

CSCF Health Check

Call Session Control Function

OPERATING INSTRUCTIONS

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	Health Check Tasks	3
3	Automated Health Check Procedures	5
3.1	Preparations	5
3.2	Automatic Health Check	5
3.3	Automatic Health Check Verdict	6
3.4	Automatic Health Check Options	7
4	Configuration File	9
4.1	Configuration File Format	9
4.2	Configuration File Location	9
4.3	Configuration Parameters	9
5	Manual Health Check Procedure	13
5.1	Preparations	13
5.2	Check Release Information	13
5.3	Verify Status of Alarms	14
5.4	Verify Controller Status	15
5.5	Verify System Status	17
5.6	Monitor Network Connectivity	18
5.7	Verify Processor Status	20
5.8	Verify Administrative and Operational State	21
5.9	Verify Diameter Stack Status	22
5.10	Verify CPU Load and Memory Use	26
5.11	Verify eVIP Status	30
5.12	Check System Environment Variables	31
5.13	Check Availability of DNS Servers	32
5.14	Check Status of SIP Interfaces	34
5.15	Store Health Check Report	35
6	Report Problems	37
7	Example of Configuration File	39



8	Example of Automatic Health Check Results	41
9	File Management	49



1 Introduction

This document describes how to perform the health check on the Call Session Control Function (CSCF). The health check tasks described in Section 2 on page 3 are recommended to be performed before and after a system update/upgrade, a normal backup, or during the periodic maintenance. For information about health check-related Key Performance Parameters (KPIs), see [Check CSCF Key Performance Indicators](#).

1.1 Prerequisites

This section describes the prerequisites for performing the health check procedure.

1.1.1 Documents

Before starting this procedure, ensure that the following information or documents are available:

- The CSCF Release Note corresponding to the current CSCF version.
- Node Configuration.
- IP Traffic Network Diagram.

1.1.2 Tools

The following tools are required:

- Workstation with Secure Shell (SSH) capabilities.
- A web browser on the workstation.

1.1.3 Conditions

The user performing the health check procedure must fulfill the following conditions:

- Know the CSCF system on a System administrator level.
- Know the external Virtual IP (VIP) Addresses and the System Controller (SC) address of the cluster node.
- Know the password of an Operation & Maintenance (O&M) user with sufficient rights to display CSCF Configuration Management (CM) and Fault Management (FM) parameters.



- Know the password for the troubleshooting user.
- Has authorized access to the CSCF with `sudo/root` privileges for the required troubleshooting and Operations, Administration, and Maintenance (OAM) actions.



2 Health Check Tasks

The most common health checks are covered by the automated procedure as described in Section 3 on page 5. In troubleshooting situations or when more control is desired, the checks can be performed using the manual procedures in Section 5 on page 13. The recommended periodicity for some of the most useful tasks is shown in Table 1.

Table 1 Critical Areas to Be Monitored Regularly or in Certain Situations

Section	Description
Alarms	Alarms often need to be monitored (once per hour).
Network connectivity	Can be often done (once per hour). Can be run on need basis or once per day.





3 Automated Health Check Procedures

This section describes how to perform the health check of the CSCF by running an automated script.

3.1 Preparations

This section describes the preparations required to execute the health check.

3.1.1 Log on to System Controller

Log on to system controller using SSH:

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

3.1.2 Obtain Persistent Storage Area Paths

To obtain the different Persistent Storage area paths for the system:

1. Enter these commands on the node:

```
<configuration-path> = cmwea config-location-get
```

```
<storage-path> = cmwea no-backup-location-get
```

3.2 Automatic Health Check

Automatic health check includes the following checks:

- Release Information
- Current alarms
- Controller status
- CPU load
- Diameter ports listening
- Administrative and operational state
- Processor outage
- Network connectivity
- Memory use
- Evolved Virtual Internet Protocol (eVIP) status



- System status
- System environment variables
- Performance indicators

The results are printed to the console and a report is created, which requires manual verification.

3.2.1 Run Automatic Health Check

To run the automatic health check:

1. Run the health check script:

CscfHealthCheck

Note: The first time the script is run, or when there are mandatory parameters in the configuration file that are missing a value, the user is prompted to enter values for these. For more information about the configuration file and configuration parameters, see Section 4 on page 9.

2. Check the results printed to the console. An example of the results is shown in Section 8 on page 41.
3. Find the generated report files with the most recent date and time in the directory:

```
<storage-path>/vcscf_cxp9034345/healthcheck/reports/<vnftype>_H
C_<vnfname>_<timestamp>_<type>.html
```

```
<storage-path>/vcscf_cxp9034345/healthcheck/reports/<vnftype>_H
C_<vnfname>_<timestamp>_<type>.xml
```

```
<storage-path>/vcscf_cxp9034345/healthcheck/reports/PM_INDICAT
ORS_Report_<nodename>_<time stamp>.html
```

Health Check report files can be fetched using File Management. For more information, see Section 9 on page 49.

Note: The time stamp of the generated healthcheck report files has format <YYYY-MM-DDTHH:MM:SSshh:mm>.

The time stamp of the generated PM_INDICATORS report file has format <YYYY-MM-DD_HH_MM_SS>.

3.3 Automatic Health Check Verdict

The verdict is a way to inform the user of the status of the individual checks. The definitions of the different verdicts are shown in Table 2. The definitions of the final Health Check verdicts are shown in Table 3.



Table 2 Verdict Definitions for Individual Checks

Verdict	Description
INFO	Information for the user, not checked by the script.
OK	Task passed.
VERIFY	Manual verification needed.
FAIL	Problem detected by the script.
ERROR	An error occurred, script update needed or system broken.

Table 3 Final Verdict Definitions

Verdict	Description
HEALTHY	All checks are successful. All individual checks have an INFO or OK verdict.
NOT_HEALTHY	At least one individual check has a VERIFY, FAIL, or ERROR verdict in the health check procedure.

3.4 Automatic Health Check Options

The behavior of the health check script can be further customized by providing command line options. Supported options are listed in Table 4.

Table 4 Command Line Options Supported by Health Check Script

Option	Meaning
-h, --help	Prints a brief help message and exits.
-r REPORT, --report REPORT	Specifies the location to save report. The default is <storage-path>/vcscf_cxp9034345/healthcheck/reports
-q, quiet	Prints only the verdict for each check in console.



Table 4 Command Line Options Supported by Health Check Script

Option	Meaning
-check CHECK [CHECK]	<p>Specifies the checks to run.</p> <p>The are possible values:</p> <ul style="list-style-type: none">• cscfprocessoroutage• pmindicator• sipinterfaces• cscfnetworkconnectivity• cscfmemusage• cscfcpload• controllerstatus• systemenvironmentvariables• cscfdnsservers• alarms• cscfisopstateandadminstate• cscfsystemstate• evipstatus• diameterports <p>It is possible to give multiple checks as input, separated with whitespaces, commas, or a combination of both.</p>
-type USECASE	<p>Specifies the use case to run.</p> <p>The possible values are basic and troubleshooting.</p>
-verbose	Prints the used input commands and their raw output.
--cpu-max=CPU_MAX	Sets the threshold, in %, that the CPU load must reach for the healthcheck script to flag VERIFY instead of OK.



4 Configuration File

This section describes the configuration file used by the automatic health check script. It contains parameters that can be configured by the user.

4.1 Configuration File Format

Each setting consists of a line with the format **key=value**. Multiple-value settings are handled by including multiple lines with the same key.

Lines that start with # are ignored.

4.2 Configuration File Location

The configuration file is stored in <config-path>/vcscf_cxp9034345/healthcheck/, and is called <user-name>.config.

The file permission is set to read and write for the user.

4.3 Configuration Parameters

If the configuration file does not exist, the script creates a default configuration file including default values. Some parameters are mandatory but are not given any default value. For the following parameters, the user is prompted to enter a value. The configuration file is updated with the following entered values:

- Cluster User
- Cluster Password
- O&M Host
- O&M User
- O&M Password

4.3.1 Cluster Port

The parameter `cluster.port` configures the port used when SSH to system controller on the cluster.

This parameter is mandatory. By default this parameter has the value 22.



4.3.2 Cluster User

The parameter `cluster.user` configures the user used when SSH to system controller on the cluster.

This parameter is mandatory. By default this parameter is not configured and must be updated for the script to work properly.

4.3.3 Cluster Password

The parameter `cluster.password` configures the password used when SSH to system controller on the cluster.

This parameter is mandatory.

Note: For security reasons, the password is not stored in the configuration file, and the password cannot be read from the configuration file. As a result, the user is always prompted for password.

4.3.4 Granularity Period

The parameter `granularity.period` configures the `cscfHealthCheck` script to select the Performance Management (PM) log files with the specific Granularity Period. The value of this parameter is in seconds. The default value for Granularity Period is 300 seconds.

Note: It is recommended to configure PM jobs with single Granularity Period.

4.3.5 O&M Host

The parameter `oam.host` configures the host address or hostname to be used when SSH to Ericsson Command-Line Interface (ECLI).

This parameter is mandatory. By default this parameter is not configured and must be updated for the script to work properly.

Note: If the O&M Movable IP (MIP) is configured on the system, the O&M host address is automatically retrieved and not prompted for user input.

4.3.6 O&M Port

The parameter `oam.ecliport` configures the port used when SSH to ECLI.

This parameter is mandatory. By default this parameter has the value 2022.

4.3.7 O&M User

The parameter `oam.user` configures the user used when SSH to ECLI.



This user must have a role that allows the user to log on to ECLI and display alarms and configuration parameters.

This parameter is mandatory. By default this parameter is not configured and must be updated for the script to work properly.

4.3.8 O&M Password

The parameter `oam.password` configures the password to be used when O&M user SSH to ECLI.

This parameter is mandatory.

Note: For security reasons, the password is not stored in the configuration file, and the password cannot be read from the configuration file. As a result, the user is always prompted for password.

4.3.9 Maximum CPU Load

The parameter `cpu.max` sets the threshold that the CPU load must reach for the healthcheck script to flag VERIFY instead of OK.

The default value is 81%.

4.3.10 Counters

The parameter `pmf.counters` configures counters from which the `CscfHealthCheck` script retrieves information.

By default, the following counters are configured:

- `cscfAcceptedRegistrations`
- `cscfExpiredRegistrations`
- `cscfRejectedRegistrations`
- `cscfFailedSessions`
- `cscfScscfAssignments`
- `cscfCxSelPullinitRegistrations`
- `cscfCxPullUnableToComplys`
- `cscfACABackup`
- `cscfNBASuccess`
- `cscfSipDigestAuthenticationSuccess`
- `scscfGibaSuccess`





5 Manual Health Check Procedure

This section describes the procedure for manually checking the health of the system for the CSCF.

5.1 Preparations

This section describes the required preparations performed before checking the node health.

5.1.1 Log on to System Controller

Log on to the system controller using SSH:

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

5.2 Check Release Information

For more information regarding the ECLI, see *Ericsson Command-Line Interface User Guide*.

To check the CSCF Release Information:

1. Log on to ECLI:

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

2. Check the release parameter in ManagedElement:

```
show ManagedElement=<nodename>
```

Example output:

```
ManagedElement=1
managedElementType="vCSCF"
release="1.8.0"
CscfFunction=1
[...]
```

The following example output shows an vCSCF 1.8.0 Emergency Package (EP) release.



```
ManagedElement=1
  managedElementType="vCSCF"
  release="1.8.1"
  CscfFunction=1
  Equipment=1
  SystemFunction
```

Note: The example shows the vCSCF 1.8.0 EP release. There are rare cases that an vCSCF EP is released only with a platform component update. Such EP releases do not result in an update of the release parameter in ManagedElement.

The application and platform component versions can be checked by following section Software Level Checks in [CSCF Troubleshooting Guideline](#).

3. Check the release parameter in CscfFunction=1:

```
show ManagedElement=<nodename>,CscfFunction=1
```

Example output:

```
CscfFunction=1
  release="CXP9035589/1 R9A (1.8.0-6)"
  userLabel=""
  CSCF-Application=CSCF
  CscfDomainRoutingApplication=CscfDomainRouting
  CscfEosApplication=CscfEos
  DIA-CFG-Application=DIA
  DNS-Application=DNS
  ExtNetSel-Application=ExtNetSelection
  ExtNetSel-Application=ExtNetSelection2
  ICMP-Application=ICMP
  LdapClientApplication=LdapClientApplication
  LI-Application=LI
  NumberNormalisation=NumberNormalisation
  SigComp-Subsystem=SigComp
```

4. Log off from the ECLI:

```
exit
```

5.3 Verify Status of Alarms

For more information regarding the ECLI, see [Ericsson Command-Line Interface User Guide](#).

To verify the status of alarms:

1. Log on to ECLI:



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

2. Verify that there are no raised alarms:

```
show ManagedElement=<nodename>,SystemFunctions=1,Fm=1 -m  
FmAlarm
```

If an alarm is found, follow the procedures in the related alarm Operating Instruction.

Example output:

```
FmAlarm=65  
  activeSeverity=WARNING  
    additionalText="Detailed Information: Link disabled by  
    OAM, IRP Cause: 14"  
  eventType=COMMUNICATIONSALARM  
  lastEventTime="2014-04-14T15:35:35+02:00"  
  majorType=193  
  minorType=2250572778  
  probableCause=14  
  sequenceNumber=65  
  source="ManagedElement=jambala, connId =conn1,  
    stack=CSCFRF,Host=LABSPTOFFCHA.ericsson.se"  
  specificProblem="Diameter, Link Disabled"  
FmAlarm=89  
  activeSeverity=WARNING  
    additionalText="Detailed Information: Link disabled by  
    OAM, IRP Cause: 14"  
  eventType=COMMUNICATIONSALARM  
  lastEventTime="2014-04-14T15:35:43+02:00"  
  majorType=193  
  minorType=2250572778  
  probableCause=14  
  sequenceNumber=89  
  source="ManagedElement=jambala, connId =conn1,  
    stack=CSCFRF,Host=LABSPTOFFCHA2.ericsson.se"  
  specificProblem="Diameter, Link Disabled"
```

3. Log off from ECLI:

```
exit
```

5.4 Verify Controller Status

To verify the controller status:

1. Check that the system controller is connected to the other half, that is, the output is Connected.

Note: If not, and if it does not resolve itself within 15 minutes, contact next level of maintenance support.



```
ssh `cmw-hostname-get SC-1` drbdadm cstate all
```

```
ssh `cmw-hostname-get SC-2` drbdadm cstate all
```

```
Connected
```

2. Check that the disk state is normal state UpToDate.

Note: If not, and if it does not resolve itself within a reasonable time frame, contact next level of maintenance support.

```
ssh `cmw-hostname-get SC-1` drbdadm dstate all
```

```
ssh `cmw-hostname-get SC-2` drbdadm dstate all
```

```
UpToDate
```

3. Check if the system controller is primary or secondary.

```
ssh `cmw-hostname-get SC-1` drbdadm role all
```

```
ssh `cmw-hostname-get SC-2` drbdadm role all
```

```
Primary/Secondary
```

On the primary controller, the field starts with Primary/, typically the value is Primary/Secondary.

On the secondary controller, the field starts with Secondary/, typically the value is Secondary/Primary.

Note: If an error has occurred, then the field can contain other values.

4. Display the state from CoreMW point of view:

```
cmw-status -v siass | grep OpenSAF -A2
```

Example output:



```
safSISU=safSu=PL-3\,safSg=NoRed\,safApp=OpenSAF,
safSi=NoRed4,safApp=OpenSAF
    HASTate=ACTIVE(1)
    HAREadinessState=READY_FOR_ASSIGNMENT(1)
safSISU=safSu=SC-2\,safSg=NoRed\,safApp=OpenSAF,
safSi=NoRed1,safApp=OpenSAF
    HASTate=ACTIVE(1)
    HAREadinessState=READY_FOR_ASSIGNMENT(1)
safSISU=safSu=SC-2\,safSg=2N\,safApp=OpenSAF,safSi=SC-2N,
safApp=OpenSAF
    HASTate=STANDBY(2)
    HAREadinessState=READY_FOR_ASSIGNMENT(1)
safSISU=safSu=PL-4\,safSg=NoRed\,safApp=OpenSAF,
safSi=NoRed3,safApp=OpenSAF
    HASTate=ACTIVE(1)
    HAREadinessState=READY_FOR_ASSIGNMENT(1)
--
safSISU=safSu=SC-1\,safSg=2N\,safApp=OpenSAF,safSi=SC-2N,
safApp=OpenSAF
    HASTate=ACTIVE(1)
    HAREadinessState=READY_FOR_ASSIGNMENT(1)
safSISU=safSu=SC-1\,safSg=NoRed\,safApp=OpenSAF,
safSi=NoRed2,safApp=OpenSAF
    HASTate=ACTIVE(1)
    HAREadinessState=READY_FOR_ASSIGNMENT(1)
```

5. Verify that HASTate is ACTIVE or STANDBY for:

```
safSISU=safSu=SC-1\,safSg=2N\,safApp=OpenSAF,safSi=SC-2N,safApp=OpenSAF
safSISU=safSu=SC-2\,safSg=2N\,safApp=OpenSAF,safSi=SC-2N,safApp=OpenSAF
```

5.5 Verify System Status

To verify the system status:

1. Display the vDicos Middleware (MW) status:

```
immlist lpmsv=LPMSvSite
```

For more information, see [vDicos Management](#).

Example output:



Name	Type	Value(s)
=====		
lpmsvState	SA_STRING_T	Idle
lpmsv	SA_NAME_T	lpmsv=LPMSvSite(15)
SaImmAttrImplementerName	SA_STRING_T	LPMSvImplementer
SaImmAttrClassName	SA_STRING_T	LPMSv
SaImmAttrAdminOwnerName	SA_STRING_T	IMMLOADER

2. Verify that lpmsvState is Idle.

5.6 Monitor Network Connectivity

To verify the network connectivity:

1. Use **ssh** to connect to a Payload (for example PL-3) using the cluster user and password:

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

2. Display Container Distribution Service (CDSv) connections of the control server:

```
clurun.sh -c ctrlsrv_print_conn
```

Example output:



Result from [.cdsv.director]:

DBSv

```
Server port:      <1.1.2:3151823157>
Client port:     <1.1.2:3151823091>
Connection status: established
Client version:  1
Server version:  1
Common version:  1
Queued messages: 0
```

LPMSv

```
Server port:      <1.1.2:3151692052>
Client port:     <1.1.2:3151757573>
Connection status: established
Client version:  1
Server version:  1
Common version:  1
Queued messages: 0
```

Result from [SC-2.cdsv.director]:

DBSv

```
Server port:      <1.1.2:3151823157>
Client port:     <1.1.2:3151823091>
Connection status: established
Client version:  1
Server version:  1
Common version:  1
Queued messages: 0
```

LPMSv

```
Server port:      <1.1.2:3151692052>
Client port:     <1.1.2:3151757573>
Connection status: established
Client version:  1
Server version:  1
Common version:  1
Queued messages: 0
```

3. Verify that Connection status is established.
4. Display CDSv Connections of the distribution server:

```
clurun.sh -c distsrv_print_conn
```

Example output:



```
Result from [.cdsv.director]:  
  
...  
  
DBSV on safAmfNode=SC-2,safAmfCluster=myAmfCluster  
  Server port:          <1.1.2:3151888694>  
  Client port:          <1.1.2:3152347376>  
  Connection status:    established  
  Client version:       1  
  Server version:       1  
  Common version:       1  
  Queued messages:      0  
  
LPMSv on safAmfNode=PL-3,safAmfCluster=myAmfCluster  
  Server port:          <1.1.2:3151692090>  
  Client port:          <1.1.3:4076011595>  
  Connection status:    established  
  Client version:       1  
  Server version:       1  
  Common version:       1  
  Queued messages:      0
```

Note: Only part of output is shown here.

5. Verify that `Connection status` is established.

5.7 Verify Processor Status

To verify the processor status:

1. Use `ssh` to connect to a Payload (for example PL-3) using the cluster user and password:

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

2. Enter the command:

```
cdsv-get-user-state
```

Example output:



```

Result from [.cdsv.user.director.DBSv]:
DBSv - cluster state: Idle
Agent server port: <1.1.2:3151692031>, listening on 92345,99
Agents (count: 4):
Agent[0x670430] node: safAmfNode=PL-3,safAmfCluster=myAmfCluster,
  port: <1.1.3:4075946061>
Idle: yes, Halting: no
Number of operation states: 0

...

Result from [.cdsv.user.director.LPMSv]:
LPMSv - cluster state: Idle
Agent server port: <1.1.2:3151692041>, listening on 12345,1
Agents (count: 4):
Agent[0x7099e0] node: safAmfNode=PL-3,safAmfCluster=myAmfCluster,
  port: <1.1.3:4075946057>
Idle: yes, Halting: no
Number of operation states: 0

...

```

Note: Only part of output is shown here.

3. Verify that DBSv - cluster state and LPMSv - cluster state are both Idle.

5.8 Verify Administrative and Operational State

For more information regarding the ECLI, see [Ericsson Command-Line Interface User Guide](#).

To verify that the CSCF is ready to handle traffic:

1. Log on to ECLI:

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

2. Display parameter cscfISPOperationalState and cscfAdministrativeState under CSCF-Application=CSCF:

```
show ManagedElement=<nodename>,CscfFunction=1,CSCFApplication=CSCF
```

Example output:



```

CSCF-Application=CSCF
  cscfActiveUserMethod
    ""
    cscfAdministrativeState=UNLOCKED
    cscfCXDestinationHost="LAB7HSS.ericsson.se"
    cscfCXDestinationRealm="cx.ericsson.se"
    cscfCXOriginHost="LAB24CSCF.ericsson.se"
    cscfCXOriginRealm="cscf.ericsson.se"
    cscfDomainAlias
      "cscf24.lab"
    cscfDomainBasedPSIRoutingEntry
      "/^psi\\.cscf24\\.lab/i"
    cscfGlobalNumberNormalizationPhoneContext=""
    cscfISPOperationalState=ENABLED
    cscfPhoneContext="+46"
    ...

```

Note: Only part of output is shown here.

3. Verify that the parameter `cscfISPOperationalState` has the value `ENABLED` and parameter `cscfAdministrativeState` has the value `UNLOCKED`.
4. Log off from ECLI:

```
exit
```

5.9 Verify Diameter Stack Status

To verify the status of the Diameter stack:

1. Use `ssh` to connect to a Payload (for example PL-3) using the cluster user and password:

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

2. Check which nodes in the cluster that are started:

```
cdsv-get-node-state -s
```

3. Check which started nodes that have `DIASharedProcPool` hosted. If already known, continue with Step 4.

```
cdsv-print-node -v | egrep -w 'DIASharedProcPool' -B 3
```

Example output:

```

Result from [.cdsv.director]:
Node[0x8ebfd0] id: 9 name: safAmfNode=PL-10,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 0100100a
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASharedPool CscfPool CscfOamProcPool

```



```

CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x8b9940] id: 8 name: safAmfNode=PL-11,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 0100100b
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x876600] id: 6 name: safAmfNode=PL-12,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 0100100c
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x822540] id: 3 name: safAmfNode=PL-3,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001003
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x7df040] id: 1 name: safAmfNode=PL-4,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001004
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x767b30] id: 0 name: safAmfNode=PL-5,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001005
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x8a8be0] id: 7 name: safAmfNode=PL-6,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001006
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x8658c0] id: 5 name: safAmfNode=PL-7,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001007
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]

```



```
Node[0x843f00] id: 4 name: safAmfNode=PL-8,  
safAmfCluster=myAmfCluster state: 2 (Started)  
flags: 00000001 address: 01001008  
  UserApp states: [ 0: 00000003 1: 00000003 ]  
  Hosted pools: [ DIASharedPool CscfPool CscfOamProcPool  
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool  
DicosDbClassPot_Pool DIASharedProcPool ]  
Node[0x7eff50] id: 2 name: safAmfNode=PL-9,  
safAmfCluster=myAmfCluster state: 2 (Started)  
flags: 00000001 address: 01001009  
  UserApp states: [ 0: 00000003 1: 00000003 ]  
  Hosted pools: [ DIASharedPool CscfPool CscfOamProcPool  
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool  
DicosDbClassPot_Pool DIASharedProcPool ]
```

4. Log on to ECLI:

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

5. Check portNr, ipAddressesList, and sctpAddressesList for each configured diameter interface. If already known, continue with Step 8.

```
show ManagedElement=<nodename>,CscfFunction=1,DIA-CFG-Applica  
tion=DIA,DIA-CFG-StackContainer=CSCFCX,DIA-CFG-OwnNodeConfig=  
CSCFCX
```

```
show ManagedElement=<nodename>,CscfFunction=1,DIA-CFG-Applica  
tion=DIA,DIA-CFG-StackContainer=CSCFRF,DIA-CFG-OwnNodeConfig=  
CSCFRF
```

```
show ManagedElement=<nodename>,CscfFunction=1,DIA-CFG-Applica  
tion=DIA,DIA-CFG-StackContainer=CSCFR0,DIA-CFG-OwnNodeConfig=  
CSCFR0
```

```
show ManagedElement=<nodename>,CscfFunction=1,DIA-CFG-Applica  
tion=DIA,DIA-CFG-StackContainer=CSCFRX,DIA-CFG-OwnNodeConfig=  
CSCFRX
```

Example output:



```

DIA-CFG-OwnNodeConfig=CSCFCX
  allowConnectFromUnknownNode=false
  diaVendorId="10415"
  enabled=true
  firmwareRevision="0"
  hostId="LAB24CSCF.ericsson.se"
  ipAddressesList
    "0:10.35.38.14"
  loadRegulationEnabled=false
  maxNumberOfRetries="2"
  maxRequestPendingTime="4"
  permissions=63
  portNr="3868"
  productName="ISP-CSCF"
  realm="cscf.ericsson.se"
  sctpHandlerLogLevel="DEFAULT"
  sendErrorAtOverload=false
  shareTree=""
  supportedAuthAppIds
    "16777216"
    "16777217"
  supportedVendorsIds
    "0"
    "10415"
    "13019"
  supportedVendorSpecificApps
    "0:0:16777216:16777216"
    "1:10415:16777216:16777216"
    "2:0:16777217:16777217"
    "3:10415:16777217:16777217"
  traceSctpHandler="DEFAULT"
  transportLayerType="1"

```

6. Make a notation of portNr, ipAddressesList, and sctpAddressesList.
7. Log off from ECLI:

```
exit
```

8. Check the Transmission Control Protocol (TCP) diameter port status and verify the ports that are available for use for each interface on each node:

```
ssh -A <node hostname> netstat -an | grep <tcp address>:<port>
```

Example output:

```
tcp      0      0 10.35.38.14:3868      0.0.0.0:*      LISTEN
```

Note: netstat cannot be used to check the status of Stream Control Transmission Protocol (SCTP) diameter ports.



5.10 Verify CPU Load and Memory Use

To verify the CPU load and memory use:

1. Log on to a Payload (for example PL-3) using the cluster user and password:

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

2. Check which nodes in the cluster that are started:

```
cdsd-get-node-state -s
```

3. Check which started nodes that have CscfPool or DIASStackProcPool hosted. If already known, continue with Step 4.

```
cdsd-print-node -v | egrep -w 'CscfPool|DIASStackProcPool' -B 3
```

Example output:

```
Result from [.cdsv.director]:
Node[0x8ebfd0] id: 9 name: safAmfNode=PL-10,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 0100100a
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x8b9940] id: 8 name: safAmfNode=PL-11,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 0100100b
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x876600] id: 6 name: safAmfNode=PL-12,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 0100100c
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x822540] id: 3 name: safAmfNode=PL-3,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001003
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASStackPool CscfPool CscfOamProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASStackProcPool ]
Node[0x7df040] id: 1 name: safAmfNode=PL-4,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001004
  UserApp states: [ 0: 00000003 1: 00000003 ]
```



```

Hosted pools: [ DIASharpPool CscfPool Cscf0amProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASharpProcPool ]
Node[0x767b30] id: 0 name: safAmfNode=PL-5,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001005
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASharpPool CscfPool Cscf0amProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASharpProcPool ]
Node[0x8a8be0] id: 7 name: safAmfNode=PL-6,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001006
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASharpPool CscfPool Cscf0amProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASharpProcPool ]
Node[0x8658c0] id: 5 name: safAmfNode=PL-7,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001007
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASharpPool CscfPool Cscf0amProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASharpProcPool ]
Node[0x843f00] id: 4 name: safAmfNode=PL-8,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001008
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASharpPool CscfPool Cscf0amProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASharpProcPool ]
Node[0x7eff50] id: 2 name: safAmfNode=PL-9,
safAmfCluster=myAmfCluster state: 2 (Started)
flags: 00000001 address: 01001009
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ DIASharpPool CscfPool Cscf0amProcPool
CMCO_VDicosPool vDicosEEPool LpmsvCommonPool vDicosOAMPool
DicosDbClassPot_Pool DIASharpProcPool ]

```

4. Display the status of the Virtual Machines (VMs) on each node that has CscfPool or DIASharpProcPool hosted:

```
clurun.sh -c vmstatus -n <node>
```

For example:

```
clurun.sh -c vmstatus -n PL-3.lpmsv.agent.vm0
```

Example output:

Result from [PL-3.lpmsv.agent.vm0]:
Configuration:



```
-----
Basic Interval:          1000 ms
Short Intervals:         1
Short Interval:          1000000 usec
Long term samples:       5

Current values:
-----
Reconfiguration ongoing: no

Resources:
-----
*CPU_AVG:
Core selection method:   avg
Limit:                   80.0%
Maint limit:             60.0%
Load:                    5.0%/4.0% (<short term>/<long term>)
Shared load:             5.0%/4.0% (<short term>/<long term>)
Rate delta:              -2138865795
Reject Rate:             0.000 (0)
Rejected:                0

CPU_CURRENT:
Core selection method:   current
Limit:                   100.0%
Maint limit:            60.0%
Load:                    4.0%/2.0% (<short term>/<long term>)
Rate delta:              -2130215175
Reject Rate:             0.000 (0)
Rejected:                0

CPU_MAX:
Core selection method:   max
Limit:                   100.0%
Maint limit:            100.0%
Load:                    26.0%/32.0% (<short term>/<long term>)
Shared load:            26.0%/32.0% (<short term>/<long term>)
Rate delta:              2095940371
Reject Rate:             0.000 (0)
Rejected:                0

Memory:
Memory limit:           100%
Usage base:             69%
Memory usage:           70% (17610153984 bytes free of 57831317504
                        total bytes)

Scaled values:
Limit:                   100.0%
Maint limit:            100.0%
Load:                    3.0%/3.0% (<short term>/<long term>)
Shared load:            3.0%/3.0% (<short term>/<long term>)
```




```

Rate delta:                -2147450880
Reject Rate:               0.000 (0)
Rejected:                  0

MultiMMap:
Multimap limit:            80%
Multimap maint limit:     60%
Usage base:                4%
Multimap usage:            4% (381371 pages allocated of 8471384 total
                             pages)

Scaled values:
Limit:                     79.0%
Maint limit:               58.0%
Load:                      0.0%/0.0% (<short term>/<long term>)
Shared load:               0.0%/0.0% (<short term>/<long term>)
Rate delta:                -2147450880
Reject Rate:               0.000 (0)
Rejected:                  0

TIPC incoming:
Tipc overload limit:       5000
Job count:                 0
Limit:                     80.0%
Maint limit:               60.0%
Load:                      0.0%/0.0% (<short term>/<long term>)
Rate delta:                -2147450880
Reject Rate:               0.000 (0)
Rejected:                  0

TIPC outgoing:
Tipc overload limit:       5000
Outgoing dialogue message count: 0
Limit:                     80.0%
Maint limit:               60.0%
Load:                      0.0%/0.0% (<short term>/<long term>)
Rate delta:                -2147450880
Reject Rate:               0.000 (0)
Rejected:                  0

Heap:
Heap limit:                80%
Heap maint limit:          60%
Usage base:                19%
Heap usage:                19% (268434432 total bytes = 52031032 used
                             bytes + 216403400 free bytes)

Scaled values:
Limit:                     75.0%
Maint limit:               50.0%
Load:                      0.0%/0.0% (<short term>/<long term>)
Rate delta:                -2147450880
Reject Rate:               0.000 (0)

```



Rejected: 0

Note: Only part of output is shown here.

5. Check the memory use, and verify that core load and core load (LT) is not more than 80%.
6. Log off from the PL:

exit

5.11 Verify eVIP Status

To verify the eVIP status:

1. Log on to the eVIP CLI:

telnet `/opt/vip/bin/getactivecontrol` 25190

2. Display eVIP link/agent status:

show agents

3. Verify that no eVIP links/agents are INACTIVE and DOWN.

Example output:



```

+-----[ ALB alb_0 (ACTIVE) ]-----+
+----- PN -----+
| pagent (4) | lbesel_pn (20) |
|[2] fe80::ff:fe01:e : ACTIVE | [2] fe80::ff:fe01:e : ACTIVE |
|[1] fe80::ff:fe01:b : ACTIVE | [1] fe80::ff:fe01:b : ACTIVE |
+-----+
| ersipc (0) | repdb (20) |
|[2] fe80::ff:fe01:e : ACTIVE | [2] fe80::ff:fe01:e : ACTIVE |
|[1] fe80::ff:fe01:b : ACTIVE | [1] fe80::ff:fe01:b : ACTIVE |
+----- LBE -----+
| lbeagent (28) | sesel_lbe (10) |
|[2] fe80::1:f4ff:fe01:4 : ACTIVE | [2] fe80::1:f4ff:fe01:4:ACTIVE |
|[1] fe80::1:f4ff:fe01:3 : ACTIVE | [1] fe80::1:f4ff:fe01:3:ACTIVE |
+----- FE -----+
| feeagent (18) | lbesel_fe (20) |
|[2] fe80::1:f6ff:fe01:9:INACTIVE DOWN | [2] fe80::1:f6ff:fe01:9:ACTIVE |
|[1] fe80::1:f6ff:fe01:7:INACTIVE DOWN | [1] fe80::1:f6ff:fe01:7:ACTIVE |
+-----+
| sesel_fe (10) | |
|[2] fe80::1:f6ff:fe01:9 : ACTIVE | |
|[1] fe80::1:f6ff:fe01:7 : ACTIVE | |
+----- SE -----+
| seagent (18) | lbesel_se (24) |
|[2] fe80::1:f5ff:fe01:6:ACTIVE RDY | [2] fe80::1:f5ff:fe01:6:ACTIVE |
|[1] fe80::1:f5ff:fe01:5:ACTIVE RDY | [1] fe80::1:f5ff:fe01:5:ACTIVE |
+-----+
| sesel_se (6) | |
|[2] fe80::1:f5ff:fe01:6 : ACTIVE | |
|[1] fe80::1:f5ff:fe01:5 : ACTIVE | |
+----- IPSEC -----+
| ikeagent (0) | ipsecuagent (10) |
| | [2] fe80::ff:fe01:10:ACTIVE RDY |
| | [1] fe80::ff:fe01:d:ACTIVE RDY |
+----- XALBSEL -----+
| xalbsel (4) | |
|[2] fe80::ff:fe01:f : ACTIVE | |
|[1] fe80::ff:fe01:c : ACTIVE | |
+-----+
eRSIP state: ACTIVE cIPSEC state: ACTIVE RDY
OK

```

4. Log off from the eVIP CLI:

exit

5.12 Check System Environment Variables

To check the environment variables:



1. Log on to a Payload (for example PL-3) using the cluster user and password:

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

2. List the environment variables:

```
for envEntry in `vdicos-envdata-list | sort`; do echo \  
$envEntry = `vdicos-envdata-get $envEntry`; done;
```

Example output:

```
...  
ASSIM_SINGLEPROCESS = 1  
CSCF_DBMONITOR_MEMORY_LIMIT = 400000000  
CSCF_ENS_RESOURCE_LIMIT = 10000  
CSCF_IPSEC_DISABLED_FOR_VEGA = 1  
CX_DIAMETER_STACKID = CSCFCX  
DIA_INSTALLER_0 = CSCFCX  
DIA_INSTALLER_1 = CSCFRF  
DIA_INSTALLER_2 = CSCFRO  
DIA_INSTALLER_3 = CSCFRX  
DIA_RESOURCE_LIMIT_CSCFCX = 80000  
DIA_RESOURCE_LIMIT_CSCFRF = 80000  
DIA_RESOURCE_LIMIT_CSCFRO = 80000  
DIA_RESOURCE_LIMIT_CSCFRX = 80000  
ICMP_CONTROLLER_RESOURCE_LIMIT = 10000  
IPMM_IS_PROCESSOR_VEGA = 1  
JIMAnonPermissions = 0  
JimDebugInfo = 255  
JimMonitorsEnabled = 0  
JimTcpPortNumber = 6497  
RF_DIAMETER_STACKID = CSCFRF  
RO_DIAMETER_STACKID = CSCFRO  
RX_DIAMETER_STACKID = CSCFRX  
SIP_TIMER_T1 = 5000  
...
```

Note: Only part of output is shown here.

3. Verify that the output is as expected (with only expected deviances for system size market adaptations, and so on).
4. Log off from the PL:

```
exit
```

5.13 Check Availability of DNS Servers

To check the availability of the Domain Name System (DNS) servers:

1. Log on to ECLI:



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

2. Check the CSCF DNS client source IP address:

```
show ManagedElement=<nodename>,CscfFunction=1,DNS-Application=DNS,dnsLocalAddress
```

Example output:

```
dnsLocalAddress
"10.50.10.1"
```

3. Check the CSCF DNS servers:

```
show ManagedElement=<nodename>,CscfFunction=1,DNS-Application=DNS,dnsServerEntry
```

Example output:

```
dnsServerEntry
"0:137.168.10.50:53"
```

4. Log off from ECLI:

```
exit
```

5. From a controller or payload, check which nodes in the cluster that are started:

```
cdsv-get-node-state -s
```

6. Check which started nodes that have CscfPool hosted.

```
cdsv-print-node -v | egrep -w 'CscfPool' -B 3
```

Example output:

```
Result from [.cdsv.director]:
Node[0x158d1a0] id: 0 name: safAmfNode=PL-3,safAmfCluster=myAmfCluster
state: 2 (Started) flags: 00000001 address: 01001003
UserApp states: [ 0: 00000003 1: 00000003 ]
Hosted pools: [ DIASStackPool CscfPool DIASStackProcPool Cscf0amProcPool
DicosDbClassPot_Pool CMC0_VDicosPool vDicosEEPool LpmsvCommonPool
vDicosOAMPool LiPool ]
Node[0x158d650] id: 1 name: safAmfNode=PL-4,safAmfCluster=myAmfCluster
state: 2 (Started) flags: 00000001 address: 01001004
UserApp states: [ 0: 00000003 1: 00000003 ]
Hosted pools: [ DIASStackPool CscfPool DIASStackProcPool Cscf0amProcPool
DicosDbClassPot_Pool CMC0_VDicosPool vDicosEEPool LpmsvCommonPool
vDicosOAMPool LiPool ]
```

7. Log on to a started node that has hosted the CscfPool:

```
ssh -A <username>@<payload>
```



8. For each DNS server, run `tracert` to check the DNS server status.

If the last hop is the destination address, the connectivity to the DNS servers is working. Otherwise, the connectivity is lost.

Note: `sudo` or root privileges are required for running `tracert`.

- a. If the source IP address (retrieved from the CSCF DNS configuration) is `0.0.0.0` (any):

```
tracert -I <DNS server IP address>
```

Example output:

```
tracert -I 137.168.10.50
tracert to 137.168.10.50 (137.168.10.50), 30 hops max,
60 byte packets
1  137.168.10.50 (137.168.10.50)  0.084 ms  0.013 ms  0.065 ms
```

- b. If the source IP address (retrieved from the CSCF DNS configuration) is **not** `0.0.0.0` (any):

```
tracert -I -s <source IP address> <DNS server IP address>
```

Example output:

```
tracert -I -s 10.50.10.1 137.168.10.50
tracert to 137.168.10.50 (137.168.10.50), 30 hops max,
60 byte packets
1  * * *
2  dns_src (10.50.10.1)  1.091 ms  1.078 ms  1.072 ms
3  192.168.216.6 (192.168.216.6)  1.284 ms  1.278 ms  1.268 ms
4  172.16.2.2 (172.16.2.2)  2.167 ms  2.169 ms  2.171 ms
5  172.16.2.1 (172.16.2.1)  2.431 ms  2.439 ms  2.457 ms
6  172.16.254.1 (172.16.254.1)  133.104 ms  132.076 ms  132.053 ms
7  137.168.10.50 (137.168.10.50)  1.106 ms  0.945 ms  0.919 ms
```

9. Log off from the node:

```
exit
```

5.14 Check Status of SIP Interfaces

To check the status of the SIP interfaces:

1. Log on to ECLI:

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

2. Navigate to the CSCF Network Interfaces:



```
ManagedElement=<node name>,CscfFunction=1,CSCF-Application=CS
CF,CscfNwIfContainer=0
```

3. For each network interface, for example IcscfNwIfs=0 and ScscfNwIfs=0, check that Status is OK.:

```
show <NetworkInterface>=<transport-protocol>:<IP
address>:<port>,<NetworkInterface>Status
```

Example:

```
show IcscfNetworkInterface=TCP:192.168.10.201:5060,icscfNetwo
rkInterfaceStatus
```

Example output:

```
icscfNetworkInterfaceStatus=OK
```

If the Status is not OK, see Section 6 on page 37.

4. Log off from ECLI:

```
exit
```

5.15 Store Health Check Report

To store the health check report: save the report in an agreed format and store it persistently.





6 Report Problems

For any abnormal situation, see [CSCF Troubleshooting Guideline](#).

If the problem still exists, report it to the next level of support.

It is also important to collect the related data. For information about how to collect the data, see [Data Collection Guideline for CSCF](#).





7 Example of Configuration File

Example 1 shows an example of the configuration file used by the automatic health check script.

```
# Configuration file for CscfHealthCheck
#
# Lines starting with # contain comments and are ignored.
#
# Information for logging in to cluster (controller)

# Port to be used when SSH to system controller on the cluster.
# Default port is 22
cluster.port=22

# User to be used when SSH to system controller on the cluster.
cluster.user=root

# Connection and authentication settings for ECLI.
# Address to be used when SSH to ECLI, usually the OAM VIP.
oam.host=192.168.10.200

# Port to be used when SSH to ECLI.
# Default port is 2022.
oam.ecliport=2022

# O&M user
oam.user=jambala_caa

# Settings for accessing PMF counter data.
# Counters to include by default. Repeat for multiple values.
# Format: NAME or NAME.KEY.
pmf.counters=cscfAcceptedRegistrations
pmf.counters=cscfExpiredRegistrations
pmf.counters=cscfRejectedRegistrations
pmf.counters=cscfFailedSessions
pmf.counters=cscfScscfAssignments
pmf.counters=cscfCxSelPullInitRegistrations
pmf.counters=cscfCxPullUnableToComplys
pmf.counters=cscfACABackup
pmf.counters=cscfNBASuccess
pmf.counters=cscfSipDigestAuthenticationSuccess
pmf.counters=scscfGibaSuccess

# the time at PM counter values should be collected from , could be the current time
# Time Format should be day/month/year hour:minutes [dd/mm/yy hh:mm]
start.time=None

# the time at PM counter values should be collected to , it determine the total time
# time duration for logs collection considering the start time
# Time Format should be day/month/year hour:minutes [dd/mm/yy hh:mm]
end.time=None

# Threshold that the CPU load must reach for the
# healthcheck script to flag VERIFY instead of OK.
# Default value is 81%.
cpu.max=81

#configure to select the PM log files with the specific granularity period,
#the value is in seconds
granularity.period=300
```

Example 1 Example of Configuration File





8 Example of Automatic Health Check Results

```
<HealthCheckReport>
  <vnfName>jambala</vnfName>
  <vnfType>vCSCF</vnfType>
  <vnfRelease>"1.8.0-6"</vnfRelease>
  <vnfProductNumber>CXP9034345/1</vnfProductNumber>
  <vnfProductRevision>R9A06</vnfProductRevision>
  <site>[]</site>
  <options>-verbose -type basic </options>
  <startDateTime>2018-09-10T17:08:22+02:00</startDateTime>
  <type>basic</type>
  <result>NOT_HEALTHY</result>
  <description>
    Report(s): /storage/no-backup/vcscf_CXP9034345/healthcheck/reports/\⇒
    vCSCF_HC_jambala_2018-09-10T17:08:22+02:00_basic.*
    INFO: Information for the user, not checked by the script
    OK: Task passed
    VERIFY: Manual verification needed
    FAIL: Problem detected by the script
    ERROR: An error occurred, script update needed or system broken
  </description>
  <Check name="FM Alarms and Notifications">
    <verdict>FAIL</verdict>
    <reason>
      VERIFY: MINOR: ManagedElement=jambala,Stack=CSCFRF,Host=LABSIMOFFCHA2.ericsson.se, \⇒
      "vDicos, Diameter Peer Node Disabled"
      VERIFY: WARNING: ManagedElement=jambala,Conn=conn1,Stack=CSCFRF,Host=LABSIMOFFCHA.ericsson.se, \⇒
      "vDicos, Diameter Link Disabled"
      <...>
      VERIFY: WARNING: ManagedElement=jambala,Conn=conn1,Stack=CSCFCX,Host=LABSPTSLF2.ericsson.se, \⇒
      "vDicos, Diameter Link Disabled"
      FAIL: MAJOR: ManagedElement=jambala,Conn=conn1,Stack=CSCFCX,Host=LAB0HSS.ericsson.se, \⇒
      "vDicos, Diameter Link Failure"
    </reason>
    <recommendedAction>Please do more advanced troubleshooting</recommendedAction>
    <description>Verify Status of Alarms</description>
    <rawinput>
      prompt @COMCLI;
      show;
      ManagedElement=jambala;
      SystemFunctions=1;
      Fm=1;
      show all;
    </rawinput>
    <rawoutput>
      ManagedElement=jambala
      Fm=1
      lastChanged="2018-09-10T16:59:11.186+02:00"
      lastSequenceNo=148
      sumCritical=0
      sumMajor=1
      sumMinor=1
      sumWarning=14
      totalActive=16
      FmAlarm=15
      activeSeverity=MINOR
      additionalText="Detailed Information: Peer Node disabled by OAM, IRP Cause: 14"
      eventType=COMMUNICATIONSALARM
      lastEventTime="2018-09-10T10:11:12.366+02:00"
      majorType=193
      minorType=2250572780
      originalAdditionalText="Detailed Information: Peer Node disabled by OAM, IRP Cause: 14"
      originalEventTime="2018-09-10T10:11:12.366+02:00"
      originalSeverity=MINOR
      probableCause=14
      sequenceNumber=20
      source="ManagedElement=jambala,Stack=CSCFRF,Host=LABSIMOFFCHA2.ericsson.se"
      specificProblem="vDicos, Diameter Peer Node Disabled"
      <...>
      FmAlarmModel=CW
      FmAlarmType=ComSaCLMClusterNodeUnavailable
    </rawoutput>
  </Check>
</HealthCheckReport>
```



```
        additionalText=""
        eventType=PROCESSINGERRORALARM
        isStateful=true
        majorType=193
        minorType=849346561
        moClasses="Amf"
        probableCause=418
        specificProblem="COM SA, CLM Cluster Node Unavailable"
    <...>
</rawoutput>
</Check>
<Check name="CSCF CPU Load">
    <verdict>OK</verdict>
    <reason>
        OK: SC-1:CPU Load Short Term: 32%
        OK:           Long Term: 56%
    </reason>
    <recommendedAction>No Action</recommendedAction>
    <description>Verify CPU Load</description>
    <rawinput>
        cdsv-get-node-state -s;
        cdsv-print-node -v;
        cdsv-get-node-state -s;
        cdsv-print-node -v;
        cmw-hostname-get SC-1;
        clurun.sh vmstatus;
    </rawinput>
    <rawoutput>
        Result from [.cdsv.director]:
        safAmfNode=SC-1,safAmfCluster=myAmfCluster is Started

        Result from [.cdsv.director]:
        Node[0x214a600] id: 0 name: safAmfNode=SC-1,safAmfCluster=myAmfCluster state: \⇒
        2 (Started) flags: 00000001 address: 01001001
        UserApp states: [ 0: 00000003 1: 00000003 ]
        Hosted pools: [ dut.rtid.1105883 dut.rtid.1107008 dut.rtid.1107094 dut.rtid.1108366 \⇒
        <...>
        Result from [.cdsv.director]:
        safAmfNode=SC-1,safAmfCluster=myAmfCluster is Started

        Result from [.cdsv.director]:
        Node[0x214a600] id: 0 name: safAmfNode=SC-1,safAmfCluster=myAmfCluster state: \⇒
        2 (Started) flags: 00000001 address: 01001001
        UserApp states: [ 0: 00000003 1: 00000003 ]
        Hosted pools: [ dut.rtid.1105883 dut.rtid.1107008 dut.rtid.1107094 dut.rtid.1108366 \⇒
        <...>
        SC-1
        Result from [SC-1.lpmv.agent]:
        Current loadbalance algorithm: table
        Update count: 25066
        Load balance timer interval: 1000
        Started processes: 0
        Last started processes: 0
        <...>
        Count of VMs: 4
        =====
        VM 0
        -----
        device: 0
        core: 0
        status: started
        core load: 32%
        core load (LT): 56%
        load balance load: 38%
        l.b. load (LT): 57%
        last token: 86
        selected: 1323
        <...>
    </rawoutput>
</Check>
<Check name="EVIP">
    <verdict>ERROR</verdict>
    <reason>ERROR: EVIP might not be installed. Please check.</reason>
    <recommendedAction>Script update is needed or system is broken</recommendedAction>
    <description>Verify eVIP Status</description>
    <rawinput>/opt/vip/bin/getactivecontrol;</rawinput>
```



```

<rawoutput>-bash: /opt/vip/bin/getactivecontrol: No such file or directory</rawoutput>
</Check>
<Check name="CSCF Memory Usage">
  <verdict>OK</verdict>
  <reason>OK: SC-1:Memory Usage: 93%</reason>
  <recommendedAction>No Action</recommendedAction>
  <description>Verify Memory Usage</description>
  <rawinput>
    cdsv-get-node-state -s;
    cdsv-print-node -v;
    cmw-hostname-get SC-1;
    clurun.sh -c printloadreg -n SC-1;
    cmw-amfnode-get SC-1;
  </rawinput>
  <rawoutput>
    Result from [.cdsv.director]:
    safAmfNode=SC-1,safAmfCluster=myAmfCluster is Started

    Result from [.cdsv.director]:
    Node[0x214a600] id: 0 name: safAmfNode=SC-1,safAmfCluster=myAmfCluster state: \⇒
    2 (Started) flags: 00000001 address: 01001001
    UserApp states: [ 0: 00000003 1: 00000003 ]
    Hosted pools: [ dut.rtid.1105883 dut.rtid.1107008 dut.rtid.1107094 dut.rtid.1108366 \⇒
    <...>
    SC-1
    Result from [SC-1.lpmsv.agent.vm0]:
    Configuration:
    -----
    Basic Interval:          1000 ms
    Short Intervals:         1
    Short Interval:          1000000 usec
    Long term samples:       5

    Current values:
    -----
    Reconfiguration ongoing: no

    Resources:
    -----
    CPU_AVG:
    Core selection method:   avg
    Limit:                   80.0%
    Maint limit:             60.0%
    Load:                   31.0%/36.0% (<short term>/<long term>)
    Shared load:             31.0%/36.0% (<short term>/<long term>)
    Rate delta:              -2105311875
    Reject Rate:             0.000 (0)
    Rejected:                0
    <...>
  </rawoutput>
</Check>
<Check name="CSCF Operational and Administrative State">
  <verdict>OK</verdict>
  <reason>
    OK: cscfISPOperationalState=ENABLED
    OK: cscfAdministrativeState=UNLOCKED
  </reason>
  <recommendedAction>No Action</recommendedAction>
  <description>Verify Administrative and Operational State</description>
  <rawinput>
    prompt @COMCLI;
    show;
    ManagedElement=jambala;
    CscfFunction=1,CSCF-Application=CSCF;
    show cscfISPOperationalState;
    show cscfAdministrativeState;
  </rawinput>
  <rawoutput>
    ManagedElement=jambala

    cscfISPOperationalState=ENABLED
    cscfAdministrativeState=UNLOCKED
  </rawoutput>
</Check>
<Check name="System Environment Variables">
  <verdict>OK</verdict>

```



```
<reason>
INFO: CSCF_DBMONITOR_MEMORY_LIMIT: 600000000
INFO: CSCF_IPSEC_DISABLED_FOR_VEGA: 1
INFO: CSCF_VEGA_SKIP_LOAD_CHECK: 1
INFO: CTF_LOG_LEVEL: WAR
INFO: CX_DIAMETER_STACKID: CSCFCX
INFO: DIA_INSTALLER_0: CSCFCX
<...>
</reason>
<recommendedAction>No Action</recommendedAction>
<description>Check System Environment Variables</description>
<rawinput>
for i in `immfind | grep -i lpmsvCfgAttr`; do echo $i; immlist -a lpmsvCfgAttrVal $i; done;
</rawinput>
<rawoutput>
lpmsvCfgAttr=CSCF_DBMONITOR_MEMORY_LIMIT,lpmsv=LPMSvSite
lpmsvCfgAttrVal=600000000
lpmsvCfgAttr=CSCF_IPSEC_DISABLED_FOR_VEGA,lpmsv=LPMSvSite
lpmsvCfgAttrVal=1
lpmsvCfgAttr=CSCF_VEGA_SKIP_LOAD_CHECK,lpmsv=LPMSvSite
<...>
</rawoutput>
</Check>
<Check name="CSCF System State">
<verdict>OK</verdict>
<reason>OK: System State: Idle</reason>
<recommendedAction>No Action</recommendedAction>
<description>Verify System Status</description>
<rawinput>immlist lpmsv=LPMSvSite;</rawinput>
<rawoutput>
Name                                     Type      Value(s)
=====
lpmsvState                             SA_STRING_T Idle
lpmsv                                   SA_NAME_T  lpmsv=LPMSvSite (15)
SaImmAttrImplementerName               SA_STRING_T LPMSvImplementer
SaImmAttrClassName                     SA_STRING_T LPMSv
SaImmAttrAdminOwnerName                 SA_STRING_T <Empty>
</rawoutput>
</Check>
<Check name="Controller status - SC-1">
<verdict>FAIL</verdict>
<reason>
OK: ro:Primary -- This SC is Primary
FAIL: cs:Connecting should be Connected
OK: ds:UpToDate/DUnknown
OK: This controller is ACTIVE on CoreMW level
</reason>
<recommendedAction>Please do more advanced troubleshooting</recommendedAction>
<description>Verify Controller Status</description>
<rawinput>
cmw-hostname-get SC-1;
/sbin/drbdadm role all;
/sbin/drbdadm cstate all;
/sbin/drbdadm dstate all;
cmw-status -v siass | grep OpenSAF -A2;
</rawinput>
<rawoutput>
SC-1
Primary
Connecting
UpToDate/DUnknown
safSISU=safSu=SC-1\,safSg=2N\,safApp=OpenSAF,safSi=SC-2N,safApp=OpenSAF
HARState=ACTIVE(1)
HARReadinessState=READY_FOR_ASSIGNMENT(1)
--
safSISU=safSu=SC-1\,safSg=NoRed\,safApp=OpenSAF,safSi=NoRed1,safApp=OpenSAF
HARState=ACTIVE(1)
HARReadinessState=READY_FOR_ASSIGNMENT(1)
</rawoutput>
</Check>
<Check name="Controller status - SC-2">
<verdict>ERROR</verdict>
<reason>
ERROR: Command failed: cmw-hostname-get SC-2
error - object or attribute does not exist
<11>Sep 10 17:08:52 CMW: ERROR (cmw-hostname-get): Host for amf-node [SC-2] not found

```




```

</reason>
<recommendedAction>Script update is needed or system is broken</recommendedAction>
<description>Verify Controller Status</description>
<rawinput>cmw-hostname-get SC-2;</rawinput>
<rawoutput>
  error - object or attribute does not exist
  <11>Sep 10 17:08:52 CMW: ERROR (cmw-hostname-get): Host for amf-node [SC-2] not found
</rawoutput>
</Check>
<Check name="CSCF configured DNS Server(s)">
  <verdict>OK</verdict>
  <reason>
    INFO: local/source: IPv4 address 0.0.0.0
    OK: DNS Server 192.168.10.50 via 0.0.0.0 is reachable
  </reason>
  <recommendedAction>No Action</recommendedAction>
  <description>Check Availability of DNS Servers</description>
  <rawinput>
    prompt @COMCLI;
    show;
    ManagedElement=jambala;
    CscfFunction=1,DNS-Application=DNS;
    show dnsLocalAddress;
    show dnsServerEntry;
    cdsv-get-node-state -s;
    cdsv-print-node -v;
    cdsv-get-node-state -s;
    cdsv-print-node -v;
    cmw-hostname-get SC-1;
    traceroute -I -s 0.0.0.0 192.168.10.50;
  </rawinput>
  <rawoutput>
    ManagedElement=jambala

    dnsLocalAddress    "0.0.0.0"
    dnsServerEntry      "0:192.168.10.50:5353"
    Result from [.cdsv.director]:
    safAmfNode=SC-1,safAmfCluster=myAmfCluster is Started

    Result from [.cdsv.director]:
    Node[0x214a600] id: 0 name: safAmfNode=SC-1,safAmfCluster=myAmfCluster state: \⇒
    2 (Started) flags: 00000001 address: 01001001
    UserApp states: [ 0: 00000003 1: 00000003 ]
    Hosted pools: [ dut.rtid.1105883 dut.rtid.1107008 dut.rtid.1107094 dut.rtid.1108366 \⇒
    <...>
    SC-1
    traceroute to 192.168.10.50 (192.168.10.50), 30 hops max, 60 byte packets
      1 192.168.10.97 (192.168.10.97)  0.541 ms  0.438 ms  0.924 ms
      2 192.168.10.50 (192.168.10.50)  1.373 ms  1.381 ms  1.327 ms
  </rawoutput>
</Check>
<Check name="SIP Interface Status">
  <verdict>OK</verdict>
  <reason>OK: All SIP interfaces statuses are OK</reason>
  <recommendedAction>No Action</recommendedAction>
  <description>Check Status of SIP Interfaces</description>
  <rawinput>
    prompt @COMCLI;
    show;
    ManagedElement=jambala;
    CscfFunction=1,CSCF-Application=CSCF,CscfNwIfContainer=0;
    show EcscfNwIfs=0;
    show EcscfNwIfs=0,EcscfNetworkInterface=TCP:192.168.10.205:8050,ecscfNetworkInterfaceStatus;
    show EcscfNwIfs=0,EcscfNetworkInterface=UDP:192.168.10.205:8050,ecscfNetworkInterfaceStatus;
    show IcscfNwIfs=0;
    <...>
  </rawinput>
  <rawoutput>
    ManagedElement=jambala
    EcscfNwIfs=0
      EcscfNetworkInterface=TCP:192.168.10.205:8050
      EcscfNetworkInterface=UDP:192.168.10.205:8050
    ecscfNetworkInterfaceStatus=OK
    ecscfNetworkInterfaceStatus=OK
    IcscfNwIfs=0
    <...>
  </rawoutput>

```



```
</rawoutput>
</Check>
<Check name="CSCF Network Connectivity">
  <verdict>OK</verdict>
  <reason>
    OK: CDSV Connections of the control server
    OK: CDSV Connections of the distribution server
  </reason>
  <recommendedAction>No Action</recommendedAction>
  <description>Monitor Network Connectivity</description>
  <rawinput>
    cmw-status -v node;
    clurun.sh -c ctrlsrv_print_conn;
    clurun.sh -c distsrv_print_conn;
  </rawinput>
  <rawoutput>
    safAmfNode=SC-1,safAmfCluster=myAmfCluster
    AdminState=UNLOCKED(1)
    OperState=ENABLED(1)
    Result from [.cdsv.director]:

    DBSv
      Server port:          <1.1.1:319307272>
      Client port:         <1.1.1:3875825354>
      Connection status:   established
      Client version:      1
      Server version:      1
      Common version:      1
      Queued messages:     0
    <...>
  </rawoutput>
</Check>
<Check name="PM Indicators">
  <verdict>OK</verdict>
  <reason>
    INFO: pm report is generated at /storage/no-backup/vcscf_CXP9034345/healthcheck/\⇒
    reports/PM_INDICATORS_Report_jambala_2018-09-10_17_08_22.csv
    INFO: pm report is generated at /storage/no-backup/vcscf_CXP9034345/healthcheck/\⇒
    reports/PM_INDICATORS_Report_jambala_2018-09-10_17_08_22.html
  </reason>
  <recommendedAction>No Action</recommendedAction>
  <description>Check PM Indicators</description>
  <rawinput>sudo -n ls -ltr /cluster/storage/no-backup/com-apr9010443/\⇒
  PerformanceManagementReportFiles//;</rawinput>
  <rawoutput>
    A20180910.1010+0200-1015+0200_jambala.xml
    A20180910.1015+0200-1020+0200_jambala.xml
    A20180910.1020+0200-1025+0200_jambala.xml
  <...>
</rawoutput>
</Check>
<Check name="CSCF Processor Outage">
  <verdict>OK</verdict>
  <reason>
    OK: DBSv - cluster state: Idle
    OK: LPMSv - cluster state: Idle
  </reason>
  <recommendedAction>No Action</recommendedAction>
  <description>Verify Processor Status</description>
  <rawinput>cdsv-get-user-state;</rawinput>
  <rawoutput>
    Result from [.cdsv.user.director.DBSv]:
    DBSv - cluster state: Idle
    Agent server port: <1.1.1:559892216>, listening on 92345,99
    Agents (count: 1):
    Agent[0xe7b840] node: safAmfNode=SC-1,safAmfCluster=myAmfCluster, port: <1.1.1:2275422080>
    Idle: yes, Halting: no
    Number of operation states: 0

    Operations (count: 0):
  <...>
</rawoutput>
</Check>
<Check name="Diameter Port Listening">
  <verdict>OK</verdict>
  <reason>OK: Diameter ports are ok</reason>
```



```

<recommendedAction>No Action</recommendedAction>
<description>Verify Diameter Stack Status</description>
<rawinput>
  cdsv-get-node-state -s;
  cdsv-print-node -v;
  cdsv-get-node-state -s;
  cdsv-print-node -v;
  cmw-hostname-get SC-1;
  prompt @COMCLI;
  show -v;
  ManagedElement=jambala;
  CscfFunction=1,DIA-CFG-Application=DIA;
  DIA-CFG-StackContainer=CSCFCX;
  DIA-CFG-OwnNodeConfig=CSCFCX;
  show portNr;
  show ipAddressList;
  show sctpAddressesList;
  show enabled;
  show transportLayerType;
  up;
  up;
  <...>
  netstat -anT | grep 192.168.10.203:3875;
  exit;
</rawinput>
<rawoutput>
  Result from [.cdsv.director]:
  safAmfNode=SC-1,safAmfCluster=myAmfCluster is Started

  Result from [.cdsv.director]:
  Node[0x214a600] id: 0 name: safAmfNode=SC-1,safAmfCluster=myAmfCluster state: \=>
  2 (Started) flags: 00000001 address: 01001001
  UserApp states: [ 0: 00000003 1: 00000003 ]
  Hosted pools: [ dut.rtid.1105883 dut.rtid.1107008 dut.rtid.1107094 dut.rtid.1108366 \=>
  dut.rtid.1124878 dut.rtid.1124884 dut.rtid.1124885 dut.rtid.1124915 dut.rtid.1127460 \=>
  dut.rtid.1127463 dut.rtid.1127502 dut.rtid.1127641\=>
  <...>
</rawoutput>
</Check>
<endTime>2018-09-10T17:09:55+02:00</endTime>
<duration>01:32</duration>
<finalVerdict>ERROR</finalVerdict>
</HealthCheckReport>

```

Example 2 Example of Automatic Health Check Result





9 File Management

The Health Check report files are exposed by File Management in the following file group structure:

- FileGroup=Cscf
 - FileGroup=HealthCheck
 - FileGroup=ReportFiles

For more information on file groups, see [Handling Files](#).