

CSCF Troubleshooting Guideline

Call Session Control Function

TROUBLESHOOTING

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
1.1	Prerequisites	1
2	Tools	3
2.1	ECLI/NETCONF	3
2.2	Tracing	5
2.3	CSCF Health Check	6
3	Troubleshooting Functions	7
3.1	Fault Isolation	8
3.2	Event History	10
3.3	Alarms and Notifications	10
3.4	Logging	11
3.5	Counters	12
3.6	CSCF Trace Logs	13
3.7	NetTrace	13
3.8	CSCF User Data Output	14
3.9	Software Level Checks	14
3.10	Data Collection	15
3.11	Check Licensing	15
3.12	Check Alarms on the NeLS	16
3.13	Check SSH Key for Authentication	17
4	Troubleshooting Procedure	21
4.1	Monitor Alarms	21
5	Application Recovery Procedure	23
5.1	Diameter Error Situations	23
5.2	Scale-In Failures	24
6	Trouble Reporting	29
7	Log Example from Fault Isolation	31





1 Introduction

This document describes how to perform the troubleshooting procedure in the Call Session Control Function (CSCF).

This document does not cover the following:

- Installation and initial configuration instructions
- Periodic maintenance tasks
- Parameter configuration

1.1 Prerequisites

It is assumed that users of this document are familiar with performing operations within the area for Operation and Maintenance (O&M) in general.

1.1.1 Documents

Before starting this procedure, ensure that the document [Emergency Recovery Procedure for CSCF](#) has been read.

1.1.2 Tools

The following tools can be used to troubleshoot the CSCF. For more information about the tools, see Section 2 on page 3.

- Ericsson Command-Line Interface (ECLI) or NETCONF through which you can perform the following:
 - View alarms and notifications
 - Configuration Management
 - Software Inventory, backups, upgrades
 - Fault Management
- Tracing
 - UserTrace
 - NetTrace
- CSCF Health Check



1.1.3

Conditions

Certain troubleshooting activities can have an impact on the node performance. For example, trace or log activation can disturb traffic, and is not recommended without first consulting Ericsson.



2 Tools

2.1 ECLI/NETCONF

The CSCF supports a common configuration and provisioning interface. The ECLI/NETCONF allows monitoring and managing the Managed Element (ME). The ECLI provides the operator with a command line user interface. With this interface, the user can perform configuration management functionality and supervise the status through the underlying Ericsson Common Information Model (ECIM) through various commands. For more information, see [Ericsson Command-Line Interface User Guide](#).

NETCONF provides mechanisms to modify the configuration of network devices, and to monitor status and statistics. For more information, see [Ericsson NETCONF Interface](#).

2.1.1 View Alarms and Notifications

The alarms can be viewed in the following ways:

- List the active alarms through ECLI. See [Check Alarm Status](#).
- View the alarms through the `alarmlog` in System Controller (SC) nodes. The files are located in

```
/cluster/storage/no-backup/coremw/var/log/saflog/FaultManagementLog/alarm
```

- List the alarms through a Simple Network Management Protocol (SNMP) view when the CSCF SNMP agent is configured. See [Create SNMP View](#).

The notification for configuration or CSCF status changes can be viewed through the NETCONF interface through successful subscription.

To view the notification or status changes:

1. Connect from a host to the CSCF system through NETCONF:

```
ssh -A <user>@<OAM MIP> -p 830
```

2. Replace the `<Time Stamp>` and `<Node Id>` with site-specific values in the `hello/subscription` messages in Example 1, and paste the messages over the connected NETCONF session:



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:notification:1.0</capability>
  </capabilities>
</hello>
]]>]]>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100">
  <create-subscription
    xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <startTime><Time Stamp></startTime>
    <filter type="subtree">
      <event>
        <filterType>objectCreated</filterType>
        <filterValue>ManagedElement=<Node Id>,.*</filterValue>
      </event>
      <event>
        <filterType>objectDeleted</filterType>
        <filterValue>ManagedElement=<Node Id>,.*</filterValue>
      </event>
      <event>
        <filterType>attributeChanged</filterType>
        <filterValue>ManagedElement=<Node Id>,.*</filterValue>
      </event>
    </filter>
  </create-subscription>
</rpc>]]>]]>
```

Example 1 Hello/Subscription Messages

3. The expected response is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100">
  <ok/>
</rpc-reply>
]]>]]>
```

Be aware of the following:

- The <Node Id> can be fetched through the ECLI.
- Element <startTime><Time Stamp></startTime> is optional. If it is omitted, the subscription is for the events from the current time and onwards. Otherwise, the <Time Stamp> needs to be the current time or the passed time in this format: yyyy-mm-ddThh:mm:ssZ, for example, 2014-11-28T19:38:39Z. The current time is found in the Linux® shell: date -u +%FT%TZ.
- When a time stamp with passed time is used, all buffered notification up to that time stamp, if still in the buffer, is sent out in the response.

2.1.2 Software Inventory, Backups, and Upgrades

The software inventory management, backup and restore, and upgrade can be accessed in the CSCF through the ECLI/NETCONF. See the following documents:

- Software Inventory Management
- System Backup and Restore



— Software Management

2.1.3 Fault Management

The Fault Management (FM) allows the operator to detect faults and malfunctions on the system. Based on the information in the notifications, the operator can locate the source of the event and the relevant Operating Instructions (OPIs). Based on this information, actions can be taken to resolve or prevent failures. For more information, see [Handling Alarms](#) and [Ericsson Alarm Interface](#).

2.2 Tracing

The AppTrace is a service available in the runtime environment. The main purpose of the AppTrace is to provide practical assistance in troubleshooting applications on live CSCF systems. With AppTrace, an AppTrace end user can gain insight into the current behavior of an application. AppTrace end users are Ericsson support personnel that do operation and service provisioning tasks in a live network. Operators are not considered AppTrace end users.

The AppTrace is used to realize the UserTrace and the NetTrace. For more information on general tracing function, see [CSCF AppTrace User Guide](#).

2.2.1 NetTrace

Note: The inherent problem with observing the behavior of a system by tracing, is the consumed capacity of the tracing itself. If the cost is too high, it can interfere with the primary function of the system, at worst even causing system failure.

NetTrace is a tool that allows the user to trace SIP transactions that traverse the CSCF depending on user-defined filter criteria. These transactions are formatted and output in standardized XML[®] file format according to a 3GPP[®] specification, see [3GPP TS 32.423, 8.1.0](#).

This feature can be used for troubleshooting by customer support centers, interoperability testing, and so on, with the benefit of improved Total Cost of Ownership (TCO). NetTrace can be set to trace on a user's all requests (PublicID) or a user initiating a request (OrigPublicID) or a user receiving a request (TermPublicID). If CSCF nodes are collocated, such as for In-Service Performance (ISP), Information Service (IS), and so on, SIP signalling between those nodes is not traced. For more information about NetTrace, see [CSCF Network Tracing](#).



2.2.2 UserTrace

Note: The inherent problem with observing the behavior of a system by tracing, is the consumed capacity of the tracing itself. If the cost is too high, it can interfere with the primary function of the system, at worst even causing system failure.

UserTrace is a tool that enables logging of trace points that traverse the CSCF application for a defined Public User ID. UserTrace can be set to trace on a user's all requests (PublicID) or a user initiating a request (OrigPublicID) or a user receiving a request (TermPublicID). Tracing on a Wildcarded Public User Identity (wIMPU) is not supported. A wIMPU represents a collection of multiple users. Tracing on a wIMPU can cause adverse effects on the system. Trace Profiles are a set of predefined AppTrace domains and Process types that can be used in a live system.

For more information about UserTrace, see [CSCF User Tracing](#).

2.3 CSCF Health Check

CSCF Health Check is a procedure for checking the health of a CSCF node. Regular execution of this procedure helps to prevent service outages and to prepare the system for software upgrades and other interventions.

For more information, see [CSCF Health Check](#).



3 Troubleshooting Functions

This section provides a general overview of the functions that can be used to resolve problems that can arise when using the CSCF.

These methods are described:

- Monitor alarms
- Error logs
- Performance measurement counter logs
- CSCF Trace Logs (UserTrace and NetTrace)
- Identify Issues

The CSCF provides several mechanisms to assist troubleshooters when failures occur. Internally, the CSCF stores events in order of occurrence in the memory.

These events are written to the log in the following scenarios:

- Capsule Abortion
- Major failure
- SIP Error transaction

When a SIP Error transaction is detected and is generated by the CSCF, it is evaluated against a SIP Error list. Proxy messages are not evaluated.

Examples:

- The S-CSCF receives the 487 response for INVITE, it will proxy back to the User Equipment (UE). The SIP 487 is not evaluated.
- The S-CSCF receives a SIP 503 response, but then sends out a SIP 500 response, so the S-CSCF maps the 503 response to the 500 response. The SIP 500 and SIP 503 responses are not evaluated in this case.
- The CSCF receives an initial INVITE request where the called user is not registered. The SIP 480 is evaluated.

The Emergency Access Transfer Function (EATF), being a Back-to-Back User Agent (B2BUA), always generates a response back towards the UE. Therefore, all SIP Error transactions are evaluated.

Some specific scenarios do not cause a dump to the log if the SIP Error Code is generated in the CSCF SIP Stack and therefore are never considered for evaluation. For example, receiving an INVITE with a non-existent session results in a 481 SIP response being sent back to the UE by the CSCF SIP Stack.



3.1 Fault Isolation

Fault Isolation provides a way to detect specific SIP Error Code and SIP Hexadecimal Error Codes, and uses Event History to dump all events to log, see the example of a log in Section 7 on page 31. It compares both the SIP Error Code generated by the CSCF and SIP Hexadecimal Error Codes, if present, against the Fault Isolation configured SIP error code list.

The Fault Isolation feature depends on the Event History feature. Therefore, to see the Events based on SIP error messages, the Event History feature `cscfEventHistoryEnabled` must be set to **true**.

3.1.1 SIP Error Code List

The SIP Error Code list is configured under the parameter `cscfFaultIsolationSipErrorCode`. The value of this parameter can be a specific SIP Error Code or a range of SIP Error Codes which is in the range of 400–699.

For example: “500” or “600–606”.

If this parameter is not configured, Fault Isolation for SIP error codes is disabled.

3.1.2 SIP Hexadecimal Error Code List

The SIP Hexadecimal Error Code list is configured under the parameter `cscfFaultIsolationHexErrorCode`. The value of this parameter is a regular expression (regexp) that can be a specific SIP Hexadecimal Error Code or a range of them.

For example: “020341A033004” or “020[3-5]41A03.*”.

If this parameter is not configured, Fault Isolation for SIP Hexadecimal Error Codes is disabled.

For more information about the valid SIP Hexadecimal Error Codes, see [CSCF SIP Hexadecimal Error Codes](#).

3.1.2.1 SIP Hexadecimal Error Code List Rules and Recommendations

The `cscfFaultIsolationHexErrorCode` parameter is a list of entries.

Each entry must comply with the following rules:

- POSIX® 1003.2 regular expression syntax
- At most 50 ASCII character long strings
- Full string matching is used



The following are recommendations:

- Apply the minimal number of entries that is sufficient
- The first entries should be the more general ones: that matches to more SIP Hexadecimal Error Codes
- The last entries to be the more specific ones: that matches to one or some SIP Hexadecimal Error Codes only
- Subset entries are merged if possible.

For example, the SIP Hexadecimal Error Code “020341A030000” and range “020341A03.*” are both subsets that can be merged into “02034.*”.

Note: The parameter is only to be used with filters provided by Ericsson support.

When the number of the entries are not enough, more general expression must be considered. As a result, several unique errors match and offline processing and filtering of the entries is advised.

3.1.2.2 SIP Hexadecimal Error Code List Entry Format

Apply asterisk (“.”) only if it is on the right end of the whole expression. See examples 2 and 3 in Table 1.

Table 1 SIP Hexadecimal Error Code List Entry Format Examples

No	Example Description	Regexp Construct Used	Expression ⁽¹⁾
1	Match to the error represented by error code 020341A030000. This means match on: <ul style="list-style-type: none"> • Version is (0x0) v0 • Use case is (0x2) Originating • Node type is (0x03) S-CSCF • Internal Node type is (0x4) I-CSCF • Reason is (0x1A03) failed to read database • Optional part is (3004) Diameter too busy. 		020341A033004
2	Match to the error represented in No 1, but ignore the content of the Optional part (if any).	“*”: asterisk	020341A03.*
3	Match to the error represented in No 1, but ignore the Reason and Optional part (if any).	“*”: asterisk	02034.*



Table 1 SIP Hexadecimal Error Code List Entry Format Examples

No	Example Description	Regex Construct Used	Expression ⁽¹⁾
4	Match on: <ul style="list-style-type: none"> Version is (0x0) v0 Use case is (0x2) Originating Node type is either (0x03) S-CSCF or (0x05) E-CSCF Internal Node type is (0x4) I-CSCF Reason is (0x1A03) failed to read database Optional part is (3004) Diameter too busy. Explicitly specified information follows the range (41A03).	“[” and “]”: bracket	020[3-5]41A03.*
5	Match on: <ul style="list-style-type: none"> Version is (0x0) v0 Use case is (0x2) Originating Node type is (0x03) S-CSCF Internal Node type is (0x4) I-CSCF Reason is either (0x1A03) failed to read database or (0x330) System is overloaded Optional part is ignored. 	“ ”: alternation “[” and “]”: bracket	02034(1A03) (3300).*
6	Match on: <ul style="list-style-type: none"> Version is (0x0) v0 Use case is (0x2) Originating Node type is (0x03) S-CSCF Internal Node type is (0x4) I-CSCF Reason is all the codes that are between 0x100 and 0x150 inclusively Optional part is ignored. 	“ ”: alternation “[” and “]”: bracket	0203401([0-4][0-9A-F]) (50).*
7	Match all the errors with Node type of (06) BCF.	“{” and “}”: fixed qualifier	[0-9A-F]{2}06.*

(1) All the examples here are for the V0 version of the SIP Hexadecimal Error Codes

3.2 Event History

Event History must be enabled to dump all its events to the log if there is a major failure or Capsule Abortion. Set the `cscfEventHistoryEnabled` parameter to **true** to enable the Event History feature.

3.3 Alarms and Notifications

The active alarms and notifications generated by the CSCF can be shown in various ways. For more information, see Section 2.1.1 View Alarms and Notifications on page 3.



3.4 Logging

Provide the event logs to the next level of maintenance support if a problem cannot be solved.

The event log files can contain information from a crash. If a crash has occurred, the files with information about the crash are collected in a tar file.

3.4.1 Where to Find Log Files and Counter Files

Detailed information about log files and counter files locations is listed in Table 2.

Table 2 Log Files and Counter Files Locations

Type of Log File	Path to Log File	Log Filename
Linux Syslog	var/log/<system_controller> Where <system_controller> is SC-1 or SC-2	messages
vDicos VM console logs	/cluster/storage/no-backup/cdclsv/log/lpmsv/	vm<n>- [SC PL] -<m>, where n is the CPU core number, and m is the SC or Payload (PL) node number
Application Logs	/cluster/storage/no-backup/coremw/var/log/saflog	<Application>_<StartDate>_<StartTime>[_<EndDate>_<EndTime>].log where <Application> can be "CSCF", "Diameter", "AppTrace", CscfEventHistory, and so on
Error Dump	/cluster/storage/no-backup/cdclsv/dumps	vDicos-<date>.<time>.<msec>.tgz
CSM log	/cluster/storage/no-backup/coremw/var/log/<system_controller>/clustermonitor/ Where <system_controller> is SC-1 or SC-2	cmw_csm.log
Counter	/cluster/storage/no-backup/com-apr9010443/PerformanceManagementReportFiles/	A<start_date>.<start_time>-<end_time>-jambala.xml

3.4.2 Linux Syslog

The Linux Syslog is a log file containing information about the main events in the system. Examples of these events are configuration changes through ECLI/NETCONF, processor disturbances, Capsule Abortions, and scale operations. The log file is located in controller blades at /var/log/<node_name>/messages.



3.4.3 vDicos VM Console Logs

The vDicos Virtual Machine (VM) console logs are used by the CSCF through the vDicos application for any debug information. The logs are rotated, collected, and stored persistently in the `/cluster/storage/no-backup/cdclsv/log/lpmsv` directory.

Each vDicos VM console log is named as `vm<n>-[SC|PL]-<m>`, where `n` is the CPU core number, and `m` is the SC or PL node number. The file is rotated with name of `vm<n>-[SC|PL]-<m>-<timestamp>` when the configurable file size is reached.

3.4.4 Application Logs

The Application Logs contain vDicos and vDicos application log entries like events, alarms, notifications, and event-history. It uses the `AppLog` function of vDicos and the logs are created when vDicos and its application start. The logs are stored continuously and provide users a summary of the application-specific information over time.

3.4.5 Error Dumps

This mechanism creates an error dump file after a process crash.

The following error dump can be created on a running system:

- Linux core dump: if a platform process crashes
- vDicosVM CA: a Capsule Abortion inside the vDicos VM

The Linux core dumps are generated by the Linux system and the error dump is located in `/cluster/dumps`. The vDicosVM CAs are generated and dumps are collected in `/cluster/storage/no-backup/cdclsv/dumps/`. The vDicos dump filenames follow this template: `vDicos<-timestamp>.tgz`.

3.4.6 CSM Log

The CBA System Model (CSM) log, `cmw_csm.log`, includes logs from the different CSM plug-ins. The CSM plug-ins are executed for scale-out and scale-in operations. For example, when scale-in fails because of veto rejection, the actual reason for rejection can be found in the `cmw_csm.log`.

3.5 Counters

Using the performance counters generated by the CSCF is another way to get useful information when troubleshooting a problem. The performance counter files are generated in 3GPP compliant Performance Management (PM) XML format, and can be transferred outside the system for post processing. For more



information about performance management and performance counters used in the CSCF, see [Performance Management and Managed Object Model \(MOM\)](#).

3.5.1 **cscfSipErrorResponse**

The `cscfSipErrorResponse` counter is used to measure the number of SIP error responses. The counter is increased when a SIP error response including a hexadecimal error code is generated. It is keyed on the SIP method, the SIP response code, and the hexadecimal error code.

3.5.2 **cscfInviteCommunicationFailure**

The `cscfInviteCommunicationFailure` counter is used to measure the number of failed SIP INVITE requests rejected by the CSCF caused by communication failures. It is keyed on the concatenation of the SIP response code, the SIP reason phrase, and the hexadecimal error code. The counter also has a SUM key to collect the total number of `cscfInviteCommunicationFailure`.

3.6 CSCF Trace Logs

3.6.1 **UserTrace**

The principle of `UserTrace` is to log trace points traversing the CSCF application for fault finding and localization purposes. Traces are for a particular `PublicID`, where a particular `PublicID` is the originator or receiver. Tracing on all CSCF trace domains for a `PublicID` is not recommended because of the impact on the processor load. Therefore, tracing on a subset of available CSCF trace domains is recommended. Specific trace domains have been preconfigured into CSCF trace profiles for a live system. They have been verified and run with acceptable performance level degradation.

To start the tracing, execute the `CscfTrace` script as follows:

```
CscfTrace {-user <public id>} <user trace profile>
```

The `-user` option is used when the operator is interested in all requests/responses for that `PublicID`. For detailed information on `UserTrace`, including the list of Trace Profiles and various command line options, see [CSCF User Tracing](#).

3.7 NetTrace

The principle of `NetTrace` is to allow a user the possibility to observe and log SIP transactions traversing the CSCF for fault finding/localization purposes. For example, `NetTrace` allows an operator to trace the requests/responses sent and received by a particular `PublicId` or several `PublicIds`.



Using CSCF AppTrace, a Trace Session can be configured to trace SIP Sessions based on the filtering of the Originating Public User IDs or Terminating Public User IDs, or both.

To start the tracing, execute the `CscfTrace` script as follows:

```
CscfTrace {-user <public id>} <Nettrace profile>
```

The `-user` option is used when the operator is interested in all requests/responses for that `PublicID`.

These two levels of net trace profiles exist, which are mutually exclusive:

- Min level, where only a subset of SIP headers is traced. In this case, it is possible to select the SIP methods of interest.
- Max level, where the complete SIP message is traced in a hexadecimal format.

For detailed information on NetTrace, including the list of Trace Profiles and various command line options, see [CSCF Network Tracing](#).

3.8 CSCF User Data Output

The user data output function enables a troubleshooter to print user data based on an individual user public identity or private identity for troubleshooting purposes. For detailed information, see [CSCF User Data Output Guideline](#).

3.9 Software Level Checks

The software level of the CSCF is important to know for the next level of support.

The platform software version can be viewed through ECLI/NETCONF. For more information, see [View Software Information](#).

The detailed platform software can also be obtained through a Linux shell command from SCs:

```
cmw-repository-list | sort | grep -v "NotUsed"
```

The details of the CSCF software version can be fetched through a Linux shell command from SCs, for example:

```
cdsv-print-node -v
```

The following command prints state of the nodes, and pickup PL-n, which is part of CscfPool:

```
clurun.sh -c list_loadmodules -n PL-n -d lpmsv.agent.vm2
```



3.10 Data Collection

The troubleshooting data must be collected and enclosed if a Customer Service Request (CSR) must be raised for a problem experienced within the CSCF. Additional information can be requested by Ericsson support personnel. For more information, see [Data Collection Guideline for CSCF](#).

3.11 Check Licensing

3.11.1 Check License Information on the NeLS

To check license information:

1. Log on to the Network License Server (NeLS) and start an ECLI session:

```
ssh <user>@<target_host> -p 2022
```

2. Navigate to the LicenseKeyFileManager Managed Object (MO):

```
>dn ManagedElement=1,NelsFunction=1,LicenseRepository\
=1,LicenseKeyFileManager=1
```

3. Show all installed licenses:

```
(LicenseKeyFileManager=1)>show -v all
```

4. Log off from the ECLI session:

```
exit
```

5. Log off from the NeLS.

3.11.2 Check License Information on the CSCF

See [View License Information](#).

3.11.3 Refresh All Licenses on the CSCF

To refresh all licenses:

1. Log on to the CSCF and start an ECLI session:

```
ssh -A <user>@<target_host> -p 2022
```

2. Navigate to the CscfLicenseModelGroup MO:

```
>dn ManagedElement=<node_id>,CscfFunction.\
applicationName=1,CSCF-Application.applicationName=\
CSCF,CscfLicenseModelGroup=0
```



3. Refresh all installed licenses:

```
(CscfLicenseModelGroup=0)>scscfLicenseRefresh=true
```

4. Log off from the ECLI session:

```
exit
```

5. Log off from the CSCF.

```
exit
```

3.11.4 View CSCF Service States of a License

To view the CSCF service states for a license:

1. Log on to the CSCF and start an ECLI session:

```
ssh -A <user>@<target_host> -p 2022
```

2. Navigate to the MO of the licensed feature:

```
>dn ManagedElement=<Node_ID>,CscfFunction.\
applicationName=1,CSCF-Application.applicationName=\
CSCF,CscfLicenseModelGroup=0,<MO_of_Licensed_Feature>
```

The following example shows navigating to the Multi-Device feature MO CscfPersonalProfileEntry:

```
>dn ManagedElement=jambala,CscfFunction.\
applicationName=1,CSCF-Application.applicationName=\
CSCF,CscfLicenseModelGroup=0,\
CscfActiveUserLicenseModel=default,\
CscfPersonalProfileEntry=personal
```

3. Refresh all service states:

```
(<MO_of_Licensed_Feature>)>show -v all
```

4. Log off from the ECLI session:

```
exit
```

5. Log off from the CSCF.

```
exit
```

3.12 Check Alarms on the NeLS

To check alarms on the NeLS:

1. Log on to the NeLS and start an ECLI session:



- ```
ssh <user>@<target_host> -p 2022
```
2. Navigate to the Fm MO:
 

```
>dn ManagedElement=1,SystemFunctions=1,Fm=1
```
  3. Show the detailed alarm status:
 

```
(Fm=1)>show-table -m FmAlarm -p fmAlarmId,source,\
activeSeverity,specificProblem,additionalText
```
  4. Log off from the ECLI session:
 

```
exit
```
  5. Log off from the NeLS.

### 3.13 Check SSH Key for Authentication

If one of the Lifecycle Manager (LCM) Workflows fails and the **Workflow Log** tab reports **Authentication failed**, there is an error regarding the SSH key. Use the following instructions to repair the SSH key between the LCM and the CSCF.

#### Prerequisites

- No documents are required.
- No tools are required.
- The following conditions must apply:
  - The user has root access in the LCM.
  - The user is able to log on as the emergency user in the CSCF.

#### 3.13.1 Verify the SSH Key in the LCM

To verify the SSH key in the Lifecycle Manager (LCM):

1. Log on to the VNF Lifecycle Manager (VNF-LCM):
 

```
ssh cloud-user@<VNF-LCM IP>
```
2. Change to the user jboss\_user:
 

```
sudo su -
```

```
su - jboss_user
```
3. Log on as the emergency user to the CSCF:
 

```
ssh <emergency-user>@<OAM MIP>
```



4. Is the system prompting for a password?

Yes: Proceed with Section 3.13.2 Check the SSH Key in the CSCF on page 18.

No: Exit these procedures; no further steps are needed.

### 3.13.2 Check the SSH Key in the CSCF

To check the SSH key in the CSCF:

1. Go to the configuration directory of the CSCF in the Lifecycle Manager (LCM):

```
ssh cloud-user@<VNF-LCM IP>

sudo su -

cd /vnflcm-ext/backups/workflows/vnfd/<VNF>/\
configurations/<VNF configuration>/
```

2. Check the public key in the LCM:

```
cat id_rsa.pub
```

3. Log on as the emergency user to the CSCF:

```
ssh <emergency-user>@<OAM MIP>
```

4. Navigate to the home directory of the emergency user in the CSCF:

```
cd ${HOME}/.ssh
```

5. Check the public key in the CSCF:

```
cat .ssh/authorized_keys
```

6. Does the public key in the LCM occur in the list of public keys in the CSCF?

Yes: Exit these procedures; no further steps are needed.

No: Proceed to Section 3.13.3 Correct the SSH Key in the CSCF on page 18.

### 3.13.3 Correct the SSH Key in the CSCF

To correct the SSH key in the CSCF:

1. Verify if there are any SSH key pairs in the Lifecycle Manager (LCM):

```
ssh cloud-user@<VNF-LCM IP>

sudo su -

su - jboss_user

cd ${HOME}/.ssh
```



```
ls -l
```

2. Are there any SSH key pairs returned to the screen?

Yes: Proceed with Step 4.

No: Continue with the next step.

3. Create an SSH key pair:

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

**Note:** Press **enter** when the system prompts a question.

Do not enter a password.

4. Copy the public key to the CSCF instance directory:

```
cp id_rsa.pub /vnflcm-ext/backups/workflows/vnfd/\
<VNFTYPE__VNFVersion>/configurations/<instance>/
```

Repeat this step for all CSCF instances that are managed by this LCM.

5. Copy the public key into the emergency user:

```
scp id_rsa.pub <emergency user>@<VNF IP>:
```

Repeat this step for all CSCF instances that are managed by this LCM.

6. Add the public key to the `authorized_keys` file:

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

Repeat this step for all CSCF instances that are managed by this LCM.

7. Log off from the CSCF:

```
exit
```

8. Verify the new SSH key by following the instructions in Section 3.13.1 Verify the SSH Key in the LCM on page 17.
9. If there are still authentication problems, contact the next level of support.







## 4 Troubleshooting Procedure

Troubleshooting a problem in the CSCF can require the use of one or more troubleshooting functions.

To assure an efficient location of the fault:

1. Execute the `CscfHealthCheck` script to have an overall view of the CSCF node status. The execution results with FAIL/VERIFY that must be checked. For more information about CSCF Health Check, see Section 2.3 CSCF Health Check on page 6.
2. Check the alarms and the notifications. For information about how to view the alarms and the notifications, see Section 2.1.1 View Alarms and Notifications on page 3. For information about the CSCF alarms, see Section 4.2.
3. Check the logs. For more information about logging, see Section 3.4 Logging on page 11.
4. Check available information related to Capsule Abortion.
5. Verify network signalling logs and if necessary, activate NetTrace. For more information on SIP error messages, possible causes and recovery procedures, see [CSCF Fault Codes Catalogue](#), [CSCF SIP Hexadecimal Error Codes](#), and counters for sip error messages, see Section 3.5 Counters on page 12.
6. Verify that the licensing is working, see Section 3.11 Check Licensing on page 15.  
  
For more information on the CSCF licenses, see [CSCF License Management](#).
7. Check already reported troubles in the specific area.
8. If writing a Trouble Report, check the software version and level. For more information about Trouble Reporting, see Section 6 on page 29. For more information about Software Level Check, see Section 3.9 Software Level Checks on page 14.

### 4.1 Monitor Alarms

Based on information displayed in alarms and notifications, the source of the event and the relevant documentation can be located. Using this information, steps can be taken to resolve or prevent failures. For more information about fault management function, see [Handling Alarms](#) and [Ericsson Alarm Interface](#).



#### 4.1.1

#### CSCF Alarms

CSCF active alarms can be listed through execution of the `CscfHealthCheck` script or through ECLI/NETCONF or SNMP receiver. For more detailed information regarding CSCF alarms, see [CSCF Alarm List](#).



## 5 Application Recovery Procedure

For information on the possible SIP error situations for the CSCF and the proposed actions on how to solve the problems, see [CSCF Fault Codes Catalogue](#) and [CSCF SIP Hexadecimal Error Codes](#).

### 5.1 Diameter Error Situations

The following Diameter error situations can occur:

- The Diameter interface is disabled
- Configuration fault
- Link inactivity
- IP network failure
- System error
- Format error of Capabilities-Exchange-Request (CER)/Capabilities-Exchange-Answer (CEA) messages

#### 5.1.1 Diameter Interface Is Disabled

The possible causes are as follows:

- The remote server is down
- Network issues (routing, and so on)
- The `stackID` under `stackContainerID` in Diameter configuration is disabled
- The `nodeID` under `peerNodeContainerId` under `stackContainerId` in Diameter configuration is disabled
- The `nodeID` under `peerNodeContainerId` under `stackContainerId` in Diameter configuration has wrong IP address or port number
- Format error of CER/CEA messages

If any diameter interface is down:

1. Wait for automatic reconnection.
2. If connection is not established, disable the neighbor node and enable it again.
3. If connection still is not established, consult next level of maintenance support.



### 5.1.2 Configuration Fault

For information on how to solve the configuration fault situation, see vDicos, Diameter Link Failure.

### 5.1.3 Link Inactivity

For information on how to solve the link inactivity situation, see vDicos, Diameter Link Failure.

### 5.1.4 IP Network Failure

For information on how to solve the IP network failure situation, see vDicos, Diameter Link Failure.

### 5.1.5 System Error

For information on how to solve the system error situation, see vDicos, Diameter Link Failure.

### 5.1.6 Format Error of CER/CEA Messages

For information on how to solve format error of CER/CEA message situation, see vDicos, Diameter Link Failure.

## 5.2 Scale-In Failures

### 5.2.1 Scale-In Failures Because of Veto Rejections

Follow this procedure if a scale-in operation fails with the following error:

```
ERROR: Transaction not committed due to validation errors
Transaction validation failed!
```

1. Check `cmw_csm.log` to see if scale-in operation is rejected because of a DBS veto or a charging backup veto.

If the reason for failure is not clear from `cmw_csm.log`, then check `/var/log/<sc>/messages` (depending on which SC is active. For example, `/var/log/SC-1/messages`) to see the reason for failure.

- If the file includes an exception with the error reason `Not enough free heap` or similar, then a DBS veto has occurred. Continue to Step 2.



- If the file includes an exception with the error reason Scale-in veto because of Diameter Backup Handler has files under PL-5 or similar, then a charging backup veto has occurred. Continue to Step 3.
- 2. If a multiple scale-in was done, try to do a single scale-in instead.
- 3. Before doing another scale-in attempt, ensure that no files with Accounting-Requests (ACRs) are stored in `/cluster/CSCF/<processorName>` for the VM or node to be scaled in.

If any files are stored in `/cluster/CSCF/<processorName>`, do not remove them manually, but ensure that the Charging Server is operational and the link to Charging Server comes up. Refer also to [CSCF Charging Request Transmission Problem](#) alarm OPI.

If `/cluster/CSCF/<processorName>` directories are empty and charging backup veto still rejects, then also check if DIABH environment variables are set, for example:

```
SC-1:~# vdicos-envdata-list | grep DIABH_PL
```

```
DIABH_PL-8_HAS_FILE
DIABH_PL-7_HAS_FILE
DIABH_PL-6_HAS_FILE
DIABH_PL-5_HAS_FILE
DIABH_PL-4_HAS_FILE
```

Contact the next level of support if environments variables are set even if there are no files.

## 5.2.2 Scale-In Rejected by Cluster Because of Insufficient Memory Capabilities of Target Sized Cluster

The cluster can reject the scale-in operation, see example output:

```
>ManagedElement=1,SystemFunctions=1,SysM=1,CrM=1
```

```
(CrM=1)>show
```

```
CrM=1
autoRoleAssignment=ENABLED
ComputeResourceRole=PL-3
ComputeResourceRole=PL-4
ComputeResourceRole=PL-5
ComputeResourceRole=PL-6
ComputeResourceRole=PL-7
ComputeResourceRole=PL-8
ComputeResourceRole=SC-1
ComputeResourceRole=SC-2
Role=Default-Role
Role=SYSTEM
```

```
(CrM=1)>ComputeResourceRole=PL-5
(ComputeResourceRole=PL-5)>configure
(config-ComputeResourceRole=PL-5)>no provides
(config-ComputeResourceRole=PL-5)>up
(config-CrM=1)commit
```

```
ERROR: Transaction not committed due to validation errors
Transaction validation failed!
```

```
(config-CrM=1)
```

One possible reason for the rejection is that the cluster estimated the needed memory for the ongoing traffic and this requirement cannot be met if the scale-in operation is performed. Therefore to avoid traffic loss the cluster rejects the operation.

The procedure checking the memory consumption of the cluster is the same as described in *DBS, Memory Limit Reached*.

Recommended solutions:

- If multiple nodes were tried to be scaled in parallel, it is recommended to scale in the nodes one by one until the memory capabilities are still enough to handle the traffic.
- If even scaling in of one node is rejected, the contraction of the system is not allowed to secure the needed capabilities.

### 5.2.3 Unexpected Cluster Reboot during Scale-In

If there is a cluster reboot during the scale-in operation, the cluster can result in an inconsistent state where the node is removed but some information remains about it. For example the Compute resource can still be seen:

```
(ComputeResourceRole=PL-5)>show
```

```
ComputeResourceRole=PL-5
adminState=LOCKED
instantiationState=UNINSTANTIATION_FAILED
operationalState=DISABLED
uses="ComputeResource=PL-5"
```



**Note:** This inconsistent state has no effect on the functional capabilities of the system. The scale-in operation was successful and the future auto scale-out operations are also not prohibited despite the inconsistent state. Though this PL-5 entry cannot be scaled in again. That has effect on the capability of the system as this `ComputeResourceRole`, PL-5, cannot be assigned any more, which means that the maximum cluster size cannot be reached any more.

Recommended solution for clearing the inconsistent information:

1. Roll back the system to the previous correct state by restoring the backup created before the scale-in operation. For more information, see [Restore Backup](#).
2. Perform the scale-in operation again.







## 6 Trouble Reporting

Problems identified that cannot be solved by using this document must be reported to the next level of maintenance support.

This is to result in a Customer Service Request (CSR).

The details of the trouble reporting process is outside the scope of this document.

When collecting information for further support, ensure that all current logs are recorded, see time/date for the logs.

For more information on how to collect information, see [Data Collection Guideline for CSCF](#).

When sending crash dumps, ensure that the dump is of the actual scenario, see time/date for the dump.





## 7 Log Example from Fault Isolation

This is an example of a log from the Fault Isolation. This log is for informational purposes only as the format or and information within the log may change.

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
CscfEventHistory:
|EventHistory caused by [fault isolation framework] for process:[CscfAppProc] pid:[7480] at:
20150617T183437091 TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
```

CscfEventHistory:

|EventHistory - Major Events for process:CscfAppProc pid:7480

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
```

CscfEventHistory:

Entry 001 ===== process:CscfAppProc pid:7480 at 20150617T183437089

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
CscfEventHistory: INVITE url:sos SIP/2.0
To: sip:u0000000000@cscf99.lab
From: u0000000000
<sip:u0000000000@cscf99.lab>;tag=6127050.9645_u0000000000
Call-ID: 6035681.6072_u0000000000
CSeq: 88584 INVITE
Max-Forwards: 70
Content-Length: 0
Via: SIP/2.0/UDP 192.168.73.1:5060;branch=z9hG4bK_prepMsg_9003944.0046_u0000000000
Contact: u0000000000 <sip:u0000000000@192.168.73.1:5060>
Allow: INVITE, ACK, BYE, CANCEL, PRACK, NOTIFY, REFER
User-Agent: userAgent1
```

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
```

CscfEventHistory:

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
```

CscfEventHistory:

Entry 002 ===== process:CscfAppProc pid:7480 at 20150617T183437091

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
```

CscfEventHistory: CscfStateHandler: [CscfInviteOrigIdleState] --> [CscfFailedState]

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
```

CscfEventHistory:

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
```

CscfEventHistory:

Entry 003 ===== process:CscfAppProc pid:7480 at 20150617T183437091

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
CscfEventHistory: SIP/2.0 416 Unsupported URI Scheme
To: sip:u0000000000@cscf99.lab;tag=528b129901d380bb0f3615b384d6
From: u0000000000
<sip:u0000000000@cscf99.lab>;tag=6127050.9645_u0000000000
Call-ID: 6035681.6072_u0000000000
CSeq: 88584 INVITE
Content-Length: 0
Via: SIP/2.0/UDP 192.168.73.1:5060;branch=z9hG4bK_prepMsg_9003944.0046_u0000000000
P-Charging-Vector: icid-value=528b129901d380bb0f3615b2fe49
```

```
TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015
```

CscfEventHistory:



TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015

CscfEventHistory:

|End of Major Events for process:CscfAppProc pid:7480

TelORB: Wed Jun 17 20:34:37 2015, Host: Wed Jun 17 20:34:45 2015

CscfEventHistory:

|EventHistory SIP framework debug info is printed in console log. process:CscfAppProc pid:7480