Oracle® Communications Core Session Manager

Essentials Guide

S-CZ9.1.5
F52423-04
January 2023

ORACLE

Oracle Communications Core Session Manager Essentials Guide, S-CZ9.1.5

F52423-04

Contents

6 The Session Load Balancer and Route Manager

7 Third Party Registration

8 References and Debugging

A Oracle Sc Interface Support

B CSM and SLRM Base Configuration Elements

C Caveats and Known Issues

About This Guide

This Essentials Guide provides information about:

- Basic concepts that apply to the key features and abilities of your Oracle Communications Core Session Manager (OCCSM)
- Information about how to load the OCCSM system software image you want to use and establish basic operating parameters
- System-level functionality for the OCCSM
- Configuration of key components of the OCCSM
- Direction to OCSBC documentation for configuration of cross-product components and features that apply to the OCCSM
- Operational description, configuration instructions and interface detail on the Oracle Communications Session Load Balancer and Route Manager component of the OCCSM

Supported Platforms

Release Version S-CZ9.1.5 includes the Oracle Core Session Manager (CSM) product. The Oracle CSM is supplied as virtual machine software or as a software-only delivery suitable for operation on server hardware as well as public and private clouds.

Related Documentation

Version S-CZ9.1.5 software relies on version SCZ910 documentation for some documentation. The following table lists the members that comprise the documentation set for this release:

Document Name	Document Description
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 Service Provider Oracle Communications Core Session Manager.
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 Oracle Communications Core Session Manager 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 Communication's enterprise MIBs, general trap information, including specific details about standard traps and enterprise traps, Simple Network Management Protocol (SNMP) GET query information (including standard and enterprise SNMP GET query names, object identifier names and numbers, and descriptions), examples of scalar and table objects.

Document Name	Document Description
Accounting Guide	Contains information about the Oracle Communications Core Session Manager's accounting support, including details about RADIUS and Diameter accounting.
HDR Resource Guide	Contains information about the Oracle Communications Core Session Manager's Historical Data Recording (HDR) feature. This guide includes HDR configuration and system-wide statistical information.
SBC Family Security Guide	Contains information about security considerations and best practices from a network and application security perspective for the Oracle Communications Core Session Manager family of products.
Installation and Platform Preparation Guide	Contains information about upgrading system images and any pre-boot system provisioning.
Call Traffic Monitoring Guide	Contains information about traffic monitoring and packet traces as collected on the system.
HMR Resource Guide	Contains information about configuring and using Header Manipulation Rules to manage service traffic.

Revision History

Date	Description
July 2022	• Initial Release
July 2022	• Adds NNC-OCSDM XSD number to Co-Product Support
December 2022	• Removes references to CSM as BGCF.
January 2023	• Adds PCFA retrieval support for S-Cz9.1.5p1.

My Oracle Support

My Oracle Support (https://support.oracle.com) is your initial point of contact for all product support and training needs. A representative at Customer Access Support (CAS) can assist you with My Oracle Support registration.

Call the CAS main number at 1-800-223-1711 (toll-free in the US), or call the Oracle Support hotline for your local country from the list at http://www.oracle.com/us/support/contact/index.html. When calling, make the selections in the sequence shown below on the Support telephone menu:

1. Select `2` for New Service Request.
2. Select `3` for Hardware, Networking, and Solaris Operating System Support.
3. Select one of the following options:
   - For technical issues such as creating a new Service Request (SR), select `1`.
   - For non-technical issues such as registration or assistance with My Oracle Support, select `2`.

You are connected to a live agent who can assist you with My Oracle Support registration and opening a support ticket.

My Oracle Support is available 24 hours a day, 7 days a week, 365 days a year.

Emergency Response

In the event of a critical service situation, emergency response is offered by the Customer Access Support (CAS) main number at 1-800-223-1711 (toll-free in the US), or call the Oracle Support hotline for your local country from the list at http://www.oracle.com/us/support/contact/index.html. The emergency response provides immediate coverage, automatic escalation, and other features to ensure that the critical situation is resolved as rapidly as possible.

A critical situation is defined as a problem with the installed equipment that severely affects service, traffic, or maintenance capabilities, and requires immediate corrective action. Critical situations affect service and/or system operation resulting in one or several of these situations:

- A total system failure that results in loss of all transaction processing capability
- Significant reduction in system capacity or traffic handling capability
- Loss of the system's ability to perform automatic system reconfiguration
- Inability to restart a processor or the system
- Corruption of system databases that requires service affecting corrective actions
- Loss of access for maintenance or recovery operations
- Loss of the system ability to provide any required critical or major trouble notification

Any other problem severely affecting service, capacity/traffic, billing, and maintenance capabilities may be defined as critical by prior discussion and agreement with Oracle.

Locate Product Documentation on the Oracle Help Center Site

Oracle Communications customer documentation is available on the web at the Oracle Help Center (OHC) site, http://docs.oracle.com. You do not have to register to access these documents. Viewing these files requires Adobe Acrobat Reader, which can be downloaded at http://www.adobe.com.

1. Access the Oracle Help Center site at http://docs.oracle.com.
2. Click **Industries**.
3. Under the Oracle Communications sub-header, click the **Oracle Communications documentation** link.
   The Communications Documentation page appears. Most products covered by these documentation sets appear under the headings "Network Session Delivery and Control Infrastructure" or "Platforms."
4. Click on your Product and then Release Number.
   A list of the entire documentation set for the selected product and release appears.
5. To download a file to your location, right-click the **PDF** link, select **Save target as** (or similar command based on your browser), and save to a local folder.

Access to Oracle Support

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

1

Introduction to the Core Session Manager S-CZ9.1.5

This Oracle Communications Core Session Manager (OCCSM) Release Introduction chapter provides the following information about this product:

- Supported platforms and hardware requirements
- An overview of the new features available in this release
-
- A summary of changes the OCCSM interfaces including the ACLI, MIB Support, and accounting interfaces.
- A summary of behavioral changes
- A summary of known issues and caveats

Review the S-CZ9.1.0 OCSBC Release Notes for further information on this S-CZ9.1.5 release of the OCCSM. There is overlap between these products, including functionality, features, compatibility, known issues and caveats. The S-CZ9.1.0 OCSBC Release Notes provides complimentary detail to this *Essentials Guide* and documents the aspects of the products that are common to both.

> **Note:**
>
> The Oracle CSM requires a license to use TLS over SIP interfaces.

New Features

The S-CZ9.1.5 release of the OCCSM supports the following new features and enhancements.

The S-CZ9.1.5 version of the OCCSM also inherits some of the features from the S-Cz9.1.0 version of the OCSBC. Not all of the OCSBC features are supported for S-CZ9.1.5, based on their relevance to the OCCSM. Contact your Oracle representative to verify whether specific OCSBC features are supported in S-CZ9.1.5 before attempting to deploy them.

TLS for Signaling Interfaces

This OCCSM software version supports the TLS protocol for signaling interfaces.

See the *Transport Layer Security* Chapter of this *OCCSM Essentials Guide*. Also see the *TLS Cipher Updates* section of this *Introduction to the Core Session Manager S-CZ9.1.5* Chapter.

PCFA Retrieval

Beginning with S-Cz9.1.5p1, you can configure the OCCSM to retrieve P-Charging Function Addresses (PCFA) information from a Home Subscriber Server (HSS). The OCCSM can get

this address from the HSS over the Cx interface during subscriber registration. The OCCSM, subsequently, can use PCFA addresses to populate the outgoing P-Charging-Function-Address headers it sends to an application server in INVITEs and registration success messages it sends to a P-CSCF.

See the *Retrieving the P-Charging Function Address from the HSS* section in the *Oracle CSM Supporting the IMS Core* chapter.

Platform Support

The OCCSM S-CZ9.1.5 software supports the following platforms.

Supported Hypervisors for Private Virtual Infrastructures

Oracle supports installation of the OCCSM on the following hypervisors:

- KVM: Linux kernel version (3.10.0-123 or later), with KVM/QEMU (2.9.0_16 or later) and libvirt (3.9.0_14 or later)
- VMware: vSphere ESXi (Version 6.5 or later)

Compatibility with OpenStack Private Virtual Infrastructures

Oracle distributes Heat templates for the Newton and Pike versions of OpenStack. Use the Newton template when running either the Newton or Ocata versions of OpenStack. Use the Pike template when running Pike or a later version of OpenStack.

Supported Public Cloud Platforms

In S-CZ9.1.5, you can run the OCCSM on the following public cloud platforms.

- Oracle Cloud Infrastructure (OCI) - After deployment, you can change the shape of your machine by, for example, adding disks and interfaces. OCI Cloud Shapes and options validated in this release are listed in the table below.

Shape	OCPUs/ VCPUs	vNICs	Tx/Rx Queues	Max Forwarding Cores	DoS Protection
VM.Standard2.4	4/8	4	2	2	Y
VM.Standard2.8	8/16	8	2	2	Y
VM.Standard2.16	16/32	16	2	2	Y

Networking using image mode [SR-IOV mode - Native] is supported on OCI. PV and Emulated modes are not currently supported.

Platform Hyperthreading Support

Of the supported hypervisors, only VMware does not expose SMT capability to the OCCSM. For public cloud deployment, OCI enables SMT by default and exposes it to the OCCSM.

Virtual Machine Platform Resources

A Virtual Network Function (VNF) requires the CPU core, memory, disk size, and network interfaces specified for operation. Deployment details, such as the use of distributed DoS protection, dictate resource utilization beyond the defaults.

Default VM Resources

VM resource configuration defaults to the following:

- 4 CPU Cores
- 8 GB RAM
- 40 GB hard disk (pre-formatted)
- 8 interfaces as follows:
  - 1 for management (wancom0)
  - 2 for HA (wancom1 and 2)
  - 1 spare
  - 4 for media

Interface Host Mode for Private Virtual Infrastructures

The OCCSM VNF supports interface architectures using Hardware Virtualization Mode - Paravirtualized (HVM-PV):

- ESXi - No manual configuration required.
- KVM - HVM mode is enabled by default. Specifying PV as the interface type results in HVM plus PV.

Supported Interface Input-Output Modes for Private Virtual Infrastructures

- Para-virtualized
- SR-IOV
- PCI Passthrough
- Emulated - Emulated is supported for management interfaces only.

Supported Ethernet Controller, Driver, and Traffic Type based on Input-Output Modes

The following table lists supported Ethernet Controllers (chipset families) and their supported driver that Oracle supports for Virtual Machine deployments. Reference the host hardware specifications, where you run your hypervisor, to learn the Ethernet controller in use. The second table provides parallel information for virtual interface support. Refer to the separate platform benchmark report for example system-as-qualified performance data.

For KVM and VMware, accelerated media/signaling using SR-IOV and PCI-pt modes are supported for the following card types.

Ethernet Controller	Driver	SR-IOV	PCI Passthrough
Intel 82599 / X520 / X540	ixgbe	M	M

Ethernet Controller	Driver	SR-IOV	PCI Passthrough
Intel i210 / i350	igb	M	M
Intel X710 / XL710	i40e	M	M
Intel X710 / XL710 / XXC710	i40en[1]	M	M
Mellanox Connect X-4	mlx5	M	M

[1] This driver is supported on VMware only.

For PV mode (default, all supported hypervisors), the following virtual network interface types are supported. You can use any make/model NIC card on the host as long as the hypervisor presents it to the VM as one of these vNIC types.

Virtual Network Interface	Driver	W/M
Emulated	e1000	W
KVM (PV)	virtio	W/M
Hyper-V (PV)	NetVSC	M
VMware (PV)	VMXNET3	W/M

Emulated NICs do not provide sufficient bandwidth/QoS, and are suitable for use as management only.

- W - wancom (management) interface
- M - media interface

> **Note:**
>
> Accelerated media/signaling using SR-IOV (VF) or PCI-pt (DDA) modes are not currently supported for Hyper-V or XEN when running on Private Virtual Infrastructures.

CPU Core Resources

The OCCSM S-CZ9.1.5 VNF requires an Intel Core2 processor or higher, or a fully emulated equivalent including 64-bit SSSE3 and TSC support.

If the hypervisor uses CPU emulation, for qemu for example, Oracle recommends that you set the deployment to pass the full set of host CPU features to the VM.

DPDK Reference

The OCCSM relies on DPDK for packet processing and related functions. You may reference the Tested Platforms section of the DPDK release notes available at https://doc.dpdk.org. This information can be used in conjunction with this Release Notes document for you to set a baseline of:

- CPU
- Host OS and version
- NIC driver and version

- NIC firmware version

> **Note:**
>
> Oracle only qualifies a specific subset of platforms. Not all the hardware listed as supported by DPDK is enabled and supported in this software.

The DPDK version used in this release is:

- 20.11.2

Upgrade Information

This section provides key information about upgrading to this software version.

Supported Upgrade Paths - Hitless

The OCCSM supports the following hitless upgrade and rollback paths:

- 7.3.5 to S-Cz9.1.5 (VMware platform)
- S-CZ8.2.5p2 and forward to S-CZ9.1.5
- S-CZ8.2.5m1 to S-CZ9.1.5
- S-CZ8.4.5 to S-CZ9.1.5

Supported Upgrade Paths - Hitless Upgrade/Non-Hitless Rollback

The OCCSM supports the following hitless upgrade and non-hitless rollback paths:

- S-CZ8.2.5 to S-CZ9.1.5
- S-CZ8.2.5p1 to S-CZ9.1.5

Non-Hitless Upgrade Path

You can perform the following non-hitless upgrade and rollback:

- 7.3.5 to S-Cz9.1.5 (KVM platform)

When upgrading to this release from a release older than the previous release, read all intermediate Release Notes for notification of incremental changes.

Upgrade Checklist

Before upgrading the Oracle Communications Core Session Manager software:

1. Obtain the name and location of the target software image file from either Oracle Software Delivery Cloud, https://edelivery.oracle.com/, or My Oracle Support, https://support.oracle.com, as applicable.
2. Provision platforms with the Oracle Communications Core Session Manager image file in the boot parameters.
3. Run the **check-upgrade-readiness** command and examine its output for any recommendations or requirements prior to upgrade.
4. Verify the integrity of your configuration using the ACLI **verify-config** command.

5. Back up a well-working configuration. Name the file descriptively so you can fall back to this configuration easily.
6. Refer to the Oracle Communications Core Session Manager Release Notes for any caveats involving software upgrades.

Upgrade and Downgrade Caveats

The following items provide key information about upgrading and downgrading with this software version.

> **Note:**
>
> Upgrading to this Release from releases earlier than S-CZ9.1.5:
> The S-CZ9.1.5 release included significant changes that hardened the security posture of the CSM. These changes required your careful evaluation regarding functionality when upgrading to S-CZ9.1.5. These changes are also applicable to customers upgrading from releases prior to S-CZ9.1.5 to this release. Take care to review this information in the S-Cz9.1.0 Release Notes.

Update known_hosts File

While there are no usability changes to SSH and SFTP, the OCCSM will regenerate its SSH host certificate after upgrading to S-CZ9.1.5 from a previous version or downgrading from S-CZ9.1.5 to a previous version. Existing keys from prior releases will not work after the upgrade. To avoid warnings about mismatched fingerprints, remove the old host keys from the known_hosts file of a system that wants to connect to the OCCSM.

SSH Keys

Before upgrading to this release, delete any imported public keys using the `ssh-pub-key delete <key-name>` command. Because the commands for SSH key management have changed from 8.2.5 to 9.1.5, you will not be able to delete old 8.2.5-type SSH keys using 9.1.5 (or later) commands. After upgrading, re-import any required public keys. See "Manage SSH Keys" in the *Configuration Guide*.

SSH Keys and Push Receivers

The OCCSM acts as an SFTP client when push-receivers are configured. If you use push-receivers and upgrade to 8.4.5 or later:

1. Because the OCCSM generates a new host key during an upgrade, the OCCSM's new host key needs to be copied to the authorized_keys file on the SFTP server. Use the command `show security public-host-key rsa` to view the OCCSM's new host key.
2. Reimport the SFTP server's host key as a known-host into the OCCSM. See "SSH Key Management" in the Configuration Guide for importing a known-host key.
3. In the **push-receiver** element, verify the **public-key** attribute is empty.

If you downgrade from 9.1.5 to a previous release, copy the public host key to the authorized_keys file of the SFTP server and reset the value of **public-key** in the **push-receiver** configuration element.

Feature Entitlements

You enable the features that you purchased from Oracle by installing license keys using the **system, license** configuration element.

> **Note:**
>
> This release does not use the **setup product** or the **setup entitlements** commands to specify product type and enable features.

Encryption for Virtual CSM

You enable encryption (TLS) for virtual system SIP interfaces using a license key.

Feature	License Key
Transport Layer Security Sessions	TLS

Request license keys at the License Codes website at http://www.oracle.com/us/support/licensecodes/acme-packet/index.html.

To enable the preceding features, you install a license key using the **system**, **license** configuration element. Refer to the *Getting Started* Chapter in the *S-CZ9.1.0 ACLI Configuration Guide* for instructions on installing and managing licenses. Note that you must install licenses on both devices to enable this feature within an HA deployment.

After you install the license keys, you must reboot the system for them to take effect and for you to see them using the **show features** command.

TLS Cipher Updates

This section gives detail on supported ciphers.

Oracle recommends the following ciphers, and includes them in the DEFAULT cipher list:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256

Oracle supports the following ciphers, but does not include them in the DEFAULT cipher list:

- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256

- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA256
- TLS_RSA_WITH_AES_256_GCM_SHA384
- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_AES_128_GCM_SHA256

Oracle supports the following ciphers for debugging purposes only:

- TLS_RSA_WITH_NULL_SHA256 (debug only)
- TLS_RSA_WITH_NULL_SHA (debug only)
- TLS_RSA_WITH_NULL_MD5 (debug only)

Oracle supports the following ciphers, but considers them not secure. They are not included in the DEFAULT cipher-list, but they are included when you set the **cipher-list** attribute to **ALL**. Note that they trigger **verify-config** error messages.

- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA

To configure TLS ciphers, use the **cipher-list** attribute in the **tls-profile** configuration element.

WARNING:

When you set **tls-version** to either **tlsv1** or **tlsv11** and you want to use ciphers that Oracle considers not secure, you must manually add them to the **cipher-list** attribute.

Note:

The default is TLSv1.2. Oracle supports TLS1.0 and TLS1.1 for backward compatibility only and they may be deprecated in the future.

System Capacities

System capacities vary across the range of platforms that support the Oracle Communications Core Session Manager. To query the current system capacities for the platform you are using, execute the **show platform limits** command.

Coproduct Support

The following products and features run in concert with the Oracle Communications Core Session Manager for their respective solutions. Contact your Sales representative for further support and requirement details.

Oracle Communications Session Delivery Manager

The following versions of Oracle Communications Session Deliver Manager (OCSDM) support this GA release of the OCCSM:

- 8.2.5
- 9.0.0

When you configure your system as an SLRM, it supports these same versions.

OCSDM support for configuration parameters that are new in OCCSM 9.1.5 is planned for the next GA release of SDM at the earliest. Parameters currently supported by OCSDM, up to and including those in S-Cz8.4.5, remain supported when using OCCSM 9.1.5.

> **Note:**
>
> Customers who wish to manage release S-Cz9.1.5 and higher with OCSDM versions 8.2.5 and above need to load an updated XSD into OCSDM. This file can be found by searching My Oracle Support for NNC-OCSDM XSD 34352323.

Oracle Communications Operations Manager

The following version of Oracle Communications Operations Manager (OCOM) support this GA release of the OCCSM:

- 4.3
- 4.4
- 5.0
- 5.1

OCOM is not supported when you configure your system as a Session Load Balancer and Route Manager (SLRM).

Documentation Changes

There are no structural changes made to the Oracle Communications Core Session Manager (OCCSM) documentation set for S-CZ9.1.5. A standard chapter titled Transport Layer Security has been added to accommodate that new feature that allows you to configure TLS over signaling interfaces.

Behavioral Changes

There are no default behavioral changes to the Oracle Communications Core Session Manager (OCCSM) in this software release.

Patches Included in This Release

The following information assures you that when upgrading, the S-CZ9.1.5 release includes defect fixes from neighboring patch releases.

Neighboring patches included in this release

- S-Cz8.2.5m1p4
- S-Cz8.4.5p1

2

Oracle Communications Core Session Manager Basics

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

Oracle Communications Core Session Manager software allows for deployment as one of two available components:

- Core Session Manager (CSM)—See the chapter titled Oracle Communications Core Session Manager Supporting the IMS Core for detailed information about Oracle Communications Core Session Manager 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 Communications Core Session Managers. 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.

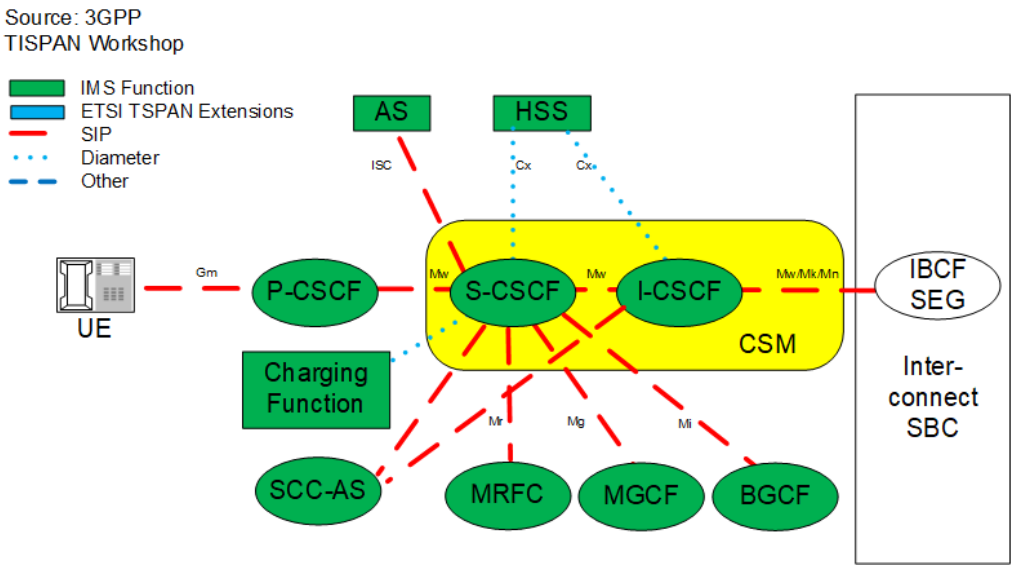
Oracle Communications Core Session Manager and IMS

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 Communications Core Session Manager 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 Communications Core Session Manager are best understood as functions of standard IMS elements. The diagram below depicts the mapping of these functions across an IMS architecture.



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.

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

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

As S-CSCF, the Oracle Communications Core Session Manager complies with 3GPP standards and Oracle Communications Core Session Manager 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 Communications Core Session Manager software that allows the network architect to establish a front end to multiple Oracle Communications Core Session Managers 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 Communications Core Session Managers 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 Communications Core Session Managers configured as S-CSCFs, thereby preventing any given S-CSCF from becoming overburdened. The user can configure any Oracle Communications Core Session Manager 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 Communications Core Session Manager and SLRM Configuration

Oracle Communications Core Session Manager (OCCSM) 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.

Oracle documents this product using an Essentials model, which results in a unique *OCCSM Essentials Guide* document, and refers to the OCSBC Documentation Set for additional, related components, features and procedures. The documentation set, listed in the front matter of this document, provides configuration information across all session control products. The OCCSM filters out configuration elements, sub-elements and parameters that do not apply to themselves, preventing you from performing invalid configuration procedures.

See the Base Configuration Elements Appendix for minimal configuration setting examples that establish an operable OCCSM or OCSLRM.

CSM Configuration Elements

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

- Boot Parameters
- Device Passwords
- Management Interfaces
- Default Gateway
- Product licensing

Required network and SIP service configuration elements, explained in multiple chapters in the *ACLI Configuration Guide*, 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 Communications Core Session Manager Supporting the IMS Core Chapter in this document, include:
  - Subscriber Database
  - SIP Registrar
  - ENUM for e.164 Translation

  - Registration Event

Common Oracle Communications Core Session Manager 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 Communications Core Session Manager 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

See the Appendix on Base Configuration Elements for a list of configuration setting examples that bring your system to a minimally operational state in an IMS environment. Change addressing and other infrastructure-dependent setting examples to match that of your environment.

High Availability

Oracle Communications Core Session Managers 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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.

3

Oracle CSM Supporting the IMS Core

General Description

The Oracle Communications Core Session Manager 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 Communications Core Session Manager can perform the SIP registrar role in conjunction with an HSS.

Message Authentication for SIP Requests

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager authenticates REGISTER request messages from all endpoints, while INVITES are only authenticated from anonymous endpoints.

User Authorization

In an IMS network, the Oracle Communications Core Session Manager requests user authorization from an HSS when receiving a REGISTER message. An HSS is defined on the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager.

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 the with Oracle Communications Core Session Manager. If the endpoint is registering, the Oracle Communications Core Session Manager 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 Communications Core Session Manager besides REGISTER, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager always sends a value of "MD5"

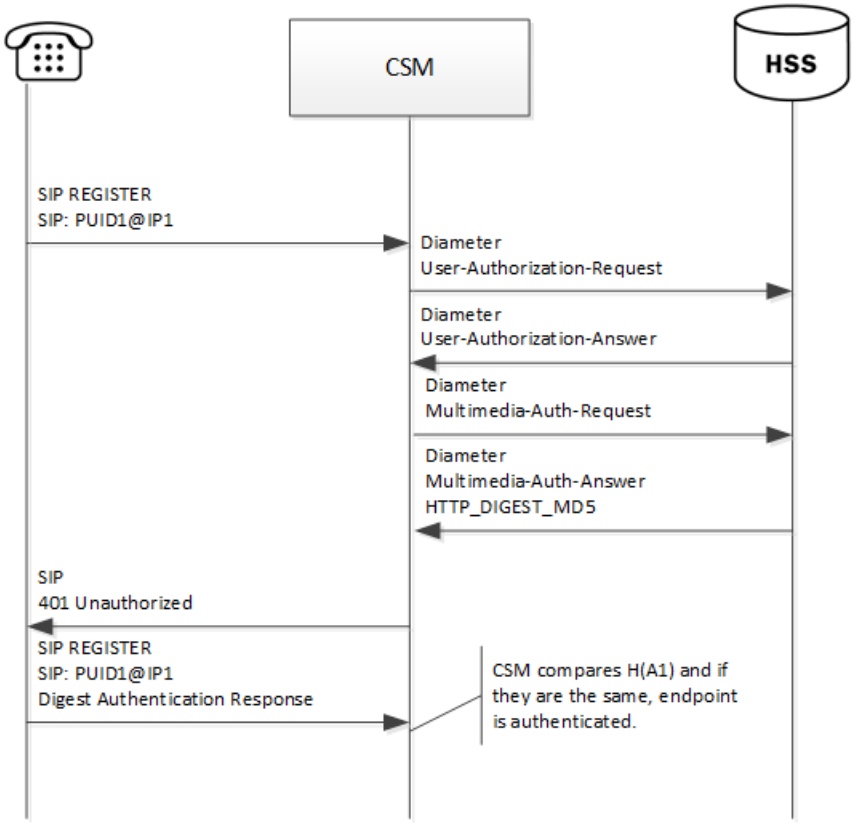
SIP Authentication Response

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

Oracle Communications Core Session Manager Authentication Check

At this point, the Oracle Communications Core Session Manager has received an MD5 hash from the HSS and an MD5 hash from the UA. The Oracle Communications Core Session Manager 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:



IMS-AKA Support

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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.

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 ACLI 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 may 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.

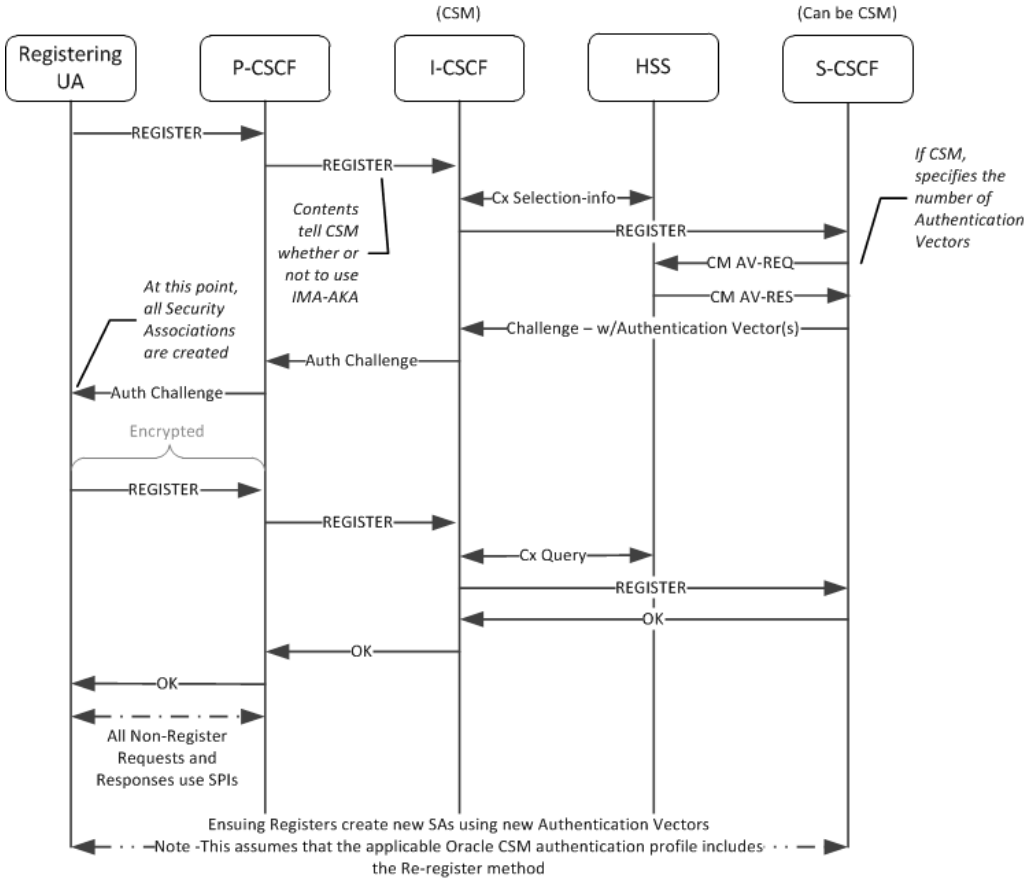
The first device to receive the REGISTER at the core is a P-CSCF. For the P-CSCF, appropriate configuration determines that it uses IMS-AKA as the authentication mechanism on the access interface. For an Oracle CSM, the presence and state of the “integrity-protected” parameter in the Authorization header 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 CSM 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 Communications Core Session Manager), 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 Communications Core Session Manager 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 it's 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 Communications Core Session Manager 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 Communications Core Session Manager can send the authentication challenge to the UA. If multiple authentication vectors were provided by the HSS, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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.

S-CSCF Selection Based on Capabilities

Within IMS environments, the I-CSCF identifies target S-CSCF's in response to SIP traffic for which the assigned S-CSCF is not known. Enhanced selection environments can include the HSS offering mandatory and optional capabilities for a user, and the I-CSCF selecting the best S-CSCF based on capabilities the S-CSCF is best suited to support (in addition to standard criteria). The user can configure the I-CSCF resident within Oracle CSM and Oracle SLRM to support this capabilities-based S-CSCF selection. Resultant operation is compliant with ETSI TS 129 228 and ETSI TS 129 229.

S-CSCF selection based on capabilities utilizes AVP information exchanged with the HSS to identify required and preferred capabilities on a per-user basis. Capabilities themselves vary widely. Examples include administrator routing preferences for divergent service types. Capabilities are manually defined at the HSS for endpoints or groups of endpoints. The Oracle CSM and Oracle SLRM user configures tables on the I-CSCF that map the S-CSCF's with the capabilities they support. Further configuration enables the I-CSCF to make the best S-CSCF selection, then forward appropriately.

Diameter messaging that can generate capabilities parsing for S-CSCF selection includes UAR/UAA and LIR/LIA traffic. Inclusion of the capabilities AVPs in the message sequence triggers this enhanced S-CSCF selection by the I-CSCF.

Configuration on the HSS and the I-CSCF must be compatible in deployments that use this feature. Configuration required on the Oracle device performing the I-CSCF function includes:

- servers-capabilities-list—A sip-registrar parameter that allows you to configure the registrar with a servers-capabilities-table.
- servers-capabilities-table—A multi-instance element that names the table and includes multiple servers-capability.
  - servers-capability—A multi-instance element within the servers-capabilities-table that includes a capability (capability value associated with users and supported by servers in the list) and a server-name-list that identifies the servers that support this capability.

The OCCSM verifies the **servers-capabilities-list** attribute with the **servers-capabilities-table** each time it loads the configuration. If the **servers-capabilities-table** with the name specified in the **servers-capabilities-list** does not exist , the system outputs the following message:

ERROR: sip-registrar [<object-name>] has invalid servers-capabilities-list entry [<entry-name>]

Server-Capabilities AVP

The Server-Capabilities AVP is a group AVP including the Mandatory-Capability AVP and Optional-Capability AVP. The number of Mandatory-Capability and Optional-Capability AVPs is not limited in a Server-Capabilities AVP. The AVP symbol notation, format and reference follows:

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.

Format definitions include:

- Server-Capabilities ::= <AVP header: 603 10415>
- *{Mandatory-Capability}
- *[Optional-Capability]
- *[Server-Name] (not supported in this release)
- *[AVP] (not supported in this release)

AVP reference, including column definition and AVP table follows:

- 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

AVP	Number	Reference	Type	Grouped
{ Server-Capabilities }	603	Base	Grouped	Mandatory-Capability Optional-Capability
{ Mandatory-Capability }	604	Base	Unsigned32	
[Optional-Capability]	605	Base	Unsigned32	

Selection Process without SLRM

The capabilities-oriented S-CSCF selection algorithm on the Oracle CSM S-CSCF include selections based on mandatory and optional capabilities information received from HSS and the configured S-CSCF Capabilities Database.

The general approach to selection within this scenario include the following principles:

- Only S-CSCFs with all mandatory capabilities can be selected.
- The process gives priority to the S-CSCF with the most optional capabilities.
- The process gives priority to the local S-CSCF.
- The system attempts to spread assignments to remote S-CSCFs of the same priority.

The capabilities-oriented S-CSCF selection algorithm uses the following high-level steps within the I-CSCF function to arrive at a selection:

1. Determine that the capabilities algorithm is required:
    a. No server-name in the LIA or UAA.
    b. Capability list exists.
    c. Assigned S-CSCF flag is not set.

   d. Mandatory/Optional Capabilities received in UAA/LIA.

2. Identify potential S-CSCFs, which must support all mandatory capabilities:
   a. Ensure the S-CSCF capabilities database is configured.
   b. Build capable S-CSCF list. This list contains all S-CSCFs from the S-CSCF capabilities database that support the Mandatory capabilities.
   c. Ensure that the capable S-CSCF list is not empty. If the capable S-CSCF list is empty, return an error to the UE.
3. Ensure that the I-CSCF is not SLRM.
4. Complete capabilities selection process using optional capabilities as criteria:
   a. An S-CSCF has the most optional capabilities.
   (If so, forward.)
   b. The local S-CSCF can take on more users, has all mandatory capabilities, and has most optional capabilities.
   (If so, forward locally.)
   c. Use round robin to select the S-CSCF that has most optional capabilities.
   (If so, forward.)
5. Forward message:
   a. Forward to selected S-CSCF.
   b. Remove selected S-CSCF from capabilities list.
   c. If there is an error, for example, the SIP response requires a re-assignment, check the assigned flag.
   d. If the assigned flag is set, return to the top.
   If the assigned is not set, return to the step that checks whether the capable S-CSCF list is empty.
   e. If the capable S-CSCF list is empty, return an error to the UE.
   If the capable S-CSCF list is not empty yet, perform capabilities selection process using optional capabilities as criteria again.

Selection Process with an SLRM

The capabilities-oriented S-CSCF selection algorithm on the Oracle SLRM uses standard Oracle CSM selection criteria in addition to capabilities criteria. This criteria includes cluster configuration, S-CSCF resource utilization and SLRM synchronization.

The general approach to selection within this scenario include the following principles:

- Only Oracle CSMs with all mandatory capabilities can be selected.
- The process gives priority to the Oracle CSMs in the cluster with the most optional capabilities, and is best able to take on new users.

The capabilities-oriented S-CSCF selection algorithm uses the following high-level steps, including the SLRM's selection steps, within the I-CSCF function to arrive at a selection:

1. Determine that the capabilities algorithm is required:
   a. No server-name in the LIA or UAA.
   b. Capability list exists.
   c. Assigned S-CSCF flag is not set.

   d. Mandatory/Optional Capabilities received in UAA/LIA.

2. Execute capabilities selection:
   a. Ensure the S-CSCF capabilities database is configured.
   b. Build capable S-CSCF list. This list contains all S-CSCFs from the S-CSCF capability database that support the Mandatory capabilities.
   c. Ensure that the capable S-CSCF list is not empty. If the capable S-CSCF list is empty, return an error to the UE.
3. Execute SLRM's selection procedure, cycle through all Oracle CSMs in the cluster:
   a. Identify applicable cluster. Begin to cycle through cluster.
   b. Determine whether Oracle CSM is in capable list.
   c. Determine whether Oracle CSM is at 100% utilization.
   d. Determine whether the next Oracle CSM support more optional capabilities.
   e. Determine whether the selected Oracle CSM is synchronized.
   f. Determine whether the next Oracle CSM using fewer resources.
4. Complete capabilities selection process using optional capabilities as criteria:
   a. An S-CSCF has the most optional capabilities.
   (If so, forward message.)
   b. The local S-CSCF can take on more users and has all mandatory capabilities and most optional capabilities.
   (If so, forward message locally.)
   c. Use round robin to select the S-CSCF that has most optional capabilities.
   (If so, forward message.)
5. Forward message:
   a. Forward to selected S-CSCF.
   b. Remove selected S-CSCF from capabilities list.
   c. If there is an error, for example, the SIP response requires a re-assignment, check the assigned flag.
   d. If the assigned flag is set, return to the top.
   If the assigned is not set, return to the step that checks whether the capable S-CSCF list is empty.
   e. If the capable S-CSCF list is empty, return an error to the UE.
   If the capable S-CSCF list is not empty yet, perform SLRM's selection procedure again.

ACLI Instructions

Configuring the server-capabilities-table

A **server-capabilities-table** is a multi-instance element that allows the user to name a **servers-capability** object and apply it to a **registrar**. A **servers-capability** object is a **server-capabilities-table** sub-element that includes a **capability** and multiple server names, which support that capability.

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 **server-capabilities-table** and press Enter to access the path.

```
ORACLE(session-router)# server-capabilities-table
ORACLE(server-capabilities-table)#
```

4. Enter a contiguous string to the **name** field. This name is the reference used in the registrar configuration to specify the use of this server capabilities table.
5. Type **servers-capability** and press Enter to access the path.

```
ORACLE(server-capabilities-table)# servers-capability
ORACLE(servers-capability)#
```

6. Enter a number to specify the capability **capability**. Valid entries range from 0 to 999999999.
7. Enter the names of the servers that belong to this **server-name-list**. Name format is the same as that used within the registrar's **home-server-route** field. The format is the URI (containing FQDN or IP address) used to identify a server to the HSS. Each entry in the list is enclosed with quotes and separated by a space.
8. Type **done** and **exit** twice to complete configuration of this **server-capabilities-table** configuration element.

Configuring the server-capabilities-list

To assign a server capabilities list to 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 to access the session router path.

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

4. Type **server-capabilities-list** and press Enter. Add a capability with associated servers.

```
ORACLE(sip-registrar)# server-capabilities-list my_capability_list1
ORACLE(sip-registrar)#
```

5. Type **done** and **exit** to complete configuration of this **sip-registrar** configuration element.

OCCSM as Registrar

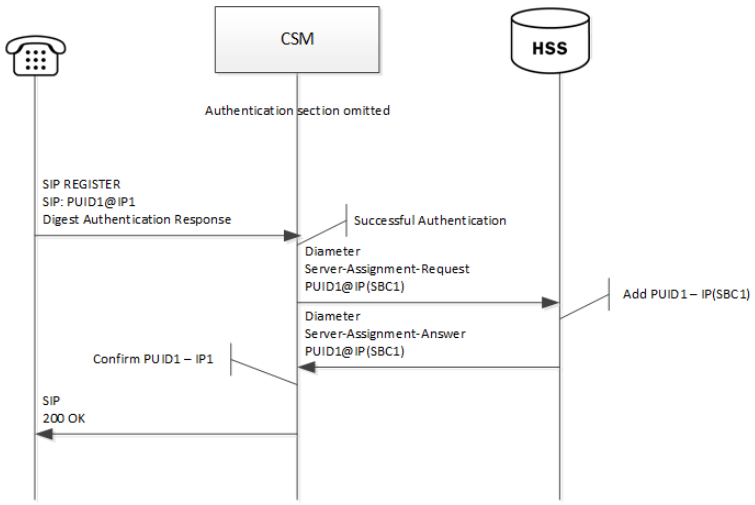
Creating a sip registrar configuration element enables the Oracle Communications Core Session Manager (OCCSM) to act as a SIP registrar. When registration functionality is enabled, the OCCSM actually registers endpoints rather than only caching and forwarding registrations to another device. OCCSM registry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

On receiving a REGISTER message, the OCCSM 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 OCCSM 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 OCCSM 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 Response with the Authentication-info Header

The Oracle Communications Core Session Manager can include the authentication-info header, as described in RFC 2617, in its 200 OK response to REGISTERs when using SIP digest. The user enables this functionality using a **sip-registrar** option.

By default, the Oracle Communications Core Session Manager supports registration with SIP digest authentication without using the authentication-info header. This is not compliant with TS 24.229. Enabling the **add-auth-info** option causes the Oracle Communications Core Session Manager to calculate and insert the required authentication-info header fields in the 200 OK.

The Oracle Communications Core Session Manager also presents this authentication header during third party registrations. The system includes the entire 200OK message in the third party registration request.

This authentication state is not shared across high availability nodes. The user can expect the Oracle Communications Core Session Manager to request re-authentication by registering UEs after failover to a backup Oracle Communications Core Session Manager.

Authentication-Info header field parameters sent by the Oracle Communications Core Session Manager include:

- **qop**—Matches the **qop** sent by the UE
- **rspauth**—A response-digest calculated as described in RFC 2617
- **cnonce**—Matches the **cnonce** sent by the UE
- **nonce-count**—Matches the **nonce-count** sent by the UE

The **nextnonce** authentication-info header field parameter, which can request a new nonce for subsequent authentication responses from the UE, is not implemented on the Oracle Communications Core Session Manager.

The ACLI syntax for enabling the **add-auth-info** option follows.

```
ORACLE(sip-registrar)#+options=add-auth-info enabled
```

The Oracle Communications Core Session Manager provides NOTICE level log entries in **log.sipd** to indicate this option's status.

Registration Handling for Online and Offline Operation Modes

The Oracle Communications Core Session Manager provides a means of running in offline mode. Offline mode provides the Oracle Communications Core Session Manager 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 Communications Core Session Manager into offline mode because the resource may not be currently required or for system maintenance.

The Oracle Communications Core Session Manager 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 Communications Core Session Manager completely out of service after it has released registrations for all its UEs.

The user can explicitly set their Oracle Communications Core Session Manager 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 Communications Core Session Manager begins taking itself out of service as an S-CSCF. This process may last a long time because the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 doe 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 Communications Core Session Manager first identifies UEs in the registration cache that have registration event subscriptions and marks them for deletion (dirty). Recall that the Oracle Communications Core Session Manager does not release any UE currently engaged in a session. For UEs in the cache without registration event subscriptions, the Oracle Communications Core Session Manager waits for registration refreshes. When they refresh, the Oracle Communications Core Session Manager marks them for release and begins to release them.

The release procedure consists of the Oracle Communications Core Session Manager 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 Communications Core Session Manager does this to remove itself as the assigned S-CSCF. This allows other Oracle Communications Core Session Managers become the S-CSCF for this UE.
3. Send deregister requests to application servers to which the Oracle Communications Core Session Manager 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 ome or more SLRMs within your deployment, the Oracle Communications Core Session Manager notifies them when it goes offline. This notification mechanism is the same as that used by the Oracle Communications Core Session Manager to indicate that it is low on resources. The Oracle Communications Core Session Manager advertises to the SLRM that it has no resources to handle registrations. When going back online, the Oracle Communications Core Session Manager simply re-advertises its resources, resuming normal operation with the SLRM.

Interaction with UEs

Actions the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager has completed originating services for an unregistered UE that is in the cache.

While in offline mode, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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`

Handling Barred PUIDs

The Oracle Communications Core Session Manager supports PUID barring functionality per 3GPP specification TS 24.229. As such, the system does not service any request method other than REGISTERs for SIP or Tel-URI PUIDs designated as barred by the HSS. The Oracle Communications Core Session Manager also complies with the requirement that it allow Push Profile Requests (PPRs) to change a PUID from barred to non-barred (and vice versa) and issues a NOTIFY of the event to subscribers. No configuration is required.

A common use case for barring information is a cell phone registering with a temporary PUID (that is barred), along with a set of non-barred PUIDs in the P-Associated User (PAU) header. After registration, the cell phone should use only the non-barred PUIDs for all ensuing methods and its contacts.

An HSS should be configured with barring information for all PUIDs. During registration procedures, the HSS provides this information to the S-CSCF. PUID information in the User Data AVP of the Diameter SAA includes a tag indicating whether the PUID is barred. The Oracle Communications Core Session Manager retains this information in the registration cache. To complete the registration, the Oracle Communications Core Session Manager replies to the UE with a list of all non-barred PUIDs in the 200OK. For all the further procedures, the UE should use a PUID from the non-barred P-Associated-URI list. If the HSS does not identify a PUID's barring status, the Oracle Communications Core Session Manager assumes it is not barred.

Typical Oracle Communications Core Session Manager behaviors related to barring include:

- Responds to ensuing requests from barred PUIDs with (403) Forbidden.
- Responds to requests that have no PSU, but include barred PUIDs in their PAI header list with 403 (Forbidden).
- Responds to requests to or from wildcarded PUIDs that match barred PUIDs with 403 (Forbidden).

- Responds to registration attempts that have all barred implicit identities with 403 (Forbidden).
- Responds to requests for termination services wherein the served user (PSU/RURI) is barred with (404) Not Found.
- Recognizes barring status during third party registration procedures and does not attempt to register a barred PUID to an AS.
- Handles related subscription scenarios as follows:
  - When receiving a subscription from a barred subscriber, responds with 403 (Forbidden).
  - When receiving a subscription for a barred user, allows the SUBSCRIBE to proceed.
  - Does not include a barred identity in any NOTIFY.
    * When receiving a subscription for a user that has barred identities in its implicit set, issues NOTIFYs that only include non-barred identities.
    * Includes only non-barred PUIDs in NOTIFY messages generated by network-initiated re-registration and authorization requests.

> **Note:**
>
> The Oracle Communications Core Session Manager does not support any PUID barring within the context of GRUU.

The user can verify PUID barring status using the **show reg sipd by-user <user> detailed** command. Example output is shown below.

```
ORACLE# show reg sipd by-user user detailed
Registration Cache (Detailed View)    Thu Jul 09 2015  15:16:08
User: sip:user_1@acme-ims.com
  Registered at:  2015-07-09-15:16:04    Surrogate User: false
  Emergency Registration? No
  ContactsPerAor Rejects 0
  ContactsPerAor OverWrites 0

  Contact Information:
    Contact:
      Name: sip:user_1@acme-ims.com
      Valid: true

...

    Associated URI(s):
      URI: sip:user_1@acme-ims.com
      Status: Barred

...
```

> **Note:**
>
> The Oracle Communications Core Session Manager replicates barred status for PUIDs to standby systems.

Releasing Unregistered Users

When a call arrives at an Oracle Communications Core Session Manager either to or from a user that is not registered at that Oracle Communications Core Session Manager, 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 Communications Core Session Manager 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 Communications Core Session Manager. Upon expiry of this feature's timer, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager rejects the register, replying with a 504 error message.
- The UE has existing calls—The Oracle Communications Core Session Manager continues to support the call, based on a stored copy of the service profile.
- A new call arrives—The Oracle Communications Core Session Manager rejects the call. The Oracle Communications Core Session Manager replies with a '480, Temporarily Unavailable' error message if the UE is the callee; the Oracle Communications Core Session Manager 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 Communications Core Session Manager does not cache unregistered users. A setting of 0 means the Oracle Communications Core Session Manager 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 Communications Core Session Manager. PSIs appear as either Distinct PSIs or Wildcarded PSIs. Similar to unregistered users, the Oracle Communications Core Session Manager 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 Communications Core Session Manager to release PSIs. This syntax is shown below.

```
ORACLE(sip-registrar)# options psi-cache-expiry=120
```

Configurable Response to Timed-Out OPTIONS Messages

The Oracle Communications Core Session Manager allows the user to configure a function by which they can cause the system to send a 408 as a response to an OPTIONS message sent to an un-responsive, registered called party. In addition, this function allows the user to specify when to send that 408.

By default, the Oracle Communications Core Session Manager does not send messages to an originating node when OPTIONS transactions time out. This complies with RFC 4321.

When registered users do not respond to OPTIONS requests, the network never informs the calling party of the called party's status. Instead, the calling party waits for the standard 32-second retry timeout to expire. If the called party was previously reachable, the calling party treats it as reachable for the entire 32-second window.

The Oracle Communications Core Session Manager includes a configuration option that:

- Starts a timer when the system forwards an applicable OPTIONS message and,
- Upon expiry of that timer, causes the system to send a 408 message to the calling party.

This option allows the network administrator to provide the calling party with this 408 response, and specify a shorter interval between request and response.

This feature works for:

- A called party that is registered via its P-CSCF, but not currently reachable.
- A called party that is reachable via an IBCF.

This function has no impact on requests that result in a response, such as SIP 480, for un-registered subscribers.

For registered users with multiple contacts, the Oracle Communications Core Session Manager uses a response from any contact as a trigger to stop the timer and not send a 408. The Oracle Communications Core Session Manager cancels all remaining OPTIONS transactions when it receives a response from a contact. In addition, if the system used parallel forking to reach multiple contacts, it waits for the timer expiry before it sends the 200OK to the caller.

The option is available via S-CSCF processing. Note that, if the called party finally responds after this timer expires and the S-CSCF logic has sent the 408, the Oracle CSM forwards it to the originating node.

The user sets the option globally in **sip-config** or on a **sip-interface**, with the **sip-interface** taking precedence. Values range from 1 to 32 seconds. Invalid ranges cause the system to use the maximum value of 32. The example below sets a sip-interface's timer to 4 seconds.

```
ORACLE(session-router)#sip-interface
ORACLE(sip-interface)#options +options-408-timeout=4
```

Option syntax on the **sip-config** and **sip-interface** configuration elements is the same.

The user must consider the infrastructure carefully. Setting the value too low can cause an inordinate number of invalid 408 responses.

Limiting REGISTER CDR Generation

The Oracle Communications Core Session Manager allows the user to generate RADIUS CDRs for REGISTER events via configuration. Large networks, however, can generate an inordinate volume of CDRs. So the Oracle Communications Core Session Manager also allows the user to reduce REGISTER CDR generation by filtering out some of the messages it sends.

When the user enables accounting with the generate-events parameter, the Oracle Communications Core Session Manager can generate CDRs for the following register and/or local register events:

- Initial REGISTER
- REGISTER refresh
- REGISTER update
- de-REGISTER

Depending on the event, the system generates per-contact start, interim and/or stop CDRs. With no other configuration, the system generates the appropriate CDRs for all of these events.

The user can prevent the system from issuing some CDR via an **account-config** option that filters, as described below, and sets a timer that restarts the CDR suppression window. Use the syntax below to set this **register-cdr-interval** option with an expiry timer value of 43200 in minutes (30 days), and limit the number of generated CDRs as described below.

```
(account-config)#options +register-cdr-interval=43200
```

When configured with this option, the Oracle Communications Core Session Manager limits the generation of CDRs for each user as follows:

1. Send a START CDR for first Register message (for first contact).
2. Don't send CDRs until the user specified time period expires. After it expires, when a Registration message causes a 'START' or 'INTERIM' CDR event to occur, send it. Then, re-set the time value. Applicable 'START' CDR events include:
   - Add new contact
   - Replace contact
   - Overwrite contact

The applicable 'INTERIM' CDR event is a Refresh Contact.

The **generate-event** parameter must also be set to **register**.

Limiting AOR Contacts

The Oracle Communications Core Session Manager allows you to limit the number of contacts that apply to AORs. It also provides a configurable behavior allowing the system to either reject a new contact or overwrite an existing contact with the new one.

The user specifies the maximum number of contacts and the operation mode on a per-registrar basis. Alternatively, the user can disable the feature. This feature is applicable to Cx and local database deployments.

The value for **max-contacts-per-aor** ranges from 0-256. A value of 0 disables the function. When **max-contacts-per-aor** is greater than zero, the Oracle Communications Core Session Manager tracks the number of contacts registered per AOR. Settings for **max-contacts-per-aor-mode** include REJECT and OVERWRITE.

If you change the configured maximum while the system is operational, your setting only applies to new registrations. If there are more contacts than your newly configured maximum, the system removes older contacts. This ensures that the contacts are always within the configured maximum.

Both **max-contacts-per-aor** and **max-contacts-per-aor-mode** are RTC supported.

Maximum Contacts REJECT Mode

If the Oracle Communications Core Session Manager 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.

Maximum Contacts OVERWRITE Mode

If the number of contacts in the initial registration exceeds the maximum, the Oracle Communications Core Session Manager selects only the highest priority contact based on q-values. If there are no q values, the Oracle Communications Core Session Manager adds contacts in the order they appear in the REGISTER message until it reaches the maximum. The system then identifies the oldest contacts for overwriting using the last registered time stamp.

In all cases, the Oracle Communications Core Session Manager follows this procedure to remove old contacts:

1. If reg-id/instance-id is present in the contact, the system simply updates the contact.
2. The system sends NOTIFY messages to the subscriber for whom the contact has been removed with a status of "terminated" and "de-activated" as the reason.
3. The system removes the contact from the registration cache.

HSS Server Assignment

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

Server Assignment Messages

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager sends an MAR to the HSS to retrieve authentication data once again).

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

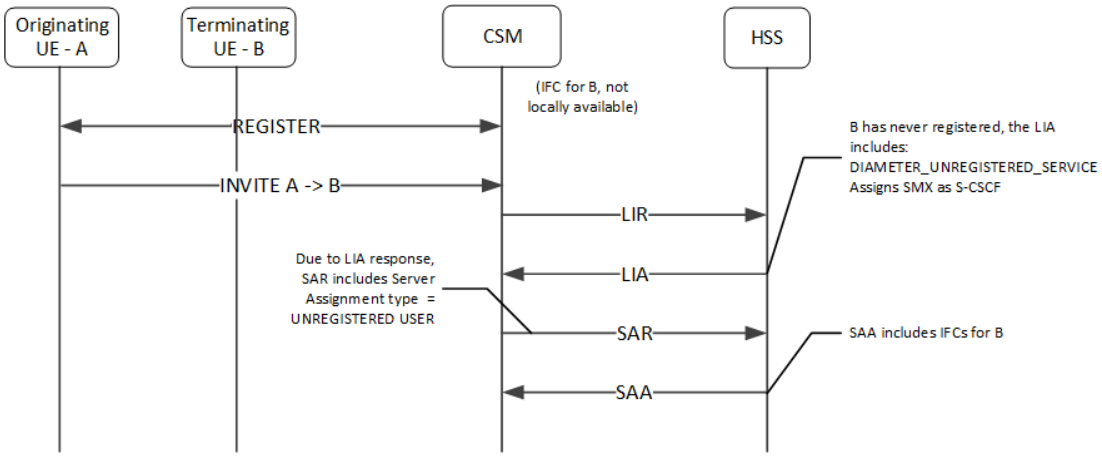
- The location update interval timer has expired—This value, configured in the sip registrar configuration element ensures that HSS database always has the correct Oracle Communications Core Session Manager address by periodically sending SARs for each registered contact.
- The message's call-id changes while the f**orward-reg-callid-change** option in the sip config configuration element is set. This covers the case where the UA changes the Oracle Communications Core Session Managers 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 Communications Core Session Manager1, moves to Oracle Communications Core Session Manager2 , and then returns to Oracle Communications Core Session Manager1.

If the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager'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 Communications Core Session Manager 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 that a completely new registration for the Contact. The Oracle Communications Core Session Manager can perform this way by considering the endpoint's reg-id and instance-id parameters defined in RFC 5626.

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager determines if it is the First hop. If there is only one Via: header in the REGISTER, the Oracle Communications Core Session Manager 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 CSM performs additional validation by checking that a Path: header corresponding to the last Via: includes a URI parameter, Outbound Registration Binding may continue.

If the Oracle Communications Core Session Manager 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 be 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 Communications Core Session Manager receives requests for the same AOR with some registrations with reg-id + instance-id and some without them, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager configuration required.

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

To allow the feature, the Oracle Communications Core Session Manager 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.

Retrieving the P-Charging Function Address from the HSS

You can configure the OCCSM to retrieve P-Charging Function Addresses (PCFA) information from a Home Subscriber Server (HSS). The OCCSM can get this address from the HSS over the Cx interface during subscriber registration. The OCCSM, subsequently, can use PCFA addresses to populate the outgoing P-Charging-Function-Address headers it sends to an application server in INVITEs and registration success messages it sends to a P-CSCF. As a result, your infrastructure can use PCFA information provisioned at the HSS as the charging addresses for your subscribers. OCCSM use of the charging information AVP is compliant with ETSI TS 129 229. The OCCSM generates the applicable P-Charging-Function-Address headers per RFC 7315.

When you enable this feature, the OCCSM, acting as S-CSCF, can decode the Charging-Information AVP, save the information locally, and insert P-Charging Function Address headers into specific message, assuming they are not already present. Applicable scenarios include registrations, registration refreshes, and dialog creating initial INVITE messages. In addition, if an applicable INVITE arrives with the PCFA header already populated, the OCCSM transparently forwards the INVITE with that PCFA information.

This feature does not apply to out of dialog messages, standalone requests or messages subsequent to an initial INVITE.

Applicable PCFA information includes IP addressing for the objects that perform the following functions:

- Charging Collection Function (CCF), which resides on the Charging Data Function (CDF) device
- Event Charging Function (ECF), which resides on the Online Charging Function (OCF) device

Depending on the scenario, the HSS provides the primary or both primary and secondary device addressing. The HSS presents the CCF for off-line charging functions, and the ECF for on-line charging functions.

You enable this functionality on the OCCSM using the **add-pcfa** parameter within the **sip-config** as shown below. The default is disabled.

```
ORACLE(sip-config)#add-pcfa enabled
```

When enabled, the OCCSM adds the following behaviors:

- Upon receiving a REGISTER request, and when the information is not already available locally, fetches PCFA information from the HSS using SAR/SAA exchange and stores PCFA information for each contact locally.
- Adds PCFA information, both primary and secondary, to the 200 OK when responding to a REGISTER message.
- If the CCF and/or ECF values are available for the originating user, the OCCSM inserts these additional CCF and/or ECF values in the PCFA header of an outgoing INVITE (Orig) towards the TAS.

> **Note:**
>
> The first instance of CCF and/or ECF header field parameters are the primary address(es).

- When forwarding an INVITE towards a registered terminating party, the OCCSM uses the value of the locally stored CCF and/or ECF to populate the PCFA headers in outgoing INVITE (Term) messages towards the TAS.
- If the CCF and/or ECF values are available for the terminating user, the OCCSM inserts these additional CCF and/or ECF values in the PCFA header of an outgoing INVITE (Term) towards the TAS.
- For un-registered terminating users for whom the OCCSM has does not have PCFA details available locally, the OCCSM performs a SAR-SAA exchange with the HSS to get this information. The OCCSM then:
  - Stores the values locally
  - Uses the values to populate the PCFA in outgoing INVITE (Term) messages towards the TAS.
  - Uses the values to populate the PCFA in outgoing INVITE messages towards core systems, including Border Gateway Control functions (BGCFs).
- For un-registered originating users for whom the OCCSM has does not have PCFA details available locally, the OCCSM performs a SAR-SAA exchange with the HSS to get this information. The OCCSM then:
  - Stores the values locally
  - Uses the values to populate the PCFA in outgoing INVITE (Orig) messages towards the TAS.
- If it receives an INVITE from a P-CSCF that has PCFA information, the OCCSM retains that information and populates it within the outgoing INVITE (Orig) towards the TAS.

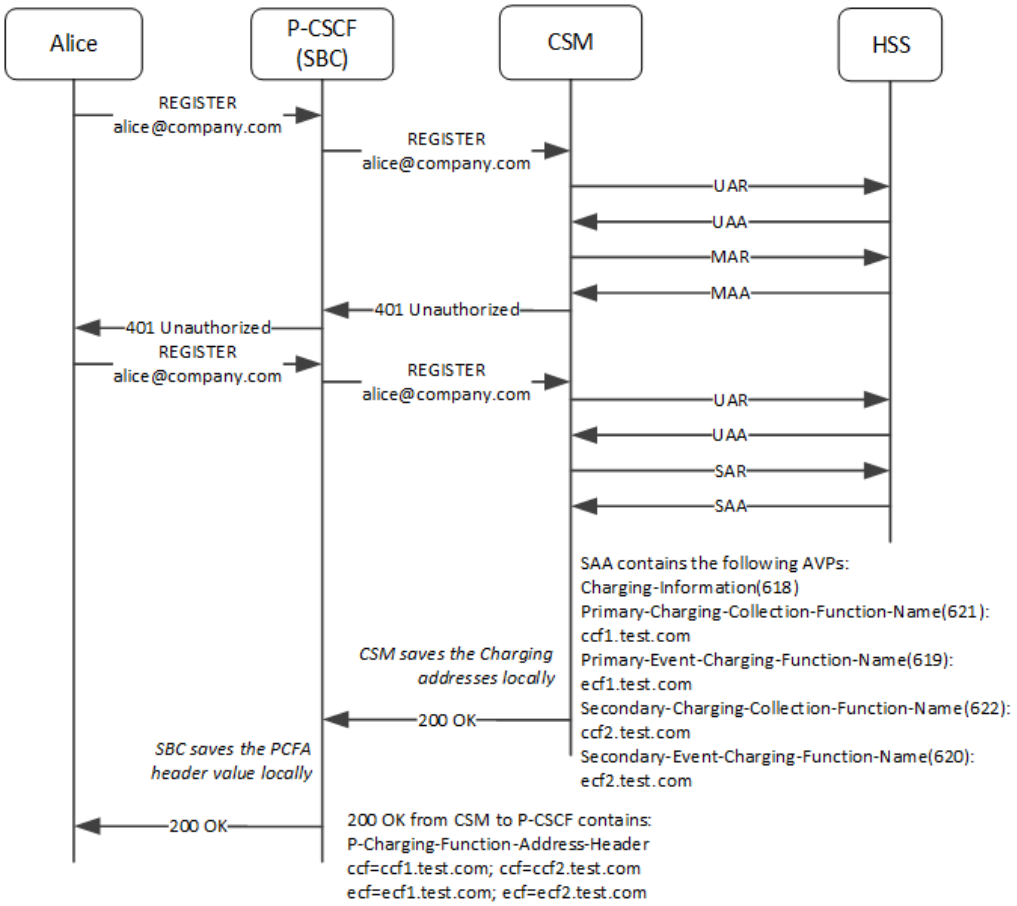
This feature supports real time configuration and HA deployments.

Call Flows for HSS-Based PCFA

This section presents key call flows that depict the OCCSM handling REGISTER and INVITE methods while configured for this HSS-based PFCA feature.

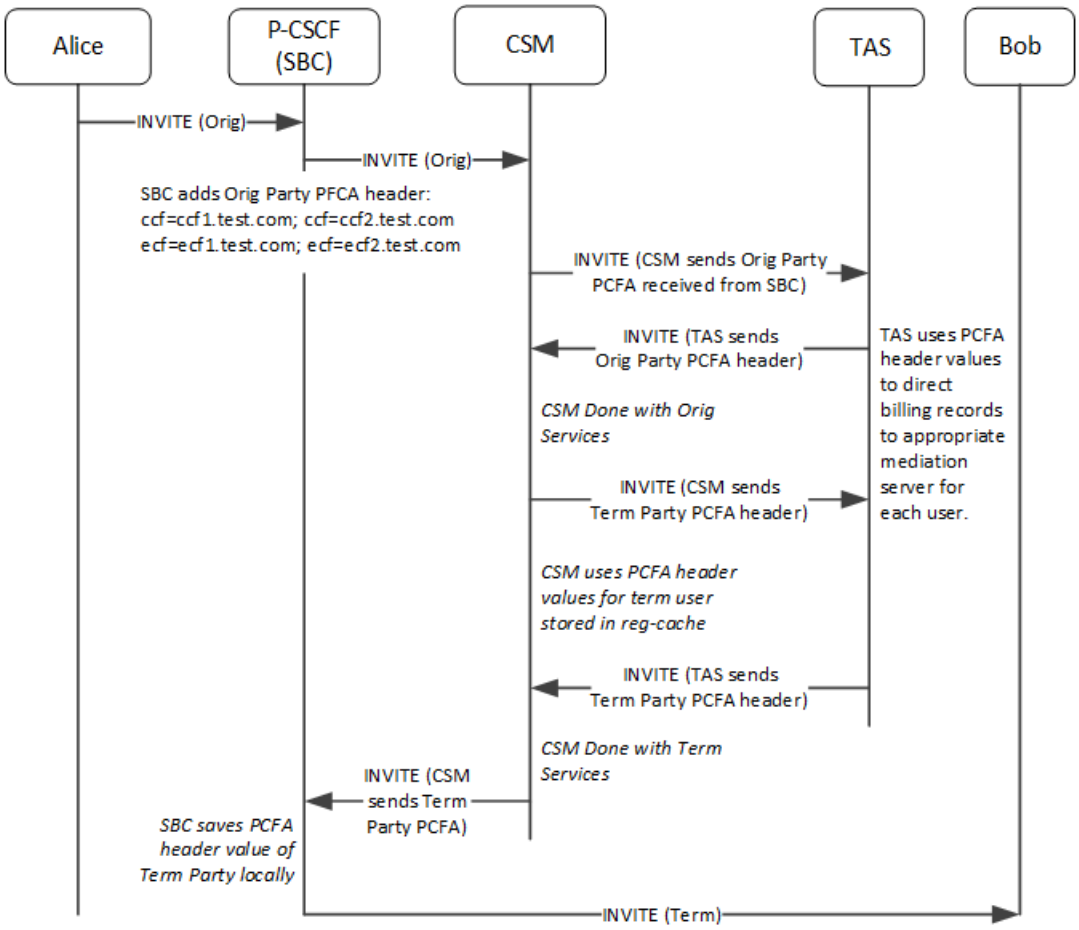
Simple Register

During registration, the OCCSM decodes the Charging-Information Grouped AVP and saves the primary and/or secondary ccf /ecf addresses it receives from the HSS an in SAA over the Cx interface. The OCCSM uses these addresses to populate the outgoing P-Charging-Function-Address header in the 200 OK response of the REGISTER message it sends to the P-CSCF (SBC).



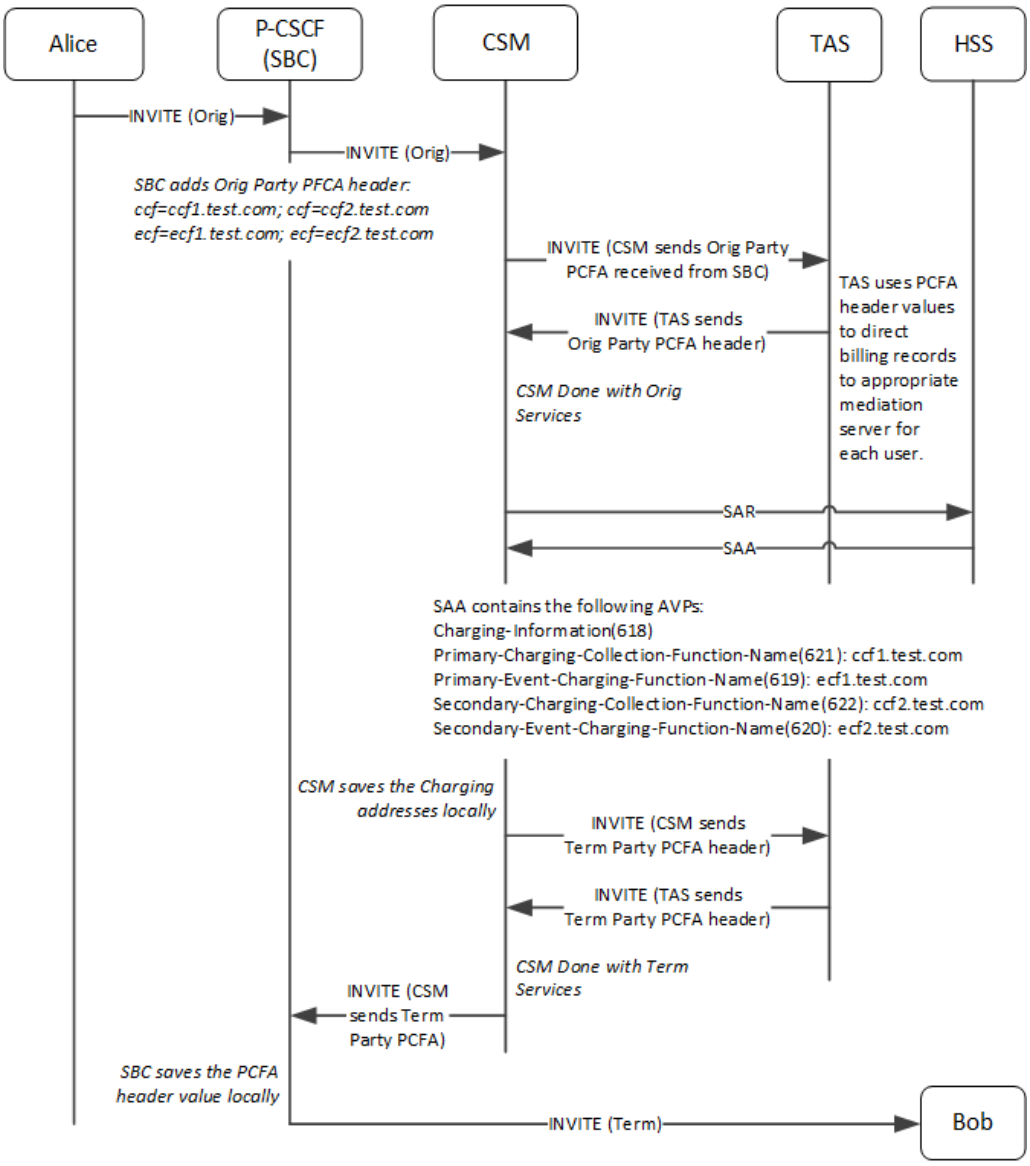
INVITE with Registered Orig and Term

In this example, the OCCSM handles an INVITE from a registered caller (Orig party) to a registered receiver (Term party). The P-CSCF, which could be an SBC, adds the PCFA header for the Orig Party with the charging addresses obtained during registration. Here, the OCCSM forwards the PCFA header it receives in the INVITE from the P-CSCF to the TAS server. After completing origination services, the OCCSM adds the Term Party's PCFA header, which includes the charging addresses saved during Term Party registration. The TAS server can then use this information for user billing purposes.



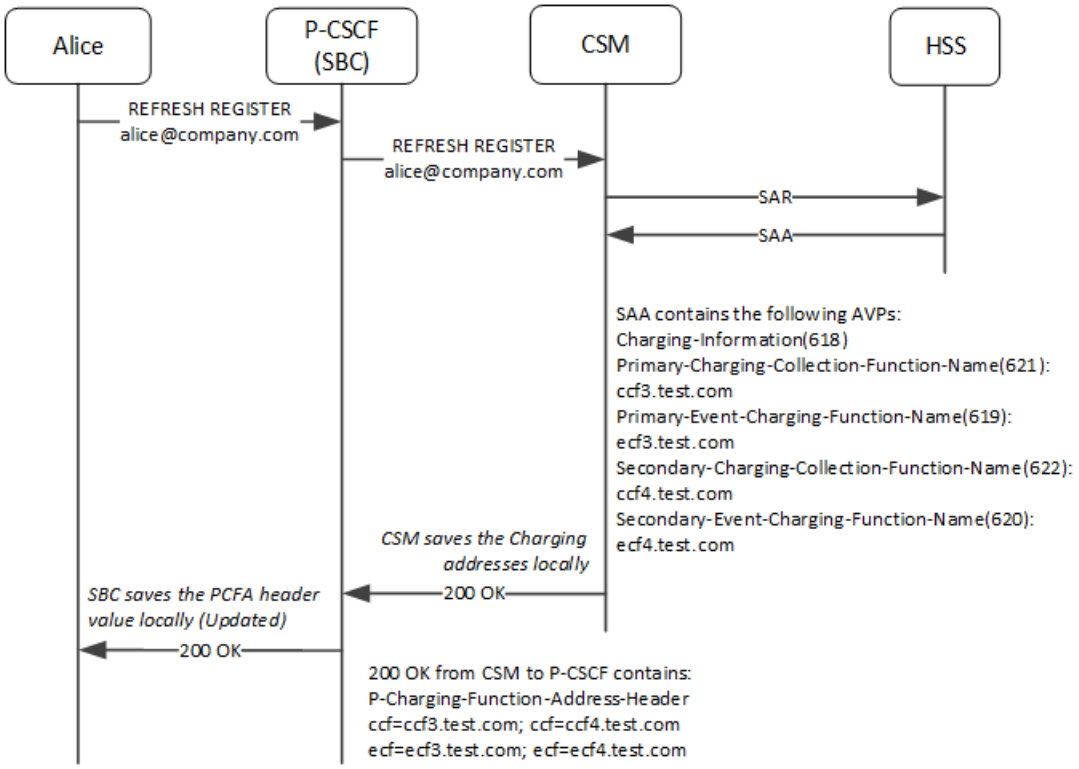
INVITE with Registered Orig and Unregistered Term

This next example expands upon the INVITE scenario by introducing an Unregistered Term Party. Again the P-CSCF adds the PCFA header for the Orig Party with the charging addresses. The OCCSM forwards the PCFA header received in the INVITE to the TAS server. After completing origination services, the OCCSM performs the SAR/SAA exchange, which provides the PCFA for the unregistered Term party and saves these locally. The OCCSM uses these charging address values to populate the PCFA header for the Term Party. The TAS server can then use this information for user billing purposes.



Refresh Register Scenario

If you upgrade your OCCSM from S-Cz8.4.5 to S-Cz9.1.5, or you enable the feature after the system has been operational, the OCCSM would receive the updated CCF/ECF addresses after a refresh register, wherein the SAR/SAA happens for the existing users.



Charging-Information AVP (618)

The charging-information AVP resides within the core registration request and answer sequence. It provides the OCCSM with the information needed to populate P-Charging Function Address headers in INVITE messages.

The Charging-Information AVP is a Grouped AVP. It contains the addresses of the primary and secondary charging functions.

AVP	Number	Reference	Type
Charging-Information ::= <AVP header>	618 10415	3GPP	Grouped
[Primary-Event-Charging-Function-Name]	619	3GPP	String
[Secondary-Event-Charging-Function-Name]	620	3GPP	String
[Primary-Charging-Collection-Function-Name]	621	3GPP	String
[Secondary-Charging-Collection-Function-Name]	622	3GPP	String

Configuring PCFA Retrieval

Perform this sequence to enable ODI preservation on the OCCSM on a global basis.

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-config** object to edit.

```
ORACLE(sip-config)# select


ORACLE(sip-config)#
```

3. **add-pcfa**—Set this parameter to **enabled**. to retrieve charging information from the HSS using the charging information AVP (618). The default value is disabled.

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

4. Save your work.

ACLI Instructions

The following configuration enables the Oracle Communications Core Session Manager to authorize and authenticate registering users. In addition it sets the Oracle Communications Core Session Manager 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. **sip-authentication-profile**—Set this to the name of an existing sip-authentication profile if you wish to authenticate per SIP interface.
7. Type **done** when finished.

SIP Registrar

To configure the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 and a mode of overwrite:

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.

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

3. Specify the contact mode to overwrite.

```
ORACLE(sip-registrar)# max-contacts-per-aor-mode overwrite
ORACLE
```

4. Type **done** and **exit** to complete configuration of this **sip-registrar** configuration element.

Response to Exceeding Maximum Contacts

To configure local response for the Oracle Communications Core Session Manager 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
```

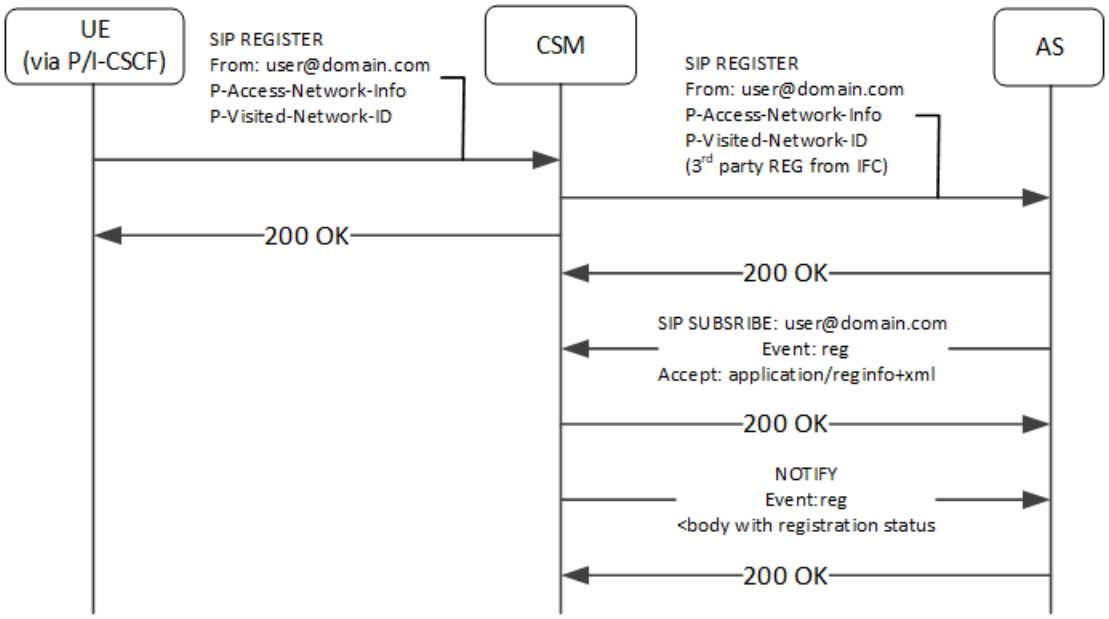
SIP Registration Event Package Support

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager is performing S-CSCF functions. You enable this feature on the Oracle Communications Core Session Manager per registrar, by simply creating a profile and applying it to the applicable registrar.

SUBSCRIBE Processing

When the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager proceeds with the request.

3. Authorizes the request. The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager only processes subscriptions for users in its registration cache, replying with a 403 Forbidden if not. For cached users, the Oracle Communications Core Session Manager 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 Communications Core Session Manager applies it's own configured minimum or the default (600000 seconds), whichever is greater.

   If the SUBSCRIBE includes an Expires header, the Oracle Communications Core Session Manager 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 Communications Core Session Manager denies the subscription, sending a 423 Too Brief message.

   When the Oracle Communications Core Session Manager 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 Communications Core Session Manager. Messages must be received from authorized subscribers and on the same realm and connection as the original SUBSCRIBE or the Oracle Communications Core Session Manager rejects the refresh request.

Reg Event NOTIFY Messages

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

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

The Oracle Communications Core Session Manager 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 Communications Core Session Manager includes:

- The Oracle Communications Core Session Manager always sends full information on all contacts, and indicates such within the reginfo element. The Oracle Communications Core Session Manager 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 Communications Core Session Manager does not include the following optional attributes within the contact element:
  - expires
  - retry-after
  - duration registered
  - display-name
- The Oracle Communications Core Session Manager 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 Communications Core Session Manager to limit notification traffic. By specifying the number of seconds between NOTIFY messages, you prevent the Oracle Communications Core Session Manager 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 Communications Core Session Manager sends out a NOTIFY for every registration event subscription. Note also that the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager's local policy routing functionality.

Within the context of local policy routing, the Oracle Communications Core Session Manager 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 theOracle Communications Core Session Manager to load balance and route around softswitch failures.

When a message arrives at the Oracle Communications Core Session Manager, it determines whether it is coming from a session agent. If so, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager is using the HSS as a resource within the context of its routing decisions.

When an INVITE arrives, the Oracle Communications Core Session Manager 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 Communications Core Session Manager proceeds as described below in the section LIR/LIA Transaction.

Note that you can configure the Oracle Communications Core Session Manager 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 for this PUID. The answer is in 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 Communications Core Session Manager, then it 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 Communications Core Session Manager, 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 Communications Core Session Managers, HSSs, and other network elements communicate and exchange SIP messaging. It is advisable to configure this parameter to ensure well defined reachability among Oracle Communications Core Session Managers.

Routing Based on UA Capabilities

In compliance with RFC 3841, the Oracle Communications Core Session Manager is able to make forwarding and forking decisions based on preferences indicated by the

UA. To do this, the Oracle Communications Core Session Manager 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 Communications Core Session Manager made routing preference decisions solely via the q value present in the contact header. In cases where the preferences were equal, the Oracle Communications Core Session Manager simply forwarded to those contacts simultaneously (parallel forking). In cases where the q value were not equal, the Oracle Communications Core Session Manager forwarded in sequence (sequential forking), forwarding to the highest q value first.

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager

Endpoints register their capabilities by listing them in the Contact headers of the REGISTER request. The Oracle Communications Core Session Manager stores these feature parameters in its registration cache along with the other contact information. In the case of ENUM databases, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager fully supports. UAs include these parameters in the accept-contact: header to further clarify capabilities requirements for the session. The Oracle Communications Core Session Manager 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 Communications Core Session Manager only forwards to contacts that support the features and have registered that support.

If only the require parameter is present, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager to perform a number normalization procedure and ensure it presents a full E.164 number for resolution. Upon successful resolution, the Oracle Communications Core Session Manager proceeds with ensuing signaling procedures.

To configure the Oracle Communications Core Session Manager 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 Communications Core Session Manager forwards applicable requests to a border gateway function via local policy.

Refer to the *Oracle Communications Session Border Controller ACLI Configuration Guide* , Session Routing and Load Balancing chapter for complete information on how to configure a **local-routing-config** and/or an **enum-config**.

Number Lookup Triggers

Use cases that are applicable to number lookups and the associated Oracle Communications Core Session Manager procedures include:

- Request from the access side:
  1. The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager requests e.164 resolution from the ENUM server(s).

Actions Based on Lookup Results

The Oracle Communications Core Session Manager forwards to the resultant SIP-URI under the following conditions:

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

PSTN Breakout Routing

The Oracle Communications Core Session Manager 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.

The Oracle Communications Core Session Manager does not validate any rn or cic information. Instead, it simply routes the request. Note that the Oracle Communications Core Session Manager uses cic information instead of rn if both are present in the request. RFC 4694 compliant circumstances under which the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager to successfully perform lookups within two domains.

HSS Initiated User Profile Changes

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager.
  - DIAMETER_ERROR_TOO_MUCH_DATA—The request failed. The Oracle Communications Core Session Manager informs to the HSS that it tried to push too much data into the Oracle Communications Core Session Manager.
  - 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 Communications Core Session Manager 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 Communications Core Session Manager, 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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.

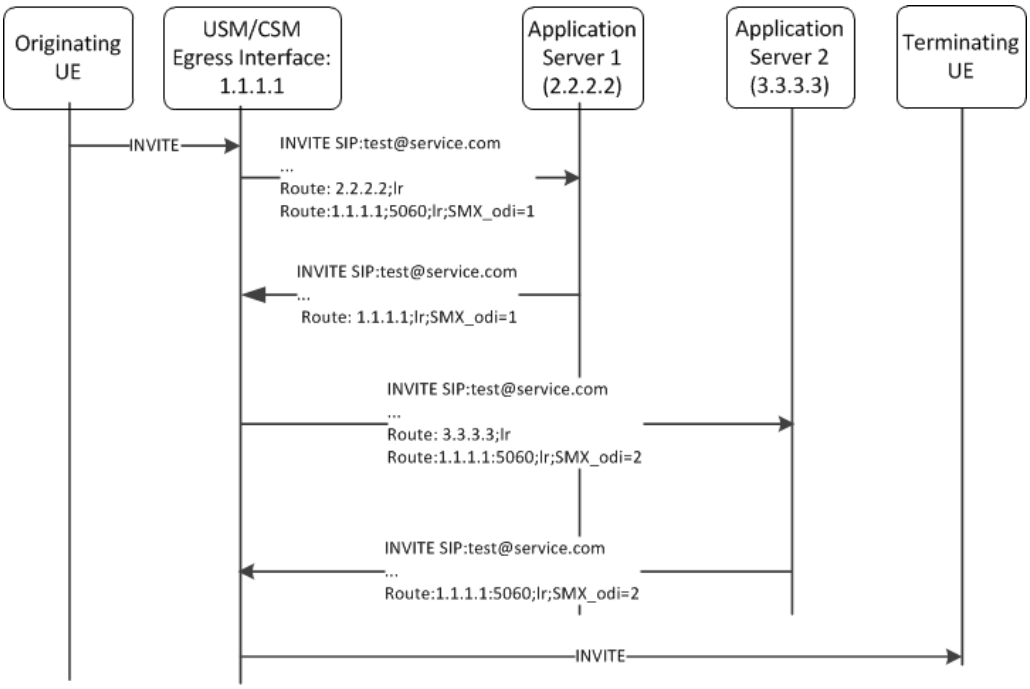
SIP Call

The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager, it is expected to include the ODI parameter that it received from the Oracle Communications Core Session Manager, 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 Communications Core Session Manager 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 Communications Core Session Manager checks if the target of the initial request is registered to itself, or not. If the UA is not registered locally the Oracle Communications Core Session Manager forwards the request by regular means into the network. If the target UA is registered locally, the Oracle Communications Core Session Manager proceeds to obtain iFCs for the target and begin iFC evaluation for the terminating side of the call.

Preserving an Original Dialog Indicator

As the Oracle Communications Core Session Manager (OCCSM) works through dialogs with Application Servers (AS), it saves and uses the Original Dialog Indicator (ODI) parameter to manage a call's service subscription sequence. By default, the OCCSM deletes the in-memory Service Profile, including the ODI, on receiving a final response to a transaction with an AS. If you enable the **preserve-odi** parameter in the **sip-config** however, the OCCSM maintains the in-memory service profiles, and each associated ODI, until it receives the **BYE** from the AS that ends the dialog between them. This is a global configuration, causing the OCCSM to maintain all ODIs for the duration of the dialog.

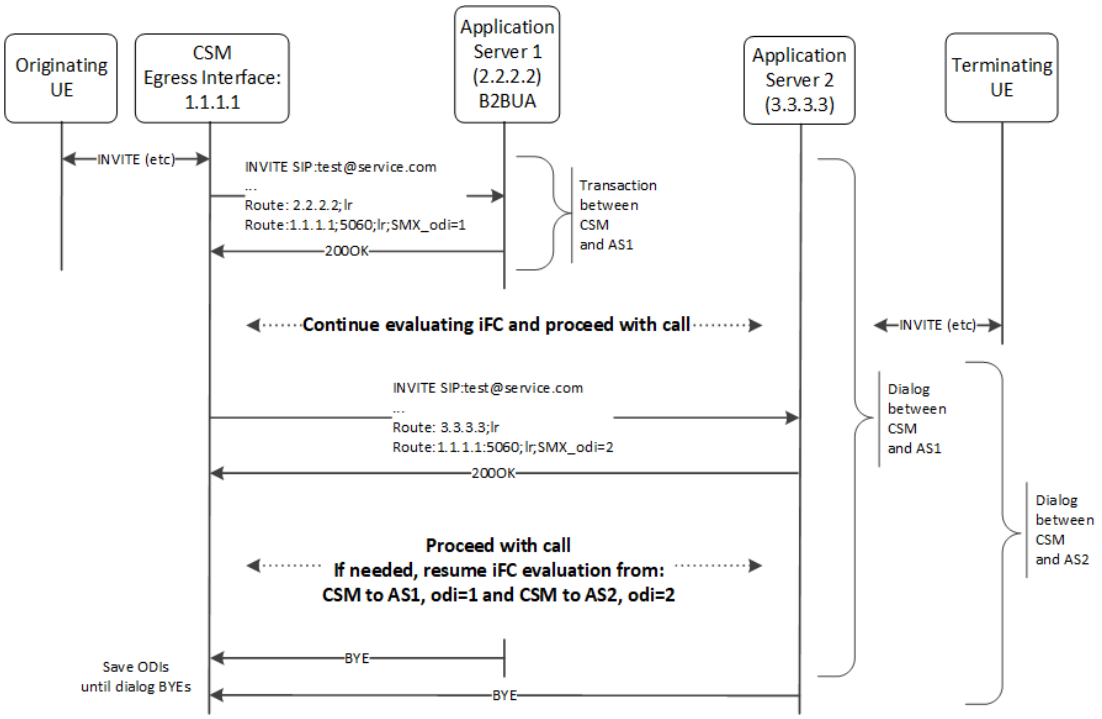
If the OCCSM discards the ODI when it receives the transaction's final message, it may experience problems, for example, with AS processes that are in B2BUA mode. In this mode, the AS may send a 200OK response to the request originator before forwarding the INVITE to the terminating side. If the OCCSM receives a message from the AS using a discarded ODI, the OCCSM restarts and repeats the entire iFC evaluation process for the call. Configuring ODI preservation provides value in applicable environments by avoiding this extraneous processing.

Enabling ODI preservation uses system resources to store this information. Balance this impact on resources with the value of deploying this behavior in your environment.

Key OCCSM behavioral detail with respect to the **preserve-odi** configuration includes:

- Upon receipt of an ODI, the OCCSM continues with iFC evaluation from the point where it left off for this ODI, using the same trigger-set and route-set as used within the initial iFC evaluation.
- If the originating user cancels a call, each AS also generates **CANCEL** message for each **INVITE**. The OCCSM clears ODIs from its memory when it processes the **CANCEL** for each original **INVITE**.
- If the HSS updates the user's service profile during iFC evaluation for a particular request message, for example, if the HSS sends a PPR to the OCCSM with a modified iFC, the OCCSM continues the current call with the existing in-memory Service Profile and uses the updated user-profile for subsequent calls.
- if an AS tries to send multiple request messages re-using an ODI, the OCCSM rejects those request messages and send 403 Forbidden response to the AS. The OCCSM only accepts the first request message from the AS using the same ODI.
- If the OCCSM does not receive the BYE, it retains the ODI until the session timers expire.
- An active OCCSM provides the service profile for each call, including all associated ODIs to its standby, enabling further service modification on a per-ODI basis despite a failover. This backup process requires a final message, a **200 OK** received from the AS. If a failover happens before the **200 OK**, the standby does not save the ODI.

The diagram below illustrates the OCCSM iFC evaluation behavior when you enable this parameter. By default, the OCCSM removes the ODI after the transaction between the OCCSM and AS1 is complete. With the **preserve-odi** parameter enabled, however, the OCCSM retains the ODI until it receives the **BYE** from AS1, which terminates the dialog between the OCCSM and AS1. If there are ODIs tracking service with other AS servers, shown below as AS2, the OCCSM retains those until the dialogs between itself and those servers terminates.



Given the diagram above, assume that the originating UE issues an invite towards the IMS core, which includes AS1 performing originating services and AS2 performing terminating services. The OCCSM initiates the iFC process with AS1, resulting in multiple transactions including the exchange of INVITEs using odi=1. When configured with **preserve-odi**, the OCCSM retains the service profile beyond the initial transaction. The OCCSM proceeds with terminating services via AS2, resulting in a similar set of transactions. In the meantime, the sequence has contacted the terminating UE. The sequence proceeds with completing the INVITE to 200OK interactions with AS1, AS2 and the terminating UE. The session proceeds until a UE issues a BYE to end the session. Subsequently, the process issues BYEs to terminate the dialogs with AS1 and AS2, at which time the OCCSM deletes those service profiles.

There are a large number of messages omitted from the diagram above for brevity and to highlight the dialogs between the OCCSM and the ASs.

Configuring ODI Preservation

Perform this sequence to enable ODI preservation on the OCCSM on a global basis.

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-config** object to edit.

```
ORACLE(sip-config)# select


ORACLE(sip-config)#
```

3. **preserve-odi**—Set this parameter to **enabled**. The default value is disabled.

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

4. Save your work.

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 Communications Core Session Manager 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 Communications Core Session Manager 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 - Registered User

When the Oracle Communications Core Session Manager receives an Initial request, it is validated as an originating request from or on behalf of a registered user when the following conditions are met:

- When the **ignore-psu-sesscase** option is not set:
  - 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.
- When the **ignore-psu-sesscase** option is set:
  - The request is a dialog creating request or a standalone request.
  - There is no "odi" parameter in the top route of the request.
  - There is an "orig" parameter in the top route of the request.
  - The served user is registered

Originating request - Unregistered User

When the Oracle Communications Core Session Manager receives an Initial request, it is validated as an originating request from or on behalf of an unregistered user when the following conditions are met:

- When the **ignore-psu-sesscase** option is not set:
  - The request is a dialog creating request or a standalone 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.
- When the **ignore-psu-sesscase** option is set:
  - The request is a dialog creating request or a standalone request.
  - There is no "odi" parameter in the top route of the request.
  - There is an "orig" parameter in the top route of the request.
  - The served user is unregistered

Terminating Requests - Registered User

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

- When the **ignore-psu-sesscase** option is not set:
  - 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 Communications Core Session Manager.
  - If the Request-URI changes when visiting an application server, the Oracle Communications Core Session Manager 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.
- When the **ignore-psu-sesscase** option is set:
  - The request is a dialog creating request or a standalone request.
  - There is no "odi" parameter in the top route of the request.
  - There is no "orig" parameter in the top route of the request.
  - The served user is registered

Terminating Requests - Unregistered User

See the IFC Support for Unregistered Users section in the Configuration Guide for this case.

- When the **ignore-psu-sesscase** option is not set:
  - If the Request-URI changes when visiting an application server, the Oracle Communications Core Session Manager 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.
- When the **ignore-psu-sesscase** option is set:
  - The request is a dialog creating request or a standalone request.
  - There is no "odi" parameter in the top route of the request.
  - The served user is not registered.

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 an unregistered user

Third Party Registration for an Implicit Registration Set

When using iFCs, the Oracle Communications Core Session Manager performs third party registrations based on the iFC downloaded for each PUID. By default, the Oracle Communications Core Session Manager performs third party registration for the service profiles of all PUID's in a user's implicit registration set. This is compliant with 3GPP specifications. The system includes any shared or default iFCs that apply to each PUID during this process. The system performs this function when it receives user-initiated de-registrations, but not when it receives RTRs. If desired, the user can configure the Oracle Communications Core Session Manager to perform third party registration for only the REGISTERED PUID in the registration using a **sip-registrar** option.

> **Note:**
>
> The Oracle Communications Core Session Manager does not attempt third party registration for any barred, tel or wildcard PUIDs.

The user can verify all third party registrations using the **show registration sipd by-user [user] detailed** command. Example output is shown below.

```
ORACLE# show registration sipd by-user 234 detailed

Registration Cache (Detailed View)    Wed Sep 16 2015  10:57:44

User: sip:234@acme-ims.com
  Registered at:  2015-09-16-10:57:40    Surrogate User: false
  Emergency Registration? No
  ContactsPerAor Rejects 0
  ContactsPerAor OverWrites 0

  Contact Information:
    Contact:
      Name: sip:234@acme-ims.com
      Valid: true
      Challenged: false
      Registered at: 2015-09-16-10:57:40
      Last Registered at: 2015-09-16-10:57:40
      Expire: 3596
      Local expire: 296
      Half: 1796

      Registrar IP: 0.0.0.0
      Transport: UDP
      Secure: false
      Local IP: 192.168.53.99:5060

      User Agent Info:
        Contact: sip:234@192.168.53.181:5060
        Realm: core
        IP: 192.168.53.181:5060
```

```
      SD Info:
        Contact: sip:234-tbcktcgo177fc@192.168.53.99:5060
Call-ID: 1-5853@192.168.53.181
      Path: <sip:234@192.168.53.181:5060;lr;p-acme-serving>

    Associated URI(s):
      URI: sip:234@acme-ims.com
      Status: Non-Barred
         Filter Criteria:
           Priority: 0
             Filter: ((method == REGISTER)) or
              ((method == INVITE))
             Application Server: sip:172.16.17.10:5060

      URI: sip:1@acme-ims.com
      Status: Non-Barred
         Filter Criteria:
           Priority: 0
             Filter: ((method == REGISTER)) or
              ((method == INVITE))
             Application Server: sip:172.16.17.10:5060
           Priority: 1
             Filter: ((method == INVITE)) or
              ((method == REGISTER))
             Application Server: sip:172.16.53.181:5065

      URI: tel:135
      Status: Barred
         Filter Criteria:
           Priority: 0
             Filter: ((method == INVITE)) or
              ((method == REGISTER))
             Application Server: sip:172.16.53.181:5065
           Priority: 1
             Filter: ((method == INVITE)) or
((method == REGISTER))
             Application Server: sip:172.16.53.181:5095


    Third Party Registration(s):
      Third Party Registration Host: 172.16.17.10
      Registration State: REGISTERED
      Last Registered at: Never
      Third Party Registration Host: 172.16.53.181
      Registration State: REGISTERED
      Last Registered at: Never
```

The user can check for third party registrations errors using the **show sipd third-party-reg all** command. Example output is shown below.

```
ORACLE# show sipd third-party-reg all
3rd Party Registrar             SA State   Requests     200OK
Timeouts    Errors
(D)111.11.17.10                   INSV         1         1
```

```
0          0
(D)111.11.53.181                    INSV          1          1          0          0
```

The user can disable the default behavior and perform third party registration only for the PUID in the REGISTER by configuration. Disabling this behavior can improve system performance by preventing the system from having to walk through large PUID sets for large numbers of ASs. The ACLI syntax for disabling this functionality using the **disable-thirdPartyReg-for-implicit-puid** setting follows.

```
ORACLE(sip-registrar)#options +disable-thirdPartyReg-for-implicit-puid
```

> **Note:**
>
> Prior to this version, the Oracle Communications Core Session Manager's default behavior was the same as if the **disable-thirdPartyReg-for-implicit-puid** option was set in the SIP registrar. Users upgrading to this version of the Oracle Communications Core Session Manager must set the **disable-thirdPartyReg-for-implicit-puid** option to retain the previous behavior.

TEL URI Replacement with SIP URI in R-URI to AS

When the OCCSM receives a request containing a TEL URI from the Media Gateway Control Function (MGCF), it sends the TEL URI as an R-URI to the Application Server (AS) to perform services. However, in some implementations, the AS does not accept TEL URI and requires the trigger to be based on SIP URI. This feature, when enabled, causes the OCCSM to replace the TEL R-URI with a SIP URI based on the first SIP user in the implicit set.

In the current implementation of the OCCSM for terminating calls, when the OCCSM receives an R-URI with SIP user=phone (for example, "sip:+359888528650@sip.mtel.bg; user=phone"), the OCCSM replaces the SIP URI with a TEL URI and further uses the TEL URI (for example, "tel:+359888528650") for Location Information Requests (LIR) and Server Assignment Requests (SAR) when the user is not in the registration cache. The Server Assignment Answer (SAA) provides the Public Identity "sip:tel.359888528650@sip.mtel.bg" in the Service Profile as it's part of the implicit registration set and the OCCSM stores it in its registration cache. Then, based on the Service Profile for the TEL URI, the OCCSM triggers the AS using the R-URI "tel:+359888528650". However, in some implementations, for requests coming from the MGCF, the OCCSM receives requests with a TEL URI which is sent as an R-URI to the AS while doing services, but the AS does not accept TEL URIs and requires the trigger to be based on a SIP URI.

To rectify this deficiency, this feature, when activated and when the OCCSM is the assigned Serving Call Session Control Function (S-CSCF), causes the OCCSM to replace the TEL R-URI with the first SIP URI in the implicit set for the TEL user; it then performs services based on the trigger for the user of the first implicit SIP URI. Once a TEL URI is changed to a SIP URI to perform services it will not be changed back to a TEL URI for the entire call flow. When this feature is enabled, the OCCSM uses the first SIP user entry in the implicit set and performs services for the user irrespective of whether the user is in the registration cache.

TEL URI Replacement with SIP URI in R-URI to AS Configuration

Configuration changes occur in real time and do not require rebooting.

1. Access the **ifc-profile** configuration element.

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

2. Select the **ifc-profile** object to edit.

```
ORACLE(ifc-profile)# select
<name>:
1:  name=ifc_appserver

ORACLE(ifc-profile)#
```

3. **options** — Set the options parameter by typing **options** , a space, the option name **replace-tel-ruri-with-implicit-sip** with a plus sign in front of it, and then press Enter. You must prepend the new option with a plus sign to append the new option to the IFC profile's options list. If you type the option without the plus sign, you will overwrite any previously configured options.

   ORACLE(ifc-profile)# options +replace-tel-ruri-with-implicit-sip

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

Additional Options

- The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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

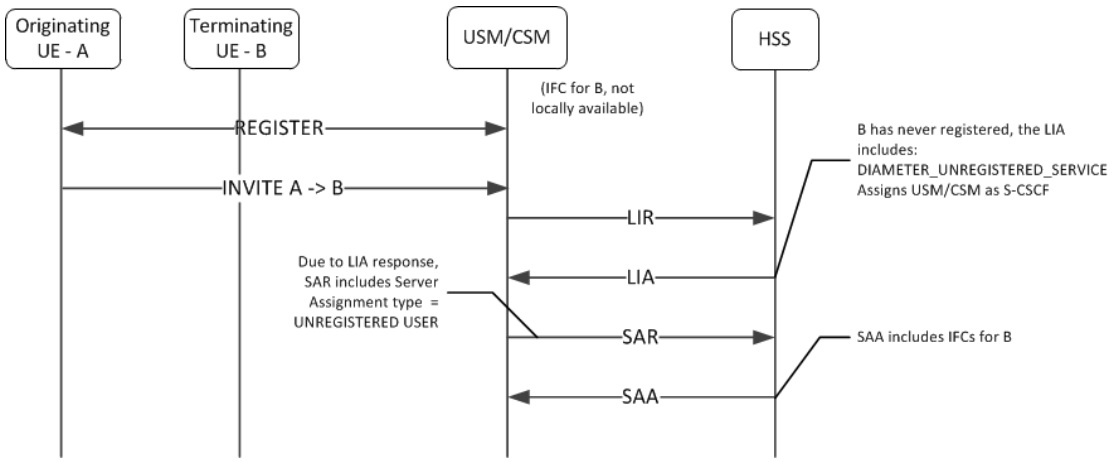
The Oracle Communications Core Session Manager can download Initial Filter Criteria (IFC) from the HSS for unregistered users. This section displays applicable message sequence diagrams.

UE-terminating requests to an unregistered user

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

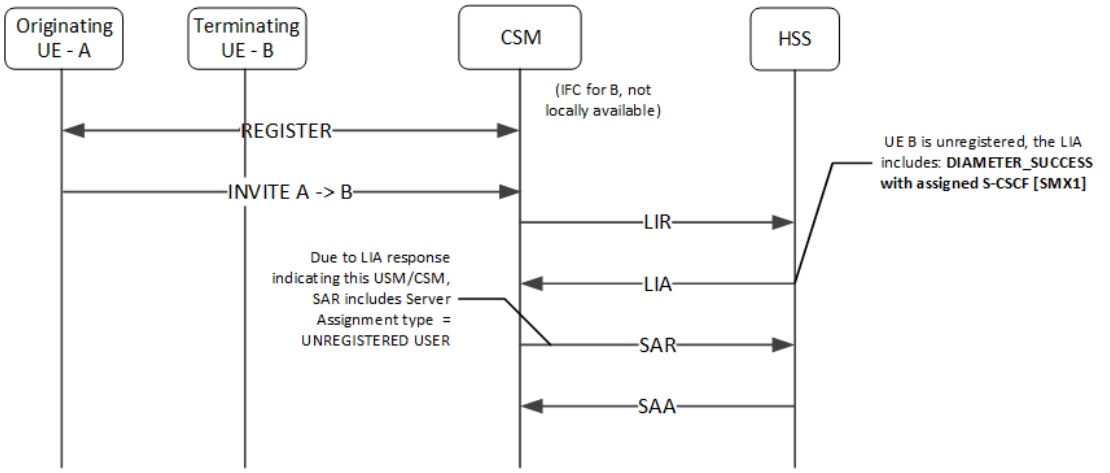
Terminating UA - Unregistered

UE has never registered.

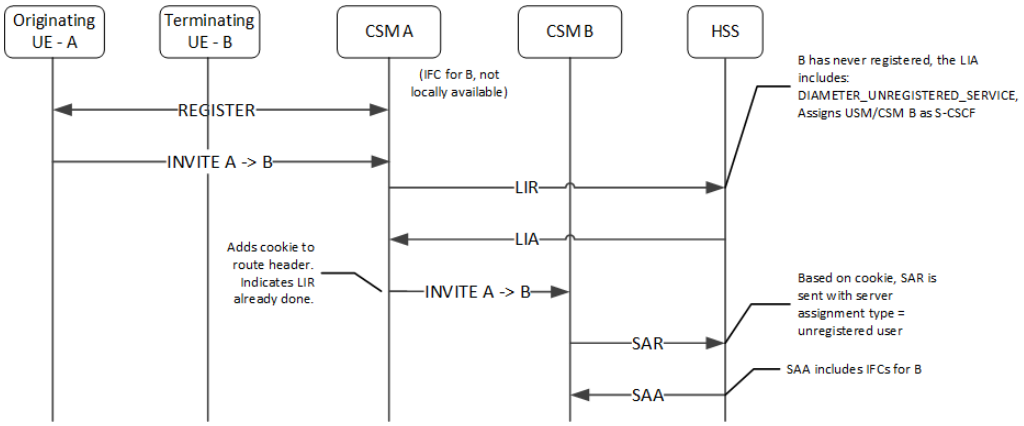


Terminating UA - Unregistered

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 Communications Core Session Manager



UE Subsequent Registration

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

Caching the Downloaded IFC

When the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager to act solely as an I-CSCF. When disabled, the Oracle Communications Core Session Manager always forwards requests from unregistered users to the serving group.
   The default is enabled, which retains the S-CSCF function on this Oracle Communications Core Session Manager.
7. **serving-group**—Set this parameter to a Session Agent Group (SAG) name. The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager itself. This establishes a backup scenario in case the HSS is not responsive.

To support the SiFC feature on the Oracle Communications Core Session Manager, 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 Communications Core Session Manager 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 Communications Core Session Manager forwards the registration to the HSS normally. Given SiFC configuration however, the HSS sends a service-profile containing the SiFC identifiers to the Oracle

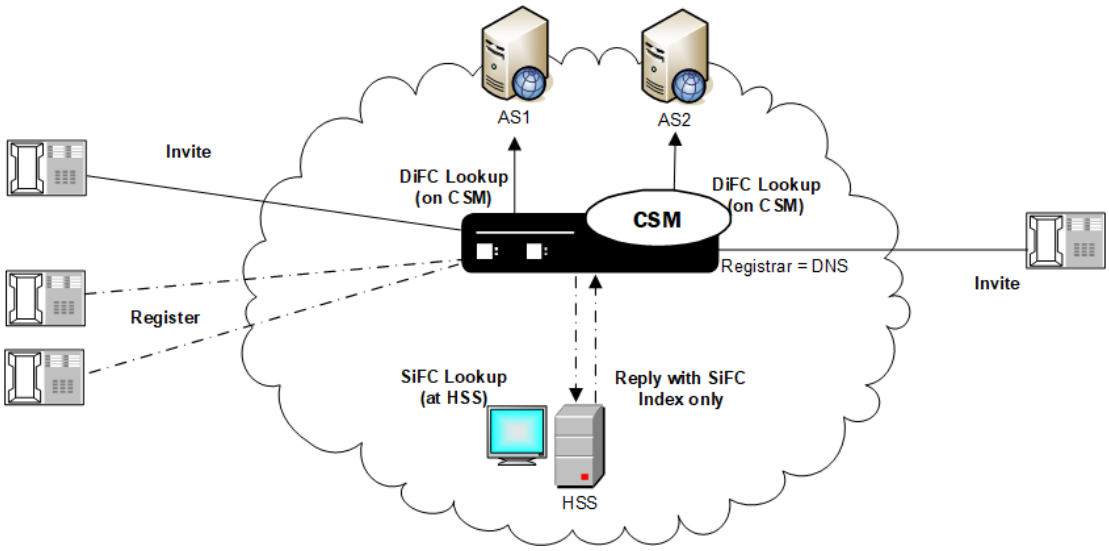
Communications Core Session Manager rather than the entire service definition. The Oracle Communications Core Session Manager parses these identifiers and maps the user to the locally stored filter criteria.

The <IFCSet id =”x”> tags in the XML file on the Oracle Communications Core Session Manager map to the HSS identifiers.

DiFC Usage

In contrast to SiFCs, the Oracle Communications Core Session Manager fires DiFCs within the context of a session. During the session, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager. 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 Communications Core Session Manager 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 Communications Core Session Manager evaluates explicitly downloaded iFCs first when determining where to look for a service. If the Oracle Communications Core Session Manager 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 Communications Core Session Manager validates the SiFC and DiFC files whenever you Activate your configuration.

SiFC and DiFC Configuration

To configure the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager caches filter criteria information that uses this wildcard syntax. This avoids the need for SAR/SAA exchanges between the Oracle Communications Core Session Manager and the HSS every time an entity requests the service. The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager.

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 Communications Core Session Manager establishes and manages multiple Cx connections with each applicable HSS. This management is achieved by connection identifiers on the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager selects the next highest priority HSS server for all operations if the first HSS fails. The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager marks that HSS as out of service.

The Oracle Communications Core Session Manager forwards unacknowledged messages to subsequent HSSs based on strategy. It changes the destination host AVP of these messages and marks then 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 Communications Core Session Manager attempts to establish diameter connections with out of service HSS servers. When those connections succeed, the Oracle Communications Core Session Manager 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 Communications Core Session Manager. 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 Communications Core Session Manager 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
```

Diameter Message Manipulations

The Oracle Communications Core Session Manager can perform manipulations on all grouped and non-grouped AVPs. This is referred to as Diameter Manipulation Rules (DMR). A message manipulation is the ability to search for a predefined string within an AVP and then replace it with another value. This is similar to the Oracle Communications Core Session Manager's header manipulation rules functionality.

A diameter manipulation configuration element is defined by a name parameter. You can optionally add a description field to the diameter manipulation. Within each diameter manipulation you can configure multiple diam manipulation rule subelements. The manipulation rule subelements are the configuration where AVPs are identified, searched, and in which the data is replaced.

The user can apply diameter manipulations to external policy server configurations. These manipulations affect traffic between the Oracle Communications Core Session Manager and the applicable policy server.

> **Note:**
>
> The Oracle Communications Core Session Manager also supports diameter manipulation across the Cx interface, with the user configuring these manipulations to home subscriber server configurations. The range of manipulation supported over the Cx interface is the same as that over the Rx interface.

Manipulation Rule

Creating a manipulation rule is divided into three parts, defining the message type and AVP where the manipulation is performed, defining how the search on the AVP is performed, and defining what to replace the found string with.

You must first define the name parameter of the diam manipulation rule configuration element. Optionally you can add a descr avp code parameter that is a description of this manipulation rule.

Naming Diameter Manipulations

The Oracle Communications Core Session Manager restricts the way you can name a diameter-manipulation rule. Specifically, observe the rules below when naming manipulation elements:

- Character limit - diameter manipulation rule names cannot be longer than 24 characters.
- Numeric characters - diameter manipulation rule names must not start with a numeric character.
- Special characters - Special characters are not supported within diameter manipulation rule names, with the exception of the underscore and hyphen characters.
- Capital letter characters - The Oracle Communications Core Session Manager includes reserved keywords that are named using all-capital letters. To avoid conflicts between keywords and diameter manipulation rules, do not configure diameter manipulation rule names using ll capital letters.

Note that, although diameter-manip-rule and avp-header-rule names have the same character-type restrictions, they do not have a character limit.

Message Based Testing

When the Oracle Communications Core Session Manager first receives a message applicable for manipulation, it checks if the message type as **request**, **response**, or **all** as configured in the msg type parameter. The Oracle Communications Core Session Manager continues to look at the message command code. Matching values are defined by configuring the msg cmd code parameter with a numeric value. You can enter a single value, multiple comma-separated values, or you can leave this parameter blank to indicate all message codes.

AVP Search Value

After the Oracle Communications Core Session Manager has identified the messages where it can look for an AVP, the avp code must be defined with a numeric AVP value to be searched. Also the AVP data type is defined so Oracle Communications Core Session Manager knows how to correctly parse the AVP once found. This is configured in the avp type parameter with valid values of: octet-string, octet-hex, integer32, unsignedint32, address, utfstring, diameteruri, or enumerated.

The comparison type is defined so that the Oracle Communications Core Session Manager knows the correct way to determine if the match value appears in the avp code. Valid comparison types are:

- Case-sensitive—The comparison-type of both case-sensitive and case-insensitive literally compares the value contained in the match-value against the received value.
- Case-insensitive—The comparison-type of both case-sensitive and case-insensitive literally compares the value contained in the match-value against the received value.
- pattern-rule—the match-value is treated as a regular expression.
- boolean—Used when it is necessary to compare the results of two or several manipulation rules with varying logic (e.g. if ($rule1 & ($rule2 | $rule3))). When the comparison-type is set to boolean, the match-value will be evaluated as a boolean expression.

Finally, the match operation is configured by defining a match value, which is the string to find. The Oracle Communications Core Session Manager evaluates if the match value is found in the avp code AVP. You may also leave the match value empty for the DMR to be applied on the AVP without testing for a match.

Reserved Keywords

The Oracle Communications Core Session Manager employs certain reserved keywords as a short hand for configuration/message parameters. These keywords are as follows:

HOSTNAME—This keyword refers to the agent hostname this rule is being referenced by. If the rule is applied to a realm/interface then the value of the hostname keyword will be an empty string. If the rule is applied to the group, then the hostname for the agent picked will be used.

ORIGINREALM—This keyword refers to the Origin-Realm value for the configured realm/interface. If the rule is applied to a Diameter Director Agent, then the origin-realm value is derived from the Diameter Director Interface the agent belongs to.

ORIGINHOST—This keyword refers to the Origin-Host value for the configured realm/interface. If the rule is applied to a Diameter Director Agent, then the origin-host value is derived from the Diameter Director Interface the agent belongs to.

Actions on Found Match Value

When the match-value is found, the Oracle Communications Core Session Manager references the action parameter. This is configured as either **none**, **add**, **delete**, **replace**, **store**, **diameter-manip**, **find-replace-all**, **log** or **group-manip** and indicates the action to perform on the string. If the match-value is not found, the Oracle Communications Core Session Manager continues processing the message without any AVP manipulation. These actions mean the following:

none

None disables a manipulation rule.

add

This action inserts the value from the **new value** parameter, creates a new AVP of the specified type and adds it to the list of AVPs at the specified position.The message length and padding are re-computed to account for this newly added AVP.

delete

This action removes the specified AVP from the list of AVPs being operated on. The message length and padding will be re-computed to account for this deleted AVP.

replace

This action substitutes the existing value with the **new value** parameter. The message length and padding and AVP length and padding will be re-computed to account for changes. This is mostly applicable to variable length AVP types such as octet-string.

store

Each manipulation rule has the ability to store the data that was contained in the AVP as a string. This is useful for creating conditional logic to make decisions whether to execute other manipulation rules or to duplicate information within the Diameter message.

Every manipulation rule performs an implicit store operation prior to executing the specified action type. All store operations are based on regular expression patterns configured in the **match value**. The information that is stored in the rule is the resultant of the regular expression applied against the specified string. The **comparison-type** is ignored when the action is set to store as the Oracle Communications Core Session Manager assumes that the **match value** is a regular expression. Conditional logic cannot be used to make a decision whether to perform a store operation or not; storing always occurs. Values stored in a manipulation rule are referred to as user defined variables.

diameter-manip

When the action is set to **diameter-manip**, the Oracle Communications Core Session Manager executes the diameter-manipulation **name** configured in the **new value**. The diameter-manipulation name in the **new value** must match another diameter-manipulation name exactly (case is sensitive).

diameter-manip action type is primarily to reuse diameter-manipulations that may be common to other use cases. A diameter-manip action should never call back to itself either directly or indirectly through a different diameter-manipulation.

find-replace-all

The **find-replace-all** action searches the object's string for the regular expression defined in the match-value and replaces every matching occurrence of that expression

with the value supplied in the **new value**. If the regular expression contains sub-groups, a specific sub-group can be specified to be replaced by adding the syntax [[:n:]] at the end of the expression, where n is the sub-group index (zero-based). When the action is find-replace-all, the comparison-type is ignored and the match-value is always treated as a regular expression.

group-manip

The **group manip** action is used to manipulate AVPs inside grouped AVPs. For this diameter manipulation, you must set the avp-type to **grouped**.

The **group manip** action is similar to the **diameter manip** action in that the Oracle Communications Core Session Manager executes the diameter-manipulation configured in the new value.

There is an important difference between **group manip** and diameter manip. The **diameter-manip** action is context insensitive, meaning when it jumps from one diameter-manipulation to the next diameter-manipulation, it starts looking for the specified AVP from the top of the message.

The **group manip** action is context sensitive, meaning when the processing jumps from one diameter-manipulation to the next diameter-manipulation, it will look for the specified AVP within the grouped AVP. In short, the **group manip** works at an AVP level. All actions are allowed in the subsequent manipulations that are invoked, with the key difference being that those manipulation rules will be applied to the current grouped AVP in the chain. Thus it is possible to apply manipulation to an AVP at any level in the hierarchy.

Consider the following examples:

In order to replace the experimental-result, experimental-result-code AVP value from 5002 to 3002, a **group manip** can be configured as follows:

```
diam-manipulation
        name                            manipExpRslt
        description
        diameter-manipulation-rule
                name                            expRslt
                avp-code                        297
                descr-avp-code
                avp-type                        grouped
                action                          group-manip
                comparison-type                 case-sensitive
                msg-type                        response
                msg-cmd-codes                   316,317,318
                match-value
                new-value                       expRsltCode
        last-modified-by                admin@console
        last-modified-date              2011-09-13 18:50:33
diam-manipulation
        name                            expRsltCode
        description
        diameter-manipulation-rule
                name                            expRsltCode
                avp-code                        298
                descr-avp-code
                avp-type                        unsignedint32
```

```
            action                          replace
            comparison-type                 case-sensitive
            msg-type                        response
            msg-cmd-codes                   316,317,318
            match-value                     5002
            new-value                       3002
    last-modified-by                admin@console
    last-modified-date              2011-09-13 18:56:14
```

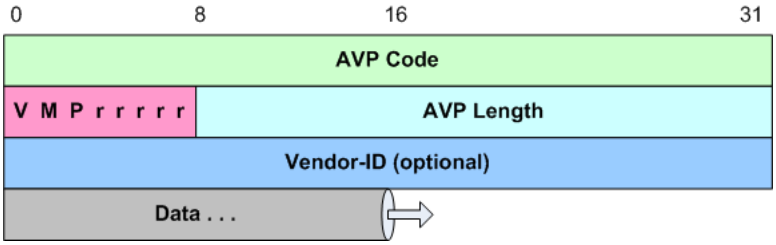
Further, if you want to add a new AVP called AvpD at the following location in the chain of AVPs Message: GrpAvpA, GrpAvpB, GrpAvpC, AvpD, then the manipulation chain would look like this

- diameter-manipulation (name=grpAvpA, action=group-manip, new-value=grpAvpB)
- diameter-manipulation (name=grpAvpB, action=group-manip, new-value=grpAvpC)
- diameter-manipulation (name=grpAvpC, action=group-manip new-value=AvpD)
- diameter-manipulation (name=AvpD action=add new-value="added new value")

> **Note:**
>
> using diameter-manip from inside the group-manip chain may have unexpected impact and must be avoided.

AVP Header Manipulation

In addition to manipulating AVPs, you can manipulate the AVP header itself. After performing AVP DMR, the AVP length and padding is recomputed. This is crucial for scenarios where a vendor-id is added or removed from the header. This functionality is configured in the avp header rules sub element. The following represents a single AVP's header:



AVP Flag Manipulation

You can manipulate the AVP flags by configuring the **header-type** parameter to **avp-flags**, this invokes operation on the flags byte in the AVP header. AVP flags are 1 byte long, where the first 3 bits represent (1) vendor, (2) must and (3) protected. The last 5 bits are reserved.

The vendor flag is critical to consider here, since it has interdependency with Vendor-Id field in the header shown above. As per RFC 3588, The 'V' bit, known as the Vendor-Specific bit, indicates whether the optional Vendor-ID field is present in the AVP header. When set, the AVP Code belongs to the specific vendor code address space. Please find specific details about the rest of the flags in RFC3588 Section 4.1.

When manipulating AVP flags, a bitwise comparison is performed between the received value and the **match value**. For ease of configuration, the **match value** is configured as a comma-separated enumerated list of vendor, must, and protected. So the **new value** and the **match value** will be used to indicate what bit in the **avp-flag** to operate on. If the **match value** is empty, the configured action is performed without any matching tests. In addition, the new value is configured using the same enumerations. The AVP header rules configuration element appears as follows:

```
avp-header-rules
   name         replaceAvpFlags
   header-type  avp-flags
   action       replace
   match-value  must,protected
   new-value         must
```

According to the example configuration, the Oracle Communications Core Session Manager makes a positive match when only the must and protected bits are set in the received avp-flags. If only the 'M' bit is set, then the match fails, the Oracle Communications Core Session Manager continues to the next header-rule.

When the match is successful (or if the **match value** is left empty), the configured **action** is performed. Consider all following actions applicable to the avp header rules sub element:

- none— no action will be performed
- add—the flags specified in the **new value** are enabled in the header, and state of the existing flags will not be changed.
  If **new value** is empty, the add operation will not be performed.

  If the **new value**=vendor, and the 'V' bit was not originally set, then the 'V' bit is now be set including a vendor-id of 9148 inserted into AVP. 9148 is the Acme Packet vendor-id assigned by IANA. It is expected that a second header-rule will be used to change this to the desired vendor-id.
- replace—all the received avp-flags will be reset. The values in the **new value** parameter will be set.
  If the **new value** is empty, the replace operation will not be performed.

  If the **new value**=vendor, and the 'V' bit was not originally set, then the 'V' bit will now be set and also a vendor-id of 9148 (Acme Packet's vendor-id) is added to the AVP header. A second header-rule may be used to change this to the desired vendor-id.

  If the **new value** does not contain vendor, and the 'V' bit was originally set, then the 'V' bit will be cleared and the vendor-id will also be set to 0 effectively removing it from the AVP header.
- delete—all flags configured in **new value**, will be deleted from the AVP header
  If the **new value** is empty, then no flags are deleted.

  If the particular flag is already empty, then it will be skipped. For example, if the **new value**=must and 'M' bit is not set, after applying the DMR the 'M' bit will still be not set.

If the **new value**=vendor, then the 'V' bit will be cleared (if not cleared already) and the vendor-id is set to 0, effectively removing the vendor-id from the avp-header.

vendor-id Manipulation

You can manipulate the Vendor ID value in the AVP header by configuring the **header-type** parameter to **avp-vendor-id**. This performs the DMR manipulation on the 4-byte vendor-id in the AVP header. AVP vendor id is present in the AVP header only when the ‘V’ bit is set in the flags. This is important because the DMR application relies upon the bit being set to determine where the data payload begins.

The avp-vendor-id search invokes an unsigned integer comparison between the received value and the **match-value**. If the **match-value** is empty, the configured action is performed without doing any match.

For the case where **match-value** is non-empty, as in the following example, the DMR engine checks whether the ‘V’ bit is set in the received header. If not, then the vendor id is not present either and the comparison is unsuccessful. If the ‘V’ bit is set, and the match succeeds, the match is successful. (An unsuccessful match has the DMR proceed to the next header-rule.)

```
avp-header-rules
   name          replaceAvpFlags
   header-type   avp-vendor-id
   action        add
   match-value   9148
   new-value     10415
```

When the match is successful (or if the **match value** is left empty), the configured **action** is performed. Consider all following actions:

- none—no action will be performed
- add—a configured vendor-id value in the **new-value** parameter is added to the AVP header and the ‘V’ bit set to indicate it’s presence. If you prefer to set the ‘V’ bit in an AVP, it is better to do an avp-vendor-id action first and then manipulate the rest of the flags.
  If the **new-value** is empty, the add operation is not performed.

  If a vendor-id already exists in the AVP header, then it is replaced by **new-value**.
- replace—the existing vendor-id value is replaced with the **new-value**.
  If the **new-value** is empty, the replace operation is not performed.

  If the vendor-id does not exist in the header, then one will be added with the **new-value** and the ‘V’ bit is set to indicate its presence.
- delete—the vendor-id will be removed from the AVP header and ‘V’ bit will be reset to indicate its absence.
  If the **new-value** is empty, then the delete operation will not be performed.

  If the vendor-id does not exist, then the delete operation is not performed.

Multi-instance AVP Manipulation

Some AVPs can appear multiple times within a message. To choose a specific occurrence of an AVP, the **avp code** parameter supports indexing logic. By default, the Oracle Communications Core Session Manager operates on all instances of the

specified AVP. However, after configuring an avp-code, you can specify in square brackets, a specific instance of the AVP on which to operate on. The indexing is zero-based. For example,

```
diameter-manipulation-rule
                name                            manip
                avp-code                        293[2]
```

Special characters that refer to non-discrete values are:

- Last occurrence—avp-code[^]
- All—avp-code [*]

The last (^) character is used to refer to the last occurring instance of that AVP. Any [^] refers to the first matching header that matches the specified conditional matching criteria. All [*] is the default behavior, and operates on all headers of the specified-type. For **group manip** action, the AVP index applies to the instance within that grouped AVP.

ACLI Instructions

Diameter Manipulation

To configure a diameter manipulation configuration element:

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

   ```
   ORACLE# configure terminal
   ```

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

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

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

   ```
   ORACLE(session-router)# diameter-manipulation
   ORACLE(diameter-manipulation)#
   ```

4. **name**—Enter the name of this Diameter manipulation element.
5. **description**—Enter an optional description for this Diameter manipulation.
6. Type **done** and continue.

Manipulation Rule

1. Type **diameter-manip-rules** to continue and enter individual policy attributes. The system prompt changes to let you know that you can begin configuring individual parameters.
2. **name**—Enter the name of this manipulation rule. This value will be referenced in when creating a chain of rules.
3. **descr-avp-code**—Enter a description of the AVP code to manipulate.

4. **msg-cmd-code**—Enter the command code number of the message to execute the manipulation on.
5. **msg-type**—Set this to the type of message this manipulation applies to as **request**, **response**, or **all**.
6. **avp-code**—Enter the AVP by code number where this manipulation applies. You can add a multi instance identifier to the end of the avp code value, enclosed in brackets.
7. **avp-type**—Set this to the data type of the content of the match field. Refer to the Diameter standards document for the encodings of individual AVPs. Valid values are:

   none | octet-string | octet-hex | integer32 | unsignedint32 | address | diameteruri | enumerated | grouped
8. **match-value**—Enter the value within the match-field to find and make a positive match on.
9. **action**—Enter either **none, add, delete, store, diameter-manip, group-manip, find-replace-all**, or **replace** as the action to take after making a positive match on the previously entered match-value.
10. **new-value**—Enter the value that should be added or replaced in the old match-value's position.
11. Type **done** and continue.

AVP Header Manipulation

1. Type **avp-header-rules** to configure AVP header manipulation rules. The system prompt changes to let you know that you can begin configuring individual parameters.
2. **name**—Enter the name of this AVP Header manipulation rule.
3. **header-type**—Set this to either avp-flag or avp-vendor-id depending on which part of the AVP header you are manipulating.
4. **action**—Enter either none, add, delete, or replace as the action to take after making a positive match on the previously entered match-value.
5. **match-value**—Enter the value in the AVP flag field or Vendor ID field to match against.

   When matching in the avp flag field, then match-value is interpreted as comma-separated list of enumerated values <vendor,protected,must>. When matching in the Vendor ID field, then match-value is interpreted as 32 bit unsigned integer <1-4294967295>
6. **new-value**—Enter the new value when the match value is found. The resultant new value is entered as the match value is configured.
7. Type **done** to save your work.

Applying the Manipulation

You can apply a diameter manipulation by name to an external policy server configuration. This element contains the two applicable parameters: diameter-in-manip and diameter-out-manip.

> **Note:**
>
> The user can also apply diameter manipulation to a home subscriber server configuration using these same parameters.

1. To navigate to external policy server configurations from 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. Enter the configuration element where you wish to apply the manipulation.
4. Type **ext-policy-server** and press Enter.

   ```
   ORACLE(media-manager)# ext-policy-server
   ORACLE(ext-policy-server)#
   ```

5. Type **select** and then choose the pre-configured external policy server you want to configure.

   ```
   ORACLE(ext-policy-server)# select
   <hostname>:
   1: ext-pol-svr-1
   2: ext-pol-svr-2
   selection: 1
   ```

   You may now add a Diameter manipulation to one or both directions of message flows.

6. **diameter-in-manip**—Enter a name of an existing diameter manipulation to apply as received by the Oracle Communications Core Session Manager on this element.
7. **diameter-out-manip**—Enter a name of an existing diameter manipulation to apply as forwarded from the Oracle Communications Core Session Manager on this element.
8. Type **done** and continue.

Diameter Manipulation Example - Supported Features AVP

This section shows you a configuration example for diameter manipulation rules. This section shows the configuration for the rule that the Oracle Communications Core Session Manager 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.

For this manipulation rule, the Oracle Communications Core Session Manager inserts a Supported Features AVP into every request.

This is a sample of the configuration:

```
diameter-manipulation
        name                                    diamManip1
        description
        diameter-manip-rule
                name                                    rule1
                avp-code                                628
                descr-avp-code
                avp-type                                grouped
                action                                  add
                msg-type                                request
                msg-cmd-code                            265
                comparison-type                         case-sensitive
                match-value
                new-value
        diameter-manip-rule
                name                                    rule2
                avp-code                                628
                descr-avp-code
                avp-type                                grouped
                action                                  group-manip
                msg-type                                any
                msg-cmd-code
                comparison-type                         case-sensitive
                match-value
                new-value                               diamManip2
```

This second rule, which defines a new value for the first rule, builds the Feature-List-ID and Feature-List AVPs to be included within the context of the Supported Features group.

```
diameter-manipulation
        name                                    diamManip2
        description
        diameter-manip-rule
                name                                    rule1
                avp-code                                266
                descr-avp-code
                avp-type                                unsignedint32
                action                                  none
                msg-type                                any
                msg-cmd-code
                comparison-type                         case-sensitive
                match-value                             10
                new-value
        diameter-manip-rule
                name                                    rule2
                avp-code                                629
                descr-avp-code
                avp-type                                unsignedint32
                action                                  none
```

```
            msg-type                        any
            msg-cmd-code
            comparison-type                 case-sensitive
            match-value                     11
            new-value
    diameter-manip-rule
            name                            rule3
            avp-code                        630
            descr-avp-code
            avp-type                        unsignedint32
            action                          none
            msg-type                        any
            msg-cmd-code
            comparison-type                 case-sensitive
            match-value                     124
            new-value
```

4

Local Subscriber Tables

Local Subscriber Table

A local subscriber table (LST) is an XML formatted file that contains one or more URIs that determine which services are provided for each URI on a per-call basis. The LST can assign services using distinct Public Service Identities (PSIs) or wildcarded PSIs via iFC processes that use the shared-ifc-set file you configure on the Oracle Communications Core Session Manager (OCCSM). The LST is saved locally on the OCCSM's file system.

LSTs enable a standalone OCCSM node or high-availability (HA) pair to forego relying on an external user database. Thus the OCCSM does not need to communicate with a server to apply services. This can reduce operational complexity and cost by alleviating the need for the OCCSM to execute the SAR queries to fetch the service profile applicable to PSI's.

> **Note:**
>
> This function uses the OCCSM as I/S-CSCF. There is no support for this function performed by the SLRM.

> **Note:**
>
> The OCCSM does not support end user authentication and registration using LST.

LST Runtime Execution

The OCCSM loads the LST when it boots up, and when the configuration is appropriately set. It can then apply incoming calls with services based on the LST. If the OCCSM cannot load an LST file:

1. The following log message is recorded at the NOTICE level:

   ```
   LST [table-name] was not loaded - [filename] has error loading XML file
   ```

2. The OCCSM prints this message on the ACLI.
3. The OCCSM sends out an SNMP trap indicating the file could not be loaded.
4. The OCCSM routes the call to the destination with no service execution.

If the OCCSM can load the LST file:

1. The OCCSM performs a look up to identify the iFC ID corresponding to the subscriber in the LST file. This can be a wildcard or a distinct PSI definition.
   a. If the subscriber does not match an LST entry, the OCCSM proceeds to step 4.

b. If the OCCSM does not find an applicable iFC ID, it proceeds to step 4.

2. Once the iFC ID is identified, the OCCSM performs a look-up in the **shared-ifcs** file to identify the service to be executed.
3. The OCCSM executes the iFC.
4. The OCCSM executes any default iFC.
5. After service execution, the OCCSM routes the call towards the destination.

LST File Format

The LST file format is as follows:

```
<localSubscriberTable>
<subscriber username="sip:alic!.*!@open-ims.lst" ifcid="1"/>
<subscriber username=”tel:4832376630!.*!”  ‟ifcid=2”/>
</localSubscriberTable>
```

The LST file's elements includes the subscribers' elements. This element has the subscriber information. And has the following attributes:

- username—Either Tel-URI or SIP-URI identifying the end station for which services are executed.
- ifcid—Pointer to service execution ID (iFC).

Overlapping LST Entries Not Supported

The best-match function for LST table entries is not supported in this release. You can create distinct PSI and wildcard PSI entries in the LST table. But the pattern of these entries cannot overlap. Currently, the OCCSM does not support recursive searches on LST entries. Therefore, the OCCSM does not identify entry best match.

For example, the two LST entries shown below are overlapping patterns, and must not be in the same LST table:

- sip:alic!.*!@open-ims.lst
- sip:ali!.*!@open-ims.lst

Note that the AOR alice@open-ims matches both entries, with “sip:alic!.*!@open-ims.lst” being the best match. In this software version, the OCCSM uses the first entry it finds instead of the best match.

LST Configuration for Service Execution

To configure the OCCSM to use LSTs for service execution, create a **local-subscriber-table** configuration element that identifies that LST. You then need to set the **subscriber-database-method** configuration to reference that LST configuration so that when messages are received and processed by a **sip-registrar** configuration element, the OCCSM then uses the identified LST for service execution.

Use the following steps:

1. Create your LST and transfer it to your OCCSM.
2. Create a **local-subscriber-table** element. Define your LST by configuring:

- The **name** parameter with a unique **local-subscriber-table** name.
- The **filename** parameter with the LST filename (and path). If you enter a filename without a path, the OCCSM looks in the default LST directory, which is /code/lst. If the LST file is located elsewhere on the OCCSM, you must specify the filename and absolute path. For example /code/path/01302012lst.xml.

3. From the applicable **sip-registrar** element, configure the **subscriber-database-method** to **LOCAL**.
4. From the applicable **sip-registrar** element, configure the **subscriber-database-config** with the **name** of the applicable **local-subscriber-table**.

ACLI Instructions

LST Table

To configure the Oracle Communications Core Session Manager to use an LST:

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

   ```
   ORACLE# configure terminal
   ```

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

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

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

   ```
   ORACLE(session-router)# local-subscriber-table
   ORACLE(local-subscriber-table)#
   ```

   You may now begin configuring the local subscriber table configuration element.

4. **name**—Enter the name of this local subscriber table configuration element that will be referenced from a SIP registrar configuration element.
5. **filename**—Enter the filename that describes this LST XML file. If no path is given, the Oracle Communications Core Session Manager looks in the /code/lst directory. You may provide a complete path if the file is located elsewhere.
6. **secret**—Not used for the PSI and wildcard PSI service execution function.
7. Type **done** when finished.

LST Redundancy for HA Systems

LSTs must be synchronized between redundant nodes to ensure that the standby node contains identical LST files. You can either SFTP the same LST file to both the active and standby node, or you can use the synchronize command. The **synchronize** command is always executed from the active system. It copies the specified file from the active to the

standby node placing the copy in the same file location on the standby node. Use the **synchronize lst** command as follows:

```
ACMESYSTEM# synchronize lst file.xml
```

> **Note:**
>
> The synchronize command does not reload the LST files.

Reloading the LST

After copying a new LST file to the Oracle Communications Core Session Manager (and its standby peer), you can reload this newer file from the ACLI using the **refresh lst** command. For example:

```
ORACLE# refresh lst <local-subscriber-table name>
```

Using the **refresh lst** command selects the LST by name to refresh. Alternatively, saving and activating the configuration will reload the configuration as well and should be used when configuration parameters have also changed.

> **Note:**
>
> In an HA pair of Oracle Communications Core Session Managers, you must independently execute the refresh lst command on both the active and standby systems.

LST File Compression

To save local disk flash space, you can compress the LST file using .gz compression. The resultant file must then have an .xml.gz extension.

5
Transport Layer Security

The Oracle Communications Core Session Manager provides support for Transport Layer Security (TLS) for SIP, which can be used to protect user and network privacy by providing authentication and guaranteeing the integrity for communications between the Oracle Communications Core Session Manager and other devices, including the following:

- An SBC or application server (AS) device in your network infrastructure (intra-network)
- Another Oracle Communications Core Session Manager when you are using a peering application (inter-network) for interior network signaling security

TLS for Signaling Interfaces

The OCCSM supports TLS on wancom management interfaces without a license. This means you can configure TLS security on the network interfaces you use to manage the OCCSM. You must, however, install a TLS license if you want to use TLS on signaling (SIP) interfaces using the **license**, **add** parameter. The TLS license is only required for signaling.

To enable TLS on signaling interfaces, you install a license key using the **system**, **license** configuration element. Refer to the *Getting Started* Chapter in the *S-CZ9.1.0 ACLI Configuration Guide* for instructions on installing and managing licenses. Note that you must install licenses on both devices to enable this feature within an HA deployment.

After you install the license keys, you must reboot the system for them to take effect and for you to see them using the **show features** command.

Supported Encryption

The Oracle Communications Core Session Manager supports the following encryption:

- TLSv1.0, TLSv1.1, TLSv1.2

> **Note:**
>
> Oracle does not support RC4 ciphers.

Diffie-Hellman Key Size

In the context of TLS negotiations on SIP interfaces, the default Diffie-Hellman key size offered by the OCCSM is 1024 bits. The key size is set in the `diffie-hellman-key-size` attribute within the **tls-global** configuration element.

While the key size can be increased, setting the key size to 2048 bits significantly decreases performance.

Suite B and Cipher List Support

The Oracle Communications Core Session Manager (OCCSM) supports full control of selecting the ciphers that you want to use for Transport Layer Security (TLS). The system defaults to DEFAULT for the Cipher List parameter in the TLS Profile configuration. Oracle recommends that you delete ALL and add only the particular ciphers that you want, choosing the most secure ciphers for your deployment.

To support Suite B, the OCCSM certificate-record configuration includes the following parameters:

- Key Algor—Public key algorithm. Supports RSA and ECDSA. Default: RSA Security. You must select ECDSA to support suite B.
- ECDSA Key Size—ECDSA key size. Supports p256 and p384.

Configure the list of ciphers that you want to use from the **cipher-list** element in the **tls-profile** configuration. Press Tab to display the list of supported ciphers. One-by-one, you can add as many ciphers as your deployment requires.

TLS Ciphers

The Oracle Communications Core Session Manager (OCCSM) supports TLS v1, TLSv1.1, and TLSv1.2 ciphers.

The **tls-profile** object contains the **cipher-list** parameter. The ACLI help for **cipher-list** parameter, displayed by appending the parameter with a question mark or by pressing the tab key after the parameter name, displays the list of ciphers that you may specify. For a complete list of supported ciphers, see the Supported TLS Ciphers topic in the Introduction chapter of this *CSM Essentials Guide*.

Minimum Advertised SSL/TLS Version

The sslmin option sets the minimum allowed TLS version. Use this option to mitigate the use of older, more vulnerable versions of TLS.

When the **tls-version** parameter is set to **compatibility** within a **tls-profile** configuration element, the **sslmin** option specifies the lowest TLS version allowed. By default, when **tls-version** is set to **compatibility**, the Oracle Communications Core Session Manager advertises only TLS1.1 and TLS1.2. To advertise TLS1.0 as well, set sslmin to **tls1.0**.

In **security-config**, the **sslmin** option values can be: tls1.0, tls1.1 or tls1.2.

Minimum Advertised SSL/TLS Version Configuration

Configure the option **sslmin** to at least tls1.0 for security purposes.

1. Access the **security-config** configuration element.

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

2. Select the **security-config** object to edit.

```
ORACLE(security-config)#

ORACLE(security-config)#
```

3. **options**—Set the options parameter by typing **options**, a space, a plus sign, the option name **sslmin=** and then one of the valid values. Valid values are:
   - tls1.0
   - tls1.1
   - tls1.2

```
ORACLE(security-config)#options +sslmin=tls1.2
```

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

Signaling Support

The Oracle Communications Core Session Manager ’s TLS functionality supports SIP and SIPS. In addition, the Oracle Communications Core Session Manager can accommodate a mixture of TLS and non-TLS sessions within a realm as because a request for TLS is controlled by the endpoint (TLS UA).

Endpoint Authentication

The Oracle Communications Core Session Manager does not operate as a CA. Instead, the Oracle Communications Core Session Manager ’s TLS implementation assumes that you are using one of the standard CAs for generating certificates:

- Verisign
- Entrust
- Thawte
- free Linux-based CA (for example, openssl)

The Oracle Communications Core Session Manager can generate a certificate request in PKCS10 format and to export it. It can also import CA certificates and a Oracle Communications Core Session Manager certificate in the PKCS7/X509 PEM format.

The Oracle Communications Core Session Manager generates the key pair for the certificate request internally. The private key is stored as a part of the configuration in 3DES encrypted form (with an internal generated password) and the public key is returned to the user along with other information as a part of PKCS10 certificate request.

The Oracle Communications Core Session Manager supports the option of importing CA certificates and marking them as trusted. However, the Oracle Communications Core Session Manager only authenticates client certificates that are issued by the CAs belonging to its trusted list. If you install only a specific vendor's CA certificate on the Oracle Communications Core Session Manager , it authenticates that vendor's endpoints. Whether the certificate is an individual device certificate or a site-to-site certificate does not matter because the Oracle Communications Core Session Manager authenticates the signature/public key of the certificate.

Key Usage Control

You can configure the role of a certificate by setting key usage extensions and extended key usage extensions. Both of these are configured in the certificate record configuration.

Key Usage List

This section defines the values you can use (as a list) in the **key-usage-list** parameter. You can configure the parameter with more than one of the possible values.

Value	Description
digitalSignature (default with keyEncipherment)	Used when the subject public key is used with a digital signature mechanism to support security services other than non-repudiation, certificate signing, or revocation information signing. Digital signature mechanisms are often used for entity authentication and data origin authentication with integrity.
nonRepudiation	Used when the subject public key is used to verify digital signatures that provide a non-repudiation service protecting against the signing entity falsely denying some action, excluding certificate or CRL signing.
keyEncipherment (default with digitalSignature)	Used with the subject public key is used for key transport. (For example, when an RSA key is to be used for key management.)
dataEncipherment	Used with the subject public key is used for enciphering user data other than cryptographic keys.
keyAgreement	Used with the subject public key is used key agreement. (For example, when a Diffie-Hellman key is to be used for a management key.)
encipherOnly	The keyAgreement type must also be set. Used with the subject public key is used only for enciphering data while performing key agreement.
decipherOnly	The keyAgreement type must also be set. Used with the subject public key is used only for deciphering data while performing key agreement.

Extended Key Usage List

This section defines the values you may use in the **extended-key-usage-list** parameter.

Value	Description
serverAuth (default)	Used while the certificate is used for TLS server authentication. In Oracle Communications Core Session Manager access-side deployments, the system typically acts as a TLS server accepting TLS connections. You might use this setting while generating the end-entity-cert.

Value	Description
clientAuth	Used while the certificate is used for TLS client authentication. In Oracle Communications Core Session Manager core-side deployments, the system typically acts as a TLS client initiating TLS connections. You might use this setting while generating the end-entity-cert.

4096-bit RSA Key Support

The Oracle Communications Core Session Manager (OCCSM) supports 4096-bit RSA keys for SIP Transport Layer Security (TLS) on all platforms. The 4096-bit support enables you to import root certificates for SIP communications secured with TLS into the OCCSM.

Use the **key-size** parameter in the certificate-record configuration to set the key size.

TLS Configuration Process

Configuring Transport Layer Security (TLS) on the Oracle Communications Core Session Manager (OCCSM) includes the following steps.

1. Configure certificates. See "Configure Certificates."
2. Configure and apply the TLS profile. See "Configure a TLS Profile."

Certificate Configuration Process

The process for configuring certificates on the Oracle Communications Core Session Manager (OCCSM) includes the following steps:

1. Configure a certificate record on the OCCSM. See "Configure Certificates."
2. Generate a certificate request by the OCCSM. See "Generate a Certificate Request."
3. Import the certificate record into the OCCSM. See "Import a Certificate Using the ACLI" and "Import a Certificate Using SFTP."
4. Reboot the system.

Configure the Certificate Record

The certificate record configuration represents either the end-entity certificate or the CA certificate on the Oracle Communications Core Session Manager (OCCSM). When you use the certificate record for an end-entity certificate, associate a private key with the certificate record configuration by using the ACLI **generate-certificate-request** command. You can import a requested certificate provided by a CA into a certificate record configuration using the ACLI **import-certificate** command.

Do not associate a private key with the certificate record configuration, if it was issued to hold a CA certificate.

> **Note:**
>
> You do not need to create a certificate record when importing a CA certificate or certificate in PKCS #12 format.

1. Access the **certificate-record** configuration element.

```
ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# certificate record
ORACLE(certificate-record)#
```

2. For the Certificate Record configuration, do the following:
   - Name—(Required) Enter the name of this certificate record.
   - Country—Enter the country name abbreviation. For example, CA for Canada. Range: 2 characters.
   - State—Enter the name of the state. For example, Quebec. Range: 1-128 characters.
   - Locality—Enter the name of the locality in the region. For example, Quebec City. Range:1-128 characters.
   - Organization—Enter the name of the organization. For example, Office of Information Technology. 1-64 characters.
   - Unit—Enter the name of the unit in the organization. For example, Global Network Security. 1-64 characters.
   - Common name—Enter the common name for the certificate record. For example, your name. Range: 1-64 characters.
   - Key algor—Set a key algorithm. Valid algorithms: rsa | ecdsa.
   - Digest algor—Set a digest algorithm. Valid values: sha1 | sha256 | sha384.
   - Key size—For the RSA key algorithm, set the RSA key size. Valid key size: 512 | 1024 | 2048 | 4096.
   - ECDSA key size—For the ECDSA key algorithm, set the ECDSA key size. Valid key size: p256 | p384.
   - Alternate name—(Optional) Enter one or more alternative names for the certificate holder.
   - Trusted—Do one of the following:
     - Select to make the certificate trusted. (Default)
     - Deselect to make the certificate un-trusted.
   - Key usage list—Set key the usage extensions you want to use with this certificate record. Multiple values allowed. Default: The combination of **digitalSignature** and **keyEncipherment**. For a list of possible values and their descriptions, see “Key Usage List.”
   - Extended key usage list—Set the extended key usage extensions you want to use with this certificate record. Default: **serverAuth**. For a list of possible values and their descriptions, see “Extended Key Usage List.”

- Options—Set any optional features or parameters that you want.

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

- Create TLS profiles, using your certificate records, to further define the encryption behavior and create the configuration element that you can apply to a SIP interface.

Generating a Certificate Request

Using the ACLI **generate-certificate-request** command allows you to generate a private key and a certificate request in PKCS10 PEM format. You take this step once you have configured a certificate record.

The Oracle Communications Core Session Manager stores the private key that is generated in the certificate record configuration in 3DES encrypted form with in internally generated password. The PKCS10 request is displayed on the screen in PEM (Base64) form.

You use this command for certificate record configurations that hold end-entity certificates. If you have configured the certificate record to hold a CA certificate, then you do not need to generate a certificate request because the CA publishes its certificate in the public domain. You import a CA certificate by using the ACLI **import-certificate** command.

This command sends information to the CA to generate the certificate, but you cannot have Internet connectivity from the Oracle Communications Core Session Manager to the Internet. You can access the internet through a browser such as Internet Explorer if it is available, or you can save the certificate request to a disk and then submit it to the CA.

To run the applicable command, you must use the value you entered in the name parameter of the certificate record configuration. You run the command from main Superuser mode command line:

```
ORACLE# generate-certificate-request acmepacket
Generating Certificate Signing Request. This can take several minutes...
-----BEGIN CERTIFICATE REQUEST-----
MIIDHzCCAoigAwIBAgIIAhMCUACEAHEwDQYJKoZIhvcNAQEFBQAwcDELMAkGA1UE
BhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExETAPBgNVBAcTCFNhbiBKb3NlMQ4w
DAYDVQQKEwVzaXBpdDEpMCcGA1UECxMgU2lwaXQgVGVzdCBDZXJ0aWZpY2F0ZSBB
dXRob3JpdHkwHhcNMDUwNDEzMjEzNzQzWhcNMDgwNDEyMjEzNzQzWjBUMQswCQYD
VQQGEwJVUzELMAkGA1UECBMCTUExEzARBgNVBAcTCkJ1cmxpbmd0b24xFDASBgNV
BAoTC0VuZ2luZWVyaW5nMQ0wCwYDVQQDEwRhY21lMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCXjIeOyFKAUB3rKkKK/+59LT+rlGuW7Lgc1V6+hfTSr0co+ZsQ
bHFUWAA15qXUUBTLJG13QN5VfG96f7gGAbWayfOS9Uymold3JPCUDoGgb2E7m8iu
vtq7gwjSeKNXAw/y7yWy/c04FmUD2U0pZX0CNIR3Mns5OAxQmq0bNYDhawIDAQAB
o4HdMIHaMBEGA1UdEQQKMAiCBnBrdW1hcjAJBgNVHRMEAjAAMB0GA1UdDgQWBBTG
tpodxa6Kmmn04L3Kg62t8BZJHTCBmgYDVR0jBIGSMIGPgBRrRhcU6pR2JYBUbhNU
2qHjVBShtqF0pHIwcDELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWEx
ETAPBgNVBAcTCFNhbiBKb3NlMQ4wDAYDVQQKEwVzaXBpdDEpMCcGA1UECxMgU2lw
aXQgVGVzdCBDZXJ0aWZpY2F0ZSBBdXRob3JpdHmCAQAwDQYJKoZIhvcNAQEFBQAD
gYEAbEs8nUCi+cA2hC/lM49Sitvh8QmpL81KONApsoC4Em24L+DZwz3uInoWjbjJ
QhefcUfteNYkbuMH7LAK0hnDPvW+St4rQGVK6LJhZj7/yeLXmYWIPUY3Ux4OGVrd
2UgV/B2SOqH9Nf+FQ+mNZOlL7EuF4IxSz9/69LuYlXqKsG4=
-----END CERTIFICATE REQUEST-----;
WARNING: Configuration changed, run save-config command.
ORACLE# save-config
Save-config received, processing.
waiting 1200 for request to finish
Request to 'SAVE-CONFIG' has Finished,
```

```
Save complete
Currently active and saved configurations do not match!
To sync & activate, run 'activate-config' or 'reboot-activate'
ORACLE# activate-config
Activate-Config received, processing.
waiting 12000 for request to finish
Add LI flows
LiSysClientMgr::handleNotifyReq
H323 Active Stack Cnt: 0
Request to 'ACTIVATE-CONFIG' has finished
Activate Complete
ORACLE#
```

Import a Certificate Using the ACLI

After the Certificate Authority (CA) generates the certificate, you can import it into the Oracle Communications Core Session Manager (OCCSM) with the **import-certificate** command.

Use the following syntax:

```
ORACLE # import-certificate [try-all|pkcs7|x509] [certificate-record
file-name]
```

1. When you use the **import-certificate** command, you can specify whether you want to use PKCS7 or X509v3 format, or try all. In the command line, you enter the command, the format specification, and the name of the certificate record.

   ```
   ORACLE# import-certificate try-all acme
   ```

   The OCCSM displays the following:

   ```
   Please enter the certificate in the PEM format.
   Terminate the certificate with ";" to exit.......
   -----BEGIN CERTIFICATE-----
   MIIDHzCCAoigAwIBAgIIAhMCUACEAHEwDQYJKoZIhvcNAQEFBQAwcDELMAkGA1UE
   BhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExETAPBgNVBAcTCFNhbiBKb3NlMQ4w
   DAYDVQQKEwVzaXBpdDEpMCcGA1UECxMgU2lwaXQgVGVzdCBDZXJ0aWZpY2F0ZSBB
   dXRob3JpdHkwHhcNMDUwNDEzMjEzNzQzWhcNMDgwNDEyMjEzNzQzWjBUMQswCQYD
   VQQGEwJVUzELMAkGA1UECBMCTUExEzARBgNVBAcTCkJ1cmxpbmd0b24xFDASBgNV
   BAoTC0VuZ2luZWVyaW5nMQ0wCwYDVQQDEwRhY21lMIGfMA0GCSqGSIb3DQEBAQUA
   A4GNADCBiQKBgQCXjIeOyFKAUB3rKkKK/+59LT+rlGuW7Lgc1V6+hfTSr0co+ZsQ
   bHFUWAA15qXUUBTLJG13QN5VfG96f7gGAbWayfOS9Uymold3JPCUDoGgb2E7m8iu
   vtq7gwjSeKNXAw/y7yWy/c04FmUD2U0pZX0CNIR3Mns5OAxQmq0bNYDhawIDAQAB
   o4HdMIHaMBEGA1UdEQQKMAiCBnBrdW1hcjAJBgNVHRMEAjAAMB0GA1UdDgQWBBTG
   tpodxa6Kmmn04L3Kg62t8BZJHTCBmgYDVR0jBIGSMIGPgBRrRhcU6pR2JYBUbhNU
   2qHjVBShtqF0pHIwcDELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWEx
   ETAPBgNVBAcTCFNhbiBKb3NlMQ4wDAYDVQQKEwVzaXBpdDEpMCcGA1UECxMgU2lw
   aXQgVGVzdCBDZXJ0aWZpY2F0ZSBBdXRob3JpdHmCAQAwDQYJKoZIhvcNAQEFBQAD
   gYEAbEs8nUCi+cA2hC/lM49Sitvh8QmpL81KONApsoC4Em24L+DZwz3uInoWjbjJ
   QhefcUfteNYkbuMH7LAK0hnDPvW+St4rQGVK6LJhZj7/yeLXmYWIPUY3Ux4OGVrd
   2UgV/B2SOqH9Nf+FQ+mNZOlL7EuF4IxSz9/69LuYlXqKsG4=
   -----END CERTIFICATE-----;
   ```

```
Certificate imported successfully....
WARNING: Configuration changed, run "save-config" command.
```

2. Save the configuration.

```
ORACLE# save-config
Save-Config received, processing.
waiting 1200 for request to finish
Request to 'SAVE-CONFIG' has Finished,
Save complete
Currently active and saved configurations do not match!
To sync & activate, run 'activate-config' or 'reboot activate'.
```

3. Synchronize and activate the configurations.

```
ORACLE# activate-config
Activate-Config received, processing.
waiting 120000 for request to finish
Add LI Flows
LiSysClientMgr::handleNotifyReq
H323 Active Stack Cnt:  0
Request to 'ACTIVATE-CONFIG' has Finished,
Activate Complete
ORACLE#
```

4. Reboot the system.

Import a Certificate Using SFTP

1. Copy the certificate to the `/opt` directory of the Oracle Communications Core Session Manager using SFTP.
2. Import the certificate with the **import-certificate** command.

   Use the following syntax:

```
import-certificate [try-all|pkcs7|x509] [certificate name] [filename]
```

   Use the value of the **name** parameter from the previously configured **certificate-record** configuration element for the certificate name argument.

```
ORACLE# import-certificate x509 acme certificate.pem
Certificate imported successfully....
WARNING: Configuration changed, run "save-config" command.
ORACLE#
```

3. Save the configuration.

```
ORACLE# save-config
```

4. Activate the configurations.

```
ORACLE# activate-config
```

5. Reboot the system.

PKCS #12 Container Import and Export Capability

The Oracle Communications Core Session Manager (OCCSM) supports Public Key Cryptography Standard (PKCS) #12 for bundling a private key with the associated X.509 public key certificate in a file for archiving, importing, and exporting. The OCCSM does not support bundling all members of the chain of trust.

> **Note:**
>
> The OCCSM only supports PKCS12 files that are bundled with either RSA or ECDSA private keys and their X.509 certificates.

OCCSM customers often need to use keys and certificates stored in the OCCSM for Transport Layer Security (TLS) packet analysis and network troubleshooting, or to share with another OCCSM or other device. The keys and certificates are packaged together and exchanged in the PKCS #12 archive file format.

> **Note:**
>
> The OCCSM supports this functionality only by way of the ACLI.

Export to a PKCS #12 File

You can export a local entity certificate from the CSM/SLRM to a PKCS #12 file by way of the ACLI.

> **Note:**
>
> When prompted for password and passphrase, use the ones that you entered in system-config.

- Run the export-certificate command.

  ```
  export-certificate <pkcs#12> <certificate-record-name> [pkcs-12-
  file-name]
  ```

  where

  - `Certificate-record-name`—the name of the local entity certificate record that you want to export.
  - `pkcs12-file-name`—the name of the target PKCS #12 file. The system creates the export file in the /opt directory. Use either .pfx or .p12 for the file extensions.

  ```
  ORACLE# export-pkcs12 masterca certificate.pfx
  Creating pkcs12 for certificate-record: (masterca)
  ```

```
PKCS12 Certificate(s) exported successfully....
ORACLE#
```

This command is supported only when using the RSA key exchange, not when using the Diffie-Hellman key exchange.

Import a PKCS #12 File

You can import a PKCS #12 key and certificate file that was generated elsewhere into the Oracle Communications Core Session Manager (OCCSM) by way of the ACLI.

Make sure that your PKCS#12 file was generated either with the `-descert` flag or the `-keypbe` and `-certpbe` options. If `rsa.key` is a private key and `cert.crt` is an X.509 certificate, either of the following commands generates a PKCS#12 file.

```
# generate using -descert
openssl pkcs12 -export -in cert.crt -inkey rsa.key -out my_pkcs12.pfx -name
"Test Cert" -descert
# generate using -keypbe and -certpbe options
openssl pkcs12 -export -in cert.crt -inkey rsa.key -out my_pkcs12.pfx -name
"Test Cert" -keypbe PBE-SHA1-3DES -certpbe PBE-SHA1-3DES
```

1. Copy the PKCS#12 file to the /opt directory using SFTP.
2. Run the import-certificate command.

   ```
   import-certificate <pkcs#12> <certificate-record-name> [pkcs-12-file-name]
   ```

   where

   - `certificate-record-name`—must be a new name that does not exist as PKCS #12. This is different from other certificate imports, where the certificate record must already exist in the target destination.
   - `pkcs12-file-name`—the name of the PKCS #12 file that you want to import.

   ```
   ORACLE# import-certificate pkcs12 newKey2 my2_pkcs12.pfx
   The specified certificate-record: (newKey2) does not exist.
   Creating one...
   Enter Import Password:
   Importing ee: newKey2
   Certificate(s) imported successfully....

   ----------------------------------------------
   WARNING:
   Configuration changed, run 'save-config' and
   'activate-config' commands to commit the changes.
   ----------------------------------------------
   ORACLE#
   ```

> **Note:**
>
> 512-bit keys are not supported.

Viewing Certificates

You can view either a brief version or detailed information about the certificates. You can also view certificates in PEM format.

Brief Version

Obtaining the brief version uses this syntax, and will appear like the following example:

```
ORACLE# show security certificates brief acmepacket
certificate-record:acmepacket
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            02:13:02:50:00:84:00:71
        Issuer:
            C=US
            ST=California
            L=San Jose
            O=sipit
            OU=Sipit Test Certificate Authority
        Subject:
            C=US
            ST=MA
            L=Burlington
            O=Engineering
            CN=acme
ORACLE#
```

Detailed Version

Obtaining the detailed version uses this syntax, and will appear like the following example:

```
ORACLE# show security certificates detail acmepacket
certificate-record:acmepacket
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            02:13:02:50:00:84:00:71
        Signature Algorithm: sha1WithRSAEncryption
        Issuer:
            C=US
            ST=California
            L=San Jose
            O=sipit
```

```
            OU=Sipit Test Certificate Authority
        Validity
            Not Before: Apr 13 21:37:43 2005 GMT
            Not After : Apr 12 21:37:43 2008 GMT
        Subject:
            C=US
            ST=MA
            L=Burlington
            O=Engineering
            CN=acme
        X509v3 extensions:
            X509v3 Subject Alternative Name:
                DNS:pkumar
            X509v3 Basic Constraints:
                CA:FALSE
ORACLE#
```

PEM Version

Obtaining the PEM version uses this syntax, and will appear like the following example:

```
ORACLE# show security certificates pem acmepacket
certificate-record: acmepacket
-----BEGIN PKCS7-----
MIIE1AYJKoZIhvcNAQcCoIIExTCCBMECAQExADAPBgkqhkiG9w0BBwGgAgQAoIIE
pTCCBKEwggOJoAMCAQICCQCoP4QO15Fv4TANBgkqhkiG9w0BAQUFADCBhDELMAkG
A1UEBhMCVVMxFjAUBgNVBAgMDU1hc3NhY2h1c2V0dHMxEzARBgNVBAcMCkJ1cmxp
bmd0b24xEzARBgNVBAoMCmFjbWVwYWNrZXQxDTALBgNVBAMMBHJvb3QxJDAiBgkq
hkiG9w0BCQEWFXRlc3RlckBhY21lcGFja2V0LmNvbTAeFw0yMjAzMTExMzI5MTha
Fw0zMjAzMDgxMzI5MThaMIGEMQswCQYDVQQGEwJVUzEWMBQGA1UECAwNTWFzc2Fj
aHVzZXR0czETMBEGA1UEBwwKQnVybGluZ3RvbjETMBEGA1UECgwKYWNtZXBhY2tl
dDENMAsGA1UEAwwEcm9vdDEkMCIGCSqGSIb3DQEJARYVdGVzdGVyQGFjbWVwYWNr
ZXQuY29tMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA5fgY0mKfJk+h
R1R1ZKBrx4wq5Hru59Ve7CIFdHghOx5r3Z+i+Drc5SZR15DqFTqmFd3GMTos3Upx
AVmvng7witqnDB49l7bCKoNAaTD8zUj7z6/rdMK8tLmaXRmN6Dv5lwTt4NQ4BBUV
xzUZb2jaCMVgYRWc9IIvLJ2qx+R/dGTzzBhbuP5Ycjm7i5nFwGpE08utvpM7J5Iu
fwP5MLEY81uH2ZS42mLFXuxo+BYSGGbQiT1va4jpET1x4pf8Hddfzk15wEySX1IY
tkoXAnm/Qu5YK3PdJ4EmADC5GJXhSC16UugrBQPDw5l0OevmjgNHERXYCsBECca7
1BEj9x/z/QIDAQABo4IBEjCCAQ4wHQYDVR0OBBYEFApphd/JurmPxU8Nx/h6gsvs
qddFMIG5BgNVHSMEgbEwga6AFApphd/JurmPxU8Nx/h6gsvsqddFoYGKpIGHMIGE
MQswCQYDVQQGEwJVUzEWMBQGA1UECAwNTWFzc2FjaHVzZXR0czETMBEGA1UEBwwK
QnVybGluZ3RvbjETMBEGA1UECgwKYWNtZXBhY2tldDENMAsGA1UEAwwEcm9vdDEk
MCIGCSqGSIb3DQEJARYVdGVzdGVyQGFjbWVwYWNrZXQuY29tggkAqD+EDteRb+Ew
CwYDVR0PBAQDAgKkMBYGA1UdEQQPMA2CC2FsdE5hbWUuY29tMAwGA1UdEwQFMAMB
Af8wDQYJKoZIhvcNAQEFBQADggEBAIALrwF4yTzG+cfYpDwOGJjONrSrLJ1T9hy4
xtRcz5gI08tEMKwquGhARQfioz9qBfPDxY/SgtOcFGukOmsezPd8OrgJdNeTkrO0
45hu42y+gRz+mogPEP1VmJmXUjDb6TAz69e/zrCzosPlJK2Yt4jlFbNDrUSy2zZl
gy2GUrz25NQ+sqXCp1hhO79cTmumHTT9lDIU9KHbT5X/MSZDv4buDWXMONrR5++z
bGJ/l8K1ATfN7bH0ruNQmHq38q6T1Mt+QvUsCmlq9bxtQH0Yj7CjfnZ+VPkyArMm
SVQ68ik0P/R9FjS3hGc6/gj87hSp0fSRBjdWOE1cKRivpgHBQT4xAA==
-----END PKCS7-----
ORACLE#
```

Configure a TLS Profile

The TLS profile configuration contains the information required to run SIP over TLS.

- Obtain the necessary certificates.
- Confirm that the system displays the Superuser mode.

When the Oracle Communications Core Session Manager (OCCSM) negotiates with TLS, it starts with the highest TLS version and works its way down until it finds a compatible version and cipher that works for the other side.

1. Access the **tls-profile** configuration element.

```
ORACLE# configure terminal
ORACLE(configure)# security
ORACLE(security)# tls-profile
ORACLE(tls-profile)#
```

2. **name**—Enter the name of the TLS profile. Required.
3. **end-entity-certificate**—Enter the name of the entity certification record.
4. **trusted-ca-certificates**—Enter the names of the trusted CA certificate records.
5. **cipher-list**—Use either the default **DEFAULT**, or enter a list of ciphers that you want to support. For a complete list of supported ciphers, see the *Oracle Communications Core Session Manager Release Notes*.
6. **verify-depth**—Specify the maximum depth of the certificate chain to verify. Default: 10. Valid range: 0-10.
7. **mutual-authenticate**—Define whether or not you want the OCCSM to mutually authenticate the client. Valid values: enabled | disabled. Default: disabled.
8. **tls-version**—Enter the TLS version that you want to use with this TLS profile. Valid values are:
   - tlsv12 (default)
   - compatibility — When the OCCSM/SLRM negotiates on TLS, it starts with the highest TLS version and works its way down until it finds a compatible version and cipher that works for the other side.
   - tlsv1
   - tlsv11

   > **Note:**
   >
   > The **sslmin** option in **security-config** specifies the lowest TLS version allowed when **tls-version** is set to **compatibility**. By default, compatibility mode excludes TLS 1.0 unless **sslmin** is set to tlsv10.

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

Applying a TLS Profile

To apply the TLS profile, you need to specify it for the SIP interface with which it will be used. You must take this step from within the SIP interface configuration.

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

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

2. 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)#
   ```

3. Select the existing SIP interface to which you want to apply the TLS profile. If you do not know the same of the profile, press Enter again after you use the select command to see a list of all SIP interfaces. Type in the number corresponding to the SIP interface you want to select, and press Enter. You will then be modifying that SIP interface.

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

4. **Type sip-ports and** Enter. The system prompt changes to let you know that you can begin configuring individual parameters.

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

5. **transport-protocol**—Change the transport protocol to **TLS**.

   ```
   ORACLE(sip-interface)# transport-protocol tls
   ```

6. **tls-profile**—Enter the name of the TLS profile you want applied. This is the same value you enter for the name parameter in the TLS profile configuration. This profile will be applied when the transport protocol is TLS.

   ```
   ORACLE(sip-interface)# tls-profile acmepacket
   ```

7. Save your updated SIP interface configuration.

Notifications for Certificate Expiration

Traps

When a security certificate is installed locally on the Oracle Communications Core Session Manager, you can poll the expiration of the certificate using the **apSecurityCertificateTable**.

You can configure the OCCSM 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-exp-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. The number of minutes between notifications sent is configured in the **security-config** parameter **local-cert-exp-trap-int**.

Alarms

The OCCSM also generates an alarm when the certificate of a **tls-profile** is about to expire or has expired. The value of **local-cert-exp-warn-period** determines the number of days before a certificate expires in which the OCCSM generates a warning alarm.

When the certificate is about to expire:

```
ORACLE# display-alarms
1 alarms to show
ID      Task    Severity        First Occurred          Last Occurred
327731  3386    6       2019-01-29 21:47:55     2019-01-29 21:47:55
Count   Description
1       Certificate: tempCert expiring on Jan 30 20:58:29 2019 GMT,

done
ORACLE#
```

When the certificate has expired:

```
ORACLE# display-alarms
1 alarms to show
ID      Task    Severity        First Occurred          Last Occurred
327730  3386    6       2019-02-01 16:20:45     2019-02-01 16:20:45
Count   Description
1       Certificate: tempCert expired on Jan 30 20:58:29 2019 GMT,

done
ORACLE#
```

Configuring Notifications for Certificate Expiration

To configure the Oracle Communications Core Session Manager to generate traps and alarms 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 trap and alarm.

   A value of 0 disables the trap and alarm.

   ```
   ORACLE(security-config)# local-cert-exp-warn-period 30
   ORACLE(security-config)#
   ```

3. Set the **local-cert-trap-int** parameter to the number of minutes between notifications sent once a certificate has expired. This value is also used for notification interval when in the pre-expiration warning period.

   A value of 0 disables the warning trap and alarm.

   ```
   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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager starts it. If the timer expires, then the Oracle Communications Core Session Manager closes the connection. However, if the endpoint is promoted to the trusted state, then the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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.

Securing Communications Between the OCCSM and SDM with TLS

You can use the Transport Layer Security (TLS) protocol to secure the communications link between the Oracle Communications Core Session Manager (OCCSM) and the Oracle Communications Session Delivery Manager (SDM). Note that the systems use Acme Control Protocol (ACP) for this messaging.

To configure the OCCSM to use TLS for this ACP messaging:

1. Configure a TLS profile. The tls-profile object is located under security, where you add certificates, select cipher lists, and specify the TLS version for each profile.
2. Configure system-config element's `acp-tls-profile` parameter to specify this TLS profile.

The `acp-tls-profile` parameter is empty by default, which means that ACP over TLS is disabled. When ACP over TLS is disabled, the SDM establishes a TCP connection with the OCCSM. When the `acp-tls-profile` parameter specifies a valid TLS profile, the OCCSM negotiates a TLS connection with SDM.

You must reboot OCCSM after configuring ACP over TLS.

6

The Session Load Balancer and Route Manager

Functional Overview

Subscriber-aware Load Balancing and Route Management (SLRM) is a proprietary mechanism within the Oracle Communications Core Session Manager 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 Communications Core Session Managers as if they are a single node. Its load balancing functions are limited to operation with other Oracle Communications Core Session Managers as targets over Diameter. Oracle has developed and maintains a proprietary interface, the Sc interface, to manage load balancing operations with target Oracle Communications Core Session Managers. This interface is documented below.

The SLRM acts as an extension upon I-CSCF operation within the Oracle Communications Core Session Manager. It dynamically discovers and evaluates resource utilization of Oracle Communications Core Session Managers deployed in the core. Having discovered and identified each Oracle Communications Core Session Manager'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 Communications Core Session Manager 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 OCCSM product group, which Oracle develops using the same software, basing the discrete operational functionality on configuration and deployment within an IMS core. The Session Load Balancer and Route Manager (SLRM) is distributed as a special configuration of an OCCSM.

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
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 Communications Core Session Managers 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.

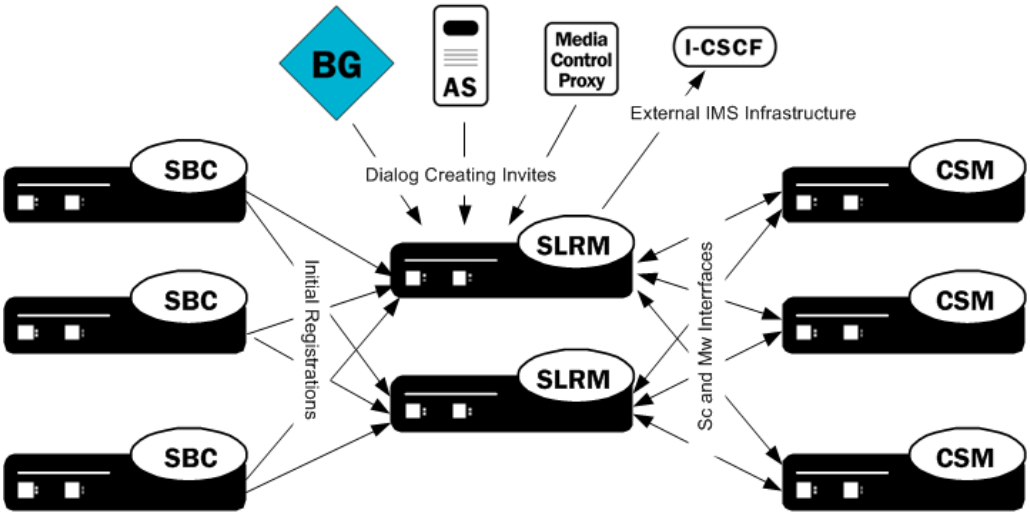
An SLRM typically resides between the network's P-CSCFs (usually Oracle SBCs) and S-CSCFs, load balancing initial registrations from the P-CSCF and INVITEs from a variety of sources. P-CSCFs send registrations. Devices from which the SLRM may receive these INVITEs include:

- 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.



Active-Active High Availability Operation

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 Communications Core Session Manager 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 Communications Core Session Managers 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 Communications Core Session Manager registrars with core names. A core name abstracts a registrar, providing a means of correlating domains serviced by a core between the Oracle Communications Core Session Manager and the SLRM.
  - Cluster membership—All Oracle Communications Core Session Managers 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 Communications Core Session Manager's known number of endpoints includes registered and unregistered users within the registration cache.
  - Maximum endpoint capacity—The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager.
- Maintains connectivity with the SLRM function via watchdog messaging

The user specifies registrars for load balancing on an Oracle Communications Core Session Manager 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 Communications Core Session Managers and SLRMs.

Having determined core membership, the SLRM determines a target Oracle Communications Core Session Manager by evaluating the endpoint capacity information provided by the Oracle Communications Core Session Managers and identifying the best target for the traffic.

Logical Deployment

The key configurations used to establish load balancing operation, includes:

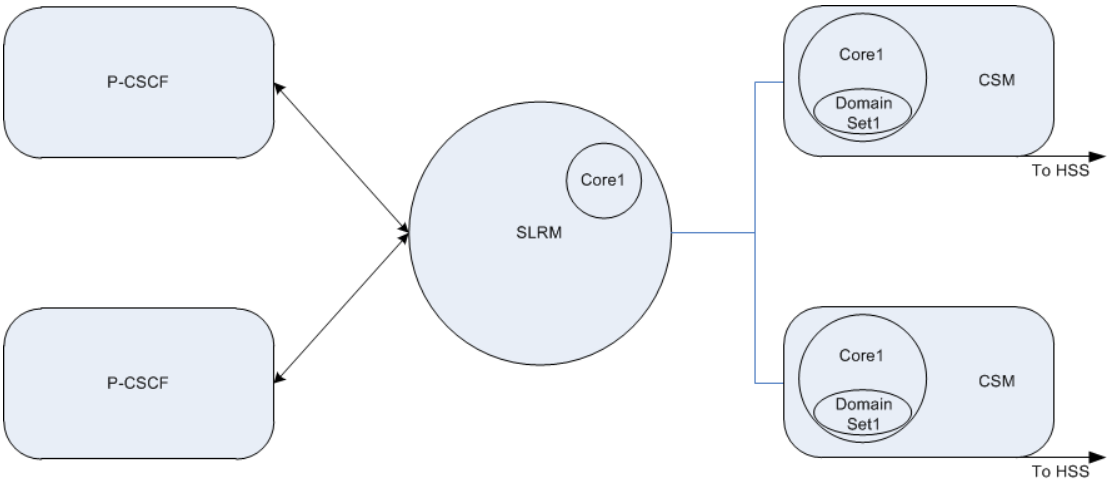
- Core—This required configuration provides a reference between Oracle Communications Core Session Manager registrars and the load balancing configuration. After configuration, each Oracle Communications Core Session Manager advertises its supported cores to the SLRM, which then creates a list of load-balance candidates for those cores.
- Cluster—This configuration refines the list of Oracle Communications Core Session Managers between which the SLRM balances traffic. The user can establish geographical preferences between Oracle SBCs and Oracle Communications Core Session Managers via cluster configuration on both devices. The default cluster ID, null, allows unconfigured Oracle Communications Core Session Managers and third party P-CSCFs to belong to clusters.

Explanations and configuration instructions for cores and clusters are presented below.

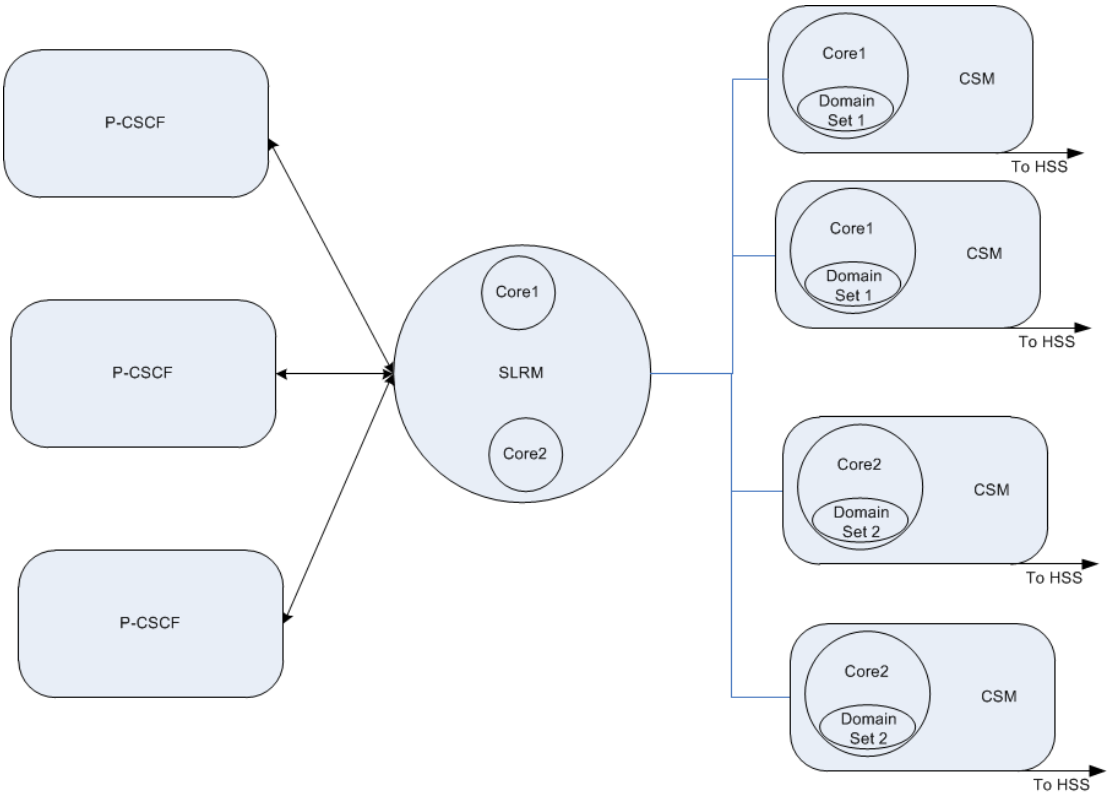
SLRM Core

An SLRM core is a required configuration that establishes a group of Oracle Communications Core Session Managers to which an SLRM load balances registrations and applicable INVITEs. The user configures cores to equate to Oracle Communications Core Session Manager 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 Oracle Communications Core Session Manager 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 Communications Core Session Managers. Based on this advertisement, the SLRM groups Oracle Communications Core Session Managers 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 Communications Core Session Managers for each core. The SLRM load balances registrations from P-CSCFs for Core1 between the Oracle Communications Core Session Managers 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 Communications Core Session Managers.

Cluster Configuration

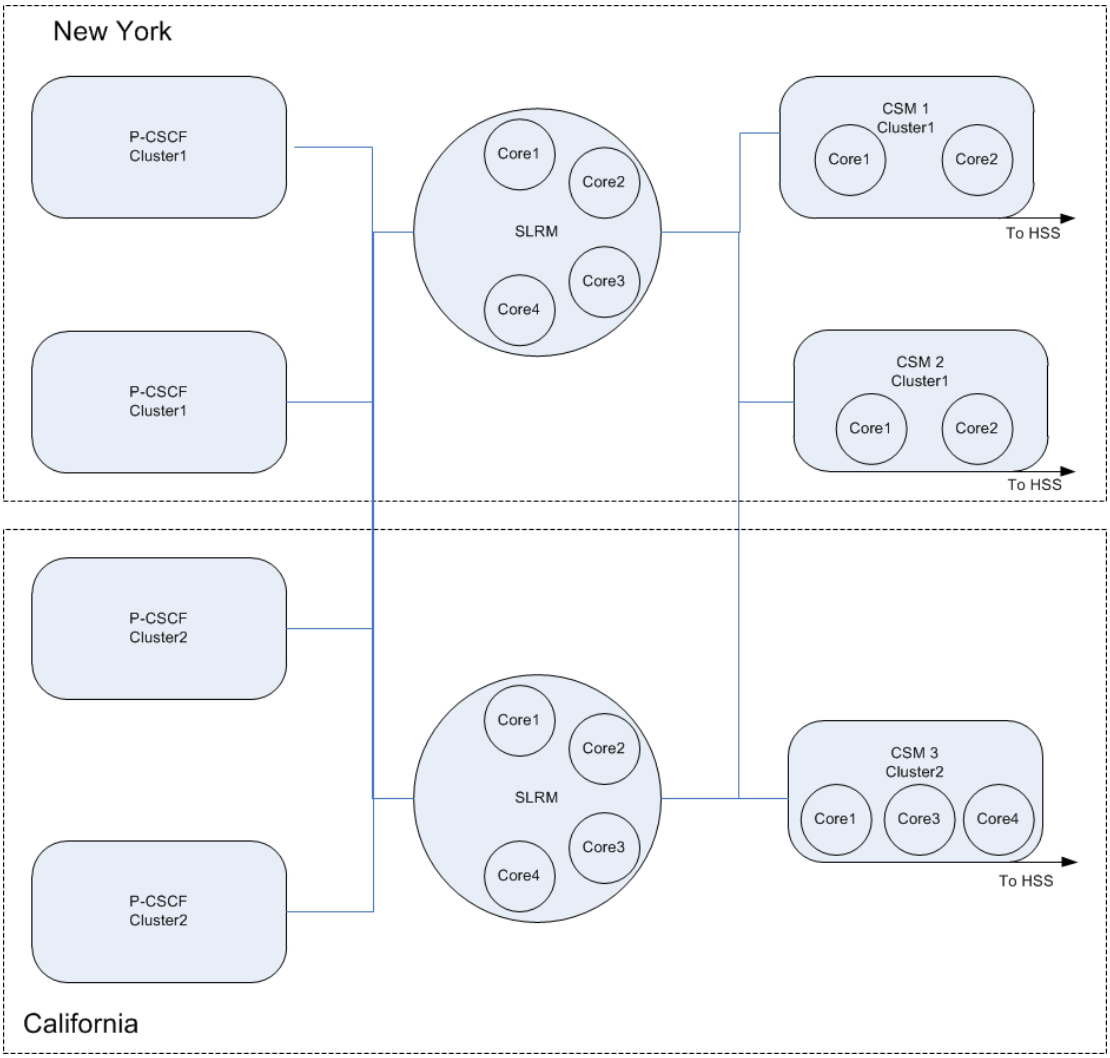
As stated, core configurations ensure that the SLRM does not send traffic to an Oracle Communications Core Session Manager that does not service that core's domain(s). Cluster configuration can refine these constraints by establishing a group of Oracle Communications

Core Session Managers for which core re-balancing is preferred. The SLRM attempts to load balance Oracle Communications Core Session Managers that belong to the same cluster first. If all Oracle Communications Core Session Managers in that cluster are unavailable, the SLRM can choose an Oracle Communications Core Session Manager that services the correct core, but belongs to a different cluster.

Each participating Oracle Communications Core Session Manager must belong to at least one cluster to participate in load balancing processes. To accommodate this requirement, all Oracle Communications Core Session Managers 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 Communications Core Session Managers. The operator can configure these cluster to establish "affinity" between P-CSCFs and Oracle Communications Core Session Managers 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Managers:

- 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 Communications Core Session Managers to load balance new registrations and applicable INVITEs. These pools include Oracle Communications Core Session Manager that service the same cores. The SLRM ranks Oracle Communications

Core Session Managers 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 Communications Core Session Managers. Key to this interaction is the Oracle Communications Core Session Manager specifying cluster membership and registering to service cores at the SLRM. Load balanced pools for a given core include only the Oracle Communications Core Session Managers registered for that core.

Supporting information over the Sc interface provides the details of the Oracle Communications Core Session Manager's registration. To this end, a client-server relationship exists, with the SLRM function acting roughly as server:

- Upon startup, each Oracle Communications Core Session Manager advertises its cluster membership, and subsequently the IMS "cores" it services and its resource utilization. This allows the SLRM function to group Oracle Communications Core Session Managers for load balancing.
- At agreed upon intervals, the Oracle Communications Core Session Manager resends advertisements to confirm or change SLRM-core registration and resource utilization.
- The Oracle Communications Core Session Manager 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 Communications Core Session Manager in the "core" lists and begins to assign users to it.
- Re-Registration — The SLRM refreshes the list of cores within which this Oracle Communications Core Session Manager participates.
- De-Registration — The SLRM removes this Oracle Communications Core Session Manager from the core list from which it is de-registering.

After a Oracle Communications Core Session Manager registers with the SLRM, the SLRM tracks its state. The SLRM only includes devices in the proper state when making load balancing calculations. Oracle Communications Core Session Manager states include:

- In Service — The Oracle Communications Core Session Manager 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 Communications Core Session Managers to which it sends traffic for a given core.

Having established each Oracle Communications Core Session Manager state, the SLRM groups all Oracle Communications Core Session Managers that service a given core into clusters. The SLRM then establishes load balance lists labeled:

- Preferred — The administrator has configured both the target Oracle Communications Core Session Manager and the source P-CSCF within the same cluster.
- Alternative — The target Oracle Communications Core Session Manager and the source P-CSCF are not in the same cluster.

> **Note:**
>
> For single cluster deployments, all Oracle Communications Core Session Managers registered to the SLRM function belong to the default cluster. If all P-CSCFs are in the default cluster, then every Oracle Communications Core Session Manager is "Preferred" for every registration.

Having categorized each Oracle Communications Core Session Manager 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 Communications Core Session Managers — Sorted by free endpoint capacity.
- Preferred, Out of Sync Oracle Communications Core Session Manager — Add to the bottom of the preferred list, sorted by free endpoint capacity.
- Alternative Oracle Communications Core Session Managers — Sort by free endpoint capacity and add to list after all preferred Oracle Communications Core Session Managers.
- Alternative, Out of Sync Oracle Communications Core Session Manager — Add to the bottom of the alternative list, sorted by free endpoint capacity.

There may be multiple Alternative Oracle Communications Core Session Manager 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 Communications Core Session Manager to the SLRM via the Sc interface.

SLRM distributes each message individually based on the criteria above. If a message fails at a Oracle Communications Core Session Manager in the list, SLRM proceeds by sending the message to the next Oracle Communications Core Session Manager in the composite list.

Re-balancing

Re-balancing is the process of taking some number of registered users from a functioning Oracle Communications Core Session Managers and redistributing them between other Oracle Communications Core Session Managers. Re-balancing occurs when manually

invoked by the user from the Oracle Communications Core Session Manager using the **release-users** command.

The Oracle Communications Core Session Manager initiates a Reg-Event process to de-register the users. This process includes the following steps:

1. The Oracle Communications Core Session Manager waits for users to send registration refresh.
2. Upon receipt of the users first registration refresh, the Oracle Communications Core Session Manager sends an Administrative_Deregistration SAR to the HSS.
3. The Oracle Communications Core Session Manager 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 Communications Core Session Manager.
6. The Oracle Communications Core Session Manager de-registers the user.
7. The Oracle Communications Core Session Manager sends a NOTIFY messages to all REGEVENT subscribers indicating the de-registration event has taken place.

Note that the Oracle Communications Core Session Manager 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 Communications Core Session Manager re-starts the process. After completion, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager and the SLRM. This is a custom diameter interface designed to include the following:

- Oracle Communications Core Session Manager registration and deregistration on the SLRM function
- Capabilities negotiation between the Oracle Communications Core Session Manager and the SLRM function
- KPIs that specify Oracle Communications Core Session Manager 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 Communications Core Session Manager. 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager uses this messaging for the following purposes:

- INITIAL — On boot up, the Oracle Communications Core Session Manager sends the request to advertise itself to the SLRM. This includes Oracle Communications Core Session Manager's registration information. This information is updated every 20 seconds.
- REFRESH — The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager cannot handle anymore endpoints.
  - The Oracle Communications Core Session Manager can handle endpoints again.
- TERM — The Oracle Communications Core Session Manager uses this message to terminate its association (de-register) with the SLRM. On receiving this message, the SLRM removes the peer Oracle Communications Core Session Manager and clear all state information.

The Oracle Communications Core Session Manager 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 Communications Core Session Manager uses to populate, refresh and update its participation within SLRM cores. This messaging is used in the following registration scenarios:

- REGISTRATION — The Oracle Communications Core Session Manager 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 Communications Core Session Manager to be timed out.

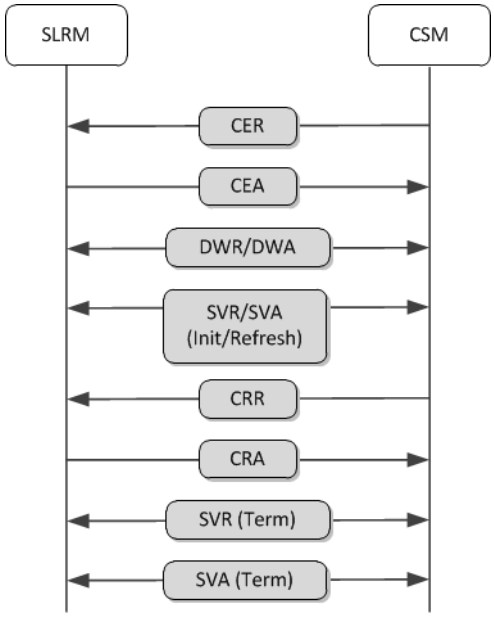
- DE-REGISTRATION — The Oracle Communications Core Session Manager sends this request to de-register a core on SLRM. Upon receipt, the SLRM removes the specified core for the Oracle Communications Core Session Manager and the Oracle Communications Core Session Manager is considered to be no longer serving the core.
- RE-REGISTRATION — The Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Managers. For the purposes of Diameter exchange, the Oracle Communications Core Session Manager 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 Communications Core Session Manager is shown below.



Sc interface messaging procedures, from the perspective of the SLRM function, includes these steps.

1. The SLRM listens on a TCP socket for connection requests from peer Oracle Communications Core Session Managers.

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 Communications Core Session Manager 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 Communications Core Session Managers. The SLRM also responds to DWR's from peer Oracle Communications Core Session Manager's
5. The SLRM responds to SVR requests sent by Oracle Communications Core Session Managers 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 Communications Core Session Manager.
   - On re-registration, the SLRM refreshes and updates the information of the cores for the Oracle Communications Core Session Manager.
   - On de-registration, the SLRM removes the core and stops managing the core for the Oracle Communications Core Session Manager.

SLRM rejects bad message requests with error responses.

Sc interface messaging procedures, from the perspective of the Oracle Communications Core Session Manager, includes these steps.

1. The Oracle Communications Core Session Manager 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 Communications Core Session Manager sends periodic DWR requests on idle connections, to check the availability of the peer SLRM. The Oracle Communications Core Session Manager also responds to DWR's from peer SLRMs.
3. The Oracle Communications Core Session Manager advertises itself and sends periodic refreshes to update current Capacity using SVR/SVA requests.
4. The Oracle Communications Core Session Manager periodically registers with the IMS cores that it is serving using CRR/CRA requests.
5. During shutdown, the Oracle Communications Core Session Manager de-registers with the SLRM using a SVR/SVA (TERM) transaction.

Sc Interface Response Codes

The Oracle Communications Core Session Manager 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 Communications Core Session Manager to the SLRM.
- REFRESH (1) — This indicates refreshes the association of this Oracle Communications Core Session Manager to the SLRM.
- TERM (2) — This indicates the termination of the association (eg, de-registration) of thisOracle Communications Core Session Manager to the SLRM.

Service-Cluster-Id AVP

Uniquely identifies the load balanced cluster to which this Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager. This AVP is included in SVR request.

> **Note:**
>
> Percentage memory is provided by Oracle Communications Core Session Manager 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 Communications Core Session Manager. 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 Communications Core Session Manager can support. This number starts as the user-configured value on the Oracle Communications Core Session Manager, but then is adjusted based on available system resources on the Oracle Communications Core Session Manager.

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 Communications Core Session Manager.
- RE-REGISTRATION (1) — This indicates refresh/update/addition of cores for the Oracle Communications Core Session Manager.
- DE-REGISTRATION (2) — This indicates de-registration of cores for the Oracle Communications Core Session Manager.

Ims-Core AVP

Specifies the IMS core served by the Oracle Communications Core Session Manager, 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager in CRR messages. This grouped AVP contains the following AVPs:

- Srv-info — Service route of IMS core on the target Oracle Communications Core Session Manager. This is a grouped AVP that includes the service info and service route AVPs.
- Ims-Core — IMS core served by Oracle Communications Core Session Manager.

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 Communications Core Session Manager.This grouped AVP has the following information in it.

- Service Info — Designation of this grouped AVP
- Service Route — The route to the target Oracle Communications Core Session Manager

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 Communications Core Session Managers (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 Communications Core Session Managers 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 Communications Core Session Manager sends Sc interface traffic.
5. **port**—Specifies the port on the SLRM interface from which the Oracle Communications Core Session Manager 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 Communications Core Session Manager.
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 Communications Core Session Manager.
5. **forwarding-realm**—Specifies the realm of the SLRM interface from which it sends Sc interface traffic.
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 Communications Core Session Manager to join an SLRM load balanced cluster. Configuration is simplified to allow for an easy and seamless migration.

Configuration required at the Oracle Communications Core Session Manager includes:

- **service-cluster-id**— A parameter within the system-config element that specifies the load balanced cluster to which this Oracle Communications Core Session Manager 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 Communications Core Session Manager.

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 Communications Core Session Manager belongs. The default value is null, which ensures that this Oracle Communications Core Session Manager 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 Communications Core Session Manager starts Sc interface signaling to this target SLRM.
4. **address**—Specifies the IP address of the SLRM to which this Oracle Communications Core Session Manager sends Sc interface traffic.
5. **port**—Specifies the port on the SLRM interface to which the Oracle Communications Core Session Manager 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 Communications Core Session Manager performing the SLRM function.

release-user

This command releases registered users from the Oracle Communications Core Session Manager on which the command is issued. The SLRM re-balances the deployment by registering these users on another Oracle Communications Core Session Manager 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 Communications Core Session Manager, 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 Communications Core Session Managers, 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 Communications Core Session Manager 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: 07ebbdebfdf64a48985bb82fa8b4c595@192.168.101.13
```

SLRM MIB Objects and Traps

The following MIB objects and traps are supported for the Oracle Communications Core Session Manager and its SLRM function. Please consult the *S-Cz7.1.2 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 Communications Core Session Manager's become unavailable for the following reasons:

- The Sc interface goes down. (0)
- The member's registration expires. (1)
- The Oracle Communications Core Session Manager does not respond to SIP requests. (3)

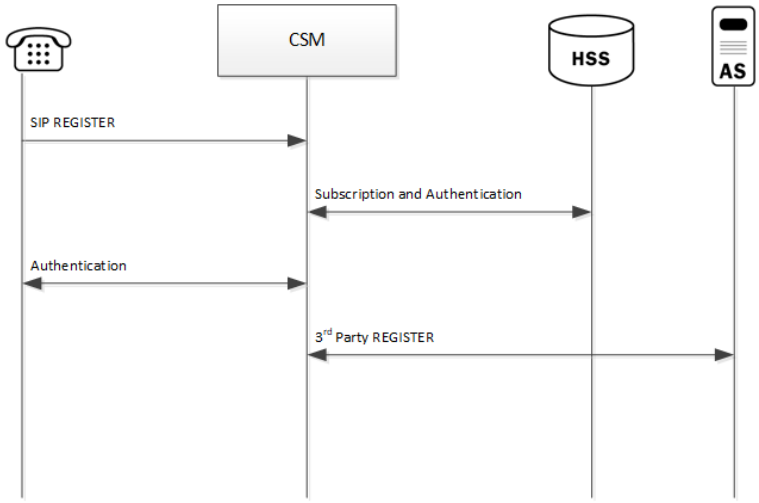
Query apCoreLBReasonCode to determine the reason code. The system sends the clear trap when the Oracle Communications Core Session Manager becomes responsive again.

7

Third Party Registration

Third party registration support on the Oracle Communications Core Session Manager 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 Communications Core Session Manager accepts incoming REGISTER requests from UAs. After the UA has been registered with the Oracle Communications Core Session Manager, the Oracle Communications Core Session Manager sends a third party REGISTER message to a third party server.



The Oracle Communications Core Session Manager supports third party registration via two methods:

- For scenarios in which UAs receive iFCs from the HSS and the Oracle Communications Core Session Manager's default iFC configuration, the Oracle Communications Core Session Manager 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 Communications Core Session Manager and response from the Oracle Communications Core Session Manager to the UA provided to them. The Oracle Communications Core Session Manager 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 Communications Core Session Manager and achieve third party registration support. For these configurations, the Oracle Communications Core Session Manager attempts third party registration to those servers for all UAs that register via the applicable Oracle Communications Core Session Manager registrar.

For both methodologies, you must configure all third party servers as session agents.

Third Party Registrations via iFCs

The Oracle Communications Core Session Manager 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 Communications Core Session Manager. 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager.
- The Oracle Communications Core Session Manager 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 confiuration may not support the optional "IncludeRegisterRequest" and "IncludeRegisterResponse". For these cases, there is a Oracle Communications Core Session Manager 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 Communications Core Session Manager.
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 Communications Core Session Manager third party registration support via iFC as oppsed to via ACLI configuration.

As is true of the method described above, third party registration is generated by the Oracle Communications Core Session Manager on behalf of the user in the To: header of REGISTER request.

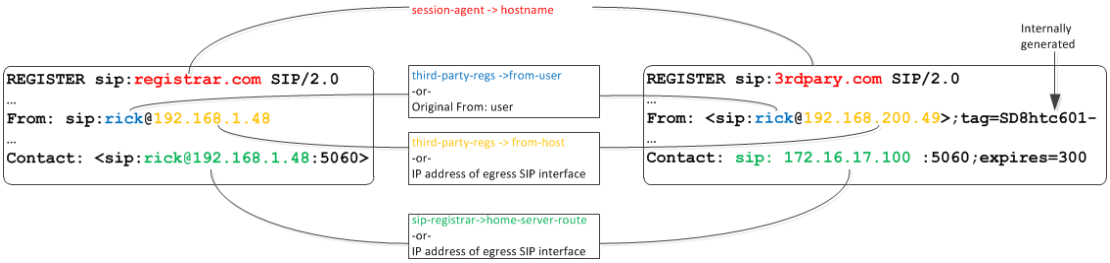
When third party registration is generated by ACLI configuration on the Oracle Communications Core Session Manager, the registration messages are generated as follows:

- 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager adheres to standard SIP session agent retransmission/ timeout procedures. If the third party server is set to out of service, the Oracle Communications Core Session Manager attempts connectivity retry procedures. The retry procedures dictate that the Oracle Communications Core Session Manager 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 ACLI 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 Communications Core Session Manager maintains a queue of outstanding third party registration requests. When the third party server returns to service, the Oracle Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager 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 Communications Core Session Manager refreshes the registration.

If the third party server responds to a REGISTER Request with a 423 (Interval Too Brief) response, the Oracle Communications Core Session Manager 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 Communications Core Session Manager-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 Communications Core Session Manager Registrar functionality. It informs the Oracle Communications Core Session Manager 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 explicity 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 Communications Core Session Manager's egress SIP interface.
8. **retry-interval**—Enter the number of seconds the Oracle Communications Core Session Manager 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.

```
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.

8

References and Debugging

ACLI Configuration Parameters

The following sections describe the Oracle Communications Core Session Manager's configuration parameters that are unique to S-CZ9.1.5.

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 Communications Core Session Manager 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

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-regs 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

ims-restoration—Enables the device to perform standards-based IMS restoration procedures with a compliant HSS deployment.

- Default: disabled
- Values: enabled | disabled

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**, and then **session-router**, and then **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**, and then **session-router**, and then **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 Communications Core Session Manager realm-config name where this home subscriber server exists.

- Default: none

multi-homed-addrs— 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
- 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 ACLI prompt is: **configure terminal**, and then **session-router**, and then **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 Communications Core Session Manager 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 Communications Core Session Manager 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 ACLI prompt is: **configure terminal**, and then **session-router**, and then **third-party-regs**.

local-subscriber-table

Parameters

name—A given name for this local subscriber table element. This name is referenced from the sip-registrar configuration element when the **credential-retrieval-method** is set to **local**.

filename—The filename of local subscriber table that this element references. If no path is provided, the default location is /code/lst.

secret—PSK used for encrypted passwords. This value is not echoed back to the screen upon viewing the configuration element.

Path

The location of this configuration element is: configure terminatl > session-router > local-subscriber-table.

enum-config

Parameters

name—Name for this enum-config to be referenced from within the system.

top-level-domain—The domain extension used to query the ENUM servers for this configuration.

realm-id—The realm-id is used to determine on which network interface to issue an ENUM query.

enum-servers—List of IP address that service the top level domain.

service-type—The ENUM service types you want supported in this ENUM configuration. Possible entries are E2U+sip and sip+E2U (the default), and the types outlines in RFCs 2916 and 3721.

- Default: E2U+sip,sip+E2U

query-method—the ENUM query distribution strategy

- Default: hunt
- Values: hunt | round-robin

timeout—The total time, in seconds, that should elapse before a query sent to a server (and its retransmissions) will timeout.

- Default: 11

cacheInactivityTimer—Enter the time interval, in seconds, after which you want cache entries created by ENUM requests deleted, if inactive for this interval.

- Default: 3600
- Values: 0-999999999

max-response-size—The maximum size in bytes for UDP datagram responses

- Defaults: 512

health-query-number—The phone number for the ENUM server health query; when this parameter is blank the feature is disabled.

health-query-interval—The interval in seconds at which you want to query ENUM server health.

- Default: 0
- Values: 0-65535

failover-to—Name of the enum-config to which you want to failover.

cache-addl-records—Set this parameter to **enabled** to add additional records received in an ENUM query to the local DNS cache.

- Default: enabled
- Values: enabled | disabled

include-source-info—Set this parameter to enabled to send source URI information to the ENUM server with any ENUM queries.

- Default: disabled
- Values: enabled | disabled

ttl—This value ets 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**, and then **session-router**, and then **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.

options—Identifies a set of features that vary depending on the configuration element in which they occur and that are enabled by invocation in the **options** parameter. Set the **options** parameter by typing "options", a Space, and then the option name preceded by a plus sign. If you type the option without the plus sign, you will overwrite any previously configured options. To append the new options to this configuration's options list, you must prefix the new option with a plus sign. Prefixing an option with a minus sign removes it from the list of options.

Path

The location of this configuration element is: **configure terminal**, and then **session-router**, and then **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**, and then **session-router**, and then **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 Communications Core Session Manager 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**, and then **session-router**, and then **session-group**.

Making Personal Data in Messaging Sent to OCOM Anonymous

When you allow people to examine SIP INVITE or SIP MESSAGE messages in the Oracle Communications Operations Monitor (OCOM), you might want to hide certain sensitive information from their view for security and confidentiality reasons. For example, you might want to hide the **SUBJECT** header in the message and in the CPIM body, as well as the MIME content of the CPIM body. Oracle's solution is to provide an option to anonymize such information for display in OCOM.

When you enable the **anonymize-invite** option, the system makes a copy of the inbound SIP INVITE and allows the original to continue on its way. In the copy, the system parses the body of the INVITE and replaces the **SUBJECT** header and MIME content with a hyphen (-). No other message content is affected, and the full functionality of the OCOM remains available. When the troubleshooter views the SIP INVITE message, OCOM displays the anonymized copy of the SIP INVITE.

You can also enable the **anonymize-message** option, which performs the same functions to the SIP MESSAGE, defined in RFC 3428, to support the transfer of Instant Messages. When enabled, this option hides the **SUBJECT** header as well as the CPIM subject and MIME content, replacing them with a hyphen (-) before sending them to OCOM.

The default setting for both options is disabled. Use the options parameter in the comm-monitor configuration to enable them.

Enabling Anonymization of Information Sent to OCOM

When you want to hide certain sensitive information in a SIP **INVITE** message that the Oracle Communications Operations Monitor (OCOM) can display, you can configure the Oracle Communications Core Session Manager (OCCSM) to anonymize the **SUBJECT** header in the message and in the CPIM body, as well as the MIME content of the CPIM body with the **anonymize-invite** option.

> **Note:**
>
> The anonymize-invite option for CommMonitor is not RTC.

You can enable the same functionality for the SIP **MESSAGE** method using the **anonymize-message** option. You can enable both options on the same **comm-monitor**, if desired using the options' plus-sign (+) syntax.

The default setting for these anonymize options is disabled. Use the options parameter in the comm-monitor configuration to enable them.

1. Access the **comm-monitor** configuration element.

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

2. Select the comm-monitor instance that you want to enable for anonymization.

3. Set the **anonymize-invite** option, referring to the syntax below, and press ENTER.

```
ORACLE(comm-monitor)#options + anonymize-invite
```

   To perform the same functionality on the SIP **MESSAGE** method, use the same syntax as above replacing the option with **anonymize-message**, and press ENTER.

4. Save and exit the configuration.

HDR Groups on HSS Data

The Oracle Communications Core Session Manager (OCCSM) provides the same Historical Data Reporting (HDR) function as provided with the Oracle Communications Session Border Controller (OCSBC). HDR is a mechanism that provides for ongoing reporting of system information. HDR descriptions and instruction are provided in the *HDR Resource Guide*.

The OCCSM includes HDR groups that are not documented in the HDR Resource Guide. Those groups include:

- **diam-stats-summary**—diameter statistics summary
- **diam-stats-detail**—diameter statistics detail
- **diam-stats-per-hss**—diameter statistics per HSS

These HDR groups align with the output of the **show home-subscriber-server** ACLI command.

You configure and work with these groups using the same approach and procedures as other HDR groups. The data provided by these groups is presented here so you can identify it and parse the HDR files correctly.

diam-stats-summary

This group presents system-wide diameter message statistics on HSS connectivity. The data is based on the **show home-subscriber-server** ACLI command.

Position	Statistic	Type	Range	Description
1	TimeStamp	Integer	0 - 2147483647	The window of time that the HDR collector used to collect the data in seconds.
2	Client Trans-Period Active	Counter	0 - 2147483647	Total number of client-initiated message transactions (sent to) all HSS servers in the current window of time.

Position	Statistic	Type	Range	Description
3	Client Trans-LifeTime Total	Counter	0 - 2147483647	Total number of client-initiated message transactions (sent to) all HSS servers.
4	Server Trans-Period Active	Counter	0 - 2147483647	Total number of server-initiated message transactions (received from) all HSS servers in the current window of time.
5	Server Trans-LifeTime Total	Counter	0 - 2147483647	Total number of server-initiated message transactions (received from) all HSS servers.
6	Connection Period Active	Counter	0 - 2147483647	Total number of message transactions associated with all client-server connections in the current window of time.
7	Connection LifeTime Total	Counter	0 - 2147483647	Total number of message transactions associated with all client-server connections.

diam-stats-detail

This group consists of statistics pertaining to diameter messaging between the OCCSM and HSS servers. The data is based on the **show home-subscriber-server stats** ACLI command.

Position	Statistic	Type	Range	Description
1	TimeStamp	Integer	0 - 2147483647	The window of time that the HDR collector used to collect the data in seconds.
2	UAR-LifeTime Total	Counter	0 - 2147483647	The total number of User-Authorization-Requests (UAR) sent to all HSS servers.

Position	Statistic	Type	Range	Description
3	UAR Error	String		The most recent error text or the string "NO ERROR".
4	UAR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with UAR messages sent to all HSS servers.
5	SAR-LifeTime Total	Counter	0 - 2147483647	The total number of Server-Assignment-Requests (SAR) sent to all HSS servers.
6	SAR Error	String		The most recent error text or the string "NO ERROR".
7	SAR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with SAR messages sent to all HSS servers.
8	MAR-LifeTime Total	Counter	0 - 2147483647	The total number of Multimedia-Auth-Request (MAR) sent to all HSS servers.
9	MAR Error	String		The most recent error text or the string "NO ERROR".
10	MAR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with MAR messages sent to all HSS servers.
11	LIR-LifeTime Total	Counter	0 - 2147483647	The total number of Location-info-Request (LIR) sent to all HSS servers.
12	LIR Error	String		The most recent error text or the string "NO ERROR".
13	LIR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with LIR messages sent to all HSS servers.

Position	Statistic	Type	Range	Description
14	RTR-LifeTime Total	Counter	0 - 2147483647	The total number of Registration-Termination-Request (RTR) sent to all HSS servers.
15	RTR Error	String		The most recent error text or the string "NO ERROR".
16	RTR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with RTR messages sent to all HSS servers.
17	PPR-LifeTime Total	Counter	0 - 2147483647	The total number of Push-Profile-Requests (PPR) sent to all HSS servers.
18	PPR Error	String		The most recent error text or the string "NO ERROR".
19	PPR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with PPR messages sent to all HSS servers.
20	CER-LifeTime Total	Counter	0 - 2147483647	The total number of Capability-Exchange-Requests (CER) sent to all HSS servers.
21	CER Error	String		The most recent error text or the string "NO ERROR".
22	CER Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with CER messages sent to all HSS servers.
23	DWR-LifeTime Total	Counter	0 - 2147483647	The total number of Device-Watchdog-Requests (DWR) sent to all HSS servers.

Position	Statistic	Type	Range	Description
24	DWR Error	String		The most recent error text or the string "NO ERROR".
25	DWR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with DWR messages sent to all HSS servers.
26	DWR Recv-LifeTime Total	Counter	0 - 2147483647	The total number of Device-Watchdog-Requests (DWR) received by the OCCSM from all HSS servers.
28	DWR Resp Sent	Counter	0 - 2147483647	The total number of responses to Device-Watchdog-Requests (DWR) sent by the OCCSM to all HSS servers.
29	DWR Resp Sent-LifeTime Total	Counter	0 - 2147483647	The total number of responses to Device-Watchdog-Requests (DWR) sent by the OCCSM to all HSS servers.
30	TCP Failures-LifeTime Total	Counter	0 - 2147483647	The total number of TCP failures that have occurred between the OCCSM and the HSS server.
31	Message Recv - LifeTime Total	Counter	0 - 2147483647	The total numbers of diameter messages received by the OCCSM from all HSS servers.
32	Message Trans-LifeTime Total	Counter	0 - 2147483647	The total numbers of diameter messages sent by the OCCSM to all HSS servers.

diam-stats-per-hss

Consists of statistics pertaining to diameter messaging between the OCCSM and individual HSS servers. The data is based on the **show home-subscriber-server stats <hss-name>** ACLI command.. These statistics are per HSS and are repeated

for each configured HSS. The OCCSM repeats all rows for each HSS. The TimeStamp is not repeated.

Position	Statistic	Type	Range	Description
1	TimeStamp	Integer	0 - 2147483647	The window of time that the HDR collector used to collect the data in seconds.
2	HSS Name	String	NA	Name of the Home Subscriber Server (HSS) server.
3	UAR-LifeTime Total	Counter	0 - 2147483647	The total number of User-Authorization-Requests (UAR) sent to this HSS.
4	UAR Error	String	NA	The most recent error text or the string "NO ERROR".
5	UAR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with UAR messages sent to this HSS.
6	SAR-LifeTime Total	Counter	0 - 2147483647	The total number of Server-Assignment-Request (SAR) sent to this HSS.
7	SAR Error	String	N/A	The most recent error text or the string "NO ERROR".
8	SAR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with SAR messages sent to this HSS.
9	MAR-LifeTime Total	Counter	0 - 2147483647	The total number of Multimedia-Auth-Request (MAR) sent to this HSS.
10	MAR Error	String	N/A	The most recent error text or the string "NO ERROR".
11	MAR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with MAR messages sent to this HSS.

Position	Statistic	Type	Range	Description
12	LIR-LifeTime Total	Counter	0 - 2147483647	The total number of Location-info-Request (LIR) sent to this HSS.
13	LIR Error	String	N/A	The most recent error text or the string "NO ERROR".
14	LIR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with LIR messages sent to this HSS.
15	RTR-LifeTime Total	Counter	0 - 2147483647	The total number of Registration-Termination-Requests (RTR) sent to this HSS.
16	RTR Error	String	N/A	The most recent error text or the string "NO ERROR".
17	RTR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with RTR messages sent to this HSS.
18	PPR-LifeTime Total	Counter	0 - 2147483647	The total number of Push-Profile-Requests (PPR) sent to this HSS.
19	PPR Error	String	N/A	The most recent error text or the string "NO ERROR".
20	PPR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with PPR messages sent to this HSS.
21	CER-LifeTime Total	Counter	0 - 2147483647	The total number of Capability-Exchange-Requests (CER) sent to this HSS.
22	CER Error	String	N/A	The most recent error text or the string "NO ERROR".
23	CER Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with CER messages sent to this HSS.

Position	Statistic	Type	Range	Description
24	DWR-LifeTime Total	Counter	0 - 2147483647	The total number of Device-Watchdog-Requests (DWR) sent to this HSS.
25	DWR Error	String	N/A	The most recent error text or the string "NO ERROR".
26	DWR Error-LifeTime Total	Counter	0 - 2147483647	The total number of errors associated with DWR messages sent to this HSS.
27	DWR Recv-LifeTime Total	Counter	0 - 2147483647	The total number of Device-Watchdog-Requests (DWR) received by the OCCSM.
28	DWR Resp SENT	String	N/A	The total number of responses to Device-Watchdog-Requests (DWR) sent by the OCCSM.
29	DWR Resp Sent-LifeTime Total	Counter	0 - 2147483647	The total number of responses to Device-Watchdog-Requests (DWR) sent by the OCCSM.
30	TCP Failures-LifeTime Total	Counter	0 - 2147483647	The total number of TCP failures that have occurred between the OCCSM and the HSS server.
31	Message Recv - LifeTime Total	Counter	0 - 2147483647	The total numbers of diameter messages received by the OCCSM from the HSS server.
32	Message Trans-LifeTime Total	Counter	0 - 2147483647	The total numbers of diameter messages sent by the OCCSM to the HSS server.

SNMP MIBs and Traps

The following MIBs and traps are supported for the Oracle Communications Core Session Manager. Please consult the Oracle Communications S-CX6.3.0 MIB Reference Guide for more SNMP information.

OCCSM 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 Communications Core Session Manager 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: 07ebbdebfdf64a48985bb82fa8b4c595@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 --------   --------- Client --------
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 --------   --------- Client --------
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 --------   --------- Client
--------
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 --------   --------- Client
--------
Message/Event       Recent      Total  PerMax   Recent      Total
PerMax
                    ------  ---------  ------   ------  ---------
------
NOTIFY Requests          0          0       0        2
2       2
Retransmissions          0          0       0       10         10
10
200 OK                   0          0       0        1
1       1
Transaction Timeouts     -          -       -        0
0       0
Locally Throttled        -          -       -        0
0       0
Avg Latency=0.000 for 0
Max Latency=0.000
```

The ACLI **show registration sipd** command includes an argument that provides information about a specific user's registration(s), as shown below.

```
ORACLE# show registration sipd by-user ra1 detailed
User: sip:ra1@apkt.com
  Registered at:  2013-06-05-19:23:40    Surrogate User: false
  Contact Information:
    Contact:
      Name: sip:ra1@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:ra1@192.168.13.1:5060
      Realm: net192
      IP: 192.168.13.1:5060
    SD Info:
      Contact: sip:ra1-1cdstqjt90hve@172.16.101.62:5060
      Realm: net172
    Call-ID: 1-28361@192.168.13.1
  Associated URI(s):
    URI: sip:ra1@apkt.com
       Filter Criteria:
         Priority: 0
           Filter: None specified
           Application Server: sip:appserv@apkt.com
  Reg Event Subscriptions Terminated locally:
    Number of Subscriptions: 1
```

Subscriber: appserv<sip:appserv@apkt.com>;tag=1 state=active exp=600114

show registration

The show registration command displays cumulative statistics on all current registrations.

```
ORACLE# show registration
15:35:43-177
SIP Registrations        -- Period -- ----- Lifetime ------
                 Active  High   Total Total  PerMax    High
User Entries          0     0       0     0       0       0
Local Contacts        0     0       0     0       0       0
Via Entries           0     0       0     0       0       0
AURI Entries          0     0       0     0       0       0
Free Map Ports        0     0       0     0       0       0
Used Map Ports        0     0       0     0       0       0
Forwards              -     -       0     0       0       0
Refreshes             -     -       0     0       0       0
Rejects               -     -       0     0       0       0
Timeouts              -     -       0     0       0       0
Fwd Postponed         -     -       0     0       0       0
Fwd Rejected          -     -       0     0       0       0
Refr Extension        0     0       0     0       0       0
Refresh Extended      -     -       0     0       0       0
ContactsPerAor Reject -     -       0     0       0       0
Surrogate Regs        0     0       0     0       0       0
Surrogate Sent        -     -       0     0       0       0
Surrogate Reject      -     -       0     0       0       0
Surrogate Timeout     -     -       0     0       0       0
```

```
HNT Entries           0     0       0     0       0       0
Non-HNT Entries       0     0       0     0       0       0
Database Regs         0     0       0     0       0       0
DDNS Entries          0     0       0     0       0       0
CX Entries            0     0       0     0       0       0
LocalDB Entries       0     0       0     0       0       0
Unreg Users           0     0       0     0       0       0
```

You can extend upon the show registration command by adding the sipd by-user <username> detail arguments. The resulting output reflects user registration information including downloaded IFCs. For example:

```
ORACLE# show registration sipd by-user +19999092907 d
Registration Cache (Detailed View)    MON JUN 25 2012  13:47:46
User: sip:+19999092907@mobile.com
  Registered at: 2012-06-25-13:43:50     Surrogate User: false
  Contact Information:
    Contact:
      Name: sip:+19999092907@mobile.com
      Valid: true
      Challenged: false
      Registered at: 2012-06-25-13:43:50
      Last Registered at: 2012-06-25-13:47:30
      Expire: 48
      Local expire: 13
      Registrar IP: 0.0.0.0
      Transport: UDP
      Secure: false
      Local IP: 155.212.214.175:5060
      User Agent Info:
        Contact: sip:
+19999092907@50.76.51.62:5762;transport=udp;acme_nat=+19999092907+50.76
.51.62@10.1.10.20:5762
        Realm: access
        IP: 50.76.51.62:5762
      SD Info:
        Contact: sip:+19999092907-rb8tulsbv3u72@108.108.108.108:5060
        Realm: core
      Call-ID: H_yvkgTAAA@10.1.10.20
    Associated URI(s):
      URI: sip:+19999092907@mobile.com
  Filter Criteria:
    Priority: 0
      Filter: ((case == 'Originating Registered') and (method ==
INVITE) and ('Accept-Contact'=='+g.app2app')) or
              ((case == 'Originating Registered') and (method ==
INVITE) and ('Contact'=='+g.app2app')) or
              ((case == 'Originating Registered') and (method ==
INVITE) and ('P-Message-Auth'=='.*')) or
              ((case == 'Originating Registered') and (method ==
INVITE) and ('P-Application-ID'=='.*'))
      Application Server: sip:pza.mobile.com:5280
Reg Event Subscriptions Received by Registrar:
Number of Subscriptions : 2
```

```
Subscriber: sip:appserv@192.168.13.1:5060; state=active; exp=59978
Subscriber: sip:pcscf@192.168.13.1:5060; state=active; exp=978
```

show home-subscriber-server

The show home-subscriber-server command displays cumulative statistics on all currently configured HSS servers.

show home-subscriber-server [stats <hss-name>| group group-name]

This command allows you to gather a set of information commonly requested by the Oracle TAC when troubleshooting customers.

The show home-subscriber-server command with no arguments displays the status of each HSS as well as the number of transactions and connections per HSS. For example:

```
ORACLE# show home-subscriber-server
Name                 Local-Address        Server-Address       Status
hss1                 192.168.207.21:45463 192.168.200.232:3872 Up
----------------------------------------
18:53:25-105
HSS Status                 -- Period -- -------- Lifetime --------
                  Active    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 describes the supported states.

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 Communications Core Session Manager has sent a CER and is waiting for a CEA response.
Inactive	The Oracle Communications Core Session Manager has sent a CER but has not received a CEA response.
Down	The Oracle Communications Core Session Manager is not attempting to establish a connection with the HSS.

Oracle Communications Core Session Manager 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                  -- Period -- -------- Lifetime --------
                  Active    High   Total      Total  PerMax    High
Client Trans           1       1       5      12157       8       1
Server Trans           0       0       0          7       2       1
Connections            1       1       0         53       2       1
                                          ---- Lifetime ----
                                  Recent      Total  PerMax
UAR                                    0          3       1
   SUBSEQ_REG (2002)                   0          3       1
SAR                                    0          6       3
   SUCCESS (2001)                      0          6       3
MAR                                    0          4       2
   SUCCESS (2001)                      0          4       2
LIR                                    0          1       1
   SUCCESS (2001)                      0          1       1
RTR                                    0          1       1
   SUCCESS (2001)                      0          1       1
PPR                                    0          1       1
   SUCCESS (2001)                      0          1       1
CER                                    0         55       3
   SUCCESS (2001)                      0         53       2
DWR                                    5      12088       5
   SUCCESS (2001)                      4      12041       5
   ERR_TIMEOUT                         0         46       1
DWR Recv                               0          5       2
   SUCCESS (2001)                      0          5       2
TCP Failures                           0        267       6
```

By entering the name of a specific HSS as an argument, the ACLI displays all HSS data for that server only. For example:

```
ACMESYSTEM# show home-subscriber-server stats hss1
```

The show home-subscriber-server command with the group argument displays the number of transactions and connections per the HSS group you specify in the command. For example:

```
ORACLE# show home-subscriber-server group hss-group1
display grp hss-group1
HSS Status                  -- Period -- -------- Lifetime --------
                  Active    High   Total      Total  PerMax    High
Client Trans           0       0       0          0       0       0
Server Trans           0       0       0          0       0       0
Sockets                0       0       0          0       0       0
Connections            0       0       0          0       0       0
                                  ---- Lifetime ----
                     Recent      Total  PerMax
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
sk3                192.168.19.1:8889            Up
12:56:41-184
HTTP Status                 -- Period -- -------- Lifetime --------
                  Active    High   Total      Total  PerMax    High
Client Trans           0       0       0          0       0       0
Server Trans           0       0       0          0       0       0
Sockets                0       0       0          0       0       0
Connections            0       0       0          0       0       0
```

You can extend upon this command to get detailed global statistics by adding the stats argument to the end of this command.

```
ORACLE# show http-server stats
Name               Server-Address               Status
sk                 host.httpsrv.com             Up
sk1                192.168.19.1:8886            Up
sk2                192.168.19.1:8887            Up
sk3                192.168.19.1:8889            Up
HTTP Status                 -- Period -- -------- Lifetime --------
                  Active    High   Total      Total  PerMax    High
Client Trans           0       0       0          0       0       0
Server Trans           0       0       0          0       0       0
Sockets                1       1       1          1       1       1
Connections            1       1       1          1       1       1
                          ---- Lifetime ----
                     Recent      Total  PerMax
Sent Requests             0          0       0
Sent Req Accepted         0          0       0
Sent Req Rejected         0          0       0
Sent Req Expired          0          0       0
HTTP Errors               0          0       0
```

You can limit this output to a single server by appending the command with the name of that server.

```
ORACLE# show http-server stats http-server1
Name = http-server1
----------------------------------------
Server-Address            Status
192.168.19.1:8886            Up
----------------------------------------
12:56:41-184
HTTP Status                 -- Period -- -------- Lifetime --------
                  Active    High   Total      Total  PerMax    High
Client Trans           0       0       0          0       0       0
Server Trans           0       0       0          0       0       0
Sockets                0       0       0          0       0       0
Connections            0       0       0          0       0       0
---- Lifetime ----
                     Recent      Total  PerMax
Sent Requests             0          0       0
Sent Req Accepted         0          0       0
Sent Req Rejected         0          0       0
Sent Req Expired          0          0       0
HTTP Errors               0          0       0
```

Verify Config

The Oracle Communications Core Session Manager performs application specific verification checks when you save a config with the save-config ACLI command. These checks are in addition to baseline Oracle Communications Core Session Manager 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 Communications Core Session Manager limits resource utilization to maintain operational stability. Resources managed this way include:

- CPU
- Memory (heap)

CPU Overload Protection

CPU overload protection on the Oracle Communications Core Session Manager is system-oriented in terms of defining the percent utilization that triggers an action. Actions are application-specific.

For the Oracle Communications Core Session Manager application, if the CPU usage exceeds the configured setting, the system sends a 5xx error in response to any initial dialog request or standalone transactions. The Oracle Communications Core Session Manager continues to accept registration refreshes and new transactions within a dialog.

> **Note:**
>
> An Oracle CSM configured to operation as an SLRM rejects all messages when CPU utilization exceeds this threshold.

By default the CPU utilization rate is 80%. This value can be changed by the following ACLI command sequence.

```
ORACLE# configure terminal
ORACLE(configure)# session-router
ORACLE(session-router)# sip-config
ORACLE(sip-config)# +options load-limit="70"
ORACLE(sip-config)# done
```

Heap Utilization

The Oracle Communications Core Session Manager limits memory utilization to maintain operational stability, as follows:

- When heap utilization exceeds the default (75%) or configured memory utilization threshold, the Oracle Communications Core Session Manager no longer accepts new registrations. The Oracle Communications Core Session Manager replies to these messages with 5xx messages. The Oracle Communications Core Session Manager continues to accept registration refreshes, in-dialog calls and subscriptions.
- When heap utilization exceeds its default (90%) or configured threshold, the Oracle Communications Core Session Manager 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 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
```

A

Oracle Sc Interface Support

The Oracle Communications Core Session Manager 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 Communications Core Session Manager.

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 Communications Core Session Manager-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
[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	

AVP	Number	Reference	Type	Grouped
{ 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 Communications Core Session Manager to provide service for the ims-core specified as a member of that core's load balanced Oracle Communications Core Session Managers.

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	Unsigned32	
{ Destination-Realm }				
{ Core-info }	4008	Base	Grouped	ims-core service-info

CRA Message Format

The core registration answer provides the Oracle Communications Core Session Manager 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	
{ 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 Communications Core Session Managers 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 Communications Core Session Manager.

AVP	Number	Reference	Type
Service info ::= <AVP header>	4015	Oracle	String
[service-route]	4011	Oracle	String

B

CSM and SLRM Base Configuration Elements

CSM Base Configuration Elements

This section provides configuration samples of the elements used for minimal Oracle CSM operation.

Consider the following configuration settings for a base CSM:

- The SIP Config must be enabled

```
sip-config
        state                                   enabled
```

- You must have a default gateway in your system-config

```
system-config
        default-gateway                          10.0.0.1
```

- You must have a core physical interface

```
phy-interface
        name                                    s1p0
        operation-type                          Media
        port                                    1
        slot                                    0
```

- You must have a core network interface

```
network-interface
        name                                    s1p0
        sub-port-id                             0
        ip-address                              192.170.2.100
        netmask                                 255.255.255.0
        gateway                                 192.170.2.1
```

- You must have a core realm

```
realm-config
        identifier                              core1
        addr-prefix                             0.0.0.0
        network-interfaces                      s1p0:0
```

- 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 section provides configuration samples of elements used for minimal Oracle SLRM operation.

SLRM Configuration

Consider the following configuration settings for a base SLRM:

- Set the component type to SLRM

```
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]?:
```

- The SIP Config must be enabled

```
sip-config
        state                                   enabled
```

- You must have a default gateway in your system-config

```
system-config
        default-gateway                          10.0.0.1
```

- You must have a core physical interface

```
phy-interface
        name                                    s1p0
        operation-type                          Media
        port                                    1
        slot                                    0
```

- You must have a core network interface

```
network-interface
        name                                    s1p0
        sub-port-id                             0
        ip-address                              192.170.2.100
        netmask                                 255.255.255.0
        gateway                                 192.170.2.1
```

- You must have a core realm

```
realm-config
        identifier                              core1
```

```
        addr-prefix                             0.0.0.0
        network-interfaces                      s0p1:0
```

- 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
```

- 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)

Consider the following configuration settings for a base CSM that uses an 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
```

C

Caveats and Known Issues

This chapter lists the caveats and known issues for this release. Oracle updates this Release Notes document to distribute issue status changes. Check the latest revisions of this document to stay informed about these issues.

Known Issues

This table lists the Oracle Communications Core Session Manager (OCCSM) known issues in version S-Cz9.1.5. You can reference known issues by Service Request number and you can identify the issue, any workaround, when the issue was found, and when it was fixed using this table. Issues not carried forward in this table from previous Release Notes are not relevant to this release. You can review delivery information, including defect fixes in this release's Build Notes.

ID	Description	Severity	Found In
30026648	The OCCSM High Availability function does not replicate sessions. This issue is based on a change, per customer enhancement request, to eliminate B2BUA proxy mode as a valid configuration for the OCCSM. The user can recognize this issue by observing that, although the sessions are not backed up, the standby still routes pertinent BYEs, CANCELs and other messages after switchover using backed up request URIs.	2	S-Cz7.3.5

ID	Description	Severity	Found In
N/A	Instead of routing a message via local policy, the OCCSM incorrectly issues an LIR when the following two conditions exist simultaneously: • The OCCSM is not configured with the **e164-primary-config** and **e164-secondary-config** options, and • The OCCSM receives a request with a tel-URI or a sip-URI with the user=phone parameter. Note that the OCCSM sends the request via local-policy if the LIA for a tel-URI or sip-URI with user=phone returns 5001 DIAMETER_ERROR_USER_UNKNOWN. For all other errors in the LIA, the OCCSM returns an error.	N/A	N/A
N/A	You must change the **sip-config**'s **registration-cache-limit** from the default of zero for an OCCSM to function with an SLRM. Oracle recommends you set this value to 500000.	N/A	N/A
N/A	When the user disables the lb-cfg on an OCCSM, breaks the SC link to the SLRM. However, instead of properly seeing the affected OCCSM as out-of-service, the SLRM sees the OCCSM as out-of-sync and may continue to forward new REGISTERs.	N/A	N/A

Known Issues Inherited from the S-CZ9.1.0 SBC

Refer to the Known Issues in the S-CZ9.1.0 OCSBC Release Notes to complete your review of issues in this release. Issues within the OCSBC, especially including

applicable VNF platform and applicable application issues apply across the S-CZ9.1.x product versions, including the OCCSM.

Resolved Known Issues

The following table provides a list of previous Known Issues that are now resolved.

ID	Description	Severity	Found In	Fixed In
33003011	Any configuration change to a **sip-registrar**, or **lb-core-config** in the case of SLRM, that is also configured with a **servers-capabilities-list** causes the associated **servers-capabilities-table** to load improperly. This is because the table needs to be activated after the **sip-registrar** (or **lb-core-config**). Workaround: After activating your **sip-registrar** (or **lb-core-config**) change, create or modify a dummy **servers-capabilities** sub-element in the applicable **servers-capabilities-table**. Then save and activate your configuration. This allows the OCCSM (or the SLRM) to load the servers-capabilities database correctly.	3	S-Cz8.2.5	S-Cz9.1.5

Caveats and Limitations

The following information lists and describes the caveats and limitations for this release. Oracle updates this Release Notes document to distribute issue status changes. Check the latest revisions of this document to stay informed about these issues.

OCCSM Caveats

This section list caveats related to Version S-CZ9.1.5 of the Oracle Communications Core Session Manager (OCCSM).

- Do not load configurations from sibling products, the Oracle SBC for example, on the OCCSM. Those configurations are incompatible with the OCCSM, causing incorrect operation. Users should configure the OCCSM from scratch or use another valid OCCSM configuration.
- Configuration elements specific to the SLRM, including lb-interface are not compatible with OCCSM configuration elements. The **set-component-type** command provides a warning indicating that the user must delete any prior configuration and create the new component type configuration from scratch to avoid potential configuration conflicts. Note the sample output below.

  –
  ```
  SCZ715_64# 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]?:
  ```

- Multi-stage routing does not work for S-CSCF routing functions.
- The OCCSM does not support 'netboot' when operating on the Oracle VM platform. The user must, instead, upload new images to the OCCSM and boot from them locally.
- By default, storage is not persistent across a reboot of a OCCSM virtual machine. You must create persistent storage space for log and dump file data.
  – Issue - Generic virtual machine installation documentation may not include the requirement to run the command format hard-disk during virtual machine installation.
  – Resolution - Run the command format hard-disk to create a persistent partition for your /opt directory, within which you can store data needed after a reboot. Perform this procedure the FIRST time you start your OCCSM.
- The OCCSM accepts only the first message received from an application server in response to messages from the OCCSM that included an ODI. If it receives subsequent messages from that application server, the OCCSM drops the reused ODI and processes the message as if they were received without an ODI.
  – Resolution - Do not configure an AS to fork responses to the OCCSM that include an ODI originally provided by the OCCSM.
- Geo-redundancy is currently not supported by the OCCSM.
- The OCCSM introduces a **system-config** parameter named **cgroup-enable**. This parameter must not be enabled. After starting up at your site, check this parameter and make sure it is disabled.
  For example. In S-Cz8.2.5p2, this parameter is enabled by default. But in S-Cz8.2.5m1, this parameter is disabled by default.

  Contact Oracle Support before you consider changing this configuration.

> **Note:**
>
> Always ensure the **cgroup-enable** parameter is disabled before an upgrade.

Caveats Inherited from the S-CZ9.1.0 SBC

Refer to the Caveats in the S-CZ9.1.0 OCSBC Release Notes to complete your review of issues in this release. Issues within the OCSBC, especially including applicable VNF platform and applicable application issues apply across the S-CZ9.1.x product versions, including the OCCSM.