

Oracle® Communications Core Session Manager

Configuration Guide

Release S-CZ7.2.5

Formerly Net-Net SIP Multimedia Xpress

March 2016

Notices

Copyright ©2015, 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

About This Guide.....	17
1 Oracle CSM Basics.....	21
Oracle CSM and IMS.....	21
Session Load Balancer and Route Manager Overview.....	22
Elements of Oracle CSM and SLRM Configuration	23
High Availability.....	24
2 Getting Started with the Oracle CSM.....	25
Oracle CSM Installation and Startup.....	25
Software Images (USM/CSM).....	26
Connecting to Your Oracle CSM.....	26
System Boot.....	28
VM Boot Parameters.....	28
Setting Up System Basics.....	31
New System Prompt.....	32
Booting an Image on Your Oracle CSM.....	32
Booting from Local Storage.....	32
Booting from an External Device.....	32
RADIUS Authentication.....	33
PAP Handshake.....	34
CHAP Handshake.....	35
MS-CHAP-v2 Handshake.....	35
Management Protocol Behavior.....	36
RADIUS Authentication Configuration.....	36
TACACS+ AAA.....	38
TACACS+ Introduction.....	38
TACACS+ Authentication.....	39
TACACS+ Authorization.....	48
TACACS+ Accounting.....	53
TACACS+ Configuration.....	61
Customizing Your ACLI Settings.....	64
Disabling the Second Login Prompt.....	64
Persistent ACLI more Parameter.....	65
Customized Login Banner.....	65
3 System Configuration.....	67
General System Information.....	67
System Identification.....	67
Connection Timeouts.....	67
Configuring General System Information.....	68
Interface Considerations for Netra Platforms.....	69
Software Adaptations for Physical Interfaces.....	69
VM Interfaces.....	69
COTS Interfaces.....	69
COTS Network Interfaces.....	69
The MACTAB File.....	70
Working with the MACTAB File.....	70
Serial Interfaces.....	72

Before You Configure.....	72
Physical Interface Configuration.....	72
Network Interfaces.....	73
IP Configuration.....	73
Network Interface Configuration.....	75
Required SIP Configuration.....	77
Enabling SIP-Config.....	77
SIP Interfaces.....	78
Digest Authentication with SIP.....	86
IP Identification (ID) Field.....	90
IP Identification Field Configuration.....	90
SNMP.....	90
Overview.....	90
Configuring SNMP.....	91
SNMP Configuration Overview.....	91
SNMP Configuration.....	91
Syslog and Process Logs.....	95
Overview.....	95
Syslog and Process Logs Configuration.....	95
Syslog Configuration.....	96
Process Log Configuration.....	96
Host Routes.....	96
Host Routes Example.....	96
Host Route Configuration.....	97
Setting Holidays in Local Policy.....	97
Holidays Configuration.....	97
Enhanced Control of UDP and TCP Ports.....	98
Port 111 Configuration.....	98
Port 3000 and 3001 Configuration.....	99
DNS Transaction Timeout.....	99
Retransmission Logic.....	100
DNS Transaction Timeout Configuration.....	100
DNS Server Status via SNMP.....	101
Persistent Protocol Tracing.....	101
About Persistent Protocol Tracing.....	101
About the Logs.....	101
Persistent Protocol Tracing Configuration.....	102
System Access Control.....	102
Adding an ACL for the Management Interface.....	103
Notes on Deleting System ACLs.....	103
System TCP Keepalive Settings.....	103
System TCP Keepalive Configuration.....	104
Configurable TCP Timers.....	104
Configuring TCP Connection Establishment.....	105
Configuring TCP Data Retransmission.....	105
Timer for Idle Connections.....	106
Historical Data Recording (HDR).....	107
RAMdrive Log Cleaner.....	107
Applicable Settings.....	107
Clean-Up Procedure.....	107
Clean-Up Frequency.....	108
RAMdrive Log Cleaner Configuration.....	108
Configurable Alarm Thresholds and Traps.....	109
SNMP Traps.....	110
Alarm Thresholds Configuration.....	111
Alarm Synchronization.....	111

Caveats.....	112
Alarm Synchronization Configuration.....	112
Accounting Configuration.....	112
Stream Control Transfer Protocol Overview.....	113
SCTP Packets.....	113
SCTP Terminology.....	113
SCTP Message Flow.....	114
Congestion Control.....	115
Multi-Streaming.....	115
Delivery Modes.....	116
Multi-Homing.....	116
Multi-Homing and Path Diversity.....	117
Monitoring Failure Detection and Recovery.....	117
Configuring SCTP Support for SIP.....	118
Configuring an SCTP SIP Port.....	118
Configuring the Realm.....	119
Configuring Session Agents.....	119
Setting SCTP Timers and Counters.....	120
Example Configurations.....	125
IPv6 Address Configuration.....	127
Access Control.....	128
Host Route.....	128
Local Policy.....	128
Network Interface.....	128
ENUM Server.....	129
Realm Configuration.....	129
Session Agent.....	129
SIP Configuration.....	129
SIP Interface SIP Ports.....	129
Steering Pool.....	129
System Configuration.....	129
IPv6 Support for Management and Telemetry.....	130
IPv6 Default Gateway.....	130
IPv6 Link Local Addresses.....	130
Network Interfaces and IPv6.....	131
IPv6 Reassembly and Fragmentation Support.....	131
Access Control List Support.....	132
Data Entry.....	132
DNS Support.....	132
Homogeneous Realms.....	133
Parent-Child Network Interface Mismatch.....	133
Address Prefix-Network Interface Mismatch.....	133
RADIUS Support for IPv6.....	133

4 Realms and Nested Realms..... 135

Overview.....	135
About Realms and Network Interfaces.....	135
About the SIP Home Realm.....	136
About Realms and Other Oracle CSM Functions.....	136
Realms.....	136
Before You Configure.....	136
Realm Configuration.....	136
Address Translation Profiles.....	138
DNS Servers.....	138
DoS ACL Configuration.....	138

Enabling RTP-RTCP UDP Checksum Generation.....	138
Aggregate Session Constraints Per Realm.....	138
UDP Checksum Generation Configuration.....	138
Nested Realms.....	139
Configuring Nested Realms.....	140
Parent and Child Realm Configuration.....	141
Aggregate Session Constraints Nested Realms.....	141
Realm-Based Packet Marking.....	142
About TOS DiffServ.....	143
Signaling Packet Marking Configuration.....	143
Using Class Profile for Packet Marking.....	144
Differentiated Services for DNS and ENUM.....	145
Differentiated Services for DNS and ENUM Configuration.....	146

5 Oracle CSM Supporting the IMS Core..... 147

General Description.....	147
Message Authentication for SIP Requests.....	147
User Authorization.....	147
UAR/UAA Transaction.....	148
SIP Digest User Authentication.....	148
Authentication via MAR/MAA.....	148
SIP Authentication Challenge.....	148
SIP Authentication Response.....	149
Oracle CSM Authentication Check.....	149
IMS-AKA Support.....	150
Authentication Sequence - Registration.....	150
Outside the Core.....	151
Authentication Success.....	151
Authentication Failure.....	152
Synchronization.....	152
Optional IMS-AKA Configuration.....	152
Oracle CSM as Registrar.....	153
New Registration.....	153
Registration Handling for Online and Offline Operation Modes.....	153
Releasing Unregistered Users.....	155
Limiting AOR Contacts.....	156
HSS Server Assignment.....	156
Server Assignment Messages.....	157
Register Refresh.....	157
Entry Unregistration.....	158
User Registration based on Reg-ID and Instance-ID (RFC 5626).....	158
reREGISTER Example.....	159
Outbound Registration Binding Processing.....	159
Wildcarded PUID Support.....	159
ACLI Instructions.....	160
home subscriber server.....	160
SIP Authentication Profile.....	160
SIP Interface.....	161
SIP Registrar.....	161
Maximum Number of Contacts.....	162
Response to Exceeding Maximum Contacts.....	162
SIP Registration Event Package Support.....	163
SUBSCRIBE Processing.....	163
SUBSCRIBE REFRESH Requests.....	164
Reg Event NOTIFY Messages.....	164

Reducing NOTIFY Traffic.....	165
Configuring Registration Event Package.....	165
Registration Event Profile Configuration.....	165
Message Routing.....	166
Registrar Routing.....	167
Default Egress Realm.....	167
Routing Based on UA Capabilities.....	167
ACLI Instructions.....	169
Tel-URI Resolution.....	170
Number Lookup Triggers.....	170
Actions Based on Lookup Results.....	170
Primary and Secondary ENUM Configuration.....	171
HSS Initiated User Profile Changes.....	172
Other Diameter Cx Configuration.....	172
Host and Realm AVP Configuration for Cx.....	172
ACLI Instructions.....	172
Initial Filter Criteria (IFC).....	173
IFC Evaluation.....	173
SIP Registration.....	173
SIP Call.....	173
Evaluating Session Case in the P-Served-User Header.....	174
Supported Sessioncase and Registration State.....	174
Additional Options.....	175
IFC Support for Unregistered Users.....	175
UE-terminating requests to an unregistered user.....	175
Caching the Downloaded IFC.....	177
Optimizing IFC Updates.....	177
Push Profile Request (PPR) updates.....	177
ACLI Instructions.....	177
SIP Registrar.....	177
SIP Registrar.....	177
Shared and Default iFCs.....	178
SiFC Usage.....	178
DiFC Usage.....	178
SiFC/DiFC File Example.....	179
iFC Execution Order.....	179
Refreshing SiFC and DiFC Files.....	179
SiFC and DiFC Configuration.....	180
Distinct and Wildcarded Public Service Identity (PSI) Support.....	180
Configuring SIP Ping OPTIONS Support.....	181
Redundancy and Load Balancing with HSS Servers.....	181
About HSS Groups.....	181
Connection Failure Detection.....	182

6 Routing with Local Policy 185

Session Agents Session Groups and Local Policy.....	185
SIP Session Agents.....	185
Session Agent Status Based on SIP Response.....	186
SIP Session Agent Continuous Ping.....	187
About Session Agents.....	188
Session Agent Groups.....	188
Request URI Construction as Forwarded to SAG-member Session Agent.....	189
SIP Session Agent Group Recursion.....	190
About Local Policy.....	190
Routing Calls by Matching Digits.....	190

Route Preference.....	191
DTMF-Style URI Routing.....	191
SIP Routing.....	192
Limiting Route Selection Options for SIP.....	192
About Loose Routing.....	192
About the Ingress Realm.....	192
About the Egress Realm.....	192
About SIP Redirect.....	194
SIP Method Matching and To Header Use for Local Policies.....	194
Load Balancing.....	196
Configuring Routing.....	197
Configuration Prerequisite.....	197
Configuration Order.....	197
Routing Configuration.....	197
SIP Session Agent DNS-SRV Load Balancing.....	210
Session Agent DNS-SRV Load Balancing Configuration.....	211
Answer to Seizure Ratio-Based Routing.....	211
ASR Constraints Configuration.....	212
ENUM Lookup.....	213
How ENUM Works.....	213
About the Oracle CSM ENUM Functionality.....	214
Custom ENUM Service Type Support.....	215
ENUM Failover and Query Distribution.....	215
ENUM Query Distribution.....	215
Failover to New enum-config.....	215
ENUM Server Operation States.....	215
Server Availability Monitoring.....	216
ENUM Server IP Address and Port.....	216
Unapplicable SNMP Traps and Objects.....	216
IPv6 ENUM SNMP Traps and Objects.....	216
Caching ENUM Responses.....	218
Source URI Information in ENUM Requests.....	218
Operation Modes.....	218
ENUM Configuration.....	220
Configuring the Local Policy Attribute.....	222
Local Policy Example.....	223
CNAM Subtype Support for ENUM Queries.....	223
CNAM Unavailable Response.....	224
SIP Profile Inheritance.....	224
CNAM Subtype Support Configuration.....	224
Using the Local Route Table (LRT) for Routing.....	224
Local Route Table (LRT) Performance.....	225
Local Routing Configuration.....	225
LRT Entry Matching.....	226
LRT String Lookup.....	227
LRT String Lookup Configuration.....	227
Directed Egress Realm from LRT ENUM.....	228
Directed Egress Realm Configuration.....	228
SIP Embedded Route Header.....	229
SIP Embedded Route Header Configuration.....	229
LRT Lookup Key Creation.....	229
Arbitrary LRT Lookup Key.....	229
Hidden Headers for HMR and LRT lookup.....	230
Compound Key LRT Lookup.....	230
Retargeting LRT ENUM-based Requests.....	230
Re-targeting LRT ENUM-based Requests Configuration.....	231

Recursive ENUM Queries.....	231
Recursive ENUM Queries Configuration.....	232
Multistage Local Policy Routing.....	232
Routing Stages.....	232
Network Applications.....	232
Multistage Routing Conceptual Example.....	233
Multistage Routing Example 2.....	233
Customizing Lookup Keys.....	235
Multistage Routing Lookup Termination.....	236
Multistage Local Policy Routing Configuration.....	236
Maintenance and Troubleshooting.....	237
Routing Based on UA Capabilities.....	237
UE Capabilities.....	238
Registering Capabilities at the Oracle CSM.....	238
Preferential Routing.....	238
Explicit Feature Preference.....	238
The “require” and explicit Feature Tag Parameters.....	239
Implicit Feature Preference.....	239
Routing-based RN and CIC.....	239
Routing-based RN Configuration.....	240

7 The Session Load Balancer and Route Manager..... 241

Functional Overview.....	241
Product Functional Matrix.....	241
Physical Deployment.....	242
Oracle CSM's Role as S-CSCF.....	243
Logical Deployment.....	243
SLRM Operation.....	245
Establishing the Load Balance Pool.....	246
Balancing.....	246
Re-balancing.....	247
I-CSCF Operation.....	247
Memory and CPU Overload Protection.....	247
The Sc Interface.....	247
Sc Interface Messages.....	248
Sc Interface Messaging.....	249
Sc Interface Response Codes.....	250
Proprietary SLRM AVP Descriptions.....	251
SLRM Configuration.....	252
set-component-type.....	252
lb-interface.....	253
lb-core-config.....	253
Oracle CSM Configuration.....	254
service-cluster-id.....	254
lb-cfg.....	254
ims-core and lb-list.....	255
Releasing Users.....	255
release-user.....	255
Obtaining SLRM-Related Information.....	255
display-component-type.....	256
show load-balancer.....	256
show sipd endpoint-ip.....	256
SLRM MIB Objects and Traps.....	256

8 Third Party Registration.....259

Third Party Registrations via iFCs.....	260
Embedded REGISTER.....	260
ACLI Instructions - Third Party Registration via iFCs.....	260
Session Agent.....	260
SIP Registrar.....	261
Third Party Registration via ACLI Configuration.....	261
Third Party Registration Server States.....	262
Third Party Registration Expiration.....	262
Defining Third Party Servers.....	263
ACLI Instructions - Third Party Server Configuration.....	263
Third Party Registrar.....	263
SIP Registrar.....	263

9 SIP Signaling Services..... 265

About the Oracle CSM and SIP.....	265
Recurse 305 Only Redirect Action.....	265
Redirect Action Process.....	265
Redirect-Action Set to Proxy.....	266
Redirect-Action Set to Recurse.....	266
Redirect-Action Set to Recurse-305-Only.....	267
Embedded Routes in Redirect Responses.....	268
SIP PRACK Interworking.....	268
UAC-Side PRACK Interworking.....	269
UAS-Side PRACK Interworking.....	269
PRACK Interworking Configuration.....	270
Global SIP Timers.....	270
Overview.....	271
Timers Configuration.....	271
SIP Timers Discreet Configuration.....	272
Session Timer Support.....	273
Call Flow Example.....	273
SIP Options Tag Handling.....	274
Overview.....	274
Configuration Overview.....	275
SIP Option Tag Handling Configuration.....	275
SIP Options.....	276
Overview.....	277
Global SIP Options.....	277
SIP Interface Options.....	283
SIP Session Agent Options.....	284
SIP Realm Options.....	284
SIP Realm Options Configuration.....	285
SIP Security.....	286
Denial of Service Protection.....	286
Configuration Overview.....	286
SIP Unauthorized Endpoint Call Routing.....	287
SIP HNT Forced Unregistration.....	287
When to Use Forced Unregistration.....	288
Caution for Using Forced Unregistration.....	288
SIP HNT Forced Unregistration Configuration.....	288
SIP IP Address Hiding and NATing in XML.....	289
Sample SIP NOTIFY with NATed XML.....	289

SIP Server Redundancy.....	290
Overview.....	290
Configuration Overview.....	290
SIP Server Redundancy Configuration.....	291
Embedded Header Support.....	291
Embedded Header Support Configuration.....	292
Static SIP Header and Parameter Manipulation.....	292
Header Manipulation Rules.....	292
Header Element Rules.....	292
About SIP Header and Parameter Manipulation.....	293
HMR \$LOCAL_PORT for Port Mapping.....	293
SIP Header and Parameter Manipulation Configuration.....	293
SIP HMR (Header Manipulation Rules).....	299
Guidelines for Header and Element Rules.....	300
Precedence.....	300
Duplicate Header Names.....	301
Performing HMR on a Specific Header.....	301
Multiple SIP HMR Sets.....	301
MIME Support.....	301
Find and Replace All.....	302
Escaped Characters.....	302
New Reserved Word.....	303
About the MIME Value Type.....	303
Back Reference Syntax.....	304
Notes on the Regular Expression Library.....	304
SIP Message-Body Separator Normalization.....	304
SIP Header Pre-Processing HMR.....	305
Best Practices.....	306
About Regular Expressions.....	306
Expression Building Using Parentheses.....	308
SIP Manipulation Configuration.....	308
Configuration Examples.....	315
Dialog-Matching Header Manipulation.....	334
About Dialog-Matching Header Manipulations.....	334
Built-In SIP Manipulations.....	336
Testing SIP Manipulations.....	337
HMR Import-Export.....	337
Exporting.....	337
Importing.....	338
Displaying Imports.....	338
Using FTP to Move Files.....	338
Removing Files.....	338
Unique HMR Regex Patterns and Other Changes.....	338
Manipulation Pattern Per Remote Entity.....	338
Reject Action.....	339
Log Action.....	341
Changes to Storing Pattern Rule Values.....	341
Removal of Restrictions.....	342
Name Restrictions for Manipulation Rules.....	342
New Value Restrictions.....	342
Dialog Transparency.....	343
Overview.....	343
Dialog Transparency Configuration.....	343
Route Header Removal.....	343
Route Header Removal Configuration.....	343
SIP Via Transparency.....	344

SIP Via Transparency Configuration.....	344
SIP Number Normalization.....	345
Terminology.....	345
Calls from IP Endpoints.....	345
Calls from IP Peer Network.....	346
SIP Number Normalization Configuration.....	346
SIP Configurable Route Recursion.....	347
Example 1.....	348
Example 2.....	348
SIP Route Recursion Configuration.....	349
SIP Proxy Subscriptions.....	350
Topology Hiding.....	350
SIP Proxy Subscription Configuration.....	351
SIP Local Response Code Mapping.....	352
SIP Local Response Code Mapping Configuration.....	352
Session Agent Ping Message Formatting.....	354
Session Agent Ping Message Formatting Configuration.....	354
Fraud Prevention.....	354
Fraud Prevention Configuration.....	355
Dynamic Transport Protocol Change.....	355
Dynamic Transport Protocol Change Configuration.....	355
SIP Privacy Extensions.....	356
Privacy Types Supported.....	356
Examples.....	357
Configuring SIP Privacy Extensions.....	357
SIP Registration Cache Limiting.....	359
About Registration Cache Additions Modifications and Removals.....	359
Registration Cache Alarm Threshold.....	359
Notes on Surrogate Registration.....	360
Monitoring Information.....	360
SIP Registration Cache Limiting Configuration.....	360
SIP Instance ID in Registration Cache.....	361
SIP Instance ID and the Registration Cache.....	361
SIP Instance ID Configuration.....	361
SIP Request Method Throttling.....	362
About Counters and Statistics.....	362
SIP Request Method Throttling Configuration.....	363
SIP Transport Selection.....	365
SIP Transport Selection Configuration.....	365
uaCSTA NAT Support.....	366
Overview.....	366
SIP Method-Transaction Statistic Enhancements.....	367
SIP Method Tracking Enhancements Configuration.....	367
SIP TCP Connection Reuse.....	367
SIP TCP Connection Reuse Configuration.....	368
SIP TCP Keepalive.....	368
SIP TCP Keepalive Configuration for Session Agents.....	369
SIP TCP Keepalive Configuration for SIP Interfaces.....	369
Local Policy Session Agent Matching for SIP.....	370
Local Policy Session Agent Matching Configuration.....	373
About Wildcarding.....	373
Enforcement Profile Configuration with subscribe-event.....	374
SIP Session Timer Feature.....	375
How the Session Timer Feature Works.....	375
SIP Session Timer Configuration.....	377
305 Response to Registrations on Secondary Interfaces.....	378

ACLI Instructions and Examples.....	378
notify sipd offload-users.....	378
show registration.....	379
show registration sipd.....	379
show sipd endpoint-ip.....	379
SNMP Configuration.....	380
SNMP.....	380

10 Number Translation..... 383

About Number Translation.....	383
Number Translation Implementation.....	383
Number Translation in SIP URIs.....	384
Number Translation Configuration Overview.....	384
Translation Rules.....	384
Translation Rules for Deleting Strings.....	385
Translation Rules for Adding Strings.....	385
Translation Rules for Replacing Strings.....	385
Session Translation.....	385
Applying Session Translations.....	386
Session Agent.....	386
Realm.....	386
Number Translation Configuration.....	386
Translation Rules.....	387
Session Translation.....	387
Number Translation Application.....	388
Other Translations.....	388
FQDN Mapping.....	388

11 Admission Control..... 389

Session Capacity- and Rate-based Admission Control.....	389
Constraints for Proxy Mode.....	389
Admission Control for Session Agents.....	390
Session Agents Admission Control Configuration.....	390
Session Agent Minimum Reserved Bandwidth.....	393
Session Agent Minimum Reserved Bandwidth Configuration.....	394
Aggregate Session Constraints for SIP.....	395
Aggregate Session Constraints Configuration.....	395
Applying Session Constraints in a SIP Interfaces.....	397
Configuring CAC Policing and Marking for non-Audio non-Video Media.....	397
CAC Utilization Statistics via SNMP.....	399
CAC utilization threshold trap on a session agent configuration.....	401
Configuring the CAC Utilization Thresholds - realm.....	402
Whitelists for SIP.....	402
What is a Whitelist.....	402
Whitelists Configuration.....	402
Configuration Exception.....	405
Verify Whitelist Configuration.....	405
How Whitelists Work.....	406
Whitelist Learning.....	406
Whitelist Learning Configuration.....	406
Rejected Messages Monitoring.....	408

12 High Availability Nodes..... 409

Overview.....	409
Establishing Active and Standby Roles.....	410
Switchovers.....	410
State Transitions.....	411
HA Features.....	411
Before Configuring a High Availability (HA) Pair.....	413
HA Node Connections.....	413
Virtual MAC Addresses.....	416
Virtual MAC Address Configuration.....	416
HA Node Connections.....	417
HA Node Connection Configuration.....	417
HA Node Parameters.....	419
Synchronizing Configurations.....	422
Synchronize HA Peers.....	422
Using Configuration Checkpointing.....	423
Manually Checking Configuration Synchronization.....	424
Media Interface Link Detection and Gateway Polling.....	425
Media Interface Link Detection and Gateway Polling Configuration.....	425
Media Interface Link Detection and Gateway Polling Configuration 2.....	426
HA Media Interface Keepalive.....	427
Impact to Boot-Up Behavior.....	427
HA Media Interface Keepalive Configuration.....	427
RTC Notes.....	428
HA.....	428
Protocol-Specific Parameters and RTC.....	428

13 Security..... 431

Security Overview.....	431
Denial of Service Protection.....	432
Levels of DoS Protection.....	433
About the Process.....	433
Trusted Path.....	434
Untrusted Path.....	435
Static and Dynamic ACL Entry Limits.....	436
Dynamic Deny for HNT.....	436
Host and Media Path Protection Process.....	436
Session Director Access Control.....	436
Access Control Endpoint Classification Capacity and DoS.....	437
Media Access Control.....	437
Host Path Traffic Management.....	437
Traffic Promotion.....	437
Malicious Source Blocking.....	437
Blocking Actions.....	438
Protecting Against Session Agent Overloads.....	438
ARP Flood Protection Enhancements.....	438
Dynamic Demotion for NAT Devices.....	438
Configuring DoS Security.....	439
Configuration Overview.....	439
Changing the Default Oracle CSM Behavior.....	439
Access Control List Configuration.....	440
Host Access Policing.....	442
Configuring ARP Flood Protection.....	443
Access Control for a Realm.....	444
Configuring Overload Protection for Session Agents.....	445
DDoS Protection from Devices Behind a NAT.....	446

TCP Synchronize Attack Prevention.....	448
About SYN.....	449
Configuring TCP SYN Attack Prevention.....	449
Host Certificate Retrieval via SNMP.....	449
Host Certificate Retrieval Configuration.....	449
Untrusted Connection Timeout for TCP and TLS.....	450
Caveats.....	450
Untrusted Connection Timeout Configuration for TCP and TLS.....	450
Online Certificate Status Protocol.....	451
Caveats.....	451
Online Certificate Status Protocol Configuration.....	451
Unreachable OCSR.....	453
OCSR Access via FQDN.....	454
Direct and Delegated Trust Models.....	455
Maintenance and Troubleshooting.....	457
show sipd acls.....	457

14 References and Debugging..... 459

ACLI Configuration Elements.....	459
sip-registrar.....	459
Parameters.....	459
Path.....	460
sip-authentication-profile.....	460
Parameters.....	460
Path.....	461
home-subscriber-server.....	461
Parameters.....	461
Path.....	462
third-party-regs.....	462
Parameters.....	462
Path.....	462
enum-config.....	462
Parameters.....	462
Path.....	464
ifc-profile.....	464
Parameters.....	464
Path.....	464
regevent-notification-profile.....	464
Parameters.....	464
Path.....	464
hss-group.....	464
Parameters.....	464
SNMP MIBs and Traps.....	465
Acme Packet License MIB (ap-license.mib).....	465
Acme Packet System Management MIB (ap-smgmt.mib).....	465
Enterprise Traps.....	466
Oracle USM Show Commands.....	466
show sipd endpoint-ip.....	466
show sipd third-party.....	466
show sipd local-subscription.....	467
show registration.....	468
show home-subscriber-server.....	470
show http-server.....	471
Verify Config.....	472
sip authentication profile (CX).....	472

sip authentication profile (ENUM).....	473
sip authentication profile (Local).....	473
sip-registrar.....	473
sip-registrar.....	473
Resource Utilization.....	474
CPU Overload Protection.....	474
Heap Utilization.....	474

A— Oracle Sc Interface Support.....477

Sc Interface and Command Codes.....	477
Diameter AVP Notation.....	477
Table Explanation.....	478
CER Message Format.....	478
CEA Message Format.....	478
DWR Message Format.....	478
DWA Message Format.....	479
SVR Message Format.....	479
SVA Message Format.....	479
CRR Message Format.....	480
CRA Message Format.....	480
Proprietary Grouped AVP Format.....	481
Core-Info AVP.....	481
Service-Info AVP Format.....	481

B— RTC Support.....483

C— CSM and SLRM Base Configuration Elements.....487

CSM Base Configuration Elements.....	487
SLRM Base Configuration Elements.....	488

About This Guide

This ACLI Configuration Guide provides information about:

- Basic concepts that apply to the key features and abilities of your Oracle CSM
- Information about how to load the Oracle CSM system software image you want to use and establish basic operating parameters
- System-level functionality for the Oracle CSM
- Configuration of all components of the Oracle CSM

Supported Platforms

Release Version S-CZ7.2.5 includes both the Oracle Core Session Manager (CSM) and Unified Session Manager (USM) products. The Oracle USM is supported on the Acme Packet 4500, 6100, and 6300 series platforms. The Oracle CSM is supplied as virtual machine software or as a software-only delivery suitable for operation on server hardware. Refer to sales documentation updates for information further specifying hardware support.

Related Documentation

The following table lists the members that comprise the documentation set for this release:

Document Name	Document Description
Acme Packet 4500 Hardware Installation Guide	Contains information about the components and installation of the Acme Packet 4500 system.
Acme Packet 4600 Hardware Installation Guide	Contains information about the components and installation of the Acme Packet 4600 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), Oracle Communications 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.

About This Guide

Document Name	Document Description
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 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.
Call Monitoring Guide	Contains information on call monitoring.

Hardware documentation is relevant only to the Oracle USM. Refer to your hardware vendor's documentation for information required for Oracle CSM operation.

Version SCZ725 software relies on version SCZ720 documentation for some documentation. This documentation includes:

- The ACLI Reference Guide
- The Troubleshooting and Maintenance Guide
- The Administrative Security Essentials Guide

Revision History

Date	Description
November 2014	<ul style="list-style-type: none">• Initial Release
December 2014	<ul style="list-style-type: none">• Updates Session Agent Ping Message Formatting Configuration task and Session Agent Ping Message Formatting Configuration topic to reflect valid descriptions for the session-agent > ping-to-user-part and session-agent > ping-from-user-part parameters.
January 2015	<ul style="list-style-type: none">• Corrects mis-categorization of standard SIP interfaces as Diameter.
February 2015	<ul style="list-style-type: none">• Clarifies the infrastructure response to terminating requests to an unregistered user.
February 2015	<ul style="list-style-type: none">• Removes LST chapter.
June 2015	<ul style="list-style-type: none">• Corrects spelling of psi-cache-expiry option.
January 2016	<ul style="list-style-type: none">• Corrects re-connection timing criteria for establishing connections to external policy servers.• Removes ENUM from the "Compound Key LRT Lookup" topic because ENUM does not support compound lookup keys.• Corrects description of parameter for source realm selection in multi-stage routing.• Adds the missing section 'About Session Agent Groups' in the chapter Session Routing and Load Balancing.• Adds the Request URI Construction as Forwarded to SAG-member Session Agent feature description to the Session Routing and Load Balancing Chapter

Date	Description
March 2016	<ul style="list-style-type: none">• Renames HIP section in Network Interfaces to Administrative Applications Over Media Interfaces. Corrects description regarding firewalls and ports. Adds gateway requirement and overlapping subnet advisory.• Minor edits to Network Interfaces conceptual and Configuration sections.• Updates HIP Address Configuration for clarity. Adds SSH parameter. Adds gateway requirement and overlapping subnet advisory.• Removes Source-based Routing section from System Configuration chapter.

Oracle CSM Basics

This chapter introduces some basic concepts that apply to the key features and abilities of your Oracle CSM. It provides an overview of the concepts related to Oracle CSM configuration and operation in your network as well as the functions it performs in an IMS core.

Oracle CSM software allows for deployment as one of two available components:

- Core Session Manager (CSM)—See the chapter titled Oracle CSM Supporting the IMS Core for detailed information about Oracle CSM configuration and operation in an IMS Core.
- Session Load Balancer and Route Manager (SLRM)—The user can configure the software to operate as a proprietary Session Load Balancer and Route Manager (SLRM). An SLRM allows you to balance traffic between multiple Oracle CSMs. SLRM configuration and operation is covered herein under the chapter titled The Session Load Balancer and Route Manager and the Sc Interface Appendix.

Any given device can be only one component. There can be multiple SLRMs serving multiple CSMs, allowing product deployments to support the largest IMS environments.

Oracle CSM and IMS

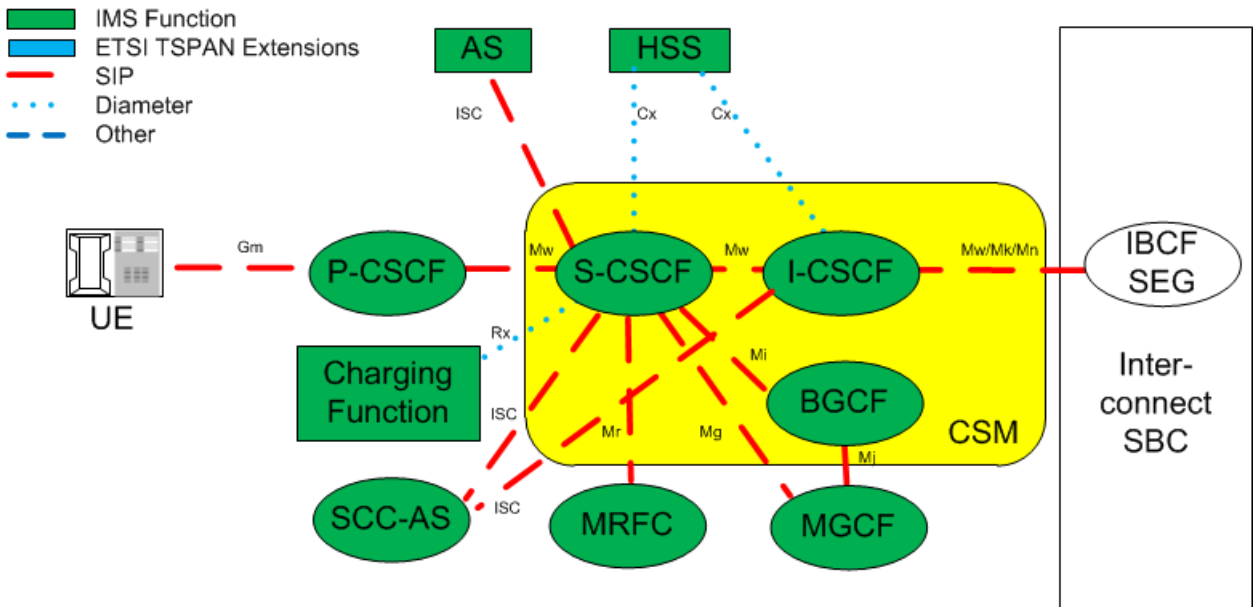
The ETSI TISPAN NGN defines several subsystems that make up the NGN architecture. The model for the target NGN architecture is depicted below. The Oracle CSM is designed to function as an integrated:

- Interrogating-Call Session Control Function (I-CSCF)
- Serving-Call Session Control Function (S-CSCF)

Deployments typically include the Oracle SBC acting as P-CSCF and the Oracle CSM acting as I-CSCF and S-CSCF.

The functions performed by the Oracle CSM are best understood as functions of standard IMS elements. The diagram below depicts the mapping of these functions across an IMS architecture.

Source: 3GPP
TISPAN Workshop



High level definitions of these functions include:

- I-CSCF—IMS passes traffic to the I-CSCF if the target S-CSCF is unknown.
- S-CSCF—Interaction with the Home Subscriber Server (HSS) determines whether and how to provide service to the endpoint.
- BGCF—The breakout gateway control function provides signaling transit to network domains external to the IMS.

Refer to 3GPP specifications for complete element definitions and explanations of the functions they can or must perform.

As I-CSCF, the Oracle CSM complies with 3GPP standards to perform the interrogating function and locate the proper S-CSCF for a given session.

As S-CSCF, the Oracle CSM complies with 3GPP standards and Oracle CSM to manage sessions. It interacts with the HSS to determine whether any given registration can reside locally, or be managed by another S-CSCF device. It also interacts with the HSS and other infrastructure components to provide applicable services within the context of a given session.

Session Load Balancer and Route Manager Overview

The Session Load Balancer and Route Manager (SLRM) is a component of the Oracle CSM software that allows the network architect to establish a front end to multiple Oracle CSMs acting as S-CSCFs.

IMS deployments typically use many S-CSCFs, often dispersed geographically, to provide location services for large numbers of endpoints. Oracle allows you to configure one or more Oracle CSMs as SLRMs to streamline endpoint access to S-CSCFs. A key extension over a standard I-CSCF is the ability of the SLRM to load balance between Oracle CSMs configured as S-CSCFs, thereby preventing any given S-CSCF from becoming overburdened. The user can configure any Oracle CSM as an SLRM without restriction. An SLRM can perform the functions of an I-CSCF, but cannot perform the functions of an S-CSCF.

See the Session Load Balancer and Route Manager chapter for complete explanation and configuration instructions on the SLRM configuration.

Elements of Oracle CSM and SLRM Configuration

Oracle CSM software is deployed as either the CSM or the SLRM component, as configured with the set component command. Each component consists of multiple configuration elements. This guide presents these elements, separating them along conceptual category with chapters roughly equating to configuration sequence. This section lists configuration elements, providing the reader with a consolidated picture of overall product configuration for both components.

See the Base Configuration Elements Appendix for minimal configuration setting examples that establish an operable Oracle CSM and SLRM.

CSM Configuration Elements

Required elements of initial device configuration for CSM, explained in the Getting Started chapter, include:

- Boot Parameters
- Device Passwords
- Management Interfaces

Required network and SIP service configuration elements, explained in multiple chapters, include:

- Enable SIP-Config—System Configuration Chapter
- Default Gateway—System Configuration Chapter
- Service physical and network interface(s)—System Configuration Chapter
- SIP Interfaces—System Configuration Chapter
- SIP Ports—System Configuration Chapter
- Realms—Realms and Nested Realms Chapter
- Required IMS Core configuration elements, explained in the Oracle CSM Supporting the IMS Core Chapter, include:
 - Subscriber Database
 - SIP Registrar
 - ENUM for e.164 Translation
 - Registration Event

Common Oracle CSM Configuration Elements

Common configuration that may be needed for your CSM deployment includes:

- Session Agents
- ENUM Routing
- High Availability (HA)
- CDR Accounting Management
- SNMP Management
- Initial Filter Criteria (iFC)
- 3rd Party Registration Service

SLRM Configuration Elements

Required elements of initial device configuration for SLRM, explained in the Getting Started chapter, include:

- Boot Parameters, including identifying the primary management port
- Device Passwords
- Management Interfaces

Required network and SIP service configuration elements, explained in multiple chapters, include:

- Enable SIP-Config—System Configuration Chapter

- Default Gateway—System Configuration Chapter
- Service physical and network interface(s)—System Configuration Chapter
- SIP Interfaces—System Configuration Chapter
- SIP Ports—System Configuration Chapter
- Realms—Realms and Nested Realms Chapter
- Session Agents—Session Routing and Load Balancing Chapter
- ENUM—Routing with Local Policy Chapter
- Local Routing—Routing with Local Policy Chapter
- Elements of IMS Core service configuration, explained in the Oracle CSM Supporting the IMS Core Chapter, include:
 - Subscriber Database
 - SIP Registrar
 - Authentication Profile
 - ENUM for e.164 Translation

Other Configuration Elements

Common secondary management element configuration includes:

- Additional management interface(s)
- CDR Accounting Management
- SNMP Management

Configuration elements that are available, but may not be required for your deployment include:

- Assorted SIP Functions
- Number Translation
- Admission Control and QoS
- DoS and other Security Functions
- Traffic Monitoring

High Availability

Oracle CSMs are deployed in pairs to deliver continuous high availability (HA) for interactive communication services. The HA design guarantees that no applicable traffic is dropped in the event of any single point failure. Furthermore, the Oracle CSM HA design provides for full registration, call and service state to be shared across an HA node. The solution uses a VRRP-like design, where the two systems share a virtual MAC address and virtual IPv4 address for seamless switchovers.

In the HA pair, one Oracle CSM is the primary system, and is used to process signaling traffic. The backup system remains fully synchronized with the primary system's session status. The primary system continuously monitors itself for connectivity and internal process health. If it detects service-disrupting conditions or degraded service levels, it will alert the backup Oracle CSM to become the active system.

The SLRM does not use HA to establish redundant operation. See the SLRM Description chapter for information on SLRM availability.

Getting Started with the Oracle CSM

Prior to configuring your Oracle CSM for service, we recommend that you review the information and procedures in this chapter.

This chapter offers information that will help you:

- Review hardware installation procedures
- Connect to your Oracle CSM using a console connection, Telnet, or SSH (secure shell)
- Become familiar with the Oracle CSM's boot parameters and how to change them if needed
- Understand and configure Oracle CSM entitlements
- Load and activate an Oracle CSM software image
- Choose a configuration mechanism: ALCI, external management application, or ACP/XML
- Enable RADIUS authentication
- Customize your login banner

Oracle CSM and Oracle USM product installation begin with the same software installation packages. The specific product deployment is based on the hardware platform over which the software is installed.

Oracle CSM is deployed on Commercial Off The Shelf (COTS) computer hardware. The Oracle CSM may also be delivered on "generic" Oracle computer hardware, such as the Netra Server X3-2 for Acme Packet. Either scenario allows for installation as virtual machine or as a standalone application directly over the hardware, sometimes referred to as 'bare metal'. This is supported by packaging the Oracle CSM with its own custom operating system.

The Oracle CSM software detects installation platform upon startup. If the hardware is not the Acme Packet 4000 or 6000 series hardware, the software configures itself as Oracle CSM instead of Oracle USM. From that point on, the user has access to only Oracle CSM commands and configuration elements. Oracle CSM deployments require that the installation team acquire applicable hardware documentation. Ensure that these installations adhere to that documentation.



Note: The Oracle CSM does not require license or entitlement configuration.

Oracle CSM Installation and Startup

The Oracle CSM is a software-only product provided in a manner that allows you to install it on the hardware you choose using the same methodology you use to install an operating system. An example is an .iso image that you can place on boot media, such as a flash drive. When you boot from this media, the Oracle CSM installation script sets up the product for first operation.

Getting Started with the Oracle CSM

Installation on virtual machines is simplified by using the virtual machine template (.ova) distribution. This method allows you to use your virtual machine manager to deploy a pre-built Oracle CSM.

After you have completed the hardware installation procedures, as explained in your hardware installation documentation, you are ready to establish a connection to your Oracle CSM. Then you can load the Oracle CSM software image you want to use and establish basic operating parameters.

Software Images (USM/CSM)

USM/CSM software image filenames for S-CZ7.2.5 and higher end with the .bz extension. They should be placed in the /boot volume.

Connecting to Your Oracle CSM

You can connect to your Oracle CSM either through a direct console connection, or by creating a remote Telnet or SSH session. Both of these access methods provide you with the full range of configuration, monitoring, and management options.



Note: By default, Telnet and SFTP connections to your Oracle CSM are enabled.

Local Connections and Time-outs (Oracle CSM)

Using a serial connection, you can connect your laptop or PC directly to the Oracle CSM. If you use a laptop, you must take appropriate steps to ensure grounding.

One end of the cable plugs into your terminal, and the other end plugs into the serial port on your hardware. Refer to your hardware documentation for the location of the serial port.

To set up a console connection to your Oracle CSM:

1. Set the connection parameters for your terminal to the default boot settings:

- Baud rate: 115,200 bits/second
- Data bits: 8
- Parity: No
- Stop bit: 1
- Flow control: None

2. Use a serial cable to connect your PC to the Oracle CSM.

3. Power on your Oracle CSM.

4. Enter the appropriate password information when prompted to log into User mode of the ACLI.

You can control the amount of time it takes for your console connection to time out by setting the console-timeout parameter in the system configuration. If your connection times out, the login sequence appears again and prompts you for your passwords. The default for this field is 0, which means that no time-out is being enforced.

Incoming Telnet Connections and Time-outs

You can Telnet to your Oracle CSM. Using remote Telnet access, you can provision the Oracle CSM remotely through the management IP interface.

The Oracle CSM, when running on Acme Packet platforms can support up to 5 concurrent Telnet sessions. When running on other platform types, only 4 concurrent Telnet sessions are available. In both cases, only one Telnet session may be in configuration mode at a time.



Note: Telnet does not offer a secure method of sending passwords. Using Telnet, passwords are sent in clear text across the network.

To Telnet to your Oracle CSM, you need to know the IP address of its administrative interface (wancom0/eth0). The wancom0/eth0 IP address of your Oracle CSM is found by checking the **inet on ethernet** value in the boot parameters or visible from the front panel display.

You can manage incoming Telnet connections from the ACLI:

- To set a time-out due to inactivity, use the **telnet-timeout** parameter in the system configuration. You can set the number of seconds that elapse before the Telnet connection or SSH connection is terminated. The default for this field is 0, which means that no time-out is being enforced.
- To view the users who are currently logged into the system, use the ACLI **show users** command. You can see the ID, timestamp, connection source, and privilege level for active connections.
- From Superuser mode in the ACLI, you can terminate the connections of other users in order to free up connections. Use the **kill user** command with the corresponding connection ID.
- From Superuser mode in the ACLI, you can globally enable and disable Telnet connections:
 - Telnet service is enabled by default unless explicitly disabled as shipped.
 - To disable Telnet, type the **management disable telnet** command at the Superuser prompt and reboot your system. The Oracle CSM then refuses any attempts at Telnet connections. If you want to restart Telnet service, type **management enable telnet**.
- If you reboot your Oracle CSM from a Telnet session, you lose IP access and therefore your connection.

SSH Remote Connections

For increased security, you can connect to your Oracle CSM using SSH. An SSH client is required for this type of connection.

The Oracle CSM supports five concurrent SSH and/or SFTP sessions.

There are two ways to use SSH to connect to your Oracle CSM. The first works the way a Telnet connection works, except that authentication takes place before the connection to the Oracle CSM is made. The second requires that you set an additional password.

1. To initiate an SSH connection to the Oracle CSM without specifying users and SSH user passwords:

- a) Open your SSH client (with an open source client, etc.).
- b) At the prompt in the SSH client, type the ssh command, a Space, the IPv4 address of your Oracle CSM, and then press Enter.

The SSH client prompts you for a password before connecting to the Oracle CSM. Enter the Oracle CSM's User mode password. After it is authenticated, an SSH session is initiated and you can continue with tasks in User mode or enable Superuser mode.



Note: You can also create connections to the Oracle CSM using additional username and password options.

2. To initiate an SSH connection to the Oracle CSM with an SSH username and password:

- a) In the ACLI at the Superuser prompt, type the ssh-password and press Enter. Enter the name of the user you want to establish. Then enter a password for that user when prompted. Passwords do not appear on your screen.

```
ORACLE# ssh-password
SSH username [saved]: MJones
Enter new password: 95X-SD
Enter new password again: 95X-SD
```

After you configure ssh-password, the SSH login accepts the username and password you set, as well as the default SSH/SFTP usernames: User and admin.

- b) Configure your SSH client to connect to your Oracle CSM's management IPv4 address using the username you just created. The standard version of this command would be:

```
ssh -l MJones 10.0.1.57
```

- c) Enter the SSH password you set in the ACLI.

```
MJones@10.0.2.54 password: 95X-SD
```

- d) Enter your User password to work in User mode on the Oracle CSM. Enable Superuser mode and enter your password to work in Superuser mode.
- e) A Telnet session window opens and you can enter your password to use the ACLI.

System Boot

When your Oracle CSM boots, the following information about the tasks and settings for the system appear in your terminal window.

- System boot parameters
- From what location the software image is being loaded: an external device or internal flash memory
- Requisite tasks that the system is starting
- Log information: established levels and where logs are being sent
- Any errors that might occur during the loading process

After the loading process is complete, the ACLI login prompt appears.

VM Boot Parameters

Boot parameters specify the information that your Oracle CSM uses at boot time when it prepares to run applications. The Oracle CSM's boot parameters:

- Allow you to set the IP address for the management interface (wancom0).
- Allow you to set a system prompt. The target name parameter also specifies the title name displayed in your web browser and SNMP device name parameters.
- Determine the software image to boot and from where the system boots that image.
- Sets up the username and password for network booting from an external FTP server.

In addition to providing details about the Oracle CSM's boot parameters, this section explains how to view, edit, and implement them.

When displaying the boot parameters, your screen shows a help menu and the first boot parameter (boot device). Press Enter to continue down the list of boot parameters.

Sample VM Boot Parameters

Boot parameters specify the information your Oracle CSM uses at boot time when it prepares to run the software image. The boot parameters:

- Show the Oracle CSM's IPv4 address for the management interface (wancom0)
- Allow you to set a system prompt
- Determine the software image and its location
- Configures the external (T)FTP server used for a net boot

Configuring boot parameters has repercussions for the Oracle CSM's physical and network interface configurations. When you configure these interfaces, you can set values that might override the boot parameters.

The following is a screen capture of the bootparam configuration list:

```
[Acme Boot]: p
Boot File      : /boot/bzImage-bones64
IP Address     : 172.44.12.89
VLAN          :
Netmask       : 255.255.0.0
Gateway       : 172.44.0.1
Host IP       :
FTP username   :
FTP password   :
Flags         : 0x00000030
Target Name    : ACMEPACKET
Console Device : COM1
Console Baudrate : 115200
Other         :

[Acme Boot]: ?
?                - print this list
```

```

@           - boot (load and go)
p           - print boot params
c           - change boot params
v           - print boot logo with version
r           - reboot
s           - show license information

```

Boot flags:

```

0x02 - enable kernel debug
0x04 - disable crashdumps
0x08 - disable bootloader countdown
0x10 - enable debug login
0x40 - use DHCP for wancom0
0x80 - use TFTP instead of FTP

```

VM Boot Parameter Definitions

The following table defines each of the Oracle CSM's boot parameters.


Boot Parameter	Description
Boot File	The name and path of the software image you are booting. Include the absolute path for a local boot from the local /boot volume and for a net boot when a path on the FTP server is needed.
IP Address	IP address of wancom0.
VLAN	VLAN of management network over which this address is accessed.
Netmask	Netmask portion of the wancom0 IP Address.
Gateway	Network Gateway that this wancom0 interface uses.
Host IP	IP Address of FTP server to download and execute a software image from.
FTP Username	FTP Server Username.
FTP Password	FTP Server Password.
Flags	<p>Codes that signal the Oracle CSM from where to boot. Also signals the system about which file to use in the booting process. This sequence always starts with 0x (these flags are hexadecimal). The most common codes are:</p> <p>0x08: Means that the system looks at the filename defined in the boot configuration parameters to determine where to boot from and what file to use. If the file name parameter contains /tffsX/filename, then the system boots off the flash memory (see options below). If the file name parameter just contains a filename, then the Oracle CSM boots off the external host defined and looks for the filename in the /tftpboot directory on that host.</p> <p>0x80008: Used for source routing.</p> <p>If your requirements differ from what these flags allow, contact your Oracle customer support representative for further codes.</p>
Target Name	<p>Name of the Oracle CSM as it appears in the system prompt. For example, ORACLE> or ORACLE#. You need to know the target name if you are setting up an HA node.</p> <p>This name is required to be unique among Oracle CSMs in your network. This name can be 64 characters or less.</p>
Console Device	Set this to COM1.

Getting Started with the Oracle CSM

Boot Parameter	Description
Console Baud Rate	The speed in bits per second which the console port operates at. It operates at 115200 BPS, 8 data bits, no stop bit, parity NONE.
Other	Unused.

Boot Parameter Changes

You can access and edit boot parameters by using either the ACLI or by interrupting the system boot process.

 **Note:** Changes to boot parameters do not go into effect until you reboot the Oracle CSM.

Oracle recommends that you use management port 0 (wancom0) as the boot interface, and that your management network is either:

- directly a part of your LAN for management port 0
- accessible through management port 0

Otherwise, your management messages may use an incorrect source address.

Change Boot Parameters from the ACLI

To access and change boot parameters from the ACLI:

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

```
ORACLE# configure terminal
```

2. Type `bootparam`, and press Enter. The boot device parameters display.

```
ORACLE(configure)# bootparam
'.' = clear field; '-' = go to previous field; ^D = quit
boot device      : eth0
```

To navigate through the boot parameters, press Enter and the next parameter appears on the following line.

You can navigate through the entire list this way. To go back to a previous line, type a hyphen (-) and press Enter. Any value that you enter entirely overwrites the existing value and does not append to it.

3. To change a boot parameter, type the new value that you want to use next to the old value. For example, if you want to change the image you are using, type the new filename next to the old one. You can clear the contents of a parameter by typing a period and then pressing Enter.

```
ORACLE(configure)# bootparam
'.' = clear field; '-' = go to previous field; ^D = quit
boot device      : eth0
processor number  : 0
host name        : goose
file name        : /boot/nnPCz100.gz /boot/nnPCz200.gz
```

When you have scrolled through all of the boot parameters, the system prompt for the configure terminal branch displays.

```
ORACLE(configure)#
```


4. Exit the configure terminal branch.
5. Reboot the Oracle CSM for the changes to take effect.

The ACLI `reboot` and `reboot force` commands initiate a reboot. With the `reboot` command, you must confirm that you want to reboot. With the `reboot force` command, you do not have to make this confirmation.

```
ORACLE# reboot force
```

The Oracle CSM completes the full booting sequence. If necessary, you can stop the auto-boot at countdown to fix any boot parameters.

If you configured boot parameters correctly, the system prompt displays and you can go ahead with configuration, management, or monitoring tasks.

-  **Note:** If you configured the boot parameters incorrectly, the Oracle CSM goes into a booting loop and displays an error message.

```
Error loading file: errno = 0x226.
Can't load boot file!!
```

Press the space bar to stop the loop. Correct the error in the boot parameter, and reboot the system.

Change Boot Parameters by Interrupting a Boot in Progress

To access and change boot parameters by interrupting a boot in progress:

1. When the Oracle CSM is in the process of booting, you can press the space bar on your keyboard to interrupt when you see the following message appear:

```
Press the space bar to stop auto-boot...
```

2. After you stop the booting process, you can enter the letter `p` to display the current parameters, the letter `c` to change the boot parameters or the `@` (at-sign) to continue booting.

```
[Acme Packet Boot]: c
'.' = clear field; '-' = go to previous field; ^D = quit
boot device      : wancom0
```

To navigate through the boot parameters, press `Enter` and the next parameter appears on the following line.

You can navigate through the entire list this way. To go back to a previous line, type a hyphen (`-`) and press `Enter`. Any value that you enter entirely overwrites the existing value and does not append to it.

3. To change a boot parameter, type the new value that you want to use next to the old value. For example, if you want to change the image you are using, type the new filename next to the old one.


```
ORACLE(configure)# bootparam
'.' = clear field; '-' = go to previous field; ^D = quit
boot device      : wancom0
processor number  : 0
host name        : goose
file name        : /code/nnPCz100.bz /code/nnPCz200.bz
```

4. After you have scrolled through the complete list of boot parameters, you return to the boot prompt. To reboot with your changes taking effect, type `@` (the at-sign), and press `Enter`.

```
[Acme Packet Boot]: @
```

The Oracle CSM completes the full booting sequence, unless there is an error in the boot parameters.

If you have configured boot parameters correctly, the system prompt displays and you can go ahead with configuration, management, or monitoring tasks.

-  **Note:** If you have configured the boot parameters incorrectly, the Oracle CSM goes into a booting loop and displays an error message.

```
Error loading file: errno = 0x226.
Can't load boot file!!
```

Press the space bar to stop the loop. Correct the error, and reboot your system.

Setting Up System Basics

Before configuring and deploying the Oracle CSM, you might want to establish some basic attributes such as a system prompt, new User and Superuser passwords, and NTP synchronization.

New System Prompt

The ACLI system prompt is set in the boot parameters. To change it, access the boot parameters and change the target name value to make it meaningful within your network. The target name may be up to 38 characters. A value that identifies the system in some way is often helpful.

Booting an Image on Your Oracle CSM

You can either boot your Oracle CSM from the system's local storage or from an external device. Both locations can store images from which the system can boot. This section describes both booting methods.

For boot parameters to go into effect, you must reboot your Oracle CSM. Since a reboot stops all call processing, Oracle recommends performing tasks that call for a reboot during off-peak hours. If your Oracle CSMs are set up in an HA node, you can perform these tasks on the standby system first.

Booting from Local Storage

Once you have installed an image, you can boot your Oracle CSM from its local storage system. With the exception of testing an image before you copy it to the Oracle CSM's storage, this is generally the method you use for booting.

To boot from your Oracle CSM local storage:

1. Confirm that the boot parameters are set up correctly, and make any necessary changes.

You can check the boot configuration parameters by accessing the bootparam command from the configure terminal menu.

```
ORACLE# configure terminal
ORACLE# bootparam
```

2. Change any boot configuration parameters that you need to change. It is especially important to change the file name boot configuration parameter. The file name parameter needs to use the /boot value so that the Oracle CSM boots from the local folder. Users can store multiple images within the system's /boot directory.
3. Reboot your Oracle CSM.
4. You are returned to the ACLI login prompt. To continue with system operations, enter the required password information.

Booting from an External Device

Booting from an external device means that your Oracle CSM connects to a server to retrieve the boot image at boot time. Rather than using an image stored on your system, it downloads the image from the external device each time it reboots.

When you are testing a new image before putting it on your Oracle CSM, you might want to boot from an external device. Ordinarily, you would not want to boot an image on your Oracle CSM this way.

To boot an image from an external device:

1. Confirm that the Oracle CSM is cabled to the network from which you are booting. This is the port designated as your management port on your hardware port 0. (You may find references to this port as wancom0.) The image is loaded from the source using FTP.
2. Log into the system you want to mount.
3. On the Oracle CSM, configure the information for the boot parameters and confirm the following:

- boot device—device to which you will FTP

This parameter value must contain the name of the applicable management interface, and then the number of the appropriate port. Usually, this value is wancom0.

- file name—name on the host of the file containing the image

The image file must exist in the home directory of the user on the image source.

- host inet—IPv4 address of the device off of which you are booting
 - gateway inet—IPv4 address of the gateway to use if the device from which you are booting is not on the same network as your Oracle CSM
 - user—username for the FTP account on the boot host
 - password—password for the FTP account on the boot host
4. Reboot your Oracle CSM.
 5. You are returned to the ACLI login prompt. To continue with system operations, enter the required password information.

RADIUS Authentication

A security feature that extends beyond the designation of ACLI User and Superuser privileges, the User Authentication and Access control feature supports authentication using your RADIUS server(s). In addition, you can set two levels of privilege, one for all privileges and more limited set that is read-only.

User authentication configuration also allows you to use local authentication, localizing security to the Oracle CSM ACLI log-in modes. These modes are User and Superuser, each requiring a separate password.

The components involved in the RADIUS-based user authentication architecture are the Oracle CSM and your RADIUS server(s). In these roles:

- The Oracle CSM restricts access and requires authentication via the RADIUS server; the Oracle CSM communicates with the RADIUS server using either port 1812 or 1645, but does not know if the RADIUS server listens on these ports
- Your RADIUS server provides an alternative method for defining Oracle CSM users and authenticating them via RADIUS; the RADIUS server supports the VSA called ACME_USER_CLASS, which specifies what kind of user is requesting authentication and what privileges should be granted.

The Oracle CSM also supports the use of the Cisco Systems Inc.™ Cisco-AVPair vendor specific attribute (VSA). This attribute allows for successful administrator login to servers that do not support the Oracle authorization VSA. While using RADIUS-based authentication, the Oracle CSM authorizes you to enter Superuser mode locally even when your RADIUS server does not return the ACME_USER_CLASS VSA or the Cisco-AVPair VSA. For this VSA, the Vendor-ID is 1 and the Vendor-Type is 9. The list below shows the values this attribute can return, and the result of each:

- shell:priv-lvl=15—User automatically logged in as an administrator
- shell:priv-lvl=1—User logged in at the user level, and not allowed to become an administrator
- Any other value—User rejected

When RADIUS user authentication is enabled, the Oracle CSM communicates with one or more configured RADIUS servers that validates the user and specifies privileges. On the Oracle CSM, you configure:

- What type of authentication you want to use on the Oracle CSM
- If you are using RADIUS authentication, you set the port from which you want the Oracle CSM to send messages
- If you are using RADIUS authentication, you also set the protocol type you want the Oracle CSM and RADIUS server to use for secure communication

Although most common set-ups use two RADIUS servers to support this feature, you are allowed to configure up to six. Among other settings for the server, there is a class parameter that specifies whether the Oracle CSM should consider a specific server as primary or secondary. As implied by these designation, the primary servers are used first for authentication, and the secondary servers are used as backups. If you configure more than one primary and one secondary server, the Oracle CSM will choose servers to which it sends traffic in a round-robin strategy. For example, if you specify three servers are primary, the Oracle CSM will round-robin to select a server until it finds an appropriate one; it will do the same for secondary servers.

The VSA attribute assists with enforcement of access levels by containing one of the three following classes:


- None—All access denied

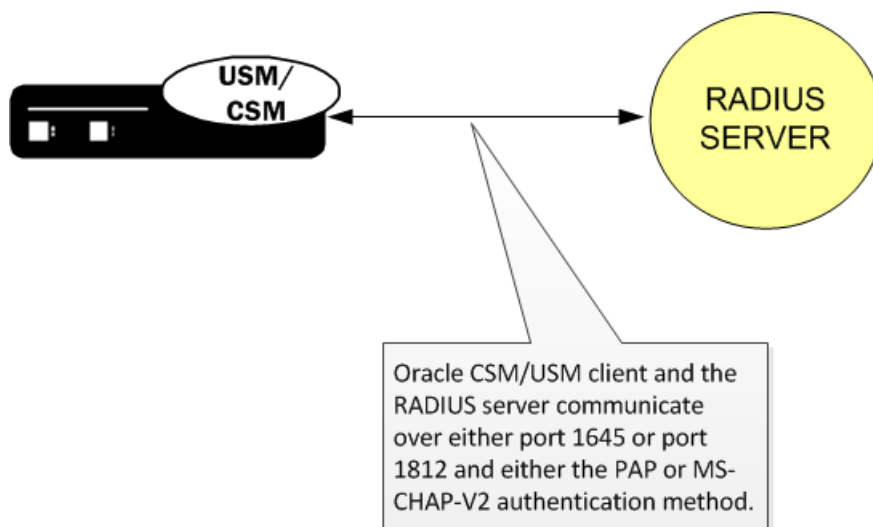
Getting Started with the Oracle CSM

- User—Monitoring privileges are granted; your user prompt will resemble ORACLE>
- Admin—All privileges are granted (monitoring, configuration, etc.); your user prompt will resemble ORACLE#

Once it has selected a RADIUS server, the Oracle CSM initiates communication and proceeds with the authentication process. The authentication process between the Oracle CSM and the RADIUS server takes place uses one of three methods, all of which are defined by RFCs:

Protocol	RFC
PAP (Password Authentication Protocol)	B. Lloyd and W. Simpson, PPP Authentication Protocols, RFC 1334, October 1992
CHAP (Challenge Handshake Authentication Protocol)	B. Lloyd and W. Simpson, PPP Authentication Protocols, RFC 1334, October 1992 W. Simpson, PPP Challenge Handshake Authentication Protocol (CHAP), RFC 1994, August 1996
MS-CHAP-V2	G. Zorn, Microsoft PPP CHAP Extensions, Version 2, RFC 2759, January 2000

 **Note:** MS-CHAP-V2 support includes authentication only; password exchange is not supported or allowed on the Oracle CSM.



PAP Handshake

For PAP, user credentials are sent to the RADIUS server include the user name and password attribute. The value of the User-Password attribute is calculated as specified in RFC 2865.

PAP Client Request Example

```
Radius Protocol
Code: Access Request (1)
  Packet identifier: 0x4 (4)
  Length: 61
  Authenticator: 0x0000708D00002C5900002EB600003F37
  Attribute value pairs
    t:User Name(1) 1:11, value:"TESTUSER1"
      User-Name: TESTUSER1
    t:User Password (2) 1:18, value:739B3A0F25094E4B3CDA18AB69EB9E4
    t:NAS IP Address(4) 1:6, value:168.192.68.8
      Nas IP Address: 168.192.68.8 (168.192.68.8)
    t:NAS Port(5) 1:6, value:118751232
```

PAP RADIUS Response

```
Radius Protocol
Code: Access Accept (2)
Packet identifier: 0x4 (4)
Length: 20
Authenticator: 0x36BD589C1577FD11E8C3B5BB223748
```

CHAP Handshake

When the authentication mode is CHAP, the user credentials sent to the RADIUS server include “username,” “CHAP-Password,” and “CHAP-Challenge.” The “CHAP-Password” credential uses MD-5 one way. This is calculated over this series of the following values, in this order: challenge-id (which for the Oracle CSM is always 0), followed by the user password, and then the challenge (as specified in RFC 1994, section 4.1).

CHAP Client Request Example

```
Radius Protocol
Code: Access Request (1)
Packet identifier: 0x5 (5)
Length: 80
Authenticator: 0x0000396C000079860000312A00006558
Attribute value pairs
  t:User Name(1) l:11, value:"TESTUSER1"
    User-Name: TESTUSER1
  t:CHAP Password (3) l:19, value:003D4B1645554E881231ED7A137DD54FBF
  t:CHAP Challenge (60) l:18, value: 000396C000079860000312A00006558
  t:NAS IP Address(4) l:6, value:168.192.68.8
    Nas IP Address: 168.192.68.8(168.192.68.8)
  t:NAS Port(5) l:6, value:118751232
```

CHAP RADIUS Response

```
Radius Protocol
Code: Access Accept (2)
Packet identifier: 0x4 (4)
Length: 20
Authenticator: 0x3BE89EED1B43D91D80EB2562E9D65392
```

MS-CHAP-v2 Handshake

When the authentication method is MS-CHAP-v2, the user credentials sent to the RADIUS server in the Access-Request packet are:

- username
- MS-CHAP2-Response—Specified in RFC 2548, Microsoft vendor-specific RADIUS attributes
- MS-CHAP2-Challenge—Serves as a challenge to the RADIUS server

If the RADIUS authentication is successful, the Access-Accept packet from the RADIUS server must include an MS-CHAP2-Success attribute calculated using the MS-CHAP-Challenge attribute included in the Access-Request. The calculation of MS-CHAP2-Success must be carried out as specified in RFC 2759. The Oracle CSM verifies that the MS-CHAP2-Success attribute matches with the calculated value. If the values do not match, the authentication is treated as a failure.

MS-CHAP-v2 Client Request Example

Some values have been abbreviated.

```
Radius Protocol
Code: Access Request (1)
Packet identifier: 0x5 (5)
Length: 80
Authenticator: 0x0000024C000046B30000339F00000B78
Attribute value pairs
```

Getting Started with the Oracle CSM

```
t:User Name(1) l:11, value:"TESTUSER1"
  User-Name: TESTUSER1
t:Vendor Specific(26) l:24, vendor:Microsoft(311)
t:MS CHAP Challenge(11) l:18, value:0000024C000046B30000339F00000B78
t:Vendor Specific(26) l:58, vendor:Microsoft(311)
t:MS CHAP2 Response(25) l:52, value:
00000000024C000046B30000339F00000B78...
t:NAS IP Address(4) l:6, value:168.192.68.8
  Nas IP Address: 168.192.68.8(168.192.68.8)
t:NAS Port(5) l:6, value:118751232
```

MS-CHAP-v2 RADIUS Response

```
Radius Protocol
Code: Access Accept (2)
Packet identifier: 0x6 (6)
Length: 179
Authenticator: 0xECB4E59515AD64A2D21FC6D5F14D0CC0
Attribute value pairs
t:Vendor Specific(26) l:51, vendor:Microsoft(311)
  t:MS CHAP Success(11) l:45, value:003533s33d3845443532443135453846313...
t:Vendor Specific(26) l:42, vendor:Microsoft(311)
  t:MS MPPE Recv Key(17) l:36, value:96C6325D22513CED178F770093F149CBBA...
t:Vendor Specific(26) l:42, vendor:Microsoft(311)
  t:MS MPPE Send Key(16) l:36, value:9EC9316DBFA701FF0499D36A1032678143...
t:Vendor Specific(26) l:12, vendor:Microsoft(311)
  t:MS MPPE Encryption Policy(7) l:6, value:00000001
t:Vendor Specific(26) l:12, vendor:Microsoft(311)
  t:MS MPPE Encryption Type(8) l:6, value:00000006
```

Management Protocol Behavior

When you use local authentication, management protocols behave the same way that they do when you are not using RADIUS servers. When you are using RADIUS servers for authentication, management protocols behave as described in this section.

- Telnet—The “user” or admin accounts are authenticated locally, not via the RADIUS server. For all other accounts, the configured RADIUS servers are used for authentication. If authentication is successful, the user is granted privileges depending on the ACME_USER_CLASS VSA attribute.
- FTP—The “user” or admin accounts are authenticated locally, not via the RADIUS server. For all other accounts, the configured RADIUS servers are used for authentication.
- SSH in pass-through mode—When SSH is in pass through mode, the Oracle CSM behave the same way that it does for Telnet.
- SSH in non-pass-through mode—When you create an SSH account on the Oracle CSM, you are asked to supply a user name and password. Once local authentication succeeds, you are prompted for the CLI user name and password. If your user CLI name is user, then you are authenticated locally. Otherwise, you are authenticated using the RADIUS server. If RADIUS authentication is successful, the privileges you are granted depend on the ACME_USER_CLASS VSA attribute.
- SFTP in pass-through mode—If you do not configure an SSH account on the Oracle CSM, the RADIUS server is contacted for authentication for any user that does not have the user name user. The Oracle CSM uses local authentication if the user name is user.
- SFTP in non-pass-through mode—The “user” or admin accounts are authenticated locally, not via the RADIUS server. For all other accounts, the configured RADIUS servers are used for authentication.

RADIUS Authentication Configuration

To enable RADIUS authentication and user access on your Oracle CSM, you need to configure global parameters for the feature and then configure the RADIUS servers that you want to use.

Global Authentication Settings

To configure the global authentication settings:

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

```
ORACLE# configure terminal
```

2. Type security and press Enter.

```
ORACLE (configure)# security
```

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

```
ORACLE (security)# authentication
ORACLE (authentication)#
```

From here, you can view the entire menu for the authentication configuration by typing a ?. You can set global parameters for authentication. You can also configure individual RADIUS servers; instructions for configuring RADIUS server appear in the next section.

4. type—Set the type of user authentication you want to use on this Oracle CSM. The default value is local. The valid values are:
 - local | radius
5. protocol—If you are using RADIUS user authentication, set the protocol type to use with your RADIUS server(s). The default is pap. The valid values are:
 - pap | chap | mschapv2
6. source-port—Set the number of the port you want to use from message sent from the Oracle CSM to the RADIUS server. The default value is 1812. The valid values are:
 - 1645 | 1812
7. allow-local-authorization—Set this parameter to enabled if you want the Oracle CSM to authorize users to enter Superuser (administrative) mode locally even when your RADIUS server does not return the ACME_USER_CLASS VSA or the Cisco-AVPair VSA. The default for this parameter is disabled.

RADIUS Server Settings

The parameters you set for individual RADIUS servers identify the RADIUS server, establish a password common to the Oracle CSM and the server, and establish trying times.

Setting the class and the authentication methods for the RADIUS servers can determine how and when they are used in the authentication process.

To configure a RADIUS server to use for authentication:

1. Access the RADIUS server submenu from the main authentication configuration:

```
ORACLE (authentication)# radius-servers
ORACLE (radius-servers)#
```

2. address—Set the remote IP address for the RADIUS server. There is no default value, and you are required to configure this address.
3. port—Set the port at the remote IP address for the RADIUS server. The default port is set to 1812. The valid values are:
 - 1645 | 1812
4. state—Set the state of the RADIUS server. Enable this parameter to use this RADIUS server to authenticate users. The default value is enabled. The valid values are:
 - enabled | disabled
5. secret—Set the password that the RADIUS server and the Oracle CSM share. This password is transmitted between the two when the request for authentication is initiated; this ensures that the RADIUS server is communicating with the correct client.

Getting Started with the Oracle CSM

6. `nas-id`—Set the NAS ID for the RADIUS server. There is no default for this parameter.
7. `retry-limit`—Set the number of times that you want the Oracle CSM to retry for authentication information from this RADIUS server. The default value is 3. The valid range is:
 - Minimum—1
 - Maximum—5

If the RADIUS server does not respond within this number of tries, the Oracle CSM marks it as dead.
8. `retry-time`—Set the amount of time (in seconds) that you want the Oracle CSM to wait before retrying for authentication from this RADIUS server. The default value is 5. The valid range is:
 - Minimum—5
 - Maximum—10
9. `dead-time`—Set the amount of time in seconds before the Oracle CSM retries a RADIUS server that it has designated as dead because that server did not respond within the maximum number of retries. The default is 10. The valid range is:
 - Minimum—10
 - Maximum—10000
10. `maximum-sessions`—Set the maximum number of outstanding sessions for this RADIUS server. The default value is 255. The valid range is:
 - Minimum—1
 - Maximum—255
11. `class`—Set the class of this RADIUS server as either primary or secondary. A connection to the primary server is tried before a connection to the secondary server is tried. The default value is primary. Valid values are:
 - `primary` | `secondary`

The Oracle CSM tries to initiate contact with primary RADIUS servers first, and then tries the secondary servers if it cannot reach any of the primary ones.

If you configure more than one RADIUS server as primary, the Oracle CSM chooses the one with which it communicates using a round-robin strategy. The same strategy applies to the selection of secondary servers if there is more than one.
12. `authentication-methods`—Set the authentication method you want the Oracle CSM to use with this RADIUS server. The default value is `pap`. Valid values are:
 - `all` | `pap` | `chap` | `mschapv2`

This parameter has a specific relationship to the global protocol parameter for the authentication configuration, and you should exercise care when setting it. If the authentication method that you set for the RADIUS server does not match the global authentication protocol, then the RADIUS server is not used. The Oracle CSM simply overlooks it and does not send authentication requests to it. You can enable use of the server by changing the global authentication protocol so that it matches.
13. Save your work and activate your configuration.

TACACS+ AAA

TACACS+ (Terminal Access Controller Access Control System Plus) is a protocol originally developed by Cisco Systems, and made available to the user community by a draft RFC, *TACACS+ Protocol, Version 1.78* (draft-grant-tacacs-02.txt). TACACS+ provides AAA (Authentication, Authorization, and Accounting) services over a secure TCP connection using Port 49.

TACACS+ Introduction

Like DIAMETER and RADIUS, TACACS+ uses a client/server model in which a Network Access Server (NAS) acts in the client role and a TACACS+ equipped device (a daemon in TACACS+ nomenclature) assumes the server role.

For purposes of the current implementation, the Oracle CSM functions as the TACACS+ client. Unlike RADIUS, which combines authentication and authorization, TACACS+ provides three distinct applications to provide finer grade access control.

Authentication is the process that confirms a user's purported identity. Authentication is most often based on a simple username/password association, but other, and more secure methods, are becoming more common. The following authentication methods are supported by the current implementation: simple password, PAP (Protocol Authentication Protocol), and CHAP (Challenge Handshake Authentication Protocol).

Authorization is the process that confirms user privileges. TACACS+ can provide extremely precise control over access to system resources. In the current implementation, TACACS+ controls access to system administrative functions.

TACACS+ provides secure communication between the client and daemon by encrypting all packets. Encryption is based on a shared-secret, a string value known only to the client and daemon. Packets are encrypted in their entirety, save for a common TACACS+ header.

The cleartext header contains, among other fields, a version number, a sequence number, and a session ID. Using a methodology described in Section 5 of the TACACS+ draft RFC, the sender encrypts outbound cleartext messages by repetitively running the MD5 hash algorithm over the concatenation of the session ID, shared-secret, version number, and sequence number values, eventually deriving a virtual one-time-pad of the same length as the message body. The sender encrypts the cleartext message with an XOR (Exclusive OR) operation, using the cleartext message and virtual one-time-pad as inputs.

The message recipient, who possesses the shared-secret, can readily obtain the version number, sequence number, session ID, and message length from the cleartext header. Consequently, the recipient employs the same methodology to derive a virtual one-time-pad identical to that derived by the sender. The recipient decrypts the encrypted message with an XOR operation, using the encrypted message and virtual one-time-pad as inputs.

Details on TACACS+ are available in the ACLI Configuration Guide.

The TACACS+ implementation is based upon the following internet draft.

draft-grant-tacacs-02.txt, *The TACACS+ Protocol Version 1.78*

Other relevant documents include

RFC 1321, *The MD-5 Message Digest Algorithm*

RFC 1334, *PPP Authentication Protocols* .

RFC 1994, *PPP Challenge Handshake Authentication Protocol (CHAP)*

TACACS+ Authentication

The Oracle CSM uses TACACS+ authentication services solely for the authentication of user accounts. Administrative users must be authenticated locally by the Oracle CSM.

The current TACACS+ implementation supports three types of user authentication: simple password (referred to as ascii by TACACS+), PAP, and CHAP.

ascii Login

ascii login is analogous to logging into a standard PC. The initiating peer is prompted for a username, and, after responding, is then prompted for a password.

PAP Login

PAP is defined in RFC 1334, *PPP Authentication Protocols*. This protocol offers minimal security in that passwords are transmitted as unprotected cleartext. PAP login differs from ascii login in that the username and password are transmitted to the authenticating peer in a single authentication packet, as opposed to the two-step prompting process used in ascii login.

CHAP Login

CHAP is defined in RFC 1994, *PPP Challenge Handshake Authentication Protocol*. CHAP is more secure than PAP in that it is based on a shared-secret (known only to the communicating peers), and therefore avoids the transmission of cleartext authentication credentials. CHAP operations can be summarized as follows.

After a login attempt, the initiator is tested by the authenticator who responds with a packet containing a challenge value — an octet stream with a recommended length of 16 octets or more. Receiving the challenge, the initiator concatenates an 8-bit identifier (carried within the challenge packet header), the shared-secret, and the challenge value, and uses the shared-secret to compute an MD-5 hash over the concatenated string. The initiator returns the hash value to the authenticator, who performs the same hash calculation, and compares results. If the hash values match, authentication succeeds; if hash values differ, authentication fails.

Authentication Message Exchange

All TACACS+ authentication packets consist of a common header and a message body. Authentication packets are of three types: START, CONTINUE, and REPLY.

START and CONTINUE packets are always sent by the Oracle CSM, the TACACS+ client. START packets initiate an authentication session, while CONTINUE packets provide authentication data requested by the TACACS+ daemon. In response to every client-originated START or CONTINUE, the daemon must respond with a REPLY packet. The REPLY packet contains either a decision (pass or fail), which terminates the authentication session, or a request for additional information needed by the authenticator.

TACACS+ Header

The TACACS+ header format is as follows.

```
+-----+-----+-----+-----+-----+
|maj |min | type  | seq_no | flags |
|ver |ver |      |      |      |
+-----+-----+-----+-----+
| session_id
+-----+-----+-----+-----+
| length
+-----+-----+-----+-----+
```

maj ver

This 4-bit field identifies the TACACS+ major protocol version, and must contain a value of 0xC .

min ver

This 4-bit field identifies the TACACS+ minor protocol version, and must contain either a value of 0x0 (identifying TACACS+ minor version 0) or a value of 0x1 . (identifying TACACS+ minor version 1). Minor versions 0 and 1 differ only in the processing of PAP and CHAP logins.

type

This 8-bit field identifies the TACACS+ AAA service as follows:

0x1 — TACACS+ Authentication

0x2 — TACACS+ Authorization

0x3 — TACACS+ Accounting

sequence-no

This 8-bit field contains the packet sequence for the current session.

The first packet of a TACACS+ session must contain the value 1; each following packet increments the sequence count by 1. As TACACS+ sessions are always initiated by the client, all client-originated packets carry an odd sequence number, and all daemon-originated packets carry an even sequence number. TACACS+ protocol strictures do not allow the sequence_no field to wrap. If the sequence count reaches 255, the session must be stopped and restarted with a new sequence number of 1.

flags

This 8-bit field contains flags as described in Section 3 of the draft RFC; flags are not under user control.

session_id

This 32-bit field contains a random number that identifies the current TACACS+ session — it is used by clients and daemons to correlate TACACS+ requests and responses.

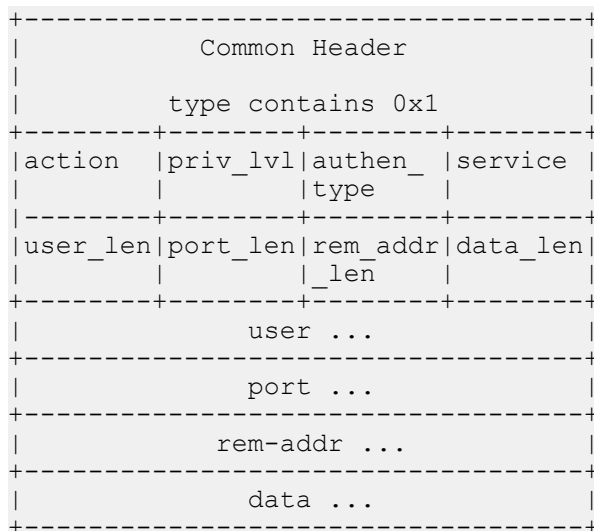
length

This 32-bit field contains the total length of the TACACS+ message, excluding the 12-octet header — in other words, the length of the message body.

Authentication START Packet

The Oracle CSM, acting as a TACACS+ client, sends an authentication START packet to the TACACS+ daemon to initiate an authentication session. The daemon must respond with a REPLY packet.

The authentication START packet format is as follows.

**action**

This 8-bit field contains an enumerated value that identifies the requested authentication action. For the current TACACS+ implementation, this field always contains a value of 0x01 , indicating user login authentication.

priv_lvl

This 8-bit field contains an enumerated value that identifies the privilege level requested by an authenticating user. For the current TACACS+ authentication implementation, this field always contains a value of 0x01 , indicating the user level.

authen-type

This 8-bit field contains an enumerated value that identifies the authentication methodology. Supported values are as follows:

0x01 ASCII — simple login, Oracle CSM prompts for username and password

0x02 PAP — as specified in RFC 1334

0x03 CHAP — as specified in RFC 1994

service

This 8-bit field contains an enumerated value that identifies the service requesting the authentication. For the current TACACS+ implementation, this field always contains a value of 0x01 , indicating user login authentication.

user_len

This 8-bit field contains the length of the user field in octets.

port_len

This 8-bit field contains the length of the port field in octets. As the port field is not used in the current TACACS+ authentication implementation, the port_len field always contains a value of 0 as specified in Section 4 of the TACACS+ draft RFC.

rem_addr_len

This 8-bit field contains the length of the rem_addr field in octets. As the rem_addr field is not used in the current TACACS+ authentication implementation, the rem_addr_len field always contains a value of 0 as specified in Section 4 of the TACACS+ draft RFC.

data_len

This 8-bit field contains the length of the data field in octets.

user

This variable length field contains the login name of the user to be authenticated.

port

This variable length field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string tty10 .

rem_addr

This variable length field contains the location of the user to be authenticated. This field contains the localhost address.

data

This optional variable length field contains miscellaneous data.

Authentication REPLY Packet

The TACACS+ daemon sends an authentication REPLY packet to the Oracle CSM in response to a authentication START or authentication CONTINUE packet. Depending on the contents of the status field, the authentication REPLY packet either ends the authentication transaction, or continues the transaction by requesting addition information needed by the authenticator.

The authentication REPLY packet format is as follows.

```
+-----+
|           Common Header           |
|           type contains 0x1       |
+-----+-----+-----+-----+
| (type field contains 0x1)         |
+-----+-----+-----+-----+
| status | flags | server_msg_len |
+-----+-----+-----+-----+
| data_len | server_msg ... |
+-----+-----+-----+-----+
|           data ...               |
+-----+
```

status

This 16-bit field contains an enumerated value that specifies the current state of the authentication process. Supported values are as follows:

0x01 PASS — the user is authenticated, thus ending the session

0x02 FAIL — the user is rejected, thus ending the session

0x04 GETUSER — daemon request for the user name

0x05 GETPASS — daemon request for the user password

0x06 RESTART — restarts the transaction, possibly because the sequence number has wrapped, or possibly because the requested authentication type is not supported by the daemon

0x07 ERROR — reports an unrecoverable error

flags

This 8-bit field contains various flags that are not under user control.

server_msg_len

This 16-bit field contains the length of the server_msg field in octets. As the server_msg field is not used in REPLY packets sent by the current TACACS+ authentication implementation, the server_msg_len field always contains a value of 0 as specified in Section 4 of the TACACS+ draft RFC.

data_len

This 16-bit field contains the length of the data field in octets. As the data field is not used in REPLY packets sent by the current TACACS+ authentication implementation, the data_len field always contains a value of 0 as specified in Section 4 of the TACACS+ draft RFC.

server_msg

This optional variable length field contains a server message intended for display to the user. The current TACACS+ authentication implementation does not use this field.

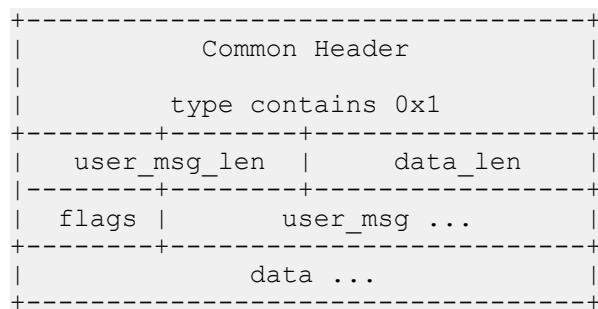
data

This optional variable length field contains data pertinent to the authentication process. The current TACACS+ authentication implementation does not use this field.

Authentication CONTINUE Packet

The Oracle CSM, acting as a TACACS+ client, sends an authentication CONTINUE packet to the TACACS+ daemon in response to a REPLY message which requested additional data required by the authenticator.

The authentication CONTINUE packet format is as follows.



user_msg_len

This 16-bit field contains the length of the user_msg field in octets.

data_len

This 16-bit field contains the length of the data field in octets. As the data field is not used in the current TACACS+ authentication implementation, the data field always contains a value of 0 as specified in Section 4 of the TACACS+ draft RFC.

flags

This 8-bit field contains various flags that are not under user control.

user_msg

Getting Started with the Oracle CSM

This variable length field contains a string that responds to an information request contained in a REPLY message.
data

This optional variable length field contains miscellaneous data, often in response to a daemon request. The current TACACS+ authentication implementation does not use the data field in Authentication CONTINUE packets.

Authentication Scenarios

Each of the supported user authentication scenarios is described in terms of packet flow in the following sections.

ASCII Authentication

The Oracle CSM initiates the authentication with an authentication START packet.

```
+-----+
|          Common Header          |
|  minor_version contains 0x0    |
|          type contains 0x1     |
+-----+
| action |priv_lvl|authen_|service |
|  0x01  |  0x01  |  0x01  |  0x01  |
+-----+
| user_len|port_len|rem_addr|data_len|
|    0    |    N    |   _len |    0    |
+-----+
|          port                   |
|          tty10                  |
+-----+
|          rem_addr               |
|          localhost address     |
+-----+
```

- The action field specifies the requested authentication action — 0x01 for TAC_PLUSAUTHEN_LOGIN (authentication of a user login).
- The priv_lvl field specifies the privilege level requested by the user — 0x01 for TAC_PLUS_PRIV_LVL_USER.
- The authen_type field specifies the authentication methodology — 0x01 for TAC_PLUS_AUTHEN_TYPE_ASCII (simple login).
- The service field specifies the requesting service — 0x01 for TAC_PLUS_AUTHEN_SVC_LOGIN (login service).
- The user_len and data_len fields contain a value of 0, as required by the TACACS+ protocol.
- The port_len and rem_addr_len fields contain the length, in octets, of the port and rem_addr fields.
- The port field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string tty10.
- The rem_addr field specifies the location of the user to be authenticated. This field contains the localhost address.

The TACACS+ daemon returns an authentication REPLY requesting the username.

```
+-----+
|          Common Header          |
|  minor_version contains 0x0    |
|          type contains 0x1     |
+-----+
| status |  flags | server_msg_len |
|  0x04  |       |           0    |
+-----+
| data_len |
|    0     |
+-----+
```

- The status field specifies a daemon request — 0x04 for TAC_PLUS_AUTH_STATUS_GETUSER (get username).

- The `server_msg_len` `data_len` fields both contain a value of 0 , as required by the TACACS+ protocol.

The Oracle CSM responds with an authentication CONTINUE packet.

Common Header		
minor_version contains 0x0		
type contains 0x1		
user_msg_len	data_len	
	0	
flags	user_msg ...	

- The `user_msg_len` field contains the length, in octets, of the `user_msg` field.
- The `data_len` field contains a value of 0 , as required by the TACACS+ protocol.
- The `user_msg` field contains the username to be authenticated.

The TCACS+ daemon returns a second authentication REPLY requesting the user password.

Common Header		
minor_version contains 0x0		
type contains 0x1		
status	flags	server_msg_len
0x05		0
data_len		
0		

- The status field specifies a daemon request — 0x05 for TAC_PLUS_AUTH_STATUS_GETPASS (get user password).
- The `server_msg_len` and `data_len` fields both contain a value of 0 , as required by the TACACS+ protocol.

The Oracle CSM responds with a second authentication CONTINUE packet.

Common Header		
minor_version contains 0x0		
type contains 0x1		
user_msg_len	data_len	
	0	
flags	user_msg ...	

- The `user_msg_len` field contains the length, in octets, of the `user_msg` field.
- The `data_len` field contains a value of 0 , as required by the TACACS+ protocol.
- The `user_msg` field contains the user password to be authenticated.
- Other, optional fields are not used.

The TCACS+ daemon returns a third authentication REPLY reporting the authentication result, and terminating the authentication session.

Common Header		
minor_version contains 0x0		
type contains 0x1		
status	flags	server_msg_len
0x01		0

Getting Started with the Oracle CSM

```
-----+
| data_len |
| 0        |
|-----+
```

- The status field specifies the authentication result — 0x01 for TAC_PLUS_AUTH_STATUS_PASS (authorization succeeds), or 0x02 for TAC_PLUS_AUTH_STATUS_FAIL (authorization fails).
- The server_msg_len, and data_len fields both contain a value of 0, as required by the TACACS+ protocol.

PAP Authentication

The Oracle CSM initiates the authentication with an authentication START packet.

```
-----+
| Common Header |
| minor_version contains 0x1 |
| type contains 0x1 |
|-----+
| action | priv_lvl | authen_ | service |
| 0x01   | 0x01    | type    | 0x01    |
|-----+
| user_len | port_len | rem_addr | data_len |
| N        | N        | _len    | N        |
|-----+
| user |
|-----+
| port |
| tty10 |
|-----+
| rem_addr |
| localhost address |
|-----+
| data ... |
|-----+
```

- The action field specifies the requested authentication action — 0x01 for TAC_PLUSAUTHEN_LOGIN (authentication of a user login).
- The priv_lvl field specifies the privilege level requested by the user — 0x01 for TAC_PLUS_PRIV_LVL_USER.
- The authen_type field specifies the authentication methodology — 0x02 for TAC_PLUS_AUTHEN_TYPE_PAP (PAP login).
- The service field specifies the requesting service — 0x01 for TAC_PLUS_AUTHEN_SVC_LOGIN (login service).
- The user_len field contains the length, in octets, of the user field.
- The port_len field contains the length, in octets, of the port field.
- The rem_addr_len field contains the length, in octets, of the rem_addr field.
- The data_len field contains the length, in octets, of the data field.
- The user field contains the username to be authenticated.
- The port field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string tty10.
- The rem_addr field specifies the location of the user to be authenticated. This field contains the localhost address.
- The data field contains the password to be authenticated.

The TACACS+ daemon returns an authentication REPLY reporting the authentication result.

```
-----+
| Common Header |
| minor_version contains 0x1 |
| type contains 0x1 |
|-----+
| status | flags | server_msg_len |
|-----+
```

0x01	0

data_len	
0	

- The status field specifies the authentication result — 0x01 for TAC_PLUS_AUTH_STATUS_PASS (authorization succeeds), or 0x02 for TAC_PLUS_AUTH_STATUS_FAIL (authorization fails).
- The server_msg_len and data_len fields both contain a value of 0, as required by the TACACS+ protocol.
- Other, optional fields are not used.

CHAP Authentication

The Oracle CSM initiates the authentication with an authentication START packet.

Common Header			
minor_version contains 0x1			
type contains 0x1			

action	priv_lvl	authen_	service
		type	
0x01	0x01	0x03	0x01

user_len	port_len	rem_addr_	data_len
		_len	
N	N	N	N

user			

port			
tty10			

rem_addr			
localhost address			

data ...			

- The action field specifies the requested authentication action — 0x01 for TAC_PLUSAUTHEN_LOGIN (authentication of a user login).
- The priv_lvl field specifies the privilege level requested by the user — 0x01 for TAC_PLUS_PRIV_LVL_USER.
- The authen_type field specifies the authentication methodology — 0x03 for TAC_PLUS_AUTHEN_TYPE_CHAP (CHAP login).
- The service field specifies the requesting service — 0x01 for TAC_PLUS_AUTHEN_SVC_LOGIN (login service).
- The user_len field contains the length, in octets, of the user field.
- The port_len field contains the length, in octets, of the port field.
- The rem_addr_len field contains the length, in octets, of the rem_addr field.
- The data_len field contains the length, in octets, of the data field.
- The user field contains the username to be authenticated.
- The port field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string tty10.
- The rem_addr field specifies the location of the user to be authenticated. This field contains the localhost address.
- The data field contains the password to be authenticated.

The TACACS+ daemon returns an authentication REPLY reporting the authentication result.

Common Header	
minor_version contains 0x1	

Getting Started with the Oracle CSM

type contains 0x1		
status	flags	server_msg_len
0x01		0
data_len		
0		

- The status field specifies the authentication result — 0x01 for TAC_PLUS_AUTH_STATUS_PASS (authorization succeeds), or 0x02 for TAC_PLUS_AUTH_STATUS_FAIL (authorization fails).
- The server_msg_len and data_len fields both contain a value of 0, as required by the TACACS+ protocol.
- Other, optional fields are not used.

TACACS+ Authorization

The Oracle CSM uses TACACS+ services to provide administrative authorization. With TACACS+ authorization enabled, each individual ACLI command issued by an admin user is authorized by the TACACS+ authorization service. The Oracle CSM replicates each ACLI command in its entirety, sends the command string to the authorization service, and suspends command execution until it receives an authorization response. If TACACS+ grants authorization, the pending command is executed; if authorization is not granted, the Oracle CSM does not execute the ACLI command, and displays an appropriate error message.

The daemon's authorization decisions are based on a database lookup. Data base records use regular expressions to associate specific command string with specific users. The construction of such records is beyond the scope of this document.

Authorization Message Exchange

All TACACS+ authorization packets consist of a common header and a message body. Authorization packets are of two types: REQUEST and RESPONSE.

The REQUEST packet, which initiates an authorization session, is always sent by the Oracle CSM. Upon receipt of every REQUEST, the daemon must answer with a RESPONSE packet. In the current TACACS+ implementation, the RESPONSE packet must contain an authorization decision (pass or fail). The exchange of a single REQUEST and the corresponding RESPONSE completes the authorization session.

Authorization REQUEST Packet

The Oracle CSM, acting as a TACACS+ client, sends an authorization REQUEST packet to the TACACS+ daemon to initiate an authorization session.

The authorization REQUEST packet format is as follows.

Common Header			
type contains 0x2			
authen_	priv_lvl	authen_	authen-
method		type	service
user_len	port_len	rem_addr	arg_cnt
		_len	
arg1_len	arg2_len	...	argN_len
user ...			
port ...			
rem-addr ...			

arg1 ...
arg2 ...
argN ...

authen_method

This 8-bit field contains an enumerated value that identifies the method used to authenticate the authorization subject — that is, an admin user. Because the admin user was authenticated locally by the Oracle CSM, this field always contains a value of 0x05 , indicating authentication by the requesting client.

priv_lvl

This 8-bit field contains an enumerated value that identifies the privilege level associated with the authorization subject. For the current TACACS+ authorization implementation, this field always contains a value of 0x00 .

authen-type

This 8-bit field contains an enumerated value that identifies the methodology. used to authenticate the authorization subject. Because the admin user was authenticated with a simple username/password exchange, this field always contains a value of 0x01 , indicating ascii login.

authen_service

This 8-bit field contains an enumerated value that identifies the service that requested authentication. Because an admin user is authenticated with a simple username/password exchange, this field always contains a value of 0x01 , the login service.

user_len

This 8-bit field contains an integer that specifies the length, in octets, of the user field.

port_len

This 8-bit field contains an integer that specifies the length, in octets, of the port field.

rem_addr_len

This 8-bit field contains an integer that specifies the length, in octets, of the rem_addr field.

arg_cnt

This 8-bit field contains an integer that specifies the number or arguments contained with the REQUEST. Given the design of the current TACACS+ implementation, this field always contains a value of 0x02 .

arg1_len

This 8-bit field contains an integer that specifies the length, in octets, of the first argument.

Subsequent fields contain the length of each sequential argument.

user

This variable length field contains the login name of the user to be authorized.

port

This variable length field contains the name of the Oracle CSM port on which authorization is taking place. Following Cisco Systems convention, this field contains the string tty10 .

rem_addr

This variable length contains the location of the user to be authorized. This field contains the localhost address.

arg...

This variable length field contains a TACACS+ attribute value pair (AVP); each arg field holds a single AVP.

Getting Started with the Oracle CSM

A TACACS+ AVP is an ASCII string with a maximum length of 255 octets. The string consists of the attribute name and its assigned value separated by either an equal sign (=) or by an asterisk (*). The equal sign (=) identifies a mandatory argument, one that must be understood and processed by the TACACS+ daemon; the asterisk (*) identifies an optional argument that may be disregarded by either the client or daemon.

Administrative authorization requires the use of only two TACACS+ AVPs: service and cmd .

The service AVP identifies the function to be authorized. In the case of the current implementation, the attribute value is always shell . Consequently the attribute takes the follow format:

```
service=shell
```

The cmd AVP identifies the specific ACLI command to be authorized. The command is passed in its entirety, from the administrative configuration root, configure terminal, through the final command argument. For example,

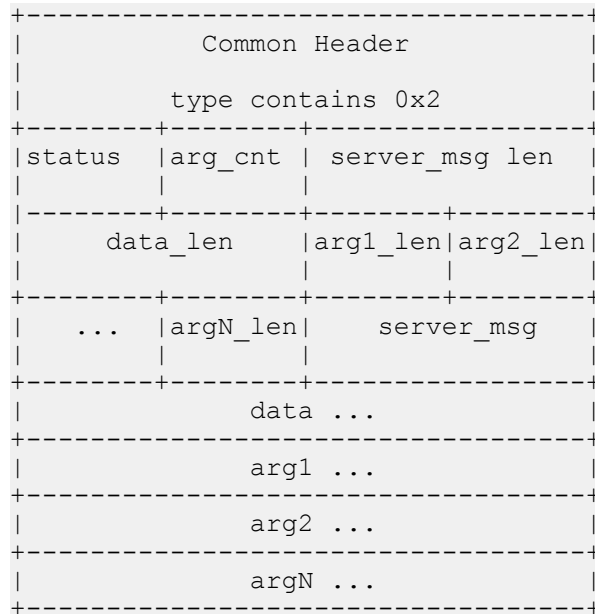
```
cmd=configure terminal security authentication type tacacsplus
```

Note the equal sign (=) used in the attribute examples, indicating that both are mandatory arguments.

Authorization RESPONSE Packet

The TACACS+ daemon sends an authorization RESPONSE packet to the Oracle CSM to report authorization results.

The authorization RESPONSE packet format is as follows.



status

This 8-bit field contains an enumerated value that specifies the results of the authorization process. Supported values are 0x01 (Pass), 0x10 (Fail), and 0x11 (Error). Fail indicates that the authorization service rejected the proposed operation, while Error indicates the authorization service failed.

If authorization succeeds (status=0x01), the ACLI command is executed; if authorization fails, for whatever the reason (status=0x10 or 0x11), the ACLI command is not executed, and an appropriate error message is generated.

arg_cnt

This 8-bit field contains an integer that specifies the number of arguments contained with the RESPONSE. Given the design of the current TACACS+ implementation, this field always contains a value of 0x02 .

server_msg_len

This 16-bit field contains an integer that specifies the length, in octets, of the server_msg field.

data_len

This 16-bit field contains an integer that specifies the length, in octets, of the data field.

arg1_len

This 8-bit field contains an integer that specifies the length, in octets, of the first argument.

Subsequent fields contain the length of each sequential argument.

server-msg

This optional variable length field contains a string that can be presented to the user.

data

This optional variable length field contains a string that can be presented to an administrative display, console, or log.

arg...

This optional variable length field contains a TACACS+ attribute value pair (AVP); each arg field holds a single AVP.

No arguments are generated in RESPONSE packets within the current TACACS+ implementation.

Authorization Scenarios

Successful and failed administrative authorization is described in terms of packet flow in the following sections.

Authorization Pass

The Oracle CSM initiates the authorization with an authorization REQUEST packet.

Common Header			
type contains 0x2			
authen_	priv_lvl	authen_	authen_
method		type	service
0x05	0x00	0x01	0x01
user_len	port_len	rem_addr	arg_cnt
N	N	len	2
arg1_len	arg2_len	user ...	
N	N	login name	
port			
tty10			
rem_addr			
localhost address			
arg1			
AVP			
service=shell			
arg2			
AVP			
cmd=configure terminal security			

- The `authen_method` field specifies the method used to authenticate the subject — 0x05 for `TAC_PLUS_AUTHEN_METHOD_LOCAL` (authentication by the client).
- The `priv_lvl` field specifies the privilege level requested by the user — 0x00 for `TAC_PLUS_PRIV_LVL_MIN`.
- The `authen_type` field specifies the authentication methodology — 0x01 for `TAC_PLUS_AUTHEN_TYPE_ASCII` (simple login).

Getting Started with the Oracle CSM

- The `authen_service` field specifies the requesting service — 0x01 for TAC_PLUS_AUTHEN_SVC_LOGIN (login service).
- The `user_len` field contains the length, in octets, of the user field.
- The `port_len` field contains the length, in octets, of the port field.
- The `rem_addr_len` field contains the length, in octets, of the `rem_addr` field.
- The `arg_cnt` field contains the number of arguments in the message body.
- The `arg1_len` field contains the length, in octets, of the service AVP.
- The `arg2_len` field contains the length, in octets, of the service AVP.
- The `user` field contains the login name of an admin user.
- The `port` field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string `tty10`.
- The `rem_addr` field specifies the location of the user to be authenticated. This field contains the localhost address.
- The `arg1` field contains the mandatory service AVP.
- The `arg2` field contains the mandatory cmd AVP.

The TCACS+ daemon returns a authorization RESPONSE reporting the status, and terminating the authorization session.

```
+-----+
|           Common Header           |
|           type contains 0x2       |
+-----+-----+-----+
| status | arg_cnt | server_msg_len |
| 0x01  | 0      | 0              |
+-----+-----+-----+
|           data_len               |
|           0                      |
+-----+
```

- The `status` field specifies the authorization status — 0x01 for TAC_PLUS_AUTHOR_STATUS_PASS_ADD (authorization approved).
- The `arg_cnt` field contains a value of 0 — the authorization RESPONSE returns no arguments.
- The `server_msg_len` and `data_len` fields both contain a value of 0, as required by the TACACS+ protocol.

Authorization Fail

The Oracle CSM initiates the authorization with an authorization REQUEST packet.

```
+-----+
|           Common Header           |
|           type contains 0x2       |
+-----+-----+-----+-----+
| authen_ | priv_lvl | authen_ | authen_ |
| method  |         | type    | service |
| 0x05    | 0x00    | 0x01    | 0x01    |
+-----+-----+-----+-----+
| user_len | port_len | rem_addr | arg_cnt |
|         |         | _len    |         |
| N       | N       | N       | 2       |
+-----+-----+-----+-----+
| arg1_len | arg2_len | user ... |
|         |         | login name |
| N       | N       |         |
+-----+-----+-----+-----+
|           port                   |
|           tty10                  |
+-----+-----+-----+-----+
|           rem_addr               |
|           localhost address      |
+-----+
```

```

+-----+
|          arg1          |
|          AVP           |
|          service=shell |
+-----+
|          arg2          |
|          AVP           |
|          cmd=configure |
|          terminal      |
|          scurity      |
+-----+

```

- The `authn_method` field specifies the method used to authenticate the administrative subject — 0x05 for TAC_PLUS_AUTHEN_METHOD_LOCAL (authentication by the client).
- The `priv_lvl` field specifies the privilege level requested by the user — 0x00 for TAC_PLUS_PRIV_LVL_MIN.
- The `authn_type` field specifies the authentication methodology — 0x01 for TAC_PLUS_AUTHEN_TYPE_ASCII (simple login).
- The `authn_service` field specifies the requesting service — 0x01 for TAC_PLUS_AUTHEN_SVC_LOGIN (login service).
- The `user_len` field contains the length, in octets, of the user field.
- The `port_len` field contains the length, in octets, of the port field.
- The `rem_addr_len` field contains the length, in octets, of the rem-addr field.
- The `arg_cnt` field contains the number of arguments in the message body.
- The `arg1_len` field contains the length, in octets, of the service AVP.
- The `arg2_len` field contains the length, in octets, of the service AVP.
- The user field contains the login name of an admin user.
- The port field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string `tty10`.
- The `rem_addr` field specifies the location of the user to be authenticated. This field contains the localhost address.
- The `arg1` field contains the mandatory service AVP.
- The `arg2` field contains the mandatory cmd AVP.

The TCACS+ daemon returns an authorization RESPONSE reporting the status, and terminating the authorization session.

```

+-----+
|          Common Header          |
|          type contains 0x2      |
+-----+
| status | arg_cnt | server_msg_len |
| 0x10  | 0      | 0              |
+-----+
|          data_len              |
|          0                     |
+-----+

```

- The status field specifies the authorization status — 0x10 for TAC_PLUS_AUTHOR_STATUS_FAIL (authorization rejected).
- The `arg_cnt` field contains a value of 0 — the authorization RESPONSE returns no arguments.
- The `server_msg_len` and `data_len` fields both contain a value of 0, as required by the TACACS+ protocol.

TACACS+ Accounting

The Oracle CSM uses TACACS+ accounting to log administrative actions. With accounting enabled, each individual CLI command executed by an admin user is logged by the accounting service.

Accounting Message Exchange

All TACACS+ accounting packets consist of a common header and a message body. Accounting packets are of two types: REQUEST and REPLY.

Getting Started with the Oracle CSM

The REQUEST packet has three variant forms. The START variant initiates an accounting session; the STOP variant terminates an accounting session; the WATCHDOG variant updates the current accounting session. REQUEST packets are always sent by the Oracle CSM. Upon receipt of every REQUEST, the daemon must answer with a REPLY packet.

A TACACS+ accounting session proceeds as follows.

1. Immediately following successful authorization of an admin user, the Oracle CSM sends an accounting REQUEST START packet.
2. The daemon responds with an accounting REPLY packet, indicating that accounting has started.
3. For each ACLI command executed by an admin user, the Oracle CSM sends an accounting REQUEST WATCHDOG packet requesting accounting of the ACLI command. As the Oracle CSM sends the WATCHDOG only after an admin user's access to the ACLI command is authorized, the accounting function records only those commands executed by the user, not those commands for which authorization was not granted.
4. The daemon responds with an accounting REPLY packet, indicating that the ACLI operation has been recorded by the accounting function.
5. Steps 3 and 4 are repeated for each authorized ACLI operation.
6. Immediately following logout (or timeout) of an admin user, the Oracle CSM sends an accounting REQUEST STOP packet.
7. The daemon responds with an accounting REPLY packet, indicating that accounting has stopped.

Accounting REQUEST Packet

The Oracle CSM, acting as a TACACS+ client, sends an accounting REQUEST START variant to the TACACS+ daemon following the successful authorization of an admin user. It sends an accounting REQUEST WATCHDOG variant to the daemon following the authorization of an admin user's access to an ACLI command. It sends an accounting REQUEST STOP variant to the daemon at the conclusion of the ACLI session.

The accounting REQUEST packet format is as follows.

```
+-----+
|           Common Header           |
|           type contains 0x3       |
+-----+-----+-----+-----+
| flags | authen_ | priv_lvl | authen- |
|       | method  |         | type   |
+-----+-----+-----+-----+
| authen_ | user_len | port_len | rem_addr |
| service |         |         | _len    |
+-----+-----+-----+-----+
| arg_cnt | arg1_len | arg2_len | argN_len |
|         |         |         |         |
+-----+-----+-----+-----+
| argN_len |         | user ... |         |
+-----+-----+-----+-----+
|         |         | port ... |         |
+-----+-----+-----+-----+
|         |         | rem-addr ... |         |
+-----+-----+-----+-----+
|         |         | arg1 ... |         |
+-----+-----+-----+-----+
|         |         | arg2 ... |         |
+-----+-----+-----+-----+
|         |         | argN ... |         |
+-----+-----+-----+-----+
```

flags

This 8-bit field contains an enumerated value that identifies the accounting REQUEST variant.

0x2 — START

0x4 — STOP

0x8 — WATCHDOG

authen_method

This 8-bit field contains an enumerated value that identifies the method used to authenticate the accounting subject — that is, an admin user. Because an admin user is authenticated locally by the Oracle CSM, this field always contains a value of 0x05 , indicating authentication by the requesting client.

priv_lvl

This 8-bit field contains an enumerated value that identifies the privilege level associated with the accounting subject. For the current TACACS+ accounting implementation, this field always contains a value of 0x00 .

authen-type

This 8-bit field contains an enumerated value that identifies the methodology. used to authenticate the accounting subject. Because an admin user is authenticated with a simple username/password exchange, this field always contains a value of 0x01 , indicating ascii login.

authen_service

This 8-bit field contains an enumerated value that identifies the service that requested authentication. Because an admin user is authenticated with a simple username/password exchange, this field always contains a value of 0x01 , the login service.

user_len

This 8-bit field contains an integer that specifies the length, in octets, of the user field.

port_len

This 8-bit field contains an integer that specifies the length, in octets, of the port field.

rem_addr_len

This 8-bit field contains an integer that specifies the length, in octets, of the rem_addr field.

arg_cnt

This 8-bit field contains an integer that specifies the number of arguments contained with the accounting REQUEST.

arg1_len

This 8-bit field contains an integer that specifies the length, in octets, of the first argument.

Subsequent fields contain the length of each sequential argument.

user

This variable length field contains the login name of the accounting subject.

port

This variable length field contains the name of the Oracle CSM port on accounting is taking place. Following Cisco System convention, this field always contains the string tty10 .

rem_addr

This variable length contains the location of the authorization subject. This field always contains the localhost address.

arg...

This variable length field contains a TACACS+ attribute value pair (AVP); each arg field holds a single AVP.

A TACACS+ AVP is an ASCII string with a maximum length of 255 octets. The string consists of the attribute name and its assigned value separated by either an equal sign (=) or by an asterisk (*). The equal sign (=) identifies a mandatory argument, one that must be understood and processed by the TACACS+ daemon; the asterisk (*) identifies an optional argument that may be disregarded by either the client or daemon.

Getting Started with the Oracle CSM

Administrative accounting requires the use of five TACACS+ AVPs: service, task-id, start_time, and stop_time.

The task_id AVP, included in accounting REQUEST START, STOP, and WATCHDOG variants, correlates session initiation, watchdog updates, and termination packets; each associated START, STOP, and WATCHDOG packet must contain matching task-id AVPs.

```
task_id=13578642
```

The start_time AVP, included in accounting REQUEST START and WATCHDOG variants, specifies the time at which a specific accounting request was initiated. The start time is expressed as the number of seconds elapsed since January 1, 1970 00:00:00 UTC.

```
start_time=1286790650
```

The stop_time AVP, included in accounting REQUEST STOP variants, specifies the time at which a specific accounting session was terminated. The stop time is expressed as the number of seconds elapsed since January 1, 1970 00:00:00 UTC.

```
stop_time=1286794250
```

The service AVP, included in accounting REQUEST START, STOP, and WATCHDOG variants, identifies the function subject to accounting. In the case of the current implementation, the attribute value is always shell . Consequently the attribute takes the follow format:

```
service=shell
```

The cmd AVP, included in accounting REQUEST WATCHDOG variants, identifies the specific ACLI command to be processed by the accounting service. The command is passed in its entirety, from the administrative configuration root, configure terminal, through the final command argument. For example,

```
cmd=configure terminal security authentication type tacacsplus
```

Note the equal sign (=) used in the attribute examples, indicating that all are mandatory arguments.

Accounting REPLY Packet

The TACACS+ daemon sends an accounting REPLY packet to the Oracle CSM to report accounting results.

The accounting REPLY packet format is as follows.

```
+-----+
|           Common Header           |
|           type contains 0x3       |
+-----+-----+-----+
| server_msg_len | data_len |
+-----+-----+-----+
| status | server_msg ... |
+-----+-----+-----+
|           data ...           |
+-----+-----+-----+
```

server_msg_len

This 16-bit field contains the length, in octets, of the server_msg field.

data_len

This 16-bit field contains the length, in octets, of the data field.

status

This 8-bit field contains the status of the previous accounting request. Supported values are:

0x1 — Success

0x2 — Error/Failure

server_msg

This optional variable length field can contain a message intended for display to the user. This field is unused in the current TACACS+ implementation.

data

This optional variable length field can contain miscellaneous data. This field is unused in the current TACACS+ implementation.

Accounting Scenario

The Oracle CSM initiates the accounting session with an accounting REQUEST START.

```

+-----+
|           Common Header           |
|           type contains 0x3       |
+-----+-----+-----+-----+
| flags | authen_ | priv_lvl | authen- |
|       | method  |         | type    |
| 0x02 | 0x05   | 0x00   | 0x01   |
+-----+-----+-----+-----+
| authen_ | user_len | port_len | rem_addr |
| service |         |         | _len    |
| 0x01   | N       | N       | N       |
+-----+-----+-----+-----+
| arg_cnt | arg1_len | arg2_len | arg3_len |
| 3       | N       | N       | N       |
+-----+-----+-----+-----+
|           user                     |
| login name of an admin user       |
+-----+-----+-----+-----+
|           port                     |
| tty10                             |
+-----+-----+-----+-----+
|           rem_addr                 |
| localhost address                 |
+-----+-----+-----+-----+
|           AVP                      |
| task-id=13578642                 |
+-----+-----+-----+-----+
|           AVP                      |
| start_time=1286790650             |
+-----+-----+-----+-----+
|           AVP                      |
| service=shell                     |
+-----+-----+-----+-----+

```

- The flags field contains an enumerated value (0x02) that identifies an accounting REQUEST START.
- The authen_method field specifies the method used to authenticate the ACCOUNTING subject — 0x05 for TAC_PLUS_AUTHEN_METHOD_LOCAL (authentication by the client).
- The priv_lvl field specifies the privilege level requested by the user — 0x00 for TAC_PLUS_PRIV_LVL_MIN.
- The authen_type field specifies the authentication methodology — 0x01 for TAC_PLUS_AUTHEN_TYPE_ASCII (simple login).
- The authen_service field specifies the requesting service — 0x01 for TAC_PLUS_AUTHEN_SVC_LOGIN (login service).
- The user_len field contains the length, in octets, of the user field.
- The port_len field contains the length, in octets, of the port field.
- The rem_addr_len field contains the length, in octets, of the rem_addr field.
- The arg_cnt field contains the number of arguments in the message body.
- The arg1_len field contains the length, in octets, of the task_id AVP.
- The arg2_len field contains the length, in octets, of the start_time AVP.

Getting Started with the Oracle CSM

- The `arg3_len` field contains the length, in octets, of the service AVP.
- The `user` field contains the login name of an admin user.
- The `port` field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string `tty10`.
- The `rem_addr` field specifies the location of the user to be authenticated. This field contains the localhost address.
- The `arg1` field contains the mandatory `task_id` AVP.
- The `arg2` field contains the mandatory `start_time` AVP.
- The `arg3` field contains the mandatory service AVP.

The TACACS+ daemon returns an accounting REPLY reporting the status, indicating that accounting has started.

```

+-----+
|           Common Header           |
|           type contains 0x3       |
+-----+-----+
| server_msg_len | data_len |
|         0      |         0 |
+-----+-----+
| status         |
| 0x01          |
+-----+

```

- The `server_msg_len` and `data_len` fields both contain a value of 0, as required by the TACACS+ protocol.
- The `status` field specifies the authorization status — 0x01 for `TAC_PLUS_ACCT_STATUS_SUCCESS` (accounting processed).

The Oracle CSM reports ACLI command execution with an accounting REQUEST WATCHDOG.

```

+-----+-----+-----+-----+
|           Common Header           |
|           type contains 0x3       |
+-----+-----+-----+-----+
| flags | authen_ | priv_lvl | authen- |
|        | method  |          | type    |
| 0x08  | 0x05   | 0x00    | 0x01   |
+-----+-----+-----+-----+
| authen_ | user_len | port_len | rem_addr |
| service |          |          | _len    |
| 0x01   | N       | N       | N       |
+-----+-----+-----+-----+
| arg_cnt | arg1_len | arg2_len | arg3_len |
| 4      | N       | N       | N       |
+-----+-----+-----+-----+
| arg4_len |          | user    |
|          | login name of admin user |
+-----+-----+-----+-----+
|          |          | port    |
|          |          | tty10  |
+-----+-----+-----+-----+
|          |          | rem_addr |
|          |          | localhost address |
+-----+-----+-----+-----+
|          |          | AVP     |
|          |          | task-id=13578642 |
+-----+-----+-----+-----+
|          |          | AVP     |
|          |          | start_time=1286790650 |
+-----+-----+-----+-----+
|          |          | AVP     |
|          |          | service=shell |
+-----+-----+-----+-----+

```

```

|          AVP          |
| cmd=configure terminal security |
+-----+

```

- The flags field contains an enumerated value (0x08) that identifies an accounting REQUEST WATCHDOG.
- The authen_method field specifies the method used to authenticate the ACCOUNTING subject — 0x05 for TAC_PLUS_AUTHEN_METHOD_LOCAL (authentication by the client).
- The priv_lvl field specifies the privilege level requested by the user — 0x00 for TAC_PLUS_PRIV_LVL_MIN.
- The authen_type field specifies the authentication methodology — 0x01 for TAC_PLUS_AUTHEN_TYPE_ASCII (simple login).
- The authen_service field specifies the requesting service — 0x01 for TAC_PLUS_AUTHEN_SVC_LOGIN (login service).
- The user_len field contains the length, in octets, of the user field.
- The port_len field contains the length, in octets, of the port field.
- The rem_addr_len field contains the length, in octets, of the rem_addr field.
- The arg_cnt field contains the number of arguments in the message body.
- The arg1_len field contains the length, in octets, of the task_id AVP.
- The arg2_len field contains the length, in octets, of the start_time AVP.
- The arg3_len field contains the length, in octets, of the service AVP.
- The arg4_len field contains the length, in octets, of the cmd AVP.
- The user field contains the login name of an admin user.
- The port field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string tty10 .
- The rem_addr field specifies the location of the user to be authenticated. This field contains the localhost address.
- The arg1 field contains the mandatory task_id AVP.
- The arg2 field contains the mandatory start_time AVP.
- The arg3 field contains the mandatory service AVP.
- The arg4 field contains the mandatory cmd AVP.

The TCACS+ daemon returns an accounting REPLY reporting the status, indicating that the ACLI operation has been processed.

```

+-----+
|          Common Header          |
|          type contains 0x3      |
+-----+-----+
| server_msg_len | data_len |
|      0         |      0   |
+-----+-----+
| status |
| 0x01  |
+-----+

```

- The server_msg_len and data_len fields both contain a value of 0 , as required by the TACACS+ protocol.
- The status field specifies the authorization status — 0x01 for TAC_PLUS_ACCT_STATUS_SUCCESS (accounting processed).

The Oracle CSM reports an admin user logout or timeout with an accounting REQUEST STOP.

```

+-----+-----+-----+-----+
|          Common Header          |
|          type contains 0x3      |
+-----+-----+-----+-----+
| flags | authen_ | priv_lvl | authen- |
|       | method  |         | type    |
| 0x04 | 0x05   | 0x00   | 0x01   |
+-----+-----+-----+-----+

```

Getting Started with the Oracle CSM

authen_	user_len	port_len	rem_addr
service			_len
0x01	N	N	N

arg_cnt	arg1_len	arg2_len	arg3_len
3	N	N	N

user			
login name of an admin user			

port			
tty10			

rem_addr			
localhost address			

AVP			
task-id=13578642			

AVP			
stop_time=1286790650			

AVP			
service=shell			

- The flags field contains an enumerated value (0x04) that identifies an accounting REQUEST STOP.
- The authen_method field specifies the method used to authenticate the ACCOUNTING subject — 0x05 for TAC_PLUS_AUTHEN_METHOD_LOCAL (authentication by the client).
- The priv_lvl field specifies the privilege level requested by the user — 0x00 for TAC_PLUS_PRIV_LVL_MIN.
- The authen_type field specifies the authentication methodology — 0x01 for TAC_PLUS_AUTHEN_TYPE_ASCII (simple login).
- The authen_service field specifies the requesting service — 0x01 for TAC_PLUS_AUTHEN_SVC_LOGIN (login service).
- The user_len field contains the length, in octets, of the user field.
- The port_len field contains the length, in octets, of the port field.
- The rem_addr_len field contains the length, in octets, of the rem_addr field.
- The arg_cnt field contains the number of arguments in the message body.
- The arg1_len field contains the length, in octets, of the task_id AVP.
- The arg2_len field contains the length, in octets, of the start_time AVP.
- The arg3_len field contains the length, in octets, of the service AVP.
- The user field contains the login name of an admin user.
- The port field contains the name of the Oracle CSM port on which authentication is taking place. Following Cisco Systems convention, this field contains the string tty10 .
- The rem_addr field specifies the location of the user to be authenticated. This field contains the localhost address.
- The arg1 field contains the mandatory task_id AVP.
- The arg2 field contains the mandatory start_time AVP.
- The arg3 field contains the mandatory service AVP.

The TCACS+ daemon returns an accounting REPLY reporting the status, indicating that accounting has terminated.

Common Header	
type contains 0x3	

server_msg_len	data_len
0	0

```
| status |
| 0x01  |
+-----+
```

- The server_msg_len and data_len fields both contain a value of 0 , as required by the TACACS+ protocol.
- The status field specifies the authorization status — 0x01 for TAC_PLUS_ACCT_STATUS_SUCCESS (accounting processed).

TACACS+ Configuration

Configuration of TACACS+ consists of the following steps.

1. Enable TACACS+ client services
2. Specify one or more TACACS+ servers (daemons)



Note: TACACS servers must be reachable from the Oracle CSM's media interfaces. Communications to and from TACACS servers can not be from the management interfaces.

Enable TACACS+ Client Services

Use the following procedure to enable specific TACACS+ client AAA services.

1. Access the **authentication** configuration element.

```
ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# authentication
ORACLE(authentication)#
```

2. Use the required type attribute to specify the authentication protocol.

Supported values are diameter, local (the default, indicating that authentication determinations are referred to a local database), radius, and tacacs.

Specify tacacs to enable the TACACS+ AAA protocol.

```
ORACLE(authentication)# type tacacs
ORACLE(authentication)#
```

3. Use tacacs-authorization to enable or disable command-based authorization of admin users. Assuming type is set to tacacs , by default, authorization is enabled.

```
ORACLE(authentication)# tacacs-authorization enabled
ORACLE(authentication)#
```

4. Use tacacs-accounting to enable or disable accounting of admin CLI operations. Assuming type is set to tacacs , by default, accounting is enabled.

```
ORACLE(authentication)# tacacs-accounting enabled
ORACLE(authentication)#
```

5. Use the server-assigned-privilege attribute to enable a proprietary TACACS+ variant that, after successful user authentication, adds an additional TACACS+ request/reply exchange. During the exchange, the Security Gateway requests the privilege level of the newly authenticated user. In response, the TACACS+ daemon returns the assigned privilege level, either user or admin . user accounts are denied access to the enable command, thus barring them from configuration level commands.

By default, this attribute is disabled — meaning that no privilege level information is exchanged.

Set this attribute to enabled to initiate the proprietary variant behavior.

```
ORACLE(authentication)# server-assigned-privilege enabled
ORACLE(authentication)#
```

6. Use the management-strategy attribute to identify the selection algorithm used to choose among multiple available TACACS+ daemons.

Retain the default value (hunt) if only a single daemon is available.

Available algorithms are hunt and roundrobin .

- **hunt** — for the first transaction the Security Gateway selects the initially configured TACACS+ daemon. As long as that daemon is online and operational, the Security Gateway directs all AAA transactions to it. Otherwise, the Security Gateway selects the second-configured daemon. If the first and second daemons are offline or non-operational, the next-configured daemon is selected, and so on through the group of available daemons.
- **roundrobin** — for the first transaction the Security Gateway selects the initially configured TACACS+ daemon. After completing the first transaction, selects each daemon in order of configuration — in theory, evenly distributing AAA transactions to each daemon over time.

```
ORACLE (authentication) # management-strategy roundrobin
ORACLE (authentication) #
```

7. Type **done** to save your configuration.

Specify TACACS+ Servers


Use the following procedure to specify one or more TACACS+ servers (daemons).

1. Access the **tacacs-servers** configuration element.

```
ORACLE# configure terminal
ORACLE (configure) # security
ORACLE (security) # authentication
ORACLE (authentication) # tacacs-servers
ORACLE (tacacs-servers) #
```

2. Use the **address** attribute to specify the IP address of this TACACS+ daemon.

```
ORACLE (tacacs-servers) # address 172.30.0.6
ORACLE (tacacs-servers) #
```

 **Note:** This address must be reachable from a media interface. The TACACS server should not be on the management interface's network.

3. Use the **port** attribute to identify the daemon port that receives TACACS+ client requests.

Provide a port number within the range 1025 through 65535, or retain the default value, 49, the well-known TACACS+ port.

```
ORACLE (tacacs-servers) # port 49
ORACLE (tacacs-servers) #
```

4. Use the **state** attribute to specify the availability of this TACACS+ daemon.


Select **enabled** (the default) or **disabled**.

Only TACACS+ daemons that are in the enabled state are considered when running the server-selection algorithm.

```
ORACLE (tacacs-servers) # state enabled
ORACLE (tacacs-servers) #
```

5. Use the **realm-id** attribute to identify the realm that provides access to this TACACS+ daemon.

```
ORACLE (tacacs-servers) # realm-id accounting
ORACLE (tacacs-servers) #
```

 **Note:** This realm must be reachable from a media interface.

6. Retain the default value for the **authentication-methods** attribute to specify support for all TACACS+ authentication methods (pap, chap, and ascii).

- **ascii** — simple login, the Session Director prompts user for username and password
- **pap** — similar to ascii method, but username and password are encapsulated in a PAP header
- **chap** — authentication based on a shared-secret, which is not passed during the authentication process

```
ORACLE (tacacs-servers) # authentication-methods all
ORACLE (tacacs-servers) #
```

- Use the secret attribute to provide the shared-secret used by the TACACS+ client and the daemon to encrypt and decrypt TACACS+ messages. The identical shared-secret must be configured on associated TACACS+ clients and daemons.

Enter a 16-digit string, and ensure that the identical value is configured on the TACACS+ daemon.

```
ORACLE(tacacs-servers) # secret 1982100754609236
ORACLE(tacacs-servers) #
```

- Use the dead-time attribute to specify, in seconds, the quarantine period imposed upon TACACS+ daemons that become unreachable. Quarantined servers are not eligible to participate in the server-selection algorithm.

Supported values are integers within the range 10 through 10000 seconds, with a default value of 10 .

```
ORACLE(tacacs-servers) # dead-interval 120
ORACLE(tacacs-servers) #
```

- Type **done** to save your configuration.

- Repeat Steps 1 through 10 to configure additional TACACS+ daemons.



Note: After configuring TACACS+ daemons, complete TACACS+ configuration by compiling a list of available daemons.

- From superuser mode, use the following command sequence to access authentication configuration mode.

```
ORACLE# configure terminal
ORACLE(configure) # security
ORACLE(security) # authentication
ORACLE(authentication) #
```

- Use the management-servers attribute to identify one or more TACACS+ daemons available to provide AAA services.

Daemons are identified by IP address and must have been previously configured as described above.

The following example identifies three available TACACS+ daemons. Note that the list is delimited by left and right parentheses, and list items are separated by space characters.

```
ORACLE(tacacs-servers) # management-servers (172.30.0.6 172.30.1.8
172.30.2.10)
ORACLE(tacacs-servers) #
```

The following example deletes the current list.

```
ORACLE(tacacs-servers) # management-servers ()
ORACLE(tacacs-servers) #
```

Managing TACACS+ Operations

TACACS+ management is supported by the following utilities.

TACACS+ MIB

An Oracle proprietary MIB provides external access to TACACS+ statistics.

MIB counters are contained in the apSecurityTacacsPlusStatsTable that is defined as follows.

```
SEQUENCE {
  apSecurityTacacsPlusCliCommands           Counter32
  apSecurityTacacsPlusSuccess Authentications Counter32
  apSecurityTacacsPlusFailureAuthentications Counter32
  apSecurityTacacsPlusSuccess Authorizations Counter32
  apSecurityTacacsPlusFailureAuthorizations Counter32
}
```

Getting Started with the Oracle CSM

apSecurityTacacsPlusStats Table (1.3.6.1.4.1.9148.3.9.9.4)		
Object Name	Object OID	Description
apSecurityTacacsCliCommands	1.3.6.1.4.1.9148.3.9.1.4.3	Global counter for ACLI commands sent to TACACS+ Accounting
apSecurityTacacsSuccess Authentications	1.3.6.1.4.1.9148.3.9.1.4.4	Global counter for the number of successful TACACS+ authentications
apSecurityTacacsFailureAuthentications	1.3.6.1.4.1.9148.3.9.1.4.5	Global counter for the number of unsuccessful TACACS+ authentications
apSecurityTacacsSuccess Authorizations	1.3.6.1.4.1.9148.3.9.1.4.6	Global counter for the number of successful TACACS+ authorizations
apSecurityTacacsFailure Authorizations	1.3.6.1.4.1.9148.3.9.1.4.7	Global counter for the number of unsuccessful TACACS+ authorizations

SNMP Trap

SNMP traps are issued when

- a TACACS+ daemon becomes unreachable
- an unreachable TACACS+ daemon becomes reachable
- an authentication error occurs
- an authorization error occurs

ACLI show Command

The show tacacs stats command displays the following statistics.

- number of ACLI commands sent for TAGACS+ accounting
- number of successful TACACS+ authentications
- number of failed TACACS+ authentications
- number of successful TACACS+ authorizations
- number of failed TACACS+ authentications
- the IP address of the TACACS+ daemon used for the last transaction

TACACS+ Logging

All messages between the Oracle CSM and the TACACS+ daemon are logged in a cleartext format, allowing an admin user to view all data exchange, except for password information.

Customizing Your ACLI Settings

This section describes several ways you can customize the way you log into the ACLI and the way the ACLI displays information. Where applicable, these descriptions also contain instructions for configuration.

Disabling the Second Login Prompt

With this feature enabled, the Oracle logs you in as a Superuser (i.e., in administrative mode) regardless of your configured privilege level for either a Telnet or an SSH session. However, if you log via SSH, you still need to enter the password for local or RADIUS authentication.

Disabling the Second Login Prompt Configuration

You disable the second login prompt in the authentication configuration.

To disable the second login prompt:

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

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

2. Type `security` and press Enter.

```
ORACLE (configure) # security
ORACLE (security) #
```

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

```
ORACLE (security) # authentication
ORACLE (authentication) #
```

4. `login-as-admin`—Set this parameter to `enabled` if you want users to be logged automatically in Superuser (administrative) mode. The default for this parameter is `disabled`.
5. Save and activate your configuration.

Persistent ACLI more Parameter

To make using the ACLI easier, the Oracle CSM provides a paging feature controlled through the `ACLI cli more` command (which you can set to `enabled` or `disabled`). Disabled by default, this feature allows you to control how the Oracle CSM displays information on your screen during a console, Telnet, or SSH session. This command sets the paging feature on a per session basis.

Customers who want to set the paging feature so that settings persist across sessions with the Oracle CSM can set a configuration parameter that controls the paging feature. Enabling this parameter lets you set your preferences once rather than having to reset them each time you initiate a new session with the Oracle CSM.

Persistent ACLI more Parameter Configuration

To set the persistent behavior of the ACLI more feature across sessions:

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

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

2. Type `system` and press Enter.

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

3. Type `system-config` and press Enter.

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

If you are adding this feature to an existing configuration, you need to select the configuration (using the `ACLI select` command) before making your changes.

4. `cli-more`—Set this parameter to `enabled` if you want the ACLI more paging feature to work persistently across console, Telnet, or SSH sessions with the Oracle CSM. If you want to continue to set this feature on a per session basis, leave this parameter set to `disabled` (default).
5. Save and activate your configuration.

Customized Login Banner

A text file can be put on the Oracle CSM to be used as a banner to be printed before each login. The file must be called `/code/banners/banner.txt`. The contents of this file will be printed before each User Access Verification login.

Getting Started with the Oracle CSM

sequence. The limits are that no more than 79 characters per line and no more than 20 lines from the banner.txt file will be printed.

The banner.txt file used for the ACLI customized login banner has to be saved in the /code/banners directory. If that directory does not already exist on your system, you do not have to create the directory prior to placing the banner file because the Oracle CSM will create it upon boot if it does not exist.

System Configuration

This chapter explains how to configure system-level functionality for the Oracle CSM. Both physical and network interfaces as well as general system parameters are required to configure your Oracle CSM for service. Accounting functionality, SNMP configurations, trap configurations, and host routes are optional.

The following configurations are explained in this chapter:

- General system parameters—used for operating and identification purposes. In general, the informational fields have no specific effect on services, but are important to keep populated. The default gateway parameter is included here. It requires special attention since its configuration is dependent on the type of traffic the Oracle CSM is servicing.
- Physical and network interfaces—enables the Oracle CSM to communicate with any network element. Interfaces are one of the most basic configurations you need to create.
- SIP Interfaces and Ports—creates an interface on the Oracle CSM through which the system can send, receive and process SIP messages.
- SNMP—used for monitoring system health throughout a network.
- Syslogs and Process logs—used to save a list of system events to a remote server for analysis and auditing purposes.
- Host routes—used to instruct the Oracle CSM host how to reach a given network that is not directly connected to a local network interface.

General System Information

This section explains the parameters that encompass the general system information on a Oracle CSM.

System Identification

Global system identification is used primarily by the Oracle CSM to identify itself to other systems and for general identification purposes.

Connection Timeouts

It is important to set administrative session timeouts on the Oracle CSM for security purposes. If you leave an active configuration session unattended, reconfiguration access is left open to anyone. By setting a connection timeout, only a short amount of time needs to elapse before the password is required for Oracle CSM access.

System Configuration

Timeouts determine the specified time period that must pass before an administrative connection is terminated. Any subsequent configuration activity can only be performed after logging in again to the Oracle CSM. The timeout parameter can be individually specified for Telnet sessions and for console port sessions.

After the Telnet timeout passes, the Telnet session is disconnected. You must use your Telnet program to log in to the Oracle CSM once again to perform any further configuration activity.

After the console timeout passes, the console session is disconnected. The current session ends and you are returned to the login prompt on the console connection into the Oracle CSM.

Configuring General System Information

This section explains how to configure the general system parameters, timeouts, and the default gateway necessary to configure your Oracle CSM.

To configure general system information:

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
```

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

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

The following is an example what a general system information configuration might look like. Parameters not described in this section are omitted below.

```
ORACLE(system-config)# show
system-config
      hostname                test1
      description             Example SD
      location                 Row 3, Rack 4, Slot 451
      default-gateway          10.0.2.1
      telnet-timeout           1000
      console-timeout          1000
      last-modified-date       2004-12-08 20:15:43
```

When showing a single-instance configuration element such as `system-config`, you must first use the `select` command to select the configuration element prior to viewing.

System Identification

You must specify identification parameters for this Oracle CSM.

Set the following parameters to configure the system identification:

1. `hostname`—Set the primary hostname used to identify the system. This parameter is used by the software for informational purposes.
2. `description`—Enter a textual description of the system. This parameter is used for informational purposes.
3. `location`—Set a location description field for your system. This parameter is used for informational purposes. For example, you could include the site name and address of the location where the Oracle CSM system chassis is located.
4. `default-gateway`—Set the default gateway for this Oracle CSM. This is the egress gateway for traffic without an explicit destination. The application of your Oracle CSM determines the configuration of this parameter.

Configuring Connection and Debug Logging Timeouts

Configure the timeouts for terminal sessions on this Oracle CSM. These parameters are optional.

Set the following parameters to configure the connection timeouts:

1. `telnet-timeout`—Set the Telnet timeout to the number of seconds you want the Oracle CSM to wait before it disconnects a Telnet session or an SSH session. The default value is 0. The valid range is:
 - Minimum—0
 - Maximum—65535
2. `console-timeout`—Set the console timeout to the number of seconds you want the Oracle CSM to wait before it ends the console session. The default value is 0. The valid range is:
 - Minimum—0
 - Maximum—65535
3. `debug-timeout`—Set the time in seconds you want to use for the debug timeout. This is the time allowed before the Oracle CSM times out log levels for system processes set to debug using the ACLI `notify` and `debug` commands.

This command does not affect log levels set in your configuration (using parameters such as `system-config>process-log-level`) or those set using the ACLI `log-level` command.

The valid range is:

- Minimum—0
- Maximum—65535

Interface Considerations for Netra Platforms

Oracle CSM management and media traffic use separate network interfaces to connect to networks. The user configures these interfaces' properties, and then adapts other configuration elements to those interfaces. This configuration requires an understanding of the platform's mechanisms for using interfaces as well as the properties they can configure, as presented in this section.

Software Adaptations for Physical Interfaces

Having been installed on a virtual machine or Netra, Oracle CSM physical interface configuration elements provide for the same properties and statistics measurements as are available for Acme Packet platform interfaces. With the exception of the MACTAB file, the abstraction of these elements from those physical ports is transparent to the user.

The user must independently understand the bus or virtual network paths used by their hardware to access physical ports. Refer to your hardware or virtual machine documentation for information about proper configuration and operation related to, for example, physical port location and speed. The user must consider these system and physical details when configuring Oracle CSM **physical-interface** elements.

VM Interfaces

Virtual machines need to interface with their virtual machine management systems to reach and share physical interfaces with other virtual machines. The Oracle CSM assigns a unique pseudo MAC address from a virtual machine management system and maps it to the Oracle CSM configuration naming syntax to direct network traffic to physical interfaces based on the type of service they support.

COTS Interfaces

Physical interfaces on COTS platforms are defined by the platform, with bussing and driver access fully independent of the Oracle CSM. Utilization of COTS interfaces by the Oracle CSM is the same as with VM interfaces. The Oracle CSM extracts physical interface MAC addressing from the platform and maps it to Oracle CSM configuration naming syntax to direct network traffic to physical interfaces based on the type of service they support.

COTS Network Interfaces

The Oracle CSM maps interfaces upon first startup, based on the hardware's NIC configuration. The software looks to configure 2 management and 2 service ports for the common 4 on-board NIC configuration. The presence of additional NIC cards maps an additional management interface to the on-board NICs, leaving only a single service

System Configuration

interface on the on-board NIC. The Oracle CSM provides a means of testing and changing physical interface assignment.

The Oracle CSM instantiates the first management port, eth0/wancom0, when it boots. The boot parameters define this addressing to enable initial management access via IP for purposes including network boot. The user does not need to configure physical (or network) interfaces for wancom0.

The MACTAB File

Interface mapping on the Oracle CSM is statically defined by a configuration file named MACTAB.

Sample default Oracle CSM interface mapping is presented below:

```
Device2# show interface-mapping
Interface Mapping Info-----
Eth-IF  MAC-Addr                Label
wancom0 00:50:88:BC:11:12         #generic
wancom1 00:50:88:BC:61:6C         #generic
wancom2 00:50:88:BC:11:C7         #generic
spare   00:50:88:BC:61:12         #generic
s0p0    00:50:88:BC:71:79         #generic
slp0    00:50:88:BC:21:FF         #generic
s0p1    00:50:88:BC:41:A2         #generic
slp1    00:50:88:BC:31:AC         #generic
s0p2    FF:FF:FF:FF:FF:FF         #dummy
slp2    FF:FF:FF:FF:FF:FF         #dummy
s0p3    FF:FF:FF:FF:FF:FF         #dummy
slp3    FF:FF:FF:FF:FF:FF         #dummy
```

The user creates physical-interface configurations using the names shown under the Eth_IF column, within service configuration elements. There are two types of physical interfaces that apply to the Oracle CSM, segregated by the naming conventions:

- Media interfaces, shown above as s0p0 and slp0.
- Management interfaces, shown above as wancom0 and wancom1

It is recommended that the user configure physical interfaces using the same naming conventions used in Oracle Session Border Controller that operates on proprietary platforms. These conventions, which simply use 's' for slot and 'p' for port, are visible in the MACTAB file.

Working with the MACTAB File

Interface identification on the Oracle CSM is based on a system-level file called MACTAB that maps interface MAC addresses to interface naming that can be applied within Oracle CSM configuration. In most cases, users retain the default mapping. The **show interface-mapping** command provide access to commands that allow the user see, change and even locate each interface based on this mapping. The MACTAB file is stored in the /boot folder. The MACTAB file ensures that interface mapping is persistent, and therefore usable, by your configuration regardless of changes to the system.

The **show interface-mapping** command displays the current mapping. An example of a MACTAB file that a user has configured is provided below.

```
CMS1# show interface-mapping
Interface Mapping Info
=====
Eth-IF      MAC-Addr                Label
wancom0     00:16:3E:30:00:2A       # ctrl port, onboard MAC
wancom1     00:16:3E:30:00:2B       # 2nd ctrl port, onboard MAC
s0p0        00:16:3E:30:00:2C       # First media interface
slp0        00:16:3E:30:00:2D       # Second media interface
=====
```

interface-mapping


The **interface-mapping** command set includes a **locate** command that causes the interface specified to blink, further helping the user to ensure they are connecting the correct port during cabling procedures.

The following table lists the **interface-mapping** subcommands along with their descriptions.

Command	Description
interface-mapping show	Display the existing content of /boot/mactab file, with the mapping information of all the available Ethernet Interface Names versus Physical Interface MAC addresses, along with any customer provided label information.
interface-mapping locate <ethernet if name> <seconds>	Lets you visually locate the Ethernet media interface. One way to achieve this is to flash the LED of the physical interface when its device name is located. This parameter indicates, in seconds, when the flashing occurs on the LED.
interface-mapping label <ethernet if name> labeling text	Lets you label the Ethernet interface identified by <eth-if-name> with a text string you define. For example, you can use a label that is meaningful for your network layout. This label is stored and then displayed as # string after the MAC address for the Ethernet interface in the /boot/mactab file.
interface-mapping delete <ethernet if name>	Delete an unused Ethernet interface. The unused Ethernet interface could be result of changing network configuration. For example, if you replace an old NIC with a new one, the system writes the new one into mactab file, but does not delete the old one. A confirmation step appears with warning message. When you confirm the action, this entry is deleted from /boot/mactab file.
interface-mapping swap <ethernet if name1> <ethernet if name2>	Swap the mapping of Ethernet interface names against the available MAC physical interfaces. For example, you can first execute the interface-mapping show command to display the current information. interface-mapping show wancom0 00:16:3E:30:00:2A # control port, onboard MAC wancom1 00:16:3E:30:00:2B # 2nd control port, onboard MAC s0p0 00:16:3E:30:00:2C # PCI left side s1p0 00:16:3E:30:00:2D # PCI right side Then you can execute the interface-mapping swap command. interface-mapping swap s0p0 s1p0 wancom0 00:16:3E:30:00:2A # control port, onboard MAC wancom1 00:16:3E:30:00:2B # 2nd control port, onboard MAC

System Configuration

Command	Description
	s0p0 00:16:3E:30:00:2D # PCI right side s1p0 00:16:3E:30:00:2C # PCI left side A warning message appears. Once you confirm the action, the MAC addresses and their corresponding labels are swapped in the /boot/mactab/file.

 **Note:** The **delete** and **swap interface-mapping** commands require a reboot to activate the new mactab.

Serial Interfaces

In lieu of IP management access, serial access provides the user with direct access to the Oracle CSM ACLI. The user must identify how their system allows for serial access. The serial interface can be a critical component of VM and COTS physical interface configuration as the user can make MACTAB changes via the serial interface without interrupting their own access during that management procedure.

Access to the Oracle CSM serial interface is dependent on platform. Familiarity with the platform is required to understand physical location and default serial configuration.

Virtual machine management software provides a simulated serial port service from which the user gets direct serial access to each system. See your virtual machine manager's documentation for instructions on accessing these serial ports. COTS systems provide serial interfaces, such as RS232, from which the user can get serial access to bare metal installations. See your COTS documentation for the location and requirements of these interfaces.

Serial port configuration, via boot parameters, is the same across all platforms.

Before You Configure

This section describes steps you should take prior to configuring physical interfaces.

Before you configure a physical interface:

1. Decide on the number and type of physical interfaces you need.

For example, you might have one media interface connecting to a private network and one connecting to the public network. You might also need to configure maintenance interfaces for HA functionality.

2. Determine the slot and port numbering you will need to enter for the physical interfaces you want to configure. The graphic above can serve as your slot and port numbering reference.
3. If you are configuring your Acme Packet 4500 for HA, refer to the HA Nodes documentation and follow the instructions there for setting special parameters in the physical interface configuration.

Physical Interface Configuration

This section describes how to configure physical interfaces.

1. Access the **phy-interface** configuration element.

```
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)#phy-interface
ORACLE(phy-interface)#
```

2. **name**—Set a name for the interface using any combination of characters entered without spaces. For example: s0p0.

3. **admin-state**—Leave the administrative state parameter set to **enabled** to receive and send traffic on this interface. Select **disabled** to prevent media and signaling from being received and sent. The default for this parameter is **enabled**.
4. **operation-type**—Select the type of physical interface connection to use. Refer to the appropriate platform section to identify how this parameter corresponds to an external interface. The default value is **control**. The valid values are:
 - **media**—Use this value for configuring the media interfaces which carry production traffic.
 - **maintenance**—Use this value for configuring the management physical interfaces, used for management protocols or HA.
 - **control**—This legacy parameter may also be used to configure the management physical interfaces.
5. **slot**—Set the slot number for this physical interface. Refer to the appropriate platform section to identify how this parameter corresponds to an external interface.
6. **port**—Set the port number for this physical interface. Refer to the appropriate platform section to identify how this parameter corresponds to an external interface.
7. **auto-negotiation**—Leave this parameter set to enabled so that the Oracle CSM and the device to which it is linked can automatically negotiate the duplex mode and speed for the link.

If auto-negotiation is enabled, the Oracle CSM begins to negotiate the link to the connected device at the duplex mode you configure. If auto-negotiation is disabled, then the Oracle CSM will not engage in a negotiation of the link and will operate only at the duplex mode and speed you set. The default is enabled. The valid values are:

- enabled | disabled
8. **duplex-mode**—Set the duplex mode. The default is full; this field is only used if the auto-negotiation field is set to disabled.

Given an operating speed of 100 Mbps, full duplex mode lets both devices on a link send and receive packets simultaneously using a total bandwidth of 200 Mbps. Given the same operating speed, half duplex mode limits the devices to one channel with a total bandwidth of 100 Mbps. The valid values are:

 - half | full
 9. **speed**—Set the speed in Mbps of the physical interfaces; this field is only used if the auto-negotiation field is set to disabled. 100 is the default. The valid values are:
 - 10 | 100 | 1000
 10. **virtual-mac**—Refer to Oracle CSM High Availability (HA) documentation to learn how to set this parameter on an HA interface.
 11. Type **done** to save your configuration.

Network Interfaces

The network interface element specifies a logical network interface. In order to use a network port on a network interface, you must configure both the physical interface and the corresponding network interface configuration elements. If the network interface does not use VLANs tagging, ensure that the **sub-port-id** parameter is set to 0, the default value. When VLAN tags are used on a network interface, the valid **sub-port-id** value can range from 1-4096. The combination of the **name** parameter and the **sub-port-id** parameter must be unique in order to identify a discrete network interface.

IP Configuration

A Oracle CSM network interface has standard parameters common to nearly all IP network interfaces. There are a few fields that are unique to the Oracle CSM.

VLANs

VLANs are used to logically separate a single physical interface into multiple network interfaces. There are several applications for this like MPLS VPNs (RFC 2547), MPLS LSPs, L2VPNs (IPSec, L2TP, ATM PVCs), reusing

address space, segmenting traffic, and maximizing the bandwidth into a switch or router. The range of services and management capabilities you can implement with VPNs is huge.

The primary applications of VLANs on the Oracle CSM are VPNs and peering. Several peering partners may terminate their connections to a Oracle CSM on a single physical interface. VLAN tags are used to segregate and correctly route the terminated traffic. The Oracle CSM can support a maximum of 1024 VLANs per physical interface. Ingress packets that do not contain the correct VLAN tag will be dropped. All packets exiting on an egress interface will have the VLAN tag appended to them.

The Oracle CSM can be included in an MPLS network through its connectivity to a PE router, which maps a MPLS VPN label to an 802.1q VLAN tag. Each Oracle CSM can terminate different 802.1q VLANs into separate network interfaces, each of which can represent a different customer VPN.

Overlapping Networks

Overlapping networks are when two or more private networks with the same addressing schemes terminate on one physical interface. The problem this creates can easily be solved by using VLAN tagging. For example, two 10.x.x.x networks terminating on one Oracle CSM network interface will obviously not work. The Oracle CSM includes the IP Address, Subnet Mask, and 802.1q VLAN tag in its Network Interface determination. This allows Oracle CSM to directly interface to multiple VPNs with overlapping address space.

HIP

By default, the Oracle CSM's FTP, ICMP, SNMP, and Telnet services cannot be accessed via the media interfaces. In order to enable these services, the Oracle CSM includes four fields that enable administrative traffic over the media interfaces. These are collectively known as the HIP, or host-in-path functions. The HIP parameters are effectively firewall functions that open the well-known ports for specified services on media interfaces.

Configurable MTU Size

Configurable MTU on per network-interface basis enables the user to set a different MTU on each network interface. It also enables the user to set a system wide default MTU for IPv6 and IPv4 network interfaces. System wide defaults can be set in **system-config** configuration object by setting **ipv6-signaling-mtu** or **ipv4-signaling-mtu**. Defaults are 1500 for both IPv6 and IPv4.

These settings can be overwritten for each network interface by setting **signaling-mtu** in **network-interface** configuration object. Default is 0 – meaning use the system wide MTU.

This feature applies to all Signaling packets generated by the Oracle CSM. All UDP packets greater than the MTU will be fragmented. For all TCP connections we advertise MSS (Maximum Segment Size) TCP option in accordance with the configured MTU. MSS option is sent in SYN and SYN/ACK packets to let the other side of the TCP connection know what your maximum segment size is. This ensures that no TCP packet is greater than the configured MTU.

1. MTU settings do not apply to media packets.
2. UDP: MTU settings apply only to packets sent by the Oracle CSM. The Oracle CSM will continue to process received packets even if they exceed to the configured MTU.
3. Security Phy (IPsec) hardware only; We subtract 100 bytes from the configured MTU to allow for extra headers added by security protocols. This happens even when Security Phy (IPsec) is in clear mode (no security is being applied). Due to hardware limitations of the Security Phy (IPsec) it only allows one MTU per physical port. The maximum MTU of all network interfaces on a given physical port will be used as the MTU for that physical port.
4. The Call Recording feature is where we make a copy of a packet, encapsulate it in an IP-in-IP header and send it to a configured Call Recording Server (CRS). When Call Recording is enabled, to allow space for IP-in-IP encapsulation we reduce the MTU of the original packets to be to be the lesser of the two options listed below.
 - Original Destination network MTU minus size of IP-in-IP header.
 - CRS network interface's MTU minus size of IP-in-IP header.



Note: This will ensure that the traffic sent to the CRS will be within the MTU constraints of CRS' network-interface.

Network Interface Configuration

This section explains how to access and configure network interface.

Special Considerations

Configuration changes to network interface parameters might have an impact on boot configuration parameters. After configuring the network interface, you might receive a message indicating that you could be changing boot config parameters under the following circumstances:

- A physical interface or network interface element matches the boot interface (for example, the physical port is the same as the boot port).
- The boot configuration parameters are modified, because the IPv4 address, netmask, or gateway is different from the corresponding boot configuration parameters.

You are asked if you want to continue. If you enter yes, the configuration will be saved and then the differing boot configuration parameters will be changed. If you enter no, then the configuration is not saved and the boot configuration parameters are not changed.

Configuring the physical and network interface elements for the first management interface is optional because that interface, eth0, is implicitly created by a valid bootparam configuration that specifies the boot device, IPv4 address, subnet, and gateway.

Network Interfaces Configuration

This section describes how to configure a network interface.

Access the **network-interface** configuration element.

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

IP Configuration and Identification

You must specify the identity and address for all network interfaces.

Set the following parameters to configure a network interface:

1. **name**—Set the name for the network interface. This must be the same name as the physical interface to which it corresponds.
2. **description**—Enter a description of the network for easier identification.
3. **hostname**—Set the hostname (FQDN) of this network interface. This parameter is optional.
4. **ip-address**—Set the IP address of this network interface.
5. **netmask**—Set the netmask of this network interface in dotted decimal notation.
6. **gateway**—Set the gateway that this network interface uses to communicate with the next hop.
7. **sec-gateway**—Set an additional optional gateway for this network interface
8. **dns-ip-primary**—Set the DNS servers. You can set an additional two DNS servers by using the **dns-ip-backup1** and **dns-ip-backup2** parameters.
9. **dns-domain**—Set the default domain name.
10. **signaling-mtu**—Sets the MTU size for IPv4 or IPv6 transmission.

VLAN Configuration

One parameter is required to configure VLANs on a Oracle CSM. The **sub-port-id** parameter located in the **network-interfaces** element adds and masks for a specific VLAN tag.

sub-port-id—Enter the identification of a specific virtual interface in a physical interface (e.g., a VLAN tag). If this network interface is not channelized, leave this field blank, and the value will correctly default to 0. The **sub-port-id** is only required if the operation type is Media. The valid range is:

- Minimum—0
- Maximum—4095.

HIP Addresses

To configure administrative service functionality on a media interface, you must define the IP addresses on the media interfaces of your Oracle CSM where you will receive administrative traffic. Adding HIP entries automatically opens the well-known port associated with a service.

Set the following parameters to configure HIP functionality on a network interface:

1. **add-hip-ip**—Set all possible IP address(es) on which you want the Oracle CSM to accept administrative traffic. Entries in this element are IP addresses of media network interfaces. This parameter can accept multiple IP addresses. You can later remove this entry by typing **remove-hip-ip** followed by the appropriate IP address.
2. **add-ftp-ip**—Set the IP address where ports 20 and 21 are opened. This allows standard FTP packets enter the Oracle CSM and reach the host. You can later remove this entry by typing **remove-ftp-ip** followed by the appropriate IP address.
3. **add-icmp-ip**—Set the IP addresses to pass standard ping packets to the host; this parameter can accommodate multiple ping IP addresses. You can later remove this entry by typing **remove-icmp-ip** followed by the appropriate IP address.

When you configure multiple ICMP ping addresses in for a network interface, you must also configure the host-in-path addresses in the `hip-ip-list` for each ICMP address. For security, if the ICMP address and the `hip-ip-list` are not added for an address, the Oracle CSM hardware discards ICMP requests or responses for the address.

To remove multiple IP addresses at one time, type the **remove-icmp-ip** and a Space, open quotation mark (“), the IP addresses you want removed from the list each separated by a space, close quotation mark (”), and then press Enter.

```
ORACLE (network-interface)# remove-icmp-ip "142.214.5.34 124.8.67.3"
```

4. **add-snmp-ip**—Set the IP address where port 161 is opened. This lets SNMP traffic enter the Oracle CSM and reach the host. You can later remove this entry by typing **remove-snmp-ip** followed by the appropriate IP address.
5. **add-telnet-ip**—Set the IP address where port 23 is opened for telnet access. You can later remove this entry by typing **remove-telnet-ip** followed by the appropriate IP address.

Configurable MTU Size

Configurable MTU on per network-interface basis enables the user to set a different MTU on each network interface. It also enables the user to set a system wide default MTU for IPv6 and IPv4 network interfaces. System wide defaults can be set in `system-config` configuration object by setting `ipv6-signaling-mtu` or `ipv4-signaling-mtu`. Defaults are 1500 for both IPv6 and IPv4.

These settings can be overwritten for each network interface by setting `signaling-mtu` in `network-interface` configuration object. Default is 0 – meaning use the system wide MTU.

This feature applies to all Signaling packets generated by the Oracle CSM. All UDP packets greater than the MTU will be fragmented. For all TCP connections we advertise MSS (Maximum Segment Size) TCP option in accordance with the configured MTU. MSS option is sent in SYN and SYN/ACK packets to let the other side of the TCP connection know what your maximum segment size is. This ensures that no TCP packet is greater than the configured MTU.

1. MTU settings do not apply to media packets.
2. UDP: MTU settings apply only to packets sent by the Oracle CSM. The Oracle CSM will continue to process received packets even if they exceed to the configured MTU.
3. Security Phy (IPsec) hardware only; We subtract 100 bytes from the configured MTU to allow for extra headers added by security protocols. This happens even when Security Phy (IPsec) is in clear mode (no security is being applied). Due to hardware limitations of the Security Phy (IPsec) it only allows one MTU per physical port. The maximum MTU of all network interfaces on a given physical port will be used as the MTU for that physical port.

- The Call Recording feature is where we make a copy of a packet, encapsulate it in an IP-in-IP header and send it to a configured Call Recording Server (CRS). When Call Recording is enabled, to allow space for IP-in-IP encapsulation we reduce the MTU of the original packets to be to be the lesser of the two options listed below.
 - Original Destination network MTU minus size of IP-in-IP header.
 - CRS network interface's MTU minus size of IP-in-IP header.



Note: This will ensure that the traffic sent to the CRS will be within the MTU constraints of CRS' network-interface.

System Wide MTU Size

To change system wide MTU settings:

- Access the **system-config** configuration element.

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

- Type **select** to begin editing the **system-config** object.

```
ORACLE(system-config)# select
ACMEPACKET(system-config)#
```

- ipv6-signaling-mtu** or **ipv4-signaling-mtu** Configure MTU in the system config and optionally in the network interface. Default will be 1500 bytes.

```
ORACLE(system-config)# ipv6-signaling-mtu 1500
ORACLE(system-config)# ipv4-signaling-mtu 1600
```

- Type **done** to save your configuration.

Required SIP Configuration

The Oracle CSM requires that SIP processing be operational to perform basic service. Required configuration, discussed within this section, includes:

- Enable SIP Config
- Configure SIP Interfaces
- Configure SIP Ports
- Configure SIP Digest Authentication

The Oracle CSM provides a myriad of other SIP controls, which you may or may not need for your deployment. See the Chapter on SIP Signaling Services for explanations of and configuration instructions to these SIP services.

Enabling SIP-Config

The Oracle CSM cannot process SIP messaging if the sip-config is not enabled. To enable sip-config:

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

```
ORACLE# configure terminal
```

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

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

- Type **sip-config** and press Enter to access the sip-config configuration elements.

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

- Type **select** and press Enter to select the sip-config configuration element.

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

Selecting the sip-config is required, as is true of all single instance element.

- Type **enable** and press Enter to enable the sip-config element.

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

6. Type done to make this setting, then save and activate your configuration.

SIP Interfaces

This section explains how to configure a SIP interface. The SIP interface defines the transport addresses (IP address and port) upon which the Oracle CSM receives and sends SIP messages.

Overview

The SIP interface defines the signaling interface. You can define a SIP interface for each network or realm to which the Oracle CSM is connected. SIP interfaces support both UDP and TCP transport, as well as multiple SIP ports (transport addresses). The SIP interface also lets Hosted NAT Traversal (HNT) be used in any realm.

The SIP interface configuration process involves configuring the following features:

- address and transport protocols (SIP ports)
- redirect action
- proxy mode
- trust mode

About SIP Ports

A SIP port defines the transport address and protocol the Oracle CSM will use for a SIP interface for the realm. A SIP interface will have one or more SIP ports to define the IP address and port upon which the Oracle CSM will send and receive messages. For TCP, it defines the address and port upon which the Oracle CSM will listen for inbound TCP connections for a specific realm.

You need to define at least one SIP port, on which the SIP proxy will listen for connections. If using both UDP and TCP, you must configure more than one port. For example, if a call is sent to the Oracle CSM using TCP, which it needs to send out as UDP, two SIP ports are needed.

Preferred SIP Port

When a SIP interface contains multiple SIP ports of the same transport protocol, a preferred SIP port for each transport protocol is selected for outgoing requests when the specific SIP port cannot be determined. When forwarding a request that matched a cached registration entry (HNT or normal registration caching), the SIP port upon which the original REGISTER message arrived is used. Otherwise, the preferred SIP port for the selected transport protocol is used. When selecting the preferred SIP port, the default SIP port of 5060 will be selected over other non-default ports.

For SIP interfaces using the SIP NAT function, the preferred SIP port address and port will take precedence over the external address of the SIP NAT when they do not match. If both TCP and UDP SIP ports are defined, the address and port of the preferred UDP port is used.

Proxy Mode

The Oracle CSM's proxy mode determines whether it forwards requests received on the SIP interface to target(s) selected from local policy; or sends a redirect response to the previous hop. Sending the redirect response causes the previous hop to contact the targets directly.

If the source of the request matches a session agent with a proxy mode already defined, that mode overrides the proxy mode defined in the SIP interface.

You can configure the proxy mode to use the Record-Route option. Requests for stateless and transaction operation modes are forwarded with a Record-Route header that has the Oracle CSM's addresses added. As a result, all subsequent requests are routed through the Oracle CSM.

Redirect Action

The redirect action is the action the SIP proxy takes when it receives a SIP Redirect (3xx) response on the SIP interface. If the target of the request is a session agent with redirect action defined, its redirect action overrides the SIP interface's.

You can set the Oracle CSM to perform a global redirect action in response to Redirect messages. Or you can retain the default behavior where the Oracle CSM sends SIP Redirect responses back to the previous hop (proxy back to the UAC) when the UAS is not a session agent.

The default behavior of the Oracle CSM is to recurse-305-only. If the Oracle CSM receives a 305 SIP redirect response (Use Proxy) on the SIP interface, it automatically redirects all requests to the Contact URI specified in the 305 response. All other 3xx responses are sent back to the previous hop.

When the redirect-action parameter is set to recurse, it does so on SIP Redirect responses received from the user agent server (UAS) and sends a new request to the Contact headers contained in the SIP Redirect response.

Instead of this behavior, the Oracle CSM can proxy the SIP Redirect response back to the user agent client (UAC) using the value in the session agent's redirect action field (when the UAS is a session agent). If there are too many UASes to define as individual session agents or if the UASs are HNT endpoints, and SIP Redirect responses need to be proxied for UASs that are not session agents; you can set the default behavior at the SIP Interface level.

SIP maddr Resolution

Release S-C6.2.0 provides enhanced resolution of addresses found in SIP contact headers, or in the maddr (multicast address) parameter of SIP 3xx REDIRECT messages. Previous releases resolved these addresses as either a host address or as a session agent name. With Release 6.2.0 these addresses can also be resolved as session agent group (SAG) names.

Support for SAG-based resolution is provide by a new sip-config parameter, sag-lookup-on-redirect. By default, SAG lookup is disabled, providing compatibility with prior releases.

The following sample SIP REDIRECT and ACLI configuration fragment illustrate enhanced processing.

```
Status-Line: SIP/2.0 302 Moved
Message Header
Via: SIP/2.0/UDP
192.168.200.224:5060;branch=z9hG4bKa0fs40009o90sc8oo780.1
From: <sip:1111@192.168.1.222:6000>;tag=1
To: sut <sip:2223@192.168.1.224:5060>;tag=11
Call-ID: 1-28515@192.168.1.222
CSeq: 1 INVITE
Contact: <sip:1111@192.168.1.223;maddr=test.acmepacket.com>
Privacy: user;id;critical;session
P-Preferred-Identity: sipp <sip:sipp@192.168.200.222:5060>
P-Asserted-Identity: abc.com
Subject: abc
Proxy-Require: privacy,prack,abc
Content-Length: 0

session-group
    group-name                test.acmepacket.com
    description
    state                      enabled
    app-protocol              SIP
    strategy                  Hunt
    dest
                                192.168.200.222
                                192.168.200.223
...
...
```

In this case, when the Oracle CSM receives the 302, it resolves the information from maddr to a SAG name. In the above example, it will resolve to the configured SAG – test.acmepacket.com. The destinations configured in SAG test.acmepacket.com will be used to route the call.

SAG-based address resolution is based on the following set of processing rules.

1. When the Contact URI does not have an maddr parameter, and the hostname is not an IP Address, the Oracle CSM will look for a SAG matching the hostname.

System Configuration

2. When the Contact URI has an maddr parameter that contains an IP address, the Oracle CSM will not look for a SAG; it will use the IP Address as the target/next-hop.
3. When the Contact URI has an maddr parameter that contains a non-IP-address value, the Oracle CSM will look for a SAG matching the maddr parameter value.

The above logic can be turned on by enabling `sag-lookup-on-redirect` in the `sip-config` object as shown below.

SIP maddr Resolution Configuration

To configure the Oracle CSM to perform SAG-based maddr resolution:

1. From superuser mode, use the following command sequence to access `sip-config` configuration mode. While in this mode, you configure SAG-based address resolution.

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

2. Use the `sag-lookup-on-redirect` parameter to enable SAG-based maddr resolution.
3. Use `done`, `exit`, and `verify-config` to complete SAG-based address resolution.

Trust Mode

The Oracle CSM supports the Calling Identity privacy requirements based on RFC 3323 and RFC 3325. The trust mode in the SIP interface determines whether the source and destination of a request is a trusted entity. With the implementation of this feature, the Oracle CSM can understand and support the privacy headers and provide the capability for anonymous packets

The Oracle CSM, which acts as a boundary device between the trusted platform and the untrusted Internet, understands the following headers:

- Privacy Header
- P-Asserted-Identity Header
- P-Preferred-Identity Header

Depending on the value of these headers and the mode in which the Oracle CSM is being operated (B2BUA or the proxy), the appropriate actions are performed.

About the Process

On receiving a message, the Oracle CSM checks whether the message source is trusted or not. It checks the SIP interface's trust mode value and, if the source is a session agent, the session agent's trust me value. Depending on these values, the Oracle CSM decides whether the request's or response's source is trusted. If it receives message from a trusted source and the message contains the P-Asserted-Identity header field, the Oracle CSM passes this message to the outgoing side. The outgoing side then decides what needs to be done with this request or response.

If the request or the response is received from an untrusted source, the Privacy header value is `id` (privacy is requested), and the P-Asserted-Identity header field is included, the Oracle CSM strips the Privacy and the P-Asserted-Identity headers and passes the request or the response to the outgoing side.

If the request or the response contains the P-Preferred-Identity header and the message source is untrusted, the Oracle CSM strips the P-Preferred-Identity header from the request or the response and passes the message to the outgoing side.

If the source is trusted or privacy is not requested (the value of the Privacy Header is not `id`) and the request or the response contains the P-Preferred-Identity header, the Oracle CSM performs the following actions:

- inserts the P-Asserted-Identity header field with the value taken from the P-Preferred-Identity header field
- deletes the P-Preferred-Identity header value
- passes this request or the response to the Outgoing side for the appropriate action, depending on the whether the destination is trusted or not

After the Oracle CSM passes the request or the response to the outgoing side, it checks whether the destination is trusted by checking the SIP interface's trust mode value and the session agent's trust me value (if the destination is configured as session agent).

- The destination is trusted

The Oracle CSM does nothing with the request or the response and passes it to the destination. If the P_Asserted_Identity headers are present, they are passed to the session agent (if the destination is configured as session agent).

- The destination is untrusted

The Oracle CSM looks at the value of the Privacy header. If set to id, the Oracle CSM removes all the P-Asserted-Identity headers (if present). It strips the Proxy-Require header if it is set to privacy. The Oracle CSM also sets the From field of SIP header to Anonymous and strips the Privacy header.

If the Privacy header is set to none, the Oracle CSM does not remove the P-Asserted-Identity header fields.

If there is no Privacy header field, the SD will not remove the P-Asserted-Identity headers.

To implement this feature, you need to configure the session agent's trust me parameter to enabled (if the message source is a session agent) and the SIP interface's trust mode to the appropriate value.

Configurable Timers and Counters

SIP timers and counters can be set in the global SIP configuration, and two can be specific for individual SIP interfaces.

You can set the expiration times for SIP messages, and you can set a counter that restricts the number of contacts that the Oracle CSM tries when it receives a REDIRECT. These are similar to two parameters in the global SIP configuration, trans-expire and invite-expire. You can also set a parameter that defines how many contacts/routes the Oracle CSM will attempt on redirect.

SIP Interface Configuration

To configure a 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 system-level configuration elements.

```
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)#
```

From this point, you can configure SIP interface parameters. To view all sip-interface parameters, enter a ? at the system prompt.

4. state—Enable or disable the SIP interface. The default is enabled. The valid values are:

- enabled | disabled

5. realm-id—Enter the name of the realm to which the SIP interface is connected.

6. sip-ports—Access the sip-ports subelement.

7. carriers—Enter the list of carriers related to the SIP interface.

Entries in this field can be from 1 to 24 characters in length and can consist of any alphabetical character (Aa-Zz), numerical character (0-9), or punctuation mark (!"#\$%^&*()+-=<>?'|{}[]@/\`~.,_ : ;) or any combination of alphabetical characters, numerical characters, or punctuation marks. For example, both 1-0288 and acme_carrier are valid carrier field formats

8. proxy-mode—Enter an option for the proxy mode parameter. Valid values are:

System Configuration

- proxy—Forward all SIP requests to selected targets.
- redirect—Send a SIP 3xx redirect response with the selected target(s) in the Contact header.
- record-route—Forward requests to selected target(s) and insert a Record-Route header with the Oracle CSM's address. For stateless and transaction mode only.

9. redirect-action—Enter the value for the redirect action. Valid values are:

- recurse-305-only—(default) If the Oracle CSM receives a 305 SIP redirect response (Use Proxy) on the SIP interface, it automatically redirects all requests to the Contact URI specified in the 305 response. All other 3xx responses are sent back to the previous hop.
- proxy—Send the SIP request back to the previous hop.
- recurse—Recurse on the Contacts in the response.

The designated proxy action will apply to SIP 3xx responses received from non-session agents and to 3xx responses received from session agents without configured SIP Redirect message actions (for example, session agents without values for the redirect action field).

10. contact-mode—Set the Contact header routing mode, which determines how the contact address from a private network is formatted.

For example, whether a maddr parameter equal to the Oracle CSM's SIP proxy needs to be added to a URI present in a Contact header.

The default is none. The valid values are:

- none—The address portion of the header becomes the public address of that private realm.
- maddr—The address portion of the header will be set to the IP address of the Oracle CSM's B2BUA.
- strict—The contents of the Request-URI is destroyed when a Record-Route header is present.
- loose—The Record-Route header is included in a Request, which means the destination of the request is separated from the set of proxies that need to be visited along the way.

11. nat-traversal—Define the type of HNT enabled for SIP. The default is none. Valid values are:

- none—HNT function is disabled for SIP.
- rport—SIP HNT function only applies to endpoints that include the rport parameter in the Via header. HNT applies when the sent-by of the topmost VIA matches the Contact-URI host address, both of which must be different from the received Layer 3 address.
- always—SIP HNT applies to requests when the sent-by of the topmost VIA matches the Contact-URI host address, both of which must be different from the received Layer 3 address. (Even when the rport parameter is not present.)

12. nat-interval—Set the expiration time in seconds for the Oracle CSM's cached registration entry for an HNT endpoint. The default is 30. The valid range is:

- Minimum—0
- Maximum—999999999

Oracle recommends setting the NAT interval to one-third of the NAT binding lifetime. A NAT binding lifetime is the network connection inactivity timeout. The value is configured (or hardwired) in the NAT device (firewall). This timer is used to cause the UA to send REGISTER messages frequently enough to retain the port binding in the NAT. Retaining the binding lets inbound requests to be sent through the NAT.

13. tcp-nat-interval—Set the registration cache expiration time in seconds to use for endpoints behind a NAT device that register using TCP. On upgrade, the Oracle CSM assigns this parameter the same value as the existing NAT interval. The default is 90. The valid range is:

- Minimum—0
- Maximum—999999999

The Oracle CSM uses the value you set for the TCP NAT interval as the expiration value passed back in SIP REGISTER (200 OK) responses to endpoints behind a NAT that register over TCP. The NAT interval value with which you are familiar from previous releases is used for endpoints behind a NAT that register over UDP. Requiring endpoints that register over TCP to send refresh requests as frequently as those registering over

UDP puts unnecessary load on the Oracle CSM. By adding a separate configuration for the TCP NAT interval, the load is reduced.

For upgrade and backward compatibility with Oracle CSM releases prior to Release 4.1, when the `tcpNatInterval` is not present in the XML for a SIP interface configuration, the value of the NAT interval (`natInterval`) is used for the TCP NAT interval as well.

- 14. registration-caching**—Enable for use with all UAs, not just those that are behind NATs. The default is disabled. The valid values are:

- enabled | disabled

If enabled, the Oracle CSM caches the Contact header in the UA's REGISTER request when it is addressed to one of the following:

- Oracle CSM
- registrar domain value
- registrar host value

The Oracle CSM then generates a Contact header with the Oracle CSM's address as the host part of the URI and sends the REGISTER to the destination defined by the registrar host value.

Whether or not SIP HNT functionality is enabled affects the value of the user part of the URI sent in the Contact header:

- HNT enabled: the Oracle CSM takes the user part of the URI in the From header of the request and appends a cookie to make the user unique. A cookie is information that the server stores on the client side of a client-server communication so that the information can be used in the future.
- HNT disabled: the user part of the Contact header is taken from the URI in the From header and no cookie is appended. This is the default behavior of the Oracle CSM.

When the registrar receives a request that matches the address-of-record (the To header in the REGISTER message), it sends the matching request to the Oracle CSM, which is the Contact address. Then, the Oracle CSM forwards the request to the Contact-URI it cached from the original REGISTER message.

- 15. min-reg-expire**—Set the time in seconds for the SIP interface. The value you enter here sets the minimum registration expiration time in seconds for HNT registration caching. The default is 300. The valid range is:

- Minimum—0
- Maximum—999999999

This value defines the minimum expiration value the Oracle CSM places in each REGISTER message it sends to the real registrar. In HNT, the Oracle CSM caches the registration after receiving a response from the real registrar and sets the expiration time to the NAT interval value.

Some UAs might change the registration expiration value they use in subsequent requests to the value specified in this field. This change causes the Oracle CSM to send frequent registrations on to the real registrar.

- 16. registration-interval**—Set the Oracle CSM's cached registration entry interval for a non-HNT endpoint. Enter the expiration time in seconds that you want the Oracle CSM to use in the REGISTER response message sent back to the UA. The UA then refreshes its registration by sending another REGISTER message before that time expires. The default is 3600. The valid range is:

- Minimum—0

A registration interval of zero causes the Oracle CSM to pass back the expiration time set by and returned in the registration response from the registrar.

- Maximum—999999999

If the expiration time you set is less than the expiration time set by and returned from the real registrar, the Oracle CSM responds to the refresh request directly rather than forwarding it to the registrar.

Although the registration interval applies to non-HNT registration cache entries, and the loosely related NAT interval applies to HNT registration cache entries, you can use the two in combination. Using a combination of the two means you can implement HNT and non-HNT architectures on the same Oracle CSM. You can then

System Configuration

define a longer interval time in the registration interval field to reduce the network traffic and load caused by excess REGISTER messages because there is no NAT binding to maintain.

17. route-to-registrar—Enable routing to the registrar to send all requests that match a cached registration to the destination defined for the registrar host; used when the Request-URI matches the registrar host value or the registrar domain value, not the Oracle CSM's address. Because the registrar host is the real registrar, it should send the requests back to the Oracle CSM with the Oracle CSM's address in the Request-URI. The default is disabled. The valid values are:

- enabled | disabled

For example, you should enable routing to the registrar if your network uses a N Oracle CSM and needs requests to go through its service proxy, which is defined in the registrar host field.

18. teluri-scheme—Enable to convert SIP URIs to tel (resources identified by telephone numbers) URIs.

If enabled, the requests generated on this SIP interface by the Oracle CSM will have a tel URI scheme instead of the SIP URI scheme. Only the Request, From, and To URIs are changed to the tel scheme. After the dialog is established, the URIs are not changed. The default is disabled. The valid values are:

- enabled | disabled

19. uri-fqdn-domain—Change the host part of the URIs to the FQDN value set here. If set to enabled, and used with an FQDN domain/host, the requests generated by the Oracle CSM on this SIP interface will have the host part of the URI set to this FQDN value. Only the Request, To, and From URIs are changed. After the dialog is established, the URIs are not changed.

20. trust-mode—Set the trust mode for the SIP interface, which is checked by the Oracle CSM when it receives a message to determine whether the message source is trusted. The default is all. Available options are:

- all—Trust all SIP elements (sources and destinations) in the realm(s), except untrusted session agents. Untrusted session agents are those that have the trust-me parameter set to disabled.
- agents-only—Trust only trusted session agents. Trusted session agents are those that have the trust-me parameter set to enabled.
- realm-prefix—Trust only trusted session agents, and source and destination IP addresses that match the IP interface's realm (or subrealm) address prefix. Only realms with non-zero address prefixes are considered.
- registered—Trust only trusted session agents and registered endpoints. Registered endpoints are those with an entry in the Oracle CSM's registration cache.
- none—Trust nothing.

Session agents must have one or more of the following:

- global realm
- same realm as the SIP interface
- realm that is a subrealm of the SIP interface's realm

21. trans-expire—Set the TTL expiration timer in seconds for SIP transactions. This timer controls the following timers specified in RFC 3261:

- Timer B—SIP INVITE transaction timeout
- Timer F—non-INVITE transaction timeout
- Timer H—Wait time for ACK receipt
- Timer TEE—Used to transmit final responses before receiving an ACK

The default is 0. If you leave this parameter set to the default, then the Oracle CSM uses the timer value from the global SIP configuration. The valid range is:

- Minimum—0
- Maximum—999999999

22. invite-expire—Set the TTL expiration timer in seconds for a SIP client/server transaction after receiving a provisional response.

You set this timer for the client and the sever by configuring it on the SIP interface corresponding to the core or access side.

The default is 0. If you leave this parameter set to the default, then the Oracle CSM uses the timer value from the global SIP configuration. The valid range is:

- Minimum—0
- Maximum—999999999

23. max-redirect-contacts—Set the maximum number of contacts or routes for the Oracle CSM to attempt in when it receives a SIP Redirect (3xx Response). The default is 0. If you leave this parameter set to the default, then the Oracle CSM will exercise no restrictions on the number of contacts or routes. The valid range is:

- Minimum—0
- Maximum—10

24. response-map—Enter the name of the SIP response map configuration that you want to apply to this SIP interfaces for outgoing responses. This parameter is blank by default.

25. local-response-map—Enter the name of the SIP response map configuration that you want to apply to this SIP interfaces for locally-generated SIP responses. This parameter is blank by default.

26. options—Optional.

Configuring SIP Ports

To configure SIP ports:

1. From sip-interface, type sip-ports and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(sip-interface)# sip-ports
ORACLE(sip-port)#
```

2. address—Enter the IP address of the host associated with the sip-port entry on which to listen. For example:

```
192.168.11.101
```

3. port—Enter the port number you want to use for this sip-port. The default is 5060. The valid range is:

- Minimum—1025
- Maximum—65535

4. transport-protocol—Indicate the transport protocol you want to associate with the SIP port. The default is UDP. The valid values are:

- TCP—Provides a reliable stream delivery and virtual connection service to applications through the use of sequenced acknowledgment with the retransmission of packets when necessary.
- UDP—Provides a simple message service for transaction-oriented services. Each UDP header carries both a source port identifier and destination port identifier, allowing high-level protocols to target specific applications and services among hosts.
- TLS—See the Security chapter for more information about configuring TLS.

5. allow-anonymous—Define the allow anonymous criteria for accepting and processing a SIP request from another SIP element.

The anonymous connection mode criteria includes admission control based on whether an endpoint has successfully registered. Requests from an existing SIP dialog are always accepted and processed. The default is all.

The following table lists the available options.

- all—All requests from any SIP element are allowed.
- agents-only—Only requests from configured session agents are allowed. The session agent must fit one of the following criteria:
 - Have a global realm.
 - Have the same realm as the SIP interface
 - Be a sub-realm of the SIP interface's realm.

When an agent that is not configured on the system sends an INVITE to a SIP interface, the Oracle CSM:

System Configuration

- Refuses the connection in the case of TCP.
- Responds with a 403 Forbidden in the case of UDP.
- realm-prefix—The source IP address of the request must fall within the realm's address prefix or a SIP interface sub-realm. A sub-realm is a realm that falls within a realm-group tree. The sub-realm is a child (or grandchild, and so on) of the SIP interface realm.

Only realms with non-zero address prefixes are considered. Requests from session agents (as described in the agents-only option) are also allowed.

- registered—Only requests from user agents that have an entry in the registration cache (regular or HNT) are allowed; with the exception of a REGISTER request. A REGISTER request is allowed from any user agent.

The registration cache entry is only added if the REGISTER is successful. Requests from configured session agents (as described in the agents-only option) are also allowed.

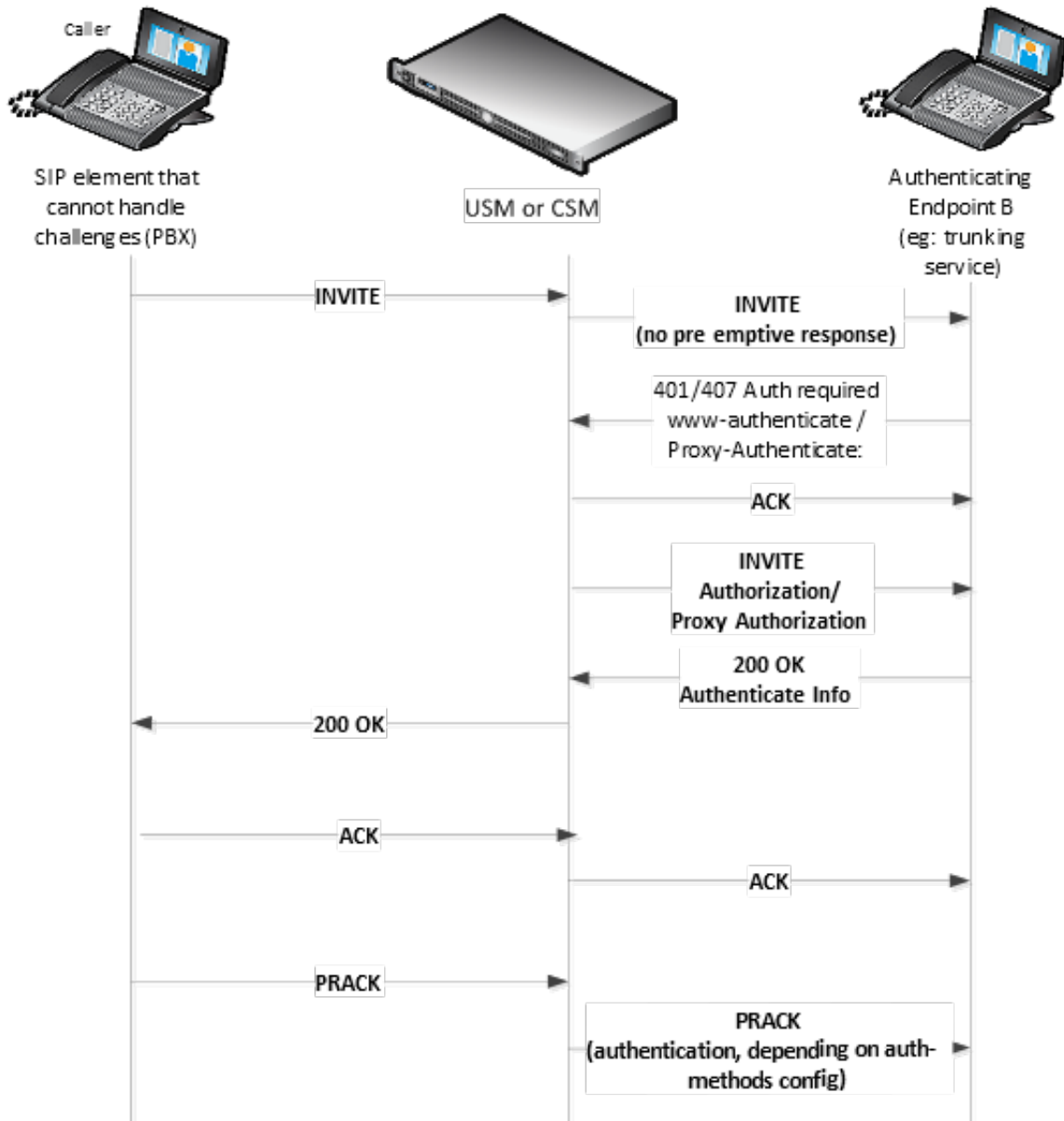
- register-prefix—Only requests from user agents that have an entry in the Registration Cache (regular or HNT) are allowed; with the exception of a REGISTER request. A REGISTER request is allowed only when the source IP address of the request falls within the realm address-prefix or a SIP interface sub-realm. Only realms with non-zero address prefixes are considered.

The Registration Cache entry is only added if the REGISTER is successful. Requests from configured session agents (as described in the agents-only option) are also allowed.

Digest Authentication with SIP

Digest authentication for Session Initiation Protocol (SIP) is a type of security feature on the Oracle CSM that provides a minimum level of security for basic Transport Control Protocol (TCP) and User Datagram Protocol (UDP) connections. Digest authentication verifies that both parties on a connection (host and endpoint client) know a shared secret (a password). This verification can be done without sending the password in the clear.

Digest authentication is disabled by default on the Oracle CSM. When digest authentication is enabled, the Oracle CSM (host) responds to authentication challenges from SIP trunking Service Providers (endpoint client). The Oracle CSM performs authentication for each IP-PBX initiating the call. However, the authentication challenge process takes place between the host and the client only since the IP-PBX cannot handle authentication challenges. The following illustration shows the digest authentication process.




The digest authentication scheme is based on a simple challenge-response paradigm. A valid response contains a checksum (by default, the MD5 checksum) of the “username” and password. In this way, the password is never sent in the clear.

By default, the Oracle CSM uses cached credentials for all requests within the same dialog, once the authentication session is established with a 200OK from the authenticating SIP element. If the in-dialog-methods attribute contains a value, it specifies the requests that have challenge-responses inserted within a dialog.

In digest authentication with SIP, the following can happen:


- More than one authenticating SIP element (IP-PBX) may be the destination of requests.
- More than one authentication challenge can occur in a SIP message. This can occur when there are additional authenticating SIP elements behind the first authenticating SIP element.
- The Oracle CSM distinguishes whether the IP-PBX is capable of handling the challenge. If Digest Authentication is disabled (no auth-attributes configured) on the Session Agent, the challenge is passed back to the IP-PBX.

System Configuration

-  **Note:** If there are multiple challenges in the request, and if the Oracle CSM has only some of the cached credentials configured, the Oracle CSM adds challenge-responses for the requests it can handle, and does not pass the challenge back to the IP-PBX.

Challenge-Responses in Requests not in the Dialog

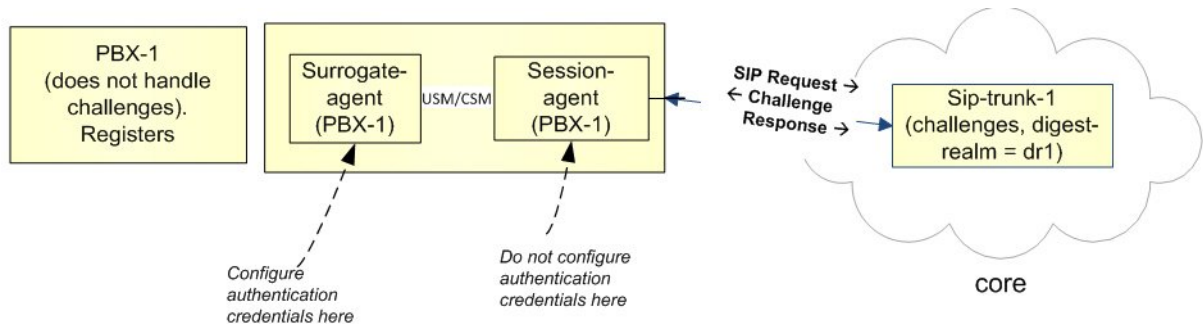
A digest authentication session starts from the client response to a www-authenticate/proxy-authenticate challenge and lasts until the client receives another challenge in the protection space defined by the auth-realm. Credentials are not cached across dialogs; however, if a User Agent (UA) is configured with the auth-realm of its outbound proxy, when one exists, the UA may cache credentials for that auth-realm across dialogs.

-  **Note:** Existing Oracle CSM behavior with surrogate-agents is that they cache credentials from REGISTER for INVITE sessions only if the Oracle CSM is considered a UA sending to its outbound proxy.

Surrogate Agents and the Oracle CSM


In the case where a surrogate-agent is configured for the IP-PBX, you do not have to configure digest authentication attributes in the session-agent object for the same IP-PBX. The surrogate-agent authentication configuration takes precedence over the session-agent authentication configuration and so it is ignored.

The following illustration shows an example of a surrogate-agent with a session-agent in the network.



Configuring Digest Authentication

In the Oracle CSM CLI, you can access the Digest Authentication object at the path session-router->session-agent->auth-attribute. If enabled, the Digest Authentication process uses the attributes and values listed in this table.

-  **Note:** If enabling Digest Authentication, all attributes listed below are required except for the in-dialog-methods attribute which is optional.

The following table lists the digest authentication object

```
ORACLE (auth-attribute) # show
auth-attribute
    auth-realm          realm01
    username            user
    password            *****
    in-dialog-methods  ACK INVITE SUBSCRIBE
```

To configure digest authentication on the Oracle CSM:

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-related objects.

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

3. Type session-agent and press Enter to access the session agent-related attributes.

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


4. Type `auth-attribute` and press Enter to access the digest authentication-related attributes.

```
ORACLE(session-agent)# auth-attribute
ORACLE(auth-attribute)#
```

5. `auth-realm` — Enter the name (realm ID) of the host realm initiating the authentication challenge. This value defines the protected space in which the digest authentication is performed. Valid value is an alpha-numeric character string. Default is blank.

```
ORACLE(auth-attribute)# auth-realm realm01
```

6. `username` — Enter the username of the client. Valid value is an alpha-numeric character string. Default is blank.

```
ORACLE(auth-attribute)# username user
```

7. `password` — Enter the password associated with the username of the client. This is required for all LOGIN attempts. Password displays while typing but is saved in clear-text (i.e., *****). Valid value is an alpha-numeric character string. Default is blank.

```
ORACLE(auth-attribute)# password *****
```

8. `in-dialog-methods` — Enter the in-dialog request method(s) that digest authentication uses from the cached credentials. Specify request methods in a list form separated by a space enclosed in parentheses. Valid values are:

- INVITE | BYE | ACK | CANCEL | OPTIONS | SUBSCRIBE | PRACK | NOTIFY | UPDATE | REFER

```
ORACLE(auth-attribute)# in-dialog-methods (ack invite subscribe)
```



Note: The methods not in this list are still resubmitted if a 401/407 response is received by the Oracle CSM.

If you do not specify any in-dialog-method value(s), digest authentication does not add challenge-responses to in-dialog requests within a dialog.

This attribute setting applies to in-dialog requests only.

Additional Notes

The following are additional notes that describe the digest authentication process:

- The Oracle CSM always challenges the first LOGIN request, and initial authentication begins with that request. The recalculated authorization key — the credentials — are then included in every subsequent request.
- If the Oracle CSM does not receive any communication from the client within the expiration period, the Oracle CSM logs the client out and tears down the transport connection. Faced with interface loss, the Oracle CSM default behavior is to flush all warrant information from the target database. This response necessitates that the client first login/re-register with the Oracle CSM, and then repopulate the empty database using a series of ADD requests. This behavior ensures that client and Oracle CSM target databases are synchronized.

Alternatively, when faced with interface loss, the Oracle CSM can retain all warrant information within the target database. This response necessitates only that the client first login/re-register with the Oracle CSM. After successful registration the client should, but is not required to, use a series of GET, ADD, and DELETE requests to ensure that the Oracle CSM and client target databases are synchronized.

- The Oracle CSM ignores the Authentication-Info header that comes in the 200OK response after digest authentication is complete. The Oracle CSM receives a 401/407 response from the client. However, some surrogate-agents may process the Authentication-Info header in a single challenge.

Digest Authentication and High Availability

The Oracle CSM supports digest authentication in high availability (HA) environments. The session-agent configuration, which includes the digest authentication parameters on the primary Oracle CSM, are replicated on the HA Oracle CSM. However, cached credentials on the primary device are not replicated on the HA device.

IP Identification (ID) Field

By default, non-fragmented UDP packets generated by media interfaces have the ID field set to 0. You can configure the Oracle CSM to populate this field with an incrementing value by adding the `increment-ip-id` option in the media manager. Every non-fragmented packet sent will have its ID increased by one from the previous packet sent.

Using a packet trace application, egress packets from the Oracle CSM will have an ID field that appears to be incrementing. Enabling the ID field can help distinguish a retransmitted non-fragmented application layer packet from a packet retransmitted by the network layer in monitoring or lab situations.

IP Identification Field Configuration

To enable ID field generation in media-manager:

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

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

2. Type `media-manager` and press Enter.

```
ORACLE (configure) # media-manager
ORACLE (media-manager) #
```

3. `options`—Set the `options` parameter by typing `options`, a Space, the option name `increment-ip-id` with a plus sign in front of it, and then press Enter.

```
ORACLE (media-manager) # options + increment-ip-id
```

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.

4. Save and activate your configuration.

SNMP

This section explains how to configure Simple Network Management Protocol (SNMP), trap receivers, and syslog servers. These features are not essential for baseline Oracle CSM service, but they are necessary to use network management systems to manage the Oracle CSM. They provide important monitoring and system health information that contribute to a robust deployment of the system.

Overview

SNMP is used to support monitoring of network-attached devices for conditions that warrant administrative attention. SNMP is comprised of three groups of settings on a Oracle CSM. These settings are system-wide configurations including MIB contact information, SNMP community settings, and trap receivers.

Basic SNMP Parameters

The Oracle CSM includes several parameters that control basic SNMP functionality. The MIB-related elements are for informational purposes, and are helpful if set. The remainder of the parameters determines if certain Oracle CSM events are reported to the SNMP system.

SNMP Community

An SNMP community is a grouping of network devices and management stations used to define where information is sent and accepted. An SNMP device or agent might belong to more than one SNMP community. SNMP communities provide a type of password protection for viewing and setting management information within a community.

SNMP communities also include access level settings. They are used to define the access rights associated with a specific SNMP community. The Oracle CSM lets you define two types of access levels: read-only and read-write.

You can define multiple SNMP communities on a Oracle CSM to segregate access modes per community and NMS host.

SNMP Trap Receiver Uses

A trap receiver is an application used to receive, log, and view SNMP traps for monitoring the Oracle CSM (CSM). An SNMP trap is the notification sent from a network device, such as the CSM, that declares a change in service. Multiple trap receivers can be defined on an CSM for either redundancy or to segregate alarms with different severity levels to individual trap receivers.

Each Oracle Communications Session Delivery Manager which manages Oracle CSMs should be configured on those SBCs as trap receivers.

Configuring SNMP

This section describes how to configure your Oracle CSM to work with external SNMP systems. Sample configurations are also provided.

SNMP Configuration Overview

1. Configure the SNMP identification information. This step includes configuring the MIB system contact, name, and location parameters.
2. Set the general SNMP parameters to enable or disable SNMP on the Oracle CSM. This step includes setting the switches that govern how the SNMP system responds to specified events.
3. Set the syslog events. This step includes setting the parameters for handling SNMP monitoring syslog events, which can trigger SNMP syslog traps.
4. Set SNMP communities. Configuration is separated into a unique configuration element.
5. Set trap receivers. Configuration is separated into a unique configuration element.

SNMP Configuration

To configure SNMP:

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
```

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

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

From this point, you can set SNMP parameters. The following is an example what an SNMP configuration might look like. Parameters not described in this section are omitted below.

```
system-config
  mib-system-contact      John Doe
  mib-system-name         Test System
  mib-system-location     Upstairs
  snmp-enabled            enabled
  enable-snmp-auth-traps  disabled
  enable-snmp-syslog-notify disabled
  enable-snmp-monitor-traps disabled
  enable-env-monitor-traps disabled
  snmp-syslog-his-table-length 1
  snmp-syslog-level       WARNING
```

System Wide Configuration for SNMP

This section describes the system-wide SNMP parameters found in the System Configuration element. These parameters set global SNMP information.

Set the following parameters to configure system wide SNMP functionality:

1. **mib-system-contact**—Set the contact information used within the system’s MIB transactions. The SNMP agent sends this information to an NMS in response to an SNMP Get for the MIB-II sysContact MIB variable. This parameter’s value can be a textual identification of your company’s contact person for the system and information about how to contact that person.
2. **mib-system-name**—Set the identification of this Oracle CSM presented within MIB transactions. This value, along with the target name of the system (identified in the boot parameters) are the values reported for MIB-II when an SNMP GET is issued by the NMS for the MIB-II sysName variable. This parameter has no direct relation to the hostname parameter in the system configuration element.

By convention, this is the node’s FQDN. For SNMP MIB-II sysName GETs, the system returns SNMP communications in the following format: <targetName>[.<mib-system-name>]

targetName is the value configured in the target name (tn) boot parameter and mib-system-name is the value configured in this field.

3. **mib-system-location**—Set the physical location of this Oracle CSM that is reported within MIB transactions. This parameter is reported when an SNMP GET is issued by the NMS for the MIB-II sysLocation variable. This parameter has no direct relation to the location field in the system configuration element.
4. **snmp-enabled**—Set the SNMP system on this Oracle CSM to **enabled** or **disabled**. By default, this parameter is set to enabled. The valid values are:
 - enabled | disabled
5. **enable-snmp-syslog-notify**—Set whether SNMP traps are sent when the system generates an alarm message. The SNMP agent sends a trap when an alarm is generated if the following conditions are met:
 - SNMP is enabled.
 - This field is enabled.
 - The syslog severity level is equal to or greater than the severity level configured in the SNMP Syslog Level field.The default is **disabled**. Valid values are:
 - **enabled | disabled**
6. **enable-snmp-monitor-traps**—When this parameter is enabled, the Oracle CSM generates traps with unique trap-IDs for each syslog event. If this parameter is disabled, a single trap-ID is used for all events, with different values in the description string. The default is disabled. The valid values are:
 - enabled | disabled
7. **enable-snmp-auth-traps**—Set whether the SNMP authentication traps are enabled. If an SNMP request fails authentication because of an IPv4 address and SNMP community mismatch, the SNMP request will be rejected. This field determines if an SNMP trap will be sent in response to the authentication failure. The default is **disabled**. Valid values for this parameter are:
 - **enabled | disabled**
8. **enable-env-monitor-traps**—Set whether or not the SNMP environment monitor traps are enabled. Environment traps include main board PROM temperature, CPU voltage, power supplies, fan speeds, etc. The default is **disabled**. Valid values for this parameter are:
 - enabled | disabled
9. **snmp-syslog-his-table-length**—Set the length of the syslog trap history table. When a syslog message that meets the SNMP syslog level field criteria is generated and SNMP is enabled, the SNMP agent adds that message to a history table. This parameter indicates the number of entries the table can contain. The default is **1**. The valid range is:
 - Minimum—1

- Maximum—500

Once the last table entry is filled, the oldest entry will be overwritten with a new entry.

- 10. snmp-syslog-level**—Set the log severity level threshold that will cause the syslog trap to be sent to an NMS. When this criteria is met and the appropriate SNMP trap is sent, an entry is written to the SNMP Syslog History Table. The default is **warning**. The following are valid values:

- emergency | critical | major | minor | warning | notice | info | trace | debug | detail

SNMP Community Configuration

To configure SNMP communities:

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
```

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

```
ORACLE(system)# snmp-community
ORACLE(snmp-community)#
```

From this point, you can set SNMP community parameters.

The following is an example what an SNMP Community configuration might look like. Parameters not described in this section are omitted below.

```
snmp-community
  community-name          public
  access-mode             READ-ONLY
  ip-addresses            10.0.1.42
```

4. **community-name**—Set the SNMP community name of an active community where this Oracle CSM can send or receive SNMP information. A community name value can also be used as a password to provide authentication, thereby limiting the NMSs that have access to this system. With this field, the SNMP agent provides trivial authentication based on the community name that is exchanged in plain text SNMP messages.
5. **access-mode**—Set the access level for all NMSs defined within this SNMP community. The access level determines the permissions that other NMS hosts can wield over this Oracle CSM. The default is read-only. The valid values are:
 - read-only—allows GET requests.
 - read-write—allows both GET and SET requests.
6. **ip-addresses**—Set one or multiple IPv4 addresses that are valid within this SNMP community. These IPv4 addresses correspond with the IPv4 address of NMS applications that monitor or configure this Oracle CSM. Include the IPv4 addresses of all servers where Element Management System is installed.

Trap Receiver Configuration

To configure trap receivers:

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
```

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

```
ORACLE(system)# trap-receiver
ORACLE(trap-receiver)#
```

System Configuration

From this point, you can set trap receivers.

The following is an example what a trap receiver configuration might look like. Parameters not described in this section are omitted below.

```
trap-receiver
  ip-address          10.0.1.42:162
  filter-level       All
  community-name     public
```

4. **ip-address**—Set the IPv4 address of an authorized NMS. This parameter is the IPv4 address of an NMS where traps are sent. If you do not specify a port number, the default SNMP trap port of 162 will be used.
5. **filter-level**—Set the filter level threshold that indicates the severity level at which a trap to be sent to this particular trap receiver. The default for this parameter is critical.

Example: A trap with a severity level of Critical is generated, the SNMP agent will only send this trap to NMSs that are configured in a trap-receiver element and have a filter-level parameter of Critical.

The following table maps Syslog and SNMP alarms to trap receiver filter levels.

Filter Level	Syslog Severity Level	(SNMP) Alarm Severity Level
Critical	Emergency (1)	Emergency
	Critical (2)	Critical
Major	Emergency (1)	Emergency
	Critical (2)	Critical
	Major (3)	Major
Minor	Emergency (1)	Emergency
	Critical (2)	Critical
	Major (3)	Major
	Minor (4)	Minor
All	Emergency (1)	Emergency
	Critical (2)	Critical
	Major (3)	Major
	Minor (4)	Minor
	Warning (5)	Warning
	Notice (6)	
	Info (7)	
	Trace (8)	
	Debug (9)	

When configuring the trap-receiver element for use with Net-Net EMS systems, Acme Packet recommends that the filter-level parameter be set to All for that configuration element that includes Net-Net EMS servers.

6. **community-name**—Set the community name to which this trap receiver belongs. This community must be defined in the SNMP community element.

Syslog and Process Logs

Logging events is a critical part of diagnosing misconfigurations and optimizing operations. Oracle CSMs can send both syslog and process log data to appropriate hosts for storage and analysis.

Overview

The Oracle CSM generates two types of logs, syslogs and process logs. Syslogs conform to the standard used for logging servers and processes as defined in RFC 3164.

Process logs are Oracle proprietary logs. Process logs are generated on a per-task basis and are used mainly for debugging purposes. Because process logs are more data inclusive than syslogs, their contents usually encompass syslog log data.

Syslog and process log servers are both identified by an IPv4 address and port pair.

Process Log Messages

Process log messages are sent as UDP packets in the following format:

```
<file-name>:<log-message>
```

In this format, <filename> indicates the log filename and <log-message> indicates the full text of the log message as it would appear if it were written to the normal log file.

Syslog and Process Logs Configuration

This section describes how to configure syslog and process log servers.

To configure syslogs and process logs:

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
```

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

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

From this point, you can set process log parameters. Skip to the following process log configuration section.

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

```
ORACLE(system-config)# syslog-server
ORACLE(syslog-server)#
```

From this point, you can set syslog parameters. The following is an example what an syslog and process log configuration might look like. Parameters not described in this section are omitted below.

```
system-log-level          WARNING
syslog-server
  address                 172.15.44.12
  port                    514
  facility                4
process-log-level        NOTICE
process-log-ip-address   0.0.0.0
process-log-port         0
```

Syslog Configuration

The Oracle CSM supports multiple syslog servers. As the number of active syslog increases, the performance level of the Oracle CSM may decrease. Therefore, we recommend configuring no more than 8 syslog servers.

Set the following parameters to configure syslog servers:

1. **address**—Set the IPv4 address of a syslog server.
2. **port**—Set the port portion of the syslog server. The default is 514.
3. **facility**—Set an integer to identify a user-defined facility value sent in every syslog message from the Oracle CSM to the syslog server. This parameter is used only for identifying the source of this syslog message as coming from the Oracle CSM. It is not identifying an OS daemon or process. The default value for this parameter is 4. RFC 3164 specifies valid facility values.

In software release versions prior to Release 1.2, the Oracle CSM would send all syslog messages with a facility marker of 4.

4. **system-log-level**—Set which log severity levels write to the system log (filename: acmelog). The default is WARNING. Valid values are:
 - EMERGENCY | CRITICAL | MAJOR | MINOR | WARNING | NOTICE | INFO | TRACE | DEBUG | DETAIL

Process Log Configuration

Set the following parameters to configure the process log server:

1. **process-log-level**—Set the starting log level all processes running on the system use. Each individual process running on the system has its own process log. The default is NOTICE. Valid values are:
 - EMERGENCY | CRITICAL | MAJOR | MINOR | WARNING | NOTICE | INFO | TRACE | DEBUG | DETAIL
2. **process-log-ip-address**—Set the IPv4 address of the process log server. The default 0.0.0.0, which causes log messages to be written to the normal log file.
3. **process-log-port**—Set the port number associated with the process log server. The default value for this parameter is 0, which causes log messages to be written to the normal log file. The valid range is:
 - Minimum—0
 - Maximum—65535.

Host Routes

Host routes let you insert entries into the Oracle CSM's routing table. These routes affect traffic that originates at the Oracle CSM's host process. Host routes are used primarily for steering management traffic to the correct network.

When traffic is destined for a network that is not explicitly defined on a Oracle CSM, the default gateway (located in the system config) is used. If you try to route traffic to a specific destination that is not accessible through the default gateway, you need to add a host route. Host routes can be thought of as a default gateway override.

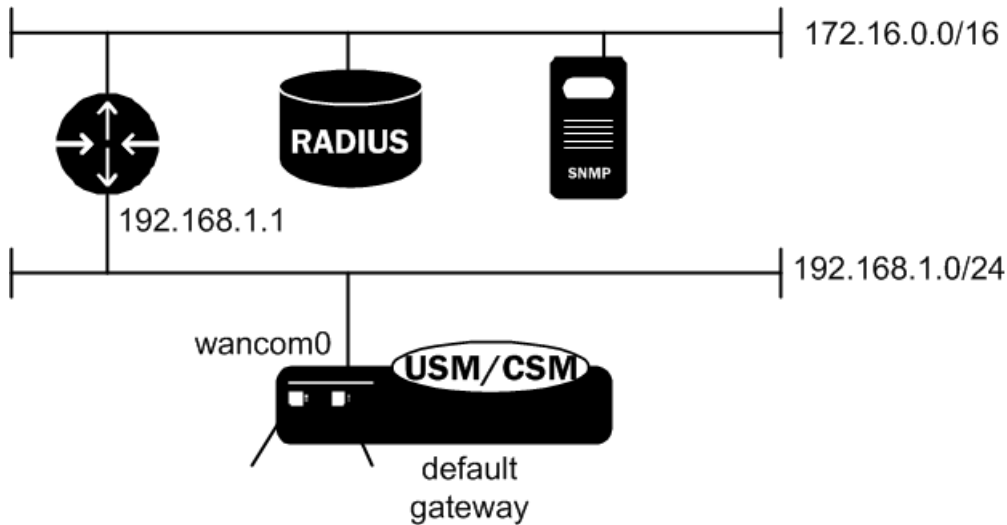
Certain SIP configurations require that the default gateway is located on a media interface. In this scenario, if management applications are located on a network connected to an administrative network, you will need to add a host route for management connectivity.

When source-based routing is used, the default gateway must exist on a media interface. Host routes might be needed to reach management applications connected to a management port in this kind of situation as well.

Host Routes Example

Because SIP signaling over media interfaces is enabled, the default gateway uses an IPv4 address assigned to a media interface. Maintenance services (SNMP and Radius) are located on a network connected to, but separate from, the 192.168.1.0/24 network on wancom0. In order to route Radius or SNMP traffic to an NMS (labeled as SNMP in the

following example), a host route entry must be a part of the Oracle CSM configuration. The host route tells the host how to reach the 172.16.0.0/16 network. The actual configuration is shown in the example in the next section of this guide.



Host Route Configuration

To configure a host route:

1. Access the **host-route** configuration element.

```
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)# host-route
ORACLE(host-route)#
```

2. **dest-network**—Set the IP address of the destination network that this host route points toward.
3. **netmask**—Set the netmask portion of the destination network for the route you are creating. The netmask is in dotted decimal notation.
4. **gateway**—Set the gateway that traffic destined for the address defined in the first two elements should use as its first hop.
5. Type **done** to save your configuration.

Setting Holidays in Local Policy

This section explains how to configure holidays on the Oracle CSM.

You can define holidays that the Oracle CSM recognizes. Holidays are used to identify a class of days on which a local policy is enacted. All configured holidays are referenced in the local-policy-attributes configuration subelement as an H in the days-of-week parameter. Because holidays are entered on a one-time basis per year, you must configure a new set of holidays yearly.

Holidays Configuration

To configure holidays:

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
```

System Configuration

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

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

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

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

From this point, you can configure the holidays subelement. To view all holidays parameters, enter a `?` at the system prompt.

```
holiday
      date           2005-01-01
      description    New Years Day
```

To configure a holiday, add an entry for the following parameters in the holidays element:

5. `date`—Enter the holiday's date in YYYY-MM-DD format.
6. `description`—Enter a short description for the holiday you are configuring. If the description contains words separated by spaces, enter the full description surrounded by quotation marks.

Enhanced Control of UDP and TCP Ports

This section explains how to configure the Oracle CSM for finer control of the set of UDP and TCP ports that on which the Oracle CSM provides services. The settings you can configure have an impact on:

- UDP/TCP port 111 (the RPC services port), which is disabled on Oracle CSM startup but can be enabled in the boot parameters
- TCP ports 3000 (used when notify commands are issued remotely, i.e. via an element management system) and 3001 (used for remote configuration, i.e. via an element management system), which can now be enabled or disabled in the system configuration

Neither configuration for these features is covered by RTC, so you must reboot your Oracle CSM for changes to take effect. Be aware that rebooting can cause system downtime, and plan accordingly.

Port 111 Configuration

To enable port 111 using Oracle CSM boot parameters:

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

```
ORACLE# configure terminal
```

2. To enter the boot parameters so that you can configure them, type `bootparam` and press Enter.

```
ORACLE(configure)# bootparam
```

3. Press Enter to scroll through the list of boot parameters until you reach the setting for flags.

To set this value correctly, you need to add the value `0x200000` to your existing flag setting in the boot parameters. In the example below, the existing flag value is `0x30008`. When the value `0x200000` is added, the result is `0x230008`. The result is the value that you need to set.

When you reach the flag setting, type the value representing the flags you need (`0x230008` in the example below) and press Enter. Continue to press Enter to finish scrolling through the rest of the boot parameters.

```
'.' = clear field; '-' = go to previous field; ^D = quit
boot device           : wancom0
processor number      : 0
host name             : acmepacket8
file name             : /tffs0/sd220p9.gz
inet on ethernet (e) : 10.0.1.57:ffff0000
```

```

inet on backplane (b)      : 0.0.0.0
host inet (h)             : 10.0.1.5
gateway inet (g)         : 10.0.0.1
user (u)                  : user
ftp password (pw)        : password
flags (f)                 : 0x30008 0x230008
target name (tn)         : acmesystem
startup script (s)       : 0
other (o)                 :
NOTE: These changed parameters will not go into effect until reboot. Also,
be aware that some boot parameters may also be changed through the PHY and
Network Interface Configurations.
ORACLE(configure)#

```

4. Type **exit** to return to the main Superuser menu so that you can reboot your Oracle CSM and apply the settings you have entered.

```
ORACLE(configure)# exit
```

5. Reboot your Oracle CSM. Type a **y** and press Enter to reboot.

```

ORACLE# reboot
-----
WARNING: you are about to reboot this SD!
-----
Reboot this SD [y/n]?:y

```

Port 3000 and 3001 Configuration

To control TCP ports 3000 and 3001 in the system configuration:

1. Access the **security-config** configuration element.

```

ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# security-config
ORACLE(security-config)#

```

2. Type **select** to begin editing the **system-config** object.

```

ORACLE(system-config)# select
ACMEPACKET(system-config)#

```

3. The parameter controlling ports 3000 and 3001 is called **remote-control**, and its default is enabled. To disable the ports, set this parameter to **disabled**.

```
ORACLE(system-config)# remote-control disabled
```

4. Type **done** to save your configuration.
5. Reboot your Oracle CSM. Type a **y** and press Enter to reboot.

```

ORACLE# reboot
-----
WARNING: you are about to reboot this SD!
-----
Reboot this SD [y/n]?:y

```

DNS Transaction Timeout

This section explains how to configure the DNS transaction timeout interval on a per network-interface basis. You can currently configure the Oracle CSM with a primary and two optional backup DNS servers. The Oracle CSM queries the primary DNS server and upon not receiving a response within the configured number of seconds, queries the backup1 DNS server and if that times out as well, then contacts the backup2 DNS server.

Retransmission Logic

The retransmission of DNS queries is controlled by three timers. These timers are derived from the configured DNS timeout value and from underlying logic that the minimum allowed retransmission interval should be 250 milliseconds; and that the Oracle CSM should retransmit 3 times before timing out to give the server a chance to respond.

- Init-timer is the initial retransmission interval. If a response to a query is not received within this interval, the query is retransmitted. To safeguard from performance degradation, the minimum value allowed for this timer is 250 milliseconds.
- Max-timer is the maximum retransmission interval. The interval is doubled after every retransmission. If the resulting retransmission interval is greater than the value of max-timer, it is set to the max-timer value.
- Expire-timer: is the query expiration timer. If a response is not received for a query and its retransmissions within this interval, the server will be considered non-responsive and the next server in the list will be tried.

The following examples show different timeout values and the corresponding timers derived from them.

```
timeout >= 3 seconds
Init-timer = Timeout/11
Max-Timer = 4 * Init-timer
Expire-Timer = Timeout
timeout = 1 second
Init-Timer = 250 ms
Max-Timer = 250 ms
Expire-Timer = 1 sec
timeout = 2 seconds
Init-Timer = 250 ms
Max-Timer = 650 ms
Expire-Timer = 2sec
```

DNS Transaction Timeout Configuration

To configure DNS transaction timeout:

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
```

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

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

From this point, you can configure network interface parameters. To view all network interface parameters, enter a ? at the system prompt.

4. dns-timeout—Enter the total time in seconds you want to elapse before a query (and its retransmissions) sent to a DNS server would timeout. The default is 11 seconds. The valid range is:
 - Minimum—1
 - Maximum—999999999.

If a query sent to the primary DNS server times out, the backup1 DNS server is queried. If the query times out after the same period of time elapses, the query continues on to the backup2 DNS server.

5. Save and activate your configuration.

DNS Server Status via SNMP

The Oracle CSM monitors the status of all configured DNS servers used by a SIP daemon. If a DNS server goes down, a major alarm is sent. If all DNS servers used by a SIP daemon are down, a critical alarm is sent. The `apAppsDnsServerStatusChangeTrap` is sent for both events.

You can poll the status of a DNS server using the `apAppsDNSServerStatusTable` in the `ap-apps.mib`.

Once the `apAppsDnsServerStatusChangeTrap` has been sent, a 30 second window elapses until the server status is checked again. At the 30 second timer expiration, if the server is still down, another trap and alarm are sent. If the server has been restored to service, the `apAppsDnsServerStatusChangeClearTrap` is sent.

Persistent Protocol Tracing

This section explains how to configure persistent protocol tracing to capture specific SIP and MGCP protocol message logs and persistently send them off the Oracle CSM, even after rebooting the system.

About Persistent Protocol Tracing

You can configure sending protocol message logs off of the Oracle CSM, and have that persist after a reboot. You no longer have to manually issue the `notify` command each time you reboot.

To support persistent protocol tracing, you configure the following system-config parameters:

- `call-trace`—Enable/disable protocol message tracing (currently only `sipmsg.log` and `alg.log`) regardless of the `process-log-level` setting. If the `process-log-level` is set to `trace` or `debug`, `call-trace` will not disable.
- `internal-trace`—Enable/disable internal ACP message tracing for all processes, regardless of `process-log-level` setting. This applies to all `*.log` (internal ACP message exchange) files other than `sipmsg.log` and `alg.log`. If the `process-log-level` is set to `trace` or `debug`, `call-trace` will not disable.
- `log-filter`—Determine what combination of protocol traces and logs are sent to the log server defined by the `process-log-ip` parameter value. You can also fork the traces and logs, meaning that you keep trace and log information in local storage as well as sending it to the server. You can set this parameter to any of the following values: `none`, `traces`, `traces-fork`, `logs`, `logs`, `all`, or `all-fork`.

The Oracle CSM uses the value of this parameter in conjunction with the `process-log-ip` and `process-log-port` values to determine what information to send. If you have configured the `proc-log-ip` and `proc-log-port` parameters, choosing `traces` sends just the trace information (provided they are turned on), `logs` sends only process logs (`log.*`), and `all` sends everything (which is the default).

About the Logs

When you configure persistent protocol tracing, you affect the following types of logs.



Note: Enabling logs can have an impact on Oracle CSM performance.

Process Logs

Events are logged to a process log flow from tasks and are specific to a single process running on the Oracle CSM. By default they are placed into individual files associated with each process with the following name format:

```
log.<taskname>
```

By setting the new `log-filter` parameter, you can have the logs sent to a remote log server (if configured). If you set `log-filter` to `logs` or `all`, the logs are sent to the log server. Otherwise, the logs are still captured at the level the `process-log-level` parameter is set to, but the results are stored on the Oracle CSM's local storage.

Communication Logs

These are the communication logs between processes and system management. The logs are usually named `<name>.log`, with `<name>` being the process name. For example, `sipd.log`.

System Configuration

This class of log is configured by the new `internal-trace` parameter.

Protocol Trace Logs

The only protocol trace logs included at this time are `sipmsg.log` for SIP and `alg.log` for MGCP. All of the logs enabled with the `call-trace` parameter are sent to remote log servers, if you also set the `log-filter` parameter to `logs` or `all`.

Persistent Protocol Tracing Configuration

Before you configure persistent protocol tracing, ensure you have configured the process logs by setting the system configuration's `process-log-ip` parameter.

To configure persistent protocol tracing:

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
```

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

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

4. `call-trace`—Set to `enabled` to enable protocol message tracing for `sipmsg.log` for SIP and `alg.log` for MGCP. The default is `disabled`. The valid values are:

- `enabled` | `disabled`

5. `internal-trace`—Set to `enabled` to enable internal ACP message tracing for all processes. The default is `disabled`. The valid values are:

- `enabled` | `disabled`

6. `log-filter`—Choose the appropriate setting for how you want to send and/or store trace information and process logs. The valid values are:

- `none`—No information will be sent or stored.
- `traces`—Sends the trace information to both the log server; includes `<name>.log` files that contain information about the Oracle CSM's internal communication processes (`<name>` is the name of the internal process)
- `traces-fork`—Sends the trace information to both the log server and also keeps it in local storage; includes `<name>.log` files that contain information about the Oracle CSM's internal communication processes (`<name>` is the name of the internal process)
- `logs`—Sends the process logs to both the log server; includes `log.*` files, which are Oracle CSM process logs
- `logs-fork`—Sends the process logs to both the log server and also keeps it in local storage; includes `log.*` files, which are Oracle CSM process logs
- `all`—Sends all logs to the log servers that you configure
- `all-fork`—Sends all logs to the log servers that you configure, and it also keeps the logs in local storage

7. Save and activate your configuration.

System Access Control

You can configure a system access control list (ACL) for your Oracle CSM that determines what traffic the Oracle CSM allows over its management interface (`wancom0`). By specifying who has access to the Oracle CSM via the management interface, you can provide DoS protection for this interface.

Using a list of IP addresses and subnets that are allowable as packet sources, you can configure what traffic the Oracle CSM accepts and what it denies. All IP packets arriving on the management interface are subject; if it does not match your configuration for system ACL, then the Oracle CSM drops it.



Note: All IP addresses configured in the SNMP community table are automatically permitted.

Adding an ACL for the Management Interface

The new subconfiguration system-access-list is now part of the system configuration, and its model is similar to host routes. For each entry, you must define an IP destination address and mask; you can specify either the individual host or a unique subnet.

If you do not configure this list, then there will be no ACL/DoS protection for the Oracle CSM's management interface.

You access the system-access-list via system path, where you set an IP address and netmask. You can configure multiple system ACLs using this configuration.

To add an ACL for the management interface:

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

```
ORACLE# configure terminal
```

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

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

3. Type system-access-list and press Enter.

```
ORACLE(system)# system-access-list
ORACLE(system-access-list)#
```

4. source-address—Enter the IP address representing for the source network for which you want to allow traffic over the management interface.
5. netmask—Enter the netmask portion of the source network for the traffic you want to allow. The netmask is in dotted decimal notation.

Notes on Deleting System ACLs

If you delete a system ACL from your configuration, the Oracle CSM checks whether or not there are any active FTP or Telnet client was granted access when the entry was being removed. If such a client were active during ACL removal, the Oracle CSM would warn you about the condition and ask you to confirm the deletion. If you confirm the deletion, then the Oracle CSM's session with the active client is suspended.

The following example shows you how the warning message and confirmation appear. For this example, an ACLI has been deleted, and the user is activating the configuration that reflects the change.

```
ORACLE # activate-config
Object deleted will cause service disruption:
system-access-list: identifier=172.30.0.24
** WARNING: Removal of this system-ACL entry will result
           in the lockout of a current FTP client
Changes could affect service, continue (y/n) y
Activate-Config received, processing.
```

System TCP Keepalive Settings

You can configure the Oracle CSM to control TCP connections by setting:

- The amount of time the TCP connection is idle before the Oracle CSM starts sending keepalive messages to the remote peer
- The number of keepalive packets the Oracle CSM sends before terminating the TCP connection

If TCP keepalive fails, then the Oracle CSM will drop the call associated with that TCP connection.

In the ALCI, a configured set of network parameters appears as follows:

System Configuration

```
network-parameters
tcp-keepinit-timer          75
tcp-keepalive-count         4
tcp-keepalive-idle-timer    400
tcp-keepalive-interval-timer 75
tcp-keepalive-mode          0
```

Then you apply these on a per-interface basis.

System TCP Keepalive Configuration

TCP settings are global, and then enabled or disabled on a per-interface basis.

To configure TCP keepalive parameters on your Oracle CSM:

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

```
ORACLE# configure terminal
```

2. Type `system` and press Enter to access the system-related configurations.

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

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

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

4. `tcp-keepinit-timer`—If a TCP connection cannot be established within some amount of time, TCP will time out the connect attempt. It can be used to set the initial timeout period for a given socket, and specifies the number of seconds to wait before the connect attempt is timed out. For passive connections, this value is inherited from the listening socket. The default is 75. The valid range is:
 - Minimum—0
 - Maximum—999999999
5. `tcp-keepalive-count`—Enter the number of packets the Oracle CSM sends to the remote peer before it terminates the TCP connection. The default is 8. The valid range is:
 - Minimum—0
 - Maximum—223-1
6. `tcp-keepalive-idle-timer`—Enter the number of seconds of idle time before TCP keepalive messages are sent to the remote peer if the `SO-KEEPALIVE` option is set. The default is 7200. The valid range is:
 - Minimum—30
 - Maximum—7200
7. `tcp-keepalive-interval-timer`—When the `SO_KEEPALIVE` option is enabled, TCP probes a connection that has been idle for some amount of time. If the remote system does not respond to a keepalive probe, TCP retransmits the probe after a set amount of time. This parameter specifies the number of seconds to wait before retransmitting a keepalive probe. The default value is 75 seconds. The valid range is:
 - Minimum—15
 - Maximum—75
8. `tcp-keepalive-mode`—Set the TCP keepalive response sequence number. The default is 0. The valid values are:
 - 0—The sequence number is sent un-incremented
 - 1—The number is incremented
 - 2—No packets are sent

Configurable TCP Timers

You can configure your Oracle CSM to detect failed TCP connections more quickly so that data can be transmitted via an alternate connection before timers expire. Across all protocols, you can now control the following for TCP:

- Connection establishment
- Data retransmission
- Timer for idle connections

These capabilities all involve configuring an options parameter that appears in the network parameters configuration.

Configuring TCP Connection Establishment

To establish connections, TCP uses a three-way handshake during which two peers exchange TCP SYN messages to request and confirm the active open connection. In attempting this connection, one peer retransmits the SYN messages for a defined period of time if it does not receive acknowledgement from the terminating peer. You can configure the amount of time in seconds between the retries as well as how long (in seconds) the peer will keep retransmitting the messages.

You set two new options in the network parameters configuration to specify these amounts of time: `atcp-syn-rxmt-interval` and `atcp-syn-rxmt-maxtime`.

Note that for all configured options, any values entered outside of the valid range are silently ignored during configuration and generate a log when you enter the activate command.

To configure TCP connection establishment:

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

```
ORACLE# configure terminal
```

2. Type `system` and press Enter.

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

3. Type `network-parameters` and press Enter.

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

4. `options`—Set the options parameter by typing `options`, a Space, the option name `atcp-syn-rxmt-interval=x` (where `x` is a value in seconds between 2 and 10) with a plus sign in front of it. Then press Enter. This value will be used as the interval between TCP SYN messages when the Oracle CSM is trying to establish a connection with a remote peer.

Now enter a second option to set the maximum time for trying to establish a TCP connection. Set the options parameter by typing `options`, a Space, the option name `atcp-syn-rxmt-maxtime=x` (where `x` is a value in seconds between 5 and 75) with a plus sign in front of it. Then press Enter.

```
ORACLE(network-parameters)# options +atcp-syn-rxmt-interval=5
ORACLE(network-parameters)# options +atcp-syn-rxmt-maxtime=30
```

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 configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.



Note: `atcp-syn-rxmt-maxtime=x` option is equivalent to the `tcp-keepinit-timer` parameter, but only affects ATCP.

5. Save and activate your configuration.

Configuring TCP Data Retransmission

TCP is considered reliable in part because it requires that entities receiving data must acknowledge transmitted segments. If data segments go unacknowledged, then they are retransmitted until they are finally acknowledged or until the maximum number of retries has been reached. You can control both the number of times the Oracle CSM tries to retransmit unacknowledged segments and the periodic interval (how often) at which retransmissions occur.

You set two new options in the network parameters configuration to specify how many retransmissions are allowed and for how long: `atcp-rxmt-interval` and `atcp-rxmt-count`.

System Configuration

To configure TCP data retransmission:

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

```
ORACLE# configure terminal
```

2. Type system and press Enter.

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

3. Type network-parameters and press Enter.

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

4. options—Set the options parameter by typing options, a Space, the option name atcp-rxmt-interval=x (where x is a value in seconds between 2 and 60) with a plus sign in front of it. Then press Enter. This value will be used as the interval between retransmission of TCP data segments that have not been acknowledged.

Now enter a second option to set the number of times the Oracle CSM will retransmit a data segment before it declares the connection failed. Set the options parameter by typing options, a Space, the option name atcp-rxmt-count=x (where x is a value between 4 and 12 representing how many retransmissions you want to enable) with a plus sign in front of it. Then press Enter.

```
ORACLE(network-parameters)# options +atcp-rxmt-interval=30
ORACLE(network-parameters)# options +atcp-rxmt-count=6
```

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 configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.

5. Save and activate your configuration.

Timer for Idle Connections

When enabled to do so, the Oracle CSM monitors inbound TCP connections for inactivity. These are inbound connections that the remote peer initiated, meaning that the remote peer sent the first SYN message. You can configure a timer that sets the maximum amount of idle time for a connection before the Oracle CSM consider the connection inactive. Once the timer expires and the connection is deemed inactive, the Oracle CSM sends a TCP RST message to the remote peer.

To configure the timer for TCP idle connections:

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

```
ORACLE# configure terminal
```

2. Type system and press Enter.

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

3. Type network-parameters and press Enter.

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

4. options—Set the options parameter by typing options, a Space, the option name atcp-idle-timer=x (where x is a value in seconds between 120 and 7200) with a plus sign in front of it. Then press Enter. This value will be used to measure the activity of TCP connections; when the inactivity on a TCP connection reaches this value in seconds, the Oracle CSM declares it inactive and drops the session.

```
ORACLE(network-parameters)# options +atcp-idle-timer=900
```

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 configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.

5. Save and activate your configuration.

Historical Data Recording (HDR)

Updated HDR and HDR configuration information resides in the Net-Net® C-Series Historical Data Recording (HDR) Resource Guide Version C6.3.0, 400-0141-63 (Net-Net 4000 S-CX6.3.0 HDR Resource Guide.pdf). This document is available with the complete Version S-CX6.3.0 documentation set.

RAMdrive Log Cleaner

The RAMdrive log cleaner allows the Oracle CSM to remove log files proactively and thereby avoid situations where running low on RAMdrive space is a danger. Because even a small amount of logging can consume a considerable space, you might want to enable the RAMdrive log cleaner.

The RAMdrive cleaner periodically checks the remaining free space in the RAMdrive and, depending on the configured threshold, performs a full check on the /ramdrv/logs directory. During the full check, the RAMdrive cleaner determines the total space logs files are using and deletes log files that exceed the configured maximum lifetime. In addition, if the cleaner finds that the maximum log space has been exceeded or the minimum free space is not sufficient, it deletes older log files until the thresholds are met.

Not all log files, however, are as active as others. This condition affects which log files the log cleaner deletes to create more space in RAMdrive. More active log files rotate through the system more rapidly. So, if the log cleaner were to delete the oldest of these active files, it might not delete less active logs files that could be older than the active ones. The log cleaner thus deletes files that are truly older, be they active or inactive.

Applicable Settings

In the system configuration, you establish a group of settings in the options parameter that control the log cleaner's behavior:

- ramdrv-log-min-free—Minimum percent of free space required when rotating log files.

When the amount of free space on the RAMdrive falls below this value, the log cleaner deletes the oldest copy of the log file. The log cleaner also uses this setting when performing period cleaning.

- ramdrv-log-max-usage—Maximum percent of the RAMdrive the log files can use.

The log cleaner removes old log files to maintain this threshold.

- ramdrv-log-min-check—Minimum percent of free space on the RAMdrive that triggers the log cleaner to perform a full check of log files.
- ramdrv-min-log-check—Minimum time (in seconds) between log cleaner checks.
- ramdrv-max-log-check—Maximum time (in seconds) between log cleaner checks. This value must be greater than or equal to the ramdrv-min-log-check.
- ramdrv-log-lifetime—Maximum lifetime (in days) for log files. You give logs unlimited lifetime by entering a value of 0.

Clean-Up Procedure

The log cleaner checks the amount of space remaining in the RAMdrive and performs a full check of the logs directory when:

- Free space is less than the minimum percent of the RAMdrive that triggers a full check of log files
- The amount of free space has changed by more than 5% of the RAMdrive capacity since the last full check
- A full check of the logs directory has not been performed in the last hour

When it checks the logs directory, the log cleaner inventories the collected log files. It identifies each files as one of these types:

- Process log—Files beginning with log.
- Internal trace file—A <task>.log file
- Protocol trace file—Call trace including sipmsg.log, dns.log, sipddns.log, and alg.log

System Configuration

- CDR file—File beginning with cdr

Next, the log cleaner determines the age of the log files using the number of seconds since the log files were created. Then it orders the files from oldest to newest. The age adjusts such that it always increases as the log file sequence number (a suffix added by file rotation) increases. The log cleaner applies an additional weighting factor to produce a weighted age that favors the preservation of protocol traces files over internal trace files, and internal trace files over process log files. The base log file and CDR files are excluded from the age list and so will not be deleted; the accounting configuration controls CDR file aging.

With the age list constructed, the log cleaner examines the list from highest weighted age to lowest. If the actual file age exceeds the RAMdrive maximum log lifetime, the log cleaner deletes it. Otherwise, the log cleaner deletes files until the maximum percent of RAMdrive that logs can use is no longer exceeded and until the minimum percent of free space required when rotating logs is available.

Clean-Up Frequency

The minimum free space that triggers a full check of log files and the maximum time between log file checks control how often the log cleaner performs the clean-up procedure. When it completes the procedure, the log cleaner determines the time interval until the next required clean-up based on the RAMdrive's state.

If a clean-up results in the deletion of one or more log files or if certain thresholds are exceeded, frequency is based on the minimum time between log cleaner checks. Otherwise, the system gradually increases the interval up to the maximum time between log cleaner checks. The system increases the interval by one-quarter of the difference between the minimum and maximum interval, but not greater than one-half the minimum interval or smaller than 10 seconds. For example, using the default values, the interval would be increased by 30 seconds.

RAMdrive Log Cleaner Configuration

You configure the log cleaner's operating parameters and thresholds in the system configuration. Note that none of these settings is RTC-supported, so you must reboot your Oracle CSM in order for them to take effect. If you are using this feature on an HA node, however, you can add this feature without impact to service by activating the configuration, rebooting the standby, switching over to make the newly booted standby active, and then rebooting the newly standby system.

Unlike other values for options parameters, the Oracle CSM validates these setting when entered using the ACLI. If any single value is invalid, they all revert to their default values.

To configure the RAMdrive log cleaner:

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

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

2. Type `system` and press Enter.

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

3. Type `system-config` and press Enter.

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

4. `options`—Set the options parameter by typing `options`, a Space, `<option name>=X` (where X is the value you want to use) with a plus sign in front of it. Then press Enter.

Remember that if any of your settings are invalid, the Oracle CSM changes the entire group of these options back to their default settings.

Option Name	Description
ramdrv-log-min-free	Minimum percent of free space required when rotating log files.

Option Name	Description
	When the amount of free space on the RAMdrive falls below this value, the log cleaner deletes the oldest copy of the log file. The log cleaner also uses this setting when performing period cleaning. Default=40; Minimum=15; Maximum=75
ramdrv-log-max-usage	Maximum percent of the RAMdrive the log files can use. The log cleaner removes old log files to maintain this threshold. Default=40; Minimum=15; Maximum=75
ramdrv-log-min-check	Minimum percent of free space on the RAMdrive that triggers the log cleaner to perform a full check of log files. Default=50; Minimum=25; Maximum=75
ramdrv-min-log-check	Maximum time (in seconds) between log cleaner checks. This value must be greater than or equal to the ramdrv-min-log-check. Default=180; Minimum=40; Maximum=1800
ramdrv--log-lifetime	Maximum lifetime (in days) for log files. You give logs unlimited lifetime by entering a value of 0. Default=30; Minimum=2; Maximum=9999

```
ORACLE (system-config) # options +ramdrv-log-min-free=50
ORACLE (system-config) # options +ramdrv-log-max-usage=50
ORACLE (system-config) # options +ramdrv-log-min-check=35
ORACLE (system-config) # options +ramdrv-min-log-check=120
ORACLE (system-config) # options +ramdrv-max-log-free=1500
ORACLE (system-config) # options +ramdrv-log-lifetime=7
```

If you type options and then the option value for either of these entries without the plus sign, you will overwrite any previously configured options. In order to append the new options to this configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.

5. Reboot your Oracle CSM.

Configurable Alarm Thresholds and Traps

The Oracle CSM supports user-configurable threshold crossing alarms. These configurations let you identify system conditions of varying severity which create corresponding alarms of varying severity. You configure an alarm threshold type which indicates the resource to monitor. The available types are:

- cpu — CPU utilization monitored as a percentage of total CPU capacity
- memory — memory utilization monitored as a percentage of total memory available
- sessions — license utilization monitored as a percentage of licensed session capacity
- space — remaining disk space (configured in conjunction with the volume parameter - see the Storage Expansion Module Monitoring section of the *Accounting Guide* for more information.)
- deny-allocation — denied entry utilization monitored as a percentage of reserved, denied entries.

For the alarm type you create, the Oracle CSM can monitor for 1 through 3 severity levels as minor, major, and critical. Each of the severities is configured with a corresponding value that triggers that severity. For example the configuration for a CPU alarm that is enacted when CPU usage reaches 50%:

```
alarm-threshold
type                               cpu
```

System Configuration

```
severity      minor
value        50
```

You may create addition CPU alarms for increasing severities. For example:

```
alarm-threshold
  type          cpu
  severity      critical
  value         90
```

The alarm state is enacted when the resource defined with the type parameter exceeds the value parameter. When the resource drops below the value parameter, the alarm is cleared.

SNMP Traps

When a configured alarm threshold is reached, the Oracle CSM sends an `apSysMgmtGroupTrap`. This trap contains the resource type and value for the alarm configured in the `alarm-threshold` configuration element. The trap does not contain information associated with configured severity for that value.

```
apSysMgmtGroupTrap      NOTIFICATION-TYPE
  OBJECTS                { apSysMgmtTrapType, apSysMgmtTrapValue }
  STATUS                 current
  DESCRIPTION
    " The trap will generated if value of the monitoring object
    exceeds a certain threshold. "
  ::= { apSystemManagementNotifications 1 }
```

When the resource usage retreats below a configured threshold, the Oracle CSM sends an `apSysMgmtGroupClearTrap`.

```
apSysMgmtGroupClearTrap NOTIFICATION-TYPE
  OBJECTS                { apSysMgmtTrapType }
  STATUS                 current
  DESCRIPTION
    " The trap will generated if value of the monitoring object
    returns to within a certain threshold. This signifies that
    an alarm caused by that monitoring object has been cleared. "
  ::= { apSystemManagementNotifications 2 }
```

The alarm and corresponding traps available through the User Configurable Alarm Thresholds functionality are summarized in the following table.

Alarm	Severity	Cause	Actions
CPU	minor major critical	high CPU usage	apSysMgmtGroupTrap sent with apSysCPUUtil apSysMgmtTrapValue
memory	minor major critical	high memory usage	apSysMgmtGroupTrap sent with apSysMemoryUtil apSysMgmtTrapValue
sessions	minor major critical	high license usage	apSysMgmtGroupTrap sent with apSysLicenseCapacity apSysMgmtTrapValue
space	minor major critical	high HDD usage, per volume	apSysMgmtStorageSpaceAvailThresholdTrap sent with: apSysMgmtSpaceAvailCurrent

Alarm	Severity	Cause	Actions
			apSysMgmtSpaceAvailMinorThreshold apSysMgmtSpaceAvailMajorThreshold apSysMgmtSpaceAvailCriticalThreshold apSysMgmtPartitionPath
deny allocation	minor major critical	high usage of denied ACL entries	apSysMgmtGroupTrap sent with apSysCurrentEndptsDenied apSysMgmtTrapValue

Alarm Thresholds Configuration

To configure alarm thresholds:

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
```

3. Type system-config and press Enter.

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

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

```
ORACLE(system-config)# alarm-threshold
ORACLE(alarm-threshold)#
```

5. type — Enter the type of resource which this alarm monitors. Valid values include:

- cpu
- memory
- sessions
- space
- deny-allocation

6. volume — Enter the logical disk volume this alarm monitors (used only in conjunction when type = space).

7. severity — Set the severity of the threshold. Valid values include:

- minor
- major
- critical

8. value — Enter the value from 1 to 99, indicating the percentage, which when exceeded generates an alarm.

9. Save and activate your configuration.

Alarm Synchronization

Two trap tables in the ap-smgmt.mib record trap information for any condition on the Oracle CSM that triggers an alarm condition. You can poll these two tables from network management systems, OSS applications, and the Session Delivery Manager to view the fault status on one or more Oracle CSM s.

The two trap tables that support alarm synchronization, and by polling them you can obtain information about the current fault condition on the Oracle CSM . These tables are:

System Configuration

- **apSysMgmtTrapTable**—You can poll this table to obtain a summary of the Oracle CSM 's current fault conditions. The table records multiples of the same trap type that have occurred within a second of one another and have different information. Each table entry contains the following:
 - Trap identifier
 - System time (synchronized with an NTP server)
 - sysUpTime
 - Instance number
 - Other trap information for this trap identifier
- **apSysMgmtTrapInformationTable**—You can poll this table to obtain further details about the traps recorded in the **apSysMgmtTrapTable** table. The following information appears:
 - Data index
 - Data type
 - Data length
 - The data itself (in octets)

Trap tables do not record information about alarm severity.

The **apSysMgmtTrapTable** can hold up to 1000 entries, and you can configure the number of days these entries stay in the table for a maximum of seven days. If you set this parameter to 0 days, the feature is disabled. And if you change the setting to 0 days from a greater value, then the Oracle CSM purges the tables.

Caveats

Note that the Oracle CSM does not replicate alarm synchronization table data across HA nodes. That is, each Oracle CSM in an HA node maintains its own tables.

Alarm Synchronization Configuration

You turn on alarm synchronization in the system configuration.

To use alarm synchronization:

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

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

2. Type `system` and press Enter.

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

3. Type `system-config` and press Enter.

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

4. **trap-event-lifetime**—To enable alarm synchronization—and cause the Oracle CSM to record trap information in the **apSysMgmtTrapTable** and the **apSysMgmtTrapInformationTable**—set this parameter to the number of days you want to keep the information. Leaving this parameter set to 0 (default) turns alarm synchronization off, and you can keep information in the tables for up to 7 days. 7 is the maximum value for this parameter.

Accounting Configuration

The Oracle CSM offers support for RADIUS, an accounting, authentication, and authorization (AAA) system. In general, RADIUS servers are responsible for receiving user connection requests, authenticating users, and returning all configuration information necessary for the client to deliver service to the user.

You can configure your Oracle CSM to send call accounting information to one or more RADIUS servers. This information can help you to see usage and QoS metrics, monitor traffic, and even troubleshoot your system.

This guide contains all RADIUS information, as well as information about:

- Local CDR storage on the Oracle CSM , including CSV file format settings
- The ability to send CDRs via FTP to a RADIUS sever (the FTP push feature)
- Per-realm accounting control
- Configurable intermediate period
- RADIUS CDR redundancy
- RADIUS CDR content control

Stream Control Transfer Protocol Overview

The Stream Control Transmission Protocol (SCTP) was originally designed by the Signaling Transport (SIGTRAN) group of IETF for Signalling 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.

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

System Configuration

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.

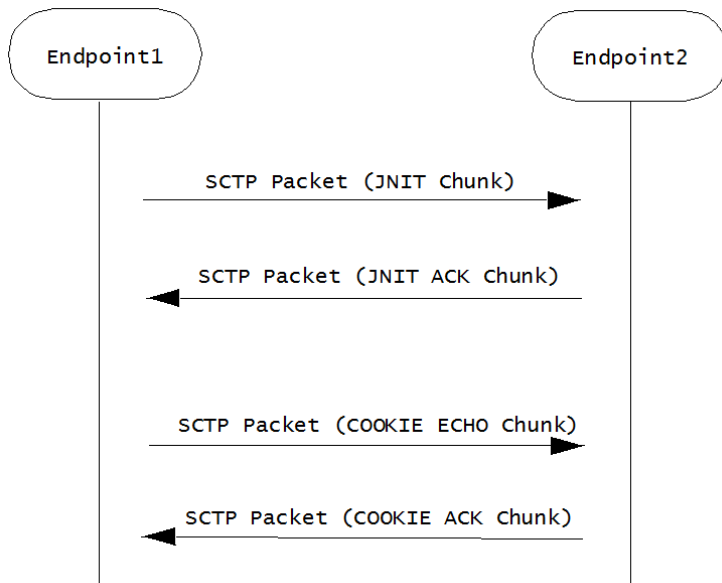
SCTP Transport Address

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.

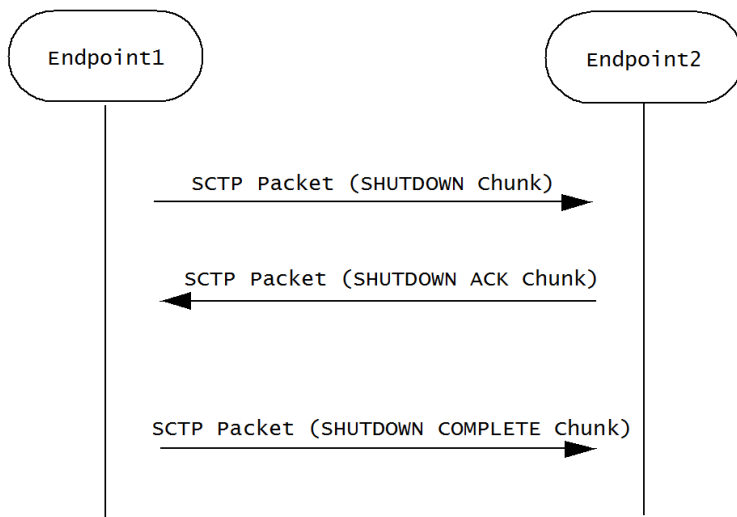
Both the INIT chunk (issued by the initiator) and INIT ACK chunk (issued by the responder) specify the number of outbound streams supported by the association, as well as the maximum inbound streams accepted from the other endpoint.

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.

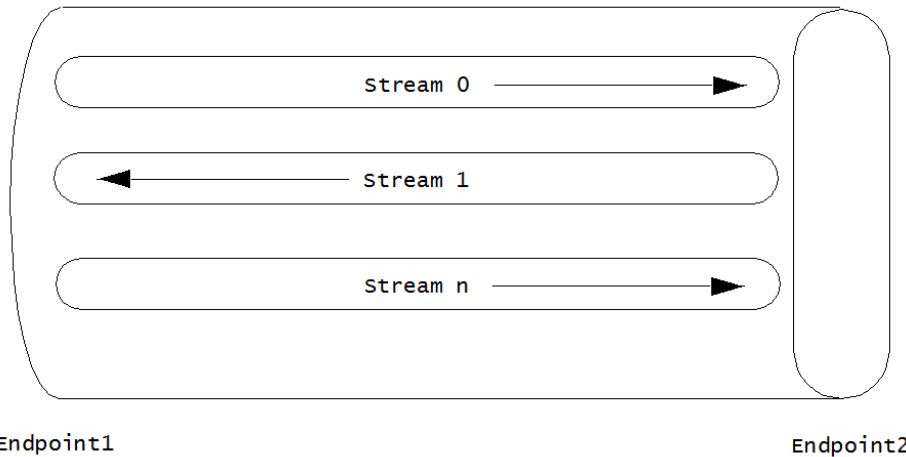


Congestion Control

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.

SIP transactions are mapped into SCTP streams as described in Section 5.1 of RFC 4168. In what it describes as the simplest way, the RFC suggests (keyword SHOULD) that all SIP messages be transmitted via Stream 0 with the U bit set to 1.

On the transmit side, the current SCTP implementation follows the RFC 4168 recommendation. On the receiving side, a SIP entity must be prepared to receive SIP messages over any stream.

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 with 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, and 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 CSM 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.

System Configuration

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.

Configuring SCTP Support for SIP

RFC 4168, *The Stream Control Transfer Protocol (SCTP) as a Transport for the Session Initiation Protocol (SIP)*, specifies the requirements for SCTP usage as a layer 4 transport for SIP. Use the following steps to:

- configure SCTP as the layer 4 transport for a SIP interface
- create an SCTP-based SIP port
- associate physical interfaces/network interfaces with SIP realms
- identify adjacent SIP servers that are accessible via SCTP
- set SCTP timers and counters (optional)

Configuring an SCTP SIP Port

SIP ports are created as part of the SIP Interface configuration process.

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

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

2. Use the address parameter to provide the IPv4 or IPv6 address of the network interface that supports the SIP port.

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

```
ORACLE(sip-port)# address 172.16.10.76
ORACLE(sip-port)#
```

3. Retain the default value, 5060 (the well-known SIP port) for the port parameter.

```
ORACLE(sip-port)# port 5060
ORACLE(sip-port)#
```

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

Supported values are UDP, TCP, TLS, and SCTP.

Select SCTP.

```
ORACLE(sip-port)# transport-protocol sctp
ORACLE(sip-port)#
```

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(sip-port)# multi-homed-addr 182.16.10.76
ORACLE(sip-port)#
ORACLE(sip-port)# multi-homed-addr (182.16.10.76 192.16.10.76
196.15.32.108)
ORACLE(sip-port)#
```

6. Remaining parameters can be safely ignored.
7. Use done, exit, and verify-config to complete configuration of this SCTP-based SIP port.

```

ORACLE(sip-port)# done
ORACLE(sip-interface)# exit
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 a SIP port which identifies primary and secondary multi-homed transport addresses, you identify the network interfaces that support the primary address and secondary addresses to the realm assigned during SIP Interface 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 (mol 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#

```

Configuring Session Agents

After configuring the realm, you identify adjacent SIP servers who will be accessed via the SCTP protocol.

1. From superuser mode, use the following command sequence to access session-agent configuration mode.

```

ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# session-agent
ORACLE(session-agent)#

```

2. Use the select command to access the target session-agent.
3. Use the transport-method parameter to select the layer 4 transport protocol.

Select staticSCTP for SCTP transport

```

ORACLE(session-agent)# transport-method staticSCTP
ORACLE(session-agent)#

```

4. Set the reuse-connections parameter to none.

Select staticSCTP for SCTP transport

```

ORACLE(session-agent)# reuse-connections none
ORACLE(session-agent)#

```

System Configuration

5. Use `done`, `exit`, and `verify-config` to complete session agent configuration.

```
ORACLE(session-agent)# done
ORACLE(session-router)# exit
ORACLE(configure)# exit
ORACLE# verify-config
-----
Verification successful! No errors nor warnings in the configuration
ORACLE#
```

6. Repeat Steps 1 through 5 as necessary to configure additional session agents who will be accessed via SCTP transport.

Setting SCTP Timers and Counters

Setting SCTP timers and counters is optional. All configurable timers and counters provide default values that conform 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.

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 itself 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#
```

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

System Configuration

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 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#
```

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)#
```

- Use the `sctp-max-burst` parameter to assign a value to the SCTP protocol parameter, `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)#
```

- 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 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.

- 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)#
```

- Use the `sctp-assoc-max-retrns` 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-retrns` defaults to 10 (transmission re-tries, the recommended default value from RFC 4960).

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

- 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#
```

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

System Configuration

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-retrns 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-retrns 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-retrns parameter is smaller than the sum of the sctp-path-max-retrns 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-retrns 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#
```

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 mode 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 #
```

Example Configurations

The following ACLI command sequences summarize required SCTP port configuration, and the configuration of required supporting elements.

- PHY interfaces
- Network interfaces
- SIP ports
- realms
- session agents

Sequences show only configuration parameters essential for SCTP operations; other parameters can retain default values, or assigned other values specific to local network requirements.

Phy Interface Configuration

The first ACLI command sequence configures a physical interface named m10, that will support an SCTP primary address; the second sequence configures an interface named m01 that will support a secondary SCTP address.

```
ORACLE# configure terminal
ORACLE (configure) # system
ORACLE (system) # phy-interface
ORACLE (phy-interface) # operation-type media
ORACLE (phy-interface) # port 0
ORACLE (phy-interface) # slot 1
ORACLE (phy-interface) # name m10
ORACLE (phy-interface) #
...
...
...
ORACLE (phy-interface) #
ORACLE# configure terminal
ORACLE (configure) # system
ORACLE (system) # phy-interface
ORACLE (phy-interface) # operation-type media
ORACLE (phy-interface) # port 1
ORACLE (phy-interface) # slot 0
ORACLE (phy-interface) # name m01
ORACLE (phy-interface) #
...
...
...
ORACLE (phy-interface) #
```

Network Interface Configuration

These ACLI command sequences configure two network interfaces. The first sequence configures a network interface named m10, thus associating the network interface with the physical interface of the same name. The ACLI ip-address command assigns the IPv4 address 172.16.10.76 to the network interface. In a similar fashion, the second command sequence associates the m01 network and physical interfaces, and assigns an IPv4 address of 182.16.10.76.

```
ORACLE# configure terminal
ORACLE (configure) # system
ORACLE (system) # network-interface
ORACLE (network-interface) # name m10
ORACLE (network-interface) # ip-address 172.16.10.76
...
...
```

System Configuration

```
...
ORACLE(network-interface)#
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)# network-interface
ORACLE(network-interface)# name m01
ORACLE(network-interface)# ip-address 182.16.10.76
...
...
...
ORACLE(network-interface)#
```

SIP Port Configuration

This CLI command sequence configures a SIP port for SCTP operations. It specifies the use of SCTP as the transport layer protocol, and assigns the existing network interface address, 172.16.10.76, as the SCTP primary address. Additionally, it identifies three other existing network addresses (182.16.10.76, 192.16.10.76, and 196.15.32.108) as SCTP secondary addresses.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-interface
ORACLE(sip-interface)# sip-ports
ORACLE(sip-port)# address 172.16.10.76
ORACLE(sip-port)# transport-protocol sctp
ORACLE(sip-port)# multi-homed-addr (182.16.10.76 192.16.10.76 196.15.32.108)
...
...
...
ORACLE(sip-port)#
```

Realm Configuration

These CLI command sequences configure a realm for SCTP operations. The first CLI sequence assigns a named realm, in this example core-172, to a SIP interface during the interface configuration process. The second sequence accesses the target realm and uses the network-interfaces command to associate the named SCTP network interfaces with the realm.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-interface
ORACLE(sip-interface)# realm-id core-172
...
...
...
ORACLE(sip-interface)#
ORACLE# configure terminal
ORACLE(configure)# media-manager
ORACLE(media-manager)# realm-config
ORACLE(realm-config)# select
identifier: core-172
1. core-172 ...
selection: 1
ORACLE(realm-config)# network-interfaces (m01 m10 ...)
...
...
...
ORACLE(realm-config)#
```

Session Agent Configuration

The final CLI command sequence enables an SCTP-based transport connection between the Acme Packet 4500 and an adjacent network element.

```

ORACLE# configure terminal
ORACLE (configure) # session-router
ORACLE (session-router) # session-agent
ORACLE (session-agent) # select
<hostname>: core-172S1
1. core-172S1 ...
selection: 1
ORACLE (session-agent) #
ORACLE (session-agent) # transport-method staticSCTP
ORACLE (session-agent) # reuse-connections none
...
...
...
ORACLE (session-agent) #


```

IPv6 Address Configuration

IPv6 can be a licensed feature on the Oracle USM. IPv6 is available on the Oracle CSM without an explicit license or entitlement.

If you want to add this license to a system, contact Oracle. Once you have the license, refer to the Getting Started chapter for instructions about how to add a license.

You do not need to take action if you are working with a new system with which the IPv6 license was purchased.

 **Note:** For ACLI parameters that support only IPv4, there are many references to that version as the accepted value for a configuration parameter or other IPv4-specific languages. For IPv6 support, these references have been edited. For example, rather than providing help that refers specifically to IPv4 addresses when explaining what values are accepted in an ACLI configuration parameter, you will now see an <ipAddr> note.

This section calls out the configurations and parameters for which you can enter IPv6 addresses. In this first IPv6 implementation, the complete range of system configurations and their parameters are available for IPv6 use.

The Oracle CSM follows RFC 3513 its definition of IPv6 address representations. Quoting from that RFC, these are the two forms supported:

- The preferred form is x:x:x:x:x:x:x, where the 'x's are the hexadecimal values of the eight 16-bit pieces of the address. Examples:

```
FEDC:BA98:7654:3210:FEDC:BA98:7654:3210
```

```
1080:0:0:0:8:800:200C:417A
```

Note that it is not necessary to write the leading zeros in an individual field, but there must be at least one numeral in every field (except for the case described in 2.).

- Due to some methods of allocating certain styles of IPv6 addresses, it will be common for addresses to contain long strings of zero bits. In order to make writing addresses containing zero bits easier a special syntax is available to compress the zeros. The use of ":" indicates one or more groups of 16 bits of zeros. The "::" can only appear once in an address. The "::" can also be used to compress leading or trailing zeros in an address. For example, the following addresses: 1080:0:0:0:8:800:200C:417A a unicast address FF01:0:0:0:0:0:101 a multicast address

```
0:0:0:0:0:0:1 the loopback address
```

```
0:0:0:0:0:0:0 the unspecified addresses
```

may be represented as:

```
1080::8:800:200C:417A a unicast address
```

```
FF01::101 a multicast address
```

```
::1 the loopback address
```

```
:: the unspecified addresses
```

System Configuration

Access Control

These are the IPv6-enabled parameters in the access-control configuration.

Parameter	Entry Format
source-address	<ip-address>[/<num-bits>][:<port>[/<port-bits>]]
destination-address	<ip-address>[/<num-bits>][:<port>[/<port-bits>]]

Host Route

These are the IPv6-enabled parameters in the host-route configuration.

Parameter	Entry Format
dest-network	<ipv4> <ipv6>
netmask	<ipv4> <ipv6>
gateway	<ipv4> <ipv6>

Local Policy

These are the IPv6-enabled parameters in the local-policy configuration.

Parameter	Entry Format
from-address	<ipv4> <ipv6> POTS Number, E.164 Number, hostname, wildcard
to-address	<ipv4> <ipv6> POTS Number, E.164 Number, hostname, wildcard

Network Interface

These are the IPv6-enabled parameters in the network-interface configuration.

Parameter	Entry Format
hostname	<ipv4> <ipv6> hostname
ip-address	<ipv4> <ipv6>
pri-utility-addr	<ipv4> <ipv6>
sec-utility-addr	<ipv4> <ipv6>
netmask	<ipv4> <ipv6>
gateway	<ipv4> <ipv6>
sec-gateway	<ipv4> <ipv6>
dns-ip-primary	<ipv4> <ipv6>
dns-ip-backup1	<ipv4> <ipv6>
dns-ip-backup2	<ipv4> <ipv6>
add-hip-ip	<ipv4> <ipv6>
remove-hip-ip	<ipv4> <ipv6>
add-icmp-ip	<ipv4> <ipv6>
remove-icmp-ip	<ipv4> <ipv6>

ENUM Server

These are the IPv6-enabled parameters in the enum-config.

Parameter	Entry Format
enum-servers	[<ipv4> <ipv6>]:port

Realm Configuration

These are the IPv6-enabled parameters in the realm-config.

Parameter	Entry Format
addr-prefix	[<ipv4> <ipv6>]/prefix

Session Agent

These are the IPv6-enabled parameters in the session-agent configuration.

Parameter	Entry Format
hostname	<ipv4> <ipv6>
ip-address	<ipv4> <ipv6>

SIP Configuration

These are the IPv6-enabled parameters in the session-config.

Parameter	Entry Format
registrar-host	<ipv4> <ipv6> hostname *

SIP Interface SIP Ports

These are the IPv6-enabled parameters in the sip-interface>sip-ports configuration.

Parameter	Entry Format
address	<ipv4> <ipv6>

Steering Pool

These are the IPv6-enabled parameters in the steering-pool configuration.

Parameter	Entry Format
ip-address	<ipv4> <ipv6>

System Configuration

These are the IPv6-enabled parameters in the system-config.

Parameter	Entry Format
default-v6-gateway	<ipv6>

IPv6 Support for Management and Telemetry

Several management-oriented parameters on the Oracle CSM may be configured with IPv6 addresses to be used within IPv6 networks.

The following parameters that are configured with IP addresses accept IPv6 addresses to be used within IPv6 address space.

You may configure the wancom0/eth0 physical interface in the bootparams with an IPv6 address and complementary IPv6 gateway via the following parameters:

- **bootparams > inet on ethernet**
- **bootparams > gateway inet**

You may configure a syslog server with an IPv6 destination address via the following parameter:

- **system > system-config > syslog-servers > address**

You may configure a system access list entry with an IPv6 source address and complementary IPv6 gateway.

- **system > system-access-list > source-address**
- **system > system-access-list > netmask**

You may configure a RADIUS server with an IPv6 destination address via the following parameter:

- **security > authentication > radius-servers > address**

IPv6 Default Gateway

In the system configuration, you configure a default gateway—a parameter that now has its own IPv6 equivalent.

To configure an IPv6 default gateway:

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

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

2. Type `system` and press Enter.

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

3. Type `system-config` and press Enter.

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

4. `default-v6-gateway`—Set the IPv6 default gateway for this Oracle CSM. This is the IPv6 egress gateway for traffic without an explicit destination. The application of your Oracle CSM determines the configuration of this parameter.
5. Save your work.

IPv6 Link Local Addresses

The Oracle CSM supports IPv6 Link Local addresses configured for a network interface's gateway.

An IPv6 link local address is signified by its first hexet set to FE80:. Even if a network interface's first hexet is not FE80, but the gateway is, the Oracle CSM will still function as expected.

show neighbor-table

The `show neighbor-table` command displays the IPv6 neighbor table and validates that there is an entry for the link local address, and the gateway uses that MAC address.

```

System# show neighbor-table
LINK LEVEL NEIGHBOR TABLE
Neighbor                               Linklayer Address  Netif Expire      S
Flags
300::100                               0:8:25:a1:ab:43   sp0 permanent ? R
871962224
400::100                               0:8:25:a1:ab:45   sp1 permanent ? R
871962516
fe80::bc02:a98f:f61e:20%sp0          be:2:ac:1e:0:20   sp0 4s            ? R
871962808
fe80::bc01:a98f:f61e:20%sp1          be:1:ac:1e:0:20   sp1 4s            ? R
871963100
-----
ICMPv6 Neighbor Table:
-----
entry: slot port vlan IP                               type      flag
pendBlk Hit MAC
-----
  5   : 1   0   0   fe80::bc01:a98f:f61e:20/64   08-DYNAMIC 1
0     1   be:01:ac:1e:00:20
  4   : 1   0   0   0.0.0.0/64                 01-GATEWAY 0
0     1   be:01:ac:1e:00:20
  3   : 1   0   0   400::/64                   02-NETWORK 0
0     1   00:00:00:00:00:00
  2   : 0   0   0   fe80::bc02:a98f:f61e:20/64   08-DYNAMIC 1
0     1   be:02:ac:1e:00:20
  1   : 0   0   0   0.0.0.0/64                 01-GATEWAY 0
0     1   be:02:ac:1e:00:20
  0   : 0   0   0   300::/64                   02-NETWORK 0
0     1   00:00:00:00:00:00
-----
-----

```

Network Interfaces and IPv6

You set many IP addresses in the network interface, one of which is the specific IP address for that network interface and others that are related to different types of management traffic. This section outlines rules you must follow for these entries.

- For the network-interface ip-address parameter, you can set a single IP address. When you are working with an IPv6-enabled system, however, note that all other addresses related to that network-interface IP address must be of the same version.
- Heterogeneous address family configuration is prevented for the dns-ip-primary, dns-ip-backup1, and dns-ip-backup2 parameters.
- For HIP addresses (add-hip-ip), you can use either IPv4 or IPv6 entries.
- For ICMP addresses (add-icmp-ip), you can use either IPv4 or IPv6 entries.
- For Telnet (add-telnet-ip), FTP (add-ftp-ip), and SNMP (add-snmp-ip), you are not allowed to use IPv6; your entries MUST use IPv4.

IPv6 Reassembly and Fragmentation Support

As it does for IPv4, the Oracle CSM supports reassembly and fragmentation for large signaling packets when you enable IPV6 on your system.

The Oracle CSM takes incoming fragments and stores them until it receives the first fragment containing a Layer 4 header. With that header information, the Oracle CSM performs a look-up so it can forward the packets to its

System Configuration

application layer. Then the packets are re-assembled at the applications layer. Media fragments, however, are not reassembled and are instead forwarded to the egress interface.

On the egress side, the Oracle CSM takes large signaling messages and encodes it into fragment datagrams before it transmits them.

Note that large SIP INVITE messages should be sent over TCP. If you want to modify that behavior, you can use the SIP interface's option parameter `max-udp-length=xx` for each SIP interface where you expect to receive large INVITE packets.

Other than enabling IPv6 on your Oracle CSM, there is no configuration for IPv6 reassembly and fragmentation support. It is enabled automatically.

Access Control List Support

The Oracle CSM supports IPv6 for access control lists in two ways:

- For static access control lists that you configure in the access-control configuration, your entries can follow IPv6 form. Further, this configuration supports a prefix that enables wildcarding the source IP address.
- Dynamic ACLs are also supported; the Oracle CSM will create ACLs for offending IPv6 endpoints.

Data Entry

When you set the source-address and destination-address parameters in the access-control configuration, you will use a slightly different format for IPv6 than for IPv4.

For the source-address, your IPv4 entry takes the following format: `<ip-address>[/<num-bits>][:<port>[/<port-bits>]]`. And for the destination-address, your IPv4 entry takes this format: `<ip-address>[:<port>[/<port-bits>]]`.

Since the colon (:) in the IPv4 format leads to ambiguity in IPv6, your IPv6 entries for these settings must have the address encased in brackets ([]): `[7777::11]/64:5000/14`.

In addition, IPv6 entries are allowed up to 128 bits for their prefix lengths.

The following is an example access control configuration set up with IPv6 addresses.

```
ORACLE (access-control) # done
access-control
    realm-id                net7777
    description
    source-address          7777::11/64:5060/8
    destination-address     8888::11:5060/8
    application-protocol    SIP
    transport-protocol      ALL
    access                  deny
    average-rate-limit      0
    trust-level             none
    minimum-reserved-bandwidth 0
    invalid-signal-threshold 10
    maximum-signal-threshold 0
    untrusted-signal-threshold 0
    deny-period             30
```

DNS Support

The Oracle CSM supports the DNS resolution of IPv6 addresses; in other words, it can request the AAAA record type (per RFC 1886) in DNS requests. In addition, the Oracle CSM can make DNS requests over IPv6 transport so that it can operate in networks that host IPv6 DNS servers.

For mixed IPv4-IPv6 networks, the Oracle CSM follows these rules:

- If the realm associated with the name resolution is an IPv6 realm, the Oracle CSM will send the query out using the AAAA record type.
- If the realm associated with the name resolution is an IPv4 realm, the Oracle CSM will send the query out using the A record type.

In addition, heterogeneous address family configuration is prevented for the `dns-ip-primary`, `dns-ip-backup1`, and `dns-ip-backup2` parameters.

Homogeneous Realms

IPv6 is supported for realms and for nested realms, as long as the parent chain remains within the same address family. If you try to configure realms with mixed IPv4-IPv6 addressing, your system will issue an error message when you try to save your configuration. This check saves you time because you do not have to wait to run a configuration verification (using the CLI `verify-config` command) to find possible errors.

Parent-Child Network Interface Mismatch

Your system will issue the following error message if parent-child realms are on different network interfaces that belong to different address families:

```
ERROR: realm-config [child] and parent [net8888] are on network interfaces
that belong to different address families
```

Address Prefix-Network Interface Mismatch

If the address family and the address-prefix you configure for the realm does not match the address family of its network interface, your system will issue the following error message:

```
ERROR: realm-config [child] address prefix and network interface [1:1:0]
belong to different address families
```

RADIUS Support for IPv6

The Oracle CSM's RADIUS support now includes:

- RADIUS CDR generation for SIPv6-SIPv6 and SIPv6-SIPv4 calls
- IPv6-based addresses in RADIUS CDR attributes

This means that for the CDR attributes in existence prior to the introduction of IPv6 to the Acme Packet 4500 are mapped to the type `ipaddr`, which indicates four-byte field. The sixteen-byte requirement for IPv6 addresses is now supported, and there are a parallel set of attributes with the type `ipv6addr`. Attributes 155-170 are reserved for the IPv6 addresses.

NAS addresses use the number 95 to specify the `NAS-IPV6-Address` attribute. And local CDRs now contain IPv6 addresses.

Supporting RADIUS VSAs

The following VSAs have been added to the Oracle RADIUS dictionary to support IPv6.

<code>Acme-Flow-In-Src-IPv6_Addr_FS1_F</code>	155	<code>ipv6addr</code>	Acme
<code>Acme-Flow-In-Dst-IPv6_Addr_FS1_F</code>	156	<code>ipv6addr</code>	Acme
<code>Acme-Flow-Out-Src-IPv6_Addr_FS1_F</code>	157	<code>ipv6addr</code>	Acme
<code>Acme-Flow-Out-Dst-IPv6_Addr_FS1_F</code>	158	<code>ipv6addr</code>	Acme
<code>Acme-Flow-In-Src-IPv6_Addr_FS1_R</code>	159	<code>ipv6addr</code>	Acme
<code>Acme-Flow-In-Dst-IPv6_Addr_FS1_R</code>	160	<code>ipv6addr</code>	Acme
<code>Acme-Flow-Out-Src-IPv6_Addr_FS1_R</code>	161	<code>ipv6addr</code>	Acme
<code>Acme-Flow-Out-Dst-IPv6_Addr_FS1_R</code>	162	<code>ipv6addr</code>	Acme
<code>Acme-Flow-In-Src-IPv6_Addr_FS2_F</code>	163	<code>ipv6addr</code>	Acme
<code>Acme-Flow-In-Dst-IPv6_Addr_FS2_F</code>	164	<code>ipv6addr</code>	Acme
<code>Acme-Flow-Out-Src-IPv6_Addr_FS2_F</code>	165	<code>ipv6addr</code>	Acme
<code>Acme-Flow-Out-Dst-IPv6_Addr_FS2_F</code>	166	<code>ipv6addr</code>	Acme

System Configuration

Acme-Flow-In-Src-IPv6_Addr_FS2_R	167	ipv6addr	Acme
Acme-Flow-In-Dst-IPv6_Addr_FS2_R	168	ipv6addr	Acme
Acme-Flow-Out-Src-IPv6_Addr_FS2_R	169	ipv6addr	Acme
Acme-Flow-Out-Dst-IPv6_Addr_FS2_R	170	ipv6addr	Acme

Realms and Nested Realms

This chapter explains how to configure realms and nested realms features.

A realm is a logical definition of a network or groups of networks made up in part by devices that provide real-time communication sessions comprised of signaling messages. These network devices might be call agents, softswitches, SIP proxies, IP PBXs, etc., that are statically defined by IPv4 addresses. These network devices might also be IPv4 endpoints: SIP phones, IADs, MAs, media gateways, etc., that are defined by an IPv4 address prefix.

Realms are the basis for defining egress and ingress traffic to the Oracle CSM—which supports the Oracle CSM's topology hiding capabilities.

Overview

Realms are a logical distinction representing routes (or groups of routes) reachable by the Oracle CSM and what kinds of resources and special functions apply to those routes. Realms are used as a basis for determining ingress and egress associations to network interfaces, which can reside in different VPNs. The ingress realm is determined by the signaling interface on which traffic arrives. The egress realm is determined by the following:

- Routing policy—Where the egress realm is determined in the session agent configuration or external address of a SIP-NAT
- Realm-bridging—As applied in the SIP-NAT configuration configurations
- Third-party routing/redirect (i.e., SIP redirect)

Realms also provide configuration support for denial of service (DoS)/access control list (ACL) functionality.

Realms can also be nested in order to form nested realm groups. Nested realms consist of separate realms that are arranged within a hierarchy to support network architectures that have separate backbone networks and VPNs for signaling and media. This chapter provides detailed information about nested realms after showing you how to configure realms on your Oracle CSM.

About Realms and Network Interfaces

All realms reference network interfaces on the Oracle CSM. This reference is made when you configure a list of network interfaces in the realm configuration.

You configure a network interface to specify logical network interfaces that correspond existing physical interfaces on the Oracle CSM. Configuring multiple network interfaces on a single physical interface creates a channelized physical interface, a VLAN. VLANs, in turn, allow you to reuse address space, segment traffic, and maximize bandwidth.

Realms and Nested Realms

In order to reach the realms you configure, you need to assign them network interfaces. The values you set for the name and port in the network interface you select then indicate where the realm can be reached.

About the SIP Home Realm

The realm configuration is also used to establish what is referred to as the SIP home realm. This is the realm where the Oracle CSM's SIP proxy sits.

In peering configurations, the SIP home realm is the internal network of the SIP proxy. In backbone access configurations, the SIP home realm typically interfaces with the backbone connected network. In additions, the SIP home realm is usually exposed to the Internet in an HNT configuration.

Although you configure a SIP home realm in the realm configuration, it is specified as the home realm in the main SIP configuration by the home realm identifier parameter. Specifying the SIP home realm means that the Oracle CSM's SIP proxy can be addressed directly by connected entities, but other connected network signaling receives layer 3 NAT treatment before reaching the internal SIP proxy.

About Realms and Other Oracle CSM Functions

Realms are referenced by other configurations in order to support this functionality across the protocols the Oracle CSM supports and to make routing decisions. Other configurations' parameters that point to realms are:

- SIP configuration: home realm identifier, egress realm identifier
- SIP-NAT configuration: realm identifier
- MGCP configuration: private realm, public realm
- Session agent configuration: realm identifier
- Media manager: home realm identifier
- Steering ports: realm identifier
- Static flow: in realm identifier, out realm identifier

Realms

Realm configuration is divided into the following functional areas, and the steps for configuring each are set out in this chapter: identity and IP address prefix, realm interfaces, realm service profiles, QoS measurement, QoS marking, address translation profiles, and DNS server configuration.

Before You Configure

Before you configure realms, you want to establish the physical and network interfaces with which the realm will be associated.

- Configure a physical interface to define the physical characteristics of the signaling line.
- Configure a network interface to define the network in which this realm is participating and optionally to create VLANs.

If you wish to use QoS, you should also determine if your Oracle CSM is QoS enabled.

Remember that you will also use this realm in other configurations to accomplish the following:

- Set a signaling port or ports at which the Oracle CSM listens for signaling messages.
- Configure sessions agents to point to ingress and egress signaling devices located in this realm in order to apply constraint for admission control.
- Configure session agents for defining trusted sources for accepting signaling messages.

Realm Configuration

To access the realm configuration parameters in the ACLI:

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


```
ORACLE# configure terminal
```

2. Type `media-manager` and press Enter to access the media-related configurations.

```
ORACLE (configure) # media-manager
```

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

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

From this point, you can configure realm parameters. To view all realm configuration parameters, enter a `?` at the system prompt.

Identity and IP Address Prefix

The first parameters you configure for a realm are its name (a unique identifier) and an IP address prefix and subnet mask.

The IP address and subnet mask establish a set of matching criteria for the realm, and distinguishes between realms that you assign to the same network interface.

To configure a realm's identity and IP address prefix in the ACLI:


1. `identifier`—Enter the name of the realm. This parameter uniquely identifies the realm. You will use this parameter in other configurations when asked for a realm identifier value.
2. `addr-prefix`—Enter the IPv4 or IPv6 address and subnet mask combination to set the criteria the Oracle CSM uses to match packets sent or received on the network interface associated with this realm. This matching determines the realm, and subsequently what resources are used for that traffic. This setting determines whether the realm is an IPv4 or IPv6 realm.

This parameter must be entered in the correct format where the IPv4 or IPv6 address comes first and is separated by a slash (/) from the subnet mask value. For example, `172.16.0.0/24`.

The default for this parameter is `0.0.0.0`. When you leave this parameter set to the default, all addresses match.

Realm Interfaces

The realm points to one network interface on the Oracle CSM.

 **Note:** Only one network-interface can be assigned to a single `realm-config` object, except for Local multi-homing SCTP deployments.

To assign interfaces to a realm:

`network-interfaces`—Enter the physical and network interface(s) that you want this realm to reference. These are the network interfaces through which this realm can be reached by ingress traffic, and through which this traffic exits the system as egress traffic.

Enter the name and port in the correct format where the name of the interface comes first and is separated by a colon (:) from the port number. For example, `f10:0`.

The parameters you set for the network interfaces must be unique.

Enter multiple network interfaces for this list by typing an open parenthesis, entering each field value separated by a Space, typing a closed parenthesis, and then pressing Enter.

```
ORACLE (realm-config) # network-interfaces f1:0
```

You must explicitly configure a realm's network interface as either IPv4 or IPv6 when the applicable interface is either dual-stack or IPv6. You do this by appending the realm's network-interface with a `.4` or a `.6`, as shown below.

```
ORACLE (realm-config) # network-interfaces f1:0.6
```

For single-stack interface configurations that do not specify this format, the Oracle CSM assumes an IPv4 interface. Dual stack interface configurations fail if this IP version family suffix is not specified.

Address Translation Profiles

If you are not using this feature, you can leave the `in-translationid` and `out-translationid` parameters blank.

DNS Servers

You can configure DNS functionality on a per-network-interface basis, or you can configure DNS servers to use per realm. Configuring DNS servers for your realms means that you can have multiple DNS servers in connected networks. In addition, this allows you to specify which DNS server to use for a given realm such that the DNS might actually be in a different realm with a different network interface.

This feature is available for SIP and MGCP only.

To configure realm-specific DNS in the ACLI:

`dns-realm`—Enter the name of the network interface that is configured for the DNS service you want to apply in this realm. If you do not configure this parameter, then the realm will use the DNS information configured in its associated network interface.

DoS ACL Configuration

If you are not using this functionality, you can leave the parameters at their default values: `average-rate-limit`, `peak-rate-limit`, `maximum-burst-size`, `access-control-trust-level`, `invalid-signal-threshold`, and `maximum-signal-threshold`.

Enabling RTP-RTCP UDP Checksum Generation

You can configure the Oracle CSM to generate a UDP checksum for RTP/ RTCP packets on a per-realm basis. This feature is useful in cases where devices performing network address translation (NAT) do not pass through packets with a zero checksum from the public Internet. These packets do not make it through the NAT, even if they have the correct to and from IP address and UDP port information. When you enable this feature, the Oracle CSM calculates a checksum for these packets and thereby enables them to traverse a NAT successfully.

If you do not enable this feature, then the Oracle CSM will not generate a checksum for RTP or RTCP packets if their originator did not include one. If a checksum is already present when the traffic arrives at the hardware, the system will relay it.

You enable this feature on the outbound realm.

Aggregate Session Constraints Per Realm

You can set session constraints for the Oracle CSM's global SIP configuration, specified session agents, and specified SIP interfaces. This forces users who have a large group of remote agents to create a large number of session agents and SIP interfaces.

With this feature implemented, however, you can group remote agents into one or more realms on which to apply session constraints.

To enable sessions constraints on a per realm basis:

`constraint-name`—Enter the name of the constraint you want to use for this realm. You set up in the `session-constraints` configuration.

UDP Checksum Generation Configuration

To enable UDP checksum generation for a realm:

`generate-udp-checksum`—Enable this parameter to generate a UDP checksum for this outbound realm. The default is disabled. Valid values are:

- `enabled` | `disabled`

Admission Control Configuration

You can set admission control based on bandwidth for each realm by setting the max-bandwidth parameter for the realm configuration. Details about admission control are covered in this guide’s *Admission Control and QoS* chapter.

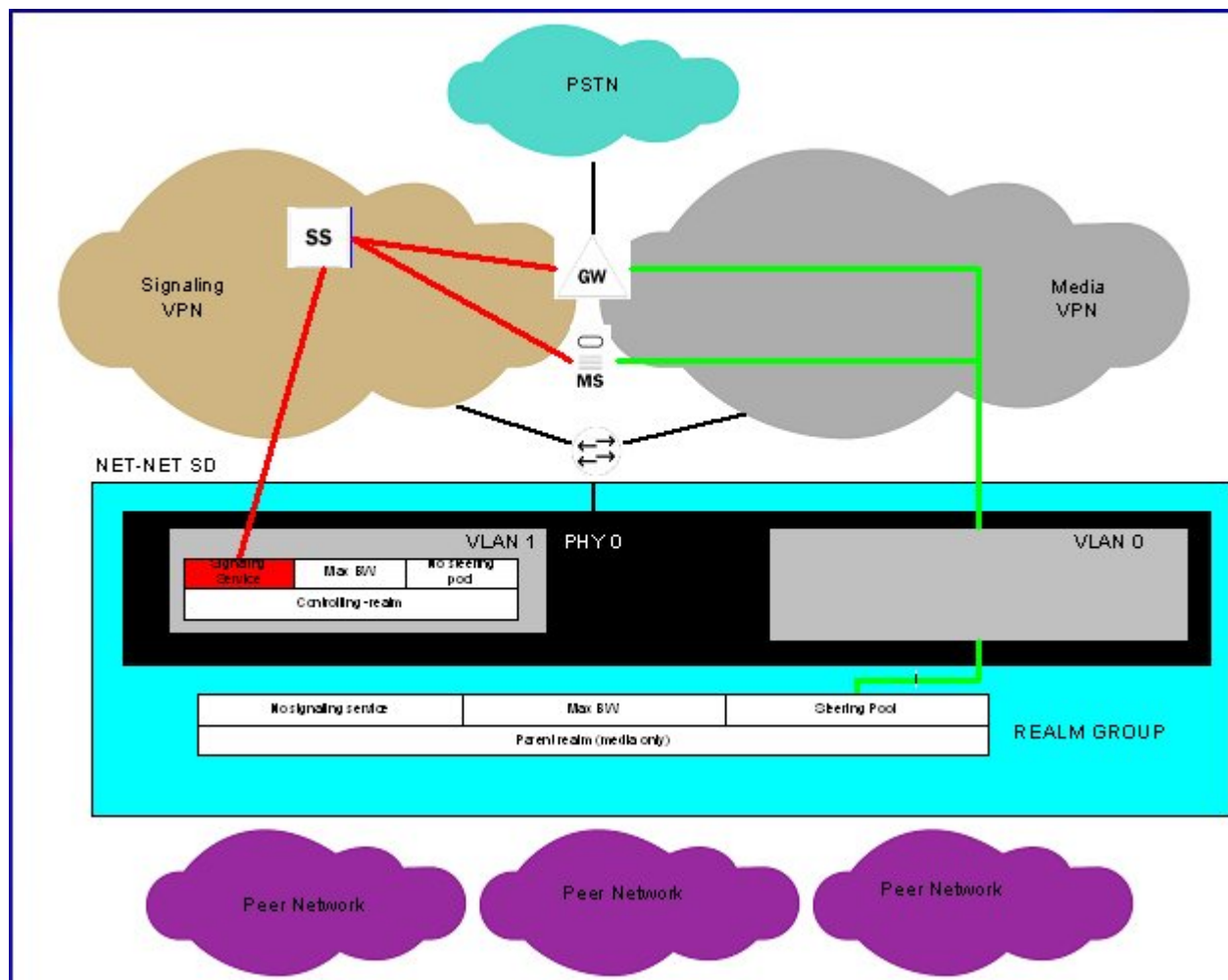
Reserved Parameters

In the ACLI, you do not need to configure the following parameters: max-latency, max-jitter, max-packet-loss, and observ-window-size.

Nested Realms

Configuring nested realms allows you to create backbone VPN separation for signaling and media. This means that you can put signaling and media on separate network interfaces, that the signaling and media VPN can have different address spaces, and that the parent realm has one media-only sub-realm.

The following figure shows the network architecture.



In addition, you can achieve enhanced scalability by using a shared service interface. A single service address is shared across many customers/peers, customer specific policies for bandwidth use and access control are preserved, and you can achieve fine-grained policy control.

These benefits are achieved when you configure these types of realms:

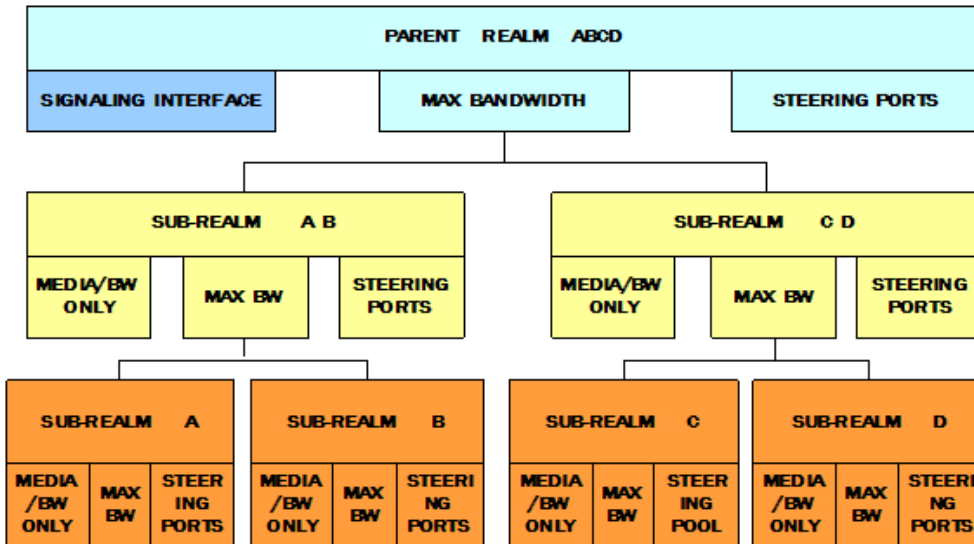
- Realm group—A hierarchical nesting of realms identified by the name of the highest order realm.

Realms and Nested Realms

- Controlling realm—A realms for which a signaling interface is configured. For example, you might configure these signaling interfaces in the following configurations: SIP-NAT or SIP port. Typically, this is the highest order realm for the parent realm in a realm group.
- Parent realm—A realm that has one or more child realms. A parent realm might also be the child realm of another realm group.
- Child realm—A realm that is associated with a single higher order parent realm. A child might also be the parent realm of another realm group. Child realms inherit all signaling and steering ports from higher order realms.
- Media-only realm—A realm for which there is no configured signaling interface directly associated. Media-only realms are nested within higher order realms.

As these definitions suggest, parent and child realms can be constructed so that there are multiple nesting levels. Lower order realms inherit the traits of the realms above them, including: signaling service interfaces, session translation tables, and steering pools.

Since realms inherit the traits of the realms above them in the hierarchy, you will probably want to map what realms should be parents and children before you start configuring them. These relationships are constructed through one parameter in the realm configuration that identifies the parent realm for the configuration. If you specify a parent realm, then the realm you are configuring becomes a child realm subject to the configured parameters you have established for that parent. And since parent realms can themselves be children of other realm, it is important that you construct these relationships with care.



Configuring Nested Realms

When you are configuring nested realms, you can separate signaling and media by setting realm parameters in the SIP interface configuration and the steering ports configuration.

- The realm identifier you set in the SIP interface configuration labels the associated realm for signaling.
- The realm identifier you set in the steering ports configuration labels the associated realm for media.

For MGCP, you set a special option that enables nested realm use.

Constructing a hierarchy of nested realms requires that you note which realms you want to handle signaling, and which you want to handle media.

In the SIP port configuration for the SIP interface, you will find an allow anonymous parameter that allows you to set certain access control measures. The table below outlines what each parameter means.

Allow Anonymous Parameter	Description
all	All anonymous connections allowed.
agents-only	Connections only allowed from configured session agents.

Allow Anonymous Parameter	Description
realm-prefix	Connections only allowed from addresses with the realm's address prefix and configured session agents.
registered	Connections allowed only from session agents and registered endpoints. (For SIP only, a REGISTER is allowed for any endpoint.)
register-prefix	Connections allowed only from session agent and registered endpoints. (For SIP only, a REGISTER is allowed for session agents and a matching realm prefix.)

Parent and Child Realm Configuration

To configure nested realms, you need to set parameters in the realm configuration.

To configure parent and child realms:

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

```
ORACLE# configure terminal
```

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

```
ORACLE (configure) # media-manager
```

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

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

4. parent-realm—Enter the identifier of the realm you want to name as the parent. Configuring this parameter makes the realm you are currently configuring as the child of the parent you name. As such, the child realm is subject to the configured parameters for the parent.

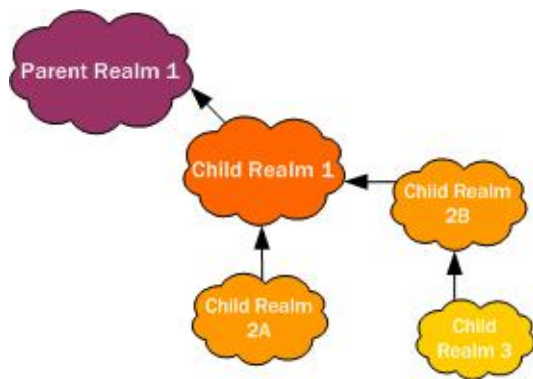
Aggregate Session Constraints Nested Realms

In addition to setting session constraints per realm for SIP sessions, you can also enable the Oracle CSM to apply session constraints across nested realms. When you set up session constraints for a realm, those constraints apply only to the realm for which they are configured without consideration for its relationship either as parent or child to any other realms.

You can also, however, enable the Oracle CSM to take nested realms into consideration when applying constraints. For example, if a call enters on a realm that has no constraints but its parent does, then the constraints for the parent are applied. This parameter is global and so applies to all realms on the system. For the specific realm the call uses and for all of its parents, the Oracle CSM increments the counters upon successful completion of an inbound or outbound call.

In the following example, you can see one parent realm and its multiple nested, child realms. Now consider applying these realm constraints:

- Parent Realm 1—55 active sessions
- Child Realm 1—45 active sessions
- Child Realm 2A—30 active sessions
- Child Realm 2B—90 active sessions
- Child Realm 3—20 active sessions



Given the realm constraints outlined above, consider these examples of how global session constraints for realms. For example, a call enters the Oracle CSM on Child Realm 2B, which has an unmet 90-session constraint set. Therefore, the Oracle CSM allows the call based on Child Realm 2B. But the call also has to be within the constraints set for Child Realm 1 and Parent Realm 1. If the call fails to fall within the constraints for either of these two realms, then the Oracle CSM rejects the call.

Impact to Other Session Constraints and Emergency Calls

You can set up session constraints in different places in your Oracle CSM configuration. Since session agents and SIP interfaces also take session constraints, it is important to remember the order in which the Oracle CSM applies them:

1. Session agent session constraints
2. Realm session constraints (including parent realms)
3. SIP interface session constraints

Emergency and priority calls for each of these is exempt from session constraints. That is, any call coming into the Oracle CSM marked priority is processed.

Session Constraints Configuration

You enabled use of session constraints for nested realms across the entire system by setting the `nested-realms-stats` parameter in the session router configuration to `enabled`.

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

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

2. Type `session-router` and press Enter.

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

3. Type `session-router` and press Enter.

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

4. `nested-realms-stats`—Change this parameter from `disabled` (default) to `enabled` if you want the Oracle CSM to apply session constraints across all nested realms (realms that are children to other realms)
5. Save and activate your configuration.

Realm-Based Packet Marking

The Oracle CSM supports TOS/DiffServ functions that allow you to

- Set up realm-based packet marking by media type, either audio-voice or video
- Set up realm-based packet marking for signaling

Upstream devices use these markings to classify traffic in order to determine the priority level of treatment it will receive.

About TOS DiffServ

TOS and DiffServ are two different mechanisms used to achieve QoS in enterprise and service provider networks; they are two different ways of marking traffic to indicate its priority to upstream devices in the network.

For more information about TOS (packet) marking, refer to:

- IETF RFC 1349 (<http://www.ietf.org/rfc/rfc1349.txt>)

For more information about DiffServ, refer to:

- IETF RFC 2474 (<http://www.ietf.org/rfc/rfc2474.txt>)
- IETF RFC 2475 (<http://www.ietf.org/rfc/rfc2475.txt>).

ToS Byte

The TOS byte format is as follows:



The TOS byte is broken down into three components:

- **Precedence**—The most used component of the TOS byte, the precedence component is defined by three bits. There are eight possible precedence values ranging from 000 (decimal 0) through 111 (decimal 7). Generally, a precedence value of 000 refers to the lowest priority traffic, and a precedence value of 111 refers to the highest priority traffic.
- **TOS**—The TOS component is defined by four bits, although these bits are rarely used.
- **MBZ**—The must be zero (MBZ) component of the TOS byte is never used.

DiffServ Byte

Given that the TOS byte was rarely used, the IETF redefined it and in doing so created the DiffServ byte.

The DiffServ byte format is as follows:



The DiffServ codepoint value is six bits long, compared to the three-bit-long TOS byte's precedence component. Given the increased bit length, DiffServ codepoints can range from 000000 (decimal 0) to 111111 (decimal 63).



Note: By default, DiffServ codepoint mappings map exactly to the precedence component priorities of the original TOS byte specification.

Signaling Packet Marking Configuration

ToS marking for signaling requires you to configure a media policy and set the name of the media policy in the appropriate realm configuration.

This section shows you how to configure packet marking for signaling.

Configuring a Media Policy for Signaling Packet Marking

To set up a media policy configuration to mark audio-voice or video packets:

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

```
ORACLE# configure terminal
```

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

Realms and Nested Realms

```
ORACLE (configure) # media-manager
```

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

```
ORACLE (media-manager) # media-policy
ORACLE (media-policy) #
```

From this point, you can configure media policy parameters. To view all media policy configuration parameters, enter a `?` at the system prompt.

4. `name`—Enter the unique name of this media policy. When you set up the class policy, this is the value you set in the `media-policy` parameter.
5. `tos-values`—Enter the list of TOS values for media types for this media policy; by default, this list is empty. These values provide a policing profile.

Using this list, you can specify one or more audio media types, one or more video media types, or both audio and video media types.

The format for `tos-values` entry must follow: `<media-type>:<tos-value>`. The `<media-type>` portion is `sip`, and the `<tos-value>` is either a decimal or hexadecimal value to insert.

You can use the `add` and `delete` commands when you enter the `tos-values` parameter to edit the list of ToS values; entering a new list of values without the `add` or `delete` commands overwrites the entire list.

Applying a Media Policy to a Realm

To apply a media policy to a realm:

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

```
ORACLE# configure terminal
```

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

```
ORACLE (configure) # media-manager
```

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

```
ORACLE (media-manager) # realm
ORACLE (realm) #
```

4. `media-policy`—Enter the unique name of the media policy you want to apply to this realm.

Using Class Profile for Packet Marking

Class profile provides an additional means of ToS marking, but only for limited circumstances. Use `class-profile` only if you are marking ToS on traffic destined for a specific To address, and when `media-policy` is not used on the same realm. Using `media-policy` for ToS marking is, by far, more common.

To configure a class profile, you prepare your desired media policy, create the class profile referencing the media policy and the To address, and set the name of the class profile in the appropriate realm configuration.

Class Profile and Class Policy Configuration

This section shows you how to configure packet marking using a class profile.

To configure the class profile and class policy:

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


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

4. Type policy and press Enter to begin configuring the class policy.

```
ORACLE(class-profile)# policy
```

From this point, you can configure class policy parameters. To view all class policy configuration parameters, enter a ? at the system prompt.

5. profile-name—Enter the unique name of the class policy. When you apply a class profile to a realm configuration, you use this value.
6. to-address—Enter a list of addresses to match to incoming traffic for marking. You can use E.164 addresses, a host domain address, or use an asterisk (*) to set all host domain addresses.
7. media-policy—Enter the name of the media policy you want to apply to this class policy.

Applying a Class Policy to a Realm

To apply a class policy to a realm:

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

```
ORACLE# configure terminal
```

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

```
ORACLE(configure)# media-manager
```

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

```
ORACLE(media-manager)# realm
ORACLE(realm)#
```

4. class-profile—Enter the name of the class profile to apply to this realm. This is the name you set in the profile-name parameter of the class-policy configuration.

Differentiated Services for DNS and ENUM

The Oracle CSM can mark DNS/ENUM packets with a configurable differentiated services code point (DSCP).

Certain service providers mandate support for Differentiated Services (DS) on all traffic streams exiting any network element. This mandate requires that network elements function as a DS marker — that is as a device that sets the Distributed Services Codepoint (DSCP) in the Differentiated Services field of the IP header.

Previous software releases provided the capabilities to mark standard SIP and Real-Time Protocol (RTP) messages. Release S-CX6.4.0M3 adds the capability to mark DNS and ENUM queries.

For basic information about DS, refer to:

- The Net-Net S-CX6.4.0 4000 CLI Configuration Guide (Realm-Based Packet Marking in Realms and Nested Realms), which provides information on currently supported DS marking of SIP and RTP packets
- *IETF RFC 2474, Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers* (<http://www.ietf.org/rfc/rfc2474.txt>)
- *IETF RFC 2475, An Architecture for Differentiated Services* (<http://www.ietf.org/rfc/rfc2475.txt>).

DS provides a mechanism to define and deliver multiple and unique service classifications that can be offered by a service provider. Specific service classifications are identified by a DSCP, essentially a numeric index. The DSCP maps to a per-hop-behavior (PHB) that defines an associated service class. PHBs are generally defined in terms of call admission controls, packet drop criteria, and queue admission algorithms. In theory, DS supports 64 distinct classifications. In practice, however, network offerings generally consist of a much smaller suite, which typically includes:

- Default PHB - required best effort service — defined in RFC 2474
- Expedited Forwarding (EF) PHB - low-loss, low-latency — defined in RFC 3246, An Expedited Forwarding PHB
- Assured Forwarding (AF) PHB - assured delivery within subscriber limits — defined in RFC 2597, *Assured Forwarding PHB Group*, and RFC 3260, *New Terminology and Clarifications for Diffserv*

Realms and Nested Realms

- Class Selector PHBs - maintain compatibility with previous TOS (type of service) precedence usage — defined in RFC 2474

Marking of DNS and ENUM queries requires the creation of a Differentiated Services media policy, and the assignment of such a policy to a specific realm.

Differentiated Services for DNS and ENUM Configuration

To create a Differentiated Services media policy:

1. From Superuser mode, use the following ACLI command sequence to move to media-policy configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# media-manager
ORACLE(media-manager)# media-policy
ORACLE(media-policy)#
```

2. Use the required name parameter to provide a unique identifier for this media-policy instance.

```
ORACLE(media-policy)# name diffServeDnsEnum
ORACLE(media-policy)#
```

3. Use the tos-settings command to move to tos-settings configuration mode.

```
ORACLE(media-policy)# tos-settings
ORACLE(tos-settings)#
```

4. Use the required media-type parameter to identify the packet-type subject to Differentiated Services marking.

For DNS/ENUM packets, use message/dns.

```
ORACLE(tos-settings)# media-type message/dns
ORACLE(tos-settings)#
```

5. Use the required tos-value parameter to specify the 6-bit DSCP to be included in the Differentiated Services field of the IP header.

Specify the DSCP as an integer (within the range from 0 to 63). The DSCP can be expressed in either decimal or hexadecimal format.

```
ORACLE(tos-settings)# tos-value 0x30
ORACLE(tos-settings)#
```

6. Other displayed parameters, media-attributes and media-sub-type, can be safely ignored.

7. Use the done and exit commands to complete Differentiated Services media policy configuration and return to media-policy configuration-mode.

```
ORACLE(tos-settings)# done
tos-settings
  media-type           message/dns
  media-sub-type
  tos-value            0x30
  media-attributes
ORACLE(tos-settings)# exit
ORACLE(media-policy)#
```

8. If necessary, repeat steps 2 through 7 to create additional Differentiated Services media policies.
9. Assign this media policy to the target realm via the media-policy parameter in a realm-config object.

Oracle CSM Supporting the IMS Core

General Description

The Oracle CSM 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 CSM can perform the SIP registrar role in conjunction with an HSS.

Message Authentication for SIP Requests

The Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM authenticates REGISTER request messages from all endpoints, while INVITES are only authenticated from anonymous endpoints.

User Authorization

In an IMS network, the Oracle CSM requests user authorization from an HSS when receiving a REGISTER message. An HSS is defined on the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM.

The Oracle CSM 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 CSM 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 CSM. If the endpoint is registering, the Oracle CSM 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 CSM besides REGISTER, the Oracle CSM 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 CSM 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 CSM always sends a value of "MD5"

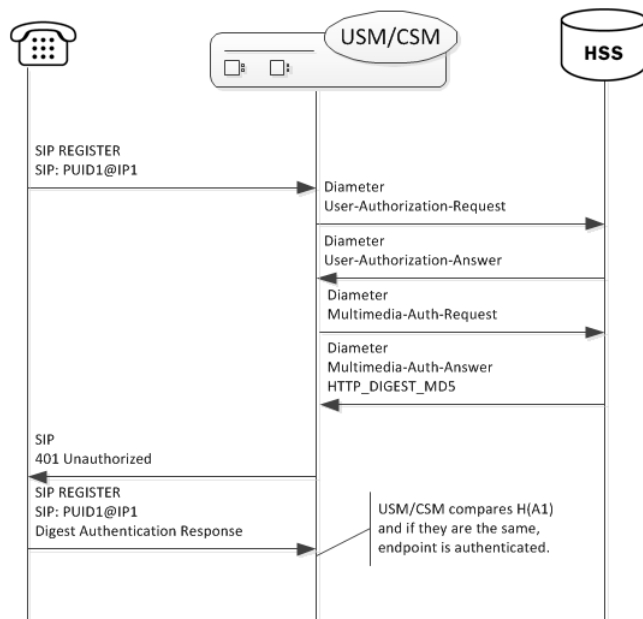
SIP Authentication Response

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

Oracle CSM Authentication Check

At this point, the Oracle CSM has received an MD5 hash from the HSS and an MD5 hash from the UA. The Oracle CSM 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 CSM acts as a SIP Registrar and updates an HSS with the state of its registrants.

IMS-AKA Support

The Oracle CSM 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 CSM 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.

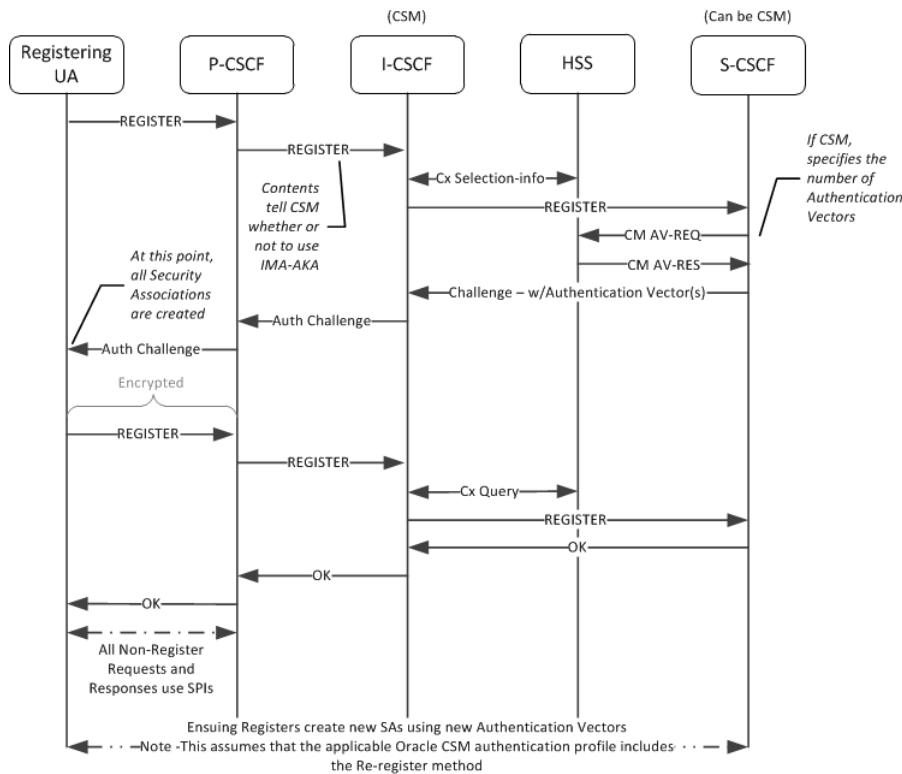
The first device to receive the REGISTER at the core is a P-CSCF, such as the Oracle USM. For the Oracle USM, appropriate configuration determines that it uses IMS-AKA as the authentication mechanism on the access interface. For an Oracle CSM, the presence and state of the “integrity-protected” parameter 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 CSM), 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 CSM 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 CSM 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 CSM can send the authentication challenge to the UA. If multiple authentication vectors were provided by the HSS, the Oracle CSM 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 CSM 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 CSM as Registrar

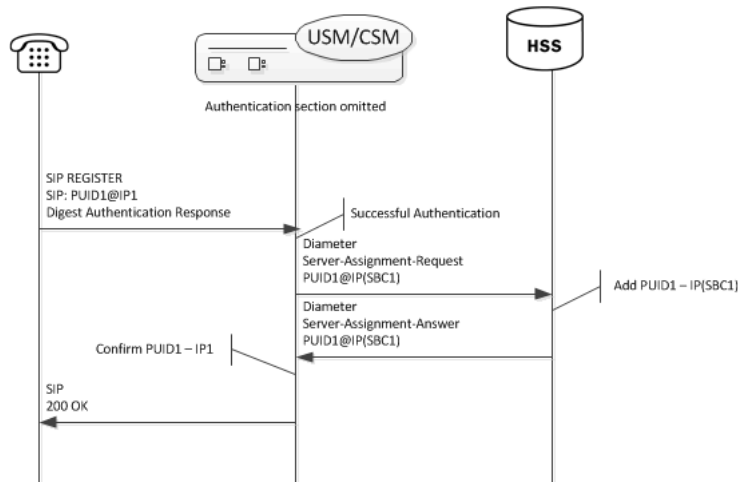
Creating a sip registrar configuration element enables the Oracle CSM to act as a SIP registrar. When registration functionality is enabled, the Oracle USM actually registers endpoints rather than only caching and forwarding registrations to another device. Oracle CSM registry services are enabled globally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

On receiving a REGISTER message, the Oracle CSM 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 CSM 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 CSM sends a 200 OK message back to the registering UA.

New Registration

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




Registration Handling for Online and Offline Operation Modes

The Oracle CSM provides a means of running in offline mode. Offline mode provides the Oracle CSM with a graceful means of taking itself out of service without impacting active sessions. This mode can operate in conjunction with other Oracle infrastructure elements, including the SLRM or network management systems. Users or management software put the Oracle CSM into offline mode because the resource may not be currently required or for system maintenance.

The Oracle CSM normally operates in online mode, fielding SIP messaging and providing location services. When set to offline mode, it begins releasing UE registrations, allowing the IMS infrastructure to move its UEs to other S-CSCFs. In addition, it stops handling calls for unregistered users. The combination of these two actions effectively takes the Oracle CSM completely out of service after it has released registrations for all its UEs.

The user can explicitly set their Oracle CSM online or offline using the command **set-system-state**. The user can confirm this operational mode using the **show system-state** command.

 **Note:** The SCZ7.2.5 release enhanced the **set-system-state** control to add registration management to the legacy system state mechanism.

Releasing Registrations

When set to offline mode, the Oracle CSM begins taking itself out of service as an S-CSCF. This process may last a long time because the Oracle CSM continues to service:

- UEs with active sessions; and
- UEs that are not yet marked for release.

Offline mode does not use timers to control its operation. Instead, the Oracle CSM simply waits for UEs to become eligible, marks them for release, and then begins to release them. The process does not need to be completed either. The user or management applications can set it back to online mode at any time.

When the Oracle CSM goes back into online mode, it begins to accept calls and registrations for all UEs that are not still marked for release. Instead, the system completes the release process for those UEs, which must then re-register. The system replies to any registration or originating service requests from UEs marked for release with a 504 message. The system continues to provide terminating services for these UEs.

Offline Registration Release Procedure

When set to offline, the Oracle CSM first identifies UEs in the registration cache that have registration event subscriptions and marks them for deletion (dirty). Recall that the Oracle CSM does not release any UE currently engaged in a session. For UEs in the cache without registration event subscriptions, the Oracle CSM waits for registration refreshes. When they refresh, the Oracle CSM marks them for release and begins to release them.

The release procedure consists of the Oracle CSM performing following steps:

1. Mark UEs in the registration cache as candidates for de-registration.
2. Send an SAR to the HSS with ADMINISTRATIVE_DEREGISTRATION action for each marked UE. The Oracle CSM does this to remove itself as the assigned S-CSCF. This allows other Oracle CSMs become the S-CSCF for this UE.
3. Send deregister requests to application servers to which the Oracle CSM performed third party registration for the UE.
4. Remove the UE from the registration cache.
5. Send reg-event NOTIFYs to application servers that have reg-event subscriptions for the UE.
6. Send reg-event NOTIFYs to all the UEs contacts.
7. Send reg-event NOTIFYs to the P-CSCFs that have reg-event subscriptions for the UE.

Interaction with SLRM

If there is one or more SLRMs within your deployment, the Oracle CSM notifies them when it goes offline. This notification mechanism is the same as that used by the Oracle CSM to indicate that it is low on resources. The Oracle CSM advertises to the SLRM that it has no resources to handle registrations. When going back online, the Oracle CSM simply re-advertises its resources, resuming normal operation with the SLRM.

Interaction with UEs

Actions the Oracle CSM may take while receiving REGISTER, De-register and REGISTER REFRESH requests in off-line mode depend on the state of the affected UEs. These actions, and the conditions that invoke them include:

- Send a 504 to the originating UE - See list of 504 cases below.

- Send a 200OK to the originating UE:
 - De-registers for users not marked for deletion, allowing normal de-registration.
- Send a 200OK, and lower the registration expiration timer to 3 minutes:
 - REGISTER refresh request for an existing contact with active sessions.
 - REGISTER refresh request for a new contact for an existing user not marked for deletion.

Actions the Oracle CSM may take while receiving INVITE requests in off-line mode, depending on the state of the affected UEs, include:

- Send a 504 to the originating UE - See list of 504 cases below.
- Perform originating services:
 - The originating user is a registered UE in the cache.
 - The originating user is an unregistered UE in the cache.
- Perform terminating services - forward to contacts:
 - The terminating user is a registered UE in the cache.
 - Originating Services performed, terminating UE is not in the cache.
- Perform unregistered services - Send 480, temporarily unavailable:
 - The Oracle CSM has completed originating services for an unregistered UE that is in the cache.

While in offline mode, the Oracle CSM returns a 504 error message to the originating UE under the following REGISTER, De-register and REGISTER REFRESH request conditions:

- Initial register for an inactive user not in cache.
- Initial register for an inactive user marked as dirty in cache.
- Refresh register for an inactive user in registered state in the cache.
- Refresh register for an inactive user marked as dirty in cache.
- Deregister for an inactive user in registered state in the cache.
- Deregister request for an inactive user marked as dirty in cache.

The Oracle CSM returns a 504 error message to the originating UE under the following INVITE request conditions:

- Call from a UE that is not in the cache.
- Call from a UE calling that is marked as dirty in cache.
- OOB call requesting orig service for a user not in the cache.
- OOB call requesting orig service for a user marked as dirty.
- OOB call requesting term service for a user is not in cache.
- OOB call requesting term service for a user marked as dirty.

The 504 Local Response Codes

The Oracle CSM uses the following two local response codes to override the 504 response for REGISTER methods and indicate why it cannot serve the UE.

- csm-releasing-users-register
- csm-releasing-users-invite

Releasing Unregistered Users

When a call arrives at an Oracle CSM either to or from a user that is not registered at that Oracle CSM, it performs a location query with the HSS to determine if the unknown UE is registered at another S-CSCF. If there is no registration, the Oracle CSM takes ownership of the UE. The system stores information about these UEs in its registration cache, labelled "NEVER REGISTERED". Barring any further, related action within the infrastructure, the UE would remain homed to the Oracle CSM. Upon expiry of this feature's timer, the Oracle CSM sends an SAR to the HSS, providing an assignment type of ADMINISTRATIVE_DEREGISTRATION for the UE. This allows the UE to be a user at a different S-CSCF the next time it is a call sender or receiver. A common use case for this scenario is a roaming UE.

When the Oracle CSM issues the SAR, it also marks the UE as 'dirty' (in the process of being de-assigned) to accommodate the following operational scenarios:

- The UE attempts to register—The Oracle CSM rejects the register, replying with a 504 error message.
- The UE has existing calls—The Oracle CSM continues to support the call, based on a stored copy of the service profile.
- A new call arrives—The Oracle CSM rejects the call. The Oracle CSM replies with a '480, Temporarily Unavailable' error message if the UE is the callee; the Oracle CSM responds with a 504 if the UE is the caller.

The user can configure the `unreg-cache-expiry` parameter in seconds on a per-registrar basis. This syntax is shown below.

```
ORACLE(sip-registrar) # unreg-cache-expiry 120
```

The parameter accepts values in the range of 0 to 604800, with 0 specifying that the Oracle CSM does not cache unregistered users. A setting of 0 means the Oracle CSM takes ownership, downloads service profiles, and then releases the user after the call without caching.

Handling Public Service Identities (PSIs)

Public Service Identities (PSI) appear as unregistered users in the Oracle CSM. PSIs appear as either Distinct PSIs or Wildcarded PSIs. Similar to unregistered users, the Oracle CSM takes ownership of the PSI if it is unassigned and a call is made to or from it. By default, PSIs are not released. However, the user can configure the `psi-cache-expiry` option in seconds on a per-registrar basis to cause the Oracle CSM to release PSIs. This syntax is shown below.

```
ORACLE(sip-registrar) # options psi-cache-expiry=120
```

Limiting AOR Contacts

The Oracle CSM allows you to limit the number of contacts that apply to AORs. If the Oracle CSM 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 CSM 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 CSM registers UAs, it requests to assign itself as the S-CSCF for the registering AoR. The Oracle CSM'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 CSM on behalf of the registered user.

Server Assignment Messages

The Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM sends an MAR to the HSS to retrieve authentication data once again).

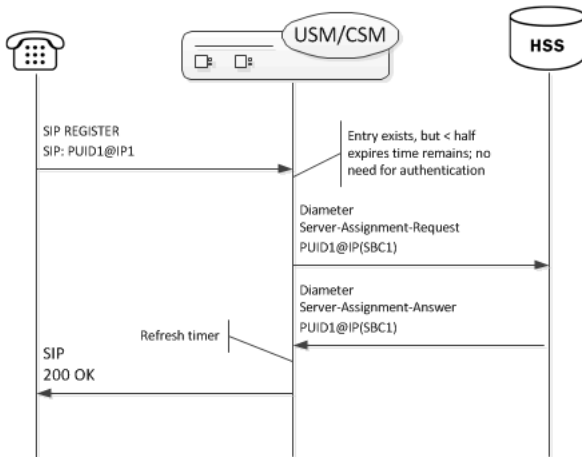
Next, the Oracle CSM determines if it can perform a local REGISTER refresh or if the HSS needs to be updated. If any of the following 3 conditions exists for the re-registering UA, the Oracle CSM updates the HSS:

- 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 CSM 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 CSMs 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 CSM1, moves to Oracle CSM2, and then returns to Oracle CSM1.

If the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM’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 CSM 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 CSM can perform this way by considering the endpoint’s reg-id and instance-id parameters defined in [RFC 5626](#).

The Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM determines if it is the First hop. If there is only one Via: header in the REGISTER, the Oracle CSM 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 CSM 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 CSM receives requests for the same AOR with some registrations with reg-id + instance-id and some without them, the Oracle CSM 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 CSM 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 CSM configuration required.

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

To allow the feature, the Oracle CSM 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.

ACLI Instructions

The following configuration enables the Oracle CSM to authorize and authenticate registering users. In addition it sets the Oracle CSM 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. Both IPv4 and IPv6 addressing is supported.
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 CSM 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 CSM 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 CSM 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
```

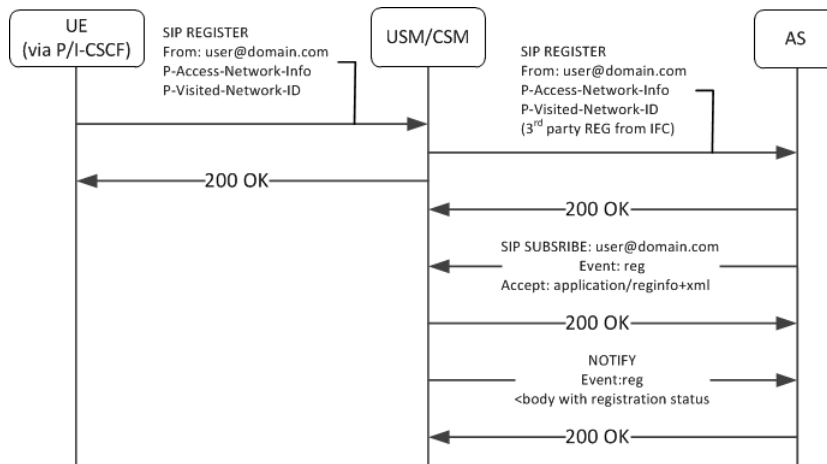
SIP Registration Event Package Support

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

SUBSCRIBE Processing

When the Oracle CSM 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 CSM 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 CSM 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 CSM proceeds with the request.

3. Authorizes the request. The Oracle CSM 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 CSM 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 CSM 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 CSM only processes subscriptions for users in its registration cache, replying with a 403 Forbidden if not. For cached users, the Oracle CSM 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 CSM applies its own configured minimum or the default (600000 seconds), whichever is greater.

If the SUBSCRIBE includes an Expires header, the Oracle CSM 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 CSM denies the subscription, sending a 423 Too Brief message.

When the Oracle CSM 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 CSM. Messages must be received from authorized subscribers and on the same realm and connection as the original SUBSCRIBE or the Oracle CSM rejects the refresh request.

Reg Event NOTIFY Messages

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

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

The Oracle CSM 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 CSM includes:

- The Oracle CSM always sends full information on all contacts, and indicates such within the reginfo element. The Oracle CSM 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 CSM does not include the following optional attributes within the contact element:
 - expires
 - retry-after
 - duration registered
 - display-name
- The Oracle CSM 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 CSM to limit notification traffic. By specifying the number of seconds between NOTIFY messages, you prevent the Oracle CSM 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 CSM sends out a NOTIFY for every registration event subscription. Note also that the Oracle CSM 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 CSM 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 CSM's local policy routing functionality.

Within the context of local policy routing, the Oracle CSM 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 CSM to load balance and route around softswitch failures.

When a message arrives at the Oracle CSM, it determines whether it is coming from a session agent. If so, the Oracle CSM 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 CSM 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 CSM 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 CSM is using the HSS as a resource within the context of its routing decisions.

When an INVITE arrives, the Oracle CSM 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 CSM proceeds as described below in the section LIR/LIA Transaction.

Note that you can configure the Oracle CSM 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 CSM, 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 CSM, 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 CSMs, HSSs, and other network elements communicate and exchange SIP messaging. It is advisable to configure this parameter to ensure well defined reachability among Oracle CSMs.

Routing Based on UA Capabilities

In compliance with RFC 3841, the Oracle CSM is able to make forwarding and forking decisions based on preferences indicated by the UA. To do this, the Oracle CSM 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 CSM made routing preference decisions solely via the q value present in the contact header. In cases where the preferences were equal, the Oracle CSM simply forwarded to those contacts simultaneously (parallel forking). In cases where the q value were not equal, the Oracle CSM forwarded in sequence (sequential forking), forwarding to the highest q value first.

The Oracle CSM 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 CSM 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 CSM

Endpoints register their capabilities by listing them in the Contact headers of the REGISTER request. The Oracle CSM stores these feature parameters in its registration cache along with the other contact information. In the case of ENUM databases, the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM fully supports. UAs include these parameters in the accept-contact: header to further clarify capabilities requirements for the session. The Oracle CSM 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 CSM only forwards to contacts that support the features and have registered that support.

If only the require parameter is present, the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM 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.

Tel-URI Resolution

The Oracle CSM 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 CSM to perform a number normalization procedure and ensure it presents a full E.164 number for resolution. Upon successful resolution, the Oracle CSM proceeds with ensuing signaling procedures.

To configure the Oracle CSM 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 CSM 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 CSM procedures include:

- Request from the access side:
 1. The Oracle CSM 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 CSM 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 CSM 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 CSM requests e.164 resolution from the ENUM server(s).

Actions Based on Lookup Results

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

- The SIP-URI is in the Oracle CSM cache, in which case the Oracle CSM performs terminating services.
- The SIP-URI is not in the Oracle CSM cache, and the Oracle CSM is configured to service the returned domain. In this case, the Oracle CSM performs the following:
 1. The Oracle CSM issues an LIR for the SIP-URI.

2. The Oracle CSM forwards the message to the correct S-CSCF.
- The SIP-URI is not in the Oracle CSM cache, and the Oracle CSM is not configured to service the returned domain.
In this case, the Oracle CSM performs refers to local policy to forward the message via local policy.

PSTN Breakout Routing

The Oracle CSM 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 CSM does not validate any rn or cic information. Instead, it simply routes the request. Note that the Oracle CSM uses cic information instead of rn if both are present in the request. RFC 4694 compliant circumstances under which the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM to successfully perform lookups within two domains.

HSS Initiated User Profile Changes

The Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM.
 - DIAMETER_ERROR_TOO_MUCH_DATA—The request failed. The Oracle CSM informs to the HSS that it tried to push too much data into the Oracle CSM.
 - DIAMETER_UNABLE_TO_COMPLY—The request failed.
- Experimental-Result—indicates diameter application (3GPP/Cx) error if present.

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 CSM 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 CSM, 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 CSM 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 CSM 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 CSM 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.

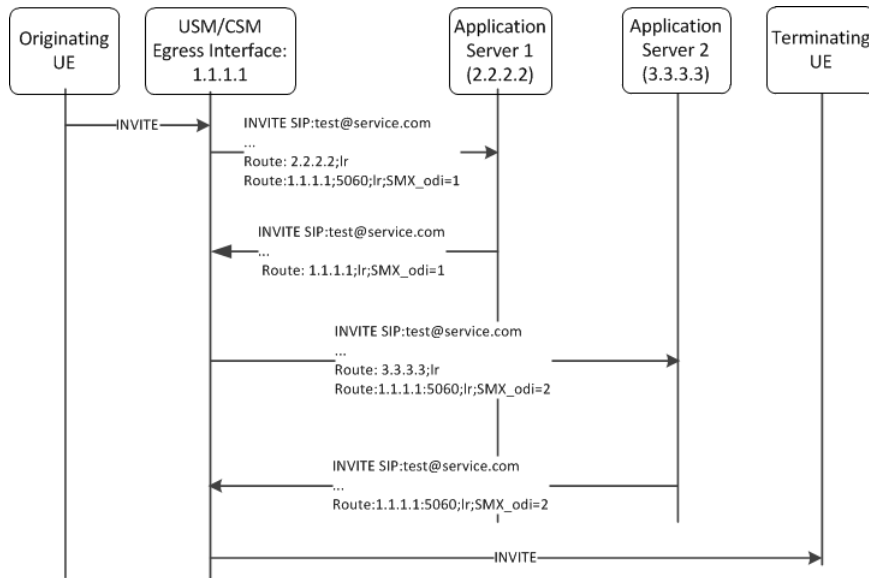
SIP Call

The Oracle CSM 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 CSM 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 CSM, it is expected to include the ODI parameter that it received from the Oracle CSM, 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 CSM 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 CSM checks if the target of the initial request is registered to itself, or not. If the UA is not registered locally the Oracle CSM forwards the request by regular means into the network. If the target UA is registered locally, the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM.
- If the Request-URI changes when visiting an application server, the Oracle CSM 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 CSM 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 CSM 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.
- The Oracle CSM provides an alternative, configurable option that allows the user to specify the use of route header information to determine Served User and Session Case for out-of-the-blue (OOTB) calls. This method is 3GPP-compliant. By default, the Oracle CSM uses information from the P-Served-User (PSU) header. The user configures this behavior by enabling the ignore-psu-sesscase option in the ifc-profile.

IFC Support for Unregistered Users

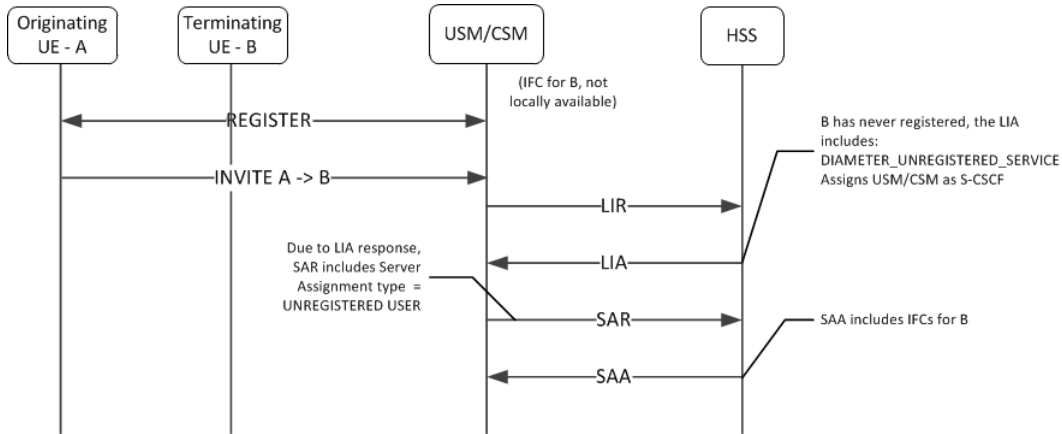
The Oracle CSM can download Initial Filter Criteria (IFC) from the HSS for unregistered users. This section displays applicable message sequence diagrams.

UE-terminating requests to an unregistered user

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

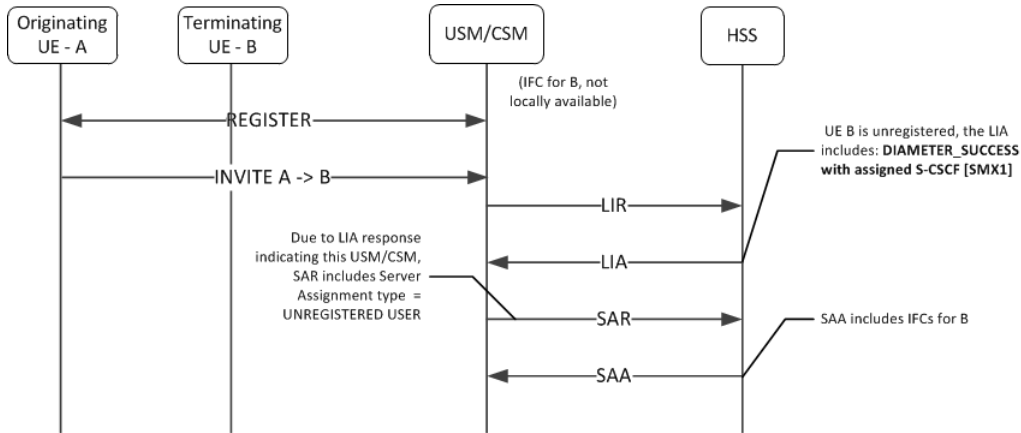
Terminating UA - Unregistered

UE has never registered.

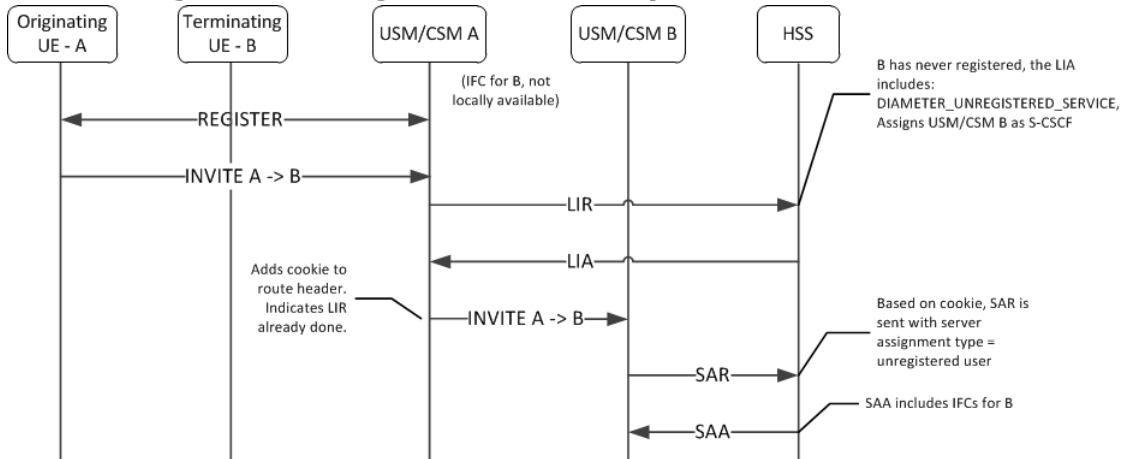


Terminating UA - Unregistered

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 CSM



UE Subsequent Registration

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

Caching the Downloaded IFC

When the Oracle CSM 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 CSM 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 CSM 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 CSM to act solely as an I-CSCF. When disabled, the Oracle CSM always forwards requests from unregistered users to the serving group. The default is enabled, which retains the S-CSCF function on this Oracle CSM.
7. serving-group—Set this parameter to a Session Agent Group (SAG) name. The Oracle CSM 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 CSM 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 CSM itself. This establishes a backup scenario in case the HSS is not responsive.

To support the SiFC feature on the Oracle CSM, 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 CSM 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

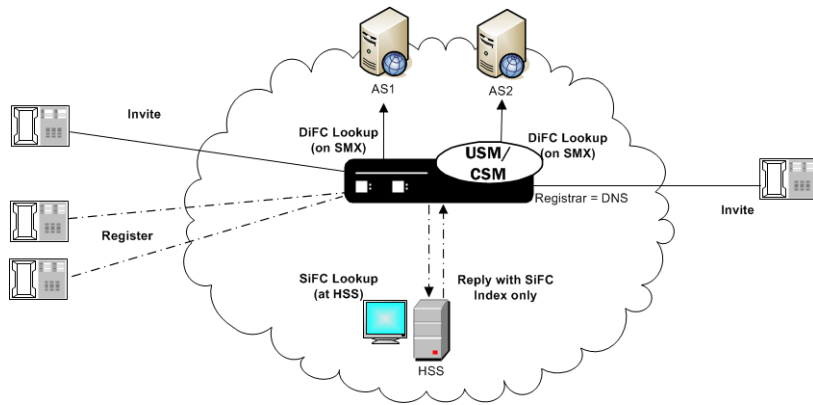
When an applicable end station registers, the Oracle CSM forwards the registration to the HSS normally. Given SiFC configuration however, the HSS sends a service-profile containing the SiFC identifiers to the Oracle CSM rather than the entire service definition. The Oracle CSM parses these identifiers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

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

DiFC Usage

In contrast to SiFCs, the Oracle CSM fires DiFCs within the context of a session. During the session, the Oracle CSM 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 CSM 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>
    </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 CSM. 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 CSM 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 CSM evaluates explicitly downloaded iFCs first when determining where to look for a service. If the Oracle CSM 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 CSM validates the SiFC and DiFC files whenever you Activate your configuration.

SiFC and DiFC Configuration

To configure the Oracle CSM 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 CSM 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 CSM 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 CSM caches filter criteria information that uses this wildcard syntax. This avoids the need for SAR/SAA exchanges between the Oracle CSM and the HSS every time an entity requests the service. The Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM.

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 CSM establishes and manages multiple Cx connections with each applicable HSS. This management is achieved by connection identifiers on the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM selects the next highest priority HSS server for all operations if the first HSS fails. The Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM marks that HSS as out of service.

The Oracle CSM 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 CSM attempts to establish diameter connections with out of service HSS servers. When those connections succeed, the Oracle CSM 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 CSM. 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 CSM 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
```

Routing with Local Policy

This chapter explains how to configure session routing and load balancing for SIP services using local policy and session agents. It contains information about configuring session agents and session agent groups, as well as local policies that can be used for routing SIP messaging.

Session Agents Session Groups and Local Policy

When you configure session routing for SIP, you can use session agents, session agent groups and local policies to define routing. (Using session agents and session agent groups is not required.)

- session agent: defines a signaling endpoint. It is a next hop signaling entity that can be configured to apply traffic shaping attributes.
- session agent group (SAG): contains individual session agents. Members of a SAG are logically equivalent (although they might vary in their individual constraints) and can be used interchangeably.

You apply an allocation strategy to the SAG to allocate traffic across the group members. Session agent groups also assist in load balancing among session agents.

- local policy: indicates where session request messages, such as SIP INVITES, are routed and/or forwarded. You use a local policy to set a preference for selecting one route over another.

Another element of routing is the realm. Realms are used when a any log level you set here overrides the log level you set in the system configuration's process log level parameter, communicates with multiple network elements over a shared intermediate connection. Defining realms allows sessions to go through a connection point between the two networks.

When you configure a realm, you give it an identifier, which stores the name of the realm associated with the any log level you set here overrides the log level you set in the system configuration's process log level parameter. The realm identifier value is also needed when you configure session agents and local policies. You can associate a realm with a session agent to identify the realm for sessions coming from or going to the session agent. You also need the realm identifier when you configure local policy to identify the egress realm (realm of the next hop).

SIP Session Agents

SIP session agents can include the following:

- softswitches
- SIP proxies
- application servers

- SIP gateways
- SIP endpoints

In addition to functioning as a single logical next hop for a signaling message (for example, where a SIP INVITE is forwarded), session agents can provide information about next or previous hops for packets in a SIP agent, including providing a list of equivalent next hops.

You can use the session agent to describe one or more SIP next or previous hops. Through the configured carriers list, you can identify the preferred carriers to use for traffic coming from the session agent. This set of carriers will be matched against the local policy for requests coming from the session agent. You can also set constraints for specific hops.

Session Agent Status Based on SIP Response

The Oracle CSM can take session agents out of service based on SIP response codes that you configure, and you can also configure SIP response codes that will keep the session agent in service.

With this feature disabled, the Oracle CSM determines session agents' health by sending them ping messages using a SIP method that you configure. Commonly, the method is an OPTIONS request. If it receives any response from the session agent, then the Oracle CSM deems that session agent available for use.

However, issues can arise when session agents are administratively out of service, but able to respond to OPTIONS requests. A session agent like this might only respond with a 200 OK when in service, and send a 4xx or 5xx message otherwise.

The session agent status feature lets you set the SIP response message that either takes a session agent out of service or allows it to remain in service when it responds to the Oracle CSM's ping request.

Details of this feature are as follows:

- The Oracle CSM only considers a session agent in service when it responds to a request method you set with the final response code that you also set. If a final response code is set, then provisional responses are not used for determining whether or not to take a session agent out of service. If the Oracle CSM receives a final response code that does not match the session agent configuration, it treats the session agent as though it had not responded.
- The Oracle CSM takes a session agent out of service when it receives an error response for dialog creating request with a response code listed in the new out-of-service-response-codes parameter.

In the case where a the session agent's response has a Retry-After header, the Oracle CSM tries to bring the session agent back into service after the period of time specified in the header. To do so, it sends another ping request.

There are two lists you can configure in the session agent configuration to determine status:

- In-service list—Set in the ACLI ping-in-service-response-codes parameter, this list defines the response codes that keep a session agent in service when they appear in its response to the Oracle CSM's ping request. Furthermore, the Oracle CSM takes the session agent out of service should a response code be used that does not appear on this list.
- Out-of-service list—Set in the ACLI out-of-service-response-codes parameter, this list defines the response codes that take a session agent out of service when they appear in its response to the Oracle CSM's ping request or any dialog-creating request.

When the Oracle CSM receives a session agent's response to its ping request, it first checks to see if there is an in-service list of responses configured for that session agent. If the list is configured and the Oracle CSM determines that there is a match, the session agent is deemed in service. Otherwise it takes the session agent out of service. In this way, the in-service list takes precedence over the out-of-service list. If you configure the in-service list, then the Oracle CSM ignores the out-of-service list.

If there is no list of in-service responses for the session agent, then the Oracle CSM checks the out of service list. If it is configured and the Oracle CSM determines that there is a match, the Oracle CSM removes that session agent from service. If there is no match, then the session agent is deemed in service.

SIP Session Agent Continuous Ping

You can configure the Oracle CSM to use either a keep-alive or continuous method for pinging SIP session agents to determine their health—i.e., whether or not the Oracle CSM should route requests to them. To summarize the two methods:

- **keep-alive**—Oracle CSM sends a ping message of a type you configure to the session agent in the absence of regular traffic. Available in Release C5.1.0 and in earlier releases.
- **continuous**—The Oracle CSM sends a ping message regardless of traffic state (regular or irregular); the Oracle CSM regularly sends a ping sent based on the configured ping interval timer. Available in Release C5.1.1p6 and in later releases.

By sending ping messages, the Oracle CSM monitors session agents' health and can determine whether or not to take a session out of service (OOS), leave it in service, or bring it back into service after being OOS.

When you set it to use the keep-alive mode of pinging (available in Release C5.1.0 and before), the Oracle CSM starts sending a configured ping message to a session agent when traffic for that session agent has become irregular. The Oracle CSM only sends the ping if there are no SIP transactions with a session agent over a configurable period of time, to which the session agent's response can have one of the following results:

- **Successful response**—A successful response is either any SIP response code or any response code not found in the `out-service-response-codes` parameter; these leave the session agent in service. In addition, any successful response or any response in the `ping-in-service-response-codes` parameter can bring a session agent from OOS to in-service status.
- **Unsuccessful response**—An unsuccessful response is any SIP response code configured in the `out-service-response-codes` parameter and takes the session agent sending it OOS. Because this parameter is blank by default, the Oracle CSM considers any SIP response code successful.
- **Transaction timeout**—A transaction timeout happens when the session agent fails to send a response to the Oracle CSM's request, resulting in the session agent's being taken OOS.

Despite the fact that the keep-alive ping mode is a powerful tool for monitoring session agents' health, you might want to use the continuous ping method if you are concerned about the Oracle CSM not distinguishing between unsuccessful responses from next-hop session agents and ones from devices downstream from the next-hop session agent. For example, if a SIP hop beyond the session agent responds with a 503 Service Unavailable, the Oracle CSM does not detect whether a session agent or the device beyond it generated the response.

When you use the continuous ping method, only the next-hop session agent responds—preventing the request from being sent to downstream devices. The Oracle CSM also sends the ping in regular traffic conditions when in continuous ping mode, so it is certain the response comes from the next hop associated with the session agent. And in continuous ping mode, only entries for the `ping-out-service-response-codes` parameter and transaction timeouts bring session agents OOS.

SIP SA Continuous Ping Configuration

You can set the ping mode in the session agent or session constraints configuration. For backward compatibility, the default for the `ping-send-mode` parameter is `keep-alive`, or the functionality available in Release C5.1.0 and in earlier releases.

To configure the ping mode for a session agent:

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

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

2. Type `session-router` and press Enter.

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

3. Type `session-agent` and press Enter.

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

Routing with Local Policy

If you are adding rate constraints to an existing configuration, then you will need to select the configuration you want to edit.

4. ping-send-mode—If to want to use continuous ping mode to send ping messages to session agents in regular traffic conditions, set this parameter to continuous. If you want to use the keep-alive mode, leave this parameter set to keep-alive (default).
5. Save and activate your configuration.

To configure the ping mode for the session constraints:

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

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

7. Type session-router and press Enter.

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

8. Type session-constraints and press Enter.

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

If you are adding rate constraints to an existing configuration, then you will need to select the configuration you want to edit.

9. ping-send-mode—If to want to use continuous ping mode to send ping messages to session agents in regular traffic conditions, set this parameter to continuous. If you want to use the keep-alive mode, leave this parameter set to keep-alive (default).
10. Save and activate your configuration.

About Session Agents

This section describes session agents. A session agent defines a signaling endpoint. It is a next hop signaling entity that can be configured to apply traffic shaping attributes. Service elements such as gateways, softswitches, and gatekeepers are defined automatically within the Oracle CSM as session agents. For each session agent, concurrent session capacity and rate attributes can be defined. You can group session agents together into session agent groups and apply allocation strategies to achieve traffic load balancing.

You can assign a media profile to a session agent.

You can configure a set of attributes and constraints for each session agent to support the following:

- session access control: Oracle CSM only accepts requests from configured session agents
- session admission control (concurrent sessions): Oracle CSM limits the number of concurrent inbound and outbound sessions for any known service element.
- session agent load balancing: session agents are loaded based on their capacity and the allocation strategy specified in the session agent group.
- session (call) gapping: Oracle CSM polices the rate of session attempts to send to and receive from a specific session agent.

Session Agent Groups

Session agent groups contain individual session agents. Members of a session agent group are logically equivalent (although they might vary in their individual constraints) and can be used interchangeably. You can apply allocation strategies to session agent groups.

Examples of session agent groups include the following:

- application server cluster
- media gateway cluster

- softswitch redundant pair
- SIP proxy redundant pair
- gatekeeper redundant pair

Session agent group members do not need to reside in the same domain, network, or realm. The Oracle CSM can allocate traffic among member session agents regardless of their location. It uses the allocation strategies configured for a SAG to allocate traffic across the group members.

Allocation strategies include the following:

Allocation Strategy	Description
Hunt	Oracle CSM selects the session agents in the order in which they are configured in the SAG. If the first agent is available, and has not exceeded any defined constraints, all traffic is sent to the first agent. If the first agent is unavailable, or is in violation of constraints, all traffic is sent to the second agent. And so on for all session agents in the SAG. When the first agent returns to service, the traffic is routed back to it.
Round robin	Oracle CSM selects each session agent in the order in which it is configured, routing a session to each session agent in turn.
Least busy	Oracle CSM selects the session agent with the least number of active sessions, relative to the maximum outbound sessions or maximum sessions constraints (lowest percent busy) of the session agent.
Proportional distribution	Session agents are loaded proportionately based upon the respective maximum session constraint value configured for each session agent.
Lowest sustained rate	Oracle CSM routes traffic to the session agent with the lowest sustained session rate, based on observed sustained session rate.

You apply allocation strategies to select which of the session agents that belong to the group should be used. For example, if you apply the Hunt strategy session agents are selected in the order in which they are listed.

Request URI Construction as Sent to SAG-member Session Agent

The Oracle CSM constructs the request URI for a session agent selected from a session agent group by using the **session-agent > hostname** value of the selected **session-agent** target. This default behavior enables features such as trunk groups and ENUM to work properly. However, care must be given when the **hostname** parameter is not a resolvable FQDN. The **sag-target-uri=<value>** option can be used to overcome the default behavior.

The value is either

- **ip** – request URI constructed from **session-agent > ip-address**
- **host** – request URI constructed from **session-agent > hostname**

This option is global and is configured in the **sip-config** configuration element.

Request URI Construction as Forwarded to SAG-member Session Agent

The Oracle CSM constructs the request URI for a session agent selected from a session agent group by using the **session-agent > hostname** value of the selected **session-agent** target. This default behavior enables features such as trunk groups and ENUM to work properly. However, care must be given when the **hostname** parameter is not a resolvable FQDN. Use the **sag-target-uri=<value>** option to override the default behavior.

The value can be set to either:

- **ip** - request URI constructed from **session-agent > ip-address**
- **host** - request URI constructed from **session-agent > hostname**

Routing with Local Policy

This option is global and is configured in the **sip-config** configuration element.

SIP Session Agent Group Recursion

You can configure a SIP session agent group (SAG) to try all of its session agents rather than to the next-best local policy match if the first session agent in the SAG fails.

With this feature disabled, the Oracle CSM performs routing by using local policies, trunk group URIs, cached services routes, and local route tables. Local policies and trunk group URIs can use SAGs to find the most appropriate next-hop session agent based on the load balancing scheme you choose for that SAG: round robin, hunt, proportional distribution, least busy, and lowest sustained rate. When it locates a SAG and selects a specific session agent, the Oracle CSM tries only that single session agent. Instead of trying other members of the SAG, the Oracle CSM recurses to the local policy that is the next best match. This happens because the Oracle CSM typically chooses a SAG based on the fact that it has not breached its constraints, but the Oracle CSM only detects failed call attempts (due to unreachable next hops, unresolved ENUM queries, or SIP 4xx/5xx/6xx failure responses) after it has checked constraints. So the Oracle CSM only re-routes if there are additional matching local policies.

When you enable SIP SAG recursion, the Oracle CSM will try the additional session agents in the selected SAG if the previous session agent fails. You can also set specific response codes in the SAG configuration that terminate the recursion. This method of terminating recursion is similar to the Oracle CSM's ability to stop recursion for SIP interfaces and session agents.

Session agents are selected according to the strategy you set for the SAG, and these affect the way that the Oracle CSM selects session agents when this feature enabled:

- Round robin and hunt—The Oracle CSM selects the first session agent according to the strategy, and it selects subsequent session agents based on the order they are entered into the configuration.
- Proportional distribution, least busy, and lowest sustained rate—The Oracle CSM selects session agents based on the list of session agents sorted by the criteria specified.

You can terminate recursion based on SIP response codes that you enter into the SAG configuration. You can configure a SAG with any SIP response code in the 3xx, 4xx, and 5xx groups. Since you can also set such a list in the session agent configuration, this list is additive to that one so that you can define additional codes for a session agent group without having to repeat the ones set for a session agent.

About Local Policy

This section explains the role of local policy. Local policy lets you indicate where session requests, such as SIP INVITES, should be routed and/or forwarded. You use a local policy to set a preference for selecting one route over another. The local policy contains the following information that affects the routing of the SIP signaling messages:

- information in the From header

Information in the message's From header is matched against the entries in the local policy's from address parameter to determine if the local policy applies.

- list of configured realms

This list identifies from what realm traffic is coming and is used for routing by ingress realm. The source realms identified in the list must correspond to the valid realm IDs you have already configured

- local policy attributes

The attributes serve as an expression of preference, a means of selecting one route over another. They contain information such as the next signaling address to use (next hop) or whether you want to select the next hop by codec, the realm of the next hop, and the application protocol to use when sending a message to the next hop. You can also use the attributes to filter specific types of traffic.

Routing Calls by Matching Digits

Local policy routing of a call can be based on matching a sequence of digits against what is defined in the local policy. This sequence refers to the first digits in the (phone) number, matching left to right.

The following examples show how the Oracle CSM matches an area code or number code against configured local policies.


- If the number or area code being matched is 1234567 (where 123 is an area code), and the from address value in one local policy is 123, and the from address value in another local policy is 12, the Oracle CSM forwards the call to the server that is defined as the next hop in the local policy with 123 as the from address value.
- If the number or area code being matched is 21234, and the from address value in one local policy is 123, and the from address value in another local policy is 12, the Oracle CSM will not find a match to either local policy because the first character of the number or area code must match the first character in a from address or to address field.

The following examples show how the Oracle CSM matches an area or number code against different local policies: the first one has a From address value of 12 and the second has a From address value of 123. The Oracle CSM chooses the route of the local policy that is configured with the most digits matching the area or number code in its From address and To address fields.

- When the two different local policies route to two different servers, and the area or number code being matched is 123, the Oracle CSM selects the second local policy based on the From address value of 123.
- When the two different local policies route to two different servers, and the area or number code being matched is 124, the Oracle CSM selects the first local policy based on the From address value of 12.

Route Preference

The Oracle CSM builds a list of possible routes based on the source realm and the From-address (From-URI) and To-address (Request-URI), which forms a subset from which preference then decides. Any local policy routes currently outside of the configured time/day are not used, if time/day are set. Also, any local policy routes not on the list of carriers (if carriers is set and the requests has a Carrier header) are not used.

 **Note:** Source realm is used in the local policy lookup process, but it is not used in route preference calculations.

The Oracle CSM applies preference to configured local policies in the following order:

1. Cost (cost in local policy attributes) is always given preference.
2. Matching media codec (media profiles option in local policy attributes).
3. Longest matching To address (to address list in local policy).
4. Shortest matching To address (to address list in local policy).
5. Longest matching From address (from address list in local policy).
6. Shortest matching From address (from address list in local policy).
7. Narrowest/strictest day of week specification (days of week option in local policy attributes).
8. Narrowest/strictest time of day specification (start time and end time options in local policy attributes).
9. Wildcard matches (use of an asterisk as a wildcard value for the from address and to address lists in local policy).
10. Wild card matches are given the least preference. A prefix value of 6 is given a higher preference than a prefix value of * even though both prefix values are, in theory, the same length.

DTMF-Style URI Routing

The Oracle CSM supports the alphanumeric characters a-d, A-D, the asterisk (*), and the ampersand (#) for local policy matching purposes. The Oracle CSM handles these characters as standards DN (POTS) or FQDN when found in the to-addr (req-uri username) or from-addr (from0uri username for SIP, SIPS, and TEL URIs).

In addition, before performing the lookup match, the Oracle CSM strips characters that provide ease-of-reading separation. For example, if the Oracle CSM were to receive a req-uri containing tel:a-#1-781-328-5555, it would treat it as tel:a#17813285555.

SIP Routing

This section describes SIP session routing. When routing SIP call requests, the Oracle CSM communicates with other SIP entities, such as SIP user devices, other SIP proxies, and so on, to decide what SIP-based network resource each session should visit next. The Oracle CSM processes SIP call requests and forwards the requests to the destination endpoints to establish, maintain, and terminate real-time multimedia sessions.

Certain items in the messages are matched with the content of the local policy, within constraints set by the previous hop session agent, and the SIP configuration information (for example, carrier preferences) to determine a set of applicable next hop destinations.

The sending session agent is validated as either a configured session agent or a valid entry in a user cache. If the session INVITATION does not match any registering user, the SIP proxy determines the destination for routing the session INVITATION.

Limiting Route Selection Options for SIP

You can configure the local policy to use the single most-preferred route. And you can configure the SIP configuration max routes option to restrict the number of routes which can be selected from a local policy lookup:

- A max-routes=1 value limits the Oracle CSM to only trying the first route from the list of available preferred routes.
- A max-routes=0 value or no max-routes value configured in the options field allows the Oracle CSM to use all of the routes available to it.

About Loose Routing

According to RFC 3261, a proxy is loose routing if it follows the procedures defined in the specification for processing of the Route header field. These procedures separate the destination of the request (present in the Request-URI) from the set of proxies that need to be visited along the way (present in the Route header field).

When the SIP NAT's route home proxy field is set to enabled, the Oracle CSM looks for a session agent that matches the home proxy address and checks the loose routing field value. If the loose routing is:

- enabled—A Route header is included in the outgoing request in accordance with RFC 3261.
- disabled—A Route header is not included in the outgoing request; in accordance with the route processing rules described in RFC 2543 (referred to as strict routing). That rule caused proxies to destroy the contents of the Request-URI when a Route header field was present.

Whether loose routing field is enabled is also checked when a local policy 's next hop value matches a session agent. Matching occurs if the hostname or the session agent's IP address field value corresponds to the next hop value. If loose routing is enabled for the matching session agent, the outgoing request retains the original Request-URI and Route header with the next hop address.

About the Ingress Realm

You can create a list of realms in your local policy that is used by the Oracle CSM to determine how to route traffic. This list determines from which realm traffic is coming and is used for routing by ingress realm.

The source realm values must correspond to valid identifier entered when the realm was configured.

About the Egress Realm

An egress realm allows SIP signaling to travel out of the Oracle CSM through a network other than the home realm. The Oracle CSM uses egress realms for signaling purposes (when matching flows). When a packet arrives at the Oracle CSM with a destination address that does not match any defined session agents, the Oracle CSM uses the address associated with the realm that is, in turn, associated with the SIP configuration's egress realm ID, as the outgoing network. With the use of the egress realm ID, it is possible to define a default route for SIP requests addressed to destinations outside the home realm. If no egress realm is defined, the home realm (default ingress realm) is used as the default egress realm.

With session agent egress realm configured, the Oracle CSM adds a default egress realm to the session agent to identify the signaling interface used for ping requests. The Oracle CSM also uses the default egress realm when the normal routing request does not yield an egress realm—for example, when a local policy does not specify the next hop's realm.

When you configure session agents, you can define them without realms or you can wildcard the realm value. These are global session agents, and multiple signaling interfaces can reach them. Then, when you use session agent pinging, the Oracle CSM sends out ping requests using the signaling interface of the default egress realm defined in the global SIP configuration. The global session agents in certain environments can cause problems when multiple global session agents residing in multiple networks, some of which might not be reachable using the default SIP interface egress realm.

The Oracle CSM uses the session agent egress realm for ping messages even when the session agent has a realm defined. For normal request routing, the Oracle CSM uses the egress realm for global session agents when local policies or SIP-NAT bridge configurations do not point to an egress realm.

Ping Message Egress Realm Precedence

For ping messages, the egress realm precedence occurs in the following way (in order of precedence):

- Egress realm identified for the session agent.
- Session agent realm (set in the realm-id parameter) or the wildcarded value
- Global SIP configuration egress realm, when configured in the egress-realm parameter
- Global SIP configuration home realm

Normal Request Egress Realm Precedence

For normal request routing, the egress realm precedence occurs in the following way (in order of precedence):

- Egress SIP-NAT realm, when the route-home-proxy parameter is set to forced and no local policy match is found
- Matching local policy realm, when configured in the local policy attributes
- Session agent realm (set in the realm-id parameter) or the wildcarded value
- Session agent egress realm, when configured in the egress-realm-id parameter
- Global SIP configuration egress realm, when configured in the egress-realm parameter
- Global SIP configuration home realm

Session Agent Egress Realm Configuration

Configuring a session agent egress realm is optional.

To configure a session agent egress realm:

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

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

2. Type session-router and press Enter.

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

3. Type session-agent and press Enter.

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

If you are adding this feature to an existing configuration, you need to select the configuration (using the ACLI select command) before making your changes.

4. egress-realm-id—Enter the name of the realm you want defined as the default egress realm used for ping messages. The Oracle CSM will also use this realm when it cannot determine the egress realm from normal routing. There is no default value for this parameter.
5. Save and activate your configuration.

About SIP Redirect

SIP redirect involves proxy redirect and tunnel redirect.

Proxy Redirect

You can configure the SIP proxy mode to define how the SIP proxy will forward requests coming from the session agent. This value is used if the session agent's proxy mode has no value (is empty).

Tunnel Redirect

You can use tunnel redirect when requests are routed to a server behind a SIP NAT that sends redirect responses with addresses that should not be modified by the SIP NAT function. For example, a provider might wish to redirect certain calls (like 911) to a gateway that is local to a the UA that sent the request. Since the gateway address is local to the realm of the UA, it should not be modified by the SIP NAT of the server's realm. Note that the server must have a session agent configured with the redirect-action field set to the proxy option in order to cause the redirect response to be sent back to the UA.

SIP Method Matching and To Header Use for Local Policies

For SIP, this feature grants you greater flexibility when using local policies and has two aspects: basing local policy routing decisions on one or more SIP methods you configure and enabling the Oracle CSM to use the TO header in REGISTER messages for routing REGISTER requests.

SIP Methods for Local Policies

This feature allows the Oracle CSM to include SIP methods in routing decisions. If you want to use this feature, you set a list of one or more SIP methods in the local policy attributes. These are the SIP methods you can enter in the list: INVITE, REGISTER, PRACK, OPTIONS, INFO, SUBSCRIBE, NOTIFY, REFER, UPDATE, MESSAGE, and PUBLISH.

After the Oracle CSM performs a local policy look-up for SIP, it then searches for local policy attributes that have this methods list configured. If it finds a a set of policy attributes that matches a method that matches the traffic it is routing, the Oracle CSM uses that set of policy attributes. This means that the Oracle CSM considers first any policy attributes with methods configured before it considers those that do not have methods. In the absence of any policy attributes with methods, the Oracle CSM uses the remaining ones for matching.

In cases where it finds neither matching policy attributes with methods or matching policy attributes without them, the Oracle CSM either rejects the calls with a 404 No Routes Found (if the request calls for a response) or drops the call.

You configure local policy matching with SIP methods in the local policy attributes parameter calls methods. This parameter is a list that takes either one or multiple values. If you want to enter multiple values, you put them in the same command line entry, enclosed in quotation marks and separated by spaces.

To configure SIP methods for local policy matching:

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

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

2. Type session-router and press Enter.

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

3. Type local-policy and press Enter. If you are adding this feature to a pre-existing local policy configuration, you will need to select and edit your local policy.

```
ORACLE (session-router) # local-policy
ORACLE (local-policy) #
```

4. Type policy-attributes and press Enter. If you are adding this feature to a pre-existing local policy configuration, you will need to select and edit your local policy.

```
ORACLE(local-policy)) # policy-attributes
ORACLE(policy-attributes) #
```

5. **methods**—Enter the SIP methods you want to use for matching this set of policy attributes. Your list can include: INVITE, REGISTER, PRACK, OPTIONS, INFO, SUBSCRIBE, NOTIFY, REFER, UPDATE, MESSAGE, and PUBLISH.

By default, this parameter is empty—meaning that SIP methods will not be taken into consideration for routing based on this set of policy attributes.

If you want to enter more than one method, your entry will resemble the following example.

```
ACMEPACKET(local-policy-attributes) # methods "PRACK INFO REFER"
```

6. Save and activate your configuration.

Routing Using the TO Header

For the Oracle CSM's global SIP configuration, you can enable the use of an ENUM query to return the SIP URI of the Registrar for a SIP REGISTER message. Without this feature enabled, the Oracle CSM uses the REQUEST URI. This ability can be helpful because REGISTER messages only have the domain in the REQUEST URI, whereas the SIP URI in the To header contains the user's identity.

There are two parts to enabling this feature. First, you must enable the register-use-to-for-lp parameter in the global SIP configuration. Then you can set the next-hop in the applicable local policy attributes set to ENUM.

To enable your global SIP configuration for routing using the TO header:

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

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

2. Type `session-router` and press Enter.

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

3. Type `sip-config` and press Enter. If you are adding this feature to a pre-existing SIP configuration, you will need to select and edit it.

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

4. **register-use-to-for-lp**—Set this parameter to `enabled` if you want the Oracle CSM to use, for routing purposes, an ENUM query to return the SIP URI of the Registrar for a SIP REGISTER message. This parameter defaults to `disabled`.

To set up your local policy attributes for routing using the TO header:

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

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

6. Type `session-router` and press Enter.

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

7. Type `local-policy` and press Enter. If you are adding this feature to a pre-existing local policy configuration, you will need to select and edit your local policy.

```
ORACLE(session-router) # local-policy
ORACLE(local-policy) #
```

8. Type `policy-attributes` and press Enter. If you are adding this feature to a pre-existing local policy configuration, you will need to select and edit your local policy.

```
ORACLE(local-policy) # policy-attributes
ORACLE(policy-attributes) #
```

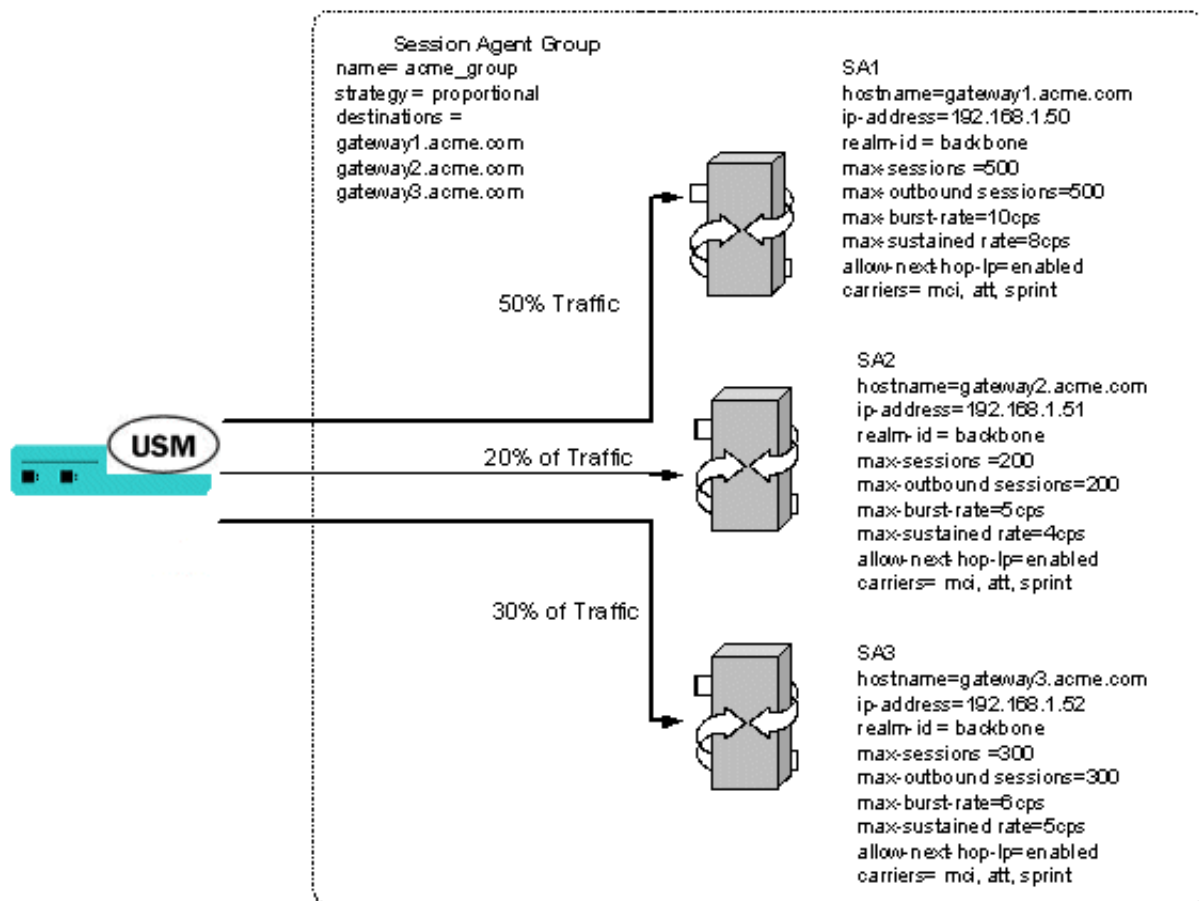
Routing with Local Policy

- next-hop—This is the next signaling host. Set this parameter to ENUM if you want to use SIP methods in local policy attribute information for routing purposes.
- Save and activate your configuration.

Load Balancing

This section describes Oracle CSM load balancing. You can use session agent groups to assist in load balancing among session agents. You define concurrent session capacity and rate attributes for each session agent and then define the session agent group. Next, you select the allocation strategy you want applied to achieve the load balancing you want.

The following example shows a configuration for load balancing gateways based on a proportional allocation strategy.



Routing and load balancing capabilities include the following:

- least cost, which includes cost-based and time-based routing
- customer preference
- traffic aggregation
- routing by media (codec) type
- capacity-based, by destination
- service element load balancing
- service element failure detection and re-route
- session agent failure
- routing by codec

Configuring Routing

This section explains how to configure routing on the Oracle CSM.

Configuration Prerequisite

You should have already configured the realms for your environment before you configure the routing elements. You need to know the realm identifier when configuring session agents and local policy.

You can use an asterisk (*) when the session agent exists in multiple realms.

Configuration Order

Recommended order of configuration:

- realm
- session agent
- session agent group
- local policy

Routing Configuration

You can enable, then configure, individual constraints that are applied to the sessions sent to the session agent. These constraints can be used to regulate session activity with the session agent. In general, session control constraints are used for session agent groups or SIP proxies outside or at the edge of a network. Some individual constraints, such as maximum sessions and maximum outbound sessions are not applicable to core proxies because they are transaction stateful, instead of session stateful. Other constraints, such as maximum burst rate, burst rate window, maximum sustained rate, and sustained rate are applicable to core routing proxies.

Configuring Session Agents

To configure session agents:

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 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. host-name—Enter the name of the host associated with the session agent in either hostname or FQDN format, or as an IP address.

If you enter the host name as an IP address, you do not have to enter an IP address in the optional IP address parameter. If you enter the host name in FQDN format, and you want to specify an IP address, enter it in the optional IP address parameter. Otherwise you can leave the IP address parameter blank to allow a DNS query to resolve the host name.

If the initial DNS query for the session agent fails to get back any addresses, the session agent is put out-of-service. When session agent is pinged, the DNS query is repeated. The ping message is not sent until the DNS query gets back one or more IP addresses. After the query receives some addresses, the ping message is sent. The session agent remains out of service until one of the addresses responds.



Note: The value you enter here must be unique to this session agent. No two session agents can have the same hostname.

The hostnames established in the session agent populate the corresponding fields in other elements.

Routing with Local Policy

5. ip-address—Optional. Enter the IP address for the hostname you entered in FQDN format if you want to specify the IP address. Otherwise, you can leave this parameter blank to allow a DNS query to resolve the host name.
6. port—Enter the number of the port associated with this session agent. Available values include:
 - zero (0)—If you enter zero (0), the Oracle CSM will not initiate communication with this session agent (although it will accept calls).
 - 1025 through 65535

The default value is 5060.



Note: If the transport method value is TCP, the Oracle CSM will initiate communication on that port of the session agent.

7. state—Enable or disable the session agent by configuring the state. By default, the session agent is enabled.
 - enabled | disabled
8. transport-method—Indicate the IP protocol to use (transport method) to communicate with the session agent. UDP is the default value. The following protocols are supported:
 - UDP—Each UDP header carries both a source port identifier and destination port identifier, allowing high-level protocols to target specific applications and services among hosts.
 - UDP+TCP—Allows an initial transport method of UDP, followed by a subsequent transport method of TCP if and when a failure or timeout occurs in response to a UDP INVITE. If this transport method is selected, INVITEs are always sent through UDP as long as a response is received.
 - DynamicTCP—dTCP indicates that dynamic TCP connections are the transport method for this session agent. A new connection must be established for each session originating from the session agent. This connection is torn down at the end of a session.
 - StaticTCP—sTCP indicates that static TCP connections are the transport method for this session agent. Once a connection is established, it remains and is not torn down.
 - DynamicTLS—dTLS indicates that Dynamic TLS connections are the transport method for this session agent. A new connection must be established for each session originating from the session agent. This connection is torn down at the end of a session.
 - StaticTLS—sTLS indicates that Static TLS connections are the transport method for this session agent. Once a connection is established, it will remain and not be torn down.
9. realm-id—Optional. Indicate the ID of the realm in which the session agent resides.

The realm ID identifies the realm for sessions coming from or going to this session agent. For requests coming from this session agent, the realm ID identifies the ingress realm. For requests being sent to this session agent, the realm ID identifies the egress realm. In a Oracle CSM, when the ingress and egress realms are different, the media flows must be steered between the realms.

- no value: the egress realm is used unless the local policy dictates otherwise
- asterisk (*): keep the egress realm based on the Request URI



Note: The realm ID you enter here must match the valid identifier value entered when you configured the realm.

10. description—Optional. Enter a descriptive name for this session agent.
11. carriers—Optional. Add the carriers list to restrict the set of carriers used for sessions originating from this session agent.

Carrier names are arbitrary names that can represent specific service providers or traditional PSTN telephone service providers (for sessions delivered to gateways). They are global in scope, especially if they are exchanged in TRIP. Therefore, the definition of these carriers is beyond the scope of this documentation.

You could create a list using carrier codes already defined in the North American Numbering Plan (NANP); or those defined by the local telephone number or carrier naming authority in another country.



Note: If this list is empty, any carrier is allowed. If it is not empty, only local policies that reference one or more of the carriers in this list will be applied to requests coming from this session agent.

12. allow-next-hop-ip—Indicate whether this session agent can be used as a next hop in the local policy.

If you retain the default value of enabled, the session agent can be used as the next hop for the local policy. Valid values are:

- enabled | disabled

13. constraints—Enable this parameter to indicate that the individual constraints you configure in the next step are applied to the sessions sent to the session agent. Retain the default value of disabled if you do not want to apply the individual constraints. Valid values are:

- enabled | disabled



Note: In general, session control constraints are used for SAGs or SIP proxies outside or at the edge of a network.

14. Enter values for the individual constraints you want applied to the sessions sent to this session agent. The following table lists the available constraints along with a brief description and available values.

Constraint	Description
maximum sessions	<p>Maximum number of sessions (inbound and outbound) allowed by the session agent. The range of values is:</p> <ul style="list-style-type: none"> • minimum: zero (0) is the default value and means there is no limit • maximum: 4294967295
maximum outbound sessions	<p>Maximum number of simultaneous outbound sessions (outbound from the Oracle CSM) that are allowed from the session agent. The range of values is:</p> <ul style="list-style-type: none"> • minimum: zero (0) is the default value and means there is no limit • maximum: 4294967295 <p>The value you enter here cannot be larger than the maximum sessions value.</p>
maximum burst rate	<p>Number of session invitations allowed to be sent to or received from the session agent within the configured burst rate window value. SIP session invitations arrive at and leave from the session agent in intermittent bursts. By entering a value in this field, you can limit the amount of session invitations that are allowed to arrive at and leave from the session-agent.</p> <p>For example, if you enter a value of 50 here and a value of 60 (seconds) for the burst rate window constraint, no more than 50 session invitations can arrive at or leave from the session agent in that 60 second time frame (window). Within that 60-second window, any sessions over the limit of 50 are rejected.</p> <p>The range of values is:</p> <ul style="list-style-type: none"> • minimum: zero (0) session invitations per second • maximum: 4294967295 session invitations per second <p>Zero is the is the default value.</p>
maximum sustain rate	<p>Maximum rate of session invitations (per second) allowed to be sent to or received from the session agent within the current window. The current rate is determined by counting the number of session invitations processed within a configured time period and dividing that number by the time period. By entering a value in this field, you can limit the amount of session invitations that are allowed to arrive at and leave from the session agent over a sustained period of time.</p>

Routing with Local Policy

Constraint	Description
	<p>For the sustained rate, the Oracle CSM maintains a current and previous window size. The period of time over which the rate is calculated is always between one and two window sizes.</p> <p>For example, if you enter a value of 5000 here and a value of 3600 (seconds) for the sustain rate window constraint, no more than 5000 session invitations can arrive at or leave from the session agent in any given 3600 second time frame (window). Within that 3600-second window, sessions over the 5000 limit are rejected.</p> <p>The range of values is:</p> <ul style="list-style-type: none"> • minimum: zero (0) invitations per second • maximum: 4294967295 invitations per second <p>Zero is the is the default value.</p> <p>The value you set here must be larger than the value you set for the maximum burst rate constraint.</p>
time to resume	<p>Time in seconds after which the SIP proxy resumes sending session invitations to this session agent. This value only takes effect when the SIP proxy stops sending invitations because a constraint is exceeded.</p> <p>The range of values is:</p> <ul style="list-style-type: none"> • minimum: zero (0) seconds • maximum: 4294967295 seconds <p>Default is zero.</p>
time to resume (ttr) no response	<p>Delay in seconds that the SIP proxy must wait from the time that it sends an invitation to the session agent and gets no response before it tries again.</p> <p>The range of values is:</p> <ul style="list-style-type: none"> • minimum: zero (0) seconds • maximum: 4294967295 seconds <p>Default is zero.</p> <p>The value you enter here must be larger than the value you enter for the time to resume constraint.</p>
in service period	<p>Amount of time in seconds the session agent must be operational (once communication is re-established) before the session agent is declared as being in-service (ready to accept session invitations). This value gives the session agent adequate time to initialize.</p> <p>The range of values is:</p> <ul style="list-style-type: none"> • minimum: zero (0) seconds • maximum: 4294967295 seconds <p>Default is zero.</p>
burst rate window	<p>Burst window period (in seconds) that is used to measure the burst rate. The term window refers to the period of time over which the burst rate is computed. Refer to the maximum burst rate information.</p> <p>The range of values is:</p>

Constraint	Description
	<ul style="list-style-type: none"> • minimum: zero (0) seconds • maximum: 4294967295seconds <p>Zero is the is the default value.</p> <p>The value you set here must be smaller than the value you set for the maximum burst rate constraint.</p>
sustain rate window	<p>Sustained window period (in seconds) that is used to measure the sustained rate. Refer to the maximum sustain rate information.</p> <p>The range of values is:</p> <ul style="list-style-type: none"> • minimum: zero (0) seconds • maximum: 4294967295seconds <p>Zero is the is the default value.</p> <p>The value you set here must be larger than the value you set for the maximum sustain rate constraint.</p>

15. req-uri-carrier-mode—SIP only. Set whether you want the selected carrier (determined by a value in the local policy) added to the outgoing message by configuring the request uri carrier mode parameter.

You can set this parameter to let the system perform simple digit translation on calls sent to gateways. A 3-digit prefix is inserted in front of the telephone number (the Request-URI) that the gateway will use to select a trunk group. Most often, the Oracle CSM needs to insert the carrier code into the signaling message that it sends on.

The default value is none. The following lists the available modes.

- none—Carrier information will not be added to the outgoing message.
- uri-param—Adds a parameter to the Request-URI. For example, cic-XXX.
- prefix—Adds the carrier code as a prefix to the telephone number in the Request-URI (in the same manner as PSTN).

16. proxy-mode—SIP only. Indicate the proxy mode to use when a SIP request arrives from this session agent.

If this field is empty (upon initial runtime or upgrade), it's value is set to the value of the SIP configuration's proxy mode by default. If no proxy mode value was entered for the SIP configuration, the default for this field is proxy.

The following are valid proxy modes:

- proxy—If the Oracle CSM is a Session Router, the system will proxy the request coming from the session agent and maintain the session and dialog state. If the Oracle CSM is a Session Director, the system behaves as a B2BUA when forwarding the request.
- redirect—The system sends a SIP 3xx reDIRECT response with contacts (found in the local policy) to the previous hop.

17. redirect-action—SIP only. Indicate the action you want the SIP proxy to take when it receives a Redirect (3XX) response from the session agent.

The following table lists the available proxy actions along with a brief description

- recurse-305-only—(default) If the Oracle CSM receives a 305 SIP redirect response (Use Proxy) on the SIP interface, it automatically redirects all requests to the Contact URI specified in the 305 response. All other 3xx responses are sent back to the previous hop.
- proxy—The SIP proxy passes the response back to the previous hop; based on the proxy mode of the original request.
- recurse—The SIP proxy serially sends the original request to the list of contacts in the Contact header of the response (in the order in which the contacts are listed in the response). For example, if the first one fails, the

request will be sent to the second, and so on until the request succeeds or the last contact in the Contact header has been tried.

18. **loose-routing**—SIP only. Enable this parameter if you want to use loose routing (as opposed to strict routing). The default is enabled. Valid values are:

- enabled | disabled


When the SIP NAT route home proxy parameter is enabled, the Oracle CSM looks for a session agent that matches the home proxy address and checks the loose routing value. If loose routing is enabled, a Route header is included in the outgoing request in accordance with RFC 3261. If loose routing is disabled, the Route header is not included in the outgoing request (in accordance with strict routing procedures defined in RFC 2543).

The loose routing value is also checked when the local policy's next hop value matches a session agent. If loose routing is set to enabled, the outgoing request retains the original Request-URI and Route header with the next hop address.

19. **send-media-session**—SIP only. Enable this parameter if you want to include a media session description (for example, SDP) in the INVITE or REINVITE message sent by the Oracle CSM. Setting this field to disabled prevents the Oracle CSM from establishing flows for that INVITE message.

The default is enabled. Valid values are:

- enabled | disabled

 **Note:** Only set send media session to disabled for a session agent that always redirects requests. It returns an error or 3xx response instead of forwarding an INVITE message.

20. **response-map**—Optional and for SIP only. Enter the name of the response map to use for this session agent. The mappings in each SIP response map is associated with a corresponding session agent. You can also configure this value for individual SIP interfaces.

21. **ping-method**—SIP only. Indicate the SIP message/method to use to ping a session agent. The ping confirms whether the session agent is in service. If this field is left empty, no session agent will be pinged.

Setting this field value to the OPTIONS method might produce a lengthy response from certain session agents and could potentially cause performance degradation on your Oracle CSM.

22. **ping-interval**—SIP only. Indicate how often you want to ping a session agent by configuring the ping interval parameter. Enter the number of seconds you want the Oracle CSM to wait between pings to this session agent. The default value is 0. The valid range is:

- Minimum: 0
- Maximum: 999999999

The Oracle CSM only sends the ping if no SIP transactions (have occurred to/from the session agent within the time period you enter here.

23. **trunk-group**—Enter up to 500 trunk groups to use with this single session agent. Because of the high number of trunk groups you can enter, the CLI provides enhanced editing mechanisms for this parameter:

- You use a plus sign (+) to add single or multiple trunk groups to the session agent's list.

When you add a single trunk group, simply use the plus sign (+) in front of the trunk group name and context. Do not use a Space between the plus sign and the trunk group name and context.

For example, you might have already configured a list of trunk groups with the following entries: tgrpA:contextA, tgrpB:contextB, and tgrpC:contextC. To add tgrp1:context1, you would make the following entry:

```
ORACLE (session-agent) # trunk-group +tgrp1:context1
```

Your list would then contain all four trunk groups.

When you add multiple trunk groups, simply enclose your entry in quotation marks (") or in parentheses (()). While you put spaces between the trunk group name and context entries, you do not use spaces with the plus sign, parentheses or quotation marks.

```
ORACLE(session-agent) # trunk-group +tgrp1:context1 tgrp2:context2
tgrp3:context3
```

- You use a minus sign (-) to delete single or multiple trunk groups from the session agent's list.

When you remove a single trunk group, simply use the minus sign (-) in front of the trunk group name and context. Do not use a Space between the minus sign and the trunk group name and context.

For example, you might have already configured a list of trunk groups with the following entries: tgrpA:contextA, tgrpB:contextB, tgrpC:contextC, and tgrp1:context1. To delete tgrp1:context1 from the list, you would make the following entry:

```
ORACLE(session-agent) # trunk-group -tgrp1:context1
```

Your list would then contain: tgrpA:contextA, tgrpB:contextB, and tgrpC:contextC.

When you add multiple trunk groups, simple enclose your entry in quotation marks (") or in parentheses (()). While you put spaces between the trunk group name and context entries, you do not use spaces with the plus sign, parentheses or quotation marks.

```
ORACLE(session-agent) # trunk-group -tgrp1:context1 tgrp2:context2
```

- You overwrite (replace) the entire list of a session agent's trunk groups by entering a list that does not use either the plus (+) or the minus (-) sign syntax.

24. ping-in-service-response-codes—SIP only. Enter the list of response codes that keep a session agent in service when they appear in its response to the Oracle CSM's ping request. The Oracle CSM takes the session agent out of service should a response code be used that does not appear on this list. Default is none.
25. out-service-response-codes—SIP only. Enter the list defines the response codes that take a session agent out of service when they appear in its response to the Oracle CSM's ping request or any in-dialog creating request (such as an INVITE, SUBSCRIBE, etc.). The Oracle CSM ignores this list if an in-service list exists.
26. options—Optional. You can add your own features and/or parameters by using the options parameter. You enter a comma-separated list of either or both of the following:


- feature=<value feature>

For example:

You can include the original address in the SIP message from the Oracle CSM to the proxy in the Via header parameter by entering the following option:

```
via-origin=<parameter-name>
```

The original parameter is included in the Via of the requests sent to the session agent. The via origin feature can take a value that is the parameter name to include in the Via. If the value is not specified for via origin, the parameter name is origin.

 **Note:** If the feature value itself is a comma-separated list, enclose it within quotation marks.

27. in-translationid—Optional. Enter the In Translation ID for a configured session translation (group of address translation rules with a single ID) if you want to apply session translation to incoming traffic.
28. out-translationid—Optional. Enter the Out Translation ID for a configured session translation (group of address translation rules with a single ID) if you want to apply session translation to outgoing traffic.

Address translations attached to session agents take precedence over address translations attached to realms. If no address translation is applied to a session agent, then the Oracle CSM will use the address translation applied to a realm. If an address translation is applied to both a realm and session agent, the translation attached to the session agent will apply. If the applicable session agent and realm have no associated translations, then the addresses will remain in their original forms and no address translations will be performed.

29. trust-me—Indicate whether this session agent is a trusted source, which the Oracle CSM checks when it receives a message to determine if the source is trusted. The default value is disabled. The valid values are:

- enabled | disabled

The following example shows a session agent with an IP address used for the hostname.

Routing with Local Policy

```
session-agent
  hostname          192.168.1.10
  ip-address        192.168.1.10
  port              5060
  state             enabled
  app-protocol      SIP
  app-type
  transport-method  UDP
  realm-id          realm-1
  description       englab
  carriers
                    carrier1
  allow-next-hop-lp enabled
  constraints       disabled
  max-sessions      355
max-inbound-sessions 4
  max-outbound-sessions 355
  max-burst-rate    0
  max-inbound-burst-rate 10
  max-outbound-burst-rate 1
  max-sustain-rate 3000
  max-inbound-sustain-rate 0
  max-outbound-sustain-rate 0
  min-seizures      5
  min-asr           0 time-to-resume 60
  ttr-no-response   0
  in-service-period 30
  burst-rate-window 60
  sustain-rate-window 3600
  req-uri-carrier-mode None
  proxy-mode        Proxy
  redirect-action    Recurse
  loose-routing     enabled
  send-media-session enabled
  response-map
  ping-method
  ping-interval     0
  media-profiles
  in-translationid
  out-translationid
  trust-me          disabled
  request-uri-headers
  stop-recurse
  local-response-map
  ping-to-user-part
  ping-from-user-part
  li-trust-me       disabled
  in-manipulationid
  out-manipulationid
  p-asserted-id
  trunk-group
  max-register-sustain-rate 0
```

Configuring Session Agent Groups

To configure session agent 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 `session-group` and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

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

4. `group-name`—Enter a unique name for the session agent group in Name format.
5. `description`—Optional. Enter descriptive information about the session agent group.
6. `state`—Enable or disable the session agent group on the Oracle CSM. The default value is enabled. Valid values are:
7. `strategy`—Indicate the session agent allocation strategy you want to use. The strategy you chose selects the session agents that will be made available by this session agent group. The default value is hunt. The valid values are:

- enabled | disabled

- `hunt`—Selects session agents in the order in which they are listed. For example, if the first agent is online, working, and has not exceeded defined constraints, all traffic is sent to the first agent. If the first agent is offline or if it exceeds a defined constraint, the second agent is selected. If the first and second agents are offline or exceed defined constraints, the third agent is selected. And so on through the list of session agents.
- `roundrobin`—Selects each session agent in the order in which they are listed in the destination list, selecting each agent in turn, one per session.
- `leastbusy`—Selects the session agent that has the fewest number of sessions relative to the maximum sessions constraint (for example, lowest percent busy) of the session agent element. The Least Busy Calculation is the result of dividing the number of active calls for a session agent by the `max-sessions` parameter within the session-agent element configuration. If the default `max-session` parameter value issued for a session agent (0), the result of the Least Busy Calculation will be 0. The Least Busy SAG Strategy will route a session to the session agent with the lowest resulting Least Busy Calculation percentage. If multiple session agents have the lowest percentage, the foremost session agent in the `Session Agent Group dest` parameter will be used.
- `propdist`—Based on programmed, constrained session limits, the Proportional Distribution strategy proportionally distributes the traffic among all of the available session agents. Sessions are distributed among session agents based on the `max-outbound-sessions` value in each session agent. The sum of `max-outbound-sessions` for every session-agent within a session group equates to 100% and the `max-outbound-sessions` value for each session-agent represents a % that total. Sessions are proportionally allocated to session agents based on their individual session agent `max-outbound-sessions` value, as a % of the total `max-outbound-sessions` for the group.
- `lowsusrate`—The Lowest Sustained Rate strategy routes to the session agent with the lowest sustained rate of session initiations/invitations (based on observed sustained session request rate).

8. `destination`—Identify the destinations (session agents) available for use by this session agent group.

A value you enter here must be a valid IP address or hostname for a configured session agent.

9. `trunk-group`—Enter trunk group names and trunk group contexts to match in either IPTTEL or custom format. If left blank, the Oracle CSM uses the trunk group in the realm for this session agent group. Multiple entries are surrounded in parentheses and separated from each other with spaces.

Entries for this list must one of the following formats: `trgp:context` or `trgp.context`.

10. `sag-recursion`—Enable this parameter if you want to use SIP SAG recursion for this SAG. The default value is disabled. Valid values are:

- enabled | disabled

11. `stop-sag-recurse`—Enter the list of SIP response codes that terminate recursion within the SAG. Upon receiving one of the specified response codes, such as 401 unauthorized, or upon generating one of the specified response codes internally, such as 408 timeout, the Oracle CSM returns a final response to the UAC. You can enter the response codes as a comma-separated list or as response code ranges.

The following example shows a session agent group using the SIP protocol and with the Hunt allocation strategy applied.

```
session-group
group-name                proxy-sag1
```

Routing with Local Policy

```
description          proxies for external domain
state                enabled
app-protocol         SIP
strategy             Hunt
dest                 gw1
                    gw2

trunk-group
tgname2:tgcontext2
sag-recursion        disabled
stop-sag-recurse    401,407
last-modified-date  2005-01-09 23:23:36
```

SAG Matching for LRT and ENUM

When this feature is enabled and a match is found, the Oracle CSM uses the matching SAG for routing. When there is no match for the SAG, the Oracle CSM processes the result as it would have if this feature had not been enabled: either matching to a session agent hostname, or performing a DNS query to resolve it.

Note that you set the state of this feature in the SIP configuration.

To configure a SAG for ENUM or LRT matching:

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

```
ORACLE# configure terminal
```

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

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

If you are adding support for this feature to a pre-existing SIP configuration, then you must select (using the ACLI `select` command) that configuration to edit it.

4. `enum-sag-match`—Set this parameter to `enabled` so the Oracle CSM will match session agent group (SAG) names with the hostname portion in the naming authority pointer (NAPTR) from an ENUM query or LRT next-hop entry. The default value is `disabled`. The valid values are:

- `enabled` | `disabled`

5. Save and activate your configuration.

Configuring Local Policy

To configure local policy:

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

```
ACMEPACKET# configure terminal
```

2. Type `session-router` and press Enter.

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

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

```
ACMEPACKET(session-router)# local-policy
ACMEPACKET(local-policy)#
```


4. `from-address`—Indicate the originating address information by entering a From address value. You can use the asterisk (*) as a wildcard to indicate this policy can be used with all originating addresses.

You can also use complete or partial E.164 addresses (strings that contain telephone keypad characters) here. Number matching works from left to right. Formats include the following:

- SIP From address
- FQDNs
- IP addresses
- 123*456

The Oracle CSM also supports the asterisk as part of the From address you configure in your local policies.

This means that for the from-address parameters of a local policy configuration, you can enter values in which an asterisk appears and match them accordingly. You might enter a value that resemble the following example:

 **Note:** After entering the from-address value, the Oracle CSM automatically saves it to the configuration when exiting from localpolicy.


5. to-address—Indicate the destination address by entering a To address value. You can use the asterisk (*) as a wildcard to indicate all this policy can be used for any destination address.

You can also use E.164 addresses (strings that contain telephone keypad characters) here. Number matching works from left to right. Formats include the following:

- SIP Request-URI
- FQDNs
- IP addresses
- 123*456

The Oracle CSM also supports the asterisk as part of the To address you configure in your local policies.

This means that for the to-address parameters of a local policy configuration, you can enter values in which an asterisk appears and match them accordingly. You might enter a value that resembles the following example:

 **Note:** After entering the to-address value, the Oracle CSM automatically saves it to the configuration when exiting from localpolicy.

6. source-realm—Enter the realm, or list of realms, you want the Oracle CSM to use to determine how to route traffic. This list identifies from what realm traffic is coming and is used for routing by ingress realm by the local policy.

You can use the asterisk (*) as a wildcard to indicate this local policy can be used with all realms. The default value is *.Or you can enter a value that corresponds to the identifier of an already configured realm. Formats include the following:

- realm ID
- customer name
- peer name
- subdomain name
- VPN identifier

7. activate-time—Set the time you want the local policy to be activated using the following syntax:

```
yyyy:mm:dd hh:mm:ss
yyyy:mm:dd-hh:mm:ss
```

8. deactivate-time—Set the time you want the local policy to be deactivated using the following syntax:

```
yyyy:mm:dd hh:mm:ss
yyyy:mm:dd-hh:mm:ss
```

9. state—Indicate whether you want the local policy to be enabled or disabled on the system. The default value is enabled. The valid values are:

- enabled | disabled

10. policy-attribute—Configure local policy attributes by following steps 8 through 21.

11. next-hop—Identify the next signaling host by entering the next hop value. You can use the following as next hops:

- IPv4 address or IPv6 address of a specific endpoint
- Hostname or IPv4 address or IPv6 address of a configured session agent

Routing with Local Policy

- Group name of a configured session agent group



Note: The group name of a configured session agent group must be prefixed with SAG:

For example:

policy-attribute

next-hop SAG:appserver

policy-attribute

next-hop lrt:routetable

policy-attribute

next-hop enum:lrg

You can also configure a next hop that has an address of 0.0.0.0, thereby creating a null route. Different from not having a local policy configured (which would trigger Oracle CSM local policy recursion), this terminates local policy recursion and immediately fails the request. In these cases, the Oracle CSM responds a request with a 404 Not Found.

12. realm—Identify the egress realm (the realm used to reach the next hop) if the Oracle CSM must send requests out from a specific realm.

The value you enter here must correspond to a valid identifier you enter when you configured the realm. If you do not enter a value here, and the next hop is a session agent, the realm identified in the session agent configuration is used for egress.

13. replace-uri—Indicate whether you want to replace the Request-URI in outgoing SIP requests with the next hop value.

14. carrier—Optional. Enter the name of the carrier associated with this route. The value you enter here must match one or more of the carrier names in the session agent configuration.

Entries in carrier fields can be from 1 to 24 characters in length and can consist of any alphabetical character (Aa-Zz), numerical character (0-9), or punctuation mark (! " # \$ % ^ & * () + - = < > ? ' | { } [] @ / \ ' ~ , . _ : ;) or any combination of alphabetical characters, numerical characters, or punctuation marks. For example, both 1-0288 and acme_carrier are valid carrier field formats.

15. start-time—Indicate the time of day (from the exact minute specified) the local policy attributes go into effect. Enter only numerical characters (0-9) and follow the 4-digit military time format. For example:

1400

The default value of 0000 implies that the defined policy attributes can be considered in effect any time after 00:00:00. The valid range is:

- Minimum—0000
- Maximum—2400

16. end-time—Indicate the time of day (from the exact minute specified) the local policy attributes are no longer in effect. Enter only numerical characters (0-9) and follow the 4-digit military time format. For example:

2400

The default value of 2400 implies that the defined policy attributes can be considered in effect any time before midnight. The valid range is:

- Minimum—0000
- Maximum—2400

17. days-of-week—Enter any combination of days of the week (plus holidays) you want the local policy attributes to be in effect. You must enter at least one day or holiday here. A holiday entry must correspond with a configured holiday established in the Session Router.

The default is U-S. The valid values are:

- U (Sunday)
- M (Monday)
- T (Tuesday)
- W (Wednesday)
- R (Thursday)
- F (Friday)
- S (Saturday)
- H (Holiday)

You can enter a range of values separated by a hyphen, for example U-S. And you can enter multiple values separated by commas, for example M,W,F. You cannot use spaces as separators.

- 18. cost**—Enter a cost value that acts as a unitless representation of the cost of a route relative to other routes reaching the same destination (To address). This value is used as a way of ranking policy attributes.

The default value is zero (0). The valid values are:

- minimum—zero (0)
- maximum—999999999

- 19. state**—Indicate whether you want to enable or disable the local policy. The default value is enabled. The valid values are:

- enabled | disabled

- 20. media-profiles**—Configure a list of media profiles if you want the local policy to route SIP traffic by the codecs specified in the SDP. The list of media profiles entered here are matched against the SDP included in SIP requests and the next hop is selected by codec.

The values in this list are matched against the `rtpmap` attribute of passed SDP, and preference weight for route selection is based on the order in which the matching payload type appears in the SDP's `m=` line.

For example when the following SDP arrives:

```
m=audio 1234 RTP/AVP 0 8 18
```

that contains the following attributes that correspond to three configured local policies with the same cost:

- `a=rtpmap:0 PCMU/8000`
- `a=rtpmap:8 PCMA/8000`
- `a=rtpmap:18 G729/8000`

the following route selection action occurs:

The local policy route that corresponds to the `a=rtpmap:0 PCMU/8000` attribute is selected because the payload type of 0 in the attribute line matches the first payload type of 0 listed in the `m=` line. The codec value of PCMU indicated in this selected attribute is used to find the local policy with the media profiles attribute that includes PCMU in the list.

Because the value you enter here is matched against the codec values included in the actual passed SDP, it must correspond to accepted industry-standard codec values.

The following example shows a local policy with a next hop value of the session agent group called `gw-sag2`.

```
local-policy
  from-address          *
  to-address            192.168.1.10
  source-realm          *
  activate-time         2005-01-20 20:30:00
  deactivate-time       N/A
  state                 enabled
  last-modified-date    2005-01-10 00:36:29
policy-attribute
  next-hop              SAG:gw-sag2
```

```
realm
replace-uri          enabled
carrier
start-time           0000
end-time              2400
days-of-week         U-S
cost                  0
app-protocol
state                enabled
media-profiles
```

Local Policy Matching for Parent Realms

You can configure the Oracle CSM to use the parent realm for routing purposes even when the source realm for an incoming message is a child realm.

With this feature disabled (default), the Oracle CSM uses the specific source realm to perform a local policy look-up. When the source realm is a child realm and any relevant local policies are configured with the parent realm, there will be no matches and the local policy look-up will fail. To avoid this issue and ensure successful look-ups, you must configure multiple local policies if you want to use a configuration with nested realms.

The Oracle CSM examines the source realm to determine if it is a parent realm with any child realms when you enable this feature. If the parent, source realm does have child realms, then the Oracle CSM creates local policy entries for the parent and all of its child realms. This operation is transparent and can save time during the configuration process.

It is possible, then, for a local policy look-up to match the same child realm in two ways:

- Through a match via the parent realm
- Through a direct match for a local policy configured with that specific child realm

In such a case, the child realm must have different costs for each type of match to avoid collisions.

This feature is enabled on a global basis in the session router configuration. Because it applies system-wide, all source realms will use this form of matching when enabled.

To enable local policy matching for parent realms:

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

```
ORACLE# configure terminal
```

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

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

3. Type session-router and press Enter.

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

4. match-lp-source-parent-realms—If you want the Oracle CSM to perform local policy realm matching based on the parent realm (so that there are local policy entries for parent and child realms), set this parameter to enabled. The default value is disabled. The valid values are:

- enabled | disabled

```
ORACLE(session-router-config)# match-lp-src-parent-realms enabled
```

5. Save and activate your configuration.

SIP Session Agent DNS-SRV Load Balancing

Prior to Release 6.2.0 the Oracle CSM provided the ability to specify an FQDN (fully qualified domain name) for a destination session-agent. During DNS lookup the FQDN could resolve to multiple SRV (Resource Record for Servers) records. Each SRV could resolve to a single IP address via A-Record query in IMS or DNS.

With Release 6.2.0 the Oracle CSM supports load balancing behavior as described in RFC 3263, Session Initiation Protocol (SIP): Locating SIP Servers.

The Oracle CSM will provide a new parameter ping-all-addresses in session-agent configuration mode to enable internal load balancing and RFC 3263 compliance. The Oracle CSM monitor the availability of the dynamically resolved IP addresses obtained from DNS server using OPTIONS ping (ping-per-DNS entry). The ping-method and ping-interval for each resolved IP addresses will be copied from original session-agent.

Status of Session-Agent:

In Service – if any of dynamically resolved IP addresses is in service

Out of service – if all dynamically resolved IP addresses is out of service.

The default of ping-all-addresses is disabled, in which case the Oracle CSM only pings the first available resolved IP addresses.

With status of each resolved IP addresses above, the Oracle CSM will recurse through the list of these in-service IP addresses dynamically resolved from DNS server on 503 response, and stop recursion based upon a configured list of response values specified by the stop-recuse parameter in sip-interface configuration mode. With internal load balancing enabled in the session-agent, the Oracle CSM provides the ability to select routing destinations based on SRV weights. The priority/weight algorithm is based on RFC 2782, *A DNS RR for specifying the location of services (DNS SRV)*.

The Oracle CSM will provide the similar functionality as that listed above for A-records, the Oracle CSM will select first available routing destinations because there is no priority/weight contained in A-records.

Session Agent DNS-SRV Load Balancing Configuration

To configure the Oracle CSM to perform Session-Agent DNS-SRV load balancing:

1. From superuser mode, use the following command sequence to access sip-config configuration mode. While in this mode, you configure SAG-based address resolution.

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

2. Use the ping-all-addresses parameter to enable Session-Agent DNS-SRV load balancing.
3. Use done, exit, and verify-config to complete Session-Agent DNS-SRV load balancing configuration.

The show agents CLI command displays the availability of dynamically resolved IP addresses

```
ORACLE# show sip agents acme.engr.com
21:46:05-51-router
Session Agent acme.engr.com(core) [In Service] NO ACTIVITY
Session Agent acme.hxu.com(core) [In Service] NO ACTIVITY
Destination: 192.168.200.235 In Service
Destination: 192.168.200.231 In Service
...
...
```

Answer to Seizure Ratio-Based Routing

New SIP session agent constraints set a threshold for Answer to Seizure Ratio (ASR) has been implemented. ASR is considered when determining whether session agents are within their constraints to route calls (in addition to session and rate constraints).

The new session agent constraints indicate the minimum acceptable ASR value and computes the ASR while making routing decisions. ASR is calculated by taking the number of successfully answered calls and dividing by the total number of calls attempted (which are known as seizures).

Routing with Local Policy

If the ASR constraints are exceeded, the session agent goes out of service for a configurable period of time and all traffic is routed to a secondary route defined in the local policy (next hop with higher cost).

The two session agent constraints are:

- **minimum seizure:** determines if the session agent is within its constraints. When the first call is made to the session agent or the if calls to the session agent are not answered, the minimum seizure value is checked.

For example, if 5 seizures have been made to the session agent and none of them have been answered, the sixth time, the session agent is marked as having exceeded its constraints and the calls will not be routed to it until the time-to-resume has elapsed.

- **minimum ASR:** considered when make routing decisions. If some or all of the calls to the session agent have been answered, the minimum ASR value is considered to make the routing decisions.

For example, if the you set the minimum ASR at 50% and the session agent's ASR for the current window falls below 50%, the session agent is marked as having exceeded its constraints and calls will not be routed to it until the time-to-resume has elapsed.

ASR Constraints Configuration

To configure ASR constraints:

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 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. If configuring an existing session agent, enter the select command to select the session agent.
5. **min-seizures**—Enter the minimum number of seizures that when exceeded, cause the session agent to be marked as having exceeded its constraints. Calls will not be routed to the session agent until the time-to-resume has elapsed. The default value is 5. The valid range is:
 - Minimum—1
 - Maximum—999999999
6. **min-asr**—Enter the percentage you want as the minimum. If the session agent's ASR for the current window falls below this percentage, the session agent is marked as having exceeded its constraints and calls will not be routed to it until the time-to-resume has elapsed. The default value is 0. The valid range is:
 - Minimum—0
 - Maximum—100
7. Save and activate your configuration.

The following example shows a session agent configuration.

```
session-agent
  hostname                192.168.1.6
  ip-address
  port                    1720
  state                   enabled
  app-protocol            H323
  app-type                H323-GW
  transport-method
  realm-id                external
  description
  carriers
  allow-next-hop-lp      enabled
```

```

constraints                               enabled
max-sessions                             0
max-inbound-sessions                      4
max-outbound-sessions                    5
max-burst-rate                           0
max-inbound-burst-rate                   10
max-outbound-burst-rate                   1
max-sustain-rate                         0
max-inbound-sustain-rate                  0
max-outbound-sustain-rate                 0
min-seizures                             5
min-asr                                   50
time-to-resume                           30
ttr-no-response                          0
in-service-period                        0
burst-rate-window                        0
sustain-rate-window                      0
req-uri-carrier-mode                     None
proxy-mode
redirect-action
loose-routing                            enabled
send-media-session                       enabled
response-map
ping-method
ping-interval                            0
media-profiles
in-translationid
out-translationid
trust-me                                  disabled
request-uri-headers
stop-recurse
local-response-map
ping-to-user-part
ping-from-user-part
li-trust-me                              disabled
in-manipulationid
out-manipulationid
p-asserted-id
trunk-group
max-register-sustain-rate                0
early-media-allow
invalidate-registrations                  disabled
last-modified-date                       2006-05-12 19:48:06

```

ENUM Lookup

Telephone Number Mapping (ENUM from TELEphone NUmber Mapping) is a suite of protocols used to unify the telephone system with the Internet by using E.164 addresses with the Domain Name System (DNS). With ENUM, an E.164 number can be expressed as a Fully Qualified Domain Name (FQDN) in a specific Internet infrastructure domain defined for this purpose (e164.arpa). E.164 numbers are globally unique, language independent identifiers for resources on Public Switched Telecommunication Networks (PSTNs). ITU-T recommendation E.164 is the international public telecommunication telephony numbering plan.

How ENUM Works

ENUM uses DNS-based architecture and protocols for mapping a complete international telephone number (for example, +1 202 123 1234) to a series of Uniform Resource Identifiers (URIs).

The protocol itself is defined in the document E.164 number and DNS (RFC 3761) that provides facilities to resolve E.164 telephone numbers into other resources or services on the Internet. The syntax of Uniform Resource Identifiers

(URIs) is defined in RFC 2396. ENUM uses Naming Authority Pointers (NAPTR) records defined in RFC 2915 in order to identify available ways or services for contacting a specific node identified through the E.164 number.

Translating the Telephone Number

A telephone number is translated into an Internet address using the following steps:

1. The number is first stored in the following format, +1-202-555-1234. 1 is the country code for the United States, Canada, and the seventeen other countries that make up the North American Numbering Plan (NANP). The + indicates that the number is a complete, international E.164 telephone number.
2. All characters are removed except for the digits. For example, 12025551234.
3. The order of the digits is reversed. For example, 43215552021. The telephone number is reversed because DNS reads addresses from right to left, from the most significant to the least significant character. Dots are placed between each digit. Example: 4.3.2.1.5.5.5.2.0.2.1. In DNS terms, each digit becomes a zone. Authority can be delegated to any point within the number.
4. A domain (for example, e164.arpa) is appended to the end of the numbers in order to create a FQDN. For example, 4.3.2.1.5.5.5.2.0.2.1.e164.arpa.
5. The domain name is queried for the resource records that define URIs necessary to access SIP-based VoIP.

Once the authoritative name server for that domain name is found, ENUM retrieves relevant records and uses that data to complete the call or service. For example, the number 12025551234 returns sip:my.name@bigcompany.com.

About NAPTR Records

ENUM uses NAPTR records for URI resource records. NAPTR records are used to translate E.164 addresses to SIP addresses. An example of a NAPTR record is:

```
$ORIGIN 4.3.2.1.5.5.5.2.0.2.1.e164.arpa.  
IN NAPTR 100 10 "u" "sip+E2U" "!^.*$!sip:phoneme@example.net!"
```

This example specifies that if you want to use the "sip+E2U" service, you should use sip:phoneme@example.net as the address.

The regular expression can be used by a telephone company to easily assign addresses to all of its clients. For example, if your number is +15554242, your SIP address is sip:4242@555telco.example.net; if your number is +15551234, your SIP address is sip:1234@555telco.example.net.

About the Oracle CSM ENUM Functionality

The ENUM functionality lets the Oracle CSM make an ENUM query for a SIP request. The ENUM lookup capability lets the Oracle CSM transform E.164 numbers to URIs during the process of routing (or redirecting) a call. During the routing of a SIP call, the Oracle CSM uses a local policy attribute to determine if an ENUM query is required and if so which ENUM server(s) need to be queried. A successful ENUM query results in a URI that is used to continue routing or redirecting the call.

Configurable Lookup Length

You can configure a lookup length in the ENUM configuration that provides for more efficient caching of URI lookup results; in it, you can specify the length of the string for the DNS request starting from the most significant digit. This provides more flexibility for length matching, which is useful given the amount of wild card matching available in ENUM services. Specific ENUM groups might only be intended to provide NPANXX or wild card results.

UDP Datagram Support for DNS NAPTR Responses

The Oracle CSM's default behavior is to conform to the DNS standard defined in RFC 1035 Domain Names: Implementation and Specification, which sets a maximum size for UDP responses of 512 bytes. This limitation means that responses larger than 512 bytes are truncated (set with the TC, or truncation, bit). In addition, this limitation protects network and system resources because using TCP consumes an undesirable amount of both.

However, you can configure support ENUM queries that manage larger UDP DNS responses as set out in RFC 2671, Extension Mechanisms for DNS (EDNS0), enabling your Oracle CSM to manage responses beyond 512 bytes. According to RFC 2671, senders can advertise their capabilities using a new resource record (OPT pseudo-RR), which contains the UDP payload size the sender can receive. When you specify a maximum response size over 512 bytes, then the Oracle CSM add the OPT pseudo-RR to the ENUM query—without which the ENUM server will truncate the response.

Custom ENUM Service Type Support

You can configure the ENUM service type that you want to use for an ENUM group. The Oracle CSM has always supported E2U+sip and sip+E2U by default, and still does. With Release S-C6.1.0, however, you are also able to configure the service type to those supported in RFCs 2916 and 3721.

For example, you can now set the service type in the ENUM configuration to support E2U+sip and E2U+voicemsg:sip. When you configure customer ENUM service types on your system, however, you should note the following:

- New entries in the service-type parameter overwrite pre-existing values, including the default values.
- Because of the overwriting noted above, you must include the defaults (if you want them configured) when you are adding additional ENUM service type support. That is, you have to also type in E2U+sip and sip+E2U if you want them to be used in addition to the customized types you are setting.

ENUM Failover and Query Distribution

ENUM Query Distribution

The Oracle CSM can intelligently distribute ENUM queries among all configured ENUM servers. By setting the enum config's query method parameter to round robin, the Oracle CSM will cycle ENUM queries, sequentially, among all configured ENUM servers. For example, query 1 will be directed to server 1, query 2 will be directed to server 2, query 3 will be directed to server 3, and so on.

The default query method, hunt, directs all ENUM queries toward the first configured ENUM server. If the first server is unreachable, the Oracle CSM directs all ENUM queries toward the next configured ENUM server, and so on.

Failover to New enum-config

When an enum-config's configured servers are unreachable via the network, i.e., no response is received on a query, the Oracle CSM can failover to a defined ENUM config that contains different enum servers to query. This failover behavior works when all servers in an enum config are unreachable, rather than when the Oracle CSM receives not-found type responses.

The Oracle CSM queries each ENUM server once before trying the next configured server, and then ultimately trying the servers listed in the failover-to enum config. If the failover-to servers also are unreachable, the Oracle CSM fails the call; the failover-to behavior does not recurse among enum-configs, it only checks the first, linked enum-config.

ENUM Server Operation States


After 5 consecutive failed attempts, an ENUM server is considered Out of Service (OOS). All subsequent queries which would be directed to the OOS servers are immediately directed to the first non-OOS server. ENUM servers return to in-service after 600 seconds. If all configured ENUM servers are OOS, the Oracle CSM fails the call.

After the first failed attempt to reach an ENUM server, it is placed in a Time Out state, which it stays in for 30 seconds. Within this 30 seconds it will not be contacted when an ENUM query is made. After the 30 seconds pass, the ENUM server goes back to an in-service state.

Server Availability Monitoring

The Oracle CSM can probe an ENUM server's health by sending it a standard ENUM NAPTR query and receiving a valid answer. The query is for the phone number defined in the health query number parameter, which should be one that the ENUM servers can positively resolve. As long as the query succeeds, that ENUM server maintains its in-service state and is available for ENUM queries. Any lack of response, whether network based (time-outs), or application based (DNS error or not found response) is considered a query failure and the server is set to OOS and unavailable for ENUM queries.

The Oracle CSM continuously checks the health of all configured ENUM servers to determine their current state and monitor for failed servers' return to service. All servers are checked for availability at the health query interval parameter, as defined in seconds.

 **Note:** When ENUM server availability monitoring is enabled, ENUM servers can only exist in an in-service or out-of-service states; Without the health query interval defined, server availability monitoring is disabled, and ENUM servers exist in three service states.

ENUM Server IP Address and Port

You can configure an IP address and port for each enum server listed in the enum-servers parameter. IP address and port are specified in XXX.XXX.XXX.XXX:YYYY format with a port value range of 1024-65535. If the port number is not specified, 53 is assumed.

The Oracle CSM supports IPv6 ENUM configurations in IPv6 realms. The enumservers parameter in the enum-config configuration parameter can be configured IPv6 addresses in addition to IPv4 addresses. When IPv6 Addresses are used, the realm configured in the realm-id parameter must be an IPv6 realm.

Unapplicable SNMP Traps and Objects

When only IPv4 ENUM servers are configured, all legacy SNMP object and trap functionality remains the same. When IPv6 addressing is used for ENUM servers, these existing SNMP objects are obsoleted.

```
apSysMgmtENUMStatusChangeTrap      NOTIFICATION-TYPE
apENUMServerStatusTable             OBJECT-TYPE
```

IPv6 ENUM SNMP Traps and Objects

New SNMP trap notifies operators of ENUM Server Status change.

```
apAppsENUMServerStatusChangeTrap    NOTIFICATION-TYPE
OBJECTS                              { apAppsENUMConfigName,
                                   apAppsENUMServerInetAddressType,
                                   apAppsENUMServerInetAddress,
                                   apAppsENUMServerStatus }
STATUS                                current
DESCRIPTION
" The trap will be generated if the reachability status of an ENUM
server changes."
::= { apAppsNotifications 1 }
```

The following objects support this trap.

```
apAppsENUMServerStatusTable         OBJECT-TYPE
SYNTAX                              SEQUENCE OF ApAppsENUMServerStatusEntry
MAX-ACCESS                            not-accessible
STATUS                                current
DESCRIPTION
" A read-only table to hold the status of configured ENUM servers,
indexed by the name of the enum server, server address type and server IP.
Please note this table is the replacement of
apENUMServerStatusTable defined in ap-smgmt.mib, where the table was
obsoleted."
::= { apAppsMIBTabularObjects 1 }
apAppsENUMServerStatusEntry         OBJECT-TYPE
```



```

SYNTAX          ApAppsENUMServerStatusEntry
MAX-ACCESS     not-accessible
STATUS         current
DESCRIPTION
    "An entry designed to hold the status of a single ENUM server"
INDEX { apAppsENUMConfigName,
        apAppsENUMServerInetAddressType,
        apAppsENUMServerInetAddress }
 ::= { apAppsENUMServerStatusTable 1 }
ApAppsENUMServerStatusEntry ::= SEQUENCE {
    apAppsENUMConfigName          DisplayString,
    apAppsENUMServerInetAddressType  InetAddressType,
    apAppsENUMServerInetAddress    InetAddress,
    apAppsENUMServerStatus         INTEGER
}
apAppsENUMConfigName      OBJECT-TYPE
SYNTAX                    DisplayString
MAX-ACCESS                read-only
STATUS                    current
DESCRIPTION
    "The name of the enum-config element that contains this
    ENUM server."
 ::= { apAppsENUMServerStatusEntry 1 }
apAppsENUMServerInetAddressType  OBJECT-TYPE
SYNTAX                    InetAddressType
MAX-ACCESS                read-only
STATUS                    current
DESCRIPTION
    "The IP address of this ENUM server."
 ::= { apAppsENUMServerStatusEntry 2 }
apAppsENUMServerInetAddress      OBJECT-TYPE
SYNTAX                    InetAddress
MAX-ACCESS                read-only
STATUS                    current
DESCRIPTION
    "The IP address of this ENUM server."
 ::= { apAppsENUMServerStatusEntry 3 }
apAppsENUMServerStatus          OBJECT-TYPE
SYNTAX                    INTEGER {
                                inservice(0),
                                lowerpriority(1),
                                oosunreachable(2)
                                }
MAX-ACCESS                read-only
STATUS                    current
DESCRIPTION
    "The status of this ENUM server."
 ::= { apAppsENUMServerStatusEntry 4 }
apAppsENUMServerStatusGroup      OBJECT-GROUP
OBJECTS {
    apAppsENUMConfigName,
    apAppsENUMServerInetAddressType,
    apAppsENUMServerInetAddress,
    apAppsENUMServerStatus
}
STATUS                    current
DESCRIPTION
    "Report the status of configured ENUM servers."
 ::= { apAppsObjectGroups 1 }
apAppsEnumServerNotificationsGroup  NOTIFICATION-GROUP
NOTIFICATIONS {
    apAppsENUMServerStatusChangeTrap
}
STATUS                    current
    
```

DESCRIPTION
"A collection of traps to extend reporting capabilities." ::= { apAppsNotificationGroups 1 }

Caching ENUM Responses

As DNS responses often lead to further DNS queries, a DNS server can send additional multiple records in a response to attempt to anticipate the need for additional queries. The Oracle CSM can locally cache additional NAPTR, SRV, and A records returned from an ENUM query to eliminate the need for unnecessary external DNS requests by enabling the cache addl records parameter. These cached records can then be accessed by internal ENUM and DNS agents.

The unprompted NAPTR, SRV, or A record returned to the Oracle CSM must include complete information to resolve a call to be added to the local DNS/ENUM cache, otherwise the Oracle CSM will preform an external query to find the address it is looking to resolve.

Cached entries are per ENUM config. That means if one ENUM config has a number of cached entries, and an ENUM request is directed through a different ENUM config, the second configuration is not privy to what the first configuration has cached.

The Oracle CSM uses the shorter lifetime of the DNS response's TTL or the server dns attribute's transaction-timeout to determine when to purge a DNS record from the local cache.

Source URI Information in ENUM Requests

ENUM queries can be configured to include the source URI which caused the ENUM request by enabling the include source info parameter. The Oracle CSM can add the P-Asserted-ID URI (only if not in an INVITE) or the From URI into an OPT-RR Additional Record to be sent to the ENUM server. It can be useful to specify the originating SIP or TEL URI from a SIP request which triggered the ENUM query, so the ENUM server can provide a customized response based on the caller.

This feature implements the functionality described in the Internet Draft, DNS Extension for ENUM Source-URI, draft-kaplan-enum-source-uri-00.

When a P-Asserted-ID is blocked or removed before the ENUM query is made, the Oracle CSM only sends the URI in the From header.

Note that to support this feature, according to the Internet draft, ENUM clients must support 1220 bytes in UDP responses. Therefore, if this feature is enabled, and the max response size parameter is not set i.e., with a 512 byte default, the Oracle CSM will set the size to 1200 on the OPT-RR records sent.

Operation Modes

There are four modes of ENUM operation that are selected on a global basis:

- stateless proxy
- transaction stateful proxy
- session stateful proxy
- B2BUA with or without media

Stateless Proxy Mode

The stateless proxy mode is the most basic form of SIP operation. The stateless proxy mode:

- Has the least number of messages per call. No record route header is added and there are no 100 Trying or BYEs.
- Does not keep transaction state (timers and retransmission). There are no session counters and no session stop time. No session stop time means no RADIUS STOP records.
- Has no limits on session state.
- Can restrict functionality by specification. This can mean no media management, limited potential for RADIUS accounting, and no CALEA (no Release/BYE messages for CDC).
- Acts primarily as a routing device, with local policy routing and ENUM routing.

Transaction Stateful Proxy

In the transaction stateful proxy mode:

- Adds state to the proxy (not dialogs).
- Has lower number of messages per call. No Record Route header added and no BYES.
- Keeps transaction state (timers and retransmissions).
- Enforces session restrictions (32k) because of state management. These restrictions can be increased.
- Can restrict functionality by specification. This can mean no media management, limited potential for RADIUS accounting, and no CALEA (no Release/BYE message for CDC).
- Acts as routing device with transaction timers, with local policy routing and ENUM routing.
- Can off-load some transactions across unreliable links.

Session Stateful Proxy

The session stateful proxy mode:

- Maintains dialog state as a proxy.
- Includes BYES (though cannot be inserted)
- Keeps transaction state (timers and retransmission)
- Provides per-session information such as session counters per session agent, RADIUS STOP accounting record generation, CALEA CDC generation.
- Enforces session restrictions (32k) because of state management.
- Does not provide media management. There is no CALEA CCC.
- Routes full sessions with transaction timers with local policy routing and ENUM routing.

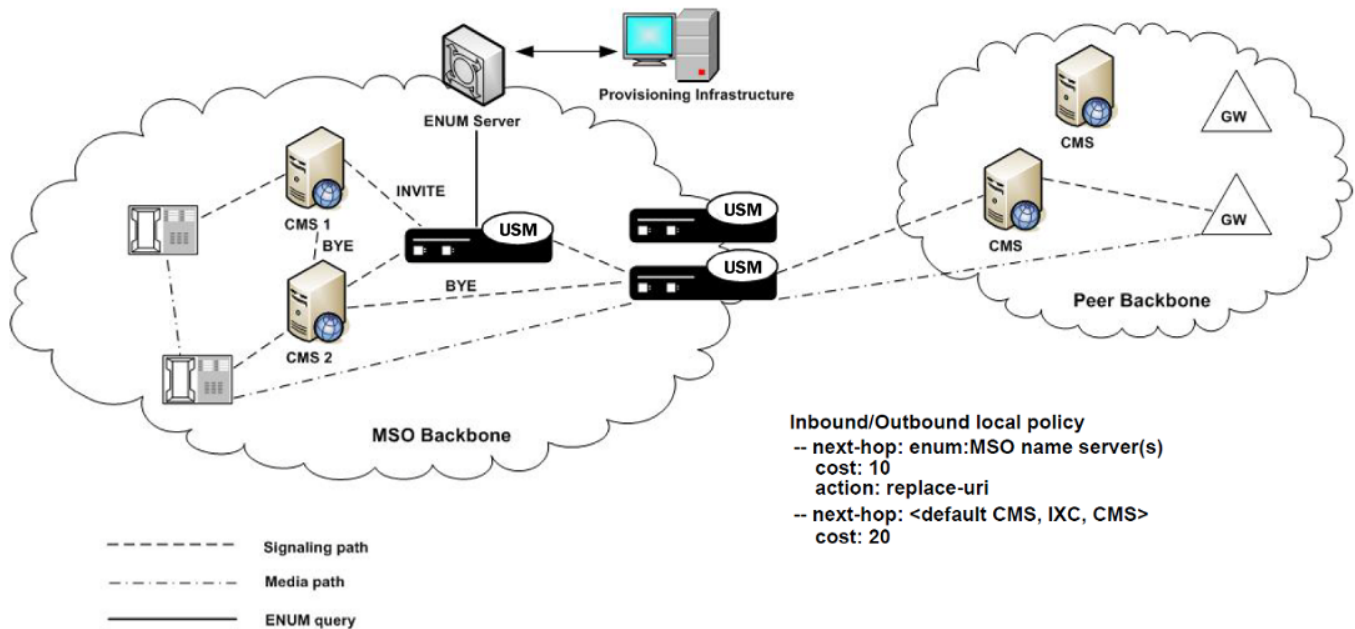
B2BUA

The B2BUA mode:

- Acts as UAS and UAC within call flow.
- Includes BYES (can be inserted).
- Keeps transaction state (timers and retransmissions)
- Provides per-session information such as session counters per session agent, RADIUS STOP accounting record generation, CALEA CDC generation.
- Enforces session restrictions (32k) because of state management.
- Can provide media management, including media routing through a single IP address with topology masking, CALEA CCC, media watchdogs for state management.
- Routes full sessions with topology masking. Includes rewriting Via, Route, Contact headers, full NATing with SIP NAT or header manipulation, direct bridging, local policy routing, and ENUM routing.

Example ENUM Stateless Proxy

The following diagram shows the Oracle CSM using ENUM to query a local subscriber database. The Oracle CSM serves as the inbound and outbound routing hub and performs media management. Calls are routed throughout the MSO network using ENUM lookup results.



ENUM Configuration

This section shows you how to configure ENUM on your Oracle CSM.

To configure ENUM:

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

```
ORACLE# configure terminal
```

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

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

4. name—Enter a string that uniquely identifies this ENUM configuration. You use this name in other areas of the Oracle CSM configuration to refer to this ENUM configuration. For example, in the local policy attributes.
5. top-level-domain—Enter the domain extension to be used when querying the ENUM servers for this configuration. For example, e164.arpa. The query name is a concatenation of the number and the domain.

For example the number is +17813334444 and the domain is e164.arpa, the query name would be 4.4.4.4.3.3.3.1.8.7.1.e164.arpa.com.

6. realm-id—Enter the realm where the ENUM servers can be reached. The realm ID is used to determine on which network interface to issue the ENUM query.
7. enum-servers—Enter the list of ENUM servers (an ENUM server and corresponding redundant servers) to be queried. Separate each server address with a space and enclose list within parentheses.

The first server on this list is the first one to be queried. If the query times out (including retransmissions) without getting a response, the next server on the list is queried and so on.

8. service-type—Enter 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.

This parameter defaults to the following service types: E2U+sip and sip+E2U.

You can enter multiple services types in the same entry, as in this example:

```
ORACLE(enum-config)# service-type E2U+sip,sip+E2U,E2U+voicemail
```

9. query-method—Set the strategy the Oracle CSM uses to contact ENUM servers. Valid values are:
 - hunt—Directs all ENUM queries toward the first configured ENUM server. If the first server is unreachable, the Oracle CSM directs all ENUM queries toward the next configured ENUM server, and so on.
 - round-robin—Cycles all ENUM queries, sequentially, among all configured in-service ENUM servers. Query 1 will be directed to server 1, query 2 will be directed to server 2, query 3 will be directed to server 3.
10. timeout—Enter the total time in seconds that should elapse before a query sent to a server (and its retransmissions) will timeout. If the first query times out, the next server is queried and the same timeout is applied. This process continues until all the servers in the list have timed out or until one of the servers responds.

The retransmission of ENUM queries is controlled by three timers. These timers are derived from this timeout value and from underlying logic that the minimum allowed retransmission interval should be 250 milliseconds; and that the Oracle CSM should retransmit 3 times before timing out to give the server a chance to respond. The valid values are:

- Init-timer—Is the initial retransmission interval. If a response to a query is not received within this interval, the query is retransmitted. To safeguard from performance degradation, the minimum value allowed for this timer is 250 milliseconds.
- Max-timer—Is the maximum retransmission interval. The interval is doubled after every retransmission. If the resulting retransmission interval is greater than the value of max-timer, it is set to the max-timer value.
- Expire-timer—Is the query expiration timer. If a response is not received for a query and its retransmissions within this interval, the server will be considered non-responsive and the next server in the list will be tried.

The following examples show different timeout values and the corresponding timers derived from them.

timeout >= 3 seconds

```
Init-timer = Timeout/11
Max-Timer = 4 * Init-timer
Expire-Timer = Timeout
```

timeout = 1 second

```
Init-Timer = 250 ms
Max-Timer = 250 ms
Expire-Timer = 1 sec
```

timeout = 2 seconds

```
Init-Timer = 250 ms
Max-Timer = 650 ms
Expire-Timer = 2sec
```

11. cache-inactivity-timer—Enter the time interval in seconds after which you want cache entries created by ENUM requests deleted, if inactive for this interval. If the cache entry gets a hit, the timer restarts and the algorithm is continued until the cache entry reaches its actual time to live.

Setting this value to zero disables caching. For optimal performance, set this to one hour. Rarely used cache entries are purged and frequently used entries are retained. The default value is 3600. The valid range is:

- Minimum—0
- Maximum—999999999

12. lookup-length—Specify the length of the ENUM query, starting from the most significant digit. The default is 0. The valid range is:

- Minimum—1
- Maximum—255

13. max-response-size—Enter the maximum size in bytes for UDP datagrams in DNS NAPTR responses. This parameter takes values from 512 (default) to 65535. Although the maximum value you can set is 65535, Oracle recommends configuring values that do not exceed 4096 bytes.

14. health-query-number—Set this parameter to a standard ENUM NAPTR query that will consistently return a positive response from the ENUM server.

Routing with Local Policy

15. `health-query-interval`—Set this parameter to the number of seconds to perpetually probe ENUM servers for health.
16. `failover-to`—Set this parameter to the name of another ENUM-config which to failover to under appropriate conditions.
17. `cache-addl-records`—Set this parameter to enabled for the Oracle CSM to add additional records received in an ENUM query to the local DNS cache.
18. `include-source-info`—Set this parameter to enabled for the Oracle CSM to send source URI information to the ENUM server with any ENUM queries.
19. Save your work.

Example


The following example shows an ENUM configuration called `enumconfig`.

```
enum-config
  name                               enumconfig
  top-level-domain
  realm-id                            public
  enum-servers                        10.10.10.10:3456
                                      10.10.10.11
  service-type                        E2U+sip,sip+E2U
  query-method                        hunt
  timeout                             11
  cacheInactivityTimer                3600
  max-response-size                   512
  health-query-number                 +17813245678
  health-query-interval                0
  failover-to                          enumconfig2
  cache-addl-records                  enabled
  include-source-info                  disabled
```

Configuring the Local Policy Attribute

You can specify that an ENUM query needs to be done for the routing of SIP calls. You do so by configuring the local policy's `next-hop` attribute with the name of a specific ENUM configuration, prefixed with the `enum:` tag. For example: `enum:test`

You can configure multiple next-hops with different ENUM servers or server groups (possibly with different top-level-domains). If the first ENUM server group you enter as the next hop is not available, one of the others can be used.

 **Note:** A new parameter called `action` has replaced the policy attribute's `replace-uri` parameter available prior to build 211p19.

To configure local policy:

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 `local-policy` and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(session-router)# local-policy
ORACLE(local-policy)#
```

4. `next-hop`—Enter the name of the ENUM configuration with the prefix `enum:`. For example, `enum:test`.
5. `action`—Set to redirect if you want to send a REDIRECT message back to the calling party with the information returned by ENUM in the Contact. The calling party then needs to send a REDIRECT using that information. The default value is `none`. Valid values are:

- none—No specific actions requested.
- replace-uri—To replace the next Request-URI with the next hop.
- redirect—To send a redirect response with this next hop as contact.

6. Save and activate your configuration.

Local Policy Example

The following example shows one local policy with the next-hop configured to use enum:test and a second with the next-hope configured to use enum:test_alterate.

```

local-policy
  from-address          *
  to-address            *
  source-realm          public
  activate-time         N/A
  deactivate-time       N/A
  state                 enabled
  last-modified-date    2006-03-09 09:18:43
  policy-attribute
    next-hop            enum:test
    realm                public
    action               none
    terminate-recursion disabled
    carrier
    start-time          0000
    end-time             2400
    days-of-week         U-S
    cost                 1
    app-protocol         SIP
    state                 enabled
  media-profiles
  policy-attribute
    next-hop            enum:test_alterate
    realm                public
    action               none
    terminate-recursion disabled
    carrier
    start-time          0000
    end-time             2400
    days-of-week         U-S
    cost                 2
    app-protocol         SIP
    state                 enabled

```

CNAM Subtype Support for ENUM Queries

CNAM, calling name, data is a string up to 15 ASCII characters of information associated with a specific calling party name. The *Internet-draft, draft-ietf-enum-cnam-08.txt*, registers the Enumservice 'pstndata' and subtype 'cnam' using the URI scheme 'pstndata:' to specify the return of CNAM data in ENUM responses. The Oracle CSM recognizes CNAM data returned via this mechanism. CNAM data is then inserted into the display name of the From: header in the original Request. If a P-Asserted-ID header is present in the original request, the CNAM data is inserted there as well.

CNAM data is identified by an ENUM response with service-type: E2U+pstndata:cnam

CNAM support is configured in the sip profile configuration element, which can then be applied to either a session agent, realm, or SIP interface.

The Oracle CSM can preform CNAM queries on the signaling message's ingress or egress from the system by setting the cnam lookup direction parameter to either ingress or egress. If the CNAM lookup direction parameters are

Routing with Local Policy

configured on both the ingress and egress sides of a call, the Oracle CSM will only preform the lookup on the ingress side of the call.

CNAM Unavailable Response

A CNAM response can include a Calling Name Privacy Indicator parameter ('unavailable=p') or Calling Name Status Indicator parameter ('unavailable=u') in responses. The Oracle CSM can insert a custom reason string into the SIP message's From and P-Asserted-ID header in the original requires.

Configuring the cnam unavailable ptype parameter inserts the specified text into the From and P-Asserted-ID headers when a CNAM response contains the unavailable=p parameter.

Configuring the cnam unavailable utype parameter inserts the specified text into the From and P-Asserted-ID headers when a CNAM response contains the unavailable=u parameter.

SIP Profile Inheritance

CNAM features, via the SIP Profile configuration element can be applied to session agents, realms, and SIP interfaces. The more generalized object inherits the more specific object's values. For example, if CNAM support via a SIP profile is configured on a session agent, the expected processing will override any SIP profile configuration on the downstream realm or SIP interface. Likewise, if CNAM support is unconfigured on the receiving session agent, but configured in the realm, CNAM configuration on the SIP interface will be ignored.

CNAM Subtype Support Configuration

To enable the Oracle CSM to preform CNAM subtype ENUM queries, you must configure a SIP profile with an enum-config object (that points to valid ENUM servers). The referenced enum-config configuration element lists the servers to contact for CNAM type queries (and other general ENUM server interaction parameters).

To configure CNAM subtype support:

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

```
ORACLE# configure terminal
```

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

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

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

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

4. name—Enter a string that uniquely identifies this SIP profile configuration. You use this name in other areas of the Oracle CSM configuration to refer to this SIP profile in session agents, realms, or SIP interfaces.
5. cnam-lookup-server—Set this parameter to the name of an ENUM-config to that will query ENUM servers for CNAM data.
6. cnam-lookup-dir—Set this parameter to ingress or egress to identify where the Oracle CSM performs a CNAM lookup with respect to where the call traverses the system. The default value is egress.
7. cnam-unavailable-ptype—Set this parameter to a string, no more than 15 characters, to indicate that the unavailable=p parameter was returned in a CNAM response.
8. cnam-unavailable-utype—Set this parameter to a string, no more than 15 characters, to indicate that the unavailable=u parameter was returned in a CNAM response.
9. Save your work.

Using the Local Route Table (LRT) for Routing


The LRT allows the Oracle CSM to determine next hops and map E.164 to SIP URIs locally for routing flexibility.

The LRT uses a local route cache that is populated by a local XML file on the Oracle CSM. Each local cache is populated from one defined XML file. For routing, the local route cache operates in a way similar to the ENUM model where a local policy next hop specifies the local route table that the Oracle CSM references. For example, you can configure one next hop to use one table, and another next hop to use a different table.

Similar to the ENUM model, the Oracle CSM typically performs a local route table lookup using the telephone number (TN) of the SIP Request-URI. This is the user portion of the URI, and the Oracle CSM ignores user parameters or non-digit characters. The local route table XML file defines the matching number and the resulting regular expression replacement value such as ENUM NAPTR entries do. The Oracle CSM uses the resulting regular expression to replace the Request-URI, and it uses the hostname or IP address portion to determine the next hop. If the hostname or IP address matches a configured session agent, the request is sent to that session agent. If the Oracle CSM does not find a matching session agent for the hostname/IP address, the Oracle CSM either performs a DNS query on the hostname to determine its IP address or sends the request directly to the IP address.

When the next hop is defined as a user-parameter lookup key, such as a routing number (RN) or carrier identification code (CIC), the defined key is used for the local route table lookup.

The Oracle CSM can attempt up to 10 next hops per LRT entry in the order in which they appear in the XML file. If the next hop is unsuccessful, the Oracle CSM tires the next hop on list. An unsuccessful hop may occur when an out-of-service session agent or the next hop responds with a failure response.

 **Note:** Entering XML comments on the same line as LRT XML data is not supported.

The Oracle CSM can perform local route table lookups for SIP requests and communicate the results to the SIP task. The new task processes the new local routing configuration objects.

When a SIP call is routed, the Oracle CSM uses local policy attributes to determine if a local route table lookup is required. If a lookup is needed, the Oracle CSM selects the local routing configuration to use. Successful local route table lookups result in URIs that can be used to continue routing and redirecting calls.

Local Route Table (LRT) Performance

Capabilities

- Loads approximately 500 LRT tables during boot time
- Loads 100,000 entries per LRT file
- Loads 2,000,000 LRT entries total per system

Constraints

- You cannot configure the Oracle CSM with 500 LRT files each with 100,000 entries.
- Actual performance that affects the interaction among the three performance attributes varies with system memory and configuration.

Local Routing Configuration

This section shows you how to:

- Set up local route configuration
- Specify that a set of local policy attributes needs to use local routing

Configure Local Routing

The local routing configuration is an element in the ACLI session-router path, where you configure a name for the local route table, the filename of the database corresponding to this table, and the prefix length (significant digits/bits) to be used for lookup.

To configure local routing:

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

```
ORACLE# configure terminal
```

Routing with Local Policy

2. Type session-router, and press Enter.

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

3. Type local-routing-config, and press Enter.

```
ORACLE (session-router) # local-routing-config
ORACLE (local-routing-config) #
```

4. name—Enter the name (a unique identifier) for the local route table; this name is used for reference in the local policy attributes when to specify that local routing should be used. There is no default for this parameter, and it is required.
5. file-name—Enter the name for the file from which the database corresponding to this local route table will be created. You should use the .gz format, and the file should be placed in the /code/lrt/ directory. There is no default for this parameter and it is required.
6. prefix-length—Enter the number of significant digits/bits to used for lookup and cache storage. The default value is 0. The valid range is:
 - Minimum—0
 - Maximum—999999999
7. Save and activate your configuration.

The following example displays a typical local routing configuration.

```
local-routing-config
  name                lookup
  file-name           abc.xml.gz
  prefix-length       3
```

Applying the Local Routing Configuration

Apply the local routing configuration by calling it to use in the local policy attributes. You do this by setting a flag in the next-hop parameter along with the name of the local routing configuration that you want to use.

To apply the local routing configuration:

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 local-policy, and press Enter.

```
ORACLE (session-router) # local-policy
ORACLE (local-policy) #
```

4. Type policy-attributes, and press Enter.

```
ORACLE (local-policy) # policy-attributes
ORACLE (local-policy-attributes) #
```

5. next-hop—In the next-hop parameter, type in lrt: followed directly by the name of the local routing configuration to be used. The lrt: tag tells the Oracle CSM that a local route table will be used.

```
ACMEPACKET (local-policy-attributes) # next-hop lrt:lookup
```

6. Save and activate the configuration.

LRT Entry Matching

When searching an LRT for a matching route, the Oracle CSM can be configured with one of three match modes with the match mode parameter in the local routing config. These modes are:

- exact—When searching the applicable LRT, the search and table keys must be an exact match.
- best—The longest matching table key in the LRT is the chosen match.

- all—The all mode makes partial matches where the table's key value is a prefix of the lookup key. For example, a lookup in the following table with a key of 123456 returns entries 1, 3, and 4. The 'all' mode incurs a performance penalty because it performs multiple searches of the table with continually shortened lookup keys to find all matching entries. This mode also returns any exact matches too.

Entry#	Key	Result
1	1	<sip:\0@host1.example.com>
2	122	<sip:\0@host22.example.com>
3	123	<sip:\0@host3.example.com>
4	1234	<sip:\0@host4.example.com>
5	1234567	<sip:\0@host7.example.com>
6	1235	<sip:\0@host5.example.com>

LRT Entry Matching Configuration

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

```
ORACLE# configure terminal
```

- Type session-router and press Enter.

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

- Type local-routing-config and press Enter.

```
ORACLE(session-router)# local-routing-config
ORACLE(local-routing-config)#
```

- match-mode—Set this parameter to either best, all, or leave it as exact which is the default. This indicates to the Oracle CSM how to determine LRT lookup matches.
- Save your work using the done command.

LRT String Lookup

The Oracle CSM can search an LRT for either E.164 or string table keys. This selection is on a global basis. When the string-lookup parameter is disabled (default) in the local routing configuration, all lookups will be E.164 type, except when:

- If eloc-str-lookup is enabled in a matching local policy's policy-attribute, E-CSCF procedures are applied and the resulting lookup type is 'string'.
- The Oracle CSM also performs string lookups exclusively when a compound lookup key is specified.

When the lookup type is 'E.164', the lookup is skipped if the lookup key is not a valid telephone number (i.e. it must contain only digits).

LRT String Lookup Configuration

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

```
ORACLE# configure terminal
```

- Type session-router and press Enter.

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

- Type local-routing-config and press Enter.

```
ORACLE(session-router)# local-routing-config
ORACLE(local-routing-config)#
```

- string-lookup—Set this parameter to enabled for the Oracle CSM to perform LRT lookups on table keys of a string data type. Leave this parameter to its default as disabled to continue using E.164 type lookups.

5. Save your work using the done command.

Directed Egress Realm from LRT ENUM

A message can be sent into a specific egress realm identified in an ENUM query or LRT lookup. The egress realm is noted by a configurable parameter in the result URI. The Oracle CSM is configured with the name of this parameter, that indicates an egress realm name, and looks for it in the returned URI.

To configure the parameter name, the egress-realm-param option is added to the sip config using the following format:

```
egress-realm-param=<name>
```

Where <name> is the parameter name to extract the egress realm name from.

When the egress realm param is defined, the ENUM and LRT results will always be checked for the presence of the URI parameter. The sip config options will apply for received SIP requests.

For example, if egress-realm-param=egress is added to the sip config, a matching entry in the LRT that specifies the egress realm core will look like this:

```
<route>
<user type="E164">+17815551212</user>
<next type="regex">!^.*$!sip:\0@core.example.com;egress=core!</next>
</route>
```

If the URI does not contain the parameter or the parameter identifies a realm that is not configured on the system, the egress realm that is normally applicable (from local policy, SIP-NAT, or session-agent data) will be used.

Directed Egress Realm Configuration

To add an egress parameter to look for in a sip-config:

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

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

2. Type session-router and press Enter.

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

3. Type sip-config and press Enter. If you are adding this feature to a pre-existing SIP configuration, you will need to select and edit it.

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

4. egress-realm-param—Configure this option with the parameter to parse for in a returned ENUM or LRT result:
For example

- options egress-realm-param=egress

In order to append the new option to the sip-config's options list, you must prepend the new option with a plus sign. For example:

```
ORACLE(sip-config)# options +egress-realm-param=egress
```

5. Save your work using the ACLI done command.



Note: The egress-realm-param option can be configured similarly in the h323-config.

SIP Embedded Route Header

The Oracle CSM examines the ENUM and LRT lookup result for embedded Route headers. In the LRT or as returned in an ENUM query a URI including an embedded route header would look like:

```
<sip:user@example.com?Route=%3Csip:host.example.com;lr%3E>
```

Using embedded Route headers is the Oracle CSM's default behavior. This can be overridden by adding the sip-config option use-embedded-route.

When the ENUM or LRT result becomes the top Route header, any embedded Route headers extracted are inserted just after that top Route (which will always be a loose route and include the "lr" URI parameter). In this case, the request will be sent to the top Route.

When the ENUM or LRT results become the Request-URI, any embedded Route headers extracted from the result are inserted before any Route headers already in the outgoing request. After that, if there are any Route headers in the outgoing request and the top Route header has an "lr" URI parameter, the request is sent to the top Route header. Otherwise, the request is sent to the Request URI.

SIP Embedded Route Header Configuration

To set the Oracle CSM's default behavior of using embedded route headers from ENUM queries or LRT lookups:

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

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

2. Type session-router and press Enter.

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

3. Type sip-config and press Enter. If you are adding this feature to a pre-existing SIP configuration, you will need to select and edit it.

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

4. use-embedded-route—Configure this as an option with one of the following arguments:

- all = use embedded routes from both ENUM and LRT results (default)
- none = do not use embedded routes
- enum = use embedded routes from ENUM results only
- lrt = use embedded routes from LRT results only

In order to append the new option to the sip-config's options list, you must prepend the new option with a plus sign. For example:

Set the options parameter by typing options, a Space, the option name use-embedded-route, and then press Enter.

```
ORACLE(sip-config)# options +use-embedded-route=none
```

5. Save your work using the ACLI done command.

LRT Lookup Key Creation

This section describes the Oracle CSM's LRT lookup key creation capability.

Arbitrary LRT Lookup Key

In addition to the standard From, To, and P-Asserted-Identity header fields the Oracle CSM can now use the values from any arbitrary SIP header as an LRT or ENUM lookup key. This is preformed by prepending a dollar sign

Routing with Local Policy

\$ by the header name whose value's userinfo portion of the URI will be used as the lookup key. For example, key=\$Refer-To would use the userinfo portion of the URI in the Refer-To header of the request as the lookup key.

An ampersand & followed by a header name will use the whole value of the header as the lookup key. For example, key=&X-Route-Key would use the whole value of the X-Route-Key as the lookup key. As a shortcut, an ampersand is not required for a "hidden" header. For example, "key=@LRT-Key" would use the value of the @LRT-Key header as the lookup key.

Hidden Headers for HMR and LRT lookup

When an LRT lookup key is more complex than just the URI's userinfo or a Tel-URI, HMR can be used to extract the data and build a special header.

By using a header name that begins with the at-sign "@" (e.g. @lrt-key), the header can be hidden and not included in outgoing SIP message, thus eliminating the need for an extra HMR rule to remove it.

Since '@' is not a valid character in a header name as defined by RFC 3261, there is no possibility of a collision between a header name defined in the future and a hidden header name beginning with @.

Compound Key LRT Lookup

LRT lookup keys can be combinations of more than one key value. For example, "key=\$FROM,\$TO" would construct a compound key with the userinfo of the From URI followed by a comma followed by the userinfo of the To URI.

If the request message contained:

```
From: <sip:1234@example.com>  
To: <sip:5678@example.com>
```

The compound key to match this From/To pair is "1234,5678".

In the table lookup, the compound key is a single key value and there is no special treatment of the comma in key matching. The comma is simply an ordinary additional character that is matched like any letter or digit (i.e. the comma must appear in the LRT entry's "type" element data). For example, if the table were configured for "best" match-mode, the lookup key "1234,5678" would match a table entry of "1234,567", but it would not match a table entry of "123,5678".

Retargeting LRT ENUM-based Requests

Request re-targeting is when a target or a request as indicated in the Request-URI, is replaced with a new URI.

This happens most commonly when the "home" proxy of the target user replaces the Request-URI with the registered contact of that user. For example, the original request is targeted at the Address-of-Record of bob (e.g. sip:bob@example.net). The "home" proxy for the domain of the original target, example.net, accesses the location service/registration database to determine the registered contact(s) for the user (e.g. sip:bob@192.168.0.10). This contact was retrieved in a REGISTER request from the user's UA. The incoming request is then re-targeted to the registered contact. When re-target-requests is enabled, or the original Request-URI is the Oracle CSM itself, the URI from the LRT lookup is used as the new Request-URI for the outgoing request.

When a request is routed rather than re-targeted, the Request-URI is not changed, but one or more Route headers may be inserted into the outgoing request. Sometimes a request which already contains Route headers will be routed without adding additional Route headers.

When the Oracle CSM routes requests and the original Request-URI was not the Oracle CSM itself, the URI from the LRT /ENUM lookup is added as the top Route: header including the "lr" parameter. The Request-URI then remains unchanged.

Whether the Oracle CSM re-targets or routes a request depends on the following:

- The target (Request-URI) of the received request
- The presence of Route headers

- Local Policy Attributes,
- Registration Cache matching.

If the original target is the Oracle CSM itself (i.e. the Request-URI contains the IP Address in the SIP interface the request was received on), the request is always re-targeted. When the original target is not the Oracle CSM and Local Policy is applied, the request will be re-targeted when the policy attribute action parameter is `replace-uri`. The request will also be re-targeted when the policy attribute specifies an ENUM or LRT lookup.

Retargetting requests can be configured in either the ENUM or LRT config depending on the request URI retrieval method chosen.

Re-targeting LRT ENUM-based Requests Configuration

This section shows you how to configure the Oracle CSM to re-target/re-route request message when performing an LRT or an ENUM lookup.

To configure the Oracle CSM to re-target or route request messages when performing an LRT lookup:

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 `local-routing-config` and press Enter.

```
ORACLE(session-router)# local-routing-config
ORACLE(local-routing-config)#
```

4. `retarget-requests`—Leave this parameter set to enabled for the Oracle CSM to replace the Request-URI in the outgoing request. Change this parameter to disabled for the Oracle CSM to route the request by looking to the Route header to determine where to send the message.
5. Save your work using the `done` command.

To configure the Oracle CSM to re-target or route request messages when performing an ENUM lookup:

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

```
ORACLE# configure terminal
```

7. Type `session-router` and press Enter.

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

8. Type `local-routing-config` and press Enter.

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

9. `retarget-requests`—Leave this parameter set to enabled for the Oracle CSM to replace the Request-URI in the outgoing request. Change this parameter to disabled for the Oracle CSM to route the request by looking to the Route header to determine where to send the message.
10. Save your work using the `done` command.

Recursive ENUM Queries

If the Oracle CSM receives an A-record in response to an ENUM query, it will reperform that ENUM query to the server received in the A-record.

If the Oracle CSM receives an NS record in response to an ENUM query, it will resend the original ENUM query to the DNS server defined in the realm of the FQDN in the NS record. It will use the response to perform a subsequent ENUM query.

This behavior is configured by setting the recursive query parameter in the enum config to enabled.

Recursive ENUM Queries Configuration

To configure the Oracle CSM to query a DNS server for a hostname returned in an ENUM lookup result:

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 local-routing-config and press Enter.

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

4. recursive-query—Set this parameter to enabled for the Oracle CSM to query a DNS server for a hostname returned in an ENUM result.
5. Save your work using the done command.


Multistage Local Policy Routing

Multistage local policy routing enables the Oracle CSM to perform multiple stages of route lookups where the result from one stage is used as the lookup key for the next routing stage.

Routing Stages

A routing stage signifies a re-evaluation of local policy based on the results of a local policy lookup. In the simplest, single stage case, the Oracle CSM performs a local policy lookup on a SIP message's Request URI. The result of that local policy lookup is a next hop FQDN, IP address, ENUM lookup, or LRT lookup; that result is where the Oracle CSM forwards the message. In the multistage routing model, that resultant next hop is used as the lookup key for a second local policy lookup.

The results of each stage do not erase the results of the previous stage. Thus, previous results are also possible routes to use for recursion, but the next stage results are tried first.

 **Note:** Setting a next hop to a SAG in a multistage scenario constitutes an error.

Multi-stage Routing Source Realm

By default, the Oracle CSM uses the realm within which a message was received as the source realm through all stages of a multistage local policy routing lookup. You can change this by setting the multi-stage-src-realm-override parameter in the session router config to enabled. Enabling this setting causes the Oracle CSM to use the next-hop realm from the current local policy stage as the source realm for the next stage of the lookup. This source realm selection process also repeats for each stage of a multistage routing scenario.

Network Applications

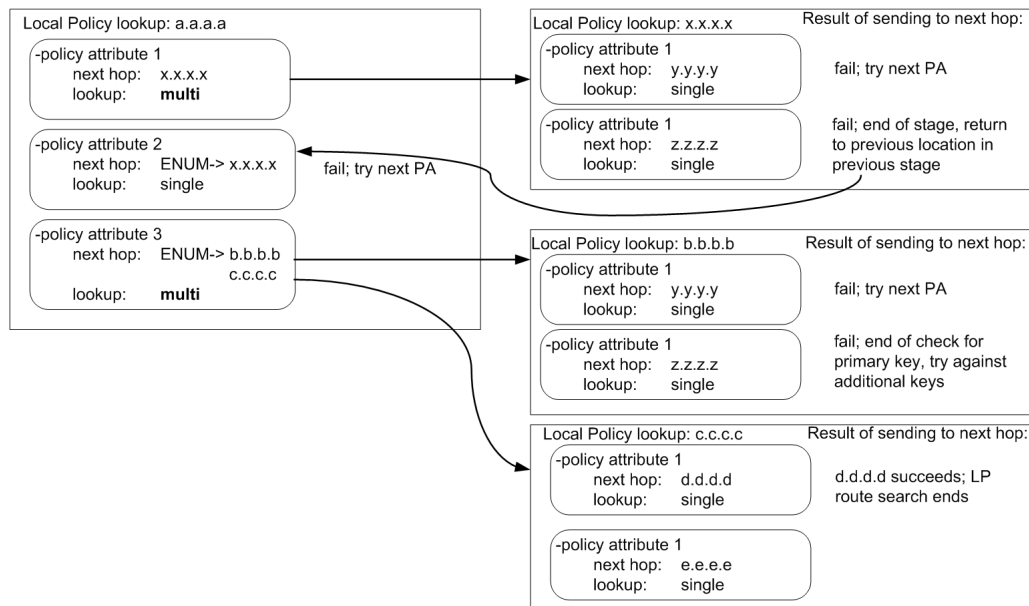
The following are typical applications of multistage routing:

- An operator might need to query an ENUM server for a destination number. Based on the NAPTR result of the ENUM query, the Oracle CSM performs a local policy lookup to decide how to route the request, perhaps based on a LRT table lookup.
- An operator might need to query one ENUM server for a number portability lookup, then based on the routing number perform a second ENUM query to a different server to learn which carrier to use for the routing number. Then, then based on the identified carrier perform a LRT lookup for what next-hop(s) to use for that carrier.
- An operator might query an LRT table to confirm the allowed source number. Then, based on the result, query an ENUM server for destination routing.

Multistage Routing Conceptual Example

Multistage routing is enabled by setting a policy attribute's lookup parameter to multi. Instead of replacing the SIP message's request URI with the policy attribute's next hop address or response from an ENUM or LRT lookup, the system uses that next hop or ENUM or LRT lookup response to reconstruct the SIP message. The reconstructed SIP message is fed again through all configured local policy configuration elements (and policy attribute sub elements). Each time the Oracle CSM re-evaluates a SIP message against local policies, it is considered an additional routing stage. When multiple records are returned from an ENUM or LRT lookup, the Oracle CSM evaluates the first response against all applicable local policies. If unsuccessful, the Oracle CSM evaluates all additional responses, in turn, against all applicable local policies.

For example:



Multistage Routing Example 2

The following three local policy configuration elements are configured in the Oracle CSM:

Routing with Local Policy

Local Policy 1		Local Policy 2		Local Policy 3	
from-address	*	from-address	*	from-address	*
to-address	159	to-address	192.168.1.49	to-address	215680000002
source-realm	private	source-realm	private	source-realm	private
policy-attribute		policy-attribute		policy-attribute	
next-hop	lrt:default-lrt	next-hop	lrt:carrier-lrt	next-hop	192.168.200.98
lookup	multi	lookup	multi	lookup	single
policy-attribute		policy-attribute		policy-attribute	
next-hop	192.168.200.50	next-hop	lrt:emergency	next-hop	192.168.200.97
lookup	single	lookup	single	lookup	single
				policy-attribute	
				next-hop	192.168.200.44
				lookup	multi

```
<route>
  <user type="E164">159</user>
  <next type="regex">!^.*$!sip:11568000000@192.168.200.47!</next>
  <next type="regex">!^.*$!sip:215680000002@192.168.200.99!</next>
  <next type="regex">!^.*$!sip:11578000000@192.168.200.44!</next>
</route>
```

```
INVITE sip:159@192.168.1.49:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.48:5060
From: sipp <sip:sipp@192.168.1.48:5060>;tag=1
To: sut <sip:159@192.168.1.49:5060>
Call-ID: 1-4576@192.168.1.48
CSeq: 1 INVITE
Contact: sip:sipp@192.168.1.48:5060
Max-Forwards: 70
Subject: Performance Test
Content-Type: application/sdp
Content-Length: 135
```

The local route table in default-lrt appears as follows:

```
<route>
  <user type="E164">159</user>
  <next type="regex">!^.*$!sip:11568000000@192.168.200.47!</next>
next>
  <next type="regex">!^.*$!sip:215680000002@192.168.200.99!</next>
  <next type="regex">!^.*$!sip:11578000000@192.168.200.44!</next>
</route>
```

1. The Oracle CSM receives an INVITE on realm, private (SDP is omitted below):

```
INVITE sip:159@192.168.1.49:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.48:5060
From: sipp <sip:sipp@192.168.1.48:5060>;tag=1
To: sut <sip:159@192.168.1.49:5060>
Call-ID: 1-4576@192.168.1.48
CSeq: 1 INVITE
Contact: sip:sipp@192.168.1.48:5060
Max-Forwards: 70
Subject: Performance Test
Content-Type: application/sdp
Content-Length: 135
```

2. The Oracle CSM performs a local policy search based on the following parameters:

```
from-address: sipp <sip:sipp@192.168.1.48:5060>;tag=1
to-address: sip:159@192.168.1.49:5060
Source Realm: private
```

3. The local policy search returns the four following routes to try:

```
lrt:default-lrt
192.168.200.50
lrt:emergency
lrt:carrier-lrt
```

The first next-hop route will be an LRT query. In addition, this policy attribute is configured with lookup=multi, meaning the results of the LRT query should be used for another local policy query, i.e., a second stage. More specifically, the request-uri that was received in response to the LRT query will be used as the to-uri in the next LP query.

The Oracle CSM performs the LRT lookup in the default-lrt configuration element and is returned the following:

```
sip:11568000000@192.168.200.47
sip:215680000002@192.168.200.99
sip:11578000000@192.168.200.44
```

The Oracle CSM attempts to use the results from the LRT query for the next stage Local Policy lookup(s). Beginning with the first route and continuing in sequential order, the Oracle CSM will try to route the outgoing INVITE message by performing additional Local Policy lookups on the remaining LRT query results, until the INVITE is successfully forwarded.

The Oracle CSM performs a local policy query on:

```
sip:11568000000@192.168.200.47
```

Which equates to a local policy lookup on:

```
from-URI=sipp <sip:sipp@192.168.1.48:5060>;
to-URI=sip:11568000000@192.168.200.47
Source Realm: private
```

The query fails because there is no Local Policy entry for 11568000000.

The Oracle CSM performs a second query on request-uri

```
sip:215680000002@192.168.200.99
```

Which equates to a local policy lookup on:

```
from-URI=sipp <sip:sipp@192.168.1.48:5060>;
to-URI=sip:215680000002@192.168.200.99
Source Realm: private
```

The LP query is successful and returns the following next- hops:

```
192.168.200.98
  192.168.200.99
192.168.200.44
```

The three routes shown above represent the next stage of the multistage routing for this INVITE. The policy attributes' lookup parameter is set to single for these next-hops. Therefore, the Oracle CSM will attempt to send the outgoing INVITE message to one or more of these next-hops; there are no more stages to check.

4. The Oracle CSM sends an INVITE to 192.168.200.98:

```
INVITE sip:215680000002@192.168.200.98;lr SIP/2.0
Via: SIP/2.0/UDP 192.168.200.49:5060
From: sipp <sip:sipp@192.168.1.48:5060>
To: sut <sip:159@192.168.1.49:5060>
Call-ID: SDnhae701-76e8c8b6e168958e385365657faab5cb-v3000i1
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.200.49:5060;transport=udp>
Max-Forwards: 69
Subject: Performance Test
Content-Type: application/sdp
Content-Length: 140
```

5. If the INVITE is sent to 192.168.200.98 successfully, the local policy routing will conclude and the call will continue processing. Otherwise the Oracle CSM will try the other next hops until a route succeeds or all next-hops have been exhausted

Customizing Lookup Keys

When the next hop parameter points to perform an ENUM or LRT lookup, it can be provisioned with a "key=" attribute in order to specify a parameter other than the username to perform the lookup on. The following table lists the header, key value, and corresponding syntax to configure the Oracle CSM with.

Routing with Local Policy

Username from Header:	Key Value	Example
To-URI	\$TO	key=\$TO
From-URI	\$FROM	key=\$FROM
P-Asserted-Identity	\$PAI	key=\$PAI

For a subsequent stage in multistage local policy routing, the lookup key to use for the next stage can be explicitly specified by configuring the next key parameter. By default, multistage lookups use the modified Request-URI returned from the ENUM/LRT response as the to-address key for the next local policy lookup. When the next key parameter is configured, its value will be used for the to-address key in the subsequent local policy lookup regardless if an ENUM or LRT lookup is configured for that policy attribute. The key syntax for this parameter is the same as with the Routing-based RN and CIC feature.

Multistage Routing Lookup Termination

It is important for the Oracle CSM to have a mechanism to stop performing additional stages of route lookups and limit the number of attempts and results to be tried. Routing termination can be performed at in the non-multistage way or at the global session router level.

Global Local Policy Termination

The Oracle CSM can be configured to limit local policy lookups based several aspects of the route lookup process:

- Limiting the number of stages per message lookup—The Oracle CSM can limit to the number of additional local policy lookup stages it will perform received message to a maximum of 5. This is configured with the additional lp lookups parameter. Leaving this parameter at its default value of 0 essentially disables multistaged local policy lookups.
- Limiting the number of routes per Local Policy lookup—The Oracle CSM can limit the number of route results to use as returned for each Local-Policy lookup. This is configured with the max lp lookups routes per lookup parameter. Leaving this parameter at its default value of 0 places no limit on the number of returned routes the Oracle CSM can try.
- Limiting the total number of routes for all local policy lookups per message request—The Oracle CSM can limit the number of route returned in total across all lookups for a given request, including additional stages. This is configured with the total lp routes parameter. Leaving this parameter at its default value of 0 places no limit on the number of returned routes the Oracle CSM can try. This parameter overrides any configured options.

Additionally, the Oracle CSM monitors for local policy lookup loops which could cause a significant deterioration in performance. If a loop is found, the Oracle CSM stops trying the looping route list and proceeds to try any remaining routes..

Multistage Local Policy Routing Configuration

To set up your local policy attributes for routing using the TO header:

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

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

2. Type session-router and press Enter.

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

3. Type local-policy and press Enter. If you are adding this feature to a pre-existing local policy configuration, you will need to select and edit a local policy.

```
ORACLE(session-router)# local-policy
ORACLE(local-policy)#
```

4. Type policy-attributes and press Enter. If you are adding this feature to a pre-existing local policy configuration, you will need to select and edit your local policy.

```
ORACLE(local-policy) # policy-attributes
ORACLE(local-policy-attributes) #
```

5. next-hop—This is the next signaling host and/or object to query. This parameter can be configured as an IP address, ENUM server, or LRT. You can also add a lookup key to an ENUM server or LRT lookup with the following syntax:

```
next-hop    enum:ENUM-object;key=$TO
```

6. terminate-recursion—Set this parameter to enabled to terminate local policy route recursion when the current stage completes.
7. lookup—Leave this parameter at the default single for single stage local policy routing or set it to multi to enable multistage local policy routing.
8. next-key—Set this parameter to \$TO, \$FROM, or \$PAI if you wish to override the recently-returned lookup key value for the next stage.
9. Save and activate your configuration.

Maintenance and Troubleshooting

The show sipd policy command includes four additional counters that refer to single and multistage local policy lookups. All counters are reported for the recent period, and lifetime total and lifetime period maximum. These counters are:

- Local Policy Inits—Number of times the Oracle CSM makes an initial local policy lookup.
- Local Policy Results Max—Number of times the Oracle CSM truncated the number of routes returned for a local policy lookup because the maximum number of routes per local policy lookup (max lp lookups routes per lookup) threshold was reached.
- Local Policy Exceeded—Number of times the Oracle CSM truncated the number of routes returned for a local policy lookup because the maximum number of routes per message request (total lp routes) threshold was reached.
- Local Policy Loops—Number of times the Oracle CSM detected a loop while performing a multistage local policy lookup.

Traps

An SNMP trap is generated to notify that the limit on the additional lp lookups threshold has been reached during the recent window period. This trap occurs a maximum of once during a window period.

```
apSysMgmtLPLookupExceededTrap NOTIFICATION-TYPE
    STATUS          current
    DESCRIPTION
        " The trap will be generated the first time the additional Local
        Policy Lookups limit is reached is in the recent window period. The trap will
        only occur once during a window period."
    ::= { apSystemManagementMonitors 65}
```

Routing Based on UA Capabilities

In compliance with RFC 3841, the Oracle CSM is able to make forwarding and forking decisions based on preferences indicated by the UA. To do this, the Oracle CSM 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 CSM made routing preference decisions solely via the q value present in the contact header. In cases where the preferences were equal, the Oracle CSM simply forwarded to those contacts simultaneously (parallel forking). In cases where the q value were not equal, the Oracle CSM forwarded in sequence (sequential forking), forwarding to the highest q value first.

The Oracle CSM 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 CSM 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 CSM

Endpoints register their capabilities by listing them in the Contact headers of the REGISTER request. The Oracle CSM stores these feature parameters in its registration cache along with the other contact information. In the case of ENUM databases, the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM 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 CSM fully supports. UAs include these parameters in the accept-contact: header to further clarify capabilities requirements for the session. The Oracle CSM 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 CSM only forwards to contacts that support the features and have registered that support.

If only the require parameter is present, the Oracle CSM 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 CSM 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 CSM 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 CSM 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 CSM uses the UE’s q value solely to determine parallel and sequential forking.

Routing-based RN and CIC

When the Oracle CSM performs local policy routing, it selects local policy entries based on from addresses, to addresses, and source realms. All three are configurable in the local policy configuration. The to addresses can either be the username in a Request-URI (if it is an E.164/phone number format), or the request-URI’s hostname or IP address. The Oracle CSM sorts matching local policies based on policy attribute entries. A policy attribute defines a next hop, which can be a session agent or a session agent group. Alternatively, the next hop might define an ENUM server group or local route table to use to find the next hop.

If the routing-based RN and CIC feature is not enabled, the Oracle CSM performs the subsequent ENUM query or local route table lookup using the Request-URI’s username, if it is a telephone number (TN). The TN is the normalized user part of the Request-URI, ignoring any user parameters or non-digit characters.

If the routing-based RN and CIC feature is enabled, the Oracle CSM instead performs the ENUM or local route table lookup based on a user parameter, which is useful for lookups based on routing number (RN) or carrier identification code (CIC):

Routing with Local Policy

- An RN is a number that identifies terminating switch nodes in Number Portability scenarios when the original TN has been moved to the switch defined by the RN.
- A CIC is the globally unique number of the terminating carrier to which a ported number has been moved.

In applications where the Oracle CSM is given the RN or the CIC in the Request-URI, this feature is useful because the Oracle CSM can perform an additional ENUM or local route table lookup to find the next hop to the RN or the CIC. Typically, ENUM servers have imported Number Portability data with which to respond to the Oracle CSM query, and (for example) the Oracle CSM can use local route tables for storing CIC values for direct carrier hand-off.

Even with this feature enabled, the Oracle CSM still performs local policy match selection based on the TN. This feature only uses the RN or CIC user-parameter for the ENUM or local route table lookup after the local policy and policy attributes have been selected.

Routing-based RN Configuration

This section shows you how to specify that a set of local policy attributes should use an RN for lookup. You can also set this value to CIC, or to any value you require.

You can set the lookup key to an RN in the local policy attributes' next-hop parameter.

To set the lookup key to RN:

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 local-policy and press Enter.

```
ORACLE(session-router)# local-policy
ORACLE(local-policy)#
```

4. Type policy-attributes and press Enter.

```
ORACLE(local-policy)# policy-attributes
ORACLE(local-policy-attributes)#
```

5. next-hop—In the next-hop parameter—after the kind of ENUM service used—type a colon (:). Then, without spaces, type in key=rn and press Enter.

```
ORACLE(local-policy-attributes)# next-hop lrt:lookup;key=rn
```

6. Save and activate your configuration.

The Session Load Balancer and Route Manager

Functional Overview

Subscriber-aware Load Balancing and Route Management (SLRM) is a proprietary mechanism within the Oracle CSM that presents a single target for devices sending SIP messages to your IMS core over the applicable interfaces. As such, SLRM provides load-balanced services connecting users to a group of Oracle CSMs as if they are a single node. Its load balancing functions are limited to operation with other Oracle CSMs as targets over Diameter. Oracle has developed and maintains a proprietary interface, the Sc interface, to manage load balancing operations with target Oracle CSMs. This interface is documented below.

The SLRM acts as an extension upon I-CSCF operation within the Oracle CSM. It dynamically discovers and evaluates resource utilization of Oracle CSMs deployed in the core. Having discovered and identified each Oracle CSM's status, the SLRM then distributes traffic between them. Applicable traffic includes:

- SIP REGISTERs;
- Out-of-the-blue SIP INVITEs from application servers; and
- SIP INVITEs from end-stations external to your network for which terminating services may apply.

The user must explicitly set their Oracle CSM to operate as an SLRM using the command **set-component-type**. The user can confirm this operational mode using the **show ims-core-product-type** or the **display-component-type** command.

Product Functional Matrix

The SLRM is a component of the Oracle CSM/USM product group, which Oracle develops using the same software, basing the discrete operational functionality on configuration and deployment within an IMS core. The Oracle USM and Oracle CSM are distributed as separate products. The Session Load Balancer and Route Manager (SLRM) is distributed as a special configuration of an Oracle CSM.

Refer to the table below to understand product nomenclature as specified by configuration and functionality. Minimum configuration excludes universally common box configurations, such as interfaces and realms.

Nomenclature	Minimum Configuration	Functionality
OC-CSM	registrar, home-subscriber-server, authentication-profile	IMS S-CSCF and I-CSCF
OC-USM	ims-access, registrar, home-subscriber-server, authentication-profile	IMS P-CSCF, I-CSCF and S-CSCF

The Session Load Balancer and Route Manager

Nomenclature	Minimum Configuration	Functionality
SLRM	set-component-type, lb-interface, lb-core-config	I-CSCF, Proprietary I-CSCF Load Balancing

References to these product names must be understood within the context of their nomenclature and configuration for the purposes of understanding which functions they perform and which functions they do not perform.

Physical Deployment

The SLRM is typically deployed in an High Availability (HA) configuration, which includes multiple Oracle CSMs operating redundantly as SLRMs. There are no limitations to the number of platforms deployed as SLRMs or the number of devices with which they interoperate.

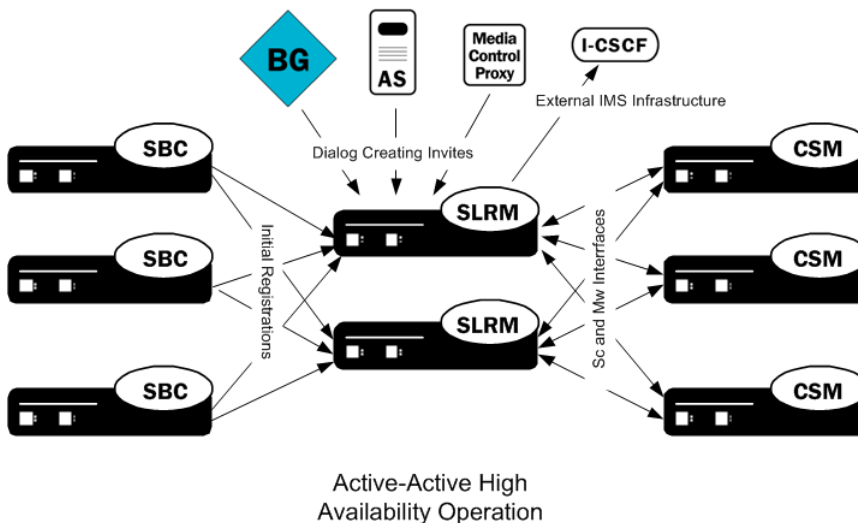
An SLRM typically resides between the network's P-CSCFs (usually Oracle SBCs) and S-CSCFs, load balancing initial registrations from the P-CSCF and INVITEs from a variety of sources. P-CSCFs send registrations. Devices from which the SLRM may receive these INVITEs include:

- AS
- BGCF
- MGCF

The SLRM may also receive traffic that it does not load balance. This includes traffic for which the target S-CSCF is already known. In these cases, the I-CSCF follows 3GPP standards for operational behavior.

Active-Active Redundancy

Multiple devices performing the SLRM function can, and should, reside in parallel to provide redundant SLRM operation. The SLRM function is not dialog stateful, which allows active-active redundancy. Typically, the SLRM configuration on each redundant device is exactly the same.



Configuring the devices running SLRM as Session Agents within Session Agent Groups on Oracle SBCs is one method of establishing redundant connectivity. A more generic means of establishing redundant connectivity could be to use DNS techniques, such as dynamic or round-robin DNS, as the means for the P-CSCFs to reach redundant SLRMs.

SLRM-Supported SIP Interfaces

Standard SIP interfaces that the SLRM may support between itself and external devices other than the Oracle CSM include:

- Mw—The SLRM load balances all initial registration traffic.
- ISC—The SLRM may be in the path for the initial dialog transaction, but is bypassed by the AS for subsequent dialog messages.

- Mi—The SLRM may be in the path for the initial dialog transaction, but is bypassed by the BGCF for subsequent dialog messages.
- Mr—The SLRM may be in the path for the initial dialog transaction, but is bypassed by the media control device for subsequent dialog messages.

Oracle CSM's Role as S-CSCF

The Oracle CSMs that participate as S-CSCFs in an SLRM load balanced deployment are responsible for performing these key functions:

- Sends information about itself to the SLRM, including:
 - Cores serviced—The user configures Oracle CSM registrars with core names. A core name abstracts a registrar, providing a means of correlating domains serviced by a core between the Oracle CSM and the SLRM.
 - Cluster membership—All Oracle CSMs reside within a default cluster (null). The user can configure specific cluster membership to establish geographic-based preferences with which the SLRM can restrict traffic unless and until outages require that the infrastructure route that traffic outside of the preferred geography.
 - Number of current endpoints—An Oracle CSM's known number of endpoints includes registered and unregistered users within the registration cache.
 - Maximum endpoint capacity—The Oracle CSM determines maximum endpoint capacity dynamically. It uses the current number of endpoints and the resources in use by those endpoints to determine maximum endpoint capacity. The SLRM uses this number as part of its criteria to establish load balance order.
 - Operational resources available—The Oracle CSM also tracks current CPU and memory utilization.
- Manages cluster membership via refresh timing.
- Manages SLRM core registration via refresh timing.
- Responds to SLRM-initiated rebalance processes.
- Supports manual rebalance processes from the Oracle CSM.
- Maintains connectivity with the SLRM function via watchdog messaging

The user specifies registrars for load balancing on an Oracle CSM using a registrar's (**ims-core**) parameter, which aligns with a core name configured on the SLRM. These configurations establish 'load-balance group' names between Oracle CSMs and SLRMs.

Having determined core membership, the SLRM determines a target Oracle CSM by evaluating the endpoint capacity information provided by the Oracle CSMs and identifying the best target for the traffic.

Logical Deployment

The key configurations used to establish load balancing operation, includes:

- Core—This required configuration provides a reference between Oracle CSM registrars and the load balancing configuration. After configuration, each Oracle CSM advertises its supported cores to the SLRM, which then creates a list of load-balance candidates for those cores.
- Cluster—This configuration refines the list of Oracle CSMs between which the SLRM balances traffic. The user can establish geographical preferences between Oracle SBCs and Oracle CSMs via cluster configuration on both devices. The default cluster ID, null, allows unconfigured Oracle CSMs and third party P-CSCFs to belong to clusters.

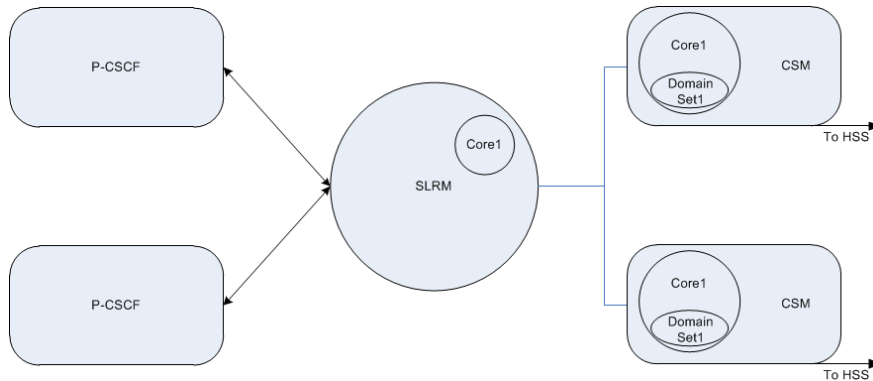
Explanations and configuration instructions for cores and clusters are presented below.

SLRM Core

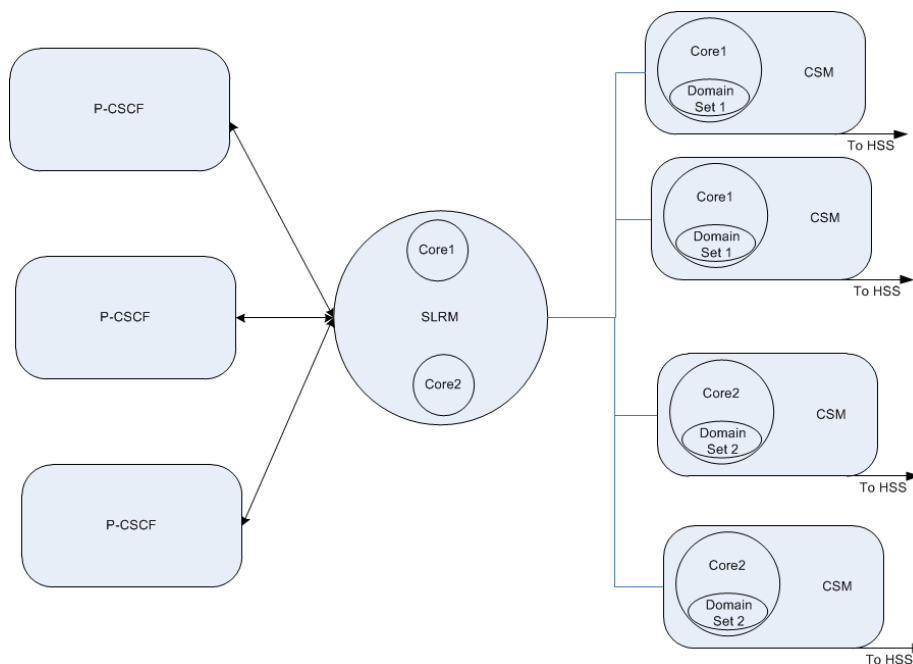
An SLRM core is a required configuration that establishes a group of Oracle CSMs to which an SLRM load balances registrations and applicable INVITEs. The user configures cores to equate to Oracle CSM SIP registrars, which service the associated set of domains at the HSS. An SLRM's core configuration includes a list of domains, that must match those of the target registrars. Although the original REGISTER or INVITE is sent by a device that is unaware of core configuration, the REGISTER or INVITE does include target domain. The SLRM recognizes the target domain and, based on the core configuration, associates the message with the applicable core.

The Session Load Balancer and Route Manager

The Oracle CSM includes a core configuration within each sip-registrar that it advertises to the SLRM. Core names must be the same on the SLRM and the Oracle CSMs. Based on this advertisement, the SLRM groups Oracle CSMs that service the same set of domains for load balancing.



The SLRM supports any number of cores. In the diagram below, the SLRM services both Core1 and Core2. There are 2 Oracle CSMs for each core. The SLRM load balances registrations from P-CSCFs for Core1 between the Oracle CSMs at the top of the diagram and those for Core2 between the bottom.



You create core configurations on both the SLRM and all applicable Oracle CSMs.

Cluster Configuration

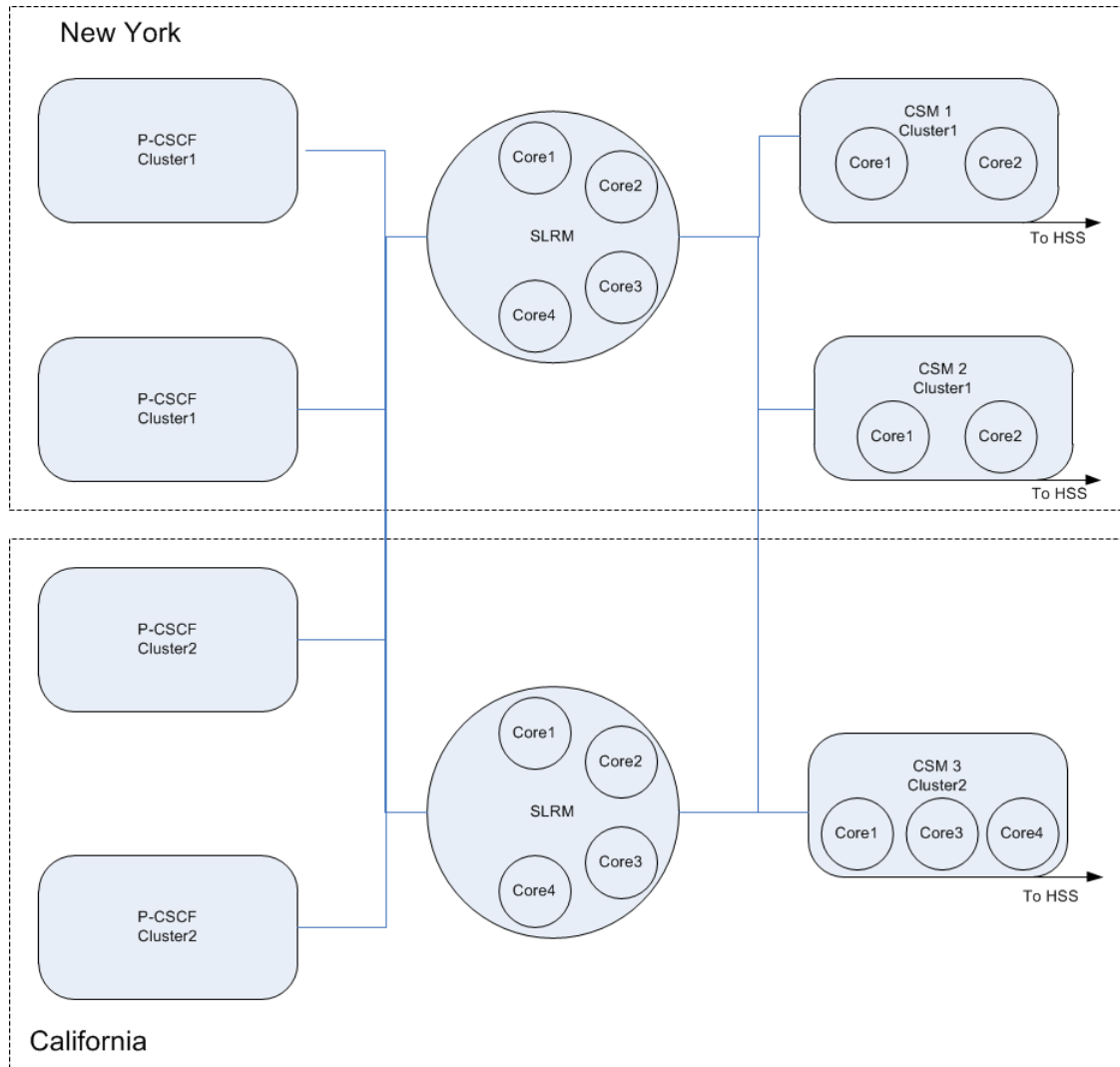
As stated, core configurations ensure that the SLRM does not send traffic to an Oracle CSM that does not service that core's domain(s). Cluster configuration can refine these constraints by establishing a group of Oracle CSMs for which core re-balancing is preferred. The SLRM attempts to load balance Oracle CSMs that belong to the same cluster first. If all Oracle CSMs in that cluster are unavailable, the SLRM can choose an Oracle CSM that services the correct core, but belongs to a different cluster.

Each participating Oracle CSM must belong to at least one cluster to participate in load balancing processes. To accommodate this requirement, all Oracle CSMs belong to the null cluster by default. (Cluster configuration is a string, which is empty by default.) In addition, the SLRM adds any P-CSCF without a cluster configuration to the null cluster.

As deployments grow, however, the operator may need to balance the requirements of maintaining connectivity during outages with the value of keeping core traffic regionally focused. The configuration of multiple clusters

logically separates groups of Oracle CSMs. The operator can configure these cluster to establish "affinity" between P-CSCFs and Oracle CSMs that reside, for example, in the same geographic region.

Noting the diagram below, a REGISTER coming from a P-CSCF in New York could be sent to a Oracle CSM that services Core1. All things equal, SLRM can choose CSM 1, CSM 2 or CSM 3. Cluster configuration, however, can defer the selection of CSM 3, thereby preventing the traffic from traversing the long span to California. Furthermore, if both CSM 1 and CSM 2 become unavailable, SLRM has the option of forwarding to CSM 3 because it supports Core1.



SLRM Operation

An Oracle CSM that is not configured to perform SLRM functions performs standard I-CSCF and S-CSCF functions. When configured for SLRM however, the S-CSCF functions are no longer available. Instead, the SLRM performs the following tasks as a 'front-end' to a pool of load balanced Oracle CSMs:

- Establishing the load balance pool
- Balancing traffic
- Re-balancing traffic

These operational functions are described in the following sections.

Establishing the Load Balance Pool

The SLRM creates pools of Oracle CSMs to load balance new registrations and applicable INVITEs. These pools include Oracle CSM that service the same cores. The SLRM ranks Oracle CSMs to create an ordered list from which it can choose registration targets.

Oracle's Diameter Sc interface includes messaging sequences and AVPs to support the interaction between the SLRM and Oracle CSMs. Key to this interaction is the Oracle CSM specifying cluster membership and registering to service cores at the SLRM. Load balanced pools for a given core include only the Oracle CSMs registered for that core.

Supporting information over the Sc interface provides the details of the Oracle CSM's registration. To this end, a client-server relationship exists, with the SLRM function acting roughly as server:

- Upon startup, each Oracle CSM advertises its cluster membership, and subsequently the IMS "cores" it services and its resource utilization. This allows the SLRM function to group Oracle CSMs for load balancing.
- At agreed upon intervals, the Oracle CSM resends advertisements to confirm or change SLRM-core registration and resource utilization.
- The Oracle CSM is also capable of initiating graceful shutdown procedures to remove itself from any load balance pool.

Sc interface registration information that aligns with the functions above include:

- New Registration — The SLRM includes this Oracle CSM in the "core" lists and begins to assign users to it.
- Re-Registration — The SLRM refreshes the list of cores within which this Oracle CSM participates.
- De-Registration — The SLRM removes this Oracle CSM from the core list from which it is de-registering.

After a Oracle CSM registers with the SLRM, the SLRM tracks its state. The SLRM only includes devices in the proper state when making load balancing calculations. Oracle CSM states include:


- In Service — The Oracle CSM has registered at the SLRM. The SLRM can include this device in its load balancing calculations and send it endpoint registrations.
- Out of Sync — Capacity information is unreliable. The Sc interface is down or the SLRM registration has timed out. This device would be selected last. The device goes back in-service if the Sc interface recovers or it re-registers with the SLRM. The system uses a back-off timing algorithm to determine when to send connectivity re-attempts, beginning with 70 seconds and proceeding by exponentially increasing the time between connectivity re-attempt until it reaches 1920 seconds (32 minutes).
- Out of Service — Not available for use by this core. The device is not responding to attempts at re-establishment. The SLRM brings the device back into service using a truncated exponential back-off method that is capped at 32 minutes.
- Destroyed — The SLRM has removed this device from this core's list because it has explicitly de-registered.

Balancing

Balancing is the act of the SLRM maintaining an ordered list of Oracle CSMs to which it sends traffic for a given core.

Having established each Oracle CSM state, the SLRM groups all Oracle CSMs that service a given core into clusters. The SLRM then establishes load balance lists labeled:

- Preferred — The administrator has configured both the target Oracle CSM and the source P-CSCF within the same cluster.
- Alternative — The target Oracle CSM and the source P-CSCF are not in the same cluster.

 **Note:** For single cluster deployments, all Oracle CSMs registered to the SLRM function belong to the default cluster. If all P-CSCFs are in the default cluster, then every Oracle CSM is "Preferred" for every registration.

Having categorized each Oracle CSM within their clusters, the SLRM then creates, and on an on-going, dynamic basis using KPIs from SLRM-registration updates, maintains the load balance order as follows:

- Preferred Oracle CSMs — Sorted by free endpoint capacity.
- Preferred, Out of Sync Oracle CSM — Add to the bottom of the preferred list, sorted by free endpoint capacity.

- Alternative Oracle CSMs — Sort by free endpoint capacity and add to list after all preferred Oracle CSMs.
- Alternative, Out of Sync Oracle CSM — Add to the bottom of the alternative list, sorted by free endpoint capacity.

There may be multiple Alternative Oracle CSM groups. Alternative groups are selected in round-robin fashion.

Free endpoint capacity is calculated as percent utilization based on supported capacity and current utilization. It is reported by the Oracle CSM to the SLRM via the Sc interface.

SLRM distributes each message individually based on the criteria above. If a message fails at a Oracle CSM in the list, SLRM proceeds by sending the message to the next Oracle CSM in the composite list.


Re-balancing

Re-balancing is the process of taking some number of registered users from a functioning Oracle CSMs and redistributing them between other Oracle CSMs. Re-balancing occurs when manually invoked by the user from the Oracle CSM using the **release-users** command.

The Oracle CSM initiates a Reg-Event process to de-register the users. This process includes the following steps:

1. The Oracle CSM waits for users to send registration refresh.
2. Upon receipt of the users first registration refresh, the Oracle CSM sends an Administrative_Deregistration SAR to the HSS.
3. The Oracle CSM sends a 504 Server Timeout to any ensuing registration refreshes by the endpoint.
4. The HSS sets the PUID to Not Registered and clears its S-CSCF association.
5. The HSS sends an SAA back to the Oracle CSM.
6. The Oracle CSM de-registers the user.
7. The Oracle CSM sends a NOTIFY messages to all REGEVENT subscribers indicating the de-registration event has taken place.

Note that the Oracle CSM can accept new registrations during the re-balance process. The process includes a time out at 30 minutes, after which the **release-user** command stops releasing users regardless of whether it has reached the configured user count. If the user issues the **release-users** command again, the Oracle CSM re-starts the process. After completion, the Oracle CSM echoes a message indicating the re-balance is complete.

 **Note:** If an HA switchover occurs before the release-users command has finished, the process does not continue to release users. If desired, the user can re-issue the command on the backup system after the switchover is complete.

I-CSCF Operation

As noted earlier, a device running the SLRM function can also act as an I-CSCF. All I-CSCF functions are 3GPP compliant.

Memory and CPU Overload Protection

The Oracle CSM protects itself from memory (heap) and CPU overload using configurable limits for their usage.

If the CPU usage exceeds the configured setting, the system sends a 503 error in response to any initial dialog request or standalone transactions. There are two memory (heap) related thresholds, the first of which generates 5xx replies, the second of which drops all messages.

This is true regardless of whether the system is performing SLRM or S-CSCF functions.

The Sc Interface

Oracle has define the Sc interface to define information exchange between the Oracle CSM and the SLRM. This is a custom diameter interface designed to include the following:

The Session Load Balancer and Route Manager

- Oracle CSM registration and deregistration on the SLRM function
- Capabilities negotiation between the Oracle CSM and the SLRM function
- KPIs that specify Oracle CSM status to the SLRM function
- Error information

Sc Interface Messages

The Sc Interface uses four message types to perform the SLRM function, including:

- Capabilities Exchange
- Device Watchdog
- Service Association
- Core Registration

Each message type uses a pre-defined request/answer sequence using timing that is dynamically managed and impacts the client-server as well as the load balance operational state between the SLRM and the Oracle CSM. Each sequence includes a response code in the answer message indicating if the request was a success or failure.

The high-level format, which shows message AVPs, for each of these messages is provided in the Sc Interface Appendix within this document. See RFC 6733 for detailed, generic information about Diameter message packet format and handling.

Capabilities Exchange Messages

The capabilities exchange message sequence, CER/CEA, is standard Diameter messaging used as a means of correlating client capabilities with server services. The CER message is used to discover peer's identity and exchange capabilities, including applications supported, vendor-Id and device addressing information. Key AVPs that must match include:

- Vendor-Id = 9148 (Oracle-Acme-Id)
- Vendor-Specific-Application-ID = 9999 (Oracle-Acme-Sc)

The Oracle CSM proceeds with presenting its cluster membership and registering its cores upon a successful CEA response.

Device Watchdog Messages

The device watchdog message sequence, DWR/DWA, is standard Diameter messaging used on idle connections to check peer availability and detect transport failures. Watchdog messaging can determine availability status between client and server. The sequence is initiated by both the SLRM function and the Oracle CSM depending on the devices' inter-operational state and the timing of the last successful exchanges.

On idle connections, this message is sent at a default interval of 60 seconds.

If watchdog messaging's is unable to confirm connectivity, the SLRM de-registers the applicable Oracle CSM and removes it from any load-balance pools.

Service Association Messages

The service association message sequence, SVR/SVA, is proprietary Diameter messaging that the Oracle CSM uses to advertise itself to an SLRM, specify its status in terms of service and capacity information, and remove its association with that SLRM. The process can be understood as a registration process. The Oracle CSM uses this messaging for the following purposes:

- INITIAL — On boot up, the Oracle CSM sends the request to advertise itself to the SLRM. This includes Oracle CSM's registration information. This information is updated every 20 seconds.
- REFRESH — The Oracle CSM uses this message to refresh/update its association (registration) with the SLRM. If the Capacity/Service info is not refreshed and it expires, the SLRM considers the Capacity/Service info to be invalid, so it changes the Oracle CSM status to "Out of Sync". Changes to the following trigger a refresh with updated info immediately:

- A change in service information, such as service-cluster-Id.
- The Oracle CSM cannot handle anymore endpoints.
- The Oracle CSM can handle endpoints again.
- TERM — The Oracle CSM uses this message to terminate its association (de-register) with the SLRM. On receiving this message, the SLRM removes the peer Oracle CSM and clear all state information.

The Oracle CSM specifies the service association request type in the SVR's Request Type AVP.

Core Registration Messages

The core registration exchange message sequence, CRR/CRA, is proprietary Diameter messaging that the Oracle CSM uses to populate, refresh and update its participation within SLRM cores. This messaging is used in the following registration scenarios:

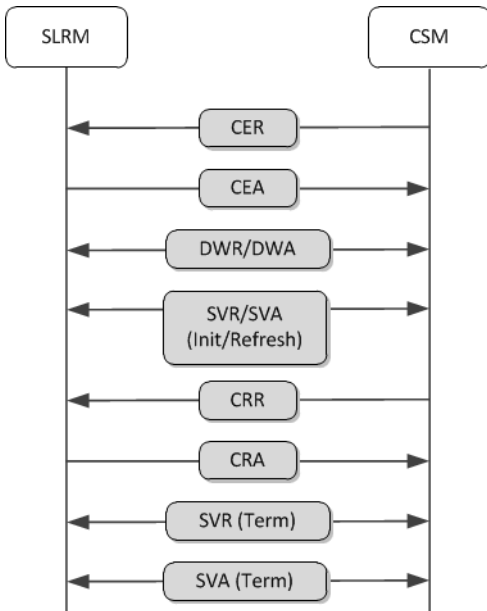
- REGISTRATION — The Oracle CSM sends the request to register the available cores to the SLRM. The registration refresh interval, which defaults to 60 seconds, is included in the Refresh-Interval AVP. If the registration is not refreshed and registration expires, then the SLRM considers that Oracle CSM to be timed out.
- DE-REGISTRATION — The Oracle CSM sends this request to de-register a core on SLRM. Upon receipt, the SLRM removes the specified core for the Oracle CSM and the Oracle CSM is considered to be no longer serving the core.
- RE-REGISTRATION — The Oracle CSM sends this message to refresh and update the registration of its cores. Upon receipt, the SLRM resets the expiration time and updates the core information. This message is sent at the default interval of 60 seconds. Additional notes on re-registration include:
 - Changes to Core-Info do not affect existing cache entries.
 - The systems use new values for future registrations/transactions.

The Oracle CSM specifies the registration scenario in the Registration Type AVP.

Sc Interface Messaging

The SLRM function uses the Sc interface to determine when and how to load balance Oracle CSMs. For the purposes of Diameter exchange, the Oracle CSM acts as client and the SLRM function acts as agent (server). This messaging includes standard Diameter exchanges, complemented with proprietary exchanges to handle all aspects of load balancing.

A typical message flow between SLRM and Oracle CSM is shown below.



Sc interface messaging procedures, from the perspective of the SLRM function, includes these steps.

The Session Load Balancer and Route Manager

1. The SLRM listens on a TCP socket for connection requests from peer Oracle CSMs.
2. After establishing a new Diameter connection, the SLRM waits for a CER from the peer CSM. The SLRM closes the connection if it does not get a CER after a timeout.
3. The SLRM exchanges the peer identity, supported vendor-ids and application-ids with the peer Oracle CSM within the CER/CEA exchange. A successful CER/CEA handshake creates a new peer relationship. If there is any error in the initial handshake, the SLRM sends an appropriate error response in the CEA and closes the connection.
4. The SLRM sends periodic DWR requests on idle connections, to check the availability of peer Oracle CSMs. The SLRM also responds to DWR's from peer Oracle CSM's
5. The SLRM responds to SVR requests sent by Oracle CSMs to advertise themselves to the SLRM.
6. The SLRM responds to CRR requests as follows.
 - On a new registration, the SLRM starts managing the cores for the Oracle CSM.
 - On re-registration, the SLRM refreshes and updates the information of the cores for the Oracle CSM.
 - On de-registration, the SLRM removes the core and stops managing the core for the Oracle CSM.

SLRM rejects bad message requests with error responses.

Sc interface messaging procedures, from the perspective of the Oracle CSM, includes these steps.

1. The Oracle CSM establishes a diameter session with peer SLRM. The initial handshake includes Diameter connection is setup and the capabilities exchange negotiation (CER/CEA).
2. The Oracle CSM sends periodic DWR requests on idle connections, to check the availability of the peer SLRM. The Oracle CSM also responds to DWR's from peer SLRMs.
3. The Oracle CSM advertises itself and sends periodic refreshes to update current Capacity using SVR/SVA requests.
4. The Oracle CSM periodically registers with the IMS cores that it is serving using CRR/CRA requests.
5. During shutdown, the Oracle CSM de-registers with the SLRM using a SVR/SVA (TERM) transaction.

Sc Interface Response Codes

The Oracle CSM and SLRM function insert a set of base protocol response codes to the Result-Code AVP of Response messages to indicate what has transpired based on the request. The sole Sc interface response code indicating success is DIAMETER_SUCCESS 2001.

- SC_DIAMETER_SUCCESS 2001
- SC_DIAMETER_FIRST_ASSOC 2002
- SC_DIAMETER_SUBSEQ_ASSOC 2003
- SC_DIAMETER_FIRST_REG 2004
- SC_DIAMETER_SUBSEQ_reg 2005

There are multiple response codes used to indicate failure, including:

- SC_DIAMETER_ERROR_CORE_NOT_FOUND 5001
- SC_DIAMETER_ERROR_PEER_NOT_FOUND 5002
- SC_DIAMETER_ERROR_PROTO_VER_MISMATCH 5003
- SC_DIAMETER_ERROR_DATABASE 5004
- SC_DIAMETER_ERROR_TIMEOUT 5005
- SC_DIAMETER_ERROR_UNABLE_COMPLY 5012

The format of each response message includes the response code AVP indicating one of the results above. Message format is provided in the Sc Interface Appendix. See RFC 6733 for detailed, generic information about Diameter response-code AVPs.

Proprietary SLRM AVP Descriptions

Req-Type AVP

The Req-Type AVP is of type Enumerated and indicates the type of registration (service association) requested by this SVR. The following values are defined:


- INITIAL (0) — This indicates the establishment of the association (eg, registration) between this Oracle CSM to the SLRM.
- REFRESH (1) — This indicates refreshes the association of this Oracle CSM to the SLRM.
- TERM (2) — This indicates the termination of the association (eg, de-registration) of this Oracle CSM to the SLRM.

Service-Cluster-Id AVP

Uniquely identifies the load balanced cluster to which this Oracle CSM belongs. The default cluster is zero. This AVP is included in the SVR request.


Pct-Used-CPU AVP

Indicates the percentage CPU used in a CSM. This AVP is included in SVR request.

 **Note:** Percentage CPU is provided by Oracle CSM for information purposes only. The SLRM displays it in show commands, but does not take any action based on this value.

Pct-Used-Mem AVP

Indicates the percentage memory of used in the Oracle CSM. This AVP is included in SVR request.

 **Note:** Percentage memory is provided by Oracle CSM for information purposes only. The SLRM displays it in show commands, but does not take any action on this value.

EP-Srv-Cnt AVP

Gives the number of endpoints currently serviced by the Oracle CSM. This AVP is included in the SVR request.

Proto-Ver AVP

Specifies the Sc interface version in SVR request. This AVP is included only in the initial SVR request.

Max-EPs-Supp AVP

Specifies the maximum number of contacts that a Oracle CSM can support. This number starts as the user-configured value on the Oracle CSM, but then is adjusted based on available system resources on the Oracle CSM.

Core-Reg-Type AVP

The Core-Reg-Type AVP is of type Enumerated and indicates the type of registration in the CRR request. The following values are defined:

- REGISTRATION (0) — This indicates registration of cores for the Oracle CSM.
- RE-REGISTRATION (1) — This indicates refresh/update/addition of cores for the Oracle CSM.
- DE-REGISTRATION (2) — This indicates de-registration of cores for the Oracle CSM.

Ims-Core AVP

Specifies the IMS core served by the Oracle CSM, as configured within the SIP registrar's ims-core setting. This AVP is included in the CRR request.

Srv-Assoc-ID AVP

This auto-generated string establishes an association relationship between the Oracle CSM and the SLRM. This AVP is included in the SVR request and answer.

Srv-Assoc-Exp

This AVP specifies the expiry time for the service association to which this Oracle CSM registered within this sequence. This AVP is included in the CRR request and is always 80 seconds with refreshes established via SVR sent every 20 seconds.

Core-Reg-Exp AVP

This AVP specifies the expiry time for the core registration to which this Oracle CSM registered within this sequence. This AVP is included in the CRR request and is always 240 seconds, with refreshes established via CRR sent every 60 seconds.

Soft-Ver AVP

Specifies the software version running on the Oracle CSM or SLRM. This AVP is included only in the initial SVR request.

Grouped AVPs

Oracle's Sc Diameter interface specifies grouped AVPs for use within its messaging. These AVPs are expanded below.

Core-Info AVP

A grouped AVP used to send service related information of a Oracle CSM in CRR messages. This grouped AVP contains the following AVPs:

- Srv-info — Service route of IMS core on the target Oracle CSM. This is a grouped AVP that includes the service info and service route AVPs.
- Ims-Core — IMS core served by Oracle CSM.

Srv-Info AVP

The Sc interface's service info AVP is a grouped AVP nested within the core-info AVP. It provides the SLRM function with the routes used to access the applicable ims-cores via this Oracle CSM. This grouped AVP has the following information in it.

- Service Info — Designation of this grouped AVP
- Service Route — The route to the target Oracle CSM

SLRM Configuration

This section explains how to configure functionality specific to the SLRM. It does not include configuration steps for elements that it shares in common with its corresponding Oracle CSMs (for example, **system-config**, **phy-interface**, **network-interface** and so forth).

SLRM configuration is quite simple. Aside from basic network connectivity, the service interfaces and the IMS core architecture, much of the configuration is otherwise learned dynamically learned from the Oracle CSMs that comprise the cluster.

Configuration elements include:

set-component-type—Defines operational behavior as either SLRM or CSM.

lb-interface—A multi-instance element identifying the Sc listening interface. There is typically only one **lb-interface** per device. Parameters include the local address, associated with realm, of the interface that the SLRM uses for SLRM signaling.

lb-core-cfg—A multi-instance element identifying every core the SLRM services, as well as the domains serviced within that core.

set-component-type

Use the **set-component-type** command to define the system's operational mode as either SLRM or CSM..

1. From superuser mode, use the following ACLI command sequence to access the **set-component-type** configuration element.

- **core-session-manager**—Defines the device as an I-CSCF and S-CSCF.
- **core-load-balancer**—Defines the device as an I-CSCF and SLRM.

The device responds by displaying user requirements for changing component type. If the user attempts to set the component type to the current component type, the system provides a message indicating this and takes no further action.

```
ORACLE# set-component-type core-load-balancer
WARNING: Changing component type is service impacting.
*****
  Ensure that you follow these steps if you choose to
  change the component type:

  1. Issue the delete-config command.
  2. Reboot.
*****
Continue with the change [y/n]?:
```

2. Type **y** to make the change. The system displays a message indicating the component type.
3. Be sure to **delete-config** and **reboot** to complete the procedure.

lb-interface

Use the following procedure to perform required **lb-interface** configuration.

1. From superuser mode, use the following ACLI command sequence to access the **lb-interface** configuration element.

```
ORACLE# configure terminal
ORACLE (configure) # session-router
ORACLE (session-router) # lb-interface
ORACLE (lb-interface) #
```

2. **name**—Use the **name** parameter to specify the name for this interface.
3. **state**—Enables or disables this interface for the Sc interface.
4. **address**—Specifies the IP address of the SLRM from which this Oracle CSM sends Sc interface traffic.
5. **port**—Specifies the port on the SLRM interface from which the Oracle CSM sends Sc interface traffic.
6. **realm**—Specifies the local realm to which this Sc interface applies.

lb-core-config

Use the following procedure to perform required **lb-core-config** configuration.

1. From superuser mode, use the following ACLI command sequence to access the **lb-core-config** configuration element.

```
ORACLE# configure terminal
ORACLE (configure) # session-router
ORACLE (session-router) # lb-core-config
ORACLE (lb-core-config) #
```

2. **core-name**—Use the **core-name** parameter to specify the name of this **lb-core-config**. This name must match the name of an **lb-config** on the Oracle CSM.
3. **state**—Enables or disables this **lb-core-config** instance.
4. **domains**—List of domains associated with this core. This list must match that of the corresponding registrar at the Oracle CSM.
5. **forwarding-realm**—Specifies the realm of the SLRM interface from which it sends Sc interface traffic.

The Session Load Balancer and Route Manager

6. **hss-config**—Specifies the name of the hss-config that matches the applicable HSS. The SRLM sends UARs and LIRs associated with this core to this HSS.

Note - The configuration options described in the Primary and Secondary ENUM Configuration section within the Diameter Oracle CSM chapter applies to the lb-core-config element. See that section for instructions on configuring those options here.

Oracle CSM Configuration

This section describes the configuration necessary to allow an Oracle CSM to join an SLRM load balanced cluster. Configuration is simplified to allow for an easy and seamless migration.

Configuration required at the Oracle CSM includes:

- **service-cluster-id**— A parameter within the system-config element that specifies the load balanced cluster to which this Oracle CSM belongs.
- **lb-cfg**—A multi-instance element identifying the Sc listening interface on the SLRM(s).
- **sip-registrar**—This configuration element has two applicable parameters.
 - **ims-core**—Parameter that specifies the matching SLRM core name to which this registrar applies.
 - **lb-cfg**—List of **lb-cfg** elements that this registrar uses to register its cores.

The following subsections explain these configurations.

service-cluster-id

Use the following procedure to perform **service-cluster-id** configuration within the system-config element on the Oracle CSM.

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

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

2. Select the **system-config** element. The **system-config** element is a single-instance element.
3. Use the **service-cluster-id** parameter to specify the load balanced cluster to which this Oracle CSM belongs. The default value is null, which ensures that this Oracle CSM can participate as a load balanced device with SLRM even though it has not been explicitly configured.

```
ORACLE(system-config)# service-cluster-id new_york
```

lb-cfg

Use the following procedure to perform required lb-cfg configuration.

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

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

2. **name**—A name for this **lb-cfg** element.
3. **state**—Enables or disables this configuration. When enabled, the Oracle CSM starts Sc interface signaling to this target SLRM.
4. **address**—Specifies the IP address of the SLRM to which this Oracle CSM sends Sc interface traffic.
5. **port**—Specifies the port on the SLRM interface to which the Oracle CSM sends Sc interface traffic.
6. **realm**—Specifies the local realm to which this Sc interface applies.

ims-core and lb-list

Use the following procedure to perform required **ims-core** and **lb-list** configuration within the selected **sip-registrar**.

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

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

2. Select the **sip-registrar** you intend to configure.
3. **ims-core**—Use the **ims-core** parameter to specify the core identification for this registrar. The domains supported by this **sip-registrar** must be the same as those in the SLRM's **lb-core-cfg** list.
4. **lb-list**—Use the **lb-list** parameter to specify an **lb-cfg** to communicate to the SLRM over the SC interface.

Releasing Users

Manual rebalancing consists of executing the release-users command from the Oracle CSM performing the SLRM function.

release-user

This command releases registered users from the Oracle CSM on which the command is issued. The SLRM rebalances the deployment by registering these users on another Oracle CSM upon the next registration cycle. This command only releases users up to the count specified. The process includes a time out at 30 minutes, after which the **release-user** command stops releasing users regardless of whether it has reached the configured user count. A user is only released if it is not in an active session.



Note: If an HA switchover occurs before the release-users command has finished, the process does not continue to release users. If desired, the user can re-issue the command on the backup system after the switchover is complete.

Parameter

- <count>** Specify the number of users that the system must release. The system marks this number of users for release, and begins to remove users until it reaches this number.
- stop** The system removes all users from the list of users currently marked for release.
- status** The system displays a list of all users marked for release from each cluster.

Path

release-user is an command available to superusers.

Release

First appearance: S-Cz7.1.5

Obtaining SLRM-Related Information

This section explains commands you can use to display or obtain SLRM load balance information. Methods of obtaining this information includes the **show load balancer** ACLI command and SNMP.

display-component-type

On an Oracle CSM, the **display-component-type** command shows the user the current operational mode as either **core-session-manager** or **core-load-balancer**.

Example

```
ORACLE# display-component-type
Component Type is: core-session-manager
SCZ715_64#
```

show load-balancer

On the device running the SLRM function and any load balanced Oracle CSMs, the **show load-balancer** command is the root command for displaying all load balance statistics. The various arguments the command supports narrows the output for clarity and specificity.

Arguments

stats [load balancer name]—Shows cumulative statistics on a per load-balancer basis. Adding a **load-balancer-name** as an argument narrows the output to the load-balancer specified.

members—Shows statistics on members in a cluster.

cores <core-name>—Shows load balance statistics on a per-core basis. Adding a **core-name** as an argument narrows the output to the core specified.

interface <argument>—Shows the cumulative statistics for all load balance interfaces on this device.

- **lb-interface-name**—Adding an **interface-name** as an argument narrows the output to the interface specified.
- **peer-address:port**—Adding **peer-address:port** as an argument narrows the output to the address/port specified.

Example

```
ORACLE# show load-balancer interface if3
```

show sipd endpoint-ip

The **show sipd endpoint-ip <user | IP address>** command displays information about each endpoint. For a supplied AoR, the Oracle CSM 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-s37q249kvluaa@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
```

SLRM MIB Objects and Traps

The following MIB objects and traps are supported for the Oracle CSM and its SLRM function. Please consult the *Scz7.1.2 MIB Reference Guide* for more SNMP information.

Oracle Communications System Management MIB (ap-corelb.mib)

The following table describes the SLRM-related SNMP GET query names for the Oracle Core Load Balancer MIB (ap-corelb.mib).

SNMP GET Query Name	Object Identifier Name: Number	Description
Object Identifier Name: apCoreLBModule (1.3.6.1.4.1.9148.3.19)		
Object Identifier Name: apCoreLBMIBObjects (1.3.6.1.4.1.9148.3.19.1)		
Object Identifier Name: apCoreLBMIBGeneralObjects (1.3.6.1.4.1.9148.3.19.1.1)		
apCoreLBMemberAddress	1.3.6.1.4.1.9148.3.19.1.1.1	This is the IP address of the CSM registered with the SLRM.
apCoreLBMemberAddressType	1.3.6.1.4.1.9148.3.19.1.1.2	This is the protocol version of the IP address of the CSM registered with the SLRM.
apCoreLBMemberPort	1.3.6.1.4.1.9148.3.19.1.1.3	This is the IP port of the CSM registered with the SLRM.
apCoreLBMemberId	1.3.6.1.4.1.9148.3.19.1.1.4	The cluster Id of the Core LB member.
apCoreLBReasonCode	1.3.6.1.4.1.9148.3.19.1.1.5	The reason for the core member failure. Values include service assoc terminated (0), service assoc timeout (1) and connection down (2).

SLRM Traps

The table below identifies traps that apply specifically to the SLRM function.

Trap Name: OID	Description
apCoreLBMemberOOSTrap	The system sends this trap when any member of a load balanced core is not responsive.
apCoreLBMemberInServiceTrap	The system sends this trap when any member of a load balanced core becomes responsive after failure.

The system sends the failure trap when the registered Oracle CSM's become unavailable for the following reasons:

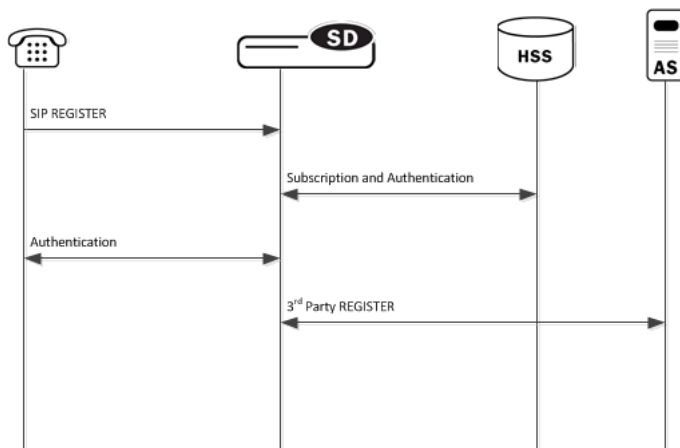
- The Sc interface goes down. (0)
- The member's registration expires. (1)
- The Oracle CSM does not respond to SIP requests. (3)

Query apCoreLBReasonCode to determine the reason code. The system sends the clear trap when the Oracle CSM becomes responsive again.

Third Party Registration

Third party registration support on the Oracle CSM 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 CSM accepts incoming REGISTER requests from UAs. After the UA has been registered with the Oracle CSM, the Oracle CSM sends a third party REGISTER message to a third party server.



The Oracle CSM supports third party registration via two methods:

- For scenarios in which UAs receive iFCs from the HSS and the Oracle CSM's default iFC configuration, the Oracle CSM 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 CSM and response from the Oracle CSM to the UA provided to them. The Oracle CSM 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 CSM and achieve third party registration support.
For these configurations, the Oracle CSM attempts third party registration to those servers for all UAs that register via the applicable Oracle CSM registrar.

For both methodologies, you must configure all third party servers as session agents.

Third Party Registrations via iFCs

The Oracle CSM 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 CSM. 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 CSM 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 CSM 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 CSM 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 CSM.
- The Oracle CSM 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 CSM 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 CSM.
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 CSM third party registration support via iFC as opposed to via ACLI configuration.

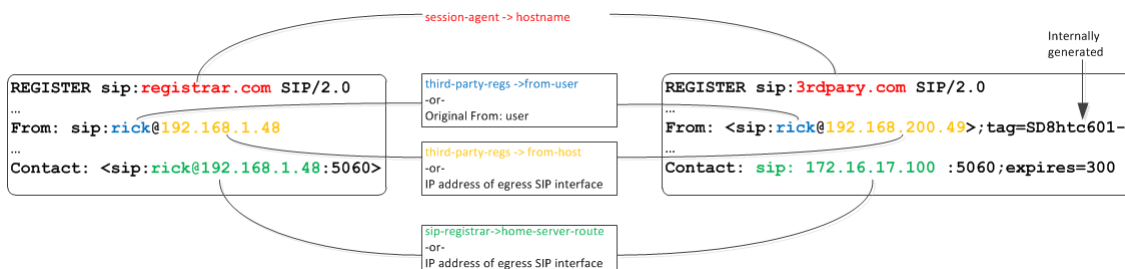
As is true of the method described above, third party registration is generated by the Oracle CSM on behalf of the user in the To: header of REGISTER request.

Third Party Registration

When third party registration is generated by CLI configuration on the Oracle CSM, 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 CSM 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 CSM 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 CSM 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 CSM adheres to standard SIP session agent retransmission/ timeout procedures. If the third party server is set to out of service, the Oracle CSM attempts connectivity retry procedures. The retry procedures dictate that the Oracle CSM 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 CSM maintains a queue of outstanding third party registration requests. When the third party server returns to service, the Oracle CSM 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 CSM 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 CSM 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 CSM refreshes the registration.

If the third party server responds to a REGISTER Request with a 423 (Interval Too Brief) response, the Oracle CSM 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 CSM-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 CSM Registrar functionality. It informs the Oracle CSM 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 CSM's egress SIP interface.
8. retry-interval—Enter the number of seconds the Oracle CSM 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.

SIP Signaling Services

This chapter explains how to configure the Oracle CSM to support Session Initiation Protocol (SIP) signaling services for hosted IP services applications. SIP is a text-based application-layer signaling protocol that creates, identifies, and terminates multimedia sessions between devices.

About the Oracle CSM and SIP


The Oracle CSM's support of SIP is broad in scope and varied in context. Beyond SIP interface configuration, the Oracle CSM offers a multitude of configurations that control its utilization of the SIP protocol. Engineers involved in network design and administration need to identify which configurations are applicable and effective in their own environment and then apply those controls using the Oracle CSM.

Recurse 305 Only Redirect Action

The Oracle CSM has a SIP feature called redirect action. This is a feature that allows the Oracle CSM, acting as a SIP Proxy or a Session Agent, to redirect SIP messages after receiving a SIP redirect (3xx) response. By default, the redirect-action parameter on the Oracle CSM's sip-interface and session-agent on the is set to recurse-305-only.

Redirect Action Process

When the redirect-action parameter is set to proxy, the Oracle CSM sends SIP Redirect responses back to the previous hop (back to the User Agent Client (UAC)) when the User Agent Server (UAS) is not a session agent. The URI in the Contact of the response is changed from the URI that was in the original request.

 **Note:** If the target of the request is a session agent, the session agent's redirect action supercedes that of the SIP interface.

When the redirect-action parameter is set to recurse, if the Oracle CSM receives a SIP redirect (3xx) response on the SIP interface, it automatically redirects all requests to the Contact URI specified in the 3xx response. The responses contain the same Contact URI that was in the original request sent to the UAS.

For example, if UAC X sends an INVITE to the Oracle CSM set up as a SIP proxy, the Oracle CSM forwards the INVITE to UAS Y (Y is not a session agent). Y then responds to the Oracle CSM with a 3xx response (redirection message) with the same URI that was in the original request. This indicates to the Oracle CSM that if it receives any future requests directed toward Y, that it should automatically redirect the request directly to Y. The Oracle CSM then recurses, or repeatedly sends subsequent incoming messages to the Contact URI specified in the Header of the 3xx responses.

SIP Signaling Services

When the redirect-action parameter is set to recurse-305-only, if the Oracle CSM receives a 305 SIP redirect response (Use Proxy) on the SIP interface, it automatically redirects all requests to the Contact URI specified in the 305 response. All other 3xx responses are sent back to the previous hop.

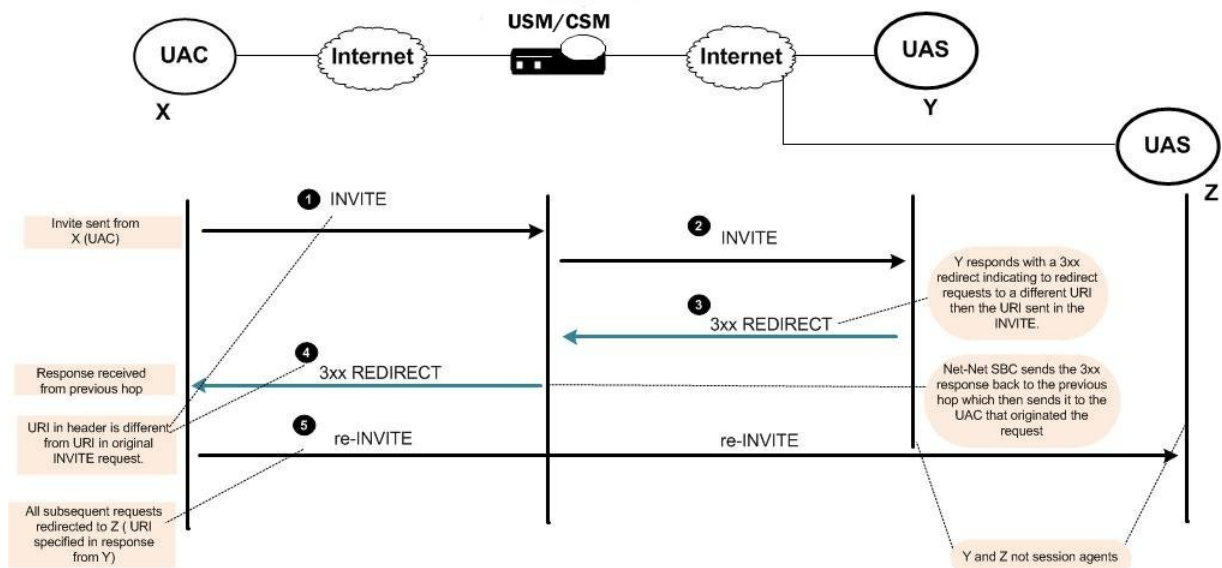
When the UAS is a session agent, the Oracle CSM can send the SIP redirect response back to the UAC using the value in the session agent's redirect action field. If there are too many UASs to define as individual session agents, or if the UASs are Hosted NAT Traversal (HNT) endpoints, and SIP redirect responses need to be proxied for UASs that are not session agents, you can set the behavior at the SIP interface level.

Redirect-Action Set to Proxy

The following occurs if you set the redirect-action parameter to proxy on the Oracle CSM:

1. X (UAC) sends an INVITE to the Oracle CSM.
2. The Oracle CSM forwards the INVITE to Y (UAS).
3. Y sends the 3xx REDIRECT response to the Oracle CSM with a different URI in the message header.
4. The Oracle CSM forwards the 3xx REDIRECT response to the previous hop. X receives the 3xx REDIRECT response from the previous hop.
5. X redirects all subsequent requests to the URI in the message header received from Y.

The following illustration shows an example of a dialog between X, Y, Z, and the Oracle CSM during a redirect-action session set to proxy.

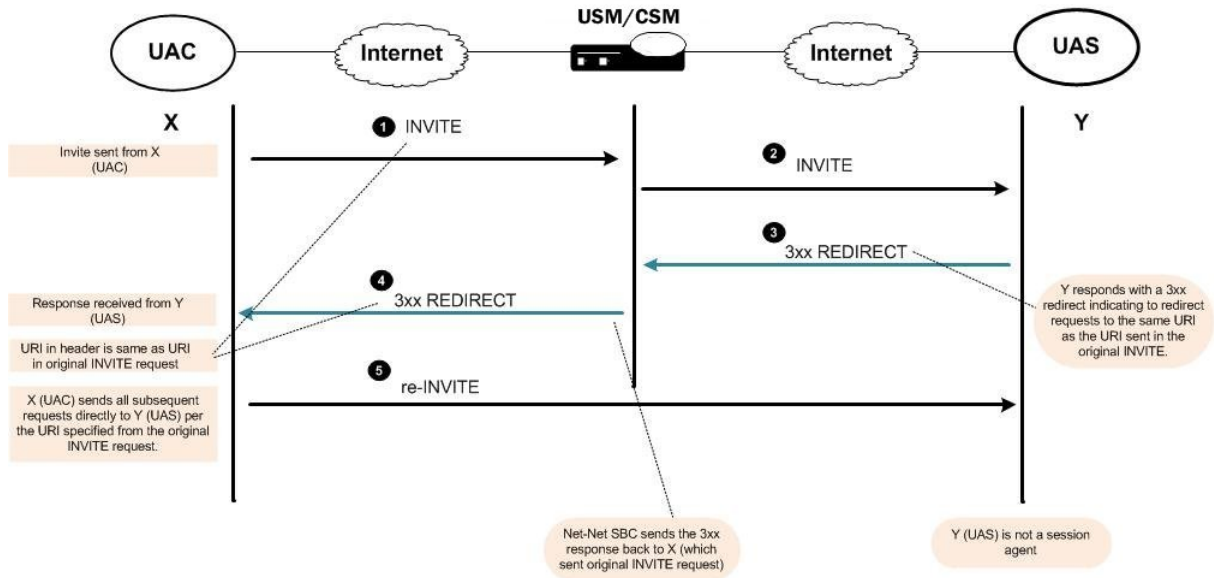


Redirect-Action Set to Recurse

The following occurs if you set the redirect-action parameter to recurse on the Oracle CSM:

1. X (UAC) sends an INVITE to the Oracle CSM.
2. The Oracle CSM forwards the INVITE to Y (UAS).
3. Y sends the 3xx REDIRECT response to the Oracle CSM with the same URI as the URI sent in the original request.
4. The Oracle CSM forwards the 3xx REDIRECT response to X (UAC).
5. X (UAC) sends all subsequent requests directly to Y (UAS) per the URI specified from the original INVITE request.

The following illustration shows an example of a dialog between X, Y, and the Oracle CSM during a redirect-action session set to recurse.

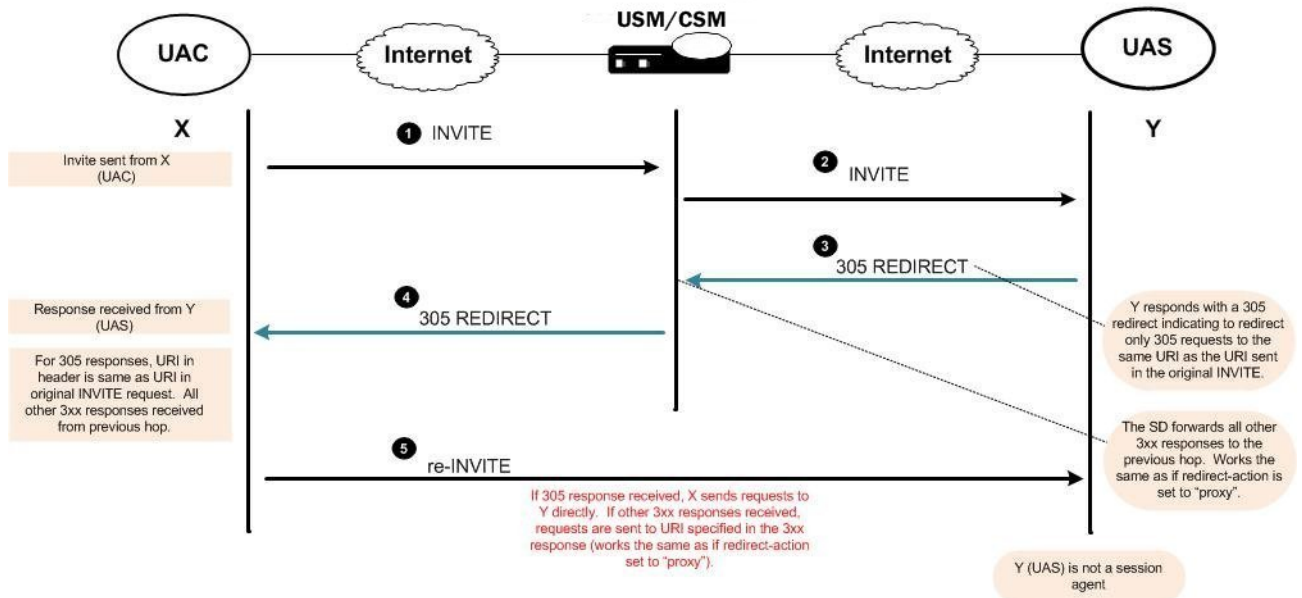


Redirect-Action Set to Recurse-305-Only

The following occurs if you set the redirect-action parameter to recurse-305-only on the Oracle CSM:

1. X (UAC) sends an INVITE to the Oracle CSM.
2. The Oracle CSM forwards the INVITE to Y (UAS).
3. Y sends a 305 REDIRECT response to the Oracle CSM with the same URI as the URI sent in the original request.
4. The Oracle CSM forwards the 305 REDIRECT response to X (UAC).
5. If 305 response received, X sends requests to Y directly. If other 3xx responses received, requests are sent to URI specified in the 3xx response (works the same as if redirect-action set to proxy).

The following illustration shows an example of a dialog between X, Y, and the Oracle CSM during a redirect-action session set to recurse-305-only.



Embedded Routes in Redirect Responses

When the Oracle CSM recurses as the result of a redirect (3xx) response, the server might need to specify one or more intermediate hops. These hops are reflected in the Contact header for the 3xx response using embedded route headers and look like this:

```
Contact: <sip:touser@server.example.com?Route=%3Cproxy.example.com%Blr%3E>
```

The Contact header shows that the request should be sent to server.example.com using proxy.example.com.

You can configure your Oracle CSM to specify that embedded headers in 3xx Contact headers are to be included in new requests such that they are tied to a session agent representing the new target (server.example.com). This behavior requires you to set the request-uri-headers parameter.

However, you can also use the use-redirect-route in global SIP configuration's options parameter so that the embedded Route header is used as the next hop to receive the new request.

When you configure this new option, the Oracle CSM constructs a new request using the redirect Contact, and the SIP URI from the Contact becomes the Request-URI. Then, the system inserts the embedded routes as Route headers in the, using the same order in which they appeared in the redirect Contact. Afterward, the Oracle CSM determines the next hop in the same way it does with any other request. If the first route is a loose route (i.e., it has the lr URI parameter), then the Oracle CSM sends a request to host indicated in the first route. Otherwise, strict routing applies, and the Oracle CSM sends the request to the host indicated in the Request-URI.

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

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

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

3. Type sip-config and press Enter.

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

If you are adding support for this feature to a pre-existing configuration, then you must select (using the ACLI select command) the configuration that you want to edit.

4. options—Set the options parameter by typing options, a Space, and then the option name.

```
ORACLE(sip-config)# options use-redirect-route
```

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

5. Save and activate your configuration.

SIP PRACK Interworking


When you configure your Oracle CSM with PRACK interworking for SIP, you enable it to interwork between endpoints that support RFC 3262, *Reliability of Provisional Responses in the Session Initiation Protocol*, and those that do not.

As its title indicates, RFC 3262 defines a reliable provisional response extension for SIP INVITEs, which is the 100rel extension tag. While some endpoints do not support the RFC, other SIP implementations require compliance with it. A session setup between two such endpoints fails. However, you can configure your Oracle CSM to supply the provisional response on behalf of endpoints that do not support it—and thereby enable sessions between those endpoints and the ones requiring RFC 3262 compliance.

You need to configure PRACK interworking for a SIP interface associated with the endpoints that need RFC 3262 support. To enable the feature, you set the 100rel-interworking option. The Oracle CSM applies PRACK

interworking for either the UAC or the UAS. The Oracle CSM checks to see whether or not it needs to apply PRACK interworking when an INVITE arrives at the ingress or egress SIP interface with the option enabled. First, it checks the Require header for the 100rel tag; if not found there, it checks the Supported header.

Since there is a slight difference in the application of this feature between the UAC and UAS, this section explains both.

 **Note:** If SDP is included in a PRACK request sent to a SIP interface where PRACK interworking is enabled, it will not be responded to, nor will any SDP be included in the locally-generated 200 OK to that PRACK.

UAC-Side PRACK Interworking

The Oracle CSM applies PRACK interworking on the UAC side when:

- A SIP INVITE does not contain a 100rel tag in a Require or Supported header
- The ingress SIP interface is enabled with the 100rel-interworking option
- The UAS fails to send reliable provisional responses

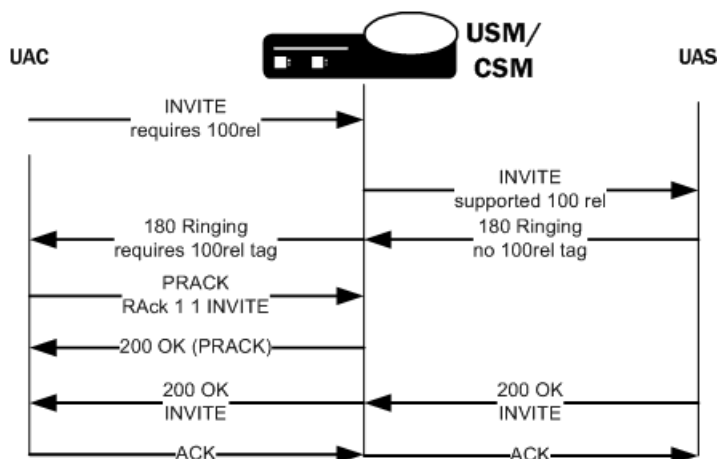
When it is to forward a non-reliable response to a UAC that requires RFC 3262 support, the Oracle CSM converts the non-reliable response to a reliable one by adding the 100rel tag to the Require header and adding an Rseq header to the response. Further, the Oracle CSM adds a Require header (complete with the 100rel tag) if there is not one already in the response, and then also adds Rseq header.

Note that the Oracle CSM sets the value of the Rseq header as 1 for the first provisional response, and then increments it by 1 for each subsequent provisional response. It also adds the PRACK method to the Allow header when that header appears.

The Oracle CSM retransmits the converted reliable provisional response in accordance with RFC 3262, until it receives a PRACK request. For the initial timeout for retransmission, the Oracle CSM uses the value you set in the init-timer parameter in the global SIP configuration. It stops retransmitting when either it receives a transmission, or when the ingress SIP interface's trans-expire timer elapses.

If it never receives a PRACK, the Oracle CSM does not generate an error response to the INVITE, relying instead on the downstream UAS to produce a final response.

The call flow for this application looks like this:



UAS-Side PRACK Interworking

The Oracle CSM applies PRACK interworking on the UAS side when:

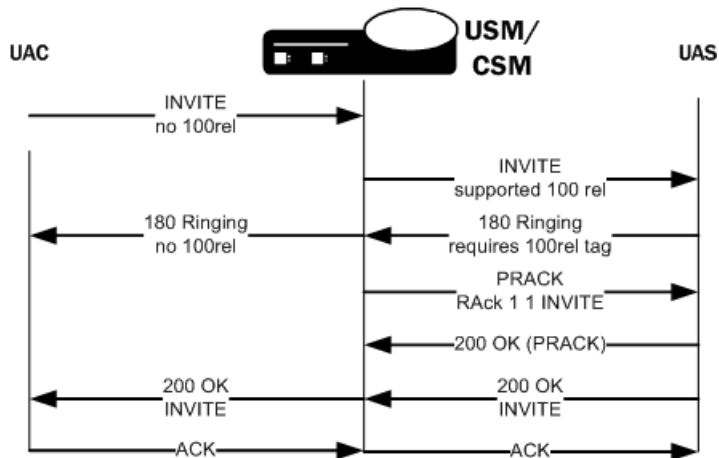
- A SIP INVITE contains the 100rel tag in a Require or Supported header
- The egress SIP interface is enabled with the 100rel-interworking option
- The UAS does send reliable provisional responses

SIP Signaling Services

When the UAC does not support RFC 3262, the Oracle CSM generates a PRACK request to acknowledge the response. It also converts the response to non-reliable by removing the 100 rel tag from the Require header and removing the RSeq header from the response.

In the case of the UAS, the Oracle CSM matches the PRACK to a converted reliable provisional response using the PRACK's RACK header. If it finds a matching response, the Oracle CSM generates a 200 OK to the PRACK. And if it finds no match, then it generates a 481 Call Leg/Transaction Does Not Exist response. The Oracle CSM generates a 400 Bad Request response if either the RACK is not in the PRACK request or it is not formatted properly.

The call flow for this application looks like this:



PRACK Interworking Configuration

You enable PRACK interworking for ingress and egress SIP interfaces. Be sure you know on what side, ingress or egress, you need this feature applied.

To configure PRACK interworking for a SIP interface:

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

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

2. Type session-router and press Enter.

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

3. Type sip-interface and press Enter. If you are editing an existing configuration, select the one on which you want to enable this feature.

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

4. options—Set the options parameter by typing options, a Space, the option name 100rel-interworking with a plus sign in front of it, and then press Enter.

```
ORACLE (sip-interface) # options +100rel-interworking
```

If you type options and then the option value for either of these entries without the plus sign, you will overwrite any previously configured options. In order to append the new option to this configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.

5. Save and activate your configuration.

Global SIP Timers

This section explains how to configure SIP retransmission and expiration timers.



Note: you can also set timers and counters per SIP interface.

Overview

SIP timers define the transaction expiration timers, retransmission intervals when UDP is used as a transport, and the lifetime of dynamic TCP connections. The retransmission and expiration timers correspond to the timers defined in RFC 3261.

- **init timer:** is the initial request retransmission interval. It corresponds to Timer T1 in RFC 3261.

This timer is used when sending requests over UDP. If the response is not received within this interval, the request is retransmitted. The retransmission interval is doubled after each retransmission.

- **max timer:** is the maximum retransmission interval for non-INVITE requests. It corresponds to Timer T2 in RFC 3261.

The retransmission interval is doubled after each retransmission. If the resulting retransmission interval exceeds the max timer, it is set to the max timer value.

- **trans expire:** is the transaction expiration timer. This value is used for timers B, D, F, H and J as defined in RFC 3261.
- **invite expire:** defines the transaction expiration time for an INVITE transaction after a provisional response has been received. This corresponds to timer C in RFC 3261.

If a final response is not received within this time, the INVITE is cancelled. In accordance with RFC 3261, the timer is reset to the invite expire value when any additional provisional responses are received.

- **Inactive dynamic conn timer** defines the idle time of a dynamic TCP connection before the connection is torn down. Idle is defined as not transporting any traffic. There is no timer in RFC 3261 corresponding to this function.

Timers Configuration

To configure timers:

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 `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. **init-timer**—Enter the initial timeout value in milliseconds for a response to an INVITE request, and it applies to any SIP request in UDP. In RFC 3261, this value is also referred to as `TIMER_T1`. The default is 500. The valid range is:

- Minimum—0
- Maximum—999999999

5. **max-timer**—Enter the maximum transmission timeout (T2) for SIP in milliseconds.

When sending SIP over UDP, a re-transmission timer is used. If the timer expires and the message is re-transmitted, the re-transmission timer is then set to twice the previous value (but will not exceed the maximum timer value). Using the default values of 500 milliseconds and 4000 milliseconds, the re-transmission timer is 0.5, then 1, 2, and finally 4. The incrementing continues until the transmission expire timer activates. The default is 4000. The valid range is:

- Minimum—0
- Maximum—999999999

SIP Signaling Services

6. `trans-expire`—Enter the transaction expire timeout value (Timer B) in seconds to set the time for SIP transactions to live. The same value is used for Timers D, F, H and J. The default is 32. The valid range is:
 - Minimum—0
 - Maximum—999999999
7. `invite-expire`—Enter the invite expire timeout value (Timer C) in seconds to indicate the time for SIP client transaction will live after receiving a provisional response. The default is 180. The valid range is:
 - Minimum—0
 - Maximum—999999999
8. `inactive-dynamic-conn`—Enter the inactive dynamic connection value in seconds to set the time limit for inactive dynamic connections.

If the connection between the SIP proxy and a session agent is dynamic (for example, through dTCP), and the connection has been idle for the amount of time specified here, the SIP proxy breaks the connection. Idle is defined as not transporting any traffic. The default value is 32. The valid range is:

- Minimum—0
- Maximum—999999999



Note: Setting this parameter to 0 disables this parameter.

The following example shows SIP config timer values for a peering network. Some parameters are omitted for brevity.

```
sip-config
  state                               enabled
  operation-mode                       dialog
dialog-transparency                    disabled
home-realm-id                          acme
  egress-realm-id
nat-mode                                Public
registrar-domain
registrar-host
registrar-port                          0
init-timer                              500
max-timer                               4000
trans-expire                            32
invite-expire                           180
inactive-dynamic-conn                   32
```

SIP Timers Discreet Configuration

Previous releases controlled various SIP timers with a single CLI command, `trans-expire`, available in both `sip-config` and `sip-interface` modes. When executed in `sip-config` mode, the command essentially established a global default transaction expiration timer value. Executed at the `sip-interface` level, the command established a local, interface-specific value that overrode the global default.

Specific timers controlled by `trans-expire` are as follows:

Timer B, the INVITE transaction timeout timer, defined in Section 17.1.1.2 and Appendix A of RFC 3261, *SIP: Session Initiation Protocol*.

Timer D, the Wait-Time for response retransmits timer, defined in Section 17.1.1.2 and Appendix A of RFC 3261, *SIP: Session Initiation Protocol*.

Timer F, the non-INVITE transaction timeout timer, defined in Section 17.1.2.2 and Appendix A of RFC 3261, *SIP: Session Initiation Protocol*.

Timer H, the Wait-Time for ACK receipt timer, defined in Section 17.2.1 and Appendix A of RFC 3261, *SIP: Session Initiation Protocol*.

Timer J, the Wait-Time for non-INVITE requests timer, defined in Section 17.2.2 and Appendix A of RFC 3261, *SIP: Session Initiation Protocol*.

A new ACLI command (`initial-inv-trans-expire`) that enables user control over SIP Timer B for initial INVITE transactions. Other timers, namely B for non-initial INVITES, D, F, H, and J remain under the control of `trans-expire`.

Use `initial-inv-trans-expire` in the `sip-config` configuration mode, to establish a global, default transaction timeout value (expressed in seconds) used exclusively for initial INVITE transactions.

```
ORACLE (sip-config) # initial-inv-trans-expire 4
ORACLE (sip-config) #
```

Allowable values are integers within the range 0 (the default) through 999999999. The default value, 0, indicates that a dedicated INVITE Timer B is not enabled. Non-default integer values enable a dedicated Timer B and set the timer value.

The default value retains compatibility with previous operational behavior in that Timers B, D, F, H, and J all remain subject to the single timer value set by `trans-expire`. However, when `initial-inv-trans-expire` is set to a supported non-zero value, SIP Timer B as it applies to initial INVITES, assumes that value rather than the value assigned by `trans-expire`. This functionality is available in both `sip-config` and in `sip-interface` objects.

If a dedicated Timer B is enabled at the `sip-config` level, you can use `initial-inv-trans-expire` in the `sip-interface` configuration mode, to establish a local interface-specific Timer B timeout value that overrides the global default value.

```
ORACLE (sip-interface) # initial-inv-trans-expire 8
ORACLE (sip-interface) #
```

Session Timer Support

The Oracle CSM partially supports RFC4028 by establishing session timers without participating in the session timer negotiation.

When a 2xx response to a Session Refresh Request arrives, the Oracle CSM will start a new timer or refresh the existing timer using the value of the `Session-Expires` header. When the session timer expires, the Oracle CSM will send a BYE to both the upstream and downstream endpoints.

When accounting is configured, the Oracle CSM will also send a RADIUS STOP record with `Acct-Terminate-Cause=Session-Timeout`.

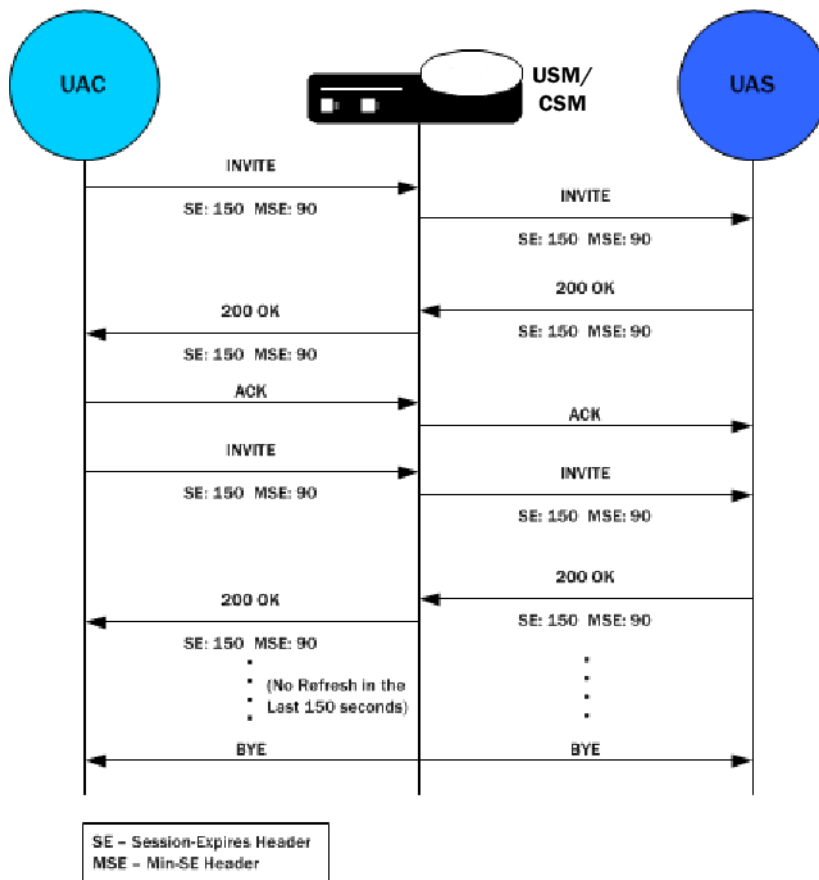
Call Flow Example

The UAS obtains the value from the `Session-Expires` header field in a 2xx response to a session refresh request that it sends.

Proxies and UACs obtain this value from the `Session-Expires` header field in a 2xx response to a session refresh request that they receive.

Once the session timer runs out, the Oracle CSM sends a BYE to both the UAC and the UAS to clear the session.

Enable this feature by adding the `session-timer-support` option to the `sip config`.



SIP Options Tag Handling

This section explains how to configure SIP options on a global or per-realm level and how to specify whether the feature treatment applies to traffic inbound to or outbound from a realm, or both.

SIP extensions that require specific behavior by UAs or proxies are identified by option tags. Option tags are unique identifiers used to designate new options (for example, extensions) in SIP. These option tags appear in the Require, Proxy-Require, and Supported headers of SIP messages.

Option tags are compatibility mechanisms for extensions and are used in header fields such as Require, Supported, Proxy-Require, and Unsupported in support of SIP.

The option tag itself is a string that is associated with a particular SIP option (i.e., an extension). It identifies this option to SIP endpoints.

Overview

The SIP specification (RFC 3261) requires that the Oracle CSM B2BUA reject any request that contains a Require header with an option tag the Oracle CSM does not support. However, many of these extensions operate transparently through the Oracle CSM's B2BUA. You can configure how SIP defines the Oracle CSMs B2BUA treatment of specific option tags.

Also, there might be certain extensions that an endpoint indicates support for by including the option tag in a Supported header. If you do not want a given extension used in your network, the you can configure SIP option tag handling to remove the undesired option tag from the Supported header. You can also specify how option tags in Proxy-Require headers are to be treated.

Configuration Overview

You configure the SIP feature element to define option tag names and their treatment by the Oracle CSM when the option tag appears in a Supported header, a Require header, and a Proxy-Require header. If an option tag is encountered that is not configured as a SIP feature, the default treatments apply. You only need to configure option tag handling in the SIP feature element when non-default treatment is required.

You can specify whether a SIP feature should be applied to a specific realm or globally across realms. You can also specify the treatment for an option based on whether it appears in an inbound or outbound packet. Inbound packets are those that are coming from a realm to the Oracle CSM and outbound packets are those which are going from the Oracle CSM to the realm.

The following tables lists the SIP option tag parameters you need to configure.

Parameter	Description
name	SIP feature tag name
realm	Realm name with which the feature will be associated. To make the feature global, leave the field empty.
support mode inbound	Action for tag in Supported header in an inbound packet.
require mode inbound	Action for tag in Require header in an inbound packet
proxy require mode inbound	Action for tag in Proxy-Require header in an inbound packet
support mode outbound	Action for tag in Supported header in an outbound packet
require mode outbound	Action for tag in Require header in an outbound packet
proxy require mode outbound	Action for tag in Proxy-Require header in an outbound packet

SIP Option Tag Handling Configuration

To configure SIP option tag handling:

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

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

From this point, you can configure SIP option tags parameters. To view all sip-feature parameters, enter a `?` at the system prompt.

4. `name`—Enter a name for the option tag that will appear in the Require, Supported, or Proxy-Require headers of inbound and outbound SIP messages.

You must enter a unique value.



Note: Valid option tags are registered with the IANA Protocol Number Assignment Services under Session Initiation Protocol Parameters. Because option tags are not registered until the SIP extension is published as a RFC, there might be implementations based on Internet-Drafts or proprietary implementations that use unregistered option tags.

5. `realm`—Enter the name of the realm with which this option tag will be associated. If you want to apply it globally across realms, leave this parameter blank.
6. `support-mode-inbound`—Optional. Indicate the support mode to define how the option tag is treated when encountered in an inbound SIP message's Supported header. The default value is `pass`. Valid values are:

SIP Signaling Services

- pass—Indicates the B2BUA should include the tag in the corresponding outgoing message.
 - strip—Indicates the tag should not be included in the outgoing message. Use strip if you do not want the extension used.
7. require-mode-inbound—Optional. Indicate the require mode to define how the option tag is treated when it is encountered in an inbound SIP message’s Require header. The default value is reject. The valid values are:
 - pass—Indicates the B2BUA should include the tag in the corresponding outgoing message.
 - reject—Indicates the B2BUA should reject the request with a 420 (Bad Extension) response. The option tag is included in an Unsupported header in the reject response.
 8. require-mode-inbound—Optional. Indicate the require proxy mode to define how the option tag is treated when encountered in an incoming SIP message’s Proxy-Require header. The default is reject. The valid values are:
 - pass—Indicates the B2BUA should include the tag in the corresponding outgoing message.
 - reject—Indicates the B2BUA should reject the request with a 420 (Bad Extension) response. The option tag is included in an Unsupported header in the reject response.
 9. support-mode-outbound—Optional. Indicate the support mode to define how the option tag is treated when encountered in an outbound SIP message’s Supported header. The default value is pass. Valid values are:
 - pass—Indicates the B2BUA should include the tag.
 - strip—Indicates the tag should not be included in the outgoing message. Use strip if you do not want the extension used.
 10. require-mode-outbound—Optional. Indicate the require mode to define how the option tag is treated when it is encountered in an outbound SIP message’s Require header. The default value is reject. Valid values are:
 - pass—Indicates the B2BUA should include the tag.
 - reject—Indicates the B2BUA should reject the request with a 420 (Bad Extension) response. The option tag is included in an Unsupported header in the reject response.
 11. require-mode-outbound—Optional. Indicate the require proxy mode to define how the option tag is treated when encountered in an outgoing SIP message’s Proxy-Require header. The default value is reject. The valid values are:
 - pass—Indicates the B2BUA should include the tag.
 - reject—Indicates the B2BUA should reject the request with a 420 (Bad Extension) response. The option tag is included in an Unsupported header in the reject response.

The following example shows SIP option tag handling configured for non-default treatment of option tags.

```
sip-feature
  name                newfeature
  realm               peer-1
  support-mode-inbound Strip
  require-mode-inbound Reject
  proxy-require-mode-inbound Pass
  support-mode-outbound Pass
  require-mode-outbound Reject
  proxy-require-mode-outbound Reject
  last-modified-date  2004-12-08 03:55:05
```

SIP Options

This section explains how you can configure a limited list of specialized SIP features and/or parameters called options. The options described here were developed to meet specific needs not addressed by the standard SIP configuration parameters. Not all users have a need for these options.



Note: Oracle recommends checking with your Oracle representative before applying any of these options.

Overview

You can configure options for the SIP configuration and SIP interface. Both elements include a parameter (options) that you use to configure the options.

Global SIP Options

The following table lists the SIP options supported by the Oracle CSM (CSM).

Option	Description
add-error-to-tag=no	If present (even when set to no), suppresses the addition of an Acme tag on 3xx-6xx responses.
add-prov-to-tag=no	Prevents the CSM from adding a tag parameter to the To header (to-tag) to non-100 provisional responses to INVITE requests. Used when a provisional (101-199) response is received from the UAS on a client transaction without a to-tag. By default, the CSM adds the tag cookie in the response (as though it had a tag) sent back to the UAC for the associated server transaction. When you include this option in the SIP configuration, and the response from the UAS does not have a to-tag, the response forwarded to the UAC will not have a to-tag.
add-reg-expires	Causes an Expires header to always be included in a REGISTER response with the registration caching and HNT traversal functions of the CSM. Use for endpoints that do not understand the Expires parameter in the Contact header.
add-ruri-user=<methods>	Causes a userinfo portion to be added to a Request-URI when one is not present. Used to support the OKI phone, which registers a Contact of just an IP-Address but rejects initial INVITEs if the Request_URI does not have a userinfo part. <methods> is a comma-separated list of methods to which the option should apply. If more than one method is listed, the list must be enclosed in quotes. This option only applies to out-of-dialog requests (no tag parameter in the To header). However, if ACK is listed, it will apply to all ACK requests because an ACK is always supposed to have a to-tag.
allow-notify-no-contact	Prevents the CSM from rejecting NOTIFYs with a 400 Bad Request response. NOTIFY requests without Contact header are allowed to pass through the CSM instead.
call-id-host=<host>	Causes the CSM to include a host part (ID@host) in the Call-ID it generated. <host> is the hostname (or IP address) that is to appear in the host part of the Call-ID. If not specified, the SIP port address is used.
contact-endpoint=<param-name>	Defines a URL parameter to report the real Contact address of an endpoint in a REGISTER message forwarded to a registrar, when the CSM is caching registration. (plain or HNT). If <param-name> is not specified, the default value endpoint is used. This parameter is added as a URL parameter in the Contact on the REGISTER message. In order for the registration cache to work properly, the softswitch/registrar is expected to include the endpoint parameter in the Request-URI of a SIP request it forwards to the address-of-record.

SIP Signaling Services

Option	Description
contact-firewall=<param-name>	<p>Defines a URL parameter to report the NAT between the CSM and the real Contact address of an endpoint in a REGISTRAR message forwarded to a registrar when the CSM is doing registration caching for NHT.</p> <p>If <param-name> is not specified, the default value firewall is used.</p> <p>This parameter will be added as a URL parameter in the Contact on the REGISTER message.</p> <p>In order for the registration cache to work properly, the softswitch/registrar is expected to include the endpoint parameter in the Request-URI of any SIP request it forwards for the address-of-record.</p>
disable-privacy	<p>Prevents the change of the P-Preferred-Identity to P-Asserted-Identity and lets the P-Preferred-Identity go through unchanged.</p>
drain-sendonly	<p>Causes the CSM to examine the SDP attributes and change sendonly mode to sendrecv. This causes the endpoint receiving the SDP to send RTP, which is required for HNT traversal endpoints to work with media servers. The CSM sets up the flow so that RTP coming from the endpoint are dropped to prevent the UA that sent the sendonly SDP from receiving packets.</p> <p>See the option video-sbc-session also.</p>
encode-contact=<prefix>	<p>Causes the CSM to encode Contact addresses into the userinfo part of the URI. It applies only to Contact address that usually get the maddr parameter. Use when the CSM needs requests sent to the URI in the Contact sent instead to the CSM. The host part of the URI will have the CSM's address.</p> <p>The <prefix> serves as a place between the original userinfo and the encoded address. If a <prefix> is specified, a default of +SD is used. Without this option, the CSM adds a maddr parameter.</p>
fix-to-header	<p>For requests that have the CSM address in both the Request-URI and the To-URI, it sets the hostport of the To-URI to a local policy's next hop target on out-of-dialog requests (no to-tag).</p> <p>This is the default IWF behavior, even without this option configured.</p>
forward-reg-callid-change	<p>Addresses the case when an endpoint reboots and performs a third party registration before its old registration expires. During this re-registration, the contact header is the same as it was pre-reregistration. As a consequence of the reboot, the SIP Call-ID changes.</p> <p>In this situation, the CSM does not forward the REGISTER to the registrar, because it believes the endpoint is already registered, based on a previous registration from the same Contact: header URI.</p> <p>To remedy this problem, the CSM now keeps track of the Call-ID in its registration cache. A new option in the SIP interface configuration element forces the CSM to forward a REGISTER message to the registrar when the Call-ID header changes in a REGISTER message received from a reregistering UAC.</p>
global-contact	<p>Addresses interoperability in the Dialog and Presence event packages that are used in hosted PBX and IP Centrex offerings. This option</p>

Option	Description
	<p>enables persistent URIs in the Contact headers inserted into outgoing SIP messages.</p> <p>If this option is not used, URIs placed in the Contact header of outgoing messages are only valid within the context of the dialog to which the message is associated.</p>
ignore-register-service-route-oos	Prohibits a Register message from using a service route if that service route is an out-of-service session agent.
load-limit=<cpu percentage>	Defines the CPU usage percentage at which the CSM should start rejecting calls. Default value is 90%.
lp-sa-match=<match strategy>	Changes the ways local policies and session agents match; accounts for realm in matching process. Strategy choices are: all, realm, sub-realm, interface, and network.
max-register-forward=<value>	<p>Defines a limit (as assigned in the value field) of REGISTERs to be forwarded to the registrar.</p> <p>During each second, the sipd counts how many REGISTERs have been sent to the registrar. It checks the threshold when it receives a REGISTER from the UA and determines that less than half the real registration lifetime is left. If the number of REGISTERs forwarded (new and updates) in the current second exceeds the configured threshold, it will respond to the UA from the cache.</p>
max-register-refresh=<value>	<p>Defines the desired limit of REGISTER refreshes from all the UAs. Each second of time, sipd counts the number of REGISTER/200-OK responses sent back. When the threshold is exceeded, it increments the expire time (based on NAT interval) by one second and resets the count.</p> <p>By default no threshold is applied. The recommended value is somewhat dependent on the CSM hardware used, but 300 can be used as an initial value.</p>
max-routes=<number of routes>	<p>Restricts the number of routes through which the sipd will iterate from a local policy lookup. For example, setting this option to 1 causes the CSM to only try the first, best, route. Setting this option to 0, or omitting it, lets the CSM use all of the routes available to it (with the priority scheme for route matching).</p> <p>When you test a policy using the test-policy CLI command, this option is not recognized and all options that match the criteria are displayed.</p>
max-udp-length=<maximum length>	<p>Setting this option to zero (0) forces sipd to send fragmented UDP packets. Using this option, you override the default value of the maximum UDP datagram size (1500 bytes; sipd requires the use of SIP/TCP at 1300 bytes).</p> <p>You can set the global SIP configuration's max-udp-length=x option for global use in your SIP configuration, or you can override it on a per-interface basis by configuring this option in a SIP interface configuration.</p>
media-release=<header-name>[;<header-param>]	Enables the multi-system media release feature that encodes IP address and port information for the media streams described by SDP. It lets another CSM decode the data to restore the original SDP, which allows

Option	Description
	<p>the media to flow directly between endpoints in the same network (that is serviced by multiple CSMs).</p> <p>The media release information can appear in the following places:</p> <ul style="list-style-type: none"> • SIP header P-Media-Release: <encoded-media-interface-information> • Header parameter on a SIP header Contact: <sip:1234@abc.com> ; acme-media=<encoded-media-interface-information> • SDP attribute in the message body a=acme-media: <encoded-media-interface-information> <p>Option includes the following:</p> <ul style="list-style-type: none"> • <header-name> is SIP header in which to put the information or the special value sdp, which indicates the information should be put into the SDP. • <header-param> is the header parameter name in which to put the information or in the case of the special header name value sdp, it is the SDP attribute name in which to put the information. <p>They identify to where the encoded information is passed. If you do not specify a header, P-Media-Release is used.</p>
no-contact-endpoint-port	<p>Enables the CSM to add a URL parameter (defined as an argument to the contact-endpoint option) to the Contact headers of REGISTER messages that it forwards to the registrar when it performs registration caching. The value of the contact-endpoint URL parameter is the real address of the endpoint; and if the endpoint is behind a NAT, this includes the IP address and a port number. However, not all network entities can parse that port number, which is included unconditionally. This feature allows you to configure the exclusion of the port number.</p> <p>Despite the fact that you set this parameter in the global SIP configuration, it is applied only to SIP interfaces. However, you can set a contact-endpoint option in the realm configuration, on which this new parameter has no effect.</p>
refer-to-uri-prefix=<prefix>	<p>Defines a prefix to be matched against the userinfo part of Contact headers (config=), of which the CSM should create a B2BUA map. This ensures that outgoing messages include the correct userinfo value. This option is used to enable add-on conferencing.</p>
reg-cache-mode=<mode>	<p>Affects how the userinfo part of Contact address is constructed with registration caching. <mode> values are:</p> <ul style="list-style-type: none"> • none: userinfo from the received (post NAT) Contact is retained • from: userinfo from the From header is copied to the userinfo of the forwarded Contact header • append: append the UA's Contact address into a cookie appended to the userinfo from the original Contact userinfo. For HNT, the NAT/firewall address is used. • append-from: takes userinfo from the From header and appends the encrypted address to the userinfo from the original Contact userinfo. For HNT, the NAT/firewall address is used.

Option	Description
	<p>The from mode is used with softswitches that do not use the cookies used by the CSM. It also helps limit the number of bytes in the userinfo; which might create duplicate contacts. For example, if the CSM address is 1.2.3.4, both 1234@5.6.7.8 and 1234@4.3.2.1 will result in a CSM contact of 1234@5.6.7.8.</p>
reg-contact-user-random	<p>Support the SIP random registered-contact feature. Gives the CSM the ability to support endpoints that randomly change their contact usernames every time they re-register. Only applicable to operators who need to support the Japan TTC standard JJ-90.22 in specific applications.</p> <p>Applies to cases when an endpoint re-registers with a different contact username, but with the same hostname/IP address and the same address of record (AoR). Without this feature enabled, the CSM forwards every re-registration to the registrar with the new contact information without it being considered a registration refresh. The CSM forwards it to the Registrar using the same sd-contact as the previous registration.</p> <p>When you set this option, the CSM does treat such a re-registration as a registration refresh when it is received prior to the half-life time for the specific contact. The CSM also uses the new contact username for the Request-URI in requests it sends to the UA, and verifies that the UA uses the correct one when that CSM is set to allow-anonymous registered mode.</p> <p>NOTE: The registration cache mode is set using the option reg-cache-mode, but regardless of how you configure it, the registration cache mode will be set to contact when SIP random registered-contact feature is enabled.</p>
register-grace-timer	<p>Makes the grace time for the SIP Registration configurable. You can configure the grace timer in seconds.</p>
reinvite-trying=[yes]	<p>Causes the CSM to send a 100 Trying for re-INVITEs, which is normally suppressed. If you enter the option name but omit the value yes, the option is still active.</p>
reject-interval=<value>	<p>Acts as a multiplier to increase the value presented to the UAC in the Retry-After field. For example, if reject-interval=5 (reject interval is set to 10); at a 90% rejection rate the CSM sends Retry-After: 45.</p> <p>When rejecting calls because of CPU load limiting, the CSM adds a Retry-After parameter to the error response (typically 503 Service Unavailable). By default the CSM sets the Retry-After value to be 1/10th of the current rejection rate.</p>
reject-register=[no refresh]	<p>Allows REGISTER messages through even during load limiting. By default, REGISTER messages are subject to load limiting.</p>
response-for-not-found=<response code>	<p>Change the 404 Not Found generated by the CSM to a different response code.</p>
route-register-no-service-route	<p>Controls how a UA is registered. Option can have three values:</p> <ul style="list-style-type: none"> route-register-no-service-route—This option prevents the use of the Service-Route procedure to route the Re-Register requests after the UA has initially registered.

Option	Description
	<ul style="list-style-type: none"> • route-register-no-service-route=all—Prevents the use of the Service-Route procedure to route the Re-Register requests for all messages, after the UA has initially registered. • route-register-no-service-route=refresh—Prevents the use of the Service-Route procedure to route the Re-Register requests for all refresh-register messages, but not de-register messages, after the UA has initially registered. <p>Addition idle argument ensures that, when enabled, the CSM follows the previously defined rules for idle calls, where idle means not engaged in any INVITE-based sessions.</p> <p>Sample syntax: route-register-no-service-route=refresh;idle</p>
sdp-insert-sendrecv	<p>When a call is initiated, the SDP communicates between call offerer and call answerer to determine a route for the media. Devices can be configured to only send media (“a=sendonly”), to only receive media (“a=recvonly”), or to do both (“a=sendrecv”). Some devices, do not disclose this information. With this option configured, when either the offerer or answerer does not disclose its directional attribute, the CSM automatically inserts a sendrecv direction attribute to the media session.</p>
set-inv-exp-at-100-resp	<p>Set Timer C when a 100 Trying response is received (instead of waiting until 1xx (> 100) is received). If the CSM does not receive a 100 Trying response within Timer B, the call should be dropped because there is a problem communicating with the next hop.</p>
strip-domain-suffix-route	<p>Causes sipd to strip any Router headers from the inbound messages coming to the external address of a SIP NAT; if the message contains a FQDN that matches the configured domain suffix for that SIP NAT.</p>
video-sbc-session	<p>Use with drain-sendonly for conference floor support. When configured with drain-sendonly and when the CSM receives an SDP, the CSM proxies the m=control and its related a= and c= unchanged. Although media streams are allocated for this m line, an actual flow is not set up.</p> <p>SDP received with the following:</p> <pre>m=video a=sendonly</pre> <p>is sent out as the following:</p> <pre>m=video a=sendonly a=X-SBC-Session</pre>
session-timer-support	<p>This option enables the CSM to start the session timer for session refreshes coming from the UAC. The CSM determines whether or not a session is active based on session refreshes or responses. It terminates the session when no session refreshes occur within the session timer interval.</p>
session-timer-support	<p>Enables RFC4028 session timer support.</p>
inmanip-before-validate	<p>Enables SIP Header Pre-processing for HMR.</p>

Option	Description
process-implicit-tel-URI	Correctly appends coodie in REGISTER message when user=phone does not exist.
offerless-bw-media	Reserves appropriate bandwidth for an INVITE with no SDP.

SIP Interface Options

The following table lists the SIP interface options supported by the Oracle CSM.

Option	Description
100rel-interworking	Enables RFC 3262, <i>Reliability of Provisional Responses in the Session Initiation Protocol</i> support.
contact-endpoint=<endpoint name>	The Oracle CSM inserts the endpoint IP address and port into the Contact headers as messages egress using that SIP interface. The inserted data is the same as the information received in the Request or Response being forwarded. If the endpoint name is not specified, the default value endpoint is used.
contact-firewall=<firewall name>	The Oracle CSM inserts the firewall IP address and port into the Contact headers as messages egress using that SIP interface. The inserted data is the same as the information received in the Request or Response being forwarded. If the endpoint name is not specified, the default value firewall is used.
contact-vlan=<VLAN/realm name>	The Oracle CSM inserts the realm and VLAN ID into the Contact headers as messages egress using that SIP interface. The inserted data is the same as the information received in the Request or Response being forwarded. If the endpoint name is not specified, the default value vlan is used.
dropResponse	The Oracle CSM drops responses by specified status codes. The option value can contain one or more status codes separated by semicolons. Error ranges can also be entered. If any of the response codes matches then a response is not sent. If the dropResponse option is set in both the sip-interface and the session-agent elements, the session-agent setting takes precedence.
max-udp-length=<maximum length>	Sets the largest UDP packers that the Oracle CSM will pass. Packets exceeding this length trigger the establishment of an outgoing TCP session to deliver the packet; this margin is defined in RFC 3261. The system default for the maximum UDP packet length is 1500. You can set the global SIP configuration's max-udp-length=x option for global use in your SIP configuration, or you can override it on a per-interface basis by configuring this option in a SIP interface configuration.
response-for-not-found=<response code>	Change the 404 Not Found generated by the SBC to a different response code.
strip-route-headers	Causes the Oracle CSM to disregard and strip all route headers for requests received on a SIP interface.

SIP Signaling Services

Option	Description
upd-fallback	When a request needs to be sent out on the SIP interface for which you have configured this option, the Oracle CSM first tries to send it over TCP. If the SIP endpoint does not support TCP, however, then the Oracle CSM falls back to UDP and tries the request again.
via-header-transparency	Enables the Oracle CSM to insert its Via header on top of the top-most Via header received from user equipment (UE). It then forwards it on to the IP Multimedia Subsystem (IMS) core with the original Via header now located as the bottom-most Via header. The Oracle CSM still replaces the Contact and other header addresses with its own, and does not pass on the core's Via headers in outbound requests.
use-redirect-route	Use Route parameter in Contact header as next-hop as received in a 3xx response.
reg-via-proxy	Enables your Oracle CSM to support endpoints that register using an intervening proxy.
lmsd-interworking	Enables 3GPP2 LMSD Interworking.
suppress-reinvite	Enables reINVITE suppression.

SIP Session Agent Options

The following table lists the SIP session agent options supported by the Oracle CSM.

Option	Description
dropResponse	The Oracle CSM drops responses by specified status codes. The option value can contain one or more status codes separated by semicolons. Error ranges can also be entered. If any of the response codes matches then a response is not sent. If the dropResponse option is set in both the sip-interface and the session-agent elements, the session-agent setting takes precedence.
trans-timeouts=<value>	Defines the number of consecutive non-ping transaction timeouts that will cause a session agent to be out of service. When the session agent is configured, i.e. when the PING options are defined, the value is 10. If not defined, the default value is 5. A Value of 0 prevents the session agent from going out of service because of a non-ping transaction timeout.
via-origin=<parameter-name>	Causes a parameter to be included in the top Via header of requests sent to the session agent. The parameter indicates the source IP address of the corresponding request received by the Oracle CSM. <parameter-name> defines the name of the parameter. If not specified, the default value origin is used.
refer-reinvite	Enables SIP REFER with Replaces.

SIP Realm Options

The following table lists the SIP session agent options supported by the Oracle CSM.

Option	Description
number-normalization	Applies to the SIP To URI. (Currently the Oracle CSM supports number normalization on From and To addresses for both inbound and outbound

Option	Description
	call legs.) Number normalization includes add, delete, and replace string functions that result in consistent number formats. Number normalization occurs on ingress traffic, prior to the generation of accounting records or local policy lookups.
refer-reinvite	Enables SIP REFER with Replaces.

SIP Realm Options Configuration

To configure options:

Labels enclosed in <> indicate that a value for the option is to be substituted for the label. For example, <value>. In order to change a portion of an options field entry, you must re-type the entire field entry.

1. Navigate to the options parameter in the SIP configuration or SIP interface elements.
2. Enter the following:

```
options Space <option name>="<value>"
```

For example, if you want to configure the refer-to-uri-prefix option (the add-on conferencing feature):

Type options, followed by a Space.

Type refer-to-uri-prefix, followed by an equal sign (=).

Type the opening quotation mark (") followed by conf, another equal sign and the closing quotation mark.

Press Enter.

For example:

```
options refer-to-uri-prefix=conf=
```

If the feature value itself is a comma-separated list, it must be enclosed in quotation marks.

Configuring Multiple Options

You can enter a list of options for this field:

1. Type options followed by a space.
2. Within quotation marks, enter the feature names and values of the parameters you need. Separate each one with a comma.
3. Close the quotation marks.
4. Press Enter.

For example:

```
ACMEPACKET(sip-config)# options "refer-to-uri-prefix="conf=", encode-  
contact="+SD", add-ruri-user=INVITE, ACK"
```

Adding an Entry

Enter the new entry with a preceding plus (+) sign. For example:

```
options +response-for-not-found
```

This format allows previously configured options field values to remain intact without requiring re-entry of the entire field value.

SIP Security

This section provides an overview of Oracle CSM's security capability. Oracle CSM security is designed to provide security for VoIP and other multi-media services. It includes access control, DoS attack, and overload protection, which help secure service and protect the network infrastructure (including the Oracle CSM). In addition, Oracle CSM security lets legitimate users to still place calls during attack conditions, protecting the service itself.

Oracle CSM security includes the Net-SAFE framework's numerous features and architecture designs. Net-SAFE is a requirements framework for the components required to provide protection for the Session Border Controller (SBC), the service provider's infrastructure equipment (proxies, gateways, call agents, application servers, and so on), and the service itself.

Denial of Service Protection

The Oracle CSM Denial of Service (DoS) protection functionality protects softswitches and gateways with overload protection, dynamic and static access control, and trusted device classification and separation at Layers 3-5. The Oracle CSM itself is protected from signaling and media overload, but more importantly the feature allows legitimate, trusted devices to continue receiving service even during an attack. DoS protection prevents the Oracle CSM host processor from being overwhelmed by a targeted DoS attack from the following:

- IP packets from an untrusted source as defined by provisioned or dynamic ACLs
- IP packets for unsupported or disabled protocols
- Nonconforming/malformed (garbage) packets to signaling ports
- Volume-based attack (flood) of valid or invalid call requests, signaling messages, and so on.
- Overload of valid or invalid call requests from legitimate, trusted sources

Levels of DoS Protection

The multi-level Oracle CSM Denial of Service protection consists of the following strategies:

- Fast path filtering/access control: involves access control for signaling packets destined for the Oracle CSM host processor as well as media (RTP) packets. The SBC accomplishes media filtering using the existing dynamic pinhole firewall capabilities. Fast path filtering packets destined for the host processor require the configuration and management of a trusted list and a deny list for each Oracle CSM realm (although the actual devices can be dynamically trusted or denied by the Oracle CSM based on configuration). You do not have to provision every endpoint/device on the Oracle CSM, but instead retain the default values.
- Host path protection: includes flow classification, host path policing and unique signaling flow policing. Fast path filtering alone cannot protect the Oracle CSM host processor from being overwhelmed by a malicious attack from a trusted source. The host path and individual signaling flows must be policed to ensure that a volume-based attack will not overwhelm the Oracle CSM's normal call processing; and subsequently not overwhelm systems beyond it. The Oracle CSM must classify each source based on its ability to pass certain criteria that is signaling- and application-dependent. At first each source is considered untrusted with the possibility of being promoted to fully trusted. The Oracle CSM maintains two host paths, one for each class of traffic (trusted and untrusted), with different policing characteristics to ensure that fully trusted traffic always gets precedence.
- Host-based malicious source detection and isolation – dynamic deny list. Malicious sources can be automatically detected in real-time and denied in the fast path to block them from reaching the host processor.

Configuration Overview

NAT table entries are used to filter out undesired IP addresses (deny list). After the packet from an endpoint is accepted through NAT filtering, policing is implemented in the Traffic Manager based on the sender's IP address. NAT table entries are used to distinguish signaling packets coming in from different sources for policing purposes.

You can configure deny rules based on the following:

- ingress realm
- source IP address
- transport protocol (TCP/UDP)

- application protocol (SIP, MGCP)

You can configure guaranteed minimum bandwidth for trusted and untrusted signaling paths.

You can configure signaling path policing parameters for individual source addresses. Policing parameters include:

- peak data rate in bits per second
- average data rate in bits per second
- maximum burst size

SIP Unauthorized Endpoint Call Routing

The Oracle CSM (CSM) can route new dialog-creating SIP INVITEs from unauthorized endpoints to a session agent or session agent group; then rejection can occur based on the allow-anonymous setting for the SIP port. This type of provisional acceptance and subsequent rejection applies only to INVITEs; the CSM continues to reject all other requests, such as SUBSCRIBE.

You might enable this feature if you have a network in which unauthorized SIP endpoints continually try to register even if the Oracle CSM has previously rejected them and never will accept them. For instance, the user account associated with the endpoint might have been removed or core registrars might be overloaded.

SIP Unauthorized Endpoint Call Routing Configuration

You enable the routing of unauthorized endpoints to session agents and session agent groups that will reject them in the SIP interface configuration.

To enable SIP unauthorized endpoint call routing:

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

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

2. Type session-router and press Enter.

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

3. Type sip-interface and press Enter.

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

If you are adding this feature to an existing configuration, then you will need to select the configuration you want to edit.

4. route-unauthorized-calls—Enter the name (or IP address) of the session agent or session agent group to which you want calls from unauthorized endpoints routed. This parameter is blank by default, meaning the SIP unauthorized call routing feature is disabled.

Remember your settings in the allow-anonymous parameter in the SIP port configuration provide the basis for rejection.

5. Save and activate your configuration.

SIP HNT Forced Unregistration

If you use HNT and experience the issue explained in this section, consider using the Oracle CSM (CSM) forced unregistration feature. When this feature is enabled and a registration entry for an endpoint expires, the CSM notifies the soft switch to remove this binding using REGISTER message. In that REGISTER message, the expires header will be set to 0 and the expires parameter in the Contact header will also be set to 0.

The benefits of using forced unregistration include:

- Leveraging existing HNT configuration to provide near real-time information about the UA's status to the registrar/softswitch

SIP Signaling Services

- Preserving resource utilization for the CSM and the softswitch by deleting a contact binding that is no longer valid or needed
- Preventing extra bindings from being generated at the softswitch (e.g., in instances when the UA or NAT restart)

This feature applies to:

- HNT endpoints with registration caching enabled by default, and when the nat-traversal parameter in the SIP interface configuration is set to always
- non-HNT endpoints with registration caching enabled, when the registration-interval parameter in the SIP interface configuration is used in the expires header sent to the UA in the 200 OK

When to Use Forced Unregistration

For typical HNT use, it is common that the registration interval between the client UA and the Oracle CSM (CSM) is between 60 and 120 seconds. This differs significantly from the re-registration interval between the CSM and the registrar, which varies from approximately 30 to 60 minutes.

If the UA fails to refresh its registration, the contact binding at the CSM is deleted after the registration expires. This expiration is determined by the expires= header in the 200 OK. The binding at the real registrar will remain intact. This creates a discrepancy between the real state of the UA and state of the softswitch. In the best case scenario, the contact binding expires at the softswitch after a few minutes.

For network management, this discrepancy can be problematic because the service provider would be unaware of the UA's status until the binding expires at the softswitch. This can take a considerable amount of time to happen.

In addition, the CSM encodes a cookie in the userinfo of the Contact header in the REGISTER message. This is a function of the source IPv4 address and port from which the request came, i.e., the ephemeral port in the NAT for DSL scenarios. Therefore, additional bindings that remain for long periods of time are created at the registrar if, for example, the:

- UA reboots
- Ethernet link between the UA and the DSL router is lost for over two minutes
- DSL crashes
- DSL/ATM layer between the DSL router

Caution for Using Forced Unregistration

You should use caution when applying SIP HNT forced unregistration for the following reasons:

- It can have an impact on the performance of your Oracle CSM and the registrar, especially when you have a large number of HNT endpoints in your configuration that become unavailable simultaneously.
- It is possible that the registrar might become vulnerable to overload in the case where the registrar must authenticate a large number of register messages generated when HNT endpoints are de-registered. It is possible that the cached registration credentials might become "stale" over time (e.g., the nonce value usually has a limited lifetime). Without proper credentials, the registrar will reject the de-registrations.

Given these concerns, we recommend that you consult with your Oracle systems engineer before adopting the use of forced unregistration.

SIP HNT Forced Unregistration Configuration

To enable SIP HNT forced unregistration:

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
```

4. Use the CLI select command so that you can work with the SIP configuration.

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

5. options—Set the options parameter by typing options, a Space, the option name force-unregistration, and then press Enter.

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

If you type options force-unregistration, you will overwrite any previously configured options. In order to append the new option to the sip-config's options list, you must prepend the new option with a plus sign as shown in the previous example.

SIP IP Address Hiding and NATing in XML

Adding to its topology hiding and NAT capabilities, the Oracle CSM now performs those functions for pertinent IP addresses that are not part of the standard SIP message header format. Previously, such addresses were visible to the next hop in the SIP session path.


Note that this feature adds to the Oracle CSM's pre-existing ability to perform this function for XML messages; this new support is specifically for the keyset-info message type.

For incoming SIP NOTIFY messages, the Oracle CSM searches for the application/keyset-info+xml content type in the message. When it finds this content type, it searches further to detect the presence of <di:remote-uri> or <di:local-uri> XML tags and then NATs the IP addresses in the tags it finds. Specifically, the Oracle CSM changes:

- The <di:remote-uri> IP address to be the egress SIP interface's IP address
- The <di:local-uri> IP address to be the IP address of the next hop to which the message is being sent

Sample SIP NOTIFY with NATed XML

The following is a sample SIP NOTIFY message as it might arrive at the Oracle CSM.

 **Note:** that it contains the <di:remote-uri> or <di:local-uri> XML tags on which the system will perform NAT; these lines appear in bold text.

```
NOTIFY sip:15615281021@10.152.128.253:5137;transport=udp SIP/2.0
To: 15615281021 <sip:15615281021@10.152.128.102:5080>;tag=5c93d019904036a
From: <sip:15615281021@10.152.128.102:5080>;tag=test_tag_0008347766
Call-ID: 3215a76a979d0c6
CSeq: 18 NOTIFY
Contact: <sip:15615281021@10.152.128.102:5080;maddr=10.152.128.102>
Via: SIP/2.0/UDP 10.152.128.102:5060;branch=z9hG4bK_brancha_0023415201
Event: keyset-info
Subscription-state: active;expires=2778
Accept: application/keyset-info+xml
Content-Type: application/keyset-info+xml
Content-Length: 599
Max-Forwards: 70
<?xml version="1.0"?>
<keyset-info xmlns="urn:ietf:params:xml:ns:keyset-info"
  version="16"
  entity="15615281021">
  <ki-data>
    <ki-state>"active"</ki-state>
    <ki-event>"unknown"</ki-event>
  </ki-data>
  <di:dialog id="dialog_id_201" call-
id="1395216611-1987932283256611-11-0884970552" local-
tag="test_tag_0008347790" direction="recipient">
    <di:state>trying</di:state>
    <di:duration>2778</di:duration>
```

SIP Signaling Services

```
<di:local-uri>sip:15615281021@10.152.128.253:5137</di:local-uri>
<di:remote-uri>sip:1004@10.152.128.102</di:remote-uri>
</di:dialog>
</keyset-info>
```

Once the Oracle CSM has completed the NAT process, the `<di:remote-uri>` and `<di:local-uri>` XML tags look like this

```
<di:local-uri>sip:15615281021@192.168.200.99:5137</di:local-uri>
<di:remote-uri>sip:1004@192.168.200.49</di:remote-uri>
```

because egress the SIP interface's IP address is 192.168.200.49 and the next hop's IP address is 192.168.200.99.

This feature does not require any configuration.

SIP Server Redundancy

This section explains how to configure SIP server redundancy. SIP server redundancy involves detecting that an upstream/downstream SIP signaling entity has failed, and adapting route policies dynamically to remove it as a potential destination.

Overview

You establish SIP server redundancy by creating session agents, which are virtual representations of the SIP signaling entities. These agents are then collected into a session agent group, which is a logical collection of two or more session agents that behaves as a single aggregate entity.

Rather than direct signaling messages to a single session agent (IP), the signaling message is directed to a session agent group (SAG). The group will have a set distribution pattern: hunt, round robin, proportionally distributed, and so on. Signaling is spread amongst the agents using this chosen pattern.

You direct the signaling message by configuring a route policy, known as a local policy, which determines where SIP REQUESTS should be routed and/or forwarded. The values in the To and From headers in the SIP REQUEST are matched with the content of the local policy within the constraints set by the session agent's previous hop value and SIP interface values such as the list of carriers.

To summarize, you need:

- two or more session agents
- a session group containing those session agents
- a local policy which directs traffic to the session agent group

Configuration Overview

You make a session agent group a target by using a local policy to select the next hop from the members of a session agent group. You need to set the replace URI field of the configured local policy to enabled; which causes NAT rules such as realm prefixing to be overridden. The replace URI field allows you to indicate whether the local policy's value is used to replace the Request-URI in outgoing requests. This boolean field can be set to either enabled or disabled.

When the SIP NAT's route home proxy field is set to forced, it forces the Request to be forwarded to the home proxy without using a local policy. When this option is set to either disabled or enabled and the Request-URI matches the external address of the SIP NAT, the local policy is used.


However, the local policy only replaces the Request-URI when the original Request-URI matches the SBC's IP address or hostname. This behavior is in accordance with that described in RFC 3261. The original Request-URI will be the home proxy address value (the home address of the SIP NAT into the backbone) and not the Oracle CSM (CSM) address.

Using strict routing, the Request-URI would be the next hop, but the message would also include a Route header with the original Request-URI. With loose routing, the Request-URI remains unchanged and the next hop value is added as the top Route header.

Sometimes the next hop field value must replace the Request-URI in the outgoing request, even if the original Request-URI is not the CSMC. To accomplish this, an option has been added to the local policy that causes the next hop value to be used as the Request-URI and prevents the addition of Route headers. This option is the replace uri value in the local policy.

The following table lists the policy attributes for the local policy:

Parameter	Description
next hop	IP address of your internal SIP proxy. This value corresponds to the IP address of the network interface associated with the SIP proxy.
realm	Number of the port associated with the SIP port.
replace uri	Stores the transport protocol used for sending an receiving signaling messages associated with the SIP port.
allow anonymous	Indicates whether this SIP port allows anonymous connections from session agents.

 **Note:** You should also define the ping method intervals for the session agents so that the CSM can detect when the agents are back in service after failure.

SIP Server Redundancy Configuration

To enable replace URI:

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

```
ORACLE# configure terminal
```

2. Type local-policy and press Enter to access the system-level configuration elements. The system prompt changes.

```
ORACLE (configure) # local-policy
ORACLE (local-policy) #
```

3. Type policy-attributes and press Enter. The system prompt changes.

```
ORACLE (local-policy) # policy-attributes
ORACLE (local-policy-attributes) #
```

From this point, you can configure policy attributes for the local policy. To see all local policy attribute options, enter a ? at the system prompt.

4. action—Set this parameter to replace-uri, which causes NAT rules such as realm prefixing to be overridden. The default value is none. Valid values are:

- none | replace-uri | redirect

The replace URI field allows you to indicate whether the local policy's value is used to replace the Request-URI in outgoing requests. This boolean field can be set to either enabled or disabled.

Embedded Header Support

This section explains how to configure embedded header support. The Oracle CSM supports methods of extracting an embedded P-Asserted-Identity header from a contact header to support E911 when integrated with certain vendor's systems. See RFC 3455 *Private Header (P-Header) Extensions to the Session Initiation Protocol (SIP) for the 3rd-Generation Partnership Project (3GPP)* for more information.

The embedded header support feature watches for a specified embedded header contained in a Contact header received in a 3XX message. When the specified embedded header is found, the full <header=value> pair is inserted as a unique header in a redirected INVITE message that exits the Oracle CSM. If the outgoing INVITE message were to contain the specified header, regardless of the use of this feature, the value extracted from the 3XX message replaces the INVITE message's specified header value.

SIP Signaling Services

If an incoming Contact header in a 3XX message looks like:

```
Contact: <ESRN@IPv4_Intrado_GW;user=phone?P-Asserted-Identity=%3Csip:+1-ESQK@IPv4_My_EAG;user=phone%3E>
```

Then, if you configure your Oracle CSM to parse for the embedded P-Asserted-Identity header to write as a unique header in the outgoing invite message, the outgoing INVITE and P-Asserted-Identity headers will look like:

```
INVITE SIP: ESRN@IPv4_Intrado_GW;user=phone  
P-Asserted-Identity: +1-ESQK@IPv4_My_EAG;user=phone
```

Embedded Header Support Configuration

Embedded header support is enabled in the session agent configuration.

To configure embedded header support:

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 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. Select the session agent where you want this feature.

```
ORACLE(session-agent)# select  
<hostname>:  
1: asd          realm=          ip=1.0.0.0  
2: SIPSA       realm=          ip=10.10.102.1  
selection:2  
ORACLE(session-agent)#
```

5. request-uri-headers—Enter a list of embedded headers extracted from the Contact header that will be inserted in the re INVITE message. To configure this parameter for multiple headers, enclose the headers in double quotes and separate them with spaces. This completes the configuration of embedded header support.

```
ORACLE(session-agent)# request-uri-headers P-Asserted-Identity
```

Static SIP Header and Parameter Manipulation

This section explains the SIP header and parameter manipulation feature, which lets the Oracle CSM add, modify, and delete SIP headers and parts of SIP headers called SIP header elements. SIP header elements are the different subparts of the header, such as the header value, header parameter, URI parameter and so on (excluding the header name).

To enable the SIP header and parameter manipulation functionality, you create header manipulation rulesets in which you specify header manipulation rules, as well as optional header element rules that operate on specified header elements. You then apply the header manipulation ruleset as inbound or outbound for a session agent or SIP interface.

Header Manipulation Rules

Header manipulation rules operate on the header you specify when you configure the rule. A header manipulation rule can also be configured with a list of element rules, each of which would specify the actions you want performed for a given element of this header.

Header Element Rules

Header element rules perform operations on the elements of a header. Header elements include all subparts of a header, excluding the header name. For example, header value, header parameter, URI parameter, and so on.

About SIP Header and Parameter Manipulation

Using the SIP header manipulation ruleset, you can cause the Oracle CSM to:

- Delete a header based on header name match.
- Delete a header based on header name match as well as header value match.
- Add a header.
- Modify the elements of a header (by configuring header element rules):

Add an element to a header.

For example, add a parameter to a header or add a URI parameter to the URI in a header.

Delete an element from a header.

For example, delete a parameter from a header or delete a URI parameter from the URI in a header.

Modify an element of a header.

For example, replace a FQDN with an IPv4 address in a header or replace the value of a parameter in the header.

Delete a message body part

For example, delete the body part if the Content-Type is application/ISUP.

HMR \$LOCAL_PORT for Port Mapping

When you configure SIP HMR and set an element-rule's new-value parameter to \$LOCAL_PORT, the Oracle CSM maps this value to the real port it uses for each signaling exchange.

Role in Trunk Group URI Feature

SIP header and parameter manipulation plays a role in the trunk group URI feature. You need to set the new-value parameter to one of the trunk group values when configuring SIP header rules, if using this feature. (In addition you can configure session agents and session agents groups on the Oracle CSM to insert trunk group URI parameters in the SIP contact header.

For all trunk group URI support, you must set the appropriate parameters in the SIP header manipulation configuration and in the session agent or session agent group configurations.

For trunk group URI support, the SIP header and parameter manipulation configuration tells the Oracle CSM where and how to manipulate the SIP message to use originating (access) and terminating (egress) trunk group URI parameters.

SIP Header and Parameter Manipulation Configuration

This section explains how to configure SIP header and parameter manipulation. First you create a SIP header manipulation ruleset, then the header manipulation rules and optional header element rules you want that ruleset to contain. You then configure a session agent or a SIP interface to use the SIP header and parameter manipulation ruleset in the inbound and outbound directions.

Creating SIP Header Manipulation Rulesets

To configure the SIP header manipulation ruleset:

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-manipulation and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

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


4. name—Enter the name you want to use for this ruleset.
5. header-rules—Define the header manipulation rules you want to include in this ruleset.

Type header-rules and press Enter.

```
ORACLE(sip-manipulation)# header-rules
ORACLE(sip-header-rules)#
```

name—Enter the name of the header to which this rule applies. (The name you enter here must match a header name.)

This is a case-insensitive string that is compared to the header name for matching. You need to create a rule using the long form of the header name and a rule using the compact form of the header name.

 **Note:** The Request-URI header is identified as request-uri.

action—Enter the action you want applied to the header specified in the name parameter. The default value is none. Valid options are:

- add—Add a new header, if that header does not already exist.
- delete—Delete the header, if it exists.
- manipulate—Elements of this header will be manipulated according to the element rules configured.
- store—Store the header.
- none—No action to be taken.

match-value—Enter the value to be matched (only an exact match is supported) with a header value. The action you specify is only performed if the header value matches.

msg-type—Enter the message type to which this header rule applies. The default value is any. Valid options are:

- any—Both Requests and Reply messages
- request—Request messages only
- reply—Reply messages only


Type show to display the header rule configuration values.

6. element-rules—Define the element rules you want to use to be performed on the elements of the header specified by the header rule.

Type element-rules and press Enter.

```
ORACLE(sip-header-rules)# element-rules
ORACLE(sip-element-rules)#
```

name—Enter the name of the element to which this rule applies.

 **Note:** The name parameter usage depends on the element type you enter in step 6. For uri-param, uri-user-param, and header-param it is the parameter name to be added, replaced, or deleted. For all other types, it serves to identify the element rule and any name can be used.

type—Enter the type of element on which to perform the action. The default value is none. Valid options are:

- header-value—Enter value of the header.
- header-param-name—Header parameter name.
- header-param—Parameter portion of the header.
- uri-display—Display of the SIP URI.
- uri-user—User portion of the SIP URI.
- uri-host—Host portion of the SIP URI.
- uri-port—Port number portion of the SIP URI.
- uri-param-name—Name of the SIP URI param.

- uri-param—Parameter included in the SIP URI.
- uri-header-name—SIP URI header name
- uri-header—Header included in a request constructed from the URI.
- uri-user-param—User parameter of the SIP URI.

action—Enter the action you want applied to the element specified in the name parameter, if there is a match value. The default value is none. Valid options are:

- none—No action is taken.
- add—Add a new element, if it does not already exist.
- replace—Replace the elements.
- delete-element—Delete the specified element if it exists. Based on the match value if entered in step 6f.
- delete-header—Delete the specified header, if it exists.
- store—Store the elements.

match-val-type—Enter the type of value that needs to be matched to the match-field entry for the action to be performed. The default value is ANY. Valid options are:

- IP—Element value in the SIP message must be a valid IP address to be compared to the match-value field entry. If the match-value field is empty, any valid IP address is considered a match. If the element value is not a valid IP address, it is not considered a match.
- FQDN—Element value in the SIP message must be a valid FQDN to be compared to the match-value field entry. If the match-value field is empty, any valid FQDN is considered a match. If the element value is not a valid FQDN, it is not considered a match.
- ANY—Element value in the SIP message is compared with the match-value field entry. If the match-value field is empty, all values are considered a match.

match-value—Enter the value you want to match against the element value for an action to be performed.

new-value—Enter the value for a new element or to replace a value for an existing element. You can enter an expression that includes a combination of absolute values, pre-defined parameters, and operators.

- Absolute values, with which you can use double quotes for clarity. You must escape all double quotes and back slashes that are part of an absolute value, and enclose the absolute value in double quotes.

For example:

sip:”+\$STRUNK_GROUP+”.\$STRUNK_GROUP_CONTEXT

- Pre-defined parameters always start with a \$. Valid pre-defined parameters are:

Parameter	Description
\$ORIGINAL	Original value of the element is used.
\$LOCAL_IP	IP address of the SIP interface on which the message was received for inbound manipulation; or sent on for outbound manipulation.
\$REMOTE_IP	IP address the message was received from for inbound manipulation; or being sent to for outbound manipulation.
\$REMOTE_VIA_HOST	Host from the top Via header of the message is used.
\$STRUNK_GROUP	Trunk group is used.
\$STRUNK_GROUP_CONTEXT	Trunk group context is used.

- Operators are:

Operator	Description
+	Append the value to the end. For example: acme”+”packet generates acmepacket

Operator	Description
+^	Prepends the value. For example: acme”+^”packet generates packetacme
-	Subtract at the end. For example: 112311”-”11 generates 1123
_^	Subtract at the beginning. For example: 112311”-^”11 generates 2311

Examples of entries for the new-value field.

```
$ORIGINAL+acme
$ORIGINAL+"my name is john"
$ORIGINAL+"my name is \"john\""
$ORIGINAL-^781+^617
```

Type show to display the element rule configuration values.

Type done to save them.

Repeat steps 6b through 6j to create additional rules.

Type exit to return to the header-rules parameters.

7. methods—Enter the SIP method names to which you want to apply this header rule. If entering multiple method names, separate them with commas. For example:

```
INVITE, ACK, BYE
```

This field is empty by default. If you leave the method field empty, the header-rule is applied to all methods.

8. Type exit to return to the sip-manipulation level.
9. Save your work using the ACLI done command.
10. If you want to save this configuration, exit out of configuration mode and type save-config.

Configuring a Session Agent

You can configure a session agent to use the SIP header manipulation ruleset.

To configure a session agent:

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. in-manipulationid—Enter the name of the SIP header manipulation ruleset you want to apply to inbound SIP packets.

```
ORACLE(session-agent)# in-manipulationid route-stripper
```


5. out-manipulationid—Enter the name of the SIP header manipulation ruleset you want to apply to outbound SIP packets.

```
ORACLE(session-agent)# out-manipulationid route-stripper
```

6. Save your work using the ACLI done command.
7. If you want to save this configuration, exit out of configuration mode and type save-config.

Configuring a SIP Interface

You can configure a interface to use a SIP header manipulation ruleset.

To configure a 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. in-manipulationid—Enter the name of the SIP header manipulation ruleset you want to apply to SIP packets in the ingress direction.

```
ORACLE(sip-interface)# in-manipulationid
```

5. out-manipulationid—Enter the name of the SIP header manipulation ruleset you want to apply to SIP packets in the egress direction.

```
ORACLE(sip-interface)# out-manipulationid
```

6. Save your work using the ACLI done command.
7. If you want to save this configuration, exit out of configuration mode and type save-config.

Example 1 Stripping All Route Headers

This example explains how to strip all route headers from a SIP packet. First, you create a header manipulation ruleset, in the example it is called route-stripper. Then you configure the list of header manipulation rules you need to strip route headers. In this case, you only need one rule named Route (to match the Route header name) with the action set to Delete.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-manipulation
ORACLE(sip-manipulation)# name route-stripper
ORACLE(sip-manipulation)# header-rules
ORACLE(sip-header-rules)# name Route
ORACLE(sip-header-rules)# action Delete
ORACLE(sip-header-rules)# done
header-rule
  name          Route
  action        delete
  match-value
  msg-type      any
ORACLE(sip-header-rules)# ex
ORACLE(sip-manipulation)# done
sip-manipulation
  name          route-stripper
  header-rule
    name        Route
    action      delete
```

match-value	
msg-type	any

Example 2 Stripping an Existing Parameter and Adding a New One

This example explains how to strip the user parameter from the Contact header URI and add the acme parameter with value as LOCAL IP, only for requests. First you create a header manipulation ruleset, in the example it is called param-stripper1. You then configure a list of header rules you need. In this case, you only need one rule named Contact (to match the Contact header name), with action set to manipulate (indicating the elements of this header would be manipulated). Next, you configure a list of element rules for the Contact header rule.

In this case you configure two element rules; one to strip the uri parameter user (the rule name user matches the param name user) and the other to add the uri parameter acme (the rule name acme matches the param name acme).

```
ORACLE# configure terminal
ORACLE (configure)# session-router
ORACLE (session-router)# sip-manipulation
ORACLE (sip-manipulation)# name param-stripper1
ORACLE (sip-manipulation)# header-rules
ORACLE (sip-header-rules)# name Contact
ORACLE (sip-header-rules)# action manipulate
ORACLE (sip-header-rules)# msg-type request
ORACLE (sip-header-rules)# element-rules
ORACLE (sip-element-rules)# name user
ORACLE (sip-element-rules)# type uri-param
ORACLE (sip-element-rules)# action delete-element
ORACLE (sip-element-rules)# done
element-rule
  name          user
  type          uri-param
  action        delete-element
  match-val-type any
  match-value
  new-value
ORACLE (sip-element-rules)# name acme
ORACLE (sip-element-rules)# action add
ORACLE (sip-element-rules)# type uri-param
ORACLE (sip-element-rules)# new-value "$LOCAL_IP"
ORACLE (sip-element-rules)# done
element-rule
  name          acme
  type          uri-param
  action        add
  match-val-type any
  match-value
  new-value    "$LOCAL_IP"
ORACLE (sip-element-rules)# ex
ORACLE (sip-header-rules)# done
header-rule
  name          Contact
  action        manipulate
  match-value
  msg-type     request
  element-rule
    name          user
    type          uri-param
    action        delete-element
    match-val-type any
    match-value
    new-value
  element-rule
    name          acme
    type          uri-param
    action        add
```

```

        match-val-type      any
        match-value
        new-value           "$LOCAL_IP"
ORACLE (sip-header-rules) # ex
ORACLE (sip-manipulation) # done
sip-manipulation
  name                      param-stripper1
  header-rule
    name                    Contact
    action                  manipulate
    match-value
    msg-type                request
    element-rule
      name                  user
      type                  uri-param
      action                delete-element
      match-val-type       any
      match-value
      new-value
element-rule
  name                      acme
  type                      uri-param
  action                    add
  match-val-type           any
  match-value
  new-value                 "$LOCAL_IP"

```

For example, if the IP address of the SIP interface (\$LOCAL_IP) is 10.1.2.3 and the Oracle CSM receives the following Contact header:

```
Contact: <sip:1234@10.4.5.6;user=phone>
```

The header rule is applied to strip the user parameter from the Contact header URI and add the acme parameter with the value 10.1.2.3:

```
Contact: <sip:1234@10.4.5.6;acme=10.1.2.3>
```

SIP HMR (Header Manipulation Rules)

SIP header manipulation can also be configured in a way that makes it possible to manipulate the headers in SIP messages both statically and dynamically. Using this feature, you can edit response headers or the Request-URI in a request, and change the status code or reason phrase in SIP responses.

Static SIP Header and Parameter Manipulation allows you to set up rules in your Oracle CSM configuration that remove and/or replace designated portions of specified SIP headers. SIP HMR allows you to set up dynamic header manipulation rules, meaning that the Oracle CSM has complete control over alterations to the header value. More specifically:

- The Oracle CSM can search header for dynamic content or patterns with the header value. It can search, for example, for all User parts of a URI that begin with 617 and end with 5555 (e.g., 617...5555).
- The Oracle CSM can manipulate any part of a patterns match with any part of a SIP header. For example, 617 123 5555 can become 617 231 5555 or 508 123 0000, or any combination of those.

To provide dynamic header manipulation, the Oracle CSM uses regular expressions to provide a high degree of flexibility for this feature. This allows you to search a specific URI when you do not know that value of the parameter, but want to use the matched parameter value as the header value. It also allows you to preserve matched sections of a pattern, and change what you want to change.

You can apply header manipulation to session agents, SIP interfaces, and realms. You do so by first setting up header manipulations rules, and then applying them in the configurations where they are needed. Within the header manipulation rules, there are sets of element rules that designate the actions that need to be performed on a given header.

Each header rule and each element rule (HMR) have a set of parameters that you configure to identify the header parts to be manipulated, and in what way the Oracle CSM is to manipulate them. These parameters are explained in detail, but the parameter that can take regular expression values is match-value. This is where you set groupings that you want to store, match against, and manipulate.

Generally, you set a header rule that will store what you want to match, and then you create subsequent rules that operate on this stored value. Because header rules and element rules are applied sequentially, it is key to note that a given rule performs its operations on the results of all the rules that you have entered before it. For example, if you want to delete a portion of a SIP header, you would create Rule 1 that stores the value for the purpose of matching, and then create Rule 2 that would delete the portion of the header you want removed. This prevents removing data that might be used in the other header rules.

Given that you are using regular expression in this type of configuration, this tightly sequential application of rules means that you must be aware of the results to be yielded from the application of the regular expressions you enter. When you set a regular expression match value for the first rule that you enter, the Oracle CSM takes the results of that match, and then a second rule might exist that tells the Oracle CSM to use a new value if it the second rule's match value finds a hit (and only 10 matches, 0-9, are permitted) for the results (yield) from applying the first rule.

Consider the example of the following regular expression entry made for a match-value parameter: 'Trunk(+)', which might be set as that match value in the first rule you configure. Given a SIP element rule called uri-param and the param-name tgid, it can yield two values:

- Grouping 0—The entire matching string (Trunk1)
- Grouping 1—The grouping (1)

In turn, these groupings can be referenced in an element rule by using this syntax:

```
$<header rule name >.$<element rule name.$<value>
```

Additional syntax options that can be used with this feature are:

- \$headerName[['index']]
- \$headerName[['index']]\$.index]
- \$headerName[['index']]\$.elementName]
- \$headerName[['index']]\$.elementName]\$.index]

Guidelines for Header and Element Rules

Header rules and element rules share these guidelines:

- References to groupings that do not exist result in an empty string.
- References to element rule names alone result in a Boolean condition of whether the expression matched or not.
- A maximum of ten matches are allowed for a regular expression. Match 0 (grouping 0) is always the match of the entire matching string; subsequent numbers are the results for other groups that match.

Precedence

The Oracle CSM applies SIP header rules in the order you have entered them. This guards against the Oracle CSM removing data that might be used in the other header rules.

This ordering also provides you with ways to use manipulations strategically. For example, you might want to use two rules if you want to store the values of a regular expression. The first rule would store the value of a matched regular expression, and the second could delete the matched value.

In addition to taking note of the order in which header rules are configured, you now must also configure a given header rule prior to referencing it. For example, you must create Rule1 with the action store for the Contact header BEFORE you can create Rule2 which uses the stored value from the Contact header.

Duplicate Header Names

If more than one header exists for the header name you have configured, the Oracle CSM stores the value where it can be referenced with the optional syntax `$<header rule name>[index]`. Additional stored header values are indexed in the order in which they appear within the SIP message, and there is no limit to the index.

Possible index values are:

- `~` — The Oracle CSM references the first matching header
- `*` — The Oracle CSM references all headers
- `^` — The Oracle CSM references the last stored header in the header rule

Performing HMR on a Specific Header

HMR has been enhanced so that you can now operate on a specific instance of a given header. The syntax you use to accomplish this is similar to that you used to refer to a specific header rule stored value instance.

Using the header-name parameter, you can now add a trailing [`<index>`] value after the header name. This [`<index>`] is a numerical value representing the specific instance of the header on which to operate. However, the Oracle CSM takes no action if the header does not exist. You can also use the caret (`^`) to reference the last header of that type (if there are multiple instances)

The count for referencing is zero-based, meaning that the first instance of the header counts as 0.

Note that the header instance functionality has no impact on HMR's add action, and you cannot use this feature to insert headers into a specific location. Headers are added to the end of the list, except that Via headers are added to the top.

Multiple SIP HMR Sets

In general you use SIP HMR by configuring rules and then applying those rules to session agents, realms, or SIP interfaces in the inbound or outbound direction. In addition, the Oracle CSM has a set method for how certain manipulation rules take precedence over others. For instance, inbound SIP manipulation rules defined in a session agent take precedence over any configured for a realm, and the rules for a realm take precedence over SIP interface manipulation rules.

The multiple SIP HMR feature gives you the ability to:

- Apply multiple inbound and outbound manipulations rules to a SIP message
- Provision the order in which the Oracle CSM applies manipulation rules

The action parameter in the header rules configuration now takes the value `sip-manip`. When you set the parameter to `sip-manip`, you then configure the `new-value` parameter with the name of a SIP manipulation rule that you want to invoke. The values for the `match-value`, `comparison-type`, and `methods` parameters for invoked rule are all supported. This means that the manipulation defined by the rules identified in the `new-value` parameter are carried out when the values for the `match-value`, `comparison-type`, and `methods` parameters are true.

The relationship between manipulation rules and manipulation rule sets is created once you load your configuration, meaning that the order in which you enter them does not matter. It also means that the Oracle CSM cannot dynamically perform validation as you enter rules, so you should use the ACLI `verify-config` command to confirm your manipulation rules contain neither invalid nor circular references. Invalid references are those that point to SIP manipulation rules that do not exist, and circular references are those that create endless loops of manipulation rules being carried out over and over. If you load a configuration exhibiting either of these issues, the Oracle CSM forces the action value for the rule to `none` and the rule will not be used.

MIME Support

Using the SIP HMR feature set, you can manipulate MIME types in SIP message bodies. While you can manipulate the body of SIP messages or a specific content type using other iterations of SIP HMR, this version gives you the power to change the MIME attachment of a specific type within the body by using regular expressions. To achieve this, you use the `find-replace-all` action type, which enables the search for a particular string and the replacement of

all matches for that type. Although you use find-replace-all to manipulate MIME attachments, it can also be used to achieve other goals in SIP HMR.

Note that using find-replace-all might consume more system resources than other HMR types. Therefore this powerful action type should only be used when another type cannot perform the type of manipulation you require.

Find and Replace All

To manipulate a particular portion of the MIME attachment, for example when removing a certain attribute within the content type of application/sdp, the Oracle CSM (CSM) would need to search the content multiple times because:

- SDP can have more than one media line
- The SIP message body can contain more than one application/sdp.

The find-replace-all action type works for SIP header rules and for element rules. You can use it for all manipulation types from the entire header value, to the URI specific parameters, to MIME attachment.

For this action type, it does not matter what you configure the comparison type, which is atypical for actions types, as the comparison type is vital to the others. Find-replace-all, however, binds the comparison type to the pattern rule. Thus, the CSM treats the match value as a regular expression, and it ignores any configured comparison type value in favor of the pattern rule. This type of action is both a comparison and action: For each regular expression match within the supplied string, the CSM substitutes the new value for that match. Yet if you want to replace a certain portion of the regular expression and not the entire matched expression, you need to use a subgroup of expressions and the right syntax to indicate the sub-group replacement index.

You can indicate the sub-group replacement syntax by adding the string `[[n:]]` to the end of the regular expression—where `n` is a number between 0 and 9. For example, given the following settings:

- `action=find-replace-all`
- `match-value=sip:(user)@host[[1:]]`
- `new-value=bob`

you create a new rule to replace only the user portion of the URI that searches for the regular expression and replaces all instances of the user subgroup with the value bob.

Taking advantage of the find-replace-all's recursive nature, you can replace all the 0 digits in a telephone number with 1:

- `action=find-replace-all`
- `match-value=0`
- `new-value=1`

So for the user portion of a URI—or for any other string—with a value 1-781-308-4400 would be replaced as 1-781-318-4411.

If you leave the new-value parameter blank for find-replace-all, the CSM replaces the matched sub-group with an empty string—an equivalent of deleting the sub-group match. You can also replace empty sub-groups, which is like inserting a value within the second sub-group match. For example, `user()@host.com[[1:]]` with a configured new-value `_bob` yields `user_bob@host.com`.

When you use find-replace-all, you cannot use the following parameter-type values: `uri-param-name`, `uri-header-name`, and `header-param-name`. These values are unusable because the CSM only uses case-sensitive matches for the match-value to find the parameter name within the URI. Since it can only be found by exact match, the CSM does not support finding and replacing that parameter.

Escaped Characters

SIP HMR's support for escaped characters allows for searches for values you would be unable to enter yourself. Because they are necessary to MIME manipulation, support for escaped characters now includes:

- `\f`
- `\n`

- \r
- \t
- \v

New Reserved Word

To allow you to search for carriage returns and new lines, the SIP HMR MIME feature also adds support for the reserved word \$CRLF. Because you can search for these values and replace them, you also must be able to add them back in when necessary. Configuring \$CRLF in the new-value parameter always resolves to /r/n, which you normally cannot otherwise enter through the ACLI.

About the MIME Value Type

Introduced to modify the MIME attachment, SIP HMR supports a mime value for the type parameter in the element rules configuration. Like the status-code and reason-phrase values, you can only use the mime type value against a specific header—which in this case, is Content-Type.

When you set the element rule type to mime, you must also configure the parameter-name with a value. This step is a requirement because it sets the content-type the Oracle CSM manipulates in a specific part of the MIME attachment. You cannot leave this parameter blank; the Oracle CSM does not let you save the configuration if you do. When you use the store action on a multi-part MIME attachment that has different attachment types, the Oracle CSM stores the final instance of the content-type because it does not support storing multiple instances of element rule stored values.

In the event you do not know the specific content-type where the Oracle CSM will find the match-value, you can wildcard the parameter-name by setting with the asterisk (*) as a value. You cannot, however, set partial content-types (i.e., application/*). So configured, the Oracle CSM loops through the MIME attachment's content types.

You can set the additional action types listed in this table with the described result:

Action Type	Description
delete-element	Removes the matched mime-type from the body. If this is the last mime-type within in message body, the Oracle CSM removes the Content-Type header.
delete-header	Removes all body content and removes the Content-Type header.
replace	Performs a complete replacement of the matched mime-type with the new-value you configure.
find-replace-all	Searches the specifies mime-type's contents and replaces all matching regular expressions with the new-value you configure
store	Stores the final instance of the content-type (if there are multi-part MIME attachments of various attachment types)
add	Not supported

MIME manipulation does not support manipulating headers in the individual MIME attachments. For example, the Oracle CSM cannot modify the Content-Type given a portion of a message body like this one:

```
--boundary-1
Content-Type: application/sdp
v=0
o=use1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 10000 RTP/AVP 8
a=rtptime:8 PCMA/8000/1
a=sendrecv
a=ptime:20
a=maxptime:200
```

Back Reference Syntax

You can use back reference syntax in the new-value parameter for header and element rules configurations. Denoted by the use of \$1, \$2, \$3, etc. (where the number refers to the regular expression's stored value), you can reference the header and header rule's stored value without having to use the header rule's name. It instead refers to the stored value of this rule.

For example, when these settings are in place:

- header-rule=changeHeader
- action=manipulate
- match-value=(.+)([^;])

you can set the new-value as sip:\$2 instead of sip:\$changeHeader.\$2.

You can use the back reference syntax for:

- Header rule actions manipulate and find-replace-all
- Element rule actions replace and find-replace-all

Using back reference syntax simplifies your configuration steps because you do not need to create a store rule and then manipulate rule; the manipulate rule itself performs the store action if the comparison-type is set to pattern-rule.

Notes on the Regular Expression Library

In the regular expression library, the dot (.) character no longer matches new lines or carriage returns. Conversely, the not-dot does match new lines and carriage returns. This change provides a safety mechanism preventing egregious backtracking of the entire SIP message body when there are no matches. Thus, the Oracle CSM reduces backtracking to a single line within the body. In addition, there is now support for:

Syntax	Description
\s	Whitespace
\S	Non-whitespace
\d	Digits
\D	Non-digits
\R	Any \r, \n, \r\n
\w	Word
\W	Non-word
\A	Beginning of buffer
\Z	End of buffer
\	Any character including newline, in the event that the dot (.) is not

In addition, there is:

- Escaped character shortcuts (\w\W\S\s\d\D\R) operating inside brackets [...]

SIP Message-Body Separator Normalization

The Oracle CSM supports SIP with Multipurpose Internet Mail Extension (MIME) attachments — up to a maximum payload size of 64KB — and has the ability to allow more than the required two CRLFs between the SIP message headers and the multipart body's first boundary. The first two CRLFs that appear in all SIP messages signify the end of the SIP header and the separation of the header and body of the message, respectively. Sometimes additional extraneous CRLFs can appear within the preamble before any text.

The Oracle CSM works by forwarding received SIP messages regardless of whether they contain two or more CRLFs. Although three or more CRLFs are legal, some SIP devices do not accept more than two.

The solution to ensuring all SIP devices accept messages sent from the Oracle CSM is to strip all CRLFs located at the beginning of the preamble before the appearance of any text, ensuring that there are no more than two CRLFs between the end of the last header and the beginning of the body within a SIP message. You enable this feature by adding the new `stripPreambleCrlf` option to the global SIP configuration.

To enable the stripping of CRLFs in the preamble:

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

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

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

```
ORACLE(configure)# session-router
ORACLE#(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. `options`—Set the `options` parameter by typing `options`, a Space, the option name `stripPreambleCrlf` with a plus sign.

```
ORACLE(sip-config)# options +stripPreambleCrlf
```

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 global SIP configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.

5. Save and activate your configuration.

SIP Header Pre-Processing HMR

By default, the Oracle CSM (CSM) performs in-bound SIP manipulations after it carries out header validation. Adding the `inmanip-before-validate` option in the global SIP configuration allows the CSM to perform HMR on received requests prior to header validation. Because there are occasional issues with other SIP implementations—causing invalid headers to be used in messages they send to the CSM—it can be beneficial to use HMR to remove or repair these faulty headers before the request bearing them are rejected.

When configured to do so, the CSM performs pre-validation header manipulation immediately after it executes the top via check. Inbound SIP manipulations are performed in order of increasing precedence: SIP interface, realm, and session agent.

The fact that the top via check happens right before the CSM carries out pre-validation header manipulations means that you cannot use this capability to repairs the first via header if it is indeed invalid. If pre-validation header manipulation were to take place at another time during processing, it would not be possible to use it for SIP session agents. The system learns of matching session agents after top via checking completes.

For logistical reasons, this capability does not extend to SIP responses. Inbound manipulation for responses cannot be performed any sooner that it does by default, a time already preceding any header validation.

To enable SIP header pre-processing:

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

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

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

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

SIP Signaling Services

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. options—Set the options parameter by typing options, a Space, the option name inmanip-before-validate with a plus sign.

```
ORACLE(sip-config)# options +inmanip-before-validate
```

This value allows a the CSM to perform pre-validation header manipulation in order of increasing precedence: SIP interface, realm, and session agent.

5. Save and activate the configuration.

Best Practices

This section lists practices that AOracle recommends you follow for successful implementation of this feature.

- Define all storage rules first.

This recommendation is made because each subsequent header rule processes against the same SIP message, so each additional header rules works off of the results from the application of the rule that precedes it.

In general, you want to store values from the original SIP header rather than from the iteratively changed versions.

- Implement rules at the element rule rather than the header rule level.

Header rules should only be a container for element rules.

- When you are creating rules to edit a header, add additional element rules to modify a single header rather than try to create multiple header rules each with one element rule. That is, create multiple element rules within a header rule rather than creating multiple header rules.
- Do not use header or element rule names that are all capital letters (i.e., \$IP_ADDRESS). Capitals currently refer to predefined rules that are used as macros, and they might conflict with a name that uses capital letters.

About Regular Expressions

Two of the most fundamental ideas you need to know in order to work with regular expressions and with this feature are:

- Regular expressions are a way of creating strings to match other string values.
- You can use groupings in order to create stored values on which you can then operate.

To learn more about regex, you can visit the following Web site, which has information and tutorials that can help to get you started:<http://www.regular-expressions.info/>.

Many of the characters you can type on your keyboard are literal, ordinary characters—they present their actual value in the pattern. Some characters have special meaning, however, and they instruct the regex function (or engine which interprets the expressions) to treat the characters in designated ways. The following table outlines these “special characters” or metacharacters.

Character	Name	Description
.	dot	Matches any one character, including a space; it will match one character, but there must be one character to match. Literally a . (dot) when bracketed ([.]), or placed next to a \ (backslash).
*	star/asterisk	Matches one or more preceding character (0, 1, or any number), bracketed carrier class, or group in parentheses. Used for quantification.

Character	Name	Description
		Typically used with a . (dot) in the format .* to indicate that a match for any character, 0 or more times. Literally an * (asterisk) when bracketed ([*]).
+	plus	Matches one or more of the preceding character, bracketed carrier class, or group in parentheses. Used for quantification. Literally a + (plus sign) when bracketed ([+]).
	bar/vertical bar/pipe	Matches anything to the left or to the right; the bar separates the alternatives. Both sides are not always tried; if the left does not match, only then is the right attempted. Used for alternation.
{	left brace	Begins an interval range, ended with } (right brace) to match; identifies how many times the previous singles character or group in parentheses must repeat. Interval ranges are entered as minimum and maximums ({minimum,maximum}) where the character/group must appear a minimum of times up to the maximum. You can also use these character to set magnitude, or exactly the number of times a character must appear; you can set this, for example, as the minimum value without the maximum ({minimum,}).
?	question mark	Signifies that the preceding character or group in parentheses is optional; the character or group can appear not at all or one time.
^	caret	Acts as an anchor to represent the beginning of a string.
\$	dollar sign	Acts as an anchor to represent the end of a string.
[left bracket	Acts as the start of a bracketed character class, ended with the] (right bracket). A character class is a list of character options; one and only one of the characters in the bracketed class must appear for a match. A - (dash) in between two character enclosed by brackets designates a range; for example [a-z] is the character range of the lower case twenty-six letters of the alphabet. Note that the] (right bracket) ends a bracketed character class unless it sits directly next to the [(left bracket) or the ^ (caret); in those two cases, it is the literal character.
(left parenthesis	Creates a grouping when used with the) (right parenthesis). Groupings have two functions:

SIP Signaling Services

Character	Name	Description
		<p>They separate pattern strings so that a whole string can have special characters within it as if it were a single character.</p> <p>They allow the designated pattern to be stored and referenced later (so that other operations can be performed on it).</p>

Expression Building Using Parentheses

You can now use parentheses () when you use HMR to support order of operations and to simplify header manipulation rules that might otherwise prove complex. This means that expressions such as (sip + urp) - (u + rp) can now be evaluated to sip. Previously, the same expression would have evaluated to sipurprp. In addition, you previously would have been required to create several different manipulation rules to perform the same expression.

SIP Manipulation Configuration

This section explains the parameters that appear in the subelements for the SIP manipulations configuration. Within the SIP manipulations configuration, you can set up SIP header rules, and within those header rules you can configure element rules.

This section also contains several configuration examples for different applications of the HMR feature.

Configuring SIP Header Manipulation Rules

To configure dynamic SIP header manipulation rules:

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

```
ORACLE# configure terminal
```

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

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

3. Type sip-manipulation and press Enter.

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

4. Type header-rules and press Enter.

```
ORACLE(sip-manipulation)# header-rules
```

5. name—Enter the unique identifier for this SIP HMR. There is no default for this value.
6. header-name—Enter the name of the header on which you want the Oracle CSM (CSM) to use this HMR. There is no default for this parameter.

Set this parameter to @status-line, where the at-sign (@)—not allowed in SIP header names—to prevent undesired matches with header having the name status-code.

7. msg-type—Specify the type of message to which this SIP HMR will be applied. The default value is any. The valid values are:

- any | request | reply

8. methods—Enter the method type to use when this SIP HMR is used, such as INVITE, ACK, or CANCEL. When you do not set the method, the CSM applies the rule across all SIP methods.

9. comparison-type—Enter the way that you want SIP headers to be compared from one of the available. This choice dictates how the CSM processes the match rules against the SIP header. the default is refer-case-sensitive. The valid values are:

- boolean | refer-case-sensitive | refer-case-insensitive | pattern-rule | case-sensitive | case-insensitive

10. action—Enter the action that you want this rule to perform on the SIP header. The default value is none. The valid values are:

- add | delete | manipulate | store | none

Remember that you should enter rules with the action type store before you enter rules with other types of actions.

When you set the action type to store, the CSM always treats the match value you enter as a regular expression. As a default, the regular expression is used for the match value is `.+` (which indicates a match value of at least one character), unless you set a more specific regular expression match value.

11. match-value—Enter the value to match against the header value in SIP packets; the CSM matches these against the entire SIP header value. This is where you can enter values to match using regular expression values. Your entries can contain Boolean operators.

When you configure HMR (using SIP manipulation rules, elements rules, etc.), you can now use escape characters in the match-value parameter to support escaping Boolean and string manipulation operators..

You can also escape the escape character itself, so that it is used as a literal string. For example, the CSM now treats the string `\+1234` as `+1234`.

The following are escape characters: `+`, `-`, `+`, `-`, `&`, `|`, `\`, `(`, `)`, `.`, `$`, `^`, and `“`.

You can also use two variables, `$REMOTE_PORT` and `$LOCAL_PORT`, which resolve respectively to the far-end and remote UDP or TCP port value.

12. new-value—When the action parameter is set to add or to manipulate, enter the new value that you want to substitute for the entire header value. This is where you can set stored regular expression values for the CSM to use when it adds or manipulates SIP headers.

When you configure HMR (using SIP manipulation rules, elements rules, etc.), you can now use escape characters in the new-value parameter to support escaping Boolean and string manipulation operators..

You can also escape the escape character itself, so that it is used as a literal string. For example, the CSM now treats the string `\+1234` as `+1234`.

The following are escape characters: `+`, `-`, `+`, `-`, `&`, `|`, `\`, `(`, `)`, `.`, `$`, `^`, and `“`.

You can also use two variables, `$REMOTE_PORT` and `$LOCAL_PORT`, which resolve respectively to the far-end and remote UDP or TCP port value.

Configuring SIP Header Manipulation Element Rules

Element rules are a subset of the SIP header manipulation rules and are applied at the element type level rather than at the entire header value.

To configure dynamic SIP header manipulation rules:

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

```
ORACLE# configure terminal
```

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

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

3. Type sip-manipulation and press Enter.

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

4. Type header-rules and press Enter.

```
ORACLE(sip-manipulation)# header-rules
```

5. Type element-rules and press Enter.

```
ORACLE(sip-header-rules)# element-rules
ORACLE(sip-element-rules)#
```

6. name—Enter the unique identifier for this element rule. There is no default for this value.
7. parameter-name—Enter the SIP header parameter/element on which you want the Oracle CSM (CSM) to use this rule. There is no default for this parameter.
8. type—Specify the type of parameter to which this element rule will be applied. The default value is none. The valid values are:
 - header-value | header-param-name | header-param | uri-display | uri-user | uri-user-param | uri-host | uri-port | uri-param-name | uri-param | uri-header-name | uri-header

To configure HMR so that there is impact only on the status-line; the value will be used for matching according to the comparison-type:

 - status-code—Designates the status code of the response line; accepts any string, but during the manipulation process only recognizes the range from 1 to 699.
 - reason-phrase—Designates the reason of the response line; accepts any string.
9. match-val-type—Enter the value type that you want to match when this rule is applied. The default value is ANY. Valid values are:
 - IP | FQDN | ANY
10. comparison-type—Enter the way that you want SIP headers to be compared from one of the available. This choice dictates how the CSM processes the match rules against the SIP header parameter/element. The default is refer-case-sensitive.
 - boolean | refer-case-sensitive | refer-case-insensitive | pattern-rule
11. action—Enter the action that you want this rule to perform on the SIP header parameter/element. The default is none. The valid rules are:
 - add | replace | delete-element | delete-header | store | none

Remember that you should enter rules with the action type store before you enter rules with other types of actions.

When you set the action type to store, the CSM always treats the match value you enter as a regular expression. As a default, the regular expression is uses for the match value is .+ (which indicates a match value of at least one character), unless you set a more specific regular expression match value.
12. match-value—Enter the value to match against the header value in SIP packets; the CSM matches these against the value of the parameter/element. This is where you can enter values to match using regular expression values, or stored pattern matches. Your entries can contain Boolean operators.

When you configure HMR (using SIP manipulation rules, elements rules, etc.), you can now use escape characters in the match-value parameter to support escaping Boolean and string manipulation operators..

You can also escape the escape character itself, so that it is used as a literal string. For example, the CSM now treats the string \+1234 as +1234.

The following are escape characters: +, -, +^, -^, &, |, \, (,), ., \$, ^, and “.

You can also use two variables, \$REMOTE_PORT and \$LOCAL_PORT, which resolve respectively to the far-end and remote UDP or TCP port value.
13. new-value—When the action parameter is set to add or to manipulate, enter the new value that you want to substitute for the entire header value. This is where you can set stored regular expression values for the CSM to use when it adds or manipulates parameters/elements.

When you configure HMR (using SIP manipulation rules, elements rules, etc.), you can now use escape characters in the new-value parameter to support escaping Boolean and string manipulation operators..

You can also escape the escape character itself, so that it is used as a literal string. For example, the CSM now treats the string \+1234 as +1234.

The following are escape characters: +, -, +^, -^, &, |, \, (,), ., \$, ^, and “.

You can also use two variables, \$REMOTE_PORT and \$LOCAL_PORT, which resolve respectively to the far-end and remote UDP or TCP port value.

Status-Line Manipulation and Value Matching

The Oracle CSM's HMR feature has been enhanced to support the ability to change the status code or reason phrase in SIP responses. This addition—the ability to edit status-lines in responses—builds on HMR's existing ability to edit response headers or the Request-URI in a request.

This section shows you how to configure SIP HMR when you want the Oracle CSM to drop a 183 Session Progress response when it does not have SDP, though flexibility is built into this feature so that you can use it to achieve other ends. In addition, you can now set the SIP manipulation's match-value parameter with Boolean parameters (AND or OR).

Setting the Header Name

SIP header rules (part of the SIP manipulation configuration) now support a new value for the header-name parameter. The value is @status-line, where the at-sign (@)—not allowed in SIP header names—prevents undesired matches with header having the name status-code.

To set the header name for SIP header rules:

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

```
ORACLE# configure terminal
```

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

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

3. Type sip-manipulation and press Enter.

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

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

```
ORACLE(session-router)# header-rules
ORACLE(sip-header-rules)#
```

5. header-name—Enter the new value for the header-name parameter: @status-line.

Setting the Element Type

In the element rules (a subset of the SIP header rules configuration), you can now set the type parameter to either of the following values, both of which will only have an impact on the status-line:

- status-code—Designates the status code of the response line; accepts any string, but during the manipulation process only recognizes the range from 1 to 699
- reason-phrase—Designates the reason of the response line; accepts any string

Like other rule types you can set, the Oracle CSM matches against the value for these using case-sensitive, case-insensitive, or pattern-rule matching (set in the comparison-type parameter for the element rule).

To set the element type:

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

```
ORACLE# configure terminal
```

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

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

3. Type sip-manipulation and press Enter.

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

4. Type header-rules and press Enter.

```
ORACLE(session-router)# header-rules
ORACLE(sip-header-rules)#
```

5. Type `element-rule` and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(sip-header-rules)# element-rules
ORACLE(sip-element-rules)#
```

6. `type`—Enter either `status-code` or `reason-phrase`, the value of which will be used for matching according to the `comparison-type`.

Setting the Match Value

Note that for the SIP header rules and for the SIP element rules, the `match-value` parameter can now be set with these Boolean operators:

- `and` (for which the syntax is the ampersand `&`)
- `or` (for which the syntax is the pipe `|`)

However, you can only use Boolean operators in this value when you set the `comparison-type` parameter to `pattern-rule` and are evaluating stored matches. The Oracle CSM evaluates these Boolean expressions from left to right, and does not support any grouping mechanisms that might change the order of evaluation. For example, the Oracle CSM evaluates the expression `A & B | C` (where `A=true`, `B=false`, and `C=true`) as follows: `A & B = false`; `false | true = true`.

You can set the `match-value` for the SIP header rules or for the SIP element rules.

To set a match value in the SIP header rules configuration:

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

```
ORACLE# configure terminal
```

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

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

3. Type `sip-manipulation` and press Enter.

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

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

```
ORACLE(session-router)# header-rules
ORACLE(sip-header-rules)#
```

5. `match-value`—Enter the value to match against the header value in SIP packets; the Oracle CSM matches these against the entire SIP header value. This is where you can enter values to match using regular expression values; your entries can contain Boolean operators.

To set a match value in the SIP element rules configuration:

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

```
ORACLE# configure terminal
```

7. Type `session-router` and press Enter to access the signaling-level configuration elements.

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

8. Type `sip-manipulation` and press Enter.

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

9. Type `header-rules` and press Enter.

```
ORACLE(session-router)# header-rules
ORACLE(sip-header-rules)#
```


10. Type `element-rule` and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(sip-header-rules)# element-rules
ORACLE(sip-element-rules)#
```

11. `match-value`—Enter the value to match against the header value in SIP packets; the Oracle CSM matches these against the value of the parameter/element. This is where you can enter values to match using regular expression values, or stored pattern matches; your entries can contain Boolean operators.

Setting the Response Code Block

To enable the SIP HMR enhancements, you need to set an option in SIP interface configuration that keeps the Oracle CSM from sending the response you designate.

Note that this example sets the `dropResponse` option to 699, where 699 is an arbitrary code used to later match the HMR.

To enable SIP response blocking for a 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 signaling-level configuration elements.

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

3. Type `sip-interface` and press Enter.

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

If you are adding support for this feature to a pre-existing SIP interface, then you must select (using the `ACLI select` command) the configuration that you want to edit.

4. `options`—Set the `options` parameter by typing `options`, a Space, the option name `dropResponse` with a plus sign in front of it, type the equal sign and the code(s) or range(s) you want blocked. If there is more than one, separate your entries with a colon. Then press Enter.

```
ORACLE(sip-interface)# options +dropResponse=699
```

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

5. Save and activate your configuration.

Configuring MIME Support

The `find-replace-all` action has been added to the header rules. Element rules support the `find-replace-all` action and the `mime` type.

To set the header rule with the `find-replace-all` action:

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

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

2. Type `session-router` and press Enter.

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

3. Type `sip-manipulation` and press Enter.

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

4. Type `header-rules` and press Enter.

```
ORACLE(sip-manipulation)# header-rules
ORACLE(sip-header-rules)#
```

5. action—Set the action parameter to find-replace-all if you want to enable SIP HMR MIME manipulation.
6. Save and activate your configuration.

To set the element rule with the find-replace-all action and MIME type:

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

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

8. Type session-router and press Enter.

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

9. Type sip-manipulation and press Enter.

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

10. Type header-rules and press Enter.

```
ORACLE(sip-manipulation)# header-rules
ORACLE(sip-header-rules)#
```

11. Type element-rules and press Enter.

```
ORACLE(sip-header-rules)# element-rules
```

12. ORACLE(sip-element-rules)#

13. action—Set the action parameter to find-replace-all if you want to enable SIP HMR MIME manipulation.

14. type—Set the type parameter to mime if you want to enable SIP HMR MIME manipulation.

15. Save and activate your configuration.

Testing Pattern Rules

The Oracle CSM it yields the results you require. This command is useful for testing the regex values that you devise because it will tell you whether that value is valid or not.

This command is called test-pattern-rule.

To test a pattern rule:

1. Type test-pattern-rule and press Enter.

```
ORACLE# test-pattern-rule
ORACLE(test-pattern-rule)#
```

2. expression—Enter the regular expression that you want to test. If there is a match, then the Oracle CSM will inform you of it; you will also be informed if there is no match.

The string against which the Oracle CSM is matching is not the string parameter that you can use for this command; it is the string value of the regular expression you entered.

```
ORACLE(test-pattern-rule)# expression '.*;tgid=(.+) .*'
```

3. string—Enter the string against which you want to compare the regular expression.

```
ORACLE(test-pattern-rule)# string sip:+17024260002@KCMGGWC;user=phone SIP/
2.0;tgid=Trunk1
expression made 3 matches against string
```

4. show—Use the show command within test-pattern-rules to view the test pattern that you entered, whether there was a match, and the number of matches.

```
ORACLE(test-pattern-rule)# show
Pattern Rule:
Expression  : .*(;tgid=(.+) ).*
String      : sip:+17024260002@KCMGGWC;user=phone SIP/2.0;tgid=Trunk1
Matched     : TRUE
```

```
Matches:
$0 sip:+17024260002@KCMGGWC;user=phone SIP/2.0;tgid=Trunk1
$1 ;tgid=Trunk1
$2 Trunk1
```

Configuring SIP HMR Sets

This section shows you how to configure your multiple SIP HMR sets.

Remember to run the ACLI `verify-config` command prior to activating your configuration so the Oracle CSM can detect any invalid or circular references.

To set the parameters enabling the use of SIP HMR sets:

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

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

2. Type `session-router` and press Enter.

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

3. Type `sip-manipulation` and press Enter.

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

4. Type `header-rules` and press Enter.

```
ORACLE (session-router) # header-rules
ORACLE (sip-header-rules) #
```

5. Type `element-rule` and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE (sip-header-rules) # element-rules
ORACLE (sip-element-rules) #
```

6. `action`—Enter the `sip-manip` value to enable use this rule for a SIP HMR set. This value then invoke the rule identified in the `new-value` parameter.

7. `new-value`—To use SIP HMR sets, enter the name of the manipulation rule you want invoked for the set.

8. Save and activate your configuration.

Configuration Examples

This section shows you several configuration examples for HMR. This section shows the configuration for the various rules that the Oracle CSM applied, and sample results of the manipulation. These examples present configurations as an entire list of fields and settings for each ruleset, nested header rules and nested element rules. If a field does not have any operation within the set, the field is shown with the setting at the default or blank.

Example 1 Removing Headers

For this manipulation rule, the Oracle CSM removes the Custom header if it matches the pattern rule. It stores the defined pattern rule for the goodBye header. Finally, it removes the goodBye header if the pattern rule from above is a match.

This is a sample of the configuration:

```
sip-manipulation
  name                               removeHeader
  header-rule
    name                               removeCustom
    header-name                         Custom
    action                              delete
    comparison-type                     boolean
    match-value                         ^This is my.*
    msg-type                             request
```

```

        new-value
        methods                               INVITE
header-rule
        name                                   goodByeHeader
        header-name                             Goodbye
        action                                   store
comparison-type    boolean
        match-value                             ^Remove (.+)
        msg-type                                 request
        new-value
        methods                               INVITE
header-rule
        name                                   goodBye
action            delete
        comparison-type                         pattern-rule
        match-value                             $goodByeHeader
        msg-type                                 request
        new-value
        methods                               INVITE

```

This is a sample of the result:

```

Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP
192.168.200.61:5060;branch=z9hG4bK0g639r10fgc0aakk26s1.1
    From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDclrm601-1
    To: sut <sip:service@192.168.1.61:5060>
    Call-ID: SDclrm601-d01673bcacfcc112c053d95971330335-06a3gu0
    CSeq: 1 INVITE
    Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
    Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
    Params: sipp <sip:sipp1@192.168.1.60:5060>
    Params: sipp <sip:sipp2@192.168.1.60:5060>
    Edit: disp <sip:user@192.168.1.60:5060>
    Max-Forwards: 69
    Subject: Performance Test
    Content-Type: application/sdp
    Content-Length: 140

```

Example 2 Manipulating the Request URI

For this manipulation rules, the Oracle CSM stores the URI parameter `tgid` in the Request URI. Then if the pattern rule matches, it adds a new header (`x-customer-profile`) with the a new header value `tgid` to the URI parameter in the request URI.

This is a sample of the configuration:

```

sip-manipulation
  name                               CustomerTgid
  header-rule
    name                               ruriRegex
    header-name                         request-uri
    action                               store
    comparison-type                     pattern-rule
    match-value
    msg-type                             request
new-value
  methods                               INVITE
  element-rule
    name                               tgidParam
    parameter-name                     tgid
    type                               uri-param
    action                               store
    match-val-type                     any

```

```

                                comparison-type
                                match-value
                                new-value
header-rule
    name                         addCustomer
    header-name                  X-Customer-Profile
    action                       add
    comparison-type              pattern-rule
    match-value                  $ruriRegex.$tgidParam
    msg-type                    request
    new-value                    $ruriRegex.$tgidParam.$0
    methods                      INVITE
header-rule
    name                         delTgid
    header-name                  request-uri
    action                       manipulate
    comparison-type              pattern-rule
    match-value                  $ruriRegex.$tgidParam
    msg-type                    request
    new-value
    methods                      INVITE
    element-rule
        name                     tgidParam
        parameter-name           tgid
        type                     uri-param
        action                   delete-element
        match-val-type           any
        comparison-type          case-sensitive
        match-value              $ruriRegex.$tgidParam.
$0
                                new-value

```

This is a sample of the result:

```

Request-Line: INVITE sip:service@192.168.200.60:5060 SIP/2.0
  Message Header
Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK0g6plv3088h03acgh6c1.1
  From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDclrg601-1
  To: sut <sip:service@192.168.1.61:5060>
  Call-ID: SDclrg601-f125d8b0ec7985c378b04cab9f91cc09-06a3gu0
  CSeq: 1 INVITE
  Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
  Goodbye: Remove Me
  Custom: This is my custom header
  Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
Params: sipp <sip:sipp1@192.168.1.60:5060>
  Params: sipp <sip:sipp2@192.168.1.60:5060>
  Edit: disp <sip:user@192.168.1.60:5060>
  Max-Forwards: 69
  Subject: Performance Test
  Content-Type: application/sdp
  Content-Length: 140
  X-Customer-Profile: 123

```

Example 3 Manipulating a Header

For this manipulation rule, the Oracle CSM stores the pattern matches for the Custom header, and replaces the value of the Custom header with a combination of the stored matches and new content.

This is a sample of the configuration:

```

sip-manipulation
  name                         modCustomHdr
  header-rule
    name                       customSearch

```

```

header-rule
    header-name      Custom
    action           store
    comparison-type  pattern-rule
    match-value      (This is my )(.+)( header)
    msg-type         request
    new-value
    methods          INVITE

name               customMod
header-name       Custom
action            manipulate
comparison-type   pattern-rule
match-value       $customSearch
msg-type          request
new-value

methods           INVITE
element-rule
    name           hdrVal
    parameter-name hdrVal
    type           header-value
    action         replace
    match-val-type any
    comparison-type case-sensitive
    match-value

new-value          $customSearch.$1+edited+$customSearch.$3

```

This is a sample of the result:

```

Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP
192.168.200.61:5060;branch=z9hG4bK20q2s820boghbacgs6o0.1
    From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDe1ra601-1
    To: sut <sip:service@192.168.1.61:5060>
    Call-ID: SDe1ra601-4bb668e7ec9eeb92c783c78fd5b26586-06a3gu0
    CSeq: 1 INVITE
    Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
    Goodbye: Remove Me
    Custom: This is my edited header
    Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
    Params: sipp <sip:sipp1@192.168.1.60:5060>
    Params: sipp <sip:sipp2@192.168.1.60:5060>
    Edit: disp <sip:user@192.168.1.60:5060>
    Max-Forwards: 69
    Subject: Performance Test
    Content-Type: application/sdp
    Content-Length: 140

```

Example 4 Storing and Using URI Parameters

For this manipulation rule, the Oracle CSM stores the value of the URI parameter tag from the From header. It also creates a new header FromTag with the header value from the stored information resulting from the first rule.

This is a sample of the configuration:

```

sip-manipulation
    name           storeElemParam
    header-rule
        name       Frohmr
        header-name From
        action     store
        comparison-type case-sensitive
        match-value
        msg-type   request
        new-value

```

```

methods                               INVITE
element-rule
  name                                 elementRule
  parameter-name                       tag
  type                                  uri-param
  action                                 store
  match-val-type                       any
  comparison-type                      case-sensitive
  match-value
  new-value

header-rule
  name                                 newHeader
  header-name                          FromTag
  action                                 add
  comparison-type                      pattern-rule
  match-value                          $FromHR.$elementRule
  msg-type                              any
  new-value                             $FromHR.$elementRule.$0
  methods

```

This is a sample of the result:

```

Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP
192.168.200.61:5060;branch=z9hG4bK4oda2e2050ih7acgh6c1.1
    From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDflre601-1
    To: sut <sip:service@192.168.1.61:5060>
    Call-ID: SDflre601-f85059e74e1b443499587dd2dee504c2-06a3gu0
    CSeq: 1 INVITE
    Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
    Goodbye: Remove Me
    Custom: This is my custom header
    Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
    Params: sipp <sip:sipp1@192.168.1.60:5060>
    Params: sipp <sip:sipp2@192.168.1.60:5060>
    Edit: disp <sip:user@192.168.1.60:5060>
    Max-Forwards: 69
    Subject: Performance Test
    Content-Type: application/sdp
Content-Length: 140
  FromTag: 1

```

Example 5 Manipulating Display Names

For this manipulation rule, the Oracle CSM sores the display name from the Display header. It replaces the two middle characters of the original display name with a new string. Then is also replaces the From header's display name with "abc 123" if it matches sipp.

This is a sample of the configuration:

```

sip-manipulation
  name                                 modDisplayParam
  header-rule
    name                                 storeDisplay
    header-name                          Display
    action                                 store
    comparison-type                      case-sensitive
    match-value
    msg-type                              request
    new-value
    methods                              INVITE
    element-rule
      name                                 displayName
      parameter-name                      display

```

```

                                type                uri-display
                                action              store
                                match-val-type      any
comparison-type                pattern-rule
                                match-value        (s) (ip) (p )
                                new-value
header-rule
    name                        modDisplay
    header-name                 Display
    action                      manipulate
    comparison-type             case-sensitive
    match-value
    msg-type                    request
    new-value
    methods                     INVITE
    element-rule
        name                    modRule
        parameter-name          display
        type                    uri-display
        action                  replace
        match-val-type          any
        comparison-type         pattern-rule
        match-value             $storeDisplay.
$displayName
    new-value                   $storeDisplay.
$displayName.$1+lur+$storeDisplay.$displayName.$3
header-rule
    name                        modFrom
    header-name                 From
    action                      manipulate
    comparison-type             pattern-rule
    match-value
    msg-type                    request
    new-value
    methods                     INVITE
    element-rule
        name                    fromDisplay
        parameter-name
        type                    uri-display
        action                  replace
        match-val-type          any
        comparison-type         pattern-rule
        match-value             sipp
        new-value               "\"abc 123\" "

```

This is a sample of the result:

```

Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP
192.168.200.61:5060;branch=z9hG4bK681kot109gp04acgs6o0.1
  From: "abc 123" <sip:sipp@192.168.1.60:5060>;tag=SD79ra601-1
  To: sut <sip:service@192.168.1.61:5060>
  Call-ID: SD79ra601-a487f1259e2370d3dbb558c742d3f8c4-06a3gu0
  CSeq: 1 INVITE
  Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
  Goodbye: Remove Me
  Custom: This is my custom header
  Display: slurp <sip:user@192.168.1.60:5060;up=abc>;hp=123
  Params: sipp <sip:sipp1@192.168.1.60:5060>
  Params: sipp <sip:sipp2@192.168.1.60:5060>
  Edit: disp <sip:user@192.168.1.60:5060>
  Max-Forwards: 69
  Subject: Performance Test

```



```
Content-Type: application/sdp
Content-Length: 140
```

Example 6 Manipulating Element Parameters

For this more complex manipulation rule, the Oracle CSM:

- From the Display header, stores the display name, user name, URI parameter up, and header parameter hp
- Adds the header parameter display to the Params header, with the stored value of the display name from the first step
- Add the URI parameter user to the Params header, with the stored value of the display name from the first step
- If the URI parameter match succeeds in the first step, replaces the URI parameter up with the Display header with the value def
- If the header parameter match succeeds in the first step, deletes the header parameter hp from the Display header

This is a sample of the configuration:

```
sip-manipulation
  name                               elemParams
  header-rule
    name                               StoreDisplay
    header-name                         Display
    action                               store
    comparison-type                     case-sensitive
    match-value
    msg-type                             request
    new-value
    methods                              INVITE
    element-rule
      name                               displayName
      parameter-name                     uri-display
      type                               any
      action                               store
      match-val-type                     pattern-rule
      comparison-type                     any
      match-value
      new-value
  element-rule
    name                               userName
    parameter-name                       user
    type                                 uri-user
    action                               store
    match-val-type                       any
    comparison-type                     pattern-rule
    match-value
    new-value
  element-rule
    name                                 uriParam
    parameter-name                       up
    type                                 uri-param
    action                               store
    match-val-type                       any
    comparison-type                     pattern-rule
    match-value
    new-value
  element-rule
    name                                 headerParam
    parameter-name                       hp
    type                                 header-param
    action                               store
    match-val-type                       any
    comparison-type                     pattern-rule
    match-value
```

```

        new-value
header-rule
    name                    EditParams
    header-name             Params
    action                  manipulate
    comparison-type        case-sensitive
    match-value
    msg-type                request
    new-value
    methods                 INVITE
    element-rule
        name                addHeaderParam
        parameter-name      display
        type                 header-param
        action              add
match-val-type             any
    comparison-type        case-sensitive
    match-value
    new-value              $StoreDisplay.
$displayName.$0
    element-rule
        name                addUriParam
        parameter-name      user
        type                 uri-param
        action              add
        match-val-type      any
        comparison-type     case-sensitive
        match-value
    new-value
$StoreDisplay.$userName.$0
    header-rule
        name                    EditDisplay
        header-name             Display
        action                  manipulate
        comparison-type        case-sensitive
        match-value
        msg-type                request
        new-value
        methods                 INVITE
        element-rule
            name                replaceUriParam
            parameter-name      up
            type                 uri-param
            action              replace
            match-val-type      any
            comparison-type     pattern-rule
            match-value        $StoreDisplay.$uriParam
            new-value          def
        element-rule
            name                delHeaderParam
            parameter-name      hp
            type                 header-param
            action              delete-element
            match-val-type      any
            comparison-type     pattern-rule
            match-value        $StoreDisplay.$headerParam
            new-value

```

This is a sample of the result:

```

Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP
192.168.200.61:5060;branch=z9hG4bK7okvei0028jgdacgh6c1.1

```

```

From: sipp <sip:sipp@192.168.1.60:5060>;tag=SD89rm601-1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: SD89rm601-b5b746cef19d0154cb1f342cb04ec3cb-06a3gu0
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
Goodbye: Remove Me
Custom: This is my custom header
Display: sipp <sip:user@192.168.1.60:5060;up=def>
Params: sipp <sip:sipp1@192.168.1.60:5060;user=user>;display=sipp
Params: sipp <sip:sipp2@192.168.1.60:5060;user=user>;display=sipp
Edit: disp <sip:user@192.168.1.60:5060>
Max-Forwards: 69
Subject: Performance Test
Content-Type: application/sdp
Content-Length: 140

```

Example 7 Accessing Data from Multiple Headers of the Same Type

For this manipulation rule, the Oracle CSM stores the user name from the Params header. It then adds the URI parameter c1 with the value stored from the first Params header. Finally, it adds the URI parameter c2 with the value stored from the second Params header.

This is a sample of the configuration:

```

sip-manipulation
  name
  header-rule
    name
    header-name
    action
    comparison-type
    match-value
    msg-type
    new-value
    methods
    element-rule
      name
      parameter-name
      type
      action
      match-val-type
      comparison-type
      match-value
      new-value
  header-rule
    name
    header-name
    action
    comparison-type
    match-value
    msg-type
    new-value
  methods
    element-rule
      name
      parameter-name
      type
      action
      match-val-type
      comparison-type
      match-value
      new-value
  $storeUserName.$0
  element-rule
    name
    parameter-name
    type
    action
    match-val-type
    comparison-type
    match-value
    new-value
    addParam1
    c1
    uri-param
    add
    any
    case-sensitive
    $storeParams[0].

```

```

name addParam2
parameter-name c2
type uri-param
action add
match-val-type any
comparison-type case-sensitive
match-value
new-value $storeParams[1].

$storeUserName.$0

```

This is a sample of the result:

```

Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP
192.168.200.61:5060;branch=z9hG4bK9g855p30cos08acgs6o0.1
    From: sipp <sip:sipp@192.168.1.60:5060>;tag=SD99ri601-1
    To: sut <sip:service@192.168.1.61:5060>
    Call-ID: SD99ri601-6f5691f6461356f607b0737e4039caec-06a3gu0
    CSeq: 1 INVITE
    Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
    Goodbye: Remove Me
    Custom: This is my custom header
    Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
    Params: sipp <sip:sipp1@192.168.1.60:5060>
    Params: sipp <sip:sipp2@192.168.1.60:5060>
    Edit: disp <sip:user@192.168.1.60:5060;c1=sipp1;c2=sipp2>
    Max-Forwards: 69
    Subject: Performance Test
    Content-Type: application/sdp
    Content-Length: 140

```

Example 8 Using Header Rule Special Characters

For this manipulation rule, the Oracle CSM:

- Stores the header value of the Params header with the given pattern rule, and stores both the user name of the Params header and the URI parameter abc
- Adds the URI parameter lpu with the value stored from the previous Params header
- If any of the Params headers match the pattern rule defined in the first step, adds the URI parameter apu with the value apu
- If all of the Params headers match the pattern rule defined in the first step, adds the URI parameter apu with the value apu
- If the first Params headers does not match the pattern rule for storing the URI parameter defined in the first step, adds the URI parameter not with the value 123
- If the first Params headers matches the pattern rule for storing the URI parameter defined in the first step, adds the URI parameter yes with the value 456

This is a sample of the configuration:

```

sip-manipulation
  name specialChar
  header-rule
    name searchParams
    header-name Params
    action store
    comparison-type pattern-rule
    match-value .*sip:(.+)@.*
    msg-type request
    new-value
    methods INVITE
  element-rule
    name userName

```

```

parameter-name
type
action
match-val-type
comparison-type
match-value
new-value
uri-user
store
any
case-sensitive

element-rule
name
parameter-name
type
action
match-val-type
comparison-type
match-value
new-value
emptyUriParam
abc
uri-param
store
any
pattern-rule

header-rule
name
header-name
action
comparison-type
match-value
msg-type
new-value
methods
element-rule
name
parameter-name
type
action
match-val-type
comparison-type
match-value
new-value
$searchParams[^].$userName.$0
lastParamUser
lpu
uri-param
add
any
case-sensitive
element-rule
name
parameter-name
type
action
match-val-type
comparison-type
match-value
new-value
anyParamUser
apu
uri-param
add
any
pattern-rule
$searchParams[~]
aup
element-rule
name
parameter-name
type
action
match-val-type
comparison-type
match-value
new-value
allParamUser
apu
header-param
add
any
pattern-rule
$searchParams[*]
apu
element-rule
name
parameter-name
type
action
match-val-type
comparison-type
match-value
notParamYes
not
uri-param
add
any
pattern-rule
!$searchParams.

$emptyUriParam
new-value
123
element-rule
name
notParamNo

```

	parameter-name	yes
	type	uri-param
	action	add
	match-val-type	any
	comparison-type	pattern-rule
	match-value	\$searchParams.
\$emptyUriParam		
	new-value	456

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP
192.168.200.61:5060;branch=z9hG4bK681m9t30e0qh6akgj2s1.1
    From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDchrc601-1
    To: sut <sip:service@192.168.1.61:5060>
    Call-ID: SDchrc601-fcf5660a56e2131fd27f12fcbd169fe8-06a3gu0
    CSeq: 1 INVITE
    Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
    Goodbye: Remove Me
    Custom: This is my custom header
    Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
    Params: sipp <sip:sipp1@192.168.1.60:5060>
    Params: sipp <sip:sipp2@192.168.1.60:5060>
    Edit: disp
<sip:user@192.168.1.60:5060;lpu=sipp2;apu=apu;not=123>;apu=apu
    Max-Forwards: 69
    Subject: Performance Test
    Content-Type: application/sdp
    Content-Length: 140
```

Example 9 Status-Line Manipulation

This section shows an HMR configuration set up for status-line manipulation.

Given that the object of this example is to drop the 183 Session Progress response when it does not have SDP, your SIP manipulation configuration needs to:

1. Search for the 183 Session Progress response
2. Determine if the identified 183 Session Progress responses contain SDP; the Oracle CSM searches the 183 Session Progress responses where the content length is zero
3. If the 183 Session Progress response does not contain SDP, change its status code to 699
4. Drop all 699 responses

```
sip-manipulation
  name
  description
  header-rule
    name
    header-name
    action
    comparison-type
    match-value
    msg-type
    new-value
    methods
  header-rule
    name
    header-name
    action
    comparison-type
    match-value
    msg-type
```

manip	IsContentLength0
	Content-Length
	store
	pattern-rule
	0
	reply
	is183
	@status-line
	store
	pattern-rule
	reply

```

new-value
methods
element-rule
name
    is183Code
    parameter-name
    type
    status-code
    action
    store
    match-val-type
    any
    comparison-type
    pattern-rule
    match-value
    183
    new-value

header-rule
    name
    change183
    header-name
    @status-line
    action
    manipulate
    comparison-type
    case-sensitive
    match-value
    msg-type
    reply
    new-value
    methods
    element-rule
        name
        make199
        parameter-name
        type
        status-code
        action
        replace
        match-val-type
        any
        comparison-type
        pattern-rule
        match-value
        $IsContentLength0 &
        new-value
        199

sip-interface
    options dropResponse=699

```

Example 10 Use of SIP HMR Sets

The following example shows the configuration for SIP HMR with one SIP manipulation configuration loading another SIP manipulation configuration. The goals of this configuration are to:

- Add a new header to an INVITE
- Store the user portion of the Request URI
- Remove all Route headers from the message only if the Request URI is from a specific user

```

sip-manipulation
    name
    deleteRoute
    description
    delete all Route Headers
    header-rule
        name
        deleteRoute
        header-name
        Route
        action
        delete
        comparison-type
        case-sensitive
        match-value
        msg-type
        request
        new-value
        methods
        INVITE
sip-manipulation
    name
    addAndDelete
    description
    Add a New header and delete Route
headers
    header-rule
        name
        addHeader
        header-name
        New
        action
        add
        comparison-type
        case-sensitive
        match-value

```

```

        msg-type          request
        new-value         "Some Value"
        methods           INVITE
header-rule
    name                 storeRURI
    header-name          request-uri
    action               store
    comparison-type      pattern-rule
    match-value
msg-type          request
new-value
methods           INVITE
element-rule
    name                 storeUser
    parameter-name
    type                 uri-user
    action               store
    match-val-type       any
    comparison-type      pattern-rule
    match-value          305.*
    new-value
header-rule
    name                 deleteHeader
    header-name          request-uri
    action               sip-manip
    comparison-type      Boolean
    match-value          $storeRURI.$storeUser
    msg-type            request
    new-value            deleteRoute
    methods              INVITE

```

Example 11 Use of Remote and Local Port Information

The following example shows the configuration for remote and local port information. The goals of this configuration are to:

- Add LOCAL_PORT as a header parameter to the From header
- Add REMOTE_PORT as a header parameter to the From header

```

sip-manipulation
    name                 addOrigIp
    description
    header-rule
        name                 addIpParam
        header-name          From
        action               manipulate
        comparison-type      case-sensitive
        match-value
    msg-type            request
    new-value
    methods              INVITE
    element-rule
        name                 addIpParam
        parameter-name        newParam
        type                 header-param
        action               add
        match-val-type       any
        comparison-type      case-sensitive
        match-value
        new-value            $LOCAL_IP
    element-rule
        name                 addLocalPort
        parameter-name        lport
        type                 header-param

```



```

        action                add
        match-val-type        any
        comparison-type       case-sensitive
        match-value
        new-value             $LOCAL_PORT
    element-rule
        name                  addRemotePort
        parameter-name        rport
        type                  header-param
        action                add
        match-val-type        any
        comparison-type       case-sensitive
        match-value
        new-value             $REMOTE_PORT

```

Example 12 Response Status Processing

Given that the object of this example is to drop the 183 Session Progress response when it does not have SDP, your SIP manipulation configuration needs to:

1. Search for the 183 Session Progress response
2. Determine if the identified 183 Session Progress responses contain SDP; the Oracle CSM searches the 183 Session Progress responses where the content length is zero
3. If the 183 Session Progress response does not contain SDP, change its status code to 699
4. Drop all 699 responses

```

sip-manipulation
  name                manip
  description
  header-rule
    name              IsContentLength0
    header-name       Content-Length
    action            store
    comparison-type   pattern-rule
    match-value       0
    msg-type          reply
    new-value
    methods
  header-rule
    name              is183
    header-name       @status-line
    action            store
    comparison-type   pattern-rule
    match-value
    msg-type          reply
    new-value
    methods
    element-rule
      name            is183Code
      parameter-name
      type            status-code
      action          store
      match-val-type  any
      comparison-type pattern-rule
      match-value     183
      new-value
  header-rule
    name              changel83
    header-name       @status-line
    action            manipulate
    comparison-type   case-sensitive
    match-value
    msg-type          reply

```

```

new-value
methods
element-rule
    name
    parameter-name
    type
    action
    match-val-type
    comparison-type
    match-value
$isl83.$isl83Code
new-value
sip-interface
    options dropResponse=699

```

The following four configuration examples are based on the this sample SIP INVITE:

```

INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary
Content-Length: 466
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 12345 RTP/AVP 18
a=rtpmap:8 G729/8000/1
a=fmtp:18 annexb=no
a=sendrecv
a=ptime:20
a=maxptime:200
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=video 12345 RTP/AVP 34
a=rtpmap:34 H263a/90000
a=ptime:30
--boundary--

```

Example 13 Remove a Line from SDP

In this example, the SIP manipulation is configured to remove all p-time attributes from the SDP.

```

sip-manipulation
    name
    description
    header-rule
        name
        header-name
        action
        comparison-type
        match-value
removePtimeFromBody
removes ptime attribute from all bodies
CTypeManp
Content-Type
manipulate
case-sensitive

```

```

msg-type request
new-value
methods INVITE
element-rule
  name remPtime
  parameter-name application/sdp
  type mime
  action find-replace-all
  match-val-type any
  comparison-type case-sensitive
  match-value a=ptime:[0-9]{1,2}(\n|
\r\n)
new-value

```

The result of manipulating the original SIP INVITE (shown above) with the configured SIP manipulation is:

```

INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary
Content-Length: 466
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 12345 RTP/AVP 18
a=rtpmap:18 G729/8000/1
a=fmtp:18 annexb=no
a=sendrecv
a=maxptime:200
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=video 12345 RTP/AVP 34
a=rtpmap:34 H263a/90000
--boundary-

```

Example 14 Back Reference Syntax

In this sample of back-reference syntax use, the goal is to change the To user. The SIP manipulation would be configured like the following:

```

sip-manipulation
  name changeToUser
  description change user in the To header
  header-rule
    name ChangeHeader
    header-name To
    action manipulate
    comparison-type case-sensitive
    match-value
    msg-type request
    new-value

```

methods	INVITE
element-rule	
name	replaceValue
parameter-name	
type	header-value
action	replace
match-val-type	any
comparison-type	pattern-rule
match-value	(.+) (service) (.+)
new-value	\$1+Bob+\$3

The result of manipulating the original SIP INVITE (shown above) with the configured SIP manipulation is:

```
INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:Bob@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary
Content-Length: 466
...
...
...
```

Example 15 Change and Remove Lines from SDP

In this sample of changing and removing lines from the SDP, the goal is to convert the G.729 codec to G.729a. The SIP manipulation would be configured like the following:

```
sip-manipulation
  name          std2prop-codec-name
  description   rule to translate standard to
proprietary codec name
  header-rule
    name          CTypeManp
    header-name   Content-Type
    action        manipulate
    comparison-type case-sensitive
    match-value
    msg-type      any
    new-value
    methods
    element-rule
      name          g729-annexb-no-std2prop
      parameter-name application/sdp
      type          mime
      action        find-replace-all
      match-val-type any
      comparison-type case-sensitive
      match-value   a=rtpmap:[0-9]{1,3}
(G729/8000/1\r\na=fmtp:[0-9]{1,3} annexb=no)[[::]]
      new-value     G729a/8000/1
```

The result of manipulating the original SIP INVITE (shown above) with the configured SIP manipulation is:

```
INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
```

```

Content-Type: multipart/mixed;boundary=boundary
Content-Length: 466
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 12345 RTP/AVP 8
a=rtpmap:18 G729a/8000/1
a=sendrecv
a=maxptime:200
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=video 12345 RTP/AVP 34
a=rtpmap:34 H263a/90000
--boundary-

```

Example 16 Change and Add New Lines to the SDP

In this sample of changing and adding lines from the SDP, the goal is to convert non-standard codec H.263a to H.263. The SIP manipulation would be configured like the following:

```

sip-manipulation
  name                prop2std-codec-name
  description         rule to translate proprietary to
  standard codec name
  header-rule
    name              CodecManp
    header-name       Content-Type
    action             manipulate
    comparison-type   case-sensitive
    match-value
    msg-type          any
    new-value
    methods
    element-rule
      name            H263a-prop2std
      parameter-name  application/sdp
      type            mime
      action          find-replace-all
      match-val-type any
      comparison-type case-sensitive
      match-value     a=rtpmap:([0-9]{1,3})
H263a/.*\r\n
  new-value           a=rtpmap:+$1+"
H263/90000"+$CRLF+a=fmtp:+$1+" QCIF=4"+$CRLF

```

The result of manipulating the original SIP INVITE (shown above) with the configured SIP manipulation is:

```

INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary

```

```
Content-Length: 466
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 12345 RTP/AVP 8
a=rtpmap:18 G729/8000/1
a=fmtp:18 annexb=no
a=sendrecv
a=maxptime:200
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=video 12345 RTP/AVP 34
a=rtpmap:34 H263/90000
a=fmtp:34 QCIF=4
--boundary-
```

Dialog-Matching Header Manipulation

The most common headers to manipulate using HMR are the To-URI and From-URI. Along with the to-tag, from-tag, and Call-ID values, these are also all headers that represent dialog-specific information that must match the UAC and UAS to be considered part of the same dialog. If these parameters are modified through HMR, the results can be that the UAC or UAS rejects messages.

While it is possible to ensure that dialog parameters match correctly using regular HMR, this feature offers a simpler and less error-prone method of doing so.

In addition, this section describes the addition of built-in SIP manipulations defined by Oracle best practices, and a new method of testing your SIP manipulations.

About Dialog-Matching Header Manipulations

The goal of this feature is to maintain proper dialog-matching through manipulation of dialog-specific information using HMR. Two fundamental challenges arise when looking at the issue of correctly parameters manipulating dialog-matching:

- Inbound HMR
- Outbound HMR

The new setting out-of-dialog (for the msg-type parameter) addresses these challenges by offering an intelligent more of dialog matching of messages for inbound and outbound HMR requests. This is a msg-type parameter, meaning that it becomes matching criteria for operations performed against a message. If you also specify methods (such as REGISTER) as matching criteria, then the rule is further limited to the designated method.

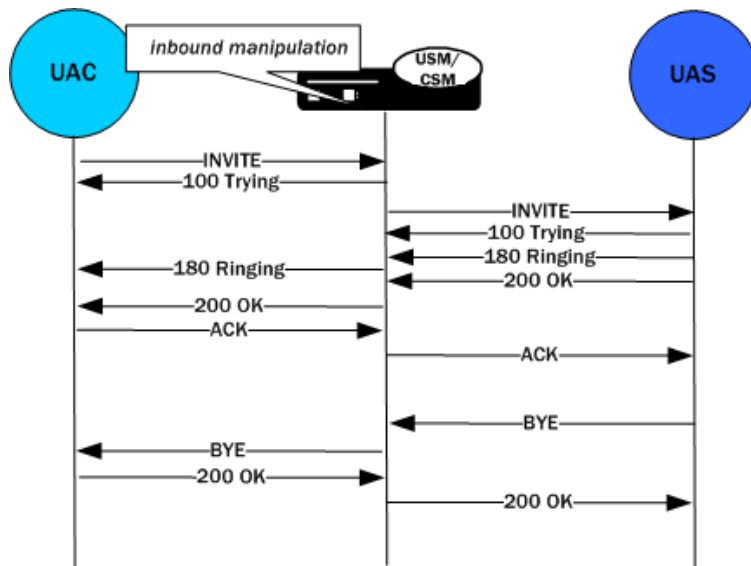
For both inbound and outbound manipulations, using the out-of-dialog setting means the message must be a request without a to-tag in order to perform the manipulation.

Inbound HMR Challenge

Since inbound manipulations take place before the message reaches the core of Oracle CSM SIP processing, the SIP proxy takes the manipulated header as what was directly received from the client. This can cause problems for requests leaving the Oracle CSM for the UAC because the dialog will not match the initial request sent.

So the unmodified header must be cached because for any subsequent request (as in the case of a BYE originating from the terminator; see the diagram below) the Oracle CSM might need to restore the original value—enabling the UAC to identify the message correctly as being part of the same dialog. For out-of-dialog requests (when the To, From, or Call-ID headers are modified) the original header will be stored in the dialog when the msg-type out-of-dialog is used.

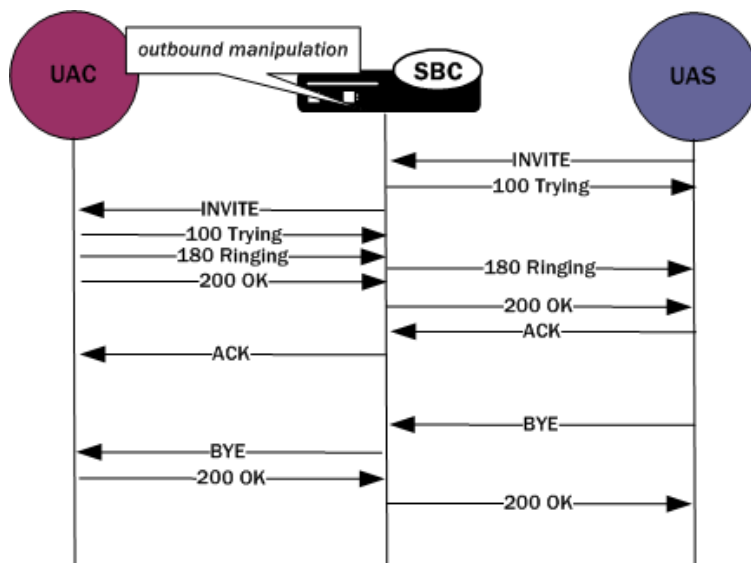
The Oracle CSM performs the restoration of original headers outside of SIP manipulations. That is, there are no manipulation rules to configure for restore the header to their original context. The Oracle CSM will recognize the headers have been modified, and restore them to their original state prior to sending the message out on the wire. Restoration takes place prior to outbound manipulations so that any outbound manipulation can those headers once they have been restored.



Outbound HMR Challenge

When you use the out-of-dialog setting for an outbound manipulation, the Oracle CSM only executes this specific SIP header rule only if the same SIP header rule was executed against the initial dialog-creating request.

For example, if the INVITE's To header was not manipulated, it would not be correct to manipulate the To header in the BYE request. To do so would render the UAC unable to properly match the dialog. And this also means that the outbound manipulation should be carried out against a To, From, or Call-ID header in the BYE request if it was manipulated in the INVITE.



Dialog-matching Header Manipulation Configuration

You using the out-of-dialog setting in the msg-type parameter, part of the SIP header rules configuration.

To enable dialog-matching header manipulation:

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

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

2. Type session-router and press Enter.

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

3. Type sip-manipulation and press Enter.

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

4. Type mime-rules and press Enter. If you are adding this feature to an existing configuration, then remember you must select the configuration you want to edit.

```
ORACLE (sip-manipulation) # header-rules
ORACLE (sip-header-rules) #
```

5. msg-type—Set this parameter to out-of-dialog to enable dialog-matching header manipulation.

6. Save your work.

Built-In SIP Manipulations

In the course of HMR use, certain rules have become commonly used. Lengthy and complex, these rules do not include any customer-specific information and do they can be used widely. To make using them easier, they have been turned into built-in rules that you can reference in the in-manipulationid and out-manipulationid parameters that are part of the realm, session agent, and SIP interfaces configurations.

Built-in rules start with the prefix ACME_, so Oracle recommends you name your own rules in a different manner to avoid conflict.

While the number of built-in manipulation rules is expected to grow, one is supported at the present time: ACME_NAT_TO_FROM_IP. When performed outbound, this rule changes:

- The To-URI hostname to the logical \$TARGET_IP and port to \$TARGET_PORT
- The From-URI to the logical \$REPLY_IP and port to be \$REPLY_PORT

Built-In SIP Manipulation Configuration

When you want to enable this feature for a realm, session agent, or SIP interface, you configure the in-manipulationid or out-manipulationid parameters with the rule.

The sample here shows this feature being applied to a session agent, but the realm and SIP interface configurations also have the same parameter you use to set up the feature.

To use built-in SIP manipulations:

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

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

2. Type session-router and press Enter.

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

3. Type session-agent and press Enter.

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


4. out-manipulationid—Enter name of the built-in rule you want to use. Remember that all built-in rules start with ACME_.
5. Save your work.

Testing SIP Manipulations

You can now use a new tool that allows you to test the outcome of your SIP manipulation and header rules without sending real traffic through the Oracle CSM to see if they work.

To use the tool, you enter the ACLI's test-sip-manipulation utility and reference the rule you want to test using this name. Then you enter a mode where you put in a SIP message entered in ASCII. You can cut and paste this message from sipmsg.log or from some other location. Using <Ctrl-D> stops the SIP message collection and parses it.

The test informs you of any parsing errors found in the SIP message. Once the message is entered, you can execute the SIP manipulation against the message. The output after this step is the modified SIP message after manipulations have been applied. You will also find a debugging option, which displays SIP manipulation logging to the screen as the manipulation takes place.

As a starting point for testing, this tool comes loaded with a default SIP message. It cannot be associated with realms, session agents, or SIP interfaces, and so it also comes with certain reserved words, such as: \$LOCAL_IP, \$TRUNK_GROUP_CONTEXT, and \$REMOTE_PORT. In addition, you can use your settings for testing across terminal sessions; if you choose to save your settings, everything (including the SIP message) will be saved, with the exception of the debugging option.

It is not recommended that you use this tool to add an ISUP message body.

HMR Import-Export

Due to the complexity of SIP manipulations rules and the deep understanding of system syntax they require, it is often difficult to configure reliable rules. This feature provides support for importing and exporting pieces of SIP manipulation configuration in a reliable way so that they can be reused.

Exporting

The SIP manipulation configuration contains an export command. When you use it, the Oracle CSM sends the configuration you have selected to a designated file. The contents are the same information you see when you use the ACLI show command in XML format; it includes the selected configuration and any changes that have been made. Because you can only export one SIP manipulation configuration at a time, you must export each one-by-one if you need more than one.

The file name can be any you selected, and would be most useful if it were to identify its contents in some way. If the file already exists, then the export fails and informs you the file already exists. A successfully-executed export simply returns you to the system prompt.

The system writes exported files to /code/imports, a new location that will be created to avoid overlap with existing backup files. The files will carry the extension .gz to show that they have been compressed with gzip.

Your export data will look like this sample:

```
<?xml version='1.0' standalone='yes'?>
<sipManipulation
  name='manip'
  description=''
  lastModifiedBy='admin@console'
  lastModifiedDate='2009-10-16 14:16:29'>
  <headerRule
    headerName='Foo'
    msgType='any'
    name='headerRule'
    action='manipulate'
    cmpType='boolean'
```

```
        matchValue=' $REGEX (" [bB] [A-Za-z] {2} ") '  
        newValue='foo'  
        methods='INVITE'>  
    </headerRule>  
</sipManipulation>
```

To avoid conflict with other objects on the system, key and object ID are not included as part of the exported XML.

Importing

Using the import command in the SIP manipulation configuration, you can import data from an exported file to a currently-selected configuration. If you have not selected a configuration into which to load the data, a new one will be created. Including the .gz extension, you enter the full name of the file you want imported. After it finds the file, the Oracle CSM unarchives it and parses its contents. If these steps fail, the Oracle CSM will alert you. If they succeed, then the configuration data loads into the object.

If you have been making changes to the configuration into which data was imported, the Oracle CSM will inform you prior to importing the data so that you will not lose any of your work. This way, you will be less likely to overwrite unsaved changes.

Once the import is complete, it will be as if you entered the configuration by hand. You only need to save your work (by typing done) to save the SIP manipulation to the global SIP configuration. Note that if for some reason the XML is malformed or contained more than one object, the import will fail.

If you attempt to import a configuration with the same key as one that already exists, the system returns an error and prevents you from saving the imported object. In this case, you can delete the object with the same key and then carry out your import, or you can select the object with the same key and perform an import that will overwrite it with new data.

Displaying Imports

You can display imported SIP manipulations data at the system prompt. The command lists all files in the exported files directory, and also tells you if there are none.

Using FTP to Move Files

You can also place exported SIP manipulation configuration files on the Oracle CSM using FTP. You need to use the same /code/imports directory to do so.

Removing Files

Using the delete-import command with the name of the file you want to delete removes it from the system. Using this command, you can delete files that are no longer useful to you. Carrying out this command is final and there is no warning before you go ahead with the deletion. A failed deletion (for instance, because there is no such file) will produce an error message; a successful deletion simply returns you to the system prompt.

Unique HMR Regex Patterns and Other Changes

In addition to the HMR support it offers, the Oracle CSM can now be provisioned with unique regex patterns for each logical remote entity. This supplement to pre-existing HMR functionality saves you provisioning time and saves Oracle CSM resources in instances when it was previously necessary to define a unique SIP manipulation per PBX for a small number of customer-specific rules.

Manipulation Pattern Per Remote Entity

On the Oracle CSM, you can configure logical remote entities (session agents, realms, and SIP interfaces) with a manipulation pattern string that the system uses as a regular expression. Then the SIP manipulation references this regular expression using the reserved word \$MANIP_PATTERN. At runtime, the Oracle CSM looks for the logical

entity configured with a manipulation pattern string in this order of preference: session agent, realm, and finally SIP interface.

On finding the logical entity configured with the manipulation string, the Oracle CSM dynamically determines the expression. When there is an invalid reference to a manipulation pattern, the pattern-rule expression that results will turn out to be the default expression (which is \,+).

When the \$MANIP_PATTERN is used in a manipulation rule's new-value parameter, it resolves to an empty string, equivalent of no value. Even though this process ends with no value, it still consumes system resources. And so Oracle recommends you do not use \$MANIP_PATTERN as a new-value value.

In the following example, the SIP manipulation references the regular expression from a realm configuration:

```
realm-config
  identifier                net200
  description
  addr-prefix              0.0.0.0
  network-interfaces      public:0
  ...
  manipulation-pattern     Lorem(.+)
sip-manipulation
  name                    manip
  description
  header-rules
    name                  headerRule
    header-name          Subject
    action                manipulate
    match-value          $MANIP_PATTERN
    msg-type              request
    comparison-type      pattern-rule
    new-value             Math
    methods               INVITE
```

Reject Action

When you use this action type and a condition matching the manipulation rule arises, the Oracle CSM rejects the request (though does not drop responses) and increments a counter.

- If the msg-type parameter is set to any and the message is a response, the Oracle CSM increments a counter to show the intention to reject the message—but the message will continue to be processed.
- If the msg-type parameter is set to any and the message is a request, the Oracle CSM performs the rejection and increments the counter.

The new-value parameter is designed to supply the status code and reason phrase corresponding to the reject. You can use the following syntax to supply this information: status-code[:reason-phrase]. You do not have to supply the status code and reason phrase information; by default, the system uses 400:Bad Request.

If you do supply this information, then the status code must be a positive integer between 300 and 699. The Oracle CSM then provides the reason phrase corresponding to the status code. And if there is no reason phrase, the system uses the one for the applicable reason class.

You can also customize a reason phrase. To do so, you enter the status code followed by a colon (:), being sure to enclose the entire entry in quotation marks (") if your reason code includes spaces.

When the Oracle CSM performs the reject action, the current SIP manipulation stops processing and does not act on any of the rules following the reject rule. This course of action is true for nested SIP manipulations that might have been constructed using the sip-manip action type.

Reject Action Configuration

To support the reject action, two parameters in the session-router-config allow you to set how many messages in a certain amount of time cause the Oracle CSM to generate an SNMP trap.

To set the reject message number and time window:

SIP Signaling Services

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

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

2. Type session-router and press Enter.

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

3. Type session-router and press Enter.

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

4. reject-message-threshold—Enter the minimum number of message rejections allowed in the reject-message-window time on the Oracle CSM (when using the SIP manipulation action reject) before generating an SNMP trap. The default is 0, meaning this feature is disabled and no trap will be sent.
5. reject-message-window—Enter the time in seconds that defines the window for maximum message rejections allowed before generating an SNMP trap.
6. Save your work.

About Counters

The Oracle CSM tracks messages that have been flagged for rejection using the reject action type. In the show sipd display, refer to the Rejected Messages category; there is no distinction between requests and responses.

```
ORACLE# show sipd
13:59:07-102
SIP Status
Active      -- Period -- ----- Lifetime -----
High Total Total PerMax High
Sessions    0      0      0      0      0      0
Subscriptions 0      0      0      0      0      0
Dialogs     0      0      0      0      0      0
CallID Map  0      0      0      0      0      0
Rejections  -      -      0      0      0
ReINVITES  -      -      0      0      0
Media Sessions 0      0      0      0      0      0
Media Pending 0      0      0      0      0      0
Client Trans 0      0      0      0      0      0
Server Trans 0      0      0      0      0      0
Resp Contexts 0      0      0      0      0      0
Saved Contexts 0      0      0      0      0      0
Sockets     0      0      0      0      0      0
Req Dropped -      -      0      0      0
DNS Trans   0      0      0      0      0      0
DNS Sockets 0      0      0      0      0      0
DNS Results 0      0      0      0      0      0
Rejected Msgs 0      0      0      0      0      0
Session Rate = 0.0
Load Rate = 0.0
Remaining Connections = 20000 (max 20000)
```

SNMP Support

The Oracle CSM provides SNMP support for the Rejected Messages data, so you can access this information externally. The new MIB objects are:

```
apSysRejectedMessages OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of messages rejected by the SD due to matching
criteria."
 ::= { apSysMgmtMIBGeneralObjects 18 }
```

```

apSysMgmtRejectedMessagesThresholdExceededTrap NOTIFICATION-TYPE
    OBJECTS          { apSysRejectedMessages }
    STATUS            current
    DESCRIPTION
        " The trap will be generated when the number of rejected messages
exceed the configured threshold within the configured window."
        ::= { apSystemManagementMonitors 57 }
apSysMgmtRejectedMessagesGroup OBJECT-GROUP
    OBJECTS {
        apSysRejectedMessages
    }
    STATUS            current
    DESCRIPTION
        "Objects to track the number of messages rejected by the SD."
        ::= { apSystemManagementGroups 18 }
apSysMgmtRejectedMessagesNotificationsGroup NOTIFICATION-GROUP
    NOTIFICATIONS {
        apSysMgmtRejectedMesagesThresholdExceededTrap
    }
    STATUS            current
    DESCRIPTION
        "Traps used for notification of rejected messages"
        ::= { apSystemManagementNotificationsGroups 26 }
apSgmtRejectedMessagesCap
    AGENT-CAPABILITIES
    PRODUCT-RELEASE "Acme Packet SD"
    STATUS            current
    DESCRIPTION      "Acme Packet Agent Capability for enterprise
system management MIB."
    SUPPORTS         APSYSGMT-MIB
    INCLUDES         {
        apSysMgmtRejectedMessagesGroup,
        apSysMgmtRejectedMessagesNotificationsGroup
    }
    ::= { apSgmtMibCapabilities 37 }

```

Log Action

When you use this action type and a condition matching the manipulation rule arises, the Oracle CSM logs information about the current message to a separate log file. This log files will be located on the same core in which the SIP manipulation occurred. On the core where sipt runs, a logfile called matched.log will appear when this action type is executed.

The matched.log file contains a timestamp, received and sent Oracle CSM network interface, sent or received IP address:port information, and the peer IP address:port information. It also specifies the rule that triggered the log action in this syntax: rule-type[rule:name]. The request URI, Contact header, To Header, and From header are also present.

```

-----
Apr 17 14:17:54.526 On [0:0]192.168.1.84:5060 sent to 192.168.1.60:5060
element-rule[checkRURIPort]
INVITE sip:service@192.168.1.84:5060 SIP/2.0
From: sipp <sip:+2125551212@192.168.1.60:5060>;tag=3035SIPpTag001
To: sut <sip:service@192.168.1.84>
Contact: sip:sipp@192.168.1.60:5060

```

Changes to Storing Pattern Rule Values

Release S-C6.2.0 introduces changes to the framework for storing regular expression results within manipulation rules, altering the way the store action works. These changes are beneficial to performance.

In previous releases, when the store action is used, the Oracle CSM stores all values matching the regular expression defined in the match-value parameter for all headers. At runtime, the system evaluates all stored values to find the correct index.

Now, you no longer need to specify the store action. The simple fact of referencing another rule tells the system it must store a value. When SIP manipulation is used, the system first checks to see if any values require storing. The add action is an exception to this process; storing happens after a header is added.

When referring to a rule, that rule still needs to have a regular expression defined in the match-value and the comparison type set to pattern-rule; else the default expression will be used.

Removal of Restrictions

The following restrictions related to HMR have been removed in Release S-C6.2.0:

- The action find-replace-all now executes all element rules. Previously, no child rules were executed.
- The action sip-manip now executes existing all element rules. Previously, no child rules were executed.
- The action store now executes existing all element rules. Previously, only child rules with the store action were executed.
- The action add now executes existing all element rules. Previously, only child rules with the add action were executed.

Name Restrictions for Manipulation Rules

Historically, you have been allowed to configure any value for the name parameter within a manipulation rule. This method of naming caused confusion when referencing rules, so now manipulation rules name must follow a specific syntax. They must match the expression `^[[:alpha:]]+[[:alnum:]]+$` and contain at least one lower case letter.

In other words, the name must:

- Start with a letter, and then it can contain any number of letters, numbers, or underscores
- Contain at least one lower case letter

All pre-existing configurations will continue to function normally. If you want to change a manipulation rule, however, you are required to change its name if it does not follow the new format.

The ACLI verify-config command warns you if the system has loaded a configuration containing illegal naming syntax.

Please note that the software allows you to make changes to HMRs, including configuring new functionality to existing rules, as long as you do not change the rule name. This results in an important consideration surrounding HMRs with hyphens in previously configured rule names.

- You can reference stored values in new value names. (Recall that stored values may be rule names.)
- You can perform subtraction in new value names.

If you use a rule names with hyphens within the REGEX of new value names, the system cannot determine whether the hyphen is part of the rule name or is intended to invoke subtraction within the REGEX. For this reason, you need to use great care with legacy HMR naming that includes hyphens.

As a general rule, create new rule names that follow the new rule naming guidelines if you intend to use new functionality in those rules.

New Value Restrictions

To simplify configuration and remove possible ambiguity, the use of boolean and equality operators (`==`, `<=`, `<`, etc.) for new-value parameter values has been banned. Since there was no specific functionality tied to their use, their ceasing to be use will have no impact to normal SIP manipulation operations.

Dialog Transparency

This section explains how to configure dialog transparency, which prevents the Oracle CSM from generating a unique Call-ID and modifying dialog tags.

Overview

With dialog transparency enabled, the Oracle CSM is prevented from generating a unique Call-ID and from modifying the dialog tags; the Oracle CSM passes what it receives. Therefore, when a call made on one Oracle CSM is transferred to another UA and crosses a second Oracle CSM, the second Oracle CSM does not note the context of the original dialog, and the original call identifiers are preserved end to end. The signalling presented to each endpoint remains in the appropriate context regardless of how many times a call crosses through a Oracle CSM or how many Oracle CSMs a call crosses.

Without dialog transparency enabled, the Oracle CSM's SIP B2BUA rewrites the Call-ID header and inserted dialog cookies into the From and To tags of all messages it processes. These dialog cookies are in the following format: SDxxxxxNN-. Using these cookies, the Oracle CSM can recognize the direction of a dialog. However, this behavior makes call transfers problematic because one Oracle CSMs' Call-ID might not be properly decoded by another Oracle CSM. The result is asymmetric header manipulation and failed call transfers.

Dialog Transparency Configuration

You set one parameter in your SIP configuration to enable dialog transparency.

- For new configurations, this feature defaults to enabled

To enable SIP dialog transparency:

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
```

4. Use the ACLI select command so that you can work with the SIP configuration.

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

5. dialog-transparency—Enter the state of SIP dialog transparency you require for your Oracle CSM. The default value is enabled. The valid values are:

- enabled | disabled

Route Header Removal

This section explains how to enable the Oracle CSM to disregard and strip all SIP Route headers. You set an option in a SIP interface configuration to strip all Route headers for SIP requests coming from this interface.

When the Oracle CSM with this option configured receives an INVITE from an interface, it removes the route headers. However, although it removes the headers, the Oracle CSM maintains backward compatibility with RFC 2543 nodes. To do so, it normalizes the request to an RFC 3261 loose routing form before it removes the headers.

Route Header Removal Configuration

The following information explains how to remove SIP route headers.

To configure SIP route header removal:

SIP Signaling Services

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

```
ORACLE# configure terminal
```

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

```
ORACLE(configure)# session-router  
ORACLE(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 options strip-route-headers and press Enter. This completes the configuration of SIP route header removal.

```
ORACLE(sip-interface)# options strip-route-headers
```

SIP Via Transparency

This section explains the inbound Via header transparency feature, which enables the Oracle CSM to insert its Via header on top of the top-most Via header received from user equipment (UE). It then forwards it on to the IP Multimedia Subsystem (IMS) core with the original Via header now located as the bottom-most Via header.

The Oracle CSM still replaces the Contact and other header addresses with its own, and does not pass on the core's Via headers in outbound requests.

This feature is targeted for the Telecoms & Internet converged Services & Protocols for Advanced Networks (TISpan) with SIP hosted NAT traversal support. It works with SIP NAT bridged, local-policy routed, and non-SIP NAT configurations, regardless of registration handling.

Some equipment acts as Proxy-CSCF (P-CSCF) and Serving-CSCF (S-CSCF) nodes, with the Oracle CSM is located between the equipment and user endpoints. The equipment needs to see the each user endpoint's original Via header in order to perform some implicit authentication, admission, and control functions in a TISpan-compliant model.

You enable Via header transparency on the access SIP interface. Received Via headers are saved for inclusion in requests going out another interface or session agent that does not have the parameter set, in other words, the core side. For any received SIP message where the inbound previous hop interface was enabled for Via header transparency, the Oracle CSM adds its own Via header as it forwards it, and it also copies the received top-most Via as the new bottom-most Via, if the outbound next hop interface/session agent is not enabled for Via header transparency. The Oracle CSM also adds a received= parameter to the copied Via header, per the SIP RFC 3261.

Any message received from an interface without Via header transparency enabled, does not have the received Via header copied over to any other direction.

For HNT, where the original top-most (and only) Via header from a UE is a private/false address, the SD should still copy that false address into the core-side, and the received= parameter will contain the real Layer-3 addressing.

SIP Via Transparency Configuration

You can configure SIP Via header transparency for the access SIP interface using the ACLI.

To configure SIP Via header transparency for an access interface:

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

```
ORACLE# configure terminal
```

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

```
ORACLE(configure)# session-router  
ORACLE(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. You can either add support to a new SIP interface configuration or to an existing SIP interface configuration:

For a new SIP interface configuration, you can add the option by typing options, a Space, and then via-header-transparency.

```
ORACLE(sip-interface) # options via-header-transparency
```

For an existing SIP interface configuration without options configured, select the SIP interface, type options followed by a Space, and then via-header-transparency.

```
ORACLE(sip-interface) # select
ORACLE(sip-interface) # options via-header-transparency
```

For an existing SIP interface configuration with options configured, select the SIP interface, type options followed by a Space, the plus sign (+), and the via-header-transparency option.

```
ORACLE(sip-interface) # select
ORACLE(sip-interface) # options +via-header-transparency
```

5. Save your work using the ACLI save or done command.

SIP Number Normalization

This section explains the SIP number normalization feature that applies to the SIP To URI. (Currently the Oracle CSM supports number normalization on From and To addresses for both inbound and outbound call legs.) Number normalization includes add, delete, and replace string functions that result in consistent number formats.

Number normalization applies to the SIP To URI. It occurs on ingress traffic, prior to the generation of accounting records or local policy lookups. RADIUS CDR attributes are populated with the normalized numbers. Local policy matching is based on the normalized numbers.

Terminology

The following lists explains the terminology used later.

- X is any digit having the value 0 through 9
- N is any digit having the value 2 through 9
- 0/1 is a digit having the value of either 0 or 1
- NXX is a form of Numbering Plan Area (NPA).
- CC is a 1, 2, or 3 digit country code used in international dialing
- NN is a national number that can be a four to fourteen digit national number used in international dialing, where the combination of CC+NN is a 7 to 15 digit number.
- + symbol in E.164 indicates that an international prefix is required
- E.164 numbers are globally unique, language independent identifiers for resources on Public Telecommunication Networks that can support many different services and protocols.
- N11 number is any of the three-digit dialing codes in the form N11 used to connect users to special services, where N is a digit between 2 and 9

Calls from IP Endpoints

The Oracle CSM uses the following number normalization rules:

- North American Numbering Plan (NANP) calls: where a number with the format 1NPANXXXXXXX is received, the Oracle CSM adds a plus sign (+) as a prefix to the NANP number. The Oracle CSM also adds the string ;user=phone after the host IP address in the SIP URI. For example:

```
sip:+1NPANXXXXXXX@ipaddr;user=phone
```

SIP Signaling Services

- International NWZ1 calls: Oracle CSM receives an international call with the format 011CCNN. The Oracle CSM deletes the 011 prefix and adds a plus sign (+) as a prefix to CC+NN; and also adds the string ;user=phone after the host IP address in the SIP URI. For example:

```
sip:+CCNN@ipaddr;user=phone
```

- Private number calls: when a private number with the format nxxxx (where n=2 through 9) is received, no number normalization is applied by the Oracle CSM.
- Calls to numbers such as N11, 0-, 0+, 00-, and 01+: the Oracle CSM adds ;phone-context=+1 after the number and also adds the string ;user=phone after the host IP address in the SIP URI. For example:

```
sip:N11;phone-context=+1@ipaddr;user=phone  
sip:01CCNN;phone-context=+1@ipaddr;user=phone
```

- Calls with numbers that are already normalized are not modified by the Oracle CSM.

Calls from IP Peer Network

For calls received from external peer networks, the Oracle CSM uses the following number normalization rules:

- Global numbers such as NANP and international E.164 numbers should have already been normalized. If not, the Oracle CSM applies the same number normalization rules listed in the prior section.
- Calls to numbers such as N11, 0-, 0+, 00-, and 01+: the Oracle CSM adds ;phone-context=+1 after the number and also adds the string ;user=phone (if absent) after the host IP address in the SIP URI.

SIP Number Normalization Configuration

You can configure SIP number normalization for the realm and session agent using the ACLI.

Realm

To configure SIP number normalization for a realm:

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

```
ORACLE# configure terminal
```

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

```
ORACLE(configure)# media-manager  
ORACLE(media-manager)#
```

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

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

4. You can either add SIP number normalization support to a new session agent configuration or to an existing session agent configuration:

- For a new realm configuration, add the option by typing options, a Space, and then number-normalization.

```
ORACLE(realm-config)# options number-normalization
```

- For an existing realm configuration without any options already configured, select the realm, type options followed by a Space, and then number-normalization.

```
ORACLE(realm-config)# select  
ORACLE(realm-config)# options number-normalization
```

- For an existing realm configuration with other options, select the realm, type options followed by a Space, the plus sign (+), and the number-normalization option.

```
ORACLE(realm-config)# select  
ORACLE(realm-config)# options +number-normalization
```

5. Save your work using the ACLI save or done command.

Session Agent

To configure SIP number normalization for a session agent:

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

```
ORACLE# configure terminal
```

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

```
ORACLE(configure)# session-router
ORACLE(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. You can either add SIP number normalization support to a new session agent configuration or to an existing session agent configuration:

- For a new a session agent configuration, add the option by typing options, a Space, and then number-normalization.

```
ORACLE(session-agent)# options number-normalization
```

- For an existing session agent configuration without any options already configured, select the session agent, type options followed by a Space, and then number-normalization.

```
ORACLE(session-agent)# select
ORACLE(session-agent)# options number-normalization
```

- For an existing session agent configuration with other options, select the session agent, type options followed by a Space, the plus sign (+), and the number-normalization option.

```
ORACLE(session-agent)# select
ORACLE(session-agent)# options +number-normalization
```

5. Save your work using the CLI save or done command.

SIP Configurable Route Recursion

When the Oracle CSM routes SIP requests from a UAC to a UAS, it might determine that there are multiple routes to try based on a matching local policy. The Oracle CSM recurses through the list of routes in a specific order according to your configuration and the quality of the match. There are other scenarios when a UAS replies with a 3xx Redirect response to the Oracle CSM, the 3xx response can include multiple Contacts to which the request should be forwarded in a specific order. In both cases, the Oracle CSM needs to recurse through a list of targets.

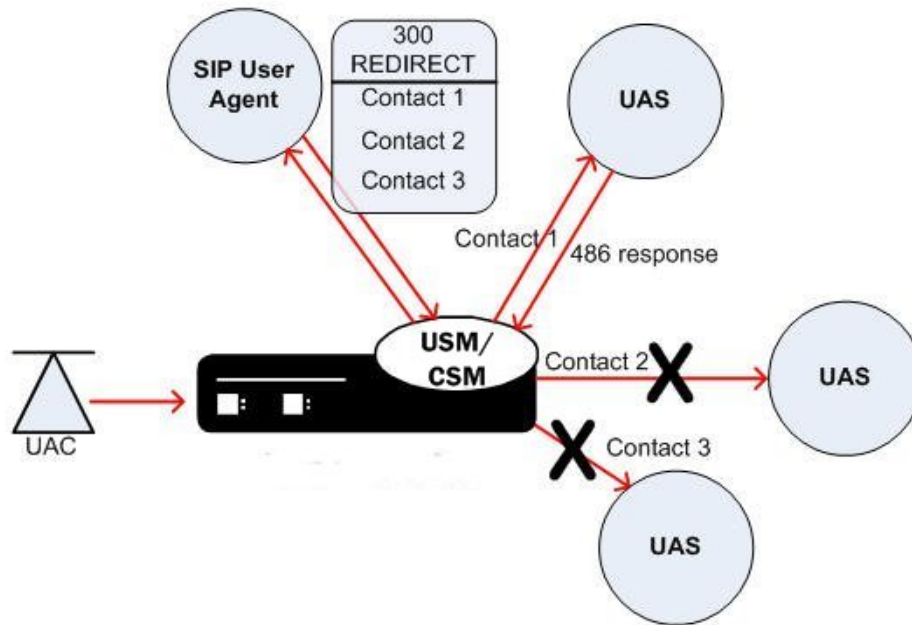
When the Oracle CSM receives a non-successful (or non-6xx response) final response from the UAS, and there are multiple targets for the original request, the Oracle CSM will forward the request to the next target and wait for a response. While the process of forwarding the request to multiple targets as explained in the previous paragraph is called serial forking, and the process of forwarding the request to contacts received in redirect responses is called recursion, the term recursion is used for both processes in this notice.

Use the SIP Route Recursion feature when you want the Oracle CSM to forward a response to the UAC and stop recursing through the target list immediately after receiving the 3xx, 4xx, or 5xx response code that you configure. When this feature is disabled, the Oracle CSM only stops recursing when it receives a message with a 401 or 407 response code. Using this feature, you can configure a specific message or range of messages to stop recursing on when received. The Oracle CSM retains its default behavior to stop recursing on a 401 or 407 response code when SIP Route Recursion is configured on a SIP interface. The Oracle CSM will always stop recursing when it receives a global failure (6xx); this behavior is not configurable.

You can disable response recursion for either a SIP interface or for a SIP session agent, providing you with flexibility for various network architectures. For instance, a PSTN gateway might be the only hop to reach a given endpoint, whereas several session agents might need to be contacted if multiple devices map to a contacted address of record.

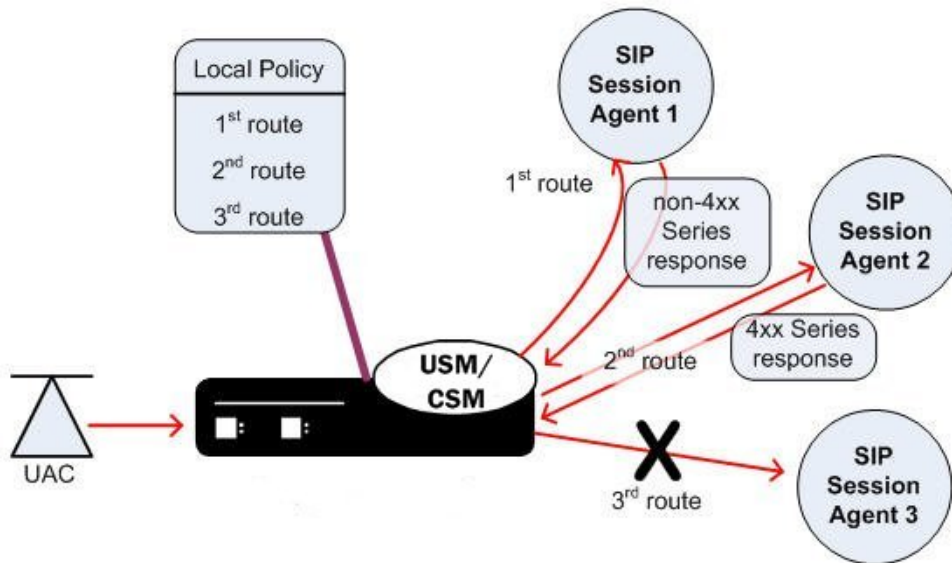
Example 1

A more detailed example is when a softswitch might return a list of contacts for multiple PSTN gateways in a Redirect message. If the PSTN target number contacted on redirection is busy, a 486 response will be sent to the Oracle CSM. Since the single target is located in the PSTN, a subsequent request through a different gateway will yield another 486 response. The Oracle CSM should be configured to return the 486 response to the UAC immediately. No other SIP requests should be sent to applicable targets/contacts that were enumerated in the redirect list. See the following example:



Example 2

The Oracle CSM might determine from a local policy lookup that several routes are applicable for forwarding a SIP message. The Oracle CSM will try each route in turn, but the SIP response recursion disable feature can be implemented to stop the route recursion when a configured responses message is received by the Oracle CSM. See the following example:



There are a few conditions on the parameter used to configure response recursion:

- SIP Route Recursion is configurable for either the SIP interface or session agent.
- 401 and 407 are preconfigured for all configured SIP interfaces. They are not configured for session agents.
- The format is a comma-separated list of response codes or response code ranges: 404, 484-486.
- Only response codes that fall within the 3xx, 4xx, and 5xx range may be specified.

SIP Route Recursion Configuration

You enable SIP route recursion either in the session agent or the SIP interface configuration.

Configuring a Session Agent for SIP Route Recursion

To configure SIP Route recursion for an existing session agent:

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. Select the session agent where you want this feature.

```
ORACLE(session-agent)# select
<hostname>:
1: asd          realm=      ip=1.0.0.0
2: SIPSA       realm=      ip=10.10.102.1
selection:2
ORACLE(session-agent)#
```

5. `stop-recurse`—Enter list of returned response codes that this session agent will watch for in order to stop recursion on the target's or contact's messages. This can be a comma-separated list or response code ranges.

```
ORACLE(session-agent)# stop-recurse 404,484-486
```

6. Save and activate your changes.

Configuring a SIP Interface for SIP Route Recursion

To configure SIP route recursion for an existing 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. Select the SIP interface to which you want to apply this feature.

```
ORACLE(sip-interface)# select
<realm-id>:
1: Acme_Realm
selection:1
ORACLE(sip-interface)#
```

5. stop-recurse—Enter a list of returned response codes that this SIP interface will watch for in order to stop recursion on the target's or contact's messages. This list can be a comma-separated list of response codes or response code ranges.

```
ORACLE(sip-interface)# stop-recurse 404,484-486
```

6. Save and activate your changes.

SIP Proxy Subscriptions

When the Oracle CSM operates in dialog mode (i.e., as a B2BUA), it creates and maintains dialog state for subscription dialogs created with SUBSCRIBE/NOTIFY messages and for INVITE-initiated dialogs. Since there can be a very large number of subscriptions per user in a Rich Communication Services (RCS) environment (especially for presence subscriptions), Oracle CSM resources can quickly become depleted.

To alleviate this consumption of resources, you can configure your Oracle CSM to operate in proxy mode for event packages that you define using the proxy-sub-event parameter in the global SIP configuration. When you define event packages in this list and the operation mode for the SIP configuration is dialog or session, the Oracle CSM processes all SUBSCRIBE and NOTIFY requests and responses for the designated event packages in transaction stateful mode.

Topology Hiding

So that it can perform topology hiding, the Oracle CSM retains necessary routing information (such as the Contact or Record-Route header values) and it encodes certain data from the messages it receives in the messages it sends. To be more specific, the Oracle CSM encodes the original URI hostport and the ingress realm name into a gr parameter it adds to the URI. The hostport information is replaced with the IP address and port of the SIP interface from which the message is sent (the egress interface). Without this information, subsequent in-dialog messages cannot be routed correctly because the Oracle CSM does not retain dialog state (i.e., the route-set or remote-target).

For example, the URI sip:td@192.168.24.121:5060 might be encoded as sip:td@192.168.24.121:5060;gr=vjml9qtd175bhmhvhkgp0jov81popvbp000040.

The Oracle CSM also performs URI encoding on the message body for Content-Type application/pdf+xml. This contains a Presence Information Data Format document (or PIDF) in PUBLISH and NOTIFY requests so subsequent SIP requests using the URIs in the PIDF document can be routed correctly.

The Oracle CSM performs URI encoding in outgoing SIP messages after SIP=NAT is applied and before outbound HMR occurs. And the system decodes URIs in SIP messages after inbound HMR takes place and before SIP-NAT is applied.


In the event a URI is encoded several times (as is the case in spiral and hairpin calls), the encoded realm+hostport values are separated by a plus sign (+), as in the following:

```
sip:td@192.168.24.121:5060; gr=vjml9qtd175bhmhvhkqp+dhfhhb0jov81opvbp
```

When the Oracle CSM receives any of the following requests, it matches the contents of the request's Event header with the list you configure in the proxy-sub-events parameter:

- PUBLISH
- SUBSCRIBE
- NOTIFY
- REFER

This is provided the operation-mode for the SIP configuration is set to either session or dialog. If it finds a match, the Oracle CSM marks the request for processing in transaction-stateful mode rather than in B2BUA mode.

 **Note:** Although PUBLISH is not a dialog-creating request, topology hiding needs to be applied to the PIDF so that subsequent NOTIFY requests containing portions of the published PIDF can be decoded properly.

When the Oracle CSM forwards the request, it will have encoded all Contact and Record-Route header information using the ingress realm. The hostport value of the URIs then has egress SIP interface's IP address and port. The Via headers in the requested the Oracle CSM received are not included in the outgoing request.


Then PUBLISH, SUBSCRIBE, NOTIFY, and REFER responses are compared to the request that was sent to determine if the response should receive transaction-stateful proxy treatment. The Oracle CSM decodes any encoded Record-Route headers back to their original values for the outgoing response. Any Record-Route headers added downstream from the Oracle CSM are encoded using the original request's egress realm (meaning the realm from which the response was received). In addition, the Contact header is encoded using the request's egress realm and ingress SIP interface and port.

Feature Interaction

When using this feature, the Oracle CSM does not keep dialog or subscription state. Therefore the Per-user SUBSCRIBE Dialog Limit feature—configured in the enforcement-profile configuration—will not function properly when a subscription is handled in proxy mode.

SIP Proxy Subscription Configuration

This section shows you how to configure a list of SIP event packages to cause the Oracle CSM to act in proxy mode.

 **Note:** The operation-mode parameter for the global SIP configuration must be set to either dialog or session in order for this feature to function as designed.

To configure a list of SIP event packages to enable SIP proxy subscriptions:

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

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

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

3. Type sip-config and press Enter.

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

If you are adding support for this feature to a pre-existing configuration, then you must select (using the ACLI select command) the configuration that you want to edit.

4. proxy-sub-events—Enter a list of SIP event package names that you want to enable the SIP proxy subscriptions feature. You can enter more than one value by enclosing multiple values in quotations marks, as in the following example.

```
ORACLE(sip-config)# proxy-sub-events presence winfo
```

SIP Local Response Code Mapping

The SIP local response code mapping feature has been added as an enhancement to the SIP response code mapping. The SIP response code map feature lets you establish a table that maps SIP response-received messages (entries) to response-to-send messages (entries).

SIP local response code mapping is used with the SIP responses generated by the Oracle CSM towards a specific SIP session agent. This feature lets you provision the mapping of the response codes used by the Oracle CSM when it generates the responses towards a session agent.

You create the SIP local response code map using the existing mapping functionality, and then assigning that map to a session agent or to a SIP interface.



Note: The configured response map is not used when the Oracle CSM is acting as proxy for the responses to this session agent.

SIP Local Response Code Mapping Configuration

The following instructions explain how to create the SIP response code map and then how to assign it to a specific session agent.

Creating a SIP Response Code Map

To create a SIP local response code map:

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

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

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

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

```
ORACLE(session-router)# sip-response-map  
ORACLE(response-map)#
```

4. name—Enter the name of the SIP response map you want to configure. This value is required and must be unique.

```
ORACLE(response-map)# name busy
```

5. entries—To configure the entries for this mapping, type entries and then press Enter. Typing a question mark will show you the response code entry parameters that you can configure.

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

recv-code—Enter original SIP response code for the recv-mode parameter. The valid range is:

- Minimum—100
- Maximum—699

```
ORACLE(response-map-entries)# recv-mode 486
```

xmit-code—Enter the SIP response code into which you want the original response code to be translated. This valid range is:

- Minimum—100
- Maximum—699

```
ORACLE(response-map-entries)# xmit-mode 600
```

reason—Enter a reason for the translated code into the reason parameter. This response comment is sent with the translated code. Make your entry in quotation marks.

```
ORACLE(response-map-entries)# reason "Busy Everywhere"
```


The following two parameters (method and register-response-expires) enable a SIP registration response mapping feature that allows you to configure the Oracle CSM to remap a SIP failure response—which it receives from another network device or that it generates locally—to a 200 OK. You might want the Oracle CSM to perform this type of mapping for circumstances where non-malicious endpoints continually attempt registration, but will stop (and still not be registered) when they receive a 200 OK. This response mapping does not actually register the client with the Oracle CSM, meaning that there is neither a registration cache entry or a CAM ACL for it.

For the 200 OK it generates, the Oracle CSM removes any Reason or Retry-After header in the 200 OK and sets the expires time. By default, the expires time is the Retry-After time (if there is one in the response) or the expires value in the Register request (if there is no Retry-After expires time). You can also set this value using the register-response-expires parameter, but the value you set should never exceed the Register request's expires time.

method—Enter the name of the received SIP failure response message you want to map to a 200 OK. There is no default for this parameter, and leaving the parameter empty turns off the SIP registration response mapping feature.

register-response-expires—Enter the time you want to use for the expires time what mapping the SIP method you identified in the method parameter from Step 4. The maximum is 999999999. By default, the expires time is the Retry-After time (if there is one in the response) or the expires value in the Register request (if there is no Retry-After expires time). Any value you configure in this parameter (when not using the defaults) should never exceed the Register request's expires time.

6. Note the name that you gave the SIP response code map so that you can use it when you configure a session agent to support SIP response code mapping.
7. Save and activate your changes.

Assigning SIP Response Code Maps to Session Agents

To assign a SIP local response code map to a session agent:

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

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

```
ORACLE(configure)# session-router
ORACLE(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. local-response-map—Enter the name of the configured SIP response map that you want to use for this session-agent and press Enter.

```
ORACLE(session-agent)# local-response-map busy
```

5. Save and activate your configuration.

Assigning SIP Response Code Maps to SIP Interfaces

To apply SIP response codes maps to a 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 signaling-level configuration elements.

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

3. Type sip-interface and press Enter.

SIP Signaling Services

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

4. **local-response-map**—Enter the name of the configured SIP response map that you want to apply to this SIP interface for locally-generated SIP responses. This parameter is blank by default.
5. Save and activate your configuration.

Session Agent Ping Message Formatting

You can configure the user portions of the Request-URI and To: headers that define the destination of a session agent ping message, and the From: header that defines the source of a session agent ping message. These headers are sent to Oracle CSM session agent. This feature is required for interoperability with certain E911 servers.

In the following example of a session agent ping-type message, you can set the user portion of the Request-URI (the text bob in the OPTIONS method line) and the user portion of the From: header (the text bob in the From: header) to the same new value. You can also set the user portion of the To: header (the text anna in the To: header) to its own new value.

```
OPTIONS sip:bob@sip.com SIP/2.0
From: UA1 <sip:bob@sip.com>
To: NUT <sip:anna@gw.sip.com>
Call-ID: 123abc@desk.sip.com
CSeq: 1 OPTIONS
Contact: <sip:UA1@client.sip.com>
Accept: application/sdp
Content-Length: 0
```

If you do not enable this feature, then the session agent ping-type message contains the text ping in all cases.

Session Agent Ping Message Formatting Configuration

1. Access the **session-agent** configuration element.

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

2. Select the **session-agent** object to edit.

```
ORACLE(session-agent)# select
<hostname>:
1: 192.168.100.101:1813

selection: 1
ORACLE(session-agent)#
```

3. **ping-from-user-part**—Set the user portion of the From: header that defines the source of a session agent ping message.

```
ORACLE(session-agent)# ping-from-user-part bob
```

4. **ping-to-user-part**—Set the user portions of the Request-URI and To: headers that define the destination of a session agent ping message.

```
ORACLE(session-agent)# ping-to-user-part anna
```

5. Type **done** to save your configuration.

Fraud Prevention

The Oracle CSM can constrain outgoing SIP messages to a maximum size in bytes in order to support fraud prevention techniques. If a message does exceed the configured size, it is dropped. A SIP message can be constrained from 0 to 65535 bytes, with a default value of 4096 bytes.

Fraud Prevention Configuration

To set a maximum SIP message size:

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

```
ORACLE# configure terminal
```

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

```
ORACLE(configure)# session-router
ORACLE(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 configure the existing sip config.

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

5. sip-message-len—Set the size constraint in bytes of a SIP message. The default is 4096. The valid range is:

- Minimum—0
- Maximum—65535

This completes the configuration.

```
ORACLE(sip-config)# sip-message-len 5000
```

6. Save your work using the ACLI done command.

Dynamic Transport Protocol Change

The Oracle CSM also uses the IP address and port in the Contact and Via headers. This is useful for cases when endpoints dynamically change transport protocols (TCP/UDP), and the port number used for sending an INVITE might not be the same one used to send a Register message.

If you do not enable this feature, when an endpoint registered with the Oracle CSM used UDP for its transport protocol, a call fails if that endpoint subsequently initiates the call using TCP. The Oracle CSM checks for the Layer 3 IP address and port, and it rejects the call if the port is changed.

With the new option reg-no-port-match added to the SIP interface configuration, the Oracle CSM will not check the Layer 3 port in the INVITE and REGISTER messages.

Dynamic Transport Protocol Change Configuration

To enable dynamic transport protocol change:

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

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

```
ORACLE(configure)# session-router
ORACLE(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 options +reg-no-port-match and press Enter.

```
ORACLE(sip-interface)# options +reg-no-port-match
```

SIP Signaling Services

If you type options `reg-no-port-match` without the plus (+) sign, you will remove any previously configured options. In order to append the new option to the options list, you must prepend the new option with a plus sign as shown in the example above.

5. Save and activate your configuration.

SIP Privacy Extensions

This section explains how you can configure privacy services to be applied only when the source is trusted and the destination is considered untrusted. (Prior to this release, the Oracle CSM always applied the privacy services, unless the source and the destination were both trusted.)

The Oracle CSM considers all user endpoints and nodes outside the core as untrusted.

The Oracle CSM acts as the boundary device between the trusted platform and the untrusted Internet, to implement privacy requirements. When it receives a message, the Oracle CSM checks whether the source is trusted. It evaluates the level of privacy requested in a Privacy header, if present.

Depending on whether the source is trusted or untrusted, the Oracle CSM can do different things when passing the message to the outgoing side. It also checks whether the destination is trusted.

Privacy Types Supported

The Oracle CSM supports the following Privacy types:

- **user**: user-level privacy function provided. Any non-essential informational headers are removed, including the Subject, Call-Info, Organization, User-Agent, Reply-To, and In-Reply-To. Possibly the original value of the From header is changed to anonymous.
- **header**: headers that cannot be set arbitrarily by the user (Contact/Via) are modified. No unnecessary headers that might reveal personal information about the originator of the request are added. (The values modified must be recoverable when further messages in the dialog need to be routed to the originator.)
- **id**: third-party asserted identity kept private with respect to SIP entities outside the trust domain with which the user authenticated.

The following SIP headers can directly or indirectly reveal identity information about the originator of a message: From, Contact, Reply-To, Via, Call-Info, User-Agent, Organization, Server, Subject, Call-ID, In-Reply-To and Warning.

user

The Oracle CSM supports the Privacy type **user**. It can remove non-essential information headers that reveal user information by:

- Setting the SIP From header and display information to anonymous
- Removing the Privacy header
- Removing Proxy-Require option tag = privacy (if present)
- Removing the following headers:

Subject

Call-Info

Organization

User-Agent

Reply-To

In-Reply-To

header

The Oracle CSM also supports the Privacy type **header**. It modifies SIP headers that might reveal the user identity by:

- Stripping the Via header
- Replacing the Contact header
- Stripping Record-Route
- Removing the Privacy header
- Removing Proxy-Require option tag = privacy (if present)

In general, the B2BUA behavior of the Oracle CSM by default provides header privacy for all sessions.

id

The Oracle CSM also supports the Privacy type id. It keeps the Network Asserted Identity private from SIP entities outside the trusted domain by:

- Stripping only P-Asserted-Identity
- Removing the Privacy header and Proxy-Require option-tag = privacy
- Setting the From header to anonymous (for the backward compatibility)

Examples

The following examples show the actions the Oracle CSM performs depending on the source and target of the calls.

Calls from Untrusted Source to Trusted Target

When calls are from an untrusted source to a trusted target and PPI is included in the INVITE sent to IP border elements, the Oracle CSM maps the PPI information to PAI in the outgoing INVITE to the trusted side (even if the Privacy header is set to id or to none). The Privacy and From headers get passed on unchanged.

IP border elements must pass PAI (if received in the ingress INVITE) and the From and Privacy headers to the egress side just as they were received on the ingress side.

The Oracle CSM maps the PPI to PAI by default, if the outgoing side is trusted. To change this behavior, you need to configure the `disable-ppi-to-pai` option.

Calls from Trusted to Untrusted

When calls are from a trusted source to an untrusted target, and the Privacy header is set to id, the Oracle CSM strips PAI, makes the From header anonymous, and strips the Privacy header.

If the Privacy header is set to none, the Oracle CSM does not change the From header and passes on the Privacy header, if there is one.

Calls from Trusted to Trusted

When calls are going from trusted source to trusted target acting as a peer network border element and PPI is included, the Oracle CSM maps PPI to PAI. The Privacy header remains the same as signaled and the Oracle CSM passes the From header and the PAI without changes.

Configuring SIP Privacy Extensions

Prior to this release the session agent's trust mode provided this functionality. Now you configure SIP interface's trust-mode as none, which means nothing is trusted for this SIP interface.

You also configure the `disable-ppi-to-pai` parameter to disable the changing of the P-Preferred header to the P-Asserted-Identity header, if the outgoing side is trusted.

Trust Mode

To configure the trust mode:

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.

SIP Signaling Services

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

3. Type `session-router` 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)#
```

From this point, you can configure SIP interface parameters. To view all `session-router` parameters, enter a `?` at the system prompt.

4. If configuring an existing interface, enter the select command to select the interface.
5. `trust-mode`—Select the trust mode for this SIP interface. The default value is `all`. The valid values are:
 - `all`—Trust all previous and next hops except untrusted session agents
 - `agents-only`—Trust only trusted session agents
 - `realm-prefix`—Trusted only trusted session agents or address matching realm prefix
 - `registered`—Trust only trusted session agents or registered endpoints
 - `none`—Trust nothing
6. Save and activate your configuration.

The following example shows the `trust-mode` set to `none`. The remaining SIP interface options are omitted for brevity.

```
session-router
state                enabled
realm-id             access1
sip-port
  address            192.168.1.30
  port               5060
  transport-protocol UDP
  allow-anonymous    all
carriers
proxy-mode           Proxy
redirect-action
contact-mode         maddr
nat-traversal        none
nat-interval         30
registration-caching disabled
min-reg-expire       300
registration-interval 3600
route-to-registrar   disabled
teluri-scheme        disabled
uri-fqdn-domain
options
trust-mode           none
```

Disabling the PPI to PAI Change

To disable the changing of PPI to PAI:

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 `session-router` and press Enter. The system prompt changes.

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

From this point, you can configure SIP configuration parameters. To view all `session-router` parameters, enter a `?` at the system prompt.

4. If configuring an existing SIP configuration, enter the select command to select it.

5. options—Enter `disable-ppi-to-pai`. If adding to an existing list of options, use a preceding plus (+) sign.

```
options +disable-ppi-to-pai
```

6. Save and activate your configuration.

SIP Registration Cache Limiting

Using SIP registration cache limiting for SIP endpoint access deployments, you can restrict the size of the SIP registration cache for the global SIP configuration.

You can implement this feature if you have been seeing issues where, either due to network failure scenarios or incorrect sizing of system capabilities, the Oracle CSM and/or the SIP registrar cannot support the number of registering endpoints. Although the Oracle CSM protects itself and the registrar against SIP REGISTER floods, conditions can still occur where too many legitimate endpoints attempt to register with the registrar via the Oracle CSM.

By enabling SIP registration cache limiting, you restrict the number of legitimate endpoints that can register. The Oracle CSM rejects any endpoints beyond the limit you set. If you do not want to use this feature, simply leave the `reg-cache-limit` parameter set to its default of 0, meaning there is no limit to the entries in the SIP registration cache.

When you limit the number of registered endpoints allowed in the Oracle CSM's registration cache, the Oracle CSM analyzes each registration before starting to process it. First, the Oracle CSM checks the contact header to determine if it is already in the list of contacts for the user. If it finds the contact in its cache list, the Oracle CSM treats the registration as a refresh; it treats any other headers as new. Note that the Oracle CSM checks the message prior to making any changes to the cache because it must either accept or reject the message as a whole.

The Oracle CSM adds the number of new contacts to the number already present in the cache, and rejects any registration with a contact that would cause it to exceed its limit. Rejection causes the Oracle CSM to send a response communicating that its registration cache is full. The default response is the 503 Registration DB-Full message, but you can use the SIP response mapping feature to use another message if required.

You can set an option in the global SIP configuration that defines the value in the `Retry-After` header. The Oracle CSM sends this header as part of its rejection response when the registration cache is full. Another option sets the percentage of the registration cache size which, if exceeded, causes the Oracle CSM to send an alarm.

About Registration Cache Additions Modifications and Removals

When it receives a REGISTER message with new contact information for a user, the Oracle CSM considers it an addition to the cache and augments the number of registration cache entries. Then the Oracle CSM forwards the message to the registrar, and—when and only when the registrar returns both the original and new contacts in the 200 OK—the registration cache count stays the same. However, if the registrar returns only the new contact (making this a case of modification), then the Oracle CSM removes the old contact information and subtracts accordingly from the number of registration cache entries.

Thus the Oracle CSM does not know whether a REGISTER might result in an addition or a modification until it receives a response from the registrar. For this reason, the Oracle CSM first assumes it is to make an addition, and then updates the registration cache and count when it has the necessary information from the registrar.

The registration cache count does not reflect removals during the rejection check because the Oracle CSM ignores registration messages or expires headers with their expires values set to zero when it counts new entries. The fact that removals take place after additions and modifications means that messages which remove one contact while adding another might be rejected. That is, the addition might exceed the registration cache limit before any removal can take place to make room for it.

Registration Cache Alarm Threshold

A percentage of the registration cache limit, the registration cache alarm threshold is a configurable value you can set to trigger an alarm when the registration cache is reaching its limit. When exceeded, this threshold triggers the generation of an alarm and SNMP trap. When registrations fall back beneath the threshold, the Oracle CSM clears the alarm and sends a clear trap.

SIP Signaling Services

This alarm is Major in severity, and its text reads as follows:

```
Number of contacts <registration count> has exceeded the registration cache
threshold <threshold %> of <registration cache limit value>.
```

Notes on Surrogate Registration

The Oracle CSM does not, under any circumstances, reject surrogate registrations on the basis of the registration cache limit. However, surrogate registrations generate contacts, and so they do add to the global registration count. In the case where the surrogate registrations add to the registration count to the extent the count exceeds the limit you configure, you will have more registrations in the cache than the configured limit.

Monitoring Information

You can monitor how many entries are in the SIP registration cache using the ACLI `show registration` command and referring to the Local Contacts statistics.

SIP Registration Cache Limiting Configuration

This section shows you how to configure the registration cache limit, and how to set the options controlling retry times and thresholds for alarm purposes.

To configure SIP registration cache limiting:

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

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

2. Type `session-router` and press Enter.

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

3. Type `sip-config` and press Enter.

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

If you are adding this feature to an existing configuration, you need to select the configuration (using the ACLI `select` command) before making your changes.

4. `registration-cache-limit`—Set the registration cache limit, or the maximum number of SIP registrations that you want to keep in the registration cache. The minimum and default value for this parameter is 0, and you can set it to a maximum value of 999999999. Leaving this parameter set to 0 means there is no limit on the registration cache (and therefore leaves this feature disabled).
5. `options`—Set the options parameter by typing `options`, a Space, the option name `reg-cache-lim-retry-after=X` (where X is the value added to the Retry-After header) with a plus sign in front of it. This option defaults to 1800, and you can enter values from 0 to 999999999.

You can configure the alarm threshold option the same way, substituting the option name `reg-cache-alarm-thresh=X` (where X is the percentage of registration cache limit that triggers an alarm). This option defaults to 95, and you can enter value from 0 to 100.

```
ORACLE (sip-config) # options +reg-cache-lim-retry-after=2500
ORACLE (sip-config) # options +reg-cache-alarm-thresh=90
```

If you type `options` and then the option value for either of these entries without the plus sign, you will overwrite any previously configured options. In order to append the new options to this configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.

6. Save and activate your configuration.

SIP Instance ID in Registration Cache

As defined in RFC 5626, Managing Client-Initiated Connections in the Session Initiation Protocol (SIP), the +sip.instance uniquely identifies a specific user agent instance. The instance-id does not change even when the User Agent is rebooted or power cycled. Using the instance-id from the +sip-instance parameter allows the SBC to locate a registered contact even when the IP address of the UA has changed.

The +sip.instance is a Contact header field parameter that contains a globally unique identifier, generally a Universally Unique Identifier (UUID) Uniform Resource Name (URN) that identifies a specific user agent client.

As defined in RFC 5626, Managing Client-Initiated Connections in the Session Initiation Protocol (SIP), the parameter contains an "instance-id" that uniquely identifies a specific user Agent instance. The instance-id does not change even when the User Agent is rebooted or power cycled (see Section 3.1 of RFC 5626). Using the instance-id from the +sip-instance parameter allows the SBC to locate a registered contact even when the IP address of the UA has changed.

An example REGISTER message containing a +sip-instance parameter and assigned value is shown below.

```
REGISTER sip:example.com SIP/2.0
Via: SIP/2.0/TCP 192.0.2.2;branch=z9hG4bK-bad0ce-11-1036
Max-Forwards: 70
From: Bob <sip:bob@example.com>;tag=d879h76
To: Bob <sip:bob@example.com>
Call-ID: 8921348ju72je840.204
CSeq: 1 REGISTER
Supported: path, outbound
Contact: <sip:line1@192.0.2.2;transport=tcp>; reg-id=1;
+sip.instance="<urn:uuid:00000000-0000-1000-8000-000A95A0E128>"
Content-Length: 0
```

The user agent calculates a +sip.instance generally using a time-stamp, a unique string -- such as a MAC address, and/or a randomly generated number. After calculating the value, the user agent saves it to persistent storage, thus ensuring that the value is unchanged by subsequent power cycles.

SIP Instance ID and the Registration Cache

Every assignment of a new IP address to a mobile user agent client results in a new REGISTER request. The receipt of the REGISTER request, in turn, generates an additional entry in the registration cache. The implementation can be simplified by including the +sip.instance value in the criteria used to match incoming REGISTER requests with existing sessions maintained in the registration cache. When a match is found, the SBC re-uses the existing registration cache rather than adding a new one.

If such a +sip.instance match is found and the contact address has changed, the SBC determines if digest authentication has been used in previous associated REGISTER requests. If so, the SBCs forwards the REGISTER request to the registrar for authentication of the new location.

SIP Instance ID Configuration

Because this configuration is dynamically performed at the **sip-config** level, it applies to all SIP interfaces and takes effect immediately

1. Access the **sip-config** configuration element.

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

2. Select the **sip-interface** object to edit.

```
ORACLE(sip-interface)# select
<RealmID>:
1: realm01 172.172.30.31:5060
```

```
selection: 1
ORACLE(sip-interface) #
```

3. **match-sip-instance**—Set this parameter to enabled to use the +sip-instance-id when matching incoming calls with the registration cache. Valid values are:

- disabled—(the default) disables the use of the +sip-instance-id when matching incoming calls with the registration cache
- enabled—enables the use of the +sip-instance-id when matching incoming calls with the registration cache

```
ACMEPACKET(sip-config) # match-sip-instance enabled
ACMEPACKET(sip-config) #
```

4. Type **done** to save your configuration.

SIP Request Method Throttling

You can configure throttling mechanisms for SIP INVITEs and REGISTERs using session agent constraints. However, you might want to throttle other types of SIP methods, and for those methods you should use the rate constraints configuration available both in the session constraints (which you then apply to a SIP interface or a realm) and the session agent configurations.

Oracle recommends you use session agent constraints for session-rate INVITE throttling and registration-rate for REGISTER throttling.

For SIP access deployments, you can configure rate constraints for individual method types along with a set of burst and sustain rates. These constraints can help to avoid overloading the core network. In addition, they restrain the load non-INVITE messages use, thus reserving capacity for INVITE-based sessions and Registrations

When you configure SIP request method throttling, you must exercise care because it is possible to reject in-dialog requests. Therefore, Oracle recommends you do NOT configure constraints—although the configuration allows you to and will not produce error messages or warnings if you set them—for the following SIP method types:

- ACK
- PRACK
- BYE
- INFO
- REFER

However, the Oracle CSM is likely to throttle NOTIFY requests despite their being part of a Subscribe dialog.

Therefore, the methods you will most likely configure for throttling are:

- NOTIFY
- OPTIONS
- MESSAGE
- PUBLISH
- REGISTER

The Oracle CSM counts Re-INVITEs and challenged responses against the throttle limit, but does not check to determine if the constraints have been exceeded for either.

You can configure separate constraints—inbound and outbound values for burst and sustain rates—for each different method type you configure. Although you should use session agent constraints (and not rate constraints) for INVITEs, if you also set up rate constraints for INVITEs, then the smallest configured value takes precedence.

About Counters and Statistics

Each rate constraint you configure for a SIP method tracks its own counters. For example, if you configure a rate constraint for the PUBLISH method, the burst and sustain rates you set for it apply only to the PUBLISH method and not to any other methods for which you might set up rate constraints. You can, however, set the burst rate window in the session constraints configuration that will apply to all methods configured as rate constraints.

The Oracle CSM captures statistics for SIP methods throttled by rate constraints for SIP interfaces and session agents; it does not capture these statistics for the global SIP configuration.

SIP Request Method Throttling Configuration

This section shows you how to set up rate constraints for session constraints (which are then applied to SIP interfaces) and session agents.

To use this feature, you must enable the `extra-method-stats` parameter in the global SIP configuration.

To set the `extra-method-stats` parameter in the global SIP configuration:

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

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

2. Type `session-router` and press Enter.

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

3. Type `sip-config` and press Enter.

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

If you are adding this feature to an existing configuration, you need to select the configuration (using the ACLI `select` command) before making your changes.

4. `extra-method-stats`—Set this parameter to `enabled`.
5. Save and activate your configuration.

Rate Constraints for SIP Interfaces

To apply rate constraints to SIP interfaces, you need to configure rate constraints in the session constraints configuration and then apply the session constraints to the SIP interface where you want them used.

Note that you need to set up the parent `session-constraint` configuration to save any rate constraints you configure.

To configure rate constraints:

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

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

2. Type `session-router` and press Enter.

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

3. Type `session-constraints` and press Enter.

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

If you are adding rate constraints to an existing configuration, then you will need to select the configuration you want to edit.

4. Type `rate-constraints` and press Enter.

```
ORACLE(session-constraints)# rate-constraints
ORACLE(rate-constraints)#
```

5. `method`—Enter the SIP method name for the method you want to throttle. Although the parameter accepts other values, your entries should come only from the following list for the feature to function properly:

- NOTIFY
- OPTIONS
- MESSAGE

- PUBLISH
 - REGISTER
6. **max-inbound-burst-rate**—For the SIP method you set in the methods parameter, enter the number to restrict the inbound burst rate on the SIP interface where you apply these constraints. The default and minimum value is 0, and the maximum is 999999999.
 7. **max-outbound-burst-rate**—For the SIP method you set in the methods parameter, enter the number to restrict the outbound burst rate on the SIP interface where you apply these constraints. The default and minimum value is 0, and the maximum is 999999999.
 8. **max-inbound-sustain-rate**—For the SIP method you set in the methods parameter, enter the number to restrict the inbound sustain rate on the SIP interface where you apply these constraints. The default and minimum value is 0, and the maximum is 999999999.
 9. **max-outbound-sustain-rate**—For the SIP method you set in the methods parameter, enter the number to restrict the outbound sustain rate on the SIP interface where you apply these constraints. The default and minimum value is 0, and the maximum is 999999999.
 10. Save your changes and apply this session constraint and its rate constraint(s) to SIP interfaces.

Applying Session and Rate Constraints to a SIP Interface

You need the name of the session constraints configuration to apply the restrictions you set up to a SIP interface.

To apply session and rate constraints to a SIP interface:

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

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

2. Type `session-router` and press Enter.

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

3. Type `sip-interface` and press Enter.

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

If you are adding this feature to an existing configuration, then you will need to select the configuration you want to edit.

4. **constraint-name**—Enter the name of the session constraint configuration where you have set up rate constraints to apply them to this SIP interface. This parameter has no default, and must be the valid name of a session constraint configuration.
5. Save and activate your configuration.

Configuring Rate Constraints for Session Agents

You can also use this feature for individual SIP session agents.

To configure rate constraints for a SIP session agent:

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

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

2. Type `session-router` and press Enter.

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

3. Type `session-agent` and press Enter.

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

If you are adding rate constraints to an existing configuration, then you will need to select the configuration you want to edit.

4. Type rate-constraints and press Enter.

```
ORACLE(session-agent)# rate-constraints
ORACLE(rate-constraints)#
```

5. method—Enter the SIP method name for the method you want to throttle. Your entries should come only from the following list:
 - NOTIFY
 - OPTIONS
 - MESSAGE
 - PUBLISH
 - REGISTER
6. max-inbound-burst-rate—For the SIP method you set in the methods parameter, enter the number to restrict the inbound burst rate on the SIP interface where you apply these constraints. The default and minimum value is 0, and the maximum is 999999999.
7. max-outbound-burst-rate—For the SIP method you set in the methods parameter, enter the number to restrict the outbound burst rate on the SIP interface where you apply these constraints. The default and minimum value is 0, and the maximum is 999999999.
8. max-inbound-sustain-rate—For the SIP method you set in the methods parameter, enter the number to restrict the inbound sustain rate on the SIP interface where you apply these constraints. The default and minimum value is 0, and the maximum is 999999999.
9. max-outbound-sustain-rate—For the SIP method you set in the methods parameter, enter the number to restrict the outbound sustain rate on the SIP interface where you apply these constraints. The default and minimum value is 0, and the maximum is 999999999.
10. Save and activate your configuration.

SIP Transport Selection

With this feature enabled, when the Oracle CSM forwards a message larger than the value specified in the maximum UDP length parameter, it attempts to open an outgoing TCP connection to do so. This connection might fail for a number of reasons; for example, an endpoint might not support UDP, or it might be behind a firewall. The UDP fallback option addresses this condition. If it is configured in SIP interfaces associated with an outgoing message and a TCP session cannot be established, the Oracle CSM falls back to UDP and transmits the message. When the option is not present, the Oracle CSM's default behavior is to return the SIP status message 513 Message too Large.

SIP Transport Selection Configuration

You enable this feature per SIP interface by setting options that control the maximum UDP length and allow UDP fallback:

- max-udp-length=X (where X is the maximum length)—Sets the largest UDP packets that the Oracle CSM will pass. Packets exceeding this length trigger the establishment of an outgoing TCP session to deliver the packet; this margin is defined in RFC 3261. The system default for the maximum UDP packet length is 1500.

You can set the global SIP configuration's max-udp-length=X option for global use in your SIP configuration, or you can override it on a per-interface basis by configuring this option in a SIP interface configuration.

- udp-fallback—When a request needs to be sent out on the SIP interface for which you have configured this option, the Oracle CSM first tries to send it over TCP. If the SIP endpoint does not support TCP, however, then the Oracle CSM falls back to UDP and tries the request again.

To enable SIP Transport Selection:

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

```
ORACLE# configure terminal
```

SIP Signaling Services

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
```

4. options—Set the options parameter by typing options, a Space, the option name max-udp-length=X (where X is the maximum UDP length you want to set), and then press Enter.

```
ORACLE(sip-interface)# options +max-udp-length=900
```

If you type options max-udp-length=X, you will overwrite any previously configured options. In order to append the new option to the sip-interface's options list, you must prepend the new option with a plus sign as shown in the previous example.

5. options—Set the options parameter by typing options, a Space, the option name udp-fallback, and then press Enter.

```
ORACLE(sip-interface)# options +udp-fallback
```

If you type options udp-fallback, you will overwrite any previously configured options. In order to append the new option to the sip-interface's options list, you must prepend the new option with a plus sign as shown in the previous example.

6. Save and activate your configuration.

uaCSTA NAT Support

The Oracle CSM offers User Agent Computer Supported Telecommunications Application (uaCSTA) support, which allows for the network address translation (NAT) of a key XML element in SIP INFO messages to use a phone's real contact URI.

Overview

Certain customers who use a uaCSTA for third party call control have encountered difficulties with the XML in their SIP messages used to support business applications. In these cases, the XML—specifically the <deviceID> XML tag—carries encoded IP addresses that need to be changed as they traverse the Oracle CSM.

The SIP business application allows users to click-to-dial another party using e-mail application clients. The user's click triggers the application server to send a uaCSTA SIP INFO message through the system to the UA/phone. These SIP INFO messages contain XML with the user's Contact-URI. But the server is only aware of the Oracle CSM's NAT'd Contact-URI and not the user's, so the XML in the SIP INFO is carrying incorrect information.

The XML element, then, needs to be NAT'd to the phone's real Contact-URI. This is especially important because of the broad use of SIP INFO messages, which instruct a phone to:

- Answer a call
- Hold a call
- Retrieve a call

All of these functions are available via a clickable interface on the e-mail application.

The Oracle CSM performs the NAT to the <deviceID> XML tag only if it is configured to perform registration caching.

When the Oracle CSM receives a SIP message from the core side and the request has:

- A Content-Type of application/csta+xml
- A Content-Length greater than 0

it parses the message's message body into an XML document. Should parsing fail, then the Oracle CSM will forward the SIP INFO request without modification to the XML message body. Otherwise, the Oracle CSM searches for the <deviceID> subelement within the XML document. If it finds the <deviceID> subelement, the Oracle CSM searches

through its registration cache for a registered Contact that matches the value of the <deviceID>. If it finds a match, the Oracle CSM replaces the value of the <deviceID> with that of the corresponding registered Contact. If the value of the <deviceID> is a Contact that the Oracle CSM generates for a registered UA, the corresponding contact from the look-up would be the Contact of the registered UA.

These functions performed, the Oracle CSM then reformats the SIP INFO request with the modified XML message body before sending it to the next hop. If there is no match found, then the Oracle CSM forwards the SIP INFO request without modifying the XML message body.

Other than ensuring your Oracle CSM is configured to perform registration caching, you do not need take any further steps.

SIP Method-Transaction Statistic Enhancements

In prior releases, the Oracle CSM tracks SIP session agents, SIP interfaces and SIP realms on a global level. Only counters that are related to session rates and constraints are displayed.

You can now enable your Oracle CSM to track transaction messages for specific SIP session agents, SIP realms, and SIP interfaces.

The following SIP methods are tracked for Recent, Total, and Period Max values:

- INVITE | ACK | BYE | REGISTER | CANCEL | PRACK | OPTIONS | INFO | SUBSCRIBE | NOTIFY | REFER | UPDATE | MESSAGE | PUBLISH | other (unknown)

With this new tracking enhancement, the show sipd command has been updated with a new method argument which allows you to query statistics for a particular method for a given SIP agent, SIP interface, or SIP realm.

SIP Method Tracking Enhancements Configuration

To enable or disable the expanded SIP Method statistics tracking:

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 sip-config and press Enter.

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

4. extra-method-stats—Enable this parameter if you want to use the expanded SIP Method tracking feature. The default is disabled. The valid values are:

- enabled | disabled

5. Save and activate your configuration.

SIP TCP Connection Reuse

You can configure your Oracle CSM to reuse TCP connections created by SIP peering devices for outgoing SIP in-dialog and out-of-dialog request transactions.

The SIP draft draft-ietf-sip-connect-reuse-07.txt describes a way for SIP UAs to reuse connections created by a remote endpoint for outgoing requests for TLS. The Oracle CSM does not support the model connection reuse signalled by a parameter; rather, it is provisioned on a per-session-agent basis.

You enable SIP TCP connection reuse on a per-session-agent basis. The Oracle CSM checks incoming TCP connection request to determine if they are from session agent that has this feature turned on. When it is, the Oracle CSM adds the connection's source address to its list of alias connections. This is a list of connections that the Oracle

SIP Signaling Services

CSM can use for outgoing requests rather than creating its own connection (as it does when this feature is not enabled). So if a preferred connection fails, the Oracle CSM can refer to this list and use the alias connection.

The presence of an alias parameter in the Via header is just one mechanism that will call the Oracle CSM to use the inbound TCP/TLS connection for outbound requests. The Oracle CSM will automatically add an alias for the inbound connections in the following circumstances:

- The other end of the connection is behind a NAT. When the Oracle CSM sees that the Via sent-by does not match the source address of the connection, it will automatically reuse the connection to deliver requests to the UA.
- The Contact address of a REGISTER request received on a TCP connection matches the source address and port. This is because the contact address is the ephemeral port the UA used to form the connection to the Oracle CSM and, therefore, will not be listening on that port for inbound connections.
- The presence of reuse-connections in the options field of the sip-interface will cause the Oracle CSM to reuse all inbound TCP connections for sending requests to the connected UA.

SIP TCP Connection Reuse Configuration

This section describes how to enable SIP TCP connection reuse for a session agent. Currently there are two options for the new reuse-connections parameter: none (which turns the feature off) and tcp (which enables the feature for TCP connections). You also set the re-connection interval.

To enable SIP TCP connection reuse for a session agent:

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

```
ORACLE# configure terminal
```

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

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

If you are adding support for this feature to a pre-existing session agent, then you must select (using the ACLI select command) the session agent that you want to edit.

4. reuse-connections—Enable or disable SIP TCP connection reuse. The default is none. This value disables the feature. The valid values are:
 - tcp | none
5. tcp-reconn-interval—Enter the amount of time in seconds before retrying a TCP connection. The default for this parameter is 0. The valid range is:
 - Minimum—0, 2
 - Maximum—300
6. Save and activate your configuration.

SIP TCP Keepalive

The Oracle CSM supports a special TCP keepalive mechanism for SIP. By enabling this feature either for a session agent or for a SIP interface, you allow the Oracle CSM to use standard keepalive probes to determine whether or not connectivity with a remote peer has been lost.

This feature adds to the Oracle CSM's pre-existing TCP keepalive functionality that you can enable in the network parameters configuration. Using existing functionality, you can customize keepalive timing by:

- Specifying the number of unacknowledged packets the Oracle CSM sends to the remote peer before it terminates the TCP connection.

- Specifying the number of seconds of idle time before TCP keepalive messages are sent to the remote peer.

You can now set three modes for TCP keepalive for session agents and SIP interfaces:

- none—(Default) Keepalives are not enabled for use with the session agent/SIP interface; when you select this setting for a session agent, it will use the setting for this feature from the SIP interface.
- enabled—Keepalives are enabled for the session agent/SIP interface.
- disabled—Keepalives are disabled for the session agent/SIP interface.

Note that the setting for this feature for a session agent takes precedence over that for a SIP interface. In addition, the session agent offers you a way to set the re-connection interval.

SIP TCP Keepalive Configuration for Session Agents

To enable SIP TCP keepalive for session agents:

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

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

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

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

3. Type session-agent and press Enter.

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

If you are adding support for this feature to a pre-existing session agent, then you must select (using the ACLI select command) the session agent that you want to edit.

4. tcp-keepalive—Enable or disable standard keepalive probes to determine whether or not connectivity with a remote peer is lost. The default value is none. The valid values are:

- none | enabled | disabled

```
ACMEPACKET (session-agent) # tcp-keepalive enabled
```

5. Save and activate your configuration.

SIP TCP Keepalive Configuration for SIP Interfaces

To enable SIP TCP keepalive for SIP interfaces:

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

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

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

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

3. Type sip-interface and press Enter.

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

If you are adding support for this feature to a pre-existing SIP interface, then you must select (using the ACLI select command) the SIP interface that you want to edit.

4. tcp-keepalive—Enable or disable SIP TCP keepalive. The default value is none. The valid values are:

- none | enabled | disabled

```
ORACLE (session-agent) # tcp-keepalive enabled
```

5. Save and activate your configuration.

Local Policy Session Agent Matching for SIP

When you enable the local policy session agent matching option in your global SIP configuration, you change the way local policies match session agents. Normally, the Oracle CSM looks up and stores matched session agents configured as next hops so it does not need to perform the lookup while processing requests. In this type of matching, the Oracle CSM does take the realm set in the local policy attributes into consideration. When the Oracle CSM performs its regular matching method and you have enabled overlapping IP addresses for session agents, the Oracle CSM might match session agents to different realms than the ones you intended when creating your configuration.

Local policy session agent matching provides a way to match session agents differently, taking realms and nested realms into consideration during the matching process. This difference is key to deployments with multiple peering partners that use the overlapping IP address feature, and have multiple local policies routing to the same IP address in different realms where some target next hops require session constraints but others do not. In the cases where no session constraints are required, session agents are not needed. But session agents still match the local policy, applying their constraints, because they match the next hop IP address.

In addition to modifying this behavior, this feature also affects the use of realms and nested realms. It triggers the use not only of realms, but of all the realms nested however deeply—thereby improving matching efficiency.

You can set the local policy session agent matching option with values that define how the Oracle CSM performs session agent matching:

- any—The Oracle CSM looks up and stores matched session agents configured as next hops so it does not need to perform the lookup while processing requests, without regard to realms.

This behavior is the default when the SIP configuration does not have the local policy session agent matching option set.

- realm—The Oracle CSM selects session agents in the realm that the local policy attribute indicates; this provides an exact match, rather than not taking the realm into consideration during session agent selection.

For example, the session agent is a match if the session agent realm-id and the local policy attribute realm parameters are an exact match.

- sub-realm—Session agents in the same realm or the same realm lineage—where session agents and realms are related to one another via realm parent-child relationships no matter the depth of realm nesting configured

For example, the session agent is a match if the local policy attribute realm is a sub-realm of the realm specified in the session agent realm-id parameter.

- interface—Session agents in the same realm or same realm lineage via the realm set in the local policy attribute, and whose realm uses the same signaling interface as the realm set in the local policy attribute

For example, the session agent is a match if the session agent realm-id is a sub-realm of the local policy attribute realm, and both referenced realms use the same SIP signaling interface.

- network—Session agents whose realm is in the realm lineage for the same realm set in the local policy attributes, and whose realm is associated with the same network interface as the realm set in the local policy attributes

For example, the session agent is a match if the session agent realm-id is a sub-realm of the local policy attribute realm, and realm reference by both use the same network interface.

If it cannot find a match, the Oracle CSM will use the IP address as the next hop. Further, requests matching local policy attributes will not be associated with session agents, and so their constraints will not be applied.

The Oracle CSM stores session agent information that it looks up when performing local policing session agent matching. To perform the lookup, it uses the session agent hostname as a key. When the hostname is an FQDN and there is a configured IP address in the ip-address parameter, the Oracle CSM uses the ip-address value as a secondary key. Given this implementation, the following are true when selecting session agents:

- If multiple session agents share the same IP address, the one with an IP address in the hostname parameter takes precedence.

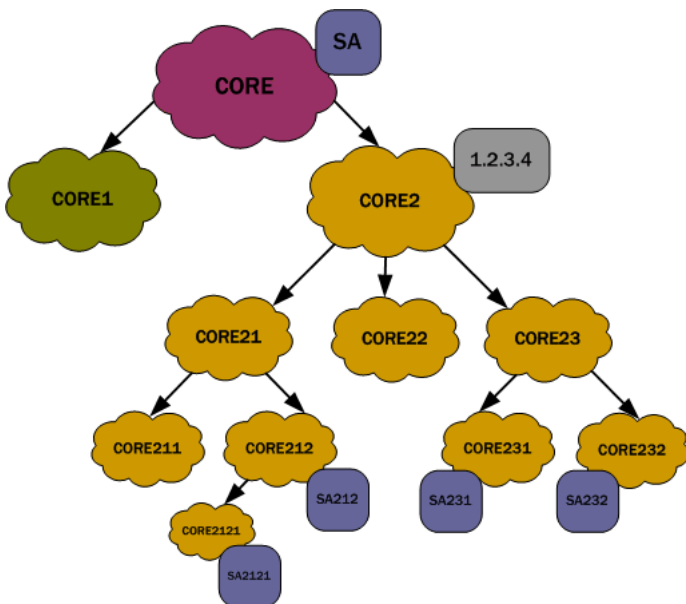
- If all session agents with the same IP address have an FQDN as their hostname, the one whose name is alphabetically lower will take precedence, where alphabetically lower means earlier in the alphabet (closer to A than to Z).
- For non-global session agents (whose realms are configured but not wildcarded) with an IP address, the Oracle CSM uses a key that is a combination of the IP address and the realm in the form <address>:<realm>.
- For a session agent whose realm has a parent realm, the Oracle CSM uses a combination of the IP address, realm, and realm-path (or lineage for the realm) in the form <address>:<realm-path>. For example, the realm path for a realm core3 with a parent core2, which in turn has a parent core would be core:core2:core3.

When it looks up a session agent with a realm, the Oracle CSM first searches for an exact match for the IP address and realm combination. If this fails, it performs a second search if the desired realm has parents or children. The Oracle CSM locates an entry in its repository of session agent information that is greater than or equal to the IP address with the base realm, which is the ancestor of the desired realm without a parent. Having gathered this set of candidates, the Oracle CSM narrows down the search for a match by comparing sub-realms and determines there is a match if either:

- The desired realm path is a sub-string of the entry’s realm path, or
- The entry’s realm path is a substring of the desired realm path (i.e., the desired realm is a sub-realm of the entry’s realm)

Then the Oracle CSM orders the candidates by depth of the entry’s realm-path, or number of levels from the base realm relative to the depth of the desired realm. By searching the ordered set until the entry’s realm depth equals the desired realm’s depth, the Oracle CSM determines a parent candidate, all subsequent entries being sub-realms of the desired realm. The Oracle CSM only considers entries at the first level deeper than the desired realm. If at this point there is only one entry, the Oracle CSM deems it a match. Otherwise, it selects the parent candidate as the matching entry. In the event the search does not yield a matching realm, the Oracle CSM uses the global session agent for the IP address, if there is one.

The following diagram shows the realm tree, where the clouds are realms and squares are session agents, representing a group of session agents sharing the IP address 1.2.3.4. The Oracle CSM searches for the session agents lower in the tree along the session agent realm-path and the desired realm.



For the diagram above, the following shows how the hostname would look for this group of session agents.

Key	Session Agent (hostname[realm])
1.2.3.4	1.2.3.4[CORE2]

SIP Signaling Services

Key	Session Agent (hostname[realm])
(This session agent owns the primary key for the IP address because its hostname is the IP address.)	
1.2.3.4:CORE (IP+realm key entry)	SA[CORE]
1.2.3.4:CORE (IP+realm key entry)	1.2.3.4[CORE2]
1.2.3.4:CORE212 (IP+realm key entry)	SA212[CORE212]
1.2.3.4:CORE2121 (IP+realm key entry)	SA2121[CORE2121]
1.2.3.4:CORE231 (IP+realm key entry)	SA231[CORE231]
1.2.3.4:CORE232 (IP+realm key entry)	SA232[CORE232]
1.2.3.4:CORE: (IP+realm-path key entry)	SA[CORE]
1.2.3.4:CORE:CORE2: (IP+realm-path key entry)	1.2.3.4[CORE2]
1.2.3.4:CORE2:CORE21:CORE212 (IP+realm-path key entry)	SA212[CORE212]
1.2.3.4:CORE2:CORE21:CORE212:CORE2121 (IP+realm-path key entry)	SA2121[CORE2121]
1.2.3.4:CORE2:CORE23:CORE231 (IP+realm-path key entry)	SA231[CORE231]
1.2.3.4:CORE2:CORE23:CORE232 (IP+realm-path key entry)	SA232[CORE232]

For each realm in the table above, the search results for each realm would look like this:

IP Address	Realm	Session Agent (hostname[realm])
1.2.3.4	CORE	SA[CORE]
1.2.3.4	CORE2	1.2.3.4[CORE2]
1.2.3.4	CORE21	SA212[CORE212]
1.2.3.4	CORE211	1.2.3.4[CORE2]

IP Address	Realm	Session Agent (hostname[realm])
1.2.3.4	CORE212	SA212[CORE212]
1.2.3.4	CORE2121	SA2121[CORE2121]
1.2.3.4	CORE22	1.2.3.4[CORE2]
1.2.3.4	CORE23	1.2.3.4[CORE2]
1.2.3.4	CORE231	SA231[CORE231]
1.2.3.4	CORE232	SA232[CORE232]

Local Policy Session Agent Matching Configuration

When you enable local policy session agent matching, remember that you can choose from five different ways to use the feature: all, realm, sub-realm, interface, and network.

This example shows you how to use the realm selection.

To enable local policy session agent matching using the realm method:

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

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

2. Type session-router and press Enter.

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

3. Type sip-config and press Enter.

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

4. options—Set the options parameter by typing options, a Space, the option name lp-sa-match=X (where X is the local policy session agent matching method you want to use) with a plus sign in front of it. Then press Enter.

Remember that if you do not specify a method, the system uses the all method.

```
ORACLE (sip-config) # options +lp-sa-match=realm
```

If you type options and then the option value for either of these entries without the plus sign, you will overwrite any previously configured options. In order to append the new options to this configuration's options list, you must prepend the new option with a plus sign as shown in the previous example.

5. Save and activate your configuration.

- Unordered—Meaning that the endpoint can deliver data within regard for their stream sequence number

You set this preference in the network parameters configuration.

About Wildcarding

The Oracle CSM supports wildcarding the event type in the subscribe-event configuration. To wildcard the value, you enter an asterisk (*) for the event-type parameter instead of typing in the name of an actual event type.

When you wildcard this value, the Oracle CSM applies the subscription limitations you set across all event types. Or, if you have entered multiple subscribe-event configurations, the Oracle CSM applies the wildcard limits across the event types for which you have not set limits.

Consider the following example of a configured enforcement profile with a wildcarded subscribe-event configuration:

```
enforcement-profile
    name                    rulefour
    allowed-methods
    sdp-address-check      disabled
```

```
subscribe-event
  event-type                *
  max-subscriptions         1
subscribe-event
  event-type                xyz
  max-subscriptions         0
last-modified-by           admin@console
last-modified-date         2008-11-11 12:49:27
```

In this example, the enforcement profile allows all subscriptions that are event type xyz for a user. But it allows only one maximum for every other subscription event type.

Monitoring

You can display the number of subscription dialogs per SUBSCRIBE event type using the ACLI show registration sipd subscriptions-by-user command. You can display this information per event type, or you can show data for all event types by wildcarding the event type argument.

Enforcement Profile Configuration with subscribe-event

This section shows you how to configure an enforcement profile with a subscribe-event configuration. Remember that you can set up multiple subscribe-event configurations to correspond with the event types you want to control. It also shows you how to apply these limitations to a realm.

Setting Up Subscribe Dialog Limits

Setting up subscribe dialog limits means setting up an enforcement profile. For the sole purpose of setting up the subscription event limits, you only need to configure the name parameters and then as many subscribe-event configurations as you require. The enforcement profile has other uses, such as SIP SDP address correlation, so only configure the parameters you need.

To configure subscribe dialog limits:

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

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

2. Type session-router and press Enter.

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

3. Type enforcement-profile and press Enter.


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

4. name—Enter a name for this enforcement profile. You will use this name later when you apply the enforcement profile to a realm; it is the value you enter into the enforcement-profile parameter in the realm configuration.
5. Still in the enforcement profile configuration, type subscribe-event and press Enter.

```
ORACLE (enforcement profile) # subscribe-event
ORACLE (subscribe-event) #
```

6. event-type—Enter the SIP subscription event type for which you want to set up limits. You can also wildcard this value (meaning that this limit is applied to all event types except the others specifically configured in this enforcement profile). To use the wildcard, enter an asterisk (*) for the parameter value.

By default, this parameter is blank.

 **Note:** The value you enter must be configured as an exact match of the event type expected in the SIP messages (except for the wildcard). Further, the value conforms to the event type BNF specified in RFC 3265.

7. max-subscriptions—Enter the maximum number of subscriptions allowed to a user for the SIP subscription event type you entered in the event-type parameter. Leaving this parameter set to 0 (default) means that there is no limit. You can set this parameter to a maximum value of 65535.

- If you are entering multiple subscribe-event configurations, then you save them each by using the ACLI done command and then repeat Steps 6 and 7 to configure a new one. If you do not save each, then you will simply overwrite the first configuration repeatedly.

```
ORACLE(subscribe-event)# done
```

- When you finish setting up subscribe-event configurations and have saved them, exit to return to the enforcement profile configuration.

```
ORACLE(subscribe-event)# exit
```

- You also need to save the enforcement profile configuration.

```
ORACLE(enforcement profile)# done
```

Applying an Enforcement Profile to a Realm

For the Oracle CSM to use the limits you have set up, you need to apply them to a realm.

To apply an enforcement profile to a realm:

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

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

- Type media-manager and press Enter.

```
ORACLE(configure)# media-manager
ORACLE(media-manager)#
```

- Type realm-config and press Enter. If you are adding this feature to a pre-existing realm configuration, you will need to select and edit your realm.

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

- enforcement-profile—Enter the name of the enforcement profile you want to apply to this realm. This value corresponds to the name parameter in the enforcement profile configuration. This parameter has no default value.
- Save and activate your configuration.

SIP Session Timer Feature

SIP does not have a keepalive mechanism for established sessions and it does not have the capability of determining whether a session is still active. User agents (UAs) may be able to determine whether a session has timed out by using session specific mechanisms, but proxies cannot always determine when sessions are still active.

The Oracle CSM provides a SIP session timer feature that, when enabled, forwards the re-INVITE or UPDATE requests from a User Agent Client (UAC) to a User Agent Server (UAS) in order to determine whether or not a session is still active. This refresh feature works for both UAs and proxies. The following paragraphs provide additional information about the session timers on the Oracle CSM.

How the Session Timer Feature Works

During an active SIP call session, when a UA fails to send a BYE message at the end of the session, or when the BYE message gets lost due to network problems, the proxy cannot determine when the session has ended. Therefore, the proxy may hold onto resources associated with the call session for indefinite periods of time causing the session to never time out.

The SIP session timer feature adds the capability to periodically refresh SIP sessions by sending repeated INVITE (re-INVITE) or UPDATE Session Refresh Requests. These requests are sent during active call legs to allow UAs or proxies to determine the status of a SIP session. The Session Refresh Requests along with the session timers determine if the active sessions stay active and completed sessions are terminated.

When you enable the session timer feature on the Oracle CSM, it periodically sends out a Session Refresh Request (re-INVITE or UPDATE). The Response that is returned to the Oracle CSM contains a success status code (2xx) that

contains a session timer interval. The Oracle CSM then refreshes the session timer each time it receives the 2xx Response containing that session timer interval.

The initial INVITE message sent from the UAC to the UAS contains two fields that make up the session timer interval in the SIP Session Header:

- Session-Expires (SE) - Specifies the maximum amount of time, in seconds, that can occur between session refresh requests in a dialog before the session is considered timed out.
- Minimum-SE (Min-SE) - Specifies the minimum allowed value, in seconds, for the session expiration.

The following displays the session timer interval values inserted in the SIP session INVITE message per RFC 4028:

```
INVITE sip:9109621001@192.168.200.99 SIP/2.0
Via: SIP/2.0/UDP 192.168.200.49:5060;branch=z9hG4bK0g6t23200gd0res41580.1
Max-Forwards: 69
From: <sip:rick@192.168.1.48>;tag=SDr08od01-188c3fbc-b01a-4d68-
b741-09e5dc98a064
To: sip:149@192.168.1.49
Contact: <sip:rick@192.168.200.49:5060;transport=udp>
Call-ID: SDr08od01-9c12b48e3b0f7fad39ff3a2e0ced5ed3-v3000i1
CSeq: 3941 INVITE
Allow: INVITE, ACK, BYE, CANCEL, UPDATE, PRACK
Supported: timer
Session-Expires: 1800
Min-SE: 90
Content-Type: application/sdp
Content-Length: 236

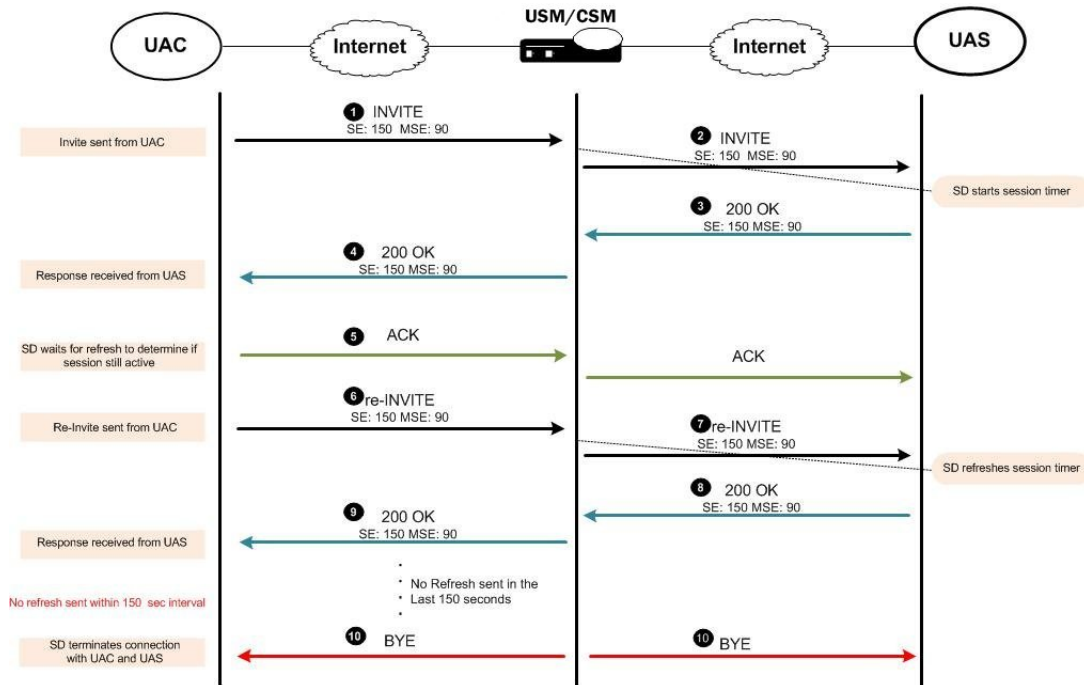
v=0
o=- 3462189550 3462189550 IN IP4 192.168.200.49
s=pjmedia
c=IN IP4 192.168.200.49
t=0 0
m=audio 20000 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=sendrecv
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
```

If the Oracle CSM receives an INVITE from the UAC with a Session-Expires header, it starts a new session timer, or refreshes an existing session timer and then forwards the INVITE to the UAS. The subsequent 2xx Responses and re-INVITES also include the session timer intervals. If the Oracle CSM does not receive a session refresh within the time specified in the session timer interval, the session timer expires, and the Oracle CSM terminates the session between the UAC and the UAS.

The following occurs when you enable the session timer feature on the Oracle CSM:

1. The UAC sends an INVITE to the Oracle CSM with the SE and min-SE values (session timer interval). The Oracle CSM starts a session timer.
2. The Oracle CSM forwards the INVITE to the User Agent Server (UAS) with the same values.
3. The UAS sends the 200 OK Response to the Oracle CSM with the session interval values.
4. The Oracle CSM forwards the Response to the UAC.
5. The UAC sends an ACK (Acknowledge) to the Oracle CSM, and the Oracle CSM forwards the ACK to the UAS.
6. The UAC sends out a re-INVITE (Session Refresh Request) to the Oracle CSM with the session interval values. The Oracle CSM refreshes the session timer.
7. The Oracle CSM forwards the re-INVITE to the UAS.
8. The UAS sends the 200 OK response to the Oracle CSM with the session interval values.
9. The Oracle CSM sends the 200 OK response to the UAC.

10. If the Oracle CSM does not receive a Response within the session interval (150 seconds in the following illustration), the timer expires, and the Oracle CSM terminates the session between the UAC and the UAS. The following illustration shows an example of a dialog between the UAC, the Oracle CSM, and the UAS during an active session.



When the Oracle CSM terminates a session it sends a BYE to both the ingress and egress call legs. If accounting is configured, the Oracle CSM also sends a RADIUS stop record with Acct-Terminate-Cause = Session-Timeout. You can enable or disable the use of the session timers using the ACLI interface at `session-router > sip-config > options`.

SIP Session Timer Configuration

You can configure the session timer feature on the Oracle CSM to periodically refresh SIP sessions and determine whether or not a session is still active. If the Oracle CSM determines that a session is no longer active, it terminates the session based on the session timer interval settings.

To configure the session timer feature on the Oracle CSM:

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-related configurations.

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

3. Type `sip-config` and press Enter to access the SIP config-related configurations. The system prompt changes to let you know that you can begin configuring individual parameters for this object.

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

4. Enter options followed by the following value:

- `+session-timer-support`

```
ORACLE(sip-config)# options +session-timer-support
```

This value enables the system to start the session timer for session refreshes coming from the UAC. The system determines whether or not a session is active based on session refreshes or responses. It terminates the

session when no session refreshes occur within the session timer interval. To disable the session timer feature, enter options followed by `-session-timer-support`.

```
Oracle CSM(sip-config)# options -session-timer-support
```



Note: To disable session timers, you must add a minus sign (-) before the `session-timers-support` value.

305 Response to Registrations on Secondary Interfaces

Certain devices are provisioned with point of contact (POC) lists for registration. In the context of geographic redundancy, if a UE is unable to register with its primary POC, it proceeds to its secondary POC. It is desirable to designate certain Oracle CSM as secondary in order to redirect the UE to re-register with the primary SBC.

This feature is designed for Oracle CSMs sitting behind Oracle Communications Subscriber-Aware Load Balancers (SLBs). This feature allows users to designate CSMs sitting behind SLBs as secondary. When registered endpoints in a realm attempt to register with the primary POC, and registration fails, the UEs proceed to their secondary point of contact (POC). Once connection to the primary POC has been restored, users can execute a command to offload the UE registrations in a given realm with a 305 response to the UEs in order for the UEs to re-register with the next POC. When the scenario consists of two POCs, the endpoint re-registers with the primary POC.

You enable this feature with an option through the SIP interface of the secondary SBC. The `notify sipd offload-users <realm>` command marks endpoints associated with a given realm to receive a 305 message. Once this command is executed, any subsequent requests from UEs in this designated realm receive a 305 message. Once a 305 message is issued, the registration cache is cleared to allow the UE request to be forwarded to the primary SBC. If the UE re-registers again with the secondary SBC, it will not receive a second 305 message. The `notify sipd offload-users <realm>` command must be executed again in order to repeat the process.

The `show registration` command is expanded to include a counter for the number of registrations received on secondary interfaces, as well as the number of endpoints marked to receive a 305.

SNMP support for this feature tracks the total count of registrations on secondary interfaces. The `apSipSecInterfaceRegThresholdExceededTrap` is generated when the total number of registrations on secondary interfaces exceeds the configured threshold, and `apSipSecInterfaceRegThresholdClearTrap` is generated when the total count falls below the configured threshold.

ACLI Instructions and Examples

To enable this feature, you must select an option for the SIP-interface of the secondary Oracle CSM:

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

```
ACMEPACKET# configure terminal
```

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

```
ACMEPACKET(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.

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

4. `options`—Type `secondary` to designate this system as the secondary point of contact.

```
ACMEPACKET(session-router)# options secondary
```

5. Save and activate your configuration.

notify sipd offload-users

The `notify sipd offload-users <realm>` command marks the endpoints associated with the given `<realm>` to receive a 305 message. Once this command is executed, any re-registrations received from an endpoint in that realm receives a 305 message.

A confirmation message is sent if the command succeeds:


```
Realm <realm> is marked to offload users at HH:MM:SS
```

If the given <realm> is not a secondary realm, the following message is displayed:

```
Realm <realm> is not secondary
```


If the given <realm> does not exist, the following message is displayed:

```
Realm <realm> is not found
```

 **Note:** This command is only valid for secondary systems. If the secondary option is not enabled for the Oracle CSM, this command produces an error message.

show registration

The show registration command displays the total number of endpoints associated with secondary interfaces, and the total number of endpoints marked to receive a 305 response.

 **Note:** The total number of SIP calls to receive a 305 response are included in the Registrations marked for 305 counter.

The total number of SIP calls associated with secondary realms are included in this count.

```
AcmePacket# show registration
11:02:13-102
SIP Registrations          -- Period --  ----- Lifetime -----
                          Active   High   Total      Total  PerMax   High
User Entries                1     1     1          1     1       1
...
Secondary Interface Registrations=NNN
Registrations marked for 305=MMM
access1: N1
access2: N2
```

show registration sipd

The show registration sipd command includes the following parameters:

show registration sipd sec-by-ip—displays the registrations sent to secondary interfaces by IP address.

```
Registration Cache          MON FEB 06 2012  16:33:49
                          Num
 IP Address      User                Contacts Registered at
-----
. . . .
```

show registration sipd sec-by-user—displays the registrations sent to secondary interfaces by user name.

```
Registration Cache          MON FEB 06 2012  16:33:49
                          Num
 Phone Number    User                Contacts Registered at
-----
. . . .
```

show sipd endpoint-ip

If the registered endpoint is associated with a secondary SIP interface, the show sipd endpoint-ip command displays SIP Interface: secondary.

```
# show sipd endpoint-ip 1
User <sip:12341@172.16.101.67>
Contact exp=290
  UA-Contact: <sip:12341@172.16.26.1:5060> UDP keep-acl
              realm=net172 local=172.16.101.67:5060 UA=172.16.26.1:5060
  SD-Contact: <sip:12341-2k86jbp9bj925@192.168.101.67:5060> realm=net192
```

```
Call-ID: 1-839@172.16.26.1'  
SA=192.168.26.2  
Service-Route='<sip:h19216806003.dg.com;lr>'  
SIP Interface: secondary
```

SNMP Configuration

To configure the threshold for the number of registrations on secondary SIP interfaces in order to generate a trap:

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

```
ACMEPACKET# configure terminal
```

2. Type session-router and press Enter.

```
ACMEPACKET(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.

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

4. options—Type sec-reg-threshold=X, where X equals the number of registrations received on all secondary interfaces. Once that number has been reached, the apSipSecInterfaceRegThresholdExceededTrap is generated.

```
ACMEPACKET(session-router)# options secondary
```

5. options—Type sec-reg-threshold-clear=Y. When the number of registrations received on all secondary interfaces drop below Y, the apSipSecInterfaceRegThresholdExceededClearTrap is generated.



Note: The value of sec-reg-threshold-clear=Y must be less than sec-reg-threshold=. The values cannot be equal.

```
ACMEPACKET(session-router)# options secondary
```

6. options—Type sec-reg-threshold-alarm-severity = [minor | major | critical] to set the alarm severity for APP_ALARM_SEC_INTF_REG_EXCEED.

7. Save and activate your configuration.

SNMP

The apSipSecInterfaceRegThresholdExceededTrap trap is generated when the total count of all registrations on the secondary interfaces cross sec-reg-threshold-exceed is exceeded.

Trap Name	Object Identifier Name: Number	Description
Object Identifier Name: apSipInterfaceRegNotificationsGroup		
apSipSecInterfaceRegThresholdExceededTrap	.1.3.6.1.4.1.9148.3.15.2.1.2.0.1	The trap will be generated if the total number of registrations on all secondary SIP interfaces exceed threshold.
apSipSecInterfaceRegThresholdClearTrap	.1.3.6.1.4.1.9148.3.15.2.1.2.0.2	The trap will be generated if the total number of registrations on all secondary SIP interfaces go below clear threshold.

An alarm is also generated for this event. The severity of the alarm is configured in the sec-reg-threshold-alarm-severity option.

Alarm Name	Alarm ID	Alarm Severity	Cause(s)	Example Log Message	Trap Generated (Trap Reference)
Hardware Alarms					
APP_ALARM_SEC_INTF_REG_EXCEED	0X50018	Minor by default. The severity is configurable.	The total number of registrations on all secondary SIP interfaces exceeds the configured threshold.	Number of Secondary Interface registrations <total> has exceeded the secondary interface registration threshold of <max>.	apSyslogMessageGenerated (ap-slog.mib) apEnvMonStatusChangeNotification (ap-env-monitor.mib) apSysMgmtFanTrap (ap-smgmt.mib)

The following objects monitor the number of registrations on secondary interfaces:

SNMP GET Query Name	Object Identifier Name: Number	Description
Object Identifier Name: apSipInterfaceRegObjectsGroup		
apSipSecInterfaceTotalRegistrations	.1.3.6.1.4.1.9148.3.15.1.1.1.1.0	The total number of registrations on all secondary SIP interfaces.
apSipSecInterfaceRegThreshold	.1.3.6.1.4.1.9148.3.15.1.1.1.2.0	The max threshold for registrations on all secondary SIP interfaces beyond which trap and alarm will be raised
apSipSecInterfaceClearThreshold	.1.3.6.1.4.1.9148.3.15.1.1.1.3.0	The threshold for registrations on all secondary SIP interfaces below which if alarm was raised before, it will be cleared.

Number Translation

About Number Translation

Oracle CSM number translation is used to change a layer-5 endpoint name according to prescribed rules. Number translations can be performed on both the inbound and the outbound call legs independently, before and after routing occurs.

Number translation takes place twice for SIP calls. The first number translation is applied to the incoming leg of the call, before the outgoing route is selected. The second number translation is applied to the outgoing leg of the call after the outgoing route is selected.

Number translation can be used to strip address prefixes added by external gateways. It can also be used to add a string tag to an address in order to implement a local policy routing scheme, and then remove the tag upon egress from the Oracle CSM. The most common use of number translation is to add or remove a “1” or a + from a phone number sent from or addressed to a device.

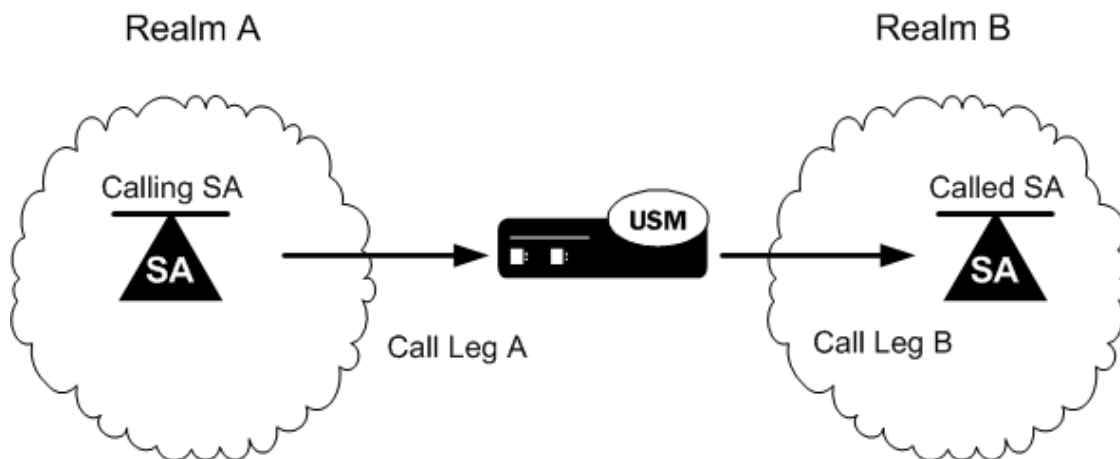
Number Translation Implementation

Oracle CSM number translations are implemented in three steps. First, the individual number translation rules are defined in the translation rules subelement. Next, the established rules are grouped in a specified order to apply to calling and called numbers. This second step occurs in the session translation element. Finally, session translations are attached to either session agents or realms in the session agent element or realm configuration element.

Number translations attached to session agents take precedence over number translations attached to realms. If no number translation is applied to a session agent, then the Oracle CSM will use the number translation applied to a realm. If a number translation is applied to both a realm and session agent, the translation attached to the session agent will apply. If session agents and realms have no associated translations, then all numbers will remain in their original forms as they pass through the Oracle CSM.

Within each realm or session agent, the number translation is applied to either the incoming or outgoing call leg. This distinction between incoming and outgoing calls is made from the point of view of the Oracle CSM. The following diagram illustrates the number translation concept.

Number Translation



The following table shows you which parameters to apply a session translation ID in order to affect the corresponding leg of the call as shown in the illustration.

Leg	Calling SA	Called SA	Realm A	Realm B
A	IN Translation ID		IN Translation ID	
B		OUT Translation ID		OUT Translation ID

Number Translation in SIP URIs

Number translations only change the user portion of the URI. A typical SIP URI looks like sip:user@hostname. The user portion can take the form of either a phone number or any other string used for identification purposes.

Within the SIP header exists a Request URI, a To URI, and a From URI. The session translation element's rules calling parameter modifies the From URI, while the rules called parameter modifies the Request URI and the To URI.

Number Translation Configuration Overview

Configuring the number translation feature requires the following steps:

1. Configure individual translation rules in the translation rules element.
2. Group these rules for use in the session translation element.
3. Apply these groups of rules on a per session agent or per realm basis using the appropriate fields in the session agent or realm configuration elements.

Translation Rules

The translation rules subelement is where the actual translation rules are created. The fields within this element specify the type of translation to be performed, the addition or deletion to be made, and where in the address that change takes place. Translations are not applied to any realm or session agent in this element.

When creating translation rules, first determine the type of translation to perform. The following table lists and describes the three types of number translations.

Field Value	Description
add	This translation type adds a character or string of characters to the address.
delete	This translation type deletes a character or string of characters from the address.

Field Value	Description
replace	This translation type replaces a character or string of characters within the address. Replace works by first applying the delete parameter then by applying the add parameter.

After you set the translation type, you define the string to add or delete. The wildcard term for a string to delete is the at-sign, @. Finally, you specify the character position in the address to make the addition or deletion.

The character position where an add or delete occurs is called an index. The index starts at 0 (immediately before the leftmost character) and increases by 1 for every position to the right you move. In order to specify the final position in an address, use the dollar-sign, \$.

To create a translation rule that deletes a string:

Translation Rules for Deleting Strings

To create a translation rule that deletes a string:

1. Enter a descriptive name for this translation in the ID field.
2. If you are deleting a specific string, enter it in the delete string field.
3. If you are deleting a portion of the address string, enter the index number in the delete index field. For this type of deletion, remember to enter the number of characters you are deleting in the form of at-signs @ in the delete string field.

The first matched string will be deleted, any remaining strings that match will remain. For example, if the address is 187521865 and the string to delete is “18,” only the first instance of 18 will be deleted. The second instance will remain after translation.


Translation Rules for Adding Strings

To create a translation rule that adds a string:

1. Enter a descriptive name for this translation in the ID field.
2. Enter the string you want to add in the add string field.
3. Enter the index of the string insertion in the add-index parameter. If you want to add a string at the end of an number, enter a dollar-sign \$ in the add index field.

Translation Rules for Replacing Strings

To create a translation rule that replaces a string:

 **Note:** A string replacement involves deleting a string followed by adding a string in the removed string’s place. The index is not used when replacing a string.

1. Enter a descriptive name for this translation in the ID field.
2. Enter the string you want to delete in the delete string field.
3. Enter the string you want to add in the add string field.

Session Translation

A session translation defines how translation rules are applied to calling and called numbers. Multiple translation rules can be referenced and applied using this element, which groups rules together and allows them to be referenced by one identifier.

There are two parameters in the session translation element. The rules calling parameter lists the translation rules to be applied to the calling number. The rules called parameter lists of translation rules to be applied to the called number.

Number Translation

The Oracle CSM applies the translation rules in the order in which they are entered. They are applied cumulatively. For example, if this field is configured with a value of rule1 rule2 rule3, rule1 will be applied to the original number first, rule2 second, and rule3 last.

To configure the session translation element:

1. Enter a descriptive name for this session translation in the ID field.
2. Enter the IDs of existing translation rules in the rules calling parameter. Multiple rules can be entered in this field. The order you enter them in is the order in which they are applied.
3. Enter the IDs of existing translation rules in the rules called parameter. Multiple rules can be entered in this field. The order you enter them in is the order in which they are applied.

Applying Session Translations

Session translations can be applied to both session agents and realms. Both session agents and realms contain the two parameters that denote incoming and outgoing call legs—in translation ID and out translation ID. These two fields are populated with session translation element IDs.

If none of these fields are populated, no number translation will take place and the original address will remain unchanged as it traverses the Oracle CSM. Further, any session translation applied to a session agent takes precedence over one applied to a realm.

Session Agent

To configure number translation for a session agent:

In the session agent element, set the in translation ID and/or the out translation ID to the appropriate ID you configured in the session translation element. There can be only one entry in each of these fields.

Realm

To configure number translation for a realm:

In the realm configuration element, set the in translation ID and/or the out translation ID to the appropriate ID you configured in the session translation element. There can be only one entry in each of these fields.

Number Translation Configuration

To create a translation rule:

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 configuration elements.

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

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

```
ORACLE(session-router)# translation-rules  
ORACLE(translation-rules)#
```

From this point, you can configure translation rules parameters. To view all translation rules parameters, enter a ? at the system prompt. The following is an example of what a translation rule configuration might look like. Parameters not described in this section are omitted below.

```
translation-rules  
  id          addplus1  
  type        add  
  add-string  +1
```

add-index	0
delete-string	
delete-index	0

Translation Rules

Set the following parameters to configure a translation rule:

1. ID—Set a descriptive ID name for this translation rule.
2. type—Set the type of translation rule you want to configure. The default value is none. The valid values are:
 - add—Adds a character or string of characters to the address
 - delete—Deletes a character or string of characters from the address
 - replace—Replaces a character or string of characters within the address
 - none—Translation rule is disabled
3. add-string—Enter the string to be added during address translation to the original address. The value in this field should always be a real value; i.e., this field should not be populated with at-signs (@) or dollar-signs (\$). The default value is a blank string.
4. add-index—Enter the position, 0 being the left most position, where you want to add the string defined in the add-string parameter. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999
5. delete-string—Enter the string to be deleted from the original address during address translation. Unspecified characters are denoted by the at-sign symbol (@).

The @ character only works if the type parameter is set to delete. This parameter supports wildcard characters or digits only. For example, valid entries are: delete-string=@@#@#@, or delete-string=123456. An invalid entry is delete-string=123@@@. When the type is set to replace, this value is used in conjunction with the add-string value. The value specified in the delete-string field is deleted and the value specified in the add-string field is inserted. If no value is specified in the delete-string parameter and the type field is set to replace, then nothing will be inserted into the address. The default value is a blank string.

6. delete-index—Enter the position, 0 being the left most spot, where you want to delete the string defined in the delete-string parameter. This parameter is only used if the delete-string parameter is set to one or more at-signs. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999

Session Translation

To configure session translations:

1. Exit out of the translation rules element and enter the session translation element.

```
ORACLE(translation-rules)# exit
ORACLE(session-router)# session-translation
ORACLE(session-translation)#
```

From this point, you can configure the session translation element. To view all session translation parameters, enter a ? at the system prompt. The following is an example of what a session translation configuration might look like:

```
session-translation
  id                               lrules-out
  rules-calling                    rule1 rule2 rule3
  rules-called                      addplus1
```

2. ID—Set a descriptive ID name for this session translation.
3. rules-calling—Enter the rules calling in the order in which they should be applied. Multiple rules should be included in quotes and separated by spaces.

Number Translation

```
ORACLE(session-translation)# rules-calling "rule1 rule2 rule3"
```

4. rules-called—Enter the rules called in the order in which they should be applied. Multiple rules should be included in quotes and separated by spaces.

Number Translation Application

To complete your number translation configuration, you must enter into a realm-config or session-agent element and assign session-translations there.

1. Navigate to the chosen element.

- To move from the session-translation element to the session-agent element, exit out of the session translation element and enter the session agent element.

```
ORACLE(session-translation)# exit
ORACLE(session-router)# session-agent
ORACLE(session-agent)#
```

- To move from the session-translation element to the realm-config element, exit from the session translation element to the configuration path.

```
ORACLE(session-translation)# exit
ORACLE(session-router)# exit
```

Navigate to the realm-config element located in the media-manager path.

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

2. In both realm-config or session agent elements, you must specify an in-translationid and/or an out-translationid in order to configure the number translation.

```
session-agent
    in-translationid
    out-translationid          lrules-out
realm-config
    in-translationid
    out-translationid          lrules-out
```

Set the following parameters to configure a translation rule:

- in-translationid—Enter the configured session translation that you want to affect the incoming traffic in this parameter.
- out-translationid—Enter the configured session translation that you want to affect the outgoing traffic in this parameter.

Other Translations

FQDN Mapping

The Oracle CSM maps FQDNs that appear in certain headers of incoming SIP messages to the IPv4 address that the Oracle CSM inserts in outgoing SIP contact headers. The mapped FQDNs are restored in the SIP headers in messages that are sent back to the originator.

This feature is useful to carriers that use IPv4 addresses in the SIP From address to create trunk groups in a PSX for routing purposes. When the carrier's peer uses FQDNs, the carrier is forced to create trunk groups for each possible FQDN that it might receive from a given peer. Similarly, this can apply to SIP Contact and P-asserted-identity headers.

Admission Control

This chapter describes how to configure the Oracle CSM for call admission control. Call admission control lets you manage call traffic based on several different policies. It is aimed at managing call admission rates in the network, enabling you to maintain suitable QoS levels. A new call is admitted only if it meets the requirements

Session Capacity- and Rate-based Admission Control

A session agent defines a signaling endpoint. It is a next hop signaling entity that can be configured to apply traffic shaping attributes. You can define concurrent session capacity and rate attributes for each session agent.

You can configure a set of attributes and constraints for each session agent to support session access control. In this configuration, the Oracle CSM only accepts requests from configured session agents. And you can set up session admission control so that the Oracle CSM limits the number of concurrent inbound and outbound sessions for any known service element.

The Oracle CSM denies a call request to any destination that has exceeded its configured policies for session capacity and session rate. The Oracle CSM might reject the call request back to the originator. If multiple destinations are available, the Oracle CSM will check current capacity and rate for each destination and attempt to route the call only to destinations whose policy limits have not been reached.

You assign a media profile to a session agent.

Constraints for Proxy Mode

The Oracle CSM applies session router and session agent constraints when it is in proxy (transaction or stateless) mode if you enable the ACLI constraints parameter for a session agent. However, the Oracle CSM does not track SIP sessions when in transaction mode, so the following session-specific constraints are not applied:

- max-sessions
- max-inbound-sessions
- max-outbound-sessions
- min-seizures
- min-asr

Constraints the Oracle CSM applies are:

- max-burst-rate
- max-inbound-burst-rate
- max-outbound-burst-rate

Admission Control

- max-sustain-rate
- max-inbound-sustain-rate
- max-outbound-sustain-rate

In order to set the desired time windows for computing burst rates and sustain rates, you also need to configure these parameters in the session agent configuration: burst-rate-window and sustain-rate-window. You can also set the time-to-resume and in-service-period parameters to control how long to wait before bringing a session agent back into service after its constraints are no longer exceeded.

Admission Control for Session Agents

This section explains how to configure session agents for admission control.

Session Agents Admission Control Configuration

To use admission control based on session rate, you need to configure session agent session rate constraints.

To configure session rates:

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 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. Enable session agent constraints and then configure the parameters related to session capacity or session rate to set admission control.

constraints—Enable this parameter. From here you can either configure admission control based on session capacity, session rates, or both. The default value is enabled. The valid values are:

- enabled | disabled

5. max-sessions—Set the maximum number of sessions (inbound and outbound) allowed by the session agent. The default value is zero (0). The valid range is:

- Minimum—0
- Maximum—4294967295

6. max-inbound-sessions—Enter the maximum number of inbound sessions allowed from this session agent. The default value is zero (0). The valid range is:

- Minimum—0
- Maximum—999999999

7. max-outbound-sessions—Enter the maximum number of concurrent outbound sessions (outbound from the Oracle CSM) that are allowed from this session agent. The default value is zero (0). The valid range is:

- Minimum—0
- Maximum—4294967295



Note: The number you enter here cannot be larger than the number you entered for max-sessions.

8. max-burst-rate—Enter a number to set how many SIP session invitations this session agent can send or receive (per second) within the configured burst rate window value. The default value is zero (0). The valid range is:

- Minimum—0
- Maximum—4294967295

For the sustained rate, the Oracle CSM maintains a current and previous window size. The period of time over which the rate is calculated is always between one and two window sizes.

For example, if you enter a value of 50 here and a value of 60 (seconds) for the burst rate window constraint, no more than 300 session invitations can arrive at or leave from the session agent in that 60 second time frame (window). Within that 60-second window, any sessions over the limit of 300 are rejected.

9. **max-inbound-burst-rate**—Enter the maximum burst rate (number of session invitations per second) for inbound sessions from this session agent. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999
10. **max-outbound-burst-rate**—Enter the maximum burst rate (number of session invitations per second) for outbound sessions to this session agent. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999
11. **max-sustain-rate**—Enter a number to set the maximum rate of session invitations (per second) this session agent can send or receive within the current window. The default value is zero (0). The valid range is:
 - Minimum—zero (0)
 - Maximum—4294967295


The number you enter here must be larger than the number you enter for max-burst-rate.

For the sustained rate, the Oracle CSM maintains a current and previous window size. The period of time over which the rate is calculated is always between one and two window sizes.

For example, if you enter a value of 50 here and a value of 36 (seconds) for the sustain rate window constraint, no more than 1800 session invitations can arrive at or leave from the session agent in any given 36 second time frame (window). Within that 36 second window, sessions over the 1800 limit are rejected.

12. **max-inbound-sustain-rate**—Enter the maximum sustain rate (of session invitations allowed within the current window) of inbound sessions from this session agent. This value should be larger than the max-inbound-burst-rate value. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999
13. **max-outbound-sustain-rate**—Enter the maximum sustain rate (of session invitations allowed within the current window) of outbound sessions to this session agent. This value should be larger than the max-outbound-burst-rate value. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999
14. **burst-rate-window**—Enter a number to set the burst window period (in seconds) that is used to measure the burst rate. The term window refers to the period of time over which the burst rate is computed. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—4294967295
15. **sustain-rate-window**—Enter a number to set the sustained window period (in seconds) that is used to measure the sustained rate. The default value is zero (0), which disables the functionality. The valid range is:
 - Minimum—10
 - Maximum—4294967295

The value you set here must be higher than or equal to the value you set for the burst rate window.

 **Note:** If you are going to use this parameter, you must set it to a minimum value of 10.

The following example shows session agent constraints that are enabled and the session capacity parameters have been configured. Other session agent parameters have been omitted for brevity.

Admission Control

```
session-agent
constraints          enabled
max-sessions         355
max-inbound-sessions 355
max-outbound-sessions 355
```

The following example shows session agent constraints are enabled and the session rate parameters have been configured. Other session agent parameters have been omitted for brevity.

```
session-agent
max-burst-rate       0
max-inbound-burst-rate 10
max-outbound-burst-rate 1
max-sustain-rate     3000
max-inbound-sustain-rate 0
max-outbound-sustain-rate 0
burst-rate-window    0
sustain-rate-window  0
```

Realm Bandwidth Configuration

To configure admission control based on bandwidth, you set the max and min bandwidth parameters in the realm configuration.

To configure realm bandwidth:

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

```
ORACLE# configure terminal
```

2. Type `media-manager` and press Enter.

```
ORACLE (configure) # media-manager
```

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

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

4. Configure the maximum bandwidth.

`max-bandwidth`—Enter a number that sets the maximum bandwidth for dynamic flows to/from the realm in kilobits (Kbps) per second. The default value is zero (0). The valid range is:

- Minimum—0
- Maximum—4294967295

The following example shows the maximum bandwidth for the realm has been configured. All other realm parameters have been omitted for brevity.

```
realm-config
max-bandwidth          64000
```

SIP Admission Control Configuration

You can configure the registered endpoint to accept and process requests from SIP realms. If a request does not meet the criteria of the option you choose here, it is rejected with a 403 (Forbidden) response.

To configure admission control:

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

```
ORACLE(sip-interface)# sip-port
ORACLE(sip-port)#
```

5. Set the criteria for admission control.

allow-anonymous—Enter the anonymous connection mode you want applied when SIP requests are processed. The default value is all.

The following are valid values:

- all—No ACL is applied and all anonymous connections are allowed.
- agents-only—Only requests from configured session agents are processed. The Oracle CSM responds to all other requests with a forbidden response.
- realm-prefix—Only requests from session agents and addresses matching the realm's address prefix are processed. All other requests are rejected with a 403 (Forbidden) response.
- registered—Only requests from session agents and registered endpoints are processed. REGISTER allowed from any endpoint.
- registered-prefix—Only requests from session agent and registered endpoint addresses that match the realm's realm prefix are processed.

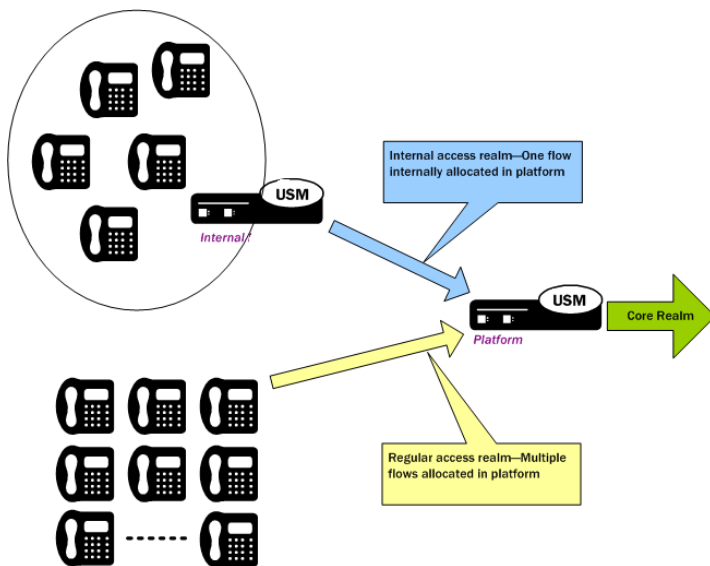
The following example shows the allow-anonymous parameter that has been configured to allow only requests from session agents and registered endpoints. All other session agent parameters following the allow-anonymous parameters are omitted for brevity.

```
sip-port
      address
      port                5060
      transport-protocol  UDP
      allow-anonymous     registered
```

Session Agent Minimum Reserved Bandwidth

You can assign session agents minimum bandwidth, applicable in access Oracle CSM deployments. Assigning a session agent minimum bandwidth can prevent overloading other network devices—such as another Oracle CSM configured as a session agent. Doing so assures signaling bandwidth and availability to the endpoints behind this Oracle CSM.

In the following diagram, the internal Oracle CSM is configured as a session agent on the platform Oracle CSM (which conveys traffic to the core realm). Setting up bandwidth reservation allows for the creation of only one allocated flow, and secures bandwidth for all the SIP clients behind the internal Oracle CSM. Contrast this scenario with the one where the platform Oracle CSM must allocate multiple flows for many SIP clients.



When you configure minimum reserved bandwidth for session agent to a non-zero value, the Oracle CSM allocates a separate pipe for per session agent. This is achieved by setting up an access control configuration in a specific way, instructing the Oracle CSM to use a minimum number of transmission timeslots the individual pipe is guaranteed to receive.

No more than 4000 session pipes are supported.

Session Agent Minimum Reserved Bandwidth Configuration

For the feature to work, you must set up an access control configuration with the settings required in the instructions and examples below.

To configure minimum reserved bandwidth for session agents:

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

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

2. Type `session-router` and press Enter.

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

3. Type `access-control` and press Enter.

```
ORACLE(session-router)# access-control
ORACLE(access-control)#
```

If you are adding this feature to an existing configuration, then you will need to select the configuration you want to edit.

4. `realm-id`—Enter the name of a valid realm.
5. `access`—Set this parameter to `permit` (default).
6. `trust-level`—Set this parameter to `high`, changing it from the default (`none`).
7. `minimum-reserved-bandwidth`—Enter the minimum reserved bandwidth you want for the session agent, and that will trigger the creation of a separate pipe for it. Only a non-zero value will allow the feature to work properly, along with the other required values set out in these instructions. The default is 0, and the maximum is `0xffffffff` (or 4294967295).
8. Save and activate your configuration.

Aggregate Session Constraints for SIP

You can set a full suite of session constraints and then apply them to a SIP interface. The session constraints configuration contains many of the same parameters as the session agent, so you can configure a group of constraints and then apply them to a SIP interface/

The SIP interface configuration's `constraint-name` parameter invokes the session constraint configuration you want to apply. Using the constraints you have set up, the Oracle CSM checks and limits traffic according to those settings for the SIP interface. Of course, if you do not set up the session constraints or you do not apply them in the SIP interface, then that SIP interface will be unconstrained. If you apply a single session-constraint element to multiple SIP interfaces, each SIP interface will maintain its own copy of the session-constraint.

SIP interfaces now have two states: "In Service" and "Constraints Exceeded." When any one of the constraints is exceeded, the status of the SIP interface changes to Constraints Exceeded and remains in that state until the time-to-resume period ends. The session constraint timers that apply to the SIP interface are the time-to-resume, burst window, and sustain window.

Aggregate Session Constraints Configuration

This section shows you how to configure aggregate session constraints and then apply them to a SIP interface.

The session constraints configuration contains many of the same parameters as the session agent does; it also incorporates the changes to the session agent parameters that are described in this section.

To configure the session constraints:

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 `session-constraints` and press Enter.

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

4. `name`—Enter the name for this session constraints configuration; this is a unique identifier that you will use in the SIP interface when you want the session constraints applied there. This is a required parameter that has no default.
5. `state`—Enable this parameter to use these session constraints. The default value is enabled. The valid values are:
 - enabled | disabled
6. `max-sessions`—Enter the maximum sessions allowed for this constraint. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999
7. `max-outbound-sessions`—Enter the maximum outbound sessions allowed for this constraint. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999
8. `max-inbound-sessions`—Enter the maximum inbound sessions allowed for this constraint. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—999999999
9. `max-burst-rate`—Enter the maximum burst rate (invites per second) allowed for this constraint. This value should be the sum of the `max-inbound-burst-rate` and the `max-outbound-burst-rate`. The default value is zero (0). The valid range is:
 - Minimum—0

- Maximum—999999999
- 10. max-sustain-rate**—Enter the maximum rate of session invitations per second allowed within the current window for this constraint. The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999
- For the sustained rate, the Oracle CSM maintains a current and previous window size. The period of time over which the rate is calculated is always between one and two window sizes.
- 11. max-inbound-burst-rate**—Enter the maximum inbound burst rate (number of session invitations per second) for this constraint. The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999
- 12. max-inbound-sustain-rate**—Enter the maximum inbound sustain rate (of session invitations allowed within the current window) for this constraint. The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999
- For the sustained rate, the Oracle CSM maintains a current and previous window size. The period of time over which the rate is calculated is always between one and two window sizes.
- 13. max-outbound-burst-rate**—Enter the maximum outbound burst rate (number of session invitations per second) for this constraint. The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999
- 14. max-outbound-sustain-rate**—Enter the maximum outbound sustain rate (of session invitations allowed within the current window) for this constraint. The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999
- For the sustained rate, the Oracle CSM maintains a current and previous window size. The period of time over which the rate is calculated is always between one and two window sizes.
- 15. time-to-resume**—Enter the number of seconds after which the SA (Session Agent) is put back in service (after the SA is taken OOS (Out Of Service) because it exceeded some constraint). The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999
- 16. ttr-no-response**—Enter the time delay in seconds to wait before the SA (Session Agent) is put back in service (after the SA is taken OOS (Out Of Service) because it did not respond to the Oracle CSM). The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999
- 17. in-service-period**—Enter the time in seconds that elapses before an element (like a session agent) can return to active service after being placed in the standby state. The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999
- 18. burst-rate-window**—Enter the time in seconds that you want to use to measure the burst rate; the window is the time over which the burst rate is calculated, and is used for the over all burst rate as well as the inbound and outbound burst rates. The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—999999999

19. `sustain-rate-window`—Enter the time in seconds used to measure the sustained rate; the window is the time over which the sustained rate is calculated, and is used for the over all sustained rate as well as the inbound and outbound sustained rates. The default value is zero (0). The valid range is:

- Minimum—0
- Maximum—999999999

Applying Session Constraints in a SIP Interfaces

In the SIP interface, there is a new parameter that allows you to use a set of session constraints for that interface; the parameter is called `constraint-name`.

To apply session constraints to a SIP interface:

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 `sip-interface` and press Enter.

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

4. `constraint-name`—Enter the name of the session constraints configuration that you want to apply to this SIP interface. There is no default for this parameter.

5. Save and activate your configuration.

Configuring CAC Policing and Marking for non-Audio non-Video Media

In the media profile and the media policy configurations, the following values have been added for the `media-type` parameter:

- `application | data | image | text`

For the media policy, these new values apply to ToS marking.

Support for the AS Bandwidth Modifier

Two new parameters have been added to the media profile configuration:

- `sdp-bandwidth`—Enable or disable the use of the AS modifier in the SDP if the `req-bandwidth` and `sdp-rate-limit-headroom` parameters are not set to valid values in a corresponding media profile. The default value is disabled. The valid values are:
 - `enabled | disabled`
- `sdp-rate-limit-headroom`—Specify the percentage of headroom to be added while using the AS bandwidth parameter while calculating the average-rate-limit (rate limit for the RTP flow). The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—100

The following conditions apply to the use and application of these two new parameters:

- If the amount of required bandwidth is not specified in the media profile (`req-bandwidth`) for the media type in the `m=` line of the SDP, then the value specified in the AS modifier is used. The Oracle CSM only uses the AS value if you set the new `sdp-bandwidth` to `enabled`.
- If the average rate limit value for RTP flows is not specified in the media profile (`average-rate-limit`) for the media type in the `m=` line of the SDP, then the value specified in the AS modifier is used. The system only uses the AS value if you set the new `sdp-bandwidth` to `enabled`. When calculating the average rate rate limit that it will use based on the AS modifier, the Oracle CSM applies the percentage set in the `sdp-rate-limit-headroom` parameter.

Admission Control

- The Oracle CSM uses the value specified in the AS modifier (if sdp-bandwidth is enabled, and req-bandwidth is set to 0) along with the user-cac-bandwidth value set in the realm configuration; this works the same way that the req-bandwidth parameter does.
- The system uses the value specified in the AS modifier (if sdp-bandwidth is enabled, and req-bandwidth is set to 0) along with the max-bandwidth value set in the realm configuration; this works the same way that the req-bandwidth parameter does.

Media Profile Configuration

To set any of the new media types in the media profile configuration:

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

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

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

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

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

4. media-type—Enter the media type that you want to use for this media profile. The valid values are:

- audio | video | application | data | image | text

5. Save and activate your configuration.

To set any of the new media types in the media policy configuration:

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

```
ORACLE# configure terminal
```

7. Type media-manager and press Enter.

```
ORACLE(configure)# media-manager
ORACLE(media-manager)#
```

8. Type media-policy and press Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

```
ORACLE(media-manager)# media-policy
ORACLE(media-policy)#
```

9. media-type—Enter the media type that you want to use for this media profile. The valid values are:

- audio | video | application | data | image | text

10. Save and activate your configuration.

AS Modifier and Headroom Configuration

To enable AS modifier use and establish the percentage of headroom to use:

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

```
ORACLE# configure terminal
```

2. Type session-router and press Enter.

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

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

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

4. sdp-bandwidth—Enable this parameter to use the AS bandwidth modifier in the SDP. The default is disabled. Valid values are:
 - enabled | disabled
5. sdp-rate-limit-headroom—Specify the percentage of headroom to be added while using the AS bandwidth parameter while calculating the average-rate-limit (rate limit for the RTP flow). The default is 0. The valid range is:
 - Minimum—0
 - Maximum—100
6. Save and activate your configuration.

CAC Utilization Statistics via SNMP

The Oracle CSM allows you to retrieve information on current session utilization and burst rate as a percentage of their configured maximums on per session-agent and/or realm basis. The Oracle CSM uses the configured max-session and max-burst-rate settings in conjunction with a percentage formula to calculate this value. The system also uses an ACLI configuration setting to establish the threshold at which trap and trap clear messages are sent from the SNMP agent to the configured manager(s).

The user must load the MIB version associated with this software version on all pertinent SNMP managers to query these CAC utilization (occupancy) values and interpret the traps. In addition, the user must configure the threshold at which the system generates the CAC utilization trap. Note that the corresponding clear trap uses the same threshold setting, sending the clear trap when utilization falls below 90% of the threshold.

SNMP Get for CAC Utilization

Using a MIB browser, the user can query the current percentage utilization values for both max-session and max-burst-rate for any session-agent or realm. The calculations for these utilization levels are:

- Session utilization level = (current session count * 100) / max-sessions
- Burst rate utilization level = (current burst rate * 100) / max-burst-rate

The MIB objects associated with these statistics are parallel for session agent and realm and include a table to contain the objects, an object associating the objects containing the values with the applicable table, and objects containing the values themselves. These objects are listed below.

The MIB objects containing CAC utilization data for Session Agents are listed below.

The object establishing the statistics table for session agent CAC utilization follows:

```
--apSip Session Agent Connection Admission Control Stats Table
apSipSaCacStatsTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF ApSipSaCacStatsEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "SIP Session Agent Connection Admission Control Stats Table."
    ::= { apSipMIBTabularObjects 5 }
```

The object establishing the session agent CAC utilization statistics objects follows:

```
apSipSaCacStatsEntry OBJECT-TYPE
    SYNTAX          ApSipSaCacStatsEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Connection Admission Control Statistics."
    AUGMENTS { apSipSessionAgentStatsEntry }
    ::= { apSipSaCacStatsTable 1 }
```

The session agent CAC utilization statistics values include:

Admission Control

```
ApSipSaCacStatsEntry ::= SEQUENCE {
    apSipSaCacSessionUtilLevel      Gauge32,
    apSipSaCacBurstRateUtilLevel    Gauge32
}
```

The above objects, specifying the CAC utilization value for sessions and burst rate utilization for session agents include:

```
apSipSaCacSessionUtilLevel      OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           "percentage"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Current session utilization level."
    ::= { apSipSaCacStatsEntry 1 }
```

```
apSipSaCacBurstRateUtilLevel    OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           "percentage"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Current burst rate utilization level."
    ::= { apSipSaCacStatsEntry 2 }
```

The MIB objects containing CAC utilization data for Realms are listed below.

The object establishing the statistics table for realm CAC utilization follows:

```
--apSig Realm Connection Admission Control Stats Table
apSigRealmCacStatsTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF ApSigRealmCacStatsEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Realm Connection Admission Control Stats Table."
    ::= { apSigMIBTabularObjects 6 }
```

The object establishing the realm CAC utilization statistics objects follows:

```
apSigRealmCacStatsEntry OBJECT-TYPE
    SYNTAX          ApSigRealmCacStatsEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Connection Admission Control Statistics."
    AUGMENTS { apSigRealmStatsEntry }
    ::= { apSigRealmCacStatsTable 1 }
```

The session agent CAC utilization statistics values include:

```
ApSigRealmCacStatsEntry ::= SEQUENCE {
    apSigRealmCacSessionUtilLevel      Gauge32,
    apSigRealmCacBurstRateUtilLevel    Gauge32
}
```

The above objects, specifying the CAC utilization value for sessions and burst rate utilization for realms include:

```
apSigRealmCacSessionUtilLevel    OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           "percentage"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Current session utilization level."
```



```

    ::= { apSigRealmCacStatsEntry 1 }

apSigRealmCacBurstRateUtilLevel          OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           "percentage"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Current burst rate utilization level."
    ::= { apSigRealmCacStatsEntry 2 }

```

CAc Utilization Traps

The Oracle CSM can issue a trap when either the value of max-session or CAC burst rate exceeds a configured value. The system only sends one trap when the threshold is exceeded. When the value falls back under 90% of this threshold, the Oracle CSM sends a clear trap.

You configure the value that triggers these traps as a percentage of the max-session and max-burst-rate settings configured for the applicable session agent and/or realm. The system uses the same setting to specify when to send both the sessions and burst rate traps. The name of this parameter is the **cac-trap-threshold**.

For realms, you configure a **session-constraint** element with the **cac-trap-threshold** setting and apply that session constraint to the realm. For a session agent however, you configure the **cac-trap-threshold** directly within the session agent's configuration.

The syntax for the command is the same within session constraint and session agent configurations.

cac-trap-threshold[0-99]

You must express the value as a number less than 100. There is no default setting; the system does not generate a trap if you have not configured this setting.

The apSipCACUtilAlertTrap identifies the threshold exceeded on a per-element and per-value (session count or burst rate) for each trap, including:

- apSipSaCacSessionUtilLevel
- apSipSaCacBurstRateUtilLevel
- apSipRealmCacSessionUtilLevel
- apSipRealmCacBurstRateUtilLevel

CAc utilization threshold trap on a session agent configuration

The CAC utilization threshold causes the system to generate a trap when session count or CAC max-burst-rate exceeds the configured percentage value of these values maximums. This setting is available within a session agent's configuration.

To configure the CAC trap threshold on a session agent, follow the procedure below.

1. Access the **session-agent** configuration element.

```

ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# session-agent
ORACLE(session-agent)

```

2. Select the **session-agent** object to edit.

```

ORACLE(session-agent)# select
<hostname>:
1: 192.168.100.101:1813

selection: 1
ORACLE(session-agent)#

```

Admission Control

3. **cac-trap-threshold**—Set the threshold when reached, expressed as a percentage of **max-sessions**, when the CAC trap is sent.
4. Type **done** to save your configuration.

Configuring the CAC Utilization Thresholds - realm

To configure the CAC trap threshold on a realm or sip interface, create a session constraint object and apply it to your realm, as shown below.

1. Use the following sequence to navigate to session constraint elements.

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

2. Select or create the desired session constraint element.

```
ORACLE(session-constraints)#name trap-at-90-percent
```

3. Configure the desired value for cac-trap-threshold expressed as a percentage value, such as 90%, as follows.

```
ORACLE(session-constraints)#cac-trap-threshold 90
```

4. Navigate to the realm-config to which you want to apply the session constraint.

```
ORACLE(realm-config)#session-constraint trap-at-90-percent
```

5. Execute the done and exit commands.
6. Save and activate your configuration.

Whitelists for SIP

Oracle CSM by default ignores and passes-through unknown SIP headers and URI parameters. However, some operators require that the Oracle CSM only accept messages with headers and URI parameters complying with those supported by their internal equipment. This section describes the use of whitelists to control unknown headers and parameters in request and response traffic.

What is a Whitelist

A whitelist is an approved list of entities for which equipment provides particular privileges, access, and recognition. The Oracle CSM can use configured whitelist profiles to control and accept specific inbound SIP headers and URI parameters that are being passed-through the Oracle CSM. When you configure this feature, the Oracle CSM rejects requests not matching the configured profile, or removes the unspecified headers or URI parameters not in the configured profile.

Whitelists Configuration

You can configure whitelist profiles or rules that allow the Oracle CSM to only accept inbound SIP headers and URI parameters that are configured in this whitelist, using the parameter `allowed-elements-profile`. You can configure the settings for this parameter using the CLI interface at `session-router>enforcement-profile`. Since the `enforcement-profile` object also pertains to session agents, realms, and SIP interfaces, you can also apply the profiles you configure to these remote entities using the CLI interface at `session-router>session-agent`, `session-router>sip-interface`, and `media-manager>realm-config`.

In the following configuration example, it is assumed that your baseline configuration passes SIP traffic, with the Oracle CSM in the role of an Access SBC. Use this procedure to configure a whitelist for the session router and optionally apply the specific whitelists to the session agent and SIP interface, as well as the media manager's realm configuration.

To configure a whitelist for the session router:

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-related configurations.

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

3. Type allowed-elements-profile and press Enter to access the enforcement profile-related parameters.

```
ORACLE(session-router)# allowed-elements-profile
ORACLE(allowed-elements-profile)#
```

4. Type allowed-elements-profile and press Enter to access the whitelist-related parameters. The system prompt changes to let you know that you can begin configuring individual parameters for this object.

```
ORACLE(enforcement-profile)# allowed-elements-profile
ORACLE(allowed-elements-profile)#
```

5. name — Enter a unique name for the whitelist you are creating. This name can be referenced when configuring the enforcement-profiles for session-agent, SIP interface, and realm-config.

```
ORACLE(allowed-elements-profile)# name whitelist1
```

6. description — Enter a description that explains the purpose of creating this whitelist. You can use any alphanumeric characters when entering the description.

```
ORACLE(allowed-elements-profile)# description Basic Whitelist
```

7. Type rule-sets and press Enter to specify the rules to match against specific incoming SIP headers and/or URI parameters. The system prompt changes to let you know that you can begin configuring individual parameters for this object.

```
ORACLE(allowed-elements-profile)# rule-sets
ORACLE(rule-sets)#
```

8. unmatched-action — Select the action for the Oracle CSM to perform when a header does not exist in an incoming message. The default value is reject. The valid values are:

- reject — Rejects all incoming messages that do not contain a header.
- delete — Deletes all incoming message that do not contain a header.



Note: This parameter applies to non-matching header names only (not non-matching URI parameters).

```
ORACLE(rule-sets)# unmatched action delete
```

9. msg-type — Specify the type of messages for which the Oracle CSM applies this whitelist configuration. The default value is any. The valid values are:

- any — Applies to all incoming messages.
- request — Applies to only incoming REQUEST messages.
- response — Applies to only incoming RESPONSE messages.

```
ORACLE(rule-sets)# msg-type any
```

10. methods — Enter the packet method(s), separated by a comma, for which this whitelist is enforced. Packet methods include, INVITE, OPTIONS, ACK, BYE, etc. If this field is left blank, the whitelist applies to all packet methods. You can enter up to a maximum of 255 characters.

```
ORACLE(rule-sets)# methods INVITE,ACK,BYE
```

11. logging — Select whether or not an incoming message is written to a log file called matched.log when the message contains an element not specified in the whitelist. The default value is disabled. The valid values are:

- enabled | disabled

```
ORACLE(rule-sets)# logging enabled
```

The matched.log contains information about the timestamp, received/sent Oracle CSM network-interface, IP address/port from which it was received or being sent from, and which peer IP address/port it was received from or sent to. The log also specifies the request URI (if applicable), and the From, To, and Contact headers in the

message, as well as which rule triggered the log action. An example of the log output of the matched.log file is as follows:

```
Dec 17 14:17:54.526 On [0:0]192.168.1.84:5060 sent to 192.168.1.60:5060
allowed-elements-profile[whitelist1(reject)]
INVITE sip:service@192.168.1.84:5060 SIP/2.0
From: sipp <sip:+2125551212@192.168.1.60:5060>;tag=3035SIPpTag001
To: sut <sip:service@192.168.1.84>
Contact: sip:sipp@192.168.1.60:5060
```

The header-rule object consists of 6 parameters that make up the header-rule:

- header-name
- unmatched-action
- allow-header-param
- allow-uri-param
- allow-uri-user-param
- allow-uri-header-name

You can configure an unlimited number of header-rules on the Oracle CSM that you can apply to your network. Use the following parameters to configure a header-rule that the Oracle CSM uses to control which incoming messages it allows.

12. header-rule — This object allows you to configure, as part of the whitelist, multiple parameters which make up the header rule that the Oracle CSM allows from incoming messages. Header-rules do NOT have to be in any specific order. The system prompt changes to let you know that you can begin configuring individual parameters for this object.


```
ORACLE(rule-set) # header-rule
ORACLE(allowed-header-rule) #
```

13. header-name — Enter the name of the header in the whitelist that the Oracle CSM allows from incoming messages. It is case-insensitive and supports abbreviated forms of header names. For example, “Via”, “via”, or “v” all match against the same header. A header name of “request-uri” refers to the request URI of requests, while a header name of * applies to any header-type not matched by any other header-rule. The default value is *. This default value provides the ability to have header-rules for commonly known headers that remove unknown parameters, but leave unknown headers alone.

```
ORACLE(allowed-header-rule) # header-name Contact
```

14. unmatched-action — Select the action for the Oracle CSM to perform when an incoming header’s parameters do not match the relevant allowed parameters specified for this header-name. The default value is reject. The valid values are:

- reject — Rejects all incoming messages that have header parameters that do not match the parameters specified in this header-name.
- delete — Deletes all incoming messages that have header parameters that do not match the parameters specified in this header-name.

 **Note:** This parameter applies to non-matching header names only (not non-matching URI parameters).

```
ORACLE(allowed-header-rule) # unmatched-action delete
```

15. allow-header-param — Enter the header parameter that the Oracle CSM allows from the headers in incoming messages. You can enter up to 255 characters, including a comma (,), semi-colon (;), equal sign (=), question mark (?), at-symbol (@), backslash (\), or plus sign (+). The default value is *, which allows all header parameters to pass through. If you leave this field empty, no header parameters are allowed.

```
ORACLE(allowed-header-rule) # allow-header-param *
```

16. allow-uri-param — Enter the URI parameter that the Oracle CSM allows from the headers in incoming messages. You can enter up to 255 characters, including a comma (,), semi-colon (;), equal sign (=), question mark (?), at-symbol (@), backslash (\), or plus sign (+). The default value is *, which allows all URI parameters to pass through. If you leave this field empty, no URI parameters are allowed.

```
ORACLE(allowed-header-rule)# allow-uri-param *
```

17. `allow-uri-user-param` — Enter the URI user parameter that the Oracle CSM allows from the headers in incoming messages. You can enter up to 255 characters, including a comma (,), semi-colon (;), equal sign (=), question mark (?), at-symbol (@), backslash (\), or plus sign (+). The default value is *, which allows all URI user parameters to pass through. If you leave this field empty, no URI user parameters are allowed.

```
ORACLE(allowed-header-rule)# allow-uri-user-param *
```

18. `allow-uri-header-name` — Enter the URI header name that the Oracle CSM allows from the headers in incoming messages. You can enter up to 255 characters, including a comma (,), semi-colon (;), equal sign (=), question mark (?), at-symbol (@), backslash (\), or plus sign (+). The default value is *, which allows all URI header name parameters to pass through. If you leave this field empty, no URI header name parameters are allowed.

```
ORACLE(allowed-header-rule)# allow-uri-header-name *
```

19. Save your work using the ACLI `done` command.

Configuration Exception

There are specific instances in incoming Request-URI messages where the Oracle CSM ignores specific parameters and automatically adds header-rules.

In a Request-URI, all parameters are URI parameters, and URI headers are not allowed. If you define values for the “`allow-header-param`”, “`allow-uri-header-name`”, and “`allow-uri-param`”, the Oracle CSM ignores these parameters in the Request-URI. Instead the Oracle CSM automatically adds header-rules for incoming “Via”, “From”, “To”, “Call-ID”, and “CSeq” messages. These are explicit header rules and cannot be deleted. Each header-rule in a Request-URI has parameters populated with the value of *. If required, a user can change the header-rule parameter values with the values identified in the following table.

Header Rule	Applicable Parameter	Required Value(s)
Via	<code>allow-header-param</code>	<ul style="list-style-type: none"> branch received rport
From	<code>allow-header-param</code>	<ul style="list-style-type: none"> tag
To	<code>allow-header-param</code>	<ul style="list-style-type: none"> tag
Call-ID	<code>allow-header-param</code>	No restrictions
CSeq	<code>allow-header-param</code>	No restrictions

Verify Whitelist Configuration

After you have configured and saved a whitelist on the Oracle CSM, you can use the existing `verify-config` command at the top level prompt to verify the saved configuration: For example:

```
ORACLE# verify-config
```

This command checks for errors in the Oracle CSM configuration. Whitelist configuration errors, specifically related to the enforcement-profile object, also display in the output of this command if applicable. The whitelist configuration error(s) display if any references to the allowed-element-profiles are improperly configured. If an error(s) exist, the following message displays:

```
-----
ERROR: enforcement-profile [ep] contains a reference to an allowed-
enforcement-profile [abc] that does not exist
-----
```

How Whitelists Work

Whitelists allow you to customize which SIP signaling messages to allow into your network and which messages to reject or delete. In the flow of SIP traffic to/from the Oracle CSM, the Oracle CSM matches any received request or response, in or out of a dialog against the configured allowed list, and rejects or deletes the non-matching element based on the actions specified in the whitelist configuration.

For responses, the Oracle CSM does not reject the message if a header or parameter is not found in the allowed list, even if the action is set to reject. Instead it deletes the offending parameter or header. In addition, if the message is a request of the method type ACK, PRACK, CANCEL or BYE, it deletes all unmatched elements, but does not reject the request, even if the action was configured to reject.

The whitelist verification performs for any method; however you can narrow this list to operate only on specific methods by defining them in the methods parameter of the configuration.

Whitelist verification occurs when a request or response is received by the Oracle CSM, but only after the Oracle CSM has processed the inbound header manipulation rule (HMR), network management controls (NMC), Resource-Priority header (RPH), and monthly-minutes checking.

The Oracle CSM responds to requests which have non-matching headers or parameters configured with an action of reject, with a "403 Forbidden" response by default. You can use a local-response event, allowed-elements-profile-rejection, to override the default reject status code and reason phrase.

The configured whitelist operates transparently on headers that have multiple URIs or multiple header values within a single header (header values separated by a comma).


Parameter parsing operates only on parameters that it can identify. For parameters that can not be parsed, for example an invalid URI (e.g. < sip:user@host.com&hp=val>), the Oracle CSM ignores this URI header parameter value of "hp" since it is not contained within a valid URI. Even though it would appear to be a URI header parameter, URI headers must come after URI parameters. Parameter matching does not occur if the headers and parameters in the URI are not well-formed. The Oracle CSM does not remove the parameter since it cannot identify it.

Whitelist Learning

You can build your whitelist configuration based on the learning capabilities of the Oracle CSM. When you enable the Oracle CSM learning mode, it acquires the knowledge of the allowable elements (headers and parameters) currently incoming to your network. The Oracle CSM collects the information about the headers received and the parameters that exist within each header. The information continues to be gathered until you disable the learning mode.

Once you disable the learning mode, the Oracle CSM prompts you to enter a name for the allowed-elements-profile. If the profile name you entered does not exist, the captured information is written to the new allowed-elements-profile configuration. The administrator can then make changes to the configuration as applicable, save the configuration, and apply it to a logical remote entity.

The new allowed-elements-profile does not contain any wildcard rules. The Oracle CSM cannot generate wildcard headers and parameters during the learning mode. The Methods object is populated from the list of methods seen by the Oracle CSM while learning.

 **Note:** Oracle recommends running the learning mode during off-peak and/or light traffic times. This mode can operate in conjunction with the execution of an allowed-elements-profile. The learning occurs just before any configured allowed-elements-profile configuration.

Whitelist Learning Configuration

The ACLI interface provides two commands that allow a Superuser to start and stop whitelist learning on the Oracle CSM:

Command	Description
start <argument> <options>	Starts whitelist learning on the Oracle CSM.

Command	Description
	<p>You must specify the argument learn-allowed-elements with this command to start the learning operation.</p> <p>Optionally, you can use method, msg-type, and params after the argument.</p>
stop <argument> <identifier>	<p>Stops the whitelist learning on the Oracle CSM and writes the learned configuration to the editing configuration on the Oracle CSM where it is saved and activated.</p> <p>You must specify the argument learn-allowed-elements with this command to stop the learning operation.</p> <p>You must specify a unique identifier that identifies the allowed-elements-profile name.</p> <p>If you specify an identifier name that already exists as a profile, the ACLI returns an error message and prompts you to enter a different name.</p>

You can use these commands at the top level ACLI prompt as required on the Oracle CSM.

You use these commands with the argument, learn-allowed-elements to start/stop the whitelist learning feature. By default, the learning mode creates a single rule-set under which all of the headers and their respective parameters are stored.

For example:

```
ORACLE# start learn-allowed-elements
Learning mode for allowed-elements-profile started.
```

In the above example, "start" is the top level ACLI command and learn-allowed-elements is the operation being performed.

Optionally, you can specify [method], [msg-type], and [params] in any order, for the Oracle CSM to learn specific rule-set elements from incoming messages and save them to the whitelist configuration.

For example:

```
ORACLE# start learn-allowed-elements method msg-type params
```

The method option creates a new rule-set per unique method. The "msg-type" option creates a new rule-set per unique message-type seen. The "params" option performs URI and header parsing to examine parameters within the message. By default, parameter parsing is disabled.

To start the whitelist learning feature:

In Superuser mode, at the top level ACLI prompt, type start learn-allowed-elements and press Enter.

```
ORACLE# start learn-allowed-elements
```

The following message displays:

```
Learning mode for allowed-elements-profile started.
```


To specify the elements of rule-sets for whitelists:

In Superuser mode, at the top level ACLI prompt, type start learn-allowed-elements method msg-type params and press Enter.

```
ORACLE# start learn-allowed-elements method msg-type params
```

The following message displays:

```
Learning mode for allowed-elements-profile started.
```

 **Note:** If you try to start a whitelist learning operation while another learning operation is already running, the following message displays:

Admission Control

```
Learning mode restarted without saving
Learning mode for allowed-elements-profile started.
```

To stop the whitelist learning feature:

In Superuser mode, at the top level ACLI prompt, type `stop learn-allowed-elements <identifier>`, where `<identifier>` is the allowed-elements-profile name, and press Enter.

```
ORACLE# stop learn-allowed-elements whitelist1
```

The following message displays:

```
Learning mode for allowed-elements-profile stopped.
```

If you specify an identifier name that already exists as a profile, the ACLI returns an error message and prompts you to enter a different name.

Rejected Messages Monitoring

Whitelists, when configured on the Oracle CSM, control whether or not the Oracle CSM allows unknown headers and URI parameters to be accepted in incoming request and response traffic. When the Oracle CSM rejects messages according to the whitelist, the rejected messages are logged to a file called “matched.log” if logging is set to enabled. You can open and view the log when required to view the rejected messages.

In addition to the rejected messages being logged to the “matched.log” file, the rejected messages are also sent through a burst counter that keeps track of the amount of messages rejected. You can enter the `show sipd` to display the number of rejected messages. The counter is titled Rejected Message.

High Availability Nodes

Oracle CSM s can be deployed in pairs to deliver high availability (HA). Two Oracle CSM s operating in this way are called an HA node. Over the HA node, media and call state are shared, keeping sessions/calls from being dropped in the event of a failure.

Two Oracle CSM s work together in an HA node, one in active mode and one in standby mode.

- The active Oracle CSM checks itself for internal process and IP connectivity issues. If it detects that it is experiencing certain faults, it will hand over its role as the active system to the standby Oracle CSM in the node.
- The standby Oracle CSM is the backup system, fully synchronized with active Oracle CSM s session status. The standby Oracle CSM monitors the status of the active system so that, if needed, it can assume the active role without the active system having to instruct it to do so. If the standby system takes over the active role, it notifies network management using an SNMP trap.

In addition to providing instructions for how to configure HA nodes and their features, this chapter explains how to configure special parameters to support HA for all protocols.

Overview

To produce seamless switchovers from one Oracle CSM to the other, the HA node uses shared virtual MAC and virtual IP addresses for the media interfaces in a way that is similar to VRRP (virtual router redundancy protocol). When there is a switchover, the standby Oracle CSM sends out gratuitous ARP messages using the virtual MAC address, establishing that MAC on another physical port within the Ethernet switch. To the upstream router, the MAC and IP are still alive, meaning that existing sessions continue uninterrupted.

Within the HA node, the Oracle CSM s advertise their current state and health to one another in checkpointing messages; each system is apprised of the other's status. Using Oracles HA protocol, the Oracle CSM s communicate with UDP messages sent out and received on the rear interfaces.

The standby Oracle CSM shares virtual MAC and IPv4 addresses for the media interfaces (similar to VRRP) with the active Oracle CSM . Sharing addresses eliminates the possibility that the MAC and IPv4 address set on one Oracle CSM in an HA node will be a single point of failure. The standby Oracle CSM sends ARP requests using a utility IPv4 address and its hard-coded MAC addresses to obtain Layer 2 bindings.

The standby Oracle CSM assumes the active role when:

- It has not received a checkpoint message from the active Oracle CSM for a certain period of time.
- It determines that the active Oracle CSM 's health score has decreased to an unacceptable level.
- The active Oracle CSM relinquishes the active role.

Establishing Active and Standby Roles

Oracle CSM s establish active and standby roles in the following ways.

- If a Oracle CSM boots up and is alone in the network, it is automatically the active system. If you then pair a second Oracle CSM with the first to form an HA node, then the second system to boot up will establish itself as the standby automatically.
- If both Oracle CSM s in the HA node boot up at the same time, they negotiate with each other for the active role. If both systems have perfect health, then the Oracle CSM with the lowest HA rear interface IPv4 address will become the active Oracle CSM . The Oracle CSM with the higher HA rear interface IPv4 address will become the standby Oracle CSM .
- If the rear physical link between the two Oracle CSM s fails during boot up or operation, both will attempt to become the active Oracle CSM . In this case, processing will not work properly.

Health Score

HA Nodes use health scores to determine their active and standby status. Health scores are based on a 100-point system. When a Oracle CSM is functioning properly, its health score is 100.

Generally, the Oracle CSM with the higher health score is active, and the Oracle CSM with the lower health score is standby. However, the fact that you can configure health score thresholds builds some flexibility into using health scores to determine active and standby roles. This could mean, for example, that the active Oracle CSM might have a health score lower than that of the standby Oracle CSM , but a switchover will not take place because the active Oracle CSM 's health score is still above the threshold you configured.

Alarms are key in determining health score. Some alarms have specific health score value that are subtracted from the Oracle CSM 's health score when they occur. When alarms are cleared, the value is added back to the Oracle CSM 's health score.

You can look at a Oracle CSM 's health score using the ACLI show health command.

Switchovers

A switchover occurs when the active Oracle CSM stops being the active system, and the standby system takes over that function. There are two kinds switchovers: automatic and manual.

Automatic Switchovers

Automatic switchovers are triggered without immediate intervention on your part. Oracle CSM s switch over automatically in the following circumstances:

- When the active Oracle CSM 's health score of drops below the threshold you configure.
- When a time-out occurs, meaning that the active Oracle CSM has not has not sent checkpointing messages to the standby Oracle CSM within the allotted time.

The active Oracle CSM might not send checkpointing messages for various reasons such as link failure, communication loss, or advertisement loss. Even if the active Oracle CSM has a perfect health score, it will give up the active role if it does not send a checkpoint message or otherwise advertise its status within the time-out window. Then the standby Oracle CSM takes over as the active system.

When an automatic switchover happens, the Oracle CSM that has just become active sends an ARP message to the switch. This message informs the switch to send future messages to its MAC address. The Oracle CSM that has just become standby ignores any messages sent to it.

Manual Switchovers


You can trigger a manual switchover in the HA node by using the ACLI notify berpd force command. This command forces the two Oracle CSM s in the HA node to trade roles. The active system becomes standby, and the standby becomes active.

In order to perform a successful manual switchover, the following conditions must be met.

- The Oracle CSM from which you trigger the switchover must be in one of the following states: active, standby, or becoming standby.
- A manual switchover to the active state is only allowed on a Oracle CSM in the standby or becoming standby state if it has achieved full media, signaling, and configuration synchronization.
- A manual switchover to the active state is only allowed on a Oracle CSM in the standby or becoming standby state if it has a health score above the value you configure for the threshold.

State Transitions

Oracle CSM s can experience series of states as they become active or become standby.

 **Note:** Packet processing only occurs on an active Oracle CSM .

State	Description
Initial	When the Oracle CSM is booting.
Becoming Active	When the Oracle CSM has negotiated to become the active system, but is waiting the time that you set to become fully active. Packets cannot be processed in this state.
Active	When the Oracle CSM is handling all media, signaling, and configuration processing.
Relinquishing Active	When the Oracle CSM is giving up its Active status, but before it has become standby. This state is very brief.
Becoming Standby	When the Oracle CSM is becoming the standby system but is waiting to become fully synchronized. It remains in this state for the period of time you set in the becoming-standby-time parameter, or until it is fully synchronized.
Standby	When the Oracle CSM is fully synchronized with its active system in the HA node.
OutOfService	When the Oracle CSM cannot become synchronized in the period of time you set in the becoming-standby-time parameter.

State Transition Sequences

When the active Oracle CSM assumes its role as the as the active system, but then changes roles with the standby Oracle CSM to become standby, it goes through the following sequence of state transitions:

- Active
- RelinquishingActive
- BecomingStandby
- Standby

When the standby Oracle CSM assumes its role as the standby system, but then changes roles with the active Oracle CSM to become active, it goes through the following sequence of state transitions:

- Standby
- BecomingActive
- Active

HA Features

HA nodes support configuration checkpointing, which you are required to set up so that the configurations across the HA node are synchronized. In addition, you can set up the following optional HA node features:

- Multiple rear interface support
- Gateway link failure detection and polling

Multiple Rear Interfaces

Configuring your HA node to support multiple rear interfaces eliminates the possibility that either of the rear interfaces you configure for HA support will become a single point of failure. Using this feature, you can configure individual Oracle CSM s with multiple destinations on the two rear interfaces, creating an added layer of failover support.

When you configure your HA node for multiple rear interface support, you can use last two rear interfaces (wancom1 and wancom2) for HA—the first (wancom0) being used for Oracle CSM management. You can connect your Oracle CSM s using any combination of wancom1 and wancom2 on both systems. Over these rear interfaces, the Oracle CSM s in the HA node share the following information:

- Health
- Media flow
- Signaling
- Configuration

For example, if one of the rear interface cables is disconnected or if the interface connection fails for some other reason, all health, media flow, signaling, and configuration information can be checkpointed over the other interface.

Health information is checkpointed across all configured interfaces. However, media flow, signaling, and configuration information is checkpointed across one interface at a time, as determined by the Oracle CSM 's system HA processes.

Configuration Checkpointing

During configuration checkpointing, all configuration activity and changes on one Oracle CSM are automatically mirrored on the other. Checkpointed transactions include adding, deleting, or modifying a configuration on the active Oracle CSM . This means that you only need to perform configuration tasks on the active Oracle CSM because the standby system will go through the checkpointing process and synchronize its configuration to reflect activity and changes.

Because of the way configuration checkpointing works, the ACLI save-config and activate-config commands can only be used on the active Oracle CSM .

- When you use the ACLI save-config command on the active Oracle CSM , the standby Oracle CSM learns of the action and updates its own configuration. Then the standby Oracle CSM saves the configuration automatically.
- When you use the ACLI activate-config command on the active Oracle CSM , the standby Oracle CSM learns of the action and activates its own, updated configuration.

The ACLI acquire-config command is used to copy configuration information from one Oracle CSM to another.

Gateway Link Failure Detection and Polling


In an HA node, the Oracle CSM s can poll for and detect media interface links to the gateways as they monitor ARP connectivity. The front gateway is assigned in the network interface configuration, and is where packets are forwarded out of the originator's LAN.

The Oracle CSM monitors connectivity using ARP messages that it exchanges with the gateway. The Oracle CSM sends regular ARP messages to the gateway in order to show that it is still in service; this is referred to as a heartbeat message. If the Oracle CSM deems the gateway unreachable for any of the reasons discussed in this section, a network-level alarm is generated and an amount you configure for this fault is subtracted from the system's health score.


The Oracle CSM generates a gateway unreachable network-level alarm if the Oracle CSM has not received a message from the media interface gateway within the time you configure for a heartbeat timeout. In this case, The Oracle CSM will send out ARP requests and wait for a reply. If no reply is received after resending the set number of ARP requests, the alarm remains until you clear it. The health score also stays at its reduced amount until you clear the alarm.

When valid ARP requests are once again received, the alarm is cleared and system health scores are increased the appropriate amount.

You can configure media interface detection and polling either on a global basis in the SD HA nodes/redundancy configuration or on individual basis for each network interface in the network interface configuration.

-  **Note:** To improve the detection of link failures, the switchport connected to the NIU should have Spanning Tree disabled. Enabling Spanning Tree stops the switchport from forwarding frames for several seconds after a reset. This prevents the NIU from reaching the gateway and generates a "gateway unreachable" network-level alarm.

Before Configuring a High Availability (HA) Pair

-  **Note:** When you configure an HA pair, you must use the same password for both Oracle CSMs.


Before configuring the parameters that support HA, complete the following steps.

1. Establish the physical connections between the Oracle CSMs. Avoid breaking the physical link (over the rear interfaces) between the Oracle CSMs in an HA node. If the physical link between the Oracle CSMs breaks, they will both attempt to become the active system and HA will not function as designed.
2. Confirm that both Oracle CSMs are set to the same time. Use the ACLI show clock command to view the system time. If the Oracle CSMs show different times, use the system-timeset command to change the time.

Oracle recommends that you use NTP to synchronize your Oracle CSMs, so that they have a common stratum time source.

3. HA nodes use ports 1 and 2 as the HA interfaces. Set port 0 on the rear panel of the Oracle CSM chassis as the boot and management interface. You configure the rear interfaces during the physical interface configuration.
4. For ACLI configuration, you need to know the target names of the Oracle CSMs making up the HA node. The target name of the system is reflected in the ACLI's system prompt. For example, in the ORACLE# system prompt, ORACLE is the target name.

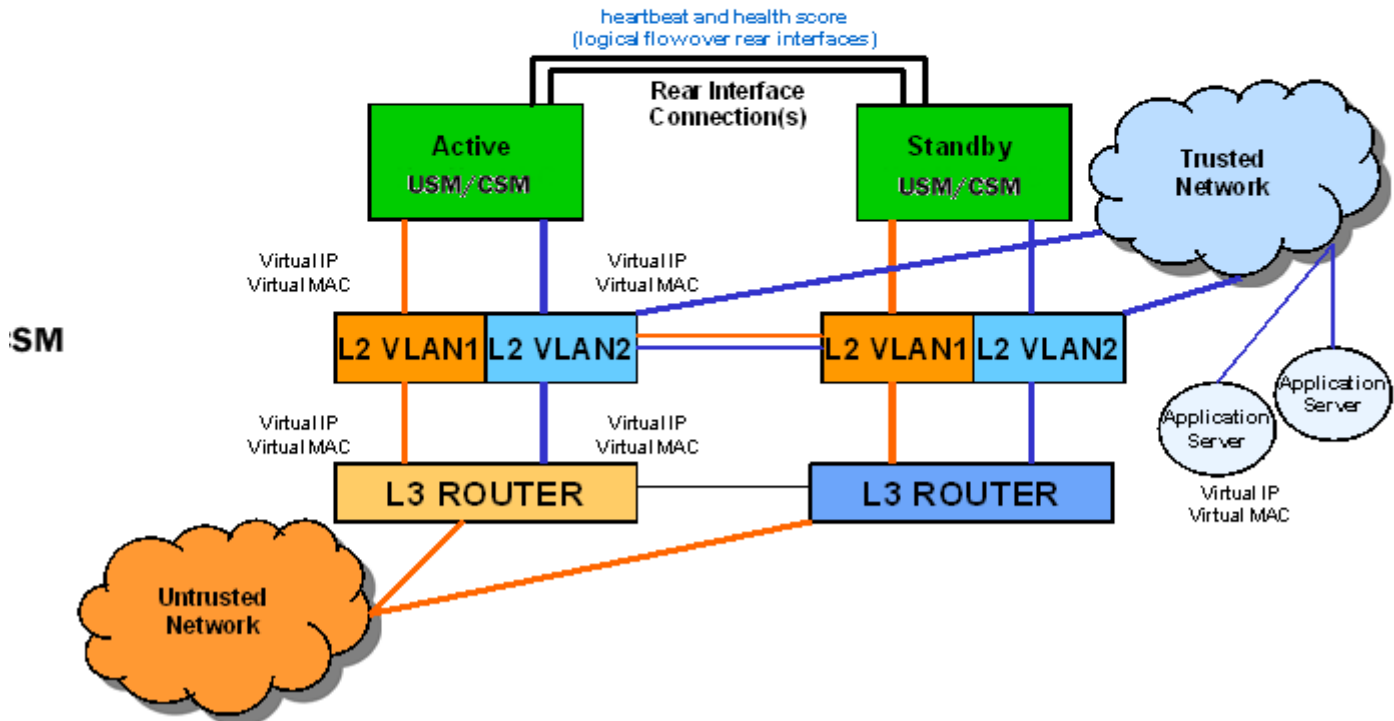
You can also see and set the target name in the Oracle CSM boot parameters.

-  **Note:** The target name is case sensitive.
5. Devise virtual MAC addresses so that, if a switchover happens, existing sessions are not interrupted.

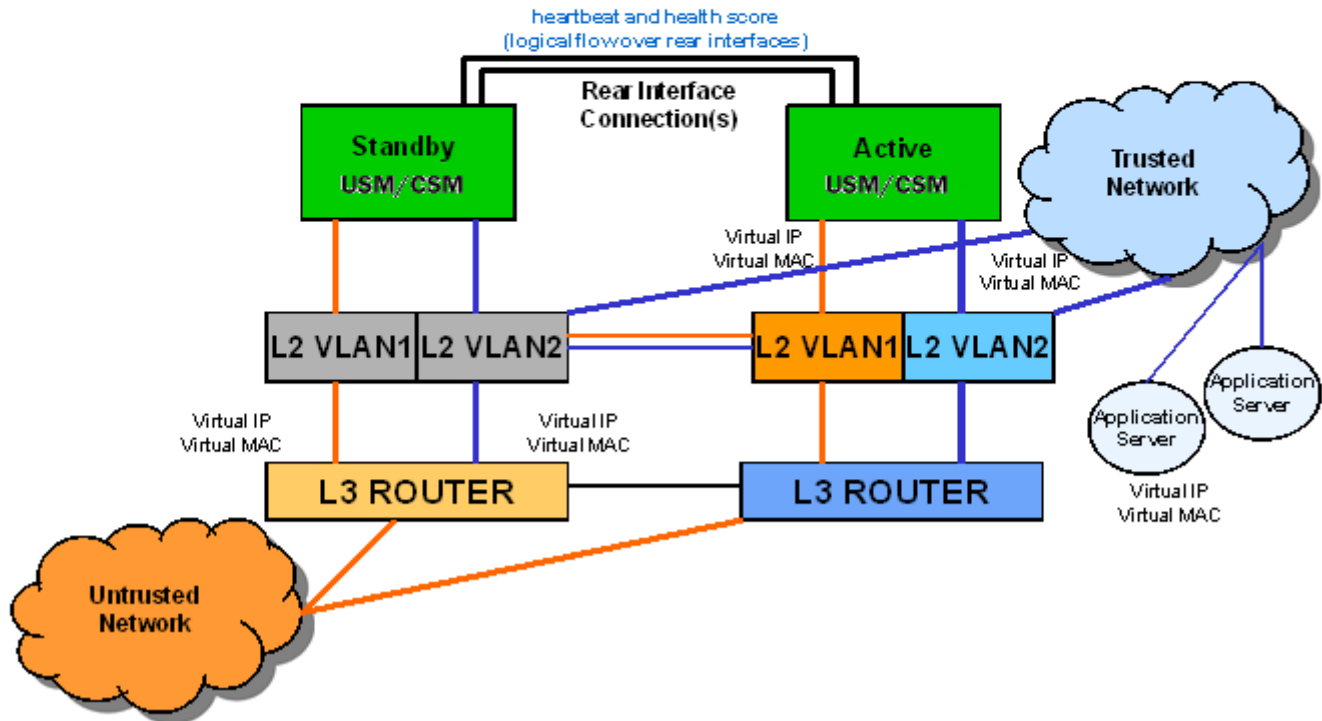
HA Node Connections


To use HA, you must establish Layer 2 and Layer 3 networks that interconnect two Oracle CSMs and support HA with the required physical network connections. The basic network set-up in the following diagram shows an HA node deployment where each system is connected to its own Layer 2 switch. This set-up provides a measure of added redundancy in the event that one of the switches fails.

Here, the active system is using the virtual MAC and IP addresses.



In the second diagram, the same network is shown with the HA node having experienced a switchover. The previously standby Oracle CSM has taken over the active role in the HA node and is using the virtual IP and MAC addresses.



 **Note:** Switches should never be in master-slave mode. If they are, HA will not work correctly.

The following are hardware set-up and location considerations for placing an HA Node:

- You must set up each Oracle CSM according to the requirements and safety precautions set out in the *Oracle Communications System Hardware Installation Guide*.
- Each Oracle CSM’s media interfaces must be connected to the same switches (or other network entities), as shown in the diagram above.
- The length of the shielded crossover 10/100 category 5 Ethernet cable that connects the Oracle CSMs from the rear interfaces must be able to reach from the configured rear interface on one Oracle CSM to the configured rear interface on the other.

HA nodes use Oraclerdr element redundancy protocol for its tasks. This protocol uses a connection between the rear interfaces of two Oracle CSMs to checkpoint the following information: health, state, media flow, signaling, and configuration.

We recommend that you use shielded category 5 (RJ45) crossover cables for all 10/100 Ethernet connections used for HA.

You can set up either single or multiple rear interface support for your HA node. For single interface support, one cable connects the two Oracle CSMs; for multiple interface support, two cables are used. However, the software configurations for each type of connection mode are different.

When you make these connections, do not use port 0 (wancom0) on the rear interface of the Oracle CSM chassis; that port should only be used for Oracle CSM management. Instead, use ports 1 and 2 (wancom1 and wancom2).

High Availability Nodes

To cable Oracle CSMs using single rear interface support:

1. Using a 10/100 category 5 crossover cable, insert one end into either port 1 (wancom1) or port 2 (wancom2) on the rear interface of the first Oracle CSM.
2. Insert the other end of the cable into port 1 or port 2 on the rear interface of the second Oracle CSM. We recommend that you use corresponding ports on the two systems. That is, use port 1 on both systems or use port 2 on both systems.
3. Perform software configuration for these interfaces as described in this chapter.

To cable Oracle CSMs using multiple rear interface support:

4. Using a 10/100 category 5 crossover cable, insert one end into port 1 on the rear interface of the first Oracle CSM.
5. Insert the other end of that cable into port 1 on the rear interface of the second Oracle CSM to complete the first physical connection.
6. Using a second 10/100 category 5 cable, insert one end into port 2 on the rear interface of the first Oracle CSM.
7. Insert the other end of this second cable in port 2 on the rear interface of the second Oracle CSM to complete the second physical connection.
8. Perform software configuration for these interfaces as described in this chapter.

Virtual MAC Addresses

In order to create the HA node, you need to create virtual MAC addresses for the media interfaces. You enter these addresses in virtual MAC address parameters for physical interface configurations where the operation type for the interface is media.

The HA node uses shared virtual MAC (media access control) and virtual IP addresses for the media interfaces. When there is a switchover, the standby Oracle CSM sends out an ARP message using the virtual MAC address, establishing that MAC on another physical port within the Ethernet switch. Virtual MAC addresses are actually unused MAC addresses that based on the Oracle CSM's root MAC address.

The MAC address is a hardware address that uniquely identifies each Oracle CSM. Given that, the virtual MAC address you configure allows the HA node to appear as a single system from the perspective of other network devices. To the upstream router, the MAC and IP are still alive, meaning that existing sessions continue uninterrupted through the standby Oracle CSM.

Depending on the type of physical layer cards you have installed, you can create MAC addresses as follows: Four Ethernet (MAC) address for each configured four-port GigE physical interface card.

Virtual MAC Address Configuration

To create a virtual MAC address:

1. Determine the Ethernet address of the Oracle CSM by using the ACLI `show interfaces` command. This command only works if you have already set up physical interface configurations. Otherwise, you will get no output.

The example below shows you where the Ethernet address information appears; this sample has been shortened for the sake of brevity. For each type of physical interface card, the Oracle CSM displays the following:

```
ORACLE# show interfaces
f00 (media slot 0, port 0)
  Flags: UP BROADCAST MULTICAST ARP RUNNING
  Type: GIGABIT_ETHERNET
  Admin State: enabled
  Auto Negotiation: enabled
  Internet address: 10.10.0.10      Vlan: 0
  Broadcast Address: 10.10.255.255
  Netmask: 0xffff0000
  Gateway: 10.10.0.1
  Ethernet address is 00:08:25:01:07:64
```

2. Identify the root portion of the Ethernet (MAC) address.

Each Oracle CSM has MAC addresses assigned to it according to the following format: 00:08:25:XX:YY:ZN where:

- 00:08:25 refers to Acme Packet
- XX:YY:ZN refers to the specific Oracle CSM
- N is a 0-f hexadecimal value available for the Oracle CSM

In this example, the root part of this address is 00:08:25:XX:YY:Z.

3. To create an unused MAC address (that you will use as the virtual MAC address) take the root MAC address you have just identified. Replace this N value with unused hexadecimal values for the Oracle CSM: 8, 9, e, or f.

In other words, you change the last digit of the MAC address to either 8, 9, e, or f depending on which of those address are not being used.

For example, for an HA node with MAC address bases of 00:08:25:00:00:00 and 00:08:25:00:00:10, the following addresses would be available for use at virtual MAC addresses:

- 00:08:25:00:00:08
- 00:08:25:00:00:09
- 00:08:25:00:00:0e
- 00:08:25:00:00:0f
- 00:08:25:00:00:18
- 00:08:25:00:00:19
- 00:08:25:00:00:1e
- 00:08:25:00:00:1f

Corresponding media interfaces in HA nodes must have the same virtual MAC addresses. Given that you have various physical interface card options, the following points illustrate how virtual MAC address can be shared:

If you are using a four-port GigE physical interface card, both the active Oracle CSM and the standby Oracle CSM might have the following virtual MAC address scheme for the slots:

- Slot 0 - 00:08:25:00:00:0e and 00:08:25:00:00:0f
- Slot 1 - 00:08:25:00:00:1e and 00:08:25:00:00:1f



Note: Note the virtual MAC addresses you have created so that you can reference them easily when you are configuring the physical interfaces for HA.

HA Node Connections

You can begin software configuration for your HA node after you have:

- Completed the steps for physical set-up and connection.
- Noted the target name of the Oracle CSMs that make up the HA node.
- Configured the virtual MAC addresses that you need, according to the type of physical interface cards installed on your Oracle CSM.

HA Node Connection Configuration

If you are using HA, you need to set the physical interface configuration parameters described in this section to establish successful connections. These parameters are for rear and media interfaces.

To access physical interface menu in the ACLI:

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
```

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

```
ORACLE(system)# phy-interface
ORACLE(phy-interface)#
```

From this point, you can configure physical interface parameters. To view all physical interfaces parameters, enter a ? at the system prompt.

Rear Interfaces

You can use port 1 (wancom1) or port 2 (wancom2) as interfaces to support HA. Do not use port 0 (wancom 0) as that port is reserved for carrying management traffic.

Make sure that the physical connections you have made on the rear panel of your Oracle CSMs correspond to the configurations you enter for physical interfaces. You can connect Oracle CSMs through multiple rear interfaces. For multiple rear interface connectivity, cable both port 1 and port 2 (wancom1 and wancom2) on one Oracle CSM to port1 and port 2 on the other Oracle CSM in the HA node.

The Oracle CSM's HA function depends heavily on health scores to determine the active and standby roles in an HA node. You can set the amount that will be subtracted from a Oracle CSM's health score in the event that a management interface fails for any reason. For example, a connection might become invalid or a cable might be removed inadvertently.

The following example shows how a configured physical interface will appear in the ACLI for an HA node:

```
phy-interface
  name                wancom1
  operation-type      Control
  port                1
  slot                0
  virtual-mac
  wancom-health-score 20
```

To establish rear interfaces for use in an HA node using the ACLI:

1. Access the physical interface menu.
2. name—Set a name for the interface using any combination of characters entered without spaces. For example: wancom1.
3. operation-type—Set this parameter to Control.
4. slot—Set this parameter to 0.
5. port—Set this parameter to 1 or 2.
6. wancom-health-score—Enter the number value between 0 and 100. This value will be subtracted from the Oracle CSM's health score in the event that a rear interface link fails. We recommend that you change this value from its default (50), and set it to 20.

This value you set here is compared to the active and emergency health score thresholds you establish in the Oracle CSM HA node (redundancy) configuration.

7. For multiple rear interface support, configure the remaining, unused rear interfaces with the appropriate values.

The following example shows configuration for multiple rear interface support.

```
ORACLE(system)# phy-interface
ORACLE(phy-interface)# name wancom1
ORACLE(phy-interface)# operation-type control
ORACLE(phy-interface)# port 1
ORACLE(phy-interface)# wancom-health-score 20
ORACLE(phy-interface)# done
ORACLE(phy-interface)# name wancom2
ORACLE(phy-interface)# operation-type control
ORACLE(phy-interface)# port 2
ORACLE(phy-interface)# wancom-health-score 20
ORACLE(phy-interface)# done
```

Media Interface Virtual MAC Addresses

To configure HA for the media interfaces in an HA node, you must set one or more virtual MAC addresses, according to the type of physical layer cards you have installed on your Oracle CSM.

To set a virtual MAC address using the ACLI:

1. Access the physical interface configuration.
2. Configure all relevant parameters as noted in the Physical Interfaces section of this guide's *System Configuration* chapter.

Since virtual MAC addresses are used for media interfaces only, verify that the operation type is set to media.

3. virtual-mac—Enter the virtual MAC address that you have created using the steps in the Virtual MAC Addresses section.

HA Node Parameters

To establish a pair of Oracle CSMs as an HA node, you need to configure basic parameters that govern how the Oracle CSMs:

- Transition on switchover
- Share media and call state information
- Checkpoint configuration data

The following example shows what an HA configuration might look like in the ACLI.

```

redundancy-config
  state                enabled
  log-level            WARNING
  health-threshold    75
  emergency-threshold 50
  port                 9090
  advertisement-time   500
  percent-drift        210
  initial-time         1250
  becoming-standby-time 45000
  becoming-active-time 100

```

You need to configure the two Oracle CSMs to be HA node peers. To enable configuration checkpointing, you must to configure two peers in the ACLI, one for the primary and one for the secondary Oracle CSM. The HA node peers configuration also allows you to configure destinations for where to send health and state information. Unless you create Oracle CSM peers and destinations configurations, HA will not work properly.

The following example shows what an HA configuration might look like in the ACLI.

```

peer
  name                netnetsd1
  state                enabled
  type                Primary
  destination
    address            169.254.1.1:9090
network-interface    wancom1:0
peer
  name                netnetsd2
  state                enabled
  type                Secondary
  destination
    address            169.254.1.2:9090
    network-interface  wancom1:0

```

HA Node Parameter Configuration

To configure general HA node parameters using the ACLI:

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

High Availability Nodes

```
ORACLE# configure terminal
```

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

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

3. Type redundancy and press Enter.

```
ORACLE(system)# redundancy
```

From here, you configure basic HA node parameters. To view all basic HA node parameters, enter a ? at the system prompt.

4. state—Leave this parameter set to enabled for HA to work. To stop HA operation, set this parameter to disabled. The default value is enabled. The valid values are:
 - enabled | disabled
5. log-level—Set the log level you want to use for the HA system process. The value you set in this field overrides any log level value you set for the entire Oracle CSM in the system configuration process log level parameter. The default value is INFO which allows you to receive a moderate amount of detail. The valid values are:
 - emergency | critical | major | minor | warning | notice | info | trace | debug | detail
6. health-threshold—Enter a value between 0 and 100 to set the health score at which the Oracle CSMs in the HA node gracefully exchange active-standby roles. The default value is 75. The valid range is:
 - Minimum—1
 - Maximum—100

For example, if this field is set to 75 and the active Oracle CSM's health score falls below that point, the standby Oracle CSM will take over the active role. However, Oracle CSM will only take over the active role if its own health score is 75 or better.

7. emergency-threshold—Enter the health score for the standby Oracle CSM to become active immediately. The default value is 50. The valid range is:
 - Minimum—0
 - Maximum—100

If the standby Oracle CSM is initializing and the active Oracle CSM's health score is below the health threshold, the standby Oracle CSM will take the active role and there will be a graceful switchover. If the active Oracle CSM's health score is below the emergency threshold, then the switchover will be immediate.

If the standby Oracle CSM has a health score below the emergency threshold and the active Oracle CSM is unhealthy, the active Oracle CSM will not give up its active role.

8. advertisement-time—Enter the number of milliseconds to set how often Oracle CSMs in an HA node inform each other of their health scores.

We recommend you leave this parameter set to its default, 500. The valid range is:

- Minimum—50
 - Maximum—999999999
9. percent-drift—Enter the percentage of the advertisement time that you want one member of the HA node to wait before considering the other member to be out of service. For the standby Oracle CSM, this is the time it will wait before taking the active role in the HA node. The default value is 210. The valid range is:
 - Minimum—100
 - Maximum—65535
 10. initial-time—Enter the number of milliseconds to set the longest amount of time the Oracle CSM will wait at boot time to change its state from initial to either becoming active or becoming standby. The default value is 1250. The valid range is:
 - Minimum—5
 - Maximum—999999999

11. `becoming-standby-time`—Enter the number of milliseconds the Oracle CSM waits before becoming standby, allowing time for synchronization. If it is not fully synchronized within this time, it will be declared out of service.

We recommend that you do not set this parameter below 45000. If a large configuration is being processed, we recommend setting this parameter to 180000 to allow enough time for configuration checkpointing. The default value is 45000. The valid range is:

- Minimum—5
- Maximum—999999999

12. `becoming-active-time`—Enter the number of milliseconds that the standby Oracle CSM takes to become active in the event that the active Oracle CSM fails or has an intolerably decreased health score. The default value is 100. The valid range is:

- Minimum—5
- Maximum—999999999

HA Node Peer Configuration

To configure a Oracle CSM as an HA node peer:

1. From the redundancy menu, type `peers` and press Enter.

```
ORACLE(system)# redundancy
ORACLE(redundancy)# peers
```


2. `state`—Enable or disable HA for this Oracle CSM. The default value is enabled. The valid values are:

- enabled | disabled

3. `name`—Set the name of the HA node peer as it appears in the target name boot parameter.

This is also the name of your system that appears in the system prompt. For example, in the system prompt `ORACLE1#`, `ORACLE1` is the target name for that Oracle CSM.

4. `type`—These values refer to the primary and secondary utility addresses in the network interface configuration. To determine what utility address to use for configuration checkpointing, set the type of Oracle CSM: primary or secondary.

 **Note:** You must change this field from unknown, its default. The valid values are:

- primary—Set this type if you want the Oracle CSM to use the primary utility address.
- secondary—Set this type if you want the Oracle CSM to use the secondary utility address.
- unknown—If you leave this parameter set to this default value, configuration checkpointing will not work.

HA Node Health And State Configuration

To configure where to send health and state information within an HA node:

1. From the peers configuration, type `destinations` and press Enter.

```
ORACLE(rdncy-peer)# destinations
ORACLE(rdncy-peer-dest)#
```

2. `address`—Set the destination IPv4 address and port of the other Oracle CSM in the HA node to which this Oracle CSM will send HA-related messages. This value is an IPv4 address and port combination that you enter as: `IPAddress:Port`. For example, `169.254.1.1:9090`.

- The IPv4 address portion of this value is the same as the IPv4 address parameter set in a network interface configuration of the other Oracle CSM in the HA node.
- The port portion of this value is the port you set in the Oracle CSM HA Node/redundancy configuration for the other Oracle CSM in the node.

3. `network-interface`—Set the name and subport for the network interface where the Oracle CSM receives HA-related messages. Valid names are `wancom1` and `wancom2`. This name and subport combination must be entered as `name:subport`; for example, `wancom1:0`.

High Availability Nodes

The network interface specified in this parameter must be linked to a physical interface configured with rear interface parameters. The physical interface's operation type must be control or maintenance, and so the subport ID portion of this parameter is 0. The subport ID is the VLAN tag.

Synchronizing Configurations

You can synchronize the Oracle CSMs (CSM) in your High Availability (HA) node in the following ways:

- Automatically — Set up configuration checkpointing within the HA node.
- Manually — Check whether or not configurations in the HA node are synchronized, and then copy configuration data from one CSM to the other.

When you initially configure a new HA node, copy the configuration data manually from one CSM to the other. When you complete the process, you can configure your HA node to automatically synchronize configurations.

Oracle recommends that you configure the HA node for configuration checkpointing because that is the most reliable way to ensure that both systems have the same configuration.

Synchronize HA Peers

The process for synchronizing the peers in a High Availability (HA) node for the first time by way of the ACLI includes the following steps.

1. Create a complete configuration on the active Oracle CSM (CSM). Include all HA node parameters and all rear interface configurations. Confirm that the rear interfaces are configured to send and receive information across the HA node.
2. On the active CSM, save the configuration.
3. On the active CSM, reboot to run the new configuration.

Use the ACLI `show health` command to see that the active CSM booted without a peer. This changes after you copy the configuration to the standby CSM and activate the configuration.

4. On the standby CSM, perform the ACLI `acquire-config` command to copy the configuration from the active CSM. Use the `acquire-config` command with the IPv4 address of `wancom 0` on the active CSM.

```
ACMEPACKET2# acquire-config 192.168.12.4
```

The IPv4 address of `wancom 0` on the active CSM is the IPv4 address portion of the value displayed for the `inet on ethernet boot` parameter. The following codeblock shows an example of the `inet on ethernet` value that the system displays when you view the boot parameters:

```
inet on ethernet (e) : 192.168.12.4:ffff0000
```

5. When the copying process (`acquire-config`) is complete, reboot the standby CSM to activate the configuration. The system boots and displays start-up information.
6. Confirm that the HA node synchronized the configurations by using the ACLI `display-current-cfg-version` and `display-running-cfg-version` commands:

```
ORACLE# display-current-cfg-version
Current configuration version is 3
ORACLE# display-running-cfg-version
Running configuration version is 3
ORACLE# display-current-cfg-version
Current configuration version is 3
ORACLE# display-running-cfg-version
Running configuration version is 3
```

In the preceding example, all configuration versions—current and running—are the same number (3).

Using Configuration Checkpointing

The Oracle CSM's primary and secondary utility addresses support configuration checkpointing, allowing the standby Oracle CSM to learn configuration changes from the active Oracle CSM. This means that you only have to enter configuration changes on the active Oracle CSM for the configurations across the HA node to be updated.

Configuration checkpointing uses parameters in the network interface and in the Oracle CSM HA Nodes/redundancy configurations.

If you are using configuration checkpointing, you also need to set up two Oracle CSM peer configurations: one the primary, and one for the secondary.

HA Configuration Checkpointing

You need to first set applicable network interface configuration parameters, and then establish applicable parameters in the Oracle CSM HA node (redundancy) configuration.

We recommend that you do not change the configuration checkpointing parameters in the redundancy configuration. Using the defaults, this feature will function as designed.

 **Note:** Remember to set the appropriate type parameter in the HA node redundancy peers configuration.

For the network interface, these parameters appear as they do in the following example when you use the ACLI. This example has been shortened for the sake of brevity.

```
pri-utility-addr      169.254.1.1
sec-utility-addr     169.254.1.2
```

For the Oracle CSM HA node (redundancy) configuration, these parameters appear as they do in the following example when you use the ACLI. This example has been shortened for the sake of brevity. You should not change these values without consultation from Oracle Technical Support or your Oracle Systems Engineer.

```
cfg-port             1987
cfg-max-trans        10000
cfg-sync-start-time  5000
cfg-sync-comp-time   1000
```

To configure HA configuration checkpointing in the ACLI:

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
```

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

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

From here, you can configure network interface parameters. To view all network interfaces parameters, enter a `?` at the system prompt.

4. `pri-utility-addr`—Enter the utility IP address for the primary HA peer in an HA architecture.

This address can be any unused IP address within the subnet defined for the network interface. For example, given a network interface of with the IPv4 address 168.0.4.15/24 (identifying the host associated with the network interface), the possible range of unused IPv4 addresses is 168.0.4.1 to 168.0.4.254. Your network administrator will know which IPv4 addresses are available for use.

5. `sec-utility-addr`—Enter the utility IP address for the secondary Oracle CSM peer in an HA architecture.

Usually, this IP address is usually the next in the sequence up from the primary utility address. It is also generated from the range of unused IP addresses within the subnet defined for the network interface.


6. Save your work and exit the network interface configuration.

High Availability Nodes

```
ORACLE(network-interface) # done
ORACLE(network-interface) # exit
ORACLE(system) #
```

7. Access the system HA node/redundancy configuration by typing redundancy at the system prompt and then press Enter.

```
ORACLE(system) # redundancy
ORACLE(redundancy) #
```

 **Note:** We strongly recommend that you keep the default settings for the parameters Steps 8 through 11.

8. `cfg-port`—Enter the port number for sending and receiving configuration checkpointing messages. Setting this to zero (0) disables configuration checkpointing. The default value is 1987. The valid values are:
 - Minimum—0, 1025
 - Maximum—65535
9. `cfg-max-trans`—Enter the number of HA configuration checkpointing transactions that you want to store. The active Oracle CSM maintains the transaction list, which is acquired by the standby Oracle CSM. Then the standby system uses the list to synchronize its configuration with active system. The default value is 10000. The valid range is:
 - Minimum—0
 - Maximum—4294967295

Transactions include: modifications, additions, and deletions. If the maximum number of stored transactions is reached, the oldest transactions will be deleted as new transactions are added.

10. `cfg-sync-start-time`—Enter the number of milliseconds before the Oracle CSM tries to synchronize by using configuration checkpointing. On the active Oracle CSM, this timer is continually reset as the Oracle CSM checks to see that it is still in the active role. If it becomes standby, it waits this amount of time before it tries to synchronize.

We recommend you leave this field at its default value, 5000, so that configuration checkpointing can function correctly. The valid range is:

- Minimum—0
- Maximum—4294967295

11. `cfg-sync-comp-time`—Enter the number of milliseconds that the standby Oracle CSM waits before checkpointing to obtain configuration transaction information after the initial checkpointing process is complete.

We recommend you leave this field at its default value, 1000, so that configuration checkpointing can function correctly. The valid range is:

- Minimum—0
- Maximum—4294967295

12. Save your work and exit the redundancy configuration.

```
ORACLE(redundancy) # done
ORACLE(redundancy) # exit
ORACLE(system) #
```

Manually Checking Configuration Synchronization

You can check that the current and active configurations are synchronized across the HA node. The current configuration is the one with which you are currently working, and the active configuration is the one active on the system.

To confirm that the systems in the HA node have synchronized configurations:

1. On the active Oracle CSM in the Superuser menu, enter the following ALCI commands and press Enter. Note the configuration version numbers for comparison with those on the standby Oracle CSM.

- `display-current-cfg-version`—Shows the version number of the configuration you are currently viewing (for editing, updating, etc.).

```
ORACLE# display-current-cfg-version
Current configuration version is 30
```

- `display-running-cfg-version`—Shows the version number of the active configuration running on the Oracle CSM.

```
ORACLE# display-running-cfg-version
Running configuration version is 30
```

2. On the standby Oracle CSM, enter the following ALCI commands and press Enter. Note the configuration version numbers for comparison with those on the active Oracle CSM.

```
ORACLE# display-current-cfg-version
Current configuration version is 30
ORACLE# display-running-cfg-version
Running configuration version is 30
```

3. Compare the configuration numbers. If the version numbers on the active Oracle CSM match those on the standby Oracle CSM, then the systems are synchronized.

If the version numbers do not match, you need to synchronize the Oracle CSMs. You can do so using the ACLI `acquire-config` command.

Media Interface Link Detection and Gateway Polling

You can use media interface link detection and gateway polling globally on the Oracle CSM, or you can override those global parameters on a per-network-interface basis.

- Use the Oracle CSM HA node (redundancy) configuration to establish global parameters. When configured globally, they will appear like this in the ACLI:

```
gateway-heartbeat-interval    0
gateway-heartbeat-retry      0
gateway-heartbeat-timeout    1
gateway-heartbeat-health     0
```

- Use the network interface's gateway heartbeat configuration to override global parameters on a per-network-interface basis. When configured for the network interface, these parameters will appear like this in the ACLI:

```
gw-heartbeat
      state           enabled
      heartbeat      0
      retry-count     0
      retry-timeout  1
      health-score    0
```

Media Interface Link Detection and Gateway Polling Configuration

To configure global media interface link detection and gateway polling:

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
```

3. Type `redundancy` and press Enter.

```
ORACLE(system)# redundancy
```

From here, you can configure gateway heartbeat parameters. To view all gateway heartbeat parameters, enter a `?` at the system prompt.

High Availability Nodes

4. gateway-heartbeat-interval—Enter the number of seconds between heartbeats for the media interface gateway. Heartbeats are sent at this interval as long as the media interface is viable. The default value is 0. The valid range is:
 - Minimum—0
 - Maximum—65535
5. gateway-heartbeat-retry—Enter the number of heartbeat retries (subsequent ARP requests) to send to the media interface gateway before it is considered unreachable. The default value is 0. The valid range is:
 - Minimum—0
 - Maximum—65535
6. gateway-heartbeat-timeout—Enter the heartbeat retry time-out value in seconds. The default value is 1. The valid range is:
 - Minimum—0
 - Maximum—65535

This parameter sets the amount of time between Oracle CSM ARP requests to establish media interface gateway communication after a media interface gateway failure.
7. gateway-heartbeat-health—Enter the amount to subtract from the Oracle CSM’s health score if a media interface gateway heartbeat fails. If the value you set in the gateway time-out retry field is exceeded, this amount will be subtracted from the system’s overall health score. The default value is 0. The valid range is:
 - Minimum—0
 - Maximum—100

Media Interface Link Detection and Gateway Polling Configuration 2

To configure media interface link detection and gateway polling on a per-network-interface basis in the CLI:

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

```
ACMEPACKET# configure terminal
```

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

```
ACMEPACKET(configure)# system
```

3. Type network-interface and press Enter.

```
ACMEPACKET(system)# network-interface
```

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

```
ACMEPACKET(network-interface)# gw-heartbeat
ACMEPACKET(gw-heartbeat)#
```

From here, you can configure gateway heartbeat parameters for the network interface. To view all gateway heartbeat parameters, enter a ? at the system prompt.

5. state—Enable or disable the gateway heartbeat feature. The default value is enabled. The valid values are:
 - enabled | disabled
6. heartbeat—Enter the number of seconds between heartbeats for the media interface gateway. Heartbeats are sent at this interval as long as the media interface is viable. The default value is zero (0). The valid range is:
 - Minimum—0
 - Maximum—65535

The value you configure in this field overrides any globally applicable value set in the gateway heartbeat interval parameter in the Oracle CSM HA node (redundancy) configuration.
7. retry-count—Enter the number of heartbeat retries that you want sent to the media interface gateway before it is considered unreachable. The default value is zero (0). The valid range is:
 - Minimum—0

- Maximum—65535
8. `retry-timeout`—Enter the heartbeat retry time-out value in seconds. The default value is 1. The valid range is:
- Minimum—1
 - Maximum—65535

This parameter sets the amount of time between system ARP requests to establish media interface gateway communication after a media interface gateway failure.

9. `health-score`—Enter the amount to subtract from the system’s health score if a media interface gateway heartbeat fails; this parameter defaults to 0. If the value you set in the `retry-time-out` field is exceeded, this amount will be subtracted from the system’s overall health score. The default value is zero (0). The valid range is:
- Minimum—0
 - Maximum—100

HA Media Interface Keepalive

In an HA node, it is possible for the two systems in the node to lose communication via the management (rear, wancom) interfaces. For example, wancom 1 and wancom 2 might become disconnected, and cause the heartbeat synchronization to fail. This type of failure causes communication errors because both systems try to assume the active role and thereby access resources reserved for the active system.

To avoid these types of conditions, you can enable an option instructing the standby system to take additional time before going to the active state. This check occurs through the system’s media interfaces. Using it, the standby can determine whether or not there has been a true active failure.

In cases when the standby determines the active system has not truly failed, it will go out of service because it will have determined it no longer has up-to-date data from its active counterpart. You can restore functionality by re-establishing management (rear) interface communication between the system in the node, and then re-synchronizes the standby by rebooting it.

When you enable the media interface keepalive, the standby system in the HA node sends ARP requests to determine if the media interfaces’ virtual IP address are active. There are two possible outcomes:

- If it receives responses to its ARP requests, the standby takes itself out of service—to prevent a conflict with the active.
- If it does not receive responses to its ARP requests within a timeout value you set, then standby assumes the active role in the HA node.

Impact to Boot-Up Behavior

With the HA media interface keepalive enabled, the Oracle CSM might be in the initial state longer than if the feature were disabled because it requires more information about the media (front) interfaces.

HA Media Interface Keepalive Configuration

You turn the HA media interface keepalive on by setting a timeout value for the standby to receive responses to its ARP requests before it assumes the active role in the HA node. Keeping this parameter set to 0, its default, disables the keepalive

To enable the HA media interface keepalive:

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

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

2. Type `system` and press Enter.

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

High Availability Nodes

3. Type redundancy and press Enter.

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

If you are adding this feature to an existing configuration, then you will need to select the configuration you want to edit.

4. `media-if-peercheck-time`—Enter the amount of time in milliseconds for the standby system in an HA node to receive responses to its ARP requests via the media interface before it takes over the active role from its counterpart.

The default is 0, which turns the HA media interface keepalive off. The maximum value is 500 milliseconds.

5. Save and activate your configuration.

RTC Notes

Starting in Release 4.1, the HA configuration is supported for real-time configuration (RTC). However, not all of the HA-related parameters are covered by RTC because of the impact on service it would cause to reconfigure these parameters dynamically.

This section sets out what parameters you should not dynamically reconfigure, or should dynamically reconfigure with care.

HA

Changes to the following ACLI parameters will have the noted consequences when dynamically reconfigured:

- `cfg-max-trans`—Changing this value could cause the activation time to lengthen slightly
- `init-time`, `becoming-standby-time`, and `becoming-active-time`—Changes take place only if the system is not transitioning between these states; otherwise the system waits until the transition is complete to make changes
- `percent-drift` and `advertisement-time`—Changes are communicated between nodes in the HA pair as part of regular health advertisements

In addition, the following parameters are not part of the RTC enhancement, for the reason specified in the right-hand column.

Parameter	Impact
<code>state</code>	Disrupts service
<code>port</code>	Disrupts service; leaves systems in an HA node without a means of communicating with each other
<code>cfg-port</code>	Disrupts service; leaves systems in an HA node without a means of communicating with each other
<code>cfg-max-trans</code>	Disrupts service
<code>cfg-sync-start-time</code>	Disrupts configuration replication
<code>cfg-sync-comp-time</code>	Disrupts configuration replication

Protocol-Specific Parameters and RTC

In addition, you should not change any of the parameters related to HA that are part of protocol or media management configurations that are used for protocol/media checkpointing. These are:

- SIP configuration
 - `red-max-trans`
 - `red-sync-start-time`

- red-sync-comp-time
- MGCP Configuration
 - red-mgcp-port
 - red-max-trans
 - red-sync-start-time
 - red-sync-comp-time
- Media Manager configuration
 - red-flow-port
 - red-mgcp-port
 - red-max-trans
 - red-sync-start-time
 - red-sync-comp-time

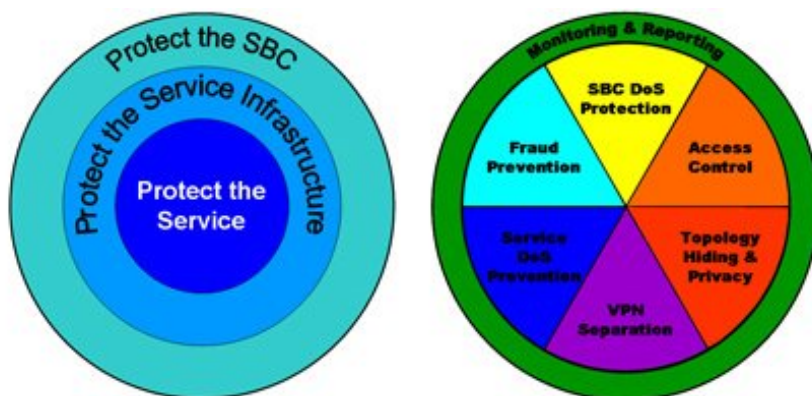
Security

This chapter explains Oracle CSM security, which is designed to provide security for administrative security, VoIP and other multimedia services. It includes Admin Security, access control, DoS attack, and overload protection, which help secure service and protect the network infrastructure (including the Oracle CSM). In addition, Oracle CSM security lets legitimate users still place calls during attack conditions; protecting the service itself.

Security Overview

Oracle CSM security includes the Net-SAFE framework's numerous features and architecture designs. Net-SAFE is a requirements framework for the components required to provide protection for the Session Border Controller (SBC), the service provider's infrastructure equipment (proxies, gateways, call agents, application servers, and so on), and the service itself.

The following diagrams illustrate Net-SAFE:



Each of Net-SAFE's seven functions consists of a collection of more specific features:

- Session border controller DoS protection: autonomic, SBC self-protection against malicious and non-malicious DoS attacks and overloads at Layers 2 to 4 (TCP, SYN, ICMP, fragments, and so on) and Layers 5 to 7 (SIP signaling floods, malformed messages, and so on).
- Access control: session-aware access control for signaling and media using static and dynamic permit/deny access control lists (ACLs) at layer 3 and 5.
- Topology hiding and privacy: complete infrastructure topology hiding at all protocol layers for confidentiality and attack prevention security. Also, modification, removal or insertion of call signaling application headers and fields. Includes support for the SIP Privacy RFC.

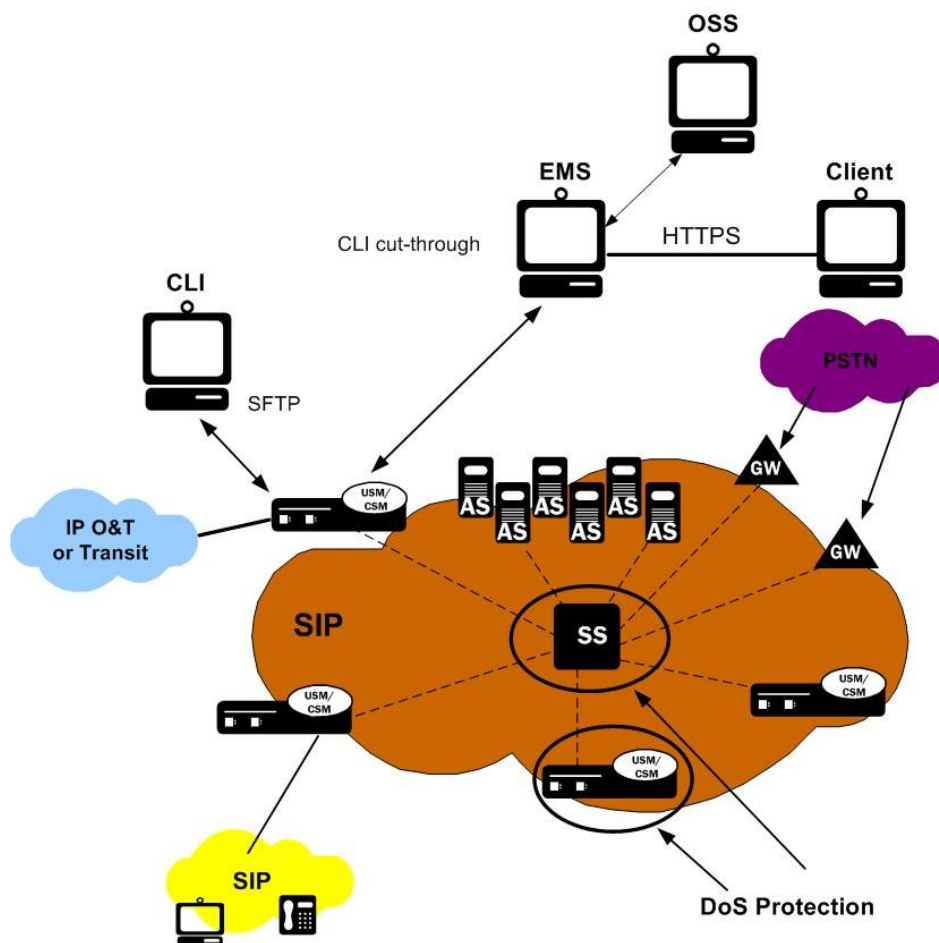
- VPN separation: support for Virtual Private Networks (VPNs) with full inter-VPN topology hiding and separation, ability to create separate signaling and media-only VPNs, and with optional intra-VPN media hair-pinning to monitor calls within a VPN.
- Service infrastructure DoS prevention: per-device signaling and media overload control, with deep packet inspection and call rate control to prevent DoS attacks from reaching service infrastructure such as SIP servers, softswitches, application servers, media servers or media gateways.
- Fraud prevention: session-based authentication, authorization, and contract enforcement for signaling and media; and service theft protection.
- Monitoring and reporting: audit trails, event logs, access violation logs and traps, management access command recording, Call Detail Records (CDRs) with media performance monitoring, raw packet capture ability and lawful intercept capability. The monitoring method itself is also secured, through the use of SSH and SFTP, and through the ability to use a separate physical Ethernet port for management access.

Denial of Service Protection

This section explains the Denial of Service (DoS) protection for the Oracle CSM. The Oracle CSM DoS protection functionality protects softswitches and gateways with overload protection, dynamic and static access control, and trusted device classification and separation at Layers 3-5. The Oracle CSM itself is protected from signaling and media overload, but more importantly the feature allows legitimate, trusted devices to continue receiving service even during an attack. DoS protection prevents the Oracle CSM host processor from being overwhelmed by a targeted DoS attack from the following:

- IP packets from an untrusted source as defined by provisioned or dynamic ACLs
- IP packets for unsupported or disabled protocols
- Nonconforming/malformed (garbage) packets to signaling ports
- Volume-based attack (flood) of valid or invalid call requests, signaling messages, and so on.
- Overload of valid or invalid call requests from legitimate, trusted sources

The following diagram illustrates DoS protection applied to the softswitch and to the Oracle CSM.



Levels of DoS Protection

The multi-level Oracle CSM DoS protection consists of the following strategies:

- Fast path filtering/access control: access control for signaling packets destined for the Oracle CSM host processor as well as media (RTP) packets. The Oracle CSM performs media filtering by using the existing dynamic pinhole firewall capabilities. Fast path filtering packets destined for the host processor require the configuration and management of a trusted list and a deny list for each Oracle CSM realm (although the actual devices can be dynamically trusted or denied by the Oracle CSM based on configuration). You do not have to provision every endpoint/device on the Oracle CSM, but instead retain the default values.
- Host path protection: includes flow classification, host path policing and unique signaling flow policing. Fast path filtering alone cannot protect the Oracle CSM host processor from being overwhelmed by a malicious attack from a trusted source. The host path and individual signaling flows must be policed to ensure that a volume-based attack will not overwhelm the Oracle CSM's normal call processing; and subsequently not overwhelm systems beyond it.

The Oracle CSM must classify each source based on its ability to pass certain criteria that is signaling- and application-dependent. At first each source is considered untrusted with the possibility of being promoted to fully trusted. The Oracle CSM maintains two host paths, one for each class of traffic (trusted and untrusted), with different policing characteristics to ensure that fully trusted traffic always gets precedence.

- Host-based malicious source detection and isolation – dynamic deny list. Malicious sources can be automatically detected in real-time and denied in the fast path to block them from reaching the host processor.

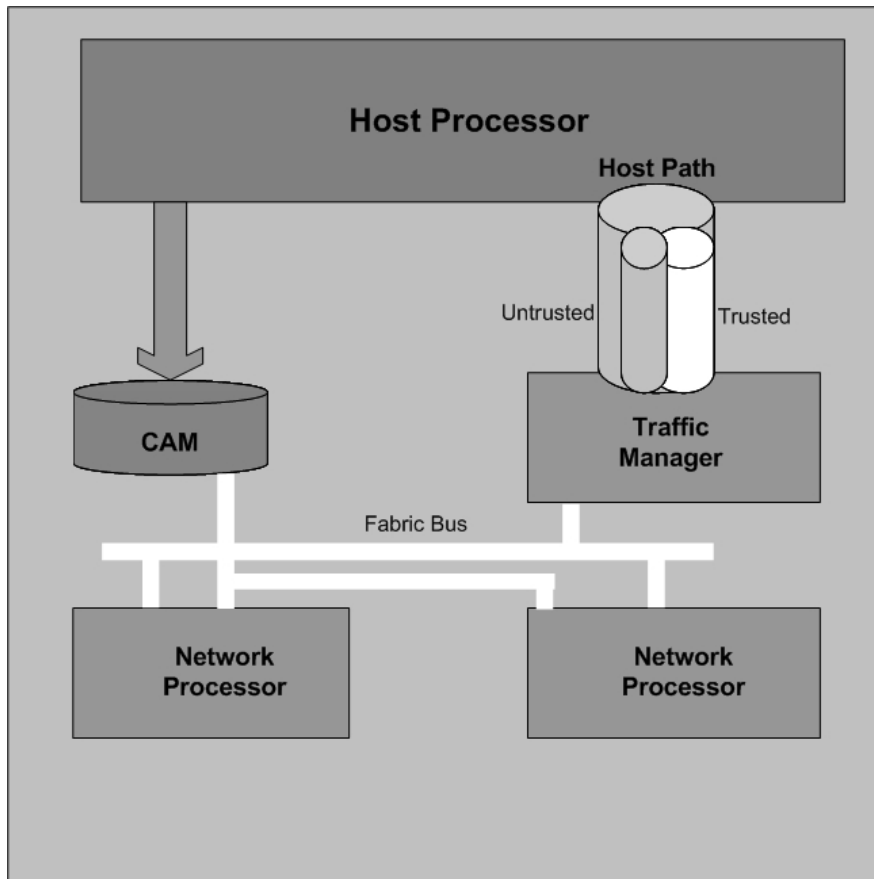
About the Process

DoS attacks are handled in the Oracle CSM's host path. The Oracle CSM uses NAT table entries to filter out undesirable IP addresses; creating a deny list. After a packet from an endpoint is accepted through NAT filtering,

Security

policing is implemented in the Traffic Manager subsystem based on the sender's IP address. NAT table entries distinguish signaling packets coming in from different sources for policing purposes. The maximum number of policed calls that the Oracle CSM can support is 16K (on 32K CAM / IDT CAM).

The Traffic Manager has two pipes, trusted and untrusted, for the signaling path. Each signaling packet destined for the host CPU traverses one of these two pipes.



Trusted Path

Packets from trusted devices travel through the trusted pipe in their own individual queues. In the Trusted path, each trusted device flow has its own individual queue (or pipe). The Oracle CSM can dynamically add device flows to the trusted list by promoting them from the Untrusted path based on behavior; or they can be statically provisioned.

Trusted traffic is put into its own queue and defined as a device flow based on the following:

- source IP address
- source UDP/TCP port number
- destination IP address
- destination UDP/TCP port (SIP or MGCP interface to which it is sending)
- realm it belongs to, which inherits the Ethernet interface and VLAN it came in on

For example, SIP packets coming from 10.1.2.3 with UDP port 1234 to the Oracle CSM SIP interface address 11.9.8.7 port 5060, on VLAN 3 of Ethernet interface 0:1, are in a separate Trusted queue and policed independently from SIP packets coming from 10.1.2.3 with UDP port 3456 to the same Oracle CSM address, port and interface.

Data in this flow is policed according to the configured parameters for the specific device flow, if statically provisioned. Alternatively, the realm to which endpoints belong have a default policing value that every device flow will use. The defaults configured in the realm mean each device flow gets its own queue using the policing values. As shown in the previous example, if both device flows are from the same realm and the realm is configured to have an

average rate limit of 10K bytes per second (10KBps), each device flow will have its own 10KBps queue. They are not aggregated into a 10KBps queue.

The individual flow queues and policing lets the Oracle CSM provide each trusted device its own share of the signaling, separate the device's traffic from other trusted and untrusted traffic, and police its traffic so that it can't attack or overload the Oracle CSM (therefore it is trusted, but not completely).

Address Resolution Protocol Flow

The Address Resolution Protocol (ARP) packets are given their own trusted flow with the bandwidth limitation of 8 Kbps. ARP packets are able to flow smoothly, even when a DoS attack is occurring.

Untrusted Path

Packets (fragmented and unfragmented) that are not part of the trusted or denied list travel through the untrusted pipe. In the untrusted path, traffic from each user/device goes into one of 2048 queues with other untrusted traffic. Packets from a single device flow always use the same queue of the 2048 untrusted queues, and 1/2048th of the untrusted population also uses that same queue. To prevent one untrusted endpoint from using all the pipe's bandwidth, the 2048 flows defined within the path are scheduled in a fair-access method. As soon as the Oracle CSM decides the device flow is legitimate, it will promote it to its own trusted queue.

All 2048 untrusted queues have dynamic sizing ability, which allows one untrusted queue to grow in size, as long as other untrusted queues are not being used proportionally as much. This dynamic queue sizing allows one queue to use more than average when it is available. For example, in the case where one device flow represents a PBX or some other larger volume device. If the overall amount of untrusted packets grows too large, the queue sizes rebalance, so that a flood attack or DoS attack does not create excessive delay for other untrusted devices.

In the usual attack situations, the signaling processor detects the attack and dynamically demotes the device to denied in the hardware by adding it to the deny ACL list. Even if the Oracle CSM does not detect an attack, the untrusted path gets serviced by the signaling processor in a fair access mechanism. An attack by an untrusted device will only impact 1/1000th of the overall population of untrusted devices, in the worst case. Even then there's a probability of users in the same 1/1000th percentile getting in and getting promoted to trusted.

IP Fragment Packet Flow

All fragment packets are sent through their own 1024 untrusted flows in the Traffic Manager. The first ten bits (LSB) of the source address are used to determine which fragment-flow the packet belongs to. These 1024 fragment flows share untrusted bandwidth with already existing untrusted-flows. In total, there are 2049 untrusted flows: 1024-non-fragment flows, 1024 fragment flows, and 1 control flow.

Fragmented ICMP packets are qualified as ICMP packets rather than fragment packets. Fragment and non-fragmented ICMP packets follow the trusted-ICMP-flow in the Traffic Manager, with a bandwidth limit of 8Kbs.

Fragment Packet Loss Prevention

You can set the maximum amount of bandwidth (in the max-untrusted-signaling parameter) you want to use for untrusted packets. However, because untrusted and fragment packets share the same amount of bandwidth for policing, any flood of untrusted packets can cause the Oracle CSM to drop fragment packets.

To prevent fragment packet loss, you can set the fragment-msg-bandwidth. When it is set to any value other than 0 (which disables it), the Oracle CSM:

- Provides for a separate policing queue for fragment packets (separate from that used for untrusted packets)
- Uses this new queue to prevent fragment packet loss when there is a flood from untrusted endpoints.

When you set up a queue for fragment packets, untrusted packets likewise have their own queue—meaning also that the max-untrusted-signaling and min-untrusted-signaling values are applied to the untrusted queue.

Static and Dynamic ACL Entry Limits

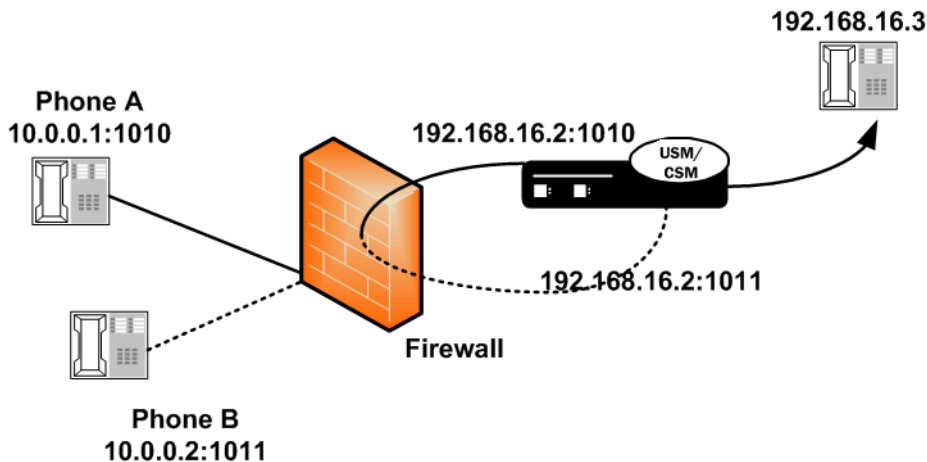
The Oracle CSM can simultaneously police a maximum of 250,000 trusted device flows, while at the same time denying an additional 32,000 attackers. If list space becomes full and additional device flows need to be added, the oldest entries in the list are removed and the new device flows are added.

Dynamic Deny for HNT

Dynamic deny for HNT has been implemented on the Oracle CSM for cases when callers are behind a NAT or firewall. Without this feature, if one caller behind a NAT or firewall were denied, the Oracle CSM would also deny all other users behind the same NAT or firewall. This would be true even for endpoints behind the firewall that had not crossed threshold limits you set for their realm; all endpoints behind the firewall would go out of service. In the following diagram, both Phone A and Phone B would be denied because their IP addresses would be translated by the firewall to the same IPv4 address (192.168.16.2).

However, dynamic deny for HNT allows the Oracle CSM to determine, based on the UDP/TCP port, which endpoints should be denied and which should be allowed. The Oracle CSM can determine that even though multiple endpoints originating behind a firewall appear with the same IPv4 address, those addresses use different ports and are unique.

As shown in the diagram below, the ports from Phone A and Phone B remain unchanged. This way, if Phone A violates the thresholds you have configured, the Oracle CSM can block traffic from Phone A while still accepting traffic from Phone B.



Host and Media Path Protection Process

The Oracle CSM Network Processors (NPs) check the deny and permit lists for received packets, and classify them as trusted, untrusted or denied (discard). Only packets to signaling ports and dynamically signaled media ports are permitted. All other packets sent to Oracle CSM ports are filtered. Only packets from trusted and untrusted (unknown) sources are permitted; any packet from a denied source is dropped by the NP hardware. The Traffic Manager manages bandwidth policing for trusted and untrusted traffic, as described earlier. Malicious traffic is detected in the host processor and the offending device is dynamically added to denied list, which enables early discard by the NP. Devices become trusted based on behavior detected by the Signaling Processor, and dynamically added to the trusted list. This process enables the proper classification by the NP hardware. All other traffic is untrusted (unknown).

Session Director Access Control

You can create static trusted/untrusted/deny lists with source IP addresses or IP address prefixes, UDP/TCP port number or ranges, and based on the appropriate signaling protocols. Furthermore, the Oracle CSM can dynamically promote and demote device flows based on the behavior, and thus dynamically creates trusted, untrusted, and denied list entries.

Access Control for Hosts

The Oracle CSM loads ACLs so they are applied when signaling ports are loaded. The following rules apply to static NAT entries based on your configuration:

- If there are no ACLs applied to a realm that have the same configured trust level as that realm, the Oracle CSM adds a default NAT entry using the realm parameters.
- If you configure a realm with none as its trust level and you have configured ACLs, the Oracle CSM only applies the ACLs.
- If you set a trust level for the ACL that is lower than the one you set for the realm, the Oracle CSM will not add a separate NAT entry for the ACL.

ACLs provide access control based on destination addresses when you configure destination addresses as a way to filter traffic. You can set up a list of access control exceptions based on the source or the destination of the traffic.

For dynamic ACLs based on the promotion and demotion of endpoints, the rules of the matching ACL are applied.

Access Control Endpoint Classification Capacity and DoS

The following capacities are for both IPv4 and IPv6 endpoints.

Platform	Denied	Trusted	Media	Untrusted	Dynamic Trusted	ARP	VLAN
AP3820	32000	8000	8000	2000	250000	4000	4000
AP4500	32000	8000	32000	2000	250000	4000	4000

Media Access Control

The media access control consists of media path protection and pinholes through the firewall. Only RTP and RTCP packets from ports dynamically negotiated through signaling are allowed, which reduces the chance of RTP hijacking. Media access depends on both the destination and source RTP/RTCP UDP port numbers being correct, for both sides of the call.

Host Path Traffic Management

The host path traffic management consists of the dual host paths discussed earlier:

- Trusted path is for traffic classified by the system as trusted. You can initially define trusted traffic by ACLs, as well as by dynamically promoting it through successful SIP or MGCP registration, or a successful call establishment. You can configure specific policing parameters per ACL, as well as define default policing values for dynamically-classified flows. Traffic for each trusted device flow is limited from exceeding the configured values in hardware. Even an attack from a trusted, or spoofed trusted, device cannot impact the system.
- Untrusted path is the default for all unknown traffic that has not been statically provisioned otherwise. For example, traffic from unregistered endpoints. Pre-configured bandwidth policing for all hosts in the untrusted path occurs on a per-queue and aggregate basis.

Traffic Promotion

Traffic is promoted from untrusted to trusted list when the following occurs:

- successful SIP registration for SIP endpoints
- successful RSIP response for MGCP endpoints
- successful session establishment for SIP or MGCP calls

Malicious Source Blocking

Malicious source blocking consists of monitoring the following metrics for each source:

- SIP transaction rate (messages per second)

- SIP call rate (call attempts per second)
- Nonconformance/invalid signaling packet rate

Device flows that exceed the configured invalid signaling threshold, or the configured valid signaling threshold, within the configured time period are demoted, either from trusted to untrusted, or from untrusted to denied classification.

Blocking Actions

Blocking actions include the following:

- Dynamic deny entry added, which can be viewed through the ACLI.
- SNMP trap generated, identifying the malicious source

Dynamically added deny entries expire and are promoted back to untrusted after a configured default deny period time. You can also manually clear a dynamically added entry from the denied list using the ACLI.

Protecting Against Session Agent Overloads

You can prevent session agent overloads with registrations by specifying the registrations per second that can be sent to a session agent.

ARP Flood Protection Enhancements

Enhancements have been made to the way the Oracle CSM provides ARP flood protection. In releases prior to Release C5.0, there is one queue for both ARP requests and responses, which the Oracle CSM polices at a non-configurable limit (eight kilobytes per second). This method of ARP protection can cause problems during an ARP flood, however. For instance, gateway heartbeats the Oracle CSM uses to verify (via ARP) reachability for default and secondary gateways could be throttled; the Oracle CSM would then deem the router or the path to it unreachable, decrement the system's health score accordingly. Another example is when local routers send ARP requests for the Oracle CSM's address are throttled in the queue; the Oracle CSM never receives the request and so never responds, risking service outage.

The solution implemented to resolve this issue is to divide the ARP queue in two, resulting in one ARP queue for requests and a second for responses. This way, the gateway heartbeat is protected because ARP responses can no longer be flooded from beyond the local subnet. In addition, the Oracle CSMs in HA nodes generate gateway heartbeats using their shared virtual MAC address for the virtual interface.

In addition, this solution implements a configurable ARP queue policing rate so that you are not committed to the eight kilobytes per second used as the default in prior releases. The previous default is not sufficient for some subnets, and higher settings resolve the issue with local routers sending ARP request to the Oracle CSM that never reach it or receive a response.

As a security measure, in order to mitigate the effect of the ARP table reaching its capacity, configuring the media-manager option, active-arp, is advised. Enabling this option causes all ARP entries to get refreshed every 20 minutes.

Dynamic Demotion for NAT Devices

In addition to the various ways the Oracle CSM already allows you to promote and demote devices to protect itself and other network elements from DoS attacks, it can now block off an entire NAT device. The Oracle CSM can detect when a configurable number of devices behind a NAT have been blocked off, and then shut off the entire NAT's access.

This dynamic demotion of NAT devices can be enabled for an access control (ACL) configuration or for a realm configuration. When you enable the feature, the Oracle CSM tracks the number of endpoints behind a single NAT that have been labeled untrusted. It shuts off the NAT's access when the number reaches the limit you set.

The demoted NAT device then remains on the untrusted list for the length of the time you set in the deny-period.

Configuring DoS Security

This section explains how to configure the Oracle CSM for DoS protection.

Configuration Overview

Configuring Oracle CSM DoS protection includes masking source IP and port parameters to include more than one match and configuring guaranteed minimum bandwidth for trusted and untrusted signaling path. You can also configure signaling path policing parameters for individual source addresses. Policing parameters are defined as peak data rate (in bytes/sec), average data rate (in bytes/sec), and maximum burst size.

You can configure deny list rules based on the following:

- ingress realm
- source IP address
- source port
- transport protocol (TCP/UDP)

Changing the Default Oracle CSM Behavior

The Oracle CSM automatically creates permit untrusted ACLs that let all sources (address prefix of 0.0.0.0/0) reach each configured realm's signaling interfaces, regardless of the realm's address prefix. To deny sources or classify them as trusted, you create static or dynamic ACLs, and the global permit untrusted ACL to specifically deny sources or classify them as trusted. Doing this creates a default permit-all policy with specific deny and permit ACLs based on the realm address prefix.

You can change that behavior by configuring static ACLs for realms with the same source prefix as the realm's address prefix; and with the trust level set to the same value as the realm. Doing this prevents the permit untrusted ACLs from being installed. You then have a default deny all ACL policy with specific static permit ACLs to allow packets into the system.

Example 1 Limiting Access to a Specific Address Prefix Range

The following example shows how to install a permit untrusted ACL of source 12.34.0.0/16 for each signalling interface/port of a realm called access. Only packets from within the source address prefix range 12.34.0.0/16, destined for the signaling interfaces/port of the realm named access, are allowed. The packets go into untrusted queues until they are dynamically demoted or promoted based on their behavior. All other packets are denied/dropped.

- Configure a realm called access and set the trust level to low and the address prefix to 12.34.0.0/16.
- Configure a static ACL with a source prefix of 12.34.0.0/16 with the trust level set to low for the realm named access.

Example 2 Classifying the Packets as Trusted

Building on Example 1, this example shows how to classify all packets from 12.34.0.0/16 to the realm signaling interfaces as trusted and place them in a trusted queue. All other packets from outside the prefix range destined to the realm's signaling interfaces are allowed and classified as untrusted; then promoted or demoted based on behavior.

You do this by adding a global permit untrusted ACL (source 0.0.0.0) for each signaling interface/port of the access realm. You configure a static ACL with a source prefix 12.34.0.0/16 and set the trust level to high.

Adding this ACL causes the Oracle CSM to also add a permit trusted ACL with a source prefix of 12.34.0.0/16 for each signaling interface/port of the access realm. This ACL is added because the trust level of the ACL you just added is high and the realm's trust level is set to low. The trust levels must match to remove the global permit trusted ACL.

Example 3 Installing Only Static ACLs

This example shows you how to prevent the Oracle CSM from installing the global permit (0.0.0.0) untrusted ACL.

- Configure a realm with a trust level of none.

- Configure static ACLs for that realm with the same source address prefix as the realm's address prefix, and set the trust level to any value.

The system installs only the static ACLs you configure.

Access Control List Configuration

To configure access control lists:

1. Access the access-control configuration element.

```
ACMEPACKET# configure terminal
ACMEPACKET(configure)# session-router
ACMEPACKET(session-router)# access-control
ACMEPACKET(access-control)#
```

2. Type select to choose and configure an existing object.

```
ACMEPACKET(access-control)# select
<src-ip>:
1: src 0.0.0.0; 0.0.0.0; realm01; ; ALL
```

3. realm-id—Enter the ID of the host's ingress realm.

4. source-address—Enter the source IPv4 address and port number for the host in the following format:

```
<IP address>[/number of address bits][:<port>][/<port bits>]
```

For example:

```
10.0.0.1/24:5000/14
10.0.0.1/16
10.0.0.1/24:5000
10.0.0.1:5000
```

You do not need to specify the number of address bits if you want all 32 bits of the address to be matched. You also do not need to specify the port bits if you want the exact port number matched. If you do not set the port mask value or if you set it to 0, the exact port number will be used for matching. The default value is 0.0.0.0.

5. destination-address—(Is ignored if you configure an application protocol in step 7.) Enter the destination IPv4 address and port for the destination in the following format:

```
<IP address>[/number of address bits][:<port>[/<port bits>]]
```

You do not need to specify the number of address bits if you want all 32 bits of the address to be matched. You also do not need to specify the port bits if you want the exact port number matched. If you do not set the port mask value or if you set it to 0, the exact port number will be used for matching. The default value is 0.0.0.0.

6. application-protocol—Enter the application protocol type for this ACL entry. The valid values are:

- SIP | None



Note: If application-protocol is set to none, the destination-address and port will be used. Ensure that your destination-address is set to a non-default value (0.0.0.0.)

7. transport-protocol—Select the transport-layer protocol configured for this ACL entry. The default value is ALL. The valid values are:

- ALL | TCP | UDP

8. access—Enter the access control type or trusted list based on the trust-level parameter configuration for this host. The default value is permit. The valid values are:

- permit—Puts the entry into the untrusted list. The entry is promoted or demoted according to the trust level set for this host.
- deny—Puts the entry in the deny list.

9. average-rate-limit—Indicate the sustained rate in bytes per second for host path traffic from a trusted source within the realm. The default value is 0. A value of 0 means policing is disabled. The valid range is:

- Minimum—0

- Maximum—999999999
- 10. trust-level**—Indicate the trust level for the host with the realm. The default value is none. The valid values are:
- none—Host is always untrusted. It is never promoted to the trusted list or demoted to the deny list.
 - low—Host can be promoted to the trusted list or demoted to the deny list.
 - medium—Host can be promoted to the trusted list but is only demoted to untrusted. It is never added to the deny list.
 - high—Host is always trusted.
- 11. invalid-signal-threshold**— Enter the number of invalid signaling messages that trigger host demotion. The value you enter here is only valid when the trust level is low or medium. Available values are:
- Minimum—Zero (0) is disabled.
 - Maximum—999999999

If the number of invalid messages exceeds this value based on the tolerance window parameter, configured in the media manager, the host is demoted.

The tolerance window default is 30 seconds. Bear in mind, however, that the system uses the same calculation it uses for specifying "recent" statistics in show commands to determine when the number of signaling messages exceeds this threshold. This calculation specifies a consistent start time for each time period to compensate for the fact that the event time, such as a user running a show command, almost never falls on a time-period's border. This provides more consistent periods of time for measuring event counts.

The result is that this invalid signal count increments for two tolerance windows, 60 seconds by default, within which the system monitors whether or not to demote the host. The signal count for the current tolerance window is always added to the signal count of the previous tolerance window and compared against your setting.

- 12. maximum-signal-threshold**—Set the maximum number of signaling messages the host can send within the tolerance window. The value you enter here is only valid when the trust level is low or medium. The default value is 0, disabling this parameter. The valid range is:
- Minimum—0
 - Maximum—999999999

If the number of messages received exceeds this value within the tolerance window, the host is demoted.

- 13. untrusted-signal-threshold**—Set the maximum number of untrusted messages the host can send within the tolerance window. Use to configure different values for trusted and un-trusted endpoints for valid signaling message parameters. Also configurable per realm. The default value is 0, disabling this parameter. The valid range is:

- Minimum—0
- Maximum—999999999

- 14. deny-period**—Indicate the time period in seconds after which the entry for this host is removed from the deny list. The default value is 30. The valid range is:
- Minimum—0
 - Maximum—999999999

- 15. nat-trust-threshold**—Enter the number of endpoints behind a NAT that must be denied for the Oracle CSM to demote the NAT device itself to denied (dynamic demotion of NAT devices). The default is 0, meaning dynamic demotion of NAT devices is disabled. The range is from 0 to 65535.

The following example shows access control configured for a host in the external realm.

```
access-control
  realm-id          external
  source-address    192.168.200.215
  destination-address 192.168.10.2:5000
  application-protocol SIP
  transport-protocol ALL
  access            permit
```

```
average-rate-limit      3343
trust-level             low
invalid-signal-threshold 5454
maximum-signal-threshold 0
untrusted-signal-threshold 0
deny-period            0
```

The following example of how to configure a black-list entry:

```
access-control
  realm-id               external
  source-address         192.168.200.200
  destination-address    192.168.10.2:5000
  application-protocol   SIP
  transport-protocol     ALL
  access                 deny
  average-rate-limit     0
  trust-level            none
  invalid-signal-threshold 0
  maximum-signal-threshold 0
  untrusted-signal-threshold 0
  deny-period           0
```

Host Access Policing

You can configure the Oracle CSM to police the overall bandwidth of the host path.

To configure host access policing:

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

```
ORACLE# configure terminal
```

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

```
ORACLE (configure) # media-manager
```

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

```
ORACLE (media-manager) # media-manager
ORACLE (media-manager-config) #
```

4. `max-signaling-bandwidth`—Set the maximum overall bandwidth available for the host path in bytes per second, which includes signaling messages from trusted and untrusted sources. It also includes any Telnet and FTP traffic on media ports. The default value is 1000000. The valid range is:
 - Minimum—71000
 - Maximum—10000000
5. `max-untrusted-signaling`—Set the percentage of the maximum signaling bandwidth you want to make available for messages coming from untrusted sources. This bandwidth is only available when not being used by trusted sources. The default value is 100. The valid range is:
 - Minimum—1
 - Maximum—100
6. `min-untrusted-signaling`—Set the percentage of the maximum signaling bandwidth you want reserved for the untrusted sources. The rest of the bandwidth is available for trusted resources, but can also be used for untrusted sources (see `max-untrusted-signaling`). The default value is 30. The valid range is:
 - Minimum—1
 - Maximum—100
7. `fragment-msg-bandwidth`—Enter the amount of bandwidth to use for the fragment packet queue. If you leave this parameter set to 0, then the Oracle CSM will use the same queue for and share bandwidth between untrusted packets and fragment packets. The default value is zero (0). The valid range is:
 - Minimum—0

- Maximum—1000000
8. tolerance-window—Set the size of the window used to measure host access limits. The value entered here is used to measure the invalid message rate and maximum message rate for the realm configuration. The default value is 30. The valid range is:
- Minimum—0
 - Maximum—99999999


The following example shows a host access policing configuration.

```
media-manager
  state                enabled
  latching             enabled
  flow-time-limit      86400
  initial-guard-timer  300
  subsq-guard-timer    300
  tcp-flow-time-limit  86400
  tcp-initial-guard-timer 300
  tcp-subsq-guard-timer 300
  tcp-number-of-ports-per-flow 2
  hnt-rtcp             disabled
  algd-log-level       WARNING
  mbc-d-log-level      WARNING
  home-realm-id
  red-flow-port        1985
  red-mgcp-port        1986
  red-max-trans        10000
  red-sync-start-time  5000
  red-sync-comp-time   1000
  max-signaling-bandwidth 1000000
  max-untrusted-signaling 50
  min-untrusted-signaling 30
  tolerance-window     30
  rtc-rate-limit       0
```

Configuring ARP Flood Protection

You do not need to configure the Oracle CSM to enable the use of two separate ARP queues; that feature is enabled automatically.

If you want to configure the ARP queue policing rate, you can do so in the media manager configuration.

 **Note:** this feature is not RTC-supported, and you must reboot your Oracle CSM in order for your configuration changes to take effect.

To set the ARP queue policing rate:

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

```
ORACLE# configure terminal
```

2. Type `media-manager` and press Enter.

```
ORACLE(configure)# media-manager
ORACLE(media-manager)#
```

3. Enter `media-manager` and press <Enter>:

```
ORACLE(media-manager)# media-manager
ORACLE(media-manager-config)#
```

4. `arp-msg-bandwidth`—Enter the rate at which you want the Oracle CSM to police the ARP queue; the value you enter is the bandwidth limitation in bytes per second. The default value is 32000. The valid range is:
 - Minimum—2000
 - Maximum—200000
5. Save your configuration.

6. Reboot your Oracle CSM.

Access Control for a Realm

Each host within a realm can be policed based on average rate, peak rate, and maximum burst size of signaling messages. These parameters take effect only when the host is trusted. You can also set the trust level for the host within the realm. All untrusted hosts share the bandwidth defined for the media manager: maximum untrusted bandwidth and minimum untrusted bandwidth.

To configure access control for a realm:

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

```
ORACLE# configure terminal
```

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

```
ORACLE(configure)# media-manager
```

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

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

4. `addr-prefix`—Set the IP address prefix used to determine if an IP address is associated with the realm. This value is then associated with the ACLs you create to determine packet access. The default value is 0.0.0.0.
5. `average-rate-limit`—Set the sustained rate for host path traffic from a trusted source within the realm in bytes per second. The default value is zero (0), disabling this parameter. The valid range is:
 - Minimum—0
 - Maximum—4294967295
6. `access-control-trust-level`—Set the trust level for the host within the realm. The default value is none. The valid values are:
 - none—Host is always untrusted. It is never promoted to the trusted list or demoted to the deny list.
 - low—Host can be promoted to the trusted list or demoted to the deny list.
 - medium—Host can be promoted to the trusted list but is only demoted to untrusted. It is never added to the deny list.
 - high—Host is always trusted.
7. `invalid-signal-threshold`— Enter the number of invalid signaling messages that trigger host demotion. The value you enter here is only valid when the trust level is low or medium. Available values are:
 - Minimum—Zero (0) is disabled.
 - Maximum—999999999

If the number of invalid messages exceeds this value based on the tolerance window parameter, configured in the media manager, the host is demoted.

The tolerance window default is 30 seconds. Bear in mind, however, that the system uses the same calculation it uses for specifying "recent" statistics in show commands to determine when the number of signaling messages exceeds this threshold. This calculation specifies a consistent start time for each time period to compensate for the fact that the event time, such as a user running a show command, almost never falls on a time-period's border. This provides more consistent periods of time for measuring event counts.

The result is that this invalid signal count increments for two tolerance windows, 60 seconds by default, within which the system monitors whether or not to demote the host. The signal count for the current tolerance window is always added to the signal count of the previous tolerance window and compared against your setting.

8. `maximum-signal-threshold`—Set the maximum number of signaling messages one host can send within the window of tolerance. The host is demoted if the number of messages received by the Oracle CSM exceeds the number set here. Valid only when the trust level is set to low or medium. The default value is zero (0), disabling this parameter. The valid range is:

- Minimum—0
 - Maximum—4294967295
9. `untrusted-signal-threshold`—Set the maximum number of untrusted messages the host can send within the tolerance window. Use to configure different values for trusted and un-trusted endpoints for valid signaling message parameters. Also configurable per realm. The default value is zero (0), disabling the parameter. The valid range is:
- Minimum—0
 - Maximum—4294967295
10. `deny-period`—Set the length of time an entry is posted on the deny list. The host is deleted from the deny list after this time period. The default value is 30. A value of 0 disables the parameter. The valid range is:
- Minimum—0
 - Maximum—4294967295
11. `nat-trust-threshold`—Enter the number of endpoints behind a NAT that must be denied for the Oracle CSM to demote the NAT device itself to denied (dynamic demotion of NAT devices). The default is 0, meaning dynamic demotion of NAT devices is disabled. The range is from 0 to 65535.

The following example shows a host access policing configuration.

```
realm-config
  identifier                private
  addr-prefix              192.168.200.0/24
  network-interfaces
  mm-in-realm              private:0
  mm-in-network            disabled
  msm-release              enabled
  qos-enable               disabled
  max-bandwidth            disabled
  ext-policy-svr           0
  max-latency              0
  max-jitter               0
  max-packet-loss         0
  observ-window-size      0
  parent-realm
  dns-realm
  media-policy
  in-translationid
  out-translationid
  class-profile
  average-rate-limit       8000
  access-control-trust-level medium
  invalid-signal-threshold 200
  maximum-signal-threshold 0
  untrusted-signal-threshold 500
  deny-period              30
  symmetric-latching      disabled
  pai-strip                disabled
  trunk-context
```

Configuring Overload Protection for Session Agents

The Oracle CSM offers two methods to control SIP registrations to smooth the registration flow.

You can limit the:

- number of new register requests sent to a session agent (using the `max-register-sustain-rate` parameter)
- burstiness which can be associated with SIP registrations

The first method guards against the Oracle CSM's becoming overwhelmed with register requests, while the second method guards against a transient registration that can require more than available registration resources.

SIP registration burst rate control allows you to configure two new parameters per SIP session agent—one that controls the registration burst rate to limit the number of new registration requests, and a second to set the time window for that burst rate. When the registration rate exceeds the burst rate you set, the Oracle CSM responds to new registration requests with 503 Service Unavailable messages.

Note that this constraint is not applied to re-registers resulting from a 401 Unauthorized challenge request.

To configure overload protection for session agents:

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 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. constraints—Enable this parameter to set the sustained rate window constraint you configure in the next step. The default value is disabled. The valid values are:
 - enabled | disabled
5. sustain-rate-window—Enter a number to set the sustained window period (in milliseconds) that is used to measure the sustained rate. The default value is zero (0). The valid range is:
 - Minimum—10
 - Maximum—4294967295

The value you set here must be higher than or equal to the value you set for the burst rate window.



Note: If you are going to use this parameter, you must set it to a minimum value of 10.

6. max-register-sustain-rate—Enter a number to set the maximum number of registrations per second you want sent to the session agent. The default value is zero (0), disabling the parameter. The valid range is:
 - Minimum—0
 - Maximum—4294967295
7. register-burst-window—Define the window size in seconds for the maximum number of allowable SIP registrations. 0 is the minimum and default value for this parameter; the maximum value is 999999999.
8. max-register-burst-rate—Enter the maximum number of new registrations you want this session agent to accept within the registration burst rate window. If this threshold is exceeded, the Oracle CSM will respond to new registration requests with 503 Service Unavailable messages. 0 is the minimum and default value for this parameter; the maximum value is 999999999.
9. Save and activate your configuration.

DDoS Protection from Devices Behind a NAT

A DDoS attack could be crafted such that multiple devices from behind a single NAT could overwhelm the Oracle CSM. The Oracle CSM would not detect this as a DDoS attack because each endpoint would have the same source IP but multiple source ports. Because the Oracle CSM allocates a different CAM entry for each source IP:Port combination, this attack will not be detected. This feature remedies such a possibility.

Restricting the Number of Endpoints behind a NAT

Each new source IP address and source IP port combination now counts as an endpoint for a particular NAT device. After the configured value of a single NAT's endpoints is reached, subsequent messages from behind that NAT are dropped and the NAT is demoted. This is set with the max-endpoints-per-nat parameter located in both the access-control and realm-config configuration elements.

Counting Invalid Messages from Endpoints behind a NAT

The Oracle CSM also counts the number of invalid messages sent from endpoints behind the NAT. Once a threshold is reached, that NAT is demoted. Numerous conditions are counted as Errors/Invalid Messages from an endpoint. The aggregate of all messages from endpoints behind the NAT are counted against the NAT device, in addition to the existing count against the endpoint. This threshold is set with the `nat-invalid-message-threshold` parameter located in both the `access-control` and `realm-config` configuration elements.

As a unique case, the absence of a REGISTER message following a 401 response is counted as an invalid message from the end point. And if that endpoint is behind a NAT, this scenario will be counted as invalid message from that NAT device as well. You set a timeout period in which the REGISTER message must arrive at the Oracle CSM. This period is set with the `wait-time-for-invalid-register` parameter located in the `realm config`.

DDoS Protection Configuration `realm-config`

To set the DDoS Protection from devices behind NATs in the `realm-config`:

1. Access the `realm-config` configuration element.

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

2. Select the `realm-config` object to edit.

```
ORACLE(realm-config)# select
identifier:
1: realm01 left-left:0 0.0.0.0

selection: 1
ORACLE(realm-config)#
```

3. `max-endpoints-per-nat`— Set the maximum number of endpoints that can exist behind a NAT before demoting the NAT device. Valid values are 0-65535, with 0 disabling this feature. This parameter is also found in the `access control` configuration element.
4. `nat-invalid-message-threshold`—Set the maximum number of invalid messages that may originate behind a NAT before demoting the NAT device. Valid values are 0-65535, with 0 disabling this feature. This parameter is also found in the `access control` configuration element.
5. `wait-time-for-invalid-register`—Set the period (in seconds) that the Oracle CSM counts before considering the absence of the REGISTER message as an invalid message.
6. Type **done** to save your configuration.

DDoS Protection Configuration `access-control`

To set the DDoS Protection from devices behind NATs in the `access-control` configuration element:

1. Access the `access-control` configuration element.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# access-control
ORACLE(access-control)#
```

2. Type `select` to choose and configure an existing object.

```
ORACLE(access-control)# select
<src-ip>:
1: src 0.0.0.0; 0.0.0.0; realm01; ; ALL
selection:1
```

3. `max-endpoints-per-nat`— Set the maximum number of endpoints that can exist behind a NAT before demoting the NAT device. Valid values are 0-65535, with 0 disabling this feature. This parameter is also found in the `access control` configuration element.

4. `nat-invalid-message-threshold`—Set the maximum number of invalid messages that may originate behind a NAT before demoting the NAT device. Valid values are 0-65535, with 0 disabling this feature. This parameter is also found in the access control configuration element.
5. Type `done` to save your work and continue.

SNMP Trap support

The following trap is sent by the Oracle CSM whenever a NAT device is blacklisted due to the triggers listed in this feature. Reasons are not included in the trap, but are available from the syslogs.

```
apSysMgmtExpDOSTrap      NOTIFICATION-TYPE
  OBJECTS                 { apSysMgmtDOSIpAddress,  apSysMgmtDOSRealmID ,
                          apSysMgmtDOSFromUri }
  STATUS                  deprecated
  DESCRIPTION
    "This trap is generated when an IP is placed on a deny list due
    to denial-of-service attempts, and provides the ip address that
    has been demoted, the realm-id of that IP, and (if available)
    the URI portion of the SIP From header of the message that
    caused the demotion."
  ::= { apSysMgmtDOSNotifications 2 }
```

Ensure that the `enable-snmp-monitor-traps` parameter in the `system-config` configuration element is enabled for the Oracle CSM to send out this trap.

Syslog Support

Set the `syslog-on-demote-to-deny` parameter in the `media-manager-config` to enabled to generate syslog on endpoint demotion from untrusted to deny. NAT device demotion will also generate a unique syslog message with accompanying text explaining that it is the NAT device demotion event.

Debugging

The `show sip acl` command now includes counts of Blocked NAT devices.

```
ACMEPACKET# show sipd acl
13:57:28-71
SIP ACL Status      -- Period -- ----- Lifetime -----
                   Active   High   Total   Total   PerMax   High
Total Entries      0     0     0     0     0     0
Trusted            0     0     0     0     0     0
Blocked            0     0     0     0     0     0
Blocked NATs       0     0     0     0     0     0
ACL Operations      ---- Lifetime ----
                   Recent   Total   PerMax
ACL Requests       0     0     0
Bad Messages       0     0     0
Promotions         0     0     0
Demotions          0     0     0
Trust->Untrust     0     0     0
Untrust->Deny     0     0     0
```

TCP Synchronize Attack Prevention

This section explains how the Oracle CSM protects itself from a Transmission Control Protocol (TCP) synchronize (SYN) packet flooding attack sourced from a remote hostile entity.

SIP signaling can be configured on the Oracle CSM to be TCP protocol-based. In this configuration, the Oracle CSM can be a target of a TCP SYN attack. The Oracle CSM C is able to service new call requests throughout the duration of an attack

About SYN

SYN is used by TCP when initiating a new connection to synchronize the sequence numbers on two connecting computers. The SYN is acknowledged by a SYN-ACK by the responding computer. After the SYN-ACK, the client finishes establishing the connection by responding with an ACK message. The connection between the client and the server is then open, and the service-specific data can be exchanged between the client and the server.

A SYN flood is a series of SYN packets from forged IP addresses. The IP addresses are chosen randomly and do not provide any hint of the attacker's location. The SYN flood keeps the server's SYN queue full. Normally this would force the server to drop connections. A server that uses SYN cookies, however, will continue operating normally. The biggest effect of the SYN flood is to disable large windows.

Server Vulnerability

Vulnerability to attack occurs when the server has sent a SYN-ACK back to client, but has not yet received the ACK message; which is considered a half-open connection. The server has a data structure describing all pending connections built in its system memory. This data structure is of finite size, and it can be made to overflow by intentionally creating too many partially-open connections.

The attacking system sends SYN messages to the server that appear to be legitimate, but in fact reference a client that is unable to respond to the SYN-ACK messages. The final ACK message is never sent to the server.

The half-open connections data structure on the server fills and no new incoming connections are accepted until the table is emptied out. Typically there is a timeout associated with a pending connection (the half-open connections will eventually expire and the server will recover). But the attacking system can continue sending IP-spoofed packets requesting new connections faster than the server can expire the pending connections. The server has difficulty in accepting any new incoming network connections.

Configuring TCP SYN Attack Prevention

No configuration is necessary to enable TCP SYN attack prevention. Internal TCP protocol changes were made to provide protection.

Host Certificate Retrieval via SNMP

When a security certificate is installed locally on the Oracle CSM, you can poll the expiration of the certificate using the `apSecurityCertificateTable`.

You can configure the Oracle CSM to generate the `apSecurityCertExpiredNotification` trap once a certificate has expired. The number of minutes between notifications sent is configured in the `security-config` parameter `local-cert-trap-int`.

To send a warning of expiration, you can set the `security-config` parameter `local-cert-exp-warn-period` to the number of days before the locally installed certificate expires in which you would like a warning.

Host Certificate Retrieval Configuration

To configure the Oracle CSM to generate traps when a certificate has or is about to expire:

1. Navigate to the `security-config` configuration element.

```
ORACLE# configure terminal
ORACLE(configure)# security#
ORACLE(security)# security-config
ORACLE(security-config)#
```

2. Set the `local-cert-exp-warn-period` parameter to the number of days before the locally installed certificate expires in order to receive a warning. A value of 0 disables the trap.

```
ORACLE(security-config)# local-cert-exp-warn-period 3
ORACLE(security-config)#
```

3. Set the `local-cert-trap-int` parameter for the number of minutes between notifications sent once a certificate has expired. A value of 0 disables the warning trap.

```
ORACLE(security-config)# local-cert-exp-trap-int 15
ORACLE(security-config)#
```

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

Untrusted Connection Timeout for TCP and TLS

You can configure the Oracle CSM for protection against starvation attacks for socket-based transport (TCP or TLS) for SIP access applications. During such an occurrence, the attacker would open a large number of TCP/TLS connections on the Oracle CSM and then keep those connections open using SIP messages sent periodically. These SIP messages act as keepalives, and they keep sockets open and consume valuable resources.

Using its ability to promote endpoints to a trusted status, the Oracle CSM now closes TCP/TLS connections for endpoints that do not enter the trusted state within the period of time set for the untrusted connection timeout. The attacking client is thus no longer able to keep connections alive by sending invalid messages.

This feature works by setting a value for the connection timeout, which the Oracle CSM checks whenever a new SIP service socket for TCP or TLS is requested. If the timer's value is greater than zero, then the Oracle CSM starts it. If the timer expires, then the Oracle CSM closes the connection. However, if the endpoint is promoted to the trusted state, then the Oracle CSM will cancel the timer.

Caveats

This connection timeout is intended for access applications only, where one socket is opened per-endpoint. This means that the timeout is not intended for using in peering applications; if this feature were enabled for peering, a single malicious SIP endpoint might cause the connection to be torn down unpredictably for all calls.

Untrusted Connection Timeout Configuration for TCP and TLS

The untrusted connection timer for TCP and TLS is set per SIP interface.

To set the untrusted connection timer for TCP and TLS:

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

```
ORACLE# configure terminal
```

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

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

3. Type `sip-interface` and press Enter.

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

If you are adding support for this feature to a pre-existing SIP configuration, then you must select (using the ACLI `select` command) the configuration that you want to edit.

4. `untrusted-conn-timeout`—Enter the time in seconds that you want the Oracle CSM to keep TCP and TLS connections open for untrusted endpoints. The default value is 0, which will not start the timer. The valid range is:
 - Minimum—0
 - Maximum—999999999
5. Save and activate your configuration.

Online Certificate Status Protocol

The Online Certificate Status Protocol (OCSP) is defined in RFC 2560, X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. The protocol enables users to determine the revocation state of a specific certificate, and may provide a more efficient source of revocation information than is possible with Certificate Revocation Lists (CRL).

The protocol specifies the data exchanged between an OCSP client (for example, the Oracle CSM) and an OCSP responder, the Certification Authority (CA), or its delegate, that issued the target certificate. An OCSP client issues a request to an OCSP responder and suspends acceptance of the certificate in question until the responder replies with a certificate status.

Certificate status is reported as

- good
- revoked
- unknown

good indicates a positive response to the status inquiry. At a minimum, this positive response indicates that the certificate is not revoked, but does not necessarily mean that the certificate was ever issued or that the time at which the response was produced is within the certificate's validity interval.

revoked indicates that the certificate has been revoked, either permanently or temporarily.

unknown indicates that the responder cannot identify the certificate.

Caveats

OCSP is currently supported only on TLS interfaces; it is not currently supported for use with IKEv1 and IKEv2.

Online Certificate Status Protocol Configuration

OCSP configuration consists of

1. Configuring one or more certificate status profiles; each profile contains information needed to contact a specific OCSP responder.
2. Enabling certificate revocation checking by assigning a certificate status profile to a previously configured TLS profile.

To create a certificate status profile:

3. From superuser mode, use the following command sequence to access cert-status-profile configuration mode. While in this mode, you provide the information required to access one or more OCSP responders.

```
ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# cert-status-profile
ORACLE(cert-status-profile)#
```

4. Use the required name parameter to identify this cert-status-profile instance — each profile instance provides configuration data for a specific OCSP responder. name is used to distinguish between multiple profile instances.
5. Use the required ip-address parameter to specify the IPv4 address of the OCSP responder.
6. Use the optional port parameter to specify the destination port.

In the absence of an explicitly configured value, the default port number of 80 is used.

7. Use the optional realm-id parameter to specify the realm used to transmit OCSP requests.

In the absence of an explicitly configured value, the default specifies service across the wancom0 interface.

8. Use the optional requester-cert parameter only if OCSP requests are signed; ignore this parameter if requests are not signed.

RFC 2560 does not require signed requests; however, local or CA policies can mandate digital signature..

9. Use the required responder-cert parameter to identify the certificate used to validate OCSP responses — a public key of the OCSP responder.

RFC 2560 requires that all OCSP responders digitally sign OCSP responses, and that OCSP clients validate incoming signatures.

Provide the name of the certificate configuration element that contains the certificate used to validate the signed OCSP response.

10. Use the optional retry-count parameter to specify the maximum number of times to retry an OCSP responder in the event of connection failure.

If the retry counter specified by this parameter is exceeded, the OCSP requester either contacts another responder (if multiple responders have been configured within this cert-status-profile) and quarantine the unavailable responder for a period defined the dead-time parameter.

In the absence of an explicitly configured value (an integer within the range 0 through 10), the default of 1 is used.

```
ORACLE(cert-status-profile)# retry-count 2
ORACLE(cert-status-profile)#
```

11. Use the optional dead-time parameter to specify the quarantine period imposed on an unavailable OCSP responder.

In the absence of an explicitly configured value (an integer within the range 0 through 3600 seconds), the default value (0) is used.

Customer implementations utilizing a single OCSP responder are encouraged to retain the default value, or to specify a brief quarantine period to prevent lengthy service outages.

12. Retain default values for the type and trans-protocol parameter to specify OCSP over an HTTP transport protocol.
13. Use done, exit, and verify-config to complete configuration of this cert-status-profile instance.
14. Repeat Steps 1 through 11 to configure additional certificate status profiles.

To enable certificate status checking:

15. Move to tls-profile configuration mode.

```
ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# tls-profile
ORACLE(tls-profile)#
```

16. Use the required cert-status-check parameter to enable OCSP in conjunction with an existing TLS profile.

17. Use the required cert-status-profile-list parameter to assign one or more cert-status-profiles to the current TLS profile.

Each assigned cert-status-profile provides the information needed to access a single OCSP responder.

Use quotation marks to assign multiple OCSP responders. The following sequence assigns three cert-status-profiles, VerisignClass3Designate, Verisign-1, and Thawte-1 to the TLS-1 profile.

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

Sample Configuration:

```
ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# cert-status-profile
ORACLE(cert-status-profile)# name VerisignClass3Designate
ORACLE(cert-status-profile)# ip-address 192.168.7.100
ORACLE(cert-status-profile)# responder-cert VerisignClass3ValidateOCSP
ORACLE(cert-status-profile)# done
ORACLE(cert-status-profile)# exit
...
...
ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# tls-profile
```

```

ORACLE(tls-profile)# select
<name>:
1. TLS-1
2. TLS-2
3. TLS-3
selection: 1
ORACLE(tls-profile)# cert-status-check enabled
ORACLE(cert-status-profile)# cert-status-profile-list
VerisignClass3Designate Verisign-1 Thawte-1
ORACLE(cert-status-profile)# done
ORACLE(cert-status-profile)# exit

```

Unreachable OCSR

With OCSF enabled, the client implementation running on the Oracle CSM supports message exchange between the Oracle CSM and an OCSR as specified in RFC 2560, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSF*. The Oracle CSM contacts the OCSR whenever a remote client attempts to establish an SSL/TLS connection with the Oracle CSM. The Oracle CSM sends a request to the OCSR to check the current status of the certificate presented by the remote client. The Oracle CSM suspends processing of the SSL/TLS connection request pending receipt of the OCSR response. In previous releases (prior to Version S-CX6.3.0), a good OCSR response resulted in the establishment of a secure SSL/TLS connection. A revoked or unknown OCSR response, or the failure to reach an OCSR, resulted in the rejection of the connection attempt.

This behavior, which adheres to the requirements of RFC 2560, conflicts with the requirements of Section 5.4.6.2.1.6.4.a.i of UCR 2008 which requires an OCSF client to attempt authentication of remote clients in the event of an unreachable OCSR.

Release S-CX6.3F1 adds a new attribute (`ignore-dead-responder`) to the TLS profile configuration element to provide compliance with DISA/DoD requirements specifying OCSF client operations when faced with unreachable OCSRs. By default, the attribute is disabled meaning that all client connections will be disallowed in the event of unreachable OCSRs.

In DISA/DoD environments `ignore-dead-responder` should be enabled, allowing local certificate-based authentication by the Oracle CSM in the event of unreachable OCSRs. Successful authentication is achieved if the certificate presented by the remote client was signed by a Certificate Authority (CA) referenced by the `trusted-ca-certificates` attribute. If the local authentication succeeds, the secure TLS/SSL connection is established; otherwise the connection is rejected.

Unreachable OCSR Configuration

The following sample configuration implements DISA/DoD-compliant client behavior in the event of an unreachable OCSR.

```

ACMEPACKET# configure terminal
ACMEPACKET(configure)# security#
ACMEPACKET(security)# tls-profile
ACMEPACKET(security)# show
tls-profile
      name                DoD
end-entity-certificate    sylvrCert-2048
trusted-ca-certificates   dod1 dod2 disaA disaB IBM1
cipher-list               all
verify-depth              10
mutual-authenticate       disabled
tls-version               tlsv1
cert-status-check         enabled
cert-status-profile-list   DoD
ignore-dead-responder     enabled
...
...
ACMEPACKET(tls-profile)#

```

OCSR Status Monitoring

OCSR monitoring is provided to track the reachability of individual OCSRs, and, in topologies containing multiple OCSRs, the overall availability of OCSR service.

If monitoring is enabled for individual OCSRs, reachability is monitored by observing responder transactions.

Initially, all OCSRs are considered reachable. If a previously reachable OCSR fails to respond to a certificate status request, the Oracle CSM marks the OCSR as unreachable, and generates an SNMP trap and log entry indicating that status. If a previously unreachable OCSR respond to a certificate status request, the Oracle CSM returns the OCSR to the reachable status, and generates an SNMP trap and log entry indicating that status change.

Use the following procedure to enable monitoring of individual OCSRs.

1. Navigate to the new security-config configuration element.

```
ORACLE# configure terminal
ORACLE(configure)# security#
ORACLE(security)# security-config
ORACLE(security-config)#
```

2. Enable monitoring of individual OCSRs by setting the ocsr-monitoring-traps attribute to enabled; this attribute is disabled by default.

```
ORACLE(security-config)# ocsr-monitoring-traps enabled
ORACLE(security-config)#
```

3. Use done, exit, and verify-config to complete required configuration.

Reachability status of individual OCSRs is aggregated to monitor the overall availability of OCSR service. Using the procedure explained above, the Oracle CSM maintains a count of all OCSRs, and of all reachable OCSRs.

- If all OCSRs are reachable, the Oracle CSM generates a trap and log entry noting this optimal state.
- If all OCSRs are unreachable, the Oracle CSM generates a trap and log entry noting this erroneous state.
- When the Oracle CSM transitions from either of the two states described above (in the optimal state, when an OCSR becomes unreachable; in the erroneous state, when an unreachable OCSR becomes reachable), the Oracle CSM generates a trap and log entry indicating that an unspecified number of OCSRs are reachable.

Monitoring of OCSR service availability is a by-product of enabling SNMP; no further configuration is required.

OCSR Access via FQDN

Prior software releases supported OCSR access only via IPv4 addresses and port numbers. In response to a DISA/DoD request, Release S-CX6.3F1 adds support for OCSR access via FQDNs. Since multiple public key infrastructure (PKI) elements capable of supporting OCSP requests can exist within a DISA/DoD environment, the Domain Name Service (DNS) lookup that resolves the FQDN can result in more than one OCSR IP address being returned to the Oracle CSM in its role of OCSP client. When processing a lookup that contains more than one IP address, the Oracle CSM uses a round-robin algorithm to select from the list of OCSR addresses.

OCSR Access via FQDN is available on all media interfaces and on the wancom0 administrative interface. Note that support for FQDN-based access is requires the configuration of DNS support.

If the realm attribute is configured in the certificate-status-profile configuration element, the required DNS query is issued on the corresponding network interface. This model requires configuration of the dns-ip-primary attribute, and optionally the dns-ip-backup1 and dns-ip-backup2 attributes for the realm's network interface.

If the realm attribute is not configured in the certificate-status-profile, the required DNS query is issued on the wancom0 interface. This model requires configuration of the dns-ip-primary attribute, and optionally the dns-ip-backup1 and dns-ip-backup2 attributes for the wancom0 interface.

Access via an FQDN is supported by a new attribute (hostname) in the cert-status-profile configuration element.

The Oracle CSM allows configuration of both an OCSR IP address and port number (using the ip-address and port attributes) and an OCSR domain (using the hostname attribute).

In such cases the verify-config command issues a warning and notes that IP address-based access will be used.

OCSR Access Configuration via IP Address

The following sample configuration accesses an OCSR at 192.168.7.100:8080.

```
ORACLE# configure terminal
ORACLE (configure) # security#
ORACLE (security) # cert-status-profile#
ORACLE (cert-status-profile) # show
cert-status-profile
  name                defaultOCSP
  ip-address          192.168.7.100
  hostname
  port                8080
  type                OCSP
  trans-PROTO         HTTP
  requestor-cert      ocspsVerisignClient
  responder-cert      VerisignCA2
  trusted-cas
  realm-id            admin
  retry-count         1
  dead-time           0
  last-modified-by
  last-modified-date
ORACLE (cert-status-profile) #
```

OCSR Access Configuration via FQDN

The following sample configuration accesses one or more OCSRs at example.disa.mil.

Note that in the absence of a specified domain, the wancom0 interface must be DNS-enabled.

```
ORACLE# configure terminal
ORACLE (configure) # security#
ORACLE (cert-status-profile) # show
cert-status-profile
  name                DISAdomain2
  ip-address
  hostname            example.disa.mil
  port
  type                OCSP
  trans-PROTO         HTTP
  requestor-cert
  responder-cert
  trusted-cas         dod1 dod2 disaA disaB IBM1
  realm-id
  retry-count         1
  dead-time           0
  last-modified-by
  last-modified-date
ORACLE (cert-status-profile) #
```

Direct and Delegated Trust Models

RFC 2560 specifies that an OCSR must digitally sign OCSP responses, and that an OCSP client must validate the received signature. In prior releases, successful validation of the signed response served to authenticate the responder. Such an authentication method is referred to as a direct trust model in that it does not require confirmation from a trusted Certificate Authority (CA). Rather it requires that the OCSP client be in possession of the public counterpart of the private key used by the OCSR to sign the response. This certificate is identified by the responder-cert attribute

in the cert-status-profile configuration element. Prior to Release S-CX6.3F1, authentication via signature validation was the only authentication method provided by the OCSP client implementation.

Release S-CX6.3F1 continues support for the direct trust model, while also supporting an alternative delegated trust model as described in Section 5.4.6.2.1.6.1.e.3.c of UCR 2010. The delegated trust model requires that OCSR be authenticated by a trusted CA. Within the DISA/DoD delegated trust model, an OCSR certificate is appended to every response, thus eliminating the need for a pre-provisioned responder certificate. The appended certificate is a signing certificate issued and signed by a DoD-approved CA that issued the certificate that is being validated. These OCSR certificates have a short lifespan and are reissued regularly.

Direct Trust Model Configuration

The direct trust model is used in virtually all commercial/enterprise environments. Configuration of the direct trust model is unchanged from that contained in the latest version of your hardware or the Oracle *CSMConfiguration Guide*.

Delegated Trust Model Configuration

The delegated trust model is used exclusively in some strict DISA/DoD environments; other DISA/DoD environments may support both the direct and delegated trust models.

Use the following procedure to configure OCSP for DISA/DoD environments.

1. From superuser mode, use the following command sequence to access cert-status-profile configuration mode. While in this mode, you configure a cert-status-profile configuration element, a container for the information required to access a single, specific OCSR.

```
ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# cert-status-profile
ORACLE(cert-status-profile)#
```

2. The name attribute differentiates cert-status-profile configuration elements one from another. Each cert-status-profile provides configuration information for a single, specific OCSP responder.
3. The type attribute selects the certificate revocation check methodology, the only currently supported methodology is OCSP.
4. Retain the default value (http) for trans-protocol attribute, which identifies the transport method used to access the OCSR.
5. The ip-address attribute works in conjunction with the port attribute to provide the IP address of the OCSR.

ip-address identifies the OCSR by its IP address. port identifies the port monitored by the HTTP server for incoming OCSP requests.

The port attribute can be safely ignored if the OCSR is specified as a FQDN by the host-name attribute, but is required if the OCSR is identified by the ip-address attribute.

Allowable port values are integers within the range 1025 through 65535. In the absence of an explicitly configured value, the system provides a default value of 80, the well-known HTTP port.

6. Alternatively, use the host-name attribute to identify the OCSR.

host-name identifies the OCSR by a FQDN.

If you provide both an IPv4 address/port number and a FQDN, the Oracle CSM uses the IP address/port number and ignores the FQDN.

If values are provided for both attributes, the Security Gateway uses the IP address and ignores the host-name value.

7. The realm-id attribute specifies the realm used to access the OCSR.

In the absence of an explicitly configured value, the Oracle CSM provides a default value of wancom0, specifying OCSP transmissions across the wancom0 management interface.

If the OCSR identified by a FQDN, the realm identified by realm-id must be DNS-enabled.

8. The requester-cert attribute is meaningful only if OCSP requests are signed; ignore this attribute if requests are not signed.

RFC 2560 does not require the digital signature of OCSP requests. OCSRs, however, can impose signature requirements.

If a signed request is required by the OCSR, provide the name of the certificate configuration element that contains the certificate used to sign OCSP requests.

9. The responder-cert attribute identifies the certificate used to validate signed OCSP response — a public key of the OCSR.

In DISA/DoD environments that support the direct trust model, optionally provide the name of the certificate configuration element that contains the certificate used to validate the signed OCSP response.

If a responder-cert is provided, it is only used if the OCSP response has no appended certificates, in which case the OCSP client attempts to validate the response signature. Depending on the validation failure or success, the response is rejected or accepted.

If the OCSP response has an appended certificate or certificate chain, the responder-cert is ignored, and the trusted-cas list is used to validate the appended certificate(s).

10. The trusted-cas attribute (a list of certificate configuration objects) identifies the approved DoD-approved CAs that sign OCSR certificates.

In DISA/DoD environments that support the delegated trust model, you must provide a list of CAs used to validate the received certificate.

If a certificate or a certificate chain is appended to the OCSP response, the OCSP client verifies that the first certificate signed the response, and that the CA is trusted by the Oracle CSM (that is, the CA certificate is contained in the trusted-cas list. The client then walks through each additional certificate (if any exist) ensuring that each certificate is also trusted. If all certificates are trusted, the OCSP response is accepted; otherwise, it is rejected.

11. The retry-count attribute specifies the maximum number of times to retry an OCSP responder in the event of connection failure.

If the retry counter specified by this attribute is exceeded, the OCSP requester contacts another responder (if multiple responders have been configured) and quarantines the unavailable responder for a period defined the dead-time attribute.

In the absence of an explicitly configured value (an integer within the range 0 through 10), the Oracle CSM provides a default value of 1 (connection retries).

12. The dead-time attribute specifies the quarantine period imposed on an unavailable OCSR.

In the absence of an explicitly configured value (an integer within the range 0 through 3600 seconds), the Oracle CSM provides a default value of 0 (no quarantine period).

Customer implementations utilizing a single OCSP responder are encouraged to retain the default value, or to specify a brief quarantine period to prevent lengthy service outages.

13. Use done, exit, and verify-config to complete configuration of this cert-status-profile instance.

14. Repeat Steps 1 through 13 to configure additional cert-status-profile configuration elements.

Maintenance and Troubleshooting

show sipd acls

The show sipd acls command includes counters that track the number of endpoints demoted from trusted to untrusted and the number of endpoints demoted from untrusted to denied. For example:

```
ORACLE# show sipd acls
...
ACL Operations          ---- Lifetime ----
```

Security

	Recent	Total	PerMax
...			
Trust->Untrust	0	1	1
Untrust->Deny	0	1	1

References and Debugging

ACLI Configuration Elements

The following sections describe the Oracle CSM's configuration elements that are unique to S-CZ7.2.5.

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 CSM 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 IPv4 or IPv6 format

port—This home subscriber server's port.

- Default: 80
- Values: 1-65535

realm—Oracle CSM 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 CSM 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 CSM 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**.

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.

- 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 CSM 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 ACLI prompt is: **configure terminal > session-router > session-group**.

SNMP MIBs and Traps

The following MIBs and traps are supported for the Oracle CSM. 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)		
apLicenseAuthFeature	apLicenseEntry: 1.3.6.1.4.1.9148.3.5.1.1.1.20	If authorization and authentication is allowed for the Oracle CSM, 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 CSM 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)		

References and Debugging

SNMP GET Query Name	Object Identifier Name: Number	Description
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 CSM 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 CSM 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-ac1
              realm=net172 local=172.16.101.13:5060 UA=172.16.17.100:5060
  SD-Contact: <sip:111111-s37q249kvluua@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: 07ebbdebfd64a48985bb82fa8b4c595@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 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
```

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
```

References and Debugging

```
Avg Latency=0.000 for 0
Max Latency=0.000
NOTIFY
----- Server -----
Message/Event      Recent      Total      PerMax      Client -----
                   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
Active  -- Period --  ----- Lifetime -----
User Entries      0      High  Total  Total  PerMax  High
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'=='.*'))
```

References and Debugging

```
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
hss1                 192.168.207.21:45463 192.168.200.232:3872 Up
-----
18:53:25-105
HSS Status
Active              -- Period -- ----- Lifetime -----
High Total        Total PerMax High
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 CSM has sent a CER and is waiting for a CEA response.
Inactive	The Oracle CSM has sent a CER but has not received a CEA response.
Down	The Oracle CSM is not attempting to establish a connection with the HSS.

Oracle CSM 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              -- Period -- ----- Lifetime -----
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
-----
Recent              Lifetime -----
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	-- Period --		----- Lifetime -----		
		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
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

References and Debugging

```
sk3          192.168.19.1:8889          Up
12:56:41-184
HTTP Status
Active      High    Total    Total    Lifetime
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
HTTP Status
Active      High    Total    Total    Lifetime
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
Active      High    Total    Total    Lifetime
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
```

Verify Config

The Oracle CSM performs application specific verification checks when you save a config with the save-config CLI command. These checks are in addition to baseline Oracle CSM 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 CSM limits resource utilization to maintain operational stability. Resources managed this way include:

- CPU
- Memory (heap)

CPU Overload Protection

The default behavior, by which the Oracle CSM limits CPU utilization to maintain operational stability, is as follows:

- When CPU exceeds 90%, the Oracle CSM no longer accepts new registrations. The Oracle CSM replies to these messages with 5xx messages. The Oracle CSM continues to accept registration refresh, in-dialog call and subscription messages.
- When CPU reaches 100%, the Oracle CSM 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 CPU utilization using the following command.

```
ORACLE# show platform cpu-load
```

The first threshold is set with the load-limit option in the sip-config object. The command sequence below is displayed as an example of the syntax, and sets the load-limit to 80%.

```
ORACLE# configure terminal
ORACLE (configure)# session-router
ORACLE (session-router)# sip-interface
ORACLE (sip-interface)# options +load-limit 80
ORACLE (sip-interface)# done
```

When the load-limit threshold is exceeded, the system rejects most out-of-dialog messages with local-response-map error messages. Reg-Event Subscriptions are the only out-of-dialog messages accepted by the system when operating above this threshold.

 **Note:** An Oracle CSM configured to operation as an SLRM rejects all messages when CPU utilization exceeds this threshold.

The second threshold is set with the transport-load-limit option in the sip-config object. The default value of transport-load-limit is 100%. The command sequence below is displayed as an example of the syntax, and sets the transport-load-limit to 90%.

```
ORACLE# configure terminal
ORACLE (configure)# session-router
ORACLE (session-router)# sip-config
ORACLE (sip-config)# transport-load-limit 90
ORACLE (sip-config)# done
```

When this threshold is exceeded, the system drops all messages.

Heap Utilization

The Oracle CSM limits memory utilization to maintain operational stability, as follows:

- When heap utilization exceeds the default (75%) or configured memory utilization threshold, the Oracle CSM no longer accepts new registrations. The Oracle CSM replies to these messages with 5xx messages. The Oracle CSM continues to accept registration refreshes, in-dialog calls and subscriptions.
- When heap utilization exceeds its default (90%) or configured threshold, the Oracle CSM 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 change the first threshold, for example from its default of 75% to 80%, using the option shown below.

```
ORACLE# configure terminal
ORACLE (configure) # session-router
ORACLE (session-router) # sip-config
ORACLE (sip-config) # +options memory-overload-protect 80
ORACLE (sip-config) # done
```

The user can change the default drop-all threshold, from 90% to 85% 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 85
ORACLE (sip-config) # done
```

Oracle Sc Interface Support

The Oracle CSM supports numerous AVPs in its Diameter-based Sc implementation. Currently AVPs belong to:

- The Diameter base AVPs found in RFC3588 and RFC4006.
- For 3GPP AVPs, if not specified by this document, their definition follows corresponding 3GPP specifications.
- Oracle proprietary Sc AVPs, described below.

Sc Interface and Command Codes

The table below provides the codes for the proprietary Sc interface commands.

Specification: Oracle Proprietary

Application-ID: 9999 (Oracle-Acme-Sc)

Vendor-ID:9148

Command-Name	Abbreviation	Code
Service Association Request	SVR	6000
Service Association Answer	SVA	6000
Core Registration Request	CRR	6001
Core Registration Answer	CRA	6001

Diameter AVP Notation

3GPP 32.299 states the following symbols are used in the message format definitions:

- <AVP> indicates a mandatory AVP with a fixed position in the message.
- {AVP} indicates a mandatory AVP in the message.
- [AVP] indicates an optional AVP in the message.
- *AVP indicates that multiple occurrences of an AVP is possible.

This syntax is used to document the Sc Interface messages herein.

Table Explanation

Each row in the following AVP tables contain:

- AVP Name
- AVP Number
- Reference where the AVP was defined
- Type of data format used to express the AVP's data
- If a grouped AVP, the names of the AVPs in the group

CER Message Format

The following table contains the top level AVPs that may be present in a message generated by the Oracle CSM.

AVP	Number	Reference	Type	Grouped
{ Session-Id }	263	Base	UTF8String	
{ Origin-Host }	264	Base	DiameterIdentity	
{ Host-IP-Address }	257	Base	Address	
{ Vendor-Id }	266	Base	Unsigned32	
{ Product-Name }	269	Base	UTF8String	
[Vendor-Specific-Application-ID]	260	Base	Grouped	

CEA Message Format

The following table contains the top level AVPs that may be present in an SLRM-generated CEA message.

AVP	Number	Reference	Type	Grouped
{ Result-Code }	268	Base	Unsigned32	
{ Origin-Host }	264	Base	DiameterIdentity	
{ Origin-Realm }	296	Base	DiameterIdentity	
{ Host-IP-Address }	257	Base	Address	
{ Vendor-Id }	266	Base	Unsigned32	
{ Product-Name }	269	Base	UTF8String	
[Vendor-Specific-Application-ID]	260	Base	Grouped	

DWR Message Format

The following table contains the top level AVPs that may be present in a DWR message.

AVP	Number	Reference	Type	Grouped
{ Origin-Host }	264	Base	DiameterIdentity	
{ Origin-Realm }	296	Base	DiameterIdentity	

DWA Message Format

The following table contains the top level AVPs that may be present in a DWA message.

AVP	Number	Reference	Type	Grouped
{ Result-Code }	268	Base	Unsigned32	
{ Origin-Host }	264	Base	DiameterIdentity	
{ Origin-Realm }	196	Base	DiameterIdentity	
[Error-Message]	281	Base	UTF8String	

SVR Message Format

The following table contains the top level AVPs present in a Oracle CSM-generated SVR message.

AVP	Number	Reference	Type	Grouped
{ Vendor-Specific-Application-ID }	260	Base	Grouped	Vendor-Id Auth-Application-ID
{ Origin-Host }	264	Base	DiameterIdentity	
{ Origin-Realm }	296	Base	DiameterIdentity	
{ Srv-Assoc-Id }	4010	Oracle	String	
{ Req-Type }	4000	Oracle	Enumerated	
{ Sc-Proto-Ver }	4005	Oracle	Unsigned32	
{ Soft-Version }	4014	Oracle	String	
{ Srv-Assoc-Exp }	4012	Oracle	Integer	
{ Destination-Realm-AVP }	283	Base	DiameterIdentity	
{ Cluster_Id }	4001	Oracle	Integer	
{ Pct-Used-Cpu }	4002	Oracle	Unsigned32	
{ Pct-Used-Mem }	4003	Oracle	Unsigned32	
{ Eps-Srv-Count }	4004	Oracle	Unsigned32	
[Max-Eps-Supp]	4006	Oracle	Unsigned32	

SVA Message Format

The following table contains the top level AVPs present in an SLRM-generated SVA message.

AVP	Number	Reference	Type	Grouped
{ Vendor-Specific-Application-ID }	260	Base	Grouped	Vendor-Id Auth-Application-ID

Oracle Sc Interface Support

AVP	Number	Reference	Type	Grouped
[Result-Code]	268	Base	Unsigned32	
{ Origin-Host }	264	Base	DiameterIdentity	
{ Origin-Realm }	296	Base	DiameterIdentity	
{ Service-Assoc-Id }	4010	Oracle	String	
{ Req-Type }	4000	Oracle	Enumerated	
{ Sc-Proto-Ver }	4005	Oracle	Unsigned32	
{ Soft-Ver }	4014	Oracle	Integer	
{ Srv-Assoc-Exp }	4012	Oracle	Unsigned32	
{ Vendor-Specific-Application-ID }	260	Base	Grouped	Vendor-Id Experimental-Result-Code

CRR Message Format

The core registration request provides and updates the SLRM function with the request from the Oracle CSM to provide service for the ims-core specified as a member of that core's load balanced Oracle CSMs.

AVP	Number	Reference	Type	Grouped
{ Vendor-Specific-Application-ID }	260	Base	Grouped	Vendor-Id Auth-Application-ID
{ Origin-Host }	264	Base	DiameterIdentity	
{ Origin-Realm }	296	Base	DiameterIdentity	
{ Srv-Assoc-Id }	4010	Oracle	String	
{ Core-Reg-Type }	4007	Oracle	Enumerated	
{ Core-Reg-Exp }	4013	Oracle	Integer	
{ Destination-Realm }	283	Base	DiameterIdentity	
{ Core-info }	4008	Base	Grouped	ims-core service-info

CRA Message Format

The core registration answer provides the Oracle CSM with the result of its attempt to register itself for servicing the core specified in the request.

AVP	Number	Reference	Type	Grouped
{ Vendor-Specific-Application-ID }	260	Base	Grouped	Vendor-Id Auth-Application-ID
[Result-Code]	268	Base	Unsigned32	

AVP	Number	Reference	Type	Grouped
{ Origin-Host }	264	Base	DiameterIdentity	
{ Origin-Realm }	296	Base	DiameterIdentity	
{ Core-Reg-Type }	4007	Oracle	Enumerated	
{ Core-Reg-Exp }	4013	Oracle	Unsigned32	
{ Vendor-Specific-Application-ID }	260	Base	Grouped	Vendor-Id Experimental-Result-Code

Proprietary Grouped AVP Format

The following sections display the format of the grouped AVPs related to SLRM.

Core-Info AVP

The core-info AVP resides within the core registration request and answer sequence. It provides the SLRM function a reference with which the SLRM can group Oracle CSMs for load balancing registrations.

AVP	Number	Reference	Type
Core-Info ::= <AVP header>	4009	Oracle	
[Ims-Core]	4008	Oracle	String
[Service-Info]	4015	Oracle	Grouped

Service-Info AVP Format

The Sc interface's service address port AVP is a grouped AVP nested within the core-info AVP. It allows for transmission of multiple service route records within the AVP. This provides the SLRM function with the routes used to access the applicable ims-cores as accessed via this Oracle CSM.

AVP	Number	Reference	Type
Service info ::= <AVP header>	4015	Oracle	String
[service-route]	4011	Oracle	String

RTC Support

This appendix summarizes real-time configuration (RTC) support status for the Oracle CSM . The table below lists which configuration elements are supported by RTC and which are not.

ACLI Configuration Elements	Parameter Details
Access Control	
Accounting Config	
Authentication	
Certificate Record	
Class Profile	
Codec Policy	
DNS ALG Service	
DNS Config	
Enum	
External Policy Server	
Host Route	
IPSEC	
Licensing	
Local Policy	
Local Response Map	
Local Routing Config	
Media Manager	The Media Manager element is supported with the exception of the following parameters: <ul style="list-style-type: none">• red-flow-port• red-max-trans• red-sync-start-time

RTC Support

ACLI Configuration Elements	Parameter Details
	<ul style="list-style-type: none"> red-sync-comp-time red-mgcp-port
Media Policy	
Media Profile	
Network Interface	
Net Management Control	
Network Parameters	<p>The Network Parameters element is supported with the exception of the following parameters:</p> <ul style="list-style-type: none"> SCTP parameters
NTP Sync	
Packet Trace Config	
Q850 SIP Map	
Realm Config	
Redundancy Config	<p>The Redundancy Config element is supported with the exception of the following parameters:</p> <ul style="list-style-type: none"> state port cfg-port cfg-max-trans cfg-sync-start-time cfg-sync-comp-time
Session Agent	
Session Group	
Session Router	
Session Constraints	
Session Translation	
SIP Config	<p>The SIP Config element is supported with the exception of the following parameters:</p> <ul style="list-style-type: none"> red-sip-port red-max-trans red-sync-start-time red-sync-stop-time
SIP Feature	
SIP Interface	collect>boot-state
SIP Manipulation	
SIP NAT	<p>The SIP NAT element is supported with the exception of the following parameters:</p>

ACLI Configuration Elements	Parameter Details
	<ul style="list-style-type: none"> ext-address
SIP Response Map	
SNMP	
Static Flow	
Steering Pool	
Surrogate Agent	
System	<p>The System Config element is supported with the exception of the following parameters:</p> <ul style="list-style-type: none"> options
Test Pattern Rule	
Test Policy	
Test Translation	
TLS Global	<p>When configured to support MSRP over TLS, this configuration element is not RTC enabled. If a single TLS profile is used by both SIP and MSRP, RTC changes will be applied for the SIP TLS configuration ONLY.</p>
TLS Profile	
Translation Rules	
Trap Receiver	

CSM and SLRM Base Configuration Elements

This appendix presents base configuration settings required for CSM and SLRM deployments.

CSM Base Configuration Elements

This appendix provides configuration samples of the elements that are required for minimal Oracle CSM operation.

The SIP Config must be enabled	
<pre>sip-config state</pre>	<pre>enabled</pre>
You must have a default gateway in your system-config	
<pre>system-config default-gateway</pre>	<pre>10.0.0.1</pre>
You must have a core physical interface	
<pre>phy-interface name operation-type port slot</pre>	<pre>s0p1 Media 1 0</pre>
You must have a core network interfaces	
<pre>network-interface name sub-port-id ip-address netmask gateway</pre>	<pre>s0p1 0 192.170.2.100 255.255.255.0 192.170.2.1</pre>
You must have a core realm	
<pre>realm-config identifier</pre>	<pre>core1</pre>

CSM and SLRM Base Configuration Elements

addr-prefix	0.0.0.0
network-interfaces	s0p1:0
You must have a core SIP interface	
sip-interface	
state	enabled
realm-id	core1
sip-port	
address	192.170.2.100
registration-caching	enabled
You must have an ENUM Configuration	
enum-config	
name	My_e164_cfg
realm-id	core1
enum-servers	192.170.2.201
You must have a Subscriber Database	
home-subscriber-server	
name	My_HSS
address	192.170.2.202
realm	core1
You must have a Registration Event Profile	
regevent-notification-profile	
name	My_reg_event_Profile
You must have an Authentication Profile	
sip-authentication-profile	
name	My_Auth_Profile
methods	REGISTER
anonymous-methods	*
digest-realm	My_Digest_Realm.com
credential-retrieval-method	Cx
credential-retrieval-config	My_HSS
You must have a Sip-Registrar	
sip-registrar	
name	My_Registrar_Name
domains	my_customer1.com
subscriber-database-method	Cx
subscriber-database-config	My_HSS
authentication-profile	My_Auth_Profile
home-server-route	sip:192.170.2.201:5060
routing-precedence	REGISTRAR
egress-realm-id	core1
options	e164-primary-config=enum:My_e164_Cfg
regevent-notification-profile	My_reg_event_Profile

SLRM Base Configuration Elements

This appendix provides configuration samples of the elements that are required for minimal Oracle SLRM operation. Configuration includes settings required for both the SLRM and the CSM.

SLRM Configuration

Set the component type to SLRM	
<pre>ORACLE# set-component-type core-load-balancer WARNING: Changing component type is service impacting. ***** Ensure that you follow these steps if you choose to change the component type: 1. Issue the delete-config command. 2. Reboot. ***** Continue with the change [y/n]?:</pre>	
The SIP Config must be enabled	
<pre>sip-config state</pre>	<pre>enabled</pre>
You must have a default gateway in your system-config	
<pre>system-config default-gateway</pre>	<pre>10.0.0.1</pre>
You must have a core physical interface	
<pre>phy-interface name operation-type port slot</pre>	<pre>s0p1 Media 1 0</pre>
You must have a core network interface	
<pre>network-interface name sub-port-id ip-address netmask gateway</pre>	<pre>s0p1 0 192.170.2.100 255.255.255.0 192.170.2.1</pre>
You must have a core realm	
<pre>realm-config identifier addr-prefix network-interfaces</pre>	<pre>core1 0.0.0.0 s0p1:0</pre>
You must have an ENUM Configuration	
<pre>enum-config name realm-id enum-servers</pre>	<pre>My_e164_cfg core1 192.170.2.201</pre>
You must have a Subscriber Database	
<pre>home-subscriber-server name address realm</pre>	<pre>My_HSS 192.170.2.202 core1</pre>

CSM and SLRM Base Configuration Elements

Configure the Load Balancer Interface

```
lb-interface
  name          My_lb_int
  address       192.170.2.101
  realm         core1
```

Configure the Load Balancer's Core Config

```
lb-core-config
  core-name      My_core_config
  domains        My_first_domain,
Other_domains
  forwarding-realm  core1
  hss-config       My_HSS
  options          e164-primary-config=enum:My_e164_Cfg
```

CSM Configuration (for SLRM)

The System Config must specify a cluster-ID

```
system-config
  service-cluster-id  cluster1
```

The SIP Config must be enabled and define a hostname

```
sip-config
  hostname          CSM1
  registration-cache-limit  500000
```

Set your Load Balancer Config

```
lb-cfg
  name          My_lb_config
  address       192.170.2.101
  realm         core1
```

Set your sip-registrar for Load Balancing

```
sip-registrar
  ims-core      My_core_config
  lb-list       My_lb_config
```