is a protocol used to determine whether a destination is unreachable. A TCP/IP-based protocol, ICMP verifies, through error and control messages between a host and an Internet gateway, the validity of an IP address. For example, ICMP functions are used by ping utilities to verify network connectivity. You configure ICMP for each IP interface that requires the functionality.

h323

Configures the ICMP protocol on the IP interface that hosts it.

Syntax

On a public IP interface:

```
config cluster box number interface ethX ip name icmp
config cluster box number interface ethX vlan number ip name icmp
config box interface ethX ip name icmp
config box interface ethX vlan number ip name icmp
```

Properties

admin: Sets the administrative state of the ICMP protocol, either **enabled** (running) or **disabled**. When disabled, you can still configure the ICMP parameters, but the parameters do not become active until the **admin** property is set to **enabled**.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

limit: Limits the number of ICMP packets that can be received per second on this IP interface.

Default: 10
Values: Min: 1 / Max: 1000
Example: **set port 10**

22

Configuring IP Objects

This chapter describes the Internet Protocol (Version 4) configuration objects in the ME. IP objects are identified by a unique string name. By using names, you can change the underlying IP address and mask without disrupting the interface. This means that you do not need to first delete an interface if you need to edit the address/mask. Interfaces that boot using the Dynamic Host Configuration Protocol (DHCP) also use the string name, since there is no IP address or mask for the interface.

ip

Opens the named IP configuration object for editing. Specify the name of the IP interface using up to 16 alphanumeric characters with no blank spaces. If you intend for the interface to be a headend interface to support load-balancing of SIP processing, see Configuring Head-End and Backing Interfaces for more information.

Tag-Based Route Selection

The ME uses classification tags to classify incoming traffic and routing tags to control the egress route for a specific service type. This may be useful, for example, in E911 applications. With inbound traffic to the ME on an interface, you may want to ensure that it always goes out on a specific interface. To do this, you would configure a classification-tag on the incoming interface that matches the routing-tag on the egress interface you desire.

When you configure an IP interface, the ME installs both a network route and a host route into the generic routing table. For example, suppose you create an IP interface named ABC with static IP address 1.1.1.1/32:

```
Generic route table for ABC
-------------------
1.1.1.1/32
1.1.1.0/24
```

If there are services configured under the interface (i.e., media, SIP, or STUN), the route is also installed in the specific service routing table. (See Services Routing Description for a general description of service route tables.) For example, if you configured SIP on interface ABC:

```
Generic route table for ABC
-------------------
1.1.1.1/32
1.1.1.0/24
SIP service route table for ABC
-------------------
1.1.1.1/32
1.1.1.0/24
```

However, if you create a routing-tag for an interface, the ME creates a separate service route table for that tag, populated with any static routes configured on that interface. When the first routing-tag is configured, the ME removes the routes associated with that interface from the default service route table. They are only available in the service route tables associated with the routing-tag(s).The ME does not install (or removes) interfaces that have a routing-tag applied from the default service routing table. For example, if you created routing-tag E911 on interface ABC:

```
Generic route table for ABC
-------------------
1.1.1.1/32
1.1.1.0/24
SIP service route table for ABC
-------------------
SIP service route table for ABC.E911
-------------------
1.1.1.1/32
1.1.1.0/24
```

To retain the route in both the default service route table and the tag-specific service route table, add a **routing-tag** named "null." This reserved routing-tag name indicates that the service routes should be installed in the default service route table as well.

Note that tag-based service route tables inherit the metrics assigned to that service type with the services-routing metrics. In addition, if a matching session configuration includes a **routing-settings > ingress-classification-tag** for incoming traffic, the session config setting takes precedence. See the **routing-tag** and **classification-tag** descriptions in this object for specific configuration requirements.

> **Note:** The preferred method for creating virtual firewalls is by using routing tags and VLANs. For sample configurations that illustrate VLANs, overlapping IP addresses, and virtual firewalls, see the *Oracle Communications WebRTC Session Controller Administration Guide*.

Syntax

```
config cluster box number interface ethX ip name
config cluster box number interface ethX vlan integer ip name
config cluster vrrp vinterface vxID ip name
config cluster vrrp vinterface vxID vlan integer ip name
config box interface ethX ip name
config box interface ethX vlan number ip name
```

Properties

admin: Enables or disables IP services on this interface.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

ip-address: Sets Dynamic Host Configuration Protocol (DHCP) IP address assignment on this interface from a DHCP server, or sets a static IP address and network mask.

For static IP addresses, specify the IP address and network mask for this Ethernet interface.

Default: dhcp
Values: DHCP | static *ipaddress/mask*
Example: **set ip-address static 192.100.10.10/32**

geolocation: Assigns a numeric to the IP interface that you can later use, for example, within a policy to identify traffic to or from that interface. To use the policy match feature, set the session configuration **routing-settings** object. You can also use this value as a filtering mechanism with the **service-route-lookup** action to return the best route to a destination.

Default: 0
Example: **set geolocation 10**

metric: Associates a cost with the interface routes (both host and network routes) that the system adds to its services route and route DB tables. The system chooses the more preferred route when there are multiple interfaces available on the same network. The lower the metric the more preferred the route. This value is carried over to the VSP **services-routing** metrics as the **user-metric** value.

Default: 1
Values: Min: 0 / Max: 4294967295
Example: **set metric 10**

classification-tag: Associates the classification-tag with the incoming service on this interface. The classification-tag applies to the ingress interface over which the system initially receives service traffic. Each IP interface can have at most one classification tag. This tag must match a configured **routing-tag** for tag-based route selection to be in effect.

You can also configure ingress or egress classification tags through the session-config **routing-settings** object. If this property is configured in both places, the **routing-settings** configuration takes precedence.

Note that this tag is case-sensitive.

Default: There is no default setting
Example: **set classification-tag E911**

routing-tag: Associates all the routes configured on an interface with this routing-tag and creates a service route table based on the routing-tag for each service enabled on this interface. The routing-tag applies to the egress interface over which the system forwards service traffic. In order to perform tag-based routing, a classification-tag must be configured on the ingress interface over which the system initially receives service traffic, and that classification tag must match the routing-tag. Each IP interface can have multiple routing tags. (Classification tags in the session-config **routing-settings** object also must match this routing tag set in the ip object.

Once a routing-tag is configured for an interface, the service routes associated with that interface are installed in the service route table associated with the routing-tag(s). In other words, the service routes are no longer installed in the default service route tables: they are only in the service route tables specified by the routing-tag(s). However, in order for tag routing to be in effect for media, the **tag-routing** property of the matching session config **media** object must be enabled (it is disabled by default).

If you create an additional routing-tag for the interface with the name “null,” the system installs the route in both the default service route table and the tag-specific service route table. Note that this tag is case-sensitive.

Default: There is no default setting
Example: **set routing-tag E911**

security-domain: Sets the informational text string that indicates the trust level of this IP interface. For example, interfaces that point to the internal network are **trusted**; interfaces that point to the public DMZ-side of the network are **untrusted**. You can then use this setting to identify an interface within your policy configuration.

Default: There is no default setting
Values: trusted | untrusted
Example: **set security-domain untrusted**

trusted-peer: Configures one or more trusted peers for this IP interface. The system accepts and processes all TCP traffic received from a trusted peer. Use this property to designate servers as trusted peers in a VRRP topology that uses TCP as the transport between the system and the server. If a failover should occur, the backup system will accept server traffic and send a TCP reset to close the connection to the failed system and establish a new one for itself.

Default: There is no default setting
Example: **set trusted-peer 10.10.10.1**

address-scope: Sets the informational text string that indicates the private or public scope of this IP interface. For example, interfaces with private IP addresses on the internal network can be configured as **private**; interfaces with public IP addresses to the external network can be tagged as **public**. You can then use this setting to identify an interface within your policy configuration.

Default: There is no default setting
Values: public | private
Example: **set address-scope private**

filter-intf: Enables or disables secure traffic filtering on this IP interface. When enabled, inbound packets that match one of the configured IP addresses on this interface are allowed to pass. All other IP packets are blocked. This enforces the concept that packets destined for an interface must actually come in on that interface.

For example, consider a box with two Ethernet interfaces: 1.1.1.1 on interface A and 2.2.2.2 on interface B. When disabled, pinging either address from the B side of the network will succeed, even though 1.1.1.1 is an A-side IP address. However, when **filter-intf** is set to **enable**, pinging 1.1.1.1 from the B side fails.

Set this to enabled to add another level of security to the ME, however, make certain that you fully understand your network structure before setting up this traffic filtering.

Default: disabled
Values: enabled | disabled
Example: **set filter-intf enabled**

netbios

Configures the NetBIOS name service. This is a service, used by Windows systems, to locate each other when configured on the same network subnet.

Syntax

```
config cluster box name interface eth ip address netbios
```

Properties

admin: Enables or disables this NetBIOS server.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

port: Specifies the port on which the NetBIOS service listens.

Default: 137

Values: Min: 1 / Max: 65535
Example: **set port 250**

netbios-name: Specifies the name to which the NetBIOS server response.

Default: There is no default setting
Example: **set netbios-name server1**

media-server

Configures an internal media server's listener port when you are configuring the multimedia streaming server (MSS) process on the ME. For more information on MSS, see the *Oracle Communications OS-E Session Services Guide*.

Syntax

```
cluster box name interface eth ip address media-server
```

Properties

rtmp: Configures a Real Time Protocol (RTMP) listener port. Enter a TCP port number to use for receiving requests.

Default: 1935
Values: Min: 0 / Max: 65535
Example: **set rtmp 1940**

rtmpt: Configures a Real Time Protocol Tunneled (RTMPT) listener port. Enter a TCP port number for receiving requests.

Default: 1935
Values: Min: 65535
Example: **set rtmpt 1945**

rtmps: Configures a Real Time Protocol Secure (RTMPS) listener port. Enter a TCP port number to use for receiving requests.

Default: 1935
Values: Min: 0 / Max: 65535
Example: **set rtmps 1930**

rtmp

Configures a Real Time Protocol (RTMP) listener port.

Syntax

```
cluster box name interface eth ip address media-server rtmp
```

Properties

app-name: Specify the server application name for this RTMP port.

Default: live
Example: **set app-name live**

rtmpt

Configures a Real Time Protocol Tunneled (RTMPT) listener port.

Syntax

```
cluster box name interface eth ip address media-server rtmp
```

Properties

app-name: Specify the server application name for this RTMPT port.

Default: live
Example: **set app-name live**

rtmps

Configures a Real Time Protocol Secure (RTMPS) listener port.

Syntax

```
cluster box name interface eth ip address media-server rtmp
```

Properties

app-name: Specify the server application name for this RTMPS port.

Default: live
Example: **set app-name live**

certificate: Specify the certificate to use for this connection. This references the **vsp > tls > certificate** object.

Default: There is no default setting
Example: **set certificate vsp\tls\certificate cert1**

23

Configuring Kernel Filter Rule Objects

Kernel filter rules provide a security mechanism that allows or denies inbound traffic on ME IP interfaces. The filter controls access to resources on the enterprise servers based on source IP address and/or subnet, source port, and protocol. When the ME processes kernel rules, it first interprets deny rules, then allow rules. Therefore, you can deny a subnet access, and then allow specific endpoints.

The ME acts on kernel rules before the other, higher level rules such as DOS policy rules. This stops traffic from known problems early, tying up fewer processing resources.

kernel-filter

Creates or edits kernel filters. Kernel filter rules allow you to deny traffic on an IP interface based on source IP address, source port number, and packet type.

> **Note:** Kernel filters are not allowed for media interfaces. If kernel filters are needed for SIP interfaces, then you must configure a separate interface to use for media.

Syntax

```
config cluster box integer interface ethX ip name kernel-filter
config cluster box integer interface ethX vlan integer ip name kernel-filter
config box interface ethX ip name kernel-filter
config box interface ethX vlan integer ip name kernel-filter
```

Properties

None

deny-rule

Creates or edits the named kernel filter deny-rule configuration. A deny rule specifies the source IP address or subnet, source port number, and protocol associated with traffic to be blocked on the current IP interface.

Specify the rule name using up to 16 alphanumeric characters, enclosing blank spaces in quotation marks.

Syntax

```
config cluster box integer interface ethX ip name kernel-filter deny-rule name
```

```
config cluster box integer interface ethX vlan integer ip name kernel-filter
deny-rule name
config box interface ethX ip name kernel-filter deny-rule name
config box interface ethX vlan integer ip name kernel-filter deny-rule name
```

Properties

admin: Sets the administrative state of this kernel filter deny rule. When enabled, network traffic is blocked using the configured IP address or subnet, port number, and packet type. When disabled, the deny rule is not in effect.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

source-address</mask ipAddress/mask>: Specifies the source IP address or subnet associated to filter (deny) on this IP interface. Specify the IP address and mask in CIDR format.

Default: 0.0.0.0/0
Example: **set source-address /mask 215.200.0.0/16**

source-port: Specifies the source port number associated with received packets to filter (deny) on this system interface.

Default: 0 (deny all ports)
Example: **set source-port 56**

protocol: Specifies the source protocol associated with received packets to filter (deny) on this system interface..

Default: all
Values: all | icmp | tcp | udp | vrrp
Example: **set protocol tcp**

allow-rule

Creates or edits the named kernel filter allow-rule configuration. An allow rule specifies the source IP address or subnet, source port number, and protocol associated with traffic to be specifically allowed on the current IP interface. Typically the allow rule is used to override the denial of an subnet by allowing specific endpoints.

Specify the rule name using up to 16 alphanumeric characters, enclosing blank spaces in quotation marks.

Syntax

```
config cluster box integer interface ethX ip name kernel-filter allow-rule name
config cluster box integer interface ethX vlan integer ip name kernel-filter
allow-rule name
config box interface ethX ip name kernel-filter allow-rule name
config box interface ethX vlan integer ip name kernel-filter allow-rule name
```

Properties

admin: Sets the administrative state of this kernel fitter allow rule. When enabled, network traffic is allowed using the configured IP address or subnet, port number, and packet type. When disabled, the allow rule is not in effect.

Default: enabled
Values: enabled | disabled

Example: **set admin disabled**

source-address</mask ipAddress/mask>: Specifies the source subnet, but more typically IP address, to allow on this IP interface. Specify the IP address and mask in CIDR format.

Default: 0.0.0.0/0
Example: **set source-address /mask 215.200.40.8/32**

source-port: Specifies the source port number associated with received packets to allow on this system interface.

Default: 0 (allow all ports)
Example: **set source-port 56**

protocol: Specifies the source protocol associated with received packets to allow on this system interface.

Default: all
Values: all | icmp | tcp | udp | vrrp
Example: **set protocol tcp**

24

Configuring Master Services Objects

The master services objects allow you to enable services for directory, accounting, authentication, database, registration, media failover, and cluster services. Each master service can run on one box in the cluster. The **host-box** property within each master service object defines the primary box for that service. You can also configure backup boxes in the event of primary failure by re-executing the **host-box** configuration.

In the example below, the first host box listed in the configuration serves as the primary host for the directory service. Subsequent host boxes (2 and 3) serve as backup.

```
NNOS-E> config master-services
config master-services> config directory
config directory> set host-box cluster box 1
config directory> set host-box cluster box 2
config directory> set host-box cluster box 3

config directory> show
 master-services
  directory
   admin enabled
   host-box[1] cluster\box 1
   host-box[2] cluster\box 2
   host-box[3] cluster\box 3
```

The first **host-box** property identifies which box runs the service. If that box does not perform, the other configured host boxes will perform in succession and attempt to boot the service.

Master Services in VRRP Configurations

Master services that are running in **vrrp** configurations can use the **group** property as an additional backup mechanism. The group property is an option to link the VRRP configuration with other services on the box, in this case the master services. If one interface of a VRRP pair is down, the group with which they are associated is considered down. If a service is associated with that group, the box hosting the downed VRRP pair stops the service, and the backup box then restarts it. (A vinterface can have more than one Ethernet interface for a given box. The ME does not bring the service down until all configured Ethernet interfaces have been established as "unavailable.")

This feature is illustrated in the following sample output. (Note that display of properties unrelated to this feature have been removed for clarity.) If either eth1 *and* eth4 or eth2 on box 1 lose link, the ME considers VRRP group 1 down. This causes both vinterface vx1 and vx2 on box 1 to go to their configured backups, resulting in box 2 becoming master for both of these VRRP interfaces.

In the master services configuration, the directory service is in group 1 but accounting is not associated with a VRRP group. This configuration results in the ME causing box 1 to stop the directory service and box 2, as backup, to restart it. The accounting service remains unchanged as it was not associated with the VRRP configuration.

```
config vrrp> show -v
cluster
  vrrp
    admin enabled
    vinterface vx1
      admin enabled
      group 1
      host-interface[1] cluster\box 1\interface eth1
      host-interface[2] cluster\box 1\interface eth4
      host-interface[3] cluster\box 2\interface eth1
    vinterface vx2
      admin enabled
      group 1
      host-interface[1] cluster\box 1\interface eth2
      host-interface[2] cluster\box 2\interface eth2

config master-services> show -v
directory
  admin enabled
  host-box[1] cluster\box 1
  host-box[2] cluster\box 2
  group 1
accounting
  admin enabled
  host-box[1] cluster\box 2
  host-box[2] cluster\box 1
  group 0
```

master-services

Opens the **master-services** configuration object.

Syntax

```
config master-services
```

Properties

advertisement-interval: *Secondary property. S*ets the interval at which the master services broadcast their locations to boxes in the cluster. This setting applies to all master services.

Default: 60
Example: **set advertisement-interval 90**

boot-interval: *Secondary property.* For Oracle debugging only. Sets the wait time for restart of master services at system boot in a non-clustered network. In a clustered network, master-services start once the clustering is established (usually less than 30 seconds).

Default: 30
Example: **set boot-interval 60**

cluster-master

Configures the box that maintains the master configuration for the cluster. It pushes out configuration changes to other boxes in the cluster. If a different box becomes cluster-master, it would then start sending out its configuration to the other boxes.

Syntax

```
config master-services cluster-master
```

Properties

admin: Enables and disables the configuration that selects a master directory service. Enabling the directory service provides a link to the system gateways.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the box that acts as cluster master. You must select a box to serve as the cluster master using this property. See Master Services Description for more information.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the cluster master with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to true, the master resumes its position. If set to false, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

directory

Opens the ME directory services object on the master. Directory services is the function that allows communication between the ME and gateway services such as Active Directory, LCS, Sametime, etc. If this object is disabled, you can still configure the enterprise gateway services (through the **directory** object), but they do not become active until you enable this master service.

Note that if you have enabled the **local-directory-based-user-services** property for VSP **settings**, you must configure directory services on at least one box in the cluster.

See Configuring Enterprise Objects for information on enabling the directory service.

Syntax

```
config master-services directory
```

Properties

admin: Enables and disables the system directory service. Enabling the directory service provides a link to the system gateways.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which directory services run and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, no directory services can run.

Default: There is no default setting
Example: **set host-box cluster box 2**

group: Associates the directory service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to true, the master resumes its position. If set to false, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

settings

Secondary object. Configures total and minimum memory allowance for the directory, accounting, and/or database processes. Use these properties to fine-tine the default settings for applications that require additional memory.

Syntax

```
config master-services directory settings
config master-services accounting settings
config master-services database settings
config master-services jtapi settings
```

Properties

heap-max: Specifies the total amount of memory that the system can allocate to the directory, accounting, or database processes.

Default: 128

Values: Min: 16 / Max: 2048
Example: **set heap-max 256**

heap-min: Specifies the minimum amount of memory that the system can allocate to the directory, accounting, or database processes.

Default: 32
Values: Min: 16 / Max: 2048
Example: **set heap-min 96**

argument: For Oracle Technical Support only.

thread-checker: Configures the system to check for blocked threads in current processes. If **true**, the system monitors for blocked threads at a regular interval. If a process is repeatedly blocked, the ME forces a restart of the process. It then writes a message to the event log, indicating the process name and ID, and recording a stack trace. This property is for Technical Support use only.

Default: false
Values: true | false
Example: **set thread-checker true**

association-min-lifetime: This is a secondary property. Controls the amount of time in seconds that "association" information (data for a given to-from URI pair) is preserved.

Default: 300
Values: Min: 0 / Max: 360000
Example: **set association-min-lifetime 25000**

track-sip-messages: Specifies whether the ME tracks the response to SIP REGISTER and INVITE messages. When enabled, the **show sip-register-responses** and **show sip-invite-responses** status providers include data indicating the type and number of responses sent and received (e.g., the number of 200 OKs, 503s, etc.)

Default: disabled
Values: enabled | disabled
Example: **set track-sip-messages enabled**

database-connection-memory-limit: Specifies the maximum memory, in kilobytes, allowed per database connection. The connection ends if this memory limit is exceeded.

Default: automatic
Values: automatic | *integer* (1000000-3000000)
Example: **set database-connection-memory-limit 2000000**

unclean-shutdown-recover: Specifies how the local database is handled during startup after an unclean shutdown. An unclean shutdown may cause corruption in the database and is usually caused by a crash.

Default: always-archive

- Values: always-archive: Always archive data if unclean shutdown is detected. You must also enter the number of times for the ME to archive (from 1 to 20).
- attempt-repair: Attempt corruption detection and repair and archive only if repair fails.

Example: **set unclean-shutdown-recover attempt-repair**

allow-route-set-duplicates: *Secondary property*. By default, the ME allows duplicate destination routes. When this property is disabled, duplicate entries are removed and the ME has only one entry in the route-set.

Default: enabled

Values: enabled | disabled
Example: **set allow-route-set-duplicates disabled**

max-proxy-transactions-per-second: *Secondary property*.

Sets the maximum number of concurrent proxy transactions that a session can have.

Default: 20
Values: Min: 1 / Max: 65535
Example: **set max-proxy-transactions-per-session 30**

unescape-header-params: *Secondary property*. When set to the default value **true**, the ME changes the To-tag parameter and messages can be ignored. To ensure the ME does not alter any of the tag parameters, set this parameter to **false**.

Default: true
Values: true | false
Example: **set unescape-header-params false**

accounting

Opens the ME accounting services object running on the master. Accounting services is the object that enables or disables all accounting functions on the ME, such as RADIUS and Diameter accounting services, system logging (syslog), the accounting database, and the accounting file system. If this object is disabled, you can still configure the accounting services, but they do not become active until you enable this master service. This setting overrides the setting of each individual accounting function.

See Configuring Accounting Objects for information on enabling and configuring accounting services. See **settings** for information on memory allowance settings.

Syntax

```
config master-services accounting
```

Properties

admin: Enables or disables accounting services on the system.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which accounting services run and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, no accounting services can run.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the accounting service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to true, the master resumes its position. If set to false, the backup service retains master control.

Default: false
Values: true | false
Example: **set do-not-disturb enabled**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

authentication

Opens the ME authentication services object running on the master. This object enables or disables all authentication functions on the ME, such as RADIUS and Diameter authentication services, and local user profiles. If this object is disabled, you can still configure the authentication services, but they do not become active until you enable this master service. This setting overrides the setting of each individual authentication function.

See Configuring RADIUS-Group Objects and Configuring Diameter Client and Server Objects for information on enabling and configuring authentication services.

Syntax

```
config master-services authentication
```

Properties

admin: Enables or disables authentication services on the system.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which authentication services run and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, no authentication services can run.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the authentication service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to true, the master resumes its position. If set to false, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each

hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

database

Opens the ME database object running on the master. This is the system database that contains the traffic data that resulted from tracing all packets in and out of the ME. The system writes the header information from all TCP, UDP, and IP packets to the database, as well as all fields in the SIP header. This database is used by the system for DOS analysis, and in the ME Management System to display call details and call sequence.

> **Note:** The database-write property must be enabled in the vsp object for the ME to write data to the database.

From the database object you can also set up operations for cleaning the database. The maintenance operations are based on the SQL VACUUM command, which reclaims storage occupied by deleted entries. Maintenance purges the database of old entries at regularly scheduled intervals. An entry is considered "old," and is therefore purged, if it is older than the day limit for the history table. Use the database object **database** subobject to set the number of days worth of entries to keep, based on table type.

If you set the maintenance period to zero, you disable the function. You can execute an immediate database purge using the top-level **database** action. Note that you may still see indication of entries in the session table if the corresponding history table entries have not yet aged out.

See **settings** for information on memory allowance settings.

Syntax

```
config master-services database
```

Properties

admin: Enables or disables the system database. If the database is disabled, the system cannot perform DOS analysis or record call details

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which database services run and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, no databases services can run.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the database service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0
Example: **set group 1**

maintenance: Sets the time for or frequency of database purging. If a daily purge is not appropriate, use the **period** property to set the number of hours between executions. However, it is advisable to run maintenance at least every 24 hours.

Default: time-of-day 03:00

- **Values: period <***hours***>**: The regularity with which maintenance should occur. Enter a number of hours. Entering 0 disables the maintenance function; this value should only be used for troubleshooting purposes.
- **time-of-day <***hours:minutes***>**: The time at which the maintenance should occur. Enter a time in 24-hour format (for example, enter 17:00 for 5:00 p.m.). The system uses local time.
- **disabled**: Turns off the database purging feature. Be aware that if you disable maintenance, the database files can get quite large. Use this option as a debugging tool and then re-enable maintenance.

Example: **set maintenance time-of-day 0:00**

media: Specifies whether the system writes media information to the database (to the media message table). The system does not create entries in the media message table if this property is not enabled. If it is enabled, use the show database-tables command to display the number of messages recorded in the table. Also, this property must be enabled to use the Play and Call-out links of the ME Call Logs tab. If disabled, there is some performance increase due to fewer writes to the database.

Default: disabled
Values: enabled | disabled
Example: **set media enabled**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to true, the master resumes its position. If set to false, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt-true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

database-threads-max: *Secondary property.* Sets the number of threads dedicated to database operation and maintenance. The minimum number of threads is two: one for writing to the database, and a separate thread for database maintenance (e.g., purging old records). By increasing the number of threads, you can improve database write performance, allowing multiple threads to write to the database simultaneously.

Default: 2
Values: Min: 2 / Max: 4
Example: **set database-threads-max 3**

sip-cache-size: *Secondary property.* Sets the number of entries allowed in the cache for the SIP database. When the system reaches the configured limit, it begins dropping entries, oldest first.

Default: 1000
Values: Min: 100 / Max: 10000
Example: **set sip-cache-size 3000**

performance: *Secondary property*. Sets the point of optimization for calls

Default: call-details

- **Values: call-rate**: The database cache works harder to increase call-rate, at the expense of call details. (This effects the ME Management System display of such things as call diagrams and the SIP Messages viewer.)
- **call-details**: Causes the system to accurately write out individual records until the database reaches the configured queue-depth. (When reached, the system begins caching the records.)

Example: **set performance call-rate**

dos-tcp-connect-multiplier: *Secondary property.* Sets the number of "hits" that the system should count for each connection (as opposed to data packets). With each data packet that matches a pattern, the system counts the match as an event; when the event count reaches a set threshold, it creates a DOS rule. Each connection, because it is more compute intensive, can count as more than one event. Set this multiplier to the number of events the system should count for each TCP connection.

Default: 5
Example: **set dos-tcp-connect-multiplier 3**

dos-tls-connect-multiplier: *Secondary property.* Sets the number of "hits" that the system should count for each connection (as opposed to data packets). With each data packet that matches a pattern, the system counts the match as an event; when the event count reaches a set threshold, it creates a DOS rule. Each connection, because it is more compute intensive, can count as more than one event. Set this multiplier to the number of events the system should count for each TLS connection.

Default: 10
Example: **set dos-tls-connect-multiplier 12**

sip-registers: *Secondary property.* Specifies whether to cache SIP REGISTER messages in the SIP message database. By disabling these messages, you can experience some performance increase. When **enabled**, the system writes them to the database in real-time. When set to **cached**, the system writes them to the database once every 5 minutes. Note that a more efficient way to disable writing of SIP REGISTER messages is to set the **message-logging** property of the **log-alert** object to **no-registers**.

Default: enabled
`Values: enabled | disabled | cached`
Example: **set sip-registers cached**

max-queue-depth: *Secondary property.* Sets the maximum number of write requests allowed in the database queue. When this value is reached, the system begins dropping requests.

Default: 4000
Example: **set max-queue-depth 4500**

caching-threshold: *Secondary property.* Sets the point in the queue at which the system begins using cached entries instead of single writes to the database. When the queue depth reaches the threshold set with this property, the system uses the resource-efficient method of scanning the cache and writing all entries for a particular row at once.

Default: 3500
Example: **set caching-threshold 4000**

write-mode: *Secondary property.* Controls the way the system writes records to the database. Typically, the system inserts records into the database. When set to **copy**, the

system activates an experimental database optimization method. Use this option only if instructed to do so by Technical Support.

Default: insert
Values: insert | copy
Example: **set edp-ping-timeout 45**

registration

Configures the master service for the registration process. In a cluster, the registration database only runs on the specified master and the selected backups. The **host-box** property establishes the master and selective mirroring. The first box listed is the master, while subsequent boxes have mirrored databases. Boxes not configured via the **host-box** property do not run the registration database. Instead, they use a local location cache. This object must be enabled for load-balancing of SIP processing (across backing interfaces configured via the **sip** object) to function correctly.

Syntax

```
config master-services registration
```

Properties

admin: Enables or disables registration services on the system. If **disabled**, the system cannot perform intracluster registration lookups. Also, you lose the persistence for registrations, which is what allows the system to perform rollover or failover operations.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which the registration service runs and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, the registration service is not available to the system.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the registration service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0
Example: **set group 1**

mirror-all-entries: Specifies whether to mirror location cache entries to all boxes in the cluster. When **enabled**, all entries are mirrored to all boxes; when **disabled**, they are not.

Mirroring helps failover recovery. In addition, you must set this property to enabled when the **route** object **alter-contact** property is set to **trunk-port-per-aor** for port forwarding.

Default: enabled
Values: enabled | disabled
Example: **set mirror-all-entries disabled**

mirror-location-cache: Specifies whether the ME mirrors the location cache to the other systems in the cluster. Disable this setting if the cluster is to handle up to one million SIP REGISTER requests. When **enabled,** the registration process mirrors the

location cache entries from the cluster to each SIP process, giving a slight performance improvement while limiting the total cluster to 250,000 users.

Default: enabled
Values: enabled | disabled
Example: **set mirror-location-cache disabled**

cache-poll-interval: Configures a timer that scans registration data and purges stale bindings and/or cache entries. This property controls how often the task repeats. To turn this feature off, set the **cache-poll-interval** to 0.

Default: 86400
Example: **set cache-poll-interval 43200**

max-poll-duration: Sets how long the database remains unlocked between polling of max-entries.

Default: 86400
Example: **set max-poll-duration 1200**

max-entries-per-poll: Sets the number of registration data entries that are scanned at a time. This is a performance optimization setting, which helps in preventing the database from being locked for excessive periods of time.

Default: 100
Example: **set max-entries-per-poll 1200**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to true, the master resumes its position. If set to false, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 100
Example: **set takeover-timer-value 200**

force-regdb-lookup: *Secondary property.* Sets whether the ME does a registration database lookup on every request. When **enabled**, the system does the lookup, ensuring that a cache entry always has cluster-wideup-to-date information. Use this when the bindings of an AOR are distributed on different backing boxes to ensure that the registration database has the complete list of bindings and the system can complete call forking in a failover scenario.

Default: disabled
Values: enabled | disabled
Example: **set force-regdb-lookup enabled**

ignore-from-tag: When enabled, the ME uses the call ID only to associate the registration with a session. When disabled, the ME uses both the call ID and the From tag to associate the registration to a session.

Default: enabled
Values: enabled | disabled
Example: **set ignore-from-tag disabled**

fatal-error-code: Enter the response code for a fatal error. The following are fatal errors:

- No registration-plan
- No proxy-contact allowed
- DNS lookup failure

Default: 403
Values: Min: 400 / Max: 699
Example: **set fatal-error-code 500**

fatal-error-string: Enter the text to be used in the fatal error.

Default: Forbidden
Example: **set fatal-error-string error1**

temporary-failure-code: Enter the response code for the temporary failure.

The following are temporary failures:

- Out of memory conditions
- Next hop host is down
- Unable to allocate trunk ports

Default: 403
Values: Min: 400 / Max: 699
Example: **set temporary-failure-code 500**

temporary-failure-string: Enter the text to be used in a temporary error.

Default: Forbidden
Example: **set temporary-failure-string failure1**

server-load

Configures the ME to calculate server load. This object must be enabled if your dial plan **arbiter** settings use **least-load** as the routing algorithm option. (The **arbiter rules** property sets the criteria by which the ME selects the server to which it forwards calls.)

Syntax

```
config master-services server-load
```

Properties

admin: Enables or disables server load calculation on the system. If **disabled**, the dial plan **arbiter** cannot use the least-load routing arbitration rule.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which the server load calculation runs and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, the calculation service is not available to the system.

Box 1 is the default box. Your first entry overwrites that default. Subsequent entries are added to the list as backup boxes.

Default: box 1
Example: **set host-box cluster box 2**

group: Associates the calculation service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to true, the master resumes its position. If set to false, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

update-timer: *Secondary property.* Sets how often the server-load master updates the other boxes in the cluster.

Default: 5000
Example: **set update-timer 3500**

call-failover

Configures failover for the media and signaling streams. As a master service, the configured host box distributes copies of the media and kernel rules to all backup boxes in a cluster. The ME uses the database on the host box, but enabling this master service ensures that there is an active copy of the database on another box in the cluster in the event of a failure.

Syntax

```
config master-services call-failover
```

Properties

admin: Enables or disables failover for media services on the system. If **disabled**, the dial plan **arbiter** cannot use the least-load routing arbitration rule.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which the media services database runs and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, the calculation service is not available to the system.

Box 1 is the default box. Your first entry overwrites that default. Subsequent entries are added to the list as backup boxes.

Default: box 1
Example: **set host-box cluster box 2**

group: Associates the media service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to true, the master resumes its position. If set to false, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

server-load: When enabled, the ME calculates the server load and distributes traffic counters around the cluster. Based on these distribution counts, each ME in a cluster knows the fail-over status.

Default: disabled
Values: enabled | disabled
Example: **set server-load enabled**

load-balancing

Configures boxes to host the load-balancing master service. These boxes are responsible for keeping the rule database up-to-date. They do not need to be the same boxes as the ones that host the head-end interfaces, although it is common to do so. (You can, for example, configure boxes in the cluster that only serve as host boxes, without any head-end interfaces or backing interfaces.)

Syntax

```
config master-services load-balancing
```

Properties

admin: Enables or disables maintenance of the rule database for the purposes of load balancing. If **disabled**, the sip **load-balancing** configuration is not operational.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which the rule database runs and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, the SIP-based load balancing is not available to the system.

Box 1 is the default box. Your first entry overwrites that default. Subsequent entries are added to the list as backup boxes.

Default: box 1
Example: **set host-box cluster box 2**

group: Associates the load balancing service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to **true**, the master resumes its position. If set to **false**, the backup service retains master control.

Default: false
`Values: true | false`
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

file-mirror

Sets all participating ME devices to share particular files (the types of files shared are preset in the ME operating system), such as media recordings, log files, etc. The file-mirror master-service distributes files to all devices listed as hosts for the service. It is used to make files highly available in the event that the box that created the file becomes unavailable. File mirroring includes keeping a record of each file in the file mirror database and also keeping a copy of each file on the local disk drive.

When configured, file mirroring works as follows:

1. When a file gets saved to the master file system, a record of the file is saved to the master database.
2. The master database then sends a message to all backup databases indicating a change and updating the backup.
3. The backup box(es) then compare their own database to their file system to determine if any files are missing (the new file is missing).
4. The backup then pulls the missing file(s) from the master file system.

Once the files are mirrored, you can play them back from any box that functions as a host. If accessing the file from a backup, the backup system first checks its database to make sure an entry is listed. It then checks its local disk for a copy of the file. If the file is not there (for example, an error during the pull operation) or is out of date, the backup again pulls the file from the master. File mirroring provides a secondary mechanism for assuring file availability. Non-host boxes also maintain a copy of the database and can pull files from the master as they are needed for processing. Use the **file-mirror-service** action to manage the mirrored files.

Syntax

```
config master-services file-mirror
```

Properties

admin: Enables or disables file mirroring on the system.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which file mirroring runs and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, the SIP-based load balancing is not available to the system.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the file mirroring process with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

file-mirror-directory: Identifies the location of the directory on the local disk to which the system writes the files. Supply an absolute path name.

Default: There is no default setting
Example: **set file-mirror-directory /cxc_common/mirror1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to **true**, the master resumes its position. If set to **false**, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

external-backup

Configures the system write all mirrored files to a backup server. With the specified frequency, the system writes all files contained in the **file-mirror file-mirror-directory** to the path specified. This is not a synchronization operation, it is strictly backup. Assuming the file transfer to the backup server was successful, the system does not attempt another transfer until the specified interval.

Syntax

```
config master-services file-mirror external-backup
```

Properties

admin: Enables or disables the configuration for the off-box backup.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

url: Identifies the location of the backup server to which the system backs up the files. Supply an absolute path name.

Default: There is no default setting
Example: **set file-mirror-directory /cxc_common/mirror1**

refresh: Specifies the frequency with which the system writes files to the backup server. See Setting Time and Time Intervals for information on entry format requirements.

Default: 15 minutes
Values: Min: 1 minute / Max: 14 days
Example: **set refresh 10:00**

route-server

Sets the route server master service, which manages the server process. The master service handles requests from local or remote ME devices or route server definitions. When presented with a request from the SIP process, the master service responds as follows, depending on the configuration:

- The master service retrieves one or more routes from the local (to the cluster) route server. This is the result if the session configuration **authorization** object is set to **Local**.
- The master service sends a Diameter request to retrieve the route(s) from the configured remote route server. This is the result if the session configuration **authorization** object is set to **Diameter**.
- The master service sends a request to an external policy service. This is the result if the session configuration **authorization** object is set to **WSDL**.

When multiple routes are returned, the dial-plan **arbiter**, if configured, resolves the best route.

The application can be configured in two ways: either intracluster or intercluster. Each has different configuration requirements, described below. Note that because the ME propagates route server rate table updates to backup boxes, you do not need to configure the **file-mirror** service for it.

See the *Oracle Communications OS-E Session Services Configuration Guide* for information on installing and implementing the route server import client, a web application that imports routes into the database.

Syntax

```
config master-services route-server
```

Properties

admin: Enables or disables the route-server master service on the system.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which route server is run and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, there is no route-server.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the route server process with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to **true**, the master resumes its position. If set to **false**, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

max-routes: *Secondary property.* Specifies the maximum number of route entries that can be imported from the rate table to the route server database. The available range for this property is determined by license restrictions.

Default: automatic
Values: automatic | *integer*
Example: **set max-routes 100000**

client-request-sender: Describes who sends requests to the route-server.

Default: only-master
- Values: only-master: The request goes from the route-server master in the client cluster.
- local-host-box: The host must be listed as a host-box for route-server master-service.

Example: **set client-request-sender local-host-max**

simple-updates: This parameter allows users to run controlled updates only without the possibility of running a simple update accidentally. When disabled and a simple update is executed, the user receives the error, "Only controlled updates are permitted by config."

Default: enabled
Values: enabled | disabled
Example: **set simple-updates disabled**

table-config

This object configures route-server tables.

Syntax

```
config master-services route-server table-config
```

Properties

table: Enter the name of the table you are creating.

Default: There is no default setting
Example: **set table table5**

description: A brief description of the table you are creating.

Default: There is no default setting
Example: **set description 1000-1050**

filename: Enter the name of the file containing the routes for this table.

Default: There is no default setting
Example: **set filename /routes**

sampling

Opens the mechanism for setting the interval at which the ME samples operational aspects of the system for, either, display in the ME Management System or for sending to an IBM Tivoli server. By setting sampling for a status provider, you can view data for that provider over a specified period of time. The ME supports two sampling targets: a Postgres SQL database and an IBM tivoli server. Set the provider data sent to the target using the **status** and **provider** objects.

When you execute a status-provider command from the CLI, the system just displays the results of the request at the time it was issued. Once you have enabled sampling, the master service stores the samples in its local database. You can you can select a status provider underneath **Trends** in the **Status** tab when using the ME Management System. The GUI trends graphs pull data from the database on the sampling master service box to display a time series graph of the results. Changes to the interval setting in the sampling subobjects do not effect the CLI results.

Note: If you have limited storage space, disable this feature if you are not using it. Otherwise, polling data is continuously written to the status database.

Configuring Summary Statistics for Display

There are certain status providers that require the **sampling** master service in order to report data.For example, the configuration required to display data for the **show sip-summary-by-box** (and related, **show sip-summary-rates-by-box**) involves the following:

1. Enable this master service (**sampling**). This begins collection of the relevant data.
2. Add a **database** target.
3. Select the **sip-summary-by-box provider**. Set the **interval** for this provider to a very short period (minimum allowed is 30 seconds).

Syntax

```
config master-services sampling
```

Properties

admin: Enables or disables status sampling on the system. In order to view the **Trends** graphs in the ME Management System, you must enable the sampling master service.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which status sampling is run and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, there is no sampling.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the status sampling process with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to **true**, the master resumes its position. If set to **false**, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

tivoli

Configures the ME to communicate with the IBM Tivoli server and to use it as a target for sampling data. (You can also send data to the local database.) The Tivoli server provide sampling status and event information to the IBM Tivoli Enterprise Manager. This information includes call information (e.g., call volumes and arrival rates, active calls, failed calls, etc.), network QoS (calculated Mean Opinion Score, Post-Dial Delay, Answer-Seize Ratio, Average Call Duration), and unit and cluster status (e.g., CPU usage, interface availability, etc.). The collected status and event information can be viewed and monitored using the IBM Tivoli Enterprise Portal desktop client.

Syntax

```
config master-services sampling tivoli hostName
```

Properties

admin: Sets the administrative status of the IBM Tivoli server configuration.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

protocol: Specifies the protocol the Tivoli server uses to communicate with the system.

Default: tcp
Values: tcp | udp
Example: **set protocol udp**

port: Specifies the port number over which the system communicates with the Tivoli server.

Default: 7500
Example: **set port 9650**

database

Configures a database for status sampling collection. When enabled, the local database is a target for sampling data. Setting these properties has no effect on other database activities (i.e., log or system databases). This data is then available as a time-series graph in the **Trends** section of the ME Management System **Status** tab.

Syntax

```
config master-services sampling database
```

Properties

admin: Sets the administrative status of the local status database. When **enabled**, the system collects data from all status providers configured under this object.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

duration: Specifies the number of days for which the system keeps collected provider data.

Default: 7
Example: **set duration 5**

status

Configures, through its subobjects, the status providers that report to the parent target. You can enable and disable the configuration through this object. There is one **status** object per target.

Syntax

```
config master-services sampling tivoli hostName status
config master-services sampling database status
```

Properties

None

provider

Selects which providers to collect data from and defines the time period for which the system displays status provider statistics. This data is then available as a time-series graph in the Trends section of the ME Management System **Status** tab (if the target is database) or sent to the tivoli server.

The following table describes each provider available for configuration through the **status** object and, if applicable, the Trend graph in which the system displays the information. The property options are the same for each.

Table 24–1 Available Providers

provider object (show command)	Reports...	Trend graph (database target only)
interface-details	...packet and error rates for each configured Ethernet interface.	interface-details
interface-throughput	...interface throughput statistics over various time intervals.	interface-throughput
system-heap	...usage and failure rates for each process running on the box.	system-heap-summary
trunk-groups	...carrier, exchange, server, trunk, as well as usage statistics.	tivoli only
server-load-db	...the peer, associated server, and aggregate load for each server that participates in load balancing.	tivoli only
switch-pool	...gateway(s) and associated swith(es), as well as configuration and usage statistics.	tivoli only
cpu-usage	...CPU utilization statistics over various time intervals.	cpu-usage
im-content-counters	...the number of times each word list or URL list has triggered a policy.	im-content-counters
sip-summary-by-box	....counters and values relevant to the SIP process. This is a base status class that is used by other status providers to calculate differential counters. See Configuring Summary Statistics for Display for configuration information.	N/A
sip-stack	...packets sent and received. Also displays total calls (both admitted and rejected) and active calls.	call-count packet-count
call-admission-control	...current and peak calls, calls in flight, active calls dropped, and in-flight calls dropped.	current-call-admission-control max-call-admission-control
tls-admission-control	...current TLS calls, as well as peak statistics for TLS calls, calls dropped, calls in-flight and, in-flight calls dropped.	tls-call-admission-control
active-calls	...maximum and average session duration.	session-duration
registration-status	...total received and declined registrations.	registrations

Table 24–1 (Cont.) Available Providers

provider object (show command)	Reports...	Trend graph (database target only)
location-auth-summary	...the number of registrations aborted (AORs challenged, those delegated to an upstream server due to lack of SIP server response, and those AORs initially unable to register because of too many concurrent requests).	registrations-aborted
location-reject-summary	...the number of registrations that were discarded, declined, or rejected.	registrations-rejected
dos-transport-counters	...the number of times the DOS transport policy has been triggered.	dos-transport-counters
dos-sip-counters	...the number of times the DOS SIP policy has been triggered.	dos-sip-counters
dos-url-counters	...the number of times the DOS URL policy has been triggered.	dos-url-counters

Syntax

```
config master-services sampling tivoli hostName status provider
config master-services sampling database status provider
```

Properties

admin: Specifies whether CPU usage statistics are stored for the specified provider in the target.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

interval: Defines how often the system polls the status provider for data. Once queried, the system writes the data to the status database. The data is kept (and displayed) for the amount of time set in the **duration** property of the target configuration. The interval property is required.

The system interprets the interval you enter from right to left, allowing you to enter only part of the time string. For example:

- 30=30 seconds
- 1:30=90 seconds
- 1:00:00=one hour

To enter a number of days, enter the number and the keyword days, and optionally, the time string. Enclose the entry in quotation marks.

Default: 1:00:00 (one hour)
Example: **set interval "5 days 12:00:00"**

jtapi

Enables integration between Microsoft OCS and third-party call control (3PCC) servers (e.g., BroadWorks, Cisco Call Manager and Avaya AES). The ME communicates with OCS using CSTA-over-SIP. (CSTA is a protocol used by OCS to communicate call state information, which can then be reflected in user presence status.) The ME communicates with the 3PCC servers using OCI (BroadWorks) or JTAPI (Cisco and Avaya). See Configuring Third-Party Call Control Server Objects, for information on configuring the servers.

Syntax

```
config master-services jtapi
```

Properties

admin: Enables and disables the system third-party call control master service.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Specifies the master box on which 3PCC services run and, optionally, backup boxes. See Master Services Description for more information. If there is no host box specified, no directory services can run.

Default: There is no default setting
Example: **set host-box cluster box 1**

group: Associates the 3PCC service with a VRRP group. Enter the number of a previously configured **vrrp** group. See Master Services in VRRP Configurations for a complete explanation.

Default: 0 (no grouping association)
Example: **set group 1**

preempt: *Secondary property.* Specifies whether the master service should retake the mastership if it has gone down and then returned to operation. If set to **true**, the master resumes its position. If set to **false**, the backup service retains master control.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property.* Specifies the number of milliseconds that the master service stays in "awaiting takeover" mode at boot time. When a box boots, each hosted master service waits for this period of time to determine if any existing boxes in the cluster are already running that service before assuming mastership.

Default: 1000
Example: **set takeover-timer-value 2000**

available-memory

The **available-memory** object allows you to enable a sampling interval for the ME to check the available memory.

Syntax

```
config master-services sampling database status available-memory
```

Properties

admin: Enables or disables the ME checking the available memory.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

interval: Defines how often the ME polls the status provider for data.

Default: 1:00:00 (one hour)
Values: Min : 30 / Max: 1036800
Example: **set interval 100**

cluster-server-load

Configures sampling for the SIP server load status.

Syntax

```
config master-services sampling database status cluster-server-load
```

Properties

admin: Enables or disables the status sampling configuration.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

interval: Defines how often the ME polls the status provider for data. The minimum polling time is 30 seconds.

Default: 1:00:00 (one hour)
Example: **set interval 1:45:00**

active-calls

Configures sampling for currently active calls.

Syntax

```
config master-services sampling database status active-calls
```

Properties

admin: Enables or disables the status sampling configuration.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

interval: Defines how often the ME polls the status provider for data. The minimum polling time is 30 seconds.

Default: 1:00:00 (one hour)
Example: **set interval 1:30:00**

events

Configures event service settings for the event core. The event core manages the distribution of events emitted from the ME. The event core sorts events into their specified channels and into a queue so they can be replayed by clients as needed.

Within the event core, there is a process known as garbage collection. Based on the configuration, garbage collection periodically scans all event channels and reclaims the oldest forwarded events, moving them back to available system memory.

Syntax

```
config master-services events
```

Properties

admin: Enables or disables the configuration that selects a master directory service.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

host-box: Defines the primary ME for the service. This references the **cluster > box** object. You can configure this property for as many **box** objects you have set to configure backup MEs in the event that the primary ME fails. The first **host-box** listed in the configuration serves as the primary host for the directory service.

Default: cluster box 1
Example: **set host-box cluster box 2**

group: Associates the master service with a VRRP group. If one interface of a VRRP pair is down, the group is considered down. If a service is associated with that group, the box hosted the downed VRRP pair stops the service and the backup box then restarts it. Enter the number of a previously configured VRRP group.

Default: 0
Values: Min: 0 / Max: 32
Example: **set group 5**

preempt: *Secondary property*. Specifies whether this master service should retake the mastership if it has gone down and then returned to operation. When true, the master resumes its position. When false, the backup service retains master control The setting applies to all master services, although only affects a service if that service becomes unavailable.

Default: false
Values: true | false
Example: **set preempt true**

takeover-timer-value: *Secondary property*. Specifies the number of milliseconds that this master service stays in "awaiting takeover" mode at boot time. When an ME boots, each hosted master service waits for this period of time to determine if any existing MEs in the cluster are already running that service before assuming mastership.

Default: 1000
Values: Min: 0 / Max: 4294967296
Example: **set takeover-timer-value 2000**

settings: *Secondary property*. Configures the settings for starting this process.

settings

Configures the settings for starting the event core process.

Syntax

```
config master-services events settings
```

Properties

channel-queue-size: Indicates the maximum number of events the ME can store for each channel.

Default: 256
Values: Min: 16 / Max: 2048
Example: **set channel-queue-size 350**

channel-queue-high-water: *Secondary property*. Enter the maximum number of worker threads all allotted to reclaim forwarded events.

Default: 224
Values: Min: 16 / Max: 2048
Example: **set channel-queue-high-water 250**

legacy-event-queue-size: *Secondary property*. Enter the maximum number of events lacking an associated channel held by the ME for forwarding.

Default: 2048
Values: Min: 32 / Max: 32768
Example: **set legacy-event-queue-size 3045**

max-channels: Enter the maximum number or active channels allowed at any given time.

Default: 32768
Values: Min: 1024 / Max: 54288
Example: **set max-channels 50000**

garbage-collection-interval: Enter the number of seconds the ME waits between garbage collection sweeps.

Default: 60
Values: Min: 0 / Max: 86400
Example: **set garbage-collection-interval 75**

gargabe-collection-reclaim: *Secondary property*. Enter the maximum percent of **channel-queue-size** the ME can reclaim per sweep.

Default: 25
Values: Min: 10 / Max: 100
Example: **set garbage-collection-reclaim 50**

garbage-collection-channel-min-idle: *Secondary property*. Enter the minimum time, in seconds, before an empty channel is reclaimed.

Default: 0 (the channel is reclaimed immediately)
Values: Min: 0 / Max: 3600
Example: **set garbage-collection-channel-min-idle 150**

25

Configuring Media Ports Objects

The media port pool defines the IP addresses and port ranges to assign to media streams on an Ethernet interface. These ports are used by the ME media services (media anchoring, NAT, recording, etc.), which are configured in the (default-) **session-config > media** object.

media-ports

Configures the media port pool on this interface, defining the available addresses and ports to use for media services. These are the services defined in the **session config > media** object. These ports can be used for media distribution, if the **cluster** object **share-media-port** property is set to true. Use the **show media-ports-process-units** command to view port limits and configuration for each processor. Use the **port-limit** object to define thresholds for active media port use.

You can configure the ME to use ports on the **cxc** processor. Do not configure media ports on ME ports (typically eth0 through eth3).

Syntax

```
config cluster box number interface ethX ip name media-ports
config cluster box number interface ethX vlan number ip name media-ports
config box interface ethX ip name media-ports
config box interface ethX vlan number ip name media-ports
```

Properties

admin: Enables or disables the media port pool on this system interface.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

base-port: Sets the base or starting port number to use for this port pool.

Default: 20000
Values: Min: 0 / Max: 65535
Example: **set base-port 35000**

count: Sets the total number of ports available for the media port pool.

A value of **0** implies the pool is empty an dis the equivalent of disabling the pool.

Default: 5000
Values: Min: 0 / Max: 65535
Example: **set count 2000**

idle-monitor: Enables or disabled the monitoring of idle ports by the ME. When ports are not in use, they should not receive traffic. When **enabled**, this property ensures that no traffic is sent to idle ports that are part of the media pool. If the system detects that an idle port is receiving traffic, the port is put into a quarantine list. An internal timer releases the port when traffic to the port stops for a period of seconds.

Default: enabled
Values: enabled | disabled
Example: **set idle-monitor disabled**

26

Configuring Messaging Objects

Messaging is the mechanism NN2600 uses to communicate among boxes. By configuring messaging, you are setting up a listening socket on an interface. This enables the interface to receive messaging traffic and participate in clustering and media partnering.

In clustering, the master box in a cluster looks through the configurations of all drones to find which interface each drone is using for messaging. (If multiple interfaces are configured, the master only communicates with one: the first it finds.) The master then communicates with the identified interface to share configuration and data. See Configuring Cluster, Box, and Interface Objects for more information.

In media partnering, you configure a specific IP address (on a different box) as a partner. On the box that houses that address, you would need to configure and enable messaging in order for partnering to work. See the cluster **media-partners** object for more information.

messaging

Configures and enables messaging for the specified interface. Messaging provides the mechanism for the ME media partners and clustering capabilities.

> **Note:** The protocol, port, and certificate set within this object must match the values set for these properties in the cluster>media-partner>partner object. If they do not match, the systems will not be able to communicate.

Syntax

```
config cluster box number interface ethX ip name messaging
config cluster box number interface ethX vlan number ip name messaging
config box interface ethX ip name messaging
config box interface ethX vlan number ip name messaging
```

Properties

admin: Sets the administrative state of the messaging configuration. When **disabled**, the parameters of messaging can still be configured, but the interface cannot participate in media partnering or clustering.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

protocol: Specifies the protocol the interface uses to communicate between systems.

Default: tcp
Values: tcp | tls
Example: **set protocol tls**

port: Identifies the Ethernet port through which the system listens for messaging sessions.

Default: 5132
Values: Min: 1 / Max: 65535
Example: **set port 13333**

certificate: Assigns the certificate that must be presented to participate in message exchanges if TLS is used as the protocol. Enter a reference to a previously configured certificate.

Default: There is no default setting
Example: **set certificate vsp tls certificate nnos-e.companyA.com**

27

Configuring Near-Side NAT Objects

You can configure the ME Engine to perform address translation on behalf of an enterprise firewall device. To do so, configure the parameters of the **near-side-nat** object to match the settings of your firewall and your media and SIP ports.

The ME uses network address translation (NAT) to change its private, backend address(es) to a public, routable address. NAT is defined in *RFC 1631, The IP Network Address Translator*. NAT ensures that internal private network addresses are rewritten so that they appear to come from the designated external network address. The ME modifies outgoing packets so that the return address is a valid Internet host (the firewall). The firewall then changes the destination address on incoming packets to the ME private address. This process protects the private addresses from public view. In addition, because the private address is not routable, any returning packets would not reach their destination. NAT provides a routable address through which the ME can maintain SIP and media connections.

The ME works with the firewall as follows:

1. You configure your firewall appropriately.
2. Configure the ME to match the firewall settings for IP addresses and ports.
3. Configure the ME **ip sip** and **ip media-ports** ports to recognize the ports specified in this object.
4. When the ME detects a packet coming from the firewall over the UDP or TCP listening port, it performs address translation where the ME changes the source address on outgoing packets from its own internal IP to the firewall public-facing address (set with the **public-ip** property).

See the following chapters for related information:

- Configuring Session Initiation Protocol Objects
- Configuring Media Ports Object

near-side-nat

Creates or edits a firewall configuration that allows the ME to perform address replacement for packets originating from the ME and destined for the Internet. By configuring the IP address of the public-facing interface on the enterprise firewall, the ME can produce a contact header that replaces its own, private IP address with the public-facing address of the firewall to allow completion of SIP calls.

When configuring the ME for address replacement, you must mirror the port forwarding of the firewall. The ports that you configure within this object indicate to the ME when it should do address replacement. For example, if you have configured

the UDP port at 5060, when the ME receives a packet from the firewall device using port 5060, it will replace its private IP address with the configured public address in its response.

Enter a name for the firewall configuration when opening this object.

Configuration Requirements

You typically configure UDP port 5060 and TCP ports 5060 and 5061 for SIP traffic. Be certain that the port numbers you enter here are the same as those you configured in the **ip sip** object.

In addition, you may configure the NAT pool addresses within this object, typically UDP ports 20000 through 30000. Be certain that the port numbers you enter here are the same as those you configured in the **ip media-ports** object.

Finally, the port numbers and IP address that you specify must match the configuration on your external firewall device.

Syntax

```
config cluster box number interface ethX ip name near-side-nat name
config cluster box number interface ethX vlan number ip name near-side-nat name
config box interface ethX ip name near-side-nat name
config box interface ethX vlan number ip name near-side-nat name
```

Properties

admin: Sets the administrative state of the external firewall configuration on the system, either **enabled** (active) or **disabled**. When disabled, you can still configure the firewall parameters, but the system will not do the address replacement necessary (if it is situated behind a near-side firewall).

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

public-ip: Sets the public-side address of the firewall positioned between the system and the Internet. The system will replace its own private, internal network address with the firewall public-facing IP address. You must supply a globally unique, routable value.

Default: There is no default setting
Example: **set public-ip 1.2.3.4**

udp-range<starting-port><count>: Specifies the UDP port number(s) that the system is listening for packets from the firewall device on. When the system receives a packet from the specified firewall port, it executes an address replacement of its own private address with the firewall public address in its response.

The typical SIP port is 5060; the typical media pool range is 20000 through 30000. Make sure that your UDP port configuration matches both the firewall configuration and the **ip sip** and **ip media-ports** object configurations.

If no UDP port is specified, the system has no port to listen for, and therefore, no address replacement occurs.

Default: There is no default setting
Example: **set udp-range 5060**

tcp-range<starting-port><count>: Specifies the TCP port numbers that the system is listening for packets from the firewall device on. When the system receives a packet

from the specified firewall port, it executes an address replacement of its own private address with the firewall public address in its response.

The typical SIP port is 5060 and 5061. Make sure that your TCP port configuration matches both the firewall configuration and the **ip sip** object configuration.

If no TCP port is specified, the system has no port to listen for, and therefore, no address replacement occurs.

Default: There is no default setting
Example: **set tcp 5060 2**

28

Configuring NTP Client and Server Objects

The ME system uses Network Time Protocol (NTP), Version 4 (described in RFC 1305) to synchronize its clock with network clocks. Systems using NTP all set and maintain their internal clock to the Coordinated Universal Time (UTC).

Synchronized time across the network provides packet and event time stamps and security certificate validation. Because computer clocks drift a few seconds a day, networked systems can be out of synchronization. NTP uses time signals from accurate time sources on the Internet to ensure all systems are synchronized.

Each ME device has a real time clock that uses NTP to maintain accurate time. You can configure the ME as an NTP client and/or as an NTP server. An NTP server responds to NTP client requests, using either external Internet time sources or the time set manually via the CLI, depending on the configuration. An NTP client queries the configured time server at the specified interval.

> **Note:** The NTP client and server objects are located in different places in the CLI hierarchy. You configure the client within the box object and the server within the interface object.

ntp-client

Opens the NTP client configuration object on the locally attached ME device, or opens the NTP client configuration object on the specified ME device in a cluster configuration.

Syntax

```
config box ntp-client
config cluster box integer ntp-client
```

Properties

admin: Enables or disables NTP client services on the local system.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

server: Specifies the IP address of the NTP server. This is the address of the device that responds to this NTP client requests.

Enter the IP address in dotted decimal format or a host name.

Default: There is no default setting

Example: **set server 192.168.10.10**

poll-interval: Sets the poll interval for NTP updates in minutes. The interval is the number of minutes between NTP client requests to the NTP server.

Default: 10
Values: Min: 1 / Max: 1440
Example: **set poll-interval 50**

ntp-server

Opens the NTP server configuration object on the specified ME Ethernet or VLAN interface. When you enable NTP server functionality, the interface will respond to NTP client requests with its current time. To set the time on the box, do one of the following:

- Use the **clock** action
- Set the server as a client to an external Internet time server using the **ntp-client** object

Syntax

```
config box interface ethX ip name ntp-server
config cluster box integer interface ethX ip name ntp-server
```

Properties

admin: Enables or disables the NTP server on the current system Ethernet or VLAN interface.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

29

Configuring Policy Objects

The policies configuration object sets both operating policy for enterprise servers and users and denial of service (DOS) policy. This chapter details server and user policy. For information on DOS policy, see Configuring Denial of Service (DOS) objects.

The policy configuration object allows you to create policies that govern the routing of SIP phone calls and instant messages to recipients, and then back to the original caller or sender. A policy is set of one or more rules, each with defined conditions that operate using a specific SIP session configuration. When a SIP message registers with the ME, the SIP message (such as a SIP INVITE) is processed against the configured policies for matching strings. If a string match occurs in any of the configured policies, then those policies are enforced on that SIP call session.

If there are no policies that match the SIP message and call registration information, the call is either forwarded to the SIP call recipient or the call is dropped based on the settings in the default session configuration.

Rules and Condition Lists

A policy is identified by a unique name and can contain one or more rules. With each rule, you configure a condition list with a set of properties, as well as the session configuration properties that control the SIP session when the policy is being enforced. Condition list configuration is described in Configuring Condition List Objects.

Session Configuration

The session configuration defines the SIP call session settings to apply to SIP calls for which a configured policy exists. When a SIP call is received at the ME, the system registers the call and checks all policies and rules to determine how the call should be processed, including those services (such as registration, location, authentication, and accounting services) that should be applied to the SIP call. See Configuring Session Configuration Objects, for a description of each session configuration object.

For more information on ME policies, refer to the *Oracle Communications OS-E Session Services Configuration Guide*.

policies

Opens the gateway to the DOS policy and session policy objects.

Syntax

```
config vsp policies
```

Properties

None

session-policies

Sets the parameters used to configure policies for this VSP, for enterprise servers, and for users.

Syntax

```
config vsp policies session-policies
```

Properties

default-policy: Sets the default policy to apply to all SIP sessions. This policy acts as a baseline policy in effect at all times. More specific policy rules may override the default settings. Enter the path to a previously configured policy.

Default: There is no default setting
Example: **set default-policy vsp policies session-policies policy default**

outbound-policy: Apply a session configuration policy to a session as it egresses the ME.

Default: There is no default setting
Example: **set outbound-policy vsp\tls\certificate policy1**

policy

Opens the new or existing named policy object for editing.

Syntax

```
config vsp policies
```

Properties

None

rule

Opens the new or existing named policy rule object for editing. Properties of this object enable or disable the policy rule, and set a user-specified description of the rule. From this object you can open the **condition-list** object to set policy statements or the **session-config** object to configure baseline SIP session characteristics.

Syntax

```
config vsp policies session-policies policy name rule name
```

Properties

admin: Enables or disables the current rule associated with the named policy. If disabled, all other enabled rules under the named policy will still be checked against incoming SIP messages.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

description: Specifies a textual description for this policy rule. Enclose the description in quotation marks and specify up to 64 characters.

Default: There is no default setting
Example: **set description "Rule to apply to SIP messages from LCS clients belonging to domain company123.com"**

session-config

Opens the session-config object for editing. It is through this object that you set the session characteristics for SIP calls matching the specified policy. For a description of all session configuration subobjects, see Configuring Session Configuration Objects.

Syntax

```
config vsp policies session-policies policy name rule name session-config
```

Properties

None

30

Configuring Preferences Objects

The preferences configuration object allows you to set operational preferences for the ME. The **preferences** object allows you to add user-defined enumeration strings to the selection of default strings that exist in the ME configuration file. When editing objects that use enumeration strings (policy predicates, for example), the ME presents both the default strings and those that you have added.

> **Note:** Although you can add enumeration strings in the CLI, they are only available through the ME Management System.

preferences

Opens the references configuration object from which you can set specific operations for the ME.

Syntax

```
config preferences
```

Properties

gui-preferences: Allows you to configure user-defined enumeration strings to the selection of default strings that exist in the ME configuration file.

click-to-call: Configures click-to-call application profiles.

gui-preferences

Adds user-defined enumeration strings to the selection of default strings that exist in the ME configuration file. When editing objects that use enumeration strings (policy predicates, for example), the ME presents both the default strings and those that you have added.

Use this object to add objects configured in other applications, for example in SIP extensions, to the ME configuration file. Often the extensions contain components that could be useful for defining policy rules. By adding the enumeration strings to the configuration file through this object, the system "remembers" the string so that you can easily use it as a building block in your definitions.

Although you can add strings with this object, the strings will not be available for use from the CLI. They are only available from the GUI. You can confirm that the strings have been added, however, through the **show** object, as shown in the example. In addition, the action of this object occurs automatically in the GUI. If you enter a

unique string in one of the categories, the ME Management System automatically retains that string in the configuration.

Syntax

```
config preferences gui-preferences
```

Properties

enum-string: Adds an entry to the list of displayed entries of the specified type. Type a question mark at the command line to see a list of available types.

Default: There is no default setting
Example: **set enum-strings timezone homeDST**

reverse-dns: Specifies whether the system should make reverse DNS lookups.

Default: false
Values: true | false
Example: **set reverse-dns true**

trap-poll-interval: Specifies how often the system checks for new traps. When a trap is detected, it is forward to the trap targets configured with the **snmp** object.

Default: 10
Values: Min: 1 / Max: 65535
Example: **set trap-poll-interval 15**

phone-path-map: Maps a phone type to the web configuration URL (if it has one). In the Tools section of the ME Management System, the Phone Registration allows you to click on the IP for a phone and the system dispatches to the vendor phone configuration URL.

This property configures the link to the web configuration URL, which is useful in cases where the configuration is not at the top most level. For example, if the phone IP address is 192.168.10.10, the web configuration application may be found at http://192.168.10.10 or it may be at http://192.168.10.10/configuration/.

The phone path map allows the administrator to supply an additional path to the web configuration or turn it off completely. For example, if you have a specific Cisco phone of the type CSCO/7, you can add a map entry of type CSCO/7 and then either turn off the configuration altogether or provide an additional path so that the Configure Phone link will go to http://192.168.10.10/configuration for that type of phone.

Enter a phone type, whether you want web configuration enabled (set to true if the phone type has management), and the URL for the web configuration page.

Default: There is no default setting
Values: phone-type true *URL* | *phone-type* false *URL*
Example: **set phone-path-map csco/7 true 192.168.10.10/configuration**

show-unlicensed-features: Sets whether the system displays all features or only those licensed for a system. If set to **true**, the system displays all features. For those that are not licensed, the system displays "Available with upgrade." When set to **false**, the unlicensed option is not displayed.

Default: true
Values: true | false
Example: **set show-unlicensed-features false**

more-than-one-session-in-call: *Secondary property.* Sets the system to group sessions in a call based on the call ID. When set to **false**, the default, the system does not group

sessions and performance is improved. Set to **true** to configure the call logs sessions search in the ME to look for more than one session in a call.

Default: false
Values: true | false
Example: **set more-than-one-session-in-call true**

channel: *Secondary property.* Swaps the display of several visual elements within the ME. By selecting one of the preconfigured channels, you change the ME display of logo and certain text to reflect the changes implemented for that channel. The default is **none**, which displays the ME images. Do not change this property unless you have reason to display different channel indicators.

Default: none
Values: none | *name*
Example: **set channel nortel**

max-config-list-size: *Secondary property.* Specifies the maximum number of items to display in a configuration list (for example, lists of interfaces or configuration pool entries). When a list contains more items than the number set here, no items of that type are shown in the configuration tree. Instead, they can be displayed on a separate page with paging and search capabilities. The tree contains a link to that page.

Default: 100
Values: Min: 10 / Max: 1000
Example: **set max-config-list-size 500**

default-call-log-search: *Secondary property.* Controls the initial display that results from clicking the **Call Logs** tab in the ME Web. If **enabled**, when the tab is clicked the system displays the Sessions page, which lists all sessions in the database for all users. When **disabled**, the database entries are not loaded; the page is blank with the links available on the left. You can then click the **Sessions** link to display the database.

Default: enabled
Values: enabled | disabled
Example: **set default-call-log-search disabled**

display-oem-documentation: *Secondary property.* When disabled, the Documentation link is not shown on the home page and the Web services portal display only shows a message indicating that no documentation is available. This property only works when the **channel** property is set to something other than **none**.

Default: disabled
Values: enabled | disabled
Example: **set display-oem-documentation enabled**

display-home-page: *Secondary property.* When **disabled**, the home page links at the left side of the web UI page are not displayed. This property only works when the **channel** property is set to something other than **none**.

Default: enabled
Values: enabled | disabled
Example: **set display-home-page disabled**

display-footer: *Secondary property.* When **disabled**, footers are not displayed when the Status, Call Logs, Event Logs, and Tools tabs are selected. This property only works when the **channel** property is set to something other than **none**.

Default: enabled
Values: enabled | disabled
Example: **set display-footer disabled**

summary-preferences

Sets the display page that appears when you either launch the ME or click on the **Home** tab from within the ME. You can set the summary display for both box and cluster displays.

Syntax

```
config preferences gui-preferences summary-preferences
```

Properties

None

cluster-summary-preferences

Sets the content of the summary page that displays in the ME when you choose "Cluster" in the **Get summary for:** pull-down. By default, the ME displays status summary for all choices. When you set **cluster-summary-preference**, the ME overwrites the selection of all status summaries with only the **cluster-summary-preference** you entered. To add additional status summaries, re-execute the command. The order in which you enter the preferences is the order in which they are displayed on the ME status summary page. If you delete all your entries, the ME returns to the default and displays all cluster status summaries. If your box is not part of a cluster, only the box-addresses and box-summary choices are available.

The following table describes each **cluster-summary-preference** option:

Table 30–1 Cluster-Summary-Preference Options

Select...	To display...
box-addresses	the IP address for each configured box in the cluster.
box-summary	the IP address, administrative state, and build information for each configured (physical) box in the cluster.
master-services	the configured host, if applicable, for each master service.
active-vrrp-interfaces	the operational state and configuration of VRRP interfaces, helping to determine interface status to more easily troubleshooting cluster problems.

Syntax

```
config preferences gui-preferences summary-preferences cluster-summary-preferences
{box-addresses | box-summary | master-services | active-vrrp-interfaces}
```

box-summary-preferences

Sets the content of status summary page that displays in the ME when you choose "Box" in the **Get summary for** pull-down. By default, the ME displays status summary for all choices. When you set **box-summary-preference**, The ME overwrites the selection of all status summaries with only the **box-summary-preference** you entered. To add additional status summaries, re-execute the command. The order in which you enter the preferences is the order in which they are displayed on the The

ME status summary page. If you delete all your entries, the ME returns to the default and displays all box status summaries.

The following table describes each **box-summary-preference** option:

Table 30–2 Box-Summary-Preference Options

Select...	To display...
box-status	the IP address, administrative state, and build information for the box. Links on the page provide access to interface, process, and sensor information.
master-services	the list of master services hosted on the selected box.
up-time	system uptime since the last reboot.
system-info	CPU utilization statistics over the time frame specified for that status provider. Links on the page provide access to memory and alert data.
registration-info	the total number of client bindings stored in the registration table, broken down into ignored, terminated, and declined registrations. A link on the page displays the total number of entries in the location cache.
call-info	the call counts of active, dropped, and total calls, as well as a link to display call trends.
call-duration	the average and maximum call duration, as well as a link for a session duration trend graph.
location-info	a count of entries in the location cache and location bindings table.

Syntax

```
config preferences gui-preferences summary-preferences box-summary-preferences
{box-status | master-services | up-time | system-info | registration-info |
call-info | call-duration | location-info}
```

monitored-calls-threshold

Sets the values that will cause the display in the ME Web Call Logs' **Monitored URIs** link to display in red, highlighting that call performance has crossed the set threshold. The **Monitored URIs** link executes and then displays the results of the QoS loopback monitoring tests.

Syntax

```
config preferences gui-preferences monitored-calls-thresholds
```

Properties

mos-threshold: Sets the value below which the Mean Opinion Score (MOS) result triggers notification. MOS is a subjective measurement and an “opinion” of the audio quality heard by the listener on a phone. The MOS measurement reveals the call quality from 1 (pure noise) to 5 (pure fidelity).

Default: 4.0
Example: **set mos-threshold 4.2**

latency-threshold: Sets the value above which the result of the QoS loopback monitoring test triggers notification. Latency is defined as the delay in getting RTP packets from the system to the endpoint (e.g., phone).

Default: 400
Example: **set latency-threshold 500**

jitter-threshold: Sets the highest allowable packet variation (jitter) allowed on a call. When the jitter exceeds this configured threshold, the system highlights the results.

Default: 60
Example: **set jitter-threshold 45**

packets-dropped-percent-threshold: Sets the maximum percentage of dropped packets allowed. When the percentage of packets dropped exceeds the configured threshold, the system highlights the results. The number of packets sent is determined by the type and duration set in the **loopback** action.

Default: 20
Example: **set packets-dropped-percent-threshold 15**

accounting-calls-preference

Manages the display of the call detail record fields in the ME Web. See the *Oracle Communications OS-E Session Services Configuration Guide* for a description of each field.

Syntax

```
config preferences gui-preferences accounting-calls-preference
```

Properties

None

select

Sets the fields to display on the **Accounting Calls** page of the ME Web **Call Logs** page. By default, fields are displayed in the order in which they are entered. To change the order, use the CLI **move** command or the arrows within the ME Web listing. If you do not configure this property, the ME displays Setup Time, Type, Method, From, To, Call ID, and the calculated MoS score on the source leg. To view all accounting fields for a call, click the Call Record link for the call on the **Accounting Calls** page.

To configure this object, re-execute the object for each column you want displayed.

Syntax

```
config preferences gui-preferences accounting-calls-thresholds select columnName
```

Properties

None

tab-order

Manages the display and/or order of tabs in the ME Web. See the ME: Using the Management Tools for a description of each tab.

Syntax

```
config preferences gui-preferences tab-order
```

Properties

None

web-tab

Sets the tabs to display in the ME Web. By default, and with full permissions, the following tabs are displayed:

- Home
- Configuration
- Status
- CallLogs
- EventLogs
- Actions
- Service
- Keys
- Access
- Tools
- Portal

If you do not see all of these tabs, it is probably because access to that capability is turned off.

Note that you must re-execute this object for each tab you want displayed. For example, if you execute only once, supplying **Actions** as the *tabName*, the **Actions** tab will be the only tab visible after you save the configuration. To re-order the tabs put continue to display them all, you must execute the object 11 times.

Syntax

```
config preferences gui-preferences tab-order web-tab tabName
```

Properties

None

click-to-call

Configures click-to-call application profiles. These profiles are then referenced with the **web-services > application** property. The click-to-call function is reserved for demonstration use only.

Syntax

```
config preferences click-to-call
```

Properties

record: Specifies whether to record the transaction. If **disabled**, the system has a record of the call but does not retain the content. The system applies the session configuration referenced in the **dont-record-entry** property. When **enabled**, the system records the call, which you can then view through the ME **Call Logs**, and applies the session configuration referenced in the **record-entry** property.

Default: enabled
Values: enabled | disabled
Example: **set record disabled**

phone-to-call: Specifies the address of record to which the system is dialing out.

Default: There is no default setting
Example: **set phone-to-call sip:5555555432@192.168.10.10**

record-entry: References a session configuration to apply if the **record** property is set to **enabled**.

Default: There is no default setting
Example: **set**

dont-record-entry: References a session configuration to apply if the **record** property is set to **disabled**.

Default: There is no default setting
Example: **set dont-record-entry "vsp session-config-pool-entry entry dont-record-click"**

31

Configuring Pre-Session Configuration Objects

The pre-session-configuration object allows you to globally apply SIP settings to your network before SIP call sessions are established. SIP methods that you want to shield from the network, for example, can be blocked using settings in the **pre-session-configuration** object. This means that you do not need to create a policy rule to block a particular SIP method that is globally forbidden from your network.

pre-session-config

Opens the **pre-session-config** object for editing. Through this object you set the parameters used by the VSP to alter SIP traffic before a session is established.

Syntax

```
config vsp pre-session-config
```

Properties

unregistered-sender-directive: Sets the action the system takes when it receives a packet with an unknown sender in the "From" field of the INVITE packet. Use the **registration-requirement-level** setting in the **route** or **source-route** object to define what is considered unknown.

Default: allow; if you select refuse, the default result code is 400

- **Values: allow**: The ME permits the packet to proceed toward its destination.
- **discard**: The ME immediately discards the packet.
- **refuse [***result-code***][***result-string***]**: The ME discards the packet but sends a response to indicate having done so. The response includes an error code (default of 400 but you can enter any value between 400 and 699) and an optional description.

Example: **set unregistered-sender-directive refuse 404 "unknown sender"**

optional-header-error-handling: Determines how the ME handles optional header parsing.

Default: strip; if you select reject, the default result code is 400 withe string "Bad Request - Optional Header Failed Parsing"

- **Values: strip**: The ME strips the malformed header in its entirety.
- **ignore**: The ME ignores the error and sends the malformed header on the egress leg.
- **discard**: The ME immediately discards the packet.

- **reject [***result-code***][***result-string***]**: The ME discards the packet but sends a response to indicate having done so. The response includes an error code (default of 400 but you can enter any value between 400 and 699) and an optional description.

Example: **set optional-header-error-handling ignore**

block-method-settings

Enables or disables blocking of SIP methods and specifies the SIP method to block. When a method type is blocked, the ME drops the packet. Repeat the command to block additional SIP methods.

The following table lists the methods that ME can block, along with brief definitions from the related RFC:

Table 31–1 Block Methods

Method	Description	Found in...
INVITE	Asks a server to establish a session.	RFC 3261, SIP: Session Initiation Protocol
ACK	Facilitates reliable message exchange for INVITEs.	RFC 3261
OPTIONS	Allows a UA to query another UA or a proxy server as to its capabilities.	RFC 3261
BYE	Terminates a specific session or attempted session.	RFC 3261
CANCEL	Asks the UAS to cease processing the request and to generate an error response to that request.	RFC 3261
REGISTER	Sends a request to a... registrar. A registrar acts as the front end to the location service for a domain, reading and writing mappings based on the contents of REGISTER requests.	RFC 3261
MESSAGE	Allows the transfer of Instant Messages.	*RFC 3428, Session Initiation Protocol (SIP) Extension for Instant Messaging*
INFO	Allow for the carrying of session related control information that is generated during a session.	*RFC 2976, The SIP INFO Method*
NOTIFY	Contains the modified session description.	*RFC 2848, The PINT Service Protocol: Extensions to SIP and SDP for IP Access to Telephone Call Services*
SUBSCRIBE	Indicates that a user wishes to receive information about the status of a service session.	RFC 2848

Table 31–1 (Cont.) Block Methods

Method	Description	Found in...
REFER	Requests that the recipient REFER to a resource provided in the request...and provides a mechanism allowing the party sending the REFER to be notified of the outcome of the referenced request.	*RFC 3515, The Session Initiation Protocol (SIP) Refer Method*
PRACK	Plays the same role as ACK, but for provisional responses.	*RFC 3262, Reliability of Provisional Responses in the Session Initiation Protocol (SIP)*
PUBLISH	Provides a framework for the publication of event state information.	*RFC 3903, Session Initiation Protocol (SIP) Extension for Event State Publication*

Syntax

```
config vsp pre-session-config block-method-settings
```

Properties

admin: Enables or disables the blocking of SIP methods before SIP sessions are established.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

block-method: Specifies the SIP method(s) to block from the network. Re-execute the command to add each block method. See the above table for method descriptions.

Default: There is no default setting
Example: **set block-method refer**

sip-header-settings

Enables or disables the SIP header rules (set with the rule object) that are applied to the network before a SIP session is established.

Syntax

```
config vsp pre-session-config sip-header-settings
```

Properties

admin: Enables or disables the SIP header policy that is applied to the network before SIP session establishment. You configure the rules of the policy with the **rule** object.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

rule

Sets an optional rule description, the conditions of the rule, and the type of action to apply to SIP headers that match those conditions. To open the rule object, specify a name. Initially, the order in which they were created establishes the precedence for the rule (if you create multiple rules). Use the **move** command to change the order.

A condition is a predicate statement that the ME matches the SIP headers against. If a header matches any of these statements, the ME takes the action defined by the **action** property. Note that the conditions are AND'd together.

Follow these rules when creating conditions:

- If you enter a header name only, the ME applies the **action** to that header.
- You can enter only one header name. To match on more than one, create multiple rules.
- If you enter a header value only, the ME applies the action if any header matches that value.
- You can enter more than one value, but should do so with extreme care, as the rule will take a single action against all matches.
- If you enter a name and value, the ME applies the action to the named header if it has the specified value.

Syntax

```
config vsp pre-session-config sip-header-settings rule name
```

Properties

description: Sets the user-specified text description for the rule. Use the **show -v** command from the **sip-header-settings** level to see all configured rules with descriptions.

Default: There is no default setting
Example: **set description "SIP header policy to apply prior to session establishment."**

condition: Sets whether to match on a name of a SIP header and/or a value in one of the fields. (See the more detailed explanation in this command description.)

Default: There is no default setting

- **Values: match-header <***header-name***>**: sets the name of the header to match on. To see possible name matches, enter **set condition match-header ?** at the prompt.
- **match-header-and-value <***header-name***><***reg-exp***>**: sets the header name and field to match against. In this case, both entries must match.
- **match-value <***reg-exp***>**: matches all SIP headers against the text string you enter. Enclose a string with spaces within quotation marks.

Example: **set condition match-header To**

action: Sets the action to apply to packets in which the conditions of this rule are met.

Default: There is no default setting

- **Values: discard-packet**: the system immediately discards the packet.
- **strip-header**: the system removes the SIP header from the packet. Use this, for example, if a particular header causes problems for another SIP device in the network.

- **alter-header <***new-sip-header***>**: the system changes the content of the header to the text you supply. If your condition list contained a match-name statement, the system alters the named header. If your condition list contained only a value, the system alters all headers that contain that value.

Example: **set action alter-header 800**

32

Configuring Processes Objects

The **processes** objects allow you to configure memory allocation settings on a per-process basis. The settings in these objects should not be modified unless specifically instructed to do so by Oracle Technical Support personnel.

When the system boots, the ME tallies the amount of physical memory in the system and scales each processes heap based on that total.

processes

Opens the **processes** object, from where you specify memory configuration settings for each process.

Syntax

```
config box processes
config cluster box number processes
```

Properties

None

process

Sets memory allocations for each specified process. When opening this object, specify the name of the process for which you would like to modify the parameters. Enter a process type to open this object. Type a question mark at the command line to display possible process types.

> **Note:** Do not change these settings without explicit instructions to do so from Technical Support.The process configuration settings are dynamic. If the ME runs out of memory in one of the heaps, you can change the configuration and the ME expands the heap. If you make a value smaller than the current heap size, the MEdoes not free any memory, but future allocation attempts fail.

When setting the heap sizes, you select either the default value, a maximum, or a specific value. You can display the current values with the **show system-heap** command.

The following table shows the available heap sizes:

Table 32–1 Available Heap Sizes

Setting	Definition
default	The ME calculates default values at boot time, based on the total amount of memory available to the system.
max	There is no limit to the heap size; the ME is free to use all the memory available on the process.
value	The ME limits the heap size to the specific value you enter. Enter the keyword **value** and the number of megabytes to assign.

Syntax

```
config box processes process processName
config cluster box number processes process processName
```

Properties

system-heap-init: Specifies the initial amount of memory that the system reserves for a process.

Default: default
Values: default | max | value *megabytes 8-3072*
Example: **set system-heap-init value 2048**

system-heap-locked: *Do not modify this value unless told to do so by Technical Support.* Specifies the amount of memory that should be locked. A value of 0 configures the system to lock no memory. Locked memory cannot be swapped out by the Linux kernel.

Default: disabled
Values: enabled | enabled *megabytes*
Example: **set system-heap-locked enabled 128**

system-heap-shring-interval: Specifies how often in minutes the system checks the system heap to see if any memory can be returned to the operating system. A value of 0, the default, specifies that the system not make that check.

Default: 60
Values: Min: 0 / Max: 1440
Example: **set system-heap-shrink-interval 90**

pools-shrink-interval: Specifies in minutes how often the system checks its memory pools to see if any memory can be returned to the system heap. A value of 0, the default, specifies that the system not make that check.

Default: 10
Values: Min: 0 / Max: 1440
Example: **set pool-shrink-interval 45**

tls-heap-max: Specifies the portion of the **system-heap-max** that is used for TLS processing. If set to **default**, the maximum TLS heap size is calculated based on the value of the **max-number-of-tls** property in the **vsp** configuration.

Default: There is no default setting
Values: default | max | value *megabytes 8-3072*
Example: **set tls-heap-max 512**

33

Configuring RADIUS-Group Objects

A RADIUS group is a uniquely named object that defines the authentication and accounting services associated with a group of RADIUS servers. Including a RADIUS group in the VSP configuration allows the ME system (the RADIUS client) to perform user authentication and forward accounting and SIP call detail records to the RADIUS servers.This means that you have flexibility to create as many unique RADIUS groups as you need, and include them with the VSPs of your choice.

Setting Server Priority

The ME allows you to set server priority to influence which server receives authentication requests. To use this feature, set the **authentication-mode** property in the **radius-group** object to **prioritized**. Set the priority for the server with the **priority** property of the **server** object. The ME then manages authentication requests using the following logic:

1. The ME always sends an authentication request to the server with the highest priority. The lower the number, the higher the priority.
2. If the request times out, the ME sends the request to the next-highest-priority server. This timeout status is applicable for that request only. The ME will forward the next request to the highest priority server.
3. The ME continues with this action until either a server replies with an Accept or a Reject, or until there are no more configured servers. If there are no more servers to try, the ME rejects the call.

Note that in **prioritized** mode, the ME does not determine that servers are dead due to consecutive failures. As long as a server is enabled in the configuration, the ME continues to forward requests, regardless of the number of failures.

When configuring for prioritization, it is important to set different priority values for the servers. Otherwise, the ME randomly selects from servers with the same value, negating the effects of prioritized mode. If that should happen, the ME generates an event indicating that multiple servers have the same priority. The following two examples illustrate how the ME forwards requests with multiple servers of the same priority:

- Server A has a priority of 1, and servers B and C have a priority of 2. The ME sends all requests to server A, with the highest priority, first. If A does not respond, the ME picks randomly between B and C.
- Servers A and B have a priority of 1, and server C has a priority of 2. The ME selects randomly between A and B, and sends all requests to that server first. If that server times out, the ME sends all requests to the other highest-priority server.

(For example, if the ME picks A first, and it times out, it then sends requests to B, not C.)

radius-group

Configures a RADIUS group, to which you add servers using the **server** object. Setting up a RADIUS group in one or more VSP configurations allows the ME system (the RADIUS client) to perform SIP traffic authentication and to forward accounting and SIP call detail records to the RADIUS servers. (To setup authentication for user access, use the access **radius** object.)

Specify the new or existing RADIUS group name using up to 16 alphanumeric characters with no blank spaces.

Syntax

```
config vsp radius-group targetname
```

Properties

admin: Enables or disables the system RADIUS server group configuration. When **enabled**, the system forwards SIP call detail records to configured RADIUS group server(s).

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

accounting-mode: Sets the RADIUS group accounting operational algorithm.

Default: duplicate

- **Values: round-robin**: If you configure multiple accounting servers in the accounting group, the round robin algorithm performs continued accounting requests to primary and secondary servers until a valid accounting response is received.
- **duplicate**: The duplicate algorithm issues multiple duplicate accounting requests to all servers in the RADIUS accounting group. A duplicate accounting request uses the same client source IP address and source UDP port.
- **fail-over <***retries***>**: If you configure multiple accounting servers, the failover algorithm forwards accounting requests to secondary servers should the current accounting server fail. You can specify up to 256 failover servers.

Example: **set accounting-mode round-robin**

authentication-mode: Sets the RADIUS group authentication operational algorithm.

Default: failover 3

- **Values: round-robin**: The round robin algorithm performs continued authentication requests to primary and secondary servers until a valid authentication response is received.
- **fail-over <***retries***>**: The failover algorithm forwards authentication requests to secondary servers should the current authentication server fail. You can specify up to 256 failover attempts to other servers.
- **prioritized**: The ME forwards authentication requests to the server with the highest assigned priority. If that server does not respond, the system forwards the request to the next highest priority server. Set the priority with the **server > priority** property. See Setting Server Priority for more information.

Example: **set authentication-mode round-robin**

type: Sets the type of SIP accounting record to use. Currently, the only valid SIP accounting record type is Cisco.

Default: cisco
Example: **set type cisco**

include-in-default: Specifies if this RADIUS group is to be included in the default RADIUS authentication and accounting target group.

If set to **true**, authentication and accounting requests are forwarded to this group if there are no configured policies that govern or redirect RADIUS requests to other servers.

Default: true
Values: true | false
Example: **set include-in-default false**

digest-attributes-format: Sets the correct Digest authentication attributes format for use with RADIUS.

Default: draft-sterman-aaa-sip-03

- **Values: draft-sterman-aaa-sip-03**: Set to this experimental format if you are using FreeRADIUS.
- **draft-ietf-radext-digest-auth-05**: Set to this early proposed standard if you are using Steel-Belted RADIUS.
- **rfc-4590**: Set to RFC 4590 is you are using the standard RADIUS.

Example: **set digest-attributes-format rfc-4590**

send-session-id: Specifies whether the system correlates RADIUS access requests with accounting requests. When **true**, the system sends the Acct-Session-ID attribute in its RADIUS auth-requests. When **false**, this attribute is sent only in accounting messages.

Default: true
Values: true | false
Example: **set send-session-id false**

include-digest-domain-in-user-name: Specifies whether to append the user's domain name to the RADIUS User-Name attribute. Enable this property if the RADIUS server requires the domain name to be included in the attribute. If the User-Name attribute already contains a domain name, the system does not take any action.

Default: disabled
Values: enabled | disabled
Example: **set include-digest-domain-in-user-name enabled**

send-user-agent: Specifies whether to include the User-Agent header value in the RADIUS Auth-Request message. If set to **true**, the ME includes the User-Agent header in the Connect-Info RADIUS attribute.

Default: false
Values: true | false
Example: **set send-user-agent true**

service-type: Maps a RADIUS service type to a SIP message type. If the system authenticates a message type that has a mapped service type, it will include that Service-Type attribute in the RADIUS request. If a service type has not been mapped to the message type the system is authenticating, but there is a mapping for the message type OTHER, the system includes the OTHER service type in the request. If there is no mapping for the actual or the OTHER method, then the system does not include any Service-Type attribute in the request.

Default: There is no default setting
Example: **set service-type other**

application: Enables or disables this normalization plan. When **enabled**, the system provides normalization for matching SIP messages. When **disabled**, you can configure the plan properties but the system does not apply it.

Default: authentication

- Values: authentication: Use SIP authentication
- routing: Use Oracle SIP routing

Example: **set application routing**

custom-accounting: Enables or disables free-form accounting services for this VSP. When enabled, free-form accounting is used. When disabled, the ME uses existing, predefined CDRs.

Default: disabled
Values: enabled | disabled
Example: **set custom-accounting enabled**

server

Identifies and configures the RADIUS servers that are part of this RADIUS group. Enter a host name or IP address to identify the server.

Syntax

```
config vsp radius-group name server serverName
```

Properties

admin: Enables or disables the system RADIUS authentication and accounting server configuration. When enabled, authentication and SIP call accounting records are forwarded to the specified server IP address and port numbers.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

authentication-port: Sets the UDP port over which the system RADIUS client sends authentication requests to the RADIUS server.

Default: 1812
Values: Min: 1 / Max: 65535
Example: **set authentication-port 1800**

authentication-sockets: Sets the number of sockets reserved for request IDs on a server. With one socket, the default, the 8-bit number space allows up to 255 outstanding requests per server. Assign additional sockets if you have a high-volume application that requires sending many requests at one time. Each additional socket increases capacity by 255 requests.

Default: 1
Values: Min: 1 / Max: 8
Example: **set association-min-lifetime 25000**

accounting-port: Sets the UDP port number over which the system RADIUS client sends accounting requests to the RADIUS server.

Default: 1813
Values: Min: 1 / Max: 65535

Example: **set accounting-port 1801**

secret-tag: Specifies the shared secret used to authenticate transactions between the system RADIUS client and the RADIUS server. See Understanding Passwords and Tags for information on the ME two-part password mechanism. Enter up to 32 alphanumeric characters.

Default: There is no default setting
Example: **set secret-tag abc123xyz**

timeout: Specifies the time (in milliseconds) to elapse before an accounting or authentication request to a RADIUS server times out. If the request times out, the system retries the request for the specified number of attempts before the request is forwarded to the next RADIUS server in the configuration.

Default: 1000
Values: Min: 1 / Max: 65535
Example: **set to-user strip-off-to 10**

retries: Sets the number of times the system retransmits an accounting or authentication request if the RADIUS server does not respond.

Default: 3
Values: Min: 2 / Max: 5
Example: **set retries 4**

window: Sets the maximum number of simultaneous requests the system client can send to the RADIUS server. Note that if you set multiple sockets with the **authentication-socket** property, this window value is a per-socket allowance.

Default: 64
Values: Min: 8 / Max: 255
Example: **set window 255**

priority: Configures a priority for the server. Set this property if the **authentication-mode** property of the **radius-group** object is set to **prioritized**. The lower the value, the higher the priority. Note that each server in a RADIUS group must have a different priority for prioritization to work correctly. See Setting Server Priority for more information.

Default: 1
Values: Min: 1 / Max: 99
Example: **set priority 5**

34

Configuring Routing Objects

The route object allows you to manually create static IP routes to destination networks and hosts (routers) connected to the Internet. A static route provide a constant route to a specific network or host router. This static route takes precedence over dynamically learned routes and is not overwritten by dynamic routing protocols (such as RIP and OSPF) running in your network.

ME uses a static route when its routing table does not have a route to other devices in the network. By defining a default route, the ME can send traffic to other devices in the network even if you do not define any other routes. You configure static routes for each IP interface that would benefit from the functionality.

routing

Opens the routing configuration object for editing. The **routing** object allows you access to the object that creates one or more static routes that are added to the system routing table.

Syntax

On a public IP interface:

```
config cluster box integer interface ethX ip name routing
config cluster vrrp vinterface vxID ip name routing
config cluster box integer interface ethX vlan integer ip name routing
config box interface ethX ip name routing
config box interface ethX vlan integer ip name routing
```

Properties

None

route

Adds to or edits a static route in the system routing table. These static routes provide a constant route to a destination host (router) or destination network that connects to the public Internet. Or, when creating a static route within a virtual firewall, you are creating entries that configure the ME to be able to reach subnets within the private network. For any static route you configure, you also define the local router or gateway as the next-hop router to the destination. You can also configure a gateway for a default route; this is the route the ME uses when no other entries match the destination.

The ME uses the route configuration to determine how to resolve destination addresses. The match is always on the most specific address available. Use the **show routing** command to display the ME routing table.

Enter a name for the route. This name appears in the routing table display. You can create as many static routes as you wish, but each must be created in an individual route object.

Syntax

On a public IP interface:

```
config cluster box integer interface ethX ip name routing route name
config cluster box integer interface ethX vlan integer ip name routing route name
config cluster vrrp vinterface vxID ip name routing route name
config box interface ethX ip name routing route name
config box interface ethX vlan integer ip name routing route name
```

Properties

admin: Enables or disables this route configuration.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

destination: Identifies the destination address that this object represents. You must also configure a corresponding **gateway** for the route. The system resolves the destination to the gateway you specify. Destination network and host addresses are added to the system routing table.

If a destination does not match any other route in the routing table, the system uses the **default** route (0.0.0.0/0) and its corresponding gateway. The default route directs any data addressed to any network numbers that are not explicitly contained in the routing table.

Without a gateway configured, this value is not functional.

Default: default

- **Values: network**: An IP address and mask to match the destination network
- **host**: A host IP address
- **default**: Creates 0.0.0.0/0

Example: **set destination network 192.168.124.0/24**

gateway: Sets the gateway or next hop IP address for the packet.

Default: 0.0.0.0
Example: **set gateway 192.168.124.6**

metric: Associates a cost with the static route that the system adds to its services route and route DB tables. The lower the metric the more preferred the route. The system chooses the more preferred route when there are multiple interfaces available on the same network.

Default: 1
Values: Min: 0 / Max: 4294967295
Example: **set metric 10**

35

Configuring Secure Shell Objects

Secure Shell (SSH) Server Version 2 on the ME provides secure client/server communications, remote logins, and file transfers using encryption and public-key authentication. To establish a secure connection and communications session, SSH uses a key pair that you generate or receive from a valid certificate authority (CA). The ME uses the OpenSSH daemon for SSH support.

An SSH session allows you to transfer files with Secure Shell File Transfer Protocol (SFTP), providing more secure transfers than FTP and an easy-to-use interface. SSH uses counters that record SFTP activity over the SSH connection.

When running SSH on the ME, the SSH session is transparent and the CLI appears just as it would if you were connecting from a console or over Telnet. The ME implementation of SSH does not support all the user-configurable attributes typically supported by SSH workstations. If you try to change an attribute that the ME does not support, you will receive a notification that the setting failed.

ssh

Configures an SSH listener on an IP interface. Note that although you can configure SSH settings on each IP interface, there is only one SSH daemon running, with one configuration. The SSH configuration is an aggregate of the separate interface SSH configurations.

Syntax

```
config cluster box number interface ethX ip name ssh
config cluster box number interface ethX vlan number ip name ssh
config box interface ethX ip name ssh
config box interface ethX vlan number ip name ssh
```

Properties

admin: Sets the administrative state of the SSH protocol, either **enabled** (running) or **disabled**. When disabled, the parameters of SSH can still be configured, but do not become active until **admin** is set to **enabled**.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

max-sessions: Sets the maximum number of concurrent SSH sessions allowed, enforced at the box level. The enforced value is an aggregate of the SSH session limits set on each IP interface that has SSH enabled. For example, to enforce a limit of five

total SSH sessions per box, you could set IP "A" to an SSH session limit of two and IP "B" to an SSH session limit of three, for a total of five.

Default: 8
Values: Min: 1 / Max: 32
Example: **set max-sessions 4**

idle-timeout: Specifies the amount of time in seconds allowed to elapse before the ME closes the SSH session due to inactivity.

Default: 600
Values: Min: 60 / Max: 86400
Example: **set idle-timeout 300**

port: Identifies the known TCP port through which the ME listens for SSH sessions.

Default: 22
Values: Min: 1 / Max: 65535
Example: **set port 25**

mode: Sets the version of SSH the system should use. Be aware multiple vulnerabilities exist in SSH version 1, and it is therefore not secure. Use the **compatibility** setting to allow the system to determine the version in use by the remote system and set its own version to match.

Default: compatibility
Values: ssh-1 | ssh-2 | compatibility
Example: **set mode ssh-1**

authentication: Sets the authentication method(s) the ME uses to authenticate users, either **password** or public key. To use public key SSH authentication on the ME, generate a public/private key pair, install the public key on the system, and install the private key on your SSH client. You can select either SSH version 1 (**RSA**) or SSH version 2 (**public-key**) authentication. You can select If you do not specify any authentication methods, the system applies the OpenSSH defaults.

Default: There is no default setting
Values: password | public-key | rsa
Example: **set authentication password**

account: Sets the account(s) to use for SSH authentication, either root or a user account (ssh). Use the **ssh password** action to set up an account password if the account type is set to **ssh**. By default, the system uses the root account. However, if you set the account type to **ssh**, the root account no longer applies. You can then add it back in using this property.

Default: root
Values: root | ssh
Example: **set account ssh**

log-level: Specifies to the SSH daemon the level of SSH events to generate and send to the ME event log. The SSH component sends all events of that level and higher.

Default: verbose
Values: quiet | fatal | error | info | verbose | debug | debug1 | debug2 | debug3
Example: **set log-level debug**

36

Configuring Server Objects

Enterprise services are SIP-enabled real-time communication systems and collaboration services. By configuring the ME to recognize a particular enterprise service, you are drawing that service under the security protection of the ME, preventing application-level attacks. These services allow an organization to support, among others:

- IP PBX hosted VoIP services
- Enterprise instant messaging systems
- Mobile devices
- Presence-based applications

Enterprise services work by establishing an application (directory) server, a SIP component in the enterprise. Client programs access the server to look up user entries, and the server expects a certain set of users to be using it. For example, a server might be an IBM/Lotus Sametime server. Using the ME, you would configure a link between that server and a directory containing the Sametime users.

Specifically, the ME supports the following enterprise servers:

- IBM Lotus Sametime Server
- Microsoft Live Communications Server (LCS) 2005
- Nortel Multimedia Communications Server (MCS)
- Avaya IP telephony PBX
- A generic SIP source/destination
- A generic SIP registration server
- A DNS group
- A SIP connection
- An H.323 gateway

> **Note:** While you can configure directory services at any time, you must enable the master-services > directory object for the ME to use the service. See Configuring Master Services Objects for more information.

For detailed information on the ME enterprise gateways, refer to the *Oracle Communications OS-E Session Services Guide*.

Normalization In the Servers Group

The server pool **server-pool-admission-control** objects provides inbound and outbound normalization settings to apply to calls going to or from the server. Use **outbound-normalization** for calls destined for a server; use **inbound-normalization** for calls received from the server. The objects properties are common for servers, gateway, and trunk groups, and are described in Configuring Normalization objects.

Server Descriptions

The following sections briefly describe each server type that is supported by the ME.

Note: The ME may present server types other than those described here. Oracle does not recommend using these servers.

SIP Gateway Description

Configures a generic SIP server. For example, it could be a SIP proxy, a SIP application server, or a PSTN gateway. By configuring the public switched telephone network (PSTN) gateway, you can configure the ME to allow enterprises to continue call operations even if their provider server is busy or down. The way that the ME handles unavailable servers and future call routing is controlled by the local-mode setting of the **routing-settings** property. See the routing-setting attribute descriptions for more information.

Specify the SIP URI for the gateway, in the form SIP:*gatewayIdentity*. For example, SIP:sip-server@broadsoft.com.

SIP Host Description

The SIP host is a generic server description that allows the ME configuration to include a server configuration for a non-explicit server type.

DNS Group Description

DNS-group is a server configuration template for servers that do not use a server pool configuration because they can be resolved by DNS. When the ME receives a REGISTER request, if the domain is the same as that configured for a dns-group, the ME clones the configuration of that dns-group for the server. The ME then does three DNS lookups: NAPTR, SRV, and A: to resolve the transport protocol, port, and address. (If multiple records are found, the ME uses the preference set in the DNS server to select the primary.) The ME then adds the server to the server pool. If the domain from the REGISTER is different from the dns-group, the ME creates a new server object and clones the configuration from the dns-group. Note that you must configure a dial plan and/or registration plan to point to the **dns-group**.

SIP Connection Description

The SIP connection server type provides a client/server model between the ME and customer premise equipment. The ME fills the server role, while the connection (line) between the CPE and the ME acts as client. This connection may be a single line, a shared line, or a group of shared lines to the enterprise or a residence. The point of connection on a shared line (the CPE) represents one or multiple direct inward dial (DID) numbers. Behind the CPE, however, may be many more endpoints. In this

configuration, the client initiates, or re-establishes in the event of failure, the connection with the ME.

Using this server type allows you to create a configuration specific to an AOR. For instance, it allows you to control the number of concurrent calls to (emission control) and from (admission control) the specific AOR. You can override the global location cache settings that set the number of concurrent calls, and allow more or fewer calls based on the connection.

Additionally, the ME can learn client transport information through dynamic registration. Within the **registration-plan**, you can reference a **sip-connection** type server. Then, when a REGISTER comes in from the CPE (sip-connection server) and matches a registration-plan, when the ME installs a location cache entry, it saves the sip-connection name and reference in the location entry. If the sip-connection has unknown transport information (host, port, transport, local port and so on), the ME can use the dynamic learn feature (if enabled), to derive the sip-connection transport information from the client registration.

server

Opens the server configuration object to allow setting the parameters for communication between the directory server and the ME, supporting the following enterprise services:

- IBM Lotus Sametime Server (sametime)
- Microsoft Live Communications Server 2005 (lcs)
- Nortel Network Multimedia Communications Server (mcs)
- Avaya IP telephony PBX (avaya)
- A generic SIP source/destination (sip-host)
- A PSTN gateway (sip-gateway)
- DNS group
- SIP connection
- H.323 gateway

The ME uses strict, tight, or loose matching rules to map. A REGISTER request or INVITE must match according to what you have configured within this object. By default, the ME uses strict rules for mapping, meaning that it only maps to names that contain an exact match of the domain name you entered. If you have configured the **domain-alias** property, the ME uses tight rules, meaning it will map on either the name or alias. If you set the **domain-subnet** property, loose matching rules are in effect.

Note: While you can configure directory services at any time, you must enable the master-services > directory object for the ME to use the service. See Configuring Master Services Objects for more information.

Note: When creating or editing a SIP gateway, specify the SIP URI for the gateway, in the form SIP:gatewayIdentity. For example, SIP:sip-server@broadsoft.com.

Routing-Setting Definitions

The **routing-setting** property allows you to select one or more server attributes. The following table describes each of these attributes in detail.

Table 36–1 Server Attributes

Attribute	Description
normalization	When the ME receives a request (e.g., an INVITE or REGISTER), it checks the host portion of the request. When **normalization** is enabled, if the host portion matches the domain name, domain name alias, the subnet, or a server-pool entry, the ME changes the host name to the server domain name. By making this change, the ME can then match the request on a configured dial or registration plan.
auto-tag-match	When enabled, if the server has a configured directory, the ME automatically creates a dial plan and registration plan for the server.
auto-domain-match	When enabled, the ME creates a domain-based dial and registration plan for the server. The plan uses the domain-exact **request-uri-match** type (matches any USER field and a HOST field containing the exact domain name specified). The ME uses the domain name configured for the server, and the resulting plans have no normalization or session configuration. The action associated with the plans is **delegate**.
pstn-backup	When a server is down (not reachable), if **pstn-backup** is not selected, the ME changes the state of the server to "not available." Any dial or registration plan with reference to that server is removed from the call routing or registration routing table. In its normal state, the ME operates in provider mode, forwarding calls to a provider application server. If the server fails, and the ME has location information for the provider, it forwards calls locally. Otherwise, the ME forwards calls to a PSTN gateway. You configure the gateway using the **pstn-gateway** server object. This is called local mode. When enabled, an unavailable server state changes to "local mode." Plan entries stay in the routing tables.
outbound-association	When enabled, the ME uses its management system to derive associations when originating a SIP message. When disabled, the ME sends the message straight through, which results in better performance.
cxc-from	When enabled, the ME changes the From header to the ME local identity when proxying registrations to an upstream server. When disabled the original URI remains in place, meaning that the REGISTER is derived directly from the sender.

Table 36–1 (Cont.) Server Attributes

Attribute	Description
local-mode	Sets the ME to always function in local mode. In *provider mode*, the normal state, the ME forwards calls to a provider application server. If the server has failed, and the ME has location information for the provider, it forwards calls locally. Otherwise, the ME forwards calls to a PSTN gateway. You configure the gateway using the sip-gateway server object. This is called *local mode*. The ME detects provider failure using the **failover-detection** property setting of the **server-pool-admission-control** object. When local mode is not selected, the system stays in local mode until the it determines that the server has resumed functionality. When the server again becomes available, the ME reverts the registrar peer back to provider mode and retries calling through the provider. If the call is successful, the ME stays in provider mode.

Service-Type Definitions

The **service-type** property allows you to set the way the ME handles INVITE and REGISTER requests and database exchanges. The following table describes each of these settings in detail.

Table 36–2 Service-Type Definitions

Attribute	Description
provider	Specifies the server as a provider peer, which means that the ME proxies INVITE and REGISTER requests. If a peer has proxy-registration configured, then the ME proxies the registration. In other words, the ME intercepts the REGISTER, stores the contact information in the location cache, and generates a new request with the ME as the contact. If the peer does not have proxy-registration configured, the ME does not proxy the registration. Instead, it checks the call routing table to see if the request URI matches a provider. If there is a match, the ME forwards the request to the peer listed in the table. If there is not a match, the ME walks the call routing table, entry by entry. If a match is found in the table, the INVITE is forwarded to the peer. Otherwise, the INVITE is forwarded to the default outbound proxy.
internal	Specifies the server as an internal peer (internal to the enterprise or subscriber community under a single service provider). If the internal peer is a SIP registrar, then the ME Engine does location database exchanges with it. When an INVITE matches a dial plan for an internal peer, the ME first looks up the location cache for forwarding information. If found, the INVITE is forwarded to the location binding. Otherwise, the INVITE is forwarded to the internal peer.

Table 36–2 (Cont.) Service-Type Definitions

Attribute	Description
external	Specifies the server as an external peer (external to the enterprise or subscriber community under a single service provider). If the external peer is a SIP registrar, then the ME challenges the REGISTER request with RADIUS or DIAMETER, and if successful, then passes the request on to the external peer. The ME stores the location binding in the cache in case the external peer forwards future INVITEs. When an INVITE matches a dial plan for an external peer, the ME first looks up the location cache for forwarding information. If found, the INVITE is forwarded to the location binding. Otherwise, the INVITE is forwarded to the external peer.

Syntax

```
config vsp enterprise servers sametime string
config vsp enterprise servers lcs string
config vsp enterprise servers mcs string
config vsp enterprise servers avaya string
config vsp enterprise servers sip-gateway SIP:gatewayIdentity
config vsp enterprise servers h323-server string
config vsp enterprise servers sip-host string
config vsp enterprise servers dns-group string
config vsp enterprise servers sip-connection string
```

Properties

description: Associates a text string with a server configuration. The string displays in some event logs and status providers to help identify the target.

Default: There is no default setting
Example: **set description E911server**

admin: Specifies whether the system uses this server in the current session. If **enabled**, the system uses this server. If **disabled**, the system does not use this server.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

carrier: Associates a text string with a server. The string can later be used to group and categorize servers.

Default: There is no default setting
Example: **set carrier server1**

domain: Identifies a domain to be used by the system for server normalization. In cases where the server is associated with:

- a single domain: Enter that domain.
- multiple domains: Enter one of the domain names.
- no domain: Enter another valid domain on the system. (This might be the case with a PSTN gateway for example.)

Also, you must set this property if you enable the **settings > local-directory-based-user-services** property without configuring the **directory**

property (to assign a directory to a server). Set this domain name to match user SIP addresses to the appropriate server (by use of the domain).

Default: There is no default setting
Example: **set domain voip.companyABC.com**

routing-tag: Controls which outbound interface SIP traffic uses. The routing-tag indicates the interface on the server where a SIP message with a matching routing-tag would be forwarded. The SIP message derives its routing-tag from the session config or IP interface classification-tag, depending on the configuration scenario. This property sets the initial routing tag for a server. If there is a policy match that applies to the server, and that configuration sets a routing tag (with the **routing-settings** ingress- and egress-classification-tag), the policy setting takes precedence.

Default: There is no default setting
Example: **set routing-tag lcs1**

failover-detection: Determines the method to use to detect when an upstream server peer is unavailable (and has resumed availability).

Default: none

- **Values: none**: The system does no checking, and the server peer always appears available, even when down.
- **auto**: The system uses an internal algorithm to count transaction failures. If a message to the server fails, the system resends the message the number of times defined in the **sip-settings > max-retransmissions** property. When the system reaches the retransmission threshold, it increments the **dead-threshold** count and then starts the retransmission process again. When the server reaches the failure threshold (set with the **dead-threshold** property), the system changes the server state to DOWN and sends no further requests. The fallback timer (set with the **dead-fallback-interval** property) activates. When the timer expires, the system decrements the server dead count by one and can again send requests to the server. If it receives no response, the system again increments the count and reaches the threshold, restarting the process. If the server responds, the system decrements the dead count again, until the count reaches 0. Note that if there is a major transport error, such as "no socket," the system skips the retransmission step and increments the **dead-threshold** count. Use this setting in the case where a server does not respond to SIP OPTIONS messages. You must also enable the **vsp > auto-server-fail-detection** property (which acts like a master switch) when using this option.
- **ping**: The system uses the sip-ping utility to check server availability. SIP ping sends SIP OPTIONS messages to a peer. When enabled, the system pings its peers at an interval defined in the **ping-interval** property. If the peer is not operational (determined by the dead-threshold property, the system switches to local mode if pstn-backup is checked (**routing-setting** property), or to unavailable mode if pstn-backup is not checked. When the system is again able to successfully ping the peer, it reverts to provider mode.
- **register**: The system determines server availability by sending a REGISTER request to the server. If there is no response from the provider, the server is assumed down. Note that you must configure a user for the server, with the appropriate password. (See the **user** and **password-tag** properties).

Example: **set failover-detection auto**

failover-termination: Sets whether calls are disconnected if the ME detects server failure. If the **failover-detection** property is enabled (set to anything other than **none**),

and a server fails, the ME terminates all calls going through that server when this property is **enabled**. When **disabled**, connections are unaffected.

Default: disabled
Values: enabled | disabled
Example: **set failover-termination enabled**

domain-alias: Sets the system to recognize an alias domain as the domain in which the server resides. You can enter as many aliases as you choose.

Default: There is no default setting
Example: **set domain-alias eng.companyABC.com**

domain-subnet: Sets the IP subnets serviced by this server.

Default: There is no default setting
Example: **set domain-subnet 1.2.3.4/16**

local: Sets the server local IP address.

Default: There is no default setting
Example: **set local 192.168.1.4**

ping-interval: Sets the number of seconds between ping packets sent between the system and the SIP registrar server.

Default: 10
Example: **set ping-interval 30**

dead-threshold: Specifies the number of transaction failures (and resulting retransmissions) a server can experience before the server state is changed to DOWN. This threshold is used in the **auto** and **ping** options of the **failover-detection** property.

Default: 4
Values: Min: 1 / Max: 255
Example: **set dead-threshold 15**

dead-fallback-interval: Sets the fallback timer for the server. During this period, the system does not send REGISTER or INVITES to the down server. After the timer expires, the system decrements the **dead-threshold** by 1. This timer is used in the **auto** and **ping** options of the **failover-detection** property.

Default: 300
Values: Min: 30 / Max: 65535
Example: **set dead-fallback-interval 450**

handle-3xx-locally-routing-lookup: Specifies whether the system should do a dial plan lookup on the REQUEST URI of a newly generated INVITE based on a 302 response received from this server. This property works in conjunction with the **handle-3xx-locally** property of the **sip-settings** session config object. If that property is enabled, the system generates a new INVITE when it receives a 3xx response. The system puts the contents of the CONTACT field in the REQUEST URI of the new INVITE. You should **disable** this feature if your server is configured to explicitly forward the message to a specified third sever. Set this to **enabled** if the message is coming from an endpoint with instructions to forward the message to a different AOR.

Default: enabled
Values: enabled | disabled
Example: **set handle-3xx-locally-routing-lookup disabled**

unregistered-sender-directive: Sets the action the system takes when it receives a packet with an unknown sender in the "From" field of the INVITE packet. Use the **registration-requirement-level** setting in the **route** or **source-route** object to define what is considered unknown.

Default: allow; if you select refuse, the default result code is 400

- **Values: allow**: The system permits the packet to proceed toward its destination.
- **discard**: The system immediately discards the packet.
- **refuse [result-code][result-string]**: The system discards the packet but sends a response to indicate having done so. The response includes an error code (default of 400 but you can enter any value between 400 and 699) and an optional description.

Example: **set unregistered-sender directive refuse 404 "unknown sender"**

inbound-session-config-pool-entry: Specifies a session configuration entry to apply to all inbound traffic destined for this server.

Default: There is no default setting
Example: **set inbound-session-config-pool-entry "vsp session-config-pool entry inboundPolicy"**

outbound-session-config-pool-entry: Specifies a session configuration entry to apply to all outbound traffic from or through this server.

Default: There is no default setting
Example: **set outbound-session-config-pool-entry "vsp session-config-pool entry outboundPolicy"**

server-type: Sets the server version or function. The type that you select is dependent on the server type that you are configuring. See below for the options for each server.

sametime: Identifies the server as version 3.1, operating as either a direct or proxy server (SIP connector).

- sametime-31
- sametime-31-sip-connector
- sametime-75
- sametime-75-sip-connector

  lcs: Identifies the LCS server version (2003 or 2005) and the function (server-only or access proxy).

- lcs-2003
- lcs-2003-access-proxy
- lcs-2005
- lcs-2005-access-proxy
- ocs-2007
- ocs-2007-edge-server

  mcs: Identifies the server as Nortel Networks MCS.

- nortel-mcs

  avaya: Identifies the server as an Avaya PBX.

- avaya

  sip-host: When using the generic SIP server, identifies the function the server is fulfilling.

- windows-messenger
- sip-proxy

- sipura
- snom
- polycom
- office-communicator
- nortel-mcp

sip-gateway: Identifies the server as a SIP gateway.

- sip-proxy
- sipx

dns-group: Identifies a DNS group.

- dns-group

sip-connection: Identifies the connection type.

- windows-messenger
- sip-proxy
- sipura
- snom
- polycom
- office-communicator
- nortel-mcp

h323-server: Identifies the server as an H.323 gateway.

- h323-gw
- h323-gatekeeper

peer-identity: *Secondary property.* Specifies a unique URI to identify a remote peer. The ME uses the peer identity (usually found in the FROM header) in peer-to-peer SIP messaging to identify where a SIP message is from. The system can use this information to identify a peer with which to swap location database records.

Default: There is no default setting
Example: **set peer-identity sip:nnos-e@companyABC.com**

directory: *Secondary property.* Creates the link between the server and the name directory it uses. Enter the full path name to a configured directory.

Also, if you enabled the **settings > local-directory-based-user-services** property**,** you must either set this property (to assign a directory to a server) or set the **domain** property to match user SIP addresses to the appropriate server (by use of the domain).

Default: There is no default setting
Example: **set directory vsp\enterprise\directories\notes-directory ABCco**

user: *Secondary property.* Assigns a user name that the system must supply when challenged by the server (the name of the person qualified to log into this directory server). Enter the name expected by the server, do not create it here. The user name and password-tag (below) are used for authentication between the system and server. This name must match the username configured on the server.

Default: There is no default setting
Example: **set user admin**

password-tag: *Secondary property.* Specifies the tag associated with the shared secret used to authenticate transactions between the system and this server. This is the tag associated with the password that the system must supply when challenged by the server. See Understanding Passwords and Tags for information on the ME two-part password mechanism.

This password associated with this tag must match the password configured on the server.

Default: There is no default setting
Example: **set password-tag secure**

routing-setting<attributes>: *Secondary property.* Sets attributes of the server. See Routing-Setting Definitions for a description of each option.

Default: normalization+outbound-association
Example: **set routing-setting auto-tag-match+auto-domain-match**

loop-detection: *Secondary property.* Sets the aggressiveness with which the system enforces call routing loop detection. (The most aggressive requires the fewest parameters to match for the system to drop the call.)

Default: tight

- **Values: strict**: If the system receives a call from a SIP proxy, and a DNS or dial-plan lookup resolves that the source and destination address are the same, the system drops the call. This is the most aggressive.
- **tight**: If the system finds the source and destination address, transport protocol, and port to be the same, it drops the call.
- **loose**: The system uses standard SIP loop detection (based on the VIA header). When the system finds its own address in the list of SIP proxies traversed, it allows the packet through.

Example: **set loop-detection strict**

service-type: *Secondary property.* Specifies the way in which the system handles INVITE and REGISTER requests and database exchanges. See Service-Type Definitions for complete descriptions of each option.

Default: provider
`Values: provider | internal | external`
Example: **set service-type internal**

peer-max-interval: *Secondary property.* Specifies the value the system writes to the max-interval setting for a peer. When doing registration delegation, the system changes the expiration value in the REGISTER request to the specified **max-interval** when delegating it to the upstream server. The system saves the expiration value recorded in the 200OK from the upstream server to its location cache. If you enter **0**, the peer value remains.

Default: 86400
Example: **set peer-max-interval 0**

peer-min-interval: *Secondary property.* Specifies the value the system writes to the min-interval setting for a peer. When doing registration delegation, the system changes the expiration value in the REGISTER request to the specified **min-interval** when delegating it to the upstream server. The system saves the expiration value recorded in the 200OK from the upstream server to its location cache. If you enter **0**, the peer value remains.

Default: 3600
Example: **set peer-min-interval 0**

registration-request-timeout: *Secondary property.* Specifies the number of seconds the system waits for a response after sending a REGISTER request to this server. If the system does not receive a response within the configured time, it sends, to the endpoint, notification that service is not available.

Default: 10
Example: **set registration-request-timeout 8**

default-policy: *Secondary property.* Sets the name of the policy to apply if no more specific policy is in place. Enter a previously configured policy reference.

Default: There is no default setting
Example: **set default-policy vsp\policies\session-policies\policy lcs**

user-group-policy: *Secondary property.* Specifies the policy to apply to users of this server who are members of the specified group. The group can be either a user group from the directory service schema or a virtual group constructed in the configuration for policy application purposes.

Enter a group name, and the system applies the specified policy to any user belonging to that group. Also enter the complete path to a previously configured policy reference.

If this value is set both here and at the enterprise level, through the **enterprise** object, the system applies both settings.

Default: There is no default setting
Example: **set user-group-policy lcsAdmin "vsp policies session-policies policy noIM"**

to-policy: *Secondary property.* References a policy to apply. If the SIP messages that start a session are directed to this server, the system applies and evaluates the referenced **to-policy**.

Default: There is no default setting
Example: **set to-policy vsp\policies\session-policies\policy toPolicy**

from-policy: *Secondary property.* References a policy to apply. If the SIP messages that start a session come from this server, the system applies and evaluates the referenced **from-policy**.

Default: There is no default setting
Example: **set from-policy vsp\policies\session-policies\policy fromPolicy**

fork-delay: *Secondary property.* Sets the period that the system waits before "ringing" another SIP device registered with a user.

Default: 0 (all destinations ring simultaneously)
Example: **set fork-delay 3**

server-pool-flush: *Secondary property.* Activates the ability to flush the server pool and relearn the entries from its configuration and the DNS server. The frequency with which the system flushes the server pool is determined by the TTL on the DNS response.

Default: disabled
Values: enabled | disabled
Example: **set**

message-filtering: *Secondary property.* Specifies a message filter to apply to traffic passing through this server. Do not change this setting from the default, **none**, unless specified to do so by Technical Support personnel.

Default: none
Values: none | st-AD *local-domain federated-domain*

Example: **set message-filtering none**

add-user-to-connect: *Secondary property*. When enabled, the ME puts the "user" (defined in the same server configuration) into the contact during register fail-over detection. For example the contact looks like this:

"Contact: sip:user@1.2.3.4:5060; transport=udp;expires=3600"

By default this parameter is disabled and the contact looks like this:

"Contact: sip:user@1.2.3.4:5060; transport=udp;expires=3600"

Default: disabled
Values: enabled | disabled
Example: **set add-user-to-connect enabled**

ping-mode: Allows you to modify how the ME decides when to mark a server up or down when it receives a response from a remote server to an OPTIONS ping.

Default: promiscuous-mode

- Values: promiscuous-mode: When an OPTIONS ping is sent to the server, if any response is received, the server is considered up. The only case where the server is marked down is a timeout from no response.
- restricted-mode: When an OPTIONS ping is sent to the server, the server must respond with a 200OK or it will be considered down.

Example: **set** ping-mode restricted-mode

DNS-Group and/or Sip-Connection Properties

domain-port: (**dns-group** only) Provides local-port functionality for a server of type dns-group. See the **server-pool-admission-control > local-port** property description for details.

Default: There is no default setting
Values: Min: 1 / Max: 65535
host: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > host** property description for details.

transport: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > transport** property description for details.

port: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > port** property description for details.

local-port: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > local-port** property description for details.

connection-role: (**sip-connection** only) See the **server-pool-admission-control > connection-role** property description for details.

admission-control: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > admission-control** property description for details.

emission-control: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > emission-control** property description for details.

max-bandwidth: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > max-bandwidth** property description for details.

max-number-of-concurrent-calls: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > max-number-of-concurrent-calls** property description for details.

max-calls-in-setup: (**dns-group** and **sip-connection** only) See the **server-pool-admission-control > max-calls-in-setup** property description for details.

default-sip-settings

Secondary object. Configures SIP communications settings for calls destined for the server. These settings override SIP settings from the default session configuration. However, these settings are overridden from outside the session configuration by SIP settings contained in any matching policy rules.

Syntax

```
config vsp enterprise servers server name default-sip-settings
```

Properties

The properties of this object are the same as those for **sip-settings**. See that object for property descriptions.

server-pool

Creates a configuration of servers that the ME uses to access enterprise information. A server pool is a logical construct used to group physical interfaces (hosts) into a shared resource for registration and INVITE requests. Add servers to the pool using the **server-pool-admission-control** subobject.

Syntax

```
config vsp enterprise servers server name server-pool
```

Properties

call-routing-on: *Secondary property.* Specifies whether the system does routing or location lookups based on the Request URI, the To URI, or an alternate setting. By default, the system performs lookups on the Request URI. Change this setting, for example, when routing information is not available in the Request URI but it is available in the To URI.

This setting applies to all servers in the pool. All calls from all servers in the pool are looked -up based on the URI set with this property.

This setting can also be configured in the **arbiter** object. If values are set in both this and the arbiter, the arbiter settings take precedence.

Default: request-uri

- **Values: request-URI**: The Request URI, which contains the hop-by-hop destination for the call
- **to-uri**: The To URI, which contains the final destination of the call
- **as-is**: The Request URI (the default) or the value set for this property in the **arbiter** object

Example: **set call-routing-on to-uri**

handle-response: Specifies the action the system should take when it receives a specific response code from this server. Enter a code, and set a handling pattern.

Default: try-next-peer

- **Values: try-next-peer**: The system forwards the message to the next server within this server pool
- **try-next-route**: The system forwards the message to the route that is the next most-specific. Use this in conjunction with the **arbiter-apply** joined-matches option (in the **arbiter** object).
- **forward**: The system returns the response to the originator of the message

Example: **set handle-response 404 forward**

dialog-failover: When enabled, the **dialog-failover** setting forces the ME to check the state of the destination SIP server before sending messages. If the destination server is down, the calls are routed to the next configured (and available) backup server.

For dialog-failover to work, the **failure-detection** property must be set to *auto*, *ping*, or *register* in the **servers** and/or **exchange** objects.

When **dialog-failover** is set to disabled, any calls in progress at the time of the failure will be retried at the original destination server until the configured timeout settings have expired.

Default: disabled
Values: enabled | disabled
Example: **set dialog-failover enabled**

server-gatekeeper-id: *Secondary property*. Specifies the way the ME reaches an H.323 Gatekeeper.

Default: dynamic
- Values: dynamic: The ME learns the Gatekeeper ID via RAS messaging.
- static: The GKId string must be configured. The ME uses this configured string to contact a remote H.323 Gatekeeper.

Example: **set server-gatekeeper-id static**

remote-web-services-fetch-timer: Configures the allowed interval to collect redirect statistics before the ME times out.

Default: 5000
Values: Min: 0 / Max: 4294967296
Example: **set remote-web-services-fetch-timer 7000**

server-pool-admission-control

Allows you to configure a server-pool CAC on any enterprise server that contains a pool.

Syntax

```
config vsp enterprise servers server name server-pool
server-pool-admission-control
```

Properties

max-bandwidth: Enter the maximum amount of bandwidth, in kbits per second, the ME allocates to the AOR. When the system reaches the maximum bandwidth limit for a server, it rejects calls until bandwidth use drops below the maximum.

Default: unlimited
Values: Min: 0 / Max: unlimited
Example: **set max-bandwidth 1000**

max-number-of-concurrent-calls: Specify the maximum number of active calls allowed for this AOR at one time. When this value is reached, the connection does not accept calls until the value drops below the threshold.

Default: 1000
Values: Min: 0 / Max: 1000000
Example: **set max-number-of-concurrent-calls 5000**

max-calls-in-setup: Sets the maximum number of simultaneous call legs in setup stage that are allowed for this AOR. A call leg in setup is much more compute-intensive than established call legs, so this value is more restrictive than the concurrent call leg value. A value of 0 causes the system to decline all calls and registrations.

Default: 30
Values: Min: 0 / Max: 10000
Example: **set max-calls-in-setup 5000**

call-rate-limiting: Limits the number of calls sent to an AOR within a certain interval in seconds. Once this interval is reached, the system rejects any calls to or from this AOR until the rate decreases, returning a response code and message. This feature sets the acceptable arrival rate for incoming calls when used with admission-control and the acceptable set-up rate when used with emission-control. When this feature is enabled, set the number of calls and the measurement interval. You can also enter a result code from 400 to 699 and a text string to accompany call rejection if no available server is found.

Default: disabled
Values: enabled | disabled
Example: **set call-rate-limiting enabled**

admission-control: Specifies whether the system considers AOR limitations when forwarding a call from the AOR. The system tracks the number of concurrent (both incoming and outgoing) active calls for this AOR.

Default: disabled
Values: enabled | disabled
Example: **set admission-control enabled**

emission-control: Specifies whether the system considers AOR limitations when forwarding a call to this AOR. The system tracks the number of concurrent (both incoming and outgoing) active calls for this AOR.

Default: disabled
Values: enabled | disabled
Example: **set emission-control enabled**

call-admission-control-error-code: Enter the call admission error code.

Default: 503
Values: Min: 400 / Max: 999
Example: **set call-admission-control-error-code 550**

call-admission-control-error-string: Enter the text string the user sees when a call admission control error occurs.

Default: There is no default setting
Example: **set call-admission-control-error-string cac error**

call-emission-control-error-code: Enter the call emission error code.

Default: 503
Values: Min: 400 / Max: 999
Example: **set call-emission-control-error-code 550**

call-emission-control-error-string: Enter the text string the user sees when a call emission control error occurs.

Default: There is no default setting
Example: **set call-emission-control-error-string cac error**

server

Adds server connections to the server pool. You must identify the server by host name or IP address. Optionally, you can set a preference, protocol, port, load-balancing, and other criteria, as well as limit connections.

Each server in a pool has an associated order of preference. The server with the lowest order is preferred, and becomes the primary. The other servers are backups. If the primary is up, as indicated by the ME monitoring process, then registrations and INVITE requests are sent there. If it is down (but the backup is up), then the ME sends requests to the backup. When the primary later becomes available again, the ME resumes sending registrations and INVITE to it.

If both the primary and backup are down, then the peer changes to local mode. In local mode, the ME does not proxy registrations to the failing peer. Instead, if the To header is addressed to a SIP phone directly connected to the ME (it has a location binding with the ME), INVITEs are switched locally. If the To header is addressed to a location out of reach of the ME, the INVITE is forwarded to the configured PSTN gateway: (The gateway is configured with the **sip-gateway server** object.)

If the primary or backup later becomes available, the peer reverts from local back to provider mode, and again forwards registrations and INVITE to the primary or backup server.

The server object allows you to configure normalization plans for outgoing and incoming calls. See Configuring Normalization objects, for a full description of the server object **outbound-normalization** and **inbound-normalization** subobjects.

Syntax

```
config vsp enterprise servers server name server-pool server name
```

Properties

host: Specifies the host name or IP address of an Internet endpoint. Enter a host name or IP address.

Default: There is no default setting
Example: **set 192.168.10.10**

endpoint: Associates a text string with a server-pool server. The string can later be used to group and categorize servers.

Default: There is no default setting
Example: **set endpoint server1**

transport: Specifies the protocol used by the connection.

Default: UDP
Values: any | UDP | TCP | TLS
Example: **set transport any**

port: Specifies the port used by the connection for SIP traffic.

Default: 5060
Example: **set port 3333**

local-port: Sets a port number for the system to use in the Contact header, Via header, and source port when it sends a Register request (and subsequent SIP messages) to an upstream server. The server caches the binding and includes the local-port when contacting the system. Additionally, the server can be configured to send SIP messages to this particular local-port without prior registration from the system.

With local-port configured, the system can tell:

- To which connection in the server pool to forward a call
- Which connection in the server pool it received the call from, when the connection sends SIP message to this local port

Using this property allows you to group traffic based on the local port number. For example, if there are multiple domains from a single physical server, the port will indicate which domain should receive the call. Or, if there is a distinct pair of physical servers to protect traffic for a domain, the Eclipse can fail over to the right backup server (in case of primary failure) for this particular domain.

Default: There is no default setting
Example: **set local-port 5050**

connection-role: Specifies the way the server behaves in establishing a TCP/TLS connection. If set to **initiator**, the server can open up a connection without any SIP traffic. If set to **responder**, the server will not open up a TCP/TLS connection until receiving SIP traffic.

Default: initiator for server-pool and responder for sip-connection
Values: initiator | responder
Example: **set connection-role responder**

connection-retry-interval: Specifies the number of seconds the system waits between attempts to open a TCP or TLS connection. This value is only meaningful if the **connection-role** property is set to **initiator**. If set to **responder**, the value is ignored.

Default: 5
Example: **set connection-retry-interval 10**

preference: Specifies the preference for the connection. The lower the value the higher the preference. If you use the value of **none**, the system uses the preference set in a different part of the configuration, such as the ordered set of arbitration rules in the **dial-plan** object.

Default: none
Values: none | *integer* [0-65535]
Example: **set preference 1**

admission-control: Specifies whether the system considers downstream server capacity when forwarding a call from the server. The ME tracks the number of concurrent calls for each server. If this property is **enabled**, the system does not forward calls from the server if the server limit has been reached and instead sends a "503 Service Unavailable" message. If **disabled**, the system does forward calls from the server. (Set the call limit with the **max-number-of-concurrent-calls** property.) See Admission Control for an AOR for specific information on CAC settings applicability for an AOR.

Default: disabled
Values: enabled | disabled
Example: **set admission-control enabled**

emission-control: Specifies whether the system considers upstream server capacity when forwarding a call to the server. The ME tracks the number of concurrent (both incoming and outgoing) active calls for the server. If this property is **enabled**, the

system does not forward calls to the server if the limit, set with the **max-number-of-concurrent-calls** property, has been reached. Instead, the system sends one of the following messages and drops the call:

If there is one outbound server/UAC/UAS, the system sends a “486 Busy” message, indicating that the route was resolved but that the AOR was unavailable.

If there are multiple outbound server/UAC/UASs and all have reached the maximum concurrent calls threshold, the system sends a “486 Busy” message.

If there are multiple outbound server/UAC/UASs and at least one has not reached the maximum concurrent calls threshold, the return code is determined by the final server that the system attempted to reach. This could be, for example, “486 busy” or a “504 server timeout” if the last server was unresponsive and the transaction timed out.

If **disabled**, the system continues to forward calls to the server.

Default: disabled
Values: enabled | disabled
Example: **set emission-control enabled**

max-bandwidth: Specifies the amount of bandwidth the system allocates to a connection. For a SIP server, the default value is **unlimited** or the server uplink bandwidth. For example, if the uplink is GigE, then bandwidth is 1 million kbps. When the system reaches the maximum bandwidth limit for a server, it rejects calls until bandwidth use drops below the maximum.

Note that the bandwidth usage value is based not on the actual traffic on the wire, but on a calculation done by the system. The calculation uses the value associated with the first known CODEC identified in the SDP for a usage rate. If there is not a known CODEC, or the value has not yet been determined from the SDP, the system uses the **default-session-bandwidth** value from the session configuration **media** object.

Set a specific bandwidth if you are using, for example, a TDM trunk or PSTN gateway with limited bandwidth. For a PSTN trunk, the usual capacity is DS0 (64 kbps bandwidth). If a gateway has 8 trunks, then the gateway has 512 kbps bandwidth.

Default: unlimited
Values: unlimited | *kbps*
Example: **set max-bandwidth 512**

max-number-of-concurrent-calls: Specifies the number of calls allowed on the connection at one time. When this value is reached, the connection will not accept calls until the value drops below the threshold.

Default: 1000
Values: Min: 0 / Max: 1000000
Example: **set max-number-of-concurrent-calls 1500**

max-calls-in-setup: Sets the maximum number of simultaneous inbound and outbound call legs in setup stage that are allowed for the connection. A call leg in setup is much more compute-intensive than established call legs, so this value is more restrictive than the concurrent call leg value. A value of 0 causes the system to decline all calls and registrations.

Default: 30
Values: Min: 0 / Max: 10000
Example: **set max-calls-in-setup 5**

call-rate-limiting: *Secondary property.* Limits the number of calls sent to a server within a certain interval. Once this interval is reached, the system hunts for the next available server. If there are no available servers, the system returns a response code and message. This feature sets the acceptable arrival rate for incoming calls when used

with **admission-control** and the acceptable set-up rate when used with **emission-control**.

If **enabled**, set the number of calls allowed and the measurement interval (in seconds). You can also enter a result code from 400 to 699 and a text string to accompany call rejection if no available server is found.

Default: disabled; if set to enabled, the default calls-per-interval is 60, the default interval is 1 second, and the default result is 486, Busy Here.
Values: enabled <*calls-per-interval*><*interval*><*result-code*><*result-string*> | disabled
Example: **set call-rate-limiting enabled 50 1 480 "Temporarily unavailable"**

max-number-of-registrations: *Secondary property.* Specifies the maximum number of registrations that can be active with a server at any one time. This property is used in conjunction with the **server-registration-balance** property of the VSP **settings** object to implement registration load balancing.

Default: 1000
Values: Min: 0 (causes the
WebRTC Session Controller Media Engine
to decline all calls and registrations) / Max: 1000000
Example: **set max-number-of-registrations 1500**

max-registrations-in-progress: *Secondary property.* Specifies the number of registrations or authentication requests per second that the system forwards to the server. Use this property as a flow control mechanism to control the system, which can process registrations much more quickly than the server. To set this, you must know the capability of your server. You also must enable the **server-registration-balance** property of the VSP **settings** object.

When a register is delegated/forwarded/tunneled to the server, the system increments a cluster-wide server counter. When the counter reaches this threshold, the system handles subsequent registrations. It responds with "200 OK," but sets a brief expiration, causing the phone to reregister almost immediately.

Default: 300
Values: Min: 0 (causes the
WebRTC Session Controller Media Engine
to decline all calls and registrations) / Max: 100000
Example: **set max-registrations-in-progress 600**

external-outbound-normalization: *Secondary property.* Specifies whether the system should perform external normalization on outbound call legs. Enter the host name of your calling plan server.

Default: no
Values: no | yes *server*
Example: **set external-outbound-normalization yes ITALKBB-EGR5**

external-inbound-normalization: *Secondary property.* Specifies whether the system should perform external normalization on inbound call legs. Enter the host name of your calling plan server.

Default: no
Values: no | yes *server*
Example: **set external-inbound-normalization yes PT1-INGRESS**

handle-unregister-locally: *Secondary property.* Specifies under what circumstances the ME can process an UNREGISTER request. When **enabled**, the system processes the request even if the REGISTER was not sent by the same registration endpoint.

Default: disabled

Values: enabled | disabled
Example: **set handle-unregister-locally enabled**

error-response-code

Opens the error-response-codes configuration object where you can specify custom admission-control, emission-control, and server down error codes and text strings that will be returned in the SIP response and captured in the event log.

Syntax

```
config vsp enterprise servers type name server-pool server name
error-response-codes
```

Properties

call-admission-control-error-code: Modifies the error code to be returned in the SIP response when a call-admission-control error occurs.

Default: 503
Values: Min: 400 / Max: 999
Example: **set call-admission-control-error-code 550**

call-admission-control-error-string: Creates a user-defined text string to be returned in the SIP response when a call-admission-control error occurs.

Default: There is no default setting
Example: **set call-admission-control-error-string cac error**

call-emission-control-error-code: Modifies the error code to be returned in the SIP response when a call-emission-control error occurs.

Default: 503
Values: Min: 400 / Max: 999
Example: **set call-emission-control-error-code 550**

call-emission-control-error-string: Creates a user-defined text string to be returned in the SIP response when a call-emission-control error occurs.

Default: There is no default setting
Example: **set call-emission-control-error-string cac error**

server-down-error-code: Modifies the error code to be returned in the SIP response when a destination SIP server is down.

Default: 503 (Bad Gateway)
Example: **set server-down-error-code 403**

server-down-error-string: Creates a user-defined text string to be returned in the SIP response when a destination SIP server is down.

Default: There is no default setting
Example: **set server-down-error-string error**

registration-proxy

Sets the characteristics of the relationship between two peers that are both proxies. When the ME acts as a proxy, it is able to supply the credentials needed for authentication challenges. It maintains a location service database to store SIP caller location (address-of-record) information. This database can be updated via the ME registration service, static address-of-records (AORs), and/or configured the ME

policies. To ensure that peer systems have and use the same database, set the properties of this object.

You can also set the registration-proxy property in the **vsp** object. In that instance, you are optimizing system performance by specifying whether the ME should walk the database.

Syntax

```
config vsp enterprise servers server name registration-proxy
```

Properties

admin: Enables or disables this proxy registration configuration. If **enabled**, the server applies these characteristics to sessions with its configured peers. If **disabled**, these characteristics are inactive.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

request-download: Automates the download of the registration database from a peer. (Peers are identified in the **server** configuration.) If set to **yes**, the peer system downloads the database to this server with the frequency set in the request interval of this property. In addition, it copies the interval to the expiration time in the REGISTER requests forwarded to peers. If set to **no**, downloads do not occur automatically. The system only learns of new or changed AORs through REGISTER requests.

Default: no; if set to yes, the default interval is 1440 minutes
Values: no | yes *minutes*
Example: **set request-download yes 1080**

network

Sets the properties specific to the server socket. To set general system network parameters, including other socket properties, use the services **network** object. This object is only applicable to sip-connection.

Syntax

```
config vsp enterprise servers sip-connection name network
```

Properties

tcp-keepalive-time: Specifies the time, in seconds, that an established TCP connection can remain idle before the system sends a keepalive to the client. The idle time expiration initiates the keepalive process.

Default: 600
Values: Min: 10 / Max: 14400
Example: **set tcp-keepalive-time 900**

tcp-keepalive-probes: Specifies the number of unanswered TCP keepalive probes that are allowed before the system determines a session is idle and disconnects it.

Default: 5
Values: Min: 2 / Max: 16
Example: **set tcp-keepalive-probes 10**

tcp-keepalive-interval: Specifies the time, in seconds, that the system waits for a response from a keepalive probe before ending the next one. The ME continues to send probes until it has sent the number specified in the **tcp-keepalive-probes** property.

Default: 6
Values: Min: 1 / Max: 60
Example: **set tcp-keepalive-interval 10**

tcp-ephemeral-port-start: *Secondary property*. Configures the local TCP ephemeral port range start. Well-known ports or ports configured for use with SIP or H.323 should not be allocated in the local port pool range. Well-known ports or ports configured for use with SIP and H.323 should not be allocated in the local port pool range.

Default: 1024
Values: Min: 1024 / Max: 65535
Example: **set tcp-ephemeral-port-start 2025**

tcp-ephemeral-port-end: Secondary property. Configures the local TCP ephemeral port range end. Well-known ports or ports configured for use with SIP or H.323 should not be allocated in the local port pool range. Well-known ports or ports configured for use with SIP and H.323 should not be allocated in the local port pool range.

Default: 4999
Values: Min: 1024 / Max: 65535
Example: **set tcp-ephemeral-port-end 3025**

ccs

Configures the ME to recognize the Avaya Converged Communication Server (CCS), a SIP proxy server that connects to the company's proprietary IP telephony solution, the Avaya Call Manager (ACM). The CCS will add SIP-based voice, presence, and Instant Messaging (IM) services.

This feature is only applicable to Avaya.

Syntax

```
config vsp enterprise servers server name ccs
```

Properties

server: Sets the IP address of the computer hosting the CCS service.

Default: There is no default setting
Example: **set server 192.168.10.10**

port: Sets the TCP port of the SIP proxy service on the computer hosting the CCS service.

Default: There is no default setting
Example: **set port 2020**

mode: Specifies the type of server this CCS server is functioning as.

Default: home-edge

- Values: edge: An edge device, allowing domain routing
- home: A home device with only one domain configured
- home-edge: A combination device

Example: **set mode edge**

h323-ras-settings

Sets the configuration for scenarios when the ME is communicating with an external H.323 GK. (This property is only applicable if the **server-type** property is set to **h323-gatekeeper**.) When the ME registers on behalf of a client, these settings allow the systems to exchange RAS messages.

Syntax

```
vsp enterprise servers h323-server h323-ras-settings
```

properties

registration-ttl: Sets the time to live (TTL), in seconds, for registration to the external GK.

Default: 3600
Values: Min: 0 / Max: 4294967295
Example: **set registration-ttl 5000**

registration-retries: Sets the number of RRQs or GRQs the ME resends to an external GK before abandoning the request. A value of **0** allows unlimited retries.

If multiple external GKs exist, this property is not used to control RRQ and GRQ retransmission.

Default: 5
Values: Min: 0 / Max: 4294967295
Example: **set registration-retries 100**

admission-retries: Sets the number of times the ME resends an ARQ to an external GK before refusing to admit the call.

Default: 3
Values: Min: 0 / Max: 4294967295
Example: **set admission-retries 5**

endpoint-alias: Assigns a string to the GRQ and RRQ to allow the external GK to identify the ME.

Default: EPAlias
Example: **set endpoint-alias alias1**

supported-prefix: Sets the value for GKs that need digits prepended to a number.

Default: +1
Example: **set supported-prefix +4**

prefix-type: Adds a voice capability supported prefix to the supported protocols identified in the ME-transmitted GRQ and RRQ.

Default: h323ID

- Values: none
- dialedDigits
- h323ID
- urlID
- emailID

Example: **set prefix-type none**

reregister-on-urq: Specifies whether the ME tries to reregister a client after having received an UNREGISTER from the GK. When enabled, the ME tries to reregister the client up to the number of times specified in the **registration-retries** property.

Default: disabled
Values: enabled | disabled
Example: **set reregister-on-urq enabled**

calls-gk-routed: When true, T_H225CallModel_gatekeeperRouted is the call model in an ME-transmitted ARQ. When false, T_H225CallMode_direct is the call mode the ME uses.

Default: false
Values: true | false
Example: **set calls-gk-routed true**

use-alternate-gks: When true, supportsAltGKPresent is present in an ME-transmitted RRQ.

Default: false
Values: true | false
Example: **set use-alternate-gks true**

retries-before-alt-gk: Specifies how many times the ME resents an RRQ or GRQ to an external GK when multiple external GKs exist.

Default: 10
Values: Min: 0 / Max: 4294967296
Example: **set retries-before-alt-gk 25**

use-lightweight-rrq: When true, the ME reregisters with an external GK using a lightweight RRQ.

Default: false
Values: true | false
Example: **set use-lightweight-rrq true**

gk-round-robin: Specifies how the ME handles external GKs that have previously rejected the ME.

Default: PollRejected
- Values: IgnoreRejected: The ME ignores rejected GKs.
- PollRejected: The
- ME
- contacts external GKs that have previously sent GRJ or RRJ to the ME.

Example: **set gk-round-robin IgnoreRejected**

wait-for-gk-response: Determines the length of time, in seconds, the ME waits for a response to a GRQ or RRQ.

Default: 15
Values: Min: 1 / Max: 4294967296
Example: **set wait-for-gk-response 44**

wait-for-admit-response: Determines the length in time, in seconds, the ME waits for a response to an ARQ.

Default: 5
Values: Min: 1 / Max: 4294967296
Example: **set wait-for-admit-response 10**

wait-for-location-response: Determines the length of time, in seconds, that the ME has to process a received LRQ before timing out.

Default: There is no default setting
Example: **set wait-for-location-response 15**

delay-all-gks-rejected: Specifies the length of time, in seconds, the ME waits before attempting external GK discovery and registration when all external GKs have rejected all GRQs and RRQs from the ME.

Default: 15
Values: Min: 1 / Max: 4294967296
Example: **set delay-all-gks-rejected 25**

use-lrqsrc-endpoint-info: Determines if the H.323 process extracts sourceEndpointInfo from a received LRQ to pass to SIP for use in destination route lookup.

Default: false
Values: true | false
Example: **set use-lrqsrc-endpoint-info true**

create-gk-sessions: When true, the ME creates sessions for all RAS discovery and registration traffic sent by the ME to external GKs.

Default: false
Values: true | false
Example: **set create-gk-sessions true**

tos: Enables or disables packet marking. Marking (tagging) a packet provides a QoS indicator, which routers along the path may act on. The ME writes this value to the ToS field of the IP header. Enter this value in hexadecimal or decimal format.

Default: There is no default setting
Values: disabled | tos <0-255>
Example: **set tos 128**

37

Configuring Services Routing Objects

The **services-routing** object applies cluster-wide routing configurations, including metrics to each of the ME service routing tables (media, SIP, H.323, and STUN) and gateway health checks. Metrics define the cost associated with each route, controlling how services are load-balanced across the cluster.

There are four service routing table types:

- Media: Used for allocating media resources across the cluster
- SIP: Used for routing SIP packets on each ME
- H.323: Used for routing H.323 packets on each ME
- STUN: Used for allocating STUN requests

When you configure an IP interface, the ME installs both a network route and a host route into the generic routing table, along with a metric (preference) for the route. If there are services configured under the interface (e.g., media, SIP, H.323, or STUN), the route is also installed in the specific service routing table. In addition, each service route table can have subtables that are created when a routing-tag is associated with an IP interface. See Tag-Based Route Selection for a description of these tag-based tables.

Understanding the Application of Route Metrics

Each service route entry has five metric fields that serve as tie breakers to determine route preference. You can assign different metric types to each of the five metric fields, controlling the preference of one route server over another. Given that there may be multiple paths available to reach a destination, using service route table metrics allows the ME to consider routes differently depending on their purpose. The lower the metric value, the higher the preference.

For each service route table (and therefore, application), you can set different criteria for route selection. For example, to influence media anchoring route selection, you can assign metrics to the media table. By default, if two or more routes exist with equivalent metric values, these routes are considered equal-cost routes. The ME uses a round-robin algorithm through all equal-cost routes when distributing a service across a cluster. You can assign metric types to a service route table using the five metric fields to prefer one route over another.

To select the most preferred route(s), the ME applies these table-specific metrics. If two or more routes exist to reach a destination, the ME then compares metric1, and so on, up to metric5, or until a route is preferred based on a metric. If all five metrics are equivalent, the routes are considered equal cost.

By default, the ME uses the **user-metric** value. This value is used as the first tiebreaker (metric1 field), and the remaining fields default to **none**.

The frequency with which the ME updates the value for calculated metrics (for all service metric tables) is set with the **vsp > services-routing > metric-timer** property. You select the metric type for a service under the **media-service-routing**, **sip-service-routing**, **h323-service-routing**, and **stun-service-routing** objects. Note that any configuration changes to the assignment of metric types causes an immediate recalculation of that service's route table. Use the **show services-routing-metrics** command to display the metric assignments.

The following lists and describes the valid metric types.

- none: The metric field is not used during route selection.
- user-metric: A user-defined static metric that can be associated with a specific IP interface or static route. This static metric can be used to determine the preference of a route. The user-metric value can be thought of as the cost associated with a route, the lower the cost the more preferred the route.

For an IP interface this metric value is configured under **box > interface > ip > metric**. For a static route this metric value is configured under **box > interface > ip > routing > route > metric.**

- intf-thruput: A route that is associated with an IP interface. This metric type uses the dynamic throughput, in kilobits per second, of the physical interface that a particular route is associated with to determine route preference. The lower the intf-thruput value, the more preferred the route.
- box-cpu-load: A route that is associated with an ME. This metric type uses the ME-wide CPU load to determine route preference. The lower an ME's CPU load is, the more preferred the route.
- box-memory: A route associated with an ME. This metric type indicates the percentage of memory allocated from the SIP Process' maximum heap. The lower the box-memory, the more preferred the route.
- box-media-load: This metric type is currently unused.

About Services Routing Load Balancing

On the ME, services routing distributes the load of a service using either a round-robin (RR) or a weighted-round-robin (WRR) load balancing algorithm. The RR algorithm loops through a set of equal cost routes, distributing the load equally across the set of routes. When dynamic metrics such as an ME's CPU utilization is used to balance the load, the RR algorithm can lead to an uneven distribution of the load. The WRR load balancing algorithm is a better choice when using dynamic metrics such as an ME's CPU utilization.

For each type of service routing, there are five service routing metric fields named metric1 through metric5. Each of the metrics can be configured to use a specific metric type to determine the services route preference.

You configure each metric to determine which load balancing algorithm, either **round-robin** or **weighted-round-robin**, is used using the **load-share-scheme** property. The default algorithm is **round-robin**.

About RR Behavior on the ME

When all metric fields are configured for RR, the ME determines which routes are part of an equal cost route set. If all metric fields configured for RR are of equal value for two or more routes, those routes are considered equal cost.

The following table shows an example. Route 1 and Route 3 are considered equal cost because metric1 and metric2 are configured as RR and both metrics have the same values. Route 2 would be a secondary route because a metric 2 value of 20 is less-preferred than Route 1 and Route 3.

Table 37–1 Determining RR Routes

Route Name	Metric1: user-metric RR	Metric2: box-cpu-load RR
Route 1	1	10
Route 2	1	20
Route 3	1	10

About WRR Behavior on the ME

If one or more metrics are configured as WWR, then those metrics are used to calculate the load-share for each route in an equal cost set. Each metric configured for WRR has a dynamic weight that is calculated based on how close a metric is to its upper bound. A route's weight is then used to calculate its load share across an equal cost route set. Services routing then balances the load across the equal cost route set based on the load share values calculated for each route.

The following table shows an example. Routes 1-3 are considered equal cost because their round robin metrics are all of equal value. Metric2 is configured to use box-cpu-load (% of CPU utilized) as a weighted-round-robin. Based on these WRR values, the routes are assigned a load-share. Since Route 1 is the least loaded, it is assigned the highest load-share (75) while route 3 receives the lowest load-share (25). Once the load-share of the equal cost route is calculated, the load is distributed appropriately across the route set. In this example, Route 1 should receive twice as much load as Route 3, and 25% more load than route 2. For example, if 150 requests are made, 75 are distributed using Route 1, 50 by Route 2, and 25 by Route 3. These values are based on the calculated loadshare value.

Table 37–2 Determining WRR Routes

Route Name	Metric1: user-metric RR	Metric2: box-cpu-load WRR
Route 1	1	25 (load-share 75)
Route 2	1	50 (load-share 50)
Route 3	1	75 (load-share 25)

The values of a route's metric fields are updated periodically based on the **vsp > services-routing > metric-time** property. Each time this metric-time expires, the load-shares are recalculated based on the latest metric values. Also, anytime an equal cost route is added or removed from an equal cost route set, new load-share values are calculated.

services-routing

Configures VSP routing to control routing in a cluster of one or more ME devices. For example, you can set the interval with which metrics used for route selection are updated. When this interval expires, the ME recalculates the selected metrics for each table and distributes the value throughout the cluster. At that point, the ME recalculates the preference of each route affected by a metric change. In addition, the subobjects provide access to the metric selection for each of the service routing tables.

Gateway Health Checks

This object configures gateway health checks for static routes, verifying reachability of a gateway. (When configuring a static route, you must supply a gateway address as the next-hop for forwarding packets to reach the destination network.) The ME sends ARP requests to each configured gateway, from each configured interface using that gateway. If a gateway is configured on multiple interfaces, the ME verifies reachability from each interface.

The ME uses a configurable timer and a maximum failure count to determine when a gateway is considered unreachable. The timer controls how often a health check is performed. The maximum failure count controls how many consecutive health checks must fail before the gateway is considered unreachable. For example, if the gateway health timer is set to ten seconds, and the gateway maximum failure count is set to three, a gateway would become unreachable after three consecutive health check failures or thirty seconds. When a gateway fails its health check on a particular IP interface, the ME considers any static routes configured with that gateway on that IP interface unreachable.

Syntax

```
config vsp services-routing
```

Properties

metric-timer: Sets the interval (in seconds) with which the system updates service-routing metrics. This interval, which applies to all route service tables, controls how quickly the system can propagate a change in a metric value throughout the cluster service route tables.

A value of **0** disables the gateway health check feature.

Default: 60
Values: Min: 0 / Max: 86400
Example: **set metric-timer 90**

gateway-health-timer: Sets the frequency with which the system sends ARP requests to a gateway. This interval applies to all gateways configured on the box. See Gateway Health Checks for more information.

A value of **0** disables the gateway health check feature.

Default: 0
Values: Min: 0 / Max: 86400
Example: **set gateway-health-timer 90**

gateway-max-failures: Specifies the number of consecutive health checks that must fail before the system considers a gateway unreachable. See Gateway Health Checks for more information.

Default: 3
Values: Min: 1 / Max: 1000

Example: **set gateway-max-failures 5**

cpu-sample-interval: Sets the sampling interval (in seconds) in which to calculate the CPU usage of the system. For example, if the **cpu-sample-interval** is 10 seconds, then 10, one second samples will be used to calculate the CPU usage. The result of this calculation is a CPU usage percentage between 0 and 100. The calculated CPU usage value is used as the **box-cpu-load** metric.

Default: 60
Values: Min: 1 / Max: 3600
Example: **set cpu-sample-interval 100**

cpu-sample-divisor: Normalizes the **box-cpu-load** metric. To do so, the raw CPU usage percentage is divided by this value and the result is used as the **box-cpu-load** metric. This provides better control of how services are load balanced, and results in a better load balancing distribution. For example, if three boxes have a CPU usage of 21, 22, and 23 percent respectively, without normalization of **box-cpu-load** metric, the box with CPU usage of 21 would be preferred over the other two boxes. By normalizing the CPU usage percentage using the default divisor of 10, the metric values for all three boxes becomes two. The load will then be balanced across all three boxes.

Default: 10
Values: Min: 1 / Max: 100
Example: **set cpu-sample-divisor 20**

media-service-routing

Sets the basis for route evaluation when selecting routes from the media service routing table. The ME uses these metrics, which set route precedence, when anchoring media.

In an ME cluster, media can be load balanced across two or more media MEs. The decision as to which ME handles the media is determined based on the signaling address from the initial INVITE. A services routing lookup is performed in a cluster-wide basis using this signaling address to determine which the ME should handle the media. Once a media ME is selected, all the media resources for the session are allocated locally from that media ME.

Delayed-Offer Topology

In a Delayed-offer topology, local media ports must be configured on the signaling ME. If you want to have the signaling ME handle only signaling and no media, you can load balance the media across the media ME. This is accomplished by configuring each of the signaling interfaces with a higher metric than any of the media interfaces on the two media MEs. Any static routes configured under these signaling interfaces should also be configured with the same higher metric. This is configured under the **ip > routing > route > metric <cost>** property. By configuring the media interfaces on the media ME with a lower metric, they are always preferred over the signaling MEs., allowing all media to load balanced across the media SBCs only.

Syntax

```
config vsp services-routing media-service-routing
```

Properties

metric1 <metric-type><load-share-scheme>: Sets the first type of metric considered by the system when it selects a route for media traffic.

Default: user-metric round-robin
Example: **set metric1 intf-throughput weighted-round-robin**

metric2 <metric-type><load-share-scheme>: Sets the second type of metric considered by the system when it selects a route for media traffic.

Default: none round-robin
Example: **set metric2 user-metric weighted-round-robin**

metric3 <metric-type><load-share-scheme>: Sets the third type of metric considered by the system when it selects a route for media traffic.

Default: none round-robin
Example: **set metric3 box-cpu-load weighted-round-robin**

metric4 <metric-type><load-share-scheme>: Sets the fourth type of metric considered by the system when it selects a route for media traffic.

Default: none round-robin
Example: **set metric2 box-memory weighted-round-robin**

metric5 <metric-type><load-share-scheme>: Sets the fifth type of metric considered by the system when it selects a route for media traffic.

Default: none round-robin
Example: **set metric5 user-metric weighted-round-robin**

sip-service-routing

Sets the basis for route evaluation when selecting routes from the SIP service routing table. The ME uses these metrics, which set route precedence, for SIP signaling. See Understanding the Application of Route Metrics for a description of using metrics for route selection.

Syntax

```
config vsp services-routing sip-service-routing
```

Properties

metric1 <metric-type><load-share-scheme>: Sets the first type of metric considered by the system when it selects a route for SIP signaling traffic.

Default: user-metric round-robin
Example: **set metric1 intf-throughput weighted-round-robin**

metric2 <metric-type><load-share-scheme>: Sets the second type of metric considered by the system when it selects a route for SIP signaling traffic.

Default: none round-robin
Example: **set metric2 user-metric weighted-round-robin**

metric3 <metric-type>< load-share-scheme>: Sets the third type of metric considered by the system when it selects a route for SIP signaling traffic.

Default: none round-robin
Example: **set metric3 box-cpu-load weighted-round-robin**

metric4 <metric-type><load-share-scheme>: Sets the fourth type of metric considered by the system when it selects a route for SIP signaling traffic.

Default: none round-robin
Example: **set metric2 box-memory weighted-round-robin**

metric5 <metric-type><load-share-scheme>: Sets the fifth type of metric considered by the system when it selects a route for SIP signaling traffic.

Default: none round-robin
Example: **set metric5 user-metric weighted-round-robin**

h323-service-routing

Sets the basis for route evaluation when selecting routes from the H.323 service routing table. The ME uses these metrics, which set route precedence, for H.323 signaling. See Understanding the Application of Route Metrics for a description of using metrics for route selection.

Syntax

```
config vsp services-routing sip-service-routing
```

Properties

metric1 <metric-type><load-share-scheme>: Sets the first type of metric considered by the system when it selects a route for H.323 signaling traffic.

Default: user-metric round-robin
Example: **set metric1 intf-throughput weighted-round-robin**

metric2 <metric-type><load-share-scheme>: Sets the second type of metric considered by the system when it selects a route for H.323 signaling traffic.

Default: none round-robin
Example: **set metric2 user-metric weighted-round-robin**

metric3 <metric-type><load-share-scheme>: Sets the third type of metric considered by the system when it selects a route for H.323 signaling traffic.

Default: none round-robin
Example: **set metric3 box-cpu-load weighted-round-robin**

metric4 <metric-type><load-share-scheme>: Sets the fourth type of metric considered by the system when it selects a route for H.323 signaling traffic.

Default: none round-robin
Example: **set metric2 box-memory weighted-round-robin**

metric5 <metric-type><load-share-scheme>: Sets the fifth type of metric considered by the system when it selects a route for H.323 signaling traffic.

Default: none round-robin
Example: **set metric5 user-metric weighted-round-robin**

stun-service-routing

Sets the basis for route evaluation when selecting routes from the STUN service routing table. The ME uses these metrics, which set route precedence, when sending requests to a STUN server. See Understanding the Application of Route Metrics for a description of using metrics for route selection.

Syntax

```
config vsp services-routing stun-service-routing
```

Properties

metric1 <metric-type><load-share-scheme>: Sets the first type of metric considered by the system when it selects a route for STUN server traffic.

Default: user-metric round-robin
Example: **set metric1 intf-throughput weighted-round-robin**

metric2 <metric-type><load-share-scheme>: Sets the second type of metric considered by the system when it selects a route for STUN server traffic.

Default: none round-robin
Example: **set metric2 user-metric weighted-round-robin**

metric3 <metric-type><load-share-scheme>: Sets the third type of metric considered by the system when it selects a route for STUN server traffic.

Default: none round-robin
Example: **set metric3 box-cpu-load weighted-round-robin**

metric4 <metric-type><load-share-scheme>: Sets the fourth type of metric considered by the system when it selects a route for STUN server traffic.

Default: none round-robin
Example: **set metric2 box-memory weighted-round-robin**

metric5 <metric-type><load-share-scheme>: Sets the fifth type of metric considered by the system when it selects a route for STUN server traffic.

Default: none round-robin
Example: **set metric5 user-metric weighted-round-robin**

38

Configuring Services Objects

The services objects allow you to configure ME event logging and virus scanning services.

Using Filters With Event Log Messages

The ME uses the filters you define in the event-log subobjects to determine the type and severity of event messages to send to the destination target. When creating a filter you specify the log class, which selects which type of event messages to send. Use the question mark at the command line to display the complete list of log classes. After selecting a class, select a severity level.

Event log messages can be written to:

- A remote syslog server
- A file
- The local database
- An external database
- The CLI window
- An SMTP server
- A Tivoli server

When configuring the message destination, you can configure one or more filters to determine which message types are written to that destination. A filter sorts messages based on the event type (log class) and the severity level.

The log class indicates the subsystem that generated the message. The severity indicates the lowest level message to display. You get messages of that class and below, with Emergency being the lowest and Debug the highest. If you set severity to **error**, you will receive Emergency, Alert, Critical and Error events. The following severity levels are recognized by the ME:

- emerg (Emergency)
- alert (Alert)
- crit (Critical)
- error (Error)
- warning (Warning)
- notice (Notice)
- info (Information)

- debug (Debug)

For a complete description of the event message types and severity levels, see *Net-Net 2600 – Using the NN2600 Management Tools*.

services

Opens the **services** object from where you configure event log settings and enable virus scanning. In addition, you define locations and schedule tasks for the ME.

Syntax

```
config services
```

Properties

None

event-log

Enables and disables global event-log administration. This control manages syslog, file, local and external database, and CLI event log storage or display.

Syntax

```
config services event-log
```

Properties

admin: Sets the global event-logging administrative state on this ME device. If **disabled**, all syslog, file system, and local-database configurations are ignored.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

snmp-trap

The ME has the ability to translate event logs into SNMP traps. When a log event is generated, the log system checks if the class and severity levels fall under any filters specified in the **event-log** config. If it is determined that this log event should be translated into an SNMP trap, the log system fills in the SNMP trap fields. Any interfaces that have SNMP targets configured transmit the trap.

The SNMP trap contains the following fields:

- Box ID
- Severity Level
- Process
- Log Class
- Log Message

The filter that the ME uses for the event-to-SNMP feature is a list of regular expression filters which you configure as a regular expression that runs on the generated log string. The ME first checks the “allowed-trap” list, then the “blocked-trap” list. When a

log string does not match either list, it is allowed through. If it matches the "allowed" list, the log message is let through and the severity is modified. If it matches the "blocked" list, the log event is not generated.

In addition, a filter for each category type has been created. This filter contains each trap that falls under the filter. The following are the eight trap categories:

- CSTA
- DOS
- H.323
- LB
- SIP
- System
- TLS
- generic

This filtering mechanism is now available under each event log type.

Syntax

```
config services event-log snmp-trap
```

Properties

admin: Enables or disables the ME's ability to translate event logs into SNMP traps.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

filter: Specifies the event message filter log class and severity level for transferring event-logs to SNMP traps. Repeat the command to specify multiple event filters.

Default: all
Example: **set filter sipRouting debug**

advanced-filters

Secondary object. This object allows you to configure more granular snmp-trap events. Specify allowed and blocked events (for example the server state change).

Syntax

```
config services event-log syslog advanced-filter
config services event-log file advanced-filter
config services event-log local-database advanced-filter
config services event-log external-database advanced-filter
config services event-log cli advanced-filter
```

Properties

allowed-event: *Secondary object*. Enter specific events you want allowed by the SNMP trap feature.

Default: There is no default setting
Example: **set allowed-event "SIP server peer (.+) server (.+) changed" info**

blocked-event: *Secondary object*. Enter specific events you want blocked by the SNMP trap feature.

Default: There is no default setting
Example: **set blocked-event (.*)**

syslog

Enables and disables a remote syslog server, specified by the syslog server IP address, and sets the filters to define which events the ME sends.

Syntax

```
config services event-log syslog ipaddress
```

Properties

admin: Enables or disables the ME event logging to the remote syslog server.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

filter <log-class><severity>: Specifies the event message filter log class and severity level for messages forwarded to the syslog server. Repeat the command to specify multiple event filters. See Using Filters with Event Log Messages for complete information.

Default: There is no default setting
Example: **set filter snmp warning**

facility: Sets the user-defined syslog facility to which the ME logs system events. Syslog facilities help isolate the origin of messages written to the syslog server.

Default: user
Values: user | local0 | local1 | local2 | local3 | local4 | local5 | local6 | local7
Example: **set facility local5**

file

Specifies the ME configuration settings for the named event log file. Specify the name using up 64 alphanumeric characters with no blank spaces. Optionally, you can specify directory file paths using the forward slash (/) character. Additionally, you set the filters to define which events the ME sends with this object.

Syntax

```
config services event-log file name
```

Properties

admin: Enables or disables the system event log file.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

filter: Specifies the event filter type and severity level for messages written to the event log. Repeat the command specify multiple event filters. See Using Filters with Event Log Messages for complete information.

Default: There is no default setting
Example: **set filter snmp warning**

size: Set the maximum allowable size, in megabytes, of the event log file.

Default: 10
Values: Min: 1 / Max: 100
Example: **set size 25**

count: Specifies the maximum number of event log files to create when the event log file reaches the maximum size in megabytes.

When the maximum count is reached, the first file in the rotation is cleared for rewriting and the count is resumed.

Default: 5
Values: Min: 1 / Max: 20
Example: **set count 15**

local-database

Specifies the configuration settings for storing events in the ME local database and sets the filters to define which events the ME sends.

Syntax

```
config services event-log local-database
```

Properties

admin: Enables or disables the system local database. When **disabled**, the system does not write event log messages to the local database. However, you can still view any messages previously in the local database using the **show event-log** command.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

filter: Specifies the event filter type and severity level for messages written to the local database. Repeat the command to specify multiple event filters. See Using Filters with Event Log Messages for complete information.

Default: There is no default setting
Example: **set filter snmp warning**

history: Sets the maximum number of days to store events in the local database. When the maximum number of days is reached, the local database is cleared and is restarted at the first day.

Default: 100
Values: Min: 1 / Max: 10000
Example: **set history 50**

external-database

Specifies the external (remote) database that serves as a target for event messages. Define this database with the **database** object. Additionally, you set the filters to define which events the ME sends with this object.

For more information on the services/database object, refer to the *Oracle Communications WebRTC Session Controller Administration Guide*.

Syntax

```
config services event-log external-database name
```

Properties

admin: Enables or disables the external database configuration. When **disabled**, the system does not write event log messages to the database.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

filter: Specifies the event filter type and severity level for messages written to the external database. Repeat the command to specify multiple event filters. See Using Filters with Event Log Messages for complete information.

Default: There is no default setting
Example: **set filter snmp warning**

history: Sets the maximum number of days to store events in the external database. When the maximum number of days is reached, the local database is cleared and is restarted at the first day.

Default: 100
Values: Min: 1 / Max: 10000
Example: **set history 50**

cli

Globally enables or disables writing of events to the CLI and sets the filters to define which events the ME sends. Use the **log-target** action to enable or disable the feature for the current CLI.

Syntax

```
config services event-log cli
```

Properties

admin: enables or disables writing event messages to the CLI.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

filter: Specifies the event filter type and severity level for messages written to the CLI. Repeat the command to specify multiple event filters. See Using Filters with Event Log Messages for complete information.

Default: There is no default setting
Example: **set filter snmp warning**

smtp

Enables and disables mailing of events to a designated SMTP server and sets the filters to define which events the ME sends. The ME then collects the events into an email and sends them to the SMTP server once every minute. Enter the host name or IP address of the SMTP server to open this object.

Syntax

```
config services event-log smtp host
```

Properties

admin: Enables or disables the SMTP server event log archiving configuration. When **enabled**, the system emails session event logs to the specified address. When **disabled**, the system does not email the log files.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

destination-mailbox: Specifies the email address to which the system sends the session event logs.

Default: There is no default setting
Example: **set destination-mailbox admin@companyABC.com**

reply-mailbox: Specifies the address that appears in the "From" field of the emailed event logs. If you do not specify this property, the "From" field is empty.

Default: There is no default setting
Example: **set reply-mailbox events@companyABC.com**

port: Specifies the port number over which the system should communicate with this SMTP server.

Default: 25
Values: Min: 1 / Max: 65535
Example: **set port 100**

connection-keepalive: Specifies the length of time, in minutes, that the system keeps the connection to the SMTP server open. This prevents opening and closing the connection with each event.

Default: 5
Values: Min: 5 / Max: 60
Example: **set connection-keepalive 20**

filter <log-class><severity>: Specifies the event filter type and severity level for messages forwarded to the SMTP server. Repeat the command to specify multiple event filters. See Using Filters with Event Log Messages for complete information.

Default: There is no default setting
Example: **set filter dns crit**

tivoli

Enables and disables sending of events to a designated Tivoli server and sets the filters to define which events the ME sends. Enter the host name or IP address of the Tivoli server to open this object.

Syntax

```
config services event-log smtp host
```

Properties

admin: Enables or disables the Tivoli server event log archiving configuration. When **enabled**, the system sends session event logs to the specified address. When **disabled**, the system does not send the log files.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

protocol: Specifies the protocol the system uses to communicate with the Tivoli server.

Default: tcp
Values: tcp | udp
Example: **set protocol udp**

port: Specifies the port number over which the system should communicate with this Tivoli server.

Default: 7500
Values: Min: 1 / Max: 65535
Example: **set port 7501**

filter <log-class><severity>: Specifies the event filter type and severity level for messages forwarded to the Tivoli server. Repeat the command to specify multiple event filters. See Using Filters with Event Log Messages for complete information.

Default: There is no default setting
Example: **set filter dns crit**

database

Defines the external database. Configure the ME to use this database as an external log target using the **external-database** object. Consult your database administrator for information regarding authentication on the remote database before configuring this object.

Syntax

```
config services database name
```

Properties

driver: Specifies the name of the Open Database Connectivity (ODBC) driver associated with the database.

Default: There is no default setting
Example: **set driver psq10DBC**

username: Specifies the user name needed for the system to access the database. This is the name the database expects to see when authenticating requests.

Default: There is no default setting
Example: **set username nnos-e**

secret-tag: Specifies the secret tag (and password) needed for the system to access the database. This is the secret the database expects to see when authenticating requests. See Understanding Passwords and Tags for information on the ME two-part password mechanism.

Default: There is no default setting
Example: **set secret-tag 123**

options: Sets the options specified by the database. Use this to identify the location of the database.

Default: There is no default setting
Example: **set options connection 192.168.100.100**

instrument

The instrumentation settings are for debugging and are intended for Technical Support use only. Do not use this object without specific instructions from Oracle personnel.

data-locations

Specifies the directory and path locations on the ME where you would like to save certain types of information. This information includes:

- Accounting records prior to their being written to accounting targets.
- RTP media and mixed, for recording and playback of recorded calls.
- File transfer records.
- Log files.

Configuring these locations is optional; the ME provides default directory path locations. If you choose not to configure locations, the default directory path for all file types is **/cxc_common/*** on the system hard-drive-1.

You can also configure multiple path locations. When you set a location, the ME adds the location to the list of possible paths. The ME uses these secondary locations when it reaches the fail-threshold (set with the **storage-device** object). You must use the **remove** command to delete an entry from the save/search list.

The ME handles the location selection as follows:

- For files being written (call recordings, file transfer records, and log files), the ME searches for an available location in the order in which the files were created. An available location is one that is mounted and not full.
- For files being read (call playback), the ME searches all locations.

You can display the default directory file paths or the search order using the **show** command.The accounting process then reads those records in and services the various accounting targets. The file system acts as a large storage queue, providing two distinct benefits:

Saving accounting records

The ME Media Engine saves all accounting records to an internal files system before writing them to defined targets. Therefore, the accounting function has a more secure backup and more efficient record writing:

- Records can be saved until they are successfully written to the target. In addition, for a configurable amount of time they can be reapplied from the file system to the destination target if the target encounters problems, providing record recovery.
- Record writing is more efficient because it is not bound to call flow or dependent on target capabilities. The queueing mechanism of an external target could cause queue overflow and data loss. With an internal file system, files can be written to the target at a rate the target can handle.

The file system is made up of a root directory and subdirectories to hold the records. See the VSP accounting object for configurable options such as subdirectory size, purge criteria, and record retention periods.

Syntax

```
config services data-locations
```

Properties

accounting-root-directory: Sets the location where the system writes accounting records prior to their being sent to various configured accounting targets. For optimal record access, the ME maintains an internal file structure of subdirectories within this specified root directory. Use the VSP **accounting** object to set the number of records stored in each subdirectory.

Default: /cxc_common/accounting
Example: **set accounting-root-directory /acme_common/acct_records**

rtp-recorded: Sets the location in which the system saves call recordings.

Default: /cxc_common/rtp_recorded; the backup is /cxc/recorded
Example: **set rtp-recorded /acme_common/rtp_recorded**

rtp-recorded-rotation: Sets the rotation scheme for writing recorded files to a directory. The system writes the files to the directories configured with the **rtp-recorded** property.

Default: first-available

- **Values: first-available**: The system writes to the first directory listed with the **rtp-recorded** property that has enough space to hold the recording. The system will continue to write to that directory until the disk fills, and then it will move to the next directory in the list.
- **round-robin**: The system rotates through all configured directories in a round-robin manner. This can allow an increase in the volume of simultaneous recorded calls by spreading the load across multiple disks.

Example: **set rtp-recorded-rotation round-robin**

rtp-mixed: Sets the location in which the system writes for playback of recorded calls. This is where RTP files are "mixed" to create files that can then be played back.

Default: /cxc_common/rtp_mixed; the backup is /cxc/mixed
Example: **set rtp-mixed /acme_common/rtp_mixed**

file-transfer-recorded: Sets the location in which the system saves file transfer records.

Default: /cxc_common/ft_recorded; the backup is /cxc/recorded
Example: **set rtp-mixed /acme_common/ft_recorded**

log: Sets the location in which the system saves log files.

Default: /cxc_common/log
Example: **set log /acme/log**

lnp-record-directory: Provides a customer-specific application implementation and is not otherwise applicable.

Default: /cxc_common/lnp
Example: **set lnp-record-directory /acme_common/lnp**

rtp-on-demand-recorded: Sets the location to which the ME writes files for on-demand recorded calls.

Default: /cxc_common/rtp_on_demand_recorded
Example: **set rtp-on-demand-recorded /recorded_directory**

rtp-on-demand-recorded-rotation: Sets the rotation scheme for writing on-demand recorded files to a directory. The ME writes the files to the directories configured in **rtp-on-demand-recorded.**

Default: first-available

- Values: first-available: The ME writes to the first directory listed in the **rtp-on-demand-recorded** property that has enough space to hold the recording. The ME continues to write to that directory until the disk fills, and then moves to the next directory in the list.
- round-robin: The ME rotates through all configured directories in a round-robin manner. This allows an increase in the volume of simultaneous recorded calls by spreading the load across multiple disks.

Example: **set rtp-on-demand-recorded-rotation round-robin**

nfs

Configures the ME as a Network File System (NFS) client. NFS is Sun Microsystems' open protocol suite that allows computers running different operating systems to access shared files and share disk storage. It is the UNIX equivalent of Server Message Block (SMB). NFS allows users to import and export local files through an interface called the Virtual File System (VFS), which runs on top of TCP/IP.

When you enable an NFS entry, the ME acts as an NFS client, enabling remote mounted disk storage. Enter a name to open the nfs object. This specifies a mount point (a node in the ME file directory). For example, if you name your **nfs** object alpha, then the external files are visible at /mnt/alpha/. You may want access to multiple servers, and/or multiple directories from a single server. In that case, each would have a separate NFS configuration object, with a unique mount point.

Syntax

```
config services data-locations nfs name
```

Properties

admin: Enables or disables the mount point.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

server: Specifies the IP address or host name of the NFS server. The system, as a NFS client, has access to the server file system.

Default: There is no default setting
Example: **set server 192.168.10.10**

share: Specifies the point in the file system that is being shared. When you configure an NFS server, you specify which directory is shared out (as well as read/write permissions and other properties).

Default: There is no default setting
Example: **set share /home/staff/nfs**

version: Specifies the version on NFS to use. The ME supports versions 2 and 3.

Default: 3

Values: 2 | 3
Example: **set version 2**

protocol: Specifies the protocol to use when communicating with the server.

Default: udp
Values: udp | tcp
Example: **set protocol tcp**

timeout: Specifies how long the system waits when trying to read from or write to the server. When the timeout value expires, the system cancels the action. If timeout is set to **default**, the driver determines the best value, depending on the version and/or protocol set.

Default: default
Values: default | custom *milliseconds (100-65535)*
Example: **set timeout 150**

storage-device

Sets the levels at which the ME warns of approaching disk capacity and the frequency of those warnings. In addition, you set the level at which writes to the disk drive fail. If you have set backup file path locations (using the **data-locations** object), when a disk drive reaches the configured fail threshold setting, The ME begins WRITE operations to the next available disk drive.

The **storage-device** object operates on all installed disk drives. If all disk drives have reached the configured free space threshold, media call recording, file transfers, and log files will no longer be written to the ME disks.

Note: Currently, the ME devices support multiple disk drives.

Syntax

```
config services storage-device
```

Properties

fail-threshold: Sets a threshold, in megabytes, at which the system no longer writes recorded calls or IM files to the disk drive. The system sends a warning message to the event log (and an SNMP trap) indicating that space on the internal disk drive has been exceeded. The system checks the fail threshold each time it receives a call.

Default: 10000
Values: Min: 200 / Max: 400000
Example: **set fail-threshold 15000**

tasks

Opens the tasks object, from where you can configure and schedule archiving and maintenance tasks for a VSP.

Syntax

```
config services tasks
```

Properties

config-update-task: Specifies the action to be performed when the configuration is modified.

Default: There is no default setting
Example: **set config-update-task restart**

task

Sets the action and the schedule for a task. A task can only contain one action. To schedule additional actions, create separate tasks. Each action uses its own set of arguments, described in the **arguments** property. (These are the same arguments you would supply when executing the function as an action instead of a task.)

Syntax

```
config services tasks task name
```

Properties

schedule: Sets the frequency with which the system executes the task. When entering the time (for **time-of-day** and **once**), you can enter the time in regular time format (for example, 3:00). The system displays the time in the format hh:mm:ss (for example, (03:00:00).

Default: disabled

- **Values: disabled**: The task is not executed, but the configuration remains.
- **period**: The task is executed with the specified frequency. Enter the interval, in hours from 1 to 288, between executions.
- **time-of-day**: The task is executed at the specified time for the number of days specified, between 1 and 12.
- **days**: The task is executed at the specified time on the days specified.
- **once**: The task is executed once at the specified time. If you do not enter a date, the system uses the current day.

Example: **set schedule time-of-day 1:00**

action: Sets the action that the task performs. Each action has a dependent set of arguments. See descriptions below for complete action/argument details.

arguments: Sets the arguments for the selected action. See descriptions below for complete action/argument details.

Note that when you are entering more than one word or value for an argument you must enclose the string in quotation marks.

The following table shows the task actions and arguments.

Table 38–1 Task Actions and Arguments

Action name	Description
action **archive**: arguments *vsp*N*ame*	Saves stored sessions for the VSP. You must also enable archiving through the **archiving** object. See the **archive** action for more information. Note that if you have the **record-count** property of the archiving object set to any value other than 0, the archiving task will fail. set arguments vsp1 The default VSP archived is **default**.
action **call-failover** arguments flush	Flushes the call-failover database of any signaling and media-session records used to maintain call state between redundant ME devices. See the **call-failover** action for more information. set arguments flush There is no default setting.
action **database** arguments {{delete | vacuum | vacuum-full | drop} *database* [*table*] | repair {translate | data-recovery} | initialize | snapshot {*integer* | force | automatic}}	Deletes or cleans database records (for databases you configured with the master services' **database** object). See the **database** action for more information. set arguments "snapshot log force" There is no default setting.
action **database-maintenance**	Executes a multistep maintenance operation on entries found in the system database tables to optimize database access. See the **database** action for argument descriptions and more information. set arguments "save backup.xml" Requires no arguments.
action **directory-clean**	Removes empty recorded media directories. You may have an empty directory, for example, if the system cleaned a directory as part of a scheduled maintenance operation. That action removes data but leaves the directories. This action takes no arguments.
action **directory-reset** arguments *vspName* [true | false]	Resets the enterprise directory, causing the system to reread the directory and update the user information. Enter the name of the VSP that houses the directory. In addition, you can set a directory purge action of true or false: **true**: Clears out the contents of the database and then repopulates it. **false**: Updates the database but leaves users that are no longer in the directory itself in the database. set arguments "vsp1 false" If you do not enter a VSP name, the system uses the VSP **default**. For the directory-reset action, the default purge action is **true**.
action **external-normalization** arguments {replace-file *fileName* | replace-url *source* | flush}	Manages the file used to maintain DNIS-to-ANI translation data. See the **external-normalization** action for more information. set arguments flush There is no default setting.

Table 38–1 (Cont.) Task Actions and Arguments

Action name	Description
action **external-presence** arguments {delete *url* | flush}	Clears all or a specified entry from the external presence cache. The external cache is the database running on the backup ME device in a cluster configuration. See the **external-presence** action for argument descriptions and more information. set arguments flush There is no default setting.
action **external-session** arguments flush	Removes all entries from the external CSTA SIP session cache. See the **external-session** action for more information. set arguments flush There is no default setting.
action **file-based-word-lists-refresh**	Rereads any saved **word-list** or **url-list file** entry into memory. See the **file-based-word-lists-refresh** action for argument descriptions and more information. set arguments “delete 5085551212@abc.com” This action takes no arguments.
action **file-transfer-delete-old** arguments *days*	Deletes all files brought on to the system, via a file transfer, that are older than the specified number of days. See the **file-transfer-delete-old** action for more information. set arguments 30 Enter a number of days between 1 and 1,000. The default number of days is 7.
action **install** arguments {file *source* [box | cluster | controlled] | url *source* [box | cluster | controlled] | nic [*model*] | nic-reinitialize | module | cancel}	Manages system software releases and network interface cards (NICs). See the **install** action for more information. set arguments “file release.tar.gz controlled” There is no default setting.
action **load-balancing-failover** arguments flush	Deletes all recorded media files older than the specified number of days. See the **media-delete-old** action for more information. set arguments flush There is no default setting.
action **location-database** arguments {merge [*filePath*] | replace [*filePath*] | save [*filePath*] | delete *aor* | flush}	Manages the location database across the cluster. See the **location-database** action for argument descriptions and more information. set arguments “save backup.xml” For merge, replace, and save options, the default location is **/cxc/location.xml**.
action **loopback** arguments *{packet | packet-init} seconds to [from] [any | udp | tcp | tls]*	Establishes an outgoing SIP loopback call. See the **loopback** action for more information. set arguments packet 10 sip:5554443211@jane.cov.com The default duration is 10 seconds.

Table 38–1 (Cont.) Task Actions and Arguments

Action name	Description
action **media-delete-old** arguments *days*	Deletes all recorded media files older than the specified number of days. See the **media-delete-old** action for more information. set arguments 30 Enter a number of days between 1 and 1,000. The default number of days is 7.
action **orderly-restart** arguments {warm | cold | halt | cluster}	Causes a restart of the type specified after gracefully terminating any existing connections. See the **orderly-restart** action for argument descriptions and more information. set arguments cluster The default type is **warm**.
action **presence** arguments {merge *fileName* | replace *fileName* | save *fileName* | delete *URL* | flush}	Manages the presence cache. See the **presence** action for argument descriptions and more information. set arguments "delete 5085551212@abc.com" The default type is **warm**.
action **restart** arguments {warm | cold | halt | cluster}	Causes an immediate restart of the type specified. See the **restart** action for argument descriptions and more information. set arguments cluster There is no default setting.
action **uri-alias** arguments {lookup *AOR* | reset | purge | seek *AOR*}	Causes an immediate restart of the type specified. See the **uri-alias** action for argument descriptions and more information. set arguments purge There is no default setting.

network

Sets system network parameters. By fine-tuning these settings, you gain greater control over network behavior. Generally the default TCP settings are adequate, however, so use caution before making changes.

Syntax

```
config services network
```

Properties

tcp-keepalive: Specifies the time, in seconds, that an established TCP connection can remain idle before the system sends a keepalive to the client. The idle time expiration initiates the keepalive process.

Default: 600
Values: Min: 30 / Max: 14400
Example: **set tcp-keepalive 1200**

tcp-keepalive-probes: Specifies the number of unanswered TCP keepalive probes that are allowed before the system disconnects an idle session.

Default: 5
Values: Min: 2 / Max: 16

Example: **set tcp-keepalive-probes 10**

tcp-keepalive-interval: Specifies the time, in seconds, that the system waits for a response from a keepalive probe before ending the next one. The system continues to send probes until it has sent the number specified in the **tcp-keepalive-probes** property.

Default: 6
Values: Min: 1 / Max: 60
Example: **set tcp-keepalive-interval 10**

tcp-max-syn-backlog: Specifies the maximum number of queued (unacknowledged) connection requests allowed before the system begins dropping requests. This value is set to help prevent a TCP SYN flood attack.

Default: 1024
Values: Min: 16 / Max: 131027
Example: **set tcp-max-syn-backlog 1536**

tcp-synack-retries: Specifies the number of times the system will retransmit a SYN-ACK in response to a SYN. If the number of retries is reached without a successful response, the system deletes the new connection from the table. This value helps minimize the effects of a SYN flooding attack.

Default: 5
Values: Min: 1 / Max: 5
Example: **set tcp-synack-retries 4**

tcp-syncookies: Enables or disables SYN cookie support in the kernel. When **enabled**, the kernel handles TCP SYN packets normally until the queue is full. Then, the kernel replies to a SYN with an intentionally modified TCP sequence number. A legitimate connection uses the number in the third packet of the three way handshake, allowing the system to verify and allow the connection, even though there is no corresponding entry in the SYN queue. An attacker would not respond with the sequence number and the connection is dropped.

Default: enabled
Values: enabled | disabled
Example: **set tcp-syncookies disabled**

tcp-fin-timeout: Specifies the number of seconds the system waits for a final FIN packet before forcibly closing the socket. The system uses the FIN packet to disconnect a TCP connection, whether it's idle or not.

Default: 60
Values: Min: 2 / Max: 300
Example: **set tcp-fin-timeout 100**

monitors

Opens the **monitors** object, through which you create monitoring configurations for tracking usage and TLS statistic threshold violations.

Syntax

```
config services monitors
```

Properties

None

monitor

Sets threshold monitors for usage and TLS activity. When the threshold is exceeded, the ME sends a message to the event log and creates an SNMP trap. You can set the interval at which the ME polls the system and compares the current statistics against parameter thresholds.

Syntax

```
config services monitors monitor name
```

Properties

interval: Specifies the number of minutes the system waits between polls of the specified parameters.

Default: 10
Values: Min: 1 / Max: 60
Example: **set interval 50**

parameter: Sets the parameter to monitor and the threshold that, when exceeded, results in a message to the event log and an SNMP trap. Re-execute the command to add parameters.

Default: There is no default setting

- Values: cpu-usage *percentage*
- memory-usage *percentage*
- kernel-memory-usage *percentage*
- memory-failures *failures*
- tls-connections *connections*
- tls-failures *failures*
- storage-devices *device percentage*
- mos-failures *value*
- syn-cookies *cookies*
- dropped-media-packets *packets*
- sip-parse-errors *errors*

Example: **set parameter cpu-usage 90**

troubleshooting

Sets the number of troubleshooting web service requests that can be handled by the ME at one time. The object also sets an allowed wait time for pending requests.

Syntax

```
config services troubleshooting
```

Properties

concurrent-requests: Specifies the number of concurrent web service troubleshooting requests the system attempts to service. If this threshold is reached, subsequent requests are queued for processing. They remain in the queue until:

- They are processed because the queue dropped below the threshold.
- They time out because they exceeded the maximum wait time assigned with the **concurrent-timeout** property.

Default: 2
Values: Min: 1 / Max: 20
Example: **set concurrent-request 5**

concurrent-timeout: Specifies the maximum amount of time a troubleshooting request waits to be serviced before the system cancels the request.

Default: 2000
Values: Min: 10 / Max: 120000
Example: **set concurrent-timeout 5000**

collect

Configures the handling of data collection output files.

Syntax

```
config services collect
```

Properties

directory: Specifies where the data collection output files will be stored. While the default directory is sufficient in most cases, if you are collecting the contents of a large database, this property allows you to specify a mount with more available disk space.

Default: /cxc_common/collect
Example: **set directory /cxc_common/collect_directory**

max-old-files: Specify the maximum number of old files the AA-SBC saves before backups are deleted.

Default: 5
Values: Min: 1 / Max: 50
Example: **set max-old-files 25**

default-collect-settings

Enables or disables the default collection parameters. When one of these properties is set to disabled, the corresponding data is not collected.

Syntax

```
config services collect default-collect-settings
```

Properties

config: Enable or disable the collection of configuration data.

Default: enabled
Values: enabled disabled
Example: **set config disabled**

certificates: Enable or disable the collection of certificate data.

Default: enabled
Values: enabled disabled

Example: **set certificates disabled**

status: Enable or disable the collection of status data.

Default: enabled
Values: enabled disabled
Example: **set status disabled**

crash-files: Enable or disable the collection of crash file data.

Default: enabled
Values: enabled disabled
Example: **set crash-files disabled**

log-files: Enable or disable the collection of log file data.

Default: enabled
Values: enabled disabled
Example: **set log-files disabled**

status-class: Enable or disable the collection of status class data.

Default: enabled
Values: enabled disabled
Example: **set status-class disabled**

database: Specifies the databases you want to collect. This property is a vector, so you can specify multiple entries. Note: Use this property with caution as it is possible to specify the collection of enormous amounts of data.

Default: There is no default setting

- Values: log
- spotlite
- status
- dos
- directory
- accounting

Example: **set database accounting**

directory: Specifies any additional directories you want collected. This property is a vector, so you can specify multiple entries.

Note: Use this property with caution as it is possible to specify the collection of enormous amounts of data.

Default: There is no default setting
Example: **set directory /cxc_common/data1/dir1**

trace-files: Enables or disables the collection of trace files on the ME.

Default: enabled
Values: enabled | disabled
Example: **set trace-files disabled**

license-files: Enables or disables the collection of license files on the ME.

Default: enabled
Values: enabled | disabled
Example: **set license-files disabled**

collect-group

Configures custom collection parameters as well as the default parameters.

Syntax

```
config services collect collect-group
```

Properties

config: Enable or disable the collection of configuration data for this collect-group.

Default: enabled
Values: enabled disabled
Example: **set config disabled**

certificates: Enable or disable the collection of certificate data for this collect-group.

Default: enabled
Values: enabled disabled
Example: **set certificates disabled**

status: Enable or disable the collection of status data for this collect-group.

Default: enabled
Values: enabled disabled
Example: **set status disabled**

crash-files: Enable or disable the collection of crash file data for this collect-group.

Default: enabled
Values: enabled disabled
Example: **set crash-files disabled**

log-files: Enable or disable the collection of log file data for this collect-group.

Default: enabled
Values: enabled disabled
Example: **set log-files disabled**

status-class: Specifies additional status classes to be collected for this collect-group. This property is a vector, so you can specify multiple entries. In addition, wildcards can be specified as well as the -v property to specify a verbose display in the status output file.

Default: There is no default setting
Example: **set status-class location-bindings-rejected -v**

database: Specifies the databases you want to collect for this collect-group. This property is a vector, so you can specify multiple entries. Note: Use this property with caution as it is possible to specify the collection of enormous amounts of data.

Default: There is no default setting

- Values: log
- spotlite
- status
- dos
- directory
- accounting

Example: **set database accounting**

directory: Specifies any additional directories you want collected for this collect-group. This property is a vector, so you can specify multiple entries.

Note: Use this property with caution as it is possible to specify the collection of enormous amounts of data.

Default: There is no default setting
Example: **set directory /cxc_common/data1/dir1**

trace-files: Enables or disables the collection of trace files on the ME.

Default: enabled
Values: enabled | disabled
Example: **set trace-files disabled**

license-files: Enables or disables the collection of license files on the ME.

Default: enabled
Values: enabled | disabled
Example: **set license-files disabled**

39

Configuring Session Configuration Objects A Through M

This chapter covers session configuration objects, A through M, alphabetically. For session configuration objects N through Z, see "Configuring Session Configuration Objects N Through Z." The session configuration objects define the way in which the Media Engine (ME) handles SIP-based signaling and media traffic. The session configuration that is applied to an active call through the ME depends on configuration of other aspects of the system.

There are several places in the configuration hierarchy through which you can access the session configuration objects. The path to these object defines in which cases the ME uses that configuration. Locations for session configuration are defined in the following table.

Table 39–1 Session Configuration Locations

Path	Defines...
vsp > default-session-config	The session configuration settings to apply to those SIP calls for which there are no configured policies. See Configuring Default Session Configuration Objects, for more information.
vsp > policies > session-policies > policy > rule	The session configuration settings to apply to SIP calls for which a configured policy exists. See Configuring Policy Objects, for more information.
vsp > dial-plan > dial-prefix **vsp > dial-plan > route** **vsp > dial-plan > source-route**	The session configuration settings to apply to calls based on the dial prefix or domain suffix. See Configuring Dial Plan Objects, for more information.
vsp > calling-group > route **vsp > calling-group > source-route**	The session configuration settings to apply to calling-group member calls based on the dial prefix or domain suffix. See Configuring Calling Group Objects, for more information.
vsp > session-config-pool > entry	A saved session configuration that can be referenced within one or more dial plans. See Configuring Session Configuration Pool Objects, for more information.

3GPP

Configures 3rd Generation Partnership Project (3GPP) systems.

Syntax

```
config vsp default-session-config 3GPP
config vsp policies session-policies policy name rule name session-config 3GPP
config vsp dial-plan dial-prefix entryName session-config 3GPP
config vsp dial-plan route name session-config 3GPP
config vsp dial-plan source-route name session-config 3GPP
config vsp session-config-pool entry name 3GPP
```

Properties

None

accounting

Sets the target destination for call detail records. By selecting a target, you are configuring the ME to provide call logging of this SIP session. The records are then sent to the server, database, or file specified in this object. Note that you must configure the destination devices first, and then reference them here.

Syntax

```
config vsp default-session-config accounting
config vsp policies session-policies policy name rule name session-config
accounting
config vsp dial-plan dial-prefix entryName session-config accounting
config vsp dial-plan route name session-config accounting
config vsp dial-plan source-route name session-config accounting
config vsp session-config-pool entry name accounting
```

Properties

target: Sets the destination for the accounting records (SIP call detail records) created for this SIP session. When you set the target, you must specify a previously configured object, dependent on the target type:

Default: There is no default setting

- **Values: radius <***radiusGroup***>**: Logs the session to the group specified in the **radius-group** object.
- **diameter <***diameterGroup***>**: Logs the session to the group specified in the **diameter-group** object.
- **database <***databaseGroup***>**: Logs the session to an internal or external database group, as specified in the database object **diameter-group** subobject.
- **syslog <***syslogGroup***>**: Logs the session to an external syslog server group, as specified in the syslog object **diameter-group** subobject.
- **file-system <***path***>**: Writes the session to a file on the ME device, as specified in the **file-system > path** subobject.

Example: **set target syslog "vsp accounting syslog group Boston"**

header: Specifies a string that is written to each accounting record. Use this, for example, to track for later analysis a header that certain user agents output in their INVITE. This header can be seen in the **Arbitrary Header** field of the Call Record displayed through the ME Management System **Call Logs** tab.

Default: There is no default setting
Example: **set header UA1**

accept-mode: Specifies the activity that initiates the connect time in the accounting record. When **disabled**, the default, the connect time is recorded when the system transmits an ACK. When **enabled**, the connect time is recorded when the system receives a 200 OK message.

Default: disabled
Values: enabled | disabled
Example: **set accept-mode enabled**

disconnect-time-upon-receipt-of-bye: Enables or disables logging of the call disconnect time entry in the call detail record when the call session terminates with a BYE request. If set to **disabled** (the default), the disconnect time is recorded with the 200OK that follows the BYE request.

Default: disabled
Values: enabled | disabled
Example: **set disconnect-time-upon-receipt-of-bye enabled**

use-short-gateway-names: Specifies how the ME handles particular gateway fields in the CDR. When **enabled** (the default), the system populates the OrigGW and TermGW fields in the CDR with the **server-pool > server-pool-admission-control** name string for the originating and terminating SIP server gateway for a call, if known.

If set to **disabled**, the ME populates the OrigGW and TermGW fields in the CDR with the server name (as configured) followed by a "- "and then the **server-pool > server-pool-admission-control** name.

Default: enabled
Values: enabled | disabled
Example: **set use-short-gateway-names disabled**

reported-failed-calls: *Secondary property*. Specifies whether to send out accounting records for calls that did not connect. When **disabled**, the records are not sent. When **enabled**, a record of the call is sent to the target configured in your session configuration.

Default: disabled
Values: enabled | disabled
Example: **set report-failed-calls enabled**

accounting-data

Adds a custom data field to the accounting record. Use this object to define the content of the field. The ME supports several predefined selections for use with the entry property. (You can also add text string values.) To enter a predefined value, precede the letter with slash (and quotation marks in the CLI). For example, "\b" to add a box identifier. The following table shows the predefined selections for use with this object.

Table 39–2 Predefined Selections For This Object

Use	For...
\b	Box identifier
\d	Digest Realm
\s	Source LNP
\e	Destination LNP
\v	Diversion Header
\p	P-Charging-Vector Header

Table 39–2 (Cont.) Predefined Selections For This Object

Use	For...
\c	Cluster Name
\r	RADIUS Caller ID
\z	Call Connected Boolean
\y	File play - scan time
\x	File play - file duration
\w	File play - playback duration
\u	Disconnect reason text
\t	Disconnect reason code
\q	Post Dial Digits Info
\j	Source Jitter
\m	Source Max Jitter
\i	Destination Jitter
\k	Destination Max Jitter

Syntax

```
config vsp default-session-config accounting-data
config vsp policies session-policies policy name rule name session-config
accounting-data
config vsp dial-plan dial-prefix entryName session-config accounting-data
config vsp dial-plan route name session-config accounting-data
config vsp dial-plan source-route name session-config accounting-data
config vsp session-config-pool entry name accounting-data
```

Properties

entry: Specifies the content of the field added to the accounting record, in the format *tag=value*. Use this property, for example, to add a source based on calls matching this session configuration. Use the predefined elements (see the Purpose) to extract data from the call SIP headers.

Default: There is no default setting
Example: **set entry cluster-name "\c"**

post-process-expression: Configures a regular expression, and replacement text, to run against the TO and FROM fields of the CDR. Use post processing to, for example, remove unwanted commas that may appear as the result of data that was imported from a CSV file. The following example turns a comma into a dash.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Example: set post-process-expression (.*0, (.*) "\1 - 1\2"

custom-data-grouping: The characters used to associate custom data tags and values.

Default: =
Example: **set custom-data-grouping ^ custom-data-delimiter**: The characters used to separate group data entries.

Default: ;

Example: **set custom-data-delimiter ^**

named-variable-entry: Specifies the content of the field added to the accounting record in the format, tag=value.

Default: There is no default setting
Example: **set named-variable-entry \s**

added-body

The **added-body** configuration object allows you to add one or more SIP message body parts.

Syntax

```
config vsp default-session-config header-settings added-body
config vsp policies session-policies policy name rule name session-config
header-settings added-body
config vsp dial-plan dial-prefix entryName session-config header-settings
added-body
config vsp dial-plan route name session-config header-settings added-body
config vsp dial-plan source-route name session-config header-settings added-body
config vsp session-config-pool entry name header-settings added-body
config vsp policies session-policies policy default rule session-config
inbound-header-settings added-body
```

Properties

admin: Indicates whether or not the ME is adding the configured SIP message body part.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

content-type: Specifies the type of message body part to add.

Example: **set content-type body**

other-headers: Specify other headers to include with this added body party.

Default: There is no default setting
Example: **set other-headers head1 info**

content: Specify the body part content being added.

Default: There is no default setting
Example: **set content message body**

apply-to-methods: Specifies the message type to which the ME applies message body part changes.

Default: There is no default setting
Example: **set apply-to-methods INVITE**

apply-to-responses: Specifies whether to apply message body part changes to SIP requests or both requests and responses.

Default: No

- Values: No: Apply to requests only.
- Yes <*response-code*>: Apply to responses with the specified response-code.
- Both: Apply to both responses and requests.

Example: **set apply-to-responses both**

apply-to-dialog: Specifies whether to apply message body part changes to a specific dialog or not.

Default: both

- Values: both: Apply to both inbound and outbound dialogs
- inbound: Apply to inbound dialog only.
- outbound: Apply to outbound dialog only.

Example: **set apply-to-dialog inbound**

cseq: *Secondary property*. Sets a mechanism that further filters which SIP messages have the body part modifications applied.

Default: 0
Example: **set cseq 5**

create-on-failed-match: *Secondary property*. Specifies whether the ME should create a header even if the expression is not a complete match.

Default: true
Values: true | false
Example: **set create-on-failed-match false**

altered-body

Modifies the SIP message body in calls matching this session configuration. The ME searches the SIP message body for the specified string and replaces the string as required.

Syntax

```
config vsp default-session-config header-settings altered-body number
config vsp policies session-policies policy name rule name session-config
header-settings altered-body number
config vsp dial-plan dial-prefix entryName session-config header-settings
altered-body number
config vsp dial-plan route name session-config header-settings altered-body number
config vsp dial-plan source-route name session-config header-settings altered-body
number
config vsp session-config-pool entry name header-settings altered-body number
```

Properties

admin: Enables or disables this configuration entry.

Default: There is no default setting
Example: **set admin enabled**

altered-body: Specifies the text to match on in the message body and the replacement text for the matched string.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Example: **set altered-body "(?ms)(.*)<address uri=\""sip:(.*)@(.*):5060;(.*)" "\1<address uri=" "sip:\2@10.1.1.1:5060;\4"**

apply-to-methods: Specifies the message type to which the system applies message body changes. The system then changes the SIP message body (if a match occurs) in all messages of that type.

When you modify this value, the system overwrites the current setting with only the message types you specify. For example, if set to the default and you enter **CONTACT**, the system only alters CONTACT messages. Enter multiple types separated by a plus sign (+) with no spaces.

Default: INVITE
Example: **set apply-to-methods INVITE+CONTACT apply-to-responses**: Specifies whether to apply message body changes to SIP requests or requests and responses. Set to **no** to apply changes only to requests. Set to **yes** to apply to responses as well. If yes, you must set the response code to which it applies. Create additional altered-body profiles to change multiple response types.

Default: no
Values: no | yes *responseCode*
Example: **set apply-to-responses yes 200**

cseq: *Secondary property.* Sets a mechanism to further filter which SIP messages have the header expression modifications applied. If **cseq** is set to zero (the default), the ME applies the changes to all SIP messages. If set to any other value, the system only applies the changes to SIP messages having a CSEQ field that matches that value.

Default: 0
Example: **set cseq 100**

remove-body: *Secondary property.* When this property is set to true, the ME removes the SIP message body from the matching of SIP messages. This includes the "Content-Type" and other related headers.

Default: false
Values: true | false
Example: set remove-body true

apply-to-dialog: Allows you to configure where to apply these options for a session.

Default: both
- Values: inbound: Apply to inbound dialog only.
- outbound: Apply to outbound dialog only.
- both: Apply to both inbound and outbound dialogs.

Example: **set apply-to-dialog inbound**

altered-header

Modifies or creates header values in calls matching this session configuration. Both this and the **reg-ex-header** objects provide this functionality. Use this object, which is simpler, when possible. Use the **reg-ex-header** object for complex modification, for example, when multiple levels of change are required. Note that you can create multiple header-altering configurations. They are processed by the system in the order that they appear in the configuration.

Syntax

```
config vsp default-session-config header-settings altered-header number
config vsp policies session-policies policy name rule name session-config
header-settings altered-header number
config vsp dial-plan dial-prefix entryName session-config header-settings
```

```
altered-header number
config vsp dial-plan route name session-config header-settings altered-header
number
config vsp dial-plan source-route name session-config header-settings
altered-header number
config vsp session-config-pool entry name header-settings altered-header number
```

Properties

admin: Enables or disables this configuration entry.

Default: There is no default setting
Values: enabled | disabled
Example: **set admin enabled**

source-header: Specifies the URI from which the system initially derives the data that is to be written to the destination header.

Default: There is no default setting
Example: **set source-header to**

source-field: Specifies the portion of the URI that the system writes to the destination. Select either **user**, **host**, **selection**, or **value**. With the **user** and **host** options, the system writes the entire field to the destination. If you choose **selection**, specify the value within the URI to match on and the replacement text to write to the destination. Or, select **value** to write a specific string to the destination. In this case, the **source-header** field is ignored.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Values: user | host | selection *regEx replacement* | value *replacement*
Example: **set source-field user**

destination-header: Specifies the header to be created or modified by the properties set in this object. The system modifies this URI with the data from the source. If the header does not exist in the message, the system creates it.

Default: There is no default setting
Example: **set destination-header request**

destination-field: Specifies the field in the specified destination URI to overwrite. Select either **user**, **host**, or **display**. To overwrite the entire selected destination URI, select **full**.

Default: There is no default setting
Values: user | host | display | full
Example: **set destination-field full**

apply-to-methods: Specifies the message type to which the system applies header value changes. The system then changes the specified URI according to the settings of the header and destination properties of this object.

When you modify this value, the system overwrites the current setting with only the message types you specify. For example, if set to the default and you enter **INVITE**, the system only authenticates INVITE messages. Enter multiple types separated by a plus sign (+) with no spaces.

Default: INVITE
Example: **set apply-to-methods INVITE+REFER**

apply-to-responses: Specifies whether to apply header value changes to SIP requests or requests and responses. Set to **no** to apply changes only to requests. Set to **yes** to apply to responses as well. If yes, you must set the response code to which it applies. Create additional altered-body profiles to change multiple response types.

Default: no
Values: no | yes *responseCode*
Example: **set apply-to-responses yes 200**

session-persistent: Specifies to which messages in a session SessionManager should apply changes made with this object. When **enabled**, the ME applies any TO, FROM, or REQUEST URI changes to the first and all subsequent messages in a session. When **disabled**, the default, the system applies the changes only to the first message in the session.

Default: disabled
Values: enabled | disabled
Example: **set session-persistent enabled**

cseq: *Secondary property.* Sets a mechanism to further filter which SIP messages have the header expression modifications applied. If **cseq** is set to zero (the default), the ME applies the changes to all SIP messages. If set to any other value, the system only applies the changes to SIP messages having a CSEQ field that matches that value.

Default: 0
Example: **set cseq 100**

apply-to-dialog: Allows you to configure where to apply these options for a session. The following are valid values:

Default: both
- Values: inbound: Apply to inbound dialog only.
- outbound: Apply to outbound dialog only.
- both: Apply to both inbound and outbound dialogs.

Example: **set apply-to-dialog inbound**

authentication

Sets the authentication mode to use on this SIP session. For authentication services that involve remote servers (such as RADIUS and DIAMETER), you must configure these servers on the ME using the either the **radius-group** or **diameter-group** configuration objects. For the directory authentication mechanism, you must first configure the directory service in the **directory** configuration object.

Syntax

```
config vsp default-session-config authentication
config vsp policies session-policies policy name rule name session-config
authentication
config vsp dial-plan dial-prefix entryName session-config authentication
config vsp dial-plan route name session-config authentication
config vsp dial-plan source-route name session-config authentication
config vsp session-config-pool entry name authentication
```

Properties

mode: Sets the type of authentication the system uses for this SIP session. Optionally, you can set whether the authentication applies to inbound-only (**enabled**) or inbound and outbound (**disabled**) traffic.

Default: none disabled

- **Values: none <enabled | disabled>**: The system performs no authentication.
- **Local <enabled | disabled>**: The system uses the username and password configured in the **user** object for authentication.
- **RADIUS <enabled | disabled> <***radiusGroupReference***>**: The system performs RADIUS authentication according to the configuration specified in the **radius-group** object.
- **DIAMETER <enabled | disabled><***diameterGroupReference***>**: The system performs DIAMETER authentication according to the configuration specified in the **diameter-group** object.
- **Directory <enabled | disabled><***directoryReference***>**: The system expects the user credentials that are specified in the **directory** service that you supply.

Example: **set mode radius enabled "vsp radius-group boston1"**

session-starter-only: Specifies which requests the system challenges. When **disabled** (the default), if authentication is enabled the system challenges all requests in a session. When **enabled**, the system only challenges the first request in a session.

Default: disabled
Values: enabled | disabled
Example: **set session-starter-only enabled**

handle-challenge-locally: Sets whether a challenge is handled locally. When **enabled**, the system terminates the original challenge response (either 401 Unauthorized or 407 Proxy Authentication Required) and generates a new request with the authentication information. When **disabled**, the system forwards the 401/407 response back to the UAC.

Default: disabled
Values: enabled | disabled
Example: **set handle-challenge-locally enabled**

challenge-response-code: Sets the response code that the system sends when it terminates the original challenge response (either 401 Unauthorized or 407 Proxy Authentication Required). This code is only applied when the **handle-challenge-locally** property is enabled.

Default: 401
Values: 401 | 407
Example: **set challenge-response-code 407**

apply-to-methods: Specifies which message types to authenticate. This setting is used by the **registration-throttling** property of the **route** and **source-route** registration plan objects to define which message types require authentication.

When you modify this value, the system overwrites the current setting with only the message types you specify. For example, if set to the default and you enter **INVITE**, the system only authenticates INVITE messages. Enter multiple types separated by a plus sign (+) with no spaces.

Default: INVITE+REGISTER+BYE
Example: **set apply-to-methods INVITE+REFER+REGISTER**

exclude-scheme-in-called: *Secondary property.* Specifies which portions of the TO URI that the ME uses for authentication. If set to **false**, the system authenticates the full TO URI. If **true**, the system uses only the User and Host portions of the TO URI.

Default: false
Values: true | false
Example: **set exclude-scheme-in-called true**

initial-challenge-stale: Specifies whether the stale parameter is included in authentication challenges, per RFCs 2069 and 3261.

Default: true
- Values: true: Includes stale="true" in the challenge.
- false: Includes stale="false" in the challenge.
- none: The stale parameter is not included in the challenge.

Example: **set initial-challenge-stale none**

authorization

Sets the type of authorization the system performs for matching sessions. Using this object, you set the protocol the system uses to get authorization data: either none, the route-server engine, or WSDL. This data results in a list of routing options for the call. For route-server, you must configure the **route-server** master service and, for intercluster lookups, a **diameter** client and **server**. For WSDL, you configure an **external-services > policy-group**.

Syntax

```
config vsp default-session-config authorization
config vsp policies session-policies policy name rule name session-config
authorization
config vsp dial-plan dial-prefix entryName session-config authorization
config vsp dial-plan route name session-config authorization
config vsp dial-plan source-route name session-config authorization
config vsp session-config-pool entry name authorization
```

Properties

mode: Sets the method to use for authorization data retrieval.

Default: none
- **Values: None**: The system performs no route-server lookup. This is the equivalent of administratively disabling the route-server service for matching calls.
- **Local**: The system performs intracluster route-server lookup; it sends the route lookup request to the system hosting the **route-server** master service.
- **WSDL <***policyGrpReference***><true | false>**: The system sends the request to the external services policy server specified in the **policy-group** configuration. Optionally, you can specify whether to send SIP message headers and/or content (both default to **false**) with the request.
- **Diameter <***diameterGrpReference***>**: The system sends the route request to the server specified in the **diameter-group** object configuration. This is typically only used for intercluster lookup configurations.
- RADIUS: The

- The ME sends a request to the RADIUS server with the to-URL and from-URL in the request. The RADIUS server responds with information that the ME uses to create session-configs that are applied to the session.

Example: **set mode Diameter "vsp diameter-group rsserver"**

always-perform-lookup: Specifies whether the system should do an authorization lookup (if configured to do so with the **mode** property). If set to **true**, the default, the system retrieves authorization data regardless of other configuration settings. If set to **false**, the system first uses internal logic to determine whether session handling data can be derived from other sources (e.g., location cache or dial plan). Set this to **false**, for example, when handling two locally registered phones calling each other.

Default: true
Values: true | false
Example: **set always-perform-lookup false**

apply-to-methods: Specifies to which message types the system applies authorization processing.

When you modify this value, the system overwrites the current setting with only the message types you specify. For example, if set to the default and you enter **INVITE**, the system only authorizes INVITE messages. Enter multiple types separated by a plus sign (+) with no spaces.

Default: INVITE
Example: **set apply-to-methods INVITE+REGISTER**

sequence: Select an existing sequence to use for querying the route server. This is a sequence you must have configured in the **vsp > route-server-config > route-server-sequence object**.

Default: There is no default setting
Example: **set sequence query1**

bodypart-type

Sets the body types that are allowed and/or prohibited during the session. This functionality is initiated (if configured) when the ME receives a SIP message that contains more than one type in the body portion of the message (when the Content Type header indicates that the message has multiple and mixed parts.) This object defines which types survive and which are deleted from the message.

Select a body type: application or text: and then a specific subtype. Use the custom MIME type to either add a body part other than application or text, or to use a subtype for application or text that is not preconfigured. To allow only a single body type, set **allowed-body-part** to the desired type and set **blocked-body-type** to **any**.

Syntax

```
config vsp default-session-config bodypart-type
config vsp policies session-policies policy name rule name session-config
bodypart-type
config vsp dial-plan dial-prefix entryName session-config bodypart-type
config vsp dial-plan route name session-config bodypart-type
config vsp dial-plan source-route name session-config bodypart-type
config vsp session-config-pool entry name bodypart-type
```

Properties

allowed-body-part: Sets the body part types allowed during the session. Re-execute the command for each type you want to allow.

Default: The default subtype setting for audio, video, and application is any; there is no default setting for custom-mime-type.
Values: audio *subType* | video *subType* | application *subType* | custom-mime-type *mimeType* subType
Example: **set allowed-body-part custom-mime-type application safe-trade**

blocked-body-part: Sets the body part types to prohibit during the session. Any body sections that contain this type are removed from the message before forwarding. Re-execute the command for each type you want to block.

Default: The default subtype setting for audio, video, and application is any; there is no default setting for custom-mime-type.
Values: audio *subType* | video *subType* | application *subType* | custom-mime-type *mimeType* subType
Example: **set blocked-body-part video mpls**

move-bp-headers: Specifies how to handle headers when there are changes to the message body. If **enabled**, when a message that has multiple parts is reduced into a single body part, the system moves the remaining body part header into the message header.

Default: disabled
Values: enabled | disabled
Example: **set move-bp-headers enabled**

call-monitoring

Provides third-party conferencing, allowing a third-party participant, such as an emergency service endpoint, to be added to a call in progress. The third-party endpoint may or may not be registered with the ME. Note that this feature differs from the **monitor-group** three-way calling feature. When using monitor groups, calls can only be listened to. Through this object a third-party can join a conversation.

Syntax

```
config vsp default-session-config media call-monitoring
config vsp policies session-policies policy name rule name session-config media
call-monitoring
config vsp dial-plan dial-prefix entryName session-config media call-monitoring
config vsp dial-plan route name session-config media call-monitoring
config vsp dial-plan source-route name session-config media call-monitoring
config vsp session-config-pool entry name media call-monitoring
```

Properties

admin: Specifies whether the system initiates conference calling. When **enabled**, and there is a session configuration match, the system makes a call to the specified third party. Upon answering, the third party is conferenced into the active call. You must also enable the **media** anchor property to use this feature.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

monitor-uri: Specifies the third party endpoint that is to be conferenced in to the active call. Note that only one call monitoring endpoint is supported per session configuration.

Default: There is no default setting
Example: **set monitor-uri http://localEmerg.Services.net/monitor0**

calling-group-settings

Specifies how matching calls are mapped to calling groups and the group reference from which they pick up their provisioning. See Configuring Calling Group Objects for complete information on calling groups.

Syntax

```
config vsp default-session-config calling-group-settings
config vsp policies session-policies policy name rule name session-config
calling-group-settings
config vsp dial-plan dial-prefix entryName session-config calling-group-settings
config vsp dial-plan route name session-config calling-group-settings
config vsp dial-plan source-route name session-config calling-group-settings
config vsp session-config-pool entry name calling-group-settings
```

Properties

type: Sets the calling-group association.

Default: There is no default setting

- Values: dynamic: Creates a calling group, and names it based on the AOR of the device. The new group then inherits the settings of the group referenced in the **calling-group** property.
- configured: Assigns the matching AOR to the group referenced with the **calling-group** property. This assignment overrides any other calling group assignment (e.g., **registration-plan > route** match).

Example: **set type dynamic**

calling-group: Specifies the referenced group for calling group parameters. If **type** is set to configured, it is the group that the AOR joins. If set to dynamic, it is the group from which the new groups settings are inherited.

Default: There is no default setting
Example: **set calling-group "vsp calling-groups groupEast"**

codec

Specifies the list of either additional codecs to allow and/or verify beyond the ME internal list, or allows overriding the parameters of the ME internal codecs. The ME codec name list is pre-populated with internal codecs by their official Session Description Protocol (SDP) tag. Enter a predefined SDP tag to open the **codec** object. (Type a question mark at the command line to see the list of predefined tags.) Or, to add a new codec, specify its SDP tag.

Syntax

```
config vsp media-verify-config rtp codec sdpTag
```

Properties

payload-type: Specifies how the ME calculates the RTP payload.

Default: automatic

- **Values: automatic**: The system determines the payload based on the SDP rtpmap attribute, indicating the payload-type of the specified codec tag. For nearly all cases, this should be left as automatic
- **manual<***hex-value***>**: Use this if the codec is always known to be constant or if the SIP client does not correctly use the rtpmap attribute. Specify the hexadecimal value that specifies the media payload type.
- **none**: N/A.

Example: **set payload-type none**

packet-size: Specifies how the ME calculates RTP packet size.

Default: automatic; if set to manual, the default packet size is 100 bytes and cannot be changed

- **Values: automatic**: The system determines the packet size based on the codec tag. For nearly all cases, this should be left as automatic as codecs use RFCs and other SDP parameters (number of channels, bit rate, sampling frequency, etc.) to determine the correct minimum and maximum sizes.
- **manual<***min-size***><***max-size***>**: Use this if a client does not specify a parameter correctly, or this is not an internal codec. Specify the minimum and maximum sizes in bytes.
- **none**: Disables packet-size checking.

Example: **set packet-size manual 100 100**

packet-rate: Specifies how the ME calculates the maximum RTP packet rate.

Default: automatic; if set to manual, the default size is 100 packets per second

- **Values: automatic**: The system determines the packet rate based on the codec tag. For nearly all cases, this should be left as automatic as codecs use RFCs and other SDP parameters (number of channels, bit rate, sampling frequency, etc.) to determine the correct maximum rate.
- **manual <max-size>**: Use this if a client does not specify a parameter correctly, or this is not an internal codec. Specify the maximum rate in packets per second.
- **none**: Disables packet-rate checking.

Example: **set packet-rate manual 50**

codec-parameters

Specifies the CODEC and the parameter to apply to it in the a=fmtp line of the SDP.

Syntax

```
vsp session-config-pool entry codec-specific-parameters codec-parameters
vsp default-session-config entry codec-specific-parameters codec-parameters
```

Properties

param: Specifies the maximum number of frames per second (FPP) for audio CODECs that the ME advertises. Enter the type of **codec** followed by the **max-fpp**.

Default: There is no default codec. The default max-fpp is 24.

- Values: any
- g722
- g7221
- g723
- g729
- g729a
- gsm
- pcma
- pcmu
- string

Values: Min: 0 / Max: 255
Example: **set param audio g729 35**

codec-payload-type-bindings

The codec-payload-type-bindings configures a binding between a codec name and a payload type. Without any codec-payload-type-bindings configured, the ME uses a default DTMF payload type of 101.

This configuration element is set when you want to change the default DTMF payload type offered by the ME. This property takes precedence over the default of 101. Codec-payload-type-bindings is used when the ME generates its own SDP for outgoing calls. The ME generates its own SDP for features like file-play or when the ME is in a Delayed-Offer/Early-Offer network.

Syntax

```
config vsp default-session-config in-media-normalization
codec-payload-type-bindings
config vsp default-session-config out-media-normalization
codec-payload-type-bindings
config vsp session-config-pool entry <name> in-media-normalization
codec-payload-type-bindings
config vsp session-config-pool entry <name> out-media-normalization
codec-payload-type-bindings
```

Properties

binding: Bind a codec name to a particular payload type.

Default: There is no default setting
Example: **set binding g729**

codec-specific-parameters

Adds an a=fmtp line to SDP. Certain endpoints require CODEC parameters to be specified; this object allows you to add the necessary information. Note that if SDP arrives with the a=fmtp line specified for a CODEC set with this object, the original line remains. Use this object in conjunction with the **sdp-regeneration** object to override the line. First, use **sdp-regeneration** to remove the received a=ftmp line from SDP, then use this object to add the line back in with the desired parameters.

Syntax

```
config vsp default-session-config codec-specific-parameters
config vsp policies session-policies policy name rule name session-config
codec-specific-parameters
config vsp dial-plan dial-prefix entryName session-config
codec-specific-parameters
config vsp dial-plan route name session-config codec-specific-parameters
config vsp dial-plan source-route name session-config codec-specific-parameters
config vsp session-config-pool entry name codec-specific-parameters
```

Properties

codec-parameters <codec><parameter>: Specifies the CODEC and the parameter to apply to it in the a=fmtp line of SDP.

Default: There is no default setting
Example: **set codec-parameters g729 annexb=yes**

contact-uri-settings-3xx-response

Specifies where the ME derives the content of the CONTACT header in 3xx (response/redirect) messages that it receives. The CONTACT headers in a 3xx response tell the recipient of the request (which caused the redirect to be generated) where the request can be sent that will yield a final response. This "alternate location" information is sent back to the UAC via the Contact: header(s) in the 3xx response. Set this object in cases when you want the ME to forward the 3xx message with a CONTACT header containing something other than the ME itself. (Note, however, that in that case, the UA does not return the updated information to the ME device.)

For example, if the user property is set to **to-uri**, the ME replaces the user field of the CONTACT header with data from the user field of the outgoing TO header in the 3xx response.

Syntax

```
config vsp default-session-config contact-uri-settings-3xx-response
config vsp policies session-policies policy name rule name session-config
contact-uri-settings-3xx-response
config vsp dial-plan dial-prefix entryName session-config
contact-uri-settings-3xx-response
config vsp dial-plan route name session-config contact-uri-settings-3xx-response
config vsp dial-plan source-route name session-config
contact-uri-settings-3xx-response
config vsp session-config-pool entry name contact-uri-settings-3xx-response
```

Properties

user: Specifies how to derive the value of the User field (the resource located at host) of the CONTACT header.

Default: contact-uri

- **Values: request-uri**: Uses the value from the incoming REQUEST URI
- **to-uri**: Uses the value from the incoming TO URI
- **from-uri**: Uses the value from the incoming FROM URI
- **next-hop**: Uses the IP address of the next-hop server
- **omit**: Leaves the field blank

- **string**: Writes the specified string to the field

Example: **set user from-uri**

user-prefix: Appends the specified string to the beginning of the User field of the CONTACT header. Use this, for example, if you need to append a "1" to a phone number for an outside call.

Default: There is no default setting
Example: **set user-prefix 1**

host: Specifies how to derive the value of the Host field (the host providing SIP resource) of the CONTACT header.

Default: next-hop-address
- **Values: cxc-address**: Uses the ME IP address as the host
- **public-address**: Uses the public address for a UAC behind a firewall or the UAC address if it is not behind a firewall
- **original-address**: The host field is not modified
- **next-hop-address**: Uses the IP address of the next-hop server. However, this value is typically used only for the
- **string**: Writes the specified string to the field

Example: **set host original-address**

port: Specifies how to derive the value of the Port field (where the request is to be sent) of the CONTACT header.

Default: omit
- **Values: cxc-local-port**: Uses the port number that the system transported the call over
- **original-port**: The port field is not modified
- **omit**: Leaves the field blank
- **string**: Writes the specified string to the Port field of the 3xx CONTACT header

Example: **set port original-port**

transport: Specifies the derivation of the transport type for the Transport field of the CONTACT header.

Default: omit
- **Values: next-hop-transport**: Uses the method used by the next-hop server
- **original-transport**: The transport field is not modified
- **omit**: Leaves the field blank
- **UDP | TCP | TLS**: Sets the transport field to the selected protocol

Example: **set transport original-transport**

add-maddr: When enabled, the ME adds a maddr URI parameter.

Default: disabled
`Values: enabled | disabled`
Example: **set add-maddr enabled**

contact-uri-settings-in-leg

Specifies where the ME derives the content of the CONTACT header from when it forwards a message to a UAC. For example, if the **user** property is set to **to-uri**, the ME replaces the User field of the CONTACT header with data from the User field of the incoming TO header. The inbound leg of the session is the portion from the ME to the call initiator (UAC).

Note that this modification does not apply to REGISTER requests. To make changes to the headers of a REGISTER request, use the properties of the **registration-plan** object.

Syntax

```
config vsp default-session-config contact-uri-settings-in-leg
config vsp policies session-policies policy name rule name session-config
contact-uri-settings-in-leg
config vsp dial-plan dial-prefix entryName session-config
contact-uri-settings-in-leg
config vsp dial-plan route name session-config contact-uri-settings-in-leg
config vsp dial-plan source-route name session-config contact-uri-settings-in-leg
config vsp session-config-pool entry name contact-uri-settings-in-leg
```

Properties

user: Specifies how to derive the value of the User field (the resource located at host) of the CONTACT header.

Default: contact-uri

- **Values: request-uri**: Uses the value from the incoming REQUEST URI
- **to-uri**: Uses the value from the incoming TO URI
- **from-uri**: Uses the value from the incoming FROM URI
- **next-hop**: Uses the IP address of the next-hop server
- **omit**: Leaves the field blank
- **string**: Writes the specified string to the field

Example: **set user from-uri**

host: Specifies how to derive the value of the Host field (the host providing SIP resource) of the CONTACT header.

Default: cxc-address

- **Values: cxc-address**: Uses the ME IP address as the host
- **public-address**: Uses the public address for a UAC behind a firewall or the UAC address if it is not behind a firewall
- **original-address**: The host field is not modified
- **next-hop-address**: Uses the IP address of the next-hop server. However, this value is typically used only with the **contact-uri-settings-3xx-response** object.
- *string*: Writes the specified string to the field.

Example: **set host original-address**

port: Specifies how to derive the value of the Port field (where the request is to be sent) of the CONTACT header.

Default: cxc-local-port

- **Values: cxc-local-port**: Uses the port number that the system transported the call over
- **original-port**: The port field is not modified
- **omit**: Leaves the field blank
- **string**: Writes the specified string to the field

Example: **set port original-port**

transport: Specifies the derivation of the transport type for the Transport field of the CONTACT header.

Default: next-hop-transport

- **Values: next-hop-transport**: Uses the method used by the next-hop server
- **original-transport**: The transport field is not modified
- **omit**: Leaves the field blank
- **UDP | TCP | TLS**: Sets the transport field to the selected protocol

Example: **set transport original-transport**

add-maddr: Specifies whether to include the MAddr parameter in the CONTACT header. If enabled, the system adds its own IP address as the MAddr parameter. If **disabled**, the system replaces the HOST with its IP address.

Default: enabled
Values: enabled | disabled
Example: **set add-maddr disabled**

use-incoming-contact: Determines the basis for creating the CONTACT header in an outbound message. When **enabled**, the system first copies the content of the inbound header to build the outbound header. Using the inbound header content, the system then applies any further changes defined in this **contact-uri-settings-in-leg** object.

Default: disabled
Values: enabled | disabled
Example: **set use-incoming-contact enabled**

from-user-contact-uri: This property is not applicable to the inbound leg of a connection.

Default: disabled
Values: enabled | disabled
Example: **set from-user-contact-uri enabled**

registration-plan-precedence: Sets whether or not the system applies the CONTACT header modifications specified within this object. When set to **true**, the system does not apply changes to messages if a registration-plan is present. When set to **false**, all message types are changed.

Default: true
Values: true | false
Example: **set registration-plan-precedence false**

add-other-params: Specifies whether the system maintains additional parameters in incoming CONTACT headers. When **enabled**, the system allows any additional parameters that were received in the CONTACT header to remain in the new CONTACT header when it rewrites it as a result of matching this session config. Additional (or "other") parameters are those found after the URI. For example, in the header "Contact: <sip:johnD@10.1.1.1:5060 udp>;team," "team" is the other parameter

and remains in the new CONTACT header. When **disabled**, the system removes additional parameters.

Default: disabled
Values: enabled | disabled
Example: **set add-other-params enabled**

always-include-contact: Sets the system verify whether there is a contact header present in each message. When enabled, the system checks to ensure that there is a Contact header present. If there is not, it creates one. The content of the header is derived from the other properties in this object. When **disabled**, the system does not check for the presence of the Contact header.

Default: disabled
Values: enabled | disabled
Example: **set always-include-contact enabled**

contact-uri-settings-out-leg

Specifies where the ME derives the content of the CONTACT header from when it forwards a message to a UAS. For example, if the **user** property is set to **to-uri**, the ME replaces the User field of the CONTACT header with data from the User field of the outgoing TO header. The outbound leg of the session is the portion from the ME to the call responder (UAS).

Note that this modification does not apply to REGISTER requests. To make changes to the headers of a REGISTER request, use the properties of the **registration-plan** object. Also, if you have enabled **enum-apply-request-result-to-contact** in the **dial-plan > normalization** object, you must set this object so that the Contact header is not overwritten on the outbound side. To do so, set **use-incoming-contact** to **enabled**, set host, port, and transport to use their original values, and set user to the Contact URI.

Syntax

```
config vsp default-session-config contact-uri-settings-out-leg
config vsp policies session-policies policy name rule name session-config
contact-uri-settings-out-leg
config vsp dial-plan dial-prefix entryName session-config
contact-uri-settings-out-leg
config vsp dial-plan route name session-config contact-uri-settings-out-leg
config vsp dial-plan source-route name session-config contact-uri-settings-out-leg
config vsp session-config-pool entry name contact-uri-settings-out-leg
```

Properties

user: Specifies how to derive the value of the User field (the resource located at host) of the CONTACT header.

Default: contact-uri

- **Values: request-uri**: Uses the value from the incoming Request URI
- **to-uri**: Uses the value from the incoming To URI
- **from-uri**: Uses the value from the incoming From URI
- **contact-uri**: Uses the value from the incoming Contact URI. Select this value to preserve changes made to the Contact header through the **dial-plan > normalization** object.
- **omit**: Leaves the field blank

- *string*: Writes the specified string to the field

Example: **set user from-uri**

host: Specifies how to derive the value of the Host field (the host providing SIP resource) of the CONTACT header.

Default: cxc-address

- **Values: cxc-address**: Uses the ME IP address as the host
- **public-address**: Uses the public address for a UAC behind a firewall or the UAC address if it is not behind a firewall
- **original-address**: The host field is not modified. Select this value to preserve changes made to the Contact header through the **dial-plan > normalization** object.
- **next-hop-address**: The IP address of the next-hop server. However, this value is typically used only for the **contact-uri-settings-3xx-response** object.
- *string*: Writes the specified string to the field

Example: **set host original-address**

port: Specifies how to derive the value of the Port field (where the request is to be sent) of the CONTACT header.

Default: cxc-local-port

- **Values: cxc-local-port**: Uses the port number that the system transported the call over
- **original-port**: The port field is not modified. Select this value to preserve changes made to the Contact header through the **dial-plan > normalization** object.
- **omit**: Leaves the field blank
- *string*: Writes the specified string to the field

Example: **set port omit**

transport: Specifies the derivation of the transport type for the Transport field of the CONTACT header.

Default: next-hop-transport

- **Values: next-hop-transport**: Uses the method used by the next-hop server
- **original-transport**: The transport field is not modified. Select this value to preserve changes made to the Contact header through the **dial-plan > normalization** object.
- **omit**: Leaves the field blank
- **UDP | TCP | TLS**: Wets the transport field to the selected protocol

Example: **set transport original-transport**

add-maddr: Specifies whether to include the MAddr parameter in the CONTACT header in an outbound message. If **enabled**, the system adds its own IP address as the MAddr parameter. If **disabled**, the system replaces the HOST with its IP address.

Default: enabled
Values: enabled | disabled
Example: **set add-maddr disabled**

use-incoming-contacts: Determines the basis for creating the CONTACT header in an outbound message. When **enabled**, the system first copies the content of the inbound header to build the outbound header. Using the inbound header content, the system then applies any further changes defined in this **contact-uri-settings-out-leg** object. Set

this property to **enabled** to preserve changes made to the Contact header through the **dial-plan > normalization** object.

Default: disabled
Values: enabled | disabled
Example: **set use-incoming-contacts enabled**

from-user-contact-uri: Specifies whether the system uses the location cache to derive the CONTACT header when forwarding a message. When **disabled**, the default, the CONTACT URI is derived from the FROM header of the original message. When this property is **enabled**, the system does a location cache lookup on the received FROM URI. If the system finds an entry, it uses the server-side contact (found in the entry) as the CONTACT URI for the outbound message.

Default: disabled
Values: enabled | disabled
Example: **set from-user-contact-uri enabled**

registration-plan-precedence: Sets whether or not the system applies the CONTACT header modifications specified within this object to REGISTER requests. When set to **true**, the system does not apply changes to REGISTER messages. When set to **false**, all message types are changed.

Default: true
Values: true | false
Example: **set registration-plan-precedence false**

add-other-params: Specifies whether the system maintains additional parameters in incoming CONTACT headers. When **enabled**, the system allows any additional parameters that were received in the CONTACT header to remain in the new CONTACT header when it rewrites it as a result of matching this session config. Additional (or "other") parameters are those found after the URI. For example, in the header "Contact: <sip:johnD@10.1.1.1:5060 udp>;team," "team" is the other parameter and remains in the new CONTACT header. When **disabled**, the system removes additional parameters.

Default: disabled
Values: enabled | disabled
Example: **set add-other-params enabled**

always-include-contact-header: Sets the system verify whether there is a contact header present in each message. When enabled, the system checks to ensure that there is a Contact header present. If there is not, it creates one. The content of the header is derived from the other properties in this object. When **disabled**, the system does not check for the presence of the Contact header.

Default: disabled
Values: enabled | disabled
Example: **set always-include-contact-header enabled**

csta-settings

Provides CSTA-to-OCI, -OCS, or - communications for enterprises using a BroadWorks, Cisco, or Avaya call manager and Microsoft OCS. The ME supports third-party call control (3PCC) for any phones connected to the 3PCC server. See Configuring Third-Party Call Control server objects for a complete description of 3PCC and information on configuring 3PCC servers.

Note that to make this application work in addition to the ME configuration, you must also point the CSTA SIP traffic at the ME so that it acts as a CSTA gateway. See the *Oracle Communications Session Services Configuration Guide* for more information.

Identifying the Active Device

Every device is mapped to a unique terminal ID. When a device logs into MOC, the ME records the terminal ID. It is not uncommon for phone address to be mapped to multiple devices, and therefore, associated with multiple IDs. Because the ME always selects the active device when initiating a session, there must be some logic configured to identify the active device so that only that device is used when an outbound call is made through MOC. This is useful, for example, when a home and work phone are mapped to the same URI. The **terminal-select-dial** property (in this object) configures the ME to call a specified number (and play a recorded file). When a user picks up in response to the call, the device used to answer is noted as the active terminal and calls to that URI are forwarded to that active device. Note that after an ID has been established through this configuration, if a different device answers a call, the ME uses the new device for the current call and then resumes use of the established device for future calls. (To change the active terminal setting, use the **jtapi-control** action.)

Using Partitions and Calling Search Spaces

The ME allows you to use the multiple partition feature of Cisco CallManager. A single phone with a single phone number (DN) can be mapped to two or more partitions, where each partition can map to a individual line on the phone. The primary partition will indicate the line to use for outbound calls, typically this will be line 1. See the Cisco online documentation, *Partitions and Calling Search Spaces*, for complete information on Cisco partitions.

Syntax

```
config vsp default-session-config csta-settings
config vsp policies session-policies policy name rule name session-config
csta-settings
config vsp dial-plan dial-prefix entryName session-config csta-settings
config vsp dial-plan route name session-config csta-settings
config vsp dial-plan source-route name session-config csta-settings
config vsp session-config-pool entry name csta-settings
```

Properties

mode: Specifies the 3PCC server type that the ME is connecting to the Microsoft OCS application. By selecting a server type, the ME acts as a translation device, converting CSTA traffic from that server type to a format the 3PCC server can recognize. Enter the type and a reference to the configured 3PCC server.

Default: none

- **Values: none <***serverReference***>**: The ME Engine does not provide 3PCC services.
- **internal <***serverReference***>**: The system acts as the PBX, resulting in phones registering with the
- ME Session Controller Media Engine
- device. This mode only works with phones registered directly to the ME device.
- **broadworks <***serverReference***>**:
- The ME converts CSTA traffic to either OCI or OCS traffic, depending on the **type** property setting of the referenced BroadWorks server.
- **cisco <***serverReference***>**:
- The ME converts CSTA traffic to for processing by the referenced Cisco server.
- **avaya <***serverReference***>**:

- The ME converts CSTA traffic to for processing by the referenced Avaya server.
- **loopback <***serverReference***>**: The ME creates a loopback session to the OCS for testing.

Example: **set mode broadworks "vsp enterprise 3pcc-servers broadworks-csta-server BWocs"**

terminal-select-dial: See Identifying the Active Divide for information on the use of this property. In the example below, the system looks changes calls in the form of tel:+1508xxxyyyy to 1508xxxyyyy@callme.com. The result is the number the system dials to play the file. The entry in the From URI field is displayed as the caller ID.

Default: any
Values: any | disabled | once-at-login <*toURIexp*><*toURIreplace*><*fromURIfile*> | action-driven <*toURIexp*><*toURIreplace*><*fromURIfile*>
Example: **set terminal-select-dial once-at-login ^tel:(\+)?((1?508)[0-9]{7}).*$\2callme.com gday.wav**

lcs-transport: Sets the transport protocol used to communicate with the third-party server. For a secure connection and to support CSTA failover operations, set transport to TLS and include a reference to a certificate on the system.

Default: any
Values: any | UDP | TCP | TLS <*certificateReference*>
Example: **set lcs-transport tls "vsp tls certificate nnos-e.abc.com**

default-partition: Sets which partition this session configuration applies to (controls). This feature only applies to Cisco CallManager partitions; see *Partitions and Calling Search Spaces* in the Cisco online documentation for more information on partitions.

Note that you can also set the default partition using the **set-default-partition** option of the **jtapi-control** action. The action setting overrides the values set with this property.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: automatic
- **Values: automatic**: Use the first address from the list retrieved from the switch.
- **specified <***value***>**: Use the partition that you specify.
- **derived <***regEx***><***replacement***>**: Use the partition name found in the Active Directory number, and derived by a regular expression rule. The AD number is in the form **tel:+number:partition**.

Example: **set default-partition derived "(?ms).*<RequestSystemStatus.*tel:**

+[0-9]*;(.*)PT""\1PT"

custom-event-fields

Adds a custom data field to the callCreated, callConnected, and callTerminated events. This object defines the content of the field.

Syntax

config vsp session-config-pool entry third-party-call-control custom-event-fields

config vsp default-session-config third-party-call-control custom-event-fields

Properties

named-variable-entry: Specifies the content of the custom data field added to the accounting record. This is in the form variable=value.

Default: There is no default setting
Example: **set named-variable-entry variable=\s**

custom-events-grouping-string: *Secondary property*. The characters used to associate an event's variable and values.

Default: =
Example: **set custom-events-grouping-string :**

custom-event-delimiter: *Secondary property.* The characters used to separate group custom event entries.

Default: ;
Example: **set custom-event-delimiter **

dns-client-settings

Configures the DNS client process and how the client communicates with the DNS service (resolver). See DNS Service Resolver and Server Objects for information on the ME DNS service.

Syntax

```
config vsp default-session-config dns-client-settings
config vsp policies session-policies policy name rule name session-config
dns-client-settings
config vsp dial-plan dial-prefix entryName session-config dns-client-settings
config vsp dial-plan route name session-config dns-client-settings
config vsp dial-plan source-route name session-config dns-client-settings
config vsp session-config-pool entry name dns-client-settings
```

Properties

admin: Specifies whether this DNS client configuration entry is applied to calls matching the session configuration.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

client-timeout: Specifies how long in milliseconds the client process waits for the DNS service (resolver) to respond. If this timer expires, the service continues to look for the entry, and on finding it, writes it to its own and to the client cache. When the client next queries for that same address, the response will come from the cache.

Default: 2000
Example: **set client-timeout 3000**

server-names: Specifies which server(s) the DNS service should use to resolve requests that originate from calls matching this session configuration. Enter a reference to a server name that you configured with the resolver **server** object. If you do not configure any server-names with this property, all configured servers are used.

Default: There is no default setting
Example: **set server-names "vsp dns resolver server name dns1"**

routing-last-resort-dns: Specifies whether the system should do a DNS lookup when it cannot determine where to forward a call based on the dial plan, registration plan, location cache, or policy. If **enabled**, the system does a DNS lookup on servers configured with the DNS resolver **server** object. If **disabled**, the system does not do a DNS lookup and returns, by default, a "404 not found" message to the caller. You can change the response code and string using the **sip-settings dns-fail-response-code** and **-string** properties.

Default: disabled
Values: enabled | disabled
Example: **set routing-last-resort-dns enabled**

add-nnos-domain: *Secondary property.* Specifies whether to add the configured domain to a single-label query. If disabled, no domain is added and the setting in the use-cxc-domain-in-search and additional-search-domains properties of the **resolver** object are applied. If enabled, the ME appends the name set with the domain-name property of the **static-stack-settings** object to a single-label query, making it a FQDN. The other **resolver** settings do not apply.

Default: disabled
Values: enabled | disabled
Example: **set add-nnos-domain enabled**

routing-lookup-type: *Secondary property.* Sets the method of server location. This property is applicable when the **routing-last-resort-dns** property is enabled.

Default: NAPTR+SRV+A

- **Values: NAPTR+SRV+A**: A NAPTR lookup, an SRV lookup on the information returned, and an A lookup from those results. It also does a lookup on the original domain name.
- **NAPTR+SRV**: A NAPTR lookup, an SRV lookup on the information returned, and an A lookup from those results. There is no lookup on the original domain name.
- **SRV+A**: An SRV lookup, followed by an A lookup on names returned by SRV. Finally, it does a lookup on the original domain name.
- **SRV**: An SRV lookup, followed by an A lookup on names returned by SRV.
- **A**: A lookup on the original domain name.

Example: **set routing-lookup-type SRV**

ras-settings: Sets the configuration for scenarios when the ME is communicating with an external H.323 gatekeeper. (This property is only applicable if the **server-type** property is set to **h323-gatekeeper**.) When the ME registers on behalf of a client, these settings allow the systems to exchange registration, admission, and status (RAS) messages.

Default: The default settings are indicated in each field description

- **Values: registration TTL**: Sets the frequency, in seconds, of the system putting forward reregistrations. The default is **3600** seconds
- **registration retries**: Sets the number of attempts the system makes to register a client. before abandoning the request. A value of zero allows unlimited retries. The default setting is **5** retries.
- **endpoint alias**: Assigns a string to identify the system to the gatekeeper. This string should be configured on the gatekeeper as well so that it can recognize calls from the ME.

- **supported prefix**: Sets the value for gatekeepers that need digits prepended to a number. This requirement would be set in the gatekeeper dial plan. Usually, this field is left blank.
- **reregister on UNREGISTER**: Sets whether the system tries to reregister a client after having received an UNREGISTER from the gatekeeper. If this is enabled, the ME tries to reregister the client up to the number f times specified with the registration-retires field. The default setting is **disabled**.
- **gatekeeper call routing**: Sets whether the system provides support for gatekeeper-routed calls. When true, the ME does provide that support. The default setting is **false**.

Example: **set ras-settings 5200 8 nnos-e-1 gk1 1! enabled true**

emergency-settings

Sets whether matching calls should be handled without limitation. When this object is administratively enabled, matching calls are not subject to emission and admission controls.

Syntax

```
config vsp default-session-config emergency-settings
config vsp policies session-policies policy name rule name session-config
emergency-settings
config vsp dial-plan dial-prefix entryName session-config emergency-settings
config vsp dial-plan route name session-config emergency-settings
config vsp dial-plan source-route name session-config emergency-settings
config vsp session-config-pool entry name emergency-settings
```

Properties

admin: Specifies whether the emergency setting should be active or inactive. When **enabled**, a matching call is handled without limitation (not subject to emission and admission controls).

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

endpoint-management

Sets the 3GPP policy configuration to apply to SIP sessions.

Syntax

```
config vsp default-session-config endpoint-management
config vsp session-config-pool entry name endpoint-management
```

Properties

admin: Enables or disables endpoint management.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

park-incoming-calls: Enable or disable automatically parking incoming calls on the ME.

Default: disabled
Values: enabled | disabled
Example: **set park-incoming-calls enabled**

park-call-greeting: Specify the audio file to be played when an incoming call is parked.

Default: There is no default setting
Example: **set park-call-greeting greet1.wav**

terminate-after-greeting: Enables or disables the ME terminating a call after parking it and playing a greeting.

Default: enabled
Values: enabled | disabled
Example: **set terminate-after-greeting disabled**

event-settings

Configures call control events as well as named-variables and user-specified channels.

Syntax

```
config vsp session-config-pool entry name event-settings
config vsp default-session-config event-settings
```

Properties

call-control-events: Enables or disables call control events.

Default: enabled
Values: enabled | disabled
Example: **set call-control-events disabled**

media-control-events: Enables or disables media control events.

Default: enabled
Values: enabled | disabled
Example: **set media-control-events disabled**

channel: Specifies channels to be used for events generated for this session. Named variables can be added into channels by using the % character to delimit the name of the variable to expand.

Default: call
Example: **set channel %$event.requestID%**

allow-event-group-events: *Secondary property*. Enables or disables the ME to send events via the legacy **event-group** object.

Default: enabled
Values: enabled | disabled
Example: **set allow-event-group-events disabled**

inbound-sip-messages: *Secondary property*. Configures events for incoming SIP messages.

outbound-sip-messages: *Secondary property*. Configures events for outgoing SIP messages.

MESSAGE-events: *Secondary property*. Enables or disables the MESSAGEevents group. This event group exists for backwards compatibility only. The SipMessageEvent class should be used instead.

Default: disabled
Values: enabled | disabled
Example: **set MESSAGE-events enabled**

named-variable-entries: Inserts named-variables into events. Before selecting named-variables to enter into events, you must configure them in either the **session-config > named-variables** object or via the **named-variables-add** action.

Default: There is no default setting
Example: **set named-variable-entries my-variable my-variable-name**

allow-empty-value: *Secondary property*. When enabled, the ME allows you to create named-value-pair entries in events without any corresponding values. The ME then looks in the configured named variable table to find a value to include in the event. If the value is empty and this property is enabled, the ME writes out the variable in the event but leaves it blank.

Default: enabled
Values: enabled | disabled
Example: **set allow-empty-value disabled**

event-filter: Configures a filter which allows you to specify the event classes that this session is allowed to emit.

Default: There is no default setting
Example: **set event-filter call-create**

event-filter: Configures a filter which allows you to specify the event classes that this session is allowed to emit.

Default: There is no default setting.
Example: **set event-filter call-control**

file-transfer

Enables recording of file transfers on the ME.

Syntax

```
config vsp default-session-config file-transfer
config vsp policies session-policies policy name rule name session-config
file-transfer
config vsp dial-plan dial-prefix entryName session-config file-transfer
config vsp dial-plan route name session-config file-transfer
config vsp dial-plan source-route name session-config file-transfer
config vsp session-config-pool entry name file-transfer
```

Properties

anchor: Enables or disables anchoring, which defines whether the system is used as an intermediary for traffic. When **enabled**, all file transfers pass through the ME device. If **disabled**, transfers circumvent the ME device. (The system would still have a record of the transfer, however, because it keeps records of all SIP transactions.) You must enable anchoring to use the recording or virus scanning features.

Default: disabled
Values: enabled | disabled

Example: **set anchor enabled**

record: Enables or disables recording of file transfers on the system in this SIP call session. The file is stored on the system and forwarded to the SIP call recipient. Anchoring must be enabled to use the recording feature.

Default: disabled
Values: enabled | disabled
Example: **set record enabled**

max-filesize: Configures the maximum file size, in bytes, allowed in a transfer through the ME device. If the size limit is exceeded, the file is dropped.

Default: 1073741824
Values: Min: 1 / Max: 1073741824
Example: **set max-filesize 50000000**

allow-non-default-ports: Specifies whether or not the system is limited in choice of ports when anchoring a file transfer. When **enabled**, the default, the system can use any port. Leave this setting for interoperability with LCS 2005. When **disabled**, the system can only use the default port that is specified by Microsoft.

Default: enabled
Values: enabled | disabled
Example: **set allow-non-default-ports disabled**

t120-anchor: Enables anchoring of the application sharing and whiteboard features found in the Office Communicator 2005 client (based on ITU T.120). If you **enable** this feature, you must also enable the anchor property of the **media** object. If you are anchoring federated traffic, you must also set the **sip-settings > lcs-compatibility** bit to compensate for a bug in the OC 2005 client that does not accept Content-Type headers with the subtype "SDP" specified in uppercase. Set the bit to 0x010032e3.

Default: disabled
Values: enabled | disabled
Example: **set t120-anchor enabled**

forking-settings

Configures the ring pattern for SIP calls from the ME to the endpoint. The ME selects ring destinations based on information in the address of record (AOR) from the locations services database.

You can also configure an ordered set of rules (metrics) to influence the final server (or next hop) selection for outbound calls with the **outbound-arbiter-rule** property. This may be necessary if you have configured multiple possible next-hop devices using the **peer** object. The arbitration calculation rules apply to all outbound devices to determine the destination. This set of rules takes precedence over any arbitration decision made through the dial-plan **arbiter** configuration. If a dial-plan references a session-config in which this object **outbound-arbiter-rule** is configured, the ME ignores the dial-plan **arbiter** configuration and uses this one instead.

Syntax

```
config vsp default-session-config forking-settings
config vsp policies session-policies policy name rule name session-config
forking-settings
config vsp dial-plan dial-prefix entryName session-config forking-settings
config vsp dial-plan route name session-config forking-settings
config vsp dial-plan source-route name session-config forking-settings
config vsp session-config-pool entry name forking-settings
```

Properties

forking-type: Sets the ME forwarding behavior when it receives a call INVITE. This is the method in which the system forwards the call to the endpoint. In order for this property to work properly, the **outbound-arbiter-rule** parameter must be configured.

Default: none

- **Values: none**: The system forwards the call to the latest binding it finds for the Request URI in the location service.
- **sequential**: The call is forwarded to each binding for each AOR stored for a destination, one at a time, until it is successful. If the system receives a fail message, busy, or timeout, it tries the next AOR. The delay between trying each device is set with the **session-provisional-timeout** property of the **sip-settings** object.
- **parallel**: The call is forwarded to each binding for all AORs for the destination.
- redirect: The redirect call statistics are used as inputs in the algorithm. This must be the configured forking-type in order for the 302 redirect feature to work.

Example: **set forking-type parallel**

max-hunt: Specifies the number of destinations to try when the system is configured to do sequential forking. Setting this value prevents the creation of a forking loop in the event that a server redirects a call to another server in the listed destinations.

Default: 50
Example: **set max-hunt 50**

outbound-arbiter-rule: Enters rules into the arbiter configuration. Enter as many rules as you wish. If you do not set any rules, the system uses the settings of the **dial-plan** arbiter (or factory defaults if the dial-plan also has no arbiter configuration). If you select least-cost, you can optionally set a maximum (or unlimited) value for call cost. It you select trunk-qos, you can optionally select a previously configured **class-of-service**. This property is required for the **forking-settings** properties to work properly.

See the Routing algorithm options table for a description of the selections.

Default: There is no default setting
Example: **set outbound-arbiter-rule least-cost 15**

max-arbitration-options: *Secondary property*. Specifies the number of potential destinations to consider when applying a rule. The smaller of this and **max-hunt** takes effect when the final destinations are determined.

Default: unlimited
Values: Min: 0 / Max: 7294967295
Example: **set max-arbitration-options 50000**

from-uri-specification

Specifies what derives the content of the fields of the FROM URI when the ME transmits a message. For example, if the **user** property is set to **from-uri**, the ME replaces the user field of the FROM URI with data from the user field of the incoming FROM URI. If set to **omit**, the user field is left blank. Or, you can enter any string that you want placed in the user field. See Altering URIs for information on replacement options.

Syntax

```
config vsp default-session-config from-uri-specification
config vsp policies session-policies policy name rule name session-config
from-uri-specification
config vsp dial-plan dial-prefix entryName session-config from-uri-specification
config vsp dial-plan route name session-config from-uri-specification
config vsp dial-plan source-route name session-config from-uri-specification
config vsp session-config-pool entry name from-uri-specification
```

Properties

user: Specifies how to derive the value of the user field of the FROM URI.

Default: from-uri
Values: request-uri | to-uri | from-uri | omit | next-hop | local | omit-phone-context | *string*
Example: **set user request-uri**

host: Specifies how to derive the value of the host field of the FROM URI.

Default: from-uri
Values: request-uri | to-uri | from-uri | omit | next-hop | next-hop-domain | local-ip | *string*
Example: **set host omit**

port: Specifies how to derive the value of the port field of the FROM URI.

Default: from-uri
Values: request-uri | to-uri | from-uri | omit | *string*
Example: **set port omit**

display: Specifies how to derive the value of the display field of the FROM URI.

Default: from-uri
Values: request-uri | to-uri | from-uri | omit | next-hop | *string*
Example: **set display next-hop**

user-agent-aware-display-translation: Specifies whether or not to apply the **displayname-character-set-info** mappings to matching calls. When **enabled**, the system checks the display name for accented characters. If found, it does a location cache lookup on the destination. If that is successful, it checks the user agent found in the location cache against the configured mappings and performs any necessary translations.

Default: disabled
Values: enabled | disabled
Example: **set user-agent-aware-display-translation enabled**

transport: Specifies the value of the transport field of the FROM URI. In addition to using the value from other fields of the incoming URI, you can set the transport method to UDP, TCP, or TLS.

You cannot enter a string for this property.

Default: from-uri
Values: request-uri | to-uri | from-uri | omit | UDP | TCP | TLS | *string*
Example: **set transport omit**

use-param: Specifies whether the User parameter in the FROM URI of the SIP header is maintained or removed when the system forwards a message. If set to **keep**, the message is forwarded with the parameter as it was received. If set to **omit**, the entire user=*param* is removed from the TO URI.

Default: omit

Values: omit | keep
Example: **set use-param keep**

user-truncate-non-digits: Specifies whether to remove non-digits from the User portion of the FROM URI in INVITE messages. When **enabled**, the system removes all non-digits.

Default: disabled
Values: enabled | disabled
Example: **set user-truncate-non-digits enabled**

uri-parameter <name><value>: Appends the specified user parameter and value to the FROM URI for matching calls. For example, the example below would result in a FROM URI that looked similar to:

<sip:spot@fun.com;BTG=trunk1>

You can control how and when the new parameter is added using the append options. Select **append-always** to have the parameter added to all matching calls. Select **append-if-does-not-exist** to add the name and value only if it does not already exist in the URI, preventing the possibility of duplicate parameters. Select **overwrite-existing** to replace any existing parameter in the FROM URI with the configured name and value, updating instead of appending to the parameter. You can append multiple user parameters.

Default: There is no default setting
Example: **set uri-parameter BTG trunk1**

header-parameter: Adds a parameter string to the SIP header (outside of the SIP URI). Use this string, for example, to identify the source of a call or group destinations. You can add as many header parameters as required. Use the format name=value. To add multiple parameters use the format name=value;name=value...

Default: There is no default setting
Example: **set header-parameter OLI=70**

add-oli-tag: Specifies the number of digits to copy from the User portion of the From header to create an Originating Line Information (OLI) tag. The OLI tag provides information on the class of service available for the call. A value of 0 disables adding the tag.

Default: 0
Example: **set add-oli-tag 2**

copy-charge-uri-user: Specifies whether to copy the User portion of the Charge URI to the User portion of the From URI. When enabled, the system changes the content of the User portion, for example, for billing purposes.

Default: disabled
Values: enabled | disabled
Example: **set copy-charge-uri-user enabled**

strip-digits: Specifies the number of digits to strip from the User portion of the From URI. Use this, for example, if the original INVITE contains extra digits that would be problematic to the downstream server. Digits are removed beginning at the string that immediately follows the sip: or sips: portion.

Default: 0
Example: **set strip-digits 2**

prepend-digits: Specifies a string to prepend to the User portion of the From URI. The string can be comprised of up to 128 ASCII characters.

Default: There is no default setting

Example: **set prepend-digits 011**

group-settings

Creates a tag (group name) that can be stored in the location cache during registration. When the ME receives a REGISTER and applies a matching session configuration, it saves out the zero or more groups (configured with this object) that are associated with that session config. When the ME receives an INVITE destined for a registered phone, a location cache lookup results in return of all the stored group names, which can then be used to further refine the selection of the applicable session config for the INVITE. You can use this feature, for example, to control outbound settings that are specific to a type of phone, such as the encryption type or CODEC preferences.

Syntax

```
config vsp default-session-config group-settings
config vsp policies session-policies policy name rule name session-config
group-settings
config vsp dial-plan dial-prefix entryName session-config group-settings
config vsp dial-plan route name session-config group-settings
config vsp dial-plan source-route name session-config group-settings
config vsp session-config-pool entry name group-settings
```

Properties

group-name: Specifies the name of the group. The system transfers each group name in a matching session config to the location cache to be stored with the registration.

Default: There is no default setting
Example: **set group-name eyebeam_group**

h225-settings

The H.225 settings configuration object allows you to configure H.225 on the ME.

Syntax

```
config vsp session-config-pool entry h225-settings
config vsp default-session-config h224-settings
```

Properties

false-start: If enabled, the ME accepts inbound H.323 fast start calls and includes fast start in SETUP messages for outbound H.323 calls. The calls fall back to slow if fast start is unsuccessful.

Default: enabled
Values: enabled | disabled
Example: **set false-start disabled**

manual-ringback: If enabled, the ME prohibits remote ringback. When this property is disabled, SIP to H.323 calls attempt to open an audio channel for remote ringback.

Default: enabled
Values: enabled | disabled
Example: **set manual-ringback disabled**

use-inbound-call-settings: When enabled for an H.323 to H.323 call, the ME uses inbound H.323 call settings for H.323 outbound calls.

Default: disabled
Values: enabled | disabled
Example: **set use-inbound-call-settings enabled**

fwd-progress-as-alerting: When enabled, the ME sends an Alerting message instead of a Progress message.

Default: disabled
Values: enabled | disabled
Example: **set fwd-progress-as-alerting enabled**

default-terminal-type: Identifies the ME terminal type for MSD.

Default: 60
Values: Min: 0 / Max: 4294967295
Example: **set default-terminal-type 75**

multiple-calls: When enabled, the ME allows calls to share an H.225 connection.

Default: disabled
Values: enabled | disabled
Example: **set multiple-calls enabled**

maintain-connection: When enabled, the ME keeps an H.225 connection open after calls are cleared.

Default: disabled
Values: enabled | disabled
Example: **set maintain-connection enabled**

conn-idle-timeout: Specifies the maximum lifetime of an idle H.225 connection. A value of 0 indicates an idle connection should never timeout.

Default: 3600
Values: Min: 300 / Max: 65535
Example: **set conn-idle-timeout 2500**

h323-user-alias: Specifies the source and destination address type in Setup, Alerting, Connect, ARQ, and LRQ messages.

Default: none
- none
- dialedDigits
- h323ID
- urlID
- emailID

Example: **set h323-user-alias urlID**

call-alerting-timeout: The maximum number in seconds the ME waits for Alerting message after sending a SETUP. The call clears if this timeout is reached.

Default: 4
Values: Min: 0 / Max: 4294967295
Example: **set call-alerting-timeout 500**

call-establishment-timeout: The maximum number in seconds the ME waits for an H.323 call to be established. The call clears if this timeout is reached.

Default: 60
Values: Min: 0 / Max: 4294967295

Example: **set call-establishment-timeout 75**

end-session-timeout: The maximum number of seconds the ME waits after sending a ReleaseComplete before call resources are reclaimed.

Default: 15
Values: Min: 0 / Max: 4294967295
Example: **set end-session-timeout 30**

h245-establish-timeout: The maximum number, in seconds, the ME waits for an H245 connection to be established. The call clears if this timeout is reached.

Default: 1
Values: Min: 0 / Max: 4294967295
Example: **set h245-establish-timeout 5**

reinvite-type: *Secondary property.* Indicates if the ME should use Terminal Capability Set or Extended Fast Connect messages to reconfigure media channels.

Default: emptyTermCapSet
Values: emptyTermCapSet | extendedFastConnect
Example: **set reinvite-type extendedFastConnect**

use-progress-inband: When enabled, inband ring information from the inbound H.323 call-leg is propagated to the outbound call-leg.

Default: enabled
Values: enabled | disabled
Example: **set use-progress-inband disabled**

fwd-retrieve-no-tx: When true, the ME does not pause remotetransmitted if media information is 0.0.0.0.

Default: true
Values: true | false
Example: **set fwd-retrieve-no-tx false**

use-server-connection: *Secondary property.* specifies whether the ME creates a new, or uses an existing, TCP connection. If true, the ME uses a TCP connection created by the remote gateway instead of creating a new outbound TCP connection. Use this property for a remote H.323 gateway using connection sharing for its H.225 traffic. (It uses a single TCP connection for multiple calls.)

Default: true
Values: true | false
Example: **set use-server-connection false**

enum-lookup-called-party: When enabled, the ME performs an ENUM lookup of the called number before making an outbound H.323 call.

Default: disabled
Values: enabled | disabled
Example: **set enum-lookup-called-party enabled**

enum-domain: The domain used for ENUM lookups.

Default: 164.arpa
Example: **set enum-domain 12025551234**

enum-returnednaptr-replace: *Secondary property*. Enter a regexp and a replacement value. When configured, if the ME performs an ENUM dip for an inbound H.323 call, the regexp and replacement string are applied to the result of the ENUM lookup. That then becomes the called party identifier.

Default: There is no default setting
Example: **set enum-returnednaptr-replace (*)\?**

session-duration-max: Sets the maximum duration of an H.323 call, in seconds. A value of **0** (the default) indicates there is no maximum lifetime.

Default: 0
Values: Min: 0 / Max: 1000000
Example: **set session-duration-max 1000**

h245-settings

The H.245 settings configuration object allows you to configure H.245 on the ME.

Syntax

```
config vsp session-config-pool entry h245-settings
config vsp default-esssion-config h245-settings
```

Properties

h245-tunnel: When enabled, the ME attempts to use an H.225.0 connection for H.245 traffic. The use of H.245 tunneling depends on indication from both H.323 terminals and gateways.

Default: enabled
Values: enabled | disabled
Example: **set h245-tunnel disabled**

early-h245: The ME does not support early H.245.

Default: notunnel
- Values: notunnel: The ME ignores the early H.245 and completes the call setup using slowstart.
- reject: The ME rejects the call.

Example: **set early-h245 reject**

wait-for-remote-tcs: When true, the ME waits to receive a Terminal Capability Set message before advertising its capabilities. When false, the ME issues a TCS message after a slowstart call is connected.

Default: true
Values: true | false
Example: **set wait-for-remote-tcs false**

clc-when-pausing-remote: *Secondary property.* When true, the ME closes its TX channels when pausing the remote H.323 terminal.

Default: false
Values: true | false
Example: **set clc-when-pausing-remote true**

send-msd-when-unpausing-remote: *Secondary property.* Specifies whether the ME conducts MSD when using TCS to unpause a remote H.323 gateway.

Default: false
Values: true | false
Example: **set send-msd-when-unpausing-remote true**

use-h450-hold-retrieve: When enabled, the ME uses H.450 supplemental service PDUs for holds and retrieves.

Default: enabled
Values: enabled | disabled

Example: **set use-h450-hold-retrieve disabled**

sip-h323-dtmf-translate <sip-dtmf-type><h323-dtmf-type>: Sets preferences for H.323-SIP DTMF interworking for a particular H.323 trunk.

Default: inband
Example: **set sip-h323-dtmf-translate RFC2833 H245SIGNAL**

codec-selection: *Secondary property.* Indicates how the ME chooses converged codecs.

Default: remote
- Values: -none: No codec is being used.
- -local: Use the highest preference common codec seen in SIP SDP.
- -remote: Use the highest preference common codec in remote TCS.
- -followMSD: Use the result of MDT to decide.

Example: **set codec-selection local**

map-ptime-to-fpp: *Secondary property.* When set to **true**, the ME uses SDP ptime parameter to set max-frames-per-packet codec value in Terminal Capability Set. Ptime and FPP are not equivalent, however, this allows compatibility in some IW scenarios.

Default: false
Values: true | false
Example: **set map-ptime-to-fpp true**

map-fpp-to-ptime: *Secondary property.* When true, the ME uses max-frames-per-packet codec value in Terminal Capability Set to set SDP ptime parameter. Ptime and FPP are not equivalent, however, this allows compatibility in some interworking scenarios.

Default: false
Values: true | false
Example: **set map-fpp-to-ptime true**

add-equivalent-codecs: *Secondary property.* When true, the ME adds equivalent codecs to Terminal Capability Set. The currently supported case is G729 present in SDP which would add both G729 and G729A in TCS.

Default: false
Values: true | false
Example: **set add equivalent-codecs true**

h323-to-sip-fromheader-spec

Specifies how to generate a SIP From header from an H.323 SETUP message. When The ME receives a message from an H.323 server via the server that contains this configuration object, it creates the From header using the parameters of this object. The From header is made up of four components defined here--scheme:user@host.suffix.

Syntax

```
config vsp session-config-pool entry h323-to-sip-fromheader-spec
config vsp default-session-config h323-to-sip-fromheader-spec
```

Properties

scheme: Specifies the Scheme to use in the From (or To) header.

Default: sip
- Values: sip: Use the SIP scheme

- tel: Use the tel scheme
- omit: Leaves the field blank. Select this if the upstream server uses the correct scheme in the H.323 SETUP message and you do not want that value changed.
- *string*: Enters the specified string in the scheme field

Example: **set scheme tel**

user: Specifies the origin of the User field content to use in the From (or To) header.

Default: calling-number

- Values: calling-number: The value in the originating phone number
- h323-id: The value from the incoming h323ID alias
- url-id: The value from the incoming url ID
- email-id: The value from the incoming email ID
- omit: Leaves the field blank
- *string*: Enters the specified string in the User field

Example: **set user h323-id**

host: Specifies the origin of the Host field content to use in the From (or To) header.

Default: h323gw-domain

- Values: h323-id: The value from the incoming h323ID alias
- url-id: The value from the incoming url ID
- email-id: The value from the incoming email ID
- h323gw-domain: Omits the value configured for the H323 gateway
- omit: Leaves the field blank
- *string*: Enters the specified string in the User field

Example: **set host omit**

suffix: Specifies the suffix to add to the From (or To) header. Enter a suffix or select omit to let the system derive the field from the SETUP message.

Default: omit

Values: omit | *string*

Example: **set suffix omit**

use-anon: When enabled, as the H.323 process builds the SIP From: header for a received H.323 SETUP message it will do the following:

- Use anonymous as the user portion of the URI if after applying h323-to-sip-fromheader-spec config the user portion is empty
- Use the IP address of the H.323 gateway which transmitted the SETUP as the host portion of the URI if, after applying h323-to-sip-fromheader-spec config, the host portion is empty

This guarantees a valid From: header URI will exist when sent to the SIP process. When set to false there is some chance an incomplete URI could be passed to SIP.

Default: false

Values: true | false

Example: **set use-anon true**

h323-to-sip-toheader-spec

Specifies how to generate a SIP To header from an H.323 SETUP message. When the ME receives a message from an H.323 server via the server that contains this configuration object, it creates the To header using the parameters of this object. The To header is made up of four components defined here: scheme:user@host.suffix.

Syntax

```
config vsp session-config-pool entry h323-to-sip-toheader-spec
config vsp default-session-config h323-to-sip-toheader-spec
```

Properties

See the **h323-to-sip-fromheader-spec** object for property descriptions.

h323-tos-settings

The ME supports Type of Service (ToS) marking for H.323 packets. The ToS value determines the quality of service that a call receives. The ToS byte in the IP header is used to mark packets for special consideration during routing. When the ME forwards a packet marked with a ToS value, this information is used by downstream routers to prioritize packet forwarding or perform other quality of service mechanisms.

When the ME receives an in-leg H.323 session with a ToS value set, you have the option to either forward this value to the out-leg session or override the in-leg ToS.

Additionally, when the ME receives an in-leg SIP session, you can either preserve the initial ToS value or override it in the out-leg during IW translation to H.323.

You configure in-leg and out-leg ToS settings for H.323 packets under this object.

For messages sent via TCP, the ECN field (the least significant two bits of the ToS) can neither be preserved nor overwritten by the ME. For these sessions, the ECN field in outgoing packets is always marked as zero, regardless of the incoming or overwritten value.

Additionally, if TLS over TCP is supported via H.235, the ToS value cannot be preserved, but it can be overwritten to the same ECN limitation as TCP.

Syntax

```
config vsp default-session-config h323-tos-settings
config vsp session-config-pool entry h323-tos-settings
```

Properties

in-leg-tos: Configures the in-leg settings for H.323 support on the ME.

out-leg-tos: Configures the out-leg settings for H.323 support on the ME.

handle-publish

Sets whether events are sent to a third-party server via.

Syntax

```
config vsp default-session-config handle-publish
```

```
config vsp policies session-policies policy name rule name session-config
handle-publish
config vsp dial-plan dial-prefix entryName session-config handle-publish
config vsp dial-plan route name session-config handle-publish
config vsp dial-plan source-route name session-config handle-publish
config vsp session-config-pool entry name handle-publish
```

Properties

third-party-interface: Sets whether events are sent to a third-party server via.

Default: none
Example: **set third-party-interface none**

handle-response

Assigns an action to a matching response code. If the ME receives a response to a SIP request (typically a UAS) that is within the 400-999 range, you can specify how the ME responds (forwarding the response message or re-sending the INVITE to a different peer or route). The purpose of this feature is to determine the destination to which the ME forks a call.

The ME operates in the following manner. If:

- The **handle-response** property is configured for the session-config and there is a matching response code, the ME takes the configured action. If the response code does not match, the ME uses the default action (try-next-peer).
- The **handle-response** property is not configured for the session-config, the ME uses the handle-response setting in the server configuration (if it exists).

When the ME receives a the specified response code from a call, it takes one of the following actions:

- **try-next-peer**: The ME forwards the message to the next server within a route.
- **try-next-route**: The ME forwards the message to the route that is the next most-specific.
- **forward**: The ME returns the response to the originator of the message.

A call can match more than one route, and each route may have more than one destination. For example, a call may route to destinations A and B. Destination A may have routes A1, A2, and A3. Destination B may have routes B1 and B2. If the current destination is A1, and a handle-response code match occurs with a setting of **try-next-peer**, the ME forwards the message to A2. If the setting is **try-next-route**, the ME forwards the message to B1.

Syntax

```
config vsp default-session-config handle-response
config vsp policies session-policies policy name rule name session-config
handle-response
config vsp dial-plan dial-prefix entryName session-config handle-response
config vsp dial-plan route name session-config handle-response
config vsp dial-plan source-route name session-config handle-response
config vsp session-config-pool entry name handle-response
```

Properties

entry: Specifies the action the system should take when it receives a specific response code from a call matching this session-config. Enter a code, and set a handling pattern.

Default: There is no default setting

- **Values: try-next-peer**: The system forwards the message to the next server for a route
- **try-next-route**: The system forwards the message to the route that is the next most-specific
- **forward**: The system returns the response to the originator of the message

Example: **set entry handle-response 404 try-next-route**

header-normalization

Modifies the User portion of the specified header. This object uses the same methodology as the **dial-plan > normalization** object.

Syntax

```
config vsp default-session-config header-settings header-normalization number
config vsp policies session-policies policy name rule name session-config
header-settings header-normalization number
config vsp dial-plan dial-prefix entryName session-config header-settings
header-normalization number
config vsp dial-plan route name session-config header-settings
header-normalization number
config vsp dial-plan source-route name session-config header-settings
header-normalization number
config vsp session-config-pool entry name header-settings header-normalization
number
```

Properties

admin: Enables or disables this configuration entry.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

destination: Specifies the header to be normalized. The system makes changes to the User field of the destination URI. Changes are applied to all messages types identified in the **apply-to-methods** property.

Default: There is no default setting
Example: **set destination Diversion**

value: Sets the type of normalization that the system applies to outgoing calls to a provider (to the USER field of the destination URI). See User Normalization Properties for property setting options and descriptions.

Default: none (no normalization is applied)
Example: **set value replace-prefix 866**

apply-to-methods: Specifies the message type to which the system applies header value changes. The system then changes the specified URI according to the settings of the **value** property of this object.

When you modify this value, the system overwrites the current setting with only the message types you specify. For example, if set to the default and you enter **NOTIFY**, the system only modifies NOTIFY messages. Enter multiple types separated by a plus sign (+) with no spaces.

Default: INVITE
Example: **set apply-to-methods INVITE+CONTACT**

apply-to-response: Specifies whether to apply header value changes to SIP requests or requests and responses. Set to **no** to apply changes only to requests. Set to **yes** to apply to responses as well. If yes, you must set the response code to which it applies. Create additional altered-body profiles to change multiple response types.

Default: no
Values: no | yes *responseCode*
Example: **set apply-to-response yes 200**

session-persistent: Specifies to which messages in a session the ME should apply changes made with this object. When **enabled**, the ME applies any TO, FROM, or REQUEST URI changes to the first and all subsequent messages in a session. When **disabled**, the default, the system applies the changes only to the first message in the session.

Default: disabled
Values: enabled | disabled
Example: **set session-persistent enabled**

cseq: *Secondary property.* Sets a mechanism to further filter which SIP messages have the header expression modifications applied. If **cseq** is set to zero (the default), the ME applies the changes to all SIP messages. If set to any other value, the system only applies the changes to SIP messages having a CSEQ field that matches that value.

Default: 0
Example: **set cseq 100**

header-settings

Configures the ME to remove, or to remove and replace the content of fields from the SIP header. In addition, you can identify header types to specifically allow or block. The allowed-headers and blocked-headers properties apply the following rules:

- The From, To, CSeq, and Call-ID headers are required and cannot be blocked.
- The allowed list overrides settings of the blocked list. If a header is explicitly allowed, it cannot then be blocked using the blocked list.
- If a header name does not match a value in either list, it is allowed.
- The ME accepts regular expressions for an entry; all special characters apply.

Syntax

```
config vsp default-session-config header-settings
config vsp policies session-policies policy name rule name session-config
header-settings
config vsp dial-plan dial-prefix entryName session-config header-settings
config vsp dial-plan route name session-config header-settings
config vsp dial-plan source-route name session-config header-settings
config vsp session-config-pool entry name header-settings
```

Properties

allowed-header: Sets the SIP headers that should be explicitly allowed to remain in the SIP message. You can enter any number of header names by re-executing the command. See the Purpose for applicable rules.

Default: There is no default setting
Example: **set allowed-header Via**

blocked-header: Sets the SIP headers that should be explicitly removed from the SIP message. You can enter any number of header names by re-executing the command. See the Purpose for applicable rules.

Default: There is no default setting
Example: **set blocked-header .***

apply-allow-block-to: Sets whether the allow and block properties of this object apply to requests only or requests and responses. When **disabled**, changes apply only to requests. When **enabled**, the default, changes apply to requests and responses.

Default: requests-and-responses
`Values: requests | responses | requests-and-responses`
Example: **set apply-allow-block-to responses**

pAssert-mode: *Secondary property.* Sets whether to strip the number in the P-Asserted-Identity field from the SIP header. When **enabled**, the system replaces the value in the From field with the value from the P-Asserted-Identity field for the outbound call leg. (Note that the system maintains the original From field value in the Contact field.)

Default: disabled
`Values: enabled | disabled`
Example: **set pAssert-mode enabled**

header-to-strip: *Secondary property.* Configures the system to strip the value of the specified field. Enter a SIP header field name.

Default: There is no default setting
Example: **set header-to-strip Remote-Party-ID**

apply-to-allow-block-to-dialog: Specifies whether the allow and block properties of this object apply to a specific dialog or not.

Default: both
- Values: inbound: Apply to the inbound dialog only
- outbound: Apply to the outbound dialog only
- both: Apply to both inbound and outbound dialogs

Example: **set apply-to-allow-block-to-dialog inbound**

sip-manipulation: Specify the configured sip-manipulation you want to associate with this header-setting. Configure the sip-manipulation in the **sip-manipulation-pool > sip-manipulation** object.

Default: There is no default setting
Example: **set sip-manipulation sipmanip1**

in-codec-preferences

Sets a preference for CODECs, influencing the ME ordering of them in the SDC on the inbound leg of a call. The ME removes those CODECs with a zero priority from the SDP. CODEC preferences do not cause the ME to add a CODEC to the SDP, but to

remove and/or reorder existing CODECs according to their priority. The ME places a CODEC whose priority is not specified in its original order, just ahead of the known auxiliary CODECs (e.g., telephone-events). This is not a mechanism to add CODECs into an SDP, only to order those that are already there (via transcoding or from the original offer/answer).

Syntax

```
config vsp default-session-config in-codec-preferences
config vsp policies session-policies policy name rule name session-config
in-codec-preferences
config vsp dial-plan dial-prefix entryName session-config in-codec-preferences
config vsp dial-plan route name session-config in-codec-preferences
config vsp dial-plan source-route name session-config in-codec-preferences
config vsp session-config-pool entry name in-codec-preferences
```

Properties

preferences<media-type>[codec][priority]: Assigns a priority to a given CODEC for inbound audio or video sessions.

Default: There are no default settings

- **Values: media type**: Select for audio or video. The associated CODEC (subtype) is preferred according to the priority for that media type.
- **codec**: The CODEC to which the priority applies. Use the question mark character at the command line to see a list of available CODECs, or enter any CODEC name.
- **priority**: Sets a preference for the CODEC. The lower the number, the more preferred the CODEC. Assigning a priority value of zero disables the CODEC for the session. The system removes these CODECs before sending the SDP offer or answer. (0-100)

Example: **set preferences video g729 1**

in-dtmf-preferences

Configures the ME's in-leg DTMF method preferences.

Syntax

```
config vsp default-session-config in-dtmf-preferences
config vsp session-config-pool entry in-dtmf-preferences
```

Properties

admin: Specifies whether or not this DTMF preference list is applied to calls matching this session configuration.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

preferences: Allows you to configure supported dtmf-types and assign them with a priority to determine the ME's preferences.

Default: audio 1
First select a DTMF method. The available DTMF methods are:

- Values: audio

- rfc-2833
- sip-info-dtmf
- sip-info-dtmf-relay
- sip-notify
- h245-alphanumeric
- h245-signal
- q931

Then assign it a priority. This can be from 0-100. A value of **0** means the method is not supported. The lower the priority, the more preferred the DTMF method.

Example: **set preferences rfc-2833 2**

in-dtmf-settings

The ME can be configured to translate one DTMF method to another. Through this object, you can control the length of play and pause time and volume for the digits that the ME plays on the in-leg.

Syntax

```
config vsp default-session-config in-dtmf-settings
config vsp session-config-pool entry out-dtmf-settings
```

Properties

digit-volume: Specifies the volume setting for the DTMF tones. The digit volume is measured in decibel (dB) of the measured power referenced to one milliwatt, measured at a zero transmission level point. The smaller the dBm0, the louder the volume.

Default: -20
Values: Min: -36 / Max: 0
Example: **set digit-volume -15**

digit-duration: Specifies the length of time, in milliseconds, that the ME plays each DTMF digit.

Default: 750
Values: Min: 100 / Max: 10000
Example: **set digit-duration 1000**

min-digit-duration: Specifies the minimum length of time, in milliseconds, that the ME plays each DTMF digit. If a DTMF event has a duration less than this value, the **digit-duration** property overrides the duration and is used to play the DTMF event.

Default: 60
Values: Min: 5 / Max: 100
Example: **set min-digit-duration 75**

max-digit-duration: Specifies the maximum length of time, in milliseconds, that the ME plays each DTMF digit. If a DTMF event has a duration greater than this value, the **digit-duration** property overrides the duration and is used to play the DTMF event.

Default: 2000
Values: Min: 100 / Max: 10000
Example: **set max-digit-duration 3000**

inter-digit-duration: Specifies the length of time, in milliseconds, that the ME pauses between playing each digit.

Default: 250
Values: Min: 0 / Max: 1000
Example: **set inter-digit-duration 500**

pause-duration: Specifies the length of time, in milliseconds that the ME pauses when it encounters a comma character in the conference code. The comma is a special character in the conference code that indicates a specified time the ME must wait before playing the next tone.

Default: 3000
Values: Min: 500 / Max: 10000
Example: **set pause-duration 4000**

minimum-duration: Specifies the minimum time, in milliseconds, between detecting RFC-2833 events.

Default: 60
Values: Min: 0 / Max: 1000
Example: **set minimum-duration 100**

as-audio: Specifies whether the ME sends audio or DTMF packets to the conference server when representing conference code tones. When true, the ME encodes the sound in the current CODEC. When false, the ME sends DTMF packets.

Default: true
Values: true | false
Example: **set as-audio false**

in-dtmf-translation

Secondary object. Controls the method used for forwarding DTMF tones in a call. The two supported methods are via the signaling stream using SIP INFO messages or via a DTMF packet that is in compliance with RFC 2833, RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals. Using this object, you can configure the ME to:

- Forward packets in the form they arrived.
- Pick them out of the RTP stream and send them in a SIP INFO message (if they arrived as DTMF packets).
- Extract them from a SIP INFO message and send them as DTMF packets (if the SIP INFO message contained a DTMF body).

Inbound DTMF translation applies to the segment from the initiator to the ME device.

Syntax

```
config vsp default-session-config in-dtmf-translation
config vsp policies session-policies policy name rule name session-config
in-dtmf-translation
config vsp dial-plan dial-prefix entryName session-config in-dtmf-translation
config vsp dial-plan route name session-config in-dtmf-translation
config vsp dial-plan source-route name session-config in-dtmf-translation
config vsp session-config-pool entry name in-dtmf-translation
```

Properties

info: Specifies the method to use for forwarding DTMF tones that were received in a SIP INFO message. If set to **info**, the system forwards the message as it was received (in an INFO message). If set to **rfc-2833**, the system extracts the DTMF body from the INFO message and sends the content via DTMF packets.

Default: info
Values: info | rfc-2833
Example: **set info rfc-2833**

drop-info: Specifies whether to drop the SIP INFO message if the info property is set to rfc-2833. If set to **true**, the system drops the INFO packet and only sends the DTMF packets. If set to **false**, the system sends both.

Default: false
Values: true | false
Example: **set drop-info true**

rfc-2833: Specifies the method to use for forwarding DTMF tones that were received in DTMF packets. If set to **rfc-2833**, the system forwards the message as it was received (in DTMF packets). If set to **info**, the system extracts the DTMF packets from the RTP stream and sends the content via a SIP INFO message. The system sends one INFO message per event detected.

Default: rfc-2833
Values: info | rfc-2833
Example: **set rfc-2833 info**

drop-rfc-2833: Specifies whether to drop the DTMF packets if the rfc-2833 property is set to info. If set to **true**, the system drops the RFC 2833 packets and only sends the SIP INFO message. If set to **false**, the system sends both.

Default: false
Values: true | false
Example: **set drop-rfc-2833 true**

info-dtmf-body: Specifies the body type to use in a SIP INFO message when converting from RFC 2833 format. When using **dtmf**, the message body contains just single character (the digit that was pressed). When set to **dtmf-relay**, the body contains the single character plus duration data.

Default: dtmf-relay
Values: dtmf-relay | dtmf
Example: **set info-dtmf-body dtmf**

timeout-rfc-2833: *Secondary property*. Sets the number of milliseconds the system waits before sending a SIP INFO message if it does not detect the end of the event. The timer is started at the start of an event. This property only applies when the forwarding method has been changed from **rfc-2833** to **info**, and is used in the event that when monitoring DTMF, the system does not detect an event end.

Default: 1000
Example: **set timeout-rfc-2833 1500**

in-echo-cancellation-settings

Configures inbound server-side acoustic echo cancellation for devices running telephony applications that do not have functional on-board echo cancellation.

Syntax

```
config vsp session-config-pool entry name in-echo-cancellation-settings
config vsp default-session-config in-echo-cancellation-settings
```

Properties

admin: Enables or disables echo cancellation on inbound call-legs.

Default: disabled
Values: enabled | disabled
Example: **set**

platform-delay: Although the maximum echo tail allowed by the echo cancellation algorithm is 200 ms, it is possible that the actual echo tail for certain devices is longer. This parameter allows you to account for any extra delay in addition to the 200 ms. For example, if your total acoustic delay is 500 ms, set **echo-tail** to 200 ms and **platform-delay** to 300 ms. This covers the entire 500 ms delay.

Default: 0
Values: Min: 0 / Max: 65535
Example: **set platform-delay 100**

echo-tail: The amount of time, in milliseconds, that it takes for the person speaking to hear his or her own echo. This is also known as echo delay. The maximum echo tail supported by the ME's algorithm is 200 ms.

Default: 16
Values: Min: 0 / Max: 200
Example: **set echo-tail 150**

noise-reduction: Allows you to reduce background noise. A value of 0 means there is no noise reduction and 5 is the maximum reduction.

Default: 2
Values: Min: 0 (no noise reduction) / Max: 5
Example: **set noise-reduction 4**

in-encryption

Sets the parameters for inbound encrypted media sessions when the ME is anchoring a call. This is the portion from the initiator to the ME device. With this object you set the encryption requirements on the call, and the encryption method used.

Syntax

```
config vsp default-session-config in-encryption
config vsp policies session-policies policy name rule name session-config
in-encryption
config vsp dial-plan dial-prefix entryName session-config in-encryption
config vsp dial-plan route name session-config in-encryption
config vsp dial-plan source-route name session-config in-encryption
config vsp session-config-pool entry name in-encryption
```

Properties

mode: Specifies the encryption requirements on the incoming call (from endpoint to the ME device). The method of encryption used is determined by the **type** property.

Default: none

- **Values: none**: The system disables the encryption put forth by the incoming endpoint (i.e, it responds “no” to the encryption portion of the authentication handshake.) If the outbound endpoint requires encryption, the call is dropped.
- **pass-thru**: The system passes cryptographic parameters through the box and does not participate in RTP encryption. This method renders some advanced media services unusable (in particular, recording, announcements, transcoding, call monitoring, RTP stats, media verification, and RTCP generation). When using this option, set mode to **pass-thru** in both in-encryption and **out-encryption**.
- **allow**: If the incoming endpoint offers encryption to the ME device, the system answers with it. If the endpoint does not offer encryption, the system does not answer with or initiate encryption.
- **follow**: If the outbound endpoint offers encryption, the system offers encryption (the type set by policy) to the inbound endpoint.
- **offer**: The system offers encryption to the inbound endpoint when the session is first established.
- **reoffer**: The system offers encryption to the inbound endpoint whether the message is an INVITE or a REINVITE. This setting is most applicable when an endpoint issues a REINVITE, and encryption was not required with the original INVITE. In this case, the system will again offer encryption when forwarding the message.
- **require**: The call must come in with encryption specified or the system drops it.

Example: **set mode allow**

type: Sets the type of encryption on inbound sessions. In choosing a type, the system uses the encryption expected by that device or application.

Default: RFC-3711

- **Values: Linksys**: Uses SIP INFO messages to exchange mini-certificates and exchange a symmetric key. The media encryption is similar to RFC-3711, but done with AES-128 Countermode and with HMAC MD5 authentication. See the **linksys** action for more certificate information.
- **RFC-1889**: Use encryption as defined in *RFC 1889, RTP: A Transport Protocol for Real-Time Applications*. This mode is used for compatibility with Windows Messenger and Microsoft Office Communicator, neither of which currently support RFC-3711 encryption. Instead, it uses a DES-CBC encryption of the entire UDP payload (including RTP headers) with no authentication.
- **RFC-3711**: Use encryption as defined in *RFC 3711, The Secure Real-time Transport Protocol (SRTP*).

Example: **set type Linksys**

require-tls: Specifies the requirements of the signaling protocol for the call inbound leg. It defines whether the system offers SRTP over a non-secure (TCP or UDP) signaling connection. The action of this property depends on the setting of the **mode** property.

In most cases, this property does not need to be modified because the system does not consider the in-leg transport (only whether or not crypto was offered). However, set this property to true to ensure that the system does not offer crypto to a client on the in-leg that is not using TLS.

Default: false

- **Values: true**: The system only offers encryption when talking to a TLS client. If TLS and SRTP are required (**mode** is set to **require**), the system fails calls going to

TCP/UDP clients. If the mode property is set to **offer** or **follow**, the system forwards the call without SRTP.

- **false**: The system offers SDP messages according to the mode setting without regard for the signaling transport. This allows keys to be exchanged in an insecure message.

Example: **set require-tls true**

priority-AES-128-CM-HMAC-SHA1-32: *Secondary property*. Sets a preference for 32-bit SHA1 authentication tags on incoming calls. The system supports (offers) both 32- and 80-bit authentication tags on ingress. A value of 0 disables support for the 32-bit tag.

Default: 1
Values: Min: 1 (most preferred) / Max: 5
Example: **set priority-AES-128-CM-HMAC-SHA1-32 2**

priority-AES-128-CM-HMAC-SHA1-80: *Secondary property*. Sets a preference for 80-bit SHA1 authentication tags on incoming calls. The system supports (offers) both 32- and 80-bit authentication tags on ingress. A value of 0 disables support for the 80-bit tag. To enable, set a value between 1 (most preferred) and 5.

Default: 2
Values: Min: 1 (most preferred) / Max: 5
Example: **set priority-AES-128-CM-HMAC-SHA1-80 1**

mki-length: *Secondary property*. Provides support for the optional Master Key Identifier bit defined in *RFC 3711, The Secure Real-time Transport Protocol (SRTP)*. The value specify sets the number of bytes in the MKI. The system then sends the negotiated identifier of that length indicating which master key to use for decryption with each SRTP packet. Note that the endpoint must support this option.

Default: 0
Values: Min: 0 / Max: 4
Example: **set mki-length 2**

mikey-offer-location: *Secondary property.* Controls where in the SDP the system stores the MIKEY offer (the "a=key-mgmt:mikey" line) when it is made. If MIKEY is offered to the system, it puts the MIKEY answer in the location where the offer was located. Set the location as the media descriptor or session level.

Default: session
Values: session | media-descriptor
Example: **set mikey-offer-location media-descriptor**

mikey-time-tolerance: *Secondary property.* Controls where in the SDP the system stores the MIKEY offer (the "a=key-mgmt:mikey" line) when it is made. If MIKEY is offered to the system, it puts the MIKEY answer in the location where the offer was located. Set the location as the media descriptor or session level.

Default: 60
Example: **set mikey-time-tolerance 90**

symmetric-address-failure: *Secondary property.* Specifies whether the system learns the source IP address from the RTP/RTCP packets, even if the packets fail decryption. When **enabled**, the first packet in a particular stream that fails SRTP decryption causes a DroppedPacket notification to be sent to the application with the address of the packet. The application treats this like a srcIPChanged notification.

Default: disabled
Values: enabled | disabled
Example: **set symmetric-address-failure enabled**

treat-as-secure: *Secondary property*. Specifies whether a proprietary security indicator is used on the SIP interface; either ST-secure or ST-insecure. This setting only operates on the X-Siemends-Call-Type header when MIKEY encryption is involved in pass-thru mode.

Default: disabled

- Values: disabled: Sets ST-insecure
- enabled: Sets ST-secure
- auto: If SRTP is active on both sides of the call, the ME allows the X-Siemends-Call-Type header to pass unchanged. If SRTP is not active on other side of the call, the header is set to ST-insecure. The "auto" value sets the interface to trusted.

Example: **set treat-as-secure enabled**

encryption-preferences: Creates a prioritized list of encryption types to offer or accept.

> **Note:** Always give DTLS a priority of 1 and RFC3711 a priority of 2.

Default: There is no default setting.
Example: **set encryption-preferences DTLS 1**

in-ice-settings

Configures the ME's ICE settings on the in-leg.

Syntax

```
config vsp default-session-config in-ice-settings
config vsp session-config-pool entry <name> in-ice-settings
```

Properties

admin: Enables or disables ICE on the in-leg.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

connectivity-check-time-out: Specifies the time in ms before a STUN connectivity check times out.

Default: 100
Values: Min: 0 | Max: 4294967296
Example: **set connectivity-check-time-out 75**

connectivity-check-max-retransmits: Specifies the number of times the ME retransmits ICE STUN connectivity checks before labeling a candidate pair as Failed.

> **Note:** To achieve maximum interoperability with Chrome, set this value to no less than **200**.

Default: 7
Values: Min: 0 | Max: 255
Example: **set connectivity-check-max-retransmits 5**

delay-stun-responses: *Secondary property*. When enabled, the ME does not respond to STUN until the 200 OK is received.

Default: disabled
Values: enabled | disabled
Example: **set delay-stun-responses enabled**

suppress-re-invites: *Secondary property*. When enabled the ME does not send a re-INVITE when ICE completes successfully.

Default: disabled
Values: enabled | disabled
Example: **set suppress-re-invites enabled**

nomination-policy: Specify the RFC-5245 nomination type.

Default: regular
Values: regular | aggressive
Example: **set nomination-policy aggressive**

delay-stun-requests: *Secondary property*. When enabled, connectivity checks do not begin until a STUN request is received.

Default: disabled
Values: enabled | disabled
Example: **set delay-stun-requests enabled**

trickle-ice: Enable or disable trickle ICE on the out-leg.

Default: disabled
Values: enabled | disabled
Example: **set trickle-ice enabled**

terminate-call-on-ice-failure: *Secondary property*. When enabled, a call is terminated if ICE fails.

Default: disabled
Values: enabled | disabled
Example: **set terminate-call-on-ice-failure enabled**

in-leg-tos

This object determines the ToS value setting for the in-leg of the session. This ToS value determines the quality of service that the call receives. The ME marks the ToS field of all packets it sends out on the in-leg with the value you specify. Enter a number that represents the 8-bit Differentiated Services (DS) field of the IP packet in decimal format, such as 26 for 00011010 or 104 for 01101000. This value can be of use to upstream devices.

Syntax

```
config vsp default-session-config h323-tos-settings in-leg-tos
config vsp session-config-pool entry h323-tos-settings in-leg-tos
```

Properties

value: Specify the ToS value setting for the in-leg of the session.

Default: 0
Values: Min: 0 / Max: 255
Example: **set value 22**

in-hold-translation

Configures the SDP hold attributes that are sent to an endpoint. The **in-hold-translation** object applies to SDP bodies sent to the endpoint that initiated the call. The **out-hold-translation** object applies to SDP bodies sent to the endpoint that initially received the call. When a call/stream is put on hold, the endpoint putting the call on hold sends an SDP offer with certain recognizable SDP characteristics: SDP connection information (c-line) and hold attributes that typically include "inactive" or "sendonly." The endpoint that is being put on hold responds with an SDP answer acknowledging the hold. This is normally expressed by including a "recvonly" or "inactive" hold attribute. Not all servers recognize all SDP hold characteristics, so these objects can be used to configure the SDP hold characteristics sent to a given server. The ME recognizes the following hold attributes (for offer or answer): sendrecv, sendonly, recvonly, and inactive. When changing or removing hold attributes, the ME will remove or overwrite any of these attributes in the SDP.

For example, an endpoint being put on hold may responds to a server offer with "a=inactive" or "a=recvonly." Some servers may interpret an SDP answer of "a=inactive" as "I'm not listening, do not send me music-on-hold." If the endpoint putting the call on hold will play music-on-hold anyway, you may configure the ME change the answer attribute to "a=recvonly." In that way, when the server receives the SDP answer with the "a=recvonly," it may still play the music-on-hold.

Syntax

```
config vsp default-session-config in-hold-translation
config vsp policies session-policies policy name rule name session-config
in-hold-translation
config vsp dial-plan dial-prefix entryName session-config in-hold-translation
config vsp dial-plan route name session-config in-hold-translation
config vsp dial-plan source-route name session-config in-hold-translation
config vsp session-config-pool entry name in-hold-translation
```

Properties

admin: Specifies whether this hold translation entry is applied to calls matching the session configuration.

Default: disabled
Values: enabled disabled
Example: **set admin enabled**

offer-address: Specifies how the ME modifies the SDP connection information (c-line) sent in an SDP offer.

Default: pass

- **Values: pass**: Not modify the c-line.
- **zero**: Change the address reported in the c-line to 0.0.0.0.
- **non-zero**: Change the address reported in the c-line to the last known address, typically the system interface address.

Example: **set offer-address zero**

offer-attribute: Specifies how the ME modifies the SDP hold attributes in the SDP offer.

Default: pass

- **Values: pass**: Not modify the hold attributes.

- **remove**: Remove any recognized hold attributes from SDP.
- **inactive**: Set the hold attribute to "inactive."
- **sendonly**: Set the hold attribute to "sendonly."
- **sendrecv**: Set the hold attribute to "sendrecv." Use with care; this setting effectively tells the endpoint that the stream is not on hold.

Example: **set offer-attribute inactive**

answer-address: Specifies how the ME modifies the SDP connection information (c-line) sent in an SDP answer.

Default: pass

- **Values: pass**: Not modify the c-line.
- **zero**: Change the address reported in the c-line to 0.0.0.0.
- **non-zero**: Change the address reported in the c-line to the last known address, typically the system interface address.

Example: **set answer-address zero**

answer-attribute: Specifies how the ME modifies the SDP hold attributes in the SDP answer.

Default: pass

- **Values: pass**: Not modify the hold attributes.
- **remove**: Remove any recognized hold attributes from SDP.
- **inactive**: Set the hold attribute to "inactive."
- **sendonly**: Set the hold attribute to "sendonly."
- **sendrecv**: Set the hold attribute to "sendrecv." Use with care; this setting effectively tells the endpoint that the stream is not on hold.

Example: **set answer-attribute remove**

remove-telephone-events: *Secondary property*. Specifies whether the system strips telephone-events from the SDP when a call is placed on hold. When set to **true**, the system does strip events, which may be necessary for some phones (Polycom, for example). When **false**, the system does not modify events in the SDP.

Default: false
Values: true | false
Example: **set remove-telephone-events true**

in-media-loss-detection

Configures in-leg media loss detection settings.

Syntax

```
config default-session-config in-media-loss-detection
config session-config-pool entry <name> in-media-loss-detection
```

Properties

admin: Enable or disable media loss detection on the incoming call leg.

Default: disabled
Values: enabled | disabled

Example: **set admin enabled**

interval: Set the interval, in seconds, in which to check for media loss on the incoming call leg.

Default: 5
Example: **set interval 3**

in-media-normalization

Changes the media descriptor string (e.g., the CODEC for audio or video) in the SDP. Use this in cases where a client is unable to understand a variation in name of a CODEC/media descriptor. For example, G729 is sometimes transmitted as G729a. Inbound media normalization applies to the segment from the initiator to the ME device.

Syntax

```
config vsp default-session-config in-media-normalization
config vsp policies session-policies policy name rule name session-config
in-media-normalization
config vsp dial-plan dial-prefix entryName session-config in-media-normalization
config vsp dial-plan route name session-config in-media-normalization
config vsp dial-plan source-route name session-config in-media-normalization
config vsp session-config-pool entry name in-media-normalization
```

Properties

normalize: Specifies, for a media type, how to normalize a CODEC/media descriptor name. The initial subtype is the type the system matches on and replaces. The alternate subtype is the type that the system then inserts in the SDP to replace the initial subtype. You can select a pre-configured type or enter a custom type.

Default: audio
Values: <audio | video | application | image | custom-mime-type *mimeType*>[*initialSubType*] [*alternateSubType*]
Example: **set normalize video g729 g729a**

in-media-scanner-settings

Configure in-media-scanner-settings. When in-media-scanner-settings are configured, a media scanner is started on the in-leg of the call and reports events based on the analysis of the received audio from the endpoint.

Syntax

```
config vsp default session-config in-media-scanner-settings
config vsp session-config-pool entry <name> in-media-scanner-settings
config vsp policies session-policies policy <name> rule <name> session-config
in-media-scanner-settings
```

Properties

admin: Enables or disables the in-media scanner settings.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

pre-scan-time: The number of milliseconds to delay before invoking the in-media scanner. This property is not applicable for the on-demand media scanner.

Default: 20
Values: Min: 0 / Max: 4294967295
low-threshold: Enter the talk or stable tone signal power threshold in dbs. Crossing this threshold indicates quiet.

Default: -36
Values: Min: -36 / Max: 3
Example: **set low-threshold -25**

high-threshold: Enter the quiet signal power threshold in dbs. Crossing this threshold indicates talking or stable tone.

Default: -36
Values: Min: -36 / Max: 3
Example: **set high-threshold -25**

low-long-duration: The number of milliseconds of detected quiet before reporting a long-pause, otherwise a short pause is reported.

Default: 2000
Values: Min: 0 / Max: 4294967295
Example: **set low-long-duration 1500**

high-long-duration: The number of milliseconds of detected talk or tone before reporting a long talk or stable tone, otherwise a short talk is reported.

Default: 900
Values: Min: 0 / Max: 4294967295
Example: **set high-long-duration 1500**

averaging-window: *Secondary property* The amount of time in dbs used when calculating signal strength.

Default: 100
Values: Min: 10 / Max: 1000
Example: **set averaging-window 500**

nominal-rounding-factor: *Secondary property.* The signal strength is rounded to the nearest multiple of this value before comparing against other signal strengths.

Default: 2
Values: Min: 1 / Max: 25
Example: **set nominal-rounding-factor 4**

event-report-frequency: The number of milliseconds the media scanner should wait between generation of media scanner events. Setting this property to 0 causes the media scanner events to be reported immediately as they occur.

Default: 1000
Values: Min: 0 / Max: 60000
Example: **set event-report-frequency 39**

event-report-count-threshold: The maximum number of media scanner events that can be pending for waiting for the event-report-frequency timer to expire before being reported. If the number of queued media scanner events reaches this count, all of the events will be immediately reported.

Default: 25
Values: Min: 1 / Max: 6000
Example: **set event-report-count-threshold 73**

event-report-flags: Set the media scanner events to report flags.

Default: report all events
Values: short-pause, long-pause, short-talk, long-talk, stable-tone
Example: set **event-report-flags short-pause**

in-msrp-session-leg

Configures in-leg Message Session Relay Protocol (MSRP) interworking.

Syntax

```
config default-session-config in-msrp-session-leg
config session-config-pool entry <name> in-msrp-session-leg
```

Properties

admin: Enable or disable MSRP interworking on this call leg.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

msrp-leg-transport: Specify the MSRP transport method for RCS or WebRTC.

Default: TCP
Values: TCP | TLS | WS | WSS
Example: set msrp-leg-transport TLS

connection-reuse: Not currently supported.

default-media-interface: Specify the local media interface to use for an MSRP connection if svc-routing fails to locate the appropriate interface.

Default: There is no default setting
Example: **set default-media-interface int1**

use-mdesc-cline-first: (*Advanced*) Specify whether the MSRP session manager attempts to use the SDP c-line (before using the path attribute) to learn the remote MSRP endpoint's media IP address.

Default: false
Values: true | false
Example: **set use-mdesc-cline-first false**

socket-read-size: (*Advanced*) Specify the MSRP socket read size to use when assembling incoming MSRP messages.

Default: 4096
Min: 0 / Max: 4294967296
Example: **set socket-read-size 5000**

partial-forward-size: (*Advanced*) Specify the threshold for forwarding buffered MSRP message content bytes.

Default: 1024
Min: 0 / Max: 4294967296
Example: **set partial-forward-size 2050**

connsrc-match-path: (*Advanced*) Specify whether the ME allows incoming MSRP connections even when a remote address does not match the SDP path attribute.

Default: false
Values: true | false
Example: **set connsrc-match-path false**

allow-missing-fingerprint: (*Advanced*) Specify whether the ME allows an MSRP secure connection even when the SDP fingerprint attribute is missing.

Default: false
Values: true | false
Example: **set allow-missing-fingerprint false**

inbound-controls

Customizes call admission control, on a per-AOR basis, for inbound calls only. This is in contrast to the admission control settings found in the **location-call-admission-control** object, which sets aggregate inbound and outbound limits. Use the **show location-cac** status provider to view call admission control settings and counters for an AOR.

Syntax

```
config vsp default-session-config location-call-admission-control
inbound-controls
config vsp policies session-policies policy name rule name
session-config location-call-admission-control inbound-controls
config vsp dial-plan dial-prefix entryName session-config
location-call-admission-control inbound-controls
config vsp dial-plan route name session-config
location-call-admission-control inbound-controls
config vsp dial-plan source-route name session-config
location-call-admission-control inbound-controls
config vsp session-config-pool entry name
location-call-admission-control inbound-controls
```

Properties

max-number-of-concurrent-calls: Specifies the maximum number of active incoming and outgoing calls allowed for this AOR at one time. When this value is reached, the connection does not accept calls until the value drops below the threshold.

A value of **0** causes the system to decline all calls and registrations.

Default: 1000
Values: Min: 0 / Max: 1000000
Example: **set max-number-of-concurrent calls 1500**

max-calls-in-setup: Sets the maximum number of simultaneous inbound and outbound call legs in setup stage that are allowed for this AOR. A call leg in setup is much more compute-intensive than established call legs, so this value is more restrictive than the concurrent call leg value. A value of 0 causes the system to decline all calls and registrations.

Default: 30
Values: Min: 0 / Max: 10000
max-bandwidth: *Secondary property.* Specifies the amount of bandwidth the system allocates to the AOR. When the system reaches the maximum bandwidth limit for a server, it rejects calls until bandwidth use drops below the maximum.

Default: unlimited
Values: unlimited | *kbps*
Example: **set max-bandwidth 512**

call-rate-limiting: *Secondary property.* Limits the number of calls sent to the AOR within a certain interval. Once this interval is reached, the system rejects any calls to

this AOR, returning a response code and message, until the rate decreases. This feature sets the acceptable arrival rate for incoming calls.

If **enabled**, set the number of calls allowed and the measurement interval (in seconds). You can also enter a result code from 400 to 699 and a text string to accompany call rejection if no available server is found.

Default: disabled; if set to enabled, the default calls-per-interval is 60, the default interval is 1, and the default result is 486, Busy Here
Values: enabled [*calls-per-interval*][*interval*][*result-code*][*result-string*] | disabled
Example: **set call-rate-limiting enabled 50 1 480 "Temporarily unavailable"**

inbound-header-settings

The inbound-header-settings configuration object allows you to set fields to remove and/or replace header settings in the SIP headers for inbound traffic.

Syntax

```
config vsp policies session-policies policy default rule session-config
inbound-header-settings
```

Properties

pAssert-mode: *Secondary property.* Sets whether or not to strip the number in the P-Asserted-Identity field from the SIP header. When enabled, the ME replaces the value in the From field with the value from the P-Asserted-Identity field for the outbound call leg. (Note that the ME maintains the original From field value in the Contact field.)

Default: disabled
Values: enabled | disabled
Example: set pAssert-mode enabled

header-to-strip: *Secondary property.* Configures the ME to strip the value of the specified field. Enter a SIP header field name.

Default: There is no default setting
Example: set header-to-strip sip1

allowed-header: Sets the SIP headers that should be explicitly allowed to remain in the SIP message. You can enter any number of header names by re-executing the command.

Default: There is no default-setting
Example: set allowed-header header1

blocked-header: Sets the SIP headers that should be explicitly removed from the SIP message. You can enter any number of header names by re-executing the command.

Example: set blocked-header header5

apply-allow-block-to: Sets whether the allow and block properties of this object apply to request messages, response messages, or both.

Default: requests-and-responses

- Values: requests: Apply to requests only
- responses: Apply to responses only
- requests-and-responses: Apply to requests and responses

Example: **set apply-allow-block-to responses apply-to-allow-block-to-dialog**: Specifies whether the allow and block properties of this object apply to a specific dialog or not.

Default: both

- Values: inbound: Apply to the inbound dialog only
- outbound: Apply to the outbound dialog only
- both: Apply to both inbound and outbound dialogs

Example: **set apply-to-allow-block-to-dialog inbound**

sip-manipulation: Specify the configured sip-manipulation you want to associate with this header-setting. Configure the sip-manipulation in the **sip-manipulation-pool > sip-manipulation** object.

Default: There is no default setting
Example: **set sip-manipulation sipmanip1**

inbound-request-uri-specification

Specifies whether the ME modifies the content of the host, port, and/or transport fields of the REQUEST URI. If set, changes are applied only to the REQUEST message traveling in the opposite direction of the session initiation REQUEST message. (Use the **request-uri-specification** object to change outbound REQUEST URIs.) These properties should only be changed to override the default behavior because of issues with an intermediary device.

Syntax

```
config vsp default-session-config inbound-request-uri-specification
config vsp policies session-policies policy name rule name session-config
inbound-request-uri-specification
config vsp dial-plan dial-prefix entryName session-config
inbound-request-uri-specification
config vsp dial-plan route name session-config inbound-request-uri-specification
config vsp dial-plan source-route name session-config
inbound-request-uri-specification
config vsp session-config-pool entry name inbound-request-uri-specification
```

Properties

host-use-next-hop: Specifies whether to change the host portion of the REQUEST URI. If **enabled**, the system sets the host portion to the IP address of the next hop. If **disabled**, the host portion remains unchanged.

Default: disabled
Values: enabled | disabled
Example: **set host-use-next-hop enabled**

port-use-next-hop: Specifies whether to change the port specified in the REQUEST URI. If **enabled**, the system sets the port number to the port used for the next hop. If **disabled**, the port number remains unchanged.

Default: disabled
Values: enabled | disabled
Example: **set port-use-next-hop enabled**

transport-use-next-hop: Specifies whether to change the transport protocol specified in the REQUEST URI. If **enabled**, the system sets the transport to the protocol used by the next hop. If **disabled**, the transport protocol remains unchanged.

Default: disabled
Values: enabled | disabled
Example: **set transport-use-next-hop enabled**

inbound-sip-messages

Configures events for incoming SIP messages. This is a secondary object.

Syntax

```
config vsp session-config-pool entry name event-settings inbound-sip-messages
config vsp default-session-config event-settings inbound-sip-messages
```

Properties

admin: Enables or disables events for incoming SIP messages.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

apply-to-methods-for-events: Select the SIP methods you want the ME to create events for.

Default: There is no default setting
Example: **set apply-to-methods-for-events INVITE**

apply-to-responses: Specifies whether to send events for SIP requests or both requests and responses. When **no**, changes are applied to requests only. When **yes**, changes are applied to requests and responses. If **yes**, you must set the response code to which it applies. A value of 0 implies all responses.

Default: no
Values: no | yes <*response code*>
Example: **set apply-to-responses yes 0**

apply-to-dialog: Specifies whether to send events for SIP messages on a specific dialog or not.

Default: both
- Values: inbound: Apply to the inbound dialog only
- outbound: Apply to the outbound dialog only
- both: Apply to both the inbound and outbound dialogs.

Example: **set apply-to-dialog inbound**

cseq: *Secondary property*. Sets a mechanism to further filter for which SIP messages have events sent. When set to **0** (the default), the ME sends events for all SIP messages. If set to any other value, the ME only sends events for SIP messages having a CSEQ field that matches that value.

Default: 0
Values: Min: 0 / Max: 4294967296
Example: **set cseq 100**

instant-messaging

Enables IM archiving, applies text stamps before or after IM messages, and sends IM message alerts to the configured the ME event log.

Syntax

```
config vsp default-session-config instant-messaging
config vsp policies session-policies policy name rule name session-config
instant-messaging
config vsp dial-plan dial-prefix entryName session-config instant-messaging
config vsp dial-plan route name session-config instant-messaging
config vsp dial-plan source-route name session-config instant-messaging
config vsp session-config-pool entry name instant-messaging
```

Properties

directive: Assigns an action to the message.

Note that if you specify the **refuse** directive with text, the text is placed on the method line of the SIP response message. That line is usually not displayed to the user. If you want a message displayed to the sender, use the **message-to-sender** property.

Default: follow-sip-directive

- **Values: allow**: Allows the message, even if higher-level policy (under instant-messaging) says to refuse or discard it.
- **discard**: Silently deletes the message instead of delivering it. No notification is sent.
- **refuse <***resultCode***><***resultString***>**: Deletes the message, but sends a SIP error response to the sending agent. Optionally, specify the result code, between 400 and 699, and/or a result string to send in the error response. The default error code is 400, with no accompanying text.
- **follow-sip-directive**: Follows whatever actions are configured at the session-level. (These are the settings under sip-directive, and/or instant-messaging.)

Example: **set directive refuse 500 Message discarded"**

alert: Sends alert messages containing IM session information and message content to the configured system event log. Specify the configured event logging target path and severity level.

Default: disabled
Values: enabled *logTargetReference severity* | disabled
Example: **set alert enabled "services event-log file messages" info**

archiving: Enables or disables archiving of SIP instant messages to the system database. When **enabled**, the system records to its database all instant messages (the text in the body of MESSAGE SIP messages) of sessions matching the policy. Messages are recorded in both directions.

Note that you must enable the **database-write** property of the **vsp** object for archiving to work.

You can view the instant messages that have been archived using the ME Management System **Call Logs** feature.

Default: disabled
Values: enabled | disabled
Example: **set archiving enabled**

pre-stamp: Prepends the user-specified text before the IM message content in this SIP session.

Default: There is no default setting
Example: **set pre-stamp "Good Morning"**

post-stamp: Appends the specified test after the IM message content in this SIP session.

Default: There is no default setting
Example: **set post-stamp "Have a great day"**

message-to-sender: Sets the text message to send back to the originating IM sender in this SIP session.

Default: There is no default setting
Example: **set message-to-sender "Messages to this user are logged."**

message-to-recipient: Sets the text message to send to the IM recipient in this SIP message. This text is in addition to the incoming message.

Default: There is no default setting
Example: **set message-to-recipient "Message is being logged"**

instant-messaging-content

Creates pointers to configured word lists and/or URL lists. For more detailed information on instant message filtering, see Configuring IM Filtering Objects. To create word lists, see the **word-list** object; to create URL lists, see the **url-list** object.

Syntax

```
config vsp default-session-config instant-messaging-content
config vsp policies session-policies policy name rule name session-config
instant-messaging-content
config vsp dial-plan dial-prefix entryName session-config
instant-messaging-content
config vsp dial-plan route name session-config instant-messaging-content
config vsp dial-plan source-route name session-config instant-messaging-content
config vsp session-config-pool entry name instant-messaging-content
```

Properties

word-list: Configures a pointer to a previously configured word list. You can include any number of words lists in your instant messaging content scan.

Default: There is no default setting
Example: **set word-list "vsp im-filtering word-list bad-words"**

url-list: Configures a pointer to a previously configured URL list. You can include any number of URL lists in your instant messaging content scan.

Default: There is no default setting
Example: **set url-list "vsp im-filtering url-list good-guys"**

location-call-admission-control

Customizes call admission control on a per-AOR basis. These parameters can also be configured in the location service **settings** object. From that object, the values are applied across the VSP. The settings override the VSP-wide settings and are applied when an AOR registers. The parameters remain in effect until the next registration

period. Use the show location-cac status provider to view call admission control settings and counters for an AOR.

Syntax

```
config vsp default-session-config location-call-admission-control
config vsp policies session-policies policy name rule name session-config
location-call-admission-control
config vsp dial-plan dial-prefix entryName session-config
location-call-admission-control
config vsp dial-plan route name session-config location-call-admission-control
config vsp dial-plan source-route name session-config
location-call-admission-control
config vsp session-config-pool entry name location-call-admission-control
```

Properties

max-number-of-concurrent-calls: Specifies the maximum number of active incoming and outgoing calls allowed for this AOR at one time. When this value is reached, the connection does not accept calls until the value drops below the threshold.

A value of 0 causes the ME to decline all calls and registrations.

Default: 1000
Values: Min: 0 / Max: 1000000
Example: **set max-number-of-concurrent calls 1500**

max-calls-in-setup: Sets the maximum number of simultaneous inbound and outbound call legs in setup stage that are allowed for this AOR. A call leg in setup is much more compute-intensive than established call legs, so this value is more restrictive than the concurrent call leg value. A value of 0 causes the system to decline all calls and registrations.

Default: 30
Values: Min: 0 / Max: 10000
Example: **set max-calls-in-setup 50**

admission-control: Specifies whether the system considers AOR limitations when forwarding a call *from* the AOR. The system tracks the number of concurrent (both incoming and outgoing) active calls for this AOR. If this property is **enabled**, the system does not forward calls from the AOR if the limit has been reached and instead sends a "603 Declined" message. If **disabled**, the system does forward calls from the AOR. (Set the call limit with the **max-number-of-concurrent-calls** property.) See Admission Control for an AOR for additional information.

Default: disabled
Values: enabled | disabled
Example: **set admission-control enabled**

max-bandwidth: *Secondary property.* Specifies the amount of bandwidth the system allocates to the AOR. When the system reaches the maximum bandwidth limit for a server, it rejects calls until bandwidth use drops below the maximum.

Default: unlimited
Values: unlimited | *kbps*
Example: **set max-bandwidth 512**

emission-control: Specifies whether the system considers AOR limitations when forwarding a call *to* this AOR. The system tracks the number of concurrent (both incoming and outgoing) active calls for the AOR. If this property is **enabled**, the system does not forward calls to the AOR if the limit, set with the

max-number-of-concurrent-calls property, has been reached. Instead, the system sends one of the following messages and drops the call:

- If there is one outbound server/UAC/UAS, the system sends a "486 Busy" message, indicating that the route was resolved but that the AOR was unavailable.
- If there are multiple outbound server/UAC/UASs and all have reached the maximum concurrent calls threshold, the system sends a "486 Busy" message.
- If there are multiple outbound server/UAC/UASs and at least one has not reached the maximum concurrent calls threshold, the return code is determined by the final server that the system attempted to reach. This could be, for example, "486 busy" or a "504 server timeout" if the last server was unresponsive and the transaction timed out.

If **disabled**, the system continues to forward calls to the AOR. See Admission Control for an AOR for more information.

Default: disabled
Values: enabled | disabled
Example: **set emission-control enabled**

call-rate-limiting: Specifies whether the system considers AOR limitations when forwarding a call *to* this AOR. The system tracks the number of concurrent (both incoming and outgoing) active calls for the AOR. If this property is **enabled**, the system does not forward calls to the AOR if the limit, set with the max-number-of-concurrent-calls property, has been reached. Instead, the system sends one of the following messages and drops the call:

- If there is one outbound server/UAC/UAS, the system sends a "486 Busy" message, indicating that the route was resolved but that the AOR was unavailable.
- If there are multiple outbound server/UAC/UASs and all have reached the maximum concurrent calls threshold, the system sends a "486 Busy" message.
- If there are multiple outbound server/UAC/UASs and at least one has not reached the maximum concurrent calls threshold, the return code is determined by the final server that the system attempted to reach. This could be, for example, "486 busy" or a "504 server timeout" if the last server was unresponsive and the transaction timed out.

If **disabled**, the system continues to forward calls to the AOR. See Admission Control for an AOR for more information.

Default: disabled; if set to enabled, the default calls-per-interval is 60, the default interval is 1, and the default result is 486, Busy Here
Values: enabled [*calls-per-interval*][*interval*][*result-code*][*result-string*] | disabled
Example: **set call-rate-limiting enabled 50 1 480 "Temporarily unavailable"**

location-events

Sets the method by which the system implements bridged line appearance (BLA) for a phone. BLA is a feature that allows a phone to display line appearance (status) on more than one phone. A single telephone number (or SIP URL) can be monitored by more than one user agent, and when a call is made to the number, all subscribed user agents can respond. The status of the call is then displayed on all phones mapped to the number. This object sets the mechanism used to do that mapping, either the default SIP mechanism, or the method put forth in the IETF draft, *Implementing Bridged Line Appearances Using Session Initiation Protocol (SIP)*.

Syntax

```
config vsp default-session-config location-events
config vsp policies session-policies policy name rule name session-config
location-events
config vsp dial-plan dial-prefix entryName session-config location-events
config vsp dial-plan route name session-config location-events
config vsp dial-plan source-route name session-config location-events
config vsp session-config-pool entry name location-events
```

Properties

bridged-line-appearance: Sets the type of bridged line appearance the phone uses. Select default for most phones. Select draft-anil-sipping-bla for the following configurations:

- When using a Sylantro phone
- When using sticky trunk ports, where the **registration-plan > route >**
- **alter-contact** property is set to trunk-port-per-aor, -endpoint, or -binding

Default: default
Values: default | draft-anil-sipping-bla
Example: **set bridged-line-appearance draft-anil-sipping-bla**

location-lookup

Customizes the manner in which the ME executes a location lookup. Within this object you can specify which headers the ME uses to perform the lookup, and in what order the fields should be searched on. Use this, for example, to send INVITEs to an AOR registered on the ME but without using the AOR of this device in the Request URI.

Syntax

```
config vsp default-session-config location-lookup
config vsp policies session-policies policy name rule name session-config
location-lookup
config vsp dial-plan dial-prefix entryName session-config location-lookup
config vsp dial-plan route name session-config location-lookup
config vsp dial-plan source-route name session-config location-lookup
config vsp session-config-pool entry name location-lookup
```

Properties

sequence: Selects the header field(s) used to do the lookup in the location cache. Enter as many headers as needed by re-executing the command.

Default: There is no default setting
Values: request-uri | to-uri | from-uri | contact | trunk-port
Example: **set sequence request-uri**

match-uri-params: Specifies which URI parameters should be used in the location lookup comparison. By default, the system does a base comparison of the received contact using the following URI information:

- scheme (i.e., sip, sips, tel)
- user
- host

- port (5060 if not otherwise specified)
- transport (UDP if not otherwise specified)
- maddr

Enter a string for the ME Engine to match against.

Default: There is no default setting
Example: **set match-uri-params state**

location-normalization

Normalizes the Request and/or To URI in a REQUEST message. The ME replaces the URI portion of the specified headers with the AOR data found in the location cache. With the properties of this object you can update the To and Request headers with the full AOR or just the host portion from the AOR in the location cache. This configuration provides a means of normalizing requests destined for an AOR (e.g., phone).

Syntax

```
config vsp default-session-config location-normalization
config vsp policies session-policies policy name rule name session-config
location-normalization
config vsp dial-plan dial-prefix entryName session-config location-normalization
config vsp dial-plan route name session-config location-normalization
config vsp dial-plan source-route name session-config location-normalization
config vsp session-config-pool entry name location-normalization
```

Properties

request-uri: Specifies the portion of the AOR to use in normalizing the Request URI. You can replace the AOR portion of the Request URI with the entire AOR from the location cache or the domain name only (AOR-host-only). By default, no replacement occurs.

Default: none
Values: none | AOR | AOR-host-only
Example: **set request-uri AOR**

to-uri: Specifies the portion of the AOR to use in normalizing the To URI. You can replace the AOR portion of the Request URI with the entire AOR from the location cache or the domain name only (AOR-host-only). By default, no replacement occurs.

Default: none
Values: none | AOR | AOR-host-only
Example: **set to-uri AOR**

log-alert

Enables or disables session logging on a per-call basis. Because session logging can result in large amounts of data, this object allows you to enable logging only on calls that match certain criteria. Note that logging must also be enabled in the **event-log** object for the session details to be written to a target.

Syntax

```
config vsp default-session-config log-alert
```

```
config vsp policies session-policies policy name rule name session-config
log-alert
config vsp dial-plan dial-prefix entryName session-config log-alert
config vsp dial-plan route name session-config log-alert
config vsp dial-plan source-route name session-config log-alert
config vsp session-config-pool entry name log-alert
```

Properties

message-logging: Enables or disables logging of individual SIP messages (INVITE, CANCEL, BYE, etc.) to the system database. Select **no-registers** to log all but SIP REGISTER messages. (This can also be accomplished with the master-services **database** object **sip-register** property, but message-logging is the preferred method.). Message logging can only be enabled if you have the appropriate license for the feature.

The system uses the database records to:

- Display the call detail diagrams at the ME Management System and logs which show the individual SIP messages involved in each session.
- Provide collected messages for detection of attack patterns, which are then used to construct DOS rules blocking the attack traffic.

Default: enabled if you have the license for message logging features and disabled if you do not

- Values: enabled: Logging is active
- disabled: Logging is inactive
- no-reregisters: Logging is active for all but REGISTER messages/sessions
- invite-session-only: Logging is active for only INVITE-based sessions
- filtered: Logging is filtered by **apply-to-methods-for-filtered logs** property.

Example: **set message-logging enabled**

apply-to-methods-for-filtered-logs: Specifies the type of SIP message to be logged if **message-logging** is set to **filtered**. If set to any other option, this property is ignored.

Default: INVITE
Example: **set apply-to-methods-for-filtered-logs INVITE+REFER**

alert: *Secondary property.* Enables or disables the sending of session alert messages to the configured system logging target. If set to **enabled**, specify the path to a previously configured logging target and a severity level. The path must be specified in quotation marks.

Default: disabled
Values: enabled *logTargetReference severity* | disabled
Example: **set alert enabled "services event-log file messages" info**

logging: *Secondary property.* Enables or disables event logging for this session. If session logging is **enabled**, and the **event-log** object enables logging, details are recorded in a target file.If session logging is **disabled**, even if the event-log is enabled, the system does not write session events to the log.

Default: disabled
Values: enabled | disabled
Example: **set logging enabled**

tracing: *Secondary property.* Enables SIP-related tracing for the session. When **enabled**, you can exit to a SIP shell and enter the **trace-filter** command to see related traces. (Note that you must have advanced CLI permissions to execute shell commands.)

Default: disabled
Values: enabled | disabled
Example: **set tracing enabled**

message-auditing: *Secondary property.* Enables the system to maintain an audit trail of changes to each SIP message.

Default: disabled; if enabled, the default severity level is error
Values: enabled *severity* | disabled
Example: **set message-auditing enabled error**

media

Configures the SIP media anchoring settings to apply to this SIP call session.

> **Note:** You must define and enable a pool of ports specifically for media services. Without these ports defined, the ME cannot establish the anchoring necessary to provide media services. See the media-ports object for more information.

RTCP Settings

Real-time Control Protocol (RTCP) is a companion protocol to RTP that gathers statistics on the performance and quality of the SIP call connection. When enabled, RTP monitors the quality of the SIP call and conveys information about the SIP call session. It is based on the periodic transmission of control packets to all participants in the session, and provides feedback on the quality of the data distribution.

RTCP statistics are used to dynamically adjust and optimize the call quality for current network conditions. You can configure the ME to drop, pass, and generate RTCP packets and to record the statistical data in its database through the **media** object.

Media Session Maintenance

The ME provides a session maintenance feature for use in cases of abnormal session terminations. Normally, SIP signaling causes the creation and termination of each media session. In the event of network or device failure, however, terminating SIP messages may not be received by the signaling system, or the signaling system may be unable to request that the resources be released by the media proxy. You can enable the **inactivity-timeout** property to recover resources for aborted media sessions.

When **inactivity-timeout is** enabled, the media proxy periodically checks for inactive media sessions. If a session timed out due to inactivity, a message is sent to the signaling the ME device, which logs the event and sends the appropriate SIP signaling messages to notify each party of the call. The media proxy then releases the resources for the inactive session.

Transcoding Media Types

The ME supports transcoding media types, which is the process of converting media from one CODEC into a different CODEC on output. This allows, in some cases, endpoints supporting different media types to communicate. You add CODECs using the **transcode-media-types** property, augmenting the list of CODECs contained in an

INVITE SDP offer. Note that the order in which a CODEC appears in the offer/answer matters in the ME selection. Original media types appear first, followed by the media types added in this object. You can re-order the media types with the **move** command, or using the **in-codec-preferences** or **out-codec-preferences** objects.

The following table illustrates and explains the transcoding process.

Table 39–3 Transcoding Media Types

Device	CODEC
Phone East	Supports A, B, Z
Phone West	Supports B, C, Z
ME	Configured with transcoding support for C
ME	Does not support Z

As a result, the following sequence occurs:

1. EAST's original INVITE/SDP offering contains CODECs A, B, Z.
2. The ME augments the list with CODEC C, resulting in an offer of A, B, Z, C in the INVITE/SDP.
3. WEST's original OK/SDP response answers with CODECs B, C, Z.
4. The ME modifies and forwards the response to EAST with CODECs B, Z.

The following table shows how the ME behaves with this configuration:

Table 39–4 ME Actions and Reasons

ME Action	Reason
Forwards Z packets in both directions unchanged.	Because CODEC Z is unsupported.
Transcodes C packets from WEST to B on their way to EAST.	Because B is the first supported CODEC in the response (on behalf of WEST) to EAST. That response is comprised of the WEST response minus CODECs that the ME added. (Note that if there are no remaining CODECs after this process, the ME adds back in the first CODEC common to EAST and the system.)
Forwards B packets from either direction, no transcoding necessary.	Because both sides understand B.

Use the **rtp-transcode-stats** and **rtp-transcode-summary** status providers to view active and summary statistics for RTP transcoding.

Syntax

```
config vsp default-session-config media
config vsp policies session-policies policy name rule name session-config media
config vsp dial-plan dial-prefix entryName session-config media
config vsp dial-plan route name session-config media
config vsp dial-plan source-route name session-config media
config vsp session-config-pool entry name media
```

Properties

anchor: Enables or disables SIP media session anchoring on this SIP call session. Media anchoring forces the SIP media session to traverse the system. The **auto** setting enables conditional anchoring. In this case, the system uses its auto-anchoring algorithms to determine anchoring necessity based on a variety of criteria, including whether you have configured smart anchoring via the **autonomous-ip** object and whether the calling devices are behind a firewall.

Default: enabled
Values: enabled | disabled
Example: **set anchor disabled**

transcode-media-types: Specifies the CODEC(s) that the system can use for transcoding media. Adding CODECs through this property augments the list contained in an INVITEs SDP offer. See Transcoding Media Types for a full description and example.

You can enter any number of valid input CODEC types. To see a list of available input CODECs, type a question mark at the command line.

Default: There is no default setting
Example: **set transcode-media-types g726-16**

auto-conference: Allows a user to log into a conference call automatically by prepending the assigned username and passcode before the number to be dialed. To do this, the SIP session establishes the initial call, and then identified DTMF tone strings are used to join the conference, creating the conference codes.

Use the **pre-** and **post-tone-delay** properties to allow announcements to play from the conference site before the digits are entered. Use the VSP **dtmf-generation** object to set parameters for the conference codes created.

Default: disabled, with no regular expression specified and the outbound side hearing the tones

- **Values: administrative state**: Turns on or off the auto-conference feature. When **enabled**, the system strips the Request URI and plays the first portion of the regular expression as tones.
- **regular expression**: Identifies the digit strings in the user portion of the Request URI. All but the last item matched are played as tones after the call comes up. The system replaces the last match as the outgoing user portion in the Request URI.
- **direction**: Specifies which side of the call hears the tone string.
- For example, if the Request URI is 1234567#123#johnd@webex.com, the resulting digit strings played are 1234567# and 123#. The Request URI becomes johnd@webex.com.

Example: **set auto-conference enabled (.*#) (.*#) (.*) out**

pre-tone-delays: Specifies the milliseconds of delay prior to playing the DTMF tone string that was determined from the **auto-conference** property. You can specify multiple delays to correspond to multiple matches of the regular expression. If the number of matches exceeds the number of delay entries, the last delay entry is repeated. For example, if you have three matches in the regular expression, and have configured delays of three and five milliseconds, the system delays 3 milliseconds, plays tone 1, delays 5 milliseconds, plays tone 2, delays 5 milliseconds, plays tone 3.

Default: There is no default setting
Example: **set pre-tone-delays 3**

post-tone-delays: Specifies the milliseconds of delay following the playing of a DTMF tone string that was determined from the **auto-conference** property. You can specify multiple delays to correspond to multiple matches of the regular expression. If the number of matches exceeds the number of delay entries, the last delay entry is repeated. For example, if you have three matches in the regular expression, and have configured delays of three and five milliseconds, the system plays tone 1, delays 3 milliseconds, plays tone 2, delays 5 milliseconds, plays tone 3, delays 5 milliseconds.

Default: There is no default setting
Example: **set post-tone-delays 3**

introduction: Specifies the path to a WAV file that plays at the introduction of a call (no audio is sent through until the introduction completes). Use the **file-play-verify** action to ensure that the recording is of a format supported by the ME device.

Default: There is no default setting
Example: **set introduction /cxc_common/intro1.wav**

music-on-hold: Specifies the path to a WAV file that plays (in a loop) while the call is on hold. Use the **file-play-verify** action to ensure that the recording is of a format supported by the ME device.

Default: There is no default setting
Example: **set music-on-hold /cxc_common/hold1.wav**

inactivity-timeout: Specifies whether the system can timeout an anchored media session due to inactivity. See Media Session Maintenance for more information. If you enable this feature, you must set the length of the inactivity timer. See Setting Time and Time Intervals for information on entry format requirements.

Default: disabled; if set to enabled, the default timer setting is 3600
Values: enabled *seconds* (greater than 60) | disabled
Example: **set inactivity-timeout enabled 1800**

inactivity-style: Specifies which parties of a call must stop sending RTP before the system activates the inactivity timer (if **enabled**). When set to **session**, the system activates the timer when all parties stop sending RTP. When set to **per-call-leg**, if one party stops sending RTP, the system activates the inactivity timer.

Default: session
Values: session | per-call-leg
Example: **set inactivity-style per-call-leg**

monitor: Associates a playback configuration with the call session. The playback function allows you to record SIP calls for playback on the ME Management System or a configured endpoint. Enter a pointer to a previously configured **monitor-group** object.

Default: There is no default setting
Example: **set playback "vsp monitor-group callRecord"**

packet-marking: Enables or disables packet marking. Marking (tagging) a packet provides a quality of service (QOS) indicator, which routers along the path may act on. You could use packet marking, for example, to give priority to voice calls over other traffic. The system writes the value you enter to the TOS (or DiffServ) field of the IP header.

Default: 0xa0
Values: disabled | tos *value* (0-255)
Example: **set packet-marking tos 128**

rtp-stats: Enables or disables the collection and logging of RTP and call quality statistics to the system database. Note that this property must be **enabled** to:

- Display Mean Opinion Score (MOS) or Quality of Service (QoS) statistics for a call at the ME Management System.
- Display RTP values in accounting files or databases that the system is writing to.

Default: disabled
Values: enabled | disabled
Example: **set rtp-stats enabled**

*rtcp <action>***<true | false>**: Specifies the handing and generation of RTCP packets in this SIP call session. When configuring this property, you set an action for call senders and a logging capability. Note that this property is not available when performing transcoding. See the Release Notes for more information.

Set an action that defines how the system responds to RTCP packets it receives from call senders.

Default: pass false
- **Values: pass**: Transmits all packets.
- **drop**: Drops any packets.
- generate-only-if-required: Generates RTCP packets if it detects that the sender is not generating them.
- **generate-always**: Always generates packets, regardless of whether the sender did.

Configure whether to log session statistics to the system database:

- **true**: System writes RTCP statistics to the database, along with its own statistics.
- **false**: System ignores statistics.

Example: **set rtcp generate-only-if-required false**

mirror: Specifies whether calls that match the defined policy are mirrored to other boxes in the cluster. To use this feature you must also set **mirror-media-settings** to true in the **cluster** object.

Default: enabled
Values: enabled | disabled
Example: **set mirror disabled**

answer-media-loopback: Sets whether the system answers a loopback call. When **enabled**, the system answers the call and generates RTP according to the negotiations. When **disabled**, the system allows the call to proceed according to the rest of the configuration. Endpoints can use these loopback calls to test the quality of the media transport, in accordance with the IETF draft-ietf-mmusic-media-loopback-07.txt.

Default: disabled
Values: enabled | disabled
Example: **set answer-media-loopback enabled**

tag-routing: Specifies whether tag routing is in use for media. If routing tags are configured for an interface (using the **ip > routing-tag** property), and that interface has media configured on it, the tags are only used when this property is **enabled**. See Tag-Based Route Selection for more information.

Default: enabled
Values: enabled | disabled
Example: **set tag-routing disabled**

encode-auto-anchor-tag: *Secondary property*. Specifies whether the content of the x-cxc-info field of the SDP is encoded or in clear text. This property is only applicable if the **anchor** property is set to **auto**. The x-cxc-info field contains the information

necessary for the system to make an auto anchoring decision. If set to **true**, the field is base-64 encoded. If set to **false** it is sent in clear text.

Default: true
Values: true | false
Example: **set encode-auto-anchor-tag false**

transcode-balance-ptime: *Secondary property.* Specifies whether the system uses signaling to attempt to "coax" the originating phone to send RTP packets at the rate of the destination phone. When transcoding, RTP packets may be arriving at departing at different rates (as determined by the CODEC in use with the phone). When **true**, the system sends the originating phone the ptime value (interval) in use by the destination phone.

Default: true
Values: true | false
Example: **set transcode-balance-ptime false**

transcode-auto-release: *Secondary property.* Specifies whether the system passes packets without transcoding, thereby releasing the transcode license for those sessions. This only applies if the system is set to auto anchor (with the anchor property set to auto) and transcode (using transcode-media-types property) and auto anchoring is required due to reachability issues between the source and destination. When this property is set to **true** in that situation, if the source and destination have the same set of CODECs, then the system passes the packets without transcoding. If **false**, the system transcodes the packets, costing the license two sessions for the duration.

Default: true
Values: true | false
Example: **set transcode-auto-release false**

decode-telephone-events: *Secondary property.* Specifies whether the system should decode DTMF packets and inject them into the audio stream. This property may be used in cases where only one of the endpoints supports DTMF per *RFC 2833, RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals.* If **false**, the default, the phone is responsible for inserting the DTMF into the audio stream. If **true**, the system decodes the DTMF packets and inserts the resulting audio into the stream.

Default: false
Values: true | false
Example: **set decode-telephone-events true**

repair-empty-codec-list: *Secondary property.* Specifies whether the system repairs an SDP that contains an illegal media description. When set to **true**, the system inserts a default well-known CODEC into the SDP, allowing the phone to process the message. It also sets the port to 0 (if it is not already), disabling the media stream. When set to **false**, the system makes no changes to the SDP.

This property only applies to illegal video and audio media types. The system inserts payload type 34 (H263) for video and payload type 0 (PCMU) for audio as the default payload types.

Default: false
Values: true | false
Example: **set repair-empty-codec-list true**

strip-blocked-stream: *Secondary property*. Specifies whether the system removes the m= line from the SDP when all CODECs within that line are blocked. CODECs can be blocked by several mechanisms: media-type filtering, CODEC preferences, and stripping of unplayable or unverifiable CODECs. If the system blocks all CODECs in an m= line, it disables the stream. If this property is set to **true**, the system removes the entire m= line from the SDP. If set to **false**, the stream is disabled (by setting the port to

zero) but remains in the SDP with some CODECs. This property is only applicable in messages that contain multiple media streams (e.g., audio and video).

Default: false
Values: true | false
Example: **set strip-blocked-stream true**

preserve-sdp-order: *Secondary property*. Sets whether the system attempts to preserve the SDP attribute order. When set to **true**, the system makes a best attempt to preserve the order. When **false**, the default, the system uses the order native to its SDP parser.

Default: false
Values: true | false
Example: **set preserve-sdp-order true**

handle-unknown-lines-in-sdp: *Secondary property.* Specifies how the system handles errored lines that it receives in an SDP. If set to **strip**, the default, the system removes the offending lines and forwards the packet. If set to **pass**, the system forwards the packet untouched. If set to **error**, the system sends a 488 message (Not Acceptable) back to the sender and logs a message to the event log.

Default: strip
Values: strip | pass | error
Example: **set handle-unknown-lines-in-sdp error**

rtp-min-consecutive: *Secondary property.* Specifies the number of consecutive RTP packets the system must receive to establish an RTP source as valid.

Default: 3
Example: **set rtp-min-consecutive 5**

rtp-sequence-discontinuity: *Secondary property.* Specifies whether the system monitors for, detects, and corrects RTP sequence number discontinuity. In some cases, a gateway may change the CODEC for a packet, but keep the same synchronization source (SSRC). If the resulting sequence numbers are discontinuous, it causes problems for SRTP processing. When this property is **enabled**, the system changes the SSRC if it detects sequence problems. When **disabled**, it does nothing.

Default: disabled
Values: enabled | disabled
Example: **set rtp-sequence-discontinuity enabled**

rtp-splice: *Secondary property.* Configures a mechanism for maintaining RTP parameters. When **enabled**, the ME attempts to maintain RTP parameters when it injects DTMF (or other RTP audio) into the RTP stream. This may result in additional processing on each RTP packet after the normal audio stream is resumed, but it is required for interoperability with some endpoints because they do not recognize the DTMF when RTP parameters change (SSRC, sequence numbers, and timestamps). Note that the **rtp-stats** property must be enabled for this property to work.

Default: disabled
Values: enabled | disabled
Example: **set rtp-splice enabled**

combine-recording-fragments: *Secondary property.* Specifies whether the system checks the state of the RTP recording file when a call ends. If **enabled**, when an ME-recorded call ends, the media master determines whether the RTP recording file is fragmented across the cluster. (Fragmenting can occur when a failover causes part of the recording to reside on one box and part on another.) If there are fragments, the system copies each to the master box and assembles the entire file. When **disabled**, the system bypasses fragment checking, which boosts performance.

Default: enabled

Values: enabled | disabled
Example: **set combine-recording-fragments disabled**

auto-anchor-consider-nat: *Secondary property.* Specifies whether to disable a portion of the ME anchoring algorithm. This property is only applicable if you have the **anchor** property set to **auto**. Typically, if the system is forwarding a call from behind a NAT, it would anchor the media stream. You may set this property to **disabled** if, for example, you have a phone behind a NAT destined for a server that can do NAT traversal and you want to release the media stream.

Default: enabled
Values: enabled | disabled
Example: **set auto-anchor-consider-nat disabled**

default-session-bandwidth: *Secondary property.* Specifies an initial bandwidth value to use when calculating the bandwidth usage for each leg of a media session. The resulting value (in accumulation with all other session values) determines whether a server pool **server** has reached the configured **max-bandwidth** setting.

Note that the bandwidth usage value is based not on the actual traffic on the wire, but on a calculation done by the ME device. The calculation uses the value associated with the first known CODEC identified in the SDP for a usage rate. If there is not a known CODEC, or the value has not yet been determined from the SDP, the system uses this setting.

If you set this property to zero, the system treats media streams with no known codecs in the SDP as using zero bandwidth. (They do not count against the bandwidth limit for a given server.)

Default: 87
Values: Min: 0 / Max: 10000
Example: **set default-session-bandwidth 31**

SIP-response-code-on-media-resource-alloc-failures: *Secondary property.* Specifies the SIP response code to send when there is a media allocation failure. The failure may occur for one of two reasons: either media ports are not configured on the interface or no ports are available because all configured media ports are in use.

Default: 488 (Service Unavailable)
Values: Min: 300 / Max: 699
Example: **set SIP-response-code-on-media-resource-alloc-failures**

attributeless-auto-anchor: *Secondary property.* When enabled in conjunction with the anchor-mode=auto, the ME attempts to auto-anchor streams without additional ME attributes in the SDP.

Default: disabled
Values: enabled | disabled
Example: **set attributeless-auto-anchor enabled**

release-provisionally-anchored-media: Release media resources that have been provisionally anchored.

When **media > anchor** is set to auto, the ME attempts to determine when anchoring resources can be released based on IP addresses and routing-tags of the communicating endpoints. When these determinations cannot be made, the media is deemed "provisionally anchored." In releases previous to Release 3.5.5, provisionally allocated media was released by default.

Default: false
Values: true | false
Example: **set release-provisionally-anchored-media false**

report-last-timestamp: When enabled, the ME reports the timestamp of the last received media packet.

Default: disabled
Values: enabled | disabled
Example: **set report-last-timestamp enabled**

monitor-rfc-2833: Specifies whether to have the ME change SSRC when it detects RTP sequence number discontinuity on active SSRC.

Default: disabled
Values: enabled | disabled
Example: **set monitor-rfc-2833 enabled**

discard-media-on-hold: *Secondary property.* Specifies whether to discard media from an endpoint when that endpoint is placed on hold.

Default: disabled
Values: enabled | disabled
Example: **set discard-media-on-hold enabled**

pass-candidate-attributes: Specifies whether or not ICE candidate SDP attributes received by the ME are forwarded to an endpoint.

Default: disabled
Values: enabled | disabled
Example: **set pass-candidate-attributes enabled**

propagate-reinvite-from-header: When enabled, when the ME receives an Invite request, the ME switches to the new From: header when it is different in a reinvite. When disabled, the ME uses the From: header received in the initial Invite.

Default: disabled
Values: enabled | disabled
Example: **set propagate-reinvite-from-header enabled**

dtmf-detected-events: Specifies whether received DTMF events are reported to web services.

Default: disabled
- Values: disabled: Provides the normal DTMF handling based on the **session-config > dtmf-preferences** settings or the older legacy **dtmf-translation** settings.
- report-and-forward: DTMF events are reported to web services and translated and forwarded based on the **session-config > dtmf-preference** settings or the older legacy **dtmf-translation** settings.
- report-and-discard: DTMF events are reported to web services and then discarded without being translated.

Example: **set dtmf-detected-events report-and-discard**

augmented-ice: Specifies whether the ME attempts augmented ICE.

Default: disabled
Values: enabled | disabled
Example: **set augmented-ice enabled**

dtls-passthru-latching: *Secondary property.* Specifies whether ME latches media based on observed DTLS traffic.

Default: enabled
Values: enabled | disabled
Example: **set dtls-passthru-latching disabled**

dtls-passthru-threshold: *Secondary Property*. Specifies required number consecutive DTLS messages from same origin before media latch.

Default: 2
Values: Min: 0 | Max: 4294967296
Example: **set dtls-passthru-threshold 5**

media-scanner-settings

Configure media scanner settings. When media-scanner-settings are enabled, the media-scanner is started after the outgoing call connects and the pre-scan-time has elapsed. The media-scanner monitors the signal strength and duration of the received audio to divide it into intervals.

Syntax

```
config vsp default session-config media-scanner-settings
config vsp session-config-pool entry <name> media-scanner-settings
config vsp policies session-policies policy <name> rule <name> session-config
media-scanner-settings
```

Properties

admin: Enables or disables the media scanner settings for play-file-broadcast.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

pre-scan-time: The number of milliseconds to delay before invoking the media scanner for speaker detection.

Default: 20
Values: Min: 0 / Max: 4294967295
Example: **set pre-scan-time 35**

max-scan-time: The maximum number of milliseconds before canceling media scanning due to timeout.

Default: 30000
Values: Min: 0 / Max: 4294967295
Example: **set max-scan-time 25000**

low-threshold: Enter the quiet signal power threshold in dbs.

Default: -36
Values: Min: -36 / Max: 3
Example: **set low-threshold -25**

high-threshold: Enter the talk or tone signal power threshold in dbs.

Default: -36
Values: Min: -36 / Max: 3
Example: **set high-threshold -25**

low-long-duration: The number of milliseconds of detected quiet before declaring a long-pause.

Default: 2000
Values: Min: 0 / Max: 4294967295
Example: **set low-long-duration 1500**

high-long-duration: The number of milliseconds of detected talk or tone before declaring a long-talk or stable-tone.

Default: 900
Values: Min: 0 / Max: 4294967295
Example: **set high-long-duration 1500**

averaging-window: *Secondary property* The window of time used when calculating signal strength.

Default: 100
Values: Min: 10 / Max: 1000
Example: **set averaging-window 500**

nominal-rounding-factor: *Secondary property.* The signal strength is rounded to the nearest multiple of the value you enter for this property.

Default: 2
Values: Min: 1 / Max: 25
Example: **set nominal-rounding-factor 4**

media-type

Sets the media types that are allowed and/or prohibited during the session. You select a media type: audio, video, application, or MIME: and then a specific subtype. Use the question mark character at the command line to see a list of available subtypes. For example:

```
config media-type> set allowed-media-types audio ?
 allow sessions to use these media types
 syntax: set allowed-media-types audio sub-type
         set allowed-media-types video sub-type
         set allowed-media-types application sub-type
         set allowed-media-types custom-mime-type mime-type sub-type
 any
 pcmu
 gsm
 g723
 dvi4
 lpc
 pcma
 g722
--more--
```

In addition to the pre-configured options, you can allow or block custom types that may be part of your enterprise.

Syntax

```
config vsp default-session-config media-type
config vsp policies session-policies policy name rule name session-config
media-type
config vsp dial-plan dial-prefix entryName session-config media-type
config vsp dial-plan route name session-config media-type
config vsp dial-plan source-route name session-config media-type
config vsp session-config-pool entry name media-type
```

Properties

allow-media-types: Sets the media types allowed during the session. Re-execute the command for each type you want to allow.

Default: The default subtype setting for audio, video, and application is any; there is no default setting for custom-mime-type.
Values: audio *subType* | video *subType* | application *subType* | custom-mime-type *mimeType subType*
Example: **set allowed-media-types custom-mime-type application safe-trade**

blocked-media-types: Sets the media types to prohibit during the session. Re-execute the command for each type you want to block.

Default: The default subtype setting for audio, video, and application is any; there is no default setting for custom-mime-type.
Values: audio *subType* | video *subType* | application *subType* | custom-mime-type *mimeType* subType
Example: **set blocked-media-types video mpls**

media-verify-config

Enables or disables media verification for RTP sessions, and sets SIP session termination based on RTP and alert messaging. The **media-verify-config** object allows you to verify RTP and RTCP media streams negotiated over SIP sessions. The settings verify that the media traffic passing through the ME matches the negotiated and legal scheme for CODECs (coder/decoders) operating on SIP signals.

Syntax

```
config vsp media-verify-config
```

Properties

admin: Enables or disables the current **media-verify-config** object. If **enabled**, the system uses these settings if this object is included in the session configuration media object.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

terminate-session: Enables or disables SIP session termination if a RTP media verification alert is generated. If **enabled**, the system terminates a media session completely upon the first media verification error (B2B mode only).

Default: enabled
Values: enabled | disabled
Example: **set terminate-session disabled**

alert-frequency: Sets the number of milliseconds the system should wait between generation of media alert messages, which indicate a media verification error.

Default: 1000
Values: Min: 10 / Max: 10000
Example: **set alert-frequency 2000**

multimedia-stream-settings

Configures multimedia stream-specific settings when configuring the multimedia stream server (MSS) process on the ME.

Syntax

```
config vsp session-config-pool entry name multimedia-stream-settings
config vsp default-session-config multimedia-stream-settings
```

Properties

stream-name: Specify how the stream name is derived. You can enter a string or keep the default value, session-id.

Default: session-id
Example: **set set-stream-name session-id2**

in-leg-permissions: Sets the stream's read and write permissions for the in-leg.

Default: read-write

- Values: read-only: The stream is read-only.
- write-only: The stream is write-only.
- read-write: The stream is read and write enabled.

Example: **set in-leg-permissions read-only**

out-leg-permissions: Sets the stream's read and write permissions for the out-leg.

Default: read-write

- Values: read-only: The stream is read-only.
- write-only: The stream is write-only.
- read-write: The stream is read and write enabled.

Example: **set out-leg-permissions read-only**

to-header-specification: Select the To: header provided to the streaming code.

Default: to-header
Example: **set to-header-specification to-header**

invert-uri-index: When enabled, the ME swaps the publish and subscribe indices in the RTMP URIs.

Default: disabled
Values: enabled | disabled
Example: **set invert-uri-index enabled**

40

Configuring Session Configuration Objects N Through Z

This chapter covers session configuration objects, N through Z, alphabetically. For session configuration objects A through M, see "Configuring Session Configuration Objects A Through M." The session configuration objects define the way in which the Media Engine (ME) handles SIP-based signaling and media traffic. The session configuration that is applied to an active call through the ME depends on configuration of other aspects of the system.

There are several places in the configuration hierarchy through which you can access the session configuration objects. The path to these object defines in which cases the ME uses that configuration. Locations for session configuration are defined in the following table.

Table 40–1 Session Configuration Locations

Path	Defines...
vsp > default-session-config	The session configuration settings to apply to those SIP calls for which there are no configured policies. See Configuring Default Session Configuration Objects, for more information.
vsp > policies > session-policies > policy > rule	The session configuration settings to apply to SIP calls for which a configured policy exists. See Configuring Policy Objects, for more information.
vsp > dial-plan > dial-prefix **vsp > dial-plan > route** **vsp > dial-plan > source-route**	The session configuration settings to apply to calls based on the dial prefix or domain suffix. See Configuring Dial Plan Objects, for more information.
vsp > calling-group > route **vsp > calling-group > source-route**	The session configuration settings to apply to calling-group member calls based on the dial prefix or domain suffix. See Configuring Calling Group Objects, for more information.
vsp > session-config-pool > entry	A saved session configuration that can be referenced within one or more dial plans. See Configuring Session Configuration Pool Objects, for more information.

named-variable-collector

The ME supports a generic database used to hold named variables. A named variable is a variable paired with a value through the reg-exp header code. This allows you to modify SIP message fields and CDR fields more generically.

This object allows you to configure the ME to look at any SIP message header, and by applying regular expressions, extract any part of the header and store the value in the specified named variable table. You can apply these settings to both requests and responses.

Under this object you assign the named variable collector a number and a name. This is also where you create the variable. To create a variable, first select the header type you want to serve as the source of the data. This can be either any valid SIP message header. Then specify a regular expression to run against the value of the source header. Finally, specify the replacement expression to apply when there is a match. If you want to append a string to the existing named-variable value, appending is done the same way as creating.

Other things you can configure under the **named-variable-collector** object include the SIP methods to apply variables, whether the ME applies variables to responses, which type of dialog variables should be applied, if a CSEQ field needs to be matched, and what actions to take on the expression if there is not a complete match.

When a session is created, a named variable list is automatically created. If any named variables are configured in the **default-session-config**, they populate this list. All variable names in the named variable list must be unique.

The ME updates the named variable list when any of the following happens.

- When the session configuration **merge-object** is set to **merge**, the named variables configured in the new session config are appended to the existing named variable list.
- When the session configuration **merge-named-variables** is set to **replace**, the existing session configuration named variables are replaced by the newly configured named variables.
- When the **header-settings > named-variable-collector** collects new named variables via the reg-exp code.
- When you specify or create a named variable list and a named variable of the same name already exists, the ME overwrites the value of the existing variable name.

Note: Variable names cannot start with a “$” and you should not use special characters such as “\”, “%”, “#”, “!”, “?”, “[“, “]”, :&”, “{“, “}”, “@” when naming variables.

Syntax

```
config vsp default-session-config header-settings named-variable-collector
```

Properties

admin: Enables or disables named variable collection on the ME.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

number: Specifies an indexed number for this entry.

Default: There is no default setting
Values: Min: / Max: 4294967296
Example: **set number 10**

named-variable: Specifies the named-variable you are creating or modifying. The value of this variable will be the replacement string.

Default: There is no default setting
Example: **set named-variable var1**

create: Identifies the header containing the data to be modified and the value assigned to the named-variable. First, select the header that serves as the source of the data. This can be **To**, **From**, or **Request**. Then specify a regular expression to run against the value of the source header. Finally, supply the replacement expression to apply if there is a match.

Default: There is no default setting
Example: **set create To (.*) \1;\1>**

append: Identifies the header containing the data to be added to the named-variable value. First, select the header that serves as the source of the data. This can be To, From, or Request. Then specify a regular expression to run against the value of the source header. Finally, supply the replacement expression to apply if there is a match. The ME appends this string to the existing named-variable value. To add spaces or commas, include them using quotation marks.

Default: There is no default setting
Example: **set append From (.*) \2;\2>**

apply-to-methods: Specifies the message type from which the ME extracts the specified named-variable value.

Default: INVITE
Example: **set apply-to-methods REGISTER**

apply-to-responses: Specifies whether to apply header value changes to SIP requests or both requests and responses. When set to **no**, changes are only applied to requests. When set to **yes**, changes are applied to both requests and responses.

Default: no
- Values: no
- yes [*response-code*]
- both [*response-code*]

Example: **set apply-to-responses both 500**

apply-to-dialog: Specifies whether to apply header value changes to a specific dialog or not.

Default: both
- Values: inbound: Apply to the inbound dialog only
- outbound: Apply to the outbound dialog only
- both: Apply to both inbound and outbound dialogs.

Example: **set apply-to-dialog inbound**

cseq: *Secondary property*. Sets a mechanism to further filter which SIP messages have the header expression modifications applied. If this property is set to **0** (the default), the ME applies the changes to all SIP messages. If set to any other value, the ME only applies the changes to SIP messages having a CSEQ field that matches that value.

Default: 0
Values: Min: 0 / Max: 4294967296
Example: **set cseq 100**

create-on-failed-match: *Secondary property*. Construct a create header even when the expression is not a complete match.

Default: true
Values: true | false
Example: **set create-on-failed-match false**

append-on-failed-match: *Secondary property*. Execute the append action event even when the create expression fails to match.

Default: true
Values: true | false
Example: **set append-on-failed-match false**

named-variables

The **named-variables** object, configured on a per-session basis, creates a static list of named variables that can be used in several features throughout the ME configuration. You create named variables under the **session-config** object and you can create a named variable list for each configured session.

Syntax

```
config vsp session-config-pool entry named-variables
config vsp default-session-config named-variables
```

Properties

named-variable: Enter the name of the named variable for this session.

Default: There is no default setting
Example: **set**

value: Enter the value of the named variable for this session.

Default: There is no default setting
Example: **set value 100**

merge-object: Specifies whether this new set of session configuration named variables overwrites the previous set of session configuration named variables, or if it is appended to the previous set.

Default: merge

- Values: merge: Append to the existing set of session configuration named variables.
- replace: Overwrite the existing set of session configuration named variables with the new one.

Example: **set merge-object replace**

nat-traversal

Specifies whether symmetric Real-time Transport Protocol (RTP) is applied to the session. When a SIP call passes through far-end NAT to reach the call recipient, the complications of NAT (or a firewall) create problems for call connections.To address this, you can configure the ME to run symmetric RTP.

RTP is a packet-based communication protocol that adds timing and sequence information to provide end-to-end network transport functions for applications transmitting real-time data, such as audio or video. With symmetric RTP on, the ME

sends return RTP messages based on the source IP address and UDP port in received RTP messages. NAT only modifies data in the IP header: the Session Description Protocol (SDP) payload is left unchanged. By using the source IP address and UDP port from the received RTP message, the ME sends traffic back to the NAT device, instead of the untranslated addresses in the SDP message.

Syntax

```
config vsp default-session-config media nat-traversal
config vsp policies session-policies policy name rule name session-config media
nat-traversal
config vsp dial-plan dial-prefix entryName session-config media nat-traversal
config vsp dial-plan route name session-config media nat-traversal
config vsp dial-plan source-route name session-config media nat-traversal
config vsp session-config-pool entry name media nat-traversal
```

Properties

admin: Enables or disables the NAT traversal configuration on this ME device. When **enabled**, the system uses the settings configured here; when **disabled** the system uses the default NAT traversal settings.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

symmetricRTP: Specifies whether to use symmetric RTP for SIP call sessions. It specifies how the system learns the public address in preference to the private address (behind the firewall). When **true**, the system uses the address and port numbers that it learns from received RTP packets as the destination address for the endpoint. When set to **false**, the system uses the address and port numbers that it learns from the SDP message as the destination address for the endpoint.

Default: false
Values: true | false
Example: **set symmetricRTP true**

asymmetric-rtp-address: Specifies addresses that are not compatible with symmetric RTP. When an address that matches this property is received in an SDP message, the system disables symmetric RTP when sending RTP to that endpoint. This property is only necessary in certain situations (e.g., when using the BroadWorks Media Server), and only when the **symmetricRTP** property is set to true.

Default: There is no default setting
Example: **set asymmetric-rt-address 192.168.10.10/32**

out-codec-preferences

Sets a preference for CODECs, influencing the ME ordering of CODECs in the SDC on the outbound leg of a call. The ME removes those CODECs with a zero priority from the SDP. CODEC preferences do not cause the ME to add a CODEC to the SDP, but to remove and/or reorder existing CODECs according to their priority. The ME places a CODEC whose priority is not specified in its original order, just ahead of the known auxiliary CODECs (e.g., telephone-events). This is not a mechanism to add CODECs into an SDP, only to order those that are already there (via transcoding or from the original offer/answer).

Syntax

```
config vsp default-session-config out-codec-preferences
config vsp policies session-policies policy name rule name session-config
out-codec-preferences
config vsp dial-plan dial-prefix entryName session-config out-codec-preferences
config vsp dial-plan route name session-config out-codec-preferences
config vsp dial-plan source-route name session-config out-codec-preferences
config vsp session-config-pool entry name out-codec-preferences
```

Properties

preferences<media-type>[codec][priority]: Assigns a priority to a given CODEC for outbound audio or video sessions.

Default: There are no default settings

- **Values: media type**: Select for audio or video. The associated CODEC (subtype) is preferred according to the priority for that media type.
- **codec**: The CODEC to which the priority applies. Use the question mark character at the command line to see a list of available CODECs, or enter any CODEC name.
- **priority**: Sets a preference for the CODEC. The lower the number, the more preferred the CODEC. Assigning a priority value of zero disables the CODEC for the session. The system removes these CODECs before sending the SDP offer or answer. (0-100)

Example: **set preferences video g729 1**

out-encryption

Sets the parameters for outbound encrypted media sessions when the ME is anchoring the call. This is the portion from the ME to the call recipient. With this object you set the encryption requirements on the call, and the encryption method used. Because the ME does not always know on the outbound leg the encryption method expected by the recipient (because that recipient is not in the registry), you must manually set the type of encryption to offer.

Note About RFC-1889 Encryption Type

When using RFC 1889, Microsoft clients do not typically do encryption unless it is offered as mandatory in the SDP by one of the clients. If you want encryption on the outbound side, you must set the **mode** property to **require**.

Syntax

```
config vsp default-session-config out-encryption
config vsp policies session-policies policy name rule name session-config
out-encryption
config vsp dial-plan dial-prefix entryName session-config out-encryption
config vsp dial-plan route name session-config out-encryption
config vsp dial-plan source-route name session-config out-encryption
config vsp session-config-pool entry name out-encryption
```

Properties

mode: Specifies the encryption requirements on the outgoing call (from system to endpoint). The method of encryption used is determined by the **type** property.

Default: none

- **Values: none**: The system disables the encryption put forth by the outbound endpoint (i.e., it responds "no" to the encryption portion of the authentication handshake.) If the inbound endpoint requires encryption, the call is dropped.
- **pass-thru**: The system passes cryptographic parameters through the box and does not participate in RTP encryption. This method renders some advanced media services unusable (in particular, recording, announcements, transcoding, call monitoring, RTP stats, media verification, and RTCP generation). When using this option, set mode to **pass-thru** in both **in-encryption** and **out-encryption**.
- **allow**: If the outgoing endpoint offers encryption to the ME device, the system answers with it. If the endpoint does not offer encryption, the system does not answer with or initiate encryption.
- **follow**: If the inbound endpoint offers encryption, the system offers encryption (the type set by policy) to the outbound endpoint.
- **offer**: The system offers encryption to the outbound endpoint when the session is first established.
- **reoffer**: The system offers encryption to the outbound endpoint whether the message is an INVITE or a REINVITE. This setting is most applicable when an endpoint issues a REINVITE, and encryption was not required with the original INVITE. In this case, the system will again offer encryption when forwarding the message.
- **require**: The call must come in with encryption specified or the system drops it.

Example: **set mode allow**

type: Sets the type of encryption on outbound sessions. In choosing a type, the system uses the encryption expected by that device or application.

Default: RFC-3711

- **Values: Linksys**: Uses SIP INFO messages to exchange mini-certificates and exchange a symmetric key. The media encryption is similar to RFC-3711, but done with AES-128 Countermode and with HMAC MD5 authentication. See the **linksys** action for more certificate information.
- **RFC-1889**: Use encryption as defined in *RFC 1889, RTP: A Transport Protocol for Real-Time Applications*. This mode is used for compatibility with Windows Messenger and Microsoft Office Communicator, neither of which currently support RFC-3711 encryption. Instead, it uses a DES-CBC encryption of the entire UDP payload (including RTP headers) with no authentication. Note that to enable outbound encryption when using FRC 1889, you must set **mode** to **require**.
- **RFC-3711**: Use encryption as defined in *RFC 3711, The Secure Real-time Transport Protocol (SRTP*).

Example: **set type RFC-1899**

require-tls: Specifies the requirements of the signaling protocol for a call outbound leg. It defines whether the system offers SRTP over a non-secure (TCP or UDP) signaling connection. The action of this property depends on the setting of the **mode** property.

Most phones follow *RFC 4568, SDP Security Descriptions for Media Streams*, and thus require that this property be set to **true**.

Default: false

- **Values: true**: The system only offers encryption when talking to a TLS client. If TLS and SRTP are required (**mode** is set to **require**), the system fails calls going to TCP/UDP clients. If the mode property is set to **offer** or **follow**, the system forwards the call without SRTP.

- **false**: The system offers SDP messages according to the mode setting without regard for the signaling transport. This allows keys to be exchanged in an insecure message.

Example: **set require-tls true**

priority-AES-128-CM-HMAC-SHA1-32: *Secondary property*. Sets a preference for 32-bit SHA1 authentication tags on out going calls. The system supports (offers) both 32- and 80-bit authentication tags on ingress. A value of 0 disables support for the 32-bit tag.

Default: 1
Values: Min: 1 (most preferred) / Max: 5
Example: **set priority-AES-128-CM-HMAC-SHA1-32 2**

priority-AES-128-CM-HMAC-SHA1-80: *Secondary property*. Sets a preference for 80-bit SHA1 authentication tags on out going calls. The system supports (offers) both 32- and 80-bit authentication tags on ingress. A value of 0 disables support for the 80-bit tag. To enable, set a value between 1 (most preferred) and 5.

Default: 2
Values: Min: 1 (most preferred) / Max: 5
Example: **set priority-AES-128-CM-HMAC-SHA1-80 1**

mki-length: *Secondary property*. Provides support for the optional Master Key Identifier bit defined in *RFC 3711, The Secure Real-time Transport Protocol (SRTP)*. The value specify sets the number of bytes in the MKI. The system then sends the negotiated identifier of that length indicating which master key to use for decryption with each SRTP packet. Note that the endpoint must support this option.

Default: 0
Values: Min: 0 / Max: 4
Example: **set mki-length 2**

mikey-offer-location: *Secondary property.* Controls where in the SDP the system stores the MIKEY offer (the "a=key-mgmt:mikey" line) when it is made. If MIKEY is offered to the system, it puts the MIKEY answer in the location where the offer was located. Set the location as the media descriptor or session level.

Default: session
Values: session | media-descriptor
Example: **set mikey-offer-location media-descriptor**

mikey-time-tolerance: *Secondary property.* Controls where in the SDP the system stores the MIKEY offer (the "a=key-mgmt:mikey" line) when it is made. If MIKEY is offered to the system, it puts the MIKEY answer in the location where the offer was located. Set the location as the media descriptor or session level.

Default: 60
Example: **set mikey-time-tolerance 90**

symmetric-address-failure: *Secondary property.* Specifies whether the system learns the source IP address from the RTP/RTCP packets, even if the packets fail decryption. When **enabled**, the first packet in a particular stream that fails SRTP decryption causes a DroppedPacket notification to be sent to the application with the address of the packet. The application treats this like a srcIPChanged notification.

Default: disabled
Values: enabled | disabled
Example: **set symmetric-address-failure enabled**

treat-as-secure: *Secondary property*. Specifies whether a proprietary security indicator is used on the SIP interface; either ST-secure or ST-insecure. This setting only operates on

the X-Siemends-Call-Type header when MIKEY encryption is involved in pass-thru mode.

Default: disabled

- Values: disabled: Sets ST-insecure
- enabled: Sets ST-secure
- auto: If SRTP is active on both sides of the call, the ME allows the X-Siemends-Call-Type header to pass unchanged. If SRTP is not active on other side of the call, the header is set to ST-insecure. The "auto" value sets the interface to trusted.

Example: **set treat-as-secure enabled**

encryption-preferences: Creates a prioritized list of encryption types to offer or accept.

> **Note:** Always give DTLS a priority of 1 and RFC3711 a priority of 2.

Default: There is no default setting.
Example: **set encryption-preferences DTLS 1**

out-dtmf-preferences

Configures the ME's out-leg DTMF method preferences.

Syntax

```
config vsp default-session-config out-dtmf-preferences
config vsp session-config-pool entry out-dtmf-preferences
```

Properties

admin: Specifies whether or not this DTMF preference list is applied to calls matching this session configuration.

Default: disabled
`Values: enabled | disabled`
Example: **set admin enabled**

preferences: Allows you to configure supported dtmf-types and assign them with a priority to determine the ME's preferences.

Default: audio 1
First select a DTMF method. The available DTMF methods are:

- Values: audio
- rfc-2833
- sip-info-dtmf
- sip-info-dtmf-relay
- sip-notify
- h245-alphanumeric
- h245-signal
- q931

Then assign it a priority. This can be from 0-100. A value of **0** means the method is not supported. The lower the priority, the more preferred the DTMF method.

Example: **set preferences rfc2833 2**

out-dtmf-settings

The ME can be configured to translate one DTMF method to another. Through this object, you can control the length of play and pause time and volume for the digits that the ME plays on the out-leg.

Syntax

```
config vsp default-session-config in-dtmf-settings
config vsp session-config-pool entry out-dtmf-settings
```

Properties

digit-volume: Specifies the volume setting for the DTMF tones. The digit volume is measured in decibel (dB) of the measured power referenced to one milliwatt, measured at a zero transmission level point. The smaller the dBm0, the louder the volume.

Default: -20
Values: Min: -36 / Max: 0

digit-duration: Specifies the length of time, in milliseconds, that the ME plays each DTMF digit.

Default: 750
Values: Min: 100 / Max: 10000
Example: **set digit-duration 1000**

min-digit-duration: Specifies the minimum length of time, in milliseconds, that the ME plays each DTMF digit. If a DTMF event has a duration less than this value, the **digit-duration** property overrides the duration and is used to play the DTMF event.

Default: 60
Values: Min: 5 / Max: 100
Example: **set min-digit-duration 75**

max-digit-duration: Specifies the maximum length of time, in milliseconds, that the ME plays each DTMF digit. If a DTMF event has a duration greater than this value, the **digit-duration** property overrides the duration and is used to play the DTMF event.

Default: 2000
Values: Min: 100 / Max: 10000
Example: **set max-digit-duration 3000**

inter-digit-duration: Specifies the length of time, in milliseconds, that the ME pauses between playing each digit.

Default: 250
Values: Min: 0 / Max: 1000
Example: **set inter-digit-duration 500**

pause-duration: Specifies the length of time, in milliseconds that the ME pauses when it encounters a comma character in the conference code. The comma is a special character in the conference code that indicates a specified time the ME must wait before playing the next tone.

Default: 3000
Values: Min: 500 / Max: 10000

Example: **set pause-duration 4000**

minimum-duration: Specifies the minimum time, in milliseconds, between detecting RFC-2833 events.

Default: 60
Values: Min: 0 / Max: 1000
Example: **set minimum-duration 100**

as-audio: Specifies whether the ME sends audio or DTMF packets to the conference server when representing conference code tones. When true, the ME encodes the sound in the current CODEC. When false, the ME sends DTMF packets.

Default: true
Values: true | false
Example: **set as-audio false**

out-dtmf-translation

Secondary object. Controls the method used for forwarding DTMF tones in a call. The two supported methods are via the signaling stream using SIP INFO messages or via a DTMF packet that is in compliance with RFC 2833, RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals. Using this object, you can configure the ME to:

- Forward packets in the form they arrived.
- Pick them out of the RTP stream and send them in a SIP INFO message (if they arrived as DTMF packets).
- Extract them from a SIP INFO message and send them as DTMF packets (if the SIP INFO message contained a DTMF body).

Outbound DTMF translation applies to the segment from the ME to the call recipient.

Syntax

```
config vsp default-session-config out-dtmf-translation
config vsp policies session-policies policy name rule name session-config
out-dtmf-translation
config vsp dial-plan dial-prefix entryName session-config out-dtmf-translation
config vsp dial-plan route name session-config out-dtmf-translation
config vsp dial-plan source-route name session-config out-dtmf-translation
config vsp session-config-pool entry name out-dtmf-translation
```

Properties

info: Specifies the method to use for forwarding DTMF tones that were received in a SIP INFO message. If set to **info**, the system forwards the message as it was received (in an INFO message). If set to **rfc-2833**, the system extracts the DTMF body from the INFO message and sends the content via DTMF packets.

Default: info
Values: info | rfc-2833
Example: **set info rfc-2833**

drop-info: Specifies whether to drop the SIP INFO message if the info property is set to rfc-2833. If set to **true**, the system drops the INFO packet and only sends the DTMF packets. If set to **false**, the system sends both.

Default: false
Values: true | false

Example: **set drop-info true**

rfc-2833: Specifies the method to use for forwarding DTMF tones that were received in DTMF packets. If set to **rfc-2833**, the system forwards the message as it was received (in DTMF packets). If set to **info**, the system extracts the DTMF packets from the RTP stream and sends the content via a SIP INFO message. The system sends one INFO message per event detected.

Default: rfc-2833
Values: info | rfc-2833

drop-rfc-2833: Specifies whether to drop the DTMF packets if the rfc-2833 property is set to info. If set to **true**, the system drops the RFC 2833 packets and only sends the SIP INFO message. If set to **false**, the system sends both.

Default: false
Values: true | false
Example: **set drop-rfc-2833 true**

info-dtmf-body: Specifies the body type to use in a SIP INFO message when converting from RFC 2833 format. When using **dtmf**, the message body contains just single character (the digit that was pressed). When set to **dtmf-relay**, the body contains the single character plus duration data.

Default: dtmf-relay
Values: dtmf-relay | dtmf
Example: **set info-dtmf-body dtmf**

timeout-rfc-2833: *Secondary property*. Sets the number of milliseconds the system waits before sending a SIP INFO message if it does not detect the end of the event. The timer is started at the start of an event. This property only applies when the forwarding method has been changed from **rfc-2833** to **info**, and is used in the event that when monitoring DTMF, the system does not detect an event end.

Default: 1000
Example: **set timeout-rfc-2833 1500**

out-echo-cancellation-settings

Configures outbound server-side acoustic echo cancellation for devices running telephony applications that do not have functional on-board echo cancellation.

Syntax

```
config vsp session-config-pool entry name out-echo-cancellation-settings
config vsp default-session-config out-echo-cancellation-settings
```

Properties

admin: Enables or disables echo cancellation on outbound call-legs.

Default: disabled
Values: enabled | disabled
Example: **set**

platform-delay: Although the maximum echo tail allowed by the echo cancellation algorithm is 200 ms, it is possible that the actual echo tail for certain devices is longer. This parameter allows you to account for any extra delay in addition to the 200 ms. For example, if your total acoustic delay is 500 ms, set **echo-tail** to 200 ms and **platform-delay** to 300 ms. This covers the entire 500 ms delay.

Default: 0

Values: Min: 0 / Max: 65535
Example: **set platform-delay 100**

echo-tail: The amount of time, in milliseconds, that it takes for the person speaking to hear his or her own echo. This is also known as echo delay. The maximum echo tail supported by the ME's algorithm is 200 ms.

Default: 16
Values: Min: 0 / Max: 200
Example: **set echo-tail 150**

noise-reduction: Allows you to reduce background noise. A value of 0 means there is no noise reduction and 5 is the maximum reduction.

Default: 2
Values: Min: 0 (no noise reduction) / Max: 5
Example: **set noise-reduction 4**

out-hold-translation

See the **in-hold-translation** object for a complete description.

Syntax

```
config vsp default-session-config out-hold-translation
config vsp policies session-policies policy name rule name session-config
out-hold-translation
config vsp dial-plan dial-prefix entryName session-config out-hold-translation
config vsp dial-plan route name session-config out-hold-translation
config vsp dial-plan source-route name session-config out-hold-translation
config vsp session-config-pool entry name out-hold-translation
```

out-leg-tos

This object determines the ToS value setting for the out-leg of the session. This ToS value determines the quality of service that the call receives. You can either preserve the ToS value in the first received message of the session or overwrite the ToS field of all packets sent out on the out-leg with the value you specify. Enter a number that represents the 8-bit Differentiated Services (DS) field of the IP packet in decimal format, such as 26 for 00011010 or 104 for 01101000. This value can be of use to upstream devices.

Syntax

```
config vsp default-session-config h323-tos-settings out-leg-tos
config vsp session-config-pool entry h323-tos-settings out-leg-tos
```

Properties

value: Specify the ToS value setting for the out-leg of the session.

Default: preserve; if you select overwrite, the default value is 0

- Values: preserve: The ME uses the ToS value in the first received message if the session.
- overwrite <*value*>: The ME marks the ToS field of all packets it sends out on the out-leg with the value you specify.

Example: **set value overwrite 22**

out-media-normalization

Changes the media descriptor string (e.g., the CODEC for audio or video) in the SDP. Use this in cases where a client is unable to understand a variation in name of a CODEC/media descriptor. For example, G729 is sometimes transmitted as G729a. Outbound media normalization applies to the segment from the ME to the call recipient.

Syntax

```
config vsp default-session-config out-media-normalization
config vsp policies session-policies policy name rule name session-config
out-media-normalization
config vsp dial-plan dial-prefix entryName session-config out-media-normalization
config vsp dial-plan route name session-config out-media-normalization
config vsp dial-plan source-route name session-config out-media-normalization
config vsp session-config-pool entry name out-media-normalization
```

Properties

normalize: Specifies, for a media type, how to normalize a CODEC/media descriptor name. The initial subtype is the type the system matches on and replaces. The alternate subtype is the type that the system then inserts in the SDP to replace the initial subtype. You can select a pre-configured type or enter a custom type.

Default: audio
Values: <audio | video | application | image | custom-mime-type *mimeType*> [*initialSubType*] [*alternateSubType*]
Example: **set normalize video g729 g729a**

out-media-loss-detection

Configures out-leg media loss detection settings.

Syntax

```
config default-session-config out-media-loss-detection
config session-config-pool entry <name> out-media-loss-detection
```

Properties

admin: Enable or disable media loss detection on the outgoing call leg.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

interval: Set the interval, in seconds, in which to check for media loss on the outgoing call leg.

Default: 5
Example: **set interval 3**

out-media-scanner-settings

Configure out-media-scanner-settings. When in-media-scanner-settings are configured, a media scanner is started on the out-leg of the call and reports events based on the analysis of the received audio from the endpoint.

Syntax

```
config vsp default session-config out-media-scanner-settings
config vsp session-config-pool entry <name> out-media-scanner-settings
config vsp policies session-policies policy <name> rule <name> session-config
out-media-scanner-settings
```

Properties

admin: Enables or disables the in-media scanner settings.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

pre-scan-time: The number of milliseconds to delay before invoking the in-media scanner. This property is not applicable for the on-demand media scanner.

Default: 20
Values: Min: 0 / Max: 4294967295
low-threshold: Enter the talk or stable tone signal power threshold in dbs.

Default: -36
Values: Min: -36 / Max: 3
Example: **set low-threshold -25**

high-threshold: Enter the quiet signal power threshold in dbs. Crossing this threshold indicates talking or stable tone.

Default: -36
Values: Min: -36 / Max: 3
Example: **set high-threshold -25**

low-long-duration: The number of milliseconds of detected quiet before reporting a long-pause, otherwise a short pause is reported.

Default: 2000
Values: Min: 0 / Max: 4294967295
Example: **set low-long-duration 1500**

high-long-duration: The number of milliseconds of detected talk or tone before reporting a long talk or stable tone, otherwise a short talk is reported.

Default: 900
Values: Min: 0 / Max: 4294967295
Example: **set high-long-duration 1500**

averaging-window: *Secondary property* The amount of time in dbs used when calculating signal strength.

Default: 100
Values: Min: 10 / Max: 1000
Example: **set averaging-window 500**

nominal-rounding-factor: *Secondary property.* The signal strength is rounded to the nearest multiple of the value for this property before comparing against other signal strengths.

Default: 2
Values: Min: 1 / Max: 25
Example: **set nominal-rounding-factor 4**

event-report-frequency: The number of milliseconds the media scanner should wait between generation of media scanner events. Setting this property to 0 causes the media scanner events to be reported immediately as they occur.

Default: 1000
Values: Min: 0 / Max: 60000
Example: **set event-report-frequency 39**

event-report-count-threshold: The maximum number of media scanner events that can be pended for waiting for the event-report-frequency timer to expire before being reported. If the number of queued media scanner events reaches this count, all of the events will be immediately reported.

Default: 25
Values: Min: 1 / Max: 6000
Example: **set event-report-count-threshold 73**

event-report-flags: Set the media scanner events to report. The options are:

Default: report all events
Values: short-pause, long-pause, short-talk, long-talk, stable-tone
Example: set **event-report-flags short-pause**

out-ice-settings

Configures the ME's ICE settings on the out-leg.

Syntax

```
config vsp default-session-config out-ice-settings
config vsp session-config-pool entry <name> out-ice-settings
```

Properties

admin: Enables or disables ICE on the out-leg.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

connectivity-check-time-out: Specifies the time in ms before a STUN connectivity check times out.

Default: 100
Values: Min: 0 | Max: 4294967296
Example: **set connectivity-check-time-out 75**

connectivity-check-max-retransmits: Specifies the number of times the ME retransmits ICE STUN connectivity checks before labeling a candidate pair as Failed.

> **Note:** To achieve maximum interoperability with Chrome, set this value to no less than 200.

Default: 7
Values: Min: 0 | Max: 255
Example: **set connectivity-check-max-retransmits 5**

delay-stun-responses: *Secondary property*. When enabled, the ME does not respond to STUN until the 200 OK is received.

Default: disabled
Values: enabled | disabled
Example: **set delay-stun-responses enabled**

suppress-re-invites: *Secondary property*. When enabled the ME does not send a re-INVITE when ICE completes successfully.

Default: disabled
Values: enabled | disabled
Example: **set suppress-re-invites enabled**

nomination-policy: Specify the RFC-5245 nomination type.

Default: regular
Values: regular | aggressive
Example: **set nomination-policy aggressive**

delay-stun-requests: *Secondary property*. When enabled, connectivity checks do not begin until a STUN request is received.

Default: disabled
Values: enabled | disabled
Example: **set delay-stun-requests enabled**

trickle-ice: Enable or disable trickle ICE on the out-leg.

Default: disabled
Values: enabled | disabled
Example: **set trickle-ice enabled**

terminate-call-on-ice-failure: *Secondary property*. When enabled, a call is terminated if ICE fails.

Default: disabled
Values: enabled | disabled
Example: **set terminate-call-on-ice-failure enabled**

out-msrp-session-leg

Configures out-leg MSRP interworking.

Syntax

```
config default-session-config out-msrp-session-leg
config session-config-pool entry <name> out-msrp-session-leg
```

Properties

admin: Enable or disable MSRP interworking on this call leg.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

msrp-leg-transport: Specify the MSRP transport method for RCS or WebRTC.

Default: TCP
Values: TCP | TLS | WS | WSS
Example: **set msrp-leg-transport TLS**

connection-reuse: Not supported in this release.

default-media-interface: Specify the local media interface to use for an MSRP connection if svc-routing fails to locate the appropriate interface.

Default: There is no default setting
Example: **set default-media-interface int1**

use-mdesc-cline-first: (*Advanced*) Specify whether the MSRP session manager attempts to use the SDP c-line (before using the path attribute) to learn the remote MSRP endpoint's media IP address.

Default: false
Values: true | false
Example: **set use-mdesc-cline-first false**

socket-read-size: (*Advanced*) Specify the MSRP socket read size to use when assembling incoming MSRP messages.

Default: 4096
Min: 0 / Max: 4294967296
Example: **set socket-read-size 5000**

partial-forward-size: (*Advanced*) Specify the threshold for forwarding buffered MSRP message content bytes.

Default: 1024
Min: 0 / Max: 4294967296
Example: **set partial-forward-size 2050**

connsrc-match-path: (*Advanced*) Specify whether the ME allows incoming MSRP connections even when a remote address does not match the SDP path attribute.

Default: false
Values: true | false
Example: set connsrc-match-path false

allow-missing-fingerprint: (*Advanced*) Specify whether the ME allows an MSRP secure connection even when the SDP fingerprint attribute is missing.

Default: false
Values: true | false
Example: **set allow-missing-fingerprint false**

outbound-controls

Customizes call admission control, on a per-AOR basis, for outbound calls only. This is in contrast to the admission control settings found in the **location-call-admission-control** object, which sets aggregate inbound and outbound limits. Use the **show location-cac** status provider to view call admission control settings and counters for an AOR.

Syntax

```
config vsp default-session-config location-call-admission-control
outbound-controls
config vsp policies session-policies policy name rule name
session-config location-call-admission-control outbound-controls
config vsp dial-plan dial-prefix entryName session-config
location-call-admission-control outbound-controls
config vsp dial-plan route name session-config
location-call-admission-control outbound-controls
config vsp dial-plan source-route name session-config
location-call-admission-control outbound-controls
config vsp session-config-pool entry name
location-call-admission-control outbound-controls
```

Properties

max-number-of-concurrent-calls: Specifies the maximum number of active outgoing calls allowed for this AOR at one time. When this value is reached, the connection does not accept calls until the value drops below the threshold.

A value of **0** causes the system to decline all calls and registrations.

Default: 1000
Values: Min: 0 / Max: 1000000
Example: **set max-number-of-concurrent calls 1500**

max-calls-in-setup: Sets the maximum number of simultaneous outbound call legs in setup stage that are allowed for this AOR. A call leg in setup is much more compute-intensive than established call legs, so this value is more restrictive than the concurrent call leg value. A value of 0 causes the system to decline all calls and registrations.

Default: 30
Values: Min: 0 / Max: 10000
max-bandwidth: *Secondary property.* Specifies the amount of bandwidth the system allocates to outbound calls to the AOR. When the system reaches the maximum bandwidth limit for a server, it rejects calls until bandwidth use drops below the maximum.

Default: unlimited
Values: unlimited | *kbps*
Example: **set max-bandwidth 512**

call-rate-limiting: *Secondary property.* Limits the number of calls sent from the AOR within a certain interval. Once this interval is reached, the system rejects any calls from this AOR, returning a response code and message until the rate decreases. This feature sets the acceptable set-up rate for incoming calls.

If **enabled**, set the number of calls allowed and the measurement interval (in seconds). You can also enter a result code from 400 to 699 and a text string to accompany call rejection if no available server is found.

Default: disabled; if set to enabled, the default calls-per-interval is 60, the default interval is 1, and the default result is 486, Busy Here
Values: enabled [*calls-per-interval*][*interval*][*result-code*][*result-string*] | disabled
Example: **set call-rate-limiting enabled 50 1 480 "Temporarily unavailable"**

outbound-sip-messages

Configures events for outgoing SIP messages. This is a secondary object.

Syntax

```
config vsp session-config-pool entry name event-settings outbound-sip-messages
config vsp default-session-config event-settings outbound-sip-messages
```

Properties

admin: Enables or disables events for outgoing SIP messages.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

apply-to-methods-for-events: Select the SIP methods you want the ME to create events for.

Default: There is no default setting
Example: **set apply-to-methods-for-events INVITE**

apply-to-responses: Specifies whether to send events for SIP requests or both requests and responses. When **no**, changes are applied to requests only. When **yes**, changes are applied to requests and responses. If **yes**, you must set the response code to which it applies. A value of 0 implies all responses.

Default: no
Values: no | yes <*response code*>
Example: **set apply-to-responses yes 0**

apply-to-dialog: Specifies whether to send events for SIP messages on a specific dialog or not.

Default: both
- Values: inbound: Apply to the inbound dialog only
- outbound: Apply to the outbound dialog only
- both: Apply to both the inbound and outbound dialogs.

Example: **set apply-to-dialog inbound**

cseq: *Secondary property*. Sets a mechanism to further filter for which SIP messages have events sent. When set to **0** (the default), the ME sends events for all SIP messages. If set to any other value, the ME only sends events for SIP messages having a CSEQ field that matches that value.

Default: 0
Values: Min: 0 / Max: 4294967296
Example: **set cseq 100**

overflow-route

Allows the ME to override the dial plan route once a configurable threshold has been reached. Under this object you can configure the session limit and the peer to use once the limit has been reached.

Syntax

```
vsp session-config-pool entry overflow-route
vsp default-session-config overflow-route
```

Properties

admin: Enables or disables the overflow route feature. When enabled, the ME overrides the dial plan route once a configurable threshold has been reached.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

limit: Enter the session limit threshold to be reached before the overflow route overrides the dial plan route.

Default: 0
Values: Min: 0 / Max: 20000

Example: **set limit 150**

peer: Specify the peer to which you want traffic forwarded once the session limit threshold has been reached. Enter the peer **type** and **address**.

Default: none

- Values: none
- server
- carrier
- exchange
- switch
- trunk
- hunt-group
- calling-group
- virtual-dial-plan

Example: **set peer server "vsp\enterprise\servers\sip-gateway RedirectClusters**

p-asserted-identity-uri-specification

Specifies what derives the content of the fields of the outgoing P-Asserted-Identity URI when the ME transmits a message. For example, if the **user** property is set to **to-uri**, the system replaces the user field of the P-Asserted-Identity URI with data from the user field of the incoming TO URI. If set to **omit**, the user field is left blank. Or, you can enter any string that you want placed in the user field. See Altering URIs for information on replacement options.

Syntax

```
config vsp default-session-config p-asserted-identity- uri-specification
config vsp policies session-policies policy name rule name session-config
p-asserted-identity-uri-specification
config vsp dial-plan dial-prefix entryName session-config
p-asserted-identity-uri-specification
config vsp dial-plan route name session-config
p-asserted-identity-uri-specification
config vsp dial-plan source-route name session-config
p-asserted-identity-uri-specification
config vsp session-config-pool entry name p-asserted-identity- uri-specification
```

Properties

user: Specifies how to derive the value of the user field of the P-Asserted-Identity URI.

Default: same-uri

Values: same-uri | request-uri | to-uri | from-uri | omit | next-hop | *string*

Example: **set user from-uri**

host: Specifies how to derive the value of the host field of the P-Asserted-Identity URI.

Default: same-uri

Values: same-uri | request-uri | to-uri | from-uri | omit | next-hop | next-hop-domain | local-ip | *string*

Example: **set host request-uri**

port: Specifies how to derive the value of the port field of the P-Asserted-Identity URI.

Default: same-uri
Values: same-uri | request-uri | to-uri | from-uri | omit | *string*
Example: **set port omit**

display: Specifies how to derive the value of the display field of the P-Asserted-Identity URI.

Default: same-uri
Values: same-uri | request-uri | to-uri | from-uri | omit | next-hop | *string*
Example: **set display to-uri**

transport: Specifies the value of the transport field of the P-Asserted-Identity URI. In addition to using the value from other fields of the incoming URI, you can set the transport method to UDP, TCP, TLS, or the method used by the next-hop server.

You cannot enter a string for this property.

Default: same-uri
Values: same-uri | request-uri | to-uri | from-uri | omit | UDP | TCP | TLS | next-hop
Example: **set transport next-hope**

user-param: Specifies whether the User parameter in the P-Asserted-Identity URI of the SIP header is maintained or removed when the system forwards a message. If set to **keep**, the message is forwarded with the parameter as it was received. If set to **omit**, the entire user=*param* is removed from the P-Asserted-Identity URI.

Default: omit
Values: omit | keep
Example: **set user-param keep**

uri-parameter <*name*><*value*>: Appends the specified user parameter and value to the P-Asserted-Identity URI for matching calls. The resulting parameter is added to the URI. For example, the example below would result in a P-Asserted-Identity URI that looked similar to:

<sip:jane@fun.com;BTG=trunk1>

You can append multiple users.

Default: There is no default setting
Example: **set uri-parameter BTG trunk1**

create: Sets whether to create a P-Asserted-Identity header if one does not already exist. If **true**, the system uses the property settings in this object to determine the value of the fields in the header. If set to **false**, the system does not create the header. If a header does already exist, this field has no effect. (The fields of the existing header are still manipulated by the values of the object's properties.)

Default: false
Values: true | false
Example: **set create true**

use-original-from-header: Specifies where the content of the P-Asserted-Identity header is derived from. (This property is only applicable if the **create** property is set to **true**.) When **use-original-from-header** is **true**, the system uses the From header as it existed, before any normalization occurred, as the p-asserted-identity header. If set to **false**, the system uses the From header as it exists when SIP call processing occurs (e.g., post normalization).

Default: false
Values: true | false
Example: **set use-original-from-header true**

peer

Sets the next-hop destination server where the ME forwards a SIP messages. If this object is not set, calculations merged from the dial plan, location cache, and other policy settings control how the ME forwards a message. These settings override the dynamic calculations, and effect the routing calculation that determines the next-hop server.

Syntax

```
config vsp default-session-config peer
config vsp policies session-policies policy name rule name session-config peer
config vsp dial-plan dial-prefix entryName session-config peer
config vsp dial-plan route name session-config peer
config vsp dial-plan source-route name session-config peer
config vsp session-config-pool entry name peer
```

Properties

peer: Specifies to which server the system should forward the call. Enter the name of a previously configured server or group of the type specified.

Default: automatic

- **Values: trunk**: Enter a reference to a carrier\exchange\switch **trunk-group** and optionally, a rate plan.
- **switch**: Enter a reference to a carrier exchange **dial-plan** and optionally, a rate plan.
- **exchange**: Enter a reference to a carrier **dial-plan** and optionally, a rate plan.
- **server**: Enter a reference to an enterprise **dial-plan**.
- **calling-group**: Enter a reference to a calling-group **dial-plan**.
- **server-member**: Enter a reference to a server-pool **server-pool-admission-control**.
- **uac**: Allows you to specify any device, using the contact string that may be found in the location binding. To use the contact string in the location binding, use the **show location-cache -v** command. If the contact address is marked as **true**, it is stored as secure, using a SIPS: indicator.
- **automatic**: Sets the system to automatically route the session. In this case, the system bases routing decisions on settings in other objects (e.g., location cache, dial plan, and other policy settings).
- named: Identifies the next-hop server by matching on enterprise server carrier and endpoint tags. If multiple matches occur, the ME hunts through all matches. To identify the server, enter the carrier tag used for an enterprise **dial-plan** and the endpoint tag used for the **server-pool > server-pool-admission-control**. If the tags configured here do not match configured tags, no server is set as the next-hop and the ME responds with a 404 (notfound) message.

Example: **set peer uac sip:cov2075556789@elmaple.com true**

hunt-timeout: Specifies the number of seconds to wait before determining a server is unavailable and trying the next configured server.

Default: 30
Example: **set hunt-timeout 25**

periodic-announcement

Specifies a WAV file that will be periodically inserted into a call. Use this to enter a recording tone or hold message. Use the **file-play-verify** action to ensure that the recording is of a format supported by the ME device.

Syntax

```
config vsp default-session-config media periodic-announcement
config vsp policies session-policies policy name rule name session-config media
periodic-announcement
config vsp dial-plan dial-prefix entryName session-config media
periodic-announcement
config vsp dial-plan route name session-config media periodic-announcement
config vsp dial-plan source-route name session-config media periodic-announcement
config vsp session-config-pool entry name media periodic-announcement
```

Properties

file: Specifies the location of the system of a WAV file containing the announcement.

Default: There is no default setting
Example: **set file /cxc/recordings/announce1**

period: Specifies the number of seconds the system waits between insertions of the announcement into the call. Note that the system starts its timer at the beginning of the announcement. The file starts playing every *period* number of seconds, regardless of the length of the recording. This means that you will hear the announcement continuously if the file, or specified **duration** (below), are longer than the specified **period**.

Default: 30
Values: Min: 10 / Max: 3600
Example: **set period 45**

duration: Specifies, in milliseconds, how much of the specified recording to play. If you specify 0, the system plays the recording in its entirety. Use this with a value set to insert a tone file, and set the number of milliseconds that would not be too intrusive.

Default: 0
Example: **set duration 500**

playback-call-settings

Enables or disables playback of the last recorded SIP call from a specific To/ From pair. Playback can be initiated using the **dial-prefix** object or any other type of policy that would trigger this object through the session configuration.

> **Note:** To use this feature, you must enable the anchor and record properties in the media object under the default session configuration.

For example, you could configure the **dial-prefix** to recognize *73. When a call came in with that prefix in the To or From URI of the SIP header, it would trigger the session configuration associated with that dial prefix plan. With this object **enabled**, the *73 would initiate playback of the last call. Therefore,

To:*73bob@phone.com

From:joe@phone.com

results in the ME playing back, to joe@phone.com, the last call exchange that was from joe@phone.com and to bob@phone.com. The ME searches the database for this call and initiates a call back to SIP phone joe@phone.com, playing back the recording of the previous call instead of connecting a new one.

You can also configure the ME to play back the last call recorded call from a phone, regardless of the destination. For joe@phone.com to hear his last recorded call, he would initiate a call to himself using the configured **dial-prefix** (*73joe@phone.com).

Syntax

```
config vsp default-session-config playback-call-settings
config vsp policies session-policies policy name rule name session-config
playback-call-settings
config vsp dial-plan dial-prefix entryName session-config playback-call-settings
config vsp dial-plan route name session-config playback-call-settings
config vsp dial-plan source-route name session-config playback-call-settings
config vsp session-config-pool entry name playback-call-settings
```

Properties

playback-last-call: Configures the system to playback the last call between the To/From call pair instead of initiating a new call. Note that the anchor and record properties must be enabled in the **media** object of the default session configuration for recording to take place.

Default: disabled
Values: enabled | disabled
Example: **set playback-last-call enabled**

pre-call-authorization

Configures the call access mechanism in the ME. When configured and enabled, the ME terminates matching calls at the box. The system then collects the DTMF digits it receives until either the timeout expires, the verify signal is received, or maximum number of digits has been reached. After digit collection is terminated, the ME compares the digits to its list of authorized numbers for that caller. If the collected digit string matches an entry in that list, the ME plays a success message, if configured, and completes the call. If the collected digit string does not match an entry in that list, the ME plays the failure message, if configured, and terminates the call. In addition, while digits are being collected, if the ME encounters the configured “cancel” digit, the system terminates the call. If the “restart” digit is enabled and encountered, all previously entered digits are removed, and collection continues.

Syntax

```
config vsp default-session-config pre-call-authorization
config vsp policies session-policies policy name rule name
session-config pre-call-authorization
config vsp dial-plan dial-prefix entryName session-config
pre-call-authorization
config vsp dial-plan route name session-config pre-call-authorization
config vsp dial-plan source-route name session-config
pre-call-authorization
config vsp session-config-pool entry name pre-call-authorization
```

Properties

admin: Sets the administrative status of the configuration entry for terminating incoming calls at the box.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

greeting-message: Sets the message that is played to incoming calls if they have been terminated. Enter a path name to the WAV file containing the message.

Default: There is no default setting
Example: **set greeting-message /cxc_common/preauth.wav**

max-number-of-digits: Specifies the maximum number of digits the ME collects before it begins evaluating the string against the authorized list (set with the **authList** property).

Default: 7
Values: Min: 1 / Max: 16
Example: **set max-number-of-digits 10**

verify: Configures a signal (telephone key pad key) that the ME waits for before it begins evaluating the string against the authorized list (set with the **authList** property). Use this to allow, for example, variable length PINs. Configure a pound (#) symbol to indicate end of collection and end PIN entries with a pound.

Default: disabled
Values: enabled *telephoneKey* | disabled
Example: **set verify enabled Pound**

cancel: Configures forced termination of a call. When **enabled**, sets a signal (telephone key pad key) that, if encountered in DTMF collection, forces the ME to terminate the call.

Default: disabled
Values: enabled *telephoneKey* | disabled
Example: **set cancel enabled Star**

restart: Configures a collection restart. When **enabled**, sets a signal (telephone key pad key) that, if encountered in DTMF collection, forces the ME to wipe whatever digits have been collected and restart collection.

Default: disabled
Values: enabled *telephoneKey* | disabled
Example: **set restart enabled Flash**

inter-digit-timeout: Configures a length of time the ME waits after button pushing has stopped and before it begins evaluating the string against the authorized list (set with the **authList** property). The inter-digit timer is reset every time a digit is received. When it times out, the preceding digits will be verified. This is used, for example, when entering a PIN. The number is not verified until the caller stops entering; after the inter-digit time out expires the ME checks the string.

Default: 2500
Values: Min: 500 / Max: 60000
Example: set inter-digit-timeout 5000

success-message: Sets the message that is played to the caller indicating that the call attempt has been successful. This is played if the collected digits match an entry in the authorized list (set with the **authList** property). Enter a path name to the WAV file containing the message.

Default: There is no default setting
Example: **set success-message /cxc_common/callSuccess.wav failure-message**: Sets the message that is played to the caller indicating that the call attempt was not successful. This is played if the collected digits do not match an entry in the authorized list (set with the **authList** property). Enter a path name to the WAV file containing the message.

Default: There is no default setting
Example: set failure-message /cxc_common/callFail.wav

authList: Adds strings of digits to a list of authorized calling numbers. The ME uses this list to match against the collected digits and determine whether a call is allowed. Re-execute the command to enter multiple strings.

Default: There is no default setting
Example: **set authList 011**

presence

Configures presence services for this SIP call session. SIP call messages containing SUBSCRIBE and NOTIFY requests and responses are associated with a presence session.

The ME stores presence information in its local databases. Presence information pertains to your SIP presence (online, away, etc.). This object enables presence services and controls whether the ME translates that presence information from one server type to another (because every server type has its own way of storing/transmitting this data).

Syntax

```
config vsp default-session-config presence
config vsp policies session-policies policy name rule name session-config presence
config vsp dial-plan dial-prefix entryName session-config presence
config vsp dial-plan route name session-config presence
config vsp dial-plan source-route name session-config presence
config vsp session-config-pool entry name presence
```

Properties

presence-services: Enables or disables system presence services on this SIP call session. If set to **enabled**, the session information is sent to the system presence database. If **disabled**, the system allows packets through but does not write any information to the database.

Default: disabled
Values: enabled | disabled
Example: **set presence-services enabled**

presence-translation: Sets the server types, and therefore the translation direction, that the system performs. The system modifies an incoming request so that the far-end client receives presence data in a recognizable format.

Default: disabled

- **Values: disabled**: The request is passed through unmodified
- **sametime-to-lcs**: Provides Sametime-to-LCS translation
- **lcs-to-sametime**: Provides LCS-to-Sametime translation
- **lcs-to-lcs**: Ensures LCS-to-LCS presence data compatibility.

- **st-to-st**: Ensures Sametime-to-Sametime presence data compatibility. This would be for use, for example, when using different versions of Sametime.
- **mcs-to-mcs**: Ensures Nortel MCS-to-MCS presence data compatibility, for example, when different versions are running.
- **voice-to-voice**: Indicates that the entity is a phone/voice (to support phones that want to SUBSCRIBE to a SIP server and announce presence). Identifying the entity as voice prevents the system from altering the SUBSCRIBE message in the presence database.
- **voice-to-lcs**: Enables forwarding of phone registrations to LCS (mostly useful in CSTA cases) to control OC client presence information.

Example: **set presence-translation sametime-to-lcs** *presence-mapping <value><mapTo>*: Configures, for a given presence status, the value to map that status to. Use this property if both clients do not recognize the same presence states. In an LCS-to-Sametime configuration, for example, if status is set to a value of "Out to Lunch" with the LCS client, it must be mapped to an option that Sametime recognizes.

Default: There is no default setting
Values: offline | away | out-to-lunch | on-the-phone | be-right-back | busy | do-not-disturb | online
Example: **set presence-mapping out-to-lunch away**

voice-lcs-transport: Sets the default protocol used to transmit voice-to-LCS SIP packets to the destination LCS server.

Default: any
Values: any | UDP | TCP | TLS
Example: **set voice-lcs-transport UDP**

federation-contact: Specifies the string that the system uses to create the SIP header URIs it sends to remote servers. The string must match the content of the Common Name field that is present in the certificate that the system provides to the LCS to prove its authenticity.

Default: There is no default setting
Example: **set federation-contact companyABC.com**

message-body-replace: Alters the body of any SIP message (for example, IM content or the SDP for an INVITE) for a matching session. Use this property with caution; you would only change the SIP message body under specific required circumstances.

In the example below, the system replaces sip:1002@company.com in the SIP message body with sip:9788231002@company.com.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Example: **set message-body-replace "(.*)sip:(\d{4})@company.com(.*)""\1sip:978823\2@company.com\3"**

st-keep-alives: Specifies whether the system responds to Sametime keep alives. When **enabled**, the system responds to keep alives on behalf of the federated domain, letting the Sametime server know that the remote peer is active. Use this in federated configurations using a version of Sametime that requires SIP keep alives. Otherwise, this property should be **disabled**.

Default: disabled
Values: enabled | disabled
Example: **set st-keep-alives enabled**

location-presence-service: Specifies whether the ME publishes device presence information to the **jtapi** master service. For Oracle Technical Support use only.

Default: disabled
Values: enabled | disabled
Example: **set location-presence-service enabled**

provisional-response

The provisional-response object allows you to add any additional provisional responses you want sent along after the "100 Trying" response at the start of a normal INVITE dialog.

Syntax

```
config vsp policies session-policies policy default rule session-config
provisional-response
```

Properties

additional-response: Enter additional provisional responses you want sent with the INVITE dialog.

Default: There is no default setting
Example: **set additional-response 101 "This is a test"**

q931-cause-sip-response-map

Allows the configuration of q931-cause and/or H.225 reason code for calls cleared by an external SIP UA. When an IW call is cleared on the SIP side, the SIP response code is used to consult an internal table for Q.931/H.225 information needed when generating the ReleaseComplete, Admission Reject, or Location Reject message. By adding a **q931-cause-sip-response-map** entry, you can override the internal table defaults.

Syntax

```
config vsp session-config-pool entry q931-cause-sip-response-map
config vsp default-session-config q931-cause-sip-response-map
```

Properties

q931-cause: Select a q931-cause to use when clearing the H.323 side of the call. If this map entry will not generate a q931-cause, or you want to use the default, select **Any**.

Default: any
Example: **set q931-cause userby**

h2250-reason: Select a h225-reason to use when clearing the H.323 side of the call.

Default: none

- Values: none: Use the default h225-reason
- lrj: Select lrj if you are generating LRJ messages and enter a relevant reason
- arj: Select arj if you are generating ARJ messages and enter a relevant reason
- any: Specifying only the q931-cause in this entry

Example: **set h2250-reason lrj**

sip-response: Select the sip-response match criteria for this entry. If this entry will not generate a q931-cause or you want to use the default, select **Any**.

Default: 0
Values: Min: 300 / Max: 699
Example: **set sip-response 500**

q931-settings

The q931-settings configuration object is used to configure Q.931 settings on the ME.

Syntax

```
config vsp session-config-pool entry q931-settings
config vsp default-session-config q931-settings
```

Properties

numbering-plan: The Q.931 numbering plan set in Calling and Called Party number information elements.

Default: ISDN
Values: unknown | ISDN | data | telex | national-standard | private | reserved
Example: **set numbering-plan data**

numbering-type: The Q.931 numbering type set in Calling and Called Party number information elements.

Default: international
Values: unknown | international | national | network-specific | subscriber | abbreviated | reserved
Example: **set numbering-type national**

presentation-indicator: Enter the static presentation value to use.

Default: allowed
Values: allowed | restricted | numberNotAvailable | reserved
Example: **set presentation-indicator restricted**

screening-indicator: Enter the static screening value to use.

Default: notScreened
Values: notScreened | verifiedPassed | verifiedFailed | networkProvided
Example: **set screening-indicator verifiedPassed**

privacy-dynamic: When true, the screening and presentation are dynamic.

Default: true
Values: true | false
Example: **set privacy-dynamic false**

use-display-ie: When true, the ME attempts to use Display IE from the SETUP message when building SIP From: header display-name.

Default: true
Values: true | false
Example: **set use-display-ie false**

add-outgoing-displaytext-ie: When true, the ME attempts to use SIP From: header display-name when building Display IE in the outgoing SETUP message.

Default: false
Values: true | false

Example: **set add-outgoing-displaytext-ie true** *q931-bearer-capability-ie [coding-standard][info-transfer-capability][transfer-mode][transfer-rate][user-info-layer1-protocol]*: Sets the Q.931 Bearer Capability values used in outgoing H.323 messages.

Example: **set q931-bearer-capability-ie nationalStd speech packetMode 128Kbps h261**

recording-policy

Specifies whether to record calls, and if so, how to handle unsupported CODECs. Note that if you enable the record property, enabling recording of the SIP session, you must also have enabled the anchor property of the **media** object.

Playing recorded calls

If the record property is enabled, you can use CLI actions or the ME Management System to replay the SIP call session. To use the CLI, use the **mix-session** action to mix the files manually. You can also:

- play the file using the **file-play** action.
- play the session using the **playback session** action.

Handling unplayable CODECs

When the ME receives the SDP message (typically in either an INVITE or 200 OK), it contains information from the originator on supported CODECs. These CODECs may or may not be mixable by the ME device. When the **strip-unplayable** property is **enabled**, the ME removes any CODECs that it cannot mix from the list, and forwards the INVITE to the destination. When **disabled**, the ME forwards the INVITE with the CODEC list unchanged. The ME then records the entire contents of the call, but when mixed, any data sent with an unsupported CODEC results in silence.

> **Note:** If The ME records a call containing an unplayable CODEC, any archiving operation that involves that record (because the include-mixed-media property of the archiving operation is set to true) will fail.

Syntax

```
config vsp default-session-config media recording-policy
config vsp policies session-policies policy name rule name session-config media
recording-policy
config vsp dial-plan dial-prefix entryName session-config media recording-policy
config vsp dial-plan route name session-config media recording-policy
config vsp dial-plan source-route name session-config media recording-policy
config vsp session-config-pool entry name media recording-policy
```

Properties

record: Enables or disables recording of the SIP session to the default directory on the system disk system. See Playing Recorded Calls for more information.p

Default: disabled
Values: enabled | disabled
Example: **set record enabled**

strip-unplayable: Specifies whether or not to strip CODECs that the system does not support. (This is only applicable if recording is enabled.) See Handling Unplayable CODECS for more information, specifically about the risk to archiving operations if set to **disabled**.

Use the **file-play-verify** action to ensure that a recording is of a format supported by the ME device.

Default: enabled
Values: enabled | disabled
Example: **set strip-unplayable disabled**

refer-settings

Enables or disables call parking compatibility settings for the Sylantro SIP for Business initiative.

Syntax

```
config vsp default-session-config refer-settings
config vsp policies session-policies policy name rule name session-config
refer-settings
config vsp dial-plan dial-prefix entryName session-config refer-settings
config vsp dial-plan route name session-config refer-settings
config vsp dial-plan source-route name session-config refer-settings
config vsp session-config-pool entry name refer-settings
```

Properties

modify-call-parking: Enables or disables call parking compatibility features. When **enabled**, the system modifies the Refer-To header. Leave this property at the default setting of **disabled** if you are not using the SIP for Business platform.

Default: disabled
Values: enabled | disabled
Example: **set modify-call-parking enabled**

fix-refer-to-call-id: Specifies whether to modify the call ID value in the Refer-to header. Enable this property if there are problems, caused by an incorrectly implemented NAT device, with the Replaces section of the Refer-to header. When **enabled**, if the system detects a *@ipAddress* string at the end of the Refer-to header call-id field, the system replaces the address with the message's remote IP address. This allows the system to later use that value to find the call leg it refers to and perform a correct translation.

Default: disabled
Values: enabled | disabled
Example: **set fix-refer-to-call-id enabled**

reg-ex-header

Provides granular modification capabilities to create or modify a header. Use this when the methods of the **altered-header** object do not achieve the necessary changes. You can create multiple **reg-ex-header** entries to accomplish complicated manipulations. The system executes each one, the order of execution based on their order within the configuration.

Syntax

```
config vsp default-session-config header-settings reg-ex-header number
config vsp policies session-policies policy name rule name session-config
header-settings reg-ex-header number
config vsp dial-plan dial-prefix entryName session-config header-settings
reg-ex-header number
config vsp dial-plan route name session-config header-settings reg-ex-header
number
config vsp dial-plan source-route name session-config header-settings
reg-ex-header number
config vsp session-config-pool entry name header-settings reg-ex-header number
```

Properties

admin: Enables or disables this configuration entry.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

destination: Specifies the header to be created or modified by the properties set in this object (the header to write the changes to). If the header doesn't exist in the message, the system creates it. When you configure this, you must also set either the create or append properties. Otherwise, there is no configuration from which to derive the new URI. You do not specify the fields within the destination to change as that level of change is accomplished with the regular expression statements. To modify an existing header, use the same header for the both the **destination** and the **create** properties.

Default: There is no default setting
Example: **set destination Diversion**

create <expression><replacement>: Identifies the header containing the data to be modified and then written to the destination header. (Use the **append** property to add data to the existing header.) First, select the header that serves as the source of the data. Then, specify a regular expression to run against the value of the source header. Also supply the replacement expression to apply if there's a match. Changes are only applied to the original expression match.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Example: **set create History-Info "([^,<]*?)<sip:([^>?]*?)\?Reason=SIP%3Bcause%3D3[0-9]{2}(.*?)>(.*?)index=(\d.\d.\d | \d.\d | \d)" "<sip:\2>;reason=deflection"**

append<expression><replacement>: Identifies the header containing the data to be added to the destination header. First, select the header that serves as the source of the data. Then, specify a regular expression to run against the value of the source header. Also supply the replacement expression to apply if there's a match. The system appends this string to the existing destination header. To add spaces or commas, be sure to include them (using quotation marks) in the replacement statement.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Example: **set append History-Info ([^,<]*?)<sip:cov.com ", <sip:cov2.com"**

apply-to-methods: Specifies the message type to which the system applies header value changes. The system then changes the specified URI according to the settings of the header and destination properties of this object.

When you modify this value, the system overwrites the current setting with only the message types you specify. For example, if set to the default and you enter **INVITE**, the system only authenticates INVITE messages. Enter multiple types separated by a plus sign (+) with no spaces.

Default: INVITE
Example: **set apply-to-methods INVITE+REFER**

apply-to-responses: Specifies whether to apply header value changes to SIP requests or requests and responses. Set to **no** to apply changes only to requests. Set to **yes** to apply to responses as well. If yes, you must set the response code to which it applies. Create additional altered-body profiles to change multiple response types.

Default: no
Values: no | yes *responseCode*
Example: set apply-to-responses yes 200

session-persistent: Specifies to which messages in a session the ME should apply changes made with this object. When **enabled**, the ME applies any TO, FROM, or REQUEST URI changes to the first and all subsequent messages in a session. When **disabled**, the default, the system applies the changes only to the first message in the session.

Default: disabled
Values: enabled | disabled
Example: **set session-persistent enabled**

cseq: *Secondary property.* Sets a mechanism to further filter which SIP messages have the header expression modifications applied. If **cseq** is set to zero (the default), the ME applies the changes to all SIP messages. If set to any other value, the system only applies the changes to SIP messages having a CSEQ field that matches that value.

Default: 0
Example: **set cseq 100**

apply-to-dialog: Allows you to configure where to apply these options for a session.

Default: both
- Values: inbound: Apply to inbound dialog only.
- outbound: Apply to outbound dialog only.
- both: Apply to both inbound and outbound dialogs.

Example: set apply-to-dialog inbound

create-on-failed-match: *Secondary property*. When **true**, construct a create header even when the expression is not a complete match.

Default: true
Values: true | false
Example: set create-on-failed-match false

append-on-failed-match: *Secondary property*. When **true**, execute the append action event when the create expression fails to match.

Default: true
Values: true | false
Example: **set append-on-failed-match false**

registration

Configures properties that are used in processing REGISTER requests. Note that for the ME to accept REGISTER requests, you must enable the **admin** property of the **registration-service** object.

Syntax

```
config vsp default-session-config registration
config vsp policies session-policies policy name rule name session-config
registration
config vsp dial-plan dial-prefix entryName session-config registration
config vsp dial-plan route name session-config registration
config vsp dial-plan source-route name session-config registration
config vsp session-config-pool entry name registration
```

Properties

admin: Specifies whether to accept or deny REGISTER requests. If set to **enabled**, the system accepts REGISTERs; the system denies REGISTERs if set to **disabled**.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

cache-lcs: Specifies whether the system should add entries for LCS clients to the registration database. Use this object in single-domain configurations, where the system sits between the client and the LCS server. When **enabled**, client requests that pass through the system are added to the database. The system adds an entry for the LCS client record into its registration database. Therefore, you can enable features that use the database, such as call forking.

When **disabled**, only client requests from external domains are added to the database. The system passes all call requests directly to the LCS server.

Default: disabled
Values: enabled | disabled
Example: **set cache-lcs enabled**

add-path-header: Specifies whether the system adds a Path: header to the packet. The Path header, with syntax similar to the Record-Route header, is used in conjunction with REGISTER requests and with 2xx messages.

Default: disabled
Values: enabled | disabled
Example: **set add-path-header enabled**

local-options-reply: Specifies how the system responds to OPTIONS requests during a registration session. When **enabled**, the system flags the matching AOR entry in the location cache with this setting. In future registration sessions for that AOR, the system will respond locally to any OPTIONS requests, rather than forwarding them to the phone. When **disabled**, OPTIONS requests are forwarded normally.

Default: disabled
Values: enabled | disabled
Example: **set local-options-reply enabled**

handle-3xx-locally: Specifies whether the system forwards responses to 3xx messages back to the UAC or resends them. If **disabled**, when the system receives a 3xx response for a REGISTER (e.g., 301 Moved Permanently or 302 Moved Temporarily), it forwards the response back to the client. When this option is **enabled**, the system does

not forward the response back. Instead, it hunts through the contact routes in the response message until it finds one that responds with a 200 OK. It then applies that route to the REGISTER message.

Default: disabled
Values: enabled | disabled
Example: **set handle-3xx-locally enabled**

delegate-aged-bindings: Specifies whether the system can redelegate a REGISTER request before both sides experience a binding expiration. By default, when the binding expires on the client-side, the system does not redelegate the REGISTER until the peer expiration. This can cause problems if, for example, you are using DNS to load balance REGISTERs between systems. In that case, the REGISTER may not be redelegated to the third-party server until the peer timer expires, making the server unaware of the relocation. When **enabled**, the system redelegates the REGISTER request to the peer even if only the client-side timer has expired.

Default: disabled
Values: enabled | disabled
Example: **set delegate-aged-bindings enabled**

unregister-aged-bindings: Specifies whether the ME allows delegation of an unregister-all (*) request. When **enabled**, the ME delegates the Contact: * UNREGISTER request unchanged and removes all bindings associated with the AOR that have this feature enabled.

Default: disabled
Values: enabled | disabled
Example: **set unregister-aged-bindings enabled**

delegate-unregister-all: Specifies whether the system sends an UNREGISTER request when a binding ages out. When enabled, if a binding ages out the system sends an UNREGISTER to the delegate. The system also resets the peer expiration so that the next REGISTER from this client is delegated. You can remove the binding without sending the UNREGISTER by using the **location flush now** action.

Default: disabled
Values: enabled | disabled
Example: **set delegate-unregister-all enabled**

broadsoft-survivability-mode: *Secondary property*. Enables or disables the Broadsoft survivability feature.

Default: disabled
Values: enabled | disabled
Example: **set broadsoft-survivability-mode enabled**

broadsoft-registration-survivability-mode: Enables or disables the Broadsoft survivability registration feature. When enabled, enter an **expression**, a **replacement**, and the **client-expiration**. The **expression** is the value in the extension provided in the 200 OK and the **replacement** is the value for the ME to use in its place to build the local location cache mapping. The **client-expiration** is the time, in seconds, that the ME waits to recheck the state of the Broadsoft server.

Default: disabled
Values: enabled <*expression*> <*replacement*> <*client-expiration*> | disabled
Example: **set broadsoft-registration-survivability-mode enabled "^sip:.'(\d\d\d\d)@""\1" 30**

remote-party-id-specification

Configures the settings for header manipulation of the User and/or Host portion of the Remote-Party-ID header of the SIP message. Within this object you can modify the existing header or create a new header if one is not present. You can then modify the content by prepending or stripping off characters, for example, to ensure interoperability between carriers. Typically these headers are only part of INVITE messages.

Syntax

```
config vsp default-session-config remote-party-id-specification
config vsp policies session-policies policy name rule name session-config
remote-party-id-specification
config vsp dial-plan dial-prefix entryName session-config
remote-party-id-specification
config vsp dial-plan route name session-config remote-party-id-specification
config vsp dial-plan source-route name session-config
remote-party-id-specification
config vsp session-config-pool entry name remote-party-id-specification
```

Properties

user-source: Specifies the URI from which the system extracts the User portion and applies it to the Remote-Party-ID header: either From, To, Request, or Contact. If left at the default, **remote-party-id-uri**, there is no change to the USER portion. For example, if you select from-uri, the system extracts the User portion of the From URI and applies it to overwrite or create the User portion of the existing or newly created Remote-Party-ID header.

Default: remote-party-id-uri
Values: remote-party-id-uri | from-uri | to-uri | request-uri | contact-uri
Example: **set user-source from-uri**

user-action: Sets the action to take once the system has derived the content of the Remote-Party-ID header. You can prepend, strip-off, or replace digits to the User portion. You can also make no modifications. (For example, you may want to create a header using the remote-party-id-specification object, but not want to make any modifications to the default header that is created.)

Default: none
Values: none | prepend *phone-prefix* | strip-off-number | replace-with *new-number*
Example: **set user-action prepend 1**

host-source: Specifies where the system extracts the Host portion for the new or existing Remote-Party-ID header. Set the host field to be derived from the remote-party-id, To, From, Request, or Contact URIs of the INVITE, or you can enter any text string.

Default: remote-party-id-uri
Values: remote-party-id-uri | from-uri | to-uri | request-uri | contact-uri | *string*
Example: **set host-source to-uri**

create: Sets whether to create a Remote-Party-ID header if one does not already exist. If **true**, the system uses the property settings in this object to determine the value of the fields in the header. If set to **false**, the system does not create the header. If a header already exists, this field has no effect. (The fields of the existing header are still manipulated by the property values.

Default: false
Values: true | false
Example: **set create true**

request-uri-specification

Specifies what derives the content of the fields of the REQUEST URI when the ME transmits a message. For example, if the **user** property is set to **to-uri**, the ME replaces the user field of the REQUEST URI with data from the user field of the incoming TO URI. If set to **omit**, the user field is left blank. Or, you can enter any string that you want placed in the user field. See Altering URIs for information on replacement options.

Syntax

```
config vsp default-session-config request-uri-specification
config vsp policies session-policies policy name rule name session-config
request-uri-specification
config vsp dial-plan dial-prefix entryName session-config
request-uri-specification
config vsp dial-plan route name session-config request-uri-specification
config vsp dial-plan source-route name session-config request-uri-specification
config vsp session-config-pool entry name request-uri-specification
```

Properties

user: Specifies how to derive the value of the user field of the REQUEST URI.

Default: request-uri
Values: request-uri | to-uri | from-uri | omit | next-hop | local | omit-phone-context | *string*
Example: **set user request-uri**

host: Specifies how to derive the value of the host field of the REQUEST URI.

Default: request-uri
Values: request-uri | to-uri | from-uri | omit | next-hop | next-hop-domain | local-ip | *string*
Example: **set host omit**

port: Specifies how to derive the value of the port field of the REQUEST URI.

Default: request-uri
Values: request-uri | to-uri | from-uri | omit | *string*
Example: **set port omit**

transport: Specifies the value of the transport field of the REQUEST URI. In addition to using the value from other fields of the incoming URI, you can set the transport method to UDP, TCP, or TLS.

You cannot enter a string for this property.

Default: request-uri
Values: request-uri | to-uri | from-uri | omit | UDP | TCP | TLS | *string*
Example: **set transport omit**

use-param: Specifies whether the User parameter in the REQUEST URI of the SIP header is maintained or removed when the system forwards a message. If set to **keep**, the message is forwarded with the parameter as it was received. If set to **omit**, the entire user=*param* is removed from the TO URI.

Default: omit

Values: omit | keep
Example: **set use-param keep**

user-truncate-non-digits: Specifies whether to remove non-digits from the User portion of the REQUEST URI in INVITE messages. When **enabled**, the system removes all non-digits.

Default: disabled
Values: enabled | disabled
Example: **set user-truncate-non-digits enabled**

uri-parameter <name><value>: Appends the specified user parameter and value to the REQUEST URI for matching calls. For example, the example below would result in a REQUEST URI that looked similar to:

<sip:spot@fun.com;BTG=trunk1>

You can control how and when the new parameter is added using the append options. Select **append-always** to have the parameter added to all matching calls. Select **append-if-does-not-exist** to add the name and value only if it does not already exist in the URI, preventing the possibility of duplicate parameters. Select **overwrite-existing** to replace any existing parameter in the REQUEST URI with the configured name and value, updating instead of appending to the parameter. You can append multiple user parameters.

Default: There is no default setting
Example: **set uri-parameter BTG trunk1**

apply-to-routing: Specifies whether the system uses the REQUEST URI changes that result from the settings for routing and normalization or for normalization only. When set to **false**, the default, the system only changes the REQUEST URI for the next-hop SIP server (normalization). When **true**, the system uses the information in the modified REQUEST URI to forward the message (routing), bypassing the functions of the dial plan and/or location cache.

Set this property to **true** only when the system is operating in proxy mode and relies on the modified REQUEST URI to correctly forward the message. For example, set this property to **true** in a CSTA application to cause the system to use the new REQUEST URI to resolve routing.

Default: false
Values: true | false
Example: **set apply-to-routing true**

use-location-cache-contact-uri: Specifies whether the system should use the location cache information or the **request-uri-specification** configuration to modify the contact field of the REQUEST message. This property is only applicable when the system uses the location service to forward the request. When **true**, the system copies the phone contact URI (as found in the location cache) to the request URI. When **false**, the system uses the settings of this object to determine the content of the REQUEST message.

Default: true
Values: true | false
Example: **set use-location-cache-contact-uri false**

response-translation-settings

Maps a new status code and, optionally, phrase to a received code.

Syntax

```
config vsp default-session-config response-translation-settings
config vsp policies session-policies policy name rule name session-config
response-translation-settings
config vsp dial-plan dial-prefix entryName session-config
response-translation-settings
config vsp dial-plan route name session-config response-translation-settings
config vsp dial-plan source-route name session-config
response-translation-settings
config vsp session-config-pool entry name response-translation-settings
```

Properties

entry <statusCode><newStatusCode>[reason][newReason]: Sets a new numeric to a reason code. Specify a code number and the replacement number to use instead. Optionally, you can specify the reason phrase and a replacement phrase. If you specify both the code and phrase, the incoming message must contain both for the replacement to take place.

Default: none
Example: **set entry 503 200 "Service Unavailable" "Service Delayed"**

routing-settings

Provides mechanisms for filtering the routing table. For example, when configured and a policy match occurs, the ME can send traffic to interfaces with the same geolocation setting. Or, the ME can control the egress interface using the routing and classification tag system. (See Tag-Based Route Selection for a complete description of how the ME classifies traffic.) Otherwise, the ME uses the entire routing table when identifying interface choices.

> **Note:** The preferred method for creating virtual firewalls is by using routing tags and VLANs. For sample configurations that illustrate VLANs, overlapping IP addresses, and virtual firewalls, see the *Oracle Communications WebRTC Session Controller Administration Guide*.

Syntax

```
config vsp default-session-config routing-settings
config vsp policies session-policies policy name rule name session-config
routing-settings
config vsp dial-plan dial-prefix entryName session-config routing-settings
config vsp dial-plan route name session-config routing-settings
config vsp dial-plan source-route name session-config routing-settings
config vsp session-config-pool entry name routing-settings
```

Properties

admin: Enables or disables this route setting configuration.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

geolocation: Assigns a numeric to match on for the **ip** interface geolocation property. When a match occurs, the system forwards traffic to an interface with a matching

geolocation. If you set the value to 0, the system uses the whole routing table when selecting an interface.

Default: 0
Example: **set geolocation 10**

ingress-classification-tag: Classifies (associates) incoming traffic with the configured tag. If you configure this property, you must configure the **egress-classification-tag** property (below) as well. The configured ingress-tag must match a configured **ip** routing-tag. The routing-tag identifies the groups of interfaces and routes that are available for the ingress-tag. See Tab-Based Route Selection for a complete description.

You can also configure a **classification-tag** through the **ip** interface object. If this property is configured in both places, the session-config setting takes precedence.

Note that this tag is case-sensitive.

Default: There is no default setting
Example: **set ingress-classification-tag E911 egress-classification-tag**: Sets an egress classification tag that is used to select the outgoing interface. That tag must then be associated with an **ip > routing-tag**, which controls the available egress interfaces and routes. See Tag-Based Route Selection for a complete description.

You can also configure a **classification-tag** through the **ip** interface object. If this property is configured in both places, the session-config setting takes precedence.

Note that this tag is case-sensitive.

Default: There is no default setting
Example: **set egress-classification-tag E911**

rtcp-header

Enables or disables verification of the Real-Time Control Protocol (RTCP) header. The ME uses RTCP to maintain quality of service and derive diagnostic data on RTP sessions.

Syntax

```
config vsp media-verify-config rtcp-header
```

Properties

header: Enables or disables verification of the RTCP header using the negotiated and agreed media.

Default: enabled
Values: enabled | disabled
Example: **set header disabled**

rtp

Enables or disables verification of the Real-Time Protocol (RTP) header. The ME uses the timing and sequence information in RTP to reassemble packets appropriately for real-time audio and video.

Syntax

```
config vsp media-verify-config rtp
```

Properties

check-header: Enables or disables verification of the RTP header using the negotiated and agreed media.

Default: enabled
Values: enabled | disabled
Example: **set check-header disabled**

Rx

Configures communication between the ME and the Camiant Policy Server (3GPP Rx). The Camiant Policy Server applies business rules that determine which and when customers, tiers, and/or applications receive bandwidth priority. The ME notifies the Camiant system about media flows so that the Camiant system can then apply policy for resource allocation (or example, reserving bandwidth or setting up QoS and forwarding rules). Communication between the two system uses a 3GPP Diameter-based Rx interface.

This object sets destination Diameter servers and configures what the ME does in response to an RAR received from the Camiant (3GPP Rx) box. Using this configuration the ME is not authenticating a message, but is instead authorizing a service and reserving bandwidth for it. The Camiant box then sends the Re-Authorization Requests (RARs) when there is a network event that requires it and the ME acts according to the mapped response.

Syntax

```
config vsp default-session-config 3GPP Rx
config vsp policies session-policies policy name rule name session-config 3GPP Rx
config vsp dial-plan dial-prefix entryName session-config 3GPP Rx
config vsp dial-plan route name session-config 3GPP Rx
config vsp dial-plan source-route name session-config 3GPP Rx
config vsp session-config-pool entry name 3GPP Rx
```

Properties

admin: Enables or disables the Rx configuration. When **enabled**, the system forwards Rx requests to the configured server and takes the configured actions on RARs. When **disabled**, the system does not send messages to or receive them from the Camiant server.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

diameter-group: Specifies a previously configured **diameter-group** to which the system sends Rx requests.

Default: There is no default setting
Example: set diameter-group “vsp diameter-group 3gppGroup”

re-auth-action <cause><action>: Specifies the action the system takes on Re-Authorization Request (RAR) messages it receives from the Camiant (3GPP Rx) box. The Camiant server sends a cause, and the system takes the action mapped to that cause with this property.

Default: disconnect release-of-bearer
- Values: causes:

- service-info-request
- charging-correlation-change
- loss-of-bearer
- recovery-of-bearer
- release-of-bearer
- establishment-of-bearer
- Values: actions:
- **none**: Take no action
- **re-authenticate**: Attempt to reauthenticate the message by sending a new Diameter Authentication-Authorization-Request
- **disconnect**: Disconnect the call

Example: set re-auth-action service-info-request re-authenticate

sdp-regeneration

Sets parameters to "regenerate" the SDP in order to more tightly control what is sent out by ME. This ensures that approved SDP format comes from the system on every call.

Manipulating Connection Information

Some phones may require configuration ensuring that connection information is not specified within the media descriptor. Using the **add-session-connection** and **remove-media-connection** properties together, you can configure the ME Engine to add a session-level c-line and remove matching c-lines from the media descriptor.

The SDP c-line, which contains connection data for the session, can be found in each media description and/or at the session level. If it appears at both the session level and in the media descriptor, the media descriptor value takes precedence (for that descriptor).

Syntax

```
config vsp default-session-config sdp-regeneration
config vsp policies session-policies policy name rule name session-config
sdp-regeneration
config vsp dial-plan dial-prefix entryName session-config sdp-regeneration
config vsp dial-plan route name session-config sdp-regeneration
config vsp dial-plan source-route name session-config sdp-regeneration
config vsp session-config-pool entry name sdp-regeneration
```

Properties

regenerate: Controls whether the system applies the settings in this object. When **enabled**, all settings are applied to the SDP and the system regenerates it before passing it on.

Default: enabled
Values: enabled | disabled
Example: **set regenerate disabled**

origin: Specifies whether the system overwrites the string that appears in the origin line (o=) of the SDP. If set to **rewrite**, the system changes the value of the username to

CSM or the value set with the **username** property. In addition, it changes the value of the session-id and session-version to zero. Otherwise, it passes the name unchanged.

Default: rewrite
Values: pass | rewrite
Example: **set origin-pass**

username: Specifies the username to be inserted into the username field of the origin line of the SDP. This value is only applicable if the origin property is set to **rewrite**.

Default: There is no default setting
Example: **set username joe@acme.com**

session-name: Specifies whether the system overwrites the textual session name that appears in the session-name (s=) line of the SDP. If set to **rewrite**, the system changes the content of the session name to Oracle or the value set with the **name** property. Otherwise, it passes the session-name unchanged.

Default: rewrite
Values: pass | rewrite
Example: **set session-name pass**

name: Specifies the name to be inserted into the session-name of the SDP. This value is only applicable if the origin property is set to **rewrite**.

Default: There is no default setting
Example: **set name "multimedia conference"**

session-info: Specifies whether to strip out or pass the textual information in the session-info (i=) line of the SDP.

Default: strip
Values: pass | strip
Example: **set session-info pass**

uri: Specifies whether to strip out or pass the uri (u=) line of the SDP, a pointer to additional information about the session.

Default: strip
Values: pass | strip
Example: **set uri pass**

e-mail-address: Specifies whether to strip out or pass the email contact information for the person responsible for the conference. This is displayed in the e= line of the SDP.

Default: strip
Values: pass | strip
Example: **set e-mail-address pass**

phone-number: Specifies whether to strip out or pass the telephone contact information for the person responsible for the conference. This is displayed in the p= line of the SDP.

Default: strip
Values: pass | strip
Example: **set phone-number pass**

bandwidth: Specifies whether to strip out or pass the proposed bandwidth to be used by the session. This is displayed in the b= line of the SDP.

Default: strip
Values: pass | strip
Example: **set bandwidth pass**

timing: Specifies whether to strip out or pass the start and stop times for a session. This is displayed in the t= line of the SDP. Note that some phones require a t-line for proper operation.

Default: strip
Values: pass | strip
Example: **set timing pass**

remove-unknown: Specifies whether to remove any unknown lines from the SDP. When **enabled**, all unknown (non-specification) lines are removed.

Default: enabled
Values: enabled | disabled
Example: **set remove-unknown disabled**

add-session-connection: Specifies whether to add session-level c-line content to the SDP if it is not already there. When **enabled**, the system inserts a session-level c-line (prior to the first m-line) in the SDP text message. The content for the line is derived from the first media descriptor c-line. Note that this property only adds the c-line at the session level. To remove c-lines from the media-descriptors, you must use the **remove-media-connection** property. See Manipulating Connection Information for more information.

Default: disabled
Values: enabled | disabled
Example: **set add-session-connection enabled**

remove-media-connection: Specifies whether to remove the c-line content from the SDP media descriptors. When **enabled**, the system removes all c-lines within media descriptors that match the session-level c-line. See Manipulating Connection Information for more information.

Default: disabled
Values: enabled | disabled
Example: **set remove-media-connection enabled**

add-rtpmaps: Specifies whether the system includes the rtpmap attributes for well-known CODECs (that is "knows" about), when the rtpmap is not included by the original endpoint. Payload types under 96 must be registered with IANA as well-known CODECs. An rtpmap attribute for well-known codecs is not required in the SDP and, therefore, not included by some endpoints. However, certain endpoints have processing problems when the rtpmap is not included. When **enabled**, the system adds the rtpmap attributes.

Default: disabled
Values: enabled | disabled
Example: **set add-rtpmaps enabled**

pass-attribute: Identifies specific attribute lines to be passed through the system unchanged. Enter as many attributes as you require. Select from a predefined list or enter the attribute name. By default the system always passes certain attributes having to do with call flow (e.g., a=sendrecv) and cryptography (e.g., a=key-mgmt, a=crypto).

Default: disabled
Values: enabled | disabled
Example: **set pass-attribute enabled**

session-control-settings

Sets whether or not the system re-evaluates policy for new requests on an existing session. A session, to the ME, is a SIP session between two parties, as defined by the To: and From: headers of the SIP messages.

A session is always started with a request message, such as an INVITE, SUBSCRIBE, NOTIFY, or REGISTER. Once a session is established, additional messages can be sent. For example, a request message (NOTIFY) or a MESSAGE message (containing an instant message text item) can come across an INVITE-initiated session.

Normally, the ME only processes policy (or runs all of the relevant policy rules by the message) for the first message of a session. With this setting enabled, the ME processes policy for each request message that comes across in a session.

Syntax

```
config vsp default-session-config session-control-settings
config vsp policies session-policies policy name rule name session-config
session-control-settings
config vsp dial-plan dial-prefix entryName session-config session-control-settings
config vsp dial-plan route name session-config session-control-settings
config vsp dial-plan source-route name session-config session-control-settings
config vsp session-config-pool entry name session-control-settings
```

Properties

re-evaluate-new-requests: Specifies whether the system should process policy for all or only the first message of a session. When **enabled**, the system processes policy for each request message. When **disabled**, it processes only the first message of a session.

Default: disabled
Values: enabled | disabled
Example: **set re-evaluate-new-requests enabled**

refused-methods <messageType><code> **<text>**: Provides a method to refuse the specified SIP message type. Enter the message type with a response code and accompanying error text.

Example: **set refused-methods REFER 499 "REFER not allowed" use-lnp-for-routing**: Provides a customer-specific application implementation that is not otherwise applicable.

Default: disabled
digest-realm-to-lnp-map: Provides a customer-specific application implementation that is not otherwise applicable.

Default: There is no default setting
source-lnp: Provides a customer-specific application implementation that is not otherwise applicable.

Default: There is no default setting
destination-lnp: Provides a customer-specific application implementation that is not otherwise applicable.

Default: There is no default setting
call-action-3pcc-server-entry: Sets the server that will be used when a call-control action is invoked. When a server is set, the ME processes any call control action on that server. If no server is specified, the ME determines which server to use.

Default: There is no default setting

Example: **set call-action-3pcc-server-entry "vsp enterprise 3pcc-servers internal-csta-server CSTA1"**

sip-directive

Sets the default instruction to apply to SIP packets arriving to open a new session. The ME can either allow the session to come up, refuse it (but send a polite error response), or discard the SIP message that started the session and ignore the request.Use the SIP directive in the default session config object to apply instructions for calls in which there are no configured policies or rules; use the policy session config object to define the action for calls that match policy.

Syntax

```
config vsp default-session-config sip-directive
config vsp policies session-policies policy name rule name session-config
sip-directive
config vsp dial-plan dial-prefix entryName session-config sip-directive
config vsp dial-plan route name session-config sip-directive
config vsp dial-plan source-route name session-config sip-directive
config vsp session-config-pool entry name sip-directive
```

Properties

directive: Sets the default SIP call directive (instruction) to apply to the SIP call session.

Default: discard

- **Values: allow**: The system allows the packet through.
- **discard**: The system immediately discards the packet.
- **refuse<***resultCode***><***textString***>**: The system discards the packet but sends a response to indicate having done so. Optionally you can specify a SIP code between 400 and 699 and a text message to be appended to a refused SIP call record. If set to the **refuse** option, the code string is not visible.

Example: **set directive refuse 401 "Server not available"**

sip-response-q931-cause-map

Allows the configuration of sip-response code for calls cleared by an external H.323 GW. When a ReleaseComplete, Admission Reject, or Location Reject message is received by the ME, the ME consults an internal table to determine the appropriate SIP response code to generate when clearing the SIP side of the call. By adding a **sip-response-q931-cause-map** entry, you can override the internal table defaults.

Syntax

```
config vsp session-config-pool entry sip-response-q931-cause-map
config vsp default-session-config sip-response-q931-cause-map
```

Properties

sip-response: Define the sip-response that will be used when clearing the SIP side of the call.

Default: 0

Values: Min: 300 / Max: 699
Example: **set sip-response 350**

q931-cause: Select a q931-cause that helps to qualify the H.323 call-clear. If this map entry does not depend on the q931-cause value, either because there is no Q.931 present or because any Q.931 cause qualifies, choose **Any**.

Default: There is no default setting
Example: **set q931-cause noresponse**

h2250-reason: Select a h2250-reason type.

Default: none

- Values: LRJ: Match an incoming LRJ message
- ARJ: Match an incoming ARJ message
- H.225: Match all other relevant traffic
- none: H.225 reason should not be used as match criteria for this entry

Example: **set h2250-reason lrj**

sip-session-timers-settings

Sets the values of SIP session timers as they are described in *RFC 4028, Session Timers in the Session Initiation Protocol (SIP)*. These timers establish periodic refreshes of SIP sessions, through re-INVITE and UPDATE requests, to help user agents and proxies determine whether the session is still active. In other words, it provides a keepalive functionality. Use of these timers helps prevent proxies that retain state information from needlessly keeping a call active. (This may happen if the proxy, for whatever reason, does not receive a BYE from the UA.)

There are two specific timer values that are used to derive the refresh interval:

- The session expiration, which defines the maximum length of a session
- The minimum allowed value for the session expiration.

The ME always sends out the largest timer value that it knows of. For example, if a UAC sends a session expiration time that is lower than the ME configured timer, the ME forwards the call with its own setting. If the session expiration set by the UAC is higher, the ME uses the UAC setting. If the session expires, the configurable **action** determines the ME response.

The timer values that you set with this object apply to all calls that are processed through the ME device.

Syntax

```
config vsp default-session-config sip-session-timers-settings
config vsp policies session-policies policy name rule name session-config
sip-session-timers-settings
config vsp dial-plan dial-prefix entryName session-config
sip-session-timers-settings
config vsp dial-plan route name session-config sip-session-timers-settings
config vsp dial-plan source-route name session-config sip-session-timers-settings
config vsp session-config-pool entry name sip-session-timers-settings
```

Properties

admin: Specifies whether the SIP session timers configuration is in use or not. If **enabled**, the system adds a Session-Expires and a Min-SE header to the INVITE request.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

preferred-refresher: Specifies which side of the connection performs the refreshes (send the re-INVITE and UPDATE requests). Select either UAC or UAS. If the selected device is not configured to act as a refresher, this property resets that configuration to enable refreshing (assuming the device implements the RFC).

Default: UAC
Values: UAC | UAS
Example: **set preferred-refresher UAS**

session-expires: Specifies the duration of the session. This is the maximum time allowed between session refresh requests before a session times out. The RFC notes that you can, but should not, set a value of less than 1800 (30 minutes), as it causes excessive messaging.

Default: 1800
Values: Min: 90 / Max: 1000000
Example: **set session-expires 2700**

min-se: Specifies the minimum allowed value for the session expiration interval. This lower floor is the fastest refresh rate a proxy servicing a request can require. A proxy can raise but not lower this minimum.

Default: 90
Values: Min: 90 / Max: 1000000
Example: **set min-se 180 action**: Specifies the action the system should take when the SIP session timers have expired.

Default: terminate
- **Values: terminate**: The system immediately removes all internal resources dedicated to the call
- **disconnect**: The system sends a BYE in an attempt to gracefully close the call before terminating it
- **nothing**: The system ignores the timer expiration and keeps the session alive until it receives a BYE

Example: **set action disconnect**

sip-settings

Configures the SIP settings that the ME applies to the SIP call session. If there are no configured policies or rules to enforce on the SIP call, then the ME applies SIP settings from the default session configuration. If the call does match policy rules, the ME applies the SIP settings defined in the session-config object.

Syntax

```
config vsp default-session-config sip-settings
config vsp policies session-policies policy name rule name session-config
sip-settings
config vsp dial-plan dial-prefix entryName session-config sip-settings
```

```
config vsp dial-plan route name session-config sip-settings
config vsp dial-plan source-route name session-config sip-settings
config vsp session-config-pool entry name sip-settings
```

Properties

mode: Sets the SIP operating mode to use with this server.

Default: auto-determine

- **Values: auto-determine**: The system determines the mode. Usually, the system is a back-to-back user agent (B2BUA) that sends INVITE traffic to the call destination. (The system B2BUA appears as the SIP call destination, but regenerates the call to the destination SIP server.) In some cases, however, the system may act as a proxy instead.
- **proxy**: The system is the SIP proxy that provides SIP registration, location, policy, and other services that determine the outcome of the SIP call.

Example: **set mode proxy**

transport: Sets the default protocol over which the SIP call session is forwarded to the destination SIP server.

Default: any

Values: any | UDP | TCP | TLS

Example: **set transport udp**

port: Specifies the destination port on the system for SIP traffic.

Default: auto-determine

- **Values: auto-determine**: The system sets the SIP port.
- **override**: Set the port number manually. Enter a port number between 1 and 65535 or leave blank to accept the default setting of 5060.

Example: **set port 1212**

route-hdr: Sets if and where to insert a Record-Route header in relevant SIP requests for outgoing system interfaces. This allows the system to remain in the SIP signaling path. (Route headers are used by SIP proxies only, so this behavior affects only proxied traffic.)

Default: none

- **Values: none**: The system does not add Record-Route headers.
- **RouteRequest**: The system inserts a Record-Route header in REQUESTs, allowing it to remain in the signaling path.
- **RouteRequestResponse**: The system both inserts a Record-Route header in SIP REQUESTs and performs a Record-Route header fix in SIP RESPONSEs. This is needed when the SIP request and corresponding response are forwarded via different interfaces on the ME device.

Example: **set route-hdrRouteRequestResponse**

route-hdr-use-fqdn: Specifies where the system derives the host portion of the Record-Route header from. When **enabled**, the system uses a domain name in the Record-Route header. That name is either:

- Configured with the **route-hdr-uri-host** property.
- Determined from a reverse lookup on the local address that the system used to transmit the SIP message.

Default: enabled

Values: enabled | disabled
Example: **set route-hdr-use-fqdn disabled**

route-hdr-uri-host: Specifies the string to insert in the host portion of the Record-Route header, identifying the ME domain name. This property is for use in conjunction with the **route-hdr-use-fqdn** property. If you set this, be sure that the string you enter is also set as the domain name or one of the domain aliases configured in the **vsp > static-stack-settings** object.

Default: There is no default setting
Example: **set route-hdr-uri-host acme.com**

route-hdr-add-register-msg: Specifies whether to add Record-Route headers to REGISTER messages. When **enabled**, the system adds Record-Route headers to REGISTER messages. Enable this for compatibility with Microsoft OCS 2007. When **disabled**, the system does not add the headers to Record-Route headers to REGISTER messages, in compliance with the suggestions of RFC 3261, SIP: Session Initiation Protocol.

Default: disabled
Values: enabled | disabled
Example: **set route-hdr-add-register-msg enabled**

route-hdr-preprocess-strip: Controls whether or not the system strips the MAddr parameter off route headers prior to processing them. This property is only for use in cases where the system receives traffic from a non-RFC3261-compliant (strict-router) SIP proxy. Do not change the value unless instructed to do so by Technical Support.

Default: disabled
Values: enabled | disabled
Example: **set route-hdr-preprocess-strip enabled**

lcs-compatibility: Enables or disables LCS compatibility. The specific bit setting required is dependent on the desired feature, and can only be determined for your system by Technical Support. For example, if you enable the **t120-anchor** property of the **file-transfer** object, you must set this bit to 0x010032e3. Do not enable this feature unless explicitly instructed to do so.

Default: disabled
Values: enabled | disabled
Example: **set lcs-compatibility enabled**

in-server: Specifies the originating server in a Sametime-to-LCS interaction. This allows the system to determine the transformations it must perform to facilitate communications. For the originating server, set the server version or function. The type that you select is dependent on the server type that you are configuring.

Use this property in conjunction with the **out-server** property. For example, in an LCS-to-Sametime setup, the **in-server** type would be lcs-200*x* and the **out-server** would be sametime-31*x*.

Default: unknown
Example: **set in-server lcs-2005**

out-server: Specifies the receiving server in a Sametime-to-LCS interaction. This allows the system to determine the transformations it must perform to facilitate communications. For the receiving server, set the server version or function. The type that you select is dependent on the server type that you are configuring.

Use this property in conjunction with the **in-server** property. For example, in an LCS-to-Sametime setup, the **in-server** type would be lcs-200*x* and the **out-server** would be sametime-31*x*.

Default: unknown
Example: **set out-server sametime-31**

utilize-contact: Determines what the system should do with messages it sends out. When **enabled**, the system sets the address in the contact header to the ME local IP address. If **disabled**, the system does not form the From header from the Contact header.

Default: enabled
Values: enabled | disabled
Example: **set utilize-contact disabled**

add-contact-nat: When **enabled**, the system appends the string "nat=true" to the contact header. If a phone is behind a firewall, and the system does not change the contact header to its own contact information, then it appends the string to the contact header.

Default: disabled
Values: enabled | disabled
Example: **set add-contact-nat enabled**

compress-signaling: When **enabled**, the system applies gzip compression to all SIP messages between the endpoints. This feature is only applicable between two ME devices.

Default: disabled
Values: enabled | disabled
Example: **set compress-signaling enabled**

preserve-call-id: Specifies whether the system generates a new call ID when forwarding an INVITE. Normally the system is a B2BUA, and therefore this property is **disabled**. As such, when it receives a call it generates a new ID for the outbound leg. If this property is **enabled**, the system uses the same call ID in the outbound INVITE that it received in the inbound INVITE.

Default: disabled
Values: enabled | disabled
Example: **set preserve-call-id enabled**

preserve-cseq: Specifies whether the system generates a new CSeq value when forwarding an INVITE. Normally the system is a B2BUA, and therefore this property is **disabled**. As such, when it receives a call it generates a new CSeq for the outbound leg. If this property is **enabled**, the system uses the same CSeq value in the outbound INVITE that it received in the inbound INVITE.

Default: disabled
Values: enabled | disabled
Example: **set preserve-cseq enabled**

proxy-generate-100-trying: Specifies on which type of SIP messages the system should generate and send a "100 trying" message to the sender. The system passes the "100 trying" when it receives it for undeclared message types.

By default, the system sends the "100 trying" as it was received (if the system is in proxy mode). Use this mode if the message contains information that must be passed from the SIP application server to the user agent. For example, additional headers in the message may contain version information necessary for proper operation of a softphone.

When selected for a message type, the system immediately sends the initiator a "100 trying" to indicate that it is processing the message and will forward it as soon as possible. Sending an immediate response can prevent longer SIP messages from timing out.

Default: There is no default setting
Example: **set proxy-generate-100-trying register+subscribe**

handle-3xx-locally: Specifies whether the system forwards responses to 3*xx* messages back to the UAC or resends it. If **disabled**, when the system receives a 3xx response for an INVITE (e.g., 300 Multiple Choices or 301 Moved Permanently), it forwards the response back to the UAC. When this option is **enabled**, the system does not forward the response back. Instead, it resends the INVITE message to the address specified in the contact header of the 3*xx* response.

Default: disabled
Values: enabled | disabled
Example: **set handle-3xx-locally enabled**

handle-3xx-locally-server-arbitration: Specifies whether the system uses or ignores the **dial-plan > arbiter** settings when forwarding a 302 Redirect message. The 302 contains the contact addresses to which the system redirects the message. If this property is **enabled**, the ME applies the arbiter selected for the original INVITE to the contact addresses to determine the redirect destination. If **disabled**, the system uses the server qvalue (preference) to select the forwarding address.

Default: disabled
Values: enabled | disabled
Example: **set handle-3xx-locally-server-arbitration enabled**

handle-exx-locally-lookup-original-invite: Specifies whether, in the event of a 3xx response from a redirect server, the system modifies the original INVITE it receives before doing a **dial-plan** lookup. If **disabled**, the system does a lookup on the original INVITE. When this option is **enabled**, the system modifies the INVITE before initiating a lookup by replacing the Request URI in the INVITE with the Contact URI found in the 3xx response. You must enable this response if you are doing a **source-route** dial-plan lookup. If you do not, there will be no source on which to base the lookup.

Default: disabled
Values: enabled | disabled
Example: **set handle-exx-locally-lookup-original-invite enabled**

session-timeout: Specifies how many seconds the system retains a session if the session did not establish successfully. For example, when authentication is required, case of authentication, the system sends a 401/407 response to the UAC to resend the request with authentication information. The session was not successful but cannot yet be terminated. This timer determines the length of time the system waits for a response.

Default: 300
Values: Min: 1 / Max: 1000000
Example: **set session-timeout 500**

session-duration-max: Specifies how many seconds the system maintains a session after the session has been successfully established. This property puts a timer on the session and forces a close when the timer expires. If set to 0 (the default), the session remains open until it is complete.

Default: 0 (disabled)
Values: Min: 0 / Max: 1000000
Example: **set session-duration-max 18000**

session-provisional-timeout: Specifies the number of seconds that the system allows the user agent server to ring before it times out the call. If **forking-settings** are configured, the system tries the next endstation when the timer expires, If not, the

system terminates the call. If set to 0, the default, session termination due to unanswered ringing is determined by the user agent client.

Default: 0 (disabled)
Values: Min: 0 / Max: 1000000
Example: **set session-provisional-timeout 30**

session-authentication-timeout: Specifies the number of seconds that an INVITE session can stay in the authentication state before timing out. Because a first INVITE typically does not contain authentication information, it results in a 401 (Unauthorized) response. This timer sets the allowable time between that response and the second INVITE (which contains authentication information). If set to 0, the default, the timer is disabled and the sip-settings **session-timeout** property determines when the session times out.

Default: 0 (disabled)
Values: Min: 0 / Max: 1000000
Example: **set session-authentication-timeout 30**

outbound-local-ip: Specifies which IP address to use to reach a destination when the ME tables contain multiple addresses to that destination. Use this in conjunction with the **outbound-local-port** property to specify an address and port combination.

Default: There is no default setting
Example: **set outbound-local-ip 10.10.10.1**

max-retransmissions: Specifies a the maximum number of times the system attempts to retransmit SIP messages at the transaction level. By setting this value through the session config, you can apply different retransmission values to different message types. For example, you might want a higher number of retransmissions for a REGISTER message, and a lower number (faster response) for an INVITE. This value does not apply to OPTIONS messages. Use the **settings > max-options-retransmissions** property to control OPTIONS retransmissions.

Default: 1
Values: Min: 1 / Max: 32
Example: **set max-retransmissions 15**

outbound-local-port: Specifies which port to use to reach a destination when the ME tables contain multiple ports to that destination. Use this in conjunction with the **outbound-local-ip** property to specify an address and port combination.

Default: There is no default setting
Example: **set outbound-local-port 3435**

message-session-timeout: Specifies the number of seconds that the system keeps alive a session that was created by a MESSAGE request after the first transaction is complete.

Default: 1800
Example: **set message-session-timeout 2400**

udp-source-port: Specifies the destination port on the system for UDP SIP traffic.

Default: auto-determine

- **Values: auto-determine**: The system sets the UDP port.
- **override**: Set the port number manually. Enter a port number or leave blank to accept the default setting of 5060. Enter a port number between 1 and 65535. The default port number is 5060.

Example: **set udp-source-port override 1212**

add-subject-header: Adds the specified string as a Subject header when a matched dial string is present in the user portion of the REQUEST, TO, or FROM URI. The system adds the header to all requests destined for devices that have successfully registered except for REGISTER, PRACK, ACK, BYE, and CANCEL.

Default: none
Values: <none | header> *string* <request-uri | to-uri | from-uri>
Example: **set add-subject-header header internalDevice=true to-uri**

strip-route-header: Specifies whether the system should strip out the route header and ignore the information it contains on both inbound and outbound traffic. The route header is a field of the SIP header which is not commonly used, but when it is, the system gives it highest priority when forwarding calls.

By default (when **disabled**), the system uses the information in the route header to forward packets. Set this property to **enabled** to ensure that manual configuration of SIP routing takes precedence.

Default: disabled
Values: enabled | disabled
Example: **set strip-route-header enabled**

strip-via-headers: Specifies whether to remove all but the specified number of Via headers when forwarding a request. When **enabled**, the system strips all but the specified number of Via headers from the request as the request passes through the box. For example, if set to 1, the system strips all Via headers except the bottom (furthest). The system always inserts itself as the top Via header, regardless of the setting. It then restores the Via headers on the response. When **disabled**, the default, the system retains all Via headers. If **enabled**, the default number of headers kept is 0 (strip all).

Default: disabled
Values: enabled | disabled *integer*
Example: **set strip-via-headers enabled 3**

sticky-via: Specifies whether to replace the port number in a VIA header. This property is only for use in cases where a unique port is assigned at connection and that port should be used in the VIA header. When this property is set to 0, the system leaves the port at the value that was received. When set to 1, the system replaces the value in the top VIA header. When set to 2, the system replaces the value in the next-to-last VIA header. Only use this property if instructed to do so by Technical Support.

Default: 0
Values: 0 | 1 | 2
Example: **set sticky-via 1**

ignore-provisional-tag: Specifies whether the system updates its internal SIP session using the TO header tag present in the provisional response sent by an intermediate server. If **enabled**, the system ignores the provisional response and updates the SIP session with the TO header tag found in the SIP final response. When **disabled**, the system updates the SIP session with the TO header from the provisional response.

Default: enabled
Values: enabled | disabled
Example: **set ignore-provisional-tag disabled**

forward-provisional-ack: Specifies which points along the path respond to a provisional ACK. When **enabled**, the mid-stations along the path do not respond to the PRACK message. Only the endpoints respond (end-to-end). When set to **disabled**, the default, the each hop along the path responds.

Default: disabled
Values: enabled | disabled
Example: **set forward provisional-ack enabled**

terminate-transaction-on-bye: Specifies system behavior when it receives a BYE message. If **enabled**, the ME device terminates any outstanding transactions on that leg when it receives a BYE message. The system applies the termination to a particular call leg for a particular session. When **disabled**, the system does not terminate outstanding transactions when it receives a BYE. Instead, it only terminates them when either it receives the final response belonging to the transaction or the transaction times out.

Default: enabled
Values: enabled | disabled
Example: **set terminate-transaction-on-bye disabled**

to-header-follows-contact-header: Specifies that the system should move the URI and header parameters from the Contact header to the To header. This property is used for a specific phone switching application only.

Default: disabled
Values: enabled | disabled
Example: **set to-header-follows-contact-header enabled**

inleg-tos: Determines the TOS value setting for the in-leg of the session. The TOS value determines the quality of service that the call receives. If set to **preserve**, the system uses the TOS value in the first received message of the session (normally a REGISTER or INVITE). If set to **overwrite**, the system marks the TOS field of all packets it sends out on the inleg with the value you specify. Enter a number that represents the 8-bit Differentiated Services (DS) field of the IP packet in decimal format, such as 26 for 00011010 or104 for 01101000. This value can be of use to upstream devices.

Default: preserve; if you select overwrite, the default value is 0
Values: preserve | overwrite *value*
Example: **set inleg-tos overwrite 22**

outleg-tos: Determines the TOS value setting for the out-leg of the session. The TOS value determines the quality of service that the call receives. If set to **preserve**, the system uses the TOS value in the first received message of the session (normally a REGISTER or INVITE). If set to **overwrite**, the system marks the TOS field of all packets it sends out on the outleg with the value you specify. Enter a number that represents 8-bit Differentiated Services (DS) field of the IP packet in decimal format, such as 26 for 00011010 or 104 for 01101000. This value can be of use to upstream devices.

Default: preserve; if you select overwrite, the default value is 0
Values: preserve | overwrite *value*
Example: **set outleg-tos overwrite 22**

generate-final-response: Sets whether the system generates a final response when it cannot forward an OPTIONS request or does not receive a response. When **enabled**, the system sends a 408 Request Timeout back to a UAC if it does not receive a response to an OPTIONS message from the UAS. When **disabled**, it does not generate a 408.

Default: enabled
Values: enabled | disabled
Example: **set generate-final-response disabled**

b2bua-generate-100-trying: Sets whether the system sends a 100 Trying message back to the caller before forwarding an INVITE. When **enabled**, the message is sent.

Default: enabled
Values: enabled | disabled
Example: **set b2bua-generate-100-trying disabled**

auto-accept-reinvite-with-no-sdp-on-in-leg: Specifies whether the system responds with a 200 OK or forwards a REINVITE. When a call is first established, the system receives an INVITE on the in-leg and forward it out the out-leg. Once the call has been established, either side may send a REINVITE. If this property is **enabled** and the call is received on the in-leg, the system responds with a 200 OK. If **disabled**, the system forwards the REINVITE to the out-leg.

Default: disabled
Values: enabled | disabled
Example: **set auto-accept-reinvite-with-no-sdp-on-in-leg enabled**

auto-accept-reinvite-with-no-sdp-out-leg: Specifies whether the system responds with a 200 OK or forwards a REINVITE. When a call is first established, the system receives an INVITE on the in-leg and forward it out the out-leg. Once the call has been established, either side may send a REINVITE. If this property is **enabled** and the call is received on the out-leg, the system responds with a 200 OK. If **disabled**, the system forwards the REINVITE to the in-leg.

Default: disabled
Values: enabled | disabled
Example: **set auto-accept-reinvite-with-no-sdp-out-leg enabled**

strip-authint-qop: Determines whether the system modifies the Quality of Protection (QoP) parameter. The QoP parameter defines the type of authentication the server requires, either auth or auth-int. Auth verifies the sender using a shared secret; auth-int verifies both the sender and the integrity of the message (as defined in RFC 2617, *HTTP Authentication: Basic and Digest Access Authentication*). When **enabled**, if the QoP parameter of a 401 or 407 challenge response offers both auth and auth-int, the system removes the auth-int option and forwards the message with just auth required. If **disabled**, the QoP parameter remains unchanged.

Default: disabled
Values: enabled | disabled
Example: **set strip-autnint-qop enabled**

ignore-cancel-branch: Specifies whether to use the branch value to cancel a transaction. By default (**disabled**) the system uses the branch value (a string that identifies the transaction) in the top VIA header. When **enabled**, the system uses the FROM and TO tags in the call ID to identify and cancel the transaction. Use this property only if a device sends the wrong branch value.

Default: disabled
Values: enabled | disabled
Example: **set ignore-cancel-branch enabled**

symmetric-signaling: Sets whether the system uses the VIA or CONTACT header to send messages for a call. An incorrectly functioning NAT device may only partially change a CONTACT or VIA header, causing the system to be unaware that the sending endpoint is behind a NAT. When **enabled**, this property ensures that the system always sends responses and future requests for that call to the IP address and port number from which the request originated. Enable this property only if instructed to do so by Technical Support.

Default: disabled
Values: enabled | disabled
Example: **set symmetric-signaling enabled**

sips-uri-scheme-setting: Controls the SIP scheme the system uses in the URI of outbound calls. For example, when one endpoint is using TLS and another UDP, the SIP scheme is inconsistent. The default setting, **auto**, allows the system to determine the scheme. To force the setting, select either **secure** (for sips) or **not-secure** (for sip).

Default: auto
Values: auto | secure | not-secure
Example: **set sips-uri-scheme-setting not-secure**

redirect-preserve-session-config: Specifies where the system derives the session configuration from when a WSDL-generated INVITE results in a 3xx response. When **enabled**, the system applies the WSDL request's session configuration to the redirect message (the session configuration within the WSDL request is maintained throughout the session). When **disabled**, the default, the system uses the session configuration associated with the original INVITE for the redirect message as well, rather than the WSDL-based session configuration.

Default: disabled
Values: enabled | disabled
Example: **set redirect-preserve-session-config enabled**

max-forwards: Specifies the value to insert into the Max-Forward header of the SIP message. This value to determine how many future hops the message is allowed, and is decremented by 1 at each hop. If a message arrives with a Max-Forwards value higher than the value set with this property, the system resets the value to this configuration setting. If the value that arrives is equal to or lower than this setting, the system decrements that value by one.

Default: 70
Example: **set max-forwards 50**

enum-fail-response: Specifies the customized response code and string that the system sends as a result of a failed ENUM lookup. When set to **ignore**, the system ignores the failed lookup and continues to try and transmit the message. (The system may be able to resolve the address via a configured dial plan, for example.) If it is not able to ultimately resolve the TO header, the system sends a 404 Not Found message back to the sender. When set to **reject**, the system rejects the incoming message immediately and sends the configured code and string to the sender. You may use this, for example, if the originating phone/gateway requires a custom cleardown (SIP response) when ENUM lookup fails.

Default: ignore
Values: ignore | reject [*result-code*][*result-string*]
Example: **set enum-fail-response reject 999 "ENUM failure"**

dns-fail-response: Sets the response code that the system sends to an endpoint when a call is received, the **dns-client-settings > routing-last-resort-dns** property is applied, and the lookup fails.

Default: 404
Example: **set dns-fail-response 400**

dns-fail-response-string: Sets the response string that the system sends to an endpoint when a call is received, the **dns-client-settings > routing-last-resort-dns** property is applied, and the lookup fails.

Default: Not Found
Example: **set dns-fail-response-string "Bad Request**

update-moc-via-csta: Specifies whether to send user state information relevant to the Microsoft Office Communicator (MOC) client. If **enabled**, the system notifies the MOC of the change in call state via CSTA when a call is connected or terminated. This

provides MOCs that are logged in with remote-call-control enabled with correct presence information (in-a-call or available).

Default: disabled
Values: enabled | disabled
Example: **set update-moc-via-csta enabled**

supported-inleg: Adds a new Supported header or overwrites the existing header on the inbound leg of the SIP message with the specified text string. Use this property, for example, to support Provisional Response Acknowledgement (RFC 3262) PRACK insertion.

Default: There is no default setting
Example: **set supported-inleg 100rel, timer**

supported-outleg: Adds a new Supported header or overwrites the existing header on the outbound leg of the SIP message with the specified text string. Use this property, for example, to support Provisional Response Acknowledgement (RFC 3262) PRACK insertion.

Default: There is no default setting
Example: **set supported-outleg "Supported: 100rel"**

persistent-destination: Specifies whether to send a message to a new remote address based on a new remote contact. When set to true, the system ignores the Contact URI and forwards messages to their original destination address. When set to false, the system uses the Contact URI in each response to determine the destination of the next message.

Default: true
Values: true | false
Example: **set persistent-destination false**

loop-detection-threshold: Specifies how many times a message can be processed by the system before it is considered to be in a loop. If this threshold is reached, the system rejects the call and sends a 482 (Loop Detected) response code back to the caller.

Default: 2
Values: Min: 1 / Max: 256
Example: **set loop-detection-threshold 4**

sip-signaling-encryption: Specifies whether a call coming in over a TLS connection is allowed or dropped.

Default: unspecified
- **Values: unspecified**: Any transport protocol is allowed; the system continues processing regardless of the connection type.
- **prohibited**: TLS is not allowed. If a call comes in over a TLS connection, the system drops the call.
- **required**: Any transport protocol is allowed; the system continues processing regardless of the connection type.

Example: **set sip-signaling-encryption required**

share-transport-connection: Specifies when the ME should reuse existing TCP or TLS connections. The setting specifies the criteria for matching between the existing connection and the current transaction (request or response). The ME reuses the connection. If there is no match, the system opens a new connection.

Default: remote-addr
- **Values: remote-ip**: The remote IP address

- **remote-addr**: The remote IP address and phone number
- **no-local-port**: The local IP address, the remote IP address, and the remote port
- **all**: The remote port and IP address and the local port and IP address

Example: **set share-transport-connection all**

propagate-RR-headers: *Secondary property.* Modifies headers in a B2B configuration. When **enabled**, the system propagates Record Route and Via headers from received to transmitted SIP messages. When **disabled**, the system creates these headers for each transmitted message. Do not modify this setting unless instructed to do so by Technical Support.

Default: disabled
Values: enabled | disabled
Example: **set propagate-RR-header enabled**

ignore-via-port: *Secondary property.* Specifies whether to ignore the port reported in the top-level Via header. Typically, when the system matches a request to an existing transaction (request/response), it checks to make sure that, among other fields, the port is the same in both. This property should be **enabled** if there is a device along the path that incorrectly changes the Via header port. By default, this property is **disabled** (port numbers are checked).

Default: disabled
Values: enabled | disabled
Example: **set ignore-via-port enabled**

ignore-route-header: *Secondary property.* Specifies whether the ME uses the Route header to forward messages. By default (**disabled**), the system does use the Route header. When **enabled**, the ME message forwarding logic does not use the Route header and instead uses other options, as set in the session configuration (Request URI, Contact URI, DNS, etc.).

Default: disabled
Values: enabled | disabled
Example: **set ignore-route-header enabled**

make-provisional-resp-reliable: *Secondary property.* Specifies a single provisional response code to which the system will add "Required: 100rel" and "RSeq: *nnn*" before forwarding. These headers are only added, however, if the initial INVITE indicates support for 100rel. Note that the **forward-provisional-ack** property should be disabled when using this so that the PRACK from the UAC wont be forwarded to the UAS. If you are using third-party call control, the **use-183-for-ringing-with-sdp** converts a 180 response to a 183. In this case, set this property to 183 instead of 180.

Default: 0
Example: **set make-provisional-resp-reliable 3**

allow-redirect: When enabled, the ME is able to redirect incoming calls to other servers.

Default: enabled
Values: enabled | disabled
Example: **set allow-redirect disabled**

strip-received-from-top-via: When enabled, the ME strips off the "Received" and "Rport" fields in the top header before it sends a 200OK response.

Default: disabled
Values: enabled | disabled
Example: **set strip-received-from-top-via enabled**

last-resort-request-uri: If both the dial plan and location cache look ups fail, when this parameter is enabled, the ME attempts to route the call using the Request-URI of the incoming INVITE. If you want to limit routing to dial plans and the location cache, set this parameter to disabled.

Default: enabled
Values: enabled | disabled
Example: **set last-resort-request-uri disabled**

preserve-session-config-on-3xx: Apply the same session configuration that was used for the initial INVITE when sending the INVITE for a 302. When disabled, the AA-SBC removes the dial plan and server session configurations from the merged configuration.

Default: disabled
Values: enabled | disabled
Example: **set preserve-session-config-on-3xx**

third-party-call-control

Configures call control, allowing the ME or a CSTA client to control (become the third party) in a call. Specifically, this object controls the WAV files that the ME should play and the external status events reported to an external server for calls created by the ME device. The third-party call controller (3PCC) functionality is used, for example, to enable the interworking of a uaCSTA with the Broadworks Open Client Interface. The ME converts between the two call control protocols. Phone control can be integrated, for example, into Microsoft Office applications using the Phone Controls interface. This object can also be enabled in certain situations involving LCS/Sametime interworking and other advanced the ME applications. In all cases, it should only be enabled at the direction of Technical Support.

When the ME functions as a 3PCC device by initiating communications to each endpoint in the session, this object configures the specific WAV file(s) to play in response to the state of the call destination. Specifically, when the ME receives an instruction from the CSTA client to establish a call, it first makes a call to the originator of the call and then to the destination. The destination responds with call progress information. If that information indicates that the phone is ringing, and the **ringback-file** property is configured, the ME plays the specified file. If the phone is busy or set to appear so, the ME plays any configured **busy-file** recording.

> **Note:** You must set the admin property of this object to enabled if you are implementing a Sametime-to-LCS federation. If you are running Sametime-to-Sametime or LCS-to-LCS, the admin property must be disabled.

Assuring Pre-Call Announcements After Failover

When the **pre-call-announcement** property is set, the ME plays a WAV file for the caller prior to the call connecting. If you set this property and want to ensure the pre-call announcement is played in the event that the master box fails over to a backup box, you must copy files to the backup boxes using the file mirror service. To do so:

1. Enable the **file-mirror** master service and configure a **file-mirror-directory**. List all boxes as possible hosts with the **host-box** property. For example:

```
config file-mirror> set admin enabled
config file-mirror> set file-mirror-directory /cxc_common/mirror
config file-mirror> set host-box cluster\box 1
```

```
config file-mirror> set host-box cluster\box 2
config file-mirror> set host-box cluster\box 3
```

1. Upload the pre-call announcement WAV file to the configured mirror directory.
2. Execute the **file-mirror-service make-available** action. For example:

```
NNOS-E> file-mirror-service make-available /cxc_common/mirror/announcement.wav
```

By executing the action after uploading files to the file mirror directory, these files will be available to master and drone in the event of failover.

Syntax

```
config vsp default-session-config third-party-call-control
config vsp policies session-policies policy name rule name session-config
third-party-call-control
config vsp dial-plan dial-prefix entryName session-config third-party-call-control
config vsp dial-plan route name session-config third-party-call-control
config vsp dial-plan source-route name session-config third-party-call-control
config vsp session-config-pool entry name third-party-call-control
```

Properties

admin: Enables or disables the ability to control externally originated calls. (If the system is the call originator, this setting is ignored.)

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

status-events: Specifies when to generate status events if the system is functioning as a third-party call controller (3PCC), initiating communications to each endpoint in the session. Specifically, when the system receives an instruction from the CSTA client to establish a call, it first makes a call to the originator of the call and then to the destination.

Default: both

- **Values: originate**: Contact established with the call originator
- **answer**: Contact established with the call destination
- **neither**: No status events are sent
- **both**: Contact established with the call originator and destination

Example: **set status-events originate**

handle-refer-locally: Specifies whether the system forwards REFER messages or whether it consumes the message and generates an INVITE for the call. When an end user wants to transfer a call, the system receives a REFER message. When this property is set to **enabled** (the default), the system terminates the message, generates an INVITE, and makes a call on behalf of that call transfer. When set to **disabled**, the system forwards the call to the upstream server for handling.

Default: enabled
Values: enabled | disabled
Example: **set handle-refer-locally disabled**

refer-maintain-identity: Specifies how the system handles the TO and FROM URIs when processing a REFER request. If set to **true**, the system keeps the original TO and FROM components of the URI instead of replacing them with the REFER destination. When set to **false**, the default, they are replaced.

For example, suppose a third-party call is initiated from A to B, and B subsequently sends a REFER to transfer the call. If this property is set to **true**, the call appears to A as

if it were still connected to B. If set to **false**, the call appears to A as if it were connected to C.

Default: false
Values: true | false
Example: **set refer-maintain-identity true**

ringback-file: Specifies the WAV file that the system plays while the remote endpoint is ringing. If the system is the call originator, and this property is configured, the system plays the file regardless of the **admin** setting. If the call is externally originated, the ringback is generated locally by the originating phone.

Default: There is no default setting
Example: **set ringback-file /cxc/cxc_common/holdRigning.wav**

busy-file: Specifies the WAV file that the system plays when the remote endpoint is set to do-not-disturb or is busy. If the system is the call originator, and this property is configured, the system plays the file regardless of the **admin** setting. If the call is externally originated, the busy signal is generated locally by the originating phone.

Default: There is no default setting
Example: **set busy-file /cxc/cxc_common/busy1.wav**

pre-call-announcement: Specifies the WAV file that the system plays for the caller when this property is set. The callee does not hear the file (the file is played before the connection to the destination is attempted). In contrast, use the introduction property of the **media** object to play an recording for both sides.

Default: There is no default setting
Example: set pre-call-announcement cxc/'cxc_common/youhavewon.wav

terminate-after-pre-call-announcement: Specifies whether to end a call once the pre-call announcement has been played. When **enabled**, the system plays the file specified with the **pre-call-announcement** property and then terminates the call.

Default: disabled
Values: enabled | disabled
Example: set terminate-after-pre-call-announcement enabled

handle-replaces-locally: Specifies whether the system forwards INVITEs with a Replaces header or whether it consumes the message and generates a new INVITE for the call. When an endpoint transfers a call, the ME receives a REFER message. When this property is **enabled**, the system terminates the message and completes the replacement locally. When **disabled**, the system forwards the call to the upstream server for handling.

Default: disabled
Values: enabled | disabled
Example: **set handle-replaces-locally enabled delayed-ack**: Specifies when the system sends an ACK in response to an originating call leg. When **enabled**, the system does not send an ACK to the originating call leg until it receives a 200/OK from the answering call leg. When **disabled**, the system sends an ACK immediately, prior to receiving a response from the destination.

Default: disabled
Values: enabled | disabled
Example: set delayed-ack enabled

include-reason-in-bye: Specifies whether to include the optional Reason header in a BYE request. Enable this to resolve the Heterogeneous Error Response Forking Problem (HERFP), as stated in RFC 3326. The Reason field helps in properly updating requests when proxy servers forward requests to multiple contacts associated with a call. When **enabled**, the default, the Reason header is included.

Default: enabled
Values: enabled | disabled
Example: set include-reason-in-bye disabled

always-apply-req-uri-spec: Specifies when to include the settings of the **request-uri-specification**. When **enabled**, the system applies the settings to the initial INVITE as well as any re-INVITEs. When **disabled**, the system only applies the settings on the initial INVITE.

Default: enabled
Values: enabled | disabled
Example: **set always-apply-req-uri-spec disabled media-shuffle**: Specifies whether to generate SDP locally in H.323-to-SIP calls when the H.323 endpoint is not in fast-start mode. (Fast-start mode is the option of having the H.323 endpoint provide media information in the initial call setup.) This property is used for injecting SDP into INVITEs for the SIP Delayed Offer to Early Offer conversion process.

In a non-fast-start case, the system has no media information when it sends out an INVITE. When this property is **enabled**, the system generates some SDP initially. When it receives the real media information from the H.323 side, the system then reinvites the SIP endpoint with the correct information. When media-shuffle is **disabled**, the system sends the INVITE with no SDP.

Default: enabled
Values: enabled | disabled
Example: set media-shuffle disabled

park-incoming-calls: Allows establishment of an inbound 3PCC session. When **enabled**, any incoming call is answered by the system (similar to the results of the **call-control park** action). Use this, for example, in conjunction with the **pre-call-announcement** property.

Default: disabled
Values: enabled | disabled
Example: set park-incoming-calls enabled

parked-call-greeting: Specifies the WAV file that the system plays for the caller when the **park-incoming-calls** property is set to enabled. The callee does not hear the file (the file is played before the connection to the destination is attempted). In contrast, use the introduction property of the **media** object to play a recording for both sides.

Default: There is no default setting
Example: **set parked-call-greeting /cxc/cxc_common/park.wav**
terminate-after-greeting: Specifies whether the ME terminates a call after playing the file specified with the **parked-call-greeting** property. The **park-incoming-calls** property must be set to enabled for this property to apply.

Default: disabled
Values: enabled | disabled
Example: set terminate-after-greeting enabled

terminate-update-locally: Specifies whether the system responds locally to UPDATE messages. When this is **enabled**, if an endpoint sends an UPDATE message, the system responds to the update with a 200 OK and the SDP from the remote endpoint. If **disabled**, the system forwards the UPDATE to the remote endpoint.

Default: disabled
Values: enabled | disabled
Example: set terminate-update-locally enabled

terminate-reinvite-locally: Specifies whether the system responds locally to REINVITE messages. When this is **enabled**, if an endpoint sends a REINVITE

message, the system responds to the reinvite with a 200 OK and the SDP from the remote endpoint. If **disabled**, the system forwards the REINVITE to the remote endpoint.

Default: disabled
Values: enabled | disabled
Example: **set terminate-reinvite-locally forking-early-media-inhibit**: Specifies whether to strip SDP from a provisional response. Use this in some cases when using sequential or parallel forking to prevent an endpoint from being swamped by provisional response content. The saved SDP is later inserted into the 200 OK (unless more current SDP is available in the 200 OK), allowing the calling phone to receive just one SDP.

Default: disabled
Values: enabled | disabled
Example: set forking-early-media-inhibit enabled

forward-unresolved-replaces: *Secondary property.* Specifies whether to forward a 481(Call Leg Not Found) message when the system cannot find the call leg information in the Replaces header. In an attended transfer, A calls B, and then B calls C, makes the connection, and completes the transfer. To do so, B sends a REFER to C. Included in that REFER is a Replaces header, with information identifying the call-leg from A to B (which will be replaced). As a B2B, the system must convert the identifying information in the call leg on the B side to that which represents the call-leg on the A side. When this property is **disabled**, if the system cannot find the call-leg in the Replaces header it responds with a 481 to the system receiving the REFER. If **enabled**, the system forwards the REFER to C without the conversion described.

Default: disabled
Values: enabled | disabled
Example: **set forward-unresolved-replaces enabled extract-refer-to-header**: *Secondary property.* Specifies whether to use the fields of the REFERTO header. When **enabled**, the ME extracts the header specifications from the REFERTO header and includes them in the resulting INVITE.

Default: disabled
Values: enabled | disabled
Example: set extract-refer-to-header enabled

reinvite-preserve-media: *Secondary property.* Specifies whether to re-use SDP. When set to **enabled**, in cases where an REINVITE is received and there is no SDP, the ME uses the previous SDP.

Default: disabled
Values: enabled | disabled
Example: set reinvite-preserve-media enabled

media-forward: *Secondary property.* Specifies whether the ME can respond to forwarded requests from the NICE media server. When this property is **enabled**, the system treats all INVITEs as requests from the NICE media server. (If the INVITE is not from the NICE media server, the system will reject the request after scanning and not finding NICE-specific content.)

Default: disabled
Values: enabled | disabled
Example: set media-forward enabled

track-to-user: *Secondary property.* Allows the NICE media server to request forwarding sessions based on the user listed in the TO URI.

Default: disabled

Values: enabled | disabled
Example: **set track-to-user enabled force-retrieve-on-delayed-offer-while-held**: *Secondary property.* Specifies system behavior when an INVITE with no SDP is received for a call on hold. When enabled, if an INVITE with no SDP is received, it is treated as a receive request and the call is taken off hold. When disabled, call status does not change from that event.

Default: disabled
Values: enabled | disabled
Example: set force-retrieve-on-delayed-offer-while-held enabled

reinvite-delayed-offer-wait-on-ack: *Secondary property.* Specifies how the ME responds to a REINVITE with no SDP. When **enabled**, the ME postpones re-inviting the remote end (accepting the REINVITE locally) until receiving the SDP contained in the ACK. The default setting is **disabled.**

Default: disabled
Values: enabled | disabled
Example: set reinvite-delayed-offer-wait-on-ack enabled

use-183-for-ringing-with-sdp: *Secondary property.* Specifies the SIP code that the ME forwards with messages it receives that contain early media. If this property is **disabled**, the system forwards the message with a 180 (Ringing) code. If **enabled**, the system changes the code to 183 (Session Progress) when forwarding the message.

Default: disabled
Values: enabled | disabled
Example: **set use-183-for-ringing-with-sdp enabled strip-require-100-rel**: *Secondary property.* Specifies whether the "Require: 100rel" line is stripped from responses. When enabled the line is removed from forwarded responses.

Default: disabled
Values: enabled | disabled
Example: **set strip-require-100-rel enabled**
forward-all-parallel-provisional-responses: *Secondary property.* Sets how provisional responses are handled when parallel forking is enabled. Parallel forking results in a single incoming INVITE generating multiple simultaneous outgoing INVITEs, which can cause issues regarding which audio stream to apply if multiple provisional responses are returned. When this property is **enabled**, if a provisional response is received during a parallel forking attempt, every 18*x* response received prior to one of the outgoing calls completing successfully is forwarded to the originating agent. If this property is **disabled**, provisional responses are only forwarded when received from the first server on the list of destinations being attempted.

Default: disabled
Values: enabled | disabled
Example: set forward-all-parallel-provisional-responses enabled

include-id-in-refer-notify: *Secondary property.* Specifies whether, when a NOTIFY is sent reporting the final status of a REFER call, the ME includes an ID field in the NOTIFY. When **enabled**, the ME includes the ID.

Default: disabled
Values: enabled | disabled
Example: **set include-id-in-refer-notify enabled notify-dtmf-event-if-allowed**: *Secondary property.* Sets whether the system sends a NOTIFY event in response to receiving DTMF. To use this property, the remote user agent must allow KPML or telephone events. When the ME receives an Allow-Events header, and this property is **enabled**, upon receiving DTMF the system sends a NOTIFY event with the RFC 2833 representation of the received DTMF.

Default: disabled
Values: enabled | disabled
Example: set notify-dtmf-event-if-allowed enabled

terminate-hold-retrieve-locally: When this property is enabled, if a re-INVITE is received with an SDP that indicates it is either a hold or retrieve request, the ME accepts the re-INVITE locally with the SDP acknowledging the hold or retrieve and the message is not forwarded. When this property is disabled, re-INVITE messages with an SDP that indicates hold or retrieve receive no special treatment.

Default: disabled
Values: enabled | disabled
Example: set terminate-hold-retrieve-locally enabled

reinvite-originator: When enabled, the ME reinvites the original UAC after the call is initially set up.

Default: disabled
Values: enabled | disabled
Example: set reinvite-originator enabled

skip-shuffle-complete-if-anchored: When enabled, no reinvite is sent forwarding the SDP contained in the ACK for calls with anchored media.

Default: disabled
Values: enabled | disabled
Example: **set skip-shuffle-complete-if-anchored enabled**
forward-302-division-header: *Secondary property*. When enabled, if a 302 Redirected response with a Diversion: header is received by the ME, the Diversion: header is forwarded in the response.

Default: enabled
Values: enabled | disabled
Example: **set forward-302-division-header disabled strip-route-headers**: *Secondary property*. Specifies which route headers back-to-back user-agents strip.

Default: all
- Values: none
- all
- top

Example: **set strip-route-headers none**

inhibit-shuffle-update: Controls whether a re-INVITE message is generated and sent to the destination server to update the SDP (as some user agents initiate SIP calls using an INVITE without an SDP.)

Normally, when the **media-shuffle** property is enabled, the SDP is generated locally for the outgoing call, and the answer SDP from the destination will be returned to the originator as the offer SDP. After the originator responds with the answer SDP, a reinvite will be generated to the destination to update the SDP.

When **inhibit-shuffle-update** is set to enabled, this re-INVITE will not be sent to update the SDP. The inhibit-shuffle-update property setting is only valid when media-anchoring is enabled. If inhibit-shuffle-update is enabled without media anchoring enabled, audio problems will result.

Default: disabled
Values: enabled | disabled
Example: set inhibit-shuffle-update enabled

refer-notify-100-trying: Controls the behavior of the ME when a REFER message is received and the referrer disconnects before the resulting transfer has completed.

Example: **set refer-notify-100-trying attempt1 forward-302-diversion-header**: When enabled, if a 302 Redirected response with a Diversion: header is received by the ME, the Diversion: header is forwarded in the response.

Default: enabled
Values: enabled | disabled
Example: **set forward-302-diversion-header disabled**

media-forward-reference-direction: Identifies which leg of a call is to the call-center PBX, mapping the Rx and Tx streams to match the NICE equipment Rx and Tx streams.

Default: out-leg
Values: in-leg | out-leg
Example: set media-forward-reference-direction in-leg

inhibit-100-trying-for-reinvite: When enabled, the ME does not send out a 100 Trying when it receives a re-INVITE. When disabled, the ME does send out a 100 Trying in response to a re-INVITE.

Default: enabled
Values: enabled | disabled
Example: set inhibit-100-trying-for-reinvite disabled

allow-lcr-for-refer: When running the route server under third-party-call-control, information may need to be obtained off of the ME, causing a delay. When this parameter is enabled, the ME can avoid potential problems caused by this delay.

Default: disabled
Values: enabled | disabled
Example: **set allow-lcr-for-refer enabled transfer-file**: Select the file of the media to be played while a blind transfer is taking place.

Default: There is no default setting
Example: set transfer-file data1

notify-dtmf-when complete: *Secondary property*. Specifies whether the ME forwards the DTMF notification via the Notify request sent at the beginning of the DTMF tone or the Notify request sent after the DTMF tone.

Default: enabled
Values: enabled | disabled
Example: **set notify-dtmf-when-complete disabled**

inhibit-provisional-response-after-prack: *Secondary property*. When enabled, the ME does not send 18x messages after receiving a Prack. This prevents problems when invalid codecs are presented in the 18x's SDP after valid codecs have already been sent.

Default: disabled
Values: enabled | disabled
Example: **set inhibit-provisional-response-after-prack enabled**

call-control-events-version: When custom-event-fields object on the ME is configured, when this property is set to **custom**, it enables the ME to add custom information in call control events (for example, callCreated, callConnected, and callTerminated).

Default: legacy

- Values: legacy: The events generated follow the legacy format.
- custom: The events generated have new custom fields.

Example: **set call-control-events-version custom**

terminate-message-locally: Indicates whether the ME forwards a received SIP MESSAGE after responding with a 200 OK.

Default: disabled
Values: enabled | disabled
Example: **set terminate-message-locally enabled**

inhibit-outgoing-reinvites: Specifies whether the ME responds locally to REINVITE messages. When **enabled**, the ME response to the REINVITE with a 200 OK and SDP from the remote endpoint before forwarding the message. When **disabled**, the ME forwards the REINVITE to the remote endpoint.

Default: disabled
Values: enabled | disabled
Example: **set inhibit-outgoing-reinvites enabled**

forward-provisional-media: Indicates whether the ME plays a ringback-file. When **enabled**, the ME does not play the ringback-file when establishing a third-party initiated call and the out-leg 18X contains SDP.

Default: disabled
Values: enabled | disabled
Example: **set forward-provisional-media enabled**

disconnect-time: *Secondary property*. Sets the number of seconds from the time a session is established until it is terminated. Reinvites and updates do not refresh this time.

Default: 0
Values: Min: 0 / Max: 10000000
Example: **set disconnect-time 100000**

inhibit-anchored-update: Specifies whether a re-INVITE message is generated and sent to the destination server to update the SDP for a third-party initiated call.

> **Note:** This property is valid only when the **session-config > media > anchor** property is set to **enabled**.

Default: disabled
Values: enabled | disabled
Example: **set inhibit-anchored-update enabled**

to-uri-specification

Specifies what derives the content of the fields of the TO URI when the ME transmits a message. For example, if the **user** property is set to **to-uri**, the ME replaces the user field of the TO URI with data from the user field of the incoming TO URI. If set to **omit**, the user field is left blank. Or, you can enter any string that you want placed in the user field.

Altering URIs

In some cases, it is necessary to change a portion of the URI (outbound only) so that the next-hop server can accept the SIP message when it arrives. For example, a server may require a part of the URI to be in a specific format. The ME allows you to modify the portions of the REQUEST, TO, and/or FROM URI so that the header matches any necessary requirements.

When you select to alter the URI, you can set the ME to replace the specified field with one of the following. Properties have some or all of the same selection options. These options apply to the **to-uri-specification**, **from-uri-specification**, **request-uri-specification**, and **inbound-controls** objects. The following defines the common options, where the ME device:

- **request-uri**: Derives values from the incoming REQUEST URI
- **to-uri**: Derives values from the incoming TO URI
- **from-uri**: Derives values from the incoming FROM URI
- **omit**: Leaves the field blank
- **next-hop**: Derives values from the IP address of the next-hop server (not applicable to the **port** field)
- **local**: Derives values from the IP address of the ME device (not applicable to the **transport** field)
- **omit-phone-context**: Does not replace the field, but removes the phone context portion, if applicable (**user** property only)
- **next-hop-domain**: Derives values from the IP address of the next-hop server or phone. If a server, the next-hop is learned from the peer property of the server; if a phone, it is the IP address of the phone (**host** property only)
- **next-hop-ip**: Derives values from the next-hop IP address, which is specified in the **peer** property of the same object (**host** property only)
- **local-ip**: Derives values from the IP address from the interface the packet goes out on (**host** property only)
- **directory** *directoryReference*: Derives values from a user alias. When selected, the ME looks for all aliases associated with the user listed in the To, Request, and From fields of the URI. The ME then uses the alias associated with the referenced directory. (**host** property only)
- **string**: Writes the specified string to the field (not applicable to the **transport** field)
- **none** or **same-uri**: The URI is not modified

For example, it is not uncommon for a carrier to change a user ANI (automatic number identification) to the modified number used by the DNIS (dialed number identification service). For the ME to correctly forward the call, it must put the user ANI back into the From header. It can do this if you configure the **host** property to a referenced directory that can identify the user DNIS alias.

Syntax

```
config vsp default-session-config to-uri-specification
config vsp policies session-policies policy name rule name session-config
to-uri-specification
config vsp dial-plan dial-prefix entryName session-config to-uri-specification
config vsp dial-plan route name session-config to-uri-specification
config vsp dial-plan source-route name session-config to-uri-specification
config vsp session-config-pool entry name to-uri-specification
```

Properties

user: Specifies how to derive the value of the user field of the TO URI.

Default: to-uri
Values: request-uri | to-uri | from-uri | omit | next-hop | local |

omit-phone-context | *string*
Example: **set user request-uri**

host: Specifies how to derive the value of the host field of the TO URI.

Default: to-uri
Values: request-uri | to-uri | from-uri | omit | next-hop | next-hop-domain | local-ip | *string*
Example: **set host omit**

port: Specifies how to derive the value of the port field of the TO URI.

Default: to-uri
Values: request-uri | to-uri | from-uri | omit | *string*
Example: **set port omit**

display: Specifies how to derive the value of the display field of the TO URI.

Default: to-uri
Values: request-uri | to-uri | from-uri | omit | next-hop | *string*
Example: **set display next-hop**

transport: Specifies the value of the transport field of the FROM URI. In addition to using the value from other fields of the incoming URI, you can set the transport method to UDP, TCP, or TLS.

You cannot enter a string for this property.

Default: to-uri
Values: request-uri | to-uri | from-uri | omit | UDP | TCP | TLS | *string*
Example: **set transport omit**

use-param: Specifies whether the User parameter in the TO URI of the SIP header is maintained or removed when the system forwards a message. If set to **keep**, the message is forwarded with the parameter as it was received. If set to **omit**, the entire user=*param* is removed from the TO URI.

Default: omit
Values: omit | keep
Example: **set use-param keep**

user-truncate-non-digits: Specifies whether to remove non-digits from the User portion of the TO URI in INVITE messages. When **enabled**, the system removes all non-digits.

Default: disabled
Values: enabled | disabled
Example: **set user-truncate-non-digits enabled**

strip-digits: Specifies the number of digits to strip from the User portion of the To URI. Use this, for example, if the original INVITE contains extra digits that would be problematic to the downstream server. Digits are removed beginning at the string that immediately follows the sip: or sips: portion.

Default: 0
Example: **set strip-digits 2**

transcoding-policy

The transcoding policy object allows you to configure the transcoding policy for the ME. This includes defining the preferred codec as deduced from SDP offers and answers, adapting to match received codecs, and rewriting rfc-2833 headers when encoding audio.

Syntax

```
config vsp session-config pool entry media transcoding-policy
```

Properties

media-types: Sets a threshold, in megabytes, at which the system no longer writes recorded calls or IM files to the disk drive. The system sends a warning message to the event log (and an SNMP trap) indicating that space on the internal disk drive has been exceeded. The system checks the fail threshold each time it receives a call.

Default: There is no default setting
Values: pcma | pcmu | g7221 | g723 | g728 | g729 | g726-16 | g726-24 | g726-30 | g726-32 | gsm | gsm-ami | iLBC
Example: **set media-types gsm**

most-preferred: When true, the ME forces audio to only use the most preferred codec.

Default: false
Values: true | false
Example: **set most-preferred true**

symmetric-codec: When true, the ME adapts and matches the correct codec when the endpoint has switched the "primary" codec.

Default: false
Values: true | false
Example: **set symmetric-codec true**

balance-ptime: When true, the ME attempts to balance RTP packet times with the SDP.

Default: true
Values: true | false
Example: **set balance-ptime false**

auto-release: When true, the ME attempts to release transcode resources when auto-anchoring is enabled.

Default: true
Values: true | false
Example: **set auto-release false**

block-unknown: When true, the ME blocks unnegotiated packet types.

Default: false
Values: true | false
Example: **set block-unknown true**

decode-telephone-events: When true, the ME decodes telephone-events into audio during transcoding when both sides do not support telephone-events.

Default: false
Values: true | false
Example: **set decode-telephone-events true**

trusted-interface-settings

Sets an interface to allow non-LCS devices to interact with LCS clients. The interface is a reference to a previously configured enterprise server: the LCS server that the ME uses as a gateway for non-LCS traffic. The server should be configured to treat the ME as authenticated.

For example, this interface would allow a SNOM phone to call an LCS Windows Messenger client. When an INVITE comes in from the SNOM phone, if the policy has

the **trusted-server** property configured, the ME forwards the INVITE to that server. Since the server has been configured to "treat as authenticated" traffic received from the ME on that interface, it does not prompt the SNOM phone, for authentication. The INVITE is forwarded to the Windows Messenger client, and the client sees the incoming call.

Syntax

```
config vsp default-session-config trusted-interface-settings
config vsp policies session-policies policy name rule name session-config
trusted-interface-settings
config vsp dial-plan dial-prefix entryName session-config
trusted-interface-settings
config vsp dial-plan route name session-config trusted-interface-settings
config vsp dial-plan source-route name session-config trusted-interface-settings
config vsp session-config-pool entry name trusted-interface-settings
```

Properties

trusted-server: Specifies the server that is configured to recognize traffic from the system as authenticated. Enter a reference to a previously configured server.

Default: There is no default setting
Example: **set trusted-server vsp enterprise servers lcs lcs-server**

uui-header

Makes modifications to the user-to-user information (UUI) header. The UUI can be used for passing the universal call ID (UCID) and other session information to the NICE media server. If this object is configured, when the ME receives a SIP message it checks for the UUI header and applies the action defined with the **replace-existing-header** property. If no UUI header exists, the ME creates one.

Syntax

```
config vsp default-session-config uui-header
config vsp policies session-policies policy name rule name session-config
uui-header
config vsp dial-plan dial-prefix entryName session-config uui-header
config vsp dial-plan route name session-config uui-header
config vsp dial-plan source-route name session-config uui-header
config vsp session-config-pool entry name uui-header
```

Properties

admin: Enables or disables this header manipulation configuration entry.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

node-id: Specifies the number to be written to the UUI header, identifying the system to the media server. This is an internal integer specific to the ME.

Default: 0
Values: Min: 1 / Max: 65535
Example: **set node-id 123**

replace-existing-header: Specifies the action the system should take if there is an existing UUI header. If **disabled**, the system appends the modified UUI header to the existing one. If **enabled**, the system replaces the existing header with the new, modified version.

Default: disabled
Values: enabled | disabled
Example: **set replace-existing-header enabled**

virtual-dialplan-settings

You can move from an existing PBX infrastructure onto the ME, while still preserving the legacy PBX dial-plan and extension number configuration.

By configuring a Virtual Dial Plan (VDP), you can virtualize the PBX onto the ME. Users that are part of the same VDP are able to reach each other via extension dialing. Any dial plan routes and digit manipulations that existed on the legacy system are applied on the ME as they would have been on the PBX.

Via the **vsp > virtual-dial-plan-pool** object, configure **virtual-dial-plans** along with the **dial-prefix**, **normalization**, **source-normalization**, **arbiter**, **route**, and **source-route** elements for each. These objects are configured the same as **vsp > dial-plan** objects. For more information on configuring VDPs, see the Dial Plan Objects.

To determine a user's VDP assignment, configure the **virtual-dialplan-settings**. Specify the assigned VDP in the **entry** property**.** You can also specify a **match-limit** in case you configure the entry points and dial plans incorrectly and a VDP loop occurs.

Syntax

```
vsp session-config-pool entry virtual-dialplan-settings
vsp default-session-config virtual-dialplan-settings
```

Properties

entry: Specify the assigned VDP for this session-config.

Default: There is no default setting
Example: **set entry vsp\tls\certificate\vdp1**

match-limit: You can also specify a **match-limit** in case you configure the entry points and dial plans incorrectly and a virtual dial plan loop occurs.

Default: 100
Values: Min: 0 / Max: 65536
Example: **set match-limit 150**

41

Configuring Session Configuration Pool Objects

The session configuration pool is a mechanism for creating a session configuration that can be referenced through other objects. This allows you to create a specific configuration and re-use it for all applicable situations. See Configuring Dial Plan Objects, for information on dial plans.

The objects available for configuration under a pool entry are the same session objects available for the default or pre-session configuration objects. See Configuring Session Configuration Objects, for a complete description of each session configuration object.

session-config-pool

Opens the session configuration pool object through which you create re-usable session configurations. You can then reference these entries in dial plan, registration plan, calling groups, and server configurations, as well as the call-control action. Note that if you do create an individual session configuration under a dial-plan entry, that local session configuration takes precedence.

Syntax

```
config vsp session-config-pool
```

Properties

None

entry

Creates an entry that can be referenced by a dial-plan or registration plan. Note that if you create an individual session configuration under a dial-plan entry, that local session configuration takes precedence. For details of the session configuration objects, see the Configuring Session Configuration Objects descriptions.

Syntax

```
config vsp session-config-pool entry string
```

Properties

None

42

Configuring Settings Objects

The **settings** object controls advanced settings for a VSP. These are properties that you typically would not need to modify.

settings

Sets a variety of VSP parameters. These are advanced settings that do not typically need modification. You can configure basic VSP parameters using the **vsp** object.

Default Plan Types For SIP Messages

Each SIP message type uses, by default, either a dial-plan or a registration-plan or no plan to determine how to handle matching messages. You can overwrite most default plan types for a given message using the sip-message-plan property. The settings that you establish with this property take precedence over the settings in the apply-to-method property of the dial- or registration-plan. Note that you cannot change the plan types for REGISTER or INVITE messages.

The following table lists the default plan types

Table 42–1 Default Plan Types

SIP message type	Plan type
INVITE	dial-plan
ACK	none
BYE	none
REGISTER	registration-plan
REFER	dial-plan
NOTIFY	registration-plan
OTHER	none
PRACK	none
CANCEL	none
SUBSCRIBE	registration-plan
OPTIONS	dial-plan
MESSAGE	dial-plan
INFO	dial-plan
PUBLISH	registration-plan

Table 42–1 (Cont.) Default Plan Types

SIP message type	Plan type
UPDATE	none
SERVICE	none

It is important to note that if you change the plan type for a message type, you must also update the applicable components of the plan through the **apply-to-methods** property. For example, NOTIFY uses a registration-plan by default. If you change it to use a dial-plan, you must modify the **apply-to-methods** property for normalization, source-normalization, arbiter, route, and/or source-route within the dial-plan to include NOTIFY as a method to which to apply the plan.

Syntax

```
config vsp settings
```

Properties

accounting-anonymous-match: Specifies whether the system should modify the From header for accounting records. When accounting is enabled, if the From header of an INVITE message matches the specified regular expression, the system saves a different header URI value as the From header value in the Call Detail Record. This property works in conjunction with the **remote-party-id-accounting** property. If that property is **enabled** and there is a match to this property, the system uses the Remote-Party-ID as the From header; if **disabled**, the system uses the Contact header. Use this property in cases where some form of call blocking causes the From header to contain an anonymous or otherwise uninformative value.

Default: anonymous@localhost.^
Example: **set accounting-anonymous-match *anonymous***

auto-server-failure-detection: Enables or disables automatic detection of server failure and failover using SIP signaling messages. If this setting is **enabled**, the system clones each SIP message for retransmission, supporting registration and call failover from one server to another. In addition, the success or failure of a transmission is dispatched to the server pool to update state information at each server. This overhead does limit call scalability. However, if this setting is **disabled**, the system cannot perform failover.

This setting is the master switch to enable this feature. You must also enable the feature for each intended server by setting the **failover-detection** property of the **server** object to **auto**, **ping**, or **register**.

Default: disabled
Values: enabled | disabled
Example: **set auto-server-failure-detection enabled**

clear-binding-on-connection-broken: Specifies whether the system deletes bindings in the location cache when a TCP or TLS connection is not in a connected state. When **enabled**, the system removes the bindings; when **disabled**, the bindings remain.

Default: enabled
Values: enabled | disabled
Example: **set clear-binding-on-connection-broken disabled**

connection-timeout: Sets the number of seconds that a SIP TLS or TCP connection can remain idle before the system closes it. Setting the value too low can cause the system to have to re-establish the connection frequently. A value of 0 disables the timeout function.

Default: 0 (disabled)
Values: Min: 0 / Max: 86400
Example: **set connection-timeout 300**

database-write: Sets whether data is written to the main database. If you disable this feature, the system writes nothing further to the database but previous records remain until they are cleaned out by the maintenance process. See the **database** object in Configuring Master Services Objects for more information on maintenance.

Default: enabled
Values: enabled | disabled
Example: **set database-write disabled**

filter-mcs-authint-to-auth: Specifies whether the system changes the Quality of Protection (QOP) value, reducing the option to only auth (authentication). This value is used between Nortel clients and servers to define the level of authentication: auth and/or authentication with integrity (authint).

When **enabled**, the system replaces the authint value in the header with the value auth. If the header contained both auth and authint, the system simply removes authint. This allows the system to rewrite the message, which is necessary for anchoring calls. When **disabled**, the header is left unchanged, but the system cannot anchor calls.

Default: enabled
Values: enabled | disabled
Example: **set filter-mcs-authint-to-auth disabled**

filter-mcs-force-IM-decrypt: Specifies whether the filter function modifies the Nortel header that specifies encryption. (Nortel encrypts all SIP payload by default.) When **enabled**, the system changes the header so that it informs the server not to re-encrypt the message when sending it to the far end. When **disabled**, the message is re-encrypted. However, the system cannot perform IM filtering on encrypted messages, so if you set it to disabled, IM filtering is disabled as a result.

Default: enabled
Values: enabled | disabled
Example: **set filter-mcs-force-IM-decrypt disabled**

filter-mcs-independent-header-schemes: Determines how the system modifies the To and From headers of the SIP message to allow compatibility with MCS (which does not currently support SIPS). As a result, the system is required to convert the “sips:” portion of header to “sip:” before delivering to MCS and then must restore the scheme for messages (of all types) returned to SIP clients.

If this property is **enabled**, the system restores the scheme part of the URI in both the To: and From: headers from the state saved in the header parameters in the message itself. If **disabled**, the system restores the scheme based on the type of tunnel connecting it to the client (TLS or non-TLS).

Default: disabled
Values: enabled | disabled
Example: **set filter-mcs-independent-header-schemes enabled**

filter-mcs-rewrite-ping-contact-hdr: Specifies whether the Nortel MCS filter in the system rewrites the PING contact header. When **enabled**, the system changes the contact header to report the client-visible public IP address instead of the client-side address of the system. This is useful if the client is behind a far-end NAT.

Default: disabled
Values: enabled | disabled
Example: **set filter-mcs-rewrite-ping-contact-hdr enabled**

filter-mcs-site-failover-threshold: Sets the amount of time the system waits for a response to a SIP message before conducting a site failover. This property is used in network configurations that use redundant sites. When the system expects some activity from a site (for example, a response to a REGISTER), it allows this interval to pass before performing a DNS lookup to determine the new, redundant site.

Default: 240 seconds (4 minutes)
Example: **set filter mcs-site-failover-threshold 300**

filter-mcs-suppress-100rel: Specifies whether the system modifies the Nortel Supported Header field (requesting acknowledgement) in the SIP request. When **enabled**, the system removes the field, causing the far-end agent not to request a reliable response. When **disabled**, the system leaves the header field untouched.

Default: enabled
Values: enabled | disabled
Example: **set filter-mcs-suppress-100rel disabled**

filter-lcs-input-remove-user-params: Enables or disables the LCS filter that strips User parameters from INVITES. By default (**enabled**), the system applies the LCS filter and strips the parameters. When **disabled**, the system leaves the user parameters in tact.

Default: enabled
Values: enabled | disabled
Example: **set filter-lcs-input-remove-user-params disabled**

filter-lcs-input-remove-record-route-hdrs: Specifies whether the system strips the Record-Route header from SIP messages. This setting is only applicable in configurations that are using messaging client tunnels (the system acts as a proxy between a messaging client and its native server). When **enabled**, the system strips the headers, when **disabled**, it leaves the message headers intact. Leave this setting at the default, enabled, for compatibility with LCS 2005.

Default: enabled
Values: enabled | disabled
Example: **set filter-lcs-input-remove-record-route-hdrs disabled**

filter-lcs-input-remove-bye-ack-params: Sets the maximum number of days to store events in the external database. When the maximum number of days is reached, the local database is cleared and is restarted at the first day.

Default: enabled
Values: enabled | disabled
Example: **set filter-lcs-input-remove-bye-ack-params disabled**

ignore-contact-on-ack: Specifies how the system updates call leg remote contact information. When **enabled**, the system does not update the call leg remote contact information from the contact header of the SIP message ACK. If **disabled**, the system uses information in the contact header of an ACK message to update the call leg remote contact field. Further, it uses that contact information when forwarding the future request message.

Default: disabled
Values: enabled | disabled
Example: **set ignore-contact-on-ack enabled**

location-cache-write-thru: Specifies when to flush modifications into the location database.

During the SIP process, location data is changed incrementally with each REGISTER and INVITE request. These incremental changes are saved in a location cache entry attached to the SIP message. If a new message has to access or modify the location data for the same AOR, the system transfers the cache copy, along with any previously

modified data, to that new message. The cache copy persists from one message to another until a cache-holder message is destroyed. At that point, the cache copy is flushed into the location database.

When **enabled**, any modification to the location data is immediately flushed into the location database. When **disabled**, the system waits until the SIP message is destroyed.

Default: disabled
Values: enabled | disabled
Example: **set location-cache-write-thru enabled**

sip-message-plan <sipMsgTypes>: Changes or assigns a plan type that the system uses for different SIP message types. These settings take precedence over the apply-to-method settings of the **dial-plan** or **registration-plan**. You can assign a plan type or no plan (**none**) to each message type. REGISTER requests always use the registration-plan and INVITEs always use the dial-plan; you cannot change these plan types. See Default Play Types for SIP Messages for a more complete explanation and important note.

Default: dial-plan
Values: none | dial-plan | registration
Example: **set sip-message-plan NOTIFY dial-plan**

request-line-routing: Determines whether message forwarding is done based on the To: header or the Request URI. If **enabled**, the system makes its forwarding decision based on the Request URI. If **disabled**, the system makes its forwarding decision based on the To: header.

Default: enabled
Values: enabled | disabled
Example: **set request-line-routing disabled**

calling-group-routing: Specifies whether to include **calling-groups** in the dial plan search criteria when determining how to forward a call. If **enabled**, the system applies the calling-group group match criteria. Incoming calls are checked to see if the IP address matches a calling group. If there is an IP match, the call is forwarded according to the routes configured under that calling group. If **disabled**, the system ignores any routes in the **calling-groups** configuration. Only the global dial-plan routes are searched when trying to route a call from a calling group.

Default: disabled
Values: enabled | disabled
Example: **set calling-group-routing enabled**

resolve-routing-through-server-domain: For Technical Support use only. Do not enable this property without explicit instructions to do so.

max-options-retransmissions: Specifies the maximum number of times the system attempts to retransmit SIP OPTIONS messages at the transaction level. By default the system only transmits OPTIONS messages once, so that it can quickly detect failure. Set the value to a higher number of attempts to allow a longer time for successful OPTIONS response. Use the session config **sip-settings > max-retransmission** property to control retransmission of other message types.

Default: 1
Values: Min: 1 / Max: 32
Example: **set max-options-retransmissions 20**

max-udp-outbound-log: Sets the total number of log entries allowed. The system creates logs when you execute the **sip > udp-log-on** action. Use this option for debugging only.

Default: 30
Values: Min: 0 / Max: 30000
Example: **set max-udp-outbound-log 150**

options-forward: Specifies whether the system, when acting as a proxy, forwards OPTIONS messages to the UAS. When **enabled**, the system forwards the messages to the provider and returns the response to the SIP client (UAC). When disabled, the system does not forward OPTIONS messages.

Default: disabled
Values: enabled | disabled
Example: **set options-forward enabled**

out-of-context-message-action <action>[resultCode][resultString]: Specifies the action to take upon receipt of a SIP message that is not affiliated with a current system session.

An out of context message is one that arrives at the system with a Call-ID different from that of any currently active session. If the arriving message is a request, and is of a method type that could start a new session (for example, INVITE, and in some cases MESSAGE, NOTIFY, SUBSCRIBE) then it is permitted. If the arriving message is a response message, or a request of a method type that can only occur within an already established session (for example, BYE, CANCEL, INFO), then it is labeled "out of context" and the prescribed action is performed.

The result code can be 400-699.

Default: discard; if you select refuse, the default result code is 400

- Values: Action options are:
- **allow**: Allow the message to be processed, and possibly forwarded, by the system SIP stack.
- **discard**: Discard out-of-context messages without a notification.
- **refuse**: Discard the packet but send a response to indicate having done so. The response includes an error code and an optional description.

Example: **set out-of-context-message-action refuse 481 "bad callID"**

out-of-context-message-media-cleanup: Specifies whether the system should tear down a media session that corresponds with an out of context message. When **enabled**, the system removes the session. See the **out-of-context-message-action** property for a description of that message type.

Default: enabled
Values: enabled | disabled
Example: **set out-of-context-message-media-cleanup disabled**

preserve-3xx-contact: Specifies whether the NAT filter service on the system should modify the contact header in a 3*xx* response message when routing an outbound call. Normally (when **disabled**), the system resets the contact header to its own local IP address. When **enabled**, the NAT filter service makes no modification to the header. Note that the preserve-3xx-contact in the **sip-settings** object controls session-based change for the SIP stack.

Default: disabled
Values: enabled | disabled
Example: **set preserve-3xx-contact enabled**

prune-associations: Specifies the event filter type and severity level for messages forwarded to the Tivoli server. Repeat the command to specify multiple event filters. See Using Filters with Event Log Messages for complete information.

Default: enabled

Values: enabled | disabled
Example: **set prune-associations disabled**

pruning-interval: Specifies the frequency with which the system attempts to reclaim inactive associations.

Default: 3600 (1 hour)
Values: Min: 1 / Max: 360000
Example: **set pruning-interval 7200**

read-header-max: Sets the maximum character length of the SIP header. This property provides buffer overflow control. If the maximum character length is exceeded, the message is discarded. Note that if the message arrived on a TCP or TLS socket (as opposed to UDP) the connection is also closed when the message is discarded.

Default: 4095
Values: Min: 64 / Max: 65535
Example: **set read-header-max 1028**

read-line-max: Sets the maximum character length for lines in the SIP message. This property provides buffer overflow control. If the maximum character length is exceeded, the entire message is discarded. If a message line has one or more continuation lines, the lengths of all these lines are added together.

Note that if the message arrived on a TCP or TLS socket (as opposed to UDP) the connection is also closed when the message is discarded.

Default: 2047
Values: Min: 64 / Max: 4095
Example: **set read-line-max 800**

read-message-max: Sets the maximum length of an entire message, including SIP header and SDP (Session Description Protocol) or other message body. This property provides buffer overflow control. If the maximum message length is exceeded, the message is discarded. Note that if the message arrived on a TCP or TLS socket (as opposed to UDP) the connection is also closed when the message is discarded.

Default: 32768
Values: Min: 64 / Max: 65535
Example: **set read-message-max 65000**

malformed-message-silent-drop: Specifies whether the system should drop malformed packets that arrive on a SIP port but do not pass SIP parsing. When **enabled**, the system drops messages identified by the kernel as malformed without sending them to the SIP process. When **disabled**, the malformed messages are sent to the SIP process, which may log a message, record the malformed message to the database for DOS pattern detection, or both, depending on other configuration settings.

Default: disabled
Values: enabled | disabled
Example: **set malformed-message-silent-drop enabled** *splittable-headers <header>*: Specifies header handling when there are multiple instances of the same header type in a message. If you specify a header type with this property and multiple instances of that type are encountered in the message, the ME splits each instance onto a separate line. Multiple instances of a header type that is not specified here results in the ME combining all instances into a single, comma separated line. You can specify as many header types as necessary by re-executing the command. Note that the following headers are always split on to multiple lines, regardless of the configuration:

- Contact
- Record-Route

- Route
- Via

The following headers are always combined, regardless of the configuration:

- Allow
- Allow-Events

Default: disabled
Example: **set splittable-headers Diversion**

server-redirect-service: Specifies whether the system saves the per-AOR redirect state of a server to its location cache. The redirect server is an alternative server listed in the Contact field of a SIP response header. If this property is **enabled**, when the system delegates a REGISTER request to a server and gets a redirect response (301/302), it performs as follows:

1. Saves the per-AOR redirect state of the server to the location cache for the AOR in concern.
2. Changes the response to a 200OK and forwards it to the destination.
3. Changes the expiration time for the register to a brief interval, causing the registering phone to reregister. When the REGISTER arrives, the system does a location cache lookup which reports the redirect state, and forwards the REGISTER to the new server.

If the system receives a call (instead of a REGISTER request) and either the To or From fields contain an AOR where the server known state is redirect, the system forwards the call to an alternative server if both of the following conditions are met:

1. The ME has previously redirected the AOR associated with the From or To fields.
2. A dial-plan lookup determines that the next-hop is a server whose state for the AOR is set to redirect.

If set to **disabled**, the system does not save the state information.

Default: disabled
Values: enabled | disabled
Example: **set server-redirect-service enabled**

server-registration-balance: Sets registration load balancing on a global level. When **enabled**, all configured servers will participate in load-balancing of REGISTER requests. Balancing is done in proportion to the maximum number of requests allowed on each server (set by the **server-pool-admission-control > max-number-of-registrations** property).

Note that once a REGISTER has been forwarded to a particular server, all future messages intended for that AOR will be forwarded to the correct server.

Note that the registration-plan **arbiter** object, if it contains a **registration-balance rule**, takes precedence over this setting.

Default: disabled
Values: enabled | disabled
Example: **set server-registration-balance enabled**

nnos-tunnel-creation: Specifies whether OC client-to-LCS server tunnels are configured via the **sip** interface or derived from the **registration-plan**. Set the property to **registration-plan** if you want to load balance across tunnels. See Configuring Load Balancing Across OC Client-to-LCX Server Tunnels for a complete description of the configuration requirements to complete tunnel load balancing.

Default: interface
Values: interface | registration-plan
Example: **set nnos-tunnel-creation registration-plan**

skip-via-transport-check: Specifies whether to ignore a mismatch in the Via header. Normally, the transport type (TCP, UDP, TLS) in the top Via header must match the transport protocol of the SIP message the system received. Use this property in cases where the client does not follow that structure. For example, the client message may have TCP in the top Via header, but actually the message was received from a TLS connection.

If set to **disabled**, when the system finds this kind of mismatch, it discards the message. If this property is **enabled**, the system does not perform the check, resulting in allowing the mismatch.

Default: disabled
Values: enabled | disabled
Example: **set skip-via-transport-check enabled**

socket-receive-buffer-size: Specifies the kernel socket buffer size for the SIP stack: for receiving SIP messages. If the system reaches the buffer size, it informs the sender so that the sender can slow transmission.

Default: 1
Values: Min: 1 / Max: 64
Example: **set socket-receive-buffer-size 2**

sockets-idle-max: Sets the maximum number of seconds that a TCP connection can remain idle before closing the connection.

Every 10 seconds, the system scans all the open TCP connections. If idle TCP connections are found, and if those connections have been idle for at least the number of seconds specified by the **sockets-idle-max** parameter, the idle TCP connections are closed. If the value is set to 0 (the default), idle sockets can remain open indefinitely, as long as their resources are not needed for a new connection.

Default: 0
Values: Min: 0 / Max: 65535
Example: **set lnp-record-directory /acme_common/lnp**

sockets-idle-min: Sets the minimum number of seconds that can transpire before a TCP connection is officially declared idle and available for a new TCP connection (socket).

If a new TCP connection is opened, and if the total current open TCP connection count exceeds the limit specified by the **sockets-per-box-max** parameter setting, the system attempts to find an open connection that has been idle for at least the number of seconds specified by the **sockets-idle-min** property. If an idle TCP connection is found, the connection is closed and available for a new TCP connection.

If there are no TCP connections that have been idle for the specified number of seconds, none are closed. If the **sockets-per-box** limit is exceeded by more than 10, new connections are refused until some of the existing sessions have closed or have been declared idle.

Default: 5
Values: Min: 1 / Max: 65535
Example: **set sockets-idle-min 10**

sockets-initial-message-timeout: Sets the number of seconds the system waits for a valid SIP message, once a TCP or TLS connection is established. If the timer expires, the system disconnects the call.

Default: 5
Values: Min: 0 / Max: 600
Example: **set sockets-initial-message-timeout 15**

sockets-per-box-max: Sets the maximum number of open TCP sockets (connections) to allow on this VSP. If the maximum number of connections is reached, the system first attempts to find idle or invalid connections to shutdown before refusing to accept new connections.

Default: 1024
Values: Min: 1 / Max: 100000
Example: **set sockets-per-box-max 1000**

sockets-per-peer-max: Sets the maximum number of open TCP sockets (connections) one remote address (a VSP peer) can have open with the system at one time. If the maximum number of connections is reached, the system first attempts to find idle or invalid connections to shutdown before refusing to accept new connections.

Note that if the per-box and per-peer maximums are the same, a peer could potentially control all connections to the box, for example, in the case of a DOS attack.

Default: 256
Values: Min: 1 / Max: 100000
Example: **set sockets-per-peer-max 1000**

stack-message-queue-max: Specifies the maximum number of messages that the system can queue in the SIP worker threads. When the processing queue length reaches the value set with this property, the system stops reading new messages from the network, and instead works on clearing out the backlog. (These messages are not deleted but saved until read.) It remains in this mode until the queue length gets down to **stack-message-queue-min**. At that point, it resumes reading new messages from the network.

Default: 8192
Values: Min: 256 / Max: 65536
Example: **set stack-message-queue-max 2048**

stack-message-queue-min: Specifies the number of messages the message queue must be reduced to before the system can begin queuing new messages. See the description for **stack-message-queue-max** for a complete description.

Default: 7168
Values: Min: 0 / Max: 65535
Example: **set stack-message-queue-min 256**

stack-message-queue-reg-clip-threshold: Specifies the maximum number of messages allowed on the SIP stack transport processing queue before the system begins discarding incoming REGISTER messages. As long as the queue length is over this number, the system continues to discard new REGISTER requests, but still queues other traffic for processing. The system reads REGISTERs from the network, but then discards them because the system is too busy to process them.

Note that if the value set for this property is higher than the value of **stack-message-queue-max**, the system will never reach the clipping threshold.

Default: 0 (disabled)
Values: Min: 0 / Max: 65536
Example: **set stack-message-queue-reg-clip-threshold 100**

supported-extensions: Specifies the name of any extensions to SIP that the system should allow for processing. Some SIP REQUEST messages contain a field indicating that the endpoint must support particular extensions. By default, the system rejects those messages. To enable passage of those messages, enter the extension name(s) in

this property. The system does not provide support for those extensions, but will recognize them and will not reject the message.

There is no default setting. The default action is for the system to reject messages containing supported extension requirements.

Default: There is no default setting
Example: **set supported-extensions ABCco-SIPvendor.extension**

translate-sips-scheme: Specifies whether to change the sips: portion of the header to sip: in the REQUEST, TO, or FROM URI and the Contact and Via headers. If enabled, the system changes the secure SIP header (sips:) to plain sip. Use this in cases where the destination server does not support secure SIP (for example, most phones do not support sips).

Default: disabled
Values: enabled | disabled
Example: **set translate-sips-scheme enabled**

tunnel-policy: Specifies whether matching policy modifies the SIP message when messaging client tunnels are configured. By default, policy does not change messages in this tunnel environment. When enabled, changes dictated by matching policy will edit the SIP message. See Configuring Messaging Client Tunnels in the **sip** object for more information.

Default: disabled
Values: enabled | disabled
Example: **set tunnel-policy enabled**

udp-ignore-content-length: Specifies whether the system ignores the content length field of the SIP header for packets coming in over UDP. The setting only applies when the content length field shows a value greater than the actual content length of the SIP message body. When set to **false**, the system does not ignore the field, and therefore discards any packet arriving with an actual content length that does not match the value in the content length field. When set to **true**, the system forwards the packet.

Set this field to **true** in a case where a SIP proxy or user agent incorrectly calculates and rewrites the content length field. This may happen, for example, when a NAT device is in use.

Default: false
Values: true | false
Example: **set udp-ignore-content-length true**

udp-tunnel-reclaim: Specifies whether the system should scan for, and tear down, inactive UDP-to-UDP tunnels. When **enabled**, the system removes tunnels that are determined inactive by expiration of the time set with the **static-stack-settings > max-udp-tunnel-inactivity** property. Use the **udp-tunnel-reclaim-scan-interval** property to set the frequency with which the system checks for idle tunnels.

Default: enabled
Values: enabled | disabled
Example: **set udp-tunnel-reclaim disabled**

udp-tunnel-reclaim-scan-interval: Sets the frequency in seconds with which the system checks for idle tunnels. The **static-stack-settings > max-udp-tunnel-inactivity** property defines the number of seconds that a tunnel can remain inactive before being deemed idle. When the udp-tunnel-reclaim property is **enabled**, the system tears down the idle tunnels found with each scan.

Default: 600
Values: Min: 60 / Max: 360000
Example: **set udp-tunnel-reclaim-scan-interval 120**

cisco-79xx-auto-ack: Specifies whether the system requires an ACK response from an INVITE challenge. Typically, when a system receives an INVITE that it must challenge, it sends a 401 challenge and awaits an ACK in response. The phone can then resend the INVITE with the appropriate Auth header information and the call can proceed.

This property is for use with Cisco phones models 7940 and 7960. These phones do not send an ACK, and instead just resend the INVITE. The system responds to the new INVITE with a 500 Server Internal Error. Set this to **enabled** if you have a Cisco 79xx model phone to signal the system to accept the new INVITE without the ACK.

Default: disabled
Values: enabled | disabled
Example: **set cisco-79xx-auto-ack enabled**

lowercase-sip-addrs: Specifies whether the system changes case for SIP addresses before storing them in the database. By default (**true**), the system changes to lowercases all addresses before storage. However, in some instances of mixed case addresses, association lookups will fail if the address has been stored as lower case. Set this property to **false** to store addresses in the database as received: either lower or mixed case.

Default: true
Values: true | false
Example: **set lowercase-sip-addrs false**

sip-process-auto-restart: Specifies the action the system should take if it detects a fatal error (e.g., deadlock) in the SIP process. If **enabled**, the system will either restart the process or, if configured with **vrrp**, failover to the backup box. If **disabled**, the process remains down and the system sends a warning message to the error log.

Default: enabled
Values: enabled | disabled
Example: **set sip-process-auto-restart disabled**

prescan-media-types: Sets whether the system prescans SIP messages for media descriptions. When **enabled**, the system preprocesses messages for CODEC-based routing. When **disabled**, it forwards messages through. You must enable this property if you have set a dial plan or policy **condition-list** to match on the sip-message-condition **media-types** attribute. Otherwise, this property should be disabled to avoid unnecessary processing overhead.

Default: disabled
Values: enabled | disabled
Example: **set prescan-media-types enabled**

backup-server: Assigns a backup server for use with the system hitless upgrade feature. There is a window of time between the beginning of the upgrade process and the point at which the system can no longer accept calls. During this time, the system redirects any calls that come in to it to the backup server specified. Therefore, a call is not started and then stranded.

Default: There is no default setting
Values: <*ipAddress*> <any | UDP | TCP | TLS>
Example: **set backup-server 10 .10.10.10.1 tls**

register-retransmit-detection: Specifies whether the system forwards request messages that were resent because the response to the original request message was dropped. When **enabled**, the system does not retransmit a client resent request message. When **disabled**, the system does resend the request.

Default: enabled
Values: enabled | disabled

Example: **set register-retransmit-detection disabled**

remote-party-id-accounting: Specifies the header value that the system should replace the From header with in the Call Detail Record in certain configurations. If the From header of an INVITE message matches the regular expression specified in the **accounting-anonymous-match** property, and this property is **enabled**, the system uses the Remote-Party-ID as the From header. If **disabled**, the system uses the Contact header as the From header.

Default: disabled
Values: enabled | disabled
Example: **set remote-party-id-accounting enabled**

apply-to-methods: Specifies the message type(s) that the system allows to be processed by an external policy server. When this message type is received by the system, if there is an external policy server configured, the system forwards a WSDL request to that server for the specific policy to apply to the session.

Default: INVITE
Example: **set apply-to-methods INVITE+REFER**

stack-socket-threads-max: *Secondary property*. Sets the number of SIP stack processing threads that should be used for TLS processing. A greater number of threads speeds up TLS connection establishment.

If you are not using TLS, set the threads to 1. If you are using TLS, set the value to 4.

Default: automatic
Values: automatic | *threads*
Example: **set stack-socket-threads-max 4**

stack-socket-event-threads-max: *Secondary property*. Sets the number of threads dedicated to servicing "RX available" events on the SIP sockets. These threads read the data from the sockets (in the case of TLS, this data is already decrypted), complete parsing of the data into complete SIP messages, and queue these messages for the worker threads to process.

In setting this value, it is best to set it lower than the value set for the **static-stack-settings > stack-worker-threads-max** property.

See Using Automatic Values for more information

Default: automatic
Values: automatic | *threads*
Example: **set stack-socket-event-threads-max 40**

sip-socket-event-queue-max: *Secondary property*. Sets the maximum number of socket events that can be queued for a single socket at any given time. Do not change this property unless instructed to do so by Technical Support.

Default: 4
Example: **set sip-socket-event-queue-max 5**

local-directory-based-user-services: *Secondary property.* Toggles whether to perform policy and user services on this VSP (whether to use policy to secure and control SIP traffic). You should only set this to **enabled** if you need to perform directory-based user services (vsp/enterprise/directory) for your SIP traffic. If enabled, you can also select which SIP messages trigger the creation of associations and user group lookups. Note that if this property is **enabled**, you must set the **directory** services for the cluster.

Also, if you enable this *without* configuring the **server > directory** property (to assign a directory to a server), you must configure the server **domain** name in order to match user SIP addresses to the appropriate server (by use of the domain).

Default: disabled
Values: enabled *msgType* disabled
Example: **set local-directory-based-user-services enabled invite+refer+register**

session-list-enable: *Secondary property.* Specifies whether or not to maintain a session list (a list of all sessions to/from a given URI). This property must be **enabled** for tone insertion to work.

If the system receives a CSTA signaling message containing DTMF digits, it replaces1 second of audio with the DTMF RTP packets (normally 50 RTP packets) in the RTP media stream. In that way, both CSTA and non-CSTA endpoints can incorporate the tones.

Default: disabled
Values: enabled disabled
Example: **set session-list-enable enabled**

sip-process-overload-restart: *Secondary property.* Restarts the SIP process if the system detects an overload of the SIP stack. This causes a crash of the SIP process and creates a crash file for debugging purposes. Do not enable this property unless instructed to do so by Technical Support.

Default: disabled
Values: enabled disabled
Example: **set sip-process-overload-restart enabled**

check-sip-sockets: *Secondary property*. Determines whether the system should check the state of all the server sockets periodically and issue a RX event, if necessary. Enable this to run a troubleshooting check if you determine that packets are not coming through. When **enabled**, the system runs a check twice a second.

Default: disabled
Values: enabled | disabled
Example: **set check-sip-sockets enabled**

timeout: *Secondary property.* Specifies the maximum amount of time an external server has to respond to a policy request. If the timer expires, the system aborts the request.

Default: 30000
Values: Min: 100 / Max: 30000
Example: **set timeout 20000**

max-queued-message: *Secondary property.* Specifies the maximum number of concurrent external policy requests allowed. When this value has been reached, the system cannot consult an external policy server to forward the SIP request. Instead, it forwards/processes requests using only the internal system policy. This condition remains until the queue drops below this limit.

Default: 0
Example: **set max-queued-message 150**

lnp-tracking: *Secondary property.* Provides a customer-specific application implementation and is not otherwise applicable.

Default: disabled
Example: **set lnp-tracking enabled**

lnp-timer: *Secondary property.* Provides a customer-specific application implementation and is otherwise not applicable.

Default: 15
lnp-removal: *Secondary property.* Provides a customer-specific application implementation and is otherwise not applicable.

Default: 1440

send-trying-before-stack: *Secondary property.* Determines at what point in processing the ME sends a "100 Trying" response to an INVITE. When this property is **enabled**, the system sends the response when low-level processing is occurring, before SIP stack processing begins. When **disabled**, the system sends a response when the SIP stack processes the INVITE.

Default: enabled
Values: enabled disabled
Example: **set send-trying-before-stack disabled**

track-sip-messages: *Secondary property.* Specifies whether the ME tracks the responses to SIP REGISTER and INVITE messages. If **enabled**, the **show sip-register-responses** and **show sip-invite-response** status providers include data indicating the type and number of responses sent and received (e.g., the number of 200 OKs, 503s, etc.).

Default: disabled
Values: enabled | disabled
Example: **set track-sip-messages enabled**

check-content-headers-method: *This is a secondary property*. Checks the incoming SIP message Content-Length and Content-Type headers to make sure they exist when the message body is not empty. This property determines which method type to do this check. By default, all the methods are subject to check. You can specify one of the following: INVITE, REFER, MESSAGE, INFO, OPTIONS, REGISTER, SUBSCRIBE, NOTIFY, PUBLISH, ACK, BYE, CANCEL, PRACK, UPDATE, SERVICE, or PING.

Default: There is no default setting
Example: **set check-content-headers-method refer+cancel**

check-content-headers-level: *This is a secondary property.* Checks the incoming SIP message Content-Length and Content-Type headers to make sure they exist when the message body is not empty. This property determines which header to check.

Default: content-type

- Values: -none: Do not check headers
- -content-type: Check Content-Type header
- Content-Type+Content-Length: Check both Content-Type and Content-Length headers.

Example: **set check-content-headers-level none**

strict-sip-parsing: *Secondary property*. When **true**, The ME performs stricter validation of parsed SIP data. Nothing is accepted that has any quirks or violates the specification in anyway.

Default: false
Values: true | false
Example: **set strict-sip-parsing true**

strict-sdp-parsing: Secondary property. When **true**, the ME performs stricter validation of parsed SDP data. Nothing is accepted that has any quirks or violates the specification in anyway.

Default: false
Values: true | false
Example: **set strict-sdp-parsing true**

match-existing-to-tag: *Secondary property*. Specifies whether the ME matches the existing To Tag of the old session leg with the To Tag of the new session leg.

Default: disabled

Values: enabled | disabled
Example: **set match-existing-to-tag enabled**

use-rfc-2543-call-id: *Secondary property*. Specifies whether the ME generates legacy RFC 2543 style call IDs using an @host argument. When enabled, this property appends the configured @host to call-IDs for b2b, 3pcc, and proxy modes.

Default: disabled
Values: enabled | disabled
Example: **set use-rfc-2543-call-id enabled localhost**

43

Configuring Session Initiation Protocol Objects

The *Session Initiation Protocol (SIP)*, described by RFC 3261, is the Internet protocol that establishes, modifies, and terminates conferencing and telephony sessions over an IP-based network using text-based messages. SIP is a major protocol in real-time collaboration networks.

You enable and configure SIP on Ethernet and VLAN interfaces. To configure load-balancing of SIP processing, see Configuring Head-End and Backing Interfaces.

Network Address Translation

Network Address Translation (NAT) takes the internal IP addresses from the private network and maps them to global public IP addresses for recipients on the public Internet. When an internal IP *address:port* (source address) is mapped to an external IP *address:port* (destination address), recipients can route traffic back to the originating IP address and port. NAT protects the private IP addresses from being exposed to clients on the public Internet.

On the ME

The ME uses NAT to ensure that SIP phone calls from internal clients on the private network can traverse enterprise firewalls en route to external clients on the public Internet. NAT operates on the two components that comprise a SIP phone call: the SIP signaling stream that sets up the phone call, and the media stream that carries RTP packets between the SIP clients. This includes:

- Re-writing IP address and TCP/UDP port information embedded in SIP/SDP messages as necessary to ensure address continuity
- Opening and closing internal media ports (“pinholes”) and controlling NAT bindings dynamically, in perfect synchronization with SIP signaling state to enable secure transit of SIP-associated media streams.

sip

Configures the Session Initiation Protocol (SIP) on an Ethernet or VLAN interface.

Syntax

```
config cluster box number interface ethX ip name sip
config cluster vrrp vinterface vxID ip name sip
config cluster box number interface ethX vlan number ip name sip
config box interface ethX ip name sip
config box interface ethX vlan number ip name sip
```

Properties

admin: Enables or disables SIP on this IP interface.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

nat-translation: Enables or disables NAT translation on this interface. See Network Address Translation for more information.

Default: disabled
Values: enabled | disabled
Example: **set nat-translation enabled**

nat-add-received-from: Sets whether the system modifies the FROM header on a NAT-translated session. If **enabled** (and far-side NAT translation is enabled), when the system transmits an INVITE, it adds a "received-from" parameter to the From: header. The property includes the public IP address on which the original REGISTER was received.

Default: disabled
Values: enabled | disabled
Example: **set nat-add-received-from enabled**

nat-add-X-Remote-Info: When enabled, if **nat-translation** is also enabled and a SIP message is received from an endpoint behind a NAT, the ME adds an X-Remote-Info header with the public IP and port of the NAT device.

Default: enabled
Values: enabled | disabled
Example: **set nat-add-X-Remote-Info disabled**

udp-port <portNumber>[fromServerReference][toServerReference]: Sets the User Datagram Protocol (UDP) port number to use when listening for SIP messages. The known UDP port number for SIP is 5060.

Optionally, you can enter a reference to a source and destination server to enable tunneling for Nortel clients. Setting the server "tells" the system that all traffic on this port is between those server types, enabling the system to filter based on that information. Use quotation marks to enter the reference. See Configuring Messaging Client Tunnels for more information.

Default: 5060
Example: **set udp-port 5060 "vsp\enterprise\servers\sip-host nortel-client" "vsp\enterprise\servers\mcs mcs-server"**

tcp-port<portNumber>[fromServerReference][toServerReference]: Sets the Transmission Control Protocol (TCP) port number to use when listening for SIP messages. The known TCP port number for SIP is 5060.

Optionally, you can enter a reference to a source and destination server to enable tunneling for Windows Messenger clients. Setting the server "tells" the system that all traffic on this port is between those server types, enabling the system to filter based on that information. Use quotation marks to enter the reference. See Configuring Messaging Client Tunnels for more information.

Default: 5060
Example: **set tcp-port 5060 "vsp\enterprise\servers\sip-host WMsgr" "vsp\enterprise\servers\lcs lcs-server"**

tls-port<portNumber>[fromServerReference][toServerReference]: Sets the TLS port number to use when listening for SIP messages. The known TLS port number for SIP is 5061.

Optionally, you can enter a reference to a source and destination server to enable tunneling for Windows Messenger or Nortel clients. Setting the server "tells" the system that all traffic on this port is between those server types, enabling the system to filter based on that information. Use quotation marks to enter the reference. See Configuring Messaging Client Tunnels for more information.

Default: 5061
Values: enabled | disabled
Example: **set tls-port 5061 "vsp\enterprise\servers sip-host WMSgr" "vsp\enteprise\servers\lcs lcs-secure"**

certificate: Assigns the certificate that must be presented to participate in SIP exchanges. Enter a reference to a previously configured certificate.

Default: There is no default setting
Example: **set certificate vsp tls certificate nnos-e.companyA.com**

load-balancing: Configures load balancing backing interfaces and distribution method.

load-balancing

Configures load balancing backing interfaces and distribution method. Note that typically load-balancing is configured on VRRP interfaces to create the redundancy. You must configure the **load-balancing** master service for load balancing to be enabled.

To load balance across tunnels, see Configuring Load Balancing Across OC Client-to-LCS Server Tunnels for complete configuration instructions.

Syntax

```
config cluster box number interface ethX ip name sip load-balancing
config cluster vrrp vinterface vxID ip name sip load-balancing
```

Properties

hash-function: Sets the hash method to use to ensure that all traffic on a connection gets forwarded to the same backing interface. The values used in the hash function are derived from the IP header on the Ethernet/IP frame.

Default: source-address-and-port

- **Values: source-address-and-port**: The source IP address and port value.
- **source-address-low-octet**: The bottom seven bits of the source port value.
- **source-address**: The source IP address value.
- **source-address-port-and-protocol**: the source IP address, port, and protocol values.
- **source-port**: The bottom seven bits of the source port value.

Example: **set hash-function source-address**

head-end-interface: Specifies a head-end interface to serve as the central distribution point of SIP traffic. The parent object of this setting becomes, by definition of the configuration, a backing interface. A backing interface can support only one head-end interface.

See Configuring Head-End and Backing Interfaces for rules on configuring the head-end interface correctly.

Default: There is no default setting

Example: **set head-end-interface “cluster vrrp vinterface vx1 ip headend1”**

44

Configuring SNMP Objects

The ME supports remote management access using the Simple Network Management Protocol (SNMP). SNMP is the Internet standard remote management protocol for network devices. Running a remote SNMP application (the SNMP manager) from a PC or workstation, you can communicate with the SNMP component on ME (the SNMP agent) to retrieve information about manageable objects on the appliance as well as edit the appliance configuration settings.

The ME imports into the enterprise MIB (Enterprise MIB objects: cxc.mib):

- SNMPv1-SMI
- SNMPv2-TC
- Standard MIB-II objects (RFC 1213-MIB)

And supports:

- GET, GETNEXT, and SET requests
- TRAP commands.

For a more detailed description of SNMP, see the *Net-Net 2600 – System Administration Guide*.

snmp

Configures the SNMP protocol on the ME.

Syntax

```
config cluster box number interface ethX ip name snmp
config cluster box number interface ethX vlan number ip name snmp
config box interface ethX ip name snmp
config box interface ethX vlan number ip name snmp
```

Properties

admin: Sets the administrative state of the SNMP protocol, either **enabled** (running) or **disabled**. When disabled, you can still configure the SNMP parameters, but the settings do not become active until the **admin** property is set to **enabled**.

Default: enabled
Values: enabled | disabled
Example: **set admin enabled**

port: Sets the UDP port over which the system listens for SNMP messages. UDP port 161 is the known port for SNMP messages; UDP port 162 is the known port for SNMP trap messages.

Default: 161
Values: Min: 1 / Max: 65535
Example: **set port 150**

version: Sets the version of SNMP to run between the remote SNMP manager and the system SNMP agent; either SNMP V1 or V2C.

Default: 2c
Values: 1 | 2c
Example: **set version 1**

community: Sets the SNMP community string that allows the remote SNMP management system to access the ME management information and statistics. The SNMP community string is similar to a user id or password that is included with all SNMP requests. If the community string is correct, the CMS Desktop responds to the SNMP request. If the community string is incorrect, the system discards the SNMP request.

Enter the SNMP community string using up to 32 alphanumeric characters with no blank spaces.

Default: public
Example: **set community private**

trap-target: Sets the IP address and optional TCP port of one or more remote hosts to which the system sends SNMP traps. You can configure up to 10 remote trap destinations.

There is no default IP address setting. The default TCP port is 162 if not specified. TCP port 162 is the known port for SNMP trap messages.

Default: There is no default setting
Example: **set trap-target 210.123.10.8**

trap-retransmit: Specifies whether the system should continue to retransmit SNMP traps until it receives an acknowledgement from the trap target. Use the **trap-reset** action to send the acknowledgement. If **enabled**, set the interval, in minutes, between retransmissions.

Default: disabled
Values: enabled *minutes* | disabled
Example: **set trap-retransmit enabled 30**

trap-filter: Specifies which categories of SNMP traps the system sends to the remote host(s) configured with the **trap-target** property. You can set as many of the pre-configured trap categories as necessary. If you do not set any trap filters, the system sends all traps. Use the **show trap-categories** command to list the possible trap types in each category.

Default: There is no default setting
Values: generic | csta | dos | sip | system | tls
Example: **set trap-filter sip**

45

Configuring Static Stack Settings Objects

The properties within the static-stack-settings object are all configuration settings that cannot be changed dynamically. Any changes to these properties do not take effect until you issue a **vsp-reset** action or restart the ME.

static-stack-settings

Modifies static properties of the VSP. Any changes to these properties do not take effect until you issue a **vsp-reset** action or restart the ME.

The ME uses the standard T1, T2, and T4 SIP timers, as described in *RFC 3261, SIP: Session Initiation Protocol*. These timers are the source for all other CXC timer settings in the SIP stack. There may be cases when you want to change the setting of these source times, for example to shorten connection times when memory is running low or to configure the ME to wait longer before initiating a retransmission.

Syntax

```
config vsp static-stack-settings
```

Properties

domain-name: Sets the DNS domain name for the VSP. The registration service only accepts registration requests matching this domain name.

Default: There is no default setting
Example: **set domain-name company.com**

domain-alias: Adds one or more aliases for the domain in which the system resides. This is useful in cases where a SIP server uses an address or other identifier instead of a domain name or where two interfaces on a CXC are configured to receive REGISTER requests. Adding an identifier: an IP address or a string: to the alias list allows the system to accept requests addressed to that identifier.

There is no limit to the number of aliases that can be added.

Default: There is no default setting
Example: **set domain-alias 198.162.10.10**

location-lookup-pattern: Specifies what portion of the URI to use when doing location cache lookups. Because vendors can change the URI format, it is difficult to maintain a consistent location cache. The system creates a URI alias table, which indexes various URI formats to an AOR.

Note that you must execute the **vsp-reset** action if you change the lookup pattern. If you do not, the alias table will be corrupt.

Default: whole-uri

- **Values: whole-uri**: The entire body of the URI. If the URI is a TEL URI, the system only considers the user portion.
- **user-host**: The user and host portion of the URI.
- **user-only**: The user portion of the URI.

Example: **set location-lookup-pattern user-host**

stack-worker-threads-max: *Secondary property*. Sets the number of SIP stack processing threads to create for this VSP. The system can support multiple execution threads, each concurrently working on a different SIP message. It is useful to configure extra threads to run, since some threads may occasionally block, for example while waiting for a response from an authentication server.

If you set the maximum to **0**, the system executes a single thread.

See Using Automatic Values for more information.

Default: automatic
Values: automatic | *threads*
Example: **set stack-worker-threads-max 60**

max-number-of-sessions: *Secondary property*. Sets the maximum number of concurrent SIP sessions that the VSP can support. This value includes all REGISTER, SUBSCRIBE, INVITE, and other sessions.the system allocates resources at boot up based on this number. (The **vsp > cac-max-number-of-calls** property creates a dynamic value for call admission control.)

Default: automatic
Values: automatic | *threads*
Example: **set max-number-of-sessions 1500**

t1: Sets the value in milliseconds of the SIP T1 timer. The T1 timer, according to RFC 3261 is: "For unreliable transports (such as UDP), the client transaction retransmits requests at an interval that starts at T1 seconds and doubles after every retransmission. T1 is an estimate of the round-trip time (RTT), and it defaults to 500 ms." The T1 full timer description can be found in section 17.1.1.1 of RFC 3261 *RFC 3261.*

Default: 2000
Values: Min: 10 / Max: 10000
Example: **set t1 30**

t2: Sets the value of the SIP T2 timer. The T2 timer, according to RFC 3261 is: "... the amount of time a non-INVITE server transaction will take to respond to a request, if it does not respond immediately." The T2 full timer description can be found in section 17.1.2.2 of *RFC 3261.*

Default: 4000
Values: Min: 1000 / Max: 10000
Example: **set t2 3000**

t4: Sets the value of the SIP T4 timer. The T4 timer, according to RFC 3261 is: "the amount of time the network will take to clear messages between client and server transactions." The T4 full timer description can be found in section 17.1.2.2 of *RFC 3261.*

Default: 5000
Values: Min: 1000 / Max: 10000
Example: **set t4 4500**

max-udp-session-linger: Sets the number of milliseconds that the system maintains the internal data structure for a SIP transaction after its apparent useful life is over.

(Keeping a session live makes it available in the event that a retransmission arrives.) You may want to reduce the session linger time if, for example, you are low on memory.

A value of 0 sets the system to remove the session immediately on receipt/transmission of the first final response.

Default: 30000
Values: Min: 0 / Max: 60000
Example: **set max-udp-session-linger 20000**

max-udp-tunnel-inactivity: Specifies the number of seconds that a UDP-to-UDP tunnel can remain inactive before it becomes eligible to be torn down. Because UDP is a connectionless protocol, it does not provide the cleanup utilities for idle sessions that a protocol such as TCP provides. Instead, the system defines inactivity with this property, and configures tunnel reclamation in the **settings** object (**udp-tunnel-reclaim** and **udp-tunnel-reclaim-scan interval** properties).

Default: 3600
Values: Min: 300 / Max: 60000
Example: **set max-udp-tunnel-inactivity 7200**

max-proxy-transactions-per-vsp: *Secondary property*. Sets the maximum number of concurrent proxy transactions for the entire VSP. You can specify an integer or use the default, **automatic**, to use the platform specific factory value.

Default: automatic
Example: **set max-proxy-transactions-per-vsp automatic**

max-redirect-sessions: *Secondary property*. When configured, this property overrides the redirect session limit enforced on the ME.

Default: automatic (The WebRTC Session Controller Media Engine uses the platform-specific factory default value)
Values: automatic | <*integer*>
Example: **set max-redirect-sessions 100**

46

Configuring STUN Server Objects

Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs) (STUN), described in RFC 3489, enables SIP clients to discover the presence and types of NATs and firewalls that exist between them and the public Internet. In addition to providing a way for an application to traverse a NAT, the protocol provides for a connectivity check between a client and server separated by a NAT. It helps prevent NAT-associated network application failures by transmitting exploratory STUN messages, over UDP port 3478, between the server and clients.

STUN allows a client to not only determine its publicly addressable IP interface and port, but to set criteria for keeping those NAT bindings open. For example, a VoIP phone or software package may include a STUN client, which will send a request to a STUN server. The server then reports back to the STUN client with:

- The public IP address of the NAT router.
- The port opened by the NAT (for that client) to allow traffic back in to the network.

ME as a STUN Relay Server

The Media Engine (ME) provides a STUN/TURN server integrated into the signaling and media proxy architecture. The TURN server provides support for straight UDP relays, as well as for TCP-to-UDP conversion. The ME supports both standard STUN (with binding discovery) and the newer version (with connectivity check usage).

As a STUN server, the ME receives STUN requests from clients and responds. (The STUN client uses DNS to find the ME/STUN server.) The ME identifies the public-side NAT details by inspecting exploratory messages from STUN-enabled clients, sent to determine which transmit and receive UDP port to use. The ME STUN server examines the incoming message and informs the client which public IP address and ports were used by the NAT. These are then used in the call establishment messages sent to the SIP destination server.

The client may also send a TURN allocate request. The ME finds an unused port on the relay-interface and sends a message back to the client with the port and IP address. The client then sends data to the ME, which forwards the data to its final destination, changing the packet to look like it originated from that port on the ME, which forwards return packets back to the client.

For complete information on STUN and TURN refer to:

- Simple Traversal Underneath Network Address Translators (NAT) (STUN)
- Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)

Understanding STUN Configuration

The following discussion only applies to RFC 3489 (standard STUN). The updated STUN protocol (RFC 3489bis, as no RFC has yet been assigned to it) does not apply to this discussion. The discussion only applies if clients in your network are using advanced features of RFC 3489 STUN.

As a STUN server, the ME listens on configured interfaces for STUN client requests. When using STUN for *Binding Discovery*, the client sends a *Binding Request* packet to the STUN server, and the server responds with a Binding Response. This response indicates the IP address and port the packet seemed to originate from (which may be different from the address/port the client sent the packet from, if there is an intervening NAT device). This address is often known as the *public address*, or *NAT mapping*. The client can then use that public IP address when registering with the ME.

Some NAT implementations base their mappings not only on the client IP address and port, but also on the server IP address and port. A packet sent from the same client address/port to a different server (or even a different port on the same server) may be given a different NAT mapping by the NAT device. Any address information learned by doing Binding Discovery with the STUN server is unusable by other devices. To determine whether or not this is the case, a STUN Binding Request can request that the STUN response be sent from a different address (presumably a different interface on the same machine), different port, or both. For the same reason, when the server sends the Binding Response, it adds information indicating which address/port it would use if the client had asked for a response from a different address/port. The client can then use this information to send a new Binding Request to the alternate address/port.

For STUN to operate properly, follow these rules when configuring STUN servers:

- Create STUN server instances in pairs
- Put each instance of the pair on a different IP address
- Assign exactly two UDP ports to each; the port number assignments must be identical for each
- The secondary interface (configured in the **stun-server** object) of each instance must point at the IP address of the other instance

For example, with a STUN server configured on interface A, ports 100 and 200, configure an additional STUN server on interface B, ports 100 and 200. In the interface A configuration, set the **secondary-interface** property to B, and vice versa.

stun-server

Configures the STUN and TURN server functionality on the ME.

Syntax

```
config cluster box number interface ethX ip name stun-server
config cluster box number interface ethX vlan number ip name stun-server
config box interface ethX ip name stun-server
config box interface ethX vlan number ip name stun-server
```

Properties

admin: Sets the administrative state of the STUN protocol, either **enabled** (running) or **disabled**.

Default: enabled

`Values: enabled | disabled`
Example: **set admin disabled**

port: Specifies the protocol and port number over which STUN messages are transmitted between a SIP client and the system STUN server.

In addition, you can specify whether to enable the TURN-redirector. When **enabled** for a port, TURN Allocate requests to that port may be redirected to another port on another box in the cluster (using the 300 Redirect error response). If no other systems in the cluster are configured to share TURN ports, or if the local machine is the best choice, the request is fulfilled locally. When the TURN-redirector is **disabled**, requests to this port are fulfilled locally or fail.

If using the TURN-redirector, you must set the **cluster > share-turn-ports** property to true to determine which systems have their TURN ports available as a target for redirection.

Default: UDP 3478 disabled
`Values: <UDP | TCP | TLS> portNumber <enabled | disabled>`
Example: **set port tcp 3478 enabled**

certificate: Specifies the certificate used to connect to the STUN server over TLS. This is the certificate that the system presents to the client.

Default: There is no default setting
Example: **set certificate vsp tls certificate my NetworkCert.pfx**

stun-auth-level: Specifies the level of authentication to require from the client for STUN requests when the transport protocol used is not TLS. (Note that this setting does not apply to TURN requests, which always use long-term authentication.)

Default: allow

- **Values: ignore**: The system does not require authentication from the client, and ignores any authentication presented. Even if the authentication is incorrect, when set to ignore, the system accepts the request.
- **allow**: The system does not require authentication from the client, but if it is presented, verifies it. If the authentication is correct or there is none, the system accepts the request. If it is incorrect, the system refuses the request.
- **short-term**: The system requires the client to use authentication that has short-term credentials. The client requests a shared secret from the system, via a TLS connection, which is given in the form of an automatically generated username and password. These credentials are used for a fixed time period of either nine minutes (when used outside the context of a TURN allocation), or for the lifetime of the allocation (when used inside the context of a TURN allocation).

Example: **set stun-auth-level short-term**

short-term-user-secret: Sets a private secret used to seed the random number generator for generated short-term credentials. This value does not need to be known by the client. If you do not configure this value, the system uses a fixed default value. If multiple systems use TURN servers cooperating in a cluster, they must all be configured with the same **short-term-user-secret** setting.

Default: There is no default setting
Example: **set short-term-user-secret pswd1**

secondary-interface: Specifies the interface to use when the CHANGE-REQUEST attribute requests a response from a different IP address. The interface you specify as the secondary interface must:

- Have a STUN server configured,

- Use the same port number assignments as the primary interface
- Have a secondary-interface that points back to this interface.

See Understanding STUN Configuration for "old-style STUN" configuration requirements.

Default: There is no default setting
Example: **set secondary-interface "cluster box 1 interface eth0 ip a"**

allow-turn: Enables or disables TURN on the system STUN server.

Default: enabled
Values: enabled | disabled
Example: **set allow-turn disabled**

relay-interface: Specifies the interface over which the client receives public visibility; the interface from which the system allocates TURN relay ports.

Create a reference to a configured interface. The interface that you select must have **media-ports** enabled and a port pool range defined, but does not require a STUN server instance configured.

Default: There is no default setting
Example: **set relay-interface "cluster box 1 interface eth1 ip z"**

allocation-lifetime-max: Specifies the maximum number of seconds that a TURN relay port allocation remains valid. Prior to expiration, the client must send a reallocation (renewal) request. The client can also send a request to immediately close the port.

Default: 3600
Values: Min: 1 / Max: 100000
Example: **set allocation-lifetime-max 750**

allocation-bandwidth-default: Specifies the amount of bandwidth allotted to a TURN relay port if the client did not request a specific amount.

Default: 150
Values: Min: 1 / Max: 1000000
Example: **set max-udp-tunnel-inactivity 7200**

ta: Sets the duration in milliseconds of the Active Destination state transition timer (as defined in draft-ietf-behave-turn-01.txt, section 8.3). This timer is used in cases when the client sets an active destination, so that all subsequent data received from the active peer is forwarded without STUN encapsulation to the client (and vice versa) and then later sets a different active destination. This property sets the amount of time, during the switch from old to new active destination, that forwarded traffic is encapsulated inside STUN Data Indications (to avoid confusion about the destination from which it came).

Leave this value set to the default unless you have pressing reason to change it.

Default: 3000
Values: Min: 0 / Max: 10000
Example: **set max-proxy-transactions-per-vsp automatic**

ltc-authentication-realm: Specifies the realm allocated for STUN LTC authentication.

Default: There is no default setting.
Example: **ltc-authentication-realm realm1**

ltc-authentication-username: Specifies the username for STUN LTC authentication.

Default: There is no default setting.
Example: **set ltc-authentication-username useradmin**

ltc-authentication-password: Specifies the password for STUN LTC authentication.

Default: There is no default setting.
Example: **set ltc-authentication-password abc123**

47

Configuring Telnet Objects

Telnet is the standard TCP/IP-based terminal emulation protocol defined in *RFC 854, Telnet Protocol Specification*, that provides a standard method for local and remote terminal communications over an IP network.

ME uses Telnet to establish a connection to the CLI. The Telnet objects allow you to configure the parameters of the Telnet session.

Basic Telnet Configuration

The following table describes the steps for connecting to the CLI over Telnet. For more detailed information about opening a Telnet session and starting the CLI, see Using the ME Command Line Interface.

1. Ensure that the ME is connected to a network that the remote system can reach.
2. Specify the Ethernet interface to be used for Telnet sessions and create a named IP configuration on that interface.
3. Start the Telnet client at the local or remote terminal.
4. Log in and start the CLI.

telnet

Configures the Telnet protocol on the ME.

Syntax

```
config cluster box number interface ethX ip name telnet
config cluster box number interface ethX vlan number ip name telnet
config box interface ethX ip name telnet
config box interface ethX vlan number ip name telnet
```

Properties

admin: Sets the administrative state of the Telnet protocol, either **enabled** (running) or **disabled**. When disabled, the parameters of Telnet can still be configured, but do not become active until **admin** is set to **enabled**.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

max-sessions: Sets the maximum number of simultaneous Telnet sessions allowed.

Default: 8

Values: Min: 1 / Max: 32
Example: **set max-sessions 10**

idle-timeout: Specifies the amount of time (in seconds) allowed to elapse before the system closes the Telnet session due to inactivity.

Default: 600
Values: Min: 60 / Max: 86400
Example: **set idle-timeout 120**

port: Identifies the Ethernet port through which the system listens for Telnet sessions.

Default: 23
Values: Min: 1 / Max: 65535
Example: **set port 21**

48

Configuring TFTP Server Objects

The ME uses a Trivial File Transfer Protocol (TFTP) server to upload and download executable images and configurations between the ME and other devices, such as IP phones. You configure TFTP servers on Ethernet and VLAN interfaces.

tftp

Opens the TFTP server configuration object on the ME and sets the administrative state and the port to which clients send and receive files. In addition, you can specify the directory to which the ME writes TFTP transfers.

Syntax

```
config cluster box number interface ethX ip name tftp
config cluster box number interface ethX vlan number ip name tftp
config box interface ethX ip name tftp
config box interface ethX vlan number ip name tftp
```

Properties

admin: Enables or disables the TFTP server on the system.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

port: Specifies the system port on which the TFTP server is listening for file transfer requests.

Default: 69
Values: Min: 1 / Max: 65535
Example: **set port 110**

directory: Specifies the default directory to which the ME writes TFTP transfers.

Default: /cxc_common/tftp
Example: **set directory /cxc_common/tftp_transfers/east**

49

Configuring Third-Party Call Control Server Objects

The ME can act as an intermediary between third-party call control (3PCC) servers (e.g., BroadWorks, Cisco Call Manager and Avaya AES) and Microsoft OCS. The ME communicates with OCS using CSTA-over-SIP. (CSTA is a protocol used by OCS to communicate call state information, which can then be reflected in user presence status.) The ME communicates with the 3PCC servers using OCI (BroadWorks) or JTAPI (Cisco and Avaya). You must configure the **jtapi** master-service to enabled third-party call control.

Third Party Call Control functionality enables Windows Communicator clients to place calls between registered SIP telephony endpoints using a 3PCC server and to place calls between any PSTN-enabled phones (wired line, mobile phones, etc.) via standard SIP gateways. In addition, subscribers can take advantage of features such as the ability to: place calls on hold; conference in third parties; transfer calls; use click-to-call dialing when calling other parties; single click to begin a conference call; accept incoming calls to the desk phone using the Communicator client; and redirect calls to any other desktop or mobile telephone.

Computer-supported telecommunications applications (CSTA) is a third-party call control protocol. It provides an abstraction layer for telecommunications applications, independent of underlying signaling protocols and devices. CSTA supplies applications services that allow a user agent to observe and control media calls (voice, IM, email, etc.). Microsoft implements SIP-based CSTA in its OCS application.

CAP is a proprietary call control interface used by Broadsoft in their SIP Application Servers to control BroadWorks calls remotely. It is an HTTPS-based protocol. When CSTA mode is set to OCI, the ME acts as a translator, allowing an OCS to view the Broadsoft BroadWorks PBX server as CSTA user agent. The LCS can then use "first-party" call control from MS Office Communicator to SIP phones in the enterprise. Communicator can originate, answer, and transfer calls (as well as execute other features) to the Broadsoft servers with the ME in between providing the conversion.

The ME supports several 3PCC server/PBX options. These include the BroadWorks, Cisco, and Avaya IP telephony platforms. The ME supports configuration to communicate with the ITC TCS server. Using this server, users can take advantage of the voice drop capabilities available through desktop environment APIs and the **call-control** action. In addition, you can configure an internal 3PCC server. An internal 3PCC server configures the ME to act as the PBX, resulting in phones registering with the ME. This mode only works with phones registered directly to with the ME. Also, you can set up a loopback configuration for testing purposes. the ME tests connectivity between itself and the OCS server.

CLI Hierarchy Information

See the following chapters for other objects in the CLI hierarchy:

- ConfiguringVirtual System Partition (VSP) Objects
- Configuring Enterprise Objects

3pcc-servers

Opens the server configuration object to allow setting the parameters for communication between the ME and the third-party call control server. The ME ports the following 3PCC servers:

- Internal CSTA server
- Broadworks CSTA server, either OCI or OCS
- Cisco Call Manager
- Avaya Communications Manager
- Loopback CSTA server for testing
- IPC CTS server

Note: While you can configure third-party call control servers at any time, you must enable the master-services jtapi object for the ME device to use the server. See Configuring Master Services Objects for more information.

Setting IPC Server Line IDs

When using the IPC CTS server with desktop APIs and the ME, you must configure a prefix and pool of available numbers used to create a unique line ID for each call. The numbers you set within this object should also be configured on the IPC server and used by it for outgoing calls to the ME. The configuration through this object allows the ME to recognize certain numbers as belonging to the IPC PBX, and can then do the appropriate mapping.

To configure the number set, first set the **number-prefix** property. Enter a string to identify the prefix, for example: 1-978-555 or tel:+555. Next you define the pool of numbers available by setting the **range-min-number** and **range-max-number**. From this pool, the IPC server will assign an extension to the prefix that is valid for the duration of the call. When the call terminates, that number is once again available. The pool can be any range or length, as long as it is configured similarly for both ends.

Syntax

```
config vsp enterprise 3pcc-servers internal-csta-server string
config vsp enterprise 3pcc-servers broadworks-csta-server string
config vsp enterprise 3pcc-servers cisco-call-manager string
config vsp enterprise 3pcc-servers avaya-communications-manager string
config vsp enterprise 3pcc-servers loopback-csta-server string
config vsp enterprise 3pcc-servers ipc-server string
```

Properties

description: Associates a text string with a server configuration. The string displays in some event logs and status providers to help identify the target.

Default: There is no default setting
Example: **set description CallMgrServer**

admin: Specifies whether the system uses this server in the current session. If **enabled**, the system uses this server. If **disabled**, the system does not use this server.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

inbound-uri-normalization: *This property is only applicable to Cisco and Avaya call managers*. Translates received Cisco or Avaya 3PCC server phone numbers into a OCS/MOC-acceptable form. The ME normalizes requests from the 3PCC server that are destined for a Microsoft OCS server using the ME. When the ME communicates with OCS/MOC, the numbers must be in an OCS-acceptable form (e.g., tel:+12126474840). Because the 3PCC server responds with and understands numbers in a different form (e.g., 6474840), the ME must perform a translation (prepending "tel:+1212") before forwarding the request.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Example: **set inbound-uri-normalization ^([0-9]{7})$ tel:+1978\1**

outbound-uri-normalization: *This property is only applicable to Cisco, BroadWorks, and Avaya call managers*. Translates phone numbers received from the Microsoft OCS server into an acceptable form for the Cisco, BroadWorks, or Avaya 3PCC server. The ME normalizes requests from the OCS server that are destined for a 3PCC server via the ME. When the ME communicates with OCS/MOC, the numbers must be in an OCS-acceptable form (e.g., tel:+12126474840). Because the 3PCC server responds with and understands numbers in a different form (e.g., 6474840), the ME must perform a translation (stripping "tel:+1212") before forwarding the request.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Example: **set outbound-uri-normalization ^([0-9]{7})$ tel:+1978\1**

server-uri-normalization: *This property is only applicable to Cisco and Avaya call managers*. Translates phone numbers coming into the ME from a 3PCC server. This normalization is used when the call manager changes a number before responding to the ME. The ME does a "first-pass" normalization of the number before applying the inbound-uri-normalization replacement and sending the call to OCS/MOC. For example, the ME may send 915085551212 when placing a call, to which the call manager may respond with 15085551212. This normalization would replace the "9" before forwarding the call.

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Default: There is no default setting
Example: **set server-uri-normalization ^9([0-9]){11}$ \1**

moc-keepalive-timeout: Specifies the number of minutes the system waits for a keepalive message from a MOC client. The client can be set to send a keepalive at a configured interval (configured at the client side). This value must be set to a value higher than the client setting. If the system does not receive a keepalive from the client within the time set with this property, it sends a message to the event log indicating a keepalive timeout. A setting of 0 (the default) disables this feature.

Default: 0
Values: Min: 0 / Max: 60
Example: **set moc-keepalive-timeout 10**

server(broadworks-csta-server only): Specifies the BroadWorks application server to which the ME connects in order to provide 3PCC services. Enter a hostname or IP address to identify the server (PBX).

Default: There is no default setting
Example: **set server 172.16.20.231**

type(broadwords-csta-server only): Specifies the type of BroadWorks PBX being used to connect calls. Also, set the port number over which the BroadWorks server is listening (the port to which the ME connects).

Default: oci 2206

- **Values: oci<*type*>**: The system acts as a translation device, converting CSTA traffic to a format the OCI server (e.g., BroadWorks PBX) can recognize. The ME sends BroadWorks CAP authentication messages to the Broadworks OCI server. The BroadWorks PBX connects the calls.
- **ocs<*type*>**: The system acts as a translation device, converting CSTA traffic to a format the OCS server (e.g., BroadWorks PBX) can recognize. The ME sends BroadWorks OCI and CAP authentication messages to the Broadworks OCS server. The BroadWorks PBX connects the calls.

Example: **set type oci 2207**

*oci-user <regEx><replacement>***(broadwords-csta-server only)**: Specifies the name that will be used to log in to the BroadWorks server. This property creates the conversion strategy to map a domain name into a BroadWorks-acceptable format.

- *regEx*: Enter a regular expression identifying the portion of the attribute to match. For example, the following expression identifies a subexpression (between the parenthesis) that matches all names:

  (.*)

- *replacement*: Enter a string that defines how to recompose the resulting regEx string. In the following example, the first component from the regular expression is substituted in place of the "1" and appended to "@company.com."

  \1@company.com

For more information regarding configuring regular expressions and replacement strings, see Using Regular Expressions.

Example: **set oci-user (.*)@company.com(.*) "\1@test.com\2"**

version (cisco-call-manager only): Specifies the version of the Cisco Call Manager that is running on the server to which the ME is connecting.

Default: cisco4.0
Values: cisco4.0 | cisco5.0 | cisco5.0
Example: **set version cisco5.0**

check-jtapi-connection (cisco-call-manager and avaya-communications-manager only): Specifies whether to verify the JTAPI connection. If set to true, when the system receives a MOC keepalive from the client, it verifies that the connection to the JTAPI server is still available. If the connection is not up, the system logs an event message.

Default: false
Values: true | false
Example: **set check-jtapi-connection true**

jtapi-debug (avaya-communications-manager only): Controls debug logging for the JTAPI library. When set to **true**, the JTAPI library creates a log file on the system and writes log messages to that file.

Default: false
Values: true | false
Example: **set jtapi-debug true**

jtapi-debug-level (avaya-communications-manager only): Specifies the message level to write to the JTAPI library log on the ME. This property is only applicable if the **jtapi-debug** property is set to **true**. Set a level, as defined below, and the system writes statements of that level to /cxc/avayaJtapiLog.OUT.

Default: 3

- **Values: 0: NONE**: No debug statements
- **1: SPY**: CstaSpy-like statements
- **2: OBJECT**: SPY + create/delete/move TS objects
- **3: HANDLER**: OBJECT + csta event handler info
- **4: EVENT**: HANDLER + jtapi event info
- **5: EXCEPTION**: EVENT + exception info
- **6: INFO**: EXCEPTION + other interesting info
- **7: ALL**: INFO + audit dump

Example: **set jtapi-debug-level 7**

host (ipc-cts-server): Specifies the ITC application server to which the ME connects in order to provide 3PCC services. Enter a hostname or IP address to identify the server (PBX).

Default: There is no default setting
Example: **set host 196.34.0.100**

port (ipc-cts-server): Specifies the port on IPC server to which the ME connects.

Default: 5712
Example: **set port 5700**

number-prefix (ipc-cts-server): Specifies the prefix that identifies calls to be terminated at the system. The prefix is completed using the **number-postfix-min** and **number-postfix-max** properties, which with the prefix, define the entire range of numbers that can be used. The prefix can take whatever form is required for the desktop application. The ME matches on the string to determine that the call is intended to be terminated.

Default: There is no default setting
Example: **set number-prefix 1-978-555**

number-postfix-min (ipc-cts-server): Specifies the starting range for numbers appended to the prefix. See Setting IPC Server Line IDs for more information.

Default: There is no default setting
Example: **set number-postfix-min 1**

number-postfix-max (ipc-cts-server): Specifies the end range for numbers appended to the prefix. See Setting IPC Server Line IDs for more information.

Default: There is no default setting
Example: **set number-postfix-max 9999**

call-delay (loopback-csta-server): Specifies the number of seconds to wait, in the event of a non-response from the call manager being loopback tested, between the origination and termination of a call.

Default: 10
Example: **set call-delay 15**

cisco-call-manager-server

Configures the JTAPI login parameters between the ME and the Cisco Call Manager server. These authentication and configuration parameters set access and connection information between the ME and the switch hosting the Cisco Call Manager. This object is only applicable to Cisco Call Manager.

Syntax

```
config vsp enterprise 3pcc-servers cisco-call-manager name
cisco-call-manager-server host
```

Properties

login: Specifies the login name that the Cisco Call manager server is configured to accept.

Default: There is no default setting
Example: **set login admin**

password: Specifies the password that the Cisco Call manager server is configured to accept.

Example: **set password admin**

connection: Specifies the number of sockets left open between the ME and the Cisco Call Manager server.

Default: 1
Values: Min: 1 / Max: 16
Example: **set connection 4**

provider-timeout: Specifies the number of minutes to wait before the system determines that the server is down or otherwise unavailable. If there is no connection or the connection fails, the system waits the amount of time specified in the **recovery-wait-time** property before testing the connection again.

Default: 1
Values: Min: 1 / Max: 60
Example: **set provider-timeout 2**

address-timeout: Specifies the number of seconds the system allows the PBX to return a verification of the address that logged in. This is the time to wait during a MOC login for the phone assigned to this user to come into service.

Default: 4
Values: Min: 1 / Max: 60
Example: **set address-timeout 10**

recovery-wait-time: Specifies the number of minutes between retries of the server connection. The system will continue retrying the connection until it is re-established or the time set with the **recovery-time-total** expires. The system begins retrying the server connection after the time set with the **provider-timeout** property has expired.

Default: 4

Values: Min: 1 / Max: 60
Example: **set recovery-wait-time 5**

recovery-time-total: Specifies the total number of minutes that the system will spend retrying the connection to the server. The system waits the number of minutes specified in **recovery-wait-time** between retries. For example, if the wait time is 5 minutes, and you want the system to retry 5 times, set this property to 25.

Default: 60
Values: Min: 5 / Max: 1440
Example: **set recovery-time-total 120**

forwarding-url: *Secondary property.* This property is a customer-specific application fix for a Cisco issue.

forwarding-cluster: *Secondary property.* This property is a customer-specific application fix for a Cisco issue.

forwarding-login: *Secondary property.* This property is a customer-specific application fix for a Cisco issue.

forwarding-password: *Secondary property.* This property is a customer-specific application fix for a Cisco issue.

aes

Configures the JTAPI login parameters between the ME and the Avaya Application Enablement Services (AES) server. These authentication and configuration parameters set access and connection information between the ME and the switch hosting the Avaya Call Manager. The Avaya implementation requires a properties file that lists addresses of the AES servers. the ME creates and maintains this file. The JTAPI implementation requires a server name to use when contacting an Avaya AES. The properties of this object configure both. This object is only applicable to Avaya Communications Manager.

Syntax

```
config vsp enterprise 3pcc-servers avaya-communications-manager name aes host
```

Properties

login: Specifies the login name that the AES server is configured to accept.

Default: There is no default setting
Example: **set login admin**

password: Specifies the password that the AES server is configured to accept.

Default: There is no default setting
Example: **set password admin**

connection: Specifies the number of sockets left open between the ME and the AES server.

Default: 1
Values: Min: 1 / Max: 16
Example: **set connection 4**

provider-timeout: Specifies the number of minutes to wait before the system determines that the server is down or otherwise unavailable. If there is no connection or the connection fails, the system waits the amount of time specified in the **recovery-wait-time** property before testing the connection again.

Default: 1
Values: Min: 1 / Max: 60
Example: **set provider-timeout**

address-timeout: Specifies the number of seconds the system allows the PBX to return a verification of the address that logged in. This is the time to wait during a MOC login for the phone assigned to this user to come into service.

Default: 4
Values: Min: 1 / Max: 60
Example: **set address-timeout 10**

recovery-wait-time: Specifies the number of minutes between retries of the server connection. The system will continue retrying the connection until it is re-established or the time set with the **recovery-time-total** expires. The system begins retrying the server connection after the time set with the **provider-timeout** property has expired.

Default: 4
Values: Min: 1 / Max: 60
Example: **set recovery-wait-time 2**

recovery-time-total: Specifies the total number of minutes that the system will spend retrying the connection to the server. The system waits the number of minutes specified in **recovery-wait-time** between retries. For example, if the wait time is 5 minutes, and you want the system to retry 5 times, set this property to 25.

Default: 60
Values: Min: 5 / Max: 1440
Example: **set recovery-time-total 120**

name: Specifies the host name of the Avaya AES server. JTAPI requires this name when responding to CSTA requests.

Default: There is no default setting
Example: **set name AV4**

backup-host-name

Configures a backup call manager for either the Cisco or Avaya platform. When the ME initiates contact, it indicates the primary PBX and if this is configured, a secondary PBX. The host names supplied here (and for the primary) must match those configured on the Cisco or Avaya device. The settings that are configured for the primary using the **cisco-call-manager-server** or **aes** objects are then applied to the backup PBX configured here.

Syntax

```
config vsp enterprise 3pcc-servers avaya-communications-manager name aes host
backup-host-name host
config vsp enterprise 3pcc-servers cisco-call-manager name
cisco-call-manager-server host backup-host-name host
```

Properties

None

50

Configuring TLS Objects

For networks running the Transport Layer Security protocol (TLS), you need to configure the certification file and the private key information required to pass SIP traffic.

TLS (sometimes referred to as Secure Socket Layer or SSL) is an encapsulation and cryptographic protocol that provides privacy and security between communicating applications over the Internet. The Media Engine (ME) uses TLS to authenticate SIP users and to encrypt/decrypt SIP traffic across participating real time collaboration networks and enterprise SIP applications.

Certificate Presentation

TLS handles presentation of certificates differently for clients (initiators) and servers (responders). Usually in a TLS connection, only the server presents a certificate: the client is only allowed to present a certificate if it is requested to do so by the server. Typically, the ME functions as a TLS server, and as such, presents a certificate to the peer. Occasionally, however, the ME functions as the client, for example, when it connects to a Microsoft LCS server. In this case, the server presents the certificate to the ME.

This operation will effect the **peer-certificate-verification** property setting of the **certificate** object. **If-presented** sets the ME to request a certificate from its peer, but allows the connection if the peer does not present. **Required** ensures that the peer presents a certificate for connection. If the certificate entry is used when the ME is a client, then **If-presented** and **Required** are equivalent. (This is because the server always presents a certificate.)

These two settings function differently when the ME is answering the connection (is the server). In the **if-presented** case, if the peer (client) doesn't present a certificate, the ME still allows the connection. If set to **required**, if the client doesn't present, the ME terminates the call. Realistically, therefore, the **if-presented** setting makes sense only for an ME.

Certificate Verification

The ME has the ability to verify a peer's certificate. By default, this behavior is disabled; all peer certificates are accepted. When enabled, however, the ME verifies a peer's certificate and rejects the connection if the certificate doesn't meet configured requirements. To verify, the ME checks the following:

- The validity of the certificate's chain. It must be signed by a trusted Certificate Authority (CA), and must not have expired.

- Clearing the Certificate Revocation List (CRL). This list tracks those certificates that a CA has revoked. If any of the certificates in the chain presented to the ME appear in the CRL, the ME rejects the connection.
- The name of the host that is presenting the certificate. If the name does not match the name the ME expects (as set in the **required-peer-name** property), then the ME rejects the certificate, even if the chain is valid.

Certificate files and CA files can be in either PEM or PKCS#12 format. CRL files must be in PEM format. For a complete description of the TLS protocol, refer to the following RFCs:

- RFC 2246, The TLS Protocol Version 1.0
- RFC 3261, Session Initiation Protocol (see Section 26.3.1)

Using Certificate Vs. Default-Outgoing-Settings

The ME uses a certificate configuration to identify the certificate file and the characteristics of the certificate. There are two types of certificate configuration: a named certificate entry that can be applied to specific TLS connections and default certificate settings for use when a specific entry was not identified.

The entry created by the **certificate** object is used when the ME functions as a server in a TLS connection. Or, it can be used in an ME-as-client setup, if you have configured the connection to use a specific certificate. For example, when you set the connection type to the LDAP server to TLS in the **directory** object, you are required to enter a named certificate.

The entry created by the **default-outgoing-settings** object is used when the ME is a client with an unspecified certificate. For example, if you were to set the protocol that the DNS resolver server uses to TLS, you are not prompted for a certificate name. The ME uses either:

- The certificate identified in the **sip-settings** object, if the session matched a configured policy
- The **default-outgoing-settings** if the session did not match a configured policy or the policy did not have a certificate specified

The **certificate** and **default-outgoing-settings** objects are otherwise the same; all certificate properties are described in the **default-outgoing-settings** object.

tls

Opens the TLS configuration object for editing. Through this object, you can configure certificate and private key information.

Syntax

```
config vsp tls
```

Properties

None

default-outgoing-settings

Creates a certificate object that sets characteristics of certificate use. This default certificate is used in cases when TLS is selected as a transport protocol and a specific certificate is not specified. See Using Certificates Vs. Default-Outgoing-Settings for more information.

The certificate for initiated connections also specifies the version of SSL/TLS to support. The ME supports SSL versions 2 and 3, and TLS version 1. You can specify any combination of these but be certain that at least one is set to **true** or the ME is not able to make connections. (By default, SSLv3 and TLSv1 are set to **true**.)

Default Vs. Specific CA and CRL Files

The ME can apply both a Certificate Authority (CA) and Certificate Revocation List (CRL) to a certificate. The CA file contains certificates that a peer can use to validate the certificate the ME presents to it during an SSL/TLS connection (see **default-ca** for more information). The CRL file is a list of certificates that a CA has revoked, and thus can no longer be trusted (see **default-crl** for more information). In each case, you have the option to configure a default file and/or a specific file to the certificate.

- **default**: A file, or set of files, that are applied to all certificates. The ME applies the files to a certificate if the **use-default-ca**/**use-default-crl** property is set to true. If set to false for a particular certificate, then that certificate does not use the default files. (You can still configure the certificate to use a CA or CRL file using the specific option, described below.)
- **specific**: A specific file that is applied either in addition to (if use-default is set to true) or instead of (if default is set to false) the default CA or CRL files for an individual certificate. You specify the specific file with the **specific-ca-file** property of this or the **certificate** objects.

Syntax

```
config vsp tls default-outgoing-settings
```

Properties

certificate-file: Specifies the name of the certificate file used to establish connections made with this object. The system supports the following certificate formats.

- **PKCS#12**: Public Key Cryptography Standard #12 format, often from Microsoft IIS Version 5 (binary)
- **PEM**: Privacy Enhanced Mail format, from any OpenSSL-based web server (ASCII)

Enter a full path name to identify the certificate location.

Default: There is no default setting
Example: **set certificate-file /cxc/certs/example.p12**

passphrase-tag: Specifies the passphrase associated with the certificate file. Use this property if the certificate file is encrypted to have its private key information protected. This passphrase must match the string that the certificate was encrypted with.

Default: There is no default setting
Example: **set passphrase-tag abc123xyz**

allow-sslv2: Specifies whether the system can negotiate Secure Socket Layer Version 2 sessions with a peer. By default, the system only supports SSLv3 or TLSv1. If you

require SSLv2 for interoperability, set this property to **true**. Note that SSLv2 is considered to suffer serious security holes.

Default: false
Values: true | false
Example: **set allow-sslv true**

allow-sslv3: Specifies whether the system can negotiate Secure Socket Layer Version 3 sessions with a peer.

Default: true
Values: true | false
Example: **set allow-sslv3 false**

allow-null-cipher: Specifies whether to use a null string in the client Hello. This setting is ignored if you have a value set in the cipher-config-string property.

Note that this property should never be enabled in a production environment, as it disables encryption. It is for debugging purposes only. If you do enable the null cipher, the client must list only the null cipher in its client Hello. Because the null cipher disables encryption, if any alternative is listed, the server will use it.

Default: disabled
Values: enabled | disabled
Example: **set allow-null-cipher enabled**

dynamic-buffers: Specifies whether to use dynamic buffers, an enhancement to the OpenSSL library. When **enabled**, the system allocates and frees transmit and receive buffers as they are needed, allowing support for many more TLS connections. When **disabled**, the system allocates both a transmit and a receive buffer at connection time, and holds the buffers open until the connection is dropped.

Default: enabled
Values: enabled | disabled
Example: **set dynamic-buffers disabled**

enable-cbc-counter-measure: Enables or disables an OpenSSL strategy the system uses when sending TLS records. The strategy is designed to prevent an attack on cipher block chaining (CBC) ciphers, which have a vulnerability in some SSL implementations. If set to **false**, you disable the protection.

Default: true
Values: true | false
Example: **set enabled -cbc-counter-measure false**

tx-record-length: Sets the record length in bytes for TLS packets. By setting the length to a value less than the default of 16,384 bytes, you reduce the amount of memory required on transmit.

Default: 2048
Values: Min: 1024 / Max: 16384
Example: **set tx-record-length 4096**

peer-certificate-verification: Specifies whether the system requests a certificate from a peer and the action in takes in response to the peer response.

Default: none

- **Values: none**: The system does not request a certificate from a peer. The system allows the connection whether or not peer presents a certificate. (If the peer does present, the system ignores the certificate.)
- **if-presented**: The system requests a certificate from the peer. If the peer presents, the certificate must pass verification for the connection to proceed. If the peer does

not present, the system allows the connection. Use this setting only when the system functions as a server. See Certificate Presentation for details.

- **required**: The system requests a certificate from the peer. If no certificate is presented, or if the presented certificate does not pass verification, the system terminates the connection.

Example: **set peer-certificate-verification required**

use-default-ca: Specifies whether to use the default revocation list(s) configured in the **default-ca** object.

Default: true
Values: true | false
Example: **set use-default-ca false**

specific-ca-file <filePath><passwordTag>: Specifies a CA file, and optionally a password, that should be used in addition to, or instead of, the default CA file(s). See Default Vs. Specific CA and CRL Files for more information.

Default: There is no default setting
Example: **set specific-ca-file cxc/certs/caZ.pem tagA**

use-default-crl: Specifies whether to use the default revocation list(s) configured in the **default-crl** object.

Default: true
Values: true | false
Example: **set use-default-crl false**

specific-crl-file <filePath><passwordTag>: Specifies a CRL file, and optionally a password, that should be used in addition to, or instead of, the default CA file(s). See Default Vs. Specific CA and CRL Files for more information.

Default: There is no default setting
Example: **set specific-crl-file cxc/certs/cr199.pem tag1**

required-peer-name: Specifies a name that must appear in the presented certificate. If you do not set this property, the system does not check the presented name.

If you do specify a name, then it must appear in either the DNS field of the altSubjectName field or in the Common Name field. To verify the peer, the system first checks to see whether there is an entry in the DNS field of the altSubjectName field. If there is, the system compares it to the required-peer-name. If it matches, the system allows the connection (and performs no further peer-name checks). If the names do not match, the system disallows the connection (and performs no further peer-name checks). If there is no entry in the DNS field, then the system checks the Common Name field. If there is a match, the system allows the connection. If the presented name does not match the required name, the system rejects the connection. You can use wildcards to express the name.

Default: There is no default setting
Example: **set required-peer-name *.companyABC.com**

cipher-config-string: *Secondary property.* Sets ciphers using the OpenSSL method. Do not change this parameter unless instructed to by Technical Support personnel.

dynamic-certificate-days-before: *Secondary Property.* Specifies how many days a dynamic certificate is marked as valid before its creation date. For example, if a certificate is created on February 25 and the value of this property is 7, the certificate is valid starting February 18.

Default: 7
Values: Min: 0 | Max: 4294967296

Example: **set dynamic-certificate-days-before 5**

dynamic-certificate-days-required: *Secondary Property.* Specifies how many days a dynamic certificate is marked as valid after its creation date. For example, if a certificate is created on February 25 and the value of this property is 7, the certificate is valid through March 4.

Default: 7
Values: Min: 0 | Max: 4294967296
Example: **set dynamic-certificate-days-required 4**

default-ca

Creates a pointer to a Certificate Authority file, which then becomes the default CA file for all certificates. This file contains certificates that a peer can use to validate the certificate the ME presents to it during an SSL/TLS connection. See Default Vs. Specific CA and CRL Files for information on using the default CA file.

CA files can be stored anywhere. However, if you store them in /cxc/certs directory, the ME copies them to all systems in the cluster. To make a CA file available for use, simply copy it on to the ME.

Syntax

```
config vsp tls default-ca
```

Properties

ca-file <path>[passwordTag]: Specifies the path to the CA file and the password (if required) to access the file. The **default-outgoing-settings** and **certificate** objects can then be configured to use or ignore these default files. You can specify multiple CA files to be used as part of the default.

Default: There is no default setting
Example: **set ca-file /cxc/certs/ca1.pem tag1**

default-crl

Creates a pointer to the one or more Certificate Revocation List (CRLs), which the ME uses for certificate validation. The CRL file is a list of certificates that a CA has revoked, and thus can no longer be trusted. If any of the certificates in the chain presented to the ME appear in the CRL, then the ME rejects the connection.

Each certificate authority maintains an up-to-date list of revoked certificates. Because there are multiple CAs, you may wish to use multiple CRLs. CRL files can be stored anywhere. However, if you store them in /cxc/certs directory, the ME copies them to all systems in the cluster. To make a CRL file available for use, simply copy it on to the ME.

See Default Vs. Specific CA and CRL Files for information on using the default CRL file.

Syntax

```
config vsp tls default-crl
```

Properties

crl-file <path>[passwordTag]: Specifies a path to a CRL file and a password for that file. The **default-outgoing-settings** and **certificate** objects can then be configured to use or ignore these default files. You can specify multiple CRL files to be used as part of the default.

Default: There is no default setting
Example: **set crl-file /cxc/certs/crl1.pem tag3**

certificate

Creates a certificate object that sets characteristics of certificate use. The object is then used in other areas of the configuration to associate a certificate with a TLS connection. See Using Certificate Vs. Default-Outgoing-Settings for more information.

Enter a certificate name to open this object.

Syntax

```
config vsp tls certificate name
```

Properties

See the **default-outgoing-settings** object.

default-dtls-settings

Sets the certificate file, requirements, and passphrase to be used for DTLS when a certificate is not specified.

Syntax

```
config vsp tls default-dtls-settings
```

Properties

allow-sslv2: (*Advanced*) Specifies whether the ME can negotiate Secure Socket Layer Version 2 sessions with a peer. By default, the ME only supports SSLv3 or TLSv1. If you require SSLv2 for interoperability, set this property to **true**.

> **Note:** For security purposes, Oracle does not recommend using SSLv2.

```
Default: false
Values: true | false
```
Example: **set allow-sslv2 true**

allow-sslv3:(*Advanced*) Specifies whether the ME can negotiate Secure Socket Layer Version 3 sessions with a peer.

Default: true
```
Values: true | false
```
Example: **set allow-sslv3 false**

allow-tlsv1: (*Advanced*) Specifies whether the ME can negotiate Transport Layer Security Version 1 sessions with a peer. (TLSv1 is the IETF-approved version of SSLv3.)

Default: true
Values: true | false
Example: **set allow-tlsv1 false**

allow-null-cipher: (*Advanced*) Specifies whether to use a null string in the client Hello. This setting is ignored if you have a value set in the **cipher-config-string** property.

> **Note:** This property should never be enabled in a production environment, as it disables encryption. It is for debugging purposes only. If you do enable the null cipher, the client must list only the null cipher in its client Hello. Because the null cipher disables encryption, if any alternative is listed, the server uses it.

Default: disabled
Values: enabled | disabled
Example: **set allow-null-cipher enabled**

dynamic-buffers: Specifies whether to use dynamic buffers, an enhancement to the OpenSSL library. When enabled, the ME allocates and frees transmit and receive buffers as needed, allowing support for many more TLS connections. When **disabled**, the ME allocates both a transmit and a receive buffer at connection time, and holds the buffers open until the connection is dropped.

Default: enabled
Values: enabled | disabled
Example: **set dynamic-buffers disabled**

enable-cbc-counter-measure: Enables or disables an OpenSSL strategy the ME uses when sending TLS records. The strategy is designed to prevent an attack on cipher block chaining (CBC) ciphers, which have a vulnerability in some SSL implementations. When **false**, the protection is disabled.

Default: true
Values: true | false
Example: **set enable-cbc-counter-measure false**

cipher-config-string: (*Advanced*) Sets ciphers using the OpenSSL method.

> **Note:** Do not change this parameter unless instructed to do so by Oracle technical support personnel.

Default: There is no default setting
Example: **set cipher-config-string SSL_TLX**

tx-record-length: Sets the record length for TLS packets. By setting the length to a value less than the default (2048 bytes), you reduce the amount of memory required on transmit.

Default: 2048
Values: Min: 1024 | Max: 16384
Example: **set tx-record-length 365**

certificate-file: (*Advanced*) Specifies the name of the certificate file used to establish connections made with this object. The ME supports the following certificate formats: PKCS#12: Public Key Cryptography Standard #12 format, often from Microsoft IIS Version 5 (binary). PEM: Privacy Enhanced Mail format, from any OpenSSL-based web server (ASCII).

Default: There is no default setting.

Example: **set certificate-file cert1**

passphrase-tag: (*Advanced*) Specifies the passphrase associated with the certificate file. Use this property if the certificate file is encrypted to have its private key information protected. This passphrase must match the string that the certificate was encrypted with.

Default:: There is no default setting.
Example: **set passphrase-tag pw123**

peer-certificate-verification: (*Advanced*) Specifies whether the ME requests a certificate from a peer and the action in takes in response to the peer's response.

Default: required-not-verified
Values:

- none: The ME does not request a certificate from a peer. The ME allows the connection whether or not peer presents a certificate. (If the peer does present, the ME ignores the certificate.)
- if-presented: The ME requests a certificate from the peer. If the peer presents, the certificate must pass verification for the connection to proceed. If the peer does not present, the ME allows the connection. Use this setting only when the ME functions as a server.
- required: The ME requests a certificate from the peer. If no certificate is presented, or if the presented certificate does not pass verification, the ME terminates the connection.
- required-not-verified: The ME requests a certificate from the peer. If no certificate is presented, the ME terminates the connection. Any certificate presented is accepted without any verification.

Example: **set peer-certificate-verification none**

use-default-ca: (*Advanced*) Specifies whether to use the default revocation list(s) configured in the default-ca object.

Default: true
Values: true | false
Example: **set use-default-ca false**

specific-ca-file: Specifies a CA file, and optionally a password, that should be used in addition to, or instead of, the default CA file(s).

Default: There is no default setting.
Example: **set specific-ca-file file456**

use-default-crl: (*Advanced*) Specifies whether to use the default revocation list(s) configured in the default-crl object.

Default: true
Values: true | false
Example: **set use-default-crl false**

specific-crl-file: Specifies a CRL file, and optionally a password, that should be used in addition to, or instead of, the default CA file(s).

Default: There is no default setting.
Example: **set specific-crl-file file_download1**

required-peer-name: (*Advanced*) Specifies a name that must appear in the presented certificate. If you do not set this property, the ME does not check the presented name.

If you do specify a name, it must appear in either the DNS field of the altSubjectName field or in the Common Name field. To verify the peer, the ME first checks to see

whether there is an entry in the DNS field of the altSubjectName field. If there is, the ME compares it to the required-peer-name. If it matches, the ME allows the connection (and performs no further peer-name checks). If the names do not match, the ME does not allow the connection (and performs no further peer-name checks). If there is no entry in the DNS field, the ME checks the Common Name field. If there is a match, the ME allows the connection. If the presented name does not match the required name, the ME rejects the connection. You can use wildcards to express the name.

Default: There is no default setting.
Example: **set required-peer-name peer1**

dynamic-certificate-country-code: Specifies the country code of the dynamically-generated certificate used for multiplexed DTLS connections.

Default: There is no default setting.
Example: **set dynamic-certificate-country-code US**

dynamic-certificate-organization-name: Specifies the organization name of the dynamically-generated certificate used for multiplexed DTLS connections.

Default: DTLS
Example: **set dynamic-certificate-organization-name DTLS**

dynamic-certificate-common-name: Specifies the common name of the dynamically-generated certificate used for multiplexed DTLS connections.

Default: dtls.invalid
Example: **set dynamic-certificate-common-name dtls.invalid**

dynamic-certificate-dns-name: Specifies the DNS name of the dynamically-generated certificate used for multiplexed DTLS connections.

Default: dtls.invalid
Example: **set dynamic-certificate-dns-name dtls.invalid**

dynamic-certificate-days-valid: Specifies the number of days the dynamically-generated certificate used for multiplexed DTLS connections is valid.

Default: 1
Values: Min: 0 | Max: 4294967296
Example: **set dynamic-certificate-days-valid 365**

dtls-cookie-exchange: When enabled, the ME forces a cookie exchange during the DTLS negotiation. This is intended to prevent DoS attacks.

> **Note:** This property must be set to **disabled** in a WebRTC environment.

Default: enabled
Values: enabled | disabled
Example: **set dtls-cookie-exchange disabled**

dynamic-certificate-days-before: *Secondary Property.* Specifies how many days a dynamic certificate is marked as valid before its creation date. For example, if a certificate is created on February 25 and the value of this property is 7, the certificate is valid starting February 18.

Default: 7
Values: Min: 0 | Max: 4294967296
Example: **set dynamic-certificate-days-before 5**

dynamic-certificate-days-required: *Secondary Property.* Specifies how many days a dynamic certificate is marked as valid after its creation date. For example, if a certificate is created on February 25 and the value of this property is 7, the certificate is valid through March 4.

Default: 7
Values: Min: 0 | Max: 4294967296
Example: **set dynamic-certificate-days-required 4**

51

Configuring User Objects

The user configuration object allows you to define in your configuration which users can pass SIP traffic on this VSP. This feature is only used if your SIP configuration requires local authentication. (Local authentication is set either in the **default-session-configuration** object under VSP, or the **session-configuration** object under policy/rule.) This object can also be used to authenticate users when **stun-server > authentication-mode** is set to **local** (used for long-term authentication).

When you enable the local authentication file, you configure ME to prompt users that are passing SIP traffic to log in. The user name and password they enter must match the entries in this file. However, you can also create policy that, for example, does not attempt to authenticate users listed in the Active Directory.

user

Configures user access to participation in SIP traffic on this VSP by adding an entry to the local authentication file. You enter each user individually, and assign a password. Use the admin property to allow or deny permission to pass SIP traffic. This file is only used if authentication is set to local in the **default-session-configuration** or **session-configuration** objects.

Enter the name of the user, up to 32 alphanumeric characters, to open the object. If the string contains delimiters (white space or \ character), it must be enclosed in double quotes (for example, “user name“). The name (and in the next level, password) that you configure are the logins needed by the user when the ME prompts.

Syntax

```
config vsp user name
```

Properties

admin: Enables or disables ability to pass SIP traffic on the VSP. You can use this property to temporarily disable a user without removing the user entry from the configuration.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

password-tag: Specifies the tag associated with the shared secret used to authenticate transactions between the system and this VSP user. See Understanding Passwords and Tags for information on the ME two-part password mechanism.

Default: There is no default setting

Example: **set password-tag abc123xyz**

52

Configuring Virtual System Partition Objects

The ME virtual system partition (VSP) is the part of the system that holds the comprehensive customer-defined configuration, which controls how the ME processes, stores, directs, and routes SIP traffic. The VSP is where you create session configurations, registration and dial plans, and policies that handle SIP request and response messages (for example, REGISTER requests and INVITE traffic). Based on the components of the VSP configuration, the ME Engine forwards traffic to SIP call destinations, authentication and accounting databases, VoIP service providers or carriers, enterprise servers, and so on.

vsp

Opens the virtual system partition (VSP) configuration object for editing. From this object you access the subobjects that set policy, session configuration characteristics, servers and directories, and many other aspects of system operation.

The properties of the VSP object include settings that limit sockets, sessions, SIP header and message sizes, and timers. You can also set the call admission control features within this object. The **settings** object contains a variety of advanced VSP parameters, which do not typically need modification.

Several properties within this object can be configured to allow the ME to determine the appropriate value (a setting of **automatic**). See Using Automatic Values for more information.

Syntax

```
config vsp
```

Properties

admin: Enables or disables this virtual system partition on this system.

Default: disabled
Values: enabled | disabled
Example: **set admin enabled**

local-identity: Sets the address-of-record for this SIP proxy. Use this property when there is more than one system (or multiple interfaces on a single system) sharing a location cache. The system writes the value of the local identifier field to the From header of a REGISTER request.

Default: There is no default setting
Example: **set local-identity sip:boston@companyABC.com**

local-normalization: Specifies whether the system changes the host portion of the SIP header to the VSP domain name. If **enabled**, when a SIP request comes in and the host portion matches the system interface address or domain alias, the system changes the Host header. If **disabled**, the Host remains as it was originally received.

Default: disabled
Values: enabled | disabled
Example: **set local-normalization enabled**

server-normalization: Specifies a first level of control that dictates whether the system changes the Host portion of the SIP header to the matching server domain name. If **enabled**, and if the matching server has the normalization routing attribute enabled, the system changes the Host header when it receives a SIP request and the Host portion matches the system interface address or domain alias. If **disabled**, the Host remains as it was originally received.

If your configuration includes many servers, you should set server-normalization to **disabled** to protect performance.

Default: disabled
Values: enabled | disabled
Example: **set server-normalization enabled**

external-inbound-normalization: Specifies whether the system should perform external normalization on inbound call legs from endpoints. If set to **no**, the system does not apply normalization settings. If set to **yes**, you must supply a tag to match against that determines the endpoints to which the system applies normalization. This name must match the table-tag column from the normalization.xml file

Default: no
Values: no | yes *tableTag*
Example: **set external-inbound-normalization yes endpnt**

registration-proxy: Specifies how the system handles registrations on this VSP. If **enabled**, for each AOR in its location cache, if there is a matching registration plan to an upstream server and the upstream server has **registration-proxy** (see below) configured and enabled, the system originates registrations to the upstream server. If **disabled**, the system does not originate a registration for AORs in its location cache.

You can also set **registration-proxy** characteristics for a server, which allows peers to receive proxy information. When set at the VSP level (this property), you are optimizing performance for your system.

Default: disabled
Values: enabled | disabled
Example: **set registration-proxy enabled**

pstn-gateway: Identifies the configured PSTN gateway for an enterprise server with a PSTN backup configuration. (This is set with the server routing setting property.) The system can allow enterprises to continue call operations even if their provider server is busy or down. You do this by configuring the public switched telephone network (PSTN) gateway.

Normally, the system forwards calls to a provider application server. If the server has failed, and the system has location information for the endpoints, it forwards calls locally. Otherwise, if configured, the system forwards calls to the PSTN gateway you configured with the sip-gateway **server** object. Enter a path to that configured server.

Default: server

- Values: none
- server *serverReference*

- carrier *carrierReference*
- gateway *gatewayReference*
- trunk *trunkGrpReference*
- trunk *trunkGrpReference*
- hunt-group *huntGrpReference*

Example: **set pstn-gateway vsp enterprise servers sip-gateway pstn**

external-policy-group: References a policy server configuration. That configuration sets the URL of the external server that maintains policy configurations to apply to a session. See **policy-service** for more information.

Default: There is no default setting
Example: **set external-policy-group "external-services policy-group myPolicy"**

external-location-group: References a location service configuration. That configuration sets the URL of the VoIP Positioning Center (VPC) providing location services (caller location) for VoIP subscribers using Location Information Services (LIS). See **red-sky-location-service**, **tcs-location-service**, or **generic-service** for more information.

Default: There is no default setting
Example: **set external-location-group "external-services location-group for E911"**

external-event-group: References an event server configuration. That configuration sets the URL of a server used for tracking system events. (These events are similar to SNMP traps.) See **event-service** for more information.

Default: There is no default setting
Example: **set external-event-group "external-services external-group myEvents"**

sip-manipulation

Configure a specific SIP manipulation. This lets you add, modify, and delete SIP headers and parts of the SIP headers called SIP header elements. SIP header elements are the different subparts of the header, such as the header value, header parameter, and URI parameter.

Syntax

```
config vsp sip-manipulation-pool sip-manipulation
```

Properties

description: Description of this SIP manipulation object.

Default: There is no default setting
Example: **set description manip10092009**

header-rules: Enter a list of header rules for this SIP manipulation.

Default: There is no default setting
Example: **set header-rule delete**

sip-manipulation-pool

Secondary object. Allows you to configure the pool of named SIP manipulation objects for the ME. These contain lists of SIP header manipulation rules and elements.

Syntax

```
config vsp sip-manipulation-pool
```

Properties

sip-manipulation: Enter the SIP manipulation you want to configure.

Default: There is no default setting
Example: **set sip-manipulation manip1**

virtual-dial-plan-pool

This object moves an existing PBX infrastructure onto an ME, creating a virtual dial plan. This preserves the legacy PBX dial-plan and extension number configuration.

Syntax

```
vsp virtual-dial-plan-pool
```

Properties

virtual-dial-plan: Configures individual virtual dial plans on the ME.

virtual-dial-plan

This object configures individual virtual dial plans on the ME.

Syntax

```
vsp virtual-dial-plan-pool virtual-dial-plan
```

Properties

name: Enter the name you want to use for this virtual dial plan.

Default: There is no default setting
Example: **set name vdp1**

dial-prefix: Applies a custom session configuration based on a dial prefix found either in the To or request URI of the SIP header.

Enter the prefix name and the prefix.

For more information on configuring dial-prefixes, see Dial Plan Objects.

Default: There is no default setting
Example: **set dial-prefix dialprefix1 555**

normalization: Facilitates routing lookup by normalizing the SIP message.

For more information on configuring dial-prefixes, see Dial Plan Objects.

Default: There is no default setting
Example: **set normalization norm1**

source-normalization: Facilitates routing lookup by normalizing the HOST portion of the SIP message. Enter the source-normalization **name**, **type**, and **source-ipnet**.

For more information on configuring dial-prefixes, see Dial Plan Objects.

Default: There is no default setting

Example: **set source-normalization snorm1 ipnet 10.10.10.10/16**

arbiter: Configures an ordered set of rules to influence the routing arbitration decision. It functions as a, "master plan," determining which metrics to use in selecting a destination server.

For more information on configuring dial-prefixes, see Dial Plan Objects.

Default: There is no default setting
Example: **set arbiter arb1**

route: Creates an entry based on the prefix and determines which part of the header to consider.

For more information on configuring dial-prefixes, see Dial Plan Objects.

Default: There is no default setting
Example: **set route route1**

source-route: Configures the ME to make call routing and forwarding decisions based on the source IP address rather than the Request URI.

Enter the source-route **name**, **type**, and **source-ipnet**.

For more information on configuring dial-prefixes, see Dial Plan Objects.

Default: There is no default setting
Example: **set source-route sroute1 15.15.15.15/16**

multimedia-streaming-config

Configure multimedia streaming when configuring the multimedia stream server (MSS) process on the ME. For more information on MSS, see the *Oracle Communications OS-E Session Services Configuration Guide*.

Syntax

config vsp multimedia-streaming-config

Properties

server: Configures a server where a multimedia stream can be published or consumed.

server

Configure multimedia streaming when configuring the multimedia stream server (MSS) process on the ME.

Syntax

```
config vsp multimedia-streaming-config server
```

Properties

name: Enter a unique name for this multimedia server.

Default: There is no default setting
Example: **set name mms1**

admin: Enables or disables this server.

Default: enabled

Values: enabled | disabled
Example: **set admin disabled**

protocol: Specifies the protocol used to communicate with this server.

Default: RTMP

- Values: RTMP: Real Time Messaging Protocol
- RTMPS: Realm Time Messaging Protocol Secure
- RTMPT: Real Time Messaging Protocol Tunneled (over HTTP)
- RTMPTS: Real Time Messaging Protocol Tunneled Secure (over HTTP)

Example: **set protocol RTMPS**

host: Specifies the hostname or IP address of the server.

Default: There is no default setting
Example: **set host 10.10.10.10**

port: Specifies the port number for the server to listen on.

Default: 1935
Values: Min: 0 / Max: 65535
Example: **set port 1945**

app-name: Specifies the path or application name where the stream can be published or consumed.

Default: live
Example: **set app-name live**

inbound-session-config-pool-entry: Specifies a session configuration entry to apply to all inbound traffic destined for this server. Enter a reference to a session configuration created in the **vsp > session-config-pool** object.

Default: There is no default setting
Example: **set inbound-session-config-pool-entry sc1**

outbound-session-config-pool-entry: Specifies a session configuration entry to apply to all outbound traffic from or through this server. Enter a reference to a session configuration created in the **vsp > session-config-pool** object.

Default: There is no default setting
Example: **set outbound-session-config-pool-entry sc1**

delegate-server: Specifies the server configured under the **vsp > enterprise > servers > sip-gateway** object.

Default: There is no default setting.
Example: **set vsp/enterprise/servers/sip-gateway server1**

route-server-config

Configures multiple route server sequences and queries on the ME. For more information on the route server, see the *Oracle Communications OS-E Session Services Configuration Guide*.

Syntax

```
config vsp route-server-config
```

Properties

route-server-sequence: Allows you to configure multiple route server queries. Provide a name for this sequence.

Default: There is no default setting.
Example: **set route-server-sequence rs_sequence1**

route-server-sequence

Configures multiple route server queries.

NOTE: When multiple route-server-sequences are configured on the ME, you can change the order of the queries using the arrows beside the route-server-sequence table.

Syntax

```
config vsp route-server-config route-server-sequence
```

Properties

description: Provide an optional description of this route server sequence.

Default: There is no default setting
Example: **set description international**

query: Configures individual route server queries. Provide a name for this query.

Default: There is no default setting
Example: **set query rs_query1**

query

Configures individual route server queries.

Syntax

```
config vsp route-server-config route-server-sequence name query
```

Properties

description: Provide an optional description of this route server query.

Default: There is no default setting
Example: **set description international**

query: Provide the following criteria for this query:

- **type**: Specify the source of the query. This can either be a value from the **header** or a **variable**. The default setting is **header**.
- **source**: Specify the source of the data to query. When the data type is **header**, the source can be either **Request**, **To**, or **From**. When the data type is **variable**, enter a string for this value. The default setting is **Request**.
- **expression**: When query **type** is **variable**, specify the regular expression to apply against the source value. The resulting value is what is queried.
- **replacement**: When query **type** is **variable**, specify the value for the ME to use in the route server query that is derived from the **expression**.

Default: header Request
Example: **set query header To**

table: Specify the table to use for this query. This is a table configured under the **route-server > table-config** object.

Default: default
Example: **set table table1**

lookup-type: Specify the type of route server lookup.

Default: route
- Values: route: Execute a route lookup.
- variable: Execute a lookup on a named variable only and do not query routes.

Example: **set lookup-type variable**

append: Specify how results of multiple queries should be appended.

Default: merge
- **Values: merge**: Routes from previous route server sequence queries are merged wiht the current query resulting routes.
- **replace**: Routes from previous route server sequence queries are replaced with the current query resulting routes.

Example: **set append replace**

variable-load: Specifies whether or not to assign route server variables to internally-named variables.

Default: none
- Values: none: No variables are expected, however, if any are returned from the route-server, the ME does not load them.
- all: Load all variables returned by the route server as-is into session-config named variables.
- assign: Assign the specified variables returned by the route server into the specified session named variables.

Example: **set variable-load all**

variable-mappings: Configures a list of mappings between route server variables returned and session-config named variables.

variable-mappings

Configures a list of mappings between route server variables returned and session-config named variables.

Syntax

```
config vsp route-server-config route-server-sequence name query name
variable-mappings
```

Properties

route-server-variable: Specify a variable name returned by the route server.

Default: There is no default setting
Example: **set route-server-variable rs_var1**

session-named-variable: Specify the session-config named variable to which you want to assign the route server variable.

Default: There is no default setting
Example: **set session-named-variable var1**

variable-ignore-additional: Specifies whether or nto to ignore any additional variables that may have been returned.

Default: false
Values: true | false
Example: **set variable-ignore-additional true**

condition-list: Defines the conditions required for the ME to execute a query.

53

Configuring Virtual Threads Objects

Sets the depth of the queues used by the ME when determining processing availability for aging the location cache.

virtual-threads

Specifies the depth of the various queues used for processing SIP messages. The ME uses these queue levels to manage purging of the location cache. See the location cache **settings > max-cache-poll-duration** property for more information.

Syntax

```
config vsp virtual-threads
```

Properties

urgent-congestion-threshold: Sets the threshold for the urgent queue. The system handles all internal signaling messages through the urgent queue.

Default: 32
Example: **set urgent-congestion-threshold 192**

priority-congestion-threshold: Sets the threshold for the priority queue. The system handles all SIP messages through the priority queue.

Default: 64
Example: **set priority-congestion-threshold 384**

regular-congestion-threshold: Sets the threshold for the regular queue. The system receives the timeout event from the location cache aging timer into this queue. Or, if aging is not complete, the event indicating the next group of messages can be aged.

Default: 128
Example: **set regular-congestion-threshold 16**

54

Configuring VLAN Objects

A virtual local area network (VLAN) is a logical grouping of systems that is not constrained by geographic boundaries. These groupings create a broadcast domain, and function just like a traditional LAN. Systems within the VLAN are not necessarily physically co-located, but do not require a router to connect them. (Routers are used to connect separate VLANs). VLANs are interconnected using system bridging software.

VLAN Tagging

The ME supports VLAN tagging, allowing a 802.1Q-compliant VLAN identifier to be added to the packet before it is sent. VLAN tagging (i.e., whether a VLAN tag is added to the packet on transmit) is enabled per VLAN, on a per interface basis. When a packet is received with a VLAN tag, the packet is accepted if the receive port is an active member of the tagged VLAN.

The ME implements its virtual firewall functionality through the use of VLAN tagging to partition private network segments. For information, see the *Oracle Communications WebRTC Session Controller System Administration Guide*.

vlan

Creates Layer 2 partitions, grouping LAN segments so that they appear to be on the same Layer 2 network. Each VLAN is identified by a VLAN ID; the ID must be unique within portions of the network, depending on the use of the VLAN.

If VRRP is not in use, a VLAN ID must be unique to each physical interface. A physical interface consists of the bottom-level interface (e.g., eth0) and all interfaces within a virtual firewall hosted on that bottom-level interface. VLAN IDs can overlap on different physical interfaces, however. For example, you could have VLAN 20 assigned on both interface eth0 and eth1.

If a cluster is configured with a VRRP interface, VLANs on the cluster must be unique among all interfaces on the cluster. (This includes the VLANs that you configure in a virtual firewall on the cluster.)

Enter a value between 2 and 4095. The ME creates a new VLAN with the specified number as an ID or opens for editing an existing VLAN.

Syntax

On a public IP interface:

```
config box interface ethX vlan integer
config cluster box integer interface ethX vlan integer
config cluster vrrp vinterface vxID vlan integer
```

Properties

admin: Enables or disables VLAN services on the specified Ethernet interface.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

arp: Enables or disables Address Resolution Protocol (ARP) services on this VLAN interface. ARP is the Internet protocol that maps IP addresses to corresponding Ethernet addresses.

Default: enabled
Values: enabled | disabled
Example: **set arp disabled**

55

Configuring VRRP Objects

The VRRP objects allow you to configure the Virtual Router Redundancy Protocol (VRRP) on ME. VRRP provides link-level failover capabilities and continued service between two or more (and up to 255) ME virtual interfaces should a network link go down in the network cluster.

Within VRRP each interface is assigned a master or backup responsibility. The priority of an interface is determined by the order in which you added the interface as a host (i.e., the first interface added has the highest priority).The VRRP interface with the highest priority is responsible for forwarding traffic and is the elected master. Other configured VRRP interfaces across the cluster with lower priorities serve as VRRP backup interfaces available to assume VRRP mastership, if necessary. You can change the order of the interfaces using the **move** command.

If the master VRRP interface becomes unavailable, the election protocol enables a backup VRRP interface to assume mastership using the next prioritized (added) interface. However, if the original master VRRP interface (the interface with the highest priority) should once again become available (and the preempt property is set to true), VRRP returns mastership to that interface.

Note that in addition to configuring VRRP for the cluster, you must also enable VRRP on one IP interface. Make sure that the parent physical interface is one that is always available, such as the management interface.

See *RFC 2338, Virtual Router Redundancy Protocol*, for detailed information about this protocol.

vrrp

Sets the administrative status of VRRP on a cluster. On the ME, VRRP functions as virtual Ethernet redundancy protocol, setting up failover for Ethernet interfaces.

In order for VRRP to work effectively on a cluster, you must enable the **share-registration-entries** property in the **cluster** object.

Syntax

```
config cluster vrrp
```

Properties

sip-fault-groups: Sets the system to initiate a VRRP failover to the backup the ME if there is a SIP crash. When you specify a group, the ME associates the SIP process with that group. (It is a good idea to apply this group to interfaces carrying SIP traffic.) In the event of a crash, the ME brings down the referenced VRRP group, causing the

failover. VRRP groups are configured using the **vinterface > group** property. A value of **0** disables this function.

Default: 0
Values: Min: 1 / Max: 32
Example: **set sip-fault-groups 5**

sip-dead-groups: Sets the system to initiate a VRRP failover to the backup ME if the local SIP process goes dead. When you specify a group, the ME associates the SIP process with that group. (It is a good idea to apply this group to interfaces carrying SIP traffic.) In the event of a crash, the ME brings down the referenced VRRP group, causing the failover. VRRP groups are configured using the **vinterface > group** property. A value of **0** disables this function.

Default: 0
Values: Min: 1 / Max: 32
Example: **set sip-dead-groups 5**

media-fault-group: *Secondary property.* Sets the system to initiate a VRRP failover to the backup box if there is a media crash. When you specify a group, the system associates the SIP process with that group. (It is a good idea to apply this group to interfaces carrying SIP traffic.) In the event of a crash, the system brings down the referenced VRRP group, causing the failover. VRRP groups are configured using the **vinterface > group** property. A value of **0** disables this function.

Default: 0
Example: **set media-fault-group 3**

vinterface

Configures a virtual interface in the cluster, for use by VRRP. A vinterface is a VRRP construction that allows a backup system for Ethernet interfaces. It does so by updating and advertising MAC-to-IP address mappings (using gratuitous ARP). A VRRP virtual router defines a MAC address for an interface. The ME loads that address onto the Ethernet interface acting as the master host. If that interface goes down, the ME moves that MAC address to the next priority interface, using ARP to broadcast the MAC-to-IP mapping.

To create or edit a vinterface, enter a virtual router interface ID in the range of **vx0** to **vx254**.

Syntax

```
config cluster vrrp vinterface vxID
```

Properties

admin: Enables and disables this VRRP virtual interface configuration. When **enabled**, the referenced interfaces participate in the failover features of VRRP. When **disabled**, the interfaces do not serve as link backup.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

group: Groups VRRP interfaces for failover purposes. If you assign a vinterface a group number of 0, it does not participate in grouping. Services that run on the box can use the VRRP grouping feature for failover as well. See Master Services in VRRP Configurations for more information.

Default: 0
Example: **set group 10**

host-interface: Sets the Ethernet interfaces to be used as part of the VRRP resources. Re-execute the command for each interface in the VRRP pool.

The order of priority is established by your order of entry (first is highest). You can configure two or more Ethernet interfaces on a box, but the boxes must be grouped. For example, all box 2 instances must occur in a row: you could configure box 2, box 2, box 1, box 3. A box must fail fully (all interfaces) before the system seeks connectivity with the next box in the configuration.

Note that each Ethernet interface should only be used once, as a master or backup. Do not re-use interfaces across a vinterface.

Default: There is no default setting
Example: **set host-interface cluster box 1 interface eth0**

preempt: *Secondary property.* Specifies whether the configured master vinterface should retake the mastership if it has gone down and then returned to operation. If set to **true**, the master resumes its position. If set to **false**, the backup interface retains master control.

Default: false
Values: true | false
Example: **set preempt true**

gratuitous-arp-count: *Secondary property.* Specifies the number of gratuitous ARP packets to send. The system uses gratuitous ARP to keep other devices informed of the IP-to-MAC address mapping of the current master vinterface.

Default: 1
Values: Min: 1 / Max: 20
Example: **set gratuitous-arp-count 3**

gratuitous-arp-interval: *Secondary property.* Specifies the number of seconds between gratuitous ARP packets if the count is set to greater than 1.

Default: 1
Values: Min: 1 / Max: 20
Example: **set gratuitous-arp-interval 10**

gratuitous-arp-period: *Secondary property.* Specifies the number of seconds the system waits between completing the previous gratuitous ARP count/interval, and beginning the next. For example, if you set the count to 2, the interval to 3, and the period to 10, the system sends out a GARP, waits 3 seconds and then sends another. After 10 seconds, the system begins again. A value of **0** sets the system to only send out the first alerting GARP packets.

Default: 0
Values: Min: 0 / Max: 65535
Example: **set gratuitous-arp-period 10**

advertisement-timer-value: *Secondary property.* Specifies how often in milliseconds the master VRRP interface advertises itself to other interfaces in the pool. Do not change this value unless instructed to do so by technical support.

Default: 100
Example: **set advertisement-timer-value 200**

heartbeat-timer-value: *Secondary property.* Sets the basis of the value used by the system to determine how long to wait before failing over to the backup interface. Do not change this value unless instructed to do so by technical support.

Default: 600
Example: **set heartbeat-timer-value 900**

takeover-skew: *Secondary property.* Sets an internal value that influences the VRRP takeover timer calculation. Do not change this value unless instructed to do so by Technical Support.

Default: 1
Values: Min: 1 / Max: 3
Example: **set takeover-skew 1**

vrrp-advertisements

Sets the interface that the ME uses to send out VRRP advertisements. Enable this object on one IP interface per box.

Syntax

```
config cluster box number interface ethX ip name vrrp-advertisements
config cluster box number interface ethX vlan number ip name vrrp-advertisements
config box interface ethX ip name vrrp-advertisements
config box interface ethX vlan number ip name vrrp-advertisements
```

Properties

admin: Sets the administrative state of VRRP advertisements on an interface. When **enabled**, the interface is used by the system to send out VRRP advertisements. When **disabled**, advertisements are not sent out over the interface. Enable this feature on only one interface per box.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

56

Configuring Web Objects

The Web object enables the Web server, providing access to the ME Management System graphical user interface. If you want to view SNMP traps through the GUI, you must also enable the server as a trap target. You enable and configure Web services on Ethernet and VLAN interfaces.

web

Configures the web server on an Ethernet or VLAN interface.

Syntax

```
config cluster box number interface ethX ip name web
config cluster box number interface ethX vlan number ip name web
config box interface ethX ip name web
config box interface ethX vlan number ip name web
```

Properties

admin: Enables or disables the system web server configuration object. Enabling a web server allows you to manage the system using the ME Management System.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

protocol: Sets the protocol to use for the ME Management System operations. After setting a protocol, you can select the web server listening port (or accept the default). This is the port the server listens on for HTTP(S) requests.

Default: https 443; the redirect port defaults to 0 which disables redirect; the default alias for the key is tomcat

- **Values: http**: Sets an insecure (unencrypted) protocol for use in web transmission.
- **https**: Provides secure transmission of web pages by using HTTP over SSL. Optionally, you can set:
- A redirect port
- A reference to a previously configured certificate (configured with the **certificate** object)
- An alias for the key in the key store (named in the certificate configuration)

Example: **set protocol https 443 0 “vsp tls certificate nnos-e.company.com” certKey**

trap-target: Configures whether the Web server collects SNMP traps. If **enabled**, the system sends traps to the Web server as well any other SNMP target. You can then

view the SNMP traps through the ME Management System **Event Logs** tab. If **disabled**, the Web does not collect SNMP traps.

Default: disabled
Values: enabled | disabled
Example: **set trap-target enabled**

jmx: Enables or disables the Java Management Extensions Managed Beans (MBeans) server. The MBean server functions as a management agent by acting as a registry for all manageable resources: applications, services, components, and devices.

When **enabled**, the system uses JMX as an interface to control resources and make them available to remote management applications. Optionally, you can specify the When **disabled**, the server does not run and you use another method as a system interface.

Default: disabled; if enabled, the default registry port is 1099 and the default server port is 1100
Values: enabled [*registryPort*][*serverPort*] | disabled
Example: **set jmx enabled 1099 1100**

max-threads: Specifies the maximum number of total worker threads, both active and spare (idle), allocated to the web server.

Default: 10
Values: Min: 1 / Max: 50
Example: **set max-threads 15**

min-spare-threads: Specifies the minimum number of inactive threads that the system must leave allocated to the web server. When the system removes idle threads, it must leave this number of spares available.

Default: 1
Values: Min: 0 / Max: 50
Example: **set min-spare-threads 20**

max-spare-threads: Specifies the maximum number of inactive threads the system can leave allocated to the web server. When the system detects idle threads, it can not maintain more than this number.

Default: 5
Values: Min: 0 / Max: 50
Example: **set max-spare-threads 8**

idle-timeout: Specifies an inactivity timeout for the ME Management System. When a session has been inactive for this number of minutes, the system logs the user off the system. A value of 0 turns off the inactivity timer.

Default: 30
Example: **set idle-timeout 45**

trusted-ips

Configures a profile of one or more IP addresses that are considered “trusted hosts” and can therefore bypass authentication. When the ME receives an HTTP request, it checks to determine whether the request came from an IP address within any trusted host profile. The ME then applies the matching permissions to the request, and bypasses the authentication (login) process. Use this to integrate selected pages into a third-party application.

Syntax

```
config cluster box number interface ethX ip name web trusted-ips name
config cluster box number interface ethX vlan number ip name web
config box interface ethX ip name web trusted-ips name
config box interface ethX vlan number ip name web trusted-ips name
```

Properties

permissions: Associates a set of **permissions** with this trusted host. Enter a reference to a previously configured set of permissions. Keep in mind that the specific access granted by the permissions profile effects the tab display in the ME Management System.

Default: There is no default setting
Example: **set permissions “access permissions EMS”**

ip-address: Sets the IP address(es) that should be allowed access to the ME Management System without further authentication. Enter the IP address of the client accessing the Web browser.

Default: There is no default setting
Example: **set ip-address 172.24.0.22**

57

Configuring Web-Service Objects

The web services object enables the Web Services Definition Language (WSDL). WSDL is an XML-based language for describing Web services, and how to access them, in a platform-independent manner. Simple Object Access Protocol SOAP (SOAP) is the communication protocol used for communication between applications, based on XML.

A WSDL document is a set of definitions that describe how to access a web service and what operations it will perform. The ME uses it in combination with SOAP and XML Schema to allow a client program connecting to a web service to determine available server functions. The actions and data types required are embedded in the WSDL file, which then may be enclosed in a SOAP envelope. The SOAP protocol supports the exchange of XML-based messages, with the ME using HTTPS.

You can configure the ME as both a WSDL client and server. Use the **external-services** object to configure it as a client; use the **web-service** object to enable the interface, allowing the ME to function as a server.

See the *Oracle Communications OS-E Management Tools* guide for a complete description of the ME WSDL implementation.

web-service

Configures the ME as a web services server by enabling WSDL on the interface. Configuring the ME as a server allows clients to make calls and requests into the box to control and manage the platform. To configure the ME as a web service client, allowing it to make web service "call outs" to get location and policy information from an external service endpoint, use the **external-services** object.

Syntax

```
config cluster box number interface ethX ip name web-service
config cluster box number interface ethX vlan number ip name web-service
config box interface ethX ip name web-service
config box interface ethX vlan number ip name web-service
```

Properties

admin: Enables or disables the use of WSDL over the selected interface. If **enabled**, the system functions as a web services server.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

protocol: Sets the protocol to use for the ME Management System operations. After setting a protocol, you can select the web service listening port (or accept the default). This is the port the server listens on for HTTP(S) requests.

Default: https 443; the redirect port defaults to 0 which disables redirect; the default alias for the key is tomcat

- **Values: http**: Sets an insecure (unencrypted) protocol for use in web transmission.
- **https**: Provides secure transmission of web pages by using HTTP over SSL. Optionally, you can set:
- A redirect port
- A reference to a previously configured certificate (configured with the **certificate** object)
- An alias for the key in the key store (named in the certificate configuration)

Example: **set protocol https 443 0 "vsp tls certificate nnos-e.company.com" certKey**

authentication: Specifies whether the web service client needs a certificate to communicate with the system. Set the property to **basic** to require HTTP basic authentication for client connections. Set the property to **certificate** to require an HTTPS certificate for authentication of client connections.

Default: basic
Values: basic | certificate *integer*
Example: **set authentication certificate "vsp tls certificate ws_cert"**

application: Identifies a **click-to-call** profile that the system web service server will host. Enter a reference to a previously configured application.

Default: There is no default setting
Example: **set application preferences click-to-call**

max-threads: Specifies the maximum number of total worker threads, both active and spare (idle), allocated to the web service.

Default: 10
Values: Min: 1 / Max: 50
Example: **set max-threads 15**

min-spare-threads: Specifies the minimum number of inactive threads that the system must leave allocated to the web service. When the system removes idle threads, it must leave this number of spares available.

Default: 1
Values: Min: 0 / Max: 50
Example: **set min-spare-threads 20**

max-spare-threads: Specifies the maximum number of inactive threads the system can leave allocated to the web service. When the system detects idle threads, it can not maintain more than this number.

Default: 5
Values: Min: 0 / Max: 50
Example: **set max-spare-threads 8**

max-message-process-threads: The maximum number of messaging processing threads.

Default: 10
Values: Min: 10 / Max: 200
Example: **set max-message-process-threads 50**

max-http-connections: The maximum number of outbound HTTP connections.

Default: 100
Values: Min: 100 / Max: 300
Example: **set max-http-connections 250**

max-http-client-connections: The maximum number of outbound HTTP connections per host.

Default: 10
Values: Min: 5 / Max: 100
Example: **set max-http-client-connections 75**

virtual-host

The ME supports an embedded Tomcat web server which allows users to host some simple Java-based web applications directly on the ME.

The ME is able to do this by means of virtual hosting. A virtual host is simply an alternate DNS name for an IP address. The ME uses a Tomcat server to process requests on a per-domain basis. Based on the configuration, you can select which virtual hosts the ME responds to.

> **Note:** Because HTTP DNS-based virtual hosting requires distinct names, two names are needed for the virtual host to work properly. You need one name to map to the ME web services server and the other to map to the applications virtual host.

For example, "ose.example.com" and "oseapps.example.com both point to the same IP. Applications are available on the virtual host oseapps.example.com name, while the applications are configured to use the ME web services available on the asc.example.com name.

For the embedded web server to work, you must specify a directory in the virtual host configuration on which to copy Web Application Resource (WAR) files you want to deploy.

> **Note:** The directory onto which you copy WAR files is always under the cxc_common directory and cannot be changed.

Syntax

```
config cluster box name interface eth ip address web-service virtual-host name
```

Properties

name: Specifies the host name for this virtual host.

Default: There is no default setting
Example: **set name vh1**

admin: Enables or disables this virtual host.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

applications-directory: The directory you are placing WAR files for applications presented by this virtual host.

Default: webapps
Example: **set applications /apps_directory**

web-app-config: Configures the web application running on this virtual host. Enter a path.

Default: There is no default setting
Example: **set web-app-config /web_app_directory**

role-mapping: Configures authentication and authorization for all web applications based on the **access permissions** configuration. Enter a name for the role to assign to this permission.

For information about roles, see http://docs.oracle.com/javee/6/tutorial/doc/gijrp.html.

Default: There is no default setting
Example: **set role-mapping admin**

access-logging: Configure the access logging for this virtual host. This is a log file to record all requests and responses to and from this virtual host.

web-app-config

Configures the web application running on this virtual host.

Syntax

```
config cluster box name interface eth ip address web-service virtual-host name
web-app-config
```

Properties

context-parameter: Sets an application-level configuration property for this web application. Enter a name and a value.

Default: There is no default setting
Example: **set context-parameter param1 phone_app**

servlets: Configures servlets within this web application. Enter a name.

For information about servlets, see http://docs.oracle.com/javaee/6/tutorial/doc/bnafd.html.

Default: There is no default setting
Example: **set servlets serv1**

servlets

Configures servlets within this web application.

Syntax

```
config cluster box name interface eth ip address web-service virtual-host name
web-app-config
```

Properties

init-parameter: Sets the servlet configuration that overrides the configuration from the web application as it is deployed. Enter a name and a value.

Default: There is no default setting
Example: **set init-parameter init1 param2**

access-logging

Configures the access logging for this virtual host. This is a log file to record all requests and responses to and from this virtual host.

Syntax

```
config cluster box name interface eth ip address web-service virtual-host name
access-logging
```

Properties

admin: Enables or disables access logging on this virtual host.

Default: enabled
Values: enabled | disabled
Example: **set admin disabled**

directory: Specifies the direcdtory on which to write the log files.

Default: /cxc_common/virtualhosts/logs
Example: **set directory /webapps_log**

prefix: Enter a prefix of the file name for the logging files.

Default: There is no default setting
Example: **set prefix wa**

pattern: Set the format to use to log requests.

Default: combined
- Values: common: A basic common format.
- combined: A combined format that extends the common format.

Example: **set pattern common**

buffered: Indicates whether log messages should be buffered before writing.

Default: false
Values: true | false
Example: **set buffered true**

maximum-file-age: Enter the maximum number of days to keep the log in the directory before deleting.

Default: 7
Values: Min: 0 / Max: 4294967296
Example: **set maximum-file-age 15**