

CSCF Configuration Management

Call Session Control Function

USER GUIDE

Copyright

© Ericsson AB 2016–2018. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

Trademark List

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



Contents

1	Introduction	1
2	Basic Concepts	3
2.1	CSCF and IMS	3
2.2	Configuration Methods	4
2.3	Authentication and User Management	5
3	Operating Procedures	7
3.1	Configure Password-Less Logon	7
3.2	Configure Basic CSCF Functionality	8
3.3	Configure Emergency Call	31
3.4	Configure Routing Functions	35
3.5	Configure External Network Selection	38
3.6	Configure End-Of-Selection Analysis	59
3.7	Configure Break-In Control Function	69
3.8	Configure Authentication	70
3.9	Configure Access Awareness	74
3.10	Configure Max Number of Contacts	76
3.11	Configure Shared IFC	78
3.12	Configure Node Default IFC	82
3.13	Configure Geographical Redundancy	86
3.14	Configure IMS Restoration	87
3.15	Configure Load Regulation and Priority	88
3.16	Configure Dynamic User Identity Support	91
3.17	Configure SIP Error Response	93
3.18	Configure Domain Routing Function	100
3.19	Configure Throttling of Diameter Traffic on Cx/Dx Interface	105
3.20	Configure Destination Blacklisting	108
3.21	Configure Re-Registration with HSS Bypass	114
3.22	Configure P-CSCF Restoration Procedure	115
3.23	Configure Media Type Counters	116
3.24	Configure Active User License Model	117
3.25	Configure Licensed Features	119
3.26	Configure I-CSCF Resource Broker Entry	121



3.27	Configure Shutting Down State in CSCF	122
3.28	Configure User Redistribution in CSCF	124
3.29	Configure DSCP in CSCF	126
3.30	Configure Removal of Cached AS Instances	128
3.31	Configure Failover Timer per FQDN	129
3.32	Configure SIP Overload Control	131
3.33	Configure Monitoring Interfaces with SIP OPTIONS	136
3.34	Configure eVIP FE-HA	140



1 Introduction

This document describes how to use the Configuration Management (CM) function in the Call Session Control Function (CSCF).

It also contains the prerequisites for using the CSCF CM function, a set of directives for how to use the CSCF CM function, and some examples of applied directives. The examples and parameter values contained in this document are to be used only as guidance.





2 Basic Concepts

This section defines the basic concepts used when working with the CM for the CSCF.

2.1 CSCF and IMS

This section describes how to configure the CSCF.

2.1.1 Simple IMS System

The IP Multimedia Subsystem (IMS) network in Figure 1 describes the interfaces of the CSCF in a high logical abstract layer.

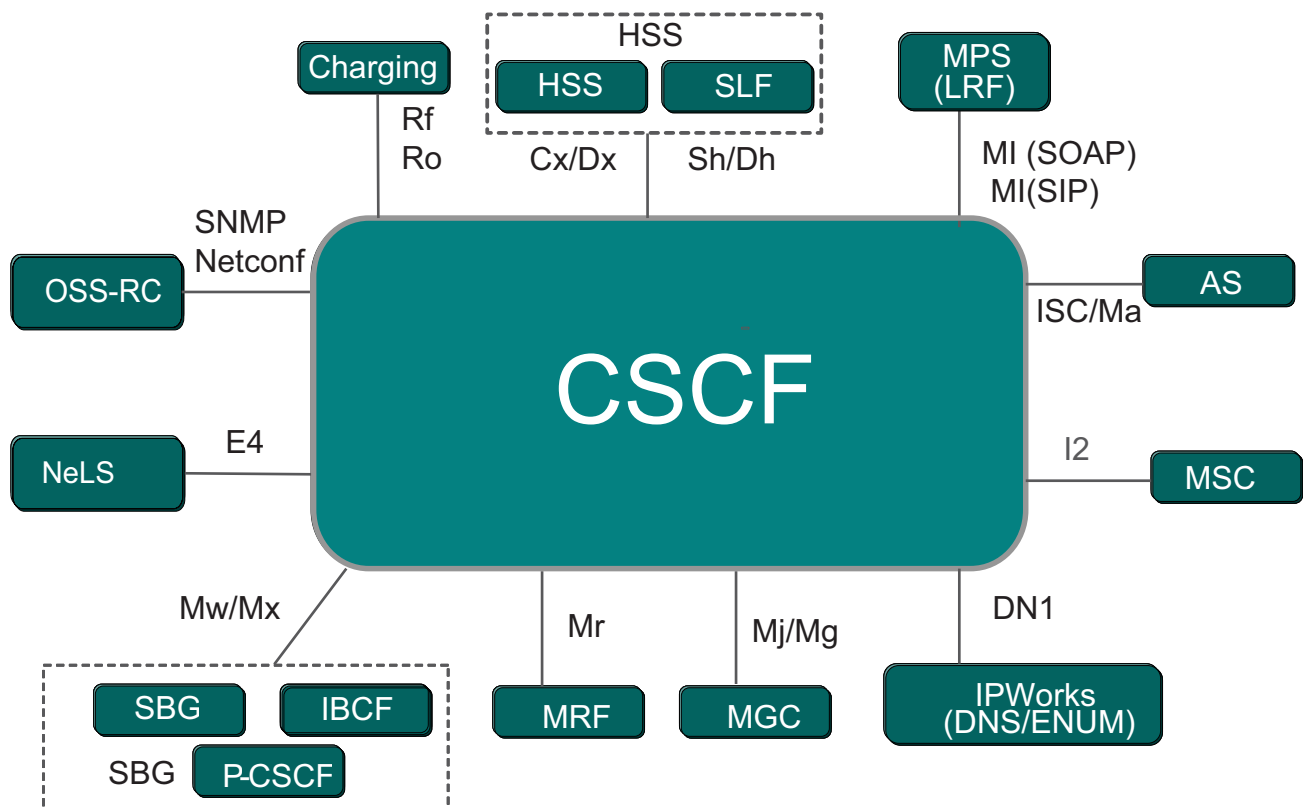


Figure 1 CSCF Logical Architecture and Interfaces

2.1.2 Originating and Terminating CSCF

All traffic in an IMS network passes through the CSCF node. Traffic in telecommunications is called a “call”, which corresponds to a Session Initiation Protocol (SIP) dialog. A call can be described by a half call model, where a half call divides a call into an originating half and a terminating half. The messages pass through the CSCF node of the operator in the originating side of a call, which serves the calling user. The originating CSCF node then forwards the messages to the matching CSCF node at the terminating side, which serves the called user. Originating and Terminating CSCF can be the same physical node. There are also cases where only the originating or only the terminating CSCF node is involved, for example, if the call is initiated by, or terminated at, an Application Server (AS).

The CSCF node is divided in Interrogating CSCF (I-CSCF), Serving CSCF (S-CSCF), Emergency CSCF (E-CSCF), and Emergency Access Transfer Function (EATF) separate node types, according to 3GPP® IMS standards. The CSCF also contains the Break-in Control Function (BCF). It is also possible to collocate I-CSCF and S-CSCF in the same physical node. The IS-CSCF is a combined CSCF node type that includes all configuration attribute settings that are used by I-CSCF and S-CSCF together.

2.2 Configuration Methods

The CM of CSCF is performed through Managed Objects (MOs). The operations on the MOs can be performed through a Common Operation and Maintenance (COM) Northbound Interface (NBI).

The CM interfaces provided by NBIs are the NETCONF and the Ericsson Command-Line Interface (ECLI). By default, the NETCONF interface and the ECLI are inactive. The activation procedure, *Enabling COM*, must be executed to be able to use these NBIs.

For more information on ECLI, refer to *Ericsson Command-Line Interface User Guide*.

The NETCONF interface is a Machine to Machine (M2M) interface, accessed through an SSH client. The default port for NETCONF SSH connection is 830.

The Ericsson Common Information Model (ECIM) compliant Managed Object Model (MOM) is used with NETCONF and ECLI.

For more information about CSCF-related configuration attributes, refer to *Managed Object Model (MOM)*.

The CSCF ECIM model, which also outlines the top-level MOM and the first-level Managed Object Classes (MOCs), is shown in Figure 2. The CSCF application-specific configuration parameters are structured into the following fragments contained in *CscfFunction*:

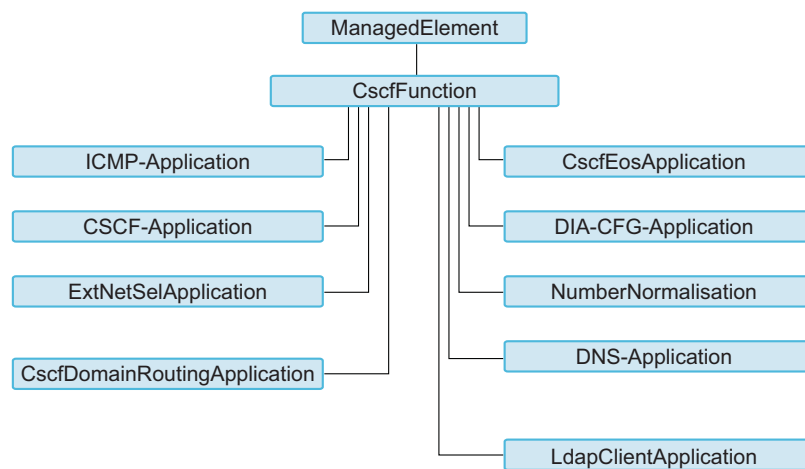


Figure 2 Simplified CSCF ECIM Model Diagram

2.3 Authentication and User Management

Only authorized users can configure the system. The user management principles, user administration, user roles, and authentication methods are described in *User Management*.

For NBI and authentication domain, there are the following role instances:

- CscfApplicationAdministrator
- CscfApplicationSecurityAdministrator
- CscfApplicationOperator
- LocalAuthenticationAdministrator
- SystemAdministrator
- SystemSecurityAdministrator

For more information, refer to *User Management*.





3 Operating Procedures

3.1 Configure Password-Less Logon

Password-less logon is used to log on to the CSCF without entering a pass-phrase or a password. By default, they are required when logging on to the CSCF. Configuring password-less logon is optional.

The logon is configured using the ssh-agent daemon. The ssh-agent is started at a client side from where logon to the CSCF is done. The private key is stored in the client. The public key is deployed in the `<user-home>/.ssh` directory of the CSCF.

After password-less logon is configured, use Secure Shell (SSH) with the `-A` option to log on to the CSCF. The `-A` option enables the forwarding of the authentication agent connection. The forwarding of the authentication agent connection must be enabled with caution.

Note: After logging on to the system, the `-A` option is needed whenever using SSH inside the CSCF. For example, for logging on to the Payload. If the SSH private key is not recognized by the CSCF, a prompt for entering the password is displayed.

To configure the password-less logon:

1. Generate private and public keys and follow the on-screen instructions to save the keys on the client side:

```
ssh-keygen -t rsa
```

Note: Do not set a passphrase. Otherwise, the passphrase is required whenever using the password-less logon.

```
> ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/cscfoperator/.ssh):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/cscfoperator/.ssh/id_rsa.
Your public key has been saved in /home/cscfoperator/.ssh/id_rsa.pub.
```

Result:

A private file `id_rsa` and a public file `id_rsa.pub` are created in the destination of choice on the client.

2. Set up the ssh-agent on the client side:

```
ssh-agent bash
```

```
ssh-add <path-to-location-of-private-key>/id_rsa
```



3. Deploy the public key to the CSCF:

- a. If the `.ssh` directory does not exist in users home directory on the CSCF, create it:

```
mkdir -p $HOME/.ssh -m700
```

- b. Copy the `id_rsa.pub` to the `.ssh` directory:

```
scp id_rsa.pub <username>@OAM IP:$HOME/.ssh
```

4. Log on to the CSCF:

```
ssh <username>@<OAM IP>
```

5. Copy the content of `id_rsa.pub` to the `authorized_keys` file:

```
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

6. Log off from the CSCF and then log on to the CSCF using the `-A` option to verify that the password-less logon is configured successfully:

```
ssh -A <username>@<OAM IP>
```

3.2 Configure Basic CSCF Functionality

This section describes how to configure the essential CSCF functionality used by several CSCF node types.

3.2.1 Configure Common Components

This section describes the common components configuration.

If the CSCF is collocated with other applications on the same physical platform, and they use the same common components, the configuration of these common components is used by all applications.

This document only considers the configuration of common components from a standalone CSCF application point of view.

3.2.1.1 Configure DNS/ENUM Client

The DNS client is a user of the DNS system and performs SIP or TEL lookups (ENUM).

The Domain Name System Server (DNS) is used by the DNS client. All queries not stored in the local cache are forwarded to an external name server. The DNS stack uses the protocol to perform DNS lookups to an external DNS server. A name server periodically checks to ensure that its zones are up to date, and if not, obtains a new copy of updated zones from master files stored locally or in another name server.



The DNS is configured using the available NBI with the *CscfFunction* fragment *DNS-Application*, see Section 2.2 Configuration Methods on page 4.

The settings for DNS are described in Table 1.

Table 1 Network Design

Attribute	Value Example
<i>dnsServerEntry</i>	IP address and port of the DNS servers. At least one entry must be set.
<i>dnsLocalAddress</i>	The source IP address for DNS queries. This attribute is to be set.
<i>dnsRetransmissionTimer</i>	10 (milliseconds, default value)
<i>dnsCacheSize</i>	2000 (number of entries, default value)
<i>dnsUnavailabilityCacheTTL</i>	60 (seconds, default value)

Note: When any of these configuration attributes are changed, the DNS reacts to the change. Some entries are discarded if the maximum size of the cache is reduced below the number of currently used cache entries.

In addition to the common component configuration, the CSCF attributes related to ENUM are shown in Table 2.

Table 2 CSCF ENUM Attributes

Attribute	Value Example
<i>cscfEnumTopDomain</i>	e164.arpa (default value)
<i>cscfEnumForLocalNumbersEnabled</i>	true

When the attribute *cscfEnumForLocalNumbersEnabled* is set to `true`, even local numbers returned from Number Normalization perform an ENUM query. When it is set to `false`, the ENUM query is suppressed when a local number is returned from Number Normalization.

The CSCF is configured using the available NBI with the *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

3.2.1.2 Configure Number Normalization

The Number Normalization function converts local, national, and international numbers as well as Operator Service Numbers (OSNs) and National Significant

Numbers (NSNs), to fully internationalized numbers. This functionality can be applied to any URI.

The Number Normalization provides profiles to cater for large data associated normalization. Profiles can be used to support number plans for more than one country. They can also be used to order numbering plans for countries on region or city basis.

The Number Normalization is configured using the available NBI with the *CscfFunction* fragment *NumberNormalisation*, see Section 2.2 Configuration Methods on page 4.

In addition to the common component configuration, the CSCF attributes that are related to Number Normalization of global (international) telephone numbers are shown in Table 3.

Table 3 Configuration Attributes for Number Normalization

Attribute	Value Example
<i>cscfEnableGlobalNumberNormalization</i>	true
<i>cscfGlobalNumberNormalizationPhoneContext</i>	Telephone number with “+” sign in the beginning or FQDN

The CSCF is configured using the available NBI with the *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

To enable Number Normalization of global numbers:

1. Configure the phone context to use for normalizing global numbers in *cscfGlobalNumberNormalizationPhoneContext*.
2. Set *cscfEnableGlobalNumberNormalization* to true.

For local numbers, the S-CSCF reads the phone context from the phone context attribute in the Request-URI. If the phone context attribute is not received in a SIP URI, the phone context is read from user data received from the Home Subscriber Server (HSS). If this is also empty, the S-CSCF reads the phone context from the configuration of attribute *cscfPhoneContext*. As a last option, if not configured, the S-CSCF name configured in the *scscfDomainNameEntry* is used. If the phone context is not received in a tel URI, the request is rejected.

Attribute *scscfAddPhoneContext* defines the originating S-CSCF handling of the phone context attribute at originating requests including a Request-URI containing a local telephone number in a SIP URI or tel URI form, as follows:

- 0: NO_MODIFICATION_OF_REQUEST_URI, S-CSCF does not modify phone context attribute in the Request-URI.



- 1: `ADD_PHONE_CONTEXT_IN_REQUEST_URI`, S-CSCF is to add a phone context attribute when receiving a Request-URI containing a local telephone number in a SIP URI without any phone context attribute.
- 2: `ADD_OR_OVERWRITE_PHONE_CONTEXT_IN_REQUEST_URI`, S-CSCF is to add or overwrite a phone context attribute when receiving a Request-URI containing a local telephone number in a SIP URI. A received tel URI gets its phone context overwritten.

Attribute *ecscfEmergencyPhoneContext* defines the emergency phone context to be used as input to Number Normalization together with the telephone number received from the Location Retrieval Function (LRF), or if the telephone number received from the LRF is a non-international number.

Attribute *scscfEmergencyPhoneContext* defines the phone context to be used as input to Number Normalization when handling emergency calls in the S-CSCF.

The CSCF does not use the error correction for missing `user=phone` provided by the Number Normalization common component. This means that the following attributes are not used by the CSCF:

- *numNormProfileUserEqPhoneEr*
- *numNormProfileDomNameEr*

The CSCF does not use the phone context provided by the Number Normalization function for NSN numbers and OSN numbers. Therefore, the first entry of attributes *numNormNsnDataNumbers* and *numNormOsnDataContextAndNumbers* are not used by the CSCF.

3.2.2 Configure Diameter Stack

The Diameter stack is configured using the available NBI with the *CscfFunction* fragment *DIA-CFG-Application*, see Section 2.2 Configuration Methods on page 4.

For more information about the `Diameter_Stack` interface, contact next level of support.

The following four CSCF interface container keys appear in the tree on the left panel:

- `ManagedElement=1,CscfFunction=1,DIA-CFG-Application=DIA,DIA-CFG-StackContainer=CSCFCX`
- `ManagedElement=1,CscfFunction=1,DIA-CFG-Application=DIA,DIA-CFG-StackContainer=CSCFRF`
- `ManagedElement=1,CscfFunction=1,DIA-CFG-Application=DIA,DIA-CFG-StackContainer=CSCFRO`

This container interface is created in `DiameterInstallerProc` when the `Diameter_Stack` starts.

`CSCFCX` is used for Cx, Dx, Sh, and Dh interfaces, `CSCFRF` is used for Rf interface, and `CSCFRO` is used for Ro interface. These attributes can be configured in the following combinations:

- Combined IS-CSCF (Cx/Dx, Rf, and Ro)
- I-CSCF (only for Cx/Dx)
- S-CSCF (Cx/Dx, Rf, and Ro)
- E-CSCF (Rf and Sh/Dh)
- BCF (only for Cx/Dx)

Stack IDs, port numbers, and example of host IDs for different node types are shown in Table 4.

Table 4 Diameter Stack

Node	stackId	portNr	hostId
IS-CSCF	CSCFCX	3868	iscscfcx.ericsson.se
	CSCFRF	3868	iscscfrf.ericsson.se
	CSCFRO	3868	iscscfro.ericsson.se
I-CSCF	CSCFCX	3868	icscfcx.ericsson.se
S-CSCF	CSCFCX	3868	scscfcx.ericsson.se
	CSCFRF	3868	scscfrf.ericsson.se
	CSCFRO	3868	scscfro.ericsson.se
E-CSCF	CSCFRF	3869	iscscfrf.ericsson.se
	CSCFCX	3870	iscscfcx.ericsson.se
BCF	CSCFCX	3868	bcfcx.ericsson.se

Note: Each interface must have different IP addresses and host IDs.

3.2.2.1 Configure SCTP

To configure SS7 CAF to operate in Stream Control Transmission Protocol (SCTP) Only Stack mode and enable Diameter over SCTP:

1. For each Diameter protocol reference point (Cx, Ro, and so on) that uses Diameter over SCTP, make the following SS7 CAF configuration changes using Signaling Manager. For information on how to make the changes, refer to *Diameter Management*.



- Modify attribute SCTPs, Sctp FE, Sctp End Points, FE: Local Address Table #1, Sctp Local Address to Diameter IP address of own node.
 - Modify attribute System Components, System Components, CP, CP, Msg Preferred Net to match the internal IP address range (internal IP addresses of the Virtual Machines (VMs)) that is specified in the `cluster.conf` file.
 - Modify the attribute System Components, System Components, CP, CP, Msg Allowed Nets to match the internal IP address range (internal IP addresses of the VMs) that is specified in the `cluster.conf` file.
2. Perform the following Evolved Virtual Internet Protocol (eVIP) configuration changes to enable SCTP:
- a. Create flow policy entries in the eVIP configuration:
 - Create ipv4 or ipv6 SCTP flow policy


```

ipv4: ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=ln_sig_sc,EvipFlowPolicies=1,EvipFlowPolicy=4cscf_diacx_sctp

ipv6: ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=ln_sig_sc,EvipFlowPolicies=1,EvipFlowPolicy=6cscf_diacx_sctp
          
```

Note: The value for `EvipAlb` differs depending on the used eVIP configuration.
 - For the created flow entry, modify attribute `addressFamily` to `ipv4` (`4cscf_diacx_sctp`) or `ipv6` (`6cscf_diacx_sctp`) depending on the type of own Diameter IP address
 - For the created flow entry, modify attribute `dest` to own Diameter IP address
 - For the created flow entry, modify attribute `protocol` to `sctp`
 - For the created flow entry, modify attribute `soGrp` to `1011250`
 - b. Create flow selection policy in the eVIP configuration as follows:
 - Create SCTP flow selection policy `ManagedElement=1,Transport=1,Evip=1,EvipSelectionPolicies=1,EvipSelectionPolicy=ss7_caf_sctp`
 - For the created policy entry, modify attribute `alb` to `ln_sig_sc`.



Note: The value of this parameter must be aligned with the assigned value of `EvipAlb`.

- For the created policy entry, modify attribute `process` to `fe_sctp`
- For the created policy entry, modify attribute `sortorder` to `20.000000`

3. For Diameter protocol communication that uses SCTP, make the following CSCF configuration changes:

Note: `<protocolStackId>` is `CSCFCX` for the Cx protocol.

- a. Modify number of Front Ends (FEs) in Diameter stack by modifying configuration attribute `numberOfFrontEnds` to 2. Configuration attribute `numberOfFrontEnds` is located under `ManagedElement=1,CscfFunction=1,DIA-CFG-Application=DIA,DIA-CFG-Configuration=Diameter`
- b. Configure the settings for own node by editing entry `ManagedElement=1,CscfFunction=1,DIA-CFG-Application=DIA,DIA-CFG-StackContainer=<protocolStackId>,DIA-CFG-OwnNodeConfig=<protocolStackId>` as follows:
 - Modify attribute `SctpAddressesList` and assign it the Diameter IP address of own node.
 - Modify attribute `maxInboundSctpStreams` to 17
 - Modify attribute `maxOutboundSctpStreams` to 17
 - Modify attribute `transportLayerType` to 3 (both SCTP and TCP).
- c. Configure the settings for each Peer node that uses Diameter over SCTP by editing entry `ManagedElement=1,CscfFunction=1,DIA-CFG-Application=DIA,DIA-CFG-StackContainer=<protocolStackId>,DIA-CFG-PeerNodeContainer=<protocolStackId>,DIA-CFG-NeighbourNode=<peerNodeId>` as follows:
 - Modify attribute `enabled` to `false`
 - Modify attribute `sctpAddressesList` and assign it the peer node IP address
 - Modify `transportLayerType` to 2 (SCTP only)
 - Modify attribute `enabled` to `true`

For more information about Diameter, contact next level of support.



Configuration options for multihomed end-to-end SCTP path diversity are available. For this configuration, a multihomed end-to-end SCTP association needs to be able to find at least two paths that do not have any physical equipment in common, to avoid fate sharing. Also, the external routers and the entire IP backbone behind them need to support path diversity end-to-end. For more information, contact the next level support.

3.2.3

Configure CSCF Common Attributes

The CSCF is configured using the available NBI with the *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

The settings for the CSCF Common Configuration are described in Table 5.

Table 5 Common Configuration Attributes for CSCF

Attribute	Value Example
<i>cscfAdministrativeState</i>	LOCKED
<i>cscfISPBehavior</i>	IS
<i>bcfEnabled</i>	true or false When set to true this enables the BCF function.
<i>ecscfEnabled</i>	true or false When set to true this enables the E-CSCF function.
<i>cscfCXOriginRealm</i>	ericsson.se
<i>cscfCXOriginHost</i>	cscf.ericsson.se
<i>cscfCxDestinationRealm</i>	ericsson.se
<i>cscfCxDestinationHost</i>	neighbour1.ericsson.se
<i>cscfDomainAlias</i>	ericsson.se
<i>icscfDomainNameEntry</i>	icscf.ericsson.se
<i>scscfDomainNameEntry</i>	scscf.ericsson.se
<i>ecscfDomainNameEntry</i>	ecscf.ericsson.se
<i>icscfNetworkInterfaceEntry</i>	Protocol:address:port
<i>scscfNetworkInterfaceEntry</i>	Protocol:address:port
<i>ecscfNetworkInterfaceEntry</i>	Protocol:address:port
<i>bcfNetworkInterfaceEntry</i>	Protocol:address:port
<i>ecscfEatfEnabled</i>	true or false When set to true this enables the E-CSCF to use the EATF function.

Attribute	Value Example
<i>eatfEnabled</i>	true or false When set to true this enables the EATF function in EATF node.
<i>eatfDomainNameEntry</i>	eatf.ericsson.se
<i>eatfNetworkInterfaceEntryId</i>	Protocol:address:port

The settings for the CSCF Charging Configuration are described in Table 6.

Table 6 Charging Configuration Attributes for CSCF

Attribute	Value Example
<i>cscfChargingInterOpId</i>	ericsson.se

3.2.3.1 cscfAdministrativeState

The *cscfAdministrativeState* must be set to LOCKED before starting network interface configuration.

3.2.3.2 cscfISPBehavior

The *cscfISPBehavior* attribute determines which of the CSCF applications I-CSCF or S-CSCF that applies.

Note: P-CSCF is not supported.

3.2.3.3 bcfEnabled

The *bcfEnabled* attribute determines if the CSCF application BCF applies.

3.2.3.4 ecscfEnabled

The *ecscfEnabled* attribute determines if the CSCF application E-CSCF applies.

3.2.3.5 ecscfEatfEnabled

The *ecscfEatfEnabled* attribute is configured in E-CSCF to trigger the invocation of the EATF. This attribute is not used to activate the EATF role in the network element.

3.2.3.6 eatfEnabled

The *eatfEnabled* attribute determines if the CSCF application EATF applies.



3.2.3.7

Cx/Dx/Sh/Dh Diameter Interface

The operator can deploy the HSS/SLF in few ways considering load sharing and redundancy aspects.

From a CSCF application point of view, configure the following four attributes for the behavior of the Cx, Dx, Sh, and Dh interfaces:

- *cscfCXOriginRealm* and *cscfCXOriginHost* identify the CSCF that sends the Diameter message to the Diameter server (the HSS/SLF). These are always configured in the same way independently of the HSS/SLF behavior.
- *cscfCxDestinationRealm* and *cscfCxDestinationHost* specify the HSS/SLF realm and host address, where the Diameter messages are sent. If an SLF or an HSS redundant cluster is used in the network, the *cscfCxDestinationHost* must be set to "NotConfigured". If a single instance HSS is used, the host address of that HSS is to be configured in *cscfCxDestinationHost*.

Apart from the CSCF application attributes, the HSS/SLF deployment depends on the Diameter configuration.

Some examples are shown in Figure 3, Figure 4, and Figure 5.

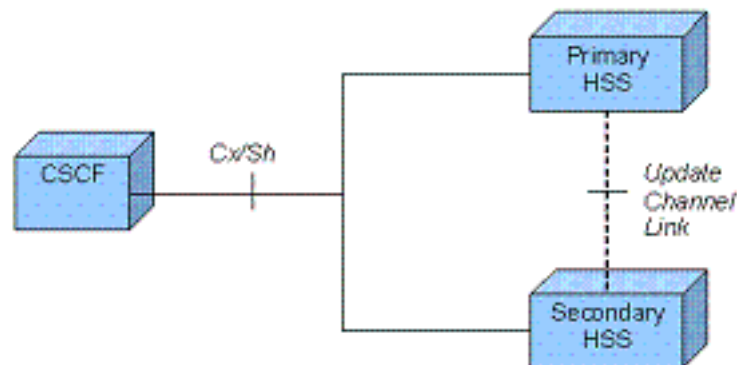


Figure 3 One Redundant HSS without SLF

One HSS in the network, which has geographical redundancy, is shown in Figure 3. In this case, *cscfCxDestinationRealm*=HSSrealm.operator.net and *cscfCxDestinationHost*=PrimaryHSS.operator.net. The Diameter configuration in the CSCF includes a Realm Routing Table for the HSSrealm.operator.net where PrimaryHSS Destination Host is a Peer Node and SecondaryHSS is configured as *secondaryNodeIds*. In addition, the two HSS nodes are configured with Diameter NetRed, but this is outside the CSCF configuration.

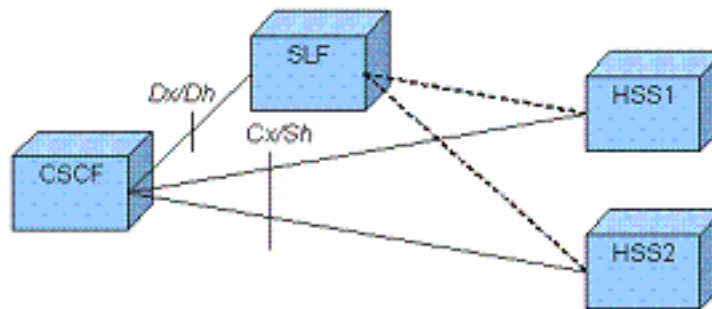


Figure 4 Multiple HSS in Classic Architecture, Redirect, or Proxy Mode

Two HSSs in the network, without any redundancy, are shown in Figure 4. An SLF is used to select which HSS that is used based on which subscriber it concerns. In this case, `cscfCxDestinationRealm=HSSrealm.operator.net` and `cscfCxDestinationHost=NotConfigured`. The Diameter configuration in the CSCF includes a Realm Routing Table for the `HSSrealm.operator.net` where SLF Destination Host is a Peer Node. Both HSS1 and HSS2 are configured as Peer Nodes to the CSCF (outside the Realm Routing Table). The CSCF configuration is independent of whether the SLF runs in a proxy or redirect mode.

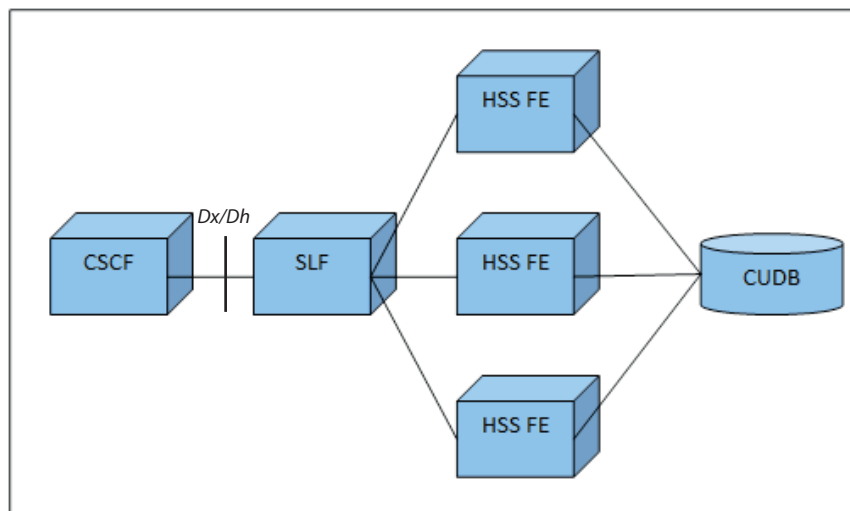


Figure 5 HSS in Layered Architecture

The HSS in a layered architecture is shown in Figure 5, where the SLF node is used to load balance between several HSS FEs and the subscription data is stored in a back-end system Centralized User Database (CUDB). In this case `cscfCxDestinationRealm=HSSrealm.operator.net` and `cscfCxDestinationHost=NotConfigured`. The Diameter configuration in the CSCF includes a Realm Routing Table for the `HSSrealm.operator.net` where SLF Destination Host is a Peer Node. None of the HSS FE nodes are configured as Peer Nodes to the CSCF which results in that the subsequent request falls back to realm routing, that is, route to the SLF node independent of which HSS FE that responded to the request from the CSCF.



3.2.3.8 **cscfDomainAlias**

cscfDomainAlias is used for the following functionality:

- S-CSCF:

Originating S-CSCF uses *cscfDomainAlias* to determine if a SIP URI that contains a telephone number is to be treated as telephone number or if the SIP URI is to be routed normally using the domain part.

Any application that requires that this type of SIP URI is to be handled as a telephone number must configure the *cscfDomainAlias* with the relevant domains, for example, “domain.com”.

cscfDomainAlias can include multiple domains.

3.2.3.9 **E/S-CSCF, BCF, EATF Domain Names**

The domain name, in the form of a Fully Qualified Domain Name (FQDN), of each CSCF node type is configured in *IcscfDomainNameEntry*, *ScscfDomainNameEntry*, *EcscfDomainNameEntry*, *BcfDomainNameEntry*, and *eatfDomainNameEntry* respectively.

3.2.3.10 **CSCF Network Interface**

The network interface is configured using the available *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

Click **CscfNwIfContainerKey=0** in the tree on the left panel.

The following CSCF interface container keys are displayed:

- BcfNwIfContainerKey=0
- IcscfNwIfContainerKey=0
- ScscfNwIfContainerKey=0
- EcscfNwIfContainerKey=0
- eatfNwIfContainerKey=0

These container interfaces are created by the system when the CSCF starts.

It is recommended to have each CSCF node in an own VIP interface. External interface is, for example, DNS, O&M, Cx, and Rf. For more information, refer to *CSCF VNF Network Connectivity Overview*.

The node ports in the CSCF are shown in Table 7.

Table 7 Ports and Traffic Direction

Node	Interface	Ports	Port Role	Protocol
I-CSCF	Messages to/from P/S-CSCF and to/from AS	5060	-	UDP / TCP
S-CSCF	Messages to/from I/P-CSCF and to/from AS	5060	-	UDP / TCP
E-CSCF	Messages to/from P-CS CF/Gateways	5060	-	UDP / TCP
BCF	Messages to/from S-CS CF/Gateways	5060	-	UDP / TCP
EATF	Messages to/from E/I-CSCF	5060	-	UDP / TCP

3.2.3.11

Enable CSCF

To enable the CSCF:

- Set the attributes described in the following sections:
 - Section 3.2.1.1 Configure DNS/ENUM Client on page 8
 - Section 3.2.1.2 Configure Number Normalization on page 9
 - Section 3.2.2 Configure Diameter Stack on page 11
 - Section 3.2.3 Configure CSCF Common Attributes on page 15
 - Section 3.2.4 Configure CSCF Charging Attributes on page 21

Note: The Number Normalization configuration attributes and the charging configuration attributes are only set if applicable except *cscfChargingInterOpId*, which is mandatory for the S-CSCF and E-CSCF.
- Connect to the CSCF node using the NBI with *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.
- Set the *cscfISPBehavior* to the desired node type.
- Set the *cscfAdministrativeState* to UNLOCKED.



When the CSCF node is in unlocked mode, it is ready to receive SIP signaling.

For enabling the E-CSCF and ETAF, see Section 3.3.2 Enable Emergency Call with E-CSCF on page 33.

For enabling the BCF, see Section 3.7 Configure Break-In Control Function on page 69.

3.2.4 Configure CSCF Charging Attributes

The S-CSCF supports both online and offline charging. The E-CSCF supports offline charging.

The charging configuration consists of the common charging attributes, and the charging triggers.

The CSCF charging trigger functionality is the means by which the CSCF is able to define few conditions allowing the determination of the charging actions to take.

There are three parts to the charging trigger, as follows:

- Charging trigger
- Charging profile
- Charging Attribute-Value Pairs (AVPs)

Figure 6 illustrates the charging configuration components. A charging profile must first be configured before it is referenced by the charging trigger.

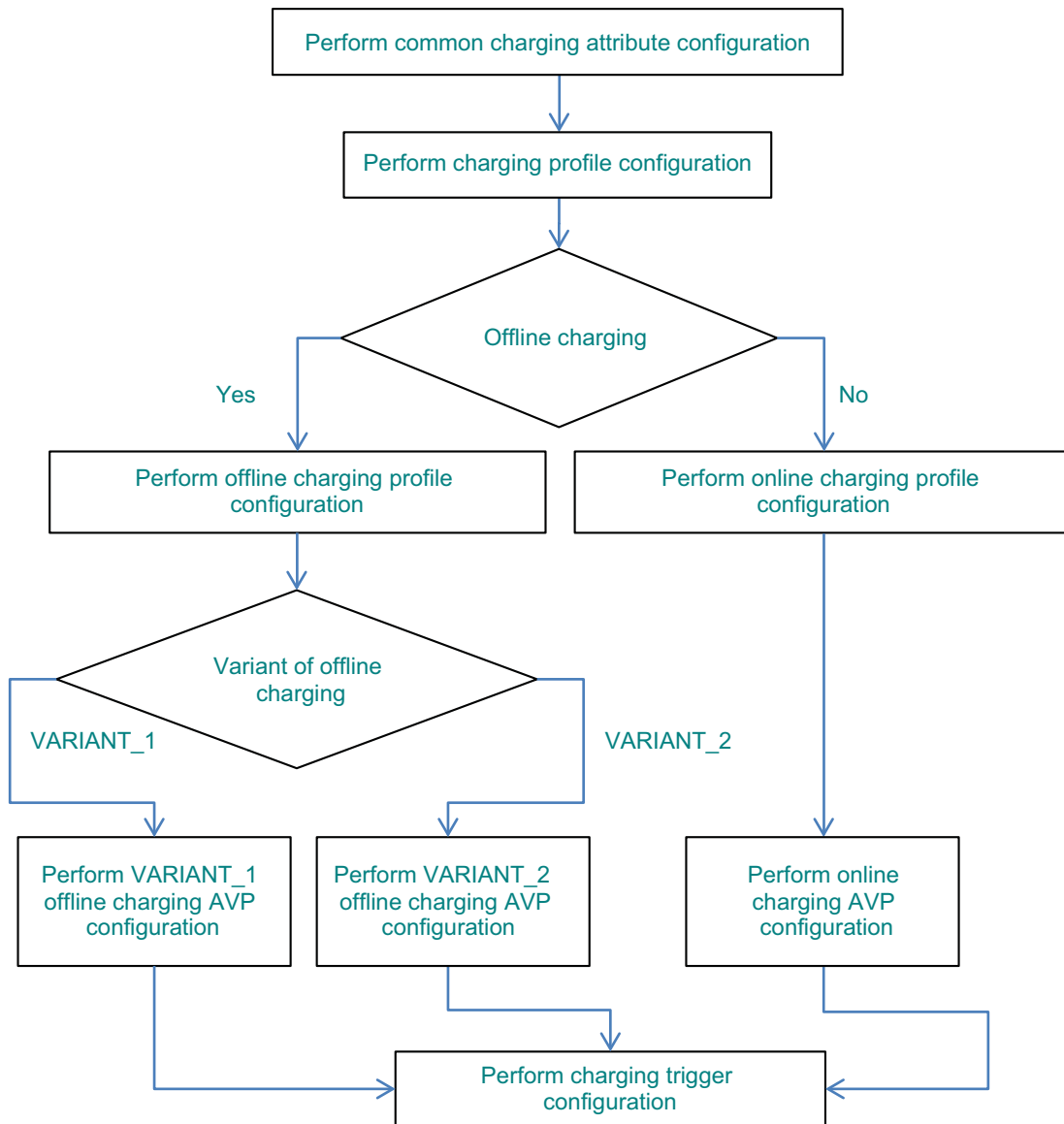


Figure 6 Charging Configuration Components

For more information regarding common charging triggers, charging triggers, charging profile, and charging AVPs, refer to *Managed Object Model (MOM)*.

3.2.4.1 Common Charging Attributes

The configuration attributes defining the general charging behavior must be defined and are shown in Table 8. These attributes are applicable to both S-CSCF and E-CSCF, unless explicitly indicated.



Table 8 Common Charging Attributes

Attribute	Value Example
<i>cscfChargingInterOpId</i>	ericsson.se
<i>cscfOwnChargingHost</i>	cscf.ericsson.se
<i>cscfOwnChargingRealm</i>	ericsson.se
<i>cscfChargingResendingTimer</i>	200
<i>cscfChargingBackupRetryInterval</i>	100
<i>cscfMaxNumChargingBackupfilesPerDestination</i>	1
<i>cscfChargingCancelCauseCode</i>	SUCCESSFUL_TRANSACTION (-1)
<i>cscfChargingSessionTimerExpiresCauseCode</i> ⁽¹⁾	UNSPECIFIED_ERROR (1)

(1) This attribute applies only to S-CSCF.

3.2.4.2

Charging Triggers

The S-CSCF and the E-CSCF start offline or online charging support, or both, after a successful validation of a set of charging trigger conditions. The charging trigger conditions are validated against the SIP information received and particular conditions of the SIP session.

A set of charging trigger conditions include the following:

- Trigger attributes
- A group of charging trigger elements

Table 9 shows the trigger attributes and the trigger elements. For more information, refer to the *Managed Object Model (MOM)*.

Table 9 Trigger Attributes and Trigger Elements

Attributes ⁽¹⁾	Value Example
Trigger Attributes	
<i>xcscfChargingTriggerName</i>	TriggerEx
<i>xcscfChargingTriggerPriority</i> ⁽²⁾	1024
<i>xcscfChargingProfileName</i> ⁽³⁾	Profile-ABC
<i>xcscfChargingTriggerConditionType</i> CNF	true
Trigger Elements	
<i>xcscfChargingTriggerGroup</i> ⁽⁴⁾	TriggerExGroup

Attributes ⁽¹⁾	Value Example
xcscfChargingSipMethod	INVITE
xcscfChargingSipMethodTriggerConditionNegated	false
xcscfChargingRequestUri	/^(sip:)(.+)\$/
xcscfChargingRequestUriTriggerConditionNegated	false
xcscfChargingSipHeader	Contact
xcscfChargingSipHeaderTriggerConditionNegated	false
xcscfChargingSipHeaderContent	/^(sip:)(.+)\$/
xcscfChargingSessionCase	0
xcscfChargingSessionCaseTriggerConditionNegated	false
scscfChargingAsInvolvementTriggerConditionNegated ⁽⁵⁾	false
scscfChargingRoamingStatus ⁽⁵⁾	0
scscfChargingRoamingStatusTriggerConditionNegated ⁽⁵⁾	false

(1) The attributes apply to S-CSCF or E-CSCF with “x” replaced by “s” or “e”, respectively.

(2) This attribute is for future extension to support multiple sets of charging trigger conditions.

(3) Multiple charging profiles can be available but only one profile can be referenced by a charging trigger.

(4) Only one trigger group per trigger can be defined.

(5) This trigger element is for S-CSCF only.

The trigger elements are optional conditions to be evaluated by the S-CSCF and the E-CSCF against the SIP session or SIP event operations. See Table 9 for a summary of the optional trigger elements. Except from the trigger element for SIP method (xcscfChargingSipMethod), each one of the other trigger elements can be configured at most once in a trigger group. For example, at most one regular expression for Request-URI matching can be configured for a Request-URI-based trigger condition (xcscfChargingRequestUri). Multiple unique elements can be configured for a SIP method-based trigger condition. For example, three instances of the SIP method trigger element (xcscfChargingSipMethod) for INVITE, SUBSCRIBE, NOTIFY can be configured as three SIP method-based trigger conditions.

Each trigger element has a “negated” configuration attribute. If this attribute is set to true, that is, condition negated, the corresponding trigger element is successfully evaluated when the configured condition is not matched. If this attribute is set to false, that is, condition not negated, the corresponding trigger element is successfully evaluated when the configured condition is matched. Otherwise, the evaluation fails.



Example 1:

The trigger element evaluation is successful if the SIP request is not a SUBSCRIBE. The evaluation fails if the SIP request is a SUBSCRIBE.

- `scscfChargingSipMethod = SUBSCRIBE`
- `scscfChargingSipMethodTriggerConditionNegated = true`

Example 2:

The trigger element evaluation is successful if the SIP request is an INVITE. The evaluation fails if the SIP request is not an INVITE.

- `scscfChargingSipMethod = INVITE`
- `scscfChargingSipMethodTriggerConditionNegated = false`

Note: re-INVITE, UPDATE, and BYE are SIP methods within a successfully established SIP session. They are not configurable SIP method trigger elements. INVITE is the only configurable SIP method trigger for E-CSCF.

Within the charging trigger group, the charging trigger element evaluation results can be logically linked through AND or OR operators.

Operators configure the trigger attribute `xcscfChargingTriggerConditionTypeCNF` to combine all the trigger element evaluation results into a logical expression for the final evaluation. If the trigger attribute is set to true, a logical expression is formed to evaluate all trigger element evaluation results within the group with OR operations. The final evaluation is successful if any of the trigger element evaluations is successful. If the trigger attribute is set to false, a logical expression is formed to evaluate all trigger element evaluation results within the group with AND operations. The final evaluation is successful if all trigger element evaluations are successful. Otherwise the final evaluation fails.

Example 3:

In this example, the final evaluation is successful and charging is triggered if the SIP method is either an INVITE or a SUBSCRIBE. Charging is not triggered otherwise for other SIP methods.

- `scscfChargingTriggerConditionTypeCNF = true`
- `scscfChargingSipMethod = INVITE`
- `scscfChargingSipMethodTriggerConditionNegated = false`
- `scscfChargingSipMethod = SUBSCRIBE`
- `scscfChargingSipMethodTriggerConditionNegated = false`

Example 4:

In this example, the final evaluation is successful and charging is triggered only for an INVITE with a tel URI in the Request-URI. Otherwise charging is not triggered.

- *scscfChargingTriggerConditionTypeCNF* = false
- *scscfChargingSipMethod* = INVITE
- *scscfChargingSipMethodTriggerConditionNegated* = false
- *scscfChargingRequestUri* = */^(.*)tel:(.+)\$/*
- *scscfChargingRequestUriTriggerConditionNegated* = false

3.2.4.3 Charging Profiles

The charging profile contains the charging actions acted upon if charging trigger conditions are met.

Multiple charging profiles can be configured in the S-CSCF while only one charging profile can be configured in the E-CSCF. One charging profile can be referenced by a charging trigger through the charging trigger attribute *scscfChargingProfileName* or *ecscfChargingProfileName* for S-CSCF or E-CSCF, respectively. The charging profile is invoked on successful final evaluation of the charging trigger that references the profile.

A charging profile contains configuration information for offline or online charging support, or both.

Each charging profile contains the attribute *scscfChargingCase* or *ecscfChargingCase* for S-CSCF or E-CSCF, respectively, for offline or online charging support control, or both. See Table 10 for a summary of the configurable case values.

Table 10 Offline and Online Charging Control

<i>scscfChargingCase</i> / <i>ecscfChargingCase</i> Value	Notes
NoCharging	For both S-CSCF and E-CSCF
OfflineChargingOnly	For both S-CSCF and E-CSCF
OfflineAndOnlineCharging	For S-CSCF only
OnlineChargingPrecedence	For S-CSCF only
OnlineChargingOnly	For S-CSCF only

For S-CSCF, a charging profile can further contain a child profile for offline charging support configuration or a child profile for online charging support configuration, or both.



For E-CSCF, a charging profile can further contain a child profile for offline charging support configuration.

The offline or online child profile, or both, must be configured to specify the desired charging behavior. For more information, see section Section 3.2.4.5 Offline Charging Configuration Steps on page 27 for charging configuration examples.

3.2.4.4 Charging AVPs

Charging AVPs are listed within the offline or online child profile, or both, of a charging profile.

S-CSCF and E-CSCF provide two variants of offline charging support. VARIANT_1 supports Diameter and 3GPP AVPs mostly specified in 3GPP specifications before release 7; VARIANT_2 supports offline charging based on 3GPP release 12 specifications. Refer to *CSCF Rf Interface* for more details.

Offline charging variant is configured through offline charging profile attribute *scscfOfflineChargingProfileVariant* and *ecscfOfflineChargingProfileVariant* for S-CSCF and E-CSCF, respectively. The new variant value is applicable to SIP sessions/events established after the configuration switch. On-going SIP sessions/events established before the switch are subject to the previous variant configuration. CDF must be able to exchange ACR/ACA messages of the applicable variants with CSCF. Offline charging variant configuration can be switched during live traffic. However, it is recommended that variant switching takes place during low or no traffic period to minimize mixed ACR/ACA messaging from the two variants.

For VARIANT_1, different AVPs have different default settings for inclusion to or exclusion from an allowed Accounting-Request (ACR) message (Start, Interim, Stop, or Event, or a combination). Operators can configure if a specific AVP is generated or not in a specific ACR message.

For VARIANT_2, S-CSCF and E-CSCF are to generate an AVP in all allowed ACR messages by default. Operators can configure if a specific AVP is not to be generated. This configuration affects all ACR messages for the AVP.

For S-CSCF online charging, charging AVPs define if an optional AVP must be included in the Credit Control Request (CCR) message. If the charging AVP is not configured, it is not included in the CCR message.

3.2.4.5 Offline Charging Configuration Steps

3.2.4.5.1 Offline Charging Configuration – VARIANT_1 Example

This section describes the E-CSCF VARIANT_1 offline charging configuration. See Table 8 for example values of the common charging attributes. These values are read by the S-CSCF and the E-CSCF for all charging support. See

Table 11 for example values of the E-CSCF charging profile which specifies VARIANT_1 offline charging support.

Table 11 Charging Profile Configuration

Charging Profile Attribute	Value Example
<i>ecscfChargingProfile</i>	ECSCF-PROF
<i>ecscfChargingCase</i>	OfflineChargingOnly
<i>ecscfOfflineChargingProfile</i>	ECSCF-PROF
<i>ecscfChargingDefaultDestinationRealm</i>	remote.se
<i>ecscfChargingEnabledCancel</i>	true
<i>ecscfOfflineChargingEnable3xx</i>	true
<i>ecscfOfflineChargingEnable4xx5xx6xx</i>	true
<i>ecscfOfflineChargingProfileVariant</i>	VARIANT_1
<i>ecscfOfflineChargingServiceContextId</i>	12.32260@3gpp.org ⁽¹⁾

(1) This attribute is not applicable to VARIANT_1

See Table 12 for example values for optional AVP configuration. For illustration purposes, the table shows only configurations of the Calling-Party-Address AVP for ACRs (Start, Interim, and Stop) and the Cause-Code AVP for ACRs (Stop and Event).

Table 12 Offline Charging VARIANT_1 AVP Configuration

VARIANT_1 AVP Attribute	Value Example
<i>ecscfOfflineChargingAVP</i>	ECSCF-PROF
<i>ecscfOfflineChargingCallingPartyAddress</i>	Start=true; Interim=true; Stop=true; Event=false
<i>ecscfOfflineChargingCauseCode</i>	Stop=true; Event=true

See Table 13 for example values of charging trigger attributes. For illustration purposes, only one trigger element is configured within the trigger group. The charging trigger references the charging profile ECSCF-PROF defined in Table 11 with the attribute *ecscfChargingProfileName*.

Table 13 Charging Trigger Configuration

Trigger Attribute	Value Example
<i>ecscfChargingTriggerName</i>	ECSCF-TRIGGER



Trigger Attribute	Value Example
<i>ecscfChargingProfileName</i>	ECSCF-PROF
<i>ecscfChargingTriggerPriority</i>	1024
<i>ecscfChargingTriggerConditionTypeCNF</i>	true
<i>ecscfChargingTriggerGroup</i>	ECSCF-TRIGGER:ID1
<i>ecscfChargingSipMethodTrigger</i>	ECSCF-TRIGGER:ID1:INVITE
<i>ecscfChargingSipMethod</i>	INVITE
<i>ecscfChargingSipMethodTriggerConditionNegated</i>	false

3.2.4.5.2 Offline Charging Configuration – VARIANT_2 Example

The next example illustrates S-CSCF VARIANT_2 offline charging configuration. See Table 8 for example values of the common charging attributes. See Table 14 for example values of the S-CSCF charging profile which specifies VARIANT_2 offline charging support only.

Table 14 Charging Profile Configuration VARIANT_2

Charging Profile Attribute	Value Example
<i>scscfChargingProfile</i>	SCSCF-PROF
<i>scscfChargingCase</i>	OfflineChargingOnly
<i>scscfOfflineChargingProfile</i>	SCSCF-PROF
<i>scscfChargingDefaultDestinationRealm</i>	remote.se
<i>scscfChargingEnabledCancel</i>	true
<i>scscfOfflineChargingEnabled3xx</i>	true
<i>scscfOfflineChargingEnabled4xx5xx6xx</i>	true
<i>scscfOfflineChargingProfileVariant</i>	VARIANT_2
<i>scscfOfflineChargingServiceContextId</i>	01.240.12.32260@3gpp.org

See Table 15 for example values for VARIANT_2 optional AVP omission configuration. For illustration purposes, the table does not configure to generate the 3GPP Message-Body AVP (Vendor Id=10415, AVP code=889), the Ericsson Event-NTP-Timestamp AVP (Vendor Id=193, AVP code=340), and the Ericsson Dial-Around-Indicator AVP (Vendor Id=193, AVP code=1160).

Table 15 Offline Charging VARIANT_2 AVP Omission Configuration

VARIANT_2 AVP Omission Attribute	Value Example
<i>scscfOfflineChargingOmitsId</i>	SCSCF-PROF
<i>scscfOfflineChargingOmitVendorId</i>	SCSCF-PROF:10415
<i>scscfOfflineChargingOmitAVP</i>	889
<i>scscfOfflineChargingOmitVendorId</i>	SCSCF-PROF:193
<i>scscfOfflineChargingOmitAVP</i>	340
<i>scscfOfflineChargingOmitAVP</i>	1160

See Table 16 for example values of charging trigger attributes. Only two trigger elements, that is, SIP methods INVITE and REGISTER, are configured within the trigger group. The charging trigger references the charging profile SCSCF-PROF defined in Table 11 with the attribute *scscfChargingProfileName*.

Table 16 Charging Trigger Attributes

Trigger Attribute	Value Example
<i>scscfChargingTriggerName</i>	SCSCF-TRIGGER
<i>scscfChargingProfileName</i>	SCSCF-PROF
<i>scscfChargingTriggerPriority</i>	1024
<i>scscfChargingTriggerConditionTypeCNF</i>	true
<i>scscfChargingTriggerGroup</i>	SCSCF-TRIGGER:ID1
<i>scscfChargingSipMethodTrigger</i>	SCSCF-TRIGGER:ID1:INVITE
<i>scscfChargingSipMethod</i>	INVITE
<i>scscfChargingSipMethodTriggerConditionNegated</i>	false
<i>scscfChargingSipMethodTrigger</i>	SCSCF-TRIGGER:ID1:REGISTER
<i>scscfChargingSipMethod</i>	REGISTER
<i>scscfChargingSipMethodTriggerConditionNegated</i>	false



3.3 Configure Emergency Call

If the E-CSCF is used to provide the emergency function, it allows a user to perform emergency calls, which are prioritized and routed to the correct emergency center, that is, Public Safety Answering Point (PSAP). The emergency center is selected depending on the dialed emergency number, for example, 112, and the location or the IP address of the user (dependent on configuration).

There are two options if the E-CSCF is not deployed in the network, as follows:

- The IMS network handles emergency calls. This means that the emergency call is handled as an ordinary call, except that the emergency call is prioritized in the network, using the normal call path for a call (external P-CSCF, S-CSCF, terminating network) and thus the external P-CSCF only allows emergency calls for registered users.
- The IMS network does not handle emergency calls. This means that emergency calls are rejected with a redirect message by the external P-CSCF.

There are two options if the E-CSCF is deployed in the network, as follows:

- The emergency call is routed directly from the external P-CSCF to E-CSCF. If so, the external P-CSCF allows emergency calls from both unregistered and registered users.
- The emergency call is routed through the S-CSCF to E-CSCF. If so, the external P-CSCF only allows emergency calls from registered users.

Callbacks from an emergency center to the user that performed the emergency call are handled as a normal call, apart from priority handling if the priority indication is set to “emergency”.

The E-CSCF is not aware of registration or barring state, and does not trigger any services for the user. Furthermore, no authentication of the user is performed. If authentication is required, the emergency call must be routed through the S-CSCF where the authentication is performed.

For Voice over LTE (VoLTE) emergency call, emergency registration is required. EATF can optionally be enabled to anchor the VoLTE emergency call and perform access transfer, if necessary. When EATF is enabled, optional redundant EATFs support can be enabled through configuration.

The E-CSCF is configured using the available NBI with *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

3.3.1 Emergency Attributes

The settings for the emergency attributes within the E-CSCF are described in Table 17.

Table 17 Configuration Attributes for E-CSCF

Attribute	Value Example
<i>ecscfDefaultPsapBehavior</i>	USE_DEFAULT_CONFIG, INVOKE_BGCF_FOR_TEL_NUMBER
<i>ecscfDefaultPSAPNumber</i>	Telephone number with or without “+” sign in the beginning
<i>ecscfEmergencyLRFAddress</i>	FQDN or IPv4 dotted decimal address for HTTP-based MI interface
<i>ecscfEmergencyPhoneContext</i>	Telephone number with “+” sign in the beginning or FQDN
<i>ecscfHttpLocalAddress</i>	IPv4 dotted decimal address
<i>ecscfHttpRequestTimer</i>	500
<i>ecscfFetchRefLocationInfo</i>	TRUE, FALSE
<i>ecscfCalledNumberManipulation</i>	TRUE, FALSE
<i>ecscfPutPsapNumberInRn</i>	TRUE, FALSE
<i>ecscfHttpDigestPw</i>	Password with 6–10 characters (A–Z, a–z, 1–9)
<i>ecscfHttpDigestUsername</i>	username
<i>ecscfSoapBehavior</i>	1 or 2
<i>ecscfEatfAddress</i>	FQDN, IPv4 dotted decimal, or IPv6
<i>ecscfEatfEnabled</i>	TRUE, FALSE

The settings for the emergency attributes within the EATF are described in Table 18.

Table 18 Configuration Attributes for EATF

Attribute	Value Example
<i>eatfEnabled</i>	TRUE, FALSE
<i>eatfPsFallbackTimer</i>	0, 30, 60
<i>eatfSessionIdentifier</i>	1, 2
<i>eatfRedundantEatfEntry</i>	1:192.15.13.4:2459 3:[541:125:258::13:65]:2254



The settings for the emergency attributes within the S-CSCF are described in Table 19.

Table 19 Configuration Attributes for S-CSCF

Attribute	Value Example
<i>cscfEmergencyCallFailureRoute</i>	<code>sip:mgc.operator.com;lr</code>
<i>cscfEmergencyCallFailureDestination</i>	<code>tel:911;phone-context=+358</code>

3.3.2 Enable Emergency Call with E-CSCF

To enable the emergency call with the E-CSCF:

1. Ensure that the attributes described in the following sections are configured:

- Section 3.2.1.1 Configure DNS/ENUM Client on page 8
- Section 3.2.1.2 Configure Number Normalization on page 9
- Section 3.2.2 Configure Diameter Stack on page 11
- Section 3.2.3 Configure CSCF Common Attributes on page 15

Note: Only attributes *cscfCXOriginRealm*, *cscfCXOriginHost*, *cscfCxDestinationRealm*, *cscfCxDestinationHost*, *cscfDomainAlias*, and *cscfChargingInterOpId* are configured.

- Section 3.2.4 Configure CSCF Charging Attributes on page 21
- Section 3.5 Configure External Network Selection on page 38

The Number Normalization configuration attributes (for the emergency phone context), the Diameter, the Cx, the External Network Selection (ENS), and the charging configuration attributes are only set if applicable.

The ENS tables can be configured to route the emergency call to a PSAP, based on the received telephone number from LRF, or a default PSAP in various error situations. For example, when no PSAP number has been received from the LRF. If *ecscfDefaultPsapBehavior* is configured to **USE_DEFAULT_CONFIG**, the ENS tables route the emergency call to the default PSAP, based on the telephone number configured in *ecscfDefaultPSAPNumber*. If *ecscfDefaultPsapBehavior* is configured to **INVOKE_BGCF_FOR_TEL_NUMBER**, the ENS tables route the emergency call to the default PSAP, based on the telephone number found in the original Request URI. See the example in Section 3.5.2 External Network Selection Table Configuration Examples on page 41.

2. If applicable, set the configuration attributes in Table 17.

3. Set the *cscfAdministrativeState* to LOCKED.
4. Set the *ecscfEnabled* to TRUE and configure the *ecscfDomainNameEntry*.
5. Configure at least one *ecscfNetworkInterfaceEntry*.
6. Set the *cscfAdministrativeState* to UNLOCKED.

When the E-CSCF node is in unlocked mode, it is ready to receive SIP signaling.

7. If the emergency call is to be routed through the S-CSCF to E-CSCF, the S-CSCF sees E-CSCF as an AS, Trigger data is to be provisioned in the HSS to accomplish the triggering of E-CSCF from S-CSCF. The trigger data can, for example, be set up to trigger on the priority header emergency, or on the dialed number.

3.3.3 Enable Emergency Call without E-CSCF

Routing emergency calls through the S-CSCF instead of the E-CSCF is determined by the external P-CSCF.

To enable multiple destinations in priority order from the ENS analysis, see Table 30. In this case, S-CSCF sends the requests one by one so that if the current request returns any negative response code or time-out, S-CSCF sends the request to the next *extNetSelPoolURI* found in the *extNetSelPoolTableEntry*.

It is possible to configure a common destination to be used when none of the configured gateways in the ENS analysis is reachable. To enable this:

1. Configure the attributes in Table 19 to the following values:
 - *cscfEmergencyCallFailureRoute* = SIP URI to be put into the Route header (this scenario always uses loose routing, so if no "lr" attribute provided, the CSCF automatically adds it)
 - *cscfEmergencyCallFailureDestination* = Optionally set to a SIP or tel URI to be used as Request-URI in the INVITE to the destination.

3.3.4 Enable Emergency Access Transfer Function

For VoLTE emergency Single Radio Voice Call Continuity (SRVCC), it is necessary to enable the invocation of EATF from the E-CSCF.

To enable the invocation of EATF:

1. Enable EATF by setting *eatfEnabled* to true and configure the *eatfDomainNameEntry*.



2. Define the SIP interface for EATF by configuration parameter *eatfNetworkInterfaceEntryId*.
3. Verify that EATF is operational by ensuring that the value of *eatfOperationalState* is 1. If it is not 1, ensure that *eatfNetworkInterfaceEntryId* is configured properly.
4. Define the EATF address in E-CSCF by configuration parameter *ecscfEatfAddress*.
5. Enable EATF in E-CSCF by configuration parameter *ecscfEatfEnabled*.
6. Verify the administrative state of the EATF, if it is locked (*CscfAdministrativeState*=0) unlock it by setting *CscfAdministrativeState* to 1.

Optionally, redirection to redundant EATFs during access transfer request handling can be configured. When an EATF cannot handle the access transfer request, because the related session is not anchored there, EATF responds with a 305 Use Proxy response including the redundant EATF addresses. The I-CSCF redirects the access transfer request to an address with the highest q-value found in the 305 Use Proxy response.

To enable redundant EATF:

1. Configure *eatfRedundantEatfEntry* with redundant EATF addresses and priorities.

3.4 Configure Routing Functions

This section describes how to configure different routing functions in the CSCF.

The CSCF and ENS are both configured using the NBI with *CscfFunction* fragment *CSCF-Application* and *ExtNetSelApplication*, see Section 2.2 Configuration Methods on page 4.

3.4.1 Configure Basic Routing Attributes

The settings of the routing attributes within the CSCF are described in Table 20.

Table 20 Routing Attributes

Attribute	Value Example
<i>cscfEnumForLocalNumbersEnabled</i>	true
<i>cscfExtNetSelOnRnBeforeEnum</i>	true
<i>cscfExtNetSelOnEnumPstn</i>	All
<i>icscfTransitEnabled</i>	true
<i>tcscfBehavior</i>	disabled
<i>icscfSipToSipTransitEnabled</i>	false

The *cscfEnumForLocalNumbersEnabled* is used to determine if ENUM is to be started when a local number is returned from the Number Normalization. When set to `true`, even local numbers returned from Number Normalization start an ENUM query. When set to `false`, the ENUM query is suppressed when a local number is returned from Number Normalization.

The *cscfExtNetSelOnRnBeforeEnum* is used to determine if an ENUM query is to be suppressed and the ENS is to be started when a Routing Number (RN) attribute is received in the Request URI by the S-CSCF. When set to `true`, the ENUM query is suppressed and the ENS is started. This is typically used for operators that use advanced routing plans to integrate with PSTN networks where modifications of RN are utilized. When set to `false`, the ENUM query is performed and the further routing depends on the ENUM response.

The *cscfExtNetSelOnEnumPstn* is used to determine for which ENUM subservices that ENS is to be started when an ENUM query results in E2U+pstn. When set to `All`, ENS is started for both the ENUM subservices E2U+pstn:tel and E2U+pstn:sip. This is typically used for operators that provision telephone users as SIP subscribers. When set to `tel`, an E2U+pstn:tel call starts ENS, while an E2U+pstn:sip call only starts ENS if the domain is known and user portion in the Request URI includes a telephone number. Other E2U+pstn:sip calls are routed on domain.

The *icscfTransitEnabled* is used to enable the possibility to transit calls at the terminating I-CSCF, when no subscriber is found in the HSS. In this case, the I-CSCF starts ENS, and the configuration of that analysis determine if the call actually breaks out.

tcscfBehavior is set to `ENABLED` in network deployments where most incoming sessions to the node is expected to be transited. *TcscfBehavior* must only be set to `ENABLED` when the CSCF is deployed as a standalone I-CSCF (*cscfISPBbehavior* = `I_CSCF`). Terminating sessions are routed to another I-CSCF where *tcscfBehavior* is not enabled. It is also recommended to have Unallocated Routing feature disabled when *TcscfBehavior* is set to `ENABLED`, to avoid making any needless subsequent HSS Lookups.

icscfSipToSipTransitEnabled enables the possibility for the I-CSCF to transit SIP requests with a target that has a non-telephone number, also referred to IMS Transit (SIP > SIP). I-CSCF transits these requests using DNS resolution. If *icscfSipToSipTransitEnabled* is disabled, the call is rejected and the “User Not Found” response is returned.

3.4.2 Configure Generic Number Portability

To enable the Number Portability All Call Query scheme, that is, Number Portability lookup performed on all telephone numbers before routing, attribute *cscfGenericNumberPortabilityEnabled* must be set to **TRUE**.

To enable the Number Portability Query on Release scheme, both attributes *cscfGenericNumberPortabilityEnabled* and *scscfNpQueryOnReleaseEnabled*, must be set to **TRUE**. The ENS



analysis must also set an `extNetSelEosCase` which invokes an End of Selection analysis, where the applicable release responses result in a `cscfEosCaseRouteName = "enum"` which triggers a Number Portability lookup.

To enable the Number Portability onward routing scheme, that is, Number Portability lookup performed by the terminating I-CSCF node, both attributes `cscfGenericNumberPortabilityEnabled` and `cscfOnwardRoutingEnabled`, must be set to **TRUE**.

A telephone number that has been ported is routed according to a Routing Number. To route the call to the appropriate gateway, the RN information is configured in the ENS RN tables.

This is configured through the following configuration attributes:

- `extNetSelRnTable`
- `extNetSelRnTableEntry`
- `extNetSelRnTablePoolName`
- `extNetSelRnTableStartRange`

For further information about the ENS, see Section 3.5 Configure External Network Selection on page 38.

3.4.3 Configure Carrier Routing

The Carrier Identification Code (CIC) routing feature is controlled by configuration attribute `cscfCicRoutingEnabled`.

If the network has an AS deployed that is configured for the Carrier Select feature, then the `cscfCicRoutingEnabled` is to be set to **ENABLED**. If the Carrier Select feature is not deployed but the ENUM database uses the CIC attribute, then the `cscfCicRoutingEnabled` attribute is to be set to **ENUM**. A CIC is used to route the call. To route the call to the appropriate gateway, the CIC information is configured in the ENS CIC tables.

This is configured through the following configuration attributes:

- `extNetSelCicTable`
- `extNetSelCicTableEntry`
- `extNetSelCicTablePoolName`
- `extNetSelCicTableStartRange`

The CIC table configuration option “ignore” has been deprecated. The ENS could be configured in the following way to get similar functionality:

- The Calling Party Table Entry includes a CIC table name.
- The selected CIC table contains a next table entry to a Request-URI table.
- The selected Request-URI table includes a regular expression that removes the CIC-parameter from the Request-URI. The next table entry in the selected Request-URI table point at the Calling Party Table where the analysis started.

For further information about the ENS, see Section 3.5 Configure External Network Selection on page 38.

3.5 Configure External Network Selection

This section describes how to configure and manage the ENS.

The CSCF starts the ENS analysis in the Breakout Gateway Control Function (BGCF) when the Request-URI contains a telephone number. The telephone number is either included in a tel URI or a SIP URI with a known domain (known domains are configured in the *cscfDomainAlias*). Selection of a routing destination can be based on few selection criteria, for example, Calling Party Domain, Calling Party Number, CIC, RN, Called Party Number, request Media Type, SIP Method, Request-URI, SIP headers, UTC time and date information, and weight-based selection. It is possible to modify the Request-URI by defining optional trunk-context and trunk group fields in the selected pool, or by regular expressions. It is also possible per pool, to specify up to four Route-URIs to be included in the outgoing request as additional Route headers. The modified Request-URI is used in further ENS analysis and sent out to the external gateway.

For more information regarding the configuration, refer to *Managed Object Model (MOM)*.

The ENS is configured using the available NBI with *CscfFunction* fragment *extNetSel-Application*, see Section 2.2 Configuration Methods on page 4.

3.5.1 Configure Global External Network Selection

It is possible to use either one or two instances of the ENS configuration, with *CscfFunction* fragment *extNetSel-Application*. Both instances have the same set of classes and attributes. ENS runs in “single configuration mode” if attribute *extNetSelectionInitialTableName* is set to *not_configured* in either of the applications. Setting *extNetSelectionInitialTableName* to *not_configured* in both applications is not allowed. An example of how to configure the global ENS configuration to enable single configuration mode is shown in Table 21.

ENS runs in “dual configuration mode” if both applications have *extNetSelectionInitialTableName* set to a valid table instance. For



an example of how to configure the global ENS configuration to enable dual configuration mode, see Table 22.

Attribute *extNetSelectionMaxTables* specifies the maximum number of navigation and matching tables allowed to be started for routing analyses. This attribute is used for ENS loop detection to avoid infinite table looping. When the number of navigation and matching tables started in an ENS session exceeds this limit, an alarm is raised and the SIP request is routed to the default network pool table entry. If the default network pool table entry is not defined, the SIP request is rejected with a SIP 500 (PSTN Gateway Unreachable) response. This attribute is included in both ENS applications. A change of the value in one application is mirrored in the attribute in the other application.

Attribute *extNetSelectionActiveConfiguration* specifies the active application (*ExtNetSelection* or *ExtNetSelection2*) when running in dual configuration mode and the other application is implicitly passive. Configuration updates are only allowed in the passive application and traffic always uses the active application. In the example shown in Table 22, all configuration updates are made in application *extNetSelection2* and traffic uses application *ExtNetSelection*. It is run in dual configuration mode so that updates to the configuration can be made without traffic disturbances.

Single configuration mode implies that the configuration instance with *extNetSelectionInitialTableName* set to *not_configured* is not to be used. Configuration changes can thus only be made on the active configuration, which is also used for traffic handling. In the example shown in Table 21, all configuration changes are made in application *ExtNetSelection*. Single configuration mode must be avoided when having large number of tables configured. Configuration changes in single configuration mode must be avoided in peak traffic hours. Configuration changes made during peak traffic hours while having large number of tables defined can result in delayed or, in worst case, failed breakout sessions.

Attribute *extNetSelectionDefaultPoolName* identifies the default network pool table entry to use when the selection algorithm does not end up in a specific entry in the network pool table. It is also used if a configuration fault has been detected in single configuration mode. For example, a configuration fault can be having an initial table without any table entries.

Attribute *extNetSelectionTablesSynchronization* is used to signal that the modifications to the ENS configuration have been completed and that it is prepared to be put in service.

Read-only attribute *extNetSelectionSynchronizationState* contains one of the values *synchronized*, *not_synchronized*, and *synchronization_failed*. The value *synchronized* indicates that the ENS configuration data has been successfully read and that it is prepared for traffic handling. Value *not_synchronized* indicates that the ENS configuration data has been updated without a consecutive synchronization. Value *synchronization_failed* indicates that a synchronization attempt of the ENS configuration data has failed.

Table 21 *ENS Global Configuration in Single Configuration Mode*

extNetSelection Global Configuration	
Attribute	Value Example
<i>extNetSelectionInitialTableName</i>	sdp-media:multimedia
<i>extNetSelectionDefaultPoolName</i>	500PstnGatewayUnreachable
<i>extNetSelectionMaxTables</i>	25
<i>ExtNetSelectionActiveConfiguration</i>	ExtNetSelection
<i>extNetSelectionTablesSynchronization</i>	false
<i>extNetSelectionSynchronizationState</i>	synchronized (read-only)
extNetSelection2 Global Configuration	
<i>extNetSelectionInitialTableName</i>	not_configured
<i>extNetSelectionDefaultPoolName</i>	<empty>
<i>extNetSelectionMaxTables</i>	25
<i>extNetSelectionActiveConfiguration</i>	ExtNetSelection
<i>extNetSelectionTablesSynchronization</i>	false
<i>extNetSelectionSynchronizationState</i>	not_synchronized (read-only)

Table 22 *ENS Global Configuration in Dual Configuration Mode*

extNetSelection Global Configuration	
Attribute	Value Example
<i>extNetSelectionInitialTableName</i>	calling:default
<i>extNetSelectionDefaultPoolName</i>	500PstnGatewayUnreachable
<i>extNetSelectionMaxTables</i>	25
<i>extNetSelectionActiveConfiguration</i>	ExtNetSelection
<i>extNetSelectionTablesSynchronization</i>	false



extNetSelection Global Configuration	
Attribute	Value Example
<i>extNetSelectionSynchronizationState</i>	synchronized (read-only)
extNetSelection2 Global Configuration	
<i>extNetSelectionInitialTableName</i>	sdp-media:multimedia
<i>extNetSelectionDefaultPoolName</i>	500PstnGatewayUnreachable
<i>extNetSelectionMaxTables</i>	25
<i>extNetSelectionActiveConfiguration</i>	ExtNetSelection
<i>extNetSelectionTablesSynchronization</i>	false
<i>extNetSelectionSynchronizationState</i>	synchronized (read-only)

3.5.2 External Network Selection Table Configuration Examples

The following sections provide examples of how the configuration tables can be populated.

3.5.2.1 Configuration Example 1

This section provides an example of how the configuration tables can be populated, as shown in Table 23 through Table 36. CIC and RN tables are omitted from the example for simplicity.

Configuration tables in this example are either placed in the active configuration instance if single configuration is used, or in the passive configuration instance if dual configuration is used, see Section 3.5.1 Configure Global External Network Selection on page 38. The global configuration is shown in Table 21 for single configuration and in Table 22 for dual configuration. The configuration changes are in this example made to `ExtNetSelection` if single configuration and to `ExtNetSelection2` if dual configuration. The configuration is made active by following the procedure described in Section 3.5.3 Manage External Network Selection in Single Configuration Mode on page 57 (single configuration) or Section 3.5.4 Manage External Network Selection in Dual Configuration Mode on page 58 (dual configuration).

When configuring the ENS, the tables must be created last one first to ensure that a table exists before it is referenced. For example; to create table `sdp-media:multimedia`, the following tables must be created first; `pool:mgcD` and `P-Asserted-Identity:sweden`.

In this configuration example, the selection process starts with the `sdp-media:multimedia` table (Table 23). To create this table, pool `mgcD` (Table 30) and the `P-Asserted-Identity:sweden` table (Table 24) must be created first.

In this configuration example, the selection process starts with the `sdp-media:multimedia` table as defined by attribute `ExtNetSelectionInitialTableName` (Table 21 or Table 22).

The `500PstnGatewayUnreachable` entry is defined as the default network pool to use if there is a configuration error. This refers to an unallocated number pool. If selected, this leads to a SIP 500 PSTN Gateway Unreachable response.

All sessions requiring video or audio media use `mgcD`. For calls requiring other media types, the selection continues using the `P-Asserted-Identity:sweden` table. All calls initiated by users from the `telia.com` domain are sent to the `mgcD` pool. For calls from other users, the selection continues using the default Calling Party Number table (Table 25). The RN and CIC table names are not specified in this example; this means that a potential CIC or RN input parameter is not used in this table.

For calling users from the +468719 area, the selection continues using the Ericsson Called Party Number table (Table 27). The Ericsson Called Party Number table leads to `mgcC` if the user dialed a number starting with 852, otherwise it uses the `ericsson:default` entry, which leads to `node2node3mgcB`.

For all other callers, the default Called Party Number table (Table 26) is used, which leads to `mgcCDE` if the dialed number starts with +10065 and to a SIP-method table (Table 31) for all other numbers (except 112).

If the called number is 112, a routing number is to be added to route the call to a PSAP. This is achieved by letting 112 lead to `Request-URI:addRnPsap:default` (Table 28), which refers to `regexAddRnPsap` (Table 29) that modifies the Request-URI by adding an RN before leading to `mgcCDE`. The `mgcCDE` entry contains a prioritized list of `ExtNetSelPoolURIs`. The alternative Pool URIs are used by the S-CSCF when doing emergency call routing.

The `SIP-Method:sweden` table (Table 31) is started for calling users not from the +468719 area and the dialed number not starting with +10065 or not being 112. In this case table navigation goes through the `SIP-Method:sweden` table (Table 31), the `Event:sweden` table (Table 32), the `P-Access-Network-Info:sweden` table (Table 33), the `Request-URI-Match:pub` table (Table 34) and the `Request-URI-Match:non-pub` table (Table 35) using the `ExtNetSelRegexMTbl` supporting table (Table 36). Table navigation leads to `mgcC` if the SIP request is a SUBSCRIBE from an xDSL caller to a dialog event, and the Request-URI contains domain string `one.net` (except `pub.one.net`). Otherwise, the navigation result in `node1mgcA`.



Note: Each Request-URI-Match table can be configured with at most one explicit match entry only to enforce ordered domain string matches such as in this example.

Table 23 *ExtNetSelTable=sdp-media:multimedia*

extNetSelTable=P-Asserted-Identity:sweden	
Attribute	Value Example
<i>extNetSelTableEntry</i>	sdp-media:multimedia
<i>extNetSelTableMatchOperand</i>	video
<i>extNetSelTableMatchOperand</i>	audio
<i>extNetSelTableResult</i>	pool:mgcD
<i>extNetSelTableEntry</i>	sdp-media:multimedia:default
<i>extNetSelTableResult</i>	P-Asserted-Identity:sweden

Table 24 *ExtNetSelTable=P-Asserted-Identity:sweden*

extNetSelTable=P-Asserted-Identity:sweden	
Attribute	Value Example
<i>extNetSelTableEntry</i>	P-Asserted-Identity:sweden:telia.com
<i>extNetSelTableMatchOperation</i>	match-exact-domainname
<i>extNetSelTableResult</i>	pool:mgcD
<i>extNetSelTableEntry</i>	P-Asserted-Identity:sweden:default
<i>extNetSelTableMatchOperation</i>	match-exact-domainname
<i>extNetSelTableResult</i>	calling:default

Table 25 *ExtNetSelCallingTable=default*

extNetSelCallingTable=default	
Attribute	Value Example
<i>extNetSelCallingTableEntry</i>	default:default
<i>extNetSelCallingTableStartRange</i>	default
<i>extNetSelCalledPartyTableName</i>	<empty>
<i>extNetSelCicTableNextTableName</i>	<empty>
<i>extNetSelRnTableNextTableName</i>	<empty>

<i>extNetSelNextTableName</i>	called:default
<i>extNetSelCallingTableEntry</i>	default:+468719
<i>extNetSelCallingTableStartRange</i>	+468719
<i>extNetSelCalledPartyTableName</i>	<empty>
<i>extNetSelCicTableNextTableName</i>	<empty>
<i>extNetSelRnTableNextTableName</i>	<empty>
<i>extNetSelNextTableName</i>	called:ericsson

Table 26 *ExtNetSelCalledTable=default*

extNetSelCalledTable=default	
Attribute	Value Example
<i>extNetSelCalledTableEntry</i>	default:default
<i>extNetSelCalledTableStartRange</i>	default
<i>extNetSelCalledTablePoolName</i>	<empty>
<i>extNetSelCalledTableNextTableName</i>	SIP-Method:sweden
<i>extNetSelCalledTableEntry</i>	default:112
<i>extNetSelCalledTablePoolName</i>	<empty>
<i>extNetSelCalledTableNextTableName</i>	Request-URI:addRnPsap:default
<i>extNetSelCalledTableEntry</i>	default:+10065
<i>extNetSelCalledTablePoolName</i>	mgcCDE
<i>extNetSelCalledTableNextTableName</i>	<empty>

Table 27 *ExtNetSelCalledTable=ericsson*

extNetSelCalledTable=ericsson	
Attribute	Value Example
<i>extNetSelCalledTableEntry</i>	ericsson:default
<i>extNetSelCalledTablePoolName</i>	node2node3mgcB
<i>extNetSelCalledTableNextTableName</i>	<empty>



<i>extNetSelCallingTableEntry</i>	ericsson:852
<i>extNetSelCalledTablePoolName</i>	mgcC
<i>extNetSelCalledTableNextTableName</i>	<empty>

Table 28 *ExtNetSelTable=Request-URI:addRnPsap*

extNetSelTable=Request-URI:addRnPsap	
Attribute	Value Example
<i>extNetSelTableEntry</i>	Request-URI:addRnPsap:default
<i>extNetSelTableMatchOperation</i>	<empty>
<i>extNetSelTableModifyOperation</i>	modify-regex
<i>extNetSelTableModifyOperand</i>	regexAddRnPsap
<i>extNetSelTableResult</i>	pool:mgcCDE

Table 29 *ExtNetSelRegexTable*

extNetSelRegexTable	
Attribute	Value Example
<i>extNetSelRegexTableEntry</i>	regexAddRnPsap
<i>extNetSelRegExpression</i>	/^(sip:.+)(@.+)\$/;rn=\+496967890\2/
<i>extNetSelRegexTableEntry</i>	regexRemoveRn
<i>extNetSelRegExpression</i>	/;rn=[^;@]*//
<i>extNetSelRegexTableEntry</i>	regexExtractCellIdMcc
<i>extNetSelRegExpression</i>	/^(3GPP.*)((utran-cell-id-3gpp[]*=[]*) ([0-9][0-9])(.)*\$/^3/

Table 30 *ExtNetSelPool*

extNetSelPool	
Attribute	Value Example
<i>extNetSelPoolTableEntry</i>	node1mgcA
<i>extNetSelPoolMode</i>	0
<i>extNetSelPoolURI</i>	1:sip:node1.net
<i>extNetSelAdditionalRoute</i>	1:sip:mgcA.net;lr



<i>extNetSelEosCase</i>	node1mgcAFail
<i>extNetSelPoolTimeOut</i>	2000
<i>extNetSelTrunkGroupAndContext</i>	tg3-1mgcA.net
<i>extNetSelUnallocatedNrResponseCode</i>	<empty>
<i>extNetSelUnallocatedNrResponsePhrase</i>	<empty>
<i>extNetSelPoolTableEntry</i>	mgcCDE
<i>extNetSelPoolMode</i>	0
<i>extNetSelPoolURI</i>	1:sip:mgcC.net
<i>extNetSelPoolURI</i>	3:sip:mgcD.net
<i>extNetSelPoolURI</i>	4:sip:mgcE.net
<i>extNetSelEosCase</i>	defaultFail
<i>extNetSelPoolTimeOut</i>	2000
<i>extNetSelTrunkGroupAndContext</i>	tg3-1mgcC.net
<i>extNetSelUnallocatedNrResponseCode</i>	<empty>
<i>extNetSelUnallocatedNrResponsePhrase</i>	<empty>
<i>extNetSelPoolTableEntry</i>	node2node3mgcB
<i>extNetSelPoolMode</i>	0
<i>extNetSelPoolURI</i>	1:sip:10.50.10.2:5060;lr
<i>extNetSelAdditionalRoute</i>	9:sip:mgcB.net:5060;lr
<i>extNetSelAdditionalRoute</i>	8:sip:10.50.10.3;lr
<i>extNetSelEosCase</i>	node2node3mgcBFail
<i>extNetSelPoolTimeOut</i>	2000
<i>extNetSelTrunkGroupAndContext</i>	tg3-1mgcB.net
<i>extNetSelUnallocatedNrResponseCode</i>	<empty>
<i>extNetSelUnallocatedNrResponsePhrase</i>	<empty>
<i>extNetSelPoolTableEntry</i>	mgcC
<i>extNetSelPoolMode</i>	0
<i>extNetSelPoolURI</i>	1:sip:mgcC.net



<i>extNetSelEosCase</i>	defaultFail
<i>extNetSelPoolTimeOut</i>	2000
<i>extNetSelTrunkGroupAndContext</i>	tg3-1mgcC.net
<i>extNetSelUnallocatedNrResponseCode</i>	<empty>
<i>extNetSelUnallocatedNrResponsePhrase</i>	<empty>
<i>extNetSelPoolTableEntry</i>	mgcD
<i>extNetSelPoolMode</i>	0
<i>extNetSelPoolURI</i>	1:sip:mgcD.net
<i>extNetSelEosCase</i>	<empty>
<i>extNetSelPoolTimeOut</i>	2000
<i>extNetSelTrunkGroupAndContext</i>	tg3-1mgcD.net
<i>extNetSelUnallocatedNrResponseCode</i>	<empty>
<i>extNetSelUnallocatedNrResponsePhrase</i>	<empty>
<i>extNetSelPoolTableEntry</i>	500PstnGatewayUnreachable
<i>extNetSelPoolMode</i>	1
<i>extNetSelPoolURI</i>	<empty>
<i>extNetSelEosCase</i>	<empty>
<i>extNetSelPoolTimeOut</i>	<empty>
<i>extNetSelTrunkGroupAndContext</i>	<empty>
<i>extNetSelUnallocatedNrResponseCode</i>	500
<i>extNetSelUnallocatedNrResponsePhrase</i>	"PSTN Gateway Unreachable"

Table 31 *ExtNetSelTable=SIP-Method:sweden*

extNetSelTable=SIP-Method:sweden	
Attribute	Value Example
<i>extNetSelTableEntry</i>	SIP-Method:sweden:INVITE
<i>extNetSelTableMatchOperation</i>	match-method
<i>extNetSelTableMatchOperand</i>	<empty>

<i>extNetSelTableResult</i>	pool:node1mgcA
<i>extNetSelTableEntry</i>	SIP-Method:sweden:SUBSCRIBE
<i>extNetSelTableMatchOperation</i>	match-method
<i>extNetSelTableMatchOperand</i>	<empty>
<i>extNetSelTableResult</i>	Event:sweden
<i>extNetSelTableEntry</i>	SIP-Method:sweden:default
<i>extNetSelTableMatchOperation</i>	<empty>
<i>extNetSelTableMatchOperand</i>	<empty>
<i>extNetSelTableResult</i>	pool:node1mgcA

Table 32 *ExtNetSelTable=Event:sweden*

extNetSelTable=Event:sweden	
Attribute	Value Example
<i>extNetSelTableEntry</i>	Event:sweden
<i>extNetSelTableMatchOperation</i>	match-regex
<i>extNetSelTableMatchOperand</i>	eventIsDialog
<i>extNetSelTableResult</i>	P-Access-Network-Info:sweden
<i>extNetSelTableEntry</i>	Event:sweden:OperandNotExist
<i>extNetSelTableMatchOperation</i>	<empty>
<i>extNetSelTableMatchOperand</i>	<empty>
<i>extNetSelTableResult</i>	pool:node1mgcA
<i>extNetSelTableEntry</i>	Event:sweden:default
<i>extNetSelTableMatchOperation</i>	<empty>
<i>extNetSelTableMatchOperand</i>	<empty>
<i>extNetSelTableResult</i>	pool:node1mgcA

Table 33 *ExtNetSelTable=P-Access-Network-Info:sweden*

extNetSelTable=P-Access-Network-Info:sweden	
Attribute	Value Example
<i>extNetSelTableEntry</i>	P-Access-Network-Info:sweden
<i>extNetSelTableMatchOperation</i>	match-regex
<i>extNetSelTableMatchOperand</i>	panilsDSL
<i>extNetSelTableResult</i>	Request-URI-Match:pub
<i>extNetSelTableEntry</i>	P-Access-Network-Info:sweden:OperandNotExist
<i>extNetSelTableMatchOperation</i>	<empty>
<i>extNetSelTableMatchOperand</i>	<empty>
<i>extNetSelTableResult</i>	pool:node1mgcA
<i>extNetSelTableEntry</i>	P-Access-Network-Info:sweden:default
<i>extNetSelTableMatchOperation</i>	<empty>
<i>extNetSelTableMatchOperand</i>	<empty>
<i>extNetSelTableResult</i>	pool:node1mgcA

Table 34 *ExtNetSelTable=Request-URI-Match:pub*

extNetSelTable=Request-URI-Match:pub	
Attribute	Value Example
<i>extNetSelTableEntry</i>	Request-URI-Match:pub
<i>extNetSelTableMatchOperation</i>	match-regex
<i>extNetSelTableMatchOperand</i>	pubURI
<i>extNetSelTableResult</i>	pool:node1mgcA
<i>extNetSelTableEntry</i>	Request-URI-Match:pub:default
<i>extNetSelTableMatchOperation</i>	<empty>
<i>extNetSelTableMatchOperand</i>	<empty>
<i>extNetSelTableResult</i>	Request-URI-Match:non-pub

Table 35 *ExtNetSelTable=Request-URI-Match:non-pub*

extNetSelTable=Request-URI-Match:non-pub	
Attribute	Value Example
<i>extNetSelTableEntry</i>	Request-URI-Match:non-pub
<i>extNetSelTableMatchOperation</i>	match-regex
<i>extNetSelTableMatchOperand</i>	nonPubURI
<i>extNetSelTableResult</i>	pool:mgcC
<i>extNetSelTableEntry</i>	Request-URI-Match:non-pub:default
<i>extNetSelTableMatchOperation</i>	<empty>
<i>extNetSelTableMatchOperand</i>	<empty>
<i>extNetSelTableResult</i>	pool:node1mgcA

Table 36 *extNetSelRegexMTbl*

extNetSelRegexMTbl	
Attribute	Value Example
<i>extNetSelRegexMTblEntry</i>	eventIsDialog
<i>extNetSelRegexM</i>	/^dialog.*i
<i>extNetSelRegexMTblEntry</i>	panIsDSL
<i>extNetSelRegexM</i>	/^.*DSL\$/
<i>extNetSelRegexMTblEntry</i>	pubURI
<i>extNetSelRegexM</i>	/pub.one.net/i
<i>extNetSelRegexMTblEntry</i>	nonPubURI
<i>extNetSelRegexM</i>	/one.net/i

Note: The SIP request is routed to the destination specified in *ExtNetSelPoolUri*. If *ExtNetSelPoolUri* specifies “lr”, the URI is added as the top Route header in the outgoing request. Any configured additional Route-URIs (*ExtNetSelAdditionalRoute*) are added as additional Route headers, according to the order indicator. For more information about the parameters, refer to *Managed Object Model (MOM)*.



The use of different Routing Alternatives, including configured Route-URIs and additional Route-URIs influence the routing of SIP requests, as shown in Figure 7.

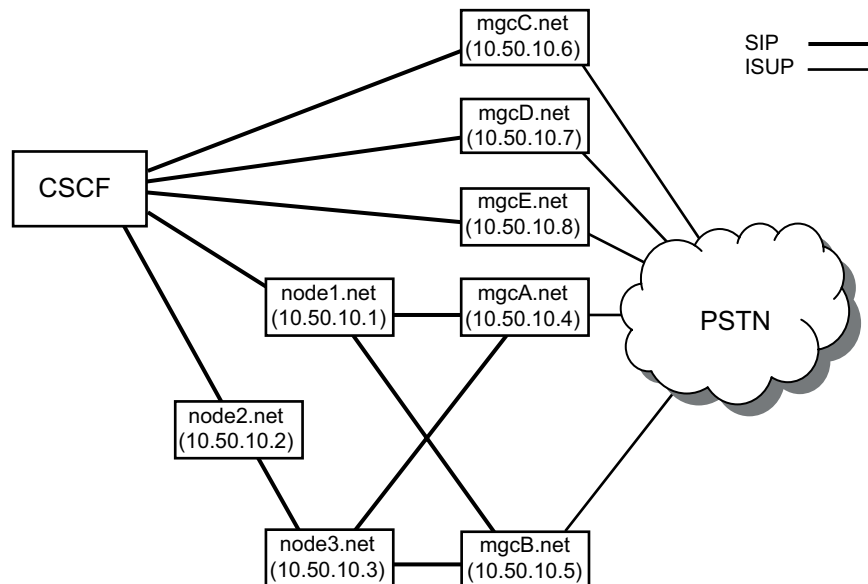


Figure 7 Network View of ENS and End-Of-Selection Routing Configurations

3.5.2.2

Configuration Example 2

This section provides an example of how the configuration tables can be populated to implement call permission services with time scheduled support, as shown in Table 37 through Table 40. The *ExtNetSelRegexTable* and the *Pool* Table in Section 3.5.2.1 Configuration Example 1 on page 41 are reused in this example.

In this configuration example, calls initiated from accesses with matching Mobile Country Code (MCC) values are routed to different target addresses before and after the call permission service is activated. Calls initiated from accesses not identified with the configured MCC values are routed to another target address.

Note: MCC is the first two digits of the “*utran-cell-id-3gpp*” parameter in the P-Access-Network-Info (PANI) header. For more information, refer to [3GPP TS 24.229](#).

In this example, call permission service is activated at 12:00:00 on 2013-05-01 UTC time.

The following target addresses are used:

- `Pool:node1mgcA`

MCC is defined in the PANI header, and it matches an ENS configured value. Call permission is activated.

- `Pool:mgcC`

MCC is defined in the PANI header, and it matches an ENS configured value. Call permission is not yet activated.

- `Pool:mgcD`

MCC is not defined in the PANI header, or it does not match any of the ENS configured values.

The selection process starts with the `SipMessage:Sweden` table (Table 37) as defined by attribute `ExtNetSelectionInitialTableName` (Table 21 or Table 22).

Note: For this example, attribute `ExtNetSelectionInitialTableName` has value `SipMessage:Sweden` for application `ExtNetSelection` in table ENS Global Configuration in single configuration mode (Table 21) and for both applications in Table 22.

The `SipMessage:Sweden` table is configured to perform routing analysis on the PANI SIP header. If the header is present in the SIP request and regular expression evaluation “`regexExtractCellIdMcc`” in the `ExtNetSelRegexTable` is successful, the first two digits of the `utran-cell-id-3gpp` parameter are extracted as a dynamic input string value to the next table, `EnsExactString:CellIdMcc`. If the header is not present, or if the regular expression evaluation fails, the pool address in `pool:mgcD` is returned.

The `EnsExactString:CellIdMcc` table (Table 38) is started with the MCC dynamic input string value from the `SipMessage:Sweden` table. If the MCC string value exactly matches anyone of the configured string values, in this case “23” only, the UTC date match table `EnsDateMatchTable:Sweden` is started. The pool address in `pool:mgcD` is returned for all other MCC values

Attribute `extNetSelMatchTableMode` in the Exact String Match table, `EnsExactString:CellIdMcc`, configures if the exact string match operation is case-sensitive or case-insensitive. It is of an enumerated string type with a value `CASE_SENSITIVE` or `CASE_INSENSITIVE`.

Input to the UTC time and date tables is generated by the ENS. The UTC Date Match table, `EnsDateMatchTable:Sweden` (Table 39), returns the pool address in `pool:node1mgcA` if the UTC date falls after the call permission activation date 2013-05-01, in entry “DaysAfter”.



Note: The stop operand “2099-12-31” is the end date of the call permission service. It is arbitrarily set in this example.

Analysis leads to the UTC Time Match table `EnsTimeMatchTable:Sweden` to verify the activation time further if the call is received on the call permission activation date. The UTC Date Match table returns the pool address in `pool:mgcC` otherwise because the UTC date falls outside the configured date ranges.

The UTC Time Match table `EnsTimeMatchTable:Sweden` (Table 40) returns the pool address in `pool:node1mgcA` if the UTC time falls in the range in entry “Enabled”. Otherwise, the table returns the pool address in `pool:mgcC`.

Table 37 `extNetSelSipMessageTableId=Sweden`

extNetSelSipMessageTableId=Sweden	
Attribute	Value Example
<code>extNetSelSipMessageTableEntryId</code>	Sweden:P-Access-Network-Info
<code>extNetSelSipMessageTableMatchOperation</code>	match-extract
<code>extNetSelSipMessageTableMatchOperand</code>	regexExtractCellIdMcc
<code>extNetSelSipMessageTableResult</code>	EnsExactString:CellIdMcc
<code>extNetSelSipMessageTableEntryId</code>	Sweden:OperandNotExist
<code>extNetSelSipMessageTableResult</code>	pool:mgcD
<code>extNetSelSipMessageTableEntryId</code>	Sweden:default
<code>extNetSelSipMessageTableResult</code>	pool:mgcD

Table 38 `extNetSelMatchTableId=EnsExactString:CellIdMcc`

extNetSelMatchTableId=EnsExactString:CellIdMcc	
Attribute	Value Example
<code>extNetSelMatchTableMode</code>	CASE_SENSITIVE
<code>extNetSelMatchTableEntryId</code>	EnsExactString:CellIdMcc:23

<i>extNetSelMatchTableResult</i>	EnsDateMatchTable:Sweden
<i>extNetSelMatchTableEntryId</i>	EnsExactString:CellIdMcc:default
<i>extNetSelMatchTableResult</i>	pool:mgcD

Table 39 *extNetSelUtcTableId=EnsDateMatchTable:Sweden*

extNetSelUtcTableId=EnsDateMatchTable:Sweden	
Attribute	Value Example
<i>extNetSelUtcTableEntryId</i>	EnsDateMatchTable:Sweden:Day 1
<i>extNetSelUtcTableStartOperand</i>	2013-05-01
<i>extNetSelUtcTableStopOperand</i>	2013-05-01
<i>extNetSelUtcTableResult</i>	EnsTimeMatchTable:Sweden
<i>extNetSelUtcTableEntryId</i>	EnsDateMatchTable:Sweden:DaysAfter
<i>extNetSelUtcTableStartOperand</i>	2013-05-02
<i>extNetSelUtcTableStopOperand</i>	2099-12-31
<i>extNetSelUtcTableResult</i>	pool:node1mgcA
<i>extNetSelUtcTableEntryId</i>	EnsDateMatchTable:Sweden:default
<i>extNetSelUtcTableResult</i>	pool:mgcC

Table 40 *extNetSelUtcTableId=EnsTimeMatchTable:Sweden*

extNetSelUtcTableId=EnsTimeMatchTable:Sweden	
Attribute	Value Example
<i>extNetSelUtcTableEntryId</i>	EnsTimeMatchTable:Sweden:Disabled ⁽¹⁾
<i>extNetSelUtcTableStartOperand</i>	00:00:00
<i>extNetSelUtcTableStopOperand</i>	11:59:59
<i>extNetSelUtcTableResult</i>	pool:mgcC
<i>extNetSelUtcTableEntryId</i>	EnsTimeMatchTable:Sweden:Enabled
<i>extNetSelUtcTableStartOperand</i>	12:00:00
<i>extNetSelUtcTableStopOperand</i>	23:59:59



<code>extNetSelUtcTableResult</code>	<code>pool:node1mgcA</code>
<code>extNetSelUtcTableEntryId</code>	<code>EnsTimeMatchTable:Sweden:default</code>
<code>extNetSelUtcTableResult</code>	<code>pool:mgcC</code>

(1) This entry is redundant in this example because the default entry gives the same result. It is included for illustration purposes.

3.5.2.3 Configuration Example 3

This section provides an example of how the configuration tables can be populated to implement weight-based routing.

Table 41 is an example where called numbers to Sweden are to be analyzed further using a weight-based formula according to table `weight:sweden` and called numbers to Norway are to be analyzed further using a weight-based formula according to table `weight:norway`.

Table 42 is an example where the weight routing is configured to reflect the weights in percentages directly. The outcome of the example is that 75% of the transactions select table `Request-URI:addCicC1:default` and 25% of the transactions select table `Request-URI:addCicC2:default`. The value range of the weight part is 0–65535, though the use of a direct percentage value is at the discretion of the operator.

Table 43 shows another way where the transactions are distributed as follows:

- 25% select `Request-URI:addCicC1:default`.
- 25% select `Request-URI:addCicC2:default`.
- 50% select `Request-URI:addCicC3:default`.
- 0% select `Request-URI:addCicC4:default`.

For example, `extNetSelWeightEntry` with the value `0:Request-URI:addCicC4:default` can be a placeholder and be modified to `400:Request-URI:addCicC4:default`. The modification changes distributions after synchronization as follows:

- 12.5% select `Request-URI:addCicC1:default`
- 12.5% select `Request-URI:addCicC2:default`
- 25% select `Request-URI:addCicC3:default`
- 50% select `Request-URI:addCicC4:default`

Table 44 shows an example where the request-URI is modified by a regular expression in Table 45 to add the URI parameter `cic=+9975` and then

the transaction is routed to the pool:mgcCDE. Request-URI tables and *ExtNetSelRegexTableEntry* for C2, C3, and C4 are not stated but can be similar to *ExtNetSelRegexTableEntry* for C1.

Table 41 Example *ExtNetSelCalledTable=nordic*

ExtNetSelCalledTable=nordic	
Attribute	Value Example
<i>ExtNetSelCalledTableEntry</i>	nordic:+46
<i>ExtNetSelCalledTablePoolName</i>	<empty>
<i>ExtNetSelCalledTableNextTableName</i>	weight:sweden
<i>ExtNetSelCalledTableEntry</i>	nordic:+47
<i>ExtNetSelCalledTablePoolName</i>	<empty>
<i>ExtNetSelCalledTableNextTableName</i>	weight:norway

Table 42 Example *ExtNetSelWeightTable=sweden*

ExtNetSelWeightTable=sweden	
Attribute	Value Example
<i>extNetSelWeightEntry</i>	75:Request-URI:addCicC1:default
<i>extNetSelWeightEntry</i>	25:Request-URI:addCicC2:default

Table 43 Example *ExtNetSelWeightTable=norway*

ExtNetSelWeightTable=norway	
Attribute	Value Example
<i>extNetSelWeightEntry</i>	100:Request-URI:addCicC1:default
<i>extNetSelWeightEntry</i>	100:Request-URI:addCicC2:default
<i>extNetSelWeightEntry</i>	200:Request-URI:addCicC3:default
<i>extNetSelWeightEntry</i>	0:Request-URI:addCicC4:default

Table 44 Example *ExtNetSelTable=Request-URI:addCicOp1*

ExtNetSelTable=Request-URI:addCicC1	
Attribute	Value Example
<i>ExtNetSelTableEntry</i>	Request-URI:addCicC1:default



ExtNetSelTable=Request-URI:addCicC1	
Attribute	Value Example
<i>ExtNetSelTableMatchOperation</i>	<empty>
<i>ExtNetSelTableModifyOperation</i>	modify-regex
<i>ExtNetSelTableModifyOperand</i>	regexAddCicC1
<i>ExtNetSelTableResult</i>	pool:mgcCDE

Table 45 Example ExtNetSelRegexTable

ExtNetSelRegexTable	
Attribute	Value Example
<i>ExtNetSelRegexTableEntry</i>	regexAddCicC1
<i>ExtNetSelRegexExpression</i>	/^(.+)(@.+)\$\1;cic=+9975\2/

3.5.3 Manage External Network Selection in Single Configuration Mode

Only one ENS application (*ExtNetSelection* or *ExtNetSelection2*) is used in single configuration mode. Attribute *extNetSelectionInitialTableName* must point to a valid table instance in the active configuration and be set to *not_configured* in the other configuration. *extNetSelectionActiveConfiguration* is set to the application where *extNetSelectionInitialTableName* points to a valid table instance. Parameter *extNetSelectionActiveConfiguration* cannot be modified while being in single configuration mode.

After changes in the ENS table configuration are complete, enable the ENS as follows:

1. Update the configuration with the required changes.
2. Set *extNetSelectionTablesSynchronization* to *true*.
3. Wait until *extNetSelectionTablesSynchronization* has been set to *false* and make sure that *extNetSelectionSynchronizationState* has taken the value *synchronized*.

Attribute *extNetSelectionSynchronizationState* is set to *synchronized* if the configuration is successfully read. Otherwise, if there was a failure while reading the configuration, the value is set to *synchronization_failed* and an alarm is raised. If the failure is caused by a configuration fault, all SIP messages to be routed to an external gateway are routed to the gateway identified in the parameter *ExtNetSelectionDefaultPoolName*, see Section 3.5.1 Configure Global External Network Selection on page 38. If the failure is caused by insufficient

process memory, all SIP messages to be routed to an external gateway are answered by a 500 Service Execution Error.

When `extNetSelectionTablesSynchronization` is set to `true`, no configuration is allowed to be changed in the ENS.

Note: A user cannot set the value to `false`. It is set to `false` when the configuration has been read.

3.5.4 Manage External Network Selection in Dual Configuration Mode

Both ENS applications (`ExtNetSelection` and `ExtNetSelection2`) are used in dual configuration mode. Attribute `extNetSelectionInitialTableName` must point to a valid table instance in both instances. `extNetSelectionActiveConfiguration` is set to either `ExtNetSelection` or `ExtNetSelection2`.

Assume that `ExtNetSelection` is active and changes must be made to the configuration. The following steps are usually to be performed:

1. Compare the currently active configuration (in this case `ExtNetSelection`) to the passive configuration (in this case `ExtNetSelection2`). Update the passive configuration in such a way that it matches the active configuration.
2. Update the passive configuration with the new changes.
3. Set `extNetSelectionTablesSynchronization` to `true` in the passive configuration (`ExtNetSelection2`).
4. Wait until `extNetSelectionTablesSynchronization` has been set to `true` and make sure that `extNetSelectionSynchronizationState` has taken the value `synchronized`.
5. Modify attribute `extNetSelectionActiveConfiguration` to refer to the current passive configuration (in this case `ExtNetSelection2`). This results in the new configuration being used for traffic handling.

Attribute `extNetSelectionSynchronizationState` is set to `synchronized` if the configuration is successfully read. Otherwise, if there was a failure while reading the configuration, the value is set to `synchronization_failed` and an alarm is raised. `extNetSelectionActiveConfiguration` is only allowed to refer to an application that has `extNetSelectionSynchronizationState` set to `synchronized`.

When `extNetSelectionTablesSynchronization` is set to `true`, no configuration is allowed to be changed in the ENS.

Note: A user cannot set the value to `false`. It is set to `false` when the configuration has been read.



3.6 Configure End-Of-Selection Analysis

To configure End-Of-Selection (EOS) Analysis to allow for one potential rerouting request:

1. Configure the maximum number of times EOS Analysis is allowed to be started per call or session, see Section 3.6.2 Maximum EOS Attempts on page 61.
2. Configure at least one Match Profile Table, see Section 3.6.3 Match Profiles on page 61.
3. Configure at least one Routing Alternative, see Section 3.6.4 Routing Alternatives on page 62.
4. Configure the EOS Case, see Section 3.6.5 EOS Cases on page 64.

For more information about the EOS parameters and attributes, refer to *Managed Object Model (MOM)*.

3.6.1 EOS Overview

A basic overview of the EOS functionality when started from a CSCF Application, is shown in Figure 8.

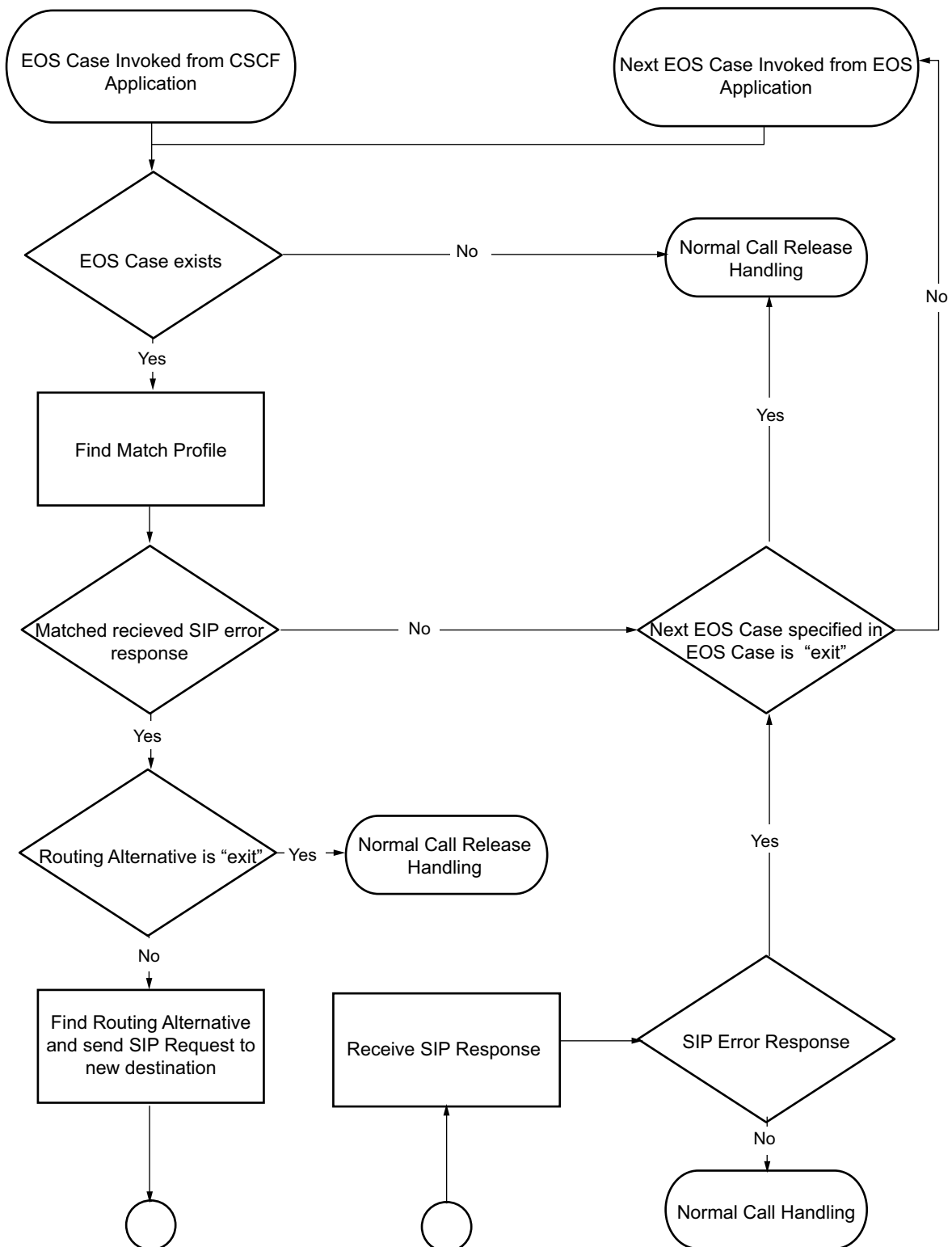


Figure 8 Basic Overview of EOS Analysis after Invocation from CSCF Application



3.6.2 Maximum EOS Attempts

Configure the maximum number of times EOS Analysis is allowed to be started per call or session as shown in Table 46.

Table 46 Configuring Maximum EOS Attempts

applicationName=CscfEos, cscfEosAnalysisId=0	
Attribute	Value Example
<i>cscfEosMaxEosAttempts</i>	2

3.6.3 Match Profiles

Configure few Match Profiles, as shown in Table 47, Table 48, and Table 49, to determine which SIP error codes and which SIP Reason Headers must be received to allow EOS Analysis to continue.

The Match Profile is referenced from one or more EOS Case Table entries.

There are no preconditions to defining Match Profiles.

Table 47 Configuring Match Profile noSipPath

cscfEosMatchProfileTableEntryId = noSipPath	
Attribute	Value Example
<i>cscfEosMatchProfileTableEntryId</i>	noSipPath
<i>cscfEosMatchSipResponseCode</i>	500,503
<i>cscfEosMatchSipReasonHeader</i>	SIP;500, SIP;503

Table 48 Configuring Match Profile nolsupRoute

cscfEosMatchProfileTableEntryId = nolsupPath	
Attribute	Value Example
<i>cscfEosMatchProfileTableEntryId</i>	nolsupPath
<i>cscfEosMatchSipResponseCode</i>	300–699
<i>cscfEosMatchSipReasonHeader</i>	Q.850;31

Table 49 Configuring Match Profile anyErrorResponse

cscfEosMatchProfileTableEntryId = anyErrorResponse	
Attribute	Value Example



cscfEosMatchProfileTableEntryId = anyErrorResponse	
<i>cscfEosMatchProfileTableEntryId</i>	anyErrorResponse
<i>cscfEosMatchSipResponseCode</i>	<empty>
<i>cscfEosMatchSipReasonHeader</i>	<empty>

If a started EOS Case references Match Profile *noSipPath*, the received SIP error response must contain a Status-Line with 500 or 503 and a Reason Header containing *protocol=SIP, cause=500* or *protocol=SIP, cause=503*. If no Reason Header is received in the response, the comparison with the profile results in a non-match.

If a started EOS Case references Match Profile *noIsupPath*, the received SIP error response (including any internally generated error response) must contain a Status-Line between 300 and 699 and a Reason Header containing *protocol=Q.850, cause=31*. If no Reason Header is received in the response, the comparison with the profile results in a match.

If a started EOS Case references Match Profile *anyErrorResponse*, the received SIP error response (including any internally generated error response) can contain any Status-Line and any protocol, and cause, combination in the Reason Header. If no Reason Header is received in the response, the comparison with the profile results in a match.

3.6.4 Routing Alternatives

Configure few Routing Alternatives as shown in Table 50, Table 51, Table 52, and Table 53 specifying which SIP Route to use if the Matching of the SIP error codes and SIP Reason Headers have been successful.

The Route Table is referenced from one or more EOS Case Table entries.

There are no preconditions to defining Routing Alternatives.

Table 50 Configuring Routing Alternative *node2node3mgcA*

cscfEosRouteTableEntryId = node2node3mgcA	
Attribute	Value Example
<i>cscfEosRouteTableEntryId</i>	node2node3mgcA
<i>cscfEosRouteUri</i>	sip:node2.net:5060;lr
<i>cscfEosRouteAdditionalRoute</i>	8:sip:10.50.10.3;lr
<i>cscfEosRouteAdditionalRoute</i>	9:sip:mgcA.net:5060;lr
<i>cscfEosRouteFailoverTimer</i>	12



Table 51 Configuring Routing Alternative node1mgcB

cscfEosRouteTableEntryId = node1mgcB	
Attribute	Value Example
<i>cscfEosRouteTableEntryId</i>	node1mgcB
<i>cscfEosRouteUri</i>	sip:node1.net;lr
<i>cscfEosRouteAdditionalRoute</i>	1:sip:10.50.10.5;lr
<i>cscfEosRouteFailoverTimer</i>	20

Table 52 Configuring Routing Alternative node2node3mgcB

cscfEosRouteTableEntryId = node2node3mgcB	
Attribute	Value Example
<i>cscfEosRouteTableEntryId</i>	node2node3mgcB
<i>cscfEosRouteUri</i>	sip:node2.net:5060;lr
<i>cscfEosRouteAdditionalRoute</i>	1:sip:node3.net;lr
<i>cscfEosRouteAdditionalRoute</i>	2:sip:mgcB.net:5060;lr
<i>cscfEosRouteFailoverTimer</i>	0

Table 53 Configuring Routing Alternative node1mgcA

cscfEosRouteTableEntryId = node1mgcA	
Attribute	Value Example
<i>cscfEosRouteTableEntryId</i>	node1mgcA
<i>cscfEosRouteUri</i>	sip:10.50.10.1
<i>cscfEosRouteAdditionalRoute</i>	7:sip:mgcA.net;lr
<i>cscfEosRouteFailoverTimer</i>	7

If an EOS Case references Routing Alternative “node2node3mgcA”, the SIP request to be rerouted is amended to include the following Route Headers:

- Route: sip:node2.net:5060;lr
- Route: sip:10.50.10.3;lr
- Route: sip:mgcA.net:5060;lr

Assuming the target (*cscfEosRouteUri*) can be resolved by DNS, the SIP request is routed onwards and supervised using the specified *cscfEosRouteFailoverTimer*.

If an EOS Case references Routing Alternative “node1mgcB”, the SIP request to be rerouted is amended to include the following Route Headers:



- Route: sip:node1.net;lr
- Route: sip:10.50.10.5;lr

Assuming the target (*cscfEosRouteUri*) can be resolved by DNS, the SIP request is routed onwards and supervised using the specified *cscfEosRouteFailoverTimer*.

If an EOS Case references Routing Alternative “node2node3mgcB”, the SIP request to be rerouted is amended to include the following Route Headers:

- Route: sip:node2.net:5060;lr
- Route: sip:node3.net;lr
- Route: sip:mgcB.net:5060;lr

Assuming the target (*cscfEosRouteUri*) can be resolved by DNS, the SIP request is routed onwards. The SIP request is supervised by *cscfSipDefaultFailoverTimer* as *cscfEosRouteFailoverTimer* is set to 0.

If an EOS Case references Routing Alternative *node1mgcA*, the SIP request to be rerouted is amended to include the following Route Header:

- Route: sip:mgcA.net;lr

The Route-URI specified in *cscfEosRouteUri* is not included as a Route Header as “lr” is not specified.

Assuming the target (*cscfEosRouteUri*) can be resolved by DNS, the SIP request is routed onwards and supervised using the specified *cscfEosRouteFailoverTimer*.

The use of different Routing Alternatives, including configured Route-URIs and additional Route-URIs influence the routing of SIP requests, as shown in Figure 7.

3.6.5 EOS Cases

The EOS Case is the first point of entry to EOS Analysis. The EOS Case consists of entries for the Match Profile to be used to compare against the received SIP error response, the routing alternative to use (if not “exit”) if the profile matching is successful and the next EOS Case to use (if not “exit”) if the profile matching fails or the specified routing alternative fails.

Configure the EOS Cases as shown in Table 54, Table 55, Table 56, Table 57, and Table 58, specifying the EOS Case Names, of which one or more corresponds to the one configured in the application that is starting EOS Analysis.



In this case, EOS Cases `node1mgcAFail` and `node2node3mgcBFail` have been configured in ENS. EOS Cases `node2node3mgcAFail`, `node1mgcBFail`, and `defaultFail` are started from within EOS.

The preconditions to defining an EOS Case are as follows:

- The required Match Profile must exist in the Match Profile Table.
- The required Routing Alternative must exist in the Routing Alternative Table if the keyword “exit” is not used.

Table 54 Configuring EOS Case node1mgcAFail

cscfEosCaseTableEntryId = node1mgcAFail	
Attribute	Value Example
<i>cscfEosCaseTableEntryId</i>	node1mgcAFail
<i>cscfEosCaseMatchProfileName</i>	noSipPath
<i>cscfEosCaseRouteName</i>	node2node3mgcA
<i>cscfEosNextEosCaseName</i>	node2node3mgcAFail

Table 55 Configuring EOS Case node2node3mgcBFail

cscfEosCaseTableEntryId = node2node3mgcBFail	
Attribute	Value Example
<i>cscfEosCaseTableEntryId</i>	node2node3mgcBFail
<i>cscfEosCaseMatchProfileName</i>	noSipPath
<i>cscfEosCaseRouteName</i>	node1mgcB
<i>cscfEosNextEosCaseName</i>	node1mgcBFail

Table 56 Configuring EOS Case node2node3mgcAFail

cscfEosCaseTableEntryId = node2node3mgcAFail	
Attribute	Value Example
<i>cscfEosCaseTableEntryId</i>	node2node3mgcAFail
<i>cscfEosCaseMatchProfileName</i>	nolsupPath
<i>cscfEosCaseRouteName</i>	node1mgcB
<i>cscfEosNextEosCaseName</i>	defaultFail

Table 57 Configuring EOS Case node1mgcBFail

cscfEosCaseTableEntryId = node1mgcBFail	
Attribute	Value Example
<i>cscfEosCaseTableEntryId</i>	node1mgcBFail



cscfEosCaseTableEntryId = node1mgcBFail	
<i>cscfEosCaseMatchProfileName</i>	nolsupPath
<i>cscfEosCaseRouteName</i>	node2node3mgcA
<i>cscfEosNextEosCaseName</i>	defaultFail

Table 58 Configuring EOS Case defaultFail

cscfEosCaseTableEntryId = defaultFail	
Attribute	Value Example
<i>cscfEosCaseTableEntryId</i>	defaultFail
<i>cscfEosCaseMatchProfileName</i>	anyErrorResponse
<i>cscfEosCaseRouteName</i>	exit
<i>cscfEosNextEosCaseName</i>	exit

3.6.6 Call Flow Example

The following subsections show examples of the use of EOS Analysis. As described in the examples, configuration can affect rerouting.

3.6.6.1 Successful Rerouting of SIP Request Example

An EOS analysis resulting in successful rerouting of a SIP request, is shown in Figure 9.

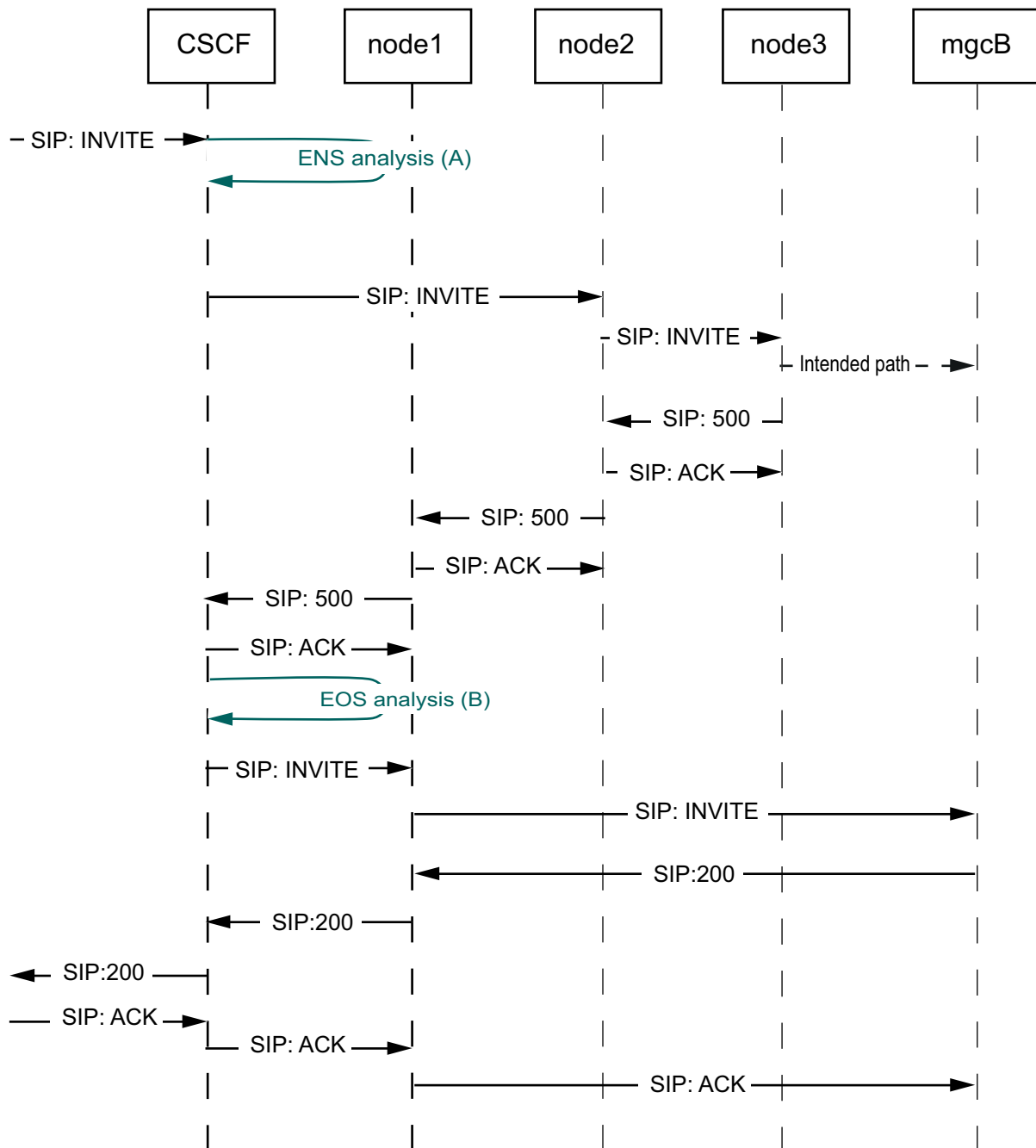


Figure 9 EOS Successful Rerouting of SIP Request

The following happens in this EOS analysis that successfully reroutes a SIP Request:

- At invocation of ENS (A), pool node2node3mgcB is used to route the request. EOS Case node2node3mgcBFail is started if a SIP failure response is received.

- At reception of SIP:500, EOS Analysis (B) is started.
- As `cscfEosMaxEosAttempts` (see Table 48) is not zero, EOS Analysis continues.
- EOS Case `node2node3mgcB` is found (see Table 55).
- The Match Profile `noSipPath` (see Table 47) is used to compare against the received SIP error response.
- A match is found so Routing Alternative `node1mgcB` is started.
- Routing Alternative `node1mgcB` is found (see Table 51)
- The failed SIP request is rerouted based on the data in Table 51 following the path CSCF – node1 – mgcB.
- The SIP request is successful and EOS Analysis is not started again for this session.

3.6.6.2 Rerouting Not Possible – Example

An EOS analysis where rerouting is not possible is shown in Figure 10.

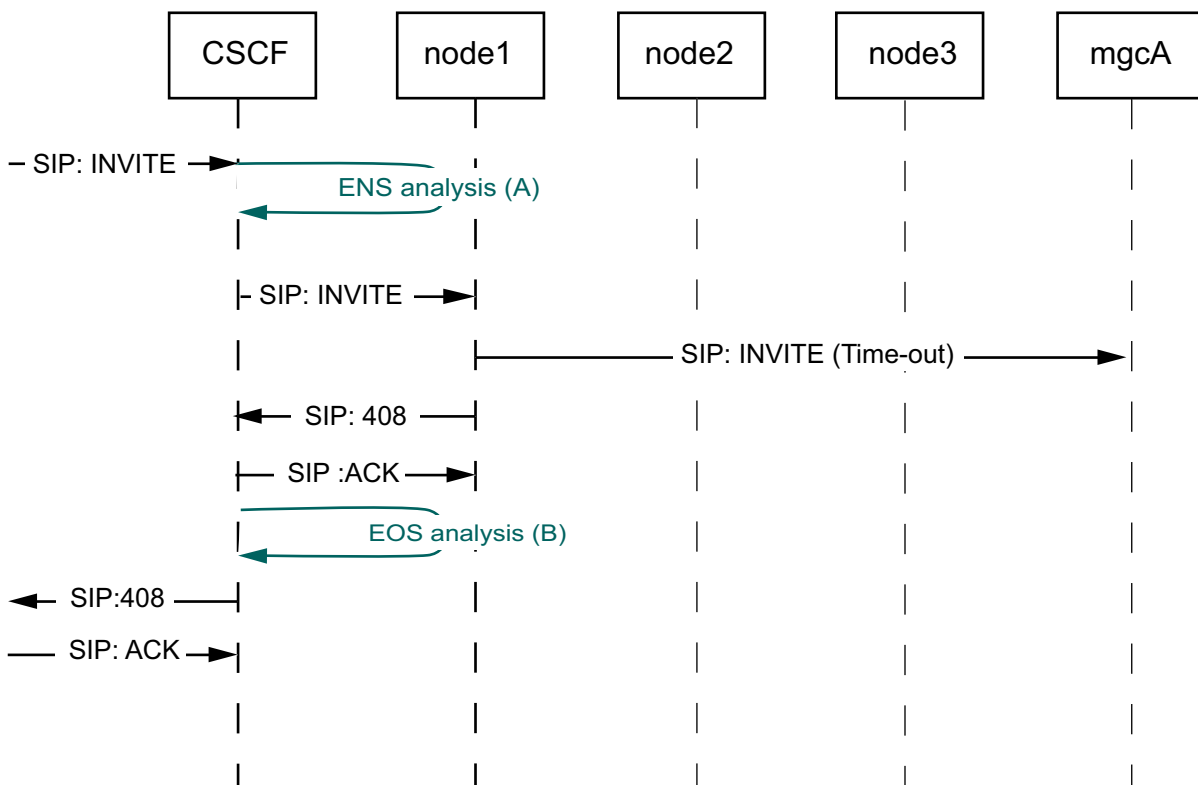


Figure 10 EOS Analysis No Profile Matching in Successive EOS Cases



The following happens in this EOS analysis that results in that it is not possible to reroute the SIP request:

- At invocation of ENS (A), pool `node1mgcA` is used to route the request. EOS Case `node1mgcAFail` is started if a SIP failure response is received.
- At reception of `SIP:408`, EOS Analysis (B) is started.
- As `cscfEosMaxEosAttempts` (see Table 48) is not zero, EOS Analysis continues.
- The EOS Case `node1mgcAFail` is found (see Table 54).
- The Match Profile `noSipPath` (see Table 47) is used to compare against the received SIP error response.
- No match is found so Next EOS Case `node2node3mgcAFail` is started.
- EOS Case `node2node3mgcAFail` is found (see Table 56).
- The Match Profile `noIsupPath` (see Table 48) is used to compare against the received SIP error response.
- No match is found so Next EOS Case `defaultFail` is started.
- EOS Case `defaultFail` is found (see Table 58).
- A match is found so Routing Option `exit` is started.
- EOS Analysis is exited and `SIP:408` is proxied backwards.

3.7 Configure Break-In Control Function

The BCF gives the possibility for users connected to other networks to execute originating IMS services. The BCF supports INVITE messages, and only for registered users.

The BCF is configured using the available NBI with *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

3.7.1 Enable BCF

To enable the BCF:

1. Make sure that the attributes described in the following sections are configured:
 - Section 3.2.1.1 Configure DNS/ENUM Client on page 8
 - Section 3.2.2 Configure Diameter Stack on page 11
 - Section 3.2.3 Configure CSCF Common Attributes on page 15

Note: Only the *CscfCXOriginRealm*, *CscfCXOriginHost*, *CscfCxDestinationRealm*, and *CscfCxDestinationHost* are configured.

2. Add the IP address of the BCF to the *cscfTrustedGateway* attribute in all the S-CSCF nodes.

Note: The S-CSCF rejects requests from a BCF if the IP address is not in the *cscfTrustedGateway* list.

3. Set the *cscfAdministrativeState* to LOCKED.
4. Set the *bcfEnabled* to true and configure the *bcfDomainNameEntry*.
5. Configure at least one *bcfNetworkInterfaceEntry*.
6. Set the *cscfAdministrativeState* to UNLOCKED.

When the BCF node is in unlocked mode, it is ready to receive SIP signaling.

3.8 Configure Authentication

The S-CSCF supports several authentication mechanisms.

The CSCF is configured using the available NBI with *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

3.8.1 Configure Trusted Gateway and Trusted AS

This section describes the trusted gateway and trusted AS configuration.

3.8.1.1 Enable Trusted Gateway

Trusted Gateway authentication is supported by the S-CSCF. At reception of a SIP request (except CANCEL and ACK), the S-CSCF verifies that the IP address from which the SIP message was originated is equal to the configured IP address of trusted gateways. If they match, processing continues without any additional authentication.

To enable this function:

1. Configure the list of trusted SIP gateways in the *cscfTrustedGateway*.
2. Make sure the *cscfOverallAuthenticationPolicyEnabled* is set to enabled.

3.8.1.2 Enable Trusted AS

Trusted AS authentication is supported by the S-CSCF. At reception of a SIP request from an AS, the S-CSCF verifies that the AS transport address is in



the list of trusted ASs (*cscfTrustedASEntry*). If not, the S-CSCF rejects the request with error response 403 (User Agent Not Authorized).

To enable this function:

1. Configure the list of trusted ASs in the *cscfTrustedASEntry*.

3.8.2 Configure NASS Bundled Authentication

NASS Bundled Authentication, refer to [3GPP TS 24.229](#), is supported by the S-CSCF. The S-CSCF compares the line identity received in the P-access-network header with the value retrieved from the HSS.

To enable NASS Bundled Authentication:

1. Define at least one access type in the *cscfNBAAccessNetworkType*.

For example, the *cscfNBAAccessNetworkType* = **ADSL2**

2. Make sure the *cscfOverallAuthenticationPolicyEnabled* is set to **ENABLED**.

When the *cscfNBAAccessNetworkType* contains at least one access type, the function is enabled. When the *cscfNBAAccessNetworkType* is empty, the function is disabled.

When the function is enabled, the CSCF is able to control that NASS Bundled Authentication is used for all users of the access networks configured in the *cscfNBAAccessNetworkType* by setting attribute *ScscfNbaAuthSchemeUnknownEnabled* to **false**.

Alternatively, if the CSCF allows the HSS to choose between NASS Bundled Authentication or Digest Authentication for those access networks, attribute *ScscfNbaAuthSchemeUnknownEnabled* is set to **true**.

3.8.3 Configure Digest Authentication

Digest Authentication, refer to [3GPP TS 24.229](#), is supported by the S-CSCF. At initial registration, the S-CSCF fetches the Digest Authentication vectors from the HSS and then challenges the user with a 401 (Unauthorized) message. The S-CSCF checks that the result of the challenge is correct.

There is no specific enabling for this function, except making sure the *cscfOverallAuthenticationPolicyEnabled* is set to **enabled**.

Digest Authentication is performed by the S-CSCF when IMS AKA, Trusted Gateway authentication, or NASS Bundled Authentication do not apply, and a SIP Authorization header exists in the SIP REGISTER message.

The *cscfSipDigestAuthenticationNonceTimeLength* can be used to change the validity time of the nonce sent by the S-CSCF to the UE in the 401 (Unauthorized) challenge message.

The *cscfSipDigestAuthenticationNonceReusabilityLimit* can be used to change the number of times the nonce can be reused.

The *cscfAuthenticationPolicyEntry* can be used to configure for which subsequent SIP request message authentication is to be performed.

Note: Authentication is never performed for SIP messages ACK and CANCEL.

At subsequent SIP request, to perform Digest Authentication, *cscfAuthenticationProcedure* is to be set to “digest”. For Optimized Digest Authentication configuration, see Section 3.8.3.3 Configure Optimized Digest Authentication on page 73.

The *cscfAuthenticationPolicyEntry* can be used to configure for which subsequent SIP request message authentication is to be performed.

Note: Authentication is never performed for SIP messages ACK and CANCEL.

3.8.3.1 Configure Digest Authentication for UEs

Some UEs do not send SIP Authorization header in the first REGISTER message when registering. The S-CSCF performs Digest Authentication by challenging such UEs with a 401 (Unauthorized) message before fetching the Digest Authentication vectors from the HSS.

To enable this function:

1. Set the *scscfSipDigestAuthenticationRealm* to the same value as in the HSS. For example, “Welcome to the ims.XYZ.net network, insert your username and password”.
2. Make sure the *cscfOverallAuthenticationPolicyEnabled* is set to enabled.
3. Make sure that Digest Authentication is used for UE access.

Digest Authentication for UEs not sending SIP Authorization header in the first REGISTER cannot be enabled at the same time as GIBA, for the same access type profile. To enable both functions in the S-CSCF, two different access profiles must be configured, see Section 3.9 Configure Access Awareness on page 74.

3.8.3.2 Configure Blacklist Function

The blacklist function enables blacklisting suspected SIP clients who attempt to pass the Digest Authentication challenge through repeatedly submitting authentication requests with different authentication credentials.



Once a SIP client is blacklisted, all authentication attempts are rejected by the S-CSCF with a 500 (Retry-After) response, for a time period, until the blacklist time period expires.

To enable this function:

1. Set the *cscfAuthenticationBlackListEnabled* to true.
2. Make sure the *cscfOverallAuthenticationPolicyEnabled* is set to enabled.
3. Make sure that Digest Authentication is used for UE access.

The *cscfBlacklistMaxAuthenticationAttempt* can be used to configure the limit of the number of consecutive authentication attempts because of a failed verification of an authentication response, before SIP client is blacklisted by the S-CSCF.

The *cscfBlackListTimer* can be used to configure the blacklist time period that requests from a blacklisted SIP client is rejected with 500 (Retry-After) response by the S-CSCF.

The *cscfBlackListLoggingFrequency* can be used to configure the maximum number of blacklist time periods before another log is issued by the S-CSCF.

Note: The S-CSCF issues a log in the beginning of the first blacklist time period.

3.8.3.3

Configure Optimized Digest Authentication

Optimized Digest is applicable to subsequent SIP requests from Digest authenticated UEs. With Optimized Digest Authentication, the received Via headers are compared to the Via headers stored at initial registration. If comparison is successful, no other checks are performed.

If comparison fails, Digest Authentication is performed.

To enable this function:

1. Set the *cscfAuthenticationProcedure* attribute to optimized.
2. Make sure the *cscfOverallAuthenticationPolicyEnabled* is set to enabled.
3. Make sure that Digest Authentication is used for UE access.

3.8.4

Configure GPRS IMS Bundled Authentication

GPRS IMS Bundled Authentication (GIBA), refer to [3GPP TS 24.229](#) and [3GPP TS 33.203](#), is supported by the S-CSCF. The S-CSCF compares the UE IP address received in the REGISTER message with the value retrieved from the HSS.

To enable GIBA:

1. Set the *scscfGibaAuthenticationEnabled* to true.
2. Make sure the *scscfGibaAuthenticationEnabled* is set to enabled.

The *cscfAuthenticationPolicyEntry* can be used to configure for which subsequent SIP request message authentication is to be performed.

Note: Authentication is never performed for SIP messages ACK and CANCEL.

GIBA cannot be enabled at the same time as Digest Authentication for UEs not sending SIP Authorization header in the first REGISTER, for the same access type profile. To enable both features in the S-CSCF, two different access profiles must be used, see Section 3.9 Configure Access Awareness on page 74.

3.9 Configure Access Awareness

To enable a more flexible configuration, it is possible to allow configuration based on the access type of the user. This makes it possible to configure different authentication and registration policies for users from different access types in the CSCF.

The S-CSCF determines the configuration to apply at reception of an initial register request by extracting the access type from the PANI header.

The access aware configuration affects the configuration of originating S-CSCF, see Section 3.9.1 Enable Access Awareness on page 74.

The CSCF is configured using the NBI with *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

3.9.1 Enable Access Awareness

To enable access awareness:

1. Configure at least one policy group, see Section 3.9.1.2 Policies on page 76.
2. Configure at least one profile to point to selected policies, see Section 3.9.1.3 Profiles on page 76.
3. Configure the mapping table to map at least one access type to each profile, see Section 3.9.1.4 Mapping Table on page 76.

3.9.1.1 Access Aware Configuration Example

To configure the access aware:



1. Create two new authentication policy groups, GIBA and Digest. Also create two registration policy groups; the MaxRefresh and the MinRefresh:

```
CscfAuthentication=GIBA, CscfAuthenticationGroup=0,
CscfAuthenticationPolicyEntry: Re-Registration:disabled
CscfAuthenticationPolicyEntry: INVITE:enabled
CscfAuthenticationPolicyEntry: NOTIFY:disabled
CscfAuthenticationPolicyEntry: UPDATE:disabled
ScscfGibaAuthenticationEnabled: true
ScscfSipDigestAuthenticationRealm: <empty>
```

```
CscfAuthentication=Digest, CscfAuthenticationGroup=0
CscfAuthenticationPolicyEntry: Re-Registration:disabled
CscfAuthenticationPolicyEntry: INVITE:enabled
CscfAuthenticationPolicyEntry: NOTIFY:disabled
CscfAuthenticationPolicyEntry: UPDATE:disabled
ScscfGibaAuthenticationEnabled: false
```

ScscfSipDigestAuthenticationRealm: set it same as in the HSS.

```
CscfRegistration=MaxRefresh, CscfRegistrationGroup=0,
CscfRegistrationRefreshMin: 60
CscfRegistrationRefreshMax: 200
CscfRegistrationRefreshDefault: 100
CscfRegistration=MinRefresh, CscfRegistrationGroup=0,
CscfRegistrationRefreshMin: 1
CscfRegistrationRefreshMax: 60
CscfRegistrationRefreshDefault: 30
```

2. Configure profiles and point to the existing policy group. Profiles are GIBAMax and DigestMin.

```
CscfProfile=GIBAMax, CscfProfileGroup=0,
CscfAuthenticationPolicy: GIBA
CscfRegistrationPolicy: MaxRefresh
CscfProfile=DigestMin, CscfProfileGroup=0,
CscfAuthenticationPolicy: Digest
CscfRegistrationPolicy: MinRefresh
```

3. Create mapping tables, including the PANI header, and map that to the *CscfProfile* to use.

```
CscfConfigProfileMappingTableEntry: IEEE-802.11-FDD:DigestMin
CscfConfigProfileMappingTableEntry: 3GPP-UTRAN-FDD:GIBAMax
```

A UE that registers with a PANI that contains access-type “IEEE-802.11” uses the configuration profile “DigestMin” and the configuration defined in the Authentication policy group “Digest” and the Registration policy group “MinRefresh”. A PANI containing “3GPP-UTRAN-FDD” uses the configuration profile “GIBAMax”. Any other empty, or missing, PANI header uses the configuration from the *CscfAuthenticationPolicy=default*

and `CscfRegistrationPolicy=default`. The configuration profile is used for all subsequent requests for this particular UE.

3.9.1.2 Policies

At initial start, a Registration policy and an Authentication policy are created. These default policy group instances are used if there is no mapping table defined, no profile maps to the received PANI access-type, or no PANI header was present in the request. Attributes that are not access aware can only be modified in the default policies. If so, other policies that have been configured are updated with the new value of those attributes. Attributes that are access aware can have different values in different policies. For more information about which attributes are access aware, refer to *Managed Object Model (MOM)*.

3.9.1.3 Profiles

The profile points to the policies to use. If no policy is specified for a newly created profile, the default Authentication/Registration policies are used.

3.9.1.4 Mapping Table

The `cscfConfigProfileMappingTableEntry` maps one or more PANI access-types to a specific `CscfProfile`.

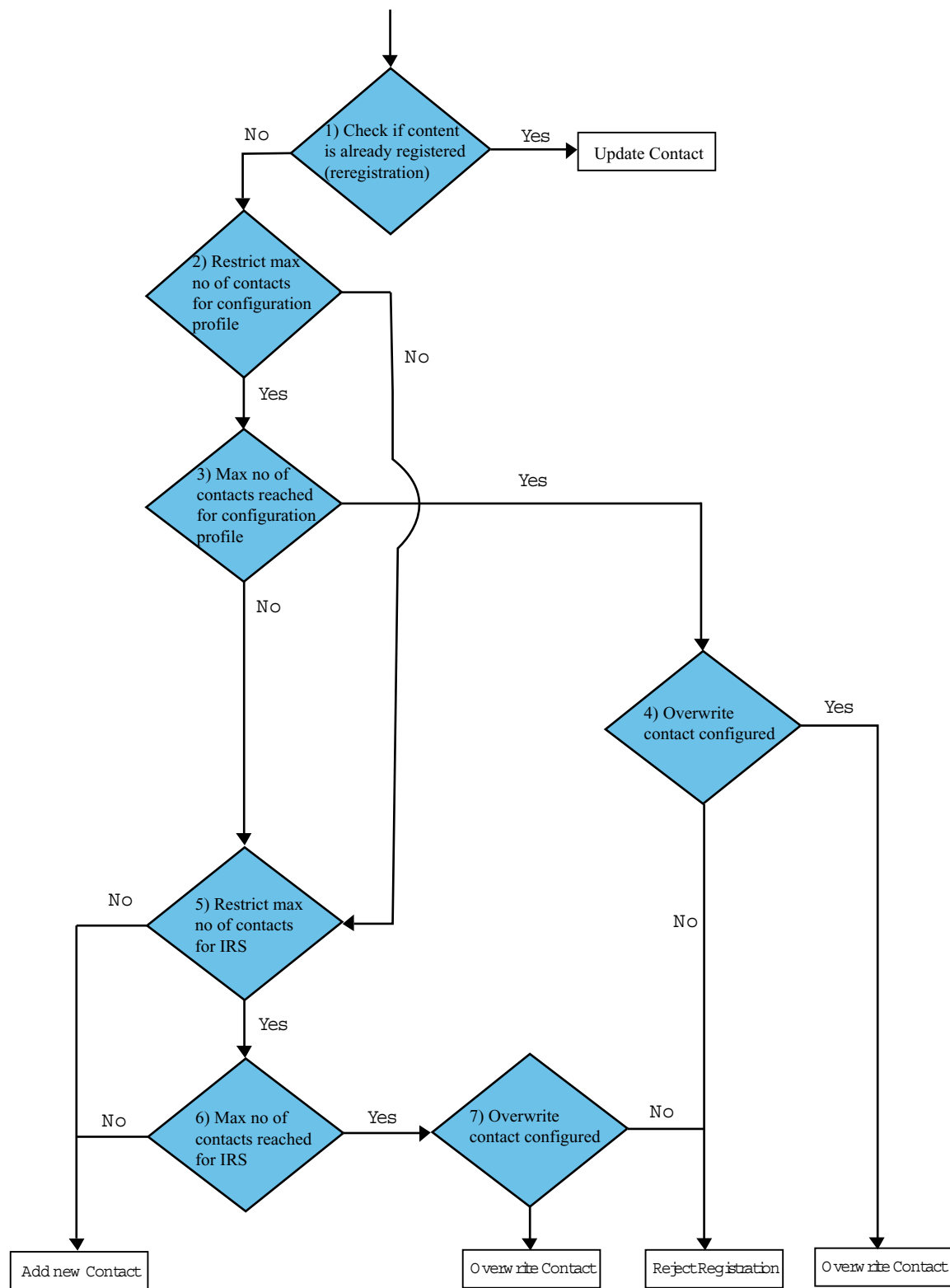
3.10 Configure Max Number of Contacts

It is possible to configure the limit of number of contacts allowed to be simultaneously registered in the S-CSCF. The behavior when the limit is exceeded can be configured to reject new registrations or to overwrite an existing contact. The feature can also be disabled, meaning that the CSCF is not enforcing any restriction on the number of contacts.

The number of contacts can be restricted based on the combination of an IMS Private Identity (IMPI), Implicit Registration Set (IRS), and the access type. This is configured by the `cscfMaxNumberContactsPerUser` and `cscfMaxContactsBehavior`.

A restriction can also be configured for all the contacts within an IRS by using the proprietary AVP “Max-No-Of-Contacts” sent from the HSS over the Cx interface. The behavior for this restriction is configured by the `cscfMaxContactsBehaviorIrs` that compares the value of the AVP with the number of contacts registered in the corresponding IRS.

For handling of these attributes by the S-CSCF, see Figure 11.



3.11 Configure Shared IFC

To start value-added services provided by ASs, certain traffic conditions must be met. These conditions are described in the Service Profile of a user with few Initial Filter Criteria (IFC). Each IFC defines some conditions, for example, all originating messages from the user where the method is INVITE. These conditions are evaluated by the S-CSCF during traffic. When the conditions are met, the S-CSCF triggers the invocation of the service by sending the traffic request to the AS responsible for the service.

The Service Profiles of a user can also express these conditions in the form of Shared Initial Filter Criteria set (SiFC) identities. An SiFC set points to a group of IFCs locally administered and stored at the S-CSCF. When receiving SiFC set information in a user profile, the S-CSCF behaves as if each of the locally defined IFCs pointed by the SiFC set had been included explicitly in the user profile received from the HSS.

3.11.1 Enable Shared IFC

To enable SiFC support:

1. Define the local SiFC set definitions.
2. Activate the new definitions by setting the *scscfSharedIfcSynchronization* attribute to *true*.
3. Set the attribute *scscfSharedIfcEnabled* to *true*.

3.11.2 Define Shared IFC Sets

An SiFC set number received in a user profile is associated with its local definition consisting of one or more IFCs. When defining an SiFC set, an SiFC set instance is first created and assigned a number. Within this instance, one or more IFC instances are created. Within each IFC instance, one or more Service Point Trigger (SPT) group instances are created. Within each SPT group instance, one or more SPT trigger instances are created.

For more information about SiFC set, IFC, SPT group, and SPT attributes, refer to *Managed Object Model (MOM)*.

All IFCs associated to a user, either received from the HSS or derived from the local definition of an SiFC set, must have a unique priority. If two IFCs have the same priority number, and one of these IFCs is derived from an SiFC set, an alarm is raised. For more information, refer to *CSCF Shared Initial Filter Criteria Priority Collision*.

Within an SPT instance, the acceptable values of the *scscfSptTriggerContext* and the *scscfSptTriggerValue* attributes depend on the type of trigger being defined. The type of trigger is specified in the *scscfSptTriggerType* attribute. When creating or modifying an SPT instance, the *scscfSptTriggerType*, *scscfSptTriggerContext*, and the



scscfSptTriggerValue attributes must be defined together and lead to a valid combination according to Figure 12.

The four types of trigger that can be defined within an SPT instance are shown in Figure 12. The associated meaning of the *scscfSptTriggerContext*, the *scscfSptTriggerValue* attributes, when the *scscfSptTriggerContext* and the *scscfSptTriggerValue* attributes are required are also shown in Figure 12.

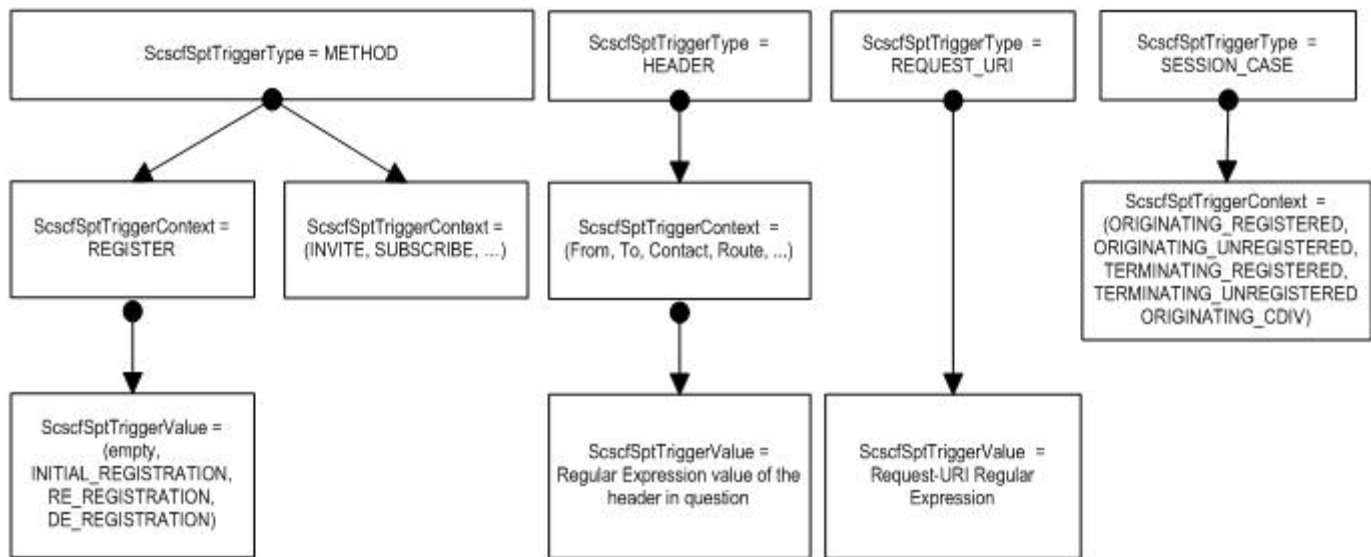


Figure 12 Service Point Trigger Type Configuration Scenarios

3.11.2.1 Shared IFC Set Local Definition Example

SiFC set ID “1”, which consists of a two IFCs named “Service1” and “Service2”, is defined in Table 59 and in the following list:

- IFC “Service1” consists of one SPT group with ID “0”. When the condition is met, AS “asname.com” is started.
- SPT group 0 consists of one SPT named “InitialReg”. The condition is met for all initial registration messages received.
- IFC Service2 consists of one SPT group with ID “0”. When the condition is met, AS “asname2.com” is started.
- SPT group 0 consists of two SPTs named “Originating” and “Invite”. The condition is met for all originating INVITE messages.

SiFC set ID “2”, which consists of one IFC named “Service3”, is defined in Table 60 and in the following list:

- IFC “Service3” consists of two SPT Groups with ID “0” and “1”. When the condition is met, AS “asname3.com” is started.

- SPT group 1 consists of one SPT named “FromEricsson”.
- SPT group 2 consists of one SPT named “Terminating”. The condition is met for all terminating messages from `Ericsson.com`.

Table 59 SiFC Set 1

Attribute	Value Example
<i>scscfSharedIfcId</i>	1
<i>scscfIfcName</i>	Service1
<i>scscfIfcAsName</i>	sip:asname.com;lr
<i>scscfIfcConditionTypeCNF</i>	true
<i>scscfIfcDefaultHandling</i>	SESSION_CONTINUED
<i>scscfIfcEnabled</i>	true
<i>scscfIfcPriority</i>	0
<i>scscfSptGroupId</i>	0
<i>scscfSptName</i>	InitialReg
<i>scscfSptConditionNegated</i>	false
<i>scscfSptTriggerType</i>	METHOD
<i>scscfSptTriggerContext</i>	REGISTER
<i>scscfSptTriggerValue</i>	INITIAL_REGISTRATION
<i>scscfIfcName</i>	Service2
<i>scscfIfcAsName</i>	sip:asname2.com;lr
<i>scscfIfcConditionTypeCNF</i>	false
<i>scscfIfcDefaultHandling</i>	SESSION_CONTINUED
<i>scscfIfcEnabled</i>	TRUE
<i>scscfIfcPriority</i>	1
<i>scscfSptGroupId</i>	0
<i>scscfSptName</i>	Originating
<i>scscfSptConditionNegated</i>	false
<i>scscfSptTriggerType</i>	SESSION_CASE
<i>scscfSptTriggerContext</i>	ORIGINATING_REGISTERED
<i>scscfSptTriggerValue</i>	
<i>scscfSptName</i>	Invite
<i>scscfSptConditionNegated</i>	false
<i>scscfSptTriggerType</i>	METHOD



<i>scscfSptTriggerContext</i>	INVITE
<i>scscfSptTriggerValue</i>	

Table 60 SiFC Set 2

Attribute	Value Example
<i>scscfSharedIfcId</i>	2
<i>scscfIfcName</i>	Service3
<i>scscfIfcAsName</i>	sip:asname3.com;lr
<i>scscfIfcConditionTypeCNF</i>	true
<i>scscfIfcDefaultHandling</i>	SESSION_CONTINUED
<i>scscfIfcEnabled</i>	true
<i>scscfIfcPriority</i>	0
<i>scscfSptGroupId</i>	0
<i>scscfSptName</i>	FromEricsson
<i>scscfSptConditionNegated</i>	false
<i>scscfSptTriggerType</i>	HEADER
<i>scscfSptTriggerContext</i>	FROM
<i>scscfSptTriggerValue</i>	/^ericsson.com\$/
<i>ScscfSptGroupId</i>	1
<i>scscfSptName</i>	Terminating
<i>scscfSptConditionNegated</i>	false
<i>scscfSptTriggerType</i>	SESSION_CASE
<i>scscfSptTriggerContext</i>	TERMINATING_REGISTERED
<i>scscfSptTriggerValue</i>	

3.11.3 Activate Shared IFC Definitions

After defining or updating the SiFC definitions, set the *scscfSharedIfcSynchronization* to `true`. The CSCF application is notified that a new configuration must be used. The application reads the new definitions and sets the *scscfSharedIfcSynchronization* to `false` when completed. The new definitions are used at the next traffic event for each user.

3.11.4 Delete Shared IFC When Referenced by User Profile or Used as Node Default Shared IFC

The CSCF allows an SiFC to be removed even if a user profile refers to it. There is no automatic support to prevent this. Therefore a thorough manual

check is required to make sure that no user profile or node default IFC refers to the SiFC that is to be removed. A more secure handling is to modify the existing SiFC or create a SiFC.

Similarly, the CSCF allows an SiFC to be removed, even if it is used as the node default SiFC. The following impacts can be observed after a referenced SiFC is deleted:

- At registration, the registration is rejected. This follows legacy behavior when a user references a non-existing SiFC.
- At re-registration, the IFC of the user is not triggered (also the IFCs that are not related to the deleted SiFC, but connected directly to the Service Profile of the user). As a consequence, third-party register messages are not sent.
- If the configured node default SiFC is not defined, or has been deleted, the CSCF behaves as if no node default SiFC is configured.
- For non-registration use cases (for example INVITE), the call continues from UE_A to UE_B as if SiFC never existed. No UE_A IFCs are triggered.

Therefore, it is not recommended to delete or remove any SiFCs if they are being referenced. Rather create a new or modify an existing SiFC.

3.12 Configure Node Default IFC

The optional feature Node Default IFC uses a locally defined SiFC set, applicable to all users in the S-CSCF. The feature is triggered when no IFCs or SiFCs from the Service Profile have been triggered for a user.

Node Default IFC feature is not executed for Call Out Of the Blue (COOB) and emergency calls. It is also not executed for registration triggers (third-party registration).

The feature uses the mechanism, configuration aspect, and data structure of the SiFC feature. To use the Node Default IFC feature, the SiFC feature must be enabled. A Node Default IFC set points to a set of IFCs locally administered and stored in the S-CSCF. Contrary to SiFC Set ID, the Node Default IFC ID is not provisioned in the HSS and not received from HSS in the user profile. Node Default IFC Set ID is only configured locally in the S-CSCF.

The feature is enabled through configuration parameter *scscfNodeDefaultIfcEnabled*. Distinguishing which SiFC set represents Node Default IFC is based on the set identity (ID).

The feature supports an “AS-less” IFC, that can be set to continue or terminate the session. A dedicated keyword is used to configure “AS-less” case. When the IFC is configured with “AS-less”, the S-CSCF does not attempt to start an AS.



3.12.1 Enable Node Default IFC

To enable Node Default IFC:

1. Enable SiFC feature, see Section 3.11.1 Enable Shared IFC on page 78.
2. Define Node Default Set ID and corresponding IFCs by using the procedure for the local SiFC set definition, see Section 3.11.2 Define Shared IFC Sets on page 78.
3. Configure attribute *scscfNodeDefaultSifcSet* with the assigned set ID value from the previous step.
4. Activate the new definitions as for any SiFC, see Section 3.11.1 Enable Shared IFC on page 78, step 3.
5. Set attribute *scscfNodeDefaultIfcEnabled* to TRUE

3.12.2 Define Node Default IFC Set

To define Node SiFC Set, see Section 3.11.2 Define Shared IFC Sets on page 78. Only one Node Default IFC Set can be defined.

Note: Node Default IFC Set ID must be configured before referring to it and must not be removed as long as it is referred to.

Specific to the Node Default IFC Set definition, attribute *scscfIfcAsName* can be defined as “AS-less”, by using dedicated string AS-NotDefinedInIFC in SIP URI format definition. In the “AS-less” case, the S-CSCF does not attempt to reach an AS, to save on unnecessary signaling, but simply continues or terminates the session, depending on the configuration of the attribute *ScscfIfcDefaultHandling*, as for any other AS defined in an IFC.

3.12.2.1 Node Default IFC Set Definition Example

A node default IFC set definition example is shown in Table 61.

Table 61 Node Default IFC Set Definition Example, Part 1

Attribute	Value Example
<i>ScscfSharedIfcId</i>	15
<i>ScscfIfcName</i>	NDSservice1
<i>ScscfIfcAsName</i>	Asname1@ericsson.com;lr
<i>ScscfIfcConditionTypeCNF</i>	TRUE
<i>ScscfIfcDefaultHandling</i>	SESSION_CONTINUED
<i>ScscfIfcEnabled</i>	TRUE
<i>ScscfIfcPriority</i>	0
<i>ScscfSptGroupId</i>	0

Attribute	Value Example
<i>ScscfSptName</i>	Originating
<i>ScscfSptConditionNegated</i>	FALSE
<i>ScscfSptTriggerType</i>	METHOD
<i>ScscfSptTriggerContext</i>	INVITE
<i>SscfSptTriggerValue</i>	
<i>ScscfIfcName</i>	NDService2
<i>ScscfIfcAsName</i>	Asname2@ericsson.com;lr
<i>ScscfIfcConditionTypeCNF</i>	TRUE
<i>ScscfIfcDefaultHandling</i>	SESSION_CONTINUED
<i>ScscfIfcEnabled</i>	TRUE
<i>ScscfIfcPriority</i>	1
<i>ScscfSptGroupId</i>	0
<i>ScscfSptName</i>	MessageSpt
<i>ScscfSptConditionNegated</i>	TRUE
<i>ScscfSptTriggerType</i>	METHOD
<i>ScscfSptTriggerContext</i>	MESSAGE
<i>SscfSptTriggerValue</i>	
<i>ScscfIfcName</i>	NDService3
<i>ScscfIfcAsName</i>	AS-NotDefinedInIFC@operator.com ;lr
<i>ScscfIfcConditionTypeCNF</i>	TRUE
<i>ScscfIfcDefaultHandling</i>	SESSION_TERMINATED
<i>ScscfIfcEnabled</i>	TRUE
<i>ScscfIfcPriority</i>	2
<i>ScscfSptGroupId</i>	0
<i>ScscfSptName</i>	FromEricsson
<i>ScscfSptConditionNegated</i>	FALSE
<i>ScscfSptTriggerType</i>	HEADER
<i>ScscfSptTriggerContext</i>	FROM
<i>SscfSptTriggerValue</i>	/*ericsson.com\$/*

3.12.2.2

Last Resort IFC

It is entirely in control of the operator to configure Node Default IFC for the desired use and matching of triggers.



If no IFC, defined in the Node Default IFC set, has been triggered, ongoing sessions continues by default.

If that behavior is not desired, the operator can configure the last resort IFC to overcome that situation, for example to terminate the ongoing session. In that case, the last resort IFC must cover any remaining condition or must contain a universal condition for matching. The last resort IFC must be configured with the lowest priority, that is, as the last IFC for execution in the Node Default IFC set.

As an extension to the previous example, an example of a last resort IFC is shown in Table 62.

Table 62 Node Default IFC Set Definition Example, Part 2: Last Resort IFC

Attribute	Value Example
<i>ScscfIfcName</i>	NDSservice4
<i>ScscfIfcAsName</i>	AS-NotDefinedInIFC@operator.com ;lr
<i>ScscfIfcConditionTypeCNF</i>	TRUE
<i>ScscfIfcDefaultHandling</i>	SESSION_TERMINATED
<i>ScscfIfcEnabled</i>	TRUE
<i>ScscfIfcPriority</i>	3
<i>ScscfSptGroupId</i>	1
<i>ScscfSptName</i>	FromEricsson
<i>ScscfSptConditionNegated</i>	TRUE
<i>ScscfSptTriggerType</i>	HEADER
<i>ScscfSptTriggerContext</i>	Contact ⁽¹⁾
<i>ScscfSptTriggerValue</i>	/.*/

(1) It is assumed that SIP Message is not a potential candidate as last resort IFC because of the lack of Contact header.

3.12.2.3 Example Error Message at Definition

If the “AS-less” value is set to an IFC that is not part of the Node Default SiFC Set, the following error message is returned:

AS-NotDefinedInIFC is not allowed as AS name for an IFC which is not part of the Node Default SiFC [*<nodedefault sifc id>*].

Any other error message applicable to SiFC definition also apply.

3.12.3 Configure `scscfNodeDefaultSifcSet`

When the Node Default IFC Set is defined, it is seen by the S-CSCF as any other SiFC Set.

To allocate a SiFC Set as a Node Default IFC Set, set attribute `scscfNodeDefaultSifcSet` with the appropriate Set ID.

For example, set `scscfNodeDefaultSifcSet` to 15.

Note: It is possible to set an SiFC as the node default SiFC, even if the SiFC is already referenced by another Service Profile. Therefore, if the SiFC is edited, the changes affect both the service profiles referencing it and when it is used as Node Default SiFC.

3.12.3.1 Example Error Message at Configuration

If an invalid value (non-integer or not empty value) is entered, this error message is returned:

```
ScscfNoDefaultSifcSet's value must be an integer.
```

3.12.4 Activate Node Default IFC Definitions

After defining or updating Node Default IFC Set definition, set the `ScscfSharedIfcSynchronization` to TRUE. For more information, see Section 3.11.3 Activate Shared IFC Definitions on page 81.

3.13 Configure Geographical Redundancy

Groupings of the CSCF nodes can be configured to be a geographical standby site for other CSCF nodes in a different geographical zone. When a UE contacts an external P-CSCF in the standby site because of a failure in the primary site, the SIP requests are handled by the CSCF nodes in the standby site. All Originating and Terminating requests are handled by the standby geographical site without trying to contact any CSCF in the failed site even if the user was registered there. When the failed site recovers and the UE contacts the primary external P-CSCF, then all the SIP requests are handled by the CSCF nodes in the primary site. This is controlled by enabling the feature Local Zone Policy through the `icscfLocalZonePolicyEnabled` configuration attribute and by configuring the `cscfResourceBrokerEntry` with only the S-CSCFs in the same geographical zone as the I-CSCF.

The CSCF is configured using the NBI with `CscfFunction` fragment `CSCF-Application`, see Section 3.2.2 Configure Diameter Stack on page 11.

3.13.1 Enable Geographical Redundancy

To enable geographical redundancy:



1. Configure the *cscfResourceBrokerEntry* attribute in the I-CSCF to contain only S-CSCFs in the same geographical zone as the I-CSCF.
2. Set the *icscfLocalZonePolicyEnabled* attribute in the I-CSCF to `true`.

3.14 Configure IMS Restoration

The Restoration Procedure provides a mechanism to ensure continuous services for users, when the S-CSCF in which they are currently registered is not reachable. The Restoration Procedure is an optional feature which can be enabled by configuration. When enabled, the S-CSCF stores and maintains the restoration information of its users in the HSS. If an S-CSCF is out of service, another S-CSCF does the restoration procedure and provide services to the restored users.

The CSCF is configured using the NBI with *CscfFunction* fragment *CSCF-Application*, see Section 2.2 Configuration Methods on page 4.

3.14.1 Enable Restoration Procedure

To enable the Restoration Procedure:

1. Set the *scscfRestorationProcedure* in the S-CSCF to 1 or 2 as needed. For information on the values of the attribute, see Table 63.
2. If the S-CSCF is planned to disable processing originating non-Register requests, set the *scscfRestorationOriginatingNonRegisterAllowed* in the S-CSCF to `FALSE`. For information on the values of the attribute, see Table 63.

Table 63 Restoration Procedure Attributes

Attribute	Description
<i>scscfRestorationProcedure</i>	<p>This attribute is used to enable or disable the Restoration Procedure.</p> <p>Values:</p> <ul style="list-style-type: none">• 0: The default value. It disables the Restoration Procedure.• 1: It enables the Restoration Procedure. If the user is not found, the S-CSCF triggers the Restoration Procedure.• 2: It enables the Restoration with Subscription Information Procedure. If a user is not found, the S-CSCF includes the subscription information that contains information in the received SUBSCRIBE message from the user and restoration information in the diameter message that is sent to the HSS.
<i>scscfRestorationOriginatingNonRegisterAllowed</i>	<p>This attribute is used only when <i>scscfRestorationProcedure</i> is enabled.</p> <p>Values:</p> <ul style="list-style-type: none">• TRUE: The default value. It enables the S-CSCF to process originating non-Register requests.• FALSE: It disables the S-CSCF to process originating non-Register requests. The S-CSCF rejects the request with a 504 error response.

3.15 Configure Load Regulation and Priority

This section describes how to configure load regulation and priority support session.



3.15.1 Configure Load Regulation

The CSCF supports 20 internal priority levels (internal priority values range from 0 to 19 where 19 is the highest priority) to be able to give preferential treatment to some services over the others.

Attribute *cscfTspPriorityMapping* is used to map the priority of SIP requests defined by the Wireless Priority Service (WPS) value (wps priority values range from 0 to 4 where 0 is the highest priority) in the Resource-Priority Header, and for normal and emergency calls to internal priority levels.

The setting of the Load Regulation attribute is shown in Table 64.

Table 64 Configuring Priority Mapping

Attribute	Value Example
<i>cscfTspPriorityMapping</i>	p0:19; p1:17; p2:15; p3:12; p4:9; ec:14; nc:0

p0-p4 are the Priority Levels of the call (where p0 is the highest, p4 is the lowest, defined by the wps value in the RPH header). Calls with the received wps priority values are mapped to internal priority levels 19-0.

ec is used to define an internal priority level to Emergency Calls.

nc is used to define an internal priority level to Normal Calls (without priority indication in the RPH header).

3.15.2 Enable Priority Support Session

The Priority Support Session facilitates emergency responses and recovery operations when the traffic intensity is high. Priority Support Session is an optional feature which can be enabled by configuration.

To enable the Priority Support Session:

1. Set attribute *cscfPrioritySupportEnabled* to false.

All the following priority-related features require Priority Support Session enabled.

3.15.2.1 Enable S-CSCF Priority Authorization

The S-CSCF Priority Authorization provides a mechanism to validate the priority in the user request against the priority in the user profile.

To enable the S-CSCF Priority Authorization:

1. Set attribute *scscfPriorityAuthorizationEnabled* to false.

When S-CSCF Priority Authorization is enabled, an option exists to set whether the SIP request is rejected if there is no priority in the user profile. By setting *scscfRejectIfNoPrioInProfileEnabled* to `true`, the S-CSCF rejects the request with SIP 403 error response in this case.

3.15.2.2 GETS-FC Translation

The S-CSCF supports the translation of Government Emergency Telecommunications Service Feature Code (GETS-FC) in the Request URI content into Resource-Priority Header. When GETS-FC is detected, the S-CSCF includes the priority found in the user profile in the Resource-Priority Header in the outgoing request. S-CSCF Priority Authorization must be enabled for GETS-FC translation.

The setting of the GETS-FC translation attribute is shown in Table 65.

Table 65 Configuring Priority Prefix

Attribute	Value Example
<i>scscfPriorityPrefix</i>	*272

SIP INVITE received by the originating S-CSCF with an R-URI that contains:
`tel:*2729271234;phone-context=+1.`

The S-CSCF matches *scscfPriorityPrefix* (*272) against the received telephone number (*2729271234) and detects that it contains a GETS-FC. The S-CSCF removes the received GETS-FC from the telephone number and includes the priority found in the user profile in the Resource-Priority Header in the outgoing request.

3.15.2.3 GETS-AN Translation

The S-CSCF supports the translation of Government Emergency Telecommunications Service Access Number (GETS-AN) in the Request URI into Resource-Priority Header. When GETS-AN is detected, the S-CSCF includes a Resource-Priority Header with `ets.0` in the outgoing request.

The setting of the GETS-AN translation attribute is shown in Table 66.

Table 66 Configuring GETS-AN

Attribute	Value Example
<i>cscfGetsAnEntry</i>	17106274387

SIP INVITE received by the originating S-CSCF with an R-URI that contains:
`tel:17106274387;phone-context=+1.`

The S-CSCF matches *cscfGetsAnEntry* (17106274387) against the received telephone number (17106274387) and detects that it is a GETS-AN. In this case, the



S-CSCF includes a Resource-Priority Header with the `ets.0` in the outgoing request.

3.15.2.4 Enable S-CSCF Priority Handling for Terminating Prioritized User

The S-CSCF Priority Handling for Terminating Prioritized User provides a mechanism to enable if the terminating S-CSCF adds a Resource-Priority Header (RPH) into the SIP request for terminating a prioritized user before sending out the request.

To enable the S-CSCF Priority Handling for Terminating Prioritized User:

1. Ensure that `cscfPrioritySupportEnabled` is set to **true**
2. Set `scscfAssignCalleePriority` to **ADD_RPH**

When the S-CSCF Priority Handling for Terminating Prioritized User is enabled and the terminating S-CSCF receives a SIP request without an RPH that is sent to a prioritized user, the S-CSCF creates an RPH based on the Service Priority Level from the user profile and adds it into the request before sending the request to next hop.

3.16 Configure Dynamic User Identity Support

Dynamic User Identity Support (DUIS) is used to support external users who do not have an IMS identity. These external users belong to a partner of the IMS network operator. DUIS is enabled using configurations. When DUIS is enabled and if the terminating request contains a global telephone number or if the external domain of the terminating user of the terminating request matches a domain in the set of preconfigured partner domains, the Dynamic User Association Router (DUA-R) function of the I-CSCF sends an LDAP SEARCH request to the Dynamic User Association Database (DUA-DB) to convert the external user identity (for example, `alice@partner.com` or `tel:+1-555-9175`) to an IMS user identity. The DUA-DB returns the specific IMS Public Identity (IMPU) of a Wildcarded Public Identity (wIMPU). The wIMPU is used as the IMS identity of the external user within the IMS network.

3.16.1 Prerequisites for DUIS

The following prerequisites are required for DUIS to perform properly:

- The support of wIMPU is required. It is enabled by configuration parameter, `scscfwImpuEnabled`.
- The support of Enhanced Caller Preferences is required. It is enabled by configuration parameter, `scscfEnhancedCallerPreferencesEnabled`.

- With DUIS, a given wIMPU can represent many external users that can generate multiple sessions in parallel. To ensure that all requests are processed, it is recommended that the number of sessions allowed per related wIMPU is configured to be unlimited. This is done by setting the value of the `MaxNoSimultaneousSessions` element in the User-Data AVP of the related wIMPU to the maximum value of `xFFFF` (decimal 65535). The User-Data AVP is provisioned in the HSS.

3.16.2 Configure DUA-DB Server

Before enabling DUIS, at least one DUA-DB server must be configured.

To configure a DUA-DB server:

1. Configure the DUA-DB servers in terms of IPv4 address in parameter `ldapServerEntryId`, for example, `192.168.28.1:7323`. The first DUA-DB server configured in `ldapServerEntryId` is the primary DUA-DB server and is connected first. Unless it is unavailable, the primary server is always the first one to use when the current server becomes unavailable.
2. Configure the Distinguished username, the administrative password, the Root DN, and the LDAP version of the DUA-DB respectively in parameters, `ldapServerAdminDn`, for example, `administratorName=telcooptr,dc=o,dc=com`, `ldapServerAdminPassword`, `ldapServerRootDn`, for example, `dc=o,dc=com`, and `ldapServerVersion` respectively.

Note: The DN used in the LDAP SEARCH request is the concatenation of `dc=duaExtId,ou=identities,o=DuaDb` and value of `ldapServerRootDn`.

3. Set the time for the DUA-R to consider a DUA-DB server unavailable after the maximum number of retries has failed. Within this time, the DUA-R is not connected to the unavailable DUA-DB server. If the default value of 30 seconds is not suitable, configure the DUA-DB server unavailable time in parameter, `ldapServerUnavailableTime`.
4. Set the maximum time that the DUA-R waits for the response to an LDAP request. When this time expires, the DUA-R resends the request to the DUA-DB server up to the maximum number of retries defined. If the default value of 500 milliseconds is not suitable, configure the maximum response time in parameter, `ldapServerResponseTime`.

Note: The product of the parameter values in `ldapServerEntryId`, `ldapServerResponseTime`, and `ldapServerMaxNumberOfRetries` must not exceed the SIP timers.

5. Set the maximum number of the resending of an LDAP request until declaring the DUA-DB server unavailable. If the default value of three retries is not suitable, configure the maximum number of retries for LDAP requests using parameter, `ldapServerMaxNumberOfRetries`.



3.16.3 Enable DUIS

To enable DUIS, after the DUA-DB servers are configured:

1. Define the External Network domains by configuring parameter, *icscfDynamicUserIdentitySupportDomainEntry*, for example, */partner.com/i,ericsson.com*, in the I-CSCF.
2. Enable DUIS by setting configuration parameter, *icscfDynamicUserIdentitySupportEnabled*, to *TRUE* in the I-CSCF.

3.16.4 Modify DUA-DB Server Configuration

Modifications to the configuration parameters *ldapServerAdminUsername*, *ldapServerAdminPassword*, *ldapServerRootDn*, and *ldapServerVersion* are possible and the changes take effect immediately. Changes to these parameters trigger a disconnection (UNBIND) and a reconnection (BIND) to the DUA-DB server.

Modifications to the configuration parameters *ldapServerUnavailableTime*, *ldapServerResponseTime*, and *ldapServerMaxNumberOfRetries* are possible and the changes take effect immediately. Disconnection and reconnection to the DUA-DB is not triggered.

Modification to the address of a DUA-DB server listed in parameter *ldapServerEntryId* is not supported. To modify the address of a DUA-DB, it is required to delete the DUA-DB with the old address (which triggers an UNBIND request to be sent to the old DUA-DB) and create a DUA-DB with the new address (which triggers a BIND request to be sent to the new DUA-DB).

3.17 Configure SIP Error Response

This section describes how to configure the SIP error responses in the CSCF.

Attention!

This function must be used with support from Ericsson since it can potentially have a serious effect on the normal operation and characteristics of the system that are difficult to foresee.

The CSCF contains a generic mechanism for modifying the SIP error responses generated or transited through the CSCF, using SIP error, SIP Reason Header, and Diameter error as input. It can be used to solve interwork problems that can occur during system integration.

3.17.1 Configure SIP Error Response

The SIP error responses are configured using the NBI using *CscfFunction* fragment *CSCF-Application*.

CM object *cscfSipErrorConfiguration=0* is preconfigured in the database.

The mechanism is activated by creating a *cscfSipErrorConfigurationTable* with identity “root” under this object and populating this table with underlying *cscfSipErrorConfigurationEntries*.

It is also possible to define subsequent *cscfSipErrorConfigurationTables* and link these to other tables to create a hierarchical structure.

The *cscfSipErrorConfigurationTable* defines the criteria that is used to look up entries in the table. This is achieved through the definition of one or more of the *cscfFilteringCriteria* attributes. For more information on the possible criteria, refer to *Managed Object Model (MOM)*.

These criteria also define the format to be used for the identities for the underlying *cscfSipErrorConfigurationEntries*. The syntax of the entry is as follows:

```
<CscfSipErrorConfigurationTable>:<Match1>[:<Match2>
[:<Match3>[:<Match4>[:<Match5>]]]]
```

Where *<cscfSipErrorConfigurationTable>* is the value of the table object where this entry is contained.

<Match1> is a value that can match the criteria defined in *CscfFilteringCriteria1*.

<Match2> is a value that can match the criteria defined in *CscfFilteringCriteria2*.

<Match3> is a value that can match the criteria defined in *CscfFilteringCriteria3*.

<Match4> is a value that can match the criteria defined in *CscfFilteringCriteria4*.

<Match5> is a value that can match the criteria defined in *CscfFilteringCriteria5*.

A match value is required for each of the filtering criteria defined in the table object. The syntax for the default entry is as follows:

```
<cscfSipErrorConfigurationTable>:default
```

For examples with different *cscfSipErrorConfigurationTables*, see Section 3.17.3 SIP Error Responses Examples on page 96.



The *cscfSipErrorConfigurationEntries* defines the actions to perform for SIP error responses that match to the entry.

The actions to perform can be one of the following:

- Continue analysis in another table through the configuration of the *cscfNextTable* attribute.
- Manipulate the response through the configuration of one or more of the attributes *cscfNewErrorCode*, *cscfNewSlogan*, and *cscfNewRetryAfter*.

It is also optional to define a default entry in the table. The default entry is matched in case no other entry is matched.

3.17.2 Operate SIP Error Responses

Before sending SIP error responses, the system looks for the “root” table. If the root table does not exist, the SIP error response is sent without any modification. Otherwise the root table is read and analyzed.

In the table, each of the configured *cscfFilteringCriteria* is referring to values based on the context of the SIP and Diameter error response. A key is generated by the CSCF for the *cscfSipErrorConfigurationEntry*, composed of the table name together with each of the *cscfFilteringCriteria* values.

Each part of the key is delimited with a colon. For example, if the *cscfSipErrorConfigurationTable=root* is configured with two criteria, *CscfFilteringCriteria1=diameter.message* and *CscfFilteringCriteria2=diameter.resultcode*, in runtime, the CSCF generates a key based on the received Diameter message and Diameter error. So if a Multimedia-Authentication-Answer (MAA) has been received with error 5003, the CSCF generates key *root:MAA:5003* and uses that to find a *cscfSipErrorConfigurationEntry* in the configuration. If no Diameter message or Diameter error is received, no key is generated, meaning that there is not any match.

If no entry matches this key, then an attempt is made to match to a default entry. The key to the default entry is composed of the table name and the keyword “default”.

If still no entry is matched, the analysis ends and the SIP error response is sent without any modification.

When an entry matches, it is analyzed. If the attribute *cscfNextTable* is configured, then the analysis continues in a similar way in that table, that is, the CSCF generates a new key based on the name and *cscfFilteringCriteria* in that table. Otherwise the *cscfNewErrorCode*, *cscfNewSlogan*, and *cscfNewRetryAfter* attributes are checked and if configured the SIP error response is manipulated accordingly.

3.17.3 SIP Error Responses Examples

The following subsections show examples of the use of the SIP Error Response mechanism. Three fictitious scenarios are described.

3.17.3.1 SIP Error Responses Example 1

This example shows the following:

- How to activate the SIP error response configuration.
- How to filter SIP error responses based on SIP error code.

The scenario for this example is that an operator discovers an interworking problem related to the transit of non-standardized error codes (601 and 603). To work around the problem, the operator decides to map these error codes to a more general 600 and remove the Retry-After header.

The configuration for this mapping is achieved by creating a *sipErrorConfigurationTable* and populating the table with entries matching to error codes 601 and 603.

The system starts the analysis by looking for the root table. The identity of the table must be “root”.

Only one filter criteria is needed to filter the SIP error messages. *CscfFilteringCriteria1* is set to value *sip.errorcode* indicating the entries in the table are found through the SIP error code, see Table 67.

Table 67 *CscfSipErrorConfigurationTable Root Table*

CscfSipErrorConfigurationTable=root	
Attribute	Value Example
<i>CscfFilteringCriteria1</i>	<i>sip.errorcode</i>
<i>CscfFilteringCriteria2</i>	
<i>CscfFilteringCriteria3</i>	
<i>CscfFilteringCriteria4</i>	
<i>CscfFilteringCriteria5</i>	

Two entries are configured in the table. The first entry matches responses with SIP error code 601. The entry must have the identity *root:601*. This indicates that the entry belongs to the “root” table and matches to the *sip.errorcode* value 601. The two parts of the identity are delimited with a colon.

Attribute *cscfNewErrorCode* is set to 600 indicating the SIP error code must be modified to value 600. Attribute *cscfNewRetryAfter* is set to *remove* indicating the retry-after header if present must be removed, see Table 68.



Table 68 CscfSipErrorConfigurationTable Root Entry 601

CscfSipErrorConfigurationTable=root, CscfSipErrorConfigurationEntry=root:601	
Attribute	Value Example
<i>cscfNextTable</i>	
<i>cscfNewErrorCode</i>	600
<i>cscfNewSlogan</i>	
<i>cscfNewRetryAfter</i>	remove

Similarly a second entry must have identity `root:603` indicating it belongs to the root table and matches to value `603`. The attributes are configured in the same way, see Table 69.

Table 69 CscfSipErrorConfigurationTable Root Entry 603

CscfSipErrorConfigurationTable=root, CscfSipErrorConfigurationEntry=root:603	
Attribute	Value Example
<i>cscfNextTable</i>	
<i>cscfNewErrorCode</i>	600
<i>cscfNewSlogan</i>	
<i>cscfNewRetryAfter</i>	remove

3.17.3.2 SIP Error Responses Example 2

This example extends the configuration in Section 3.17.3.1 SIP Error Responses Example 1 on page 96 and demonstrates how to perform the following:

- Define a secondary table that filters with different criteria
- Filter error responses based on their Reason Header
- Modify the SIP error slogan

The scenario for this example is that an operator wants to remap an error discovered by a downstream node. The specific case they want to handle is a SIP error code 480 containing the Reason Header with protocol Q.850 and cause 58. Responses matching these criteria are mapped to a 488 error code and the slogan is modified.

The solution extends the configuration described in Table 67.

A new table is created to filter based on the SIP Reason header. The table is given identity “example2”. Two filtering criteria are specified to filter

based on the fields in the Reason Header “sip.reason.protocol” and the “sip.reason.cause”, see Table 70.

Table 70 CscfSipErrorConfigurationTable Example 2

CscfSipErrorConfigurationTable=example2	
Attribute	Value Example
CscfFilteringCriteria1	sip.reason.protocol
CscfFilteringCriteria2	sip.reason.cause
CscfFilteringCriteria3	
CscfFilteringCriteria4	
CscfFilteringCriteria5	

One entry is added to the table. The entry matches SIP error responses with Reason Header protocol Q.850 and cause 58. To achieve this entry must have an identity value composed of the table name (example2) the protocol (Q.850) and the cause (58), see Table 71. Each value is delimited with a colon.

Table 71 CscfSipErrorConfigurationTable Entry example2:Q.850:58

CscfSipErrorConfigurationTable=example2, CscfSipErrorConfigurationEntry=example2:Q.850:58	
Attribute	Value Example
cscfNextTable	
cscfNewErrorCode	488
cscfNewSlogan	“Not Acceptable in Adjacent Network”
cscfNewRetryAfter	

The CscfSipErrorConfigurationTable “example2” is not used until a reference to it is created from the root table, see Table 72. The intention is for this table to be executed for SIP errors with error code 480. So a new entry is added to the root table that matches SIP error responses with 480. The only attribute configured for this entry is the cscfNextTable=“example2”. This indicates that all 480 errors must continue analysis in the table named “example2”.

Table 72 Reference to Table Example2 from Root Table

CscfSipErrorConfigurationTable=root, CscfSipErrorConfigurationEntry=root:480	
Attribute	Value Example
cscfNextTable	example2
cscfNewErrorCode	



CscfSipErrorConfigurationTable=root, CscfSipErrorConfigurationEntry=root:480	
<i>cscfNewSlogan</i>	
<i>cscfNewRetryAfter</i>	

3.17.3.3 SIP Error Responses Example 3

This example shows how to extend Section 3.17.3.1 SIP Error Responses Example 1 on page 96 and Section 3.17.3.2 SIP Error Responses Example 2 on page 97 further, to demonstrate the following:

- The use of the default entry
- Diameter message and result codes as filter criteria

The scenario for this example is that an operator wants to remap a diameter error discovered over the Cx interface. The specific cases they want to handle are diameter result code 3004 received in User-Authorization-Answer (UAA) or Location Information Answer (LIA) messages. These messages are mapped to SIP error code 500 and the Retry-after header is modified to 10 seconds.

A new table is created and given the name “example3”. Two criteria are used the diameter message and the diameter result code, see Table 73.

Table 73 *CscfSipErrorConfigurationTable Example3*

CscfSipErrorConfigurationTable=example3	
Attribute	Value Example
<i>CscfFilteringCriteria1</i>	diameter.message
<i>CscfFilteringCriteria2</i>	diameter.resultcode
<i>CscfFilteringCriteria3</i>	
<i>CscfFilteringCriteria4</i>	
<i>CscfFilteringCriteria5</i>	

An entry is added to the table to match to the diameter messages LIA with result code 3004, see Table 74.

Table 74 *CscfSipErrorConfigurationTable Example3 Entry LIA:3004*

CscfSipErrorConfigurationTable=example3, CscfSipErrorConfigurationEntry=example3:LIA:3004	
Attribute	Value Example
<i>cscfNextTable</i>	
<i>cscfNewErrorCode</i>	500



CscfSipErrorConfigurationTable=example3, CscfSipErrorConfigurationEntry=example3:LIA:3004	
<i>cscfNewSlogan</i>	
<i>cscfNewRetryAfter</i>	10

An entry is added to the table to match to the diameter messages UAA with result code 3004, see Table 75.

Table 75 *CscfSipErrorConfigurationTable Example3 Entry UAA:3004*

CscfSipErrorConfigurationTable=example3, CscfSipErrorConfigurationEntry=example3:UAA:3004	
Attribute	Value Example
<i>cscfNextTable</i>	
<i>cscfNewErrorCode</i>	500
<i>cscfNewSlogan</i>	
<i>cscfNewRetryAfter</i>	10

Add a reference to the “example3” table from the root table, see Table 76. It is decided to define a default entry in the root table. Default entries are used to match when nothing else matches. So our default entry is matched when there is no match to the Error Code 601, 603 and 480. The default rule then references the “example3” table where the analysis continues.

Table 76 *Reference to Table Example3 from Root Table*

CscfSipErrorConfigurationTable=root, CscfSipErrorConfigurationEntry=root:default	
Attribute	Value Example
<i>cscfNextTable</i>	example3
<i>cscfNewErrorCode</i>	
<i>cscfNewSlogan</i>	
<i>cscfNewRetryAfter</i>	

3.18 Configure Domain Routing Function

This section describes how to configure and manage the Domain Routing Function (DRF).

DRF provides DNS-like functionality based on a set of static configuration tables local to the CSCF. DRF is available to all CSCF roles (I-/S-/E-), including BGCF and BCF. When started, DRF attempts to map the next hop address of an SIP request in FQDN format to another next hop addresses, which can be in



FQDN, IPv4, or IPv6 format. The SIP request is routed to the new next hop addresses in serial mode on successful DRF table lookups.

If DRF is not started, for example, owing to absence of a SIP request next hop address in FQDN format or that a DRF table lookup fails, the SIP request is routed to the original next hop address.

For more information regarding DRF configuration parameters, refer to *Managed Object Model (MOM)*.

DRF is configured with *CscfFunction* fragment *cscfDomainRoutingApplication* through the NBI, see Section 2.2 Configuration Methods on page 4.

3.18.1 Configure Domain Routing Function Global Attributes

DRF has one instance of configuration with *CscfFunction* fragment *cscfDomainRoutingApplication*. The application can support multiple sets of configuration tables. Each set of configuration tables uses one matching table and one result table. On a successful matching table lookup to the FQDN, the matching table entry references a result table entry which specifies the new next hop addresses to route the SIP request.

The DRF global configuration attributes are listed in Table 77.

Attribute *cscfDomainRoutingEnabled* is an operator configurable attribute to enable or disable DRF. When set to *false*, DRF is disabled. All SIP requests are routed to the original target addresses. When set to *true*, DRF is enabled and can be started for CSCF outgoing SIP requests.

Attribute *cscfDomainRoutingActiveConfig* is a read-only attribute that identifies the set of configuration tables corresponding to the existing runtime data for traffic handling. Attribute *cscfDomainRoutingSelectedConfig* is an attribute that identifies the set of configuration tables to be converted to runtime data for traffic handling, replacing the existing one.

Attribute *cscfDomainRoutingActiveMatchingTable* is an operator configurable attribute which identifies the matching table within a configuration table set for DRF table lookup.

In the current delivery, DRF supports only one set of configuration tables and one matching table within the configuration set. The value of attribute *cscfDomainRoutingSelectedConfig* is hard-coded to default, and the value of *cscfDomainRoutingActiveConfig* equals default only. The value of attribute *cscfDomainRoutingActiveMatchingTable* is also hard-coded to 0. Modifications are allowed to the configuration table set default and matching table 0 only for runtime data update.

Operators set attribute *cscfDomainRoutingSyncConfig* to *true* to initiate the synchronization process for runtime data update based on the configuration table set identified by attribute *cscfDomainRoutingSelectedConfig*. The attribute value is reset to *false* automatically when synchronization terminates.



Attribute *cscfDomainRoutingSyncState* is a read-only attribute showing the result of the synchronization process. It can have one of three values: *synchronized*, *not_synchronized*, and *synchronization_failed*.

On a successful synchronization, the existing runtime data is replaced with the newly generated runtime data for traffic handling. On a synchronization failure, synchronization is ended and traffic handling is resumed with the existing runtime data.

Update to the configuration tables can be made without traffic disturbances. Minor traffic disturbances can be present during synchronization of a configuration table set.

Table 77 DRF Global Configuration Example

Attribute	Value Example
<i>cscfDomainRoutingEnabled</i>	true
<i>cscfDomainRoutingSelectedConfig</i> ⁽¹⁾	default
<i>cscfDomainRoutingActiveConfig</i> ⁽¹⁾	default
<i>cscfDomainRoutingActiveMatchingTable</i> ⁽¹⁾	0
<i>cscfDomainRoutingSyncConfig</i>	false
<i>cscfDomainRoutingSyncState</i> ⁽¹⁾	synchronized

(1) This attribute is not user configurable.

3.18.2 Domain Routing Function Table Configuration Examples

Each set of configuration tables consists of a Domain Routing matching table and a Domain Routing result table. This section illustrates how the DRF matching and result tables can be configured.

When configuring DRF tables, the tables must be created last one first to ensure that a table exists before it is referenced. Domain Routing result table *cscfDomainRoutingResultTableId* must therefore be created first before Domain Routing matching table *cscfDomainRoutingMatchingTableId*.

DRF is started for Domain Routing matching table lookup to the original address in FQDN format. If a SIP request is not to be routed originally to an address in FQDN format or the table lookup fails, the SIP request is routed to the original address.

When specified as part of a next hop target address, the port and transport protocol information is used to route the request to the address. Otherwise, the default port and transport protocol value is used.



Each next hop target address within a result table entry must be assigned a unique priority value. On a successful FQDN match resulting in multiple next hop target addresses, the SIP request is routed to the addresses in serial mode according to the priority setting. The next hop address with the highest priority value is attempted first and the address with the lowest priority value is attempted last.

Table 78 shows a Domain Routing matching table configuration example and Table 79 shows a Domain Routing result table configuration example.

Table 78 Example of a Domain Routing Matching

Domain Routing Matching Table	
Attribute	Value Example
<i>cscfDomainRoutingMatchingTableEntryId</i>	one.xyz.net
<i>cscfDomainRoutingMatchingTableEntryMode</i>	1
<i>cscfDomainRoutingMatchingTableEntryResult</i>	NextHop1
<i>cscfDomainRoutingMatchingTableEntryId</i>	two.xyz.net
<i>cscfDomainRoutingMatchingTableEntryMode</i>	2
<i>cscfDomainRoutingMatchingTableEntryResult</i>	NextHop2
<i>cscfDomainRoutingMatchingTableEntryId</i>	xyz.net
<i>cscfDomainRoutingMatchingTableEntryMode</i>	3
<i>cscfDomainRoutingMatchingTableEntryResult</i>	NextHop3
<i>cscfDomainRoutingMatchingTableEntryId</i>	abc.xyz.net
<i>cscfDomainRoutingMatchingTableEntryMode</i>	1
<i>cscfDomainRoutingMatchingTableEntryResult</i>	NextHop1
<i>cscfDomainRoutingMatchingTableEntryId</i>	net

Domain Routing Matching Table	
<code>cscfDomainRoutingMatchingTableEntryMode</code>	1
<code>cscfDomainRoutingMatchingTableEntryResult</code>	NextHop2

Table 79 Example of a Domain Routing Result

Domain Routing Result	
Attribute	Value Example
<code>cscfDomainRoutingResultTableEntryId</code>	NextHop1
<code>cscfDomainRoutingResultTargetAddress</code>	abc.xyz.net ; priority = 1
<code>cscfDomainRoutingResultTargetAddress</code>	def.xyz.net :5062 ; transport=UDP ; priority = 2
<code>cscfDomainRoutingResultTableEntryId</code>	NextHop2
NextHop2	123.123.123.123 ; transport = TCP ; priority = 1
<code>cscfDomainRoutingResultTableEntryId</code>	NextHop3
<code>cscfDomainRoutingResultTargetAddress</code>	111.111.111.111 ; priority = 1

Match result: match to "one.xyz.net"

Next hop address list: (In Mode-1, the original FQDN is not included.)

abc.xyz.net ; priority = 1

def.xyz.net :5062 ; transport=UDP ; priority = 2

Example 1 Original FQDN: one.xyz.net

Match result: match to "two.xyz.net"

Next hop address list: (In Mode-2, the original FQDN is included ⇒ with the lowest priority, 0 in this example)

123.123.123.123 ; transport = TCP ; priority = 1

two.xyz.net ; priority = 0

Example 2 Original FQDN: two.xyz.net



```
Match result: Original FQDN does not match any configured domain name =>
entries in the matching table. The SIP request is routed to the =>
original FQDN, one.xyz.com.
Next hop address list:
  one.xyz.com
```

Example 3 Original FQDN: one.xyz.com

3.18.3 Manage Domain Routing Function

This section describes how to manage the DRF.

To update the DRF:

1. Update the configuration tables with the new changes
2. Set `cscfDomainRoutingSyncConfig` to `true`
3. Wait until `cscfDomainRoutingSyncConfig` has been set to `False` and make sure that `cscfDomainRoutingSyncState` has taken the value `synchronized`

Attribute `cscfDomainRoutingSyncState` is set to `synchronized` if the configuration is successfully converted to runtime data. The runtime data is used to handle traffic when the attribute assumes the value `synchronized`.

If synchronization fails, the value is set to `synchronization_failed` and an alarm is raised identifying the reason for the failure. After the cause of the failure is corrected, `cscfDomainRoutingSyncConfig` can be set to `True` again to synchronize the data.

Users do not set the value of `cscfDomainRoutingSyncConfig` to `false`. It is set to `false` automatically when synchronization is completed.

3.19 Configure Throttling of Diameter Traffic on Cx/Dx Interface

When the CSCF sends a request to an HSS node that is experiencing overload, the processing of the request can worsen the overload situation on that node. The CSCF can reduce the number of sent requests to a node that it detects to be overloaded. This process is called throttling. Throttling aids recovery of the overloaded node.

Note: Throttling is a robustness function to avoid HSS node failures, the consequence can however be that the traffic throughput is reduced.

The CSCF monitors HSS responses on Cx/Dx interface over a period (measurement window). Based on the percentage of busy responses received

per requests sent, the CSCF can determine if HSS is overloaded. The CSCF can also determine if the overload is improving or worsening over time.

When HSS overload has been determined, Cx/Dx requests are throttled at a configured rate until HSS is no longer found to be overloaded. A throttled Cx/Dx request is not delivered to HSS. Processing continues as though the request was sent but a busy response was received.

For an overview of the throttling algorithm used, see Figure 13.

The percentage of busy responses received for the total requests sent over a measurement window are shown as solid dots, as follows:

- When a measurement is lower than the configurable lower threshold, the throttling rate is decreased by a configurable amount.
- When the measurement is higher than the configurable upper threshold, the throttling rate is increased by a configurable amount.
- When the measurement is between the upper and lower threshold, the throttling rate remains unchanged.

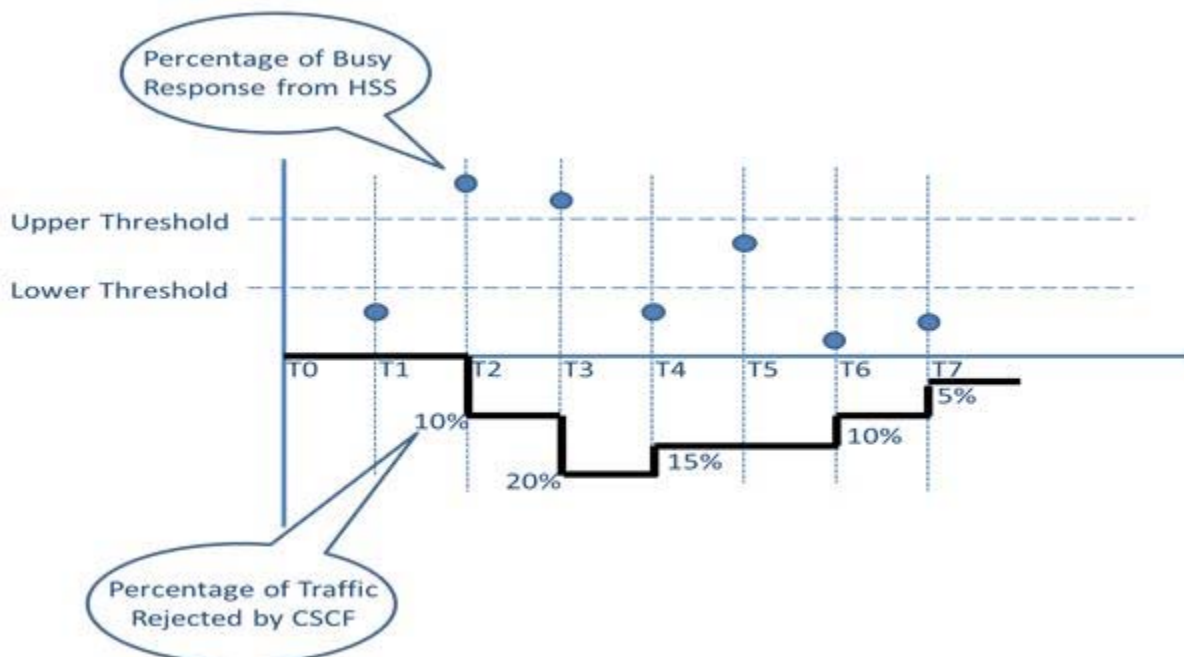


Figure 13 Throttling Algorithm

CM object `cscfThrottledInterfaceId=CxDx` is preconfigured in the database.

3.19.1 Enable Throttling Feature

To enable the throttling feature:



1. Set configuration parameter *cscfThrottlingEnabled* to `true`.

3.19.2 Modify Throttling Attributes

To modify the duration of the measurement window:

1. Set configuration parameter *cscfThrottlingWindowLength* to the duration in seconds.

The window duration must be configured to allow the feature to be responsive to the HSS overload situation without over-reacting to small glitches.

To modify the throttling rate steps (increases and decreases):

1. Set the configuration parameters *cscfThrottlingRateIncrease* and *cscfThrottlingRateDecrease* to a percentage value.

The steps must be small enough to ensure smooth functioning of the feature without making the recovery too long.

To modify the upper and lower thresholds:

1. Set configuration parameters *cscfUpperThrottlingThreshold* and *cscfLowerThrottlingThreshold* to a percentage value.

The upper threshold must be greater than the lower threshold. The thresholds must be configured to allow the feature to be responsive to the HSS overload situation without over-reacting to small glitches.

When modifying any of the parameters in the procedures, the new values take effect in the next measurement window.

The example values of throttling attributes are listed in Table 80.

Table 80 Example Values of Throttling Attributes

Attribute	Value Example ⁽¹⁾
<i>cscfThrottlingEnabled</i>	true
<i>cscfThrottlingWindowLength</i>	4
<i>cscfThrottlingRateIncrease</i>	7
<i>cscfThrottlingRateDecrease</i>	15
<i>cscfUpperThrottlingThreshold</i>	15
<i>cscfLowerThrottlingThreshold</i>	5

(1) The chosen values are very much dependent on the HSS characteristics to manage overload situations, and the traffic model that applies in the live network. This example is a suggestion that manages 200% overload traffic towards an Ericsson HSS. To ensure optimum performance of this function on a specific site, these values can be properly tuned based on its specific traffic model, network setup, and the characteristics and dimensioning of the HSS that I/S-CSCF communicates with.

3.20 Configure Destination Blacklisting

This section describes how to configure destination blacklisting in the CSCF.

If a destination host, usually the next hop, for some reason indicates that it is not available to receive traffic, the host can be blacklisted in the CSCF.

The reason can be any of the following:

- A SIP 503 Response with Retry-After header for all SIP Requests (INVITE or Non-INVITE) is received.
- A SIP 503 Response without Retry-After header for all SIP Requests (INVITE or Non-INVITE) is received.
- A transaction time-out for SIP INVITE.
- Fatal transport errors (socket errors or indicated by the Internet Control Message Protocol (ICMP)) for all SIP Requests are received.

There are five configuration parameters for configuring blacklisting thresholds, one for each blacklisting reason, as follows:

- *cscfBlacklistingSipTransactionTimeoutThreshold*
- *cscfBlacklistingSip503WithRetryAfterThreshold*
- *cscfBlacklistingSip503WithoutRetryAfterThreshold*
- *cscfBlacklistingTransportErrorThreshold*
- *icmpIntermittentTolerance*

For more information about the configuration parameters, see Section 3.20.2 Configure Blacklisting Threshold on page 109.

Depending on the reason, an entire host or individual ports (including its transport protocols) is blacklisted when the number of error events exceeds the configured threshold. A destination can be removed from the blacklist when its timer configuration expires, see Section 3.20.1 Configure Timer on page 109. There is no specific parameter to enable or disable the blacklisting function. But in practice, it is possible to disable the blacklisting function by configuring the thresholds to high values.

Besides the default configuration for destination blacklisting, the CSCF supports configuring destination blacklisting per FQDN. If one or more FQDNs of destinations are defined by attribute *cscfDestinationSipAddress*, one or more blacklisting configurations per FQDN can exist. The configuration parameters for configuring blacklisting thresholds and timer per FQDN of destination are as follows:

- *cscfDestinationSipAddress*



- `cscfDestinationUnavailabilityTimerDest`
- `cscfBlacklistingSipTransactionTimeoutThresholdDest`
- `cscfBlacklistingSip503WithRetryAfterThresholdDest`
- `cscfBlacklistingSip503WithoutRetryAfterThresholdDest`
- `cscfBlacklistingTransportErrorThresholdDest`

3.20.1 Configure Timer

Attribute `icmpBarringTime` is used to define the time in seconds to regard destinations as unreachable. This configuration is only used for blacklisting a destination because of ICMP.

Both attributes `CscfDestinationUnavailabilityTimer` and `cscfDestinationUnavailabilityTimerDest` are used to define the time in seconds for how long the CSCF blacklists an unavailable destination in the network for the following blacklisting reasons:

- SIP 503 without Retry After header
- Transaction Time-out
- Fatal Transport Error

For the blacklist reason 503 with Retry-After header, the timer is taken directly from the value in the header.

If `cscfDestinationUnavailabilityTimerDest` is set to **default**, when the FQDNs of destinations are set in attribute `cscfDestinationSipAddress`, `CscfDestinationUnavailabilityTimer` is used. Otherwise, `cscfDestinationUnavailabilityTimerDest` is used.

3.20.2 Configure Blacklisting Threshold

Each blacklisting reason has a configurable threshold that must be exceeded before the destination host becomes blacklisted. Each threshold is counted individually per destination host and reason.

The threshold level is measured during a measurement period. The measurement period starts when the first blacklisting event occurs and is prolonged every time the same blacklisting event occurs within its timer configuration (see Section 3.20.1 Configure Timer on page 109). The measurement period ends if no additional blacklisting event occurs within its timer configuration (see Section 3.20.1 Configure Timer on page 109), or when the threshold level is exceeded and the destination host becomes blacklisted. The measurement period is restarted after every new blacklisting event occurs. The counting for threshold is reset when the measurement period is ended.

Attribute *cscfBlacklistingSipTransactionTimeoutThreshold* is used to configure the threshold for blacklisting destinations because of SIP transaction time-out.

Attribute *cscfBlacklistingSip503WithRetryAfterThreshold* is used to configure the threshold for blacklisting destinations because of 503 Response with Retry-After.

Attribute *cscfBlacklistingSip503WithoutRetryAfterThreshold* is used to configure the threshold for blacklisting destinations because of 503 Response without Retry-After.

Attribute *cscfBlacklistingTransportErrorThreshold* is used to configure the threshold for blacklisting destinations because of fatal transport error (socket error).

Attribute *icmpIntermittentTolerance* is used to configure the threshold for blacklisting destinations because of ICMP.

If the FQDNs of destinations are set in attribute *cscfDestinationSipAddress*, the following attributes are used to configure the thresholds for blacklisting destinations per FQDN:

- Attribute *cscfBlacklistingSipTransactionTimeoutThresholdDest* is used to configure the threshold for blacklisting destinations because of SIP transaction time-out. The value range is 0–2147483647. The default value is **default**. When **default** is used, *cscfBlacklistingSipTransactionTimeoutThreshold* is used.
- Attribute *cscfBlacklistingSip503WithRetryAfterThresholdDest* is used to configure the threshold for blacklisting destinations because of 503 Response with Retry-After. The value range is 0–2147483647. The default value is **default**. When **default** is used, *cscfBlacklistingSip503WithRetryAfterThreshold* is used.
- Attribute *cscfBlacklistingSip503WithoutRetryAfterThresholdDest* is used to configure the threshold for blacklisting destinations because of 503 Response without Retry-After. The value range is 0–2147483647. The default value is **default**. When **default** is used, *cscfBlacklistingSip503WithoutRetryAfterThreshold* is used.
- Attribute *cscfBlacklistingTransportErrorThresholdDest* is used to configure the threshold for blacklisting destinations because of Fatal transport error (socket error). The value range is 0–2147483647. The default value is **default**. When **default** is used, *cscfBlacklistingTransportErrorThreshold* is used.

3.20.3 Configure Blacklisting Bypass

It is possible to configure the CSCF to bypass the blacklisting when all alternative transport addresses towards a destination are blacklisted. The



attribute *CscfBlacklistingBypassThrottle* is used to control how much of initial SIP requests can be sent to destinations that have been blacklisted as unreachable for any reason, except SIP Transaction Time-out. The percentage of requests that can be sent in this case can be defined. If all destinations are blacklisted, it is possible to override some or all blacklisted destinations for sending requests.

The *CscfBlacklistingSipTransactionTimeoutBypassThrottle* works the same way but only has effect if the reason for blacklisting was SIP Transaction Time-out.

The *cscfBlacklistingInsideDialogRequestBypassThrottle* parameter controls how much of inside dialogue SIP requests are sent to destinations that have been blacklisted as unreachable for any reason. The percentage of inside dialogue SIP requests that are sent in this case can be defined.

3.20.4

Example Configuration of Destination Blacklisting

If *CscfBlacklistingSipTransactionTimeoutBypassThrottle* = 0 and *CscfBlacklistingBypassThrottle* = 100, the CSCF does not ever send an initial SIP request towards a destination that is blacklisted because of SIP Transaction Time-out, but sends all initial SIP requests towards destinations blacklisted because of any other reason.

If *CscfBlacklistingSipTransactionTimeoutBypassThrottle* = 0, *CscfBlacklistingBypassThrottle* = 0, and *cscfBlacklistingInsideDialogRequestBypassThrottle* = 100, the CSCF sends inside dialogue requests instead of initial SIP requests towards destinations that are blacklisted for any reason.

Table 81 shows an example configuration of destination blacklisting.

Table 81 Example Configuration of Destination Blacklisting

Attribute	Value Example
<i>CscfDestinationUnavailabilityTimer</i>	60
<i>icmpBarringTime</i>	20
<i>cscfBlacklistingSipTransactionTimeoutThreshold</i>	0
<i>cscfBlacklistingSip503WithRetryAfterThreshold</i>	5
<i>cscfBlacklistingSip503WithoutRetryAfterThreshold</i>	5
<i>cscfBlacklistingTransportErrorThreshold</i>	5

Attribute	Value Example
<i>icmpIntermittentTolerance</i>	0
<i>CscfBlacklistingBypassThrottle</i>	100
<i>CscfBlacklistingSipTransactionTimeoutBypassThrottle</i>	0
<i>cscfBlacklistingInsideDialogRequestBypassThrottle</i>	100

3.20.5

Example Configuration of Destination Blacklisting per FQDN

Table 82 shows an example of single configuration of destination blacklisting for one FQDN. In this case, one group of attributes with the key as defined by *cscfSipConfigDestProfileId* are set and one FQDN is set by *cscfDestinationSipAddress*.

Table 82 Example Single Configuration of Destination Blacklisting for One FQDN

Attribute	Value Example
<i>cscfSipConfigDestProfileId</i>	ERICSSON_AS
<i>cscfDestinationSipAddress</i>	mtas1.com
<i>cscfDestinationUnavailabilityTimerDest</i>	50
<i>cscfBlacklistingSipTransactionTimeoutThresholdDest</i>	default
<i>cscfBlacklistingSip503WithRetryAfterThresholdDest</i>	5
<i>cscfBlacklistingSip503WithoutRetryAfterThresholdDest</i>	10
<i>cscfBlacklistingTransportErrorThresholdDest</i>	default

Table 83 shows an example of single configuration of destination blacklisting for multiple FQDNs. In this case, one group of attributes with the key as defined by *cscfSipConfigDestProfileId* are set and more than one FQDN (two FQDNs for this example) are set by *cscfDestinationSipAddress*.

Table 83 Example Single Configuration of Destination Blacklisting for Multiple FQDN

Attribute	Value Example
<i>cscfSipConfigDestProfileId</i>	SBG
<i>cscfDestinationSipAddress</i>	sbg1.com, sbg2.com



Attribute	Value Example
<i>cscfDestinationUnavailabilityTimerDest</i>	50
<i>cscfBlacklistingSipTransactionTimeoutThresholdDest</i>	default
<i>cscfBlacklistingSip503WithRetryAfterThresholdDest</i>	5
<i>cscfBlacklistingSip503WithoutRetryAfterThresholdDest</i>	10
<i>cscfBlacklistingTransportErrorThresholdDest</i>	default

Table 84 shows an example of multiple configurations of destination blacklisting for multiple FQDNs. In this case, more than one group of attributes (two groups for this example) with keys as defined by *cscfSipConfigDestProfileId* and one FQDN is set by *cscfDestinationSipAddress* per group.

Table 84 Example Multiple Configuration of Destination Blacklisting for Multiple FQDNs

Attribute	Value Example
<i>cscfSipConfigDestProfileId</i>	MAILSERVER_AS
<i>cscfDestinationSipAddress</i>	mail.com
<i>cscfDestinationUnavailabilityTimerDest</i>	50
<i>cscfBlacklistingSipTransactionTimeoutThresholdDest</i>	default
<i>cscfBlacklistingSip503WithRetryAfterThresholdDest</i>	5
<i>cscfBlacklistingSip503WithoutRetryAfterThresholdDest</i>	10
<i>cscfBlacklistingTransportErrorThresholdDest</i>	default
<i>cscfSipConfigDestProfileId</i>	ERICSSON_AS
<i>cscfDestinationSipAddress</i>	as5070.com
<i>cscfDestinationUnavailabilityTimerDest</i>	20
<i>cscfBlacklistingSipTransactionTimeoutThresholdDest</i>	5
<i>cscfBlacklistingSip503WithRetryAfterThresholdDest</i>	default

Attribute	Value Example
<i>cscfBlacklistingSip503WithoutRetryAfterThresholdDest</i>	5
<i>cscfBlacklistingTransportErrorThresholdDest</i>	10

3.21 Configure Re-Registration with HSS Bypass

When the re-registration with HSS bypass feature is enabled, I-CSCF routes the request to S-CSCF indicated in the Route header if the HSS inquiry fails because of HSS overloaded, no reply, or the request not sent because of throttling, see Section 3.19 Configure Throttling of Diameter Traffic on Cx/Dx Interface on page 105. This feature is enabled or disabled by using a parameter *cscfReRegWithHssBypass*. This parameter must be enabled in both external P-CSCF and I-CSCF if standalone deployment for the feature to work.

The re-registration with Authentication bypass feature allows the S-CSCF to process the re-registration without authentication if the HSS authentication inquiry fails because of HSS overloaded, no reply, or the authentication request not sent because of throttling. This feature is enabled or disabled by using a parameter *scscfReRegWithAuthBypass*, which is independent of the re-registration With HSS Bypass feature parameter *cscfReRegWithHssBypass*.

For the CSCF to process the re-registration when HSS inquiry fails because of the reasons listed in this section, the two parameters must be enabled.

3.21.1 Enable Re-Registration with HSS Bypass

To enable re-registration with HSS bypass:

1. Set the attribute *cscfReRegWithHssBypass* to `BYPASS_IF_FAILED`.

3.21.2 Disable Re-Registration with HSS Bypass

To disable re-registration with HSS bypass (default configuration):

1. Set the attribute *cscfReRegWithHssBypass* to `NO_BYPASS`.

3.21.3 Enable Re-Registration with Authentication Bypass

To enable re-registration with Authentication Bypass:

1. Set the attribute *scscfReRegWithAuthBypass* to `BYPASS_IF_FAILED`.

3.21.4 Disable Re-Registration with Authentication Bypass

To disable re-registration with Authentication Bypass (default configuration):



1. Set the attribute *scscfReRegWithAuthBypass* to NO_BYPASS.

3.22 Configure P-CSCF Restoration Procedure

This section describes how to enable and configure the P-CSCF Restoration Procedure.

When the S-CSCF tries to send a terminating request to a P-CSCF and the P-CSCF is blacklisted or responds with a SIP error response code matching an error code in parameter, *scscfPcscfRestorationSipErrorCodes*, S-CSCF triggers the HSS-based P-CSCF restoration procedure for the terminating user. It forces the affected UE to perform an initial registration and then a working P-CSCF is assigned.

Enabling this feature is beneficial because users registered in a P-CSCF that is not responding properly, can resume the services if a terminating call is received before any invocation of the IMS system.

3.22.1 Enable P-CSCF Restoration Procedure

This section described how to enable the P-CSCF Restoration Procedure.

To enable the P-CSCF Restoration Procedure:

1. If it is preferred to trigger P-CSCF Restoration Procedure with SIP error codes, configure the SIP error codes for triggering the P-CSCF restoration in the configuration parameter, *scscfPcscfRestorationSipErrorCodes*.

Note: Multiple SIP error codes can be defined by configuring multiple *scscfPcscfRestorationSipErrorCodes* attribute instances.

2. Enable the P-CSCF Restoration Procedure by setting the configuration parameter, *scscfPcscfRestorationEnabled*, to TRUE.

If *scscfPcscfRestorationSipErrorCodes* is not configured and *scscfPcscfRestorationEnabled* is enabled, the P-CSCF restoration is triggered only when the P-CSCF is blacklisted for reasons except receiving SIP 503 with or without a Retry-After header.

After enabling, any changes to *scscfPcscfRestorationSipErrorCodes* will take effect in the next terminating request.

3.22.2 P-CSCF Restoration Procedure Configuration Example

This section gives an example of a possible P-CSCF Restoration Procedure configuration when the P-CSCF Restoration Procedure is to be triggered on multiple SIP error codes, for example 404, 408, 486, 500, 503, and 504. In such cases, multiple instances of *scscfPcscfRestorationSipErrorCodes* must be configured.

Table 85 *P-CSCF Restoration Procedure Configuration Example when Triggered on Multiple SIP Error Codes*

Attributes	Value Examples
<code>scscfPcscfRestorationEnabled</code>	TRUE
<code>scscfPcscfRestorationSipErrorCodes</code>	404
<code>scscfPcscfRestorationSipErrorCodes</code>	408
<code>scscfPcscfRestorationSipErrorCodes</code>	486
<code>scscfPcscfRestorationSipErrorCodes</code>	500
<code>scscfPcscfRestorationSipErrorCodes</code>	503
<code>scscfPcscfRestorationSipErrorCodes</code>	504

3.23 Configure Media Type Counters

To measure statistics of INVITE sessions based on different media types, the following counters, called media type counters, are provided in S-CSCF:

- `scscfOrigAttemptedSessionPerMedia`
- `scscfOrigSuccessfulEstablishedSessionPerMedia`
- `scscfOrigAnsweredSessionPerMedia`
- `scscfOrigFailedSessionPerMedia`
- `scscfOrigAccumulatedCallTimePerMedia`
- `scscfTermAttemptedSessionPerMedia`
- `scscfTermSuccessfulEstablishedSessionPerMedia`
- `scscfTermAnsweredSessionPerMedia`
- `scscfTermFailedSessionPerMedia`
- `scscfTermAccumulatedCallTimePerMedia`

For more information about the media type counters, refer to *Managed Object Model (MOM)*.

It is possible to enable or disable media type counters by using configuration parameter `scscfMediaTypeCountersEnabled`.



The media type counters are incremented only if the parameter *cscfMediaTypeCountersEnabled* is set to `true`. If the parameter is set to `false`, the counters are not incremented.

The default value is `false`.

3.24 Configure Active User License Model

This section describes how to configure active user license model functionality.

Active users can be of three types, Personal, Advanced, or Professional, each type contains two user characteristics; number of contacts, IMS Public Users (IMPU).

If the active user has surpassed any of the user characteristics of Personal Active Users, the active user is counted as an Advanced Active User as well. If the active user has surpassed any of the user characteristics of Advanced Active Users, the active user is counted as a Professional Active User as well.

The active user license model feature is controlled by the configuration attribute *cscfActiveUserLicenseModelEnabled*.

To enable the active user license model feature:

1. Set the *cscfActiveUserMeasurementInterval* attribute to the measurement interval as the unit of time that applies to the active user licenses (possible values: `oneDay`, `oneWeek`, `tenDays`, `oneMonth`).

If the Measurement interval configuration value is changed for any of the following values, the first report is partial:

- `oneDay`: the measurement interval is the period from midnight to the following midnight. If, for example, the configuration value is changed to `oneDay` at 5 PM, the active user counter would reflect the value from 5 PM to midnight, and then a new counter value would begin from midnight to the following midnight
 - `oneWeek`: the measurement interval is based it on the calendar year, that is, weeks 1–52.
 - `tenDays`: the measurement interval is from day 1 of the calendar month to day 10, day 11 to day 20, day 21 to day 28/29/30, and day 31 (if needed)
 - `oneMonth`: the measurement interval is the whole calendar month (January, February, and so on)
2. Set Personal Active User attributes as described in Section 3.24.1 Personal Active User on page 118.
 3. Set Advanced Active User attributes as described in Section 3.24.2 Advanced Active User on page 118.



4. Set *cscfActiveUserLicenseModelEnabled* attribute to true.

3.24.1 Personal Active User

The maximum number of IMPUs, maximum number of contacts, and MultiDevice configuration settings can be configured for Personal Active Users.

Three configuration parameters for Personal Active Users are *cscfPersonalMaxNumOfIMPUs*, *cscfPersonalMaxNumOfContacts*, and *cscfMultiDeviceNumberOfContacts*.

The attribute *cscfPersonalMaxNumOfIMPUs* is used to indicate the maximum number of IMPUs for the Personal Active User Profile. The attribute *cscfPersonalMaxNumOfIMPUs* cannot be greater than *cscfAdvancedMaxNumOfIMPUs*.

The attribute *cscfPersonalMaxNumOfContacts* is used to indicate the maximum number of Contacts for the Personal Active User Profile. The attribute *cscfPersonalMaxNumOfContacts* cannot be greater than *cscfAdvancedMaxNumOfContacts* or *cscfMultiDeviceNumberOfContacts*(if present).

The attribute *cscfMultiDeviceNumberOfContacts* is the maximum number of contacts for a Personal Active User if the Multi-Device feature is enabled. The attribute must be greater than *cscfPersonalMaxNumOfContacts* and must be lower than *cscfAdvancedMaxNumOfContacts*. After the attribute *cscfMultiDeviceNumberOfContacts* is created, it must be deleted from the configuration if the Multi-Device feature is to be disabled. For more information on the Multi-Device feature, see Section 3.25.1 Configure the Multi-Device Feature on page 119.

An example setting of the Personal Active User attributes is shown in Table 86.

Table 86 Personal Active User Attributes

Attribute	Value Example
<i>cscfPersonalMaxNumOfContacts</i>	2
<i>cscfPersonalMaxNumOfIMPUs</i>	3
<i>cscfMultiDeviceNumberOfContacts</i>	6

3.24.2 Advanced Active User

The maximum number of IMPUs, maximum number of contacts, and Professional feature configuration settings can be configured for Advanced Active Users. Three configuration parameters can be updated for Advanced Active Users *cscfAdvancedMaxNumOfIMPUs*, *cscfAdvancedMaxNumOfContacts*, and *cscfProfessionalActiveUsersEnabled*.



The attribute `cscfAdvancedMaxNumOfIMPUs` is used to indicate the maximum number of IMPUs for the Advanced Active User Profile. The attribute `cscfAdvancedMaxNumOfIMPUs` must be greater than `cscfPersonalMaxNumOfIMPUs`.

The attribute `cscfAdvancedMaxNumOfContacts` is used to indicate the maximum number of Contacts for the Advanced Active User Profile. The attribute `cscfAdvancedMaxNumOfContacts` must be greater than `cscfPersonalMaxNumOfContacts` and `cscfMultiDeviceNumberOfContacts` (if present).

An example setting of the Advanced Active User attributes is shown in Table 87.

Table 87 Advanced Active User Attributes

Attribute	Value Example
<code>cscfAdvancedMaxNumOfContacts</code>	10
<code>cscfAdvancedMaxNumOfIMPUs</code>	10
<code>cscfProfessionalActiveUsersEnabled</code>	true

The attribute `cscfProfessionalActiveUsersEnabled` enables or disables the Professional Active User Function. The Professional Active Users have user characteristics that surpass the user characteristics of Advanced Active Users. For example, if the attribute `cscfProfessionalActiveUsersEnabled` = true, the `cscfAdvancedMaxNumOfContacts` = 10, and a user has 11 contacts, then the user is counted as a Professional Active User.

3.25 Configure Licensed Features

This section describes how to configure licensed features.

Note: Before configuring a licensed feature, make sure that a license for the licensed feature is installed. If the license is not installed, an alarm that indicates the unavailable license is raised by the license server when configuring the feature.

For information on installing a license, refer to *CSCF License Management*.

3.25.1 Configure the Multi-Device Feature

This section describes how to configure the Multi-Device feature. The feature can be enabled when the Active User License Model feature is enabled. Using the Multi-Device feature expands the limitation of the number of contacts from two to six for a Personal User. For information on the active user license model feature, see Section 3.24 Configure Active User License Model on page 117.

To configure the Multi-Device feature:

1. Make sure that the *cscfMultiDeviceNumberOfContacts* attribute is set in Section 3.24 Configure Active User License Model on page 117.
2. Set the *scscfMultiDeviceAdministrativeState* attribute to UNLOCKED.

3.25.1.1 Multi-Device Attributes

For the attribute *cscfMultiDeviceNumberOfContacts* configuration description, see Section 3.24.1 Personal Active User on page 118.

The attribute *scscfMultiDeviceAdministrativeState* locks or unlocks the Multi-Device feature. For *scscfMultiDeviceAdministrativeState* to be UNLOCKED, the attribute *cscfMultiDeviceNumberOfContacts* must be set.

This attribute *scscfMultiDeviceServiceState* is used to show whether the Multi-Device feature is ENABLED or DISABLED as follows:

- If *scscfMultiDeviceServiceState* is to be ENABLED, the attribute *scscfMultiDeviceAdministrativeState* must be UNLOCKED and the Multi-Device license must be available.
- If *scscfMultiDeviceServiceState* is DISABLED, the functionality of the feature is not executed. *cscfMultiDeviceNumberOfContacts* is not used to determine the maximum number of contacts for the Personal Active User Profile. As a result, active users with more than two contacts are considered as advanced users.

An example setting of the Multi-Device attributes is shown in Table 88.

Table 88 Multi-Device Attributes

Attribute	Value Example
<i>scscfMultiDeviceAdministrativeState</i>	UNLOCKED
<i>scscfMultiDeviceServiceState</i>	ENABLED

3.25.2 Configure Wi-Fi Calling Feature

This section describes how to configure the Wi-Fi Calling feature. The CSCF supports devices for making calls over a Wi-Fi access network. If the PANI includes a Wi-Fi access type and a license is not available for the Wi-Fi calling feature, the CSCF blocks SIP requests except emergency requests.

To configure the Wi-Fi Calling feature:

1. Set the *scscfWifiCallingAdministrativeState* attribute to UNLOCKED.



3.25.2.1 Wi-Fi Calling Attributes

The attribute *scscfWifiCallingAdministrativeState* locks or unlocks the Wi-Fi Calling feature.

The attribute *scscfWifiCallingServiceState* is used to show whether the Wi-Fi Calling feature is ENABLED or DISABLED as follows:

- If *scscfWifiCallingServiceState* is to be ENABLED, the attribute *scscfWifiCallingAdministrativeState* must be UNLOCKED and the Wi-Fi calling license must be available.
- If *scscfWifiCallingServiceState* is DISABLED, the functionality of the feature is not executed. For every SIP request, the PANI access type is checked against a Wi-Fi access type. A Wi-Fi access type includes IEEE-802.11. For example, *access-type = IEEE-802.11n*. If a match is found, then SIP 403 Forbidden is returned to the user.

An example setting of the Wi-Fi calling attributes is shown in Table 89.

Table 89 Wi-Fi Calling Attributes

Attribute	Value Example
<i>scscfWifiCallingAdministrativeState</i>	UNLOCKED
<i>scscfWifiCallingServiceState</i>	ENABLED

3.26 Configure I-CSCF Resource Broker Entry

This section describes how to configure server addresses in *cscfResourceBrokerEntry*.

The server address is typically an S-CSCF address, but can also be an AS address.

cscfResourceBrokerEntry is used by the I-CSCF to select a server address, based on matching the capabilities received from the HSS to the provisioned capabilities stored in the resource broker with the list of server addresses.

Each *cscfResourceBrokerEntry* entry defines a server address with supported capabilities and with a unique priority indication. A maximum of 500 server address entries can be configured, and each server address can have a maximum of 16 capabilities. An empty *cscfResourceBrokerEntry* indicates that the collocated S-CSCF must be selected (if there is one).

The syntax of the *cscfResourceBrokerEntry* is:

<Priority-nr>:<Capabilities>:<SIP-URI>

Priority-nr is an unsigned integer value. A higher value of **Priority-nr** means lower priority, that is, 0 = max priority.

Capabilities is a comma-separated list of maximum 16 positive numbers. Each number can only exist ones per entry. 0 (zero) is not considered as a valid value.

SIP-URI is a standard SIP URI that contains an S-CSCF name but no lr-parameter.

See Table 90 for an example configuration of *cscfResourceBrokerEntry*.

Table 90 Example Configuration of *cscfResourceBrokerEntry*

applicationName=Cscf, CscfResourceBroker=0	
Attribute	Value Example
<i>cscfResourceBrokerEntry</i>	0:1,2:sip:scscf1.com
<i>cscfResourceBrokerEntry</i>	15:1,2,3,4,5,6:sip:[1234::1234:1234]

Note: Performance of initial registration and some traffic use cases are significantly affected if more than 100 entries are configured in the parameter. Especially, if optional capabilities are provisioned in HSS, the performance can be significantly degraded. If many entries are required, it is recommended to avoid provisioning optional capabilities in HSS. An alternative to configure many entries in the resource broker, is to provision specific Server-Names directly in HSS.

3.27 Configure Shutting Down State in CSCF

Each CSCF Network Element has one administrative state configuration parameter whether the CSCF is a standalone node or a collocated node of any eligible combinations. For example, in a collocated IS-CSCF node with the E-CSCF enabled, the I-CSCF, the S-CSCF, and the E-CSCF have the same administrative state.

The other administrative states are **Locked** (0) and **Unlocked** (1). **Unlocked** takes the CSCF into operation and **Locked** takes the CSCF out of operation after Network Initiated Deregistering the users, leading to the graceful termination of calls (**BYEs**), third-party deregistration, and possible **RegEvent** for all registered users. The administrative state is **Locked** immediately, but it can take a considerable time before the operational state becomes **Disabled**.

The CSCF is gracefully taken out of operation with minimal traffic disturbance when it is in **Shutting Down** state. When the node can be taken out of operation without any end-user disturbance, it automatically goes to administrative state **Locked**. However, it is possible to set the administrative state manually to **Locked** or **Unlocked** when in **Shutting Down**. Users can be moved with minimal disturbance to redundant CSCF nodes during the period that the CSCF is in **Shutting Down** state.



3.27.1 S-CSCF “Shutting Down” State Behavior

It is recommended that the S-CSCF Restoration Procedure, see Section 3.14 Configure IMS Restoration on page 87, is enabled in the IMS network to minimize disturbances for users during the *Shutting Down* of an S-CSCF.

A standalone and collocated S-CSCF accepts and processes all initial REGISTER, re-REGISTER, and REGISTER of a new contact if the user has an ongoing INVITE session. Otherwise, the S-CSCF can invoke the user redistribution function, see Section 3.28.2 Configure User Redistribution in S-CSCF on page 125, or reject the user registration.

A standalone and collocated S-CSCF accepts and processes all REGISTER QUERY, de-REGISTER, and non-REGISTER requests as if the node was in *Unlocked state*.

This means that the registered users are gradually moved to other serving nodes.

To configure *Shutting Down* for an S-CSCF:

1. Set *scscfRegisteredUserThreshold* to 0
2. Set *cscfAdministrativeState* to **SHUTTING_DOWN** (2)

Note: When all registered users are moved and all ongoing sessions are terminated, the node is automatically set to the administrative state *Locked* (0).

According to 3GPP TS 24.229, a user can re-register 10 minutes before expiration. Considering the configured maximum time *cscfRegistrationRefreshMax* is set to 1 hour by default, most registered users are moved within 50 minutes.

If the administrative state remains **SHUTTING_DOWN** (2) after 50 minutes, *cscfAdministrativeState* can be manually set to *Locked* (0). After the administrative state *cscfAdministrativeState* of the node is set to *Locked* (0), the S-CSCF triggers session termination and network initiated de-registration for all remaining users that are in the registered and unregistered states. The session termination and network initiated de-registration are operated in a pace of 100 users per second by default. To monitor the number of registered users in the node, check the PM counter *cscfConcurrentRegisteredUserProfiles*.

3.27.2 I-CSCF and BCF Shutting Down State Behavior

A standalone I-CSCF and BCF transit to administrative state *Locked* directly, as they do not keep any user or dialogue states.

A collocated I-CSCF and BCF handle all SIP requests as if the CSCF was in `Unlocked` state, and goes to `Locked` state once the collocated P- or S-node is locked.

To configure `Shutting Down` for an I-CSCF and BCF:

1. Set `cscfAdministrativeState` to 2

3.27.3 E-CSCF and EATF Shutting Down State Behavior

Standalone and collocated E-CSCF and EATF handle all SIP requests and treat the requests in `Unlocked` state. They go to administrative state `Locked` when no new emergency traffic is received and all ongoing sessions are terminated.

To configure `Shutting Down` for an E-CSCF and EATF:

1. Set `cscfAdministrativeState` to 2

3.28 Configure User Redistribution in CSCF

The user redistribution function is used to redistribute users from one S-CSCF to another S-CSCF.

It is possible to redistribute registered users in the S-CSCF using several methods. One method is triggered in the I-CSCF and the others are triggered in the S-CSCF. It is recommended that user redistribution is enabled in one node at a time. If enabled in both, the redistribution that is triggered by the S-CSCF overrides the redistribution attempt from the I-CSCF.

Note: As a prerequisite for this function to work, the HSS must allow changing the stored S-CSCF host in HSS. This requires that IMS Restoration must be enabled in the HSS or that authentication function in targeting S-CSCF is enabled to send a Cx MAR to the HSS for allowing a new S-CSCF host to be stored in the HSS.

The following are example scenarios where the functionality is useful:

- A new S-CSCF node is added to the IMS network
- After a previously failed S-CSCF that is back in service
- An S-CSCF node is to be shut down for maintenance
- An S-CSCF put back into service after maintenance
- To rebalance the number of registered users in the different S-CSCF nodes



3.28.1 Configure User Redistribution in I-CSCF Based on S-CSCF Capabilities

This method is recommended when S-CSCF capabilities are provisioned for users in the HSS for selecting preferred S-CSCF nodes, see Section 3.26 Configure I-CSCF Resource Broker Entry on page 121.

This method is enabled and disabled in the I-CSCF using the configuration parameter, *IcscfForcedFallbackEnabled*. It is applicable for all REGISTER requests except Register Query: initial registration, re-registration, and deregistration. When enabled, the I-CSCF explicitly requests the S-CSCF capabilities from the HSS and, based on the capabilities received, the I-CSCF selects the best matched S-CSCF. Originating and Terminating non-Register requests do not trigger the User Redistribution function.

The User Redistribution function is executed by the I-CSCF as long as *IcscfForcedFallbackEnabled* is set to **TRUE**. It is recommended that the User Redistribution is enabled for a maximum registration period, as defined by *CscfRegistrationRefreshMax*, to allow all users to register again and to be moved to another S-CSCF if necessary. After the maximum registration period, all related users have been moved to the best matched S-CSCF selected from the Resource Broker List. Then, the operator needs to disable the User Redistribution function to prevent additional signaling.

To configure the user distribution function in the I-CSCF:

1. Set *IcscfForcedFallbackEnabled* to **TRUE**.

3.28.2 Configure User Redistribution in S-CSCF

The user redistribution in the S-CSCF is triggered by setting the CSCF administrative state to *Shutting Down*, see Section 3.27 Configure Shutting Down State in CSCF on page 122.

When the S-CSCF enters the *Shutting Down* state, it monitors the number of registered users in the S-CSCF. When the number of registered users reaches the configured threshold *scscfRegisteredUserThreshold*, the S-CSCF changes the administrative state to *Unlocked* or *Locked*.

When the threshold *scscfRegisteredUserThreshold* is set to 0, the S-CSCF automatically changes the administrative state of the S-CSCF to *Locked* when there are no remaining registered users in the S-CSCF.

When the threshold *scscfRegisteredUserThreshold* is set to a number greater than 0, the S-CSCF automatically changes the administrative state of the S-CSCF to *Unlocked* when the threshold is reached.

The following are two methods to redistribute the registered users in the S-CSCF:

- Redistribution to a specific redundant S-CSCF node

- Redistribution to any redundant S-CSCF node

3.28.2.1 Configure User Redistribution to Specific Redundant S-CSCF Node

This method is recommended when the registered users in the S-CSCF are to be moved to a certain S-CSCF node, which is indicated in the SIP signaling towards the I-CSCF through the SIP 305 response code including the SIP URI configured in *scscfRedundantScscfEntry*.

To configure the user redistribution function for redistribution to a specific redundant node:

1. Set *scscfRedundantScscfEntry* to `sip:scscf2.tcp.ics.se`.
2. Set *scscfRegisteredUserThreshold* to 10000.

This automatically changes the administrative state to `Unlocked` when 10000 registered users remain.

3. Set *cscfAdministrativeState* to 2.

3.28.2.2 Configure User Redistribution to Any Redundant S-CSCF Node

This method is recommended when the registered users in the S-CSCF are to be moved to any S-CSCF node, selected by the I-CSCF based on the HSS and I-CSCF configuration, see Section 3.26 Configure I-CSCF Resource Broker Entry on page 121, which is indicated in the SIP signaling towards the I-CSCF through the SIP 480 response code. The parameter *scscfRedundantScscfEntry* must be empty.

To configure the user redistribution function for redistribution to any redundant S-CSCF node:

1. Set *scscfRedundantScscfEntry* to `<empty>`.
2. Set *scscfRegisteredUserThreshold* to 0.

This automatically changes the administrative state to `Locked` when no registered users remain.

3. Set *cscfAdministrativeState* to 2.

3.29 Configure DSCP in CSCF

The Differentiated Services Code Point (DSCP) provides Quality of Service (QoS) and classifies the network traffic. Therefore, DSCP can be used to prioritize different traffic types in the IP Multimedia Subsystem (IMS) system. The recommended DSCP values per traffic type are listed in Table 91. For more information about DSCP, refer to [RFC 2474](#).



Table 91 Recommended Traffic Types and DSCP Values

Types	Differentiated Services	DSCP
Signaling: <ul style="list-style-type: none"> • IMS signaling (SIP, Diameter, DNS) • Online Charging 	CS5	40
Offline Charging	CS2	16
O&M High Priority (SNMP Traps)	CS4	32
O&M undifferentiated (SSH, CLI, NETCONF)	CS2	16
O&M low priority (SFTP)	CS1	8

To set the DSCP configuration permanently, a cluster reboot is needed. When the DSCP settings need to be changed but a reboot is unsuitable at that time, the DSCP settings can also be set temporarily at first.

To update the DSCP settings:

1. Change one DSCP setting at a time in a Linux® shell for each payload or controller, using this **iptables** command example:

```
iptables control -t mangle -A OUTPUT -p TCP --sport 162
-s 10.50.41.50 -j DSCP --set-dscp-class cs4
```

This command adds one DSCP configuration based on the IP address of the interface.

Result:

The DSCP settings are now set temporarily and do not require a cluster reboot before taking effect.

2. Open the file `/cluster/etc/cluster.conf`.
3. Add the suitable **iptables** commands to the `cluster.conf` file, using this **iptables** command example:

```
iptables control -t mangle -A OUTPUT -p TCP --sport 162
-s 10.50.41.50 -j DSCP --set-dscp-class cs4
```

This command adds one DSCP configuration based on the IP address of the interface.

4. Reload the configuration:

```
cluster config -r -a
```

5. Reboot the cluster:

```
cluster reboot -a
```

Note: The cluster reboot can be done at any time after configuring the `cluster.conf` file. The new DSCP configuration settings take effect after the cluster reboot.

Result:

The DSCP settings are now set permanently.

3.30

Configure Removal of Cached AS Instances

This section describes how to configure the AS IP addresses in `scscfClearAsCache` for removing AS-instances. After a cached AS instance is removed, a new AS instance is cached for each user among the available AS instances according to the DNS results when next time the AS is triggered.

To remove cached AS-instance, configure parameter `scscfClearAsCache` using one of following:

- A list of specific IP addresses of application servers
- keyword `all`

When the parameter is set with a list of specific IP addresses of application servers, cached AS-instances that matched the configured IP addresses are removed per user when they are to be used or at the latest at the next reregistration. When parameter `scscfClearAsCache` is set to keyword `all`, all AS-instances are removed. A total number of 10 IPv4 or IPv6 can be set in the parameter, and if more than 10 IP addresses are needed keyword `all` have to be used.

Examples of `scscfClearAsCache` configured value are listed in Table 92.

Table 92 Configuration of IP Addresses in Parameter `scscfClearAsCache`

Attribute	Example Values
<code>scscfClearAsCache</code>	192.168.10.50
<code>scscfClearAsCache</code>	fc01::250:56ff:fec1:1b
<code>scscfClearAsCache</code>	192.168.10.50, 192.168.10.53
<code>scscfClearAsCache</code>	all

To disable the remove AS-instances function, manually reset parameter `scscfClearAsCache` to empty value with following command (in ECLI):
`scscfClearAsCache= []`

If parameter `scscfClearAsCache` is not manually reset to empty, S-CSCF automatically clears all the values of the parameter `scscfClearAsCache` after the longest period of the configured value of parameters `cscfRegistrationRefreshMax` or `scscfUnregisteredProfileTimer`. After those longest periods, all the users that are registered before the



time of setting the parameter *scscfClearAsCache* must be exposed for a REGISTER or an activity as unregistered user. This means the AS-instances of all such users that matched with configured IP addresses of parameter *scscfClearAsCache* are removed or recached during those longest periods.

3.31 Configure Failover Timer per FQDN

This section describes how to configure the failover timer per FQDN in the CSCF.

To supervise the lifetime of a transaction within the network, the following SIP timers are used for different types of requests and the FQDNs of destinations:

- Attribute *cscfFailoverTimeInviteDest* is used to define the time in seconds for failover for all interfaces for INVITE requests per FQDN that is defined by attribute *cscfDestinationSipAddress*.

The value range is 0–32. The default value is 0 for specifying that no specific configuration is made and the timer is disabled. When the timer is disabled, *CscfScscfFailoverTimer*, *cscfASFailoverTimeInvite*, *extNetSelPoolTimeOut*, or *CscfSipDefaultFailoverTimer* is used depending on the interface. If no timer is configured, the value 64×T1 is used.

- Attribute *cscfFailoverTimeRegisterDest* is used to define the time in seconds for failover for all interfaces for REGISTER requests per FQDN that is defined by attribute *cscfDestinationSipAddress*.

The value range is 0–32. The default value is 0 for specifying that no specific configuration is made and the timer is disabled. When the timer is disabled, *cscfScscfFailoverTimer*, *cscfASFailoverTimeNonInvite*, or *cscfSipDefaultFailoverTimer* is used depending on the interface. If no timer is configured, the value 64×T1 is used.

- Attribute *cscfFailoverTimeNonInviteDest* is used to define the time in seconds for failover for all interfaces for non-INVITE (except REGISTER) requests per FQDN that is defined by attribute *cscfDestinationSipAddress*.

The value range is 0–32. The default value is 0 for specifying that no specific configuration is made and the timer is disabled. When the timer is disabled, *icscfNonInviteFailoverTimer*, *cscfASFailoverTimeNonInvite*, *extNetSelPoolTimeOut*, or *cscfSipDefaultFailoverTimer* is used depending on the interface. If no timer is configured, the value 64×T1 is used.

Table 93 shows an example of single configuration of failover timer per FQDN. In this case, one group of attributes with the key as defined by *cscfSipConfigDestProfileId* and one FQDN is set by *cscfDestinationSipAddress*.

Table 93 Example Single Configuration of Failover Timer per FQDN

Attribute	Value Example
<i>cscfSipConfigDestProfileId</i>	ERICSSON_AS
<i>cscfDestinationSipAddress</i>	mtas1.com
<i>cscfFailoverTimeInviteDest</i>	10
<i>cscfFailoverTimeRegisterDest</i>	9
<i>cscfFailoverTimeNonInviteDest</i>	5

Table 94 shows an example of single configuration of failover timer for multiple FQDNs. In this case, one group of attributes with the key as defined by *cscfSipConfigDestProfileId* and more than one FQDN is set by *cscfDestinationSipAddress*.

Table 94 Example Single Configuration of Failover Timer per FQDN for Multiple FQDNs

Attribute	Value Example
<i>cscfSipConfigDestProfileId</i>	AS
<i>cscfDestinationSipAddress</i>	as5071.com, as5070.com
<i>cscfFailoverTimeInviteDest</i>	8
<i>cscfFailoverTimeRegisterDest</i>	7
<i>cscfFailoverTimeNonInviteDest</i>	5

Table 95 shows an example of single configuration of failover timer per FQDN for different groups. In this case, more than one group of attributes with the key as defined by *cscfSipConfigDestProfileId* and one FQDN is set by *cscfDestinationSipAddress* per group.

Table 95 Example Single Configuration of Failover Timer per FQDN for Different Groups

Attribute	Value Example
<i>cscfSipConfigDestProfileId</i>	LRF
<i>cscfDestinationSipAddress</i>	lrf1.com
<i>cscfFailoverTimeInviteDest</i>	10
<i>cscfFailoverTimeRegisterDest</i>	8
<i>cscfFailoverTimeNonInviteDest</i>	5



Attribute	Value Example
<i>cscfSipConfigDestProfileId</i>	AS
<i>cscfDestinationSipAddress</i>	as5070.com
<i>cscfFailoverTimeInviteDest</i>	5
<i>cscfFailoverTimeRegisterDest</i>	9
<i>cscfFailoverTimeNonInvitedDest</i>	15

3.32 Configure SIP Overload Control

This mechanism is a hop-by-hop traffic regulator that makes it possible to gracefully reduce the traffic load by rejecting or redirecting traffic to an alternative destination to avoid overload.

SIP Overload Control is configured in the following two roles:

- Reporting

The CSCF node reports its current need of traffic reduction.

- Reacting

The CSCF node reduces the traffic towards a reporting role node.

When the functionality is enabled, SIP messages are tagged with the `oc` parameter. This parameter, with value `oc-value`, indicates the traffic percentage to redirect or reject because of overload. For more information, refer to [RFC 7339](#).

3.32.1 Configure Reporting Role for SIP Overload Control

To configure the Reporting Role for SIP Overload Control:

1. Configure the threshold of activating SIP Overload Control Reporting, see the procedure in Section 3.32.1.1 Configure the Threshold of Activating SIP Overload Control Reporting on page 132.
2. Configure the Fairness Behavior, see the procedure in Section 3.32.1.2 Configure Fairness Behavior on page 132.
3. Enable the Reporting Role for SIP Overload Control, see the procedure in Section 3.32.1.3 Enable Reporting Role for SIP Overload Control on page 133.

3.32.1.1 Configure the Threshold of Activating SIP Overload Control Reporting

The parameter *cscfSipOverloadOnset* configures the average cluster resource utilization level, in percentage, that must be exceeded to activate the SIP Overload Control reporting function.

The CSCF checks the average resource utilization for the cluster at every second.

When the resource utilization level exceeds *cscfSipOverloadOnset*, the CSCF maps the average resource utilization level to an *oc-value*. The CSCF requests the upstream nodes that support SIP Overload Control to reduce their traffic towards the CSCF with the percentage in the *oc-value*. For upstream nodes that are not supporting SIP overload control, *cscfSipOverloadControlReportingFairnessBehavior* applies, see Section 3.32.1.2 Configure Fairness Behavior on page 132.

The default value for the *cscfSipOverloadOnset* is 75.

To configure the threshold of activating SIP Overload Control Reporting:

1. Navigate to the Managed Object Class (MOC) *CscfSipOverloadControl*.
2. Set *cscfSipOverloadOnset* with the percentage of the traffic load that activates SIP Overload Control.

3.32.1.2 Configure Fairness Behavior

The parameter *cscfSipOverloadControlReportingFairnessBehavior* defines if the CSCF is to reject an amount of incoming traffic from a node that does not support SIP Overload Control.

If an upstream node does not support SIP Overload Control, the CSCF rejects the same percentage as it requests other upstream nodes that do support SIP Overload Control to reduce their traffic. This way, all upstream nodes, with or without support for SIP Overload Control, have the same percentage reduction of messages to the CSCF node.

The SIP methods that CSCF rejects through the SIP Overload Control fairness function are REGISTER, INVITE, and SUBSCRIBE.

cscfSipOverloadControlReportingFairnessBehavior can have the following values:

- **NoFairness**

This disables the fairness behavior.

- **Sip503Response**

Traffic is rejected through sending a SIP 503 (Service Unavailable) response without a Retry-After header.



- **Sip500Response**

Traffic is rejected through sending a SIP 500 (Server Unavailable) response without a Retry-After header.

The default value for *cscfSipOverloadControlReportingFairnessBehavior* is **Sip503Response**.

To configure the fairness behavior:

1. Navigate to the MOC *CscfSipOverloadControl*.
2. Set *cscfSipOverloadControlReportingFairnessBehavior* with the appropriate option for the network.

3.32.1.3 Enable Reporting Role for SIP Overload Control

The CSCF supports the Reporting Role for the mandatory loss-based algorithm defined in [RFC 7339](#).

To enable the Reporting Role for SIP Overload Control:

1. Navigate to the MOC *CscfSipOverloadControl*.
2. Set the parameter *cscfSipOverloadControlReportingEnabled* to **true**.

The parameters *cscfSipOverloadOnset* and *cscfSipOverloadControlReportingFairnessBehavior* are only applicable when the Reporting Role for the SIP Overload Control is enabled.

The SIP Overload Control reporting starts when the CSCF cluster resource utilization level exceeds the value configured in *cscfSipOverloadOnset*.

3.32.2 Configure Reacting Role for SIP Overload Control

To configure the Reacting Role for SIP Overload Control in CSCF:

1. Configure the Reacting Role priorities for SIP Overload Control, see the procedure in Section 3.32.2.1 Configure Reacting Role Priorities for SIP Overload Control on page 134.
2. Configure the Reacting Measurement window, see the procedure in Section 3.32.2.2 Configure Reacting Measurement Window on page 135.
3. Enable bypassing throttling towards the own node, see the procedure in Section 3.32.2.3 Enable Bypassing Throttling Towards Own Node on page 135.
4. Enable the Reacting Role for SIP Overload Control, see the procedure in Section 3.32.2.4 Enable Reacting Role for SIP Overload Control on page 136.

3.32.2.1 Configure Reacting Role Priorities for SIP Overload Control

The CSCF supports 16 configurable priorities to consider when deciding which requests to redirect or reject: 0 for the highest priority messages, 1–14 for the medium priority messages, 1–14 for the medium priority messages, and 15 for the lowest priority messages.

In addition to these priorities, there is another non-configurable priority that is never redirected or rejected.

During overload, messages with the lowest priority are rejected or redirected randomly first, while high priority messages continue as normal. When the overload is large and cannot be solved through rejecting and redirecting all low priority messages, also messages with the next higher priority are randomly rejected or redirected.

The SIP message `ACK` is never redirected or rejected. The SIP message `OPTIONS`, that is sent to monitor the blacklisted destination, is never redirected or rejected.

Attribute *cscfSipOverloadControlReactingTrafficPriorities* is used to map the priority of the SIP requests.

The following are types of SIP requests:

- Emergency

Emergency service requests

- RphWps<X>

All SIP requests for priority service, where <X> is in the range of 0–4, where 0 is the highest and 4 the lowest priority, defined in the `wps` namespace value in the Resource Priority Header. This indicates the priority level of the user or session.

- Inside

All in-dialog requests.

- Default

All other requests.

To set the priorities level:

1. Navigate to the Managed Object Class (MOC) *CscfSipOverloadControl*.
2. Set *cscfSipOverloadControlReactingTrafficPriorities* with the appropriate option for the network.



The input format for *cscfSipOverloadControlReactingTrafficPriorities* is 0:<request types for the highest priority separated with comma>;X from 1 to 14:<request types for the medium priority separated with comma>;15:<request types for the lowest priority separated with comma>

The default value is 0:RphWps0,RphWps1;1:RphWps2,RphWps3;2:RphWps4;3:Emergency;6:Inside;15:Default.

3.32.2.2 Configure Reacting Measurement Window

The parameter *cscfSipOverloadControlReactingTrafficMeasurementInterval* configures a sliding time window in milliseconds during which the amount of outgoing SIP requests per destination and priority is measured. This data is used to forecast the ongoing traffic levels to be redirected or rejected towards downstream nodes.

The forecast is based on the mandatory loss-based algorithm defined in [RFC 7339](#).

The shorter the measurement interval, the faster the traffic adjustments are. The transition, however, is less smooth. The longer the measurement interval, the slower the traffic adjustments are. The transition, however, is smoother.

To set the reacting measurement window:

1. Navigate to the MOC *CscfSipOverloadControl*.
2. Set *cscfSipOverloadControlReactingTrafficMeasurementInterval* to the appropriate value for the network.

The allowed values for *cscfSipOverloadControlReactingTrafficMeasurementInterval* are: 1–4294967295 and the default is 1000.

3.32.2.3 Enable Bypassing Throttling Towards Own Node

The parameter *cscfWhiteListOwnNodeBehavior* enables that all own network interfaces and own domains can be white-listed for Overload Control.

White-listing manages collocated CSCF instances, for example when the originating and the terminating CSCF for the same session is handled by the same managed element. Therefore, the SIP requests sent to any network interface or domain name that is configured as the own CSCF node, bypass the SIP Overload Control throttling. This means that these messages are not to be redirected or rejected.

To enable white-listing for Overload Control:

1. Navigate to the MOC *CscfSipProtocolClass*.
2. Set *cscfWhiteListOwnNodeBehavior* to **enabled**.

3.32.2.4 Enable Reacting Role for SIP Overload Control

The CSCF supports the Reacting Role for the mandatory loss-based algorithm defined in [RFC 7339](#).

To enable the Reacting Role for SIP Overload Control:

1. Navigate to the MOC *CscfSipOverloadControl*.
2. Set *cscfSipOverloadControlReactingEnabled* to **true**.

When *cscfSipOverloadControlReactingEnabled* is **true**, the attributes *cscfSipOverloadControlReactingTrafficPriorities*, *cscfSipOverloadControlReactingTrafficMeasurementInterval*, and *cscfWhiteListOwnNodeBehavior* apply.

3.33 Configure Monitoring Interfaces with SIP OPTIONS

This section describes how to configure monitoring of SIP interfaces by sending SIP OPTIONS to destination nodes and how to avoid monitoring certain nodes.

After enabling monitoring, as described in Section 3.33.1 Enable Monitoring on page 136, set the monitoring mode to any of the following:

- Monitoring enabled at blacklisting, see Section 3.33.2 Configure Blacklist Monitoring on page 137.

When destination nodes are blacklisted, SIP OPTIONS is sent to the blacklisted nodes to determine when a node is ready to bring back into service.

- Monitoring suppressed at blacklisting, see Section 3.33.3 Configure Suppress Monitoring on page 137.

When destination nodes are blacklisted, SIP OPTIONS is not sent. Some nodes ignore SIP OPTIONS, which results in the nodes staying blacklisted forever.

- Proactive monitoring, see Section 3.33.4 Configure Proactive Monitoring on page 138.

Destination nodes are periodically monitored independent of if they are blacklisted or not.

3.33.1 Enable Monitoring

To enable monitoring:

1. Navigate to MOC *CscfMonitoredInterfaceClass*.
2. Set *cscfMonitorEnabled* to **true**.



3.33.2 Configure Blacklist Monitoring

When destinations are blacklisted and `cscfMonitorEnabled` is set to `true`, the CSCF sends SIP OPTIONS to the blacklisted destinations to monitor their SIP status. This removes the destination from the blacklist when it is back in service or prolongs the blacklisting as long as the node is not in service.

Any SIP response to SIP OPTIONS requests, except SIP 503, removes the SIP destinations from the blacklist and ceases the alarm *CSCF, SIP Monitored Interface Unreachable*.

The parameter `cscfMonitorFallbackCheckTimer` is an integer value in the range 1–3600, used for all blacklisted destinations. It defines the period in seconds between each SIP OPTIONS transaction.

To configure `cscfMonitorFallbackCheckTimer`:

1. Navigate to MOC *CscfMonitoredInterfaceClass*.
2. Set `cscfMonitorFallbackCheckTimer` according to network needs.

The parameter `cscfAlarmOnSIP503Behavior` is used to configure whether to raise the alarm *CSCF, SIP Monitored Interface Unreachable* when a destination is blacklisted because it receives a SIP 503 response. All other blacklisting reasons always raise the alarm, because it is likely that there is a need for a manual action to cease the alarm. Blacklisting because of receiving a SIP 503 response ceases automatically within a short time period in most cases.

To configure `cscfAlarmOnSIP503Behavior`:

1. Navigate to MOC *CscfMonitoredInterfaceClass*.
2. Set `cscfAlarmOnSIP503Behavior` according to network needs.

3.33.3 Configure Suppress Monitoring

The parameter `cscfSipMonitoringSuppressDestinationEntry` is a multi-value attribute where each entry specifies a destination address with, optionally, a destination port `<dest_address>[:<dest_port>]`. When the port is not configured, SIP OPTIONS is suppressed towards all ports. The CSCF does not monitor the configured destinations when they become blacklisted.

To configure the suppress list:

1. Navigate to the MOC *CscfMonitoredInterfaceClass*.
2. Set `cscfSipMonitoringSuppressDestinationEntry` with destinations that SIP OPTIONS is not sent to.

For example values, see Table 96.

Note: It is not possible to configure a destination to *cscfSipMonitoringSuppressDestinationEntry*, if the destination address exists in the proactive list described in section Section 3.33.4 Configure Proactive Monitoring on page 138.

Table 96 Example Configuration of *cscfSipMonitoringSuppressDestinationEntry*

Attribute	Value Example
<i>cscfSipMonitoringSuppressDestinationEntry</i>	192.168.10.51:1025
<i>cscfSipMonitoringSuppressDestinationEntry</i>	192.168.10.50
<i>cscfSipMonitoringSuppressDestinationEntry</i>	[fc01::250:56ff:fec1:1b]
<i>cscfSipMonitoringSuppressDestinationEntry</i>	[fc01::250:56ff:fec1:1b]:49151

3.33.4 Configure Proactive Monitoring

3.33.4.1 Configure Proactive List

The parameter *cscfProactiveMonitoredSipInterfaceEntry* is a key attribute of a multi-instance MOC where each entry specifies the source transport address together with the destination address in the following format:

```
<protocol>:<src_address>:<src_port>-<dest_address>[:<dest_port>]
```

Where:

- **protocol:** UDP or TCP.
- **src_address:** an IPv4 dotted decimal address or IPv6 address of the source node. Both **src_address** and **dest_address** must use the same IP version (IPv4 or IPv6).
- **src_port:** the port of the source node, in the range of 1024–49151.
- **dest_address:** an IPv4 dotted decimal address or IPv6 address of the destination node. Both **src_address** and **dest_address** must use the same IP version (IPv4 or IPv6).
- **dest_port:** the destination port. An optional parameter in the range of 1024–49151. If the port is not specified, port 5060 to this destination is used.

The CSCF monitors the configured destinations. For periodic SIP OPTIONS, the first SIP blacklisting failure blacklists the destination, independent of



configured blacklisting thresholds. A periodic SIP OPTIONS never bypasses the blacklisting, independent of bypass throttle configurations. When a destination becomes blacklisted because of proactive monitoring, the blacklist monitoring raises the alarm and monitors the destination.

To configure the proactive list:

1. Navigate to the MOC *CscfMonitoredInterfaceClass*.
2. Navigate to *CscfProactiveMonitoredSipInterfaceGroup*.
3. Set *cscfProactiveMonitoredSipInterfaceEntry* with the transport protocol, the source address, and the destination address that need to be monitored.

For example values, see Table 97.

Note: The configuration of *cscfProactiveMonitoredSipInterfaceEntry* has the following constraints:

- Source addresses must be configured among own network interfaces of the CSCF, see Section 3.2.3.10 CSCF Network Interface on page 19.
- Destinations must not be configured in *cscfSipMonitoringSuppressDestinationEntry* in section Section 3.33.3 Configure Suppress Monitoring on page 137.
- It is not possible to remove the network interface of source addresses that exist in *cscfProactiveMonitoredSipInterfaceEntry*.

Table 97 Example Configuration of *cscfProactiveMonitoredSipInterfaceEntry*

Attribute	Value Example
<i>cscfProactiveMonitoredSipInterfaceEntry</i>	UDP:192.168.10.202:7025-192.168.10.51:1025
<i>cscfProactiveMonitoredSipInterfaceEntry</i>	TCP:192.168.10.201:5060-192.168.10.50
<i>cscfProactiveMonitoredSipInterfaceEntry</i>	UDP:[fc01::250:56ff:fec1:201]:5060-[2000::4:1]
<i>cscfProactiveMonitoredSipInterfaceEntry</i>	TCP:[fc01::250:56ff:fec1:202]:7025-[2000::4:1]:1024

3.33.4.2 Configure Proactive Monitoring Interval

The parameter *cscfProactiveMonitoringInterval* is used for all *cscfProactiveMonitoredSipInterfaceEntry* instances. It is an integer between 0-100. It defines the period in seconds between each SIP OPTIONS transaction towards the same destination.

The periodic sending of SIP `OPTIONS` to the different destinations is spread out over the complete `cscfProactiveMonitoringInterval` to smoothen out the traffic. When changing the interval or changing the number of entries, all destinations are rescheduled according to the number of entries and interval. Value 0 means that the proactive monitoring is disabled or discontinued regardless of `cscfMonitorEnabled` configuration.

To configure `cscfProactiveMonitoringInterval`:

1. Navigate to MOC `CscfMonitoredInterfaceClass`.
2. Set `cscfProactiveMonitoringInterval` according to network needs.

3.34 Configure eVIP FE-HA

The Evolved Virtual Internet Protocol (eVIP) Front-End High Availability (FE-HA) enables the operator to use a network configuration that does not use Bidirectional Forwarding Detection (BFD) or Open Shortest Path First (OSPF) from the Cloud Edge switch towards the Virtual Network Function (VNF). This limits the requirements that the VNF has on the infrastructure and makes it possible to deploy the VNF when BFD is not supported.

When using eVIP FE-HA, the eVIP Front-End Elements (FEEs) are always available. If one Virtual Machine (VM) that hosts an FEE fails, this Front-End Element is relocated automatically to another VM without an FEE. If no VM without an FEE exists, the Front-End Element is moved to a VM that already hosts an FEE. When the move is complete, it is announced through graceful Address Resolution Protocol (ARP).

The eVIP FE-HA can only be used for configurations with static routing without BFD. Therefore, the Front-End Elements included in an Abstract Load Balancer (ALB) must be reconfigured to use static routing. Also, a first hop redundancy protocol, such as Virtual Router Redundancy Protocol (VRRP), is required in the Cloud Edge switch for redundancy.

3.34.1 Modify eVIP Configuration for eVIP FE-HA

The default eVIP configuration is delivered in the CSCF installation package (included in the `evip.xml` configuration file) to which the site-specific parameters (IP addresses) are injected from HOT or OVF files. Modify the configuration using ECLI or NETCONF.

Note: In the following procedure, the eVIP configuration is modified through ECLI as an example.

To modify the eVIP configuration for eVIP FE-HA:

1. Create a backup, refer to *Create Backup*.
2. Update the default eVIP configuration for eVIP FE-HA through ECLI:



- a. Log on to the ECLI:

```
# ssh <oam-user>@<OAM-MIP> -p 2022
```

- b. Copy the following command lines to the console for deleting current eVIP FEEs:

```
configure
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=ln_sig_sc,EvipFees=1
no EvipFee=fee_sig_3
no EvipFee=fee_sig_4
no EvipFee=fee_sig_5
no EvipFee=fee_sig_6
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=ln_cha_sc,EvipFees=1
no EvipFee=fee_cha_3
no EvipFee=fee_cha_4
no EvipFee=fee_cha_5
no EvipFee=fee_cha_6
commit
```

- c. Copy the following command lines to the console for adding and configuring new HA Signaling FEEs:

Note: EvipParam=gateway, value=192.168.216.252 is the Virtual Router Redundancy Protocol (VRRP) for signaling in the Cloud Edge switches to provide redundancy.

```
configure
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=ln_sig_sc
EvipFees=1
EvipFee=fee_sig_3
externalInterface=eth1
extIfBridging="0"
node=3
EvipRoutingSetup=static
EvipParam=gateway
value=192.168.216.252
up
EvipParam=local_address
value=192.168.216.3/24
up
up
EvipRoutingSetup=static6
EvipParam=gateway
value=fd44:e781:d225::252
up
EvipParam=local_address
value=fd44:e781:d225::3/64
commit
up
up
up
configure
EvipFee=fee_sig_4
externalInterface=eth1
extIfBridging="0"
node=4
EvipRoutingSetup=static
EvipParam=gateway
value=192.168.216.252
up
EvipParam=local_address
value=192.168.216.4/24
up
up
EvipRoutingSetup=static6
EvipParam=gateway
value=fd44:e781:d225::252
up
```



```
EvipParam=local_address
value=fd44:e781:d225::4/64
commit
up
up
up
configure
EvipFee=fee_sig_5
externalInterface=eth1
extIfBridging="0"
node=5
EvipRoutingSetup=static
EvipParam=gateway
value=192.168.216.252
up
EvipParam=local_address
value=192.168.216.5/24
up
up
EvipRoutingSetup=static6
EvipParam=gateway
value=fd44:e781:d225::252
up
EvipParam=local_address
value=fd44:e781:d225::5/64
commit
up
up
up
configure
EvipFee=fee_sig_6
externalInterface=eth1
extIfBridging="0"
node=6
EvipRoutingSetup=static
EvipParam=gateway
value=192.168.216.252
up
EvipParam=local_address
value=192.168.216.6/24
up
up
EvipRoutingSetup=static6
EvipParam=gateway
value=fd44:e781:d225::252
up
EvipParam=local_address
value=fd44:e781:d225::6/64
commit
```

- d. Copy the following command lines to the console for adding and configuring new HA Charging Fees:

Note: EvipParam=gateway, value=192.168.217.252 is the VRRP for charging in the Cloud Edge switches to provide redundancy.

```
configure
ManagedElement=1,Transport=1,Evip=1,EvipAlbs=1,EvipAlb=ln_cha_sc
EvipFees=1
EvipFee=fee_cha_3
externalInterface=eth2
extIfBridging="0"
node=3
EvipRoutingSetup=static
EvipParam=gateway
value=192.168.217.252
up
EvipParam=local_address
value=192.168.217.3/24
up
up
EvipRoutingSetup=static6
```



```

EvipParam=gateway
value=fd44:e781:d226::252
up
EvipParam=local_address
value=fd44:e781:d226::3/64
commit
up
up
up
configure
EvipFee=fee_cha_4
externalInterface=eth2
extIfBridging="0"
node=4
EvipRoutingSetup=static
EvipParam=gateway
value=192.168.217.252
up
EvipParam=local_address
value=192.168.217.4/24
up
up
EvipRoutingSetup=static6
EvipParam=gateway
value=fd44:e781:d226::252
up
EvipParam=local_address
value=fd44:e781:d226::4/64
commit
up
up
up
configure
EvipFee=fee_cha_5
externalInterface=eth2
extIfBridging="0"
node=5
EvipRoutingSetup=static
EvipParam=gateway
value=192.168.217.252
up
EvipParam=local_address
value=192.168.217.5/24
up
up
EvipRoutingSetup=static6
EvipParam=gateway
value=fd44:e781:d226::252
up
EvipParam=local_address
value=fd44:e781:d226::5/64
commit
up
up
up
configure
EvipFee=fee_cha_6
externalInterface=eth2
extIfBridging="0"
node=6
EvipRoutingSetup=static
EvipParam=gateway
value=192.168.217.252
up
EvipParam=local_address
value=192.168.217.6/24
up
up
EvipRoutingSetup=static6
EvipParam=gateway
value=fd44:e781:d226::252
up
EvipParam=local_address
value=fd44:e781:d226::6/64
commit

```



3. Create a backup, refer to *Create Backup*.