

Oracle® Communications Unified Session Manager

Essentials Guide

Release S-CZ7.1.5 M1

Formerly Net-Net SIP Multimedia Xpress

December 2014

Notices

Copyright ©2014, 2013, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

1 Oracle USM Supporting the IMS Core.....	13
General Description.....	13
Message Authentication for SIP Requests.....	13
User Authorization.....	13
UAR/UAA Transaction.....	14
SIP Digest User Authentication.....	14
Authentication via MAR/MAA.....	14
SIP Authentication Challenge.....	14
SIP Authentication Response.....	15
Oracle USM Authentication Check.....	15
IMS-AKA Support.....	16
Authentication Sequence - Registration.....	16
Outside the Core.....	17
Authentication Success.....	17
Authentication Failure.....	18
Synchronization.....	18
Optional IMS-AKA Configuration.....	18
Oracle USM as Registrar.....	19
New Registration.....	19
Limiting AOR Contacts.....	19
HSS Server Assignment.....	20
Server Assignment Messages.....	20
Register Refresh.....	20
Entry Unregistration.....	21
User Registration based on Reg-ID and Instance-ID (RFC 5626).....	22
reREGISTER Example.....	22
Outbound Registration Binding Processing.....	22
Wildcarded PUID Support.....	23
Oracle USM Licensing.....	23
ACLI Instructions.....	23
home subscriber server.....	23
SIP Authentication Profile.....	24
SIP Interface.....	24
SIP Registrar.....	25
Maximum Number of Contacts.....	25
Response to Exceeding Maximum Contacts.....	25
SIP Registration Event Package Support.....	26
SUBSCRIBE Processing.....	27
SUBSCRIBE REFRESH Requests.....	27
Reg Event NOTIFY Messages.....	27
Reducing NOTIFY Traffic.....	28
Configuring Registration Event Package.....	29
Registration Event Profile Configuration.....	29
Message Routing.....	29
Registrar Routing.....	30
Default Egress Realm.....	30
RX Interface Features.....	31
Routing Based on UA Capabilities.....	31
ACLI Instructions.....	33

Supporting Media Sessions Established via ICE.....	33
Configuring Support for ICE.....	34
Tel-URI Resolution.....	35
Number Lookup Triggers.....	35
Actions Based on Lookup Results.....	36
Primary and Secondary ENUM Configuration.....	36
HSS Initiated User Profile Changes.....	37
Licensing and Database Registration Limits.....	37
Database Registration Limit Alarm.....	38
3GPP Compliance.....	38
P-Asserted-Id in Requests and Dialogs.....	38
P-Associated-URI in 200 OK.....	38
Other Diameter Cx Configuration.....	39
Host and Realm AVP Configuration for Cx.....	39
ACLI Instructions.....	39
Initial Filter Criteria (IFC).....	39
IFC Evaluation.....	39
SIP Registration.....	40
SIP Call.....	40
Evaluating Session Case in the P-Served-User Header.....	40
Supported Sessioncase and Registration State.....	41
Additional Options.....	41
IFC Support for Unregistered Users.....	42
UE-terminating requests to an unregistered user.....	42
Caching the Downloaded IFC.....	43
Optimizing IFC Updates.....	43
Push Profile Request (PPR) updates.....	43
ACLI Instructions.....	43
SIP Registrar.....	43
SIP Registrar.....	44
Shared and Default iFCs.....	44
SiFC Usage.....	45
DiFC Usage.....	45
SiFC/DiFC File Example.....	45
iFC Execution Order.....	46
Refreshing SiFC and DiFC Files.....	46
SiFC and DiFC Configuration.....	46
Distinct and Wildcarded Public Service Identity (PSI) Support.....	47
Configuring SIP Ping OPTIONS Support.....	47
Redundancy and Load Balancing with HSS Servers.....	48
About HSS Groups.....	48
Connection Failure Detection.....	49
Stream Control Transfer Protocol Overview.....	50
SCTP Packets.....	50
SCTP Terminology.....	50
SCTP Message Flow.....	51
Congestion Control.....	52
Multi-Streaming.....	52
Delivery Modes.....	53
Multi-Homing.....	53
Multi-Homing and Path Diversity.....	54
Monitoring Failure Detection and Recovery.....	54
ACLI Instructions for Configuring SCTP for DIAMETER Transport.....	55
Configuring an HSS Server for SCTP.....	55
Configuring the Realm.....	56
Setting SCTP Timers and Counters.....	56

2 ENUM Based Oracle USM..... 63

Message Authentication for SIP Requests.....	63
Credential Retrieval.....	63
User Authentication Query.....	64
SIP Digest User Authentication.....	64
SIP Authentication Challenge.....	64
Authentication Header Elements.....	64
SIP Authentication Response.....	64
Oracle USM Authentication Check.....	64
Oracle USM as Registrar.....	65
DDNS Update to User Subscriber Database.....	65
ENUM Database Correlation.....	65
Register Refresh.....	66
Limiting AOR Contacts.....	67
User Registration based on Reg-ID and Instance-ID (RFC 5626).....	68
reREGISTER Example.....	68
Outbound Registration Binding Processing.....	68
ENUM Database Update.....	68
NAPTR Update Format.....	69
Oracle USM Licensing.....	69
ACLI Instructions.....	69
ENUM Configuration.....	69
SIP Authentication Profile.....	70
SIP Registrar.....	70
Maximum Number of Contacts.....	71
Response to Exceeding Maximum Contacts.....	71
Update to ENUM Database on Endpoint Connection Loss.....	71
Connection Reuse.....	72
Unreachability Determination.....	72
Registration Cache and User Database Removal.....	73
ACLI Instructions.....	73
OAuth 2.0 Support.....	75
OAuth Operation.....	75
Configuring OAuth Support.....	76
Enabling the SPL Plug-in.....	76
Configuring the Plug-in Option.....	77
Message Routing.....	78
Registrar Routing.....	78
Default Egress Realm.....	78
Segmentation of ENUM Zones.....	79
Configuring Support for DDNS Server Caching.....	80
Tel-URI Resolution.....	81
Number Lookup Triggers.....	81
Actions Based on Lookup Results.....	81
Primary and Secondary ENUM Configs.....	82
Licensing and Database Registration Limits.....	83
Database Registration Limit Alarm.....	83
Extended ENUM Record Length.....	83
NAPTR and TXT Record Creation and Association.....	83
NAPTR Record Format.....	84
TXT Record Retrieval.....	84
Requirements.....	84
SIP User Parts - RFC 3261 Character Set Support.....	84
Encoding Alpha-Numerics.....	84

Multiple DNS Zone Support.....	85
Alpha-Numeric Name Support.....	85
Configuring SIP Ping OPTIONS Support.....	85

3 Local Subscriber Tables..... 87

Local Subscriber Table.....	87
LST Runtime Execution.....	87
LST Configuration.....	87
ACLI Instructions.....	88
LST Table.....	88
SIP authentication profile.....	88
LST Redundancy for HA Systems.....	88
Reloading the LST.....	89
LST File Compression.....	89
LST File Format.....	89
LST Subscriber Hash and Encryption.....	89

4 Third Party Registration..... 91

Third Party Registrations via iFCs.....	92
Embedded REGISTER.....	92
ACLI Instructions - Third Party Registration via iFCs.....	92
Session Agent.....	92
SIP Registrar.....	93
Third Party Registration via ACLI Configuration.....	93
Third Party Registration Server States.....	94
Third Party Registration Expiration.....	94
Defining Third Party Servers.....	95
ACLI Instructions - Third Party Server Configuration.....	95
Third Party Registrar.....	95
SIP Registrar.....	95

5 RADIUS Accounting of REGISTERS..... 97

CDR Generation for REGISTER Events.....	97
REGISTER Scenarios.....	97
REGISTER VSA Format.....	100
CDR Generation Configuration.....	101
Example CDRs.....	101
Local CDR CSV Orientation.....	104
Start Record.....	104
Interim Record.....	108
Stop Record.....	114

6 References and Debugging..... 121

ACLI Configuration Elements.....	121
sip-registrar.....	121
Parameters.....	121
Path.....	122
sip-authentication-profile.....	122
Parameters.....	122
Path.....	123
home-subscriber-server.....	123
Parameters.....	123
Path.....	124

third-party-regs.....	124
Parameters.....	124
Path.....	124
local-subscriber-table.....	124
Parameters.....	124
Path.....	125
enum-config.....	125
Parameters.....	125
Path.....	126
ifc-profile.....	126
Parameters.....	126
Path.....	126
regevent-notification-profile.....	126
Parameters.....	126
Path.....	127
hss-group.....	127
Parameters.....	127
SNMP MIBs and Traps.....	127
Acme Packet License MIB (ap-license.mib).....	127
Acme Packet System Management MIB (ap-smgmt.mib).....	128
Enterprise Traps.....	128
Oracle USM Show Commands.....	128
show sipd endpoint-ip.....	128
show sipd third-party.....	129
show sipd local-subscription.....	129
show registration.....	131
show home-subscriber-server.....	132
show http-server.....	134
Supporting Configuration.....	135
Session Load Balancer Support.....	135
Verify Config.....	135
sip authentication profile (CX).....	135
sip authentication profile (ENUM).....	136
sip authentication profile (Local).....	136
sip-registrar.....	136
sip-registrar.....	136
Resource Utilization.....	137
CPU Overload Protection.....	137
Heap Utilization.....	137

About this Guide

Oracle® Communications Unified Session Manager (USM) applies core session control to reduce the complexity and cost of delivering high-value, revenue generating SIP multimedia services. Oracle USM can be used to support a broad range of SIP services including residential or business voice, GSMA-defined Rich Communication Suite (RCS) services and fixed mobile convergence (FMC) for small subscriber populations or initial service rollouts.

Release Version S-Cz7.1.5 M1 is supported on the Acme Packet 4500 and Acme Packet 6300 series hardware.

Related Documentation

The following table lists the members that comprise the documentation set for this release. Refer to version SCZ7.1.2 of the applicable software documents:

Document Name	Document Description
Acme Packet 4600 Hardware Installation Guide	Contains information about the components and installation of the Acme Packet 4600 system.
Acme Packet 4500 Hardware Installation Guide	Contains information about the components and installation of the Acme Packet 4500 system.
Acme Packet 6100 Hardware Installation Guide	Contains information about the components and installation of the Acme Packet 6100 system.
Acme Packet 6300 Hardware Installation Guide	Contains information about the components and installation of the Acme Packet 6300 system.
Release Notes	Contains information about the current documentation set release, including new features and management changes.
ACLI Configuration Guide	Contains information about the administration and software configuration of the Oracle Communications Session Border Controller.
ACLI Reference Guide	Contains explanations of how to use the ACLI, as an alphabetical listings and descriptions of all ACLI commands and configuration parameters.
Maintenance and Troubleshooting Guide	Contains information about logs, performance announcements, system management, inventory management, upgrades, working with configurations, and managing backups and archives.
MIB Reference Guide	Contains information about Management Information Base (MIBs), Acme Packet's enterprise MIBs, general trap information, including specific details about standard traps and enterprise traps, Simple Network Management Protocol (SNMP) GET query information (including standard and enterprise SNMP GET query names, object identifier names and numbers, and descriptions), examples of scalar and table objects.
Accounting Guide	Contains information about accounting support, including details about RADIUS accounting.
HDR Resource Guide	Contains information about the Historical Data Recording (HDR) feature. This guide includes HDR configuration and system-wide statistical information.
Administrative Security Essentials	Contains information about Administrative Security license support.
Security Guide	Contains information about security considerations and best practices from a network and application security perspective for the Oracle Communications

About this Guide

Document Name	Document Description
	Session Border Controller family of products. The Oracle USM and the Oracle CSM are members of the Oracle Communications Session Border Controller family of products.

Release Caveats

The following sections list caveats related to Version S-Cz7.1.5 M1 of the Oracle USM.

- Configuring support for SNMPv3 is not supported as a real-time configuration change. Reboot the system after establishing an SNMPv3 configuration on the Oracle USM.
- Do not load configurations from sibling products, the Oracle SBC for example, on the Oracle USM. Those configurations are incompatible with the Oracle USM, causing incorrect operation. Users should configure the Oracle USM from scratch or use another valid Oracle USM configuration.
- The ISC interface does not work when dialog transparency is enabled on the Oracle USM.
 - Resolution - Do not enable dialog transparency if your Oracle USM must support ISC.
- The fallback-to-local-policy option only works with ENUM-based deployments. Do not configure this option for Cx or LST-based deployments.
- The Oracle USM does not work with an iFC when its default handling is set to “SESSION CONTINUED”.
- Multi-stage routing does not work for S-CSCF routing functions.
- The Oracle USM accepts only the first message received from an application server in response to messages from the Oracle USM that included an ODI. The Oracle USM drops any subsequent messages with the same ODI.
 - Resolution - Do not configure an AS to fork responses to the Oracle USM that include an ODI originally provided by the Oracle USM.
- A Oracle USM registrar set to DDNS that also has the following option set does not work properly:
e164-<primary|secondary>-config=enum:<enum-config-name>
 - Issue - Rather than forwarding a call via the DDNS database, the Oracle USM is referring to local policy configuration and forwarding the call to the next hop that results from the local policy lookup.
 - Resolution - Do not configure the option shown above on a registrar that operates with a DDNS database (subscriber-database-method set to DDNS).
- The Oracle USM does not send third party registration for the entire implicit registration set. It only sends this for the specific public user identity that is registering, de-registering or re-registering.
- Instead of routing a message via local policy, the Oracle USM incorrectly issues an LIR under the following conditions:
 - The Oracle USM is not configured with the e164-primary-config and e164-secondary-config options.
 - The Oracle USM receives a request with a tel-URI or a sip-URI with the user=phone parameter.Note that the Oracle USM returns an error if the LIA does not include a server, and routes the message to the server if the LIA includes a server.
- The Oracle USM supports only the following network interface cards:
 - 4 x 1 GigE NIU (SFP)
 - 4 x 1 GigE Copper
 - 4 x 1 GigE Copper
- Remove any other NIU other than those listed above from the chassis before booting your system with an S-CZ7.1.5 image. IPSEC and QoS PHYs are NOT supported with S-CZ7.1.5.

Revision History

Date	Revision Number	Description
April 8, 2014	Revision 1.00	<ul style="list-style-type: none"><li data-bbox="732 302 922 327">• Initial Release
May 9, 2014	Revision 1.10	<ul style="list-style-type: none"><li data-bbox="732 371 1409 430">• Clarifies a caveat indicating that the fallback-to-local-policy option works only with DDNS databases for this release.
November 26, 2014	Revision 1.20	<ul style="list-style-type: none"><li data-bbox="732 470 1450 558">• Clarifies conditions when the use of e164 resolution configuration even if Tel-URI or SIP_URI with (user=phone) is present in the registration cache.
December 26, 2014	Revision 1.30	<ul style="list-style-type: none"><li data-bbox="732 598 1414 657">• Adds caveat indicating PHY card limitations, which exclude IPSEC and QoS PHYs.

Oracle USM Supporting the IMS Core

General Description

The Oracle USM functions in an IMS core. It communicates with the HSS to obtain Authorization, Authentication, S-CSCF assignment, and ultimately routing instructions. To accomplish these functions, the Oracle USM can perform the SIP registrar role in conjunction with an HSS.

Message Authentication for SIP Requests

The Oracle USM authenticates requests by configuring the sip authentication profile configuration element. The name of this configuration element is either configured as a parameter in the sip registrar configuration element's authentication profile parameter or in the sip interface configuration element's sip-authentication-profile parameter. This means that the Oracle USM can perform SIP digest authentication either globally, per domain of the Request URI or as received on a SIP interface.

After naming a sip authentication profile, the received methods that trigger digest authentication are configured in the methods parameter. You can also define which anonymous endpoints are subject to authentication based on the request method they send to the Oracle USM by configuring in the anonymous-methods parameter. Consider the following three scenarios:

- By configuring the methods parameter with REGISTER and leaving the anonymous-methods parameter blank, the Oracle USM authenticates only REGISTER request messages, all other requests are unauthenticated.
- By configuring the methods parameter with REGISTER and INVITE, and leaving the anonymous-methods parameter blank, the Oracle USM authenticates all REGISTER and INVITE request messages from both registered and anonymous endpoints, all other requests are unauthenticated.
- By configuring the methods parameter with REGISTER and configuring the anonymous-methods parameter with INVITE, the Oracle USM authenticates REGISTER request messages from all endpoints, while INVITES are only authenticated from anonymous endpoints.

User Authorization

In an IMS network, the Oracle USM requests user authorization from an HSS when receiving a REGISTER message. An HSS is defined on the Oracle USM by creating a home subscriber server configuration element that includes a name, ip address, port, and realm as its basic defining data.

UAR/UAA Transaction

Before requesting authentication information, the Oracle USM sends a User Authorization Request (UAR) to the HSS for the registering endpoint to determine if this user is allowed to receive service. The Oracle USM populates the UAR's AVPs as follows:

- Public-User-Identity—the SIP AOR of the registering endpoint
- Visited-Network-Identity—the value of the network-id parameter from the ingress sip-interface.
- Private-User-Identity—the username from the SIP authorization header, if it is present. If not, this value is the public User ID.
- User-Authorization-Type—always set to REGISTRATION_AND_CAPABILITIES (2)

The Oracle USM expects the UAA to be either:

- DIAMETER_FIRST_REGISTRATION
- DIAMETER_SUBSEQUENT_REGISTRATION

Any of these responses result in the continued processing of the registering endpoint. Any other result code results in an error and a 403 returned to the registering UA (often referred to as a UE). The next step is the authentication and request for the H(A1) hash.

SIP Digest User Authentication

Authentication via MAR/MAA

To authenticate the registering user, the Oracle USM needs a digest realm, QoP, and the H(A1) hash. It requests these from a server, usually the HSS, by sending it a Multimedia Auth Request (MAR) message. The MAR's AVPs are populated with:

- Public-User-Identity—the SIP AOR of the endpoint being registered (same as UAR)
- Private-User-Identity—the username from the SIP authorization header or the SIP AOR if the AOR for PUID parameter is enabled. (Same as UAR)
- SIP-Number-Auth-Items—always set to 1
- SIP-Auth-Data-Item -> SIP-Item-Number—always set to 1
- SIP-Auth-Data-Item -> SIP-Authentication-Scheme—always set to SIP_DIGEST
- Server-Name—the home-server-route parameter in the sip registrar configuration element. It is the URI (containing FQDN or IP address) used to identify and route to this Oracle USM.

The Oracle USM expects the MAA to include a SIP-Auth-Data-Item VSA, which includes digest realm, QoP and H(A1) information as defined in RFC2617. The information is cached for subsequent requests. Any result code received from the HSS other than DIAMETER_SUCCESS results in a 403 error response returned for the original request.

The MAR/MAA transaction is conducted with the server defined in the credential retrieval config parameter found in the sip-authentication profile configuration element. This parameter is populated with the name of a home-subscriber-server configuration element.

SIP Authentication Challenge

When the Oracle USM receives a response from the HSS including the hash value for the user, it sends a SIP authentication challenge to the endpoint, if the endpoint did not provide any authentication headers in its initial contact with Oracle USM. If the endpoint is registering, the Oracle USM replies with a 401 Unauthorized message with the following WWW-Authenticate header:

```
WWW-Authenticate: Digest realm="atlanta.com", domain="sip:boxesbybob.com",  
qop="auth", nonce="f84f1cec41e6cbe5aea9c8e88d359", opaque="", stale=FALSE,  
algorithm=MD5
```

If the endpoint initiates any other request to the Oracle USM besides REGISTER, the Oracle USM replies with a 407 Proxy Authentication Required message with the following Proxy-Authenticate header:

```
Proxy-Authenticate: Digest realm="atlanta.com", qop="auth",
nonce="f84f1cec41e6cbe5aea9c8e88d359", opaque="", stale=FALSE, algorithm=MD5
```

Authentication Header Elements

- Domain—A quoted, space-separated list of URIs that defines the protection space. This is an optional parameter for the "WWW-Authenticate" header.
- Nonce—A unique string generated each time a 401/407 response is sent.
- Qop—A mandatory parameter that is populated with a value of "auth" indicating authentication.
- Opaque—A string of data, specified by the Oracle USM which should be returned by the client unchanged in the Authorization header of subsequent requests with URIs in the same protection space.
- Stale—A flag indicating that the previous request from the client was rejected because the nonce value was stale. This is set to true by the SD when it receives an invalid nonce but a valid digest for that nonce.
- Algorithm—The Oracle USM always sends a value of "MD5"

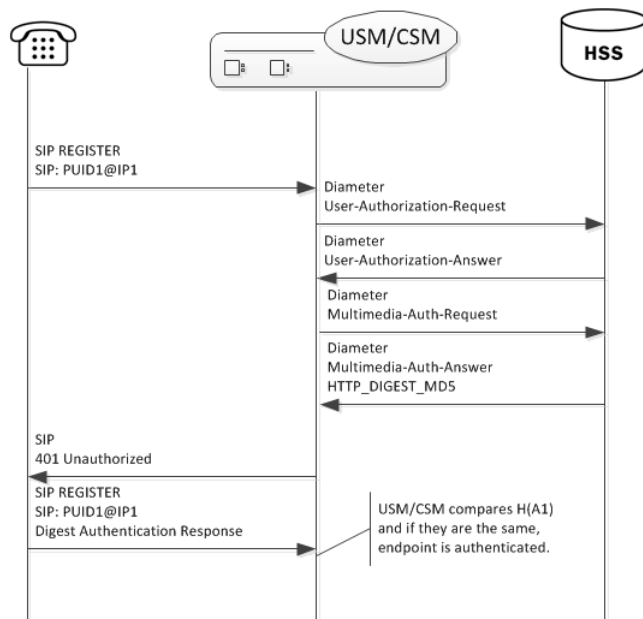
SIP Authentication Response

After receiving the 401/407 message from the Oracle USM, the UA resubmits its original request with an Authorization: header including its own internally generated MD5 hash.

Oracle USM Authentication Check

At this point, the Oracle USM has received an MD5 hash from the HSS and an MD5 hash from the UA. The Oracle USM compares the two values and if they are identical, the endpoint is successfully authenticated. Failure to match the two hash values results in a 403 or 503 sent to the authenticating endpoint.

The following image shows the User Authorization and Authentication process:



Note: Diagram information states "USM/CSM" when the applicable content applies to both the Oracle USM and the Oracle CSM.

The Oracle USM acts as a SIP Registrar and updates an HSS with the state of its registrants.

IMS-AKA Support

The Oracle USM also supports IMS-AKA for secure authentication end-to-end between UAs in an LTE network and an IMS core. It supports IMS-AKA in compliance with 3GPP specifications TS 33-203 and TS 33-102.

The goal of IMS-AKA is to achieve mutual authentication between end station termination mechanisms, such as an IP Multimedia Services Identity Module (ISIM), and the Home Network (IMS Core). Achieving this goal requires procedures both inside and outside the core. Ultimately, IMS performs the following:

- Uses the IMPI to authenticate the home network as well as the UA;
- Manages authorization and authentication information between the HSS and the UA;
- Enables subsequent authentication via authentication vectors and sequence information at the ISIM and the HSS.

The Oracle USM authenticates registrations only. This registration authentication process is similar to SIP Digest. The process accepts REGISTER requests from UAs, conducts authorization procedures via UAR/UAA exchanges and conducts authentication procedures via MAR/MAA exchanges and challenges with the UA.

Configuration and operational support are not the same on the Oracle USM and Oracle CSM. This is because the Oracle USM can perform the P-CSCF role as well as the I-CSCF and S-CSCF roles. Applicable configuration to support IMS-AKA on the P-CSCF access interface is documented in the Security chapter of the *Oracle Communications Session Border Controller CLI Configuration Guide*. This configuration includes defining an IMS-AKA profile, enabling the **sip-interface** for IMS-AKA and configuring the **sip-port** to use the profile.

There is no configuration required for the S-CSCF role, but there is an optional configuration that specifies how many authentication vectors it can accept from the HSS. The S-CSCF stores these authentication vectors for use during subsequent authentications. Storing vectors limits the number of times the device needs to retrieve them from the HSS. The default number of authentication vectors is three.

Authentication Sequence - Registration

UAs get service from an IMS core after registering at least one IMPU. To become registered, the UA sends REGISTER requests to the IMS core, which then attempts to authenticate the UA.

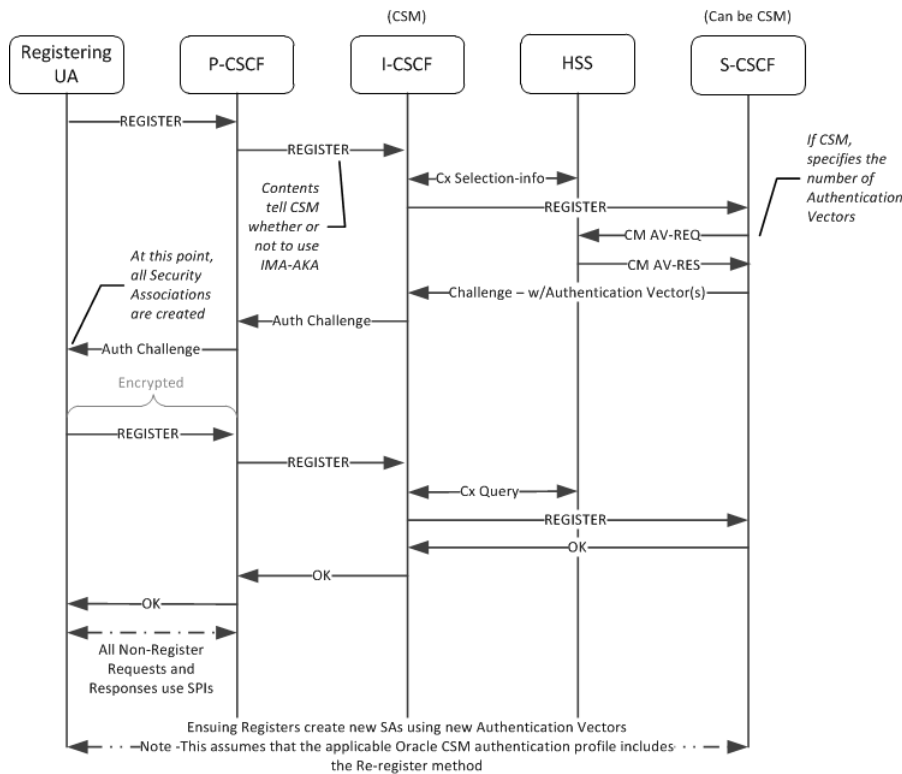
The first device to receive the REGISTER at the core is a P-CSCF, such as the Oracle USM. For the Oracle USM, appropriate configuration determines that it uses IMS-AKA as the authentication mechanism on the access interface. For an Oracle CSM, the presence and state of the “integrity-protected” parameter in the Authorization header of a REGISTER triggers the use of IMS-AKA. If the value of this parameter is either “yes” or “no”, IMS-AKA is invoked. If the parameter is not present, or it is set to any other value, the Oracle USM falls back to SIP Digest authentication.

To proceed with IMS-AKA authentication, the P-CSCF engages in S-CSCF selection procedures via the I-CSCF to identify the target S-CSCF. Having identified the S-CSCF (your Oracle USM), the I-CSCF forwards the REGISTER to it. The I-CSCF next engages in standard UAR and MAR procedures. For IMS-AKA deployments, the HSS follows procedures defined in TS 33-203 to create authentication vectors for the UA. The HSS provides the vectors to the S-CSCF, which then proceeds with authentication procedures defined in TS 33-203.

After processing, the S-CSCF uses authentication vectors to challenge the UA. The UA uses the information in this challenge to, first, authenticate the Home Network. Having confirmed the network, the UA then prepares and sends its authentication information back towards the S-CSCF. The S-CSCF is then responsible for authenticating the UA. The S-CSCF sends a 200OK back to the UA upon successful authentication, allowing the UA to get service from the HN.

The Oracle USM caches the AOR’s registration and stores authentication vectors for subsequent authentications, thereby minimizing the work required by the HSS.


The overall sequence is depicted below.



Outside the Core

LTE networks include UAs that have an IP Multimedia Service Identity Module (ISIM) or equivalent. ISIMs are configured with a long-term key used to authenticate and calculate cipher keys, as well as IP Multimedia Private and Public Identities (IMPI and IMPU). The ISIM serves as the means of authenticating the home network to the UA. The UA, in turn, sends information based on its ISIM configuration to the home network, which can then authenticate the UA.

Establishment of Security Associations (SAs) to and from the UA are the responsibility of the P-CSCF. The P-CSCF should also be capable of managing the processes when the UA is behind a NAT.

 **Note:** Within the context of IMS-AKA, only traffic between the P-CSCF and the UA is encrypted.

Authentication Success

When using IMS-AKA, successful registration of a UA consists of registering at least one IMPU and the IMPI authenticated within IMS. The UA begins this process by sending it REGISTER request to the P-CSCF properly specifying IMS-AKA authentication. IMS then performs standard procedures to identify the appropriate S-CSCF. Upon receipt of the REGISTER, the S-CSCF checks for the presence of an authentication vector. If it is present the S-CSCF issues the authentication challenge; if not, it requests authentication vector(s) from the HSS. Note that the Oracle USM allows you to request multiple authentication vectors via configuration. The HSS provides the following components within an authentication vector:

- RAND—random number
- XRES—expected response
- CK—cipher key
- IK—integrity key
- AUTN—authentication token

The MAR provided to the S-CSCF differ from that of SIP digest authentication requests as follows:

- The SIP-Number-Auth-Items AVP specifies the number of authentication vectors, which is equal to the home-subscriber-server's num-auth-vectors setting.
- The SIP-Authentication-Scheme AVP specifies the authentication scheme, Digest-AKAv1-MD5.

At this point, the Oracle USM can send the authentication challenge to the UA. If multiple authentication vectors were provided by the HSS, the Oracle USM can independently authenticate the UA until the pool is exhausted. The S-CSCF stores the RAND it sends to the UA to resolve future synchronization errors, if any. No authentication vector can be used more than once. This is validated by the ISIM, using a sequence number (SQN).

When a P-CSCF receives an authentication challenge, it removes and stores the CK and the IK. The P-CSCF forward the rest of the information to the UA.

The UA is responsible for verifying the home network. Having received the AUTN from the P-CSCF, the UA derives MAC and SQN values. Verifying both of these, the UA next generates a response including a shared secret and the RAND received in the challenge. The UA also computes the CK and IK.

Upon receipt of this response, IMS provides the message to the S-CSCF, which determines that the XRES is correct. If so, it registers the IPMU and, via IMS sends the 200 OK back to the UA.

Authentication Failure

Either the UA or IMS can deny authentication via IMS-AKA. In the case of the UA, this is considered a network failure; in the case of IMS there would be a user authentication failure.

Network Authentication Failure

The UA determines that the HN has failed authentication, it sends a REGISTER request with an empty authorization header parameter and no authentication token for synchronization (AUTS). This indicates that the MAC parameter was invalid as determined by the UA. In this case, the S-CSCF sends a 403 Forbidden message back to the UA.

User Authentication Failure

IMS-AKA determines user authentication failure as either:

- IK incorrect—If the REGISTER includes a bad IK, the P-CSCF detects this and discards the packet at the IPSEC layer. In this case, the REGISTER never reaches the S-CSCF.
- XRES incorrect—In this case, the REGISTER reaches the S-CSCF. The S-CSCF detects the incorrect XRES, the S-CSCF sends a 4xxx Auth_Failure message back to the UA via IMS.

Synchronization

Synchronization refers to authentication procedures when the (REFRESH TIMING) is found to be stale. This is not an authentication failure.

The UA may send an AUTS in response to the challenge, indicating that the authentication vector sequence is "out-of-range". Upon receipt of the AUTS, the S-CSCF sends a new authorization vector request to the HSS. The HSS checks the AUTS and, if appropriate sends a new set of authentication vectors back the the S-CSCF. Next the S-CSCF sends 401 Unauthorized back to the UA. Assuming the UA still wants to register, this would trigger a new registration procedure.

Optional IMS-AKA Configuration

The following configuration enables the Oracle USM to specify, on a per-HSS basis, the number of authentication vectors it can download per MAR. Making this setting is not required as it has a valid default entry (3).

home subscriber server

To configure the number of authentication vectors to download from a home subscriber server (HSS):

1. In Superuser mode, type **configure terminal** and press Enter.

```
ORACLE# configure terminal
```

2. Type **session-router** and press Enter to access the session router path.

```
ORACLE (configure) # session-router
```

3. Type **home-subscriber-server** and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE (session-router) # home-subscriber-server
ORACLE (home-subscriber-server) #
```

4. **Select**—If already configured, choose the home subscriber server for which you want to set the number of authentication vectors.
5. **num-auth-vector**— [1-10] 3 default - The number of authentication vectors downloaded from HSS per MAR. The range is from 1-10 with 3 as the default.
6. Type **done** when finished.

Oracle USM as Registrar

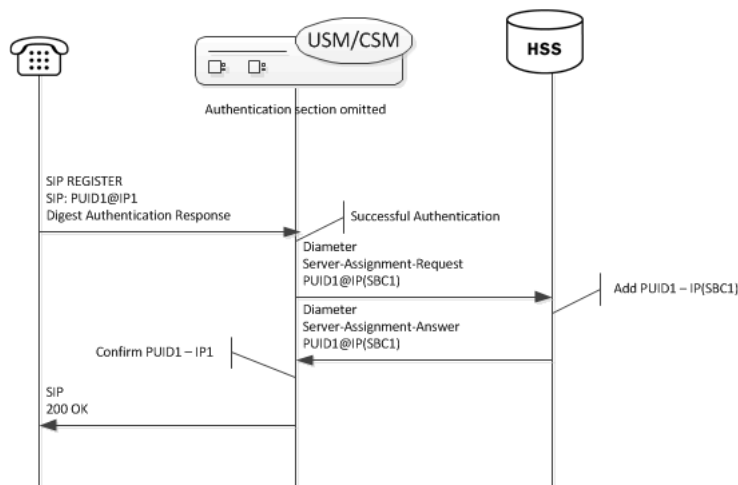
Creating a sip registrar configuration element enables the Oracle USM to act as a SIP registrar. When registration functionality is enabled, the Oracle USM actually registers endpoints rather than only caching and forwarding registrations to another device. Oracle USM registry services are enabled globally per domain, not on individual SIP interfaces or other remote logical entities.

On receiving a REGISTER message, the Oracle USM checks if it is responsible for the domain contained in the Request-URI as defined by the domains parameter and finds the corresponding sip registrar configuration. This is a global parameter—all messages are checked against all sip registrar domains. Thus you could create one sip registrar configuration element to handle all .com domains and one sip registrar configuration element to handle all .org domains. The Oracle USM begins registrar functions for all requests that match the configured domain per sip-registrar configuration element.

A UA is considered registered once a SAA assignment is received from the HSS, after which the Oracle USM sends a 200 OK message back to the registering UA.

New Registration

The following image shows a simplified call flow for a registering user:



Limiting AOR Contacts

The Oracle USM allows you to limit the number of contacts that apply to AORs. If the Oracle USM receives a registration request that exceeds the maximum that you configured, it responds with a local response, a 403 Forbidden by default, and does not register the additional contact. The system only rejects registration requests that exceed the maximum. Existing contacts persist normally.

The system checks against the maximum in the following circumstances:

- A new registration is received
- The location-update-interval expires
- A call-id changes (and the forward-reg-callid-change option is enabled)
- A registrar message sequence number has skipped a number
- There is any change to the contact list

If the number of contacts in the initial registration exceeds the maximum, the Oracle USM rejects the entire registration. In addition, if you configure this feature while the system is operational, your setting only applies to new registrations.

You configure these maximums on a per-registrar basis. The value ranges from 0-256. The feature is RTC supported.

HSS Server Assignment

As the Oracle USM registers UAs, it requests to assign itself as the S-CSCF for the registering AoR. The Oracle USM's S-CSCF identity is configured in the home-server-route parameter in sip-registrar configuration element. This is entered as a SIP URI (containing FQDN or IP address) and is used to identify and route messages to this Oracle USM on behalf of the registered user.

Server Assignment Messages

The Oracle USM sends a Server Assignment Request (SAR) to the HSS requesting to confirm the SIP or SIPS URI of the SIP server that is currently serving the user. The SAR message also serves the purpose of requesting that the Diameter server send the user profile to the SIP server. The SAR's AVPs are populated as follows:

- Public-User-Identity—the SIP AOR of the endpoint being registered (same as UAR)
- Private-User-Identity—the username from the SIP authorization header, if it is present. If not, this value is the public User ID. (Same as UAR)
- Server-Name—the home server route parameter in the sip-registrar configuration element. It is the FQDN or IP address used to identify and route to this Oracle USM sent as a URI.
- Server-Assignment-Type—the value of this attribute depends upon the registration state:
 - REGISTRATION (1)—for all new and refreshing registrations.
 - Set to TIMEOUT_DEREGISTRATION (4)—when the contact is unregistered due to expiration. This occurs if the force-unregistration option is configured in the sip config.
 - USER_DEREGISTRATION (5)—when the contact is unregistered by the user (contact parameter expires=0).
- User-Data-Already-Available—always set to USER_DATA_ALREADY_AVAILABLE (1)

Server-Assignment-Response

The Oracle USM expects a DIAMETER_SUCCESS code in the SAA to indicate that the assignment was successful. Then a 200 OK response is returned to the registering user. Any other Diameter result code is an error and results in an error response for the original REGISTER request (by default 503) and the contacts to be invalidated in the registration cache.

Register Refresh

When a UA sends a register refresh, the Oracle USM first confirms that the authentication exists for that UE's registration cache entry, and then is valid for the REGISTER refresh. (If a valid hash does not exist for that AoR, then the Oracle USM sends an MAR to the HSS to retrieve authentication data once again).

Next, the Oracle USM determines if it can perform a local REGISTER refresh or if the HSS needs to be updated. If any of the following 3 conditions exists for the re-registering UA, the Oracle USM updates the HSS:

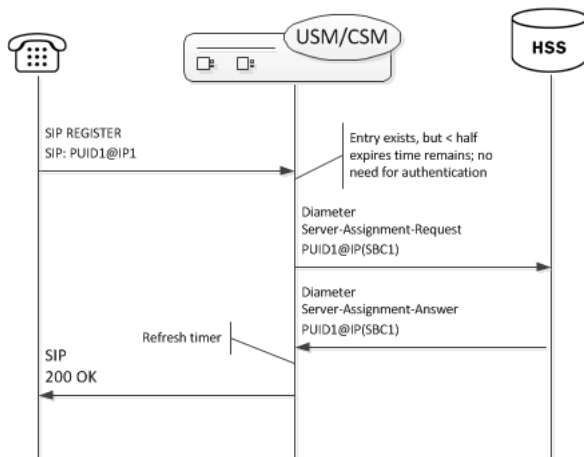
- The location update interval timer has expired—This value, configured in the sip registrar configuration element ensures that HSS database always has the correct Oracle USM address by periodically sending SARs for each registered contact.

- The message’s call-id changes while the forward-reg-callid-change option in the sip config configuration element is set. This covers the case where the UA changes the Oracle USMs through which it attaches to the network.
- The REGISTER message’s Cseq has skipped a number. This covers the case in which a user registered with Oracle USM1, moves to Oracle USM2, and then returns to Oracle USM1.

If the Oracle USM updates the HSS database because of matching one of the above conditions, the access side expiration timer per contact is reset to the REGISTER message’s Expires: header value, and returned in the 200 OK. This happens even in the case when the reREGISTER was received in the first half of the previous Expires period. In addition, the core-side location update interval timer are refreshed on both active and standby.

When the above three conditions are not met, the registration expiration proceeds normally.

If the timer has not exceeded half of its lifetime, a 200 OK is returned to the UA. If the timer has exceeded half of its lifetime, the Oracle USM just refreshes the access-side expiration timer; the registration cache expiration timer for that AoR begins its count again.



Core-side SAR Lifetime

The Oracle USM maintains a timer for user registrations per SAR on the core side as specified above. The core-side SAR lifetime timer is configured in the location update interval parameter in the sip registrar configuration element. This timer ensures that the HSS always has the correct Oracle USM address, by sending SAR messages periodically.

Entry Unregistration

Because AoRs and not contacts are referenced by the HSS, an AoR is valid and should not be removed from HSS until all associated contacts have been removed or expired. If all the contacts are removed for an AoR by receiving REGISTER messages with Expires:0 header, then the SAR sent to the HSS includes Server-Assignment-Type of USER_DEREGISTRATION (5).

When the force-unregister option in the sip config is enabled, then the HSS is explicitly updated when all of the contacts for an AoR have expired. This event prompts the Oracle USM to send a SAR to the HSS using the Server-Assignment-Type of TIMEOUT_DEREGISTRATION (4).

The HSS can send a Registration-Termination-Request to request removing a registration, which corresponds to entries in the Oracle USM’s registration cache. When an RTR is received, the following AVPs are expected:

- Private-User-Identity—Username of the user, which is being de-registered.
- Associated-Identities—The Private-Id's in the same subscription which need to be de-registered. (optional)
- Public-Identity—One or more public-Id's of the user being de-registered. (optional)

For the AoR specified by the Private-User-Identity AVP, all associated contacts are removed in the registration cache. The Oracle USM sends a Registration Termination Answer to the HSS to indicate success.

User Registration based on Reg-ID and Instance-ID (RFC 5626)

Sometimes a user's device reregisters from a different network than its original registration. This event should be considered a location update rather than a completely new registration for the Contact. The Oracle USM can perform this way by considering the endpoint's reg-id and instance-id parameters defined in [RFC 5626](#).

The Oracle USM identifies new REGISTER requests received on a different access network as a location update of the existing binding between the Contact and AoR. Without this feature, the Oracle USM would create a new binding and leave the old binding untouched in the local registration cache/ENUM database. This scenario is undesirable and leads to unnecessary load on various network elements including the Oracle USM itself.

The following conditions must be matched to equate a newly registering contact as a location update:

For a received REGISTER:

- The message must not have more than 1 Contact header while 1 of those Contact headers includes a reg-id parameter. (failure to pass this condition prompts the Oracle USM to reply to the requester with a 400 Bad Request).
- The Supported: header contains outbound value
- The Contact header contains a reg-id parameter
- The Contact header contains a +sip.instance parameter

After these steps are affirmed, the Oracle USM determines if it is the First hop. If there is only one Via: header in the REGISTER, the Oracle USM determines it is the first hop and continues to perform Outbound Registration Binding processing.

If there is more than 1 Via: header in the REGISTER message, the Oracle USM performs additional validation by checking that a Path: header corresponding to the last Via: includes an ob URI parameter, Outbound Registration Binding may continue.

If the Oracle USM is neither the first hop nor finds an ob URI in Path headers, it replies to the UA's REGISTER with a 439 First Hop Lack Outbound Support reply.

reREGISTER Example

The user (AoR) bob@example.com registers from a device +sip.instance= <urn:uuid:0001> with a reg-id="1", contact URI = sip:1.1.1.1:5060. A binding is created for bob@example.com+<urn:uuid:0001>+reg-id=1 at sip:1.1.1.1:5060.

Next, Bob@example.com sends a reREGISTER with the same instance-id but with a different reg-id = 2 and contact URI = sip:2.2.2.2:5060.

The previous binding is removed. A binding for the new contact URI and reg-id is created. bob@example.com +<urn:uuid:0001>+reg-id=2 at sip:2.2.2.2:5060

Outbound Registration Binding Processing

An outbound registration binding is created between the AoR, instance-id, reg-id, Contact URI, and other contact parameters. This binding also stores the Path: header.

Matching re-registrations update the local registration cache as expected. REGISTER messages are replied to including a Require: header containing the outbound option-tag.

If the Oracle USM receives requests for the same AOR with some registrations with reg-id + instance-id and some without them, the Oracle USM will store them both as separate Contacts for the AOR; The AoR+sip.instance+reg-id combination becomes the key to this entry.

Wildcarded PUID Support

The Oracle USM supports the use of wildcarded Public User IDs (PUIDs), typically for registering multiple endpoints on a PBX with a single PUID. A wildcard is composed of a regular expression that, when used in a PUID prefix, represents multiple UEs. The group of UEs is referred to as an implicit registration set and share a single service profile. This support is typically implemented to reduce HSS resource requirements. The regular expressions themselves are in form of Perl Compatible Extended Regular Expressions (PCRE).

Each implicit registration set is associated with an explicitly registered distinct PUID. Typically, this distinct PUID is the PBX itself. The implicit registration set is dependent on the distinct PUID, including the distinct PUID's registration status.

There is no Oracle USM configuration required.

Wildcarded PUID support is applicable to both I-CSCF and S-CSCF operation. In addition, all Oracle USMs in the applicable data paths must be in the same trust domain.

To allow the feature, the Oracle USM supports:

- Wildcarded PUID AVP in the LIR, SAR and SAA
- User Profile AVP in the SAA
- P-Profile-Key across the Mw interface, as defined in RFC 5002

Note also that the HSS must support the wildcarded-public-Identify AVP.

Oracle USM Licensing

The Oracle USM requires three licenses: Registration Cache Limit, Cx, SIP Authorization/Authentication.

For CX-based Oracle USM, the Cx license reveals the home subscriber server configuration element and the SIP Authorization/Authentication license reveals the SIP Authentication Profile configuration element. Configuring both configuration elements is required to operate a Oracle USM. Refer to the Licensing and Database Registration Limits section for the third license required for Oracle USM operation.

Refer to the Net-Net SBC ACLI Configuration guide, Getting Started chapter for how to install licenses in your system.

ACLI Instructions

The following configuration enables the Oracle USM to authorize and authenticate registering users. In addition it sets the Oracle USM to request itself as the S-CSCF for the registering users.

home subscriber server

To configure a home subscriber server (HSS):

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type home-subscriber-server and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# home-subscriber-server
ORACLE(home-subscriber-server)#
```

4. name—Enter the name for this home subscriber server configuration element to reference from other configuration elements.
5. state—Set this to enabled to use this configuration element.

6. address—Enter the IP address of this HSS.
7. port—Enter the port which to connect on of this HSS, the default value is 80.
8. realm—Enter the realm name where this HSS exists.
9. Type done when finished.

SIP Authentication Profile

To configure the SIP Authentication Profile:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type sip-authentication-profile and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-authentication-profile
ORACLE(sip-authentication-profile)#
```

You may now begin configuring the SIP Authentication Profile configuration element.

4. name—Enter the name of this SIP authentication profile that will be referenced from a SIP registrar (or a SIP interface) configuration element.
5. methods—Enter all the methods that should be authenticated. Enclose multiple methods in quotes and separated by commas.
6. anonymous-methods—Enter the methods from anonymous users that require authentication. Enclose multiple methods in quotes and separated by commas.
7. digest-realm—Leave this blank for Cx deployments.
8. credential-retrieval-method—Enter CX.
9. credential-retrieval-config—Enter the home-subscriber-server name used for retrieving authentication data.
10. Type done when finished.

SIP Interface

The full SIP interface should be configured according to your network needs. Please refer to the Oracle SBC ACLI Configuration Guide.

To configure a SIP Digest Authentication on a specific SIP Interface:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type sip-interface and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-interface
ORACLE(sip-interface)#
```

4. Type select and choose the number of the pre-configured sip interface you want to configure.

```
ORACLE(sip-interface)# select
<realm-id>:
1: private 192.168.101.17:5060
2: public 172.16.101.17:5060
selection: 1
```

5. registration-caching—Set this parameter to enabled.

6. `ims-access`—Set this parameter to enabled for access interfaces, when applicable. Core interfaces should have this feature disabled.
7. `sip-authentication-profile`—Set this to the name of an existing sip-authentication profile if you wish to authenticate per SIP interface.
8. Type done when finished.

SIP Registrar

To configure the Oracle USM to act as a SIP Registrar:

1. In Superuser mode, type `configure terminal` and press Enter.

```
ORACLE# configure terminal
```

2. Type `session-router` and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type `sip-registrar` and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)#
```

4. `name`—Enter a name for this SIP registrar configuration element.
5. `state`—Set this to enabled to use this SIP registrar configuration element.
6. `domains`—Enter one or more domains that this configuration element will invoke SIP registration for. Wildcards are valid for this parameter. Multiple entries can be entered in quotes, separated by commas.
7. `subscriber-database-method`—Set this to CX.
8. `subscriber-database-config`—Enter the home-subscriber-server configuration element name that will handle REGISTER messages for this domain. The HSS configuration element includes the actual IP address of the server that SAR's are sent to.
9. `authentication-profile`—Enter a sip-authentication-profile configuration element's name. The sip authentication profile object referenced here will be looked up for a REGISTER message with a matching domain in the request URI. You may also leave this blank for the receiving SIP Interface to handle which messages require authentication if so configured.
10. `home-server-route`—Enter the identification for this Oracle USM that will be sent as the Server-Name in MAR and SAR messages to the HSS. This value should be entered as a SIP URI.
11. `location-update-interval`—Keep or change from the default of 1400 minutes (1 day). This value is used as the timer lifetime for core-side HSS updates.
12. Type done when finished.

Maximum Number of Contacts

To configure a sip-registrar with a maximum of 10 contacts per AOR:

1. From superuser mode, use the following command sequence to access sip-registrar element.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)# select
```

Select the registrar you want to configure.

2. Specify the number of contacts.

```
AORACLE(sip-registrar)# max-contacts-per-aor 10
AORACLE(sip-registrar)# done
```

Response to Exceeding Maximum Contacts

To configure local response for the Oracle USM to issue when `max-contacts-per-aor` is exceeded:

Oracle USM Supporting the IMS Core

1. From superuser mode, use the following command sequence to access local-response and add an entry.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# local-response-map
```

2. Access the entries configuration.

```
ORACLE(local-response-map)# entries
```

3. Specify the local error you need to configure.

```
ORACLE(local-response-map-entry)# local-error contacts-per-aor-exceed
```

4. Specify the sip-reason for this error.

```
ORACLE(local-response-map-entry)# sip-reason forbidden
```

5. Specify the error code for this error.

```
ORACLE(local-response-map-entry)# sip-status 403
ORACLE(local-response-map-entry)# done
local-response-map-entry
    local-error                contacts-per-aor-exceed
    sip-status                  403
    q850-cause                  0
    sip-reason                  forbidden
    q850-reason
    method
    register-response-expires
ORACLE(local-response-map-entry)# exit
```

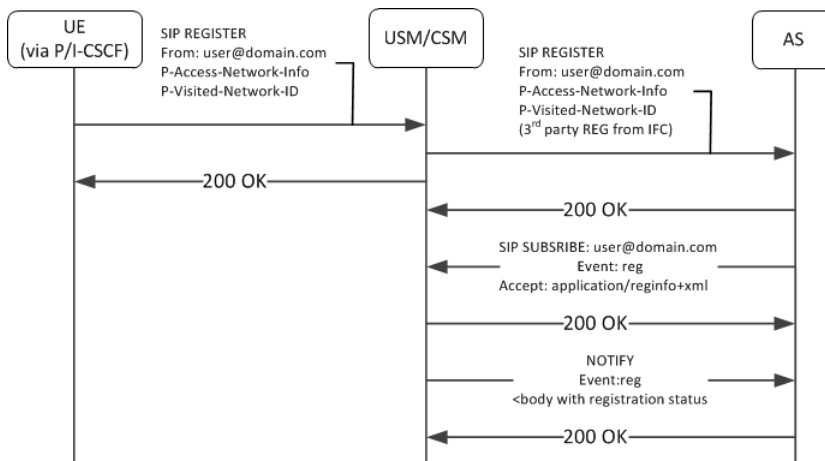
SIP Registration Event Package Support

The Oracle USM supports UA subscriptions to the registration event package, as defined in RFC3680. As such, it maintains contact with entities, often application servers, that need to know about UA registration events and provides those application servers with notifications when registration events occur.

Common usage for this functionality includes:

- Forcing re-authentication
- The provision of welcome notices to users who need information or instructions customized to their location

An operational example, shown below, begins with the Oracle USM performing 3rd party registration on behalf of a UA to an AS, based on the iFC request from the UA. The AS, being an appropriately authorized UA itself, subscribes to NOTIFY messages on reg events for the initial UA. The Oracle USM sends a 200OK to the AS, and then proceeds to forward NOTIFY messages about that UE's registration events to the AS.



This feature is relevant when the Oracle USM is performing S-CSCF functions. You enable this feature on the Oracle USM per registrar, by simply creating a profile and applying it to the applicable registrar.

SUBSCRIBE Processing

When the Oracle USM has the reg-event notification function enabled for a registrar, it includes the allow-events header in its 200OK replies to successful REGISTERS. This lets UEs know that they can subscribe to registration event packages.

When the Oracle USM receives reg-event subscription requests, it follows the sequence below to process SUBSCRIBE requests for reg events:

1. Determines validity of the request.

Subscriptions cannot include more than one package name. If there is more than one package name in the request, the Oracle USM replies with a 400 Bad Request message.

2. Determines if it can be a notifier, as follows:

- The SUBSCRIBE must include EVENT=reg.
- The requesting UA must be in the same domain as the registrar.

If both of the above are true, the Oracle USM proceeds with the request.

3. Authorizes the request. The Oracle USM only authorizes requests from UEs that come from the same realm and layer 2 connection on which it received the initial REGISTER.

Furthermore, the Oracle USM only authorizes the following UEs:

- Public user identities from UEs that are subscribing to their own registration events.
- Public user identities that this user owns. Examples include implicitly registered public user identities.
- Entities that were included in the PATH header of the target UE's registration.
- All ASs that are listed in the UE's iFC and that are part of the trust domain.

If all of the above are true, the Oracle USM proceeds with the request. If not, it sends 403 Forbidden to the requester.

4. Determines how it is functionally related to the UA. The Oracle USM only processes subscriptions for users in its registration cache, replying with a 403 Forbidden if not. For cached users, the Oracle USM forwards the request to the registrar if it is the P-CSCF. If it is the S-CSCF, it sends a 200 OK and begins to act as notifier.

5. Identifies the subscription duration, as follows, and sends the 200 OK to the UE:

If there is no Expires header in the UE's 200OK message, the Oracle USM applies its own configured minimum or the default (600000 seconds), whichever is greater.

If the SUBSCRIBE includes an Expires header, the Oracle USM honors the request unless it is less than the configured minimum.

If the SUBSCRIBE's Expires header is less than the minimum subscription time configured in the registration event profile, the Oracle USM denies the subscription, sending a 423 Too Brief message.

When the Oracle USM encounters an Expires header set to 0, it terminates the subscription. This is referred to as unsubscribing.

SUBSCRIBE REFRESH Requests

Subscriptions must be refreshed to keep them from expiring. ASs accomplish this by sending SUBSCRIBE REFRESH messages to the Oracle USM. Messages must be received from authorized subscribers and on the same realm and connection as the original SUBSCRIBE or the Oracle USM rejects the refresh request.

Reg Event NOTIFY Messages

When configured, the Oracle USM issues NOTIFY messages to subscribed ASs when significant registration events occur. NOTIFY messages sent by the Oracle USM comply fully with RFC3680. Events that trigger NOTIFY messages include:

- Registered
- Registration refreshed
- Registration expired
- Registration deactivated
- UE unregistered

The Oracle USM does not send NOTIFY messages for the following events:

- Registration created
- Registration shortened
- Registration probation
- Registration rejected

Additional detail about NOTIFY messages that is specific to the Oracle USM includes:

- The Oracle USM always sends full information on all contacts, and indicates such within the reginfo element. The Oracle USM does not utilize the partial state described within RFC 3680.
- Wildcarded PUIDs are included, enclosed in the <wildcardedIdentity> tag within the <registration> element.
- The Oracle USM does not include the following optional attributes within the contact element:
 - expires
 - retry-after
 - duration registered
 - display-name
- The Oracle USM uses the optional unknown-param element within the contact element to convey UA capabilities and distribute reg-id, sip.instance and header filed attributes.

An example of the XML body of a NOTIFY message below documents the registration status for the AOR joe@example.com.

```
<reginfo xmlns="urn:ietf:params:xml:ns:reginfo" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="0" state="full">
  <registration aor="sip:joe@example.com" id="as9" state="active">
    <contact id="6" state="active" event="registered">
      <uri>sip:joe@pc887.example.com</uri>
    </contact>
    <contact id="7" state="terminated" event="expired">
      <uri>sip:joe@university.edu</uri>
    </contact>
  </registration>
</reginfo>
```

Use the show registration and show sipd subscription commands to display all information about each subscription.

Reducing NOTIFY Traffic

RFC 3265 stipulates that the Subscription server sends NOTIFY messages to all subscribers when a UA sends a registration refresh. This can generate excessive NOTIFY traffic. You, however, can mitigate this by configuring the Oracle USM to limit notification traffic. By specifying the number of seconds between NOTIFY messages, you prevent the Oracle USM from sending notifications upon events that do not generate a change in the registration database.

Database changes that trigger notifications when this option is configured include:

- The Cseq number of the REGISTER message increases by more than 1
- The call-ID changes
- A contact parameter changes
- The number of contacts changes

Upon expiry of this timer, the Oracle USM sends out a NOTIFY for every registration event subscription. Note also that the Oracle USM does not send the cseq attribute in the CONTACT element when this interval is configured.

Configuring Registration Event Package

This section shows you how to create reg-event profiles and apply those profiles to sip-registrars. These profiles enable the monitoring of UA registration events and the delivery of state change notifications to each UA that subscribes to the package. The procedure includes:

- Create one or more registration-event profiles
- Apply each profile to the applicable sip-registrar
- Optionally specify the registration event notification interval timer

Registration Event Profile Configuration

To configure a registration event profile:

1. From superuser mode, use the following command sequence to access regevent-notification-profile command.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# regevent-notification-profile
ORACLE(registration-event-profile)#
```

2. To define the profile, simply name it and specify a timeout in seconds.

```
ORACLE(registration-event-profile)# name reg-event-profile1
ORACLE(registration-event-profile)# min-subscription-duration 2500
ORACLE(registration-event-profile)# done
ORACLE(registration-event-profile)# exit
```

3. Navigate to the registrar for which you want registration event package support.

```
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)# regevent-notification-profile reg-event-profile1
ORACLE(sip-registrar)# done
ORACLE(sip-registrar)# exit
```

Optional NOTIFY Refresh Frequency

To specify optional NOTIFY refresh frequency:

1. From superuser mode, use the following command sequence to access registration-event-profile command within session router.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# regevent-notification-profile
ORACLE(registration-event-profile)#
```

2. To enable NOTIFY, set the send-notify-for-reg-refresh option to the time, in seconds,

```
ORACLE(registration-event-profile)# options notify-refresh-interval=1800
ORACLE(registration-event-profile)# done
ORACLE(registration-event-profile)# exit
```

Prepend the option with the + sign if you have multiple options configured that you want to retain.

```
ORACLE(registration-event-profile)# options +notify-refresh-interval=1800
```

Running the command without the + character causes the system to remove any previously configured options.

Message Routing

The Oracle USM provides two major types of routing that use the routing precedence parameter in the sip registrar. Routing precedence can be set to either registrar (HSS) or local policy. Routing precedence is set to registrar by default. There are additional controls that the user may configure to refine message routing.

Registrar routing uses the configured subscriber database and registration cache to route the call. Local policy routing lets you configure routing decisions within the Oracle USM's local policy routing functionality.

Within the context of local policy routing, the Oracle USM chooses the next hop through the network for each SIP session based on information received from routing policies and constraints. Routing policies can be as simple as routing all traffic to a proxy or routing all traffic from one network to another. Routing policies can also be more detailed, using constraints to manage the volume and rate of traffic that can be routed to a specific network. For example, you can manage volume and rate of traffic to enable the Oracle USM to load balance and route around softswitch failures.

When a message arrives at the Oracle USM, it determines whether it is coming from a session agent. If so, the Oracle USM checks whether that session agent is authorized to make the call. Local policy is then checked to determine where to send the message.

Depending on whether the Oracle USM is performing originating or terminating services for the call, described in the chapter on operations within the IMS core, it performs those services prior to routing to the endpoint.

If the Oracle USM is unable to proceed with routing a request, it replies to the UA that sent the request with a 4xx response.

This chapter provides an overview of registrar routing for perspective, but focuses on local policy routing. Local policy routing is configuration intensive, allowing precise route specification. As a result, configuring local policy routing is a complex process requiring that the user understand the purpose and interaction of multiple configuration elements. This chapter also provides descriptions and configuration instruction on additional routing controls, such as the use of multistage and UA capability routing.

Registrar Routing

When the routing precedence parameter is set to registrar, the Oracle USM is using the HSS as a resource within the context of its routing decisions.

When an INVITE arrives, the Oracle USM checks its own registration cache for a pre-existing matching contact in the INVITE. If it finds a match, it forwards the request to that location. If it does not find a match, it issues an Location Information Request (LIR) to the HSS. If the HSS's response, called an LIA, provides an assigned S-CSCF for that UA, the Oracle USM proceeds as described below in the section LIR/LIA Transaction.

Note that you can configure the Oracle USM to fallback to a local policy lookup if the lookup via the registrar fails. Configure this by adding the fallback-to-localpolicy option to the sip-registrar configuration element.

For situations where the database routing decision needs to be done in lieu of the default, you can set routing precedence to local-policy. Note that you can configure a routing entry that points to an HSS by setting a policy attribute with a next-hop of cx:<home-subscriber-server-name> within the local-policy.

LIR/LIA Transaction

An LIR includes the Public-User-Identity AVP, which contain a UA's actual PUID. The HSS responds with the assigned S-CSCF server (often a Oracle USM) for this PUID. The answer is the form of a Location Info Answer (LIA). The LIA includes the assigned S-CSCF in the Server Name AVP.

If the S-CSCF returned in the LIR is this Oracle USM, then the Oracle USM performs unregistered termination services for this UA. (This situation indicates that the UA is currently unregistered.) Such services could include directing the call to voice mail. If the HSS returns an S-CSCF in the LIA that is not this Oracle USM, it forwards the request to that S-CSCF.

Default Egress Realm

The sip registrar configuration element should be configured with a default egress realm id. This is the name of the realm config that defines the IMS control plane, through which all Oracle USMs, HSSs, and other network elements communicate and exchange SIP messaging. It is advisable to configure this parameter to ensure well defined reachability among Oracle USMs.

RX Interface Features

The Oracle USM can run the Rx interface over a Diameter connection and act as a P-CSCF communicating with a PCRF. The Rx interface supports quality of service and policy management within applicable network infrastructures. See the Oracle SBC ACLI Configuration Guide for full descriptions of this functionality.

Routing Based on UA Capabilities

In compliance with RFC 3841, the Oracle USM is able to make forwarding and forking decisions based on preferences indicated by the UA. To do this, the Oracle USM evaluates each callee's AOR contact to determine the capabilities advertised by the UA and uses this information to make forwarding and forking decisions.

Prior to this support, the Oracle USM made routing preference decisions solely via the q value present in the contact header. In cases where the preferences were equal, the Oracle USM simply forwarded to those contacts simultaneously (parallel forking). In cases where the q value were not equal, the Oracle USM forwarded in sequence (sequential forking), forwarding to the highest q value first.

The Oracle USM now extends upon this functionality by scoring contacts, based on their capabilities, and making forwarding decisions using that score in addition to the q value.

There is no additional Oracle USM configuration required to enable or invoke this processing. This functionality is supported for HSS, ENUM and Local Database configurations.

UE Capabilities

RFC2533 includes a framework that defines feature sets. Feature sets make up a group of media capabilities supported by a UA, individually referred to as media feature tags. In session networks, feature tag information is converted to a form specified in RFC3840 and exchanged between devices in the network to establish lists of UA capabilities. Based on these capabilities, session operation procedures are performed that facilitate preferred communications modalities.

RFC3840 defines:

- The format a UA uses to specify feature sets
- How a UA registers capabilities within the network
- An extension to the contact header so that it can include feature parameters
- The media tags that specify each capability

The full list of applicable media tags is presented in RFC 3840. Examples of tags include audio, automata, data, mobility, application and video.

Registering Capabilities at the Oracle USM

Endpoints register their capabilities by listing them in the Contact headers of the REGISTER request. The Oracle USM stores these feature parameters in its registration cache along with the other contact information. In the case of ENUM databases, the Oracle USM also sends capabilities information to the ENUM infrastructure so that it can maintain capabilities records.

In addition to the standard set of tags, the Oracle USM supports storing custom feature tags. Tags formatted with a + sign preceding the tag are recognized as custom tags. The exception to this are tags formatted using +sip.<tagname>, which are registered sip media feature tags.

An example of a contact header specifying audio, video and methods capabilities is shown below:

```
Contact: sip:u1@h.example.com;audio;video;methods="INVITE,BYE";q=0.2
```

Preferential Routing

The Oracle USM routes using UA capabilities only when acting as S-CSCF. It calculates preferred forwarding and forking using this information in conjunction with UA requests. This calculation is based on Preferential Routing, as defined in RFC3841. Note that the q value is used in this calculation.

Using Preferential Routing, the Oracle USM creates a target UA list from applicable contacts by matching capabilities with preferences. After creating the match list, it scores UEs based on how closely they match the preferred criteria. The system determines the forwarding order referring to the q value first and then the routing score. UEs for which both scores are equal are forwarded at the same time. All remaining UEs are forwarded sequentially.

The table below presents an example wherein the result of matching and scoring calculations causes the Oracle USM to forwards sequentially to UE3, then UE2, then UE1.

User Agent	q Value	Preferential Score
UE3	1000	1000
UE1	500	1000
UE2	1000	700

UAs may or may not include capability request information in their messages. Preferential routing processing accounts for this by defining both explicit and implicit feature preference processing procedures.

Explicit Feature Preference

RFC3841 defines the two headers that a UA can use to explicitly specify the features the target UA must support, including:

Accept-Contact: — UEs the session initiator would like to reach

Reject-Contact: — UEs the session initiator does not want to reach

When the Oracle USM receives messages that includes these headers, it gathers all the contacts associated with the AOR and creates a target list using preferential routing calculations. The example below, drawn from RFC 3841, specifies the desire to route to a mobile device that can accept the INVITE method.

Accept-Contact: *;mobility="mobile";methods="INVITE"

The “require” and explicit Feature Tag Parameters

RFC 3841 defines operational procedures based on the require and explicit feature tag parameters, which the Oracle USM fully supports. UAs include these parameters in the accept-contact: header to further clarify capabilities requirements for the session. The Oracle USM can use these parameters to exclude contacts or specify the forwarding order.

To summarize the use of these parameters per RFC 3841:

When both parameters are present, the Oracle USM only forwards to contacts that support the features and have registered that support.

If only the require parameter is present, the Oracle USM includes all contacts in the contact list, but uses a forwarding sequence that places the “best” match (with the most matching capabilities) first from those with the same q value.

If only the explicit parameter is present, the Oracle USM includes all contacts in the contact list, but uses a forwarding sequence that places contacts that have explicitly indicated matching capabilities before those with the same q value. Unlike requests that specify both require and explicit, non-matching contacts may be tried if the matching ones fail.

If neither parameter is present, the Oracle USM includes all contacts in the contact list, but determines a “best” match based on the “closest” match to the desired capabilities. Again the forwarding order starts with contacts that have the same q value.

Note that this preferential routing sequence can proceed with attempts to reach contacts with a lower q value after the sequences above are exhausted. Note also that the orders calculated by preferential routing never override any forwarding order specified by the UA.

Implicit Feature Preference

If the caller does not include accept-contact or reject-contacts in the message, the Oracle USM makes implicit feature preference assumptions. Implicit feature preference forwards messages to target UEs that support the applicable method, and, in the case of SUBSCRIBE requests, that support the applicable event.

For implicit feature preference cases, the Oracle USM uses the UE's q value solely to determine parallel and sequential forking.

ACLI Instructions

Configuring the SIP Registrar's Routing Precedence

To configure a SIP registrar configuration element for message routing:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE (configure) # session-router
```

3. Type sip-registrar and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE (session-router) # sip-registrar
ORACLE (sip-registrar) #
```

4. Type select and choose the number of the pre-configured sip interface you want to configure.
5. routing-precedence— Set this to either registrar or local-policy depending on your deployment.
6. egress-realm-id—Enter the default egress realm for Oracle USM messaging.
7. Type done when finished.

Home Subscriber Server

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE (configure) # session-router
```

3. Type home-subscriber-server and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE (session-router) # home-subscriber-server
ORACLE (home-subscriber-server) #
```

4. Begin configuring your HSS, or type select and choose the number of the pre-configured HSS you want to configure.
5. Type done when finished.

Supporting Media Sessions Established via ICE

The Oracle USM supports media session establishment for stations running Interactive Connectivity Establishment (ICE) despite the addressing changes made during normal operations. By inserting Oracle USM media interface addresses as relay candidates for all components (RTP, RTCP, and so forth) within the SDP offer/answer, the Oracle USM prevents ICE mismatch detection that would otherwise stop further ICE processing, including pair nomination and status updating. If the Oracle USM did not do this, the call would then proceed under the control of the B2BUA, which would select the media path instead of ICE.

As defined in RFC 5245, ICE defines the insertion of "candidate" addressing within the SDP offer/answer messaging. In conjunction with STUN (RFC 5389), the associated call setup procedure includes connectivity checks to these addresses, which must pass before a media session can proceed. This mechanism helps overcome a variety of network intricacies, including NAT traversal. In addition, RFC 5245 specifies a security mechanism that requires the address

outlined in the default media destination be included as a candidate. This default media destination is presented in the SDP's "m" and "c" lines. Normal SBC operations, to which the Oracle USM complies, includes altering this data to support media negotiation and/or media transport. This scenario causes an ICE mismatch. This feature prevents these mismatches.

Users configure this support on the Oracle USM using a **sip-config** option called **turn-on-the-fly**. With this option enabled, the Oracle USM inserts its own addressing as a relay candidate. This mechanism is based on RFC 5766 Traversal Using Relays around NAT (TURN), which itself is an extension of RFC 5389, Session Traversal Utilities for NAT (STUN).

The Oracle USM inserts itself as a lowest priority candidate. This ensures that ICE would choose the Oracle USM as a participant in the media path last.

In the SDP example below, the Oracle USM has replaced address and port information in the c and m lines with its own media port information. In this case, ICE checks determine that the Oracle USM address is not in the candidate list, to the media flows normally without further ICE processing.

```
c=IN IP4 172.16.101.15
m=audio 15086 RTP/AVP 103

a=candidate:5c0a80165 1 UDP
2130706431 172.16.100.166 2340 typ srflx
raddr 192.168.1.101 rport 2340
a=candidate:Hc0a80165 1 UDP
1694498815 192.168.1.101 2340 typ host
```

In the SDP example below, the user has enabled the **turn-on-the-fly** option, so the Oracle USM inserts itself as a relay candidate. The ICE checks pass and the media proceeds using ICE processing. Note also the priority designation of 255, which is ICE's lowest priority.

```
c=IN IP4 172.16.101.15
m=audio 15086 RTP/AVP 103

a=candidate:5c0a80165 1 UDP
2130706431 172.16.100.166 2340 typ srflx
raddr 192.168.1.101 rport 2340
a=candidate:Hc0a80165 1 UDP
1694498815 192.168.1.101 2340 typ host
a=candidate:C214c2Jj 1 UDP 255
172.16.101.15 15086 typ relay raddr
172.16.101.15 rport 15086
```

Further implementation details include:

- The Oracle USM uses latching to specify media flow source and destination addressing for endpoints behind a NAT. This is also true for RTCP, requiring the user to enable the **hnt-rtcp** parameter in the **media-manager** element.
- To ensure that the IP address and port pair used by the applicable outbound flow is reused for the inbound flow, enable the **symmetric-latching** parameter in the access signaling **realm** element.
- The Oracle USM disables flow guard timing for calls whose media paths are identified by ICE. Normal SIP session timers take down calls that fail to terminate.
- The Oracle USM sets up NAT entries assuming the RTCP address is (RTP IP address:RTP port +1).
- Use media management on the Oracle USM in conjunction with this **turn-on-the-fly** configuration. If not, the resultant media release by the Oracle USM causes unpredictable media path selection.

Configuring Support for ICE

Use the two configurations, shown below, to enable the Oracle USM to properly manage SDP for ICE processing.

1. From superuser mode, use the following command sequence to access **sip-config** configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-config
```

2. Use the **select** command to access the **sip-config** element.
3. Enable the **turn-on-the-fly** option.

```
ORACLE(sip-config)# +option turn-on-the-fly enabled
ORACLE(sip-config)# done
ORACLE(sip-config)# exit
```

4. Use the **hnt-rtcp** command to enable the insertion of addressing to accommodate NAT traversal by RTCP.

```
ORACLE(configure)# media-manager
ORACLE(media-manager)# hnt-rtcp enabled
```

5. Navigate to the access realm element to enable **symmetric latching**.

```
ORACLE(media-manager)# done
ORACLE(media-manager)# realm-config
ORACLE(realm-config)# select
```

Select the realm to which the source endpoint belongs.

```
ORACLE(realm-config)# symmetric-latching enabled
```

6. Use **done**, exit configuration mode, and run **verify-config** to complete ICE support configuration.

```
ORACLE(realm-config)# done
ORACLE(media-manager)# done
ORACLE(media-manager)# exit
ORACLE(configure)# exit
ORACLE# verify-config
```

```
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Tel-URI Resolution

The Oracle USM can initiate number resolution procedures for requests that have tel-URI or SIP-URI (with user=phone) numbers in the R-URI. It does this by querying number resolutions services, including the local routing table(s) or ENUM server(s) to resolve the R-URI to a SIP URI. In addition, the original R-URI may not include a full E.164 number. As such, you can also configure the Oracle USM to perform a number normalization procedure and ensure it presents a full E.164 number for resolution. Upon successful resolution, the Oracle USM proceeds with ensuing signaling procedures.

To configure the Oracle USM to perform these lookups, you create applicable local-routing-config or enum-config elements and set an option within the sip-registrar that specifies a primary and, optionally, a secondary local-routing-config or enum-config that the sip-registrar uses for LRT or ENUM lookups. If there is no ENUM configuration on the sip-registrar, the Oracle USM forwards applicable requests to a border gateway function via local policy.

Refer to the Net-Net SD ACLI Configuration guide, Session Routing and Load Balancing chapter for complete information on how to configure a local-routing-config and an enum-config.

Number Lookup Triggers

Use cases that are applicable to number lookups and the associated Oracle USM procedures include:

- Request from the access side:
 1. The Oracle USM performs originating services.
 2. If the R-URI is a tel-URI or SIP-URI (with user=phone), it requests e.164 resolution from the ENUM server(s), regardless of its presence in the registration cache.
- Request from core side including request for originating services:
 1. The Oracle USM performs originating services.

2. If the R-URI is a tel-URI or SIP-URI (with user=phone), it requests e.164 resolution from the ENUM server(s), regardless of its presence in the registration cache.
- Request from core side, for terminating services only:
 1. If the R-URI is a tel-URI or SIP-URI (with user=phone) and is not in the Oracle USM cache, it performs an LIR.
 2. If the LIA reply indicates the tel-URI or SIP-URI (with user=phone) is not provisioned, the Oracle USM requests e.164 resolution from the ENUM server(s).

Actions Based on Lookup Results

The Oracle USM forwards to the resultant SIP-URI under the following conditions:

- The SIP-URI is in the Oracle USM cache, in which case the Oracle USM performs terminating services.
- The SIP-URI is not in the Oracle USM cache, and the Oracle USM is configured to service the returned domain. In this case, the Oracle USM performs the following:
 1. The Oracle USM issues an LIR for the SIP-URI.
 2. The Oracle USM forwards the message to the correct S-CSCF.
- The SIP-URI is not in the Oracle USM cache, and the Oracle USM is not configured to service the returned domain. In this case, the Oracle USM performs refers to local policy to forward the message via local policy.

PSTN Breakout Routing

The Oracle USM complies with RFC 4694 for operation with request-URIs that include carrier identification code/route number/number portability database dip indicator (cic/rn/npdi) information and routes those requests according to the rn information. The routing process includes utilization of local policy configured to break the request out of the home network via gateways such as a BGCF.

The Oracle USM does not validate any rn or cic information. Instead, it simply routes the request. Note that the Oracle USM uses cic information instead of rn if both are present in the request. RFC 4694 compliant circumstances under which the Oracle USM does not use rn, cic and npdi information include:

- Invalid routing information, including rn present, but npdi missing.
- Invalid routing information, including npdi present, but rn missing.
- Request uses a sip-URI presented without user=phone.

If the request includes originating services as well as cic/rn/npdi information, the Oracle USM performs those services rather than break out. If, after completing originating services, the request still includes cic/rn/npdi information, the system performs this breakout.

Primary and Secondary ENUM Configuration

For the purpose of redundancy, the Oracle USM allows you to configure these number lookups to use a backup resource in case the lookup at the primary fails. Such scenarios include losing contact with the primary ENUM/LRT server config (query time-out) and the entry is not found at the primary (LRT or ENUM).

To apply primary and secondary number lookup resources to a sip-registrar:

1. From superuser mode, use the following command sequence to access the sip-registrar element and select the registrar you want to configure.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)# select
```

2. Specify the resources to use with the options command.

Prepend the option with the + character if you have multiple options configured that you want to retain. Running the command without the + character causes the system to disable any previously configured options.

To specify primary and secondary ENUM servers:

```
ORACLE(sip-registrar) # options +e164-primary-config=enum:<enum-config name>
ORACLE(sip-registrar) # options +e164-secondary-config=enum:<enum-config
name>
ORACLE(sip-registrar) # done
```

To specify primary and secondary LRT resources:

```
ORACLE(sip-registrar) # options +e164-primary-config=lrt:<lrt-config name>
ORACLE(sip-registrar) # options +e164-secondary-config=lrt:<lrt-config name>
ORACLE(sip-registrar) # done
```

Bear in mind that an enum-config can reference multiple servers. When the Oracle USM references an enum-config, queries follow the normal enum-config sequence, checking each referenced server in order. If the lookup is not successful at the primary, the Oracle USM checks the servers in the registrar's e164-secondary-config.

In addition, each enum-config may refer to a different top-level-domain. This allows you to configure the Oracle USM to successfully perform lookups within two domains.

HSS Initiated User Profile Changes

The Oracle USM can receive Push Profile Request (PPR) messages from an HSS and update the state of the IMS User Profile and associated subscription information it has cached locally. The SIP digest authentication information can also be updated and reassociated with an AoR in case that has changed too. The Oracle USM expects to receive the following AVPs in a PPR message.

- Private-User-Identity—the username, whose subscription/authentication data has changed.
- SIP-Auth-Data-Item—if present new authentication data is included here.
- User-Data—if present new User data is included here.
- Charging-Information—if present new charging information is included here.

The Oracle USM replies to an HSS's PPR in a PPA message with the following AVPs:

- Result-Code—indicates Diameter base protocol error. Valid errors for in a PPA are:
 - DIAMETER_SUCCESS—The request succeeded.
 - DIAMETER_ERROR_NOT_SUPPORTED_USER_DATA—The request failed. The Oracle USM informs HSS that the received user information contained information, which was not recognized or supported.
 - DIAMETER_ERROR_USER_UNKNOWN—The request failed because the Private Identity is not found in Oracle USM.
 - DIAMETER_ERROR_TOO_MUCH_DATA—The request failed. The Oracle USM informs to the HSS that it tried to push too much data into the Oracle USM.
 - DIAMETER_UNABLE_TO_COMPLY—The request failed.
- Experimental-Result—indicates diameter application (3GPP/Cx) error if present.

Licensing and Database Registration Limits

The Oracle USM limits the number of unexpired registration cache entries globally. The total number of system registrations is configured with the registration cache limit parameter in the sip config configuration element.

The Oracle USM also limits the number of registration cache entries that were obtained from a User Subscriber Database; only REGISTERS that prompted the database query are counted here. As User Subscriber Database entries are added and removed, this counter is updated accordingly. Note that it is the actual number of SD-contacts that count against the license limit. Discrete database registration license values range from 20,000 through 500,000 in increments of 20,000.

When a registering contact is rejected because it will exceed one of these limits, the Oracle USM sends a 503 message to the registering endpoint.

Refer to the Getting Started chapter for information about install license management.

Database Registration Limit Alarm

By default, a major alarm is enabled when 98% or more of the licensed number of Database Registrations are used. This alarm is cleared when the number of database registrations falls below 90%. You can configure minor and critical alarms when crossing configured thresholds and you can also reassign the major alarm. This is configured in by creating a **system-config > alarm-threshold** sub element with type of database-registration.

3GPP Compliance

P-Asserted-Id in Requests and Dialogs

When an AoR is successfully registered through the Oracle USM, the list of implicitly registered public IDs is returned from the HSS. The set of implicitly registered public IDs includes the explicitly registered Public-id and may include wild-carded public-ids. If there are no implicitly registered public-ids, then the implicit set returned by HSS will at least contain the explicitly registered public-id.

Based on local configuration and network conditions, the registering UE may or may not be trusted.

Non-trusted UE

When a non-trusted UE sends an initial request for a dialog or a request for a standalone transaction to the Oracle USM, the P-Asserted-Identity to be inserted in the outgoing request is formed as follows:

- If the request does not have a P-Preferred-Identity header field or none of the P-Preferred-Identity header fields included in the request match any of the registered public user identities, then the Oracle USM inserts the default Public-Identity in the outgoing P-Asserted-Identity header.
- If the request includes one or more P-Preferred-Identity header fields which match the registered public user identities, then the Oracle USM includes only the first P-Preferred-Identity header field.
- If the request includes one or more P-Asserted-Identity header fields which do not match the registered public user identities, then the Oracle USM inserts the default Public-Identity in the outgoing P-Asserted-Identity header.

Trusted UE

When a trusted UE sends an initial request for a dialog or a request for a standalone transaction to the Oracle USM, the P-Asserted-Identity to be inserted in the outgoing request is formed as follows:

- If the request does not have a P-Preferred-Identity header field or none of the P-Preferred-Identity header fields included in the request match any of the registered public user identities, then the Oracle USM inserts the default Public-Identity in the outgoing P-Asserted-Identity header.
- If the request includes one or more P-Preferred-Identity header fields which match the registered public user identities, then the Oracle USM inserts the first P-Preferred-Identity header field.
- If the request includes one or more P-Asserted-Identity header fields, then the Oracle USM will use the first P-Asserted-Identity header field.
 - The contents of the From header field do not form any part of this decision process.
 - P-Preferred-Identity header fields will always be removed.

The Default Public Identity is the first appearing, non-barred identity in the set of implicitly registered Public User Identities.

To enable this behavior, add the `pai-comply-to-3gpp` option in the sip config configuration element.

P-Associated-URI in 200 OK

In a 200 OK response to a UE on a successful registration, the Oracle USM includes a P-Associated-URI header. This header includes the list of registered, distinct public user identities and the associated set of implicitly registered distinct public user identities.

When there are no associated implicit public identities, only the explicitly registered Public User Identity is included.

Other Diameter Cx Configuration

Host and Realm AVP Configuration for Cx

You can configure the values sent in the origin-host, origin-realm and destination-host AVPs when the Oracle USM communicates with a server over the Cx interface. Configure destination-host when you want to precisely specify the HSS with which these Cx exchanges take place.

The applicable configuration parameters are located in the home-subscriber-server configuration element. The parameters used to configured the AVPs are origin-realm, origin-host-identifier and destination-host-identifier. The AVPs are constructed as follows:

```
Origin Host AVP = <origin-host-identifier>.<origin-realm>
Origin Realm AVP = <origin-realm>
Destination Host AVP = <destination-host-identifier>.<destination-realm>
```

If the origin-realm is not configured, then the realm parameter in the home-subscriber-server configuration element will be used as the default. If origin-host-identifier is not configured, then the name parameter in the home-subscriber-server configuration element will be used as the default.

If these parameters are not configured, then the AVPs are constructed as follows:

```
Origin Host = <HSS Config name>.<HSS Config realm>.com
Origin Realm AVP = <HSS Config realm>
Destination Host = <HSS Config name>.<HSS Config realm>.com
```

ACLI Instructions

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type home-subscriber-server and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# home-subscriber-server
ORACLE(home-subscriber-server)#
```

4. origin-realm—Set this to a string for use in constructing unique Origin Host and Origin Realm AVPs.
5. origin-host-identifier—Set this to a string for use in constructing a unique Origin Host AVP.
6. destination-host-identifier—Set this to a string for use in constructing a unique Destination Host AVP.
7. Save your work.

Initial Filter Criteria (IFC)

The Oracle USM, acting as a S-CSCF, downloads a set of rules known as Initial Filter Criteria (IFC) from the HSS/AS. IFCs are downloaded over the Cx interface.

iFCs are a way for an S-CSCF to evaluate which ASs should be involved in the call flow for a given user agent (UA). iFCs are functionally defined by Boolean statements, whose component parts are expressed in XML; they reference the destination AS(s) where a desired service is provided.

IFC Evaluation

IFCs are evaluated as described in 3GPP TS 29.228. The Oracle USM supports all tags found in the 3GPP initial filter criteria specifications. An IFC is evaluated until its end, after which the call flow continues as expected.

SIP Registration

When the Oracle USM receives an authenticated REGISTER request from a served UA, it sends an SAR request to the HSS to obtain an SAA which includes iFCs associated with the UE’s subscription. Within the context of registration, the Oracle USM also manages third party registration procedures in conjunction with iFC exchanges or manually via the ACLI. These procedures are described in the Third Party Registration chapter.

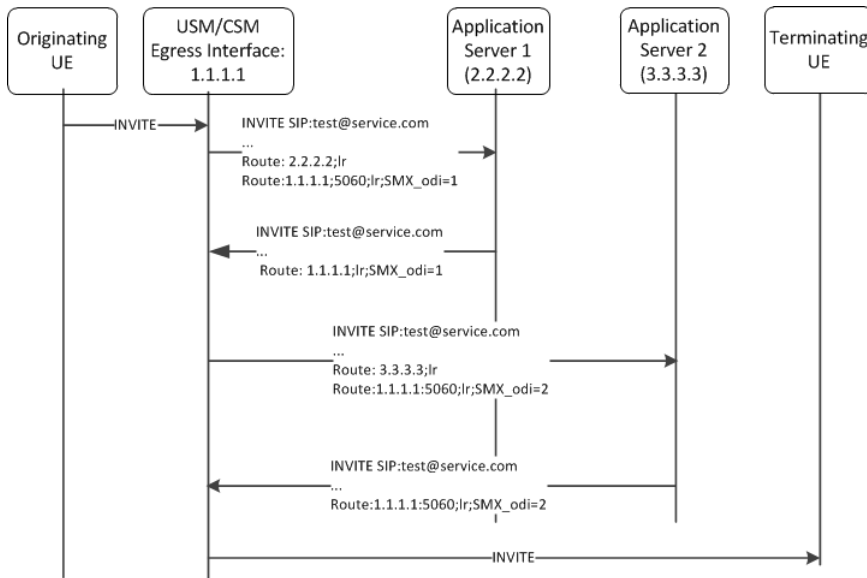
SIP Call

The Oracle USM evaluates all iFC logic to determine that messages with matching characteristics are sent to the proper AS specified in the iFC evaluation using the IP Multimedia Service Control (ISC) interface. In this INVITE, the Oracle USM adds two Route headers. The first (top) route header contains the target AS’s URI. The second Route parameter is built using the IP address of the egress SIP interface and contains the ODI as a parameter. For example:

```
INVITE SIP:test@service.com
...
Route:2.2.2.2;lr
Route:1.1.1.1:5060;lr;smx_odi=1
```

If the AS routes the call back to the Oracle USM, it is expected to include the ODI parameter that it received from the Oracle USM, unchanged. The presence of the ODI parameter indicates that iFC evaluation needs to continue from where it left off for this call. If this continuation of iFC evaluation results in another AS URI, the Oracle USM initiates a request towards that AS this time with a new ODI. In this way, the ODI is a state-signifier of Service Point Triggers.

The process continues until iFC evaluation is completed. Below is an example of an iFC evaluation completing after two iterations.



The iFC continues to be evaluated completely which may result in the INVITE being forwarded to additional ASs. At the conclusion of evaluating the iFC, the Oracle USM checks if the target of the initial request is registered to itself, or not. If the UA is not registered locally the Oracle USM forwards the request by regular means into the network. If the target UA is registered locally, the Oracle USM proceeds to obtain iFCs for the target and begin iFC evaluation for the terminating side of the call.

Evaluating Session Case in the P-Served-User Header

The P-served-user header field conveys the identity of the served user, the session case that applies to the particular communication session, and application invocation, as defined in RFC 5502 and TS 24.229. The Session Case (sescase) and Registration State (regstate) parameters are either populated by the UA originating the message or by the Oracle USM after it determines if a request is originating or terminating, and registered or unregistered

The P-served-user header is created and added to an outgoing request if the next hop is trusted. A trusted next hop is an entity defined by a session agent whose trust-me parameter is enabled. Likewise, the P-served-user header is stripped if the request is forwarded to a next hop that is not known to be trusted.

When the Oracle USM creates a P-served-User header, the user value in the originating case is the user value found in the P-Asserted-Identity header field. In the terminating case, the user value is taken from the Request URI.

Supported Sessioncase and Registration State

The following cases are supported for IFC evaluation. Conditions for classifying the calls as such are listed below.

Originating request by a UA, Registered User

When the Oracle USM receives an Initial request, it is validated as an originating request from a registered user when the following conditions are met:

- The request is a dialog creating request or a standalone request.
- There is no "odi" parameter in the top route of the request.
- The regstate and sescase parameters of the P-served-user indicate for this to be treated as originating request for a registered user OR "The request is received from a registered contact.

Originating request by a UA, Unregistered User

When the Oracle USM receives an Initial request, it is validated as an originating request from an unregistered user when the following conditions are met:

- The request is a dialog creating request or a standalone request.
- There is no "orig" parameter in the top route of the request.
- The served user is unregistered.
- The request is from an AS or I-CSCF and the top route header contains the orig parameter OR The regstate and sescase of the P-served-user header indicates that the request is an originating request for an unregistered user.

Terminating Requests to a UA, Registered User

When the Oracle USM receives an Initial request, it is validated as a terminating request towards a registered user when the following conditions are met:

- The request is a dialog creating request or a standalone request.
- There is no "orig" parameter in the top route of the request.
- There is no "odi" parameter in the top route of the request.
- The regstate and sescase parameters of the P-served-user indicate for this to be treated as terminating request for a registered user OR the request is finished with originating services if applicable and the request is destined to a user who is currently registered with the Oracle USM.
- If the Request-URI changes when visiting an application server, the Oracle USM terminates the checking of filter criteria and routes the request based on the changed value of the Request-URI, per 3GPP Specification TS 23.218.

Terminating Requests to a UA, Unregistered User

See the IFC Support for Unregistered Users section for this case.

- If the Request-URI changes when visiting an application server, the Oracle USM terminates the checking of filter criteria and routes the request based on the changed value of the Request-URI, per 3GPP Specification TS 23.218.

Additional Options

- The Oracle USM can populate the top Route: header with the sescase value for ASs that require it. In such a case, the parameter is created as either call=orig or call=term. This behavior is enabled by configuring the add-sescase-to-route option in the ifc-profile.
- When the dialog-transparency parameter in the sip-config is set to enabled and your network includes multiple ASs, you should add the dialog-transparency-support option in the ifc-profile.

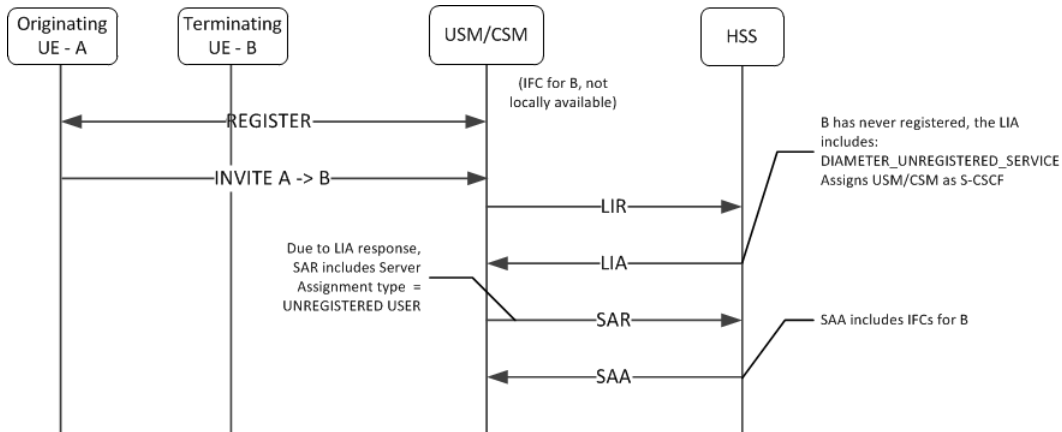
IFC Support for Unregistered Users

The Oracle USM can download Initial Filter Criteria (IFC) from the HSS for unregistered users. This section displays applicable message sequence diagrams.

UE-terminating requests to an unregistered user

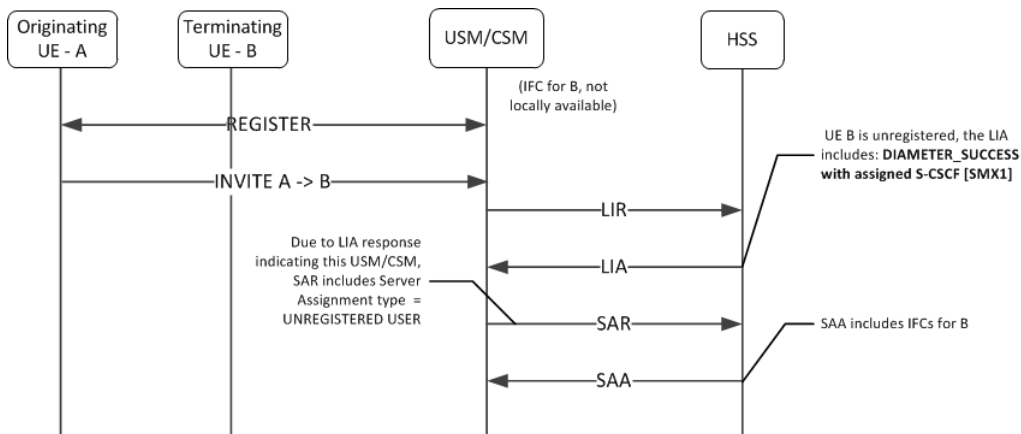
The Oracle USM downloads and executes IFCs for the terminating end of calls. The following call flows indicate possible cases for the terminating unregistered user.

Terminating UA - Not Registered

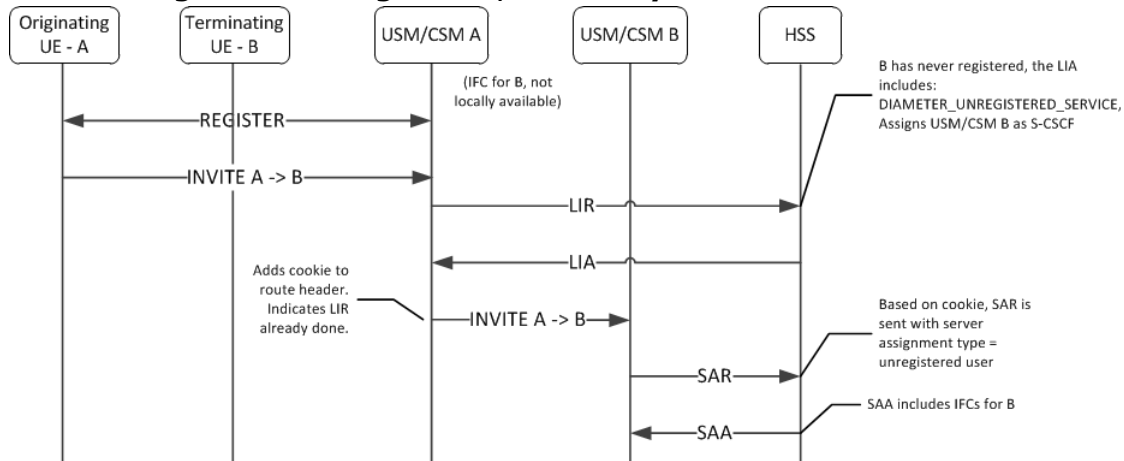


Terminating UA - Not registered

UE originally registered as a consequence of an originating or terminating request or an S-CSCF has stored the user profile.



Terminating UA - Not Registered, Served by other Oracle USM



UE Subsequent Registration

If the Oracle USM has a cached IFC downloaded for an unregistered UA who later registers to that Oracle USM, the cached IFC will be cleared and updated with the IFC downloaded by the registration process.

Caching the Downloaded IFC

When the Oracle USM downloads IFCs for unregistered users, they are saved to a local cache. If the IFC cache fills up, an older cached IFC for a user is released.

Optimizing IFC Updates

The Oracle USM aims to reduce the number of IFC updates traversing the network to save bandwidth and transactional overhead. Unless the unregistered UE's IFC entry has been deleted because of exhausting cache space, the following optimizations are performed:

- If IFCs are available locally, then an SAR/SAA operation to download IFCs will not be performed.
- If a previous IFC download operation did not return any IFCs, then subsequent calls to that unregistered user will not invoke the SAR/SAA messaging to download IFCs.

Push Profile Request (PPR) updates

The HSS can push service profile updates for private IDs. The Oracle USM can process PPR updates for unregistered entities. If the user entry has been deleted because IFC cache space has been exhausted, the PPRs will not be processed.

ACLI Instructions

SIP Registrar

To create an IFC Profile:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type ifc-profile and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# ifc-profile
ORACLE(ifc-profile)#
```

4. name—Enter a name for this IFC profile.
5. state—Set this to enabled to use this ifc-profile.
6. options—Set the options parameter by typing options, a Space, the option name with a plus sign in front of it, and then press Enter.

If you type the option without the plus sign, you will overwrite any previously configured options. In order to append the new options to the options list, you must prepend the new option with a plus sign.

The options included in this section are: add-sescase-to-route and dialog-transparency-support.

7. Type done when finished.

SIP Registrar

To enable IFC support in a SIP Registrar:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type sip-registrar and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)#
```

4. Type select and choose the number of the pre-configured SIP registrar configuration element you want to configure.

```
ORACLE(sip-registrar)# select
name:
1: registrar1
selection:1
ORACLE(sip-registrar)#
```

5. ifc-profile—Set this parameter to the name of the IFC profile you just created.
6. serving-function—Set this parameter to disabled when you want the Oracle USM to act solely as an I-CSCF. When disabled, the Oracle USM always forwards requests from unregistered users to the serving group. The default is enabled, which retains the S-CSCF function on this Oracle USM.
7. serving-group—Set this parameter to a Session Agent Group (SAG) name. The Oracle USM forwards requests from unregistered users to this group when the serving function parameter is disabled. Use of this parameter requires the prior configuration of a SAG that includes all prospective S-CSCFs. The name you give to that group is the name you specify as an argument to this parameter.
8. Type done when finished.

Shared and Default iFCs

The Oracle USM supports Shared iFCs (SiFC), as defined by TS 29.229 and Default iFCs, which are an Oracle extension upon SiFCs. SiFCs provide an operator with the ability to create iFC templates and apply them to a large number of UEs. The SiFC process optimizes the provisioning, storage and transfer of service profile information. The default IFC (DiFC) establishes a configuration wherein the iFC associations are available on the Oracle USM itself. This establishes a backup scenario in case the HSS is not responsive.

To support the SiFC feature on the Oracle USM, you create a profile that refers to a local, XML-formatted file. This file specifies the iFCs to be shared. You apply these profiles to registrars to specify where they are used.

When an SiFC configuration is in place, the Oracle USM notifies the HSS that it supports SiFCs within the Supported-Features AVP in the SAR. The HSS replies to confirm that it supports SiFCs within the SAA. The SiFC feature must be enabled on the HSS.

Note that the form and function of the SiFC and DiFC files are compatible. You can use the same file for both SiFC and DiFC configuration, if desired.

SiFC Usage

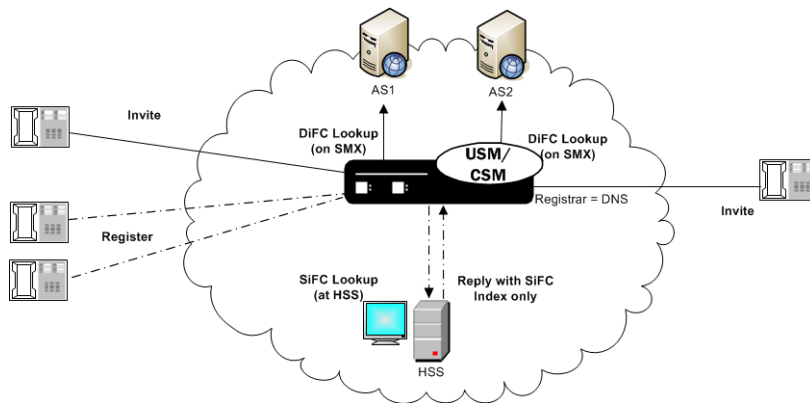
When an applicable end station registers, the Oracle USM forwards the registration to the HSS normally. Given SiFC configuration however, the HSS sends a service-profile containing the SiFC identifiers to the Oracle USM rather than the entire service definition. The Oracle USM parses these identifiers and maps the user to the locally stored filter criteria.

The <IFCSet id="x"> tags in the XML file on the Oracle USM map to the HSS identifiers.

DiFC Usage

In contrast to SiFCs, the Oracle USM fires DiFCs within the context of a session. During the session, the Oracle USM associates the iFCs within the DiFC file with the user, as needed. DiFC usage is invoked during session initiation.

Note that DiFCs are database agnostic. You can use DiFCs for HSS, ENUM and local database configurations. An operational overview of SiFCs and DiFCs is shown below.



SiFC/DiFC File Example

An example of a Oracle USM local SiFC/DiFC XML file, including a single iFC Set containing a single iFC, is presented below.

```
<?xml version="1.0" encoding="UTF-8"?>
<IFCsets>
  <IFCSet id="0">
    <InitialFilterCriteria>
      <Priority>0</Priority>
      <TriggerPoint>
        <ConditionTypeCNF>0</ConditionTypeCNF>
        <SPT>
          <ConditionNegated>0</ConditionNegated>
          <Group>0</Group>
          <Method>INVITE</Method>
          <Extension></Extension>
        </SPT>
      </TriggerPoint>
    <ApplicationServer>
      <ServerName>sip:172.16.101.26:5060</ServerName>
      <DefaultHandling>0</DefaultHandling>
    </ApplicationServer>
    <ProfilePartIndicator>0</ProfilePartIndicator>
  </IFCSet>
</IFCsets>
```

```
</InitialFilterCriteria>  
</IFCSet>  
</IFCsets>
```

Note that the Shared IFCSet contains the integer value property (id="0") that associates these filter criteria with users registered with the Oracle USM. In the case of SiFC, it is the value that the HSS should send when referencing shared sets. In the case of DiFC, the integer is meaningless. The Oracle USM loads and executes default iFCs in the order they appear within the XML file.

iFC Execution Order

Within the context of the 3GPP standards, the Oracle USM evaluates explicitly downloaded iFCs first when determining where to look for a service. If the Oracle USM cannot execute on the service based on explicitly downloaded iFCs, it refers to the SiFC, then the DiFC information to identify an AS that supports the service.

Refreshing SiFC and DiFC Files

Given the nature of local file usage, an ACLI command is available to allow the user to refresh SiFC and DiFC contexts in memory after the user has saved changes to the SiFC and DiFC files. Run the following command to deploy these changes:

```
ORACLE# refresh ifc <ifc-profile name>
```

Note also that the Oracle USM validates the SiFC and DiFC files whenever you Activate your configuration.

SiFC and DiFC Configuration

To configure the Oracle USM to use Shared and Default IFCs:

1. From superuser mode, use the following command sequence to access ifc-profile element.

```
ORACLE# configure terminal  
ORACLE(configure)# session-router  
ORACLE(session-router)# ifc-profile
```

2. Define your profile.

3. name—Enter a name for this IFC profile.

```
ORACLE(ifc-profile)# name acmeTelecomIFC
```

4. state—Set this to enabled to use this ifc-profile.

```
ORACLE(ifc-profile)# state enabled
```

5. default-ifc-filename—Specify filename and, if not stored in the default directory /code/ifc, the applicable pathname.

```
ORACLE(ifc-profile)# default-ifc-filename Afile.xml.gz
```

6. shared-ifc-filename—Specify filename and, if not stored in the default directory /code/ifc, the applicable pathname.

```
ORACLE(ifc-profile)# shared-ifc-filename Bfile.xml.gz
```

7. options—Set the options parameter by typing options, a Space, the option name with a plus sign in front of it, and then press Enter.

```
ORACLE(ifc-profile)# done
```

8. Apply the ifc-profile to your sip registrar.

```
ORACLE# configure terminal  
ORACLE(configure)# session-router  
ORACLE(session-router)# sip-registrar
```

Select the registrar you want to configure and apply the profile.

```
ORACLE(sip-registrar)# select  
ORACLE(sip-registrar)# ifc-profile acmeTelecomIFC  
ORACLE(sip-registrar)# done
```

Distinct and Wildcarded Public Service Identity (PSI) Support

The Oracle USM supports the use of distinct Public Service Identity (PSI) and wildcarded PSIs, typically for specifying access to a service. There is no configuration required on the Oracle USM to enable this support.

Administrators use individual PSI entries and/or wildcarded PSIs as service identifiers on an HSS. These identifiers provide the information needed to direct applicable messages to applicable application servers. Distinct PSIs can reside within individual PSI entries; wildcarded PSI entries are managed within iFC lists. Wildcarded PSI support is typically implemented to reduce HSS resource requirements. By configuring a wildcarded PSI, administrators can use a single record within the iFC to manage multiple resources.

A wildcard is composed of an expression that, when used in a user part, provides for access to multiple service resources. The regular expressions themselves are in form of Perl Compatible Extended Regular Expressions (PCRE).

For example, consider the following two service resources:

- sip:chatroom-12@core.com
- sip:chatroom-64@core.com

These two service resources can be represented simultaneously at the HSS using the following syntax:

- sip:chatroom-!.*!@core.com

The Oracle USM caches filter criteria information that uses this wildcard syntax. This avoids the need for SAR/SAA exchanges between the Oracle USM and the HSS every time an entity requests the service. The Oracle USM is equally capable of caching distinct PSIs, which similarly offloads the need for SAR/SAA exchanges during service resource location processes.

For most call flows, the Oracle USM does not evaluate the expression for the purpose of finding a match. Instead, it keeps the syntax provided by the HSS in its cache and provides the wildcarded syntax in the applicable AVP.

To allow the feature, the Oracle USM supports:

- Wildcarded public user identity AVP in the LIA, SAR and SAA
- User Profile AVP in the SAA
- P-Profile-Key across the Mw interface, as defined in RFC 5002

Configuring SIP Ping OPTIONS Support

You can configure the Oracle USM to respond to SIP ping OPTIONS. This support is typically configured on an S-CSCF so it can respond to pings OPTIONS sent by a P-CSCF:

To configure an SIP Options Ping response support:

1. From superuser mode, use the following command sequence to access ping-response command on a sip-interface element.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-interface
ORACLE(sip-interface)# sel
```

2. Enable the support with the ping-response command.

```
ORACLE(http-config)# ping-response enabled
ORACLE(http-config)# done
```

ping-response—Enable ping-response to allow your device to respond to ping OPTIONS. For example, this feature is useful within hybrid deployment environments on a P-CSCF as a means of verifying the S-CSCF's availability. This configuration allows the S-CSCF to respond to SIP ping OPTIONS.

Redundancy and Load Balancing with HSS Servers

The Oracle USM allows you to operate with multiple HSS servers, supporting:

- Redundancy - Continue normal operation despite HSS failure.
- Load Balancing - Divide the traffic load between HSS servers in a group of HSSs. Preference is based on the HSS list order configured on the Oracle USM.

You configure HSS servers within HSS Groups to support this functionality. For redundancy, you create and assign HSS groups, and apply either the hunt or fail-over strategy to your HSS group. To implement load balancing, you configure the applicable HSS group with a the round-robin server allocation strategy. This functionality assumes the HSS infrastructure itself is configured for redundancy.

The Oracle USM establishes and manages multiple Cx connections with each applicable HSS. This management is achieved by connection identifiers on the Oracle USM that allow it to distinguish between connections. This provides the network with the flexibility of being able to use multiple paths to a given HSS regardless of AVP values.

About HSS Groups

You configure HSS groups based on your redundancy and failover design. You accomplish this by configuring your HSS groups with the applicable HSS servers. You then assign your group to a registrar. HSS group configuration does not preclude assigning an HSS in the group to a registrar individually.

HSS groups can contain individual HSSs. Members of an HSS group are prioritized by the server list; the first server in the list takes the highest priority; the last takes the lowest. You can manually disable an HSS group, if desired, which prevents the Oracle USM from attempting to access any of the HSS servers via that group.

HSS group members do not need to reside in the same domain, network, or realm. The Oracle USM can allocate traffic among member HSSs regardless of their location. It uses the allocation strategies you configure for the group to distribute traffic across the group members.

Group allocation strategies define how the Oracle USM selects an HSS. For example, the hunt strategy selects HSSs in the order in which they are listed. Allocation strategies include the following:

Allocation Strategy	Description
failover	For HSS redundancy deployments, the failover strategy specifies that the Oracle USM selects the next highest priority HSS server for all operations if the first HSS fails. The Oracle USM does not resume operation with the initial HSS when it comes back into service.
hunt	For HSS redundancy deployments, the hunt strategy specifies that the Oracle USM select HSSs in the order in which they are configured in the HSS group. If the first HSS is available, all traffic is sent to the first HSS. If the first HSS is unavailable, all traffic is sent to the second HSS. The system follows this process for all HSS servers in the group. When a higher priority HSS returns to service, all traffic is routed back to it.
roundrobin	This strategy targets HSS load balancing deployments. The Oracle USM selects each HSS in the order in which it appears in the group list, routing diameter requests to each HSS in turn.

Paths taken by specific messaging is constrained by the purpose of that messaging, and refined by a group’s allocation strategy. Applicable messaging includes UAR/UAA, MAR/MAA, SAR/SAA and LIR/LIA. For both failover and hunt strategies, all messaging is sent to the current active server. For the round-robin strategy, messaging is distributed to group members sequentially, using the member list order.

Connection Failure Detection

The Oracle USM detects that a connection between itself and a given HSS has failed if either a diameter request fails or the diameter DWR/DWA handshake fails. If the HSS does not respond to five requests, the Oracle USM marks that HSS as out of service.

The Oracle USM forwards unacknowledged messages to subsequent HSSs based on strategy. It changes the destination host AVP of these messages and marks them with the T flag. The HSS recognizes the T flag as an indication that the request may be a duplicate, caused by a problem in the network.

Periodically, the Oracle USM attempts to establish diameter connections with out of service HSS servers. When those connections succeed, the Oracle USM marks the HSS as in-service and resumes using it within the context of the configured redundancy and load balancing strategy.

Configuring HSS Groups

To configure HSS groups:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the system-level configuration elements.

```
ORACLE (configure) # session-router
```

3. Type hss-group and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE (session-router) # hss-group
ORACLE (hss-group) #
```

4. name—Enter a unique name for the HSS group in Name format.
5. state—Enable or disable the HSS group on the Oracle USM. The default value is enabled. Valid values are:
 - enabled | disabled
6. origin-host-identifier— Set this to a string for use in constructing a unique Origin Host AVP. This setting always takes precedence over the origin-host-identifier configured for the home-subscriber-server. This setting is required.
7. strategy—Indicate the HSS server allocation strategy you want to use. The strategy you chose selects the HSS servers that will be made available by this hss-group. The default value is hunt. The valid values are:
 - hunt—Selects HSS servers in the order in which they are listed. For example, if the first server is online, all traffic is sent to the first server. If the first server is offline, the second server is selected. If the first and second servers are offline, the third server is selected. When the Oracle USM detects that a higher priority HSS is back in service, it routes all subsequent traffic to that HSS.
 - roundrobin—Selects each HSS server in the order in which they are listed in the destination list, selecting each server in turn, one per session.
 - failover — Selects the first server in the list until failure is detected. Subsequent signaling goes to the next server in the list.
8. hss-configs—Identify the HSS servers available for use by this hss-group. This list can contain as many HSS servers as is necessary. An hss-config list value must correspond to a valid hss-config.

Display syntax for the hss-configs parameter by typing the question mark character after the parameter name on the ACLI.

```
ORACLE (hss-group) # hss-configs ?
<string> list of home-subscriber-server configs for this group
for single-entry: hss1
for multi-entry: (hss1 hss2)
for adding an entry to an existing list: +hss3
for deleting an entry from an existing list: -hss3
for multiple entries add/remove from a list: +/- (hss1 hss2)
```

The following example shows an HSS group using the hunt allocation strategy applied.

```
hss-group
  name          group-test1
  state         enabled
  origin-host-identifier
  strategy      hunt
  hss-configs   hss1, hss2
  last-modified-by admin@console
  last-modified-date 2013-05-13 14:58:01
```

Stream Control Transfer Protocol Overview

The Oracle USM supports the use of the Stream Control Transmission Protocol (SCTP) as transport protocol for connections with an HSS (Cx interface). This section explains SCTP and provides instructions for SCTP configuration on an Oracle USM home-subscriber-server configuration element.

SCTP was originally designed by the Signaling Transport (SIGTRAN) group of IETF for Signaling System 7 (SS7) transport over IP-based networks. It is a reliable transport protocol operating on top of an unreliable connectionless service, such as IP. It provides acknowledged, error-free, non-duplicated transfer of messages through the use of checksums, sequence numbers, and selective retransmission mechanism.

SCTP is designed to allow applications, represented as endpoints, communicate in a reliable manner, and so is similar to TCP. In fact, it has inherited much of its behavior from TCP, such as association (an SCTP peer-to-peer connection) setup, congestion control and packet-loss detection algorithms. Data delivery, however, is significantly different. SCTP delivers discrete application messages within multiple logical streams within the context of a single association. This approach to data delivery is more flexible than the single byte-stream used by TCP, as messages can be ordered, unordered or even unreliable within the same association.

Support is compliant with RFC 4960, Stream Control Transmission Protocol.

SCTP Packets

SCTP packets consist of a common header and one or more chunks, each of which serves a specific purpose.

- DATA chunk — carries user data
- INIT chunk — initiates an association between SCTP endpoints
- INIT ACK chunk — acknowledges association establishment
- SACK chunk — acknowledges received DATA chunks and informs the peer endpoint of gaps in the received subsequences of DATA chunks
- HEARTBEAT chunk — tests the reachability of an SCTP endpoint
- HEARTBEAT ACK chunk — acknowledges reception of a HEARTBEAT chunk
- ABORT chunk — forces an immediate close of an association
- SHUTDOWN chunk — initiates a graceful close of an association
- SHUTDOWN ACK chunk — acknowledges reception of a SHUTDOWN chunk
- ERROR chunk — reports various error conditions
- COOKIE ECHO chunk — used during the association establishment process
- COOKIE ACK chunk — acknowledges reception of a COOKIE ECHO chunk
- SHUTDOWN COMPLETE chunk — completes a graceful association close

SCTP Terminology

This section defines some terms commonly found in SCTP standards and documentation.

SCTP Association is a connection between SCTP endpoints. An SCTP association is uniquely identified by the transport addresses used by the endpoints in the association. An SCTP association can be represented as a pair of SCTP endpoints, for example, `assoc = { [IPv4Addr : PORT1], [IPv4Addr1, IPv4Addr2: PORT2]}`.

Only one association can be established between any two SCTP endpoints.

SCTP Endpoint is a sender or receiver of SCTP packets. An SCTP endpoint may have one or more IP address but it always has one and only one SCTP port number. An SCTP endpoint can be represented as a list of SCTP transport addresses with the same port, for example, endpoint = [IPv6Addr, IPv6Addr: PORT].

An SCTP endpoint may have multiple associations.

SCTP Path is the route taken by the SCTP packets sent by one SCTP endpoint to a specific destination transport address or its peer SCTP endpoint. Sending to different destination transport addresses does not necessarily guarantee separate routes.

SCTP Primary Path is the default destination source address, the IPv4 or IPv6 address of the association initiator. For retransmissions however, another active path may be selected, if one is available.

SCTP Stream is a unidirectional logical channel established between two associated SCTP endpoints. SCTP distinguishes different streams of messages within one SCTP association. SCTP makes no correlation between an inbound and outbound stream.

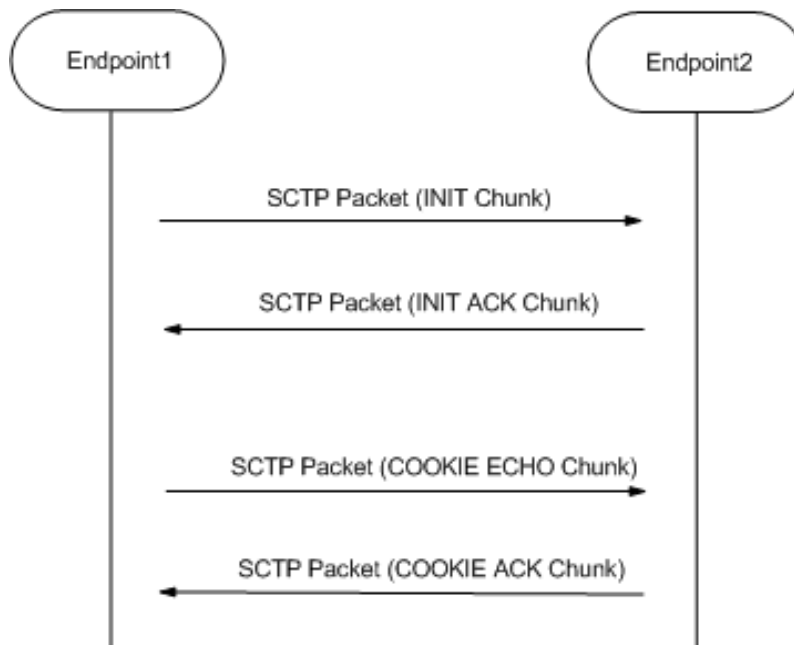
SCTP Transport Address is the combination of an SCTP port and an IP address. For the current release, the IP address portion of an SCTP Transport Address must be a routable, unicast IPv4 or IPv6 address.

An SCTP transport address binds to a single SCTP endpoint.

SCTP Message Flow

Before peer SCTP users (commonly referred to as endpoints) can send data to each other, an association (an SCTP connection) must be established between the endpoints. During the association establishment process a cookie mechanism is employed to provide protection against security attacks. The following figure shows a sample SCTP association establishment message flow.

Endpoint1 initiates the association by sending Endpoint2 an SCTP packet that contains an INIT chunk, which can include one or more IP addresses used by the initiating endpoint. Endpoint2 acknowledges the initiation of an SCTP association with an SCTP packet that contains an INIT_ACK chunk. This chunk can also include one or more IP addresses at used by the responding endpoint.



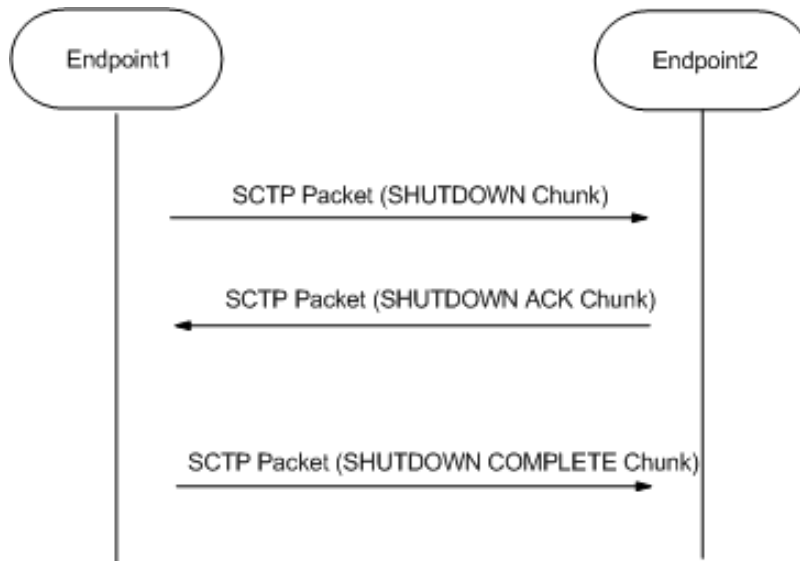
Both the INIT chunk (issued by the initiator) and INIT ACK chunk (issued by the responder) specify the number of outbound streams supported by the association, as well as the maximum inbound streams accepted from the other endpoint.

Association establishment is completed by a COOKIE ECHO/COOKIE ACK exchange that specifies a cookie value used in all subsequent DATA exchanges.

Once an association is successfully established, an SCTP endpoint can send unidirectional data streams using SCTP packets that contain DATA chunks. The recipient endpoint acknowledges with an SCTP packet containing a SACK chunk.

SCTP monitors endpoint reachability by periodically sending SCTP packets that contain HEARTBEAT chunks. The recipient endpoint acknowledges receipt, and confirms availability, with an SCTP packet containing a HEARTBEAT ACK chunk.

Either SCTP endpoint can initiate a graceful association close with an SCTP packet that contains a SHUTDOWN chunk. The recipient endpoint acknowledges with an SCTP packet containing a SHUTDOWN ACK chunk. The initiating endpoint concludes the graceful close with an SCTP packet that contains a SHUTDOWN COMPLETE chunk.

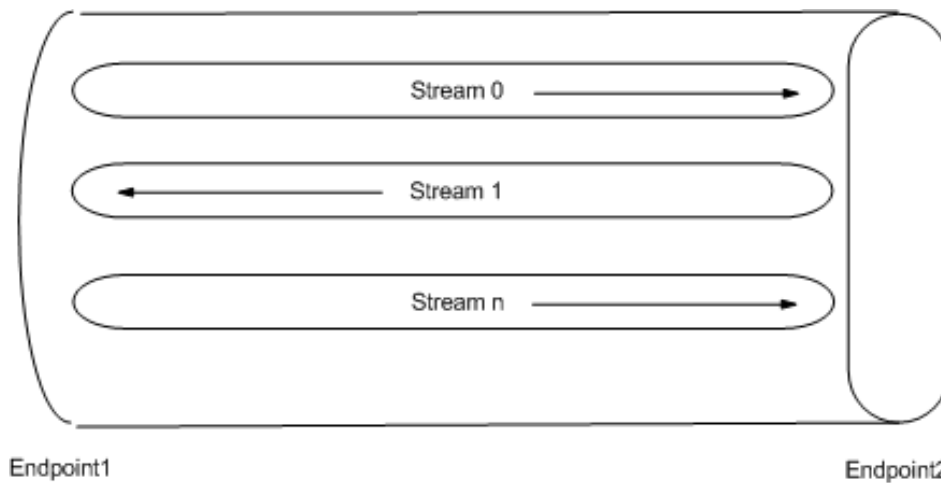


Congestion Control

SCTP congestion control mechanism is similar to that provided by TCP, and includes slow start, congestion avoidance, and fast retransmit. In SCTP, the initial congestion window (cwnd) is set to the double of the maximum transmission unit (MTU) while in TCP, it is usually set to one MTU. In SCTP, cwnd increases based on the number of acknowledged bytes, rather than the number of acknowledgements in TCP. The larger initial cwnd and the more aggressive cwnd adjustment provided by SCTP result in a larger average congestion window and, hence, better throughput performance than TCP.

Multi-Streaming

SCTP supports streams as depicted in the following figure which depicts an SCTP association that supports three streams,



The multiple stream mechanism is designed to solve the head-of-the-line blocking problem of TCP. Therefore, messages from different multiplexed flows do not block one another.

A stream can be thought of as a sub-layer between the transport layer and the upper layer. SCTP supports multiple logical streams to improve data transmission throughput. As shown in the above figure, SCTP allows multiple unidirectional streams within an association. This multiplexing/de-multiplexing capability is called multi-streaming and it is achieved by introducing a field called Stream Identifier (contained in every DATA chunk) that is used to differentiate segments in different streams.

Delivery Modes

SCTP supports two delivery modes, ordered and unordered. Delivery mode is specified by the U bit in the DATA chunk header — if the bit is clear (0), ordered delivery is specified; if the bit is set (1), unordered delivery is specified.

Within a stream, an SCTP endpoint must deliver ordered DATA chunks (received with the U bit set to 0) to the upper layer protocol according to the order of their Stream Sequence Number. Like the U bit, the Stream Sequence Number is a field within the DATA chunk header, and serves to identify the chunk's position within the message stream. If DATA chunks arrive out of order of their Stream Sequence Number, the endpoint must delay delivery to the upper layer protocol until they are reordered and complete.

Unordered DATA chunks (received with the U bit set to 1) are processed differently. When an SCTP endpoint receives an unordered DATA chunk, it must bypass the ordering mechanism and immediately deliver the data to the upper layer protocol (after reassembly if the user data is fragmented by the sender). As a consequence, the Stream Sequence Number field in an unordered DATA chunk has no significance. The sender can fill it with arbitrary value, but the receiver must ignore any value in field.

When an endpoint receives a DATA chunk with the U flag set to 1, it must bypass the ordering mechanism and immediately deliver the data to the upper layer (after reassembly if the user data is fragmented by the data sender).

Unordered delivery provides an effective way of transmitting out-of-band data in a given stream. Note also, a stream can be used as an unordered stream by simply setting the U bit to 1 in all DATA chunks sent through that stream.

Multi-Homing

Call control applications for carrier-grade service require highly reliable communication with no single point of failure. SCTP can assist carriers with its multi-homing capabilities. By providing different paths through the network over separate and diverse means, the goal of no single point of failure is more easily attained.

SCTP built-in support for multi-homed hosts allows a single SCTP association to run across multiple links or paths, hence achieving link/path redundancy. With this capability, an SCTP association can be made to achieve fast failover from one link/path to another with little interruption to the data transfer service.

Multi-homing enables an SCTP host to establish an association with another SCTP host over multiple interfaces identified by different IP addresses. With specific regard to the Oracle USM these IP addresses need not be assigned to the same physical interface, or to the same physical Network Interface Unit.

If the SCTP nodes and the according IP network are configured in such a way that traffic from one node to another travels on physically different paths if different destination IP address are used, associations become tolerant against physical network failures and other problems of that kind.

An endpoint can choose an optimal or suitable path towards a multi-homed destination. This capability increases fault tolerance. When one of the paths fails, SCTP can still choose another path to replace the previous one. Data is always sent over the primary path if it is available. If the primary path becomes unreachable, data is migrated to a different, affiliated address — thus providing a level of fault tolerance. Network failures that render one interface of a server unavailable do not necessarily result in service loss. In order to achieve real fault resilient communication between two SCTP endpoints, the maximization of the diversity of the round-trip data paths between the two endpoints is encouraged.

Multi-Homing and Path Diversity

As previously explained, when a peer is multi-homed, SCTP can automatically switch the subsequent data transmission to an alternative address. However, using multi-homed endpoints with SCTP does not automatically guarantee resilient communications. One must also design the intervening network(s) properly.

To achieve fault resilient communication between two SCTP endpoints, one of the keys is to maximize the diversity of the round-trip data paths between the two endpoints. Under an ideal situation, one can make the assumption that every destination address of the peer will result in a different, separate path towards the peer. Whether this can be achieved in practice depends entirely on a combination of factors that include path diversity, multiple connectivity, and the routing protocols that glue the network together. In a normally designed network, the paths may not be diverse, but there may be multiple connectivity between two hosts so that a single link failure will not fail an association.

In an ideal arrangement, if the data transport to one of the destination addresses (which corresponds to one particular path) fails, the data sender can migrate the data traffic to other remaining destination address(es) (that is, other paths) within the SCTP association.

Monitoring Failure Detection and Recovery

When an SCTP association is established, a single destination address is selected as the primary destination address and all new data is sent to that primary address by default. This means that the behavior of a multi-homed SCTP association when there are no network losses is similar to behavior of a TCP connection. Alternate, or secondary, destination addresses are only used for redundancy purposes, either to retransmit lost packets or when the primary destination address cannot be reached.

A failover to an alternate destination is performed when the SCTP sender cannot elicit an acknowledgement — either a SACK for a DATA chunk, or a HEARTBEAT ACK for a HEARTBEAT chunk — for a configurable consecutive number of transmissions. The SCTP sender maintains an error-counter is maintained for each destination address and if this counter exceeds a threshold (normally six), the address is marked as inactive, and taken out of service. If the primary destination address is marked as inactive, all data is then switched to a secondary address to complete the failover.

If no data has been sent to an address for a specified time, that endpoint is considered to be idle and a HEARTBEAT packet is transmitted to it. The endpoint is expected to respond to the HEARTBEAT immediately with a HEARTBEAT ACK. As well as monitoring the status of destination addresses, the HEARTBEAT is used to obtain RTT measurements on idle paths. The primary address becomes active again if it responds to a heartbeat.

The number of events where heartbeats were not acknowledged within a certain time, or retransmission events occurred is counted on a per association basis, and if a certain limit is exceeded, the peer endpoint is considered unreachable, and the association is closed.

The threshold for detecting an endpoint failure and the threshold for detecting a failure of a specific IP addresses of the endpoint are independent of each other. Each parameter can be separately configured by the SCTP user. Careless configuration of these protocol parameters can lead the association onto the dormant state in which all the destination

addresses of the peer are found unreachable while the peer still remains in the reachable state. This is because the overall retransmission counter for the peer is still below the set threshold for detecting the peer failure.

ACLI Instructions for Configuring SCTP for DIAMETER Transport

Use the following steps to configure SCTP as the layer 4 transport for an HSS.

- Create an SCTP-based home-subscriber-server
- Associate network interfaces with existing realms
- Set SCTP timers and counters

Configuring an HSS Server for SCTP

HSS servers are created during the IMS environment configuration process.

1. From superuser mode, use the following command sequence to access home-subscriber-server configuration.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# home-subscriber-server
ORACLE(home-subscriber-server)#
ORACLE(home-subscriber-server)#
```

2. Use the address parameter to provide the IP address of the network interface that supports the Cx port.

This is the primary address of a the local multi-homed SCTP endpoint.

```
ORACLE(home-subscriber-server)# address 172.16.10.76
ORACLE(home-subscriber-server)#
```

3. Retain the default value, 3868 (the well-known DIAMETER port) for the port parameter.

```
ORACLE(home-subscriber-server)# port 3868
ORACLE(home-subscriber-server)#
```

4. Use the transport-protocol parameter to identify the layer 4 protocol.

Supported values are TCP and SCTP.

Select SCTP.

```
ORACLE(home-subscriber-server)# transport-protocol sctp
ORACLE(home-subscriber-server)#
```

5. Use the multi-homed-addr parameter to specify one or more local secondary addresses of the SCTP endpoint.

Multi-homed addresses must be of the same type (IPv4 or IPv6) as that specified by the address parameter. Like the address parameter, these addresses identify SD physical interfaces.

To specify multiple addresses, bracket an address list with parentheses.

```
ORACLE(home-subscriber-server)# multi-homed-addr 182.16.10.76
ORACLE(home-subscriber-server)#
```

To specify multiple addresses, bracket an address list with parentheses.

```
ORACLE(home-subscriber-server)# multi-homed-addr (182.16.10.76
192.16.10.76 196.15.32.108)
ORACLE(home-subscriber-server)#
```

6. Remaining parameters can be safely ignored.

7. Use done, exit, and verify-config to complete configuration of the home-subscriber-server.

```
ORACLE(home-subscriber-server)# done
ORACLE(session-router)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
```

```
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Configuring the Realm

After configuring an HSS server, which identifies primary and secondary multi-homed transport addresses, you list the network interfaces that support these primary and secondary addresses in the realm assigned during **realm-config** configuration.

1. From superuser mode, use the following command sequence to access realm-config configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# media-manager
ORACLE(media-manager)# realm-config
ORACLE(realm-config)#
```

2. Use the select command to access the target realm.
3. Use the network-interfaces command to identify the network interfaces that support the SCTP primary and secondary addresses.

Network interfaces are identified by their name.

Enter a list of network interface names using parentheses as list brackets. The order of interface names is not significant.

```
ORACLE(realm-config)# network-interfaces (m01 m10)
ORACLE(realm-config)#
```

4. Use done, exit, and verify-config to complete realm configuration.

```
ORACLE(realm-config)# done
ORACLE(media-manager)# exit
ORACLE(configure)# exit
ORACLE# verify-config
```

```
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Setting SCTP Timers and Counters

Setting SCTP timers and counters is optional. All configurable timers and counters provide default values and most default to recommended values as specified in RFC 4960, Stream Control Transmission Protocol.

Management of Retransmission Timer, section 6.3 of RFC 4960 describes the calculation of a Retransmission Timeout (RTO) by the SCTP process. This calculation involves three SCTP protocol parameters: RTO.Initial, RTO.Min, and RTO.Max. Suggested SCTP Protocol Parameter Values section 15 of RFC 4960 lists recommended values for these parameters.

The following shows the equivalence of recommended values and ACLI defaults.

RTO.Initial = 3 seconds sctp-rto-initial = 3000 ms (default value)

RTO.Min = 1 second sctp-rto-min = 1000 ms (default value)

RTO.Max = 60 seconds sctp-rto-max = 60000 ms (default value)

Path Heartbeat, section 8.3 of RFC 4960 describes the calculation of a Heartbeat Interval by the SCTP process. This calculation involves the current calculated RTO and a single SCTP protocol parameter — HB.Interval.

The following shows the equivalence of recommended the value and ACLI default.

HB.Interval = 30 seconds sctp-hb-interval = 3000 ms (default value)

Acknowledgement on Reception of DATA Chunks, section 6.2 of RFC 4960 describes requirements for the timely processing and acknowledgement of DATA chunks. This section requires that received DATA chunks must be acknowledged within 500 milliseconds, and recommends that DATA chunks should be acknowledged with 200

milliseconds. The interval between DATA chunk reception and acknowledgement is specific by the ACLI `sctp-sack-timeout` parameter, which provides a default value of 200 milliseconds and a maximum value of 500 milliseconds.

Transmission of DATA Chunks, section 6.1 of RFC 4960 describes requirements for the transmission of DATA chunks. To avoid network congestion the RFC recommends a limitation on the volume of data transmitted at one time. The limitation is expressed in terms of DATA chunks, not in terms of SCTP packets. The maximum number of DATA chunks that can be transmitted at one time is specified by the ACLI `sctp-max-burst` parameter, which provides a default value of 4 chunks, the limit recommended by the RFC.

SCTP Network Parameters are not RTC

SCTP configuration parameters within the `network-parameters` element are not real-time configuration (RTC) supported. Reboot your system for changes to these parameter to take effect.

Specifying the Delivery Mode

As described in Delivery Modes, SCTP support two delivery modes, ordered and unordered.

1. From superuser mode, use the following command sequence to access `network-parameters` configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)# network-parameters
ORACLE(network-parameters)#
```

2. Use the `sctp-send-mode` parameter to select the preferred delivery mode.

Choose ordered or unordered.

```
ORACLE(network-parameters)# sctp-send-mode unordered
ORACLE(network-parameters)#
```

3. Use `done`, `exit`, and `verify-config` to complete delivery mod configuration.

```
ORACLE(network-parameters)# done
ORACLE(system)# exit
ORACLE(configure)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Setting Path Failure Detection

As described in Monitoring, Failure Detection and Recovery, when its peer endpoint is multi-homed, an SCTP endpoint maintains a count for each of the peer's destination transport addresses.

Each time the T3-rtx timer expires on any address, or when a HEARTBEAT sent to an idle address is not acknowledged within an RTO, the count for that specific address is incremented. If the value of a specific address count exceeds the SCTP protocol parameter `Path.Max.Retrans`, the endpoint marks that destination transport address as inactive.

The endpoint resets the counter when (1) a DATA chunk sent to that peer endpoint is acknowledged by a SACK, or (2) a HEARTBEAT ACK is received from the peer endpoint.

When the primary path is marked inactive (due to excessive retransmissions, for instance), the sender can automatically transmit new packets to an alternate destination address if one exists and is active. If more than one alternate address is active when the primary path is marked inactive, a single transport address is chosen and used as the new destination transport address.

Use the following procedure to configure path failure detection.

1. From superuser mode, use the following command sequence to access `network-parameters` configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# system
```

```
ORACLE(system)# network-parameters
ORACLE(network-parameters)#
```

2. Use the `sctp-path-max-retrans` parameter to assign a value to the SCTP protocol parameter `Path.Max.Retrans`.

Allowable values are integers within the range 0 through 4294967295 that specify the maximum number of RTOs and unacknowledged HEARTBEATS. In the absence of an explicitly configured integer value, `sctp-path-max-retrans` defaults to 5 (RTO and/or HEARTBEAT errors per transport address, the recommended default value from RFC 4960).

When configuring endpoint and path failure detection, ensure that the value of the `sctp-assoc-max-retrans` parameter is smaller than the sum of the `sctp-path-max-retrans` values for all the remote peer's destination addresses. Otherwise, all the destination addresses can become inactive (unable to receive traffic) while the endpoint still considers the peer endpoint reachable.

```
ORACLE(network-parameters)# sctp-path-max-retrans 5
ORACLE(network-parameters)#
```

3. Use `done`, `exit`, and `verify-config` to complete path failure detection configuration.

```
ORACLE(network-parameters)# done
ORACLE(system)# exit
ORACLE(configure)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Setting Endpoint Failure Detection

As described in *Monitoring, Failure Detection and Recovery*, a single-homed SCTP endpoint maintains a count of the total number of consecutive failed (unacknowledged) retransmissions to its peer. Likewise, a multi-homed SCTP endpoint maintains a series of similar, dedicated counts for all of its destination transport addresses. If the value of these counts exceeds the limit indicated by the SCTP protocol parameter `Association.Max.Retrans`, the endpoint considers the peer unreachable and stops transmitting any additional data to it, causing the association to enter the CLOSED state.

The endpoint resets the counter when (1) a DATA chunk sent to that peer endpoint is acknowledged by a SACK, or (2) a HEARTBEAT ACK is received from the peer endpoint.

Use the following procedure to configure endpoint failure detection.

1. From superuser mode, use the following command sequence to access `network-parameters` configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)# network-parameters
ORACLE(network-parameters)#
```

2. Use the `sctp-assoc-max-retrans` to assign a value to the SCTP protocol parameter `Association.Max.Retrans`.

Allowable values are integers within the range 0 through 4294967295 which specify the maximum number of transmission requests. In the absence of an explicitly configured integer value, `sctp-assoc-max-retrans` defaults to 10 (transmission re-tries, the recommended default value from RFC 4960).

```
ORACLE(network-parameters)# sctp-assoc-max-retrans 10
ORACLE(network-parameters)#
```

3. Use `done`, `exit`, and `verify-config` to complete endpoint failure detection configuration.

```
ORACLE(network-parameters)# done
ORACLE(system)# exit
ORACLE(configure)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
```

```
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Limiting DATA Bursts

Section 6.1 of RFC 4960 describes the SCTP protocol parameter, Max.Burst, used to limit the number of DATA chunks that are transmitted at one time.

Use the following procedure to assign a value to the SCTP protocol parameter, Max.Burst.

1. From superuser mode, use the following command sequence to access network-parameters configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)# network-parameters
ORACLE(network-parameters)#
```

2. Use the sctp-max-burst parameter to assign a value to Max.Burst.

Allowable values are integers within the range 0 through 4294967295 that specify the maximum number of DATA chunks that will be sent at one time. In the absence of an explicitly configured integer value, sctp-max-burst defaults to 4 (DATA chunks, the recommended default value from RFC 4960).

```
ORACLE(network-parameters)# sctp-max-burst 4
ORACLE(network-parameters)#
```

3. Use done, exit, and verify-config to complete configuration of DATA burst limitations.

```
ORACLE(network-parameters)# done
ORACLE(system)# exit
ORACLE(configure)# exit
ORACLE(configure)# exit
ORACLE# verify-config
```

```
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Setting the SACK Delay Timer

An SCTP Selective Acknowledgement (SACK) is sent to the peer endpoint to acknowledge received DATA chunks and to inform the peer endpoint of gaps in the received subsequences of DATA chunks. Section 6.2 of RFC 4960 sets a specific requirement for a SACK Delay timer that specifies the maximum interval between the reception of an SCTP packet containing one or more DATA chunks and the transmission of a SACK to the packet originator.

Use the following procedure to set the SACK Delay timer.

1. From superuser mode, use the following command sequence to access network-parameters configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)# network-parameters
ORACLE(network-parameters)#
```

2. Use the sctp-sack-timeout parameter to assign a value to the SACK Delay timer.

Allowable values are integers within the range 0 through 500 which specify the maximum delay (in milliseconds) between reception of a SCTP packet containing one or more Data chunks and the transmission of a SACK to the packet source. The value 0 indicates that a SACK is generated immediately upon DATA chunk reception

In the absence of an explicitly configured integer value, sctp-sack-timeout defaults to 200 ms (the recommended default value from RFC 4960).

```
ORACLE(network-parameters)# sctp-sack-timeout 200
ORACLE(network-parameters)#
```

3. Use done, exit, and verify-config to complete configuration of the SACK Delay timer.

```
ORACLE(network-parameters)# done
ORACLE(system)# exit
ORACLE(configure)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Setting the Heartbeat Interval

Both single-homed and multi-homed SCTP endpoints test the reachability of associates by sending periodic HEARTBEAT chunks to UNCONFIRMED or idle transport addresses.

Use the following procedure to assign values used in Heartbeat Interval calculation.

1. From superuser mode, use the following command sequence to access network-parameters configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)# network-parameters
ORACLE(network-parameters)#
```

2. Use the `sctp-hb-interval` parameter to assign an initial Heartbeat Interval duration.

Allowable values are integers within the range 0 through 4294967295 that specify the initial Heartbeat Interval in milliseconds. In the absence of an explicitly configured integer value, `sctp-hb-interval` defaults to 30000 milliseconds (30 seconds, the recommended default value from RFC 4960).

As described in Section 8.3 of RFC 4960, the value specified by `sctp-hb-interval` is assigned to the SCTP protocol parameter `HB.Interval`, which provides a default interval until actual calculations have derived a fluctuating interval based on network usage. The value specified by the `sctp-hb-interval` parameter is used during these calculations.

```
ORACLE(network-parameters)# sctp-hb-interval 30000
ORACLE(network-parameters)#
```

3. Use `done`, `exit`, and `verify-config` to complete Heartbeat Interval configuration.

```
ORACLE(network-parameters)# done
ORACLE(system)# exit
ORACLE(configure)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Setting the RTO

An SCTP endpoint uses a retransmission timer to ensure data delivery in the absence of any feedback from its peer. RFC 4960 refers to the timer as `T3-rtx` and to the timer duration as `RTO` (retransmission timeout).

When an endpoint's peer is multi-homed, the endpoint calculates a separate `RTO` for each IP address affiliated with the peer. The calculation of `RTO` in SCTP is similar to the way TCP calculates its retransmission timer. `RTO` fluctuates over time in response to actual network conditions. To calculate the current `RTO`, an endpoint maintains two state variables per destination IP address — the `SRTT` (smoothed round-trip time) variable, and the `RTTVAR` (round-trip time variation) variable.

Use the following procedure to assign values used in `RTO` calculation.

1. From superuser mode, use the following command sequence to access network-parameters configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# system
```

```
ORACLE(system)# network-parameters
ORACLE(network-parameters)#
```

2. Use the `sctp-rto-initial` parameter to assign an initial timer duration.

Allowable values are integers within the range 0 through 4294967295 that specify the initial duration in milliseconds. In the absence of an explicitly configured integer value, `sctp-rto-initial` defaults to 3000 milliseconds (3 seconds, the recommended default value from RFC 4960).

As described in Section 6.3 of RFC 4960, the value specified by `sctp-rto-initial` is assigned to the SCTP protocol parameter `RTO.Initial`, which provides a default RTO until actual calculations have derived a fluctuating duration based on network usage. The value specified by the `sctp-rto-initial` parameter seeds these calculations.

```
ORACLE(network-parameters)# sctp-rto-initial 3000
ORACLE(network-parameters)#
```

3. Use the `sctp-rto-min` and `sctp-rto-max` parameters to assign an RTO floor and ceiling.

Allowable values are integers within the range 0 through 4294967295 that specify the minimum and maximum durations in milliseconds. In the absence of an explicitly configured integer value, `sctp-rto-min` defaults to 1000 ms

(1 second, the recommended default value from RFC 4960), and `sctp-rto-max` defaults to 60000 ms (60 seconds, the recommended default value from RFC 4960.)

As described in Section 6.3 of RFC 4960, the values specified by `sctp-rto-min` and `sctp-rto-max` are assigned to the SCTP protocol parameters, `RTO.min` and `RTO.max` that limit RTO calculations. If a calculated RTO duration is less than `RTO.min`, the parameter value is used instead of the calculated value; likewise, if a calculated RTO duration is greater than `RTO.max`, the parameter value is used instead of the calculated value.

```
ORACLE(network-parameters)# sctp-rto-min 1000
ORACLE(network-parameters)# sctp-rto-max 60000
ORACLE(network-parameters)#
```

4. Use `done`, `exit`, and `verify-config` to complete RTO configuration.

```
ORACLE(network-parameters)# done
ORACLE(system)# exit
ORACLE(configure)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

ENUM Based Oracle USM

The Oracle USM contacts an ENUM server via DNS for two purposes:

- to obtain authentication data for a registering UA (often referred to as a UE)
- to query the user subscriber database (ENUM) regarding the registration state of the AoR and update the database with the latest information

Message Authentication for SIP Requests

The Oracle USM authenticates requests by configuring the sip authentication profile configuration element. The name of this configuration element is either configured as a parameter in the sip registrar configuration element's authentication profile parameter or in the sip interface configuration element's sip-authentication-profile parameter. This means that the Oracle USM can perform SIP digest authentication either globally, per domain of the Request URI or as received on a SIP interface.

After naming a sip authentication profile, the received methods that trigger digest authentication are configured in the methods parameter. You can also define which anonymous endpoints are subject to authentication based on the request method they send to the Oracle USM by configuring in the anonymous-methods parameter. Consider the following three scenarios:

1. By configuring the methods parameter with REGISTER and leaving the anonymous-methods parameter blank, the Oracle USM authenticates only REGISTER request messages, all other requests are unauthenticated.
2. By configuring the methods parameter with REGISTER and INVITE, and leaving the anonymous-methods parameter blank, the Oracle USM authenticates all REGISTER and INVITE request messages from both registered and anonymous endpoints, all other requests are unauthenticated.
3. By configuring the methods parameter with REGISTER and configuring the anonymous-methods parameter with INVITE, the Oracle USM authenticates REGISTER request messages from all endpoints, while INVITES are only authenticated from anonymous endpoints.

Credential Retrieval

The Oracle USM requests authentication information from an ENUM server via DNS when it receives a REGISTER or other message from an endpoint. This server, which provides authentication information, is defined on the Oracle USM in an enum-config configuration element that includes an enum-servers (the IP addresses of the servers) and realm parameters. Together, these two parameters define the DNS/ENUM server(s) which provide authentication data.

ENUM Based Oracle USM

The target ENUM server is determined first by setting the credential retrieval config parameter to enum-config so the Oracle USM will reference that enum config. Next, set the credential retrieval config parameter to the name of an enum config configuration element which is populated with the ENUM servers' IP addresses.

User Authentication Query

As soon as a request is received on a SIP interface and has been determined to require authentication, the Oracle USM attempts to authenticate the endpoint. It sends a DNS TXT query including the UA's AoR to an ENUM database and expects the H(A1) defined in RFC2617 for the user being authenticated.

SIP Digest User Authentication

SIP Authentication Challenge

When the Oracle USM receives a response from the ENUM server including the hash value for the user, it sends a SIP authentication challenge to the endpoint, if the endpoint did not provide any authentication headers in its initial contact with Oracle USM. If the endpoint is registering, the Oracle USM replies with a 401 Unauthorized message with the following WWW-Authenticate header:

```
WWW-Authenticate: Digest realm="atlanta.com", domain="sip:boxesbybob.com",
qop="auth", nonce="f84f1cec41e6cbe5aea9c8e88d359", opaque="", stale=FALSE,
algorithm=MD5
```

If the endpoint initiates any other request to the Oracle USM besides REGISTER, the Oracle USM replies with a 407 Proxy Authentication Required message with the following Proxy-Authenticate header:

```
Proxy-Authenticate: Digest realm="atlanta.com", qop="auth",
nonce="f84f1cec41e6cbe5aea9c8e88d359", opaque="", stale=FALSE, algorithm=MD5
```

Authentication Header Elements

- **Digest Realm**—This value is configured in the digest-realm parameter in the sip-registrar configuration element. This parameter is mandatory when using the "ENUM-TXT" credential retrieval method.
- **Domain**—A quoted, space-separated list of URIs that defines the protection space. This is an optional parameter for the "WWW-Authenticate" header.
- **Nonce**—A unique string generated each time a 401/407 response is sent.
- **Qop**—A mandatory parameter that is populated with a value of "auth" indicating authentication.
- **Opaque**—A string of data, specified by the Oracle USM which should be returned by the client unchanged in the Authorization header of subsequent requests with URIs in the same protection space.
- **Stale**—A flag indicating that the previous request from the client was rejected because the nonce value was stale. This is set to true by the SD when it receives an invalid nonce but a valid digest for that nonce.
- **Algorithm**—The Oracle USM always sends a value of "MD5"

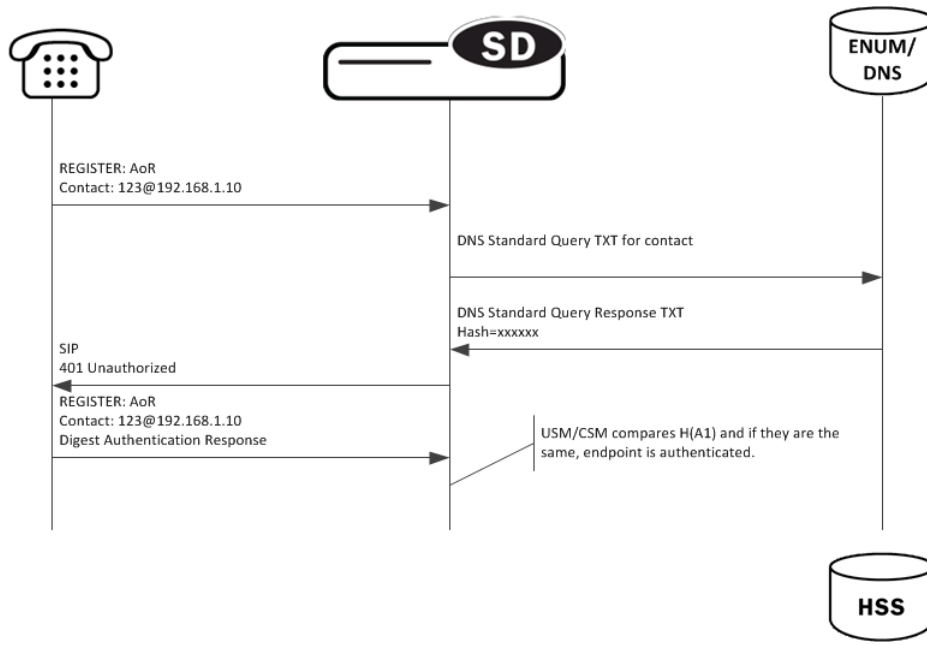
SIP Authentication Response

After receiving the 401/407 message from the Oracle USM, the UA resubmits its original request with an Authorization: header including its own internally generated MD5 hash.

Oracle USM Authentication Check

At this point, the Oracle USM has received an MD5 hash from the ENUM server and an MD5 hash from the UA. The Oracle USM compares the two values and if they are identical, the endpoint is successfully authenticated. Failure to match the two hash values results in a 403 or 503 sent to the authenticating endpoint.

The following image shows the user authentication process.



Oracle USM as Registrar

DDNS Update to User Subscriber Database

As REGISTER messages are received, the Oracle USM updates the ENUM database via DDNS UPDATE messages as defined by RFC2136. New registrations are added to the database while expired or deleted registrations are removed.

The Oracle USM acts as a registrar by configuring the sip registrar configuration element. When registrar functionality is enabled, the Oracle USM acts as a registrar rather than only caching and forwarding registrations to another device. Oracle USM registry services are enabled globally per domain, not on individual SIP interfaces or other remote logical entities.

On receiving a REGISTER message, the Oracle USM checks if it is responsible for the domain contained in the Request-URI as defined by the domains parameter and finds the corresponding sip registrar configuration. This is a global parameter and all messages are checked against all sip registrar domains. Thus you could create one sip registrar configuration element to handle all *.com domains and one sip registrar configuration element to handle all *.org domains. The Oracle USM begins registrar functions for all requests that match the configured domain per sip-registrar configuration element.

A UA is considered registered after the Oracle USM updates the ENUM server with a DDNS dynamic update. After this action, the Oracle USM sends a 200 OK message back to the registering UA.

TTL

Part of the Oracle USM architecture includes a local ENUM cache which maintains the results from ENUM queries locally. To enable Oracle USM, the TTL value in the enum config must be set to 0. This ensures that whenever an ENUM query is required, the Oracle USM consults the central User Subscriber Database to have the latest network-wide information about the UA it is trying to reach, instead of its ENUM cache.

ENUM Database Correlation

When a UA registers, as the number of associated contacts for an AoR grows or shrinks, the ENUM-based User Subscriber Database is updated in turn with the latest information using a DDNS UPDATE. After a REGISTER including authentication information is received from a UA, the Oracle USM sends a Standard NAPTR query for the AoR to the ENUM server. The ENUM server replies with a Standard Query response, including the NAPTR records.

ENUM Based Oracle USM

After receiving the entries from the ENUM database via the ENUM query, the list must be correlated with the Oracle USM's view of the AoR's registration state. The differences will be resolved by sending a DDNS UPDATE to add or remove entries from the ENUM server. The database correlation phase only occurs when endpoints register.

Contacts that need to be added are put on the UPDATE add list. Contacts that are being unregistered (contact with Expires=0) or have expired timestamp are added to the UPDATE remove list.

Entry Expiration

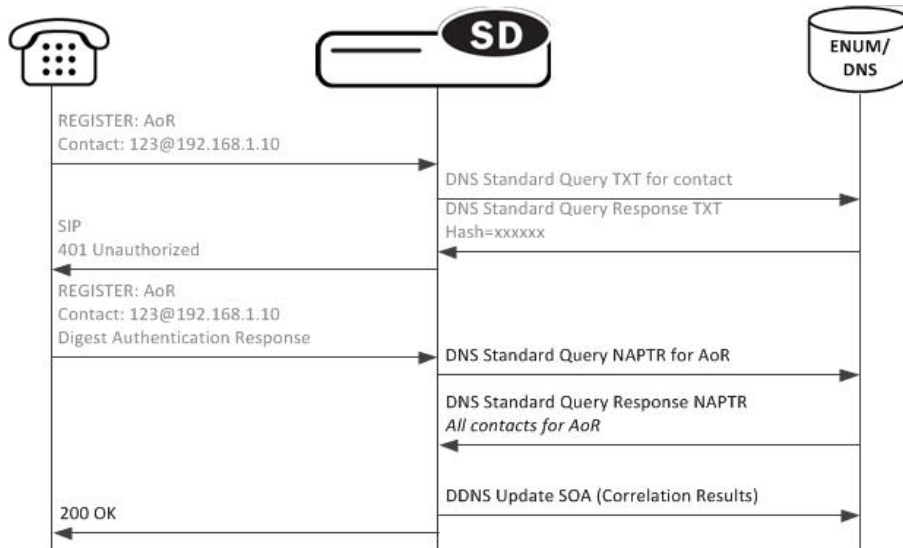
The Oracle USM employs a process to remove expired contacts from the ENUM database whether they were entered by the active Oracle USM or other Oracle USM. After determining that a contact is expired, the Oracle USM removes the contact in a subsequent DDNS update.

To do this, the Oracle USM includes an expiration parameter in the Contacts it insert into the ENUM database. Expiration is indicated with a ts= parameter. This parameter's value is set to the initial registration time measured on the Oracle USM plus the REGISTER message's Expires: header value or Expires parameter in the Contact header value. This value is measured in seconds after the epoch. In a DDNS update, a ts= parameter appears as follows:

```
Regex: "!^.*$!sip:234-hchse6c0d01u2@172.16.101.51:5060;ts=1313493824!"
```

After the Oracle USM retrieves contacts for an AoR in a NAPTR record that are expired, based on ts= parameters, the Oracle USM's next Dynamic update tells the server to remove those contacts from the ENUM database.

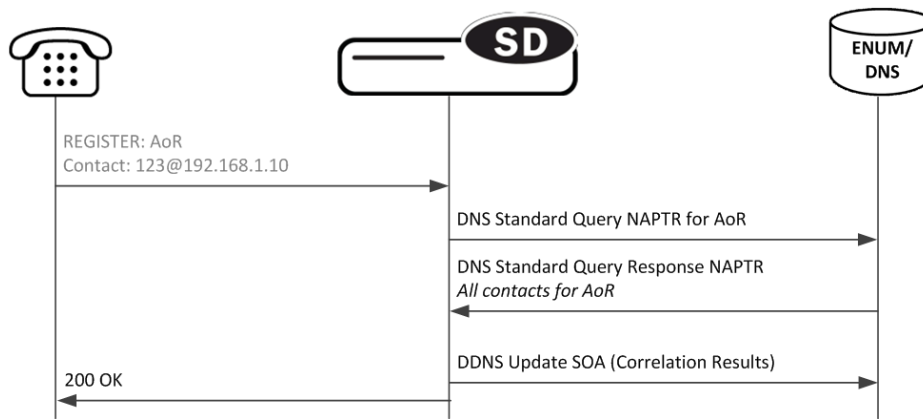
In this way, all Oracle USM members of a domain may remove expired contacts from the ENUM database.



Register Refresh

When a UA sends a register refresh, the Oracle USM first confirms that the authentication exists for that UA's registration cache entry, and then is valid for the REGISTER refresh. Then, the lifetime timer (value in Expires: header) for that registration cache entry is checked.

If the timer has not exceeded half of its lifetime, only a 200 OK is sent back to the UA. If the timer has exceeded half of its lifetime, the Oracle USM sends a NAPTR update to the ENUM database.



In addition to the baseline Oracle USM REGISTER refresh conditions, an ENUM database update is required when one of the following conditions is satisfied:

- The location update interval timer has expired—This value, configured in the sip registrar configuration element ensures that that ENUM database always has the latest user information by periodically sending Standard Queries.
- The message's call-id changes while the forward-reg-callid-change option in the sip config configuration element is set. This covers the case where the UA changes the Oracle USM through which it attaches to the network.
- The REGISTER message's Cseq has skipped a number. This covers the case in which a user registered with Oracle USM1, moves to Oracle USM2, and then returns to Oracle USM1.
- The REGISTER message's contact list has changed.

After receiving the entries from the ENUM database via the NAPTR query, the list is correlated with the internal registration cache. Appropriate DDNS updates are performed (see: ENUM Database Correlation).

If the Oracle USM updates the ENUM database due to one of the above conditions, the UA's access-side Expires timer is reset to the REGISTER message's Expires: header value, and returned in the 200 OK. This happens even in the case when the reREGISTER was received in the first half of the previous Expires period. In addition, the core-side location update interval timer is refreshed on both active and standby.

When the above four conditions are not met, the registration and reregistration expiration proceeds normally. If the access-side expiration timer has not exceeded half of its lifetime, only a 200 OK is sent back to the UA. If the timer has exceeded half of its lifetime, the Oracle USM refreshes the registration to the ENUM server. Note: Upon a Call-id or contact list change, both the registration cache timer and the ENUM database are updated.

Limiting AOR Contacts

The Oracle USM allows you to limit the number of contacts that apply to AORs. If the Oracle USM receives a registration request that exceeds the maximum that you configured, it responds with a local response, a 403 Forbidden by default, and does not register the additional contact. The system only rejects registration requests that exceed the maximum. Existing contacts persist normally.

The system checks against the maximum in the following circumstances:

- A new registration is received
- The location-update-interval expires
- A call-id changes (and the forward-reg-callid-change option is enabled)
- A registrar message sequence number has skipped a number
- There is any change to the contact list

If the number of contacts in the initial registration exceeds the maximum, the Oracle USM rejects the entire registration. In addition, if you configure this feature while the system is operational, your setting only applies to new registrations.

You configure these maximums on a per-registrar basis. The value ranges from 0-256. The feature is RTC supported.

User Registration based on Reg-ID and Instance-ID (RFC 5626)

Sometimes a user's device reregisters from a different network than its original registration. This event should be considered a location update rather than a completely new registration for the Contact. The Oracle USM can perform this way by considering the endpoint's reg-id and instance-id parameters defined in [RFC 5626](#).

The Oracle USM identifies new REGISTER requests received on a different access network as a location update of the existing binding between the Contact and AoR. Without this feature, the Oracle USM would create a new binding and leave the old binding untouched in the local registration cache/ENUM database. This scenario is undesirable and leads to unnecessary load on various network elements including the Oracle USM itself.

The following conditions must be matched to equate a newly registering contact as a location update:

For a received REGISTER:

- The message must not have more than 1 Contact header while 1 of those Contact headers includes a reg-id parameter. (failure to pass this condition prompts the Oracle USM to reply to the requester with a 400 Bad Request).
- The Supported: header contains outbound value
- The Contact header contains a reg-id parameter
- The Contact header contains a +sip.instance parameter

After these steps are affirmed, the Oracle USM determines if it is the First hop. If there is only one Via: header in the REGISTER, the Oracle USM determines it is the first hop and continues to perform Outbound Registration Binding processing.

If there is more than 1 Via: header in the REGISTER message, the Oracle USM performs additional validation by checking that a Path: header corresponding to the last Via: includes an ob URI parameter, Outbound Registration Binding may continue.

If the Oracle USM is neither the first hop nor finds an ob URI in Path headers, it replies to the UA's REGISTER with a 439 First Hop Lack Outbound Support reply.

reREGISTER Example

The user (AoR) bob@example.com registers from a device +sip.instance= <urn:uuid:0001> with a reg-id="1", contact URI = sip:1.1.1.1:5060. A binding is created for bob@example.com+<urn:uuid:0001>+reg-id=1 at sip:1.1.1.1:5060.

Next, Bob@example.com sends a reREGISTER with the same instance-id but with a different reg-id = 2 and contact URI = sip:2.2.2.2:5060.

The previous binding is removed. A binding for the new contact URI and reg-id is created. bob@example.com +<urn:uuid:0001>+reg-id=2 at sip:2.2.2.2:5060

Outbound Registration Binding Processing

An outbound registration binding is created between the AoR, instance-id, reg-id, Contact URI, and other contact parameters. This binding also stores the Path: header.

Matching re-registrations update the local registration cache as expected. REGISTER messages are replied to including a Require: header containing the outbound option-tag.

If the Oracle USM receives requests for the same AOR with some registrations with reg-id + instance-id and some without them, the Oracle USM will store them both as separate Contacts for the AOR; The AoR+sip.instance+reg-id combination becomes the key to this entry.

ENUM Database Update

When a REGISTER message is received:

1. The ENUM user database is queried for the AoR and any existing entries.
2. If there are any entries with the same instance-id as the current REGISTER request in the ENUM query response, then those entries will be marked for subsequent removal in the ENUM database.
3. The ENUM database is updated with a NAPTR request. This request adds the new Contact URI for that AOR +instance-id and removes any existing entries for the same AOR+instance-id.

NAPTR Update Format

The ENUM database update includes the instance-id and reg-id when those parameters are present in a registration. These values are appended to the regex replacement field. For example:

```
!^.*$!sip:3556-1cdstqjt90hve@172.16.101.62:5060;sip.instance=<urn:uuid:00000000-0000-1000-8000-000A95A0E128>;reg-id=1;ts=1326568408;!
```

Oracle USM Licensing

The Oracle USM connected to an ENUM database requires two licenses: Registration Cache Limit, SIP Authorization/Authentication.

For ENUM-based Oracle USM, the SIP Authorization/Authentication license reveals the SIP Authentication Profile configuration element. Configuring both configuration elements is required to operate a Oracle USM. Refer to the Licensing and Database Registration Limits section for the third license required for Oracle USM operation.

Refer to the Oracle SBC ACLI Configuration guide, Getting Started chapter for how to install licenses in your system.

ACLI Instructions

ENUM Configuration

First the server used for authentication and as the User Subscriber Database is created.

To configure the ENUM Configuration:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the media-related configurations.

```
ORACLE(configure)# session-router
```

3. Type enum-config and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# enum-config
ORACLE(enum-config)#
```

You may now begin configuring the enum-config configuration element.

4. name—Set a name to use to reference this enum configuration from within the Oracle USM.
5. top-level-domain—Enter the domain which this ENUM server(s) services and returns results for.
6. realm-id—Enter the realm name where this ENUM server exists.
7. enum-servers—Enter the IP address of one or more ENUM servers used for registration. Multiple entries are separated by commas.
8. service-type—Leave this as its default.
9. ttl—Leave this at the default of 0 to set the TTL value (in seconds) for NAPTR entries as populated when sending a DNS update to the ENUM server.
10. order—Enter the value to populate the order field with when sending NAPTR entries to the ENUM server.

11. preference—Enter the value to populate the preference field with when sending NAPTR entries to the ENUM server.
12. Type done when finished.

SIP Authentication Profile

To configure the SIP Authentication Profile:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the media-related configurations.

```
ORACLE(configure)# session-router
```

3. Type sip-authentication-profile and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-authentication-profile
ORACLE(sip-authentication-profile)#
```

You may now begin configuring the SIP Authentication Profile configuration element.

4. name—Enter the name of this SIP authentication profile that will be referenced from a SIP registrar (or SIP interface).
5. methods—Enter all the methods that should be authenticated. Enclose multiple methods in quotes and separated by commas.
6. anonymous-methods—Enter the methods from anonymous users that require authentication. Enclose multiple methods in quotes and separated by commas.
7. digest-realm—enter the digest realm sent in an authentication challenge (401/407) sent to a UA. This is required in ENUM/DNS deployments,.
8. credential-retrieval-method—Enter ENUM-TXT.
9. credential-retrieval-config—Enter the enum-config name used for retrieving authentication data.
10. Type done when finished.

SIP Registrar

To configure the Oracle USM to act as a SIP Registrar:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type sip-registrar and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)#
```

4. name—Enter a name for this SIP registrar configuration element.
5. state—Set this to enabled to use this SIP registrar configuration element.
6. domains—Enter one or more domains in the R-URI this configuration element handles. Wildcards are valid for this parameter. Multiple entries can be entered in quotes, separated by commas.
7. subscriber-database-method—Set this to DDNS.
8. subscriber-database-config—Enter the enum-config configuration element name that will handle REGISTER messages for this domain. This should be the same element used for requesting authentication data.
9. authentication-profile—Enter a sip-authentication-profile configuration element's name. The sip authentication profile object referenced here will be looked up for a REGISTER message with a matching domain in the request

URI. You may also leave this blank for the receiving SIP Interface to handle which messages require authentication if so configured.

10. location-update-interval—Keep or change from the default of 1400 minutes (1 day).
11. Type done when finished.

Maximum Number of Contacts

To configure a sip-registrar with a maximum of 10 contacts per AOR:

1. From superuser mode, use the following command sequence to access sip-registrar element.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)# select
```

Select the registrar you want to configure.

2. Specify the number of contacts.

```
AORACLE(sip-registrar)# max-contacts-per-aor 10
AORACLE(sip-registrar)# done
```

Response to Exceeding Maximum Contacts

To configure local response for the Oracle USM to issue when max-contacts-per-aor is exceeded:

1. From superuser mode, use the following command sequence to access local-response and add an entry.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# local-response-map
```

2. Access the entries configuration.

```
ORACLE(local-response-map)# entries
```

3. Specify the local error you need to configure.

```
ORACLE(local-response-map-entry)# local-error contacts-per-aor-exceed
```

4. Specify the sip-reason for this error.

```
ORACLE(local-response-map-entry)# sip-reason forbidden
```

5. Specify the error code for this error.

```
ORACLE(local-response-map-entry)# sip-status 403
ORACLE(local-response-map-entry)# done
local-response-map-entry
  local-error                contacts-per-aor-exceed
  sip-status                 403
  q850-cause                 0
  sip-reason                 forbidden
  q850-reason
  method
  register-response-expires
ORACLE(local-response-map-entry)# exit
```

Update to ENUM Database on Endpoint Connection Loss

The Oracle USM can monitor an endpoint's transport-layer status for loss of connectivity in multiple ways. Then, when the endpoint's connection to the Oracle USM has been terminated, the Contact is removed from the registration cache, as associated under a registered AoR. In addition, the user database is immediately updated.

These transactions contribute to the registration cache and ENUM user database being updated in real time to retain only reachable contacts for a registered AoR. This feature helps to alleviate:

- unnecessary transactions and system load spent on attempting to reach an unreachable endpoint
- incorrect and out of date statistics

Connection Reuse

The connection between the Oracle USM and an endpoints must employ the connection reuse mechanism to enable this feature's de-registration and user database updates. Connection reuse is when an endpoint registers to the Oracle USM and all subsequent signaling between the Oracle USM and that endpoint reuses the same socket pair. There are four cases when connection reuse is enabled:

- Connection reuse is enabled on the SIP interface facing the endpoint(s). This is enabled by adding the reuse-connections=yes option on a SIP interface.
- The endpoint is behind a NAT. Because of the fundamental method that the Oracle USM uses for maintaining its connection to an endpoint behind a NAT, this case will always force connection reuse.
- The endpoint includes the alias parameter in the Via: header in its REGISTER message to the Oracle USM (RFC 5923).
- If the endpoint is configured as a session agent, the reuse-connections parameter must be set to TCP. When receiving signaling from a remote logical entity such as a session agent defined for an endpoint, if the reuse-connections parameter is set to tcp, the Oracle USM enables connection reuse between itself and the UA.

Unreachability Determination


There are four ways that the Oracle USM determines endpoint is not reachable:

- No CRLF message is returned to the Oracle USM within the expected time frame: this is based on the Oracle USM's RFC5626 support.
- No TCP Keepalive is returned to the Oracle USM within the expected timeframe. This is based on configuring the network parameter configuration element per application interface.
- The endpoint explicitly terminates its transport-layer connection to the Oracle USM.
- The endpoint is otherwise labeled as unreachable from a non-explicit fault condition for a call made to an unreachable endpoint.

RFC 5635 Failure

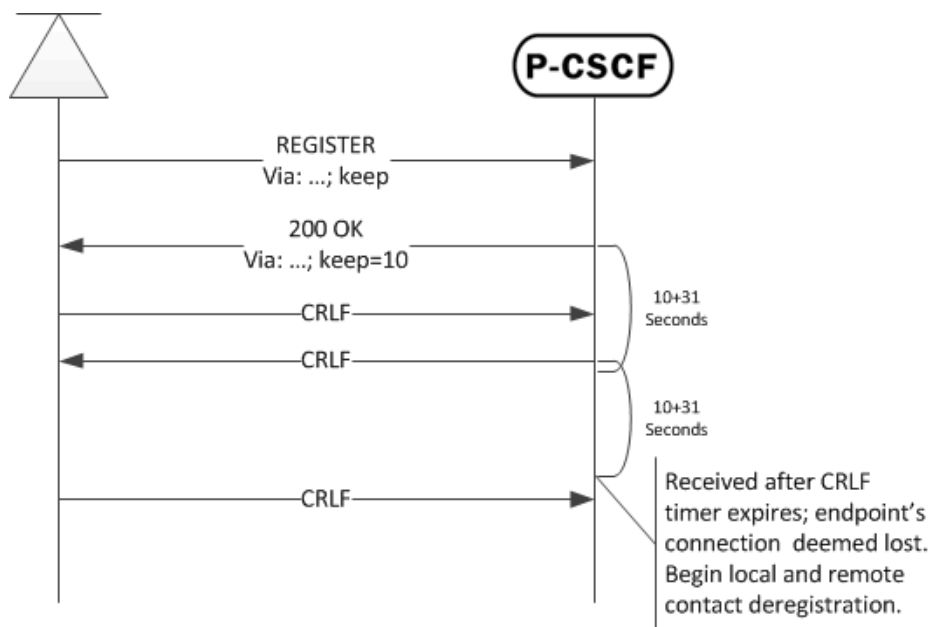
The Oracle USM supports the RFC 5635 method of SIP application keepalives, which are endpoint-initiated, i.e., the endpoint starts the mechanism by including the keep parameter in the initial Via: header. Endpoint reachability is determined the receipt or loss of a CR/LF ping-pong message. In the SIP interface configuration element, you set the register keep alive parameter to always or bnat (behind NAT), to enable RFC 5635 functionality. This applicable to TCP or TLS connections. Always forces the Oracle USM to always return a CRLF reply when the keep parameter is in the initial Via: header. bnat forces the Oracle USM to replies to RFC 5635 requests when the endpoint is located behind a NAT.

Next, you can accept the Oracle USM's default keep alive window of 30 seconds, or you may set your own by configuring the tcp nat interval or inactive con timeout value, both found in the sip interface configuration element. The Oracle USM uses the smaller of the two configured values. The chosen value is inserted into the keep parameter in the 200 OK message returned to the registering endpoint.

 **Note:** The inactive con timeout value otherwise disconnects a TCP/TLS connection after the configured value elapses. The tcp nat interval value is also inserted into the expires parameter in the Contact: header for devices identified as behind a NAT.

In addition, the Oracle USM maintains a keep timer. The Oracle USM adds 31 seconds to the keep value it returns and begins counting down. 31 is the chosen margin to account for any network or application delay.

If the endpoint returns the CRLF before the timer expires, the endpoint is considered up and a new CRLF ping is sent to the endpoint; the timer begins counting down again. If the Oracle USM fails to receive the CRLF ping before the timer expires, then the UA is considered unreachable. Provisions begin to remove that contact from the registration cache and then the ENUM user database.



TCP Keepalive Failure

Endpoint reachability can be determined from TCP keepalives, as enabled per SIP interface. This method of determining connectedness is based on configuring the global network parameters configuration element that is enabled on a SIP interface with the `tcp-keepalive` parameter. See the System TCP Keepalive Settings section of the Oracle SBC ACLI Configuration guide for how to configure the TCP keepalive feature. When an endpoint fails the TCP keepalive test, provisions begin to remove that contact from the registration cache and then the ENUM user database.

Explicit and undetermined connection termination

Ideally, a UA will gracefully close its TCP connection to the Oracle USM, and in turn the socket pair will be considered closed with the UA being unreachable. In less ideal cases, the UA goes dark and no response is received when expected. The Oracle USM considers the unresponsive UA as unreachable as call attempts time out. Provisions then begin to remove that contact from the registration cache and then the ENUM user database.

Registration Cache and User Database Removal

When the Oracle USM sets up the initial REGISTER request from a UA, an internal binding is created between the contact and the AoR for that user. If a UA is considered unreachable by either of the four conditions explained in the previous section, the following occur:

- The contact's entry in the registration cache is removed. If this is the last contact registered for an AoR, the entire entry for the AoR is removed from the registration cache.
- The Oracle USM sends an UPDATE to the ENUM user database removing the Contact.

To enable these actions, you must configure the `force-unregistration` option sip config configuration element and also set the `unregister on connection loss` parameter in the sip interface configuration element to enabled.

ACLI Instructions

To globally enable force-unregistration at the sip-config level:

1. In Superuser mode, type `configure terminal` and press Enter.

```
ORACLE# configure terminal
```

2. Type `session-router` and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type sip-config and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-config
ORACLE(sip-config)#
```

4. Type select to continue.

```
ORACLE(sip-config)# select
ORACLE(sip-config)#
```

5. options—Set the options parameter by typing options, a Space, force-unregistration with a plus sign in front of it, and then press Enter.

```
ORACLE(sip-config)# options +force-unregistration
```

If you type the option without the plus sign, you will overwrite any previously configured options. In order to append the new options to the realm configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.

6. Type done and exit to complete configuration of this sip-config configuration element.

SIP Interface Configuration

To configure the SIP Interface configuration element portion of the ENUM Database update feature:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type sip-interface and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-interface
ORACLE(sip-interface)#
```

4. Type select and choose the number of the pre-configured sip interface you want to configure.

```
ORACLE(sip-interface)# select
<realm-id>:
1: private 192.168.101.17:5060
2: public 172.16.101.17:5060
selection: 1
```

5. unregister-on-connection-loss—Set this parameter to enabled for the Oracle USM to update the ENUM server when an endpoint is deemed failed.

Parameters for determining endpoint reachability:

6. tcp-keepalive—You may set this parameter to enabled to enforce the network-parameters configuration element located in the system-config path. See the Oracle SBC ACLI Configuration guide, System TCP Keepalive Settings section for more information.
7. register-keep-alive—Set this parameter to always for the Oracle USM to return the keep parameter, with optional value (as configured in the next two steps) in the Via: header to an endpoint including an empty keep value in its initial REGISTER message.

You may set none, one, of both the following for a keep value returned to the initiating endpoint. Read the RFC 5635 Failure section for how the actual value is determined:

8. inactive con timeout—Set this parameter value to the value in seconds inserted in the returned keep parameter for RFC 5635 support
9. tcp nat interval—Set this parameter value to the value in seconds inserted in the returned keep parameter for RFC 5635 support

Type done and exit to complete configuration of this sip-interface configuration element.

OAuth 2.0 Support

The Oracle USM supports Open Authorization (OAuth) in addition to SIP digest authentication for user authorization within ENUM deployments. Both authorization methods can be operational simultaneously, allowing some users to authorize via OAuth and others via SIP digest. Applicable scenarios include authorizing registrations, subscriptions and invites.

OAuth uses HTTP to provide end users with access to services from OAuth 2.0 protected resources using various clients. OAuth also allows users to authorize third-party access to their services using user-agent redirections rather than sharing username password pairs. Any party presenting the proper bearer token can be authenticated. The methodology avoids the use of cryptographic keys, and requires protection from token disclosure during transit and storage. OAuth assumes a secure exchange of credential validation information between end points, specifically an OAuth client and server, prior to operation.

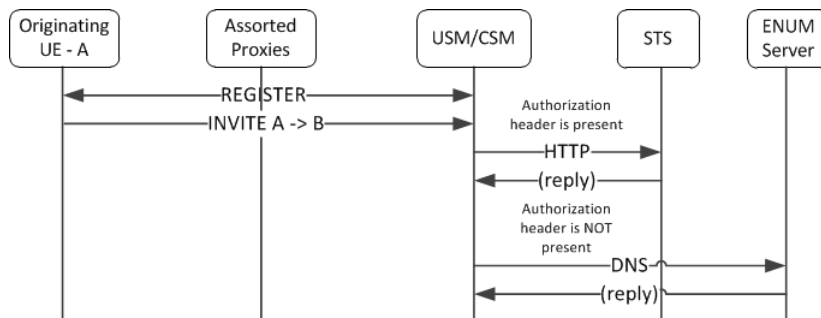
The Oracle USM implements OAuth in compliance with RFCs 6749 and 6750. OAuth typically requires deployment-specific compliance beyond RFC compliance. The Oracle USM allows for typical deployment environments via configuration.

You configure the Oracle USM to use OAuth by creating OAuth profiles and applying them globally and/or to specific interfaces. Interface profiles take precedence. In addition, you specify the server that the Oracle USM must contact for authentication using Oracle's Session Processing Language (SPL). Note that this functionality requires that you upload a script file, provided by Oracle, and run an ACLI command to configure the script to use your server. This script file is referred to as a plug-in.

OAuth Operation

The Oracle USM assumes a client requesting OAuth authentication has previously acquired a token from its authorization server. This and token refresh procedures are performed outside of the scope of Oracle USM operation. Within the context of OAuth processes, the Oracle USM performs the functions of a resource server only.

Upon receiving applicable requests from a UA, the Oracle USM attempts to authenticate using OAuth or Digest. If the request includes a bearer string and a token, the Oracle USM initiates OAuth authentication by sending an HTTP query (GET request) to a Secure Token Service Server (STS), as shown below.



If the STS returns a 200 OK, the Oracle USM proceeds with authentication. If the STS does not reply or is not reachable, the Oracle USM replies to the UA with a 500 Internal server error. If the STS replies with any message other than a 200OK, the Oracle USM replies to the UA with a 403 Forbidden.

The Oracle USM uses the User ID within the STS 200OK to authenticate and authorize service, as follows:

- For register requests, the system compares this to the user ID field in the TO header of the SIP request.
- For other requests, the system compares it to the user ID field in the FROM header.

If the user ID matches, the Oracle USM proceeds with authentication. If this procedure concludes with matches, the Oracle USM provides service to the UA. If not, it replies to the UA with a 403 forbidden.

Note the following operational caveats:

- If the bearer token is present, but the Oracle USM is not configured for OAuth, the Oracle USM responds with a 500 internal service error message. Such misconfiguration includes the OAuth SPL plugin, described below, being absent or disabled.
- If the bearer token is not in the request, the Oracle USM proceeds with SIP digest authentication.
- If the request arrives over a secure channel, the Oracle USM follows the procedure defined by the applicable sip-authentication-profile, which may not require authentication.

Configuring OAuth Support

Configuring the Oracle USM for OAuth support consists of:

- Creating one or more OAuth profiles
- Applying each profile globally or to sip-interfaces

To configure an OAuth profile:

1. From superuser mode, use the following command sequence to access http-config element and define your profile.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# http-config
ORACLE(http-config)# sel
```

2. Name your profile for reference within your configuration.
3. You can set the host parameter to an FQDN or an ip address. If you use an FQDN, the Oracle USM resolves it to an ip address when the configuration is loaded.
4. The num-connections parameter specifies the number of connections to be setup with the STS server. The Oracle USM establishes this number of connections when it boots or when the configuration is activated.
5. The max-outstanding-msgs parameter specifies the maximum number of outstanding messages per connection. An outstanding message is a request from the Oracle USM that has not had a response. When reaching this threshold, the Oracle USM stops sending requests on the connection until the number of outstanding messages falls back below this maximum. If the max-outstanding-msgs parameter is set to 1, the Oracle USM waits for a response before sending another request on a connection.
6. The Oracle USM uses the port parameter differently depending on whether the host parameter is configured as an IP address or an FQDN. If the host parameter is an FQDN name, the Oracle USM performs a lookup at a DNS server. In this case, you may or may not configure a port. If you configure the port parameter, the Oracle USM uses the configured port ignoring any port specified in the record returned by the DNS server. If you do not configure the port parameter, the Oracle USM uses the port returned by the DNS server. If the host parameter is set to an ip-address, you must configure the port parameter and the Oracle USM always uses the port parameter's value.

Enabling the SPL Plug-in

Enabling the SPL plug-in is a four step process.

1. Upload the SPL plug-in to a Oracle USM.
2. Add the SPL plug-in to the Oracle USM configuration.
3. Execute the SPL plug-in on the Oracle USM.
4. Synchronize the plug-in across HA pairs.

Uploading the Plug-in

The plug-in must be manually FTPed to the Oracle USM's /code/spl directory using any CLI or GUI-based FTP or SFTP application. The Oracle USM's FTP/SFTP server may be reached from the system's wancom or eth0 management physical interface.

Adding the Plug-in to Your Configuration

The plug-in must be configured in the spl-config configuration element. If multiple plugins are configured on the Oracle USM, the plug-ins are executed in the order of configuration.

To add the SPL Plugin to the configuration:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type system and press Enter to access the system-level configuration elements.

```
ORACLE(configure)# system
ORACLE(system)#
```

3. Type spl-config and press Enter.

```
ORACLE(system)# spl-config
ORACLE(spl-config)#
```

4. Type plugins and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ACMESYSTEM(spl-config)# plugins
ACMESYSTEM(spl-plugins)#
```

5. Type name, a <space>, and the name of the SPL plug-in file.

```
ACMESYSTEM(spl-plugins)#name SMXOAuth2.spl
```

6. Type done to save your work.

Executing SPL Files

There are two ways to execute SPL files:

1. Perform a save-config and activate-config after exiting the configuration menu.
2. Execute the reset spl command—all configured SPL files are refreshed by with the reset spl command. You can also refresh a specific file by typing reset spl <spl-file>.



Note: Oracle suggests that scripts are only refreshed during planned maintenance windows.

If an SPL file exists in the /code/spl directory, but is not configured in the spl-files parameter, it will be ignored when the ACLI User Interface is booting.

Synchronizing SPL Files Across HA Pairs

When running in an HA configuration, both the active and the standby systems must have the same version of the SPL plugins installed. To facilitate configuring the standby system, you can executing the synchronize spl ACLI command (without any arguments) to copy all files in the /code/spl directory from the active system to the same directory on the standby, overwriting any existing files with the same name.

By adding the specific filename as an argument to the synchronize spl command, the individual, specified scripts are copied between systems. For example:

```
ORACLE#synchronize spl SMXOAuth2.spl
```

The synchronize spl command can only be executed from the active system in a HA pair. There is no means to synchronize SPL files automatically during a save and activate of the Oracle USM.

To synchronize all SPL Plug-ins to the configuration:

In Superuser mode, type synchronize spl and press Enter.

```
ORACLE# synchronize spl
```

Configuring the Plug-in Option

For the SPL to work, you must configure it to recognize your http-server configuration. You do this by setting the sts-server option at the sip-registrar. This option is required for plug-in functionality.

To configure the sts-server option:

1. In Superuser mode, type configure terminal and press Enter.

ENUM Based Oracle USM

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

```
ORACLE(configure)# session-router
```

3. Access the target sip-registrar by typing sip-registrar, selecting your registrar and pressing Enter.

```
ORACLE(session-router)# sip-registrar
```

```
ORACLE(sip-registrar)# select
```

4. Type sts-server=<http-config>, where "<http-config>" is the name of your http configuration element, and press Enter.

```
ACMESYSTEM(sip-registrar)# spl-options sts-server=http-server1
```

5. Type done to save your work.

Message Routing

The Oracle USM performs routing in two ways depending on the routing precedence parameter in the sip registrar. Routing precedence can be set to either registrar (ENUM) or local policy. Routing precedence is set to registrar by default.

Registrar routing uses the configured SIP registrar/ENUM server for destination address queries. Local policy routing lets you configure routing decisions within the Oracle USM's local policy routing functionality.

If the Oracle USM is performing any services for the call it performs those services prior to routing.

If, for any reason, the Oracle USM is unable to proceed with routing a request, it replies to the station that sent the request with a 4xx response.

Registrar Routing

When the routing precedence parameter is set to registrar, the Oracle USM is using the ENUM server as a resource within the context of its routing decisions.

When an INVITE arrives, the Oracle USM issues a query to the ENUM server. If the query's reply includes a match, the Oracle USM proceeds by forwarding the message via the information in the reply.

Note that you can configure the Oracle USM to fallback to a local policy lookup if the lookup via the registrar fails. Configure this by adding the fallback-to-localpolicy option to the sip-registrar configuration element.

For situations where the database routing decision needs to be done in lieu of the default, you can set routing precedence to local-policy. Note that you can configure a routing entry that points to an HSS by setting a policy attribute with a next-hop of enum:<server-name> within the local-policy.

Default Egress Realm

The sip registrar configuration element should be configured with a default egress realm id. This is the name of the realm config which defines the IMS control plane through which all Oracle USMs, ENUM servers, and other network elements communicate and exchange SIP messaging. It is advisable to configure this parameter in order to ensure well defined reachability among Oracle USMs.

SIP Registrar

To configure a SIP registrar configuration element for message routing:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type sip-registrar and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router) # sip-registrar
ORACLE(sip-registrar) #
```

4. Type select and choose the number of the pre-configured sip registrar you want to configure.
5. routing-precedence— Set this to either registrar or local-policy depending on your deployment.
6. egress-realm-id—Enter the default egress realm for Oracle USM messaging.
7. Type done when finished.

Segmentation of ENUM Zones

The Oracle USM supports operations with an ENUM root server, from which it can obtain partition delegation information and dynamically reach delegated authoritative ENUM servers for access information by ENUM zone. Obtaining this information is dynamic and supports query re-direction to other relevant authoritative servers the Oracle USM may learn from the root server. This configuration applies only to DDNS-based registration deployments. The user configures the **ddns-ns-caching** option to enable the Oracle USM for dynamically learned ENUM server caching.

For DDNS queries, the Oracle USM uses a manually-configured ENUM server configuration to reach a DNS root server. The root server can respond with the authoritative name server that manages the zone. The Oracle USM establishes dynamic entries in its ENUM server cache, with detail about the zones those servers service. This allows it to issue subsequent lookup queries directly to the correct server based on a match between telephone number and zone prefixes.

This feature assumes ENUM servers that are bind 8.0 compliant and return authority sections compliant with RFC 1304, section 4.3.2.

Additional operational details include:

- Resolution lookups are longest match, providing the Oracle USM with the ability to send a query to authoritative servers servicing zones and sub-zones.
- Dynamic server timeouts are equal to the TTL values received in the name server's resource record, provided in the NS response. The Oracle USM removes dynamic servers, as well as any associated ACL entries upon timeout.
- Redundancy for DDNS name server lookups is round-robin, triggered by standard DNS query timeout procedures, and following the order the system learns servers for the target zone.
- If a dynamic server fails to respond to a DDNS lookup, the Oracle USM removes that server from the lookup list. If there are no further servers in the list, the Oracle USM sends a 404 back to the station that originated the request. The Oracle USM then starts any subsequent lookups for that zone by querying the root server.
- Multiple NS records are supported. The Oracle USM creates NS records for the same zones in the order provided to it to establish name server redundancy.
- Should the Oracle USM need to obtain a resolution for a prefix that is unknown, the Oracle USM queries the original root server again.
- The Oracle USM removes any dynamically-learned name server from a query list when any query fails. That server is only added back to a list when it is dynamically re-learned.
- Should the root server become unavailable, the Oracle USM sends applicable queries to the next server in your manually-configured ENUM list.
- The Oracle USM does not execute its configurable health-query check processes with dynamically learned name servers.
- With respect to high availability deployments, the server list is not maintained on a standby Oracle USM. The ENUM configuration, however, is maintained on the standby via configuration synchronization. In the event of a failover, the standby Oracle USM learns all ENUM servers using the root server as a starting point.
- This **ddns-ns-caching** function interoperates with the Oracle USM's alphanumeric user name function. The zone match is done on the 3-bit checksum of the alpha-numeric user.



Note: Name server records must present the Oracle USM with IP addresses for the server. Name server records using FQDNs are not supported.

Obtaining Information about Dynamic ENUM Servers

The Oracle USM provides ACLI commands that display real-time information about ENUM server interaction, including dynamically learned servers. The **show enum** command includes a section of statistics on cached entries

```
ORACLE# show enum stats localenum
Parameter-> localenum lastArg 3 enumStatsType 0

SIP ENUM Statistics:
16:01:51-48
ENUM Agent localenum

-- Period -- ----- Lifetime -----
Active High Total Total PerMax High
Queries - - 3 3 3 -
  Successful - - 3 3 3 -
  NotFound - - 0 0 0 -
  TimedOut - - 0 0 0 -
  Bad Status - - 3 3 3 -
  Other Failures - - 0 0 0 -
Transactions 0 1 4 4 4 1
Cache Hits
  Successful - - 0 0 0 -
  NotFound - - 0 0 0 -
Cache Entries
  Successful 0 0 0 0 0 0
  NotFound 0 0 0 0 0 0
DropdCacheEntries - - 0 0 0 -
```

When appended with the **cache-console** argument, **show enum** displays detailed information about dynamic server configuration and status.

```
ORACLE# show enum status all cache-console
Showing 2 Enum Agents: state=Active cache
-----
Enum Agent:      localenum
  Realm:         net192.4
  EnumServers:   192.168.53.199
  Query Timeout: 11
  Lookup Length: 3
  Health Query:  ' every 0 sec
  Failover To:
  IncludeSrcInfo: disabled
  Options:
  RecursiveQuery: disabled
DNS Agent:      EnumAgent[0x3b75a628(3)] ENUM w/o-stats
  Domain:       bogus.gy
  Local Address: [256:0]192.168.53.170
  Service Types: E2U+sip sip+E2U
  Trans ID:     7
  Cache Size:   0(0) inact-tmr=0 cache-addl=no
  Max Resp Size: 512
  Query Method: hunt
servers:
  1=[256:0]192.168.53.199:53 OK
dynamic servers:
  1=[256:0]192.168.53.205:53 OK
```

Configuring Support for DDNS Server Caching

Use the procedure below to enable the Oracle USM to cache and manage DDNS servers for the purpose of efficiently forwarding ENUM queries to the servers responsible for the DNS zones identified in applicable SIP requests.

1. From superuser mode, use the following command sequence to access **enum-config** configuration mode.


```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# enum-config
```

2. Use the **select** command to access the **enum-config** element or create a new one for the target ENUM root server.
3. Enable the **ddns-ns-caching** option.

```
ORACLE(enum-config)# +option ddns-ns-caching
```

4. Use **done**, exit configuration mode, and run **verify-config** to complete DDNS cache support configuration.

```
ORACLE(enum-config)# done
ORACLE(enum-config)# exit
ORACLE(session-router)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

Tel-URI Resolution

The Oracle USM can initiate number resolution procedures for requests that have tel-URI or SIP-URI (with user=phone) numbers in the R-URI. It does this by querying number resolutions services, including the local routing table(s) or ENUM server(s) to resolve the R-URI to a SIP URI. In addition, the original R-URI may not include a full E.164 number. As such, you can also configure the Oracle USM to perform a number normalization procedure and ensure it presents a full E.164 number for resolution. Upon successful resolution, the Oracle USM proceeds with ensuing signaling procedures.

To configure the Oracle USM to perform these lookups, you create applicable local-routing-config or enum-config elements and set an option within the sip-registrar that specifies a primary and, optionally, a secondary local-routing-config or enum-config that the sip-registrar uses for LRT or ENUM lookups. If there is no ENUM configuration on the sip-registrar, the Oracle USM forwards applicable requests to a border gateway function via local policy.

Refer to the Net-Net SD ACLI Configuration guide, Session Routing and Load Balancing chapter for complete information on how to configure a local-routing-config and an enum-config.

Number Lookup Triggers

Use cases that are applicable to number lookups and the associated Oracle USM procedures include:

- Request from the access side:
 1. The Oracle USM performs originating services
 2. If the R-URI is a tel-URI or SIP-URI (with user=phone) and is not in the Oracle USM cache, it requests e.164 resolution from the ENUM server(s).
- Request from core side including request for originating services:
 1. The Oracle USM performs originating services
 2. If the R-URI is a tel-URI or SIP-URI (with user=phone) and is not in the Oracle USM cache, it requests e.164 resolution from the ENUM server(s).
- Request from core side, for terminating services only:
 1. If the R-URI is a tel-URI or SIP-URI (with user=phone) and is not in the Oracle USM cache, it performs a NAPTR lookup.
 2. If the reply indicates the tel-URI or SIP-URI (with user=phone) is not provisioned, the Oracle USM requests e.164 resolution from the ENUM server(s).

Actions Based on Lookup Results

The Oracle USM forwards to the resultant SIP-URI under the following conditions:

ENUM Based Oracle USM

- The SIP-URI is in the Oracle USM cache, in which case the Oracle USM performs terminating services.
- The SIP-URI is not in the Oracle USM cache, and the Oracle USM is configured to service the returned domain. In this case, the Oracle USM performs the following:
 1. The Oracle USM issues an LIR for the SIP-URI.
 2. The Oracle USM forwards the message to the correct S-CSCF.
- The SIP-URI is not in the Oracle USM cache, and the Oracle USM is not configured to service the returned domain. In this case, the Oracle USM performs refers to local policy to forward the message via local policy.

PSTN Breakout Routing

The Oracle USM complies with RFC 4694 for operation with request-URIs that include carrier identification code/route number/number portability database dip indicator (cic/rn/npdi) information and routes those requests according to the rn information. The routing process includes utilization of local policy configured to break the request out of the home network via gateways such as a BGCF.

The Oracle USM does not validate any rn or cic information. Instead, it simply routes the request. Note that the Oracle USM uses cic information instead of rn if both are present in the request. RFC 4694 compliant circumstances under which the Oracle USM does not use rn, cic and npdi information include:

- Invalid routing information, including rn present, but npdi missing.
- Invalid routing information, including npdi present, but rn missing.
- Request uses a sip-URI presented without user=phone.

If the request includes originating services as well as cic/rn/npdi information, the Oracle USM performs those services rather than break out. If, after completing originating services, the request still includes cic/rn/npdi information, the system performs this breakout.

Primary and Secondary ENUM Configs

For the purpose of redundancy, the Oracle USM allows you to configure these number lookups to use a backup resource in case the lookup at the primary fails. Such scenarios include losing contact with the primary ENUM/LRT server config (query time-out) and the entry is not found at the primary (LRT or ENUM).

To apply primary and secondary number lookup resources to a sip-registrar:

1. From superuser mode, use the following command sequence to access the sip-registrar element and select the registrar you want to configure.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)# select
```

2. Specify the resources to use with the options command.

Prepend the option with the + character if you have multiple options configured that you want to retain. Running the command without the + character causes the system to disable any previously configured options.

To specify primary and secondary ENUM servers:

```
ORACLE(sip-registrar)# options +e164-primary-config=enum:<enum-config name>
ORACLE(sip-registrar)# options +e164-secondary-config=enum:<enum-config name>
ORACLE(sip-registrar)# done
```

To specify primary and secondary LRT resources:

```
ORACLE(sip-registrar)# options +e164-primary-config=lrt:<lrt-config name>
ORACLE(sip-registrar)# options +e164-secondary-config=lrt:<lrt-config name>
ORACLE(sip-registrar)# done
```

Bear in mind that an enum-config can reference multiple servers. When the Oracle USM references an enum-config, queries follow the normal enum-config sequence, checking each referenced server in order. If the lookup is not successful at the primary, the Oracle USM checks the servers in the registrar's e164-secondary-config.

In addition, each enum-config may refer to a different top-level-domain. This allows you to configure the Oracle USM to successfully perform lookups within two domains.

Licensing and Database Registration Limits

The Oracle USM (and Net-Net SBC) limit the number of unexpired registration cache entries globally. The total number of system registrations is configured with the registration cache limit parameter in the sip config configuration element.

The Oracle USM also limits the number of registration cache entries that were obtained from a User Subscriber Database; only REGISTERS that prompted the database query are counted here. As User Subscriber Database entries are added and removed, this counter is updated accordingly. Note that it is the actual number of SD-contacts that count against the license limit. Discrete database registration license values range from 20,000 through 500,000 in increments of 20,000.

When a registering contact is rejected because it will exceed one of these limits, the Oracle USM sends a 503 message to the registering endpoint.

Refer to the Net-Net 4000 ACLI Configuration Guide, chapter 2 Getting Started, Software Licensing section for how to install a license.

Database Registration Limit Alarm

By default, a major alarm is enabled when 98% or more of the licensed number of Database Registrations are used. This alarm is cleared when the number of database registrations falls below 90%. You can configure minor and critical alarms when crossing configured thresholds and you can also reassign the major alarm. This is configured in by creating a system-config > alarm-threshold sub element with type of database-registration.

Extended ENUM Record Length

In some cases, when a user registers a contact from a UA, the Contact: header's contents are too long to be inserted into the DNS-based user database as presented in the NAPTR record.

To mitigate this, the Oracle USM stores contact-related metadata, i.e. parameter key and value pairs, in a related TXT record.

If the contents of the Contact: header do not exceed 255 bytes, there is generally no need to extract the metadata to store in an additional TXT record, so the Oracle USM will not enact this process.

NAPTR and TXT Record Creation and Association

When a user initially registers to the Oracle USM, the DDNS update sent to the user database will include a NAPTR record and a TXT record. The NAPTR record contains the Contact: header's SIP URI as a regexp replacement. All URI and header parameters from the Contact: header are excluded from the NAPTR record and are inserted into the accompanying TXT record.

The NAPTR record and the TXT record both contain a Oracle USM-generated key that binds the Contact metadata and the Contact URI sent in the separate records. This common key indicates the data in the NAPTR and TXT record belong to the same registering contact, and is used to recreate the Contact: header for later use. The format of the common key is:

```
p-acme-ckey=<SD-Core-IP>:<integer id>
```

ENUM Based Oracle USM

The <SD-Core-IP> is the IP address from which the Oracle USM communicates with the DNS server. The <integer id> enumerates each contact. Contacts are enumerated in the integer-id element because they can be non-unique when a user behind a NAT registers more than one contact from behind the NAT. For example:

```
TXT "p-acme-ckey=172.16.101.61:1" "$key=value" "^key=value"
NAPTR 1 1 "u" "E2U+sip" "!^.*$!sip:642-10u72@172.16.101.61:5060\";p-acme-
key=172.16.101.61:1!"
```

In the previous example, the key=value pair represent parameters in the Contact header and Contact SIP URI. A \$key= indicates the parameter existed in the Contact: SIP URI. A ^key= indicates the parameter existed in a Contact: header parameter. The Oracle USM uses this convention to reconstruct the parameters' placement in the original Contact: header.

NAPTR Record Format

Oracle USMs learn that an associated TXT record exists for the from a NAPTR lookup for the queried SIP URI.

In the DDNS update, the Oracle USM indicates an associated TXT record by setting the NAPTR resource record with:

- m in the flags field (in addition to the u flag)
- E2U+sip:contact in the service type field

When no additional metadata and no parameters need to stored in ENUM server, the flags field contains u.

TXT Record Retrieval

A Oracle USM performs a separate query to the ENUM server for the TXT metadata records when it detects the 'm' flag in a NAPTR result (and the one-query-txt-naptr option is not configured). You can set the one-query-txt-naptr option to enabled in the enum-config configuration element to force the Oracle USM to request the TXT and NAPTR records both in one query from the ENUM server.

Once the Oracle USM receives the metadata stored in the TXT records, it restores the header and URI parameters that were present on the original registration.

Requirements

BIND 9.8.1-P1 or later hosting the ENUM server supports this feature.

SIP User Parts - RFC 3261 Character Set Support

RFC 3261 specifies the range of characters allowed in a user-part, all of which are supported by the Oracle USM. ENUM databases, however, do not support all these characters.

By default, the Oracle USM presents SIP messages to your DNS server unchanged, assuming that your deployment uses telephone numbers only as URI user parts. You configure the Oracle USM to support the entire range of characters allowed in user parts by enabling it on your ENUM server configuration. Upon configuration, the Oracle USM grooms the user part appropriately for the ENUM server.

Encoding Alpha-Numerics

If your deployment uses non-numeric characters in user parts, you can explicitly enable the Oracle USM to change the user part to be compatible with the ENUM database. In these cases, the Oracle USM encodes SIP message user parts using a proprietary encoding. The Oracle USM provides the encoded string to the DNS server, which then creates the applicable record(s). The Oracle USM decodes these strings as necessary during subsequent interactions with the DNS server and the UAs.

In some cases, deployments include SIP messaging in your environment that does not require encoding. These cases include messages with SIP URI user parts composed of:

- tel-uri

- sip-uri with all numeric characters
- sip-uri with all numeric characters except for a leading +

In these cases, the Oracle USM does not encode these user parts. In addition, for all-numeric user parts preceded by the + character, the Oracle USM strips the + character, reverses the digits, separates the digits with periods, and sends the message to the ENUM server with the user part unencoded.

Multiple DNS Zone Support

For smaller deployments, one can assume a single DNS zone within which all applicable UEs reside. Large deployments, however, may use multiple DNS zones, allowing the network administrator to segregate and more easily manage large numbers of UEs.

When the Oracle USM encodes the user part, it performs a modulo 1000 operation on a 24-bit checksum of the user part and appends the resulting 3 digits to the encoded user part. You can configure your DNS zones for these digits to organize users into zones.

For example, encoding the user part `acme_user` gives the string `acmeX5Fuser.2.7.7`. Assuming a domain of `acme-ims.com`, the ENUM entry for this user would be `acmeX5Fuser.2.7.7.acme-ims.com`. Applicable zone configuration can use the `.2.7.7` portion of this string to determine this user's zone.

Alpha-Numeric Name Support

To enable to use of alpha characters in SIP message user parts for a given sip-registrar:

1. From superuser mode, use the following command sequence to access sip-registrar element.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# enum-config
ORACLE(enum-config)# select
```

Select the enum-config you want to configure.

2. Specify alpha-numeric name support for this enum-config and specify the number of DNS zones in your deployment.

```
ORACLE(enum-config)# alpha-numeric-user-support enabled
ORACLE(enum-config)# done
```

Note that the parameter `alpha-numeric-user-support` is RTC supported.

Configuring SIP Ping OPTIONS Support

You can configure the Oracle USM to respond to SIP ping OPTIONS. This support is typically configured on an S-CSCF so it can respond to pings OPTIONS sent by a P-CSCF:

To configure an SIP Options Ping response support:

1. From superuser mode, use the following command sequence to access ping-response command on a sip-interface element.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-interface
ORACLE(sip-interface)# sel
```

2. Enable the support with the ping-response command.

```
ORACLE(http-config)# ping-response enabled
ORACLE(http-config)# done
```

`ping-response`—Enable ping-response to allow your device to respond to ping OPTIONS. For example, this feature is useful within hybrid deployment environments on a P-CSCF as a means of verifying the S-CSCF's availability. This configuration allows the S-CSCF to respond to SIP ping OPTIONS.

Local Subscriber Tables

Local Subscriber Table

A local subscriber table (LST) is an XML formatted file that contains one or more usernames associated with a hash as encrypted or plaintext. The LST is saved locally on the Oracle USM's file system.

LSTs enable a standalone Oracle USM node or high-availability (HA) pair to forego relying on an external user database. Thus the Oracle USM does not need to communicate with a server to authenticate users. This can eliminate the operational complexity of deploying a highly available credential storage system.

LST Runtime Execution

The LST is loaded on boot up when the configuration is appropriately set. Incoming messages thereafter can then be authenticated based on the credentials in the LST. If the Oracle USM can not load an LST file, three things occur:

1. The following log message is recorded at the NOTICE level:

```
LST [table-name] was not loaded - [filename] has error loading XML file
```

2. The message stated above is printed on the ACLI.
3. A 503 Response is returned to the UA that sent the initial REGISTER message to the Oracle USM.

LST Configuration

To configure the Oracle USM to use LSTs for authentication, you need to create a local subscriber table configuration element that identifies that LST. You then need to set the sip authentication profile configuration to reference that LST configuration so that when messages requiring authentication are received and processed by a sip registrar configuration element, the Oracle USM will use the identified LST for authentication.

In a local subscriber table configuration, you must define an object name, identify the specific LST filename (and path). If the filename is entered without a path, the Oracle USM looks in the default LST directory, which is /code/lst. If the LST file is located elsewhere on the Oracle USM, you must specify the filename and absolute path. For example /code/path/01302012lst.xml.

The corresponding sip authentication profile must be set to use the local subscriber table configuration element you just created. First set credential retrieval method to local, set the digest realm appropriately (this is required for authentication), and finally set the credential retrieval config parameter to the name of the local subscriber table configuration element that you just created. At this point you may save and activate your configuration.

Unencrypted passwords for each user in the table is computed with the MD5 hash function as follows:

```
MD5(username:digest-realm:password)
```

ACLI Instructions

LST Table

To configure the Oracle USM to use an LST:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE (configure) # session-router
```

3. Type local-subscriber-table and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE (session-router) # local-subscriber-table  
ORACLE (local-subscriber-table) #
```

You may now begin configuring the local subscriber table configuration element.

4. name—Enter the name of this local subscriber table configuration element that will be referenced from a SIP registrar configuration element.
5. filename—Enter the filename that describes this LST XML file. If no path is given, the Oracle USM looks in the /code/lst directory. You may provide a complete path if the file is located elsewhere.
6. secret—Enter the PSK used in encryption and decryption of the passwords in the XML file. Once saved, this value is not echoed back to the screen in plaintext format. See LST Subscriber Hash and Encryption.
7. Type done when finished.

SIP authentication profile

To configure the Oracle USM to utilize an LST, continuing from the previous step:

1. Type exit to return to the session router path.
2. Type sip-authentication-profile and press Enter.
3. Type select to choose the existing sip-authentication-profile configuration element you wish to use LST for authentication.

```
ACMESYSTEM (sip-authentication-profile) # select  
<name>:  
1: name=sipAuthSMX1 digest-realm=acme.com credential-retrieval-method=local  
selection: 1  
ACMESYSTEM (sip-authentication-profile) #
```

4. digest-realm—Enter the digest realm used for authenticating here.
5. credential-retrieval-method—Set this parameter to local to use an LST.
6. credential-retrieval-config—Enter the name of the LST configuration you just configured.
7. Type done when finished.

LST Redundancy for HA Systems

LSTs must be synchronized between redundant nodes to ensure that the standby node contains identical LST files. You can either SFTP the same LST file to both the active and standby node, or you can use the synchronize command. The synchronize command is always executed from the active system. It copies the specified file from the active to the standby node placing the copy in the same file location on the standby node. Use the synchronize lst command as follows:

```
ACMESYSTEM# synchronize lst file.xml
```




Note: The synchronize command does not reload the LST files.

Reloading the LST

After copying a new LST file to the Oracle USM (and its standby peer), you can reload this newer file from the ACLI using the refresh lst command. For example:

```
ORACLE# refresh lst <local-subscriber-table name>
```

Using the refresh lst command selects the LST by name to refresh. Alternatively, saving and activating the configuration will reload the configuration as well and should be used when configuration parameters have also changed.

 **Note:** In an HA pair of Oracle USMs, you must independently execute the refresh lst command on both the active and standby systems.

LST File Compression

To save local disk flash space, you can compress the LST XML file using .gz compression. The resultant file must then have an .xml.gz extension.

LST File Format

The LST file format is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<localSubscriberTable encrypt-algo="aes-128-cbc">
<subscriber username="alice@apkt.com" hash="02:5E:78:D8:7E:75:A3:39"
encrypted="true"/>
<subscriber username="bob@apkt.com" hash="bc4b2a76b9719d911017c59"
encrypted="false"/>
<subscriber username="acme@apkt.com" hash="5d41402abc4b2a76b9719d9"
encrypted="false"/>
</localSubscriberTable>
```

The LST file's elements are as follows:

localSubscriberTable

This is the head element in the XML file. Each file can have only one head element. The following attribute is found in this element:

- **encrypt-algo**—This indicates the algorithm type used to encrypt the hash in the XML file. The key for this encryption will be a preshared key and is configurable in the local subscriber table configuration element with the secret parameter.
- The value in this element is for display purposes only.
- Currently AES-128-CBC is the only supported encryption algorithm.

subscriber

This element has the subscriber information. And has the following 3 attributes:

- **username**—The value given in the username attribute must be same as the username that will be sent in the Authorization Header in the Request message from the users. Refer RFC 2617 Http Authentication for details.
- **hash**—The hash provided in the XML must be an MD5 hash of the Username, digest-realm and the password of the user. This is same as the H(A1) described in RFC 2617.

$$\text{hash} = \text{md5}(\text{username}:\text{digest-realm}:\text{password})$$
- **encrypted**—The encrypted flag indicates if the "hash" given in the XML file is encrypted or not

LST Subscriber Hash and Encryption

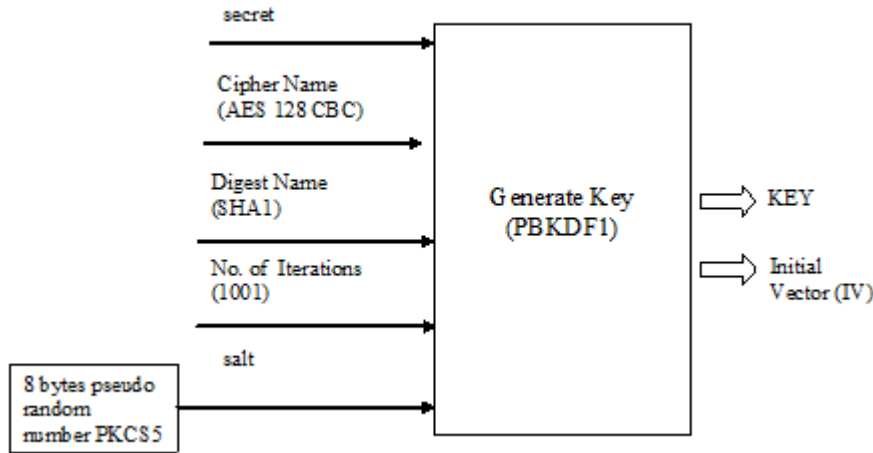
You may additionally use AES-128 CBC to encrypt the hash in the subscriber element in the LST XML file. The PSK used for encryption is configured in the secret parameter and an 8-byte pseudo random number is used as the salt. The LST file must set the encrypted attribute per subscriber element to true. To derive the final encrypted data you place

Local Subscriber Tables

in the XML file, three steps are performed according to the following blocks. The output of the last step, Formatting final Encrypted Data, is inserted into the LST files, subscriber element's hash value, when the encrypted attribute is set to true.

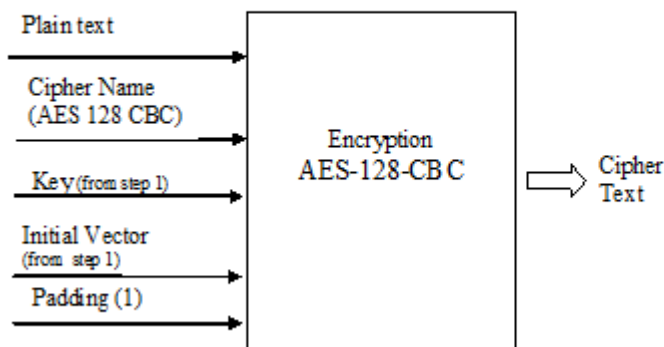
Key Initialization Vector

STEP 1: Key /IV Generation



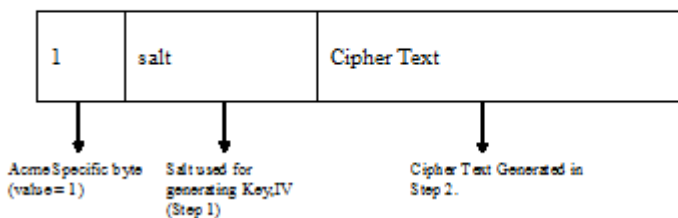
Encryption

STEP 2: Encryption



Formatting final Encrypted Data

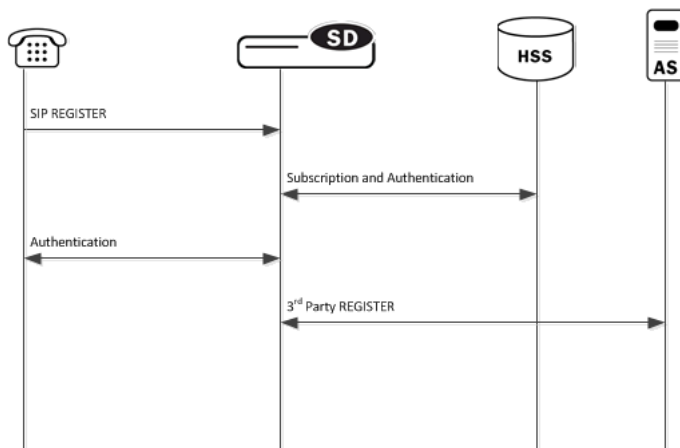
STEP 3: Final Encrypted Data



Third Party Registration

Third party registration support on the Oracle USM provides a mechanism for sending registration information to a third party server. An IM (Instant Messaging) server might be the recipient of a third party REGISTER message.

The Oracle USM accepts incoming REGISTER requests from UAs. After the UA has been registered with the Oracle USM, the Oracle USM sends a third party REGISTER message to a third party server.



The Oracle USM supports third party registration via two methods:

- For scenarios in which UAs receive iFCs from the HSS and the Oracle USM's default iFC configuration, the Oracle USM generates third party registration requests and responses for matching triggers in its iFC evaluation.

Some third party servers may want the UA's entire original request to the Oracle USM and response from the Oracle USM to the UA provided to them. The Oracle USM supports these scenarios, in some cases requiring additional configuration.

- For scenarios in which the UA needs a third party registration that is not explicitly prescribed within iFCs, you can configure a third party server on the Oracle USM and achieve third party registration support.

For these configurations, the Oracle USM attempts third party registration to those servers for all UAs that register via the applicable Oracle USM registrar.

For both methodologies, you must configure all third party servers as session agents.

Third Party Registrations via iFCs

The Oracle USM performs third party registrations based on the iFC downloaded for the user. If the filter criteria successfully evaluates to a third party server, a third party registration entry is dynamically added in the Oracle USM. The dynamic entry is automatically deleted if there are no more registrations being handled for that third party registration host.

When third party registration is performed by iFCs, the Oracle USM generates the registration messages as follows:

- The Contact: header is populated with the URI from the home server route configuration of the sip-registrar associated with the registration. If the home server route is left blank, the Oracle USM uses the IP address of the egress interface.
- The From: header of the new REGISTER message is the same as the FROM in the original message.
- The To: header of the new REGISTER message is the the same as the TO in original message (AOR).

Embedded REGISTER

As an option within standard iFC third party registration support, the Oracle USM supports 3GPP's methodology of embedding the original UE registration (and/or its response from the S-CSCF/Registrar) as a MIME body in the third party REGISTER sent from the S-CSCF to the third party server. This methodology, presented in 3GPP TS 23.218 and 29.228, uses an optional iFC extension ("IncludeRegisterRequest" and "IncludeRegisterResponse") that tells the third party server to expect the entire original REGISTER request and/or REGISTER 200OK in the mime of the third party REGISTER.

Implementation details for this methodology include the following:

- There may be further configuration required on the Oracle USM.
- The Oracle USM does not embed original registration requests or responses to any third party server outside its trust domain.
- The HSS or configured iFCs must be preconfigured for embedded third party registrations.

An HSS configuration may not support the optional "IncludeRegisterRequest" and "IncludeRegisterResponse". For these cases, there is a Oracle USM configuration option that allows you to control this inclusion, as follows:

- If the iFCs specify inclusion in an environment where you do not want it, you can set a registrar option to never include the original REGISTER
- If the iFCs do not specify inclusion in an environment where you want it, you can set a registrar option to always include the original REGISTER.

You can set these options for either the third party register, the 200 OK, or both.

ACLI Instructions - Third Party Registration via iFCs

Session Agent

To create a session agent to represent the third party server:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type session-agent and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# session-agent  
ORACLE(session-agent)#
```

4. hostname—Enter the name for this session agent.

5. ip-address—Enter the IP address for this session agent. This value must be the same as the registrar-host parameter in the third party regs configuration element to which this session agent definition corresponds.
Continue configuring this session agent’s parameters. Not all session agent functionality is applicable to the Oracle USM.
6. Type done when finished.

SIP Registrar

Option to set the SIP Registrar to perform embedded REGISTRATION support for third party registration:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type sip-registrar and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# sip-registrar
ORACLE(sip-registrar)#
```

4. Type select and choose the number of the pre-configured SIP registrar configuration element you want to configure.

```
ORACLE(sip-registrar)# select
name:
1: registrar1
selection:1
ACMEPACKET(sip-registrar)#
```

5. option +include-register-request—Set this option to control SIP REGISTER embedding in the third party registration.

```
ORACLE(sip-registrar)#options +include-register-request=true
```

Set this option to true to always embed the original REGISTER in the third party registration.

In some cases, the include may already be specified by the iFCs, even though you do not want it used. In these cases, configure the option to false

```
ORACLE(sip-registrar)#options +include-register-request=false
```

6. option +include-register-response—Set this option to control SIP REGISTER 200 OK embedding in the third party registration the S-CSCF sends to the AS.

```
ORACLE(sip-registrar)#options +include-register-response=true
```

Set this option to true to always embed the original REGISTER in the third party registration 200 OK.

In some cases, the include may already be specified by the iFCs, even though you do not want it used. In these cases, configure the option to false.

```
ACMEPACKET(sip-registrar)#options +include-register-response=false
```

7. Type done when finished.

Third Party Registration via ACLI Configuration

This section specifies the differences between Oracle USM third party registration support via iFC as opposed to via ACLI configuration.

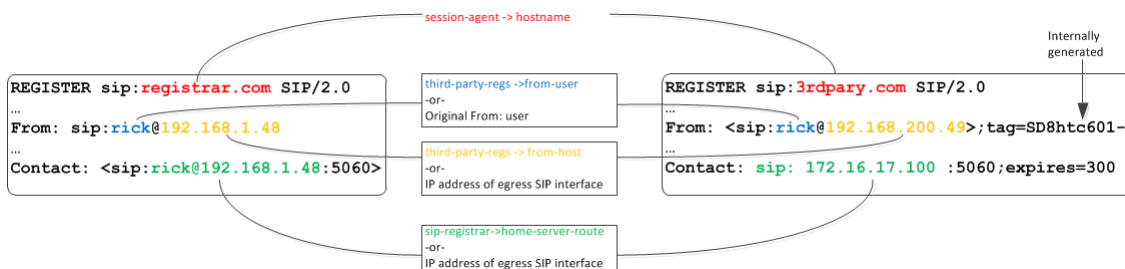
As is true of the method described above, third party registration is generated by the Oracle USM on behalf of the user in the To: header of REGISTER request.

Third Party Registration

When third party registration is generated by CLI configuration on the Oracle USM, the registration messages are generated as follows:

- The request URI of the new REGISTER message uses the value of the hostname parameter in the session agent configuration element.
- The From: header of the new REGISTER message uses the value of the from-user parameter in the third party regs configuration element as the user portion of the URI. If the from-user parameter is left blank, the Oracle USM uses the user in the original From: header.
- The From: header of the new REGISTER message uses the value of the from-host parameter in the third party regs configuration element as the host portion of the URI. If the from-host parameter is left blank, the Oracle USM uses the IP address of the egress SIP interface as the host portion of the from header.
- The Contact: header of the new REGISTER message uses the home server route parameter in the sip registrar configuration element. If the home server route parameter is left blank, the Oracle USM uses the IP address of the egress interface.

See the following diagram:



Third Party Registration Server States

If the third party server does not respond to a REGISTER request, the Oracle USM adheres to standard SIP session agent retransmission/ timeout procedures. If the third party server is set to out of service, the Oracle USM attempts connectivity retry procedures. The retry procedures dictate that the Oracle USM periodically send a REGISTER message to the third party server to check if connectivity has come back. The time interval for checking connectivity to a third party server is set with the retry interval parameter. Retries continue forever or until the third party server responds. The retry mechanism may be disabled by setting the retry interval parameter to 0.

Note: When using the CLI generated third party registration method, the time interval for checking connectivity to a third party server is set with the retry interval parameter in the third party regs configuration element.

When a third party server is out of service, the Oracle USM maintains a queue of outstanding third party registration requests. When the third party server returns to service, the Oracle USM gracefully flushes the queue of outstanding requests. This prevents a registration flood from being directed at the third party server.

Third Party Registration Expiration


The REGISTER message sent from the Oracle USM to the third party server uses the Expires: value returned from the User Subscriber Database or HSS. The third party server sends a 200 OK message containing Contact bindings and an expires value chosen by the third party server itself. The Oracle USM checks each contact address to determine if it created it. For those addresses it created (as SD-Contacts), the Expires value from the 200 OK is used as the final value.

Once the expires timer has reached half the expires period as returned from the third party server, the Oracle USM refreshes the registration.

If the third party server responds to a REGISTER Request with a 423 (Interval Too Brief) response, the Oracle USM updates the contact's expiration interval to the Min-Expires value of the 423 response. It then submits a new REGISTER Request with the updated expires value.

Defining Third Party Servers

To send third party registrations that are generated via ACLI configuration to a third party server, three configuration elements are required. The primary configuration element is the third party regs. One or more may be configured in order to send the REGISTER message to multiple registration servers. You need to configure a name and set the state to enabled. The registrar host must be configured to indicate the value to insert into the Oracle USM-generated request URI in the REGISTER message.

 **Note:** It is recommended that the list of third party registration servers be restricted to a maximum of 3.

A session agent needs to represent the third party server. Create a session agent as the third party server and note its name. Next, configure the registrar-host parameter with a session agent hostname in the third-party-reg configuration element. This specifies the session agent to be used as the registrar.

Finally, the address of the third party server must be added to the third-party-registrars parameter in the sip-registrar configuration element. This does not supercede any core Oracle USM Registrar functionality. It informs the Oracle USM of the third party server to send messages to after initial registration. Thus the value configured here must exist in the third-party-regs configuration element's registrar-host parameter list.

ACLI Instructions - Third Party Server Configuration

Recall that the configuration below is only required for scenarios in which the iFC does not explicitly specify registration for the servers you configure below.

Third Party Registrar

To configure a third party server:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE(configure)# session-router
```

3. Type third-party-regs and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# third-party-regs
ACMEPACKET(third-party-regs)#
```

4. state—Set this to enabled to use this configuration.
5. registrar-host—Set this value to the complementary session agents' hostname parameter to include those session agents as third party servers. This parameter may be modified like an options parameter. This value also appears in the request URI of the outgoing REGISTER message being sent to the third party server.
6. from-user—Configure this parameter to be the user portion of the From: header of the outgoing REGISTER message being sent to the third party server. Leaving this blank sets the user portion that in the original From: header
7. from-host—Configure this parameter to be the host portion of the From: header of the outgoing REGISTER message being sent to the third party server. Leaving this blank sets the host portion to the Oracle USM's egress SIP interface.
8. retry-interval—Enter the number of seconds the Oracle USM waits before retrying a third party server after a failed registration. Enter 0 to disable this feature.
9. Type done when finished.

SIP Registrar

To indicate to a local SIP Registrar when and what third party server to send third party registrations to:

1. In Superuser mode, type configure terminal and press Enter.

Third Party Registration

```
ORACLE# configure terminal
```

2. Type session-router and press Enter to access the session router path.

```
ORACLE (configure) # session-router
```

3. Type sip-registrar and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE (session-router) # sip-registrar  
ACMEPACKET (sip-registrar) #
```

4. Type select and choose the number of the pre-configured SIP registrar configuration element you want to configure.

```
ORACLE (sip-registrar) # select  
name:  
1: registrar1  
selection:1  
ACMEPACKET (sip-registrar) #
```

5. home-server-route—Enter the value to insert into the REGISTER message's request URI as sent to the third party server. Leaving this blank uses the AoR (or To: header) in the original REGISTER message.
6. third-party-registrars—Enter the name of a third party regs configuration element registrar-host parameter to send third part registrations associated with that SIP registrar.
7. Type done when finished.

RADIUS Accounting of REGISTERs

CDR Generation for REGISTER Events

The Oracle USM can generate RADIUS CDRs, per Contact's event, for registration, refresh registration, and registration removal. A single REGISTER message can generate multiple RADIUS CDRs since that message may contain multiple contacts.

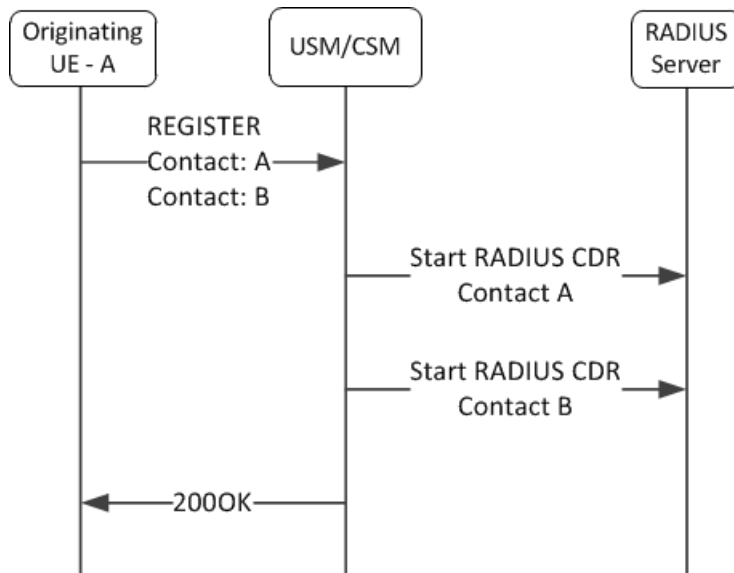
To enable CDR generation for received REGISTERs, you set the generate event parameter in the account-config configuration element to register or local-register. The register value may exist with other events such as invite.

REGISTER Scenarios

RADIUS CDRs are generated for each registration change per Contact. There are 5 main scenarios which cover CDR creation.

Initial REGISTER

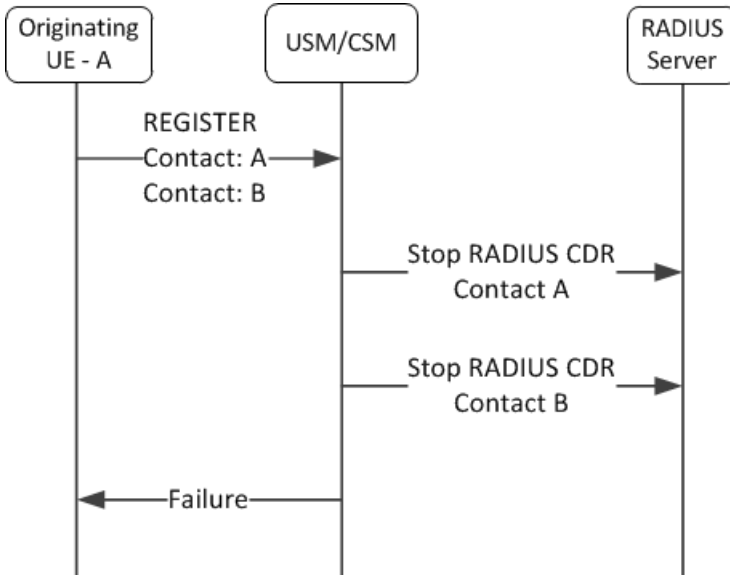
One or more RADIUS Start CDRs are sent to the RADIUS server for each contact in a successful REGISTER message before the Oracle USM replies to the endpoint with a 200 OK.



RADIUS Accounting of REGISTERS

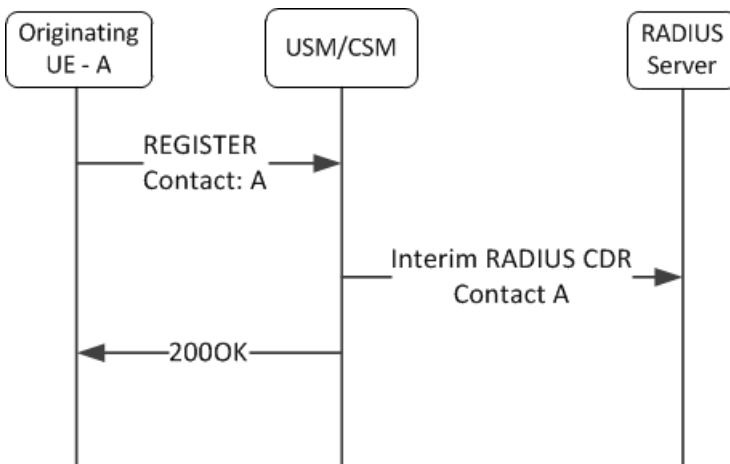
Failed REGISTER

One or more RADIUS Stop CDRs are sent to the RADIUS server for each contact in an unsuccessful REGISTER message before the Oracle USM replies to the endpoint with a SIP Final Response (4xx or 5xx) message.



REGISTER Refresh

One or more RADIUS Interim CDRs are sent to the RADIUS server for each contact in a successful reREGISTER message before the Oracle USM replies to the endpoint with a 200 OK. This happens when a database query is made and succeeds.

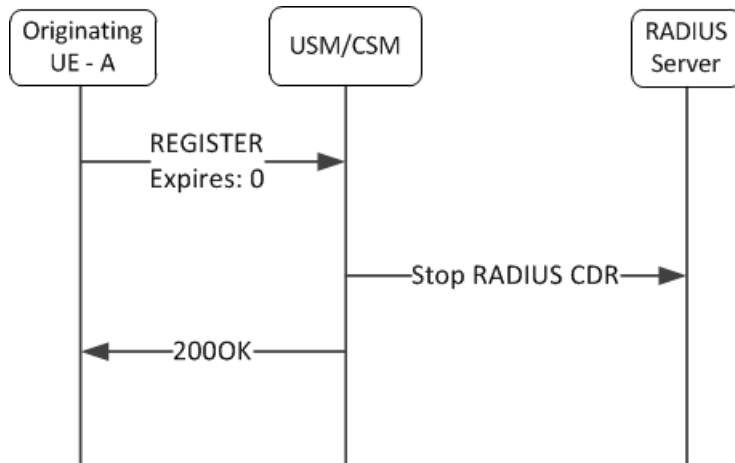


Failed REGISTER Refresh

One or more RADIUS Interim CDRs are sent to the RADIUS server for each contact in an unsuccessful reREGISTER message before the Oracle USM replies to the endpoint with a SIP Final Response (4xx or 5xx) message.

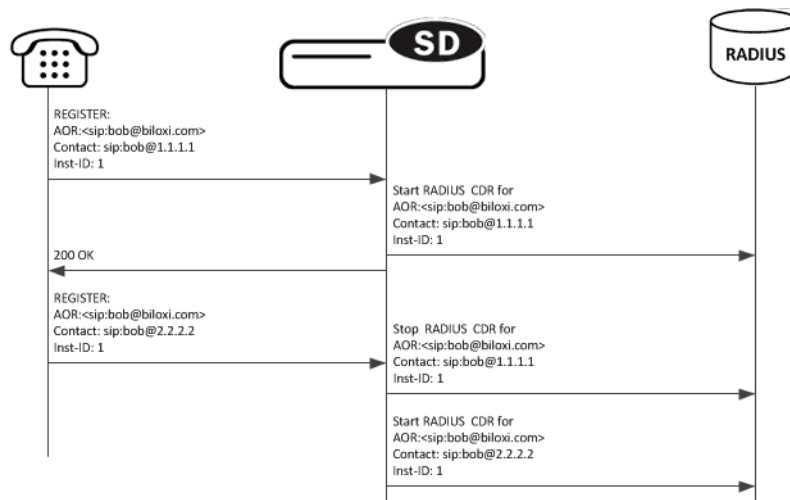
deREGISTER

One or more RADIUS Stop CDRs are sent to the RADIUS server for the contacts in a deREGISTER message before the Oracle USM replies to the endpoint with a 200 OK. The Oracle USM interprets an expires=0 parameter in a Contact: header as only removing the registration (and sending a corresponding stop record) for that contact, or an Expires: 0 header prompts Stop RADIUS records for all contacts.



Registration Update

For each Contact’s registration update with an existing Instance-ID and AoR, the Oracle USM sequentially sends a RADIUS Stop CDR for the original contact address and then a RADIUS Start CDR for the new contact address.

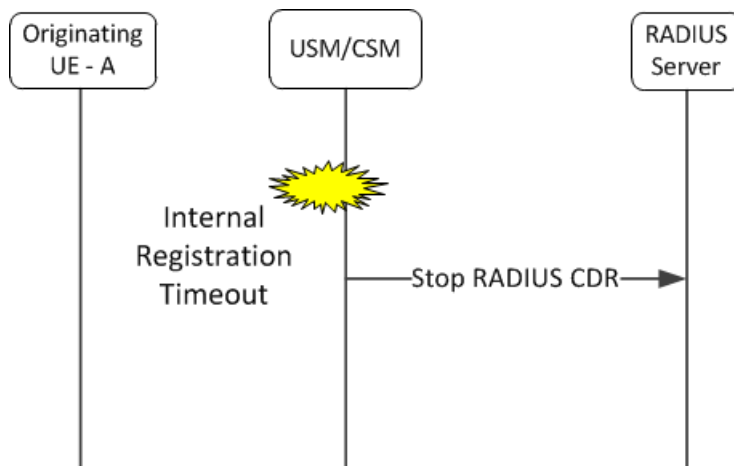


Similarly, when the Oracle USM receives a REGISTER for an existing Contact and AoR but with a different Instance-ID, the registration is updated by the Oracle USM. The same corresponding stop and start CDRs are sent to the RADIUS server.

Internal Deregistration

When a condition occurring within the Oracle USM causes a contact’s registration to be removed, a RADIUS Stop CDR is sent to the RADIUS server. In such a case, generally no indication is sent to the UA. Examples are if the registration times out, or if the contact is manually removed at the ACLI.

RADIUS Accounting of REGISTERS



REGISTER VSA Format

The following table lists new VSAs introduced with this feature.

Parameter Name	Dictionary Index	Data Type	Valid Records	Definition
Acme-SIP-Method-Type	235	String	Start, Interim, Stop	This is the SIP method type that is associated to the CDR. Possible values are INVITE, BYE, or REGISTER.
Acme-Domain-Name	236	String	Start, Interim, Stop	This is domain name of the request URI.
Acme-SIP-Contact	237	String	Start, Interim, Stop	This is contact from SIP message. This is not the entire contact header.
Acme-SIP-Expires	238	Integer	Start, Interim	This is the expires value of the Expires: header from the sip message.
Acme-Reason-Phrase	239	String	Interim, Stop	This is the Oracle USM reason code. This will not be set for all Stop and Interim

The following table lists the new definitions of existing VSAs when CDRs are created on REGISTER messages.

Parameter Name	Dictionary Index	Data Type	Valid Records	Definition
Acme-SIP-Status	71	String	Interim	
Acct-Session-Id	44	String	Start, Interim, Stop	This is the a unique string assigned to the contact. The string is made up of the system name concatenated with a timestamp and an 8 digit hex number. It is of form <system name>-<timestamp>-<number>.
Called-Station-Id		String	Start, Interim, Stop	When in a stop record, this value is populated when an internal reason

Parameter Name	Dictionary Index	Data Type	Valid Records	Definition
				causes the stop record to be generated. in this case it is the AoR.

The following pairs of Acme-Disconnect-Initiator and Acme-Disconnect-Cause VSAs, as presented in a RADIUS stop CDR that corresponds to a REGISTER message are defined in the Reason column.

Reason	Acme-Disconnect-Initiator	Acme-Disconnect-Cause
UA requested deregistration	1 - CALLING_PARTY_DISCONNECT	1-PW_CAUSE_USER_REQUEST
Contact Times Out	3 - INTERNAL_DISCONNECT	4 -PW_CAUSE_IDLE_TIMEOUT
REGISTER error on establishment.	1 - CALLING_PARTY_DISCONNECT	17-PW - CAUSE_USER_ERROR
RTR from HSS specifies user's private ID	3 - INTERNAL_DISCONNECT	20 - PW_CAUSE_RTR_REQUEST_PRIVATE
RTR from HSS specifies user's public ID	3 - INTERNAL_DISCONNECT	21 - PW_CAUSE_RTR_REQUEST_PUBLIC
Contact's registration is removed via ACLI command	3 - INTERNAL_DISCONNECT	1-PW_CAUSE_USER_REQUEST
Reuse of ID by a different Contact	3 - INTERNAL_DISCONNECT	22-PW_CAUSE_REUSE_ID

CDR Generation Configuration

To add CDR generation on REGISTER messages:

1. In Superuser mode, type configure terminal and press Enter.

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

```
ORACLE(configure)# session-router
```

3. Type account-config and press Enter.

```
ORACLE(session-router)# account-config
```

From this point, you can reach the individual parameters for duplicate RADIUS attribute prevention and for RADIUS attribute selection.

4. generate-events—Set this parameter to register and/or local-register.
5. Save and activate your configuration.

Example CDRs

The following examples list when basic registrations create CDRs.

Initial Registration CDR

A Start CDR is created for an initial REGISTER received on the Oracle USM's 192.168.101.20 interface from 192.168.12.12.

RADIUS Accounting of REGISTERS

REGISTER message:

```
REGISTER sip:registrar.biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
User-Agent: Softphone Beta1.5
Authorization: Digest username="Bob", realm=" biloxi.com",
    nonce="84a4cc6f3082121f32b42a2187831a9e",
    response="7587245234b3434cc3412213e5f113a5432"
CSeq: 1826 REGISTER
Contact: <sip:bob@192.168.12.12>
Expires: 7200
Content-Length: 0
```

Start CDR Message, selected attributes:

```
NAS-Identifier = "abc123"
Acct-Status-Type = Start
NAS-IP-Address = 172.16.101.20
NAS-Port = 5060
Acct-Session-Id = "Iapetus-C00000001"
Acme-Session-Ingress-CallId = "843817637684230@998sdasdh09"
Acme-Session-Protocol-Type = "SIP"
Calling-Station-Id = " Bob <sip:bob@biloxi.com>;tag=456248"
Called-Station-Id = " Bob <sip:bob@biloxi.com>"
Acme-Ingress-Network-Interface-Id = "M00"
Acme-Ingress-Vlan-Tag-Value = 0
Acme-Session-Ingress-Realm = "net192"
Acme-Firmware-Version = "SCX6.3.3 F-1 GA (WS Build 18)"
Acme-Local-Time-Zone = "Time Zone Not Set"
Acme-Ingress-Local-Addr = "192.168.101.20:5060"
Acme-Ingress-Remote-Addr = "192.168.12.12:5060"
Acme-SIP-Method-Type = "REGISTER"
Acme-Domain-Name = "registrar.biloxi.com"
Acme-SIP-Contact = "sip:bob@192.168.12.12"
Acme-SIP-Expires = 7200
Acme-CDR-Sequence-Number = 1
Client-IP-Address = 172.30.70.121
Acct-Unique-Session-Id = "51a15d4381d9fe38"
Timestamp = 1329241213
```

Interim Registration CDR

An interim CDR is created for a REGISTER refresh received on the Oracle USM's 192.168.101.20 interface from 192.168.12.12.

REGISTER message:

```
REGISTER sip:registrar.biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
User-Agent: Softphone Beta1.5
Authorization: Digest username="Bob", realm=" biloxi.com",
    nonce="84a4cc6f3082121f32b42a2187831a9e",
    response="7587245234b3434cc3412213e5f113a5432"
CSeq: 1826 REGISTER
Contact: <sip:bob@192.168.12.12>
Expires: 7200
Content-Length: 0
```

Interim CDR Message, selected attributes:

```
NAS-Identifier = "abc123"
  Acct-Status-Type = Interim-Update
  NAS-IP-Address = 172.16.101.20
  NAS-Port = 5060
  Acct-Session-Id = " Iapetus-C00000001"
  Acme-Session-Ingress-CallId = "843817637684230@998sdasdh09"
  Acme-Session-Protocol-Type = "SIP"
  Calling-Station-Id = " Bob <sip:bob@biloxi.com>;tag=456248"
  Called-Station-Id = " Bob <sip:bob@biloxi.com>"
  Acme-Ingress-Network-Interface-Id = "M00"
  Acme-Ingress-Vlan-Tag-Value = 0
  Acme-Session-Ingress-Realm = "net192"
  Acme-Firmware-Version = "SCX6.3.3 F-1 GA (WS Build 18)"
  Acme-Local-Time-Zone = "Time Zone Not Set"
  Acme-Ingress-Local-Addr = "192.168.101.20:5060"
  Acme-Ingress-Remote-Addr = "192.168.12.12:5060"
Acme-SIP-Method-Type = "REGISTER"
Acme-Domain-Name = "registrar.biloxi.com"
Acme-SIP-Contact = "sip:bob@192.168.12.12"
Acme-SIP-Expires = 7200
Acme-SIP-Status = "200"
Acme-Reason-Phrase = "OK"
  Acme-CDR-Sequence-Number = 1
  Client-IP-Address = 172.30.70.121
  Acct-Unique-Session-Id = "51a15d4381d9fe38"
  Timestamp = 1329241213
```

STOP CDR on REGISTER message

Stop CDRs are generated for REGISTER messages with Expires of 0 from a user agent, a failed initial registration, or Oracle USM removing the registration. A Stop CDR can take one of two forms:

1. The Stop CDR is generated as part of receiving a response to a REGISTER message.
2. The Stop CDR is generated as part of an internal event such as a Contact timing out.

REGISTER message:

The following REGISTER contains an expires of 0 received on interface 192.168.101.20 from 192.168.12.12:

```
REGISTER sip:registrar.biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
User-Agent: Softphone Beta1.5
Authorization: Digest username="Bob", realm=" biloxi.com",
  nonce="84a4cc6f3082121f32b42a2187831a9e",
  response="7587245234b3434cc3412213e5f113a5432"
CSeq: 1826 REGISTER
Contact: <sip:bob@192.168.12.12>
Expires: 0
Content-Length: 0
```

Stop CDR Message, selected attributes:

```
NAS-Identifier = "abc123"
  Acct-Status-Type = Stop
  NAS-IP-Address = 172.16.101.20
  NAS-Port = 5060
  Acct-Session-Id = " Iapetus-C00000001"
  Acme-Session-Ingress-CallId = "843817637684230@998sdasdh09"
  Acme-Session-Protocol-Type = "SIP"
  Calling-Station-Id = " Bob <sip:bob@biloxi.com>;tag=456248"
```

RADIUS Accounting of REGISTERS

```
Called-Station-Id = " Bob <sip:bob@biloxi.com>"
Acme-Ingress-Network-Interface-Id = "M00"
Acme-Ingress-Vlan-Tag-Value = 0
Acme-Session-Ingress-Realm = "net192"
Acme-Firmware-Version = "SCX6.3.3 F-1 GA (WS Build 18)"
Acme-Local-Time-Zone = "Time Zone Not Set"
Acme-Ingress-Local-Addr = "192.168.101.20:5060"
Acme-Ingress-Remote-Addr = "192.168.12.12:5060"
Acme-Disconnect-Initiator = 1
  Acme-Disconnect-Cause = 1
Acme-SIP-Status = "200"
Acme-SIP-Method-Type = "REGISTER"
Acme-Domain-Name = "registrar.biloxi.com"
Acme-SIP-Contact = "sip:bob@192.168.12.12"
Acme-Reason-Phrase = "OK"
Acme-CDR-Sequence-Number = 1
  Client-IP-Address = 172.30.70.121
  Acct-Unique-Session-Id = "51a15d4381d9fe38"
  Timestamp = 1329241213
```

Oracle USM Initiated Deregistration

Since no SIP Result code is returned to an endpoint when it was internally deregistered by the Oracle USM, its corresponding VSA is not created and will not appear in the CDR.

No REGISTER message received.

Stop CDR Message, selected attributes:

```
NAS-Identifier = "abc123"
  Acct-Status-Type = Stop
  NAS-IP-Address = 172.16.101.20
  NAS-Port = 5060
  Acct-Session-Id = " Iapetus-C00000001"
  Called-Station-Id = "sip:bob@biloxi.com"
  Acme-Session-Protocol-Type = "SIP"
Acme-Disconnect-Initiator = 3
  Acme-Disconnect-Cause = 4
Acme-SIP-Method-Type = "REGISTER"
Acme-Domain-Name = "registrar.biloxi.com"
Acme-SIP-Contact = "sip:bob@192.168.12.12"
  Acme-CDR-Sequence-Number = 1
  Client-IP-Address = 172.30.70.121
  Acct-Unique-Session-Id = "51a15d4381d9fe38"
  Timestamp = 1329241213
```

Local CDR CSV Orientation

This section lists the order of VSAs (and other statistics) in local CDR files.

Start Record

CSV Placement	Attribute Name	ACME VSA ID
1	Acct-Status-Type	
2	NAS-IP-Address	
3	NAS-Port	
4	Acct-Session-Id	
5	Acme-Session-Ingress-CallId	3

CSV Placement	Attribute Name	ACME VSA ID
6	Acme-Session--Egress-CallId	4
7	Acme-Session-Protocol-Type	43
8	Acme-Session-Forked-Call-Id	171
9	Acme-Session--Generic-Id	40
10	Calling-Station-Id	
11	Called-Station-Id	
12	h323-setup-time	
13	h323-connect-time	
14	Acme-Egress-Network-Interface-Id	139
15	Acme-Egress-Vlan-Tag-Value	140
16	Acme-Ingress-Network-Interface-Id	137
17	Acme-Ingress-Vlan-Tag-Value	138
18	Acme-Session-Egress-Realm	42
19	Acme-Session-Ingress-Realm	41
20	Acme-FlowId_FS1_F	1
21	Acme-FlowType_FS1_F	2
22	Acme-Flow-In-Realm_FS1_F	10
23	Acme-Flow-In-Src-Addr_FS1_F	11
24	Acme-Flow-In-Src-Port_FS1_F	12
25	Acme-Flow-In-Dst-Addr_FS1_F	13
26	Acme-Flow-In-Dst-Port_FS1_F	14
27	Acme-Flow-Out-Realm_FS1_F	20
28	Acme-Flow-Out-Src-Addr_FS1_F	21
29	Acme-Flow-Out-Src-Port_FS1_F	22
30	Acme-Flow-Out-Dst-Addr_FS1_F	23
31	Acme-Flow-Out-Dst-Port_FS1_F	24
32	Acme-FlowID_FS1_R	78
33	Acme-FlowType_FS1_R	79
34	Acme-Flow-In-Realm_FS1_R	80
35	Acme-Flow-In-Src-Addr_FS1_R	81
36	Acme-Flow-In-Src-Port_FS1_R	82
37	Acme-Flow-In-Dst-Addr_FS1_R	83
38	Acme-Flow-In-Dst-Port_FS1_R	84
39	Acme-Flow-Out-Realm_FS1_R	85

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
40	Acme-Flow-Out-Src-Addr_FS1_R	86
41	Acme-Flow-Out-Src-Port_FS1_R	87
42	Acme-Flow-Out-Dst-Addr_FS1_R	88
43	Acme-Flow-Out-Dst-Port_FS1_R	89
44	Acme-FlowID_FS2_F	90
45	Acme-FlowType_FS2_F	91
46	Acme-Flow-In-REALM_FS2_F	92
47	Acme-Flow-In-Src-Addr_FS2_F	93
48	Acme-Flow-In-Src-Port_FS2_F	94
49	Acme-Flow-In-Dst-Addr_FS2_F	95
50	Acme-Flow-In-Dst-Port_FS2_F	96
51	Acme-Flow-Out-REALM_FS2_F	97
52	Acme-Flow-Out-Src-Addr_FS2_F	98
53	Acme-Flow-Out-Src-Port_FS2_F	99
54	Acme-Flow-Out-Dst-Addr_FS2_F	100
55	Acme-Flow-Out-Dst-Port_FS2_F	101
56	Acme-FlowID_FS2_R	112
57	Acme-FlowType_FS2_R	113
58	Acme-Flow-In-REALM_FS2_R	114
59	Acme-Flow-In-Src-Addr_FS2_R	115
60	Acme-Flow-In-Src-Port_FS2_R	116
61	Acme-Flow-In-Dst-Addr_FS2_R	117
62	Acme-Flow-In-Dst-Port_FS2_R	118
63	Acme-Flow-Out-REALM_FS2_R	119
64	Acme-Flow-Out-Src-Addr_FS2_R	120
65	Acme-Flow-Out-Src-Port_FS2_R	121
66	Acme-Flow-Out-Dst-Addr_FS2_R	122
67	Acme-Flow-Out-Dst-Port_FS2_R	123
68	Acme-Session-Charging-Vector	54
69	Acme-Session-Charging-Function_Address	55
70	Acme-Firmware-Version	56
71	Acme-Local-Time-Zone	57
72	Acme-Post-Dial-Delay	58
73	Acme-Primary-Routing-Number	64

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
74	Acme-Originating-Trunk-Group	65
75	Acme-Terminating-Trunk-Group	66
76	Acme-Originating-Trunk-Context	67
77	Acme-Terminating-Trunk-Context	68
78	Acme-P-Asserted-ID	69
79	Acme-Ingress-Local-Addr	74
80	Acme-Ingress-Remote-Addr	75
81	Acme-Egress-Local-Addr	76
82	Acme-Egress-Remote-Addr	77
83	Acme-SIP-Diversion	70
84	Acme-Egress-Final-Routing-Number	134
85	Acme-Session-Ingress-RPH	135
86	Acme-Session-Egress-RPH	136
87	Acme-Custom-VSA-200	200
88	Acme-Custom-VSA-201	201
89	Acme-Custom-VSA-202	202
90	Acme-Custom-VSA-203	203
91	Acme-Custom-VSA-204	204
92	Acme-Custom-VSA-205	205
93	Acme-Custom-VSA-206	206
94	Acme-Custom-VSA-207	207
95	Acme-Custom-VSA-208	208
96	Acme-Custom-VSA-209	209
97	Acme-Custom-VSA-210	210
98	Acme-Custom-VSA-211	211
99	Acme-Custom-VSA-212	212
100	Acme-Custom-VSA-213	213
101	Acme-Custom-VSA-214	214
102	Acme-Custom-VSA-215	215
103	Acme-Custom-VSA-216	216
104	Acme-Custom-VSA-217	217
105	Acme-Custom-VSA-218	218
106	Acme-Custom-VSA-219	219
107	Acme-Custom-VSA-220	220

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
108	Acme-Custom-VSA-221	221
109	Acme-Custom-VSA-222	222
110	Acme-Custom-VSA-223	223
111	Acme-Custom-VSA-224	224
112	Acme-Custom-VSA-225	225
113	Acme-Custom-VSA-226	226
114	Acme-Custom-VSA-227	227
115	Acme-Custom-VSA-228	228
116	Acme-Custom-VSA-229	229
117	Acme-Custom-VSA-230	230
118	Acme-Flow-Calling-Media-Stop-Time_FS1	231
119	Acme-Flow-Called-Media-Stop-Time_FS1	232
120	Acme-Flow-Calling-Media-Stop-Time_FS2	233
121	Acme-Flow-Called-Media-Stop-Time_FS2	234
122	Acme-FlowMediaType_FS1_F	142
123	Acme-FlowMediaType_FS1_R	143
124	Acme-FlowMediaType_FS2_F	144
125	Acme-FlowMediaType_FS2_R	145
126	Acme-SIP-Method-Type	235
127	Acme-Domain-Name	236
128	Acme-SIP-Contact	237
129	Acme-SIP-Expires	238
130	Acme-CDR-Sequence-Number	59

Interim Record

CSV Placement	Attribute Name	ACME VSA ID
1	Acct-Status-Type	
2	NAS-IP-Address	
3	NAS-Port	
4	Acct-Session-Id	
5	Acme-Session-Ingress-CallId	3
6	Acme-Session--Egress-CallId	4
7	Acme-Session-Protocol-Type	43
9	Acme-Session-Forked-Call-Id	171

CSV Placement	Attribute Name	ACME VSA ID
8	Acme-Session--Generic-Id	40
10	Calling-Station-Id	
11	Called-Station-Id	
12	h323-setup-time	
13	h323-connect-time	
14	Acme-Egress-Network-Interface-Id	139
15	Acme-Egress-Vlan-Tag-Value	140
16	Acme-Ingress-Network-Interface-Id	137
17	Acme-Ingress-Vlan-Tag-Value	138
18	Acme-Session-Egress-Realm	42
19	Acme-Session-Ingress-Realm	41
20	Acme-FlowId_FS1_F	1
21	Acme-FlowType_FS1_F	2
22	Acme-Flow-In-Realm_FS1_F	10
23	Acme-Flow-In-Src-Addr_FS1_F	11
24	Acme-Flow-In-Src-Port_FS1_F	12
25	Acme-Flow-In-Dst-Addr_FS1_F	13
26	Acme-Flow-In-Dst-Port_FS1_F	14
27	Acme-Flow-Out-Realm_FS1_F	20
28	Acme-Flow-Out-Src-Addr_FS1_F	21
29	Acme-Flow-Out-Src-Port_FS1_F	22
30	Acme-Flow-Out-Dst-Addr_FS1_F	23
31	Acme-Flow-Out-Dst-Port_FS1_F	24
32	Acme-Calling-RTCP-Packets-Lost_FS1	32
33	Acme-Calling-RTCP-Avg-Jitter_FS1	33
34	Acme-Calling-RTCP-Avg-Latency_FS1	34
35	Acme-Calling-RTCP-MaxJitter_FS1	35
36	Acme-Calling-RTCP-MaxLatency_FS1	36
37	Acme-Calling-RTP-Packets-Lost_FS1	37
38	Acme-Calling-RTP-Avg-Jitter_FS1	38
39	Acme-Calling-RTP-MaxJitter_FS1	39
40	Acme-Calling-Octets_FS1	28
41	Acme-Calling-Packets_FS1	29
42	Acme-Calling-R-Factor	151

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
43	Acme-Calling-MOS	152
44	Acme-FlowID_FS1_R	78
45	Acme-FlowType_FS1_R	79
46	Acme-Flow-In-Realm_FS1_R	80
47	Acme-Flow-In-Src-Addr_FS1_R	81
48	Acme-Flow-In-Src-Port_FS1_R	82
49	Acme-Flow-In-Dst-Addr_FS1_R	83
50	Acme-Flow-In-Dst-Port_FS1_R	84
51	Acme-Flow-Out-Realm_FS1_R	85
52	Acme-Flow-Out-Src-Addr_FS1_R	86
53	Acme-Flow-Out-Src-Port_FS1_R	87
54	Acme-Flow-Out-Dst-Addr_FS1_R	88
55	Acme-Flow-Out-Dst-Port_FS1_R	89
56	Acme-Called-RTCP-Packets-Lost_FS1	46
57	Acme-Called-RTCP-Avg-Jitter_FS1	47
58	Acme-Called-RTCP-Avg-Latency_FS1	48
59	Acme-Called-RTCP-MaxJitter_FS1	49
60	Acme-Called-RTCP-MaxLatency_FS1	50
61	Acme-Called-RTP-Packets-Lost_FS1	51
62	Acme-Called-RTP-Avg-Jitter_FS1	52
63	Acme-Called-RTP-MaxJitter_FS1	53
64	Acme-Called-Octets_FS1	44
65	Acme-Called-Packets_FS1	45
66	Acme-Called-R-Factor	153
67	Acme-Called-MOS	154
68	Acme-FlowID_FS2_F	90
69	Acme-FlowType_FS2_F	91
70	Acme-Flow-In-Realm_FS2_F	92
71	Acme-Flow-In-Src-Addr_FS2_F	93
72	Acme-Flow-In-Src-Port_FS2_F	94
73	Acme-Flow-In-Dst-Addr_FS2_F	95
74	Acme-Flow-In-Dst-Port_FS2_F	96
75	Acme-Flow-Out-Realm_FS2_F	97
76	Acme-Flow-Out-Src-Addr_FS2_F	98

CSV Placement	Attribute Name	ACME VSA ID
77	Acme-Flow-Out-Src-Port_FS2_F	99
78	Acme-Flow-Out-Dst-Addr_FS2_F	100
79	Acme-Flow-Out-Dst-Port_FS2_F	101
80	Acme-Calling-RTCP-Packets-Lost_FS2	104
81	Acme-Calling-RTCP-Avg-Jitter_FS2	105
82	Acme-Calling-RTCP-Avg-Latency_FS2	106
83	Acme-Calling-RTCP-MaxJitter_FS2	107
84	Acme-Calling-RTCP-MaxLatency_FS2	108
85	Acme-Calling-RTP-Packets-Lost_FS2	109
86	Acme-Calling-RTP-Avg-Jitter_FS2	110
87	Acme-Calling-RTP-MaxJitter_FS2	111
88	Acme-Calling-Octets_FS2	102
89	Acme-Calling-Packets_FS2	103
90	Acme-FlowID_FS2_R	112
91	Acme-FlowType_FS2_R	113
92	Acme-Flow-In-Realm_FS2_R	114
93	Acme-Flow-In-Src-Addr_FS2_R	115
94	Acme-Flow-In-Src-Port_FS2_R	116
95	Acme-Flow-In-Dst-Addr_FS2_R	117
96	Acme-Flow-In-Dst-Port_FS2_R	118
97	Acme-Flow-Out-Realm_FS2_R	119
98	Acme-Flow-Out-Src-Addr_FS2_R	120
99	Acme-Flow-Out-Src-Port_FS2_R	121
100	Acme-Flow-Out-Dst-Addr_FS2_R	122
101	Acme-Flow-Out-Dst-Port_FS2_R	123
102	Acme-Called-RTCP-Packets-Lost_FS2	126
103	Acme-Called--RTCP-Avg-Jitter_FS2	127
104	Acme-Called--RTCP-Avg-Latency_FS2	128
105	Acme-Called--RTCP-MaxJitter_FS2	129
106	Acme-Called-RTCP-MaxLatency_FS2	130
107	Acme-Called-RTP-Packets-Lost_FS2	131
108	Acme-Called-RTP-Avg-Jitter_FS2	132
109	Acme-Called-RTP-MaxJitter_FS2	133
110	Acme-Called-Octets_FS2	124

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
111	Acme-Called-Packets_FS2	125
112	Acme-Session-Charging-Vector	54
113	Acme-Session-Charging-Function_Address	55
114	Acme-Firmware-Version	56
115	Acme-Local-Time-Zone	57
116	Acme-Post-Dial-Delay	58
117	Acme-Primary-Routing-Number	64
118	Acme-Originating-Trunk-Group	65
119	Acme-Terminating-Trunk-Group	66
120	Acme-Originating-Trunk-Context	67
121	Acme-Terminating-Trunk-Context	68
122	Acme-P-Asserted-ID	69
123	Acme-Ingress-Local-Addr	74
124	Acme-Ingress-Remote-Addr	75
125	Acme-Egress-Local-Addr	76
126	Acme-Egress-Remote-Addr	77
127	Acme-SIP-Diversion	70
128	Acme-Intermediate_Time	63
129	Acct-Session-Time	
130	Acme-Egress-Final-Routing-Number	134
131	Acme-Session-Ingress-RPH	135
132	Acme-Session-Egress-RPH	136
133	Acme-Custom-VSA-200	200
134	Acme-Custom-VSA-201	201
135	Acme-Custom-VSA-202	202
136	Acme-Custom-VSA-203	203
137	Acme-Custom-VSA-204	204
138	Acme-Custom-VSA-205	205
139	Acme-Custom-VSA-206	206
140	Acme-Custom-VSA-207	207
141	Acme-Custom-VSA-208	208
142	Acme-Custom-VSA-209	209
143	Acme-Custom-VSA-210	210
144	Acme-Custom-VSA-211	211

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
145	Acme-Custom-VSA-212	212
146	Acme-Custom-VSA-213	213
147	Acme-Custom-VSA-214	214
148	Acme-Custom-VSA-215	215
149	Acme-Custom-VSA-216	216
150	Acme-Custom-VSA-217	217
151	Acme-Custom-VSA-218	218
152	Acme-Custom-VSA-219	219
153	Acme-Custom-VSA-220	220
154	Acme-Custom-VSA-221	221
155	Acme-Custom-VSA-222	222
156	Acme-Custom-VSA-223	223
157	Acme-Custom-VSA-224	224
158	Acme-Custom-VSA-225	225
159	Acme-Custom-VSA-226	226
160	Acme-Custom-VSA-227	227
161	Acme-Custom-VSA-228	228
162	Acme-Custom-VSA-229	229
163	Acme-Custom-VSA-230	230
164	Acme-Flow-Calling-Media-Stop-Time_FS1	231
165	Acme-Flow-Called-Media-Stop-Time_FS1	232
166	Acme-Flow-Calling-Media-Stop-Time_FS2	233
167	Acme-Flow-Called-Media-Stop-Time_FS2	234
168	Acme-FlowMediaType_FS1_F	142
169	Acme-FlowMediaType_FS1_R	143
170	Acme-FlowMediaType_FS2_F	144
171	Acme-FlowMediaType_FS2_R	145
172	Acme-SIP-Method-Type	235
173	Acme-Domain-Name	236
174	Acme-SIP-Contact	237
175	Acme-SIP-Expires	238
176	Acme-SIP-Status	71
177	Acme-Reason-Phrase	239
178	Acme-CDR-Sequence-Number	59

Stop Record

CSV Placement	Attribute Name	ACME VSA ID
1	Acct-Status-Type	
2	NAS-IP-Address	
3	NAS-Port	
4	Acct-Session-Id	
5	Acme-Session-Ingress-CallId	3
6	Acme-Session--Egress-CallId	4
7	Acme-Session-Protocol-Type	43
8	Acme-Session-Forked-Call-Id	171
9	Acme-Session--Generic-Id	40
10	Calling-Station-Id	
11	Called-Station-Id	
12	Acct-Terminate-Cause	
13	Acct-Session-Time	
14	h323-setup-time	
15	h323-connect-time	
16	h323-disconnect-time	
17	h323-disconnect-cause	
18	Acme-Egress-Network-Interface-Id	139
19	Acme-Egress-Vlan-Tag-Value	140
20	Acme-Ingress-Network-Interface-Id	137
21	Acme-Ingress-Vlan-Tag-Value	138
22	Acme-Session-Egress-Realm	42
23	Acme-Session-Ingress-Realm	41
24	Acme-FlowId_FS1_F	1
25	Acme-FlowType_FS1_F	2
26	Acme-Flow-In-Realm_FS1_F	10
27	Acme-Flow-In-Src-Addr_FS1_F	11
28	Acme-Flow-In-Src-Port_FS1_F	12
29	Acme-Flow-In-Dst-Addr_FS1_F	13
30	Acme-Flow-In-Dst-Port_FS1_F	14
31	Acme-Flow-Out-Realm_FS1_F	20
32	Acme-Flow-Out-Src-Addr_FS1_F	21
33	Acme-Flow-Out-Src-Port_FS1_F	22

CSV Placement	Attribute Name	ACME VSA ID
34	Acme-Flow-Out-Dst-Addr_FS1_F	23
35	Acme-Flow-Out-Dst-Port_FS1_F	24
36	Acme-Calling-RTCP-Packets-Lost_FS1	32
37	Acme-Calling-RTCP-Avg-Jitter_FS1	33
38	Acme-Calling-RTCP-Avg-Latency_FS1	34
39	Acme-Calling-RTCP-MaxJitter_FS1	35
40	Acme-Calling-RTCP-MaxLatency_FS1	36
41	Acme-Calling-RTP-Packets-Lost_FS1	37
42	Acme-Calling-RTP-Avg-Jitter_FS1	38
43	Acme-Calling-RTP-MaxJitter_FS1	39
44	Acme-Calling-Octets_FS1	28
45	Acme-Calling-Packets_FS1	29
46	Acme-Calling-R-Factor	151
47	Acme-Calling-MOS	152
48	Acme-FlowID_FS1_R	78
49	Acme-FlowType_FS1_R	79
50	Acme-Flow-In-Realm_FS1_R	80
51	Acme-Flow-In-Src-Addr_FS1_R	81
52	Acme-Flow-In-Src-Port_FS1_R	82
53	Acme-Flow-In-Dst-Addr_FS1_R	83
54	Acme-Flow-In-Dst-Port_FS1_R	84
55	Acme-Flow-Out-Realm_FS1_R	85
56	Acme-Flow-Out-Src-Addr_FS1_R	86
57	Acme-Flow-Out-Src-Port_FS1_R	87
58	Acme-Flow-Out-Dst-Addr_FS1_R	88
59	Acme-Flow-Out-Dst-Port_FS1_R	89
60	Acme-Called-RTCP-Packets-Lost_FS1	46
61	Acme-Called-RTCP-Avg-Jitter_FS1	47
62	Acme-Called-RTCP-Avg-Latency_FS1	48
63	Acme-Called-RTCP-MaxJitter_FS1	49
64	Acme-Called-RTCP-MaxLatency_FS1	50
65	Acme-Called-RTP-Packets-Lost_FS1	51
66	Acme-Called-RTP-Avg-Jitter_FS1	52
67	Acme-Called-RTP-MaxJitter_FS1	53

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
68	Acme-Called-Octets_FS1	44
69	Acme-Called-Packets_FS1	45
70	Acme-Called-R-Factor	153
71	Acme-Called-MOS	154
72	Acme-FlowID_FS2_F	90
73	Acme-FlowType_FS2_F	91
74	Acme-Flow-In-REALM_FS2_F	92
75	Acme-Flow-In-Src-Addr_FS2_F	93
76	Acme-Flow-In-Src-Port_FS2_F	94
77	Acme-Flow-In-Dst-Addr_FS2_F	95
78	Acme-Flow-In-Dst-Port_FS2_F	96
79	Acme-Flow-Out-REALM_FS2_F	97
80	Acme-Flow-Out-Src-Addr_FS2_F	98
81	Acme-Flow-Out-Src-Port_FS2_F	99
82	Acme-Flow-Out-Dst-Addr_FS2_F	100
83	Acme-Flow-Out-Dst-Port_FS2_F	101
84	Acme-Calling-RTCP-Packets-Lost_FS2	104
85	Acme-Calling-RTCP-Avg-Jitter_FS2	105
86	Acme-Calling-RTCP-Avg-Latency_FS2	106
87	Acme-Calling-RTCP-MaxJitter_FS2	107
88	Acme-Calling-RTCP-MaxLatency_FS2	108
89	Acme-Calling-RTP-Packets-Lost_FS2	109
90	Acme-Calling-RTP-Avg-Jitter_FS2	110
91	Acme-Calling-RTP-MaxJitter_FS2	111
92	Acme-Calling-Octets_FS2	102
93	Acme-Calling-Packets_FS2	103
94	Acme-FlowID_FS2_R	112
95	Acme-FlowType_FS2_R	113
96	Acme-Flow-In-REALM_FS2_R	114
97	Acme-Flow-In-Src-Addr_FS2_R	115
98	Acme-Flow-In-Src-Port_FS2_R	116
99	Acme-Flow-In-Dst-Addr_FS2_R	117
100	Acme-Flow-In-Dst-Port_FS2_R	118
101	Acme-Flow-Out-REALM_FS2_R	119

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
102	Acme-Flow-Out-Src-Addr_FS2_R	120
103	Acme-Flow-Out-Src-Port_FS2_R	121
104	Acme-Flow-Out-Dst-Addr_FS2_R	122
105	Acme-Flow-Out-Dst-Port_FS2_R	123
106	Acme-Called-RTCP-Packets-Lost_FS2	126
107	Acme-Called--RTCP-Avg-Jitter_FS2	127
108	Acme-Called--RTCP-Avg-Latency_FS2	128
109	Acme-Called--RTCP-MaxJitter_FS2	129
110	Acme-Called-RTCP-MaxLatency_FS2	130
111	Acme-Called-RTP-Packets-Lost_FS2	131
112	Acme-Called-RTP-Avg-Jitter_FS2	132
113	Acme-Called-RTP-MaxJitter_FS2	133
114	Acme-Called-Octets_FS2	124
115	Acme-Called-Packets_FS2	125
116	Acme-Session-Charging-Vector	54
117	Acme-Session-Charging-Function-Address	55
118	Acme-Firmware-Version	56
119	Acme-Local-Time-Zone	57
120	Acme-Post-Dial-Delay	58
121	Acme-Primary-Routing-Number	64
122	Acme-Originating-Trunk-Group	65
123	Acme-Terminating-Trunk-Group	66
124	Acme-Originating-Trunk-Context	67
125	Acme-Terminating-Trunk-Context	68
126	Acme-P-Asserted-ID	69
127	Acme-Ingress-Local-Addr	74
128	Acme-Ingress-Remote-Addr	75
129	Acme-Egress-Local-Addr	76
130	Acme-Egress-Remote-Addr	77
131	Acme-SIP-Diversion	70
132	Acme-Session-Disposition	60
133	Acme-Disconnect-Initiator	61
134	Acme-Disconnect-Cause	62
135	Acme-SIP-Status	71

RADIUS Accounting of REGISTERS

CSV Placement	Attribute Name	ACME VSA ID
136	Acme-Egress-Final-Routing-Number	134
137	Acme-Session-Ingress-RPH	135
138	Acme-Session-Egress-RPH	136
139	Acme-Refer-Call-Transfer-Id	141
140	Acme-Custom-VSA-200	200
141	Acme-Custom-VSA-201	201
142	Acme-Custom-VSA-202	202
143	Acme-Custom-VSA-203	203
144	Acme-Custom-VSA-204	204
145	Acme-Custom-VSA-205	205
146	Acme-Custom-VSA-206	206
147	Acme-Custom-VSA-207	207
148	Acme-Custom-VSA-208	208
149	Acme-Custom-VSA-209	209
150	Acme-Custom-VSA-210	210
151	Acme-Custom-VSA-211	211
152	Acme-Custom-VSA-212	212
153	Acme-Custom-VSA-213	213
154	Acme-Custom-VSA-214	214
155	Acme-Custom-VSA-215	215
156	Acme-Custom-VSA-216	216
157	Acme-Custom-VSA-217	217
158	Acme-Custom-VSA-218	218
159	Acme-Custom-VSA-219	219
160	Acme-Custom-VSA-220	220
161	Acme-Custom-VSA-221	221
162	Acme-Custom-VSA-222	222
163	Acme-Custom-VSA-223	223
164	Acme-Custom-VSA-224	224
165	Acme-Custom-VSA-225	225
166	Acme-Custom-VSA-226	226
167	Acme-Custom-VSA-227	227
168	Acme-Custom-VSA-228	228
169	Acme-Custom-VSA-229	229

RADIUS Accounting of REGISTERs

CSV Placement	Attribute Name	ACME VSA ID
170	Acme-Custom-VSA-230	230
171	Acme-Flow-Calling-Media-Stop-Time_FS1	231
172	Acme-Flow-Called-Media-Stop-Time_FS1	232
173	Acme-Flow-Calling-Media-Stop-Time_FS2	233
174	Acme-Flow-Called-Media-Stop-Time_FS2	234
175	Acme-FlowMediaType_FS1_F	142
176	Acme-FlowMediaType_FS1_R	143
177	Acme-FlowMediaType_FS2_F	144
178	Acme-FlowMediaType_FS2_R	145
179	Acme-SIP-Method-Type	235
180	Acme-Domain-Name	236
181	Acme-SIP-Contact	237
182	Acme-Reason-Phrase	239
183	Acme-CDR-Sequence-Number	59

References and Debugging

ACLI Configuration Elements

The following sections describe the Oracle USM's unique configuration elements.

sip-registrar

Parameters

name—Configured name of this sip registrar.

- Default: empty

state—Running status of this policy-director-group.

- Default: enabled
- Values: enabled | disabled

domains—List of registration domains that this Oracle USM is responsible for. * means all domains. These domains are compared for an exact match with the domain in the request-uri of the REGISTER message. the wildcard '*' can also be entered as part of this parameter. This is entered as the domains separated by a space in quotes. No quotes required if only one domain is being configured. "+" and "-" are used to add to subtract from the list.

- Default: empty

subscriber-database-method—Protocol used to connect to User Subscriber Database server.

- Default: CX
- Values: CX | DDNS | local

subscriber-database-config—The configuration element that defines the server used for retrieving user subscriber data. For Cx deployments it is a home-subscriber-server name. For ENUM deployments it is an enum-config name.

- Default: empty

authentication-profile—Name of the sip-authentication-profile configuration used to retrieve authentication data when an endpoint is not authenticated.

- Default: empty

References and Debugging

home-server-route—The value inserted into the Server Name AVP in an MAR message. This should be entered as a SIP URI as per 3gpp TS 24229 & RFC 3261. The host can be FQDN or IPv4 address, and the port portion should be in the 1025 - 65535 range. Examples: SIP:12.12.12.12:5060

- Default: empty

third-party-registrars—The third-party-reg configuration element names where third party REGISTER messages will be forwarded to.

- Default: empty

routing-precedence—Indicates whether INVITE routing lookup should use the user database (via the registrar configuration element) or perform local policy lookup immediately.

- Default: registrar
- Values: registrar | local-policy

egress-realm-id—Indicates the default egress/core realm for SIP messaging.

- Default: empty

location-update-interval—Sets the maximum period in minutes in which the core-side user subscriber database is refreshed, per user.

- Default: 1440
- Values: 0-999999999

ifc-profile—References the ifc-profile configuration element's name that is applied to this sip-registrar.

max-contacts-per-aor—Limit to the number of contacts allowed for a given AOR.

- Default: 0 (disabled)
- Values: 1 - 256

Path

This sip-registrar configuration element is a element in the session-router path. The full path from the topmost ACLI prompt is: **configure terminal > session-router > sip-registrar**.

sip-authentication-profile

Parameters

name—Configured name of this sip-authentication profile.

methods—List of SIP methods that prompt authentication. This is entered as the methods separated by a space in quotes. No quotes required if only one method is being configured. "+" and "-" are used to add to subtract from the list.

- Default: empty

anonymous-methods—List of SIP methods that prompt authentication when received from anonymous sources. This is entered as the methods separated by a space in quotes. No quotes required if only one method is being configured. "+" and "-" are used to add or subtract from the list.

- Default: empty

digest-realm—The value inserted into the digest-realm parameter in an authentication challenge header as sent to UA. (not used for Cx deployments)

- Default: empty

credential-retrieval-method—Protocol used to connect to the server providing authentication data.

- Default: ENUM-TXT

- Values: ENUM-TXT | CX

credential-retrieval-config—The home-subscriber-server name used for retrieving authentication data.

- Default: empty

Path

This sip-authentication-profile configuration element is a element in the session-router path. The full path from the topmost ACLI prompt is: **configure terminal > session-router > sip-authentication-profile**.

home-subscriber-server

Parameters

name—Configured name of this home subscriber server.

- Default: empty

state—Running status of this home subscriber server.

- Default: enabled
- Values: enabled | disabled

transport— The layer 4 protocol used to communicate with this home subscriber server.

- Default: tcp
- Values: tcp | sctp

address—This home subscriber server's IP address.

- Default: none
- Values: IP address in dotted decimal notation

port—This home subscriber server's port.

- Default: 80
- Values: 1-65535

realm—Oracle USM realm-config name where this home subscriber server exists.

- Default: none

multi-homed-addr— Specifies one or more local secondary addresses of the SCTP endpoint. This setting is only applicable to SCTP transport. To enter multiple addresses, bracket an address list with parentheses. At least one address is required if transport is set to SCTP.

Multi-homed addresses must be of the same type (IPv4 or IPv6) as that specified by the address parameter. Like the address parameter, these addresses identify SD physical interfaces.

origin-host-identifier—Used to create segment before the dot in the Origin Host AVP.

- Default: none

origin-realm—Populates the value of the Origin Realm AVP. Populates the segment after the dot in the Origin Host AVP.

- Default: none

destination-host-identifier—Used to create segment before the dot in the Destination Host AVP.

- Default: none

watchdog-ka-timer— The interval in seconds of the watchdog/keep-alive messages.

- Default: 0

References and Debugging

- Values: 0-65535

num-auth-vector—The number of authentication vectors downloaded from the HSS per MAR.

- Default: 3
- Values: 1-10

Path

This home-subscriber-server configuration element is a element in the session-router path. The full path from the topmost CLI prompt is: **configure terminal > session-router > home-subscriber-server**.

third-party-regs

Parameters

state—Running status of this third party registration configuration element.

- Default: enabled
- Values: enabled | disabled

name—Configured name of this third party registration configuration element.

- Default: none

registrar-host—hostname of the configured session agent that will be third party server. This value is also used in the request-uri that is sent to the third party server.

- Default: none

from-user—The user part of the From URI in the REGISTER Request that is sent to the third party server in the REGISTER message. When this parameter is blank the user part of the From header from the incoming REGISTER Request will be used.

- Default: none

from-host—The host part of the From URI in the REGISTER Request that is sent the third party server in the REGISTER message. When this parameter is blank the Oracle USM uses the egress hostname/ IP address as the host.

- Default: none
- Values: Format this the same as the "registrar-host" in sip-config.

retry-interval—number of seconds the Oracle USM waits before retrying a 3rd Party Registration server after a failed registration.

- Default: 32
- Values: 0 - 3600

Path

This third-party-regs configuration element is a element in the session-router path. The full path from the topmost CLI prompt is: **configure terminal > session-router > third-party-regs**.

local-subscriber-table

Parameters

name—A given name for this local subscriber table element. This name is referenced from the sip-registrar configuration element when the credential-retrieval-method is set to local.

filename—The filename of local subscriber table that this element references. If no path is provided, the default location is /code/lst.

secret—PSK used for encrypted passwords. This value is not echoed back to the screen upon viewing the configuration element.

Path

The location of this configuration element is: `configure terminal > session-router > local-subscriber-table`.

enum-config

Parameters

name—Name for this enum-config to be referenced from within the system.

top-level-domain—The domain extension used to query the ENUM servers for this configuration.

realm-id—The realm-id is used to determine on which network interface to issue an ENUM query.

enum-servers—List of IP address that service the top level domain.

service-type—The ENUM service types you want supported in this ENUM configuration. Possible entries are E2U +sip and sip+E2U (the default), and the types outlines in RFCs 2916 and 3721.

- Default: E2U+sip,sip+E2U

query-method—the ENUM query distribution strategy

- Default: hunt
- Values: hunt | round-robin

timeout—The total time, in seconds, that should elapse before a query sent to a server (and its retransmissions) will timeout.

- Default: 11

cacheInactivityTimer—Enter the time interval, in seconds, after which you want cache entries created by ENUM requests deleted, if inactive for this interval.

- Default: 3600
- Values: 0-999999999

max-response-size—The maximum size in bytes for UDP datagram responses

- Defaults: 512

health-query-number—The phone number for the ENUM server health query; when this parameter is blank the feature is disabled.

health-query-interval—The interval in seconds at which you want to query ENUM server health.

- Default: 0
- Values: 0-65535

failover-to—Name of the enum-config to which you want to failover.

cache-addl-records—Set this parameter to enabled to add additional records received in an ENUM query to the local DNS cache.

- Default: enabled
- Values: enabled | disabled

include-source-info—Set this parameter to enabled to send source URI information to the ENUM server with any ENUM queries.

References and Debugging

- Default: disabled
- Values: enabled | disabled

ttl—This value sets the TTL value (in seconds) for NAPTR entries in the local ENUM cache and populates when sending a NAPTR entry to the ENUM server.

- Default: 0
- Values: 1-2592000

order—This parameter value populates the order field with when sending NAPTR entries to the ENUM server.

- Default: 1
- Values: 0-65535

preference—This parameter value populates the preference field with when sending NAPTR entries to the ENUM server.

- Default: 1
- Values: 0-65535

Path

This enum-config configuration element is a element in the session-router path. The full path from the topmost ACLI prompt is: **configure terminal > session-router > enum-config**.

ifc-profile

Parameters

name—A given name for this ifc profile element. This name is referenced from the sip-registrar configuration element's ifc-support parameter.

state—Running status of this ifc-profile.

- Default: enabled
- Values: enabled | disabled

shared-ifc-filename—The name of the file referenced for shared iFC function.

default-ifc-filename—The name of the file referenced for default iFC function. This file may be the same as that used for the shared iFC function.

Path

The location of this configuration element is: **configure terminal > session-router > ifc-profile**.

regevent-notification-profile

Parameters

name—A given name for this registration event notification profile element. This name is referenced from the sip-registrar configuration element.

min-subscription-duration—The amount of time, in seconds, before the subscription expires, unless it is refreshed.

- Default: 3761 seconds
- Values: 180-6000005 seconds

Path

The location of this configuration element is: **configure terminal > session-router > regevent-notification-profile**.

hss-group

Parameters

name—Enter the name of the hss-group element. This required entry must follow the Name Format, and it must be unique.

state—Enable or disable the hss-group element.

- Default: enabled
- Values: enabled | disabled

origin-host-identifier—Set this to a string for use in constructing a unique Origin Host AVP.

strategy—Select the HSS allocation options for the hss-group. Strategies determine how HSSs will be chosen by this hss-group element.

- Default: hunt
- Values:
 - **hunt**—Selects HSSs in the order in which they are listed. For example, if the first server is online, all traffic is sent to the first server. If the first server is offline, the second server is selected. If the first and second servers are offline, the third server is selected. When the Oracle USM detects that a higher priority HSS is back in service, it routes all subsequent traffic to that HSS.
 - **roundrobin**—Selects each HSS in the order in which they are listed in the dest list, selecting each HSS in turn, one per session. After all HSSs have been used, the first HSS is used again and the cycle continues.
 - **failover**—Selects the first sever in the list until failure is detected. Subsequent signaling goes to the next server in the list.

hss-configs—Identify the home-subscriber-servers available for use by this hss-group. This list can contain as many home subscriber servers as is necessary. An hss-config list value must correspond to a valid hss-group name in another group or to a valid hostname of a configured home-subscriber-server.

A value you enter here must correspond to a valid group name for a configured home-subscriber-server or a valid hostname or IP address for a configured home-subscriber-server.

hss-group is an element under the session-router path. The full path from the topmost CLI prompt is: **configure terminal > session-router > session-group**.

SNMP MIBs and Traps

The following MIBs and traps are supported for the Oracle USM. Please consult the Net-Net 4000 S-CX6.3.0 MIB Reference Guide for more SNMP information.

Acme Packet License MIB (ap-license.mib)

The following table describes the SNMP GET query names for the Oracle License MIB (ap-license.mib).

SNMP GET Query Name	Object Identifier Name: Number	Description
	Object Identifier Name: apLicenseEntry (1.3.6.1.4.1.9148.3.5.1.1.1)	

References and Debugging

SNMP GET Query Name	Object Identifier Name: Number	Description
apLicenseAuthFeature	apLicenseEntry: 1.3.6.1.4.1.9148.3.5.1.1.1.20	If authorization and authentication is allowed for the Oracle USM, the value is true. If disabled, the value is false.
apLicenseDatabaseRegFeature	apLicenseEntry: 1.3.6.1.4.1.9148.3.5.1.1.1.21	If the Oracle USM is configured as a registrar, the value is true. If registrar functionality is not enabled, this value is false.
apLicenseDatabaseRegCap	apLicenseEntry: 1.3.6.1.4.1.9148.3.5.1.1.1.22	The database registration contact capacity.

Acme Packet System Management MIB (ap-smgmt.mib)

The following table describes the SNMP GET query names for the Oracle System Management MIB (ap-smgmt.mib).

SNMP GET Query Name	Object Identifier Name: Number	Description
Object Identifier Name: apSysMgmtMIBObjects (1.3.6.1.4.1.9148.3.2.1)		
Object Identifier Name: apSysMgmtGeneralObjects (1.3.6.1.4.1.9148.3.2.1.1)		
apSysSipStatsActiveDatabaseContacts	apSysMgmtGeneralObjects: 1.3.6.1.4.1.9148.3.2.1.1.24.0	Number of database-type contacts in the registration cache.

Enterprise Traps

The following table identifies the proprietary traps that Oracle USM system supports.

Trap Name: OID	Description
apSysMgmtDatabaseRegCacheCapTrap: 1.3.6.1.4.1.9148.3.2.6.0.76	Generated when the number of database-type contacts stored in the registration cache exceeds the license threshold.
apSysMgmtDatabaseRegCacheCapClearTrap: 1.3.6.1.4.1.9148.3.2.6.0.77	Trap is generated when the number of database-type contacts stored in the registration cache falls below the license threshold.

Oracle USM Show Commands

show sipd endpoint-ip

The show sipd endpoint-ip <user | IP address> command displays information about each endpoint. For a supplied AoR, the Oracle USM displays all associated contacts (both access and core side), the expiration of each contact entry and associated 3rd Party Registration information. For example:

```
ORACLE# show sipd endpoint-ip 11111
User <sip:111111@172.16.17.100>
Contact exp=1198
  UA-Contact: <sip:111111@172.16.17.100:5060> UDP keep-acl
              realm=net172 local=172.16.101.13:5060 UA=172.16.17.100:5060
  SD-Contact: <sip:111111-s37q249kvluuaa@192.168.101.13:5060> realm=net192
Call-ID: 1-15822@172.16.17.100'
```



```
Third Party Registration:
Third Party Reg User=<sip:111111@172.16.17.100> state: REGISTERED
Expire Secs=298 seqNum= 1 refreshInterval=300
Call-ID: d355a67277d9158e7901e46a12719663@192.168.101.13
Third Party Reg User=<sip:111111@172.16.17.100> state: REGISTERED
Expire Secs=178 seqNum= 1 refreshInterval=180
Call-ID: 07ebbdebdf64a48985bb82fa8b4c595@192.168.101.13
```

show sipd third-party

The show sipd third-party command displays the current status of third party servers and statistics for messages. The format is:

```
show sipd third-party <all | name>
```

The name argument allows status to be displayed for just the server specified by the name. Not specifying a name results in status being displayed for all third party servers. For example:

```
ORACLE# show sipd third-party-reg all
3rd Party Registrar  SA State  Requests  200OK  Timeouts  Errors
192.168.17.101      INSV      9          9       0         0
192.168.17.102      INSV     14         14       0         0
```

Column definitions are as follows:

- IP Address —IP Address of third party server
- Status —Session Agent State
- Requests —Register requests sent
- 200 OK —200 OK Responses received
- Timeouts —Requests timed out
- Error —Error Responses

show sipd local-subscription

The ACLI show sipd command includes an argument that provides information about local subscriptions, as shown below.

```
ORACLE# show sipd local-subscription
19:22:18-152
SIP Local Subscription Status  -- Period --  ----- Lifetime -----
                          Active  High  Total Total  PerMax  High
Server Subscription          0    1    1    1    1    1
Message Statistics
SUBSCRIBE
----- Server -----
Message/Event  Recent  Total  PerMax  Client -----
                          Recent  Total  PerMax
SUBSCRIBE Requests      2      2      2      0      0      0
Retransmissions        0      0      0      0      0      0
200 OK                 1      1      1      0      0      0
403 Forbidden          1      1      1      0      0      0
Response Retrans       0      0      0      0      0      0
Transaction Timeouts   -      -      -      0      0      0
Locally Throttled      -      -      -      0      0      0
Avg Latency=0.000 for 0
Max Latency=0.000
NOTIFY
----- Server -----
Message/Event  Recent  Total  PerMax  Client -----
                          Recent  Total  PerMax
NOTIFY Requests      0      0      0      2      2      2
Retransmissions      0      0      0      10     10     10
200 OK              0      0      0      1      1      1
Transaction Timeouts -      -      -      0      0      0
```

References and Debugging

```
Locally Throttled          -          -          -          0          0          0
Avg Latency=0.000 for 0
Max Latency=0.000
```

You can extend upon this ACLI show sipd command to include an argument that provides information about registration event package traffic, as shown below.

```
ORACLE# show sipd local-subscription regevent
19:23:08-103
SIP Local Subscription Status -- Period -- ----- Lifetime -----
                Active High   Total Total PerMax   High
Server Subscription          0    1     1     1     1     1
Message Statistics
SUBSCRIBE
----- Server -----
Message/Event   Recent   Total   PerMax   Recent   Total   PerMax
-----
SUBSCRIBE Requests      2         2         2         0         0         0
Retransmissions         0         0         0         0         0         0
200 OK                  1         1         1         0         0         0
403 Forbidden           1         1         1         0         0         0
Response Retrans        0         0         0         0         0         0
Transaction Timeouts    -         -         -         0         0         0
Locally Throttled      -         -         -         0         0         0
Avg Latency=0.000 for 0
Max Latency=0.000
NOTIFY
----- Server -----
Message/Event   Recent   Total   PerMax   Recent   Total   PerMax
-----
NOTIFY Requests      0         0         0         2         2         2
Retransmissions         0         0         0         10        10        10
200 OK              0         0         0         1         1         1
Transaction Timeouts -         -         -         0         0         0
Locally Throttled  -         -         -         0         0         0
Avg Latency=0.000 for 0
Max Latency=0.000
```

The ACLI show registration sipd command includes an argument that provides information about a specific user's registration(s), as shown below.

```
ORACLE# show registration sipd by-user ral detailed
User: sip:ral@apkt.com
Registered at: 2013-06-05-19:23:40      Surrogate User: false
Contact Information:
Contact:
  Name: sip:ral@apkt.com
  Valid: true
  Challenged: false
  Registered at: 2013-06-05-19:23:40
  Last Registered at: 2013-06-05-19:23:40
  Expire: 3581
  Local expire: 41
  Half: 1781
  Registrar IP: 0.0.0.0
  Transport: UDP
  Secure: false
  Local IP: 192.168.101.62:5060
  User Agent Info:
    Contact: sip:ral@192.168.13.1:5060
    Realm: net192
    IP: 192.168.13.1:5060
  SD Info:
    Contact: sip:ral-1cdstqjt90hve@172.16.101.62:5060
    Realm: net172
```

```

Call-ID: 1-28361@192.168.13.1
Associated URI(s):
  URI: sip:ral@apkt.com
  Filter Criteria:
    Priority: 0
    Filter: None specified
    Application Server: sip:appserv@apkt.com
Reg Event Subscriptions Terminated locally:
  Number of Subscriptions: 1

```

Subscriber: appserv<sip:appserv@apkt.com>;tag=1 state=active exp=600114

show registration

The show registration command displays cumulative statistics on all current registrations.

```

ORACLE# show registration
15:35:43-177
SIP Registrations
-- Period -- ----- Lifetime -----
Active High Total Total PerMax High
User Entries 0 0 0 0 0 0
Local Contacts 0 0 0 0 0 0
Via Entries 0 0 0 0 0 0
AURI Entries 0 0 0 0 0 0
Free Map Ports 0 0 0 0 0 0
Used Map Ports 0 0 0 0 0 0
Forwards - - 0 0 0 0
Refreshes - - 0 0 0 0
Rejects - - 0 0 0 0
Timeouts - - 0 0 0 0
Fwd Postponed - - 0 0 0 0
Fwd Rejected - - 0 0 0 0
Refr Extension 0 0 0 0 0 0
Refresh Extended - - 0 0 0 0
ContactsPerAor Reject - - 0 0 0 0
Surrogate Regs 0 0 0 0 0 0
Surrogate Sent - - 0 0 0 0
Surrogate Reject - - 0 0 0 0
Surrogate Timeout - - 0 0 0 0
HNT Entries 0 0 0 0 0 0
Non-HNT Entries 0 0 0 0 0 0
Database Regs 0 0 0 0 0 0
DDNS Entries 0 0 0 0 0 0
CX Entries 0 0 0 0 0 0
LocalDB Entries 0 0 0 0 0 0
Unreg Users 0 0 0 0 0 0

```

You can extend upon the show registration command by adding the sipd by-user <username> detail arguments. The resulting output reflects user registration information including downloaded IFCs. For example:

```

ORACLE# show registration sipd by-user +19999092907 d
Registration Cache (Detailed View) MON JUN 25 2012 13:47:46
User: sip:+19999092907@mobile.com
Registered at: 2012-06-25-13:43:50 Surrogate User: false
Contact Information:
Contact:
  Name: sip:+19999092907@mobile.com
  Valid: true
  Challenged: false
  Registered at: 2012-06-25-13:43:50
  Last Registered at: 2012-06-25-13:47:30
  Expire: 48
  Local expire: 13
  Registrar IP: 0.0.0.0
  Transport: UDP

```

```

Secure: false
Local IP: 155.212.214.175:5060
User Agent Info:
  Contact: sip:+19999092907@50.76.51.62:5762;transport=udp;acme_nat=
+19999092907+50.76.51.62@10.1.10.20:5762
  Realm: access
  IP: 50.76.51.62:5762
SD Info:
  Contact: sip:+19999092907-rb8tulsbv3u72@108.108.108.108:5060
  Realm: core
  Call-ID: H_yvkgTAAA@10.1.10.20
Associated URI(s):
  URI: sip:+19999092907@mobile.com
Filter Criteria:
  Priority: 0
  Filter: ((case == 'Originating Registered') and (method == INVITE) and
('Accept-Contact'=='+g.app2app')) or
          ((case == 'Originating Registered') and (method == INVITE) and
('Contact'=='+g.app2app')) or
          ((case == 'Originating Registered') and (method == INVITE) and
('P-Message-Auth'=='.*')) or
          ((case == 'Originating Registered') and (method == INVITE) and
('P-Application-ID'=='.*'))
  Application Server: sip:pza.mobile.com:5280
Reg Event Subscriptions Received by Registrar:
Number of Subscriptions : 2
Subscriber: sip:appserv@192.168.13.1:5060; state=active; exp=59978
Subscriber: sip:pcscf@192.168.13.1:5060; state=active; exp=978

```

show home-subscriber-server

The show home-subscriber-server command displays cumulative statistics on all currently configured HSS servers.

```
show home-subscriber-server [stats <hss-name>] group group-name ]
```

This command allows you to gather a set of information commonly requested by the Oracle TAC when troubleshooting customers.

The show home-subscriber-server command with no arguments displays the status of each HSS as well as the number of transactions and connections per HSS. For example:

```

ORACLE# show home-subscriber-server
Name                Local-Address        Server-Address        Status
hssl                 192.168.207.21:45463 192.168.200.232:3872 Up
-----
18:53:25-105
HSS Status
Active      -- Period --  ----- Lifetime -----
Client Trans    0         1         4         12152    8         1
Server Trans    0         0         0          7         2         1
Connections     1         1         0          53        2         1

```

Note that the Connections statistic indicates the number of connections after successful CER/CEA handshake.

The table below documents the states the

Field	Description
Active	This status is related to HSS failover and load balancing configurations. The diameter connection is up and being used.
Standby	This status is related to HSS failover and load balancing configurations. The diameter connection is up, but is not being used.
Pending	The Oracle USM has sent a CER and is waiting for a CEA response.

Field	Description
Inactive	The Oracle USM has sent a CER but has not received a CEA response.
Down	The Oracle USM is not attempting to establish a connection with the HSS.

Oracle USM reports on each HSS.

The show home-subscriber-server command with the stats argument displays the number of transactions and connections per HSS as well as the number of messages exchanged with all HSS servers per message type. For example:

```
ORACLE# show home-subscriber-server stats
veloster2# show home-subscriber-server stats
Name                          Local-Address                Server-Address                Status
hss1                           192.168.207.21:45463        192.168.200.232:3872        Up
-----
18:55:03-103
HSS Status
Active      High      Total      Total      PerMax      High
Client Trans 1         1         5         12157      8          1
Server Trans 0         0         0          7          2          1
Connections  1         1         0          53         2          1
-----
----- Lifetime -----
Recent      Total      PerMax
UAR          0          3          1
  SUBSEQ_REG (2002) 0          3          1
SAR          0          6          3
  SUCCESS (2001)   0          6          3
MAR          0          4          2
  SUCCESS (2001)   0          4          2
LIR          0          1          1
  SUCCESS (2001)   0          1          1
RTR          0          1          1
  SUCCESS (2001)   0          1          1
PPR          0          1          1
  SUCCESS (2001)   0          1          1
CER          0          55         3
  SUCCESS (2001)   0          53         2
DWR          5          12088      5
  SUCCESS (2001)   4          12041     5
  ERR_TIMEOUT      0          46         1
DWR Recv     0          5          2
  SUCCESS (2001)   0          5          2
TCP Failures 0          267        6
```

By entering the name of a specific HSS as an argument, the ACLI displays all HSS data for that server only. For example:

```
ACMESYSTEM# show home-subscriber-server stats hss1
```

The show home-subscriber-server command with the group argument displays the number of transactions and connections per the HSS group you specify in the command. For example:

```
ORACLE# show home-subscriber-server group hss-group1
display grp hss-group1
HSS Status
Active      High      Total      Total      PerMax      High
Client Trans 0         0         0          0          0          0
Server Trans 0         0         0          0          0          0
Sockets      0         0         0          0          0          0
Connections  0         0         0          0          0          0
-----
----- Lifetime -----
Recent      Total      PerMax
UAR          0          0          0
```

References and Debugging

```
SAR          0          0          0
MAR          0          0          0
LIR          0          0          0
RTR          0          0          0
PPR          0          0          0
Sent Requests 0          0          0
Sent Req Accepted 0        0          0
Sent Req Rejected 0        0          0
Sent Req Expired 0          0          0
Sent Req Error 0          0          0
Recv Requests 0          0          0
Recv Req Accepted 0         0          0
Recv Req Rejected 0         0          0
HSS Errors   0          0          0
```

show http-server

The ACLI show http-server command provides basic OAuth information as shown below. The command without arguments displays basis statistics on all servers.

```
ORACLE# show http-server
Name          Server-Address          Status
sk            host.httpsrv.com       Up
sk1           192.168.19.1:8886     Up
sk2           192.168.19.1:8887     Up
sk3           192.168.19.1:8889     Up
12:56:41-184
HTTP Status   -- Period --  ----- Lifetime -----
              Active   High   Total   Total   PerMax   High
Client Trans  0         0     0       0       0       0
Server Trans  0         0     0       0       0       0
Sockets       0         0     0       0       0       0
Connections   0         0     0       0       0       0
```

You can extend upon this command to get detailed global statistics by adding the stats argument to the end of this command.

```
ORACLE# show http-server stats
Name          Server-Address          Status
sk            host.httpsrv.com       Up
sk1           192.168.19.1:8886     Up
sk2           192.168.19.1:8887     Up
sk3           192.168.19.1:8889     Up
12:56:41-184
HTTP Status   -- Period --  ----- Lifetime -----
              Active   High   Total   Total   PerMax   High
Client Trans  0         0     0       0       0       0
Server Trans  0         0     0       0       0       0
Sockets       1         1     1       1       1       1
Connections   1         1     1       1       1       1
----- Lifetime -----
              Recent   Total   PerMax
Sent Requests  0         0       0
Sent Req Accepted 0         0       0
Sent Req Rejected 0         0       0
Sent Req Expired 0         0       0
HTTP Errors    0         0       0
```

You can limit this output to a single server by appending the command with the name of that server.

```
ORACLE# show http-server stats http-server1
Name = http-server1
-----
Server-Address          Status
192.168.19.1:8886      Up
-----
```

```

12:56:41-184
HTTP Status
-- Period -- ----- Lifetime -----
Active High Total Total PerMax High
Client Trans 0 0 0 0 0 0
Server Trans 0 0 0 0 0 0
Sockets 0 0 0 0 0 0
Connections 0 0 0 0 0 0
---- Lifetime ----
Recent Total PerMax
Sent Requests 0 0 0
Sent Req Accepted 0 0 0
Sent Req Rejected 0 0 0
Sent Req Expired 0 0 0
HTTP Errors 0 0 0

```

Supporting Configuration

The following configuration elements which are not mentioned in this guide are required for the Oracle USM to function. Please refer to the Net-Net 4000 ACLI Configuration Guide for details about configuring all supporting elements.

- network-interface
- physical-interface
- realm-config
- sip-config
- system-config

The following configuration elements are mentioned in this guide briefly and still require configuration:

- local-policy
- session-agent
- sip-interface

Session Load Balancer Support

In order to rapidly increase the number of supported endpoints, the Oracle USM can interoperate with the Net-Net SLB. When paired with a Net-Net SLB, the Oracle USM maintains its ability to function with an HSS or ENUM database.

In addition, the Oracle USM and Net-Net SLB pair supports Cx or ENUM based registrations.

To communicate with a Net-Net SLB, in addition to all baseline Net-Net SBC SIP functionality, the Oracle USM advertises its registration capacity to the Net-Net SLB. This value is defined in the SIP interface as the reg cache limit parameter. Since the Oracle USM has a database registrar license, the lower of the two will be advertised to the SLB.

Verify Config

The Oracle USM performs application specific verification checks when you save a config with the save-config ACLI command. These checks are in addition to baseline Net-Net SBC verification checks.

sip authentication profile (CX)

If session-router > sip-authentication-profile > credential-retrieval-method = CX then confirm

session-router > sip-authentication-profile > credential-retrieval-config value =

any existing session-router > home-subscriber-server configuration > name value

Error

If the above check fails:

1. A WARNING is displayed on the ACLI.
2. An INFO log message is generated.

sip authentication profile (ENUM)

If session-router > sip-authentication-profile > credential-retrieval-method = ENUM-TXT then confirm

session-router > sip-authentication-profile > credential-retrieval-config value =

any existing session-router > enum-config > name value

Error

If the above check fails:

1. A WARNING is displayed on the ACLI.
2. An INFO log message is generated.

sip authentication profile (Local)

If session-router > sip-authentication-profile > credential-retrieval-method = local then confirm

session-router > sip-authentication-profile > credential-retrieval-config =

session-router > local-subscriber-table > ame Error

If the above check fails:

1. A WARNING is displayed on the ACLI.
2. An INFO log message is generated.

sip-registrar

If session-router > sip-registrar > subscriber-database-method = DDNS then confirm

session-router > sip-registrar > subscriber-database-config value =

any existing session-router > enum-config > name value

Error

If the above check fails:

1. A WARNING is displayed on the ACLI.
2. An INFO log message is generated.

sip-registrar

If session-router > sip-registrar > authentication-profile is configured, then confirm its value is any existing:

session-router > sip-authentication-profile > name value

Error

If the above check fails:

1. A WARNING is displayed on the ACLI.
2. An INFO log message is generated.

Resource Utilization


The Oracle USM limits resource utilization to maintain operational stability. Resources managed this way include:

- CPU
- Memory (heap)

CPU Overload Protection

CPU overload protection on the Oracle USM is system-oriented in terms of defining the percent utilization that triggers an action. Actions are application-specific.

For the Oracle USM application, if the CPU usage exceeds the configured setting, the system sends a 5xx error in response to any initial dialog request or standalone transactions. The Oracle USM continues to accept registration refreshes and new transactions within a dialog.

 **Note:** An Oracle CSM configured to operation as an SLRM rejects all messages when CPU utilization exceeds this threshold.

By default the CPU utilization rate is 80%. This value can be changed by the following ACLI command sequence.

```
ORACLE# configure terminal
ORACLE (configure)# session-router
ORACLE (session-router)# sip-config
ORACLE (sip-config)# options +load-limit="70"
ORACLE (sip-config)# done
```

Heap Utilization

The Oracle USM limits memory utilization to maintain operational stability, as follows:

- When heap utilization exceeds 75%, the Oracle USM no longer accepts new registrations. The Oracle USM replies to these messages with 5xx messages. The Oracle USM continues to accept registration refreshes, in-dialog calls and subscriptions.
- When heap utilization exceeds 90%, the Oracle USM drops all messages.

The user can change these thresholds to higher or lower values to best accommodate their operational environment. The user can also determine current memory utilization using the following command and referring to the heap utilization value, towards the bottom of the command's output.

```
ORACLE# show platform heap-statistics
```

The user can disable the heap utilization at 75% functionality using the option shown below.

```
ORACLE# configure terminal
ORACLE (configure)# session-router
ORACLE (session-router)# sip-config
ORACLE (sip-config)# +options disable-memory-overload-protect
ORACLE (sip-config)# done
```

The user can change the default drop-all threshold, from 90% to 80% for example, using the option shown below.

```
ORACLE# configure terminal
ORACLE (configure)# session-router
ORACLE (session-router)# sip-config
ORACLE (sip-config)# +options heap-threshold 80
ORACLE (sip-config)# done
```

