ERICSSON

ENM Operators Guide

User Guide

1/1553-AOM 901 151 Uen CN

Copyright

© Ericsson AB 2017, 2018, 2019, 2020. 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	**ENM Operators Guide**	**1**
2	**Configuration Management Overview**	**2**
2.1	Add a Node in ENM	2
2.2	Enable CM Supervision and Node Synchronization in ENM	4
2.3	Node Version and the ossModelIdentity	13
2.4	Node Version Matching for MINI-LINK Family Nodes	14
2.5	Treat-As	14
2.6	Treat-As for MINI-LINK Family Nodes	17
2.7	Replication Activities for Newly Added Nodes	17
3	**Network Element Integration Configuration Overview**	**18**
3.1	DHCP Configuration for Network Element Autointegration	18
3.2	DNS Configuration for Network Element Integration	35
3.3	NTP Configuration for Network Element Integration	42
3.4	Trusted NTP Server Configuration	46
3.5	E2E Offline Enrollment Procedure for Baseband Radio Node	62
3.6	User Exposure READ ONLY for FM Model	71
3.7	Online Certificate Enrollment on RadioNode	72
3.8	Online Certificate Enrollment on Router 6000 Family	81
3.9	E2E Offline Certificate Enrollment on Router 6000 Family	87
3.10	LDAP Configuration for Router 6000 Family	101
3.11	Configure LDAP on BSC Node in SSH	104
3.12	Launch EM with SSO for Generic ESA	109
3.13	Configure HLR/MSC with TLS	110
3.14	Move BSC Node from SSH to TLS	114
3.15	Move HLR Node from SSH to TLS	125
3.16	Move MSC/HLR Variants from SSH to TLS	132
3.17	Install Certificate to Launch SBG(IS) Element Manager	140
3.18	Command to Set Release Attribute for ENIQ	142
3.19	D-SC Dashboard Launch	143
3.20	Install Certificate to Launch MSC-BC-IS Element Manager	143
3.21	Install Certificate to Launch HLR-FE (IS) Element Manager	144
3.22	Overview of GSM and AXE Node Support	144

3.23	PM Job Name and File Path for CUDB/vCUDB Nodes	149
3.24	LDAP Support for EIR-FE/vEIR-FE (ESA-Based Node)	150
3.25	Auto-Configure SNMP Security for Baseband RadioNode and VNF	151
3.26	Performance Management for GenericESA-based Nodes	155
3.27	Accessing Element Manager GUI for SCEF Node	155
3.28	Online Certificate Enrollment on Fronthaul 6020	156
4	**Software and Hardware Manager Overview**	**159**
4.1	Node Software Upgrade using Software and Hardware Manager	161
4.2	Node Software Upgrade Using Automated Software Upgrade Flow	175
4.3	Cancel Job Using Software and Hardware Manager	189
4.4	Create and Manage Node Backups Using SHM Backup Job Functionality	190
4.5	Restoring a Node Backup in Software and Hardware Manager	195
4.6	Housekeeping Backups on Nodes Using Software Hardware Manager	199
4.7	Install License Key in Software and Hardware Manager	202
4.8	Upgrade Package State in Software Inventory	204
4.9	Delete Upgrade Packages Using Software and Hardware Manager	207
4.10	Onboard and Delete VNF Software Packages	209
4.11	Automatic Import of Software Packages from Network Element SoftwareStore (CAS-C)	211
4.12	SHM Load Controller	213
4.13	Instantaneous Licensing Functionality - LKF Request Job	216
5	**Node Health Check Overview**	**219**
5.1	Create Node Health Check Report	220
5.2	Create Health Check Profile	222
6	**Fault Management Overview**	**223**
6.1	Fault Management Concepts	224
6.2	Fault Management Flows	235
6.3	Fault Management Processing	236
6.4	Fault Management Formats	244
7	**Automatic Alarm Handling (FMX) Overview**	**253**
7.1	OSS-RC FMX vs ENM FMX: Compatibility and Delta functionality notes	253

8	**Scripting Support Overview**	**260**
8.1	Log on to a Scripting KVM	260
8.2	Running Scripts	261
8.3	Reschedule Failover Cron Services	261
8.4	Load Balancing for AMOS and General Scripting VMs	263
8.5	Automated House Keeping on All Scripting VMs	263
8.6	Copy and Paste Long Lines in AMOS and General Scripting Shell Terminal	264
9	**Automatic Link Discovery (ALD)**	**265**
9.1	Automatic Discovery for Radio Link	265
9.2	Automatic Discovery for Ethernet Link	266

1 ENM Operators Guide

> The Operators Guide contains general application information to provide an overview of the context in which specific applications operate.

For all applications, detailed online help is available on the User Interface. It can be accessed by selecting the `App Help` option in the `Help Menu`.

2 Configuration Management Overview

Configuration Management (CM) in ENM allows the following tasks:

— Adding a Node to ENM.

— Enabling CM Supervision.

It also introduces the following concepts:

— Model Identity.

— Treat-as.

Basic CM Workflow

From a CM point of view the order of the following workflow is critical:

1. Add the Node to ENM.
2. Enable CM Supervision.
3. Use other CM features such as updating attributes on the node.

2.1 Add a Node in ENM

> Adding a node to ENM defines a representation of the node inside ENM. ENM is then used to supervise or manage that node.

It is used to collect Fault Management (FM) alarms and Performance Management (PM) Report Output Period (ROP) files from the node. It is used for Configuration Management tasks on the node. A node can be added to ENM. It does not yet have to exist in the network, or have yet been powered on.

Nodes can be added to ENM using ENM CLI commands, the Add Node Web application, or the Auto-Provisioning application. The process is similar for all node types supported by ENM. There are some differences depending on the platform the node is developed on. This is detailed in the *Add a Node Tutorial* of the ENM CLI online help. The node is added to ENM by creating the `NetworkElement` Managed Object (MO) and the `ConnectivityInformation` MO. These must be created in this order, and automatically creates additional MOs.

NetworkElement Managed Object

There is one `NetworkElement` MO in the system for every node. The `NetworkElement` MO does not exist on the node. It is used in ENM to represent all node types in a generic way.

The `NetworkElement` MO has several child MOs. One of these must be created by the user to complete the addition of the node. Other MOs are created by the system, and store useful information for that node.

SubNetwork Managed Object

`SubNetwork` MOs can be used for logical grouping of nodes within ENM. The `SubNetwork` MOs do not exist on any of the nodes. ENM allows a node to be assigned to a SubNetwork when the node is being added to the system. SubNetworks can contain one or more child SubNetworks.

For example, a node added under a SubNetwork named Stockholm, which in turn has a parent SubNetwork named Sweden. Many nodes can be grouped underneath a single SubNetwork. This MO is optional, and is automatically created as part of creating the `NetworkElement` MO if needed.

MeContext and ManagedElement Managed Objects

The `MeContext` MO is used in ENM to distinguish between nodes that cannot be distinguished from one another in the management system. This is the case for nodes based on CPP. Here the root MO on every node has an FDN of ManagedElement=1, which does not uniquely identify the node. Therefore, the `MeContext` MO can be inserted into the FDNs in ENM directly above the `ManagedElement` MO. This allows each node to have a unique identifier.

For example, two nodes called MeContext=Stockholm1 and MeContext=Stockholm2. Their `ManagedElement` MOs have unique FDNs. These are MeContext=Stockholm1, ManagedElement=1, and MeContext=Stockholm2, ManagedElement=1. They can now be distinguished from one another in ENM.

For nodes based on CPP, the MeContext is automatically created when the `NetworkElement` MO is created during the addition of the node. For other node types, the creation of the `MeContext` MO is optional. It depends on the specifics of the NetworkElement `create` command as described in the online help.

ConnectivityInformation Managed Object

The `ConnectivityInformation` MO contains connectivity information. This includes the `ipAddress` and `port`. This MO is the second MO that must be explicitly created to complete the process of adding the node to ENM. It is created as a child MO of the NetworkElement. Its creation results in the automatic creation of some system created MOs. These are needed to manage the node. These MOs are used for supervising the node, and other functions described by various other use cases and applications:

- `CmNodeHeartbeatSupervision`
- `SecurityFunction`
- `FmFunction`
- `FmAlarmSupervision`
- `InventorySupervision`
- `CmFunction`
- `PMICScannerInfo`
- `PmFunction`
- `SHMFunction`

There is a specific `ConnectivityInformation` MO type created depending on the platform type of the node.

After adding the node to ENM, the security credentials must be set. This is necessary for some node communications. ENM can now be used to supervise or manage the node. For example, CM Supervision can be enabled.

The detailed procedure for adding a node to ENM is detailed in the *Add a Node Tutorial* in the online help for the ENM CLI.

See "*Configure the Connectivity Information SNMP Parameters*" in *Fault Management Administration Tasks* section in *ENM Monitoring System Administrator Guide*.

Note: ESA based network elements that are added with `neType GenericESA` support virtualization.

2.2 Enable CM Supervision and Node Synchronization in ENM

> Enabling CM Supervision means that from a CM point of view, ENM is now supervising the node.

ENM can do the following:

- Create MOs on the node.
- Read MOs and or attributes from the node.
- Update attributes on the node.

— Delete MOs from the node.

— Run actions for various MOs on the node.

If the CM Supervision is disabled for a node, none of the above operations involving interactions with the node are allowed.

In ENM all the CM data is made available, and presented exactly as it exists on the nodes. To facilitate this, ENM maintains an internal copy of part of the configuration data of the node. This information is stored in an internal ENM database. Each node is represented by the hierarchical MO structure from the node. This is stored under a node root MO, which is either an `MeContext` or a `ManagedElement` MO.

This enables use cases that span many nodes to be executed in a performance manner. It allows applications to have model-based validation to be done without contacting the nodes. To enable this, node models are mirrored unchanged. Attributes that do not issue notifications are not be stored in this way. Instead they are fetched on demand from the node.

ENM reads an initial copy of the node CM data when CM Supervision of the node is turned on. This is referred to as performing a full CM sync of the node. It synchronizes the configuration data on the node with the ENM internal mirror of that data at that point in time.

It uses these notifications to keep the copy up-to-date. When configuration data on the node changes, ENM processes the resulting notification to ensure that the internal mirror of the data is kept up-to-date.

ENM maintains an internal 'heartbeat' for each node. It checks the node to make sure it is still subscribed to receive notifications from it. It does this by periodically checking the status of the notification subscription on the node. Seven minutes for ERBS and ECIM nodes, except MSRBS-V1 node that accept values in the range 240–280. In there is any problem, it attempts to resync the node and resubscribe for notifications.

`HeartbeatInterval` has threshold values, both minimum and maximum, managed by capabilities in the node common model. They are checked in the heartbeat handler.

For every node, the minimum and the maximum value can be defined by capabilities.

The minimum value can be equal to or greater than 240. The maximum value is not defined, except on pico node (MSRBS_V1) where the maximum value is set to 280.

A value greater than 280 set by operator, after the heartbeat handler check, is reset to 280 (maximum value allowed for this type of node) while a value smaller than 240 is reset to 240.

BSC Nodes, with GEP1 HW/boards, takes longer time for CM Synchronization. To avoid duplicate or multiple synchronization requests, increase the CM heartbeatInterval period for BSC nodes with GEP1 HW only. Using the following command, increase the CM HB interval period to 25 minutes:

```
cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 heartbeatInter →
val = 1500
```

If everything is working properly, the node is considered 'Synchronized'. This means that ENM has an up-to-date copy of the node CM data. If something goes wrong with the heartbeat check, the node is considered `unsynchronized`.

A node can be `unsynchronized` for various reasons. Some reasons include:

— Connectivity to the node is lost because of network issues.

— The node was restarted by the user.

— The node was restarted by the system during a software upgrade.

— The node invalidated the notification subscription.

  For example, because of notification overload on the node.

— CM Supervision is disabled in ENM.

The synchronization status of a node is checked by looking at the `syncStatus` attribute on the CmFunction MO, as follows:

```
>> cmedit get <nodeName> CmFunction.syncStatus
```

For command details, see the *ENM CLI online help*.

This could be extended to, for example, search for all nodes that are currently `unsynchronized`:

```
>> cmedit get * CmFunction.syncStatus==UNSYNCHRONIZED
```

Sometimes, when supervision is enabled, you want to force a full sync of a node. This is done by running the explicit sync action command:

```
>> cmedit action NetworkElement=<nodeName>,CmFunction=1 sync
```

An example of a scenario to execute a full sync on a node would be to populate a new attribute after an ENM upgrade. The proposed procedure to populate the attribute radioAccessTechnology is as follows:

Example:

Use ENM CLI to check what RadioNodes are not migrated to the version where attribute radioAccessTechnology is populated. These RadioNodes can be migrated by being synchronized individually.

1. Execute **cmedit** command to get the radioAccessTechnology attribute value on the NetworkElement for all RadioNodes on the ENM system:

   ```
   >> cmedit get * NetworkElement.radioAccessTechnology --netype=RadioNode
   ```

2. Analyze the output:

   The RadioNodes that have not been migrated show the radioAccessTechnology attribute value as an empty list.

   ```
   >> FDN : NetworkElement=<nodeName> radioAccessTechnology : []
   ```

3. For each RadioNode required for migration, pick out the node name, and perform a synchronization on this node.

   Note: Sync nodes individually.

   ```
   >> cmedit action NetworkElement=<nodeName>,CmFunction=1 sync
   ```

4. Execute **cmedit** command on this node to verify that migration has been performed, and the attribute radioAccessTechnology has been populated.

   Input:

   ```
   >> cmedit get <nodeName> NetworkElement.radioAccessTechnology
   ```

   Output:

   ```
   >> FDN : NetworkElement=<nodeName> radioAccessTechnology : [4G, 5G]
   ```

5. To resync all Radionodes after upgrade, execute the following command:

   ```
   >> cmedit action * CmFunction sync -neType=RadioNode
   ```

   Resyncs using partial node names can be performed:

   - ```
     >> cmedit action <partialNodeName>* CmFunction sync -neType=RadioNode
     ```

     Example:

     ```
     cmedit action LTE* CmFunction sync -neType=RadioNode
     ```

   - ```
     >> cmedit action *<partialNodeName> CmFunction sync -neType=RadioNode
     ```

Example:

```
cmedit action *002 CmFunction sync -neType=RadioNode
```

- `>> cmedit action *<partialNodeName>* CmFunction sync -neType=RadioNode`

  Example:

  ```
  cmedit action *ERBS* CmFunction sync -neType=RadioNode
  ```

If a Node has become `UNSYNCHRONIZED`, identify at what point synchronization was lost. The `lostSynchronization` attribute on the `CmFunction` MO contains the date and time that synchronization was lost. As with any modeled information in ENM, to retrieve a description of the attributes on the `CmFunction` MO use the `describe` command. For example:

```
>> cmedit describe CmFunction.*
```

If the node becomes `UNSYNCHRONIZED`, ENM tries to recover this automatically. If necessary, it performs a `delta sync`. This means it retrieves information missing since it was last synchronized. This is a faster and more efficient process than a full sync of the node. If a `delta sync` is not appropriate, then a full sync of the node is performed.

ENM also allows the configuration data on the node to be modified. This is possible by allowing the application to create or delete MOs, or modify attribute values in the internal ENM database. These changes are then written to the node.

— For MINI-LINK Indoor Network Elements, the CM model is only synchronized to support SHM functionality, CM operations are not supported.

— For CISCO-ASR9000, CISCO-ASR900, and JUNIPER-MX Network Elements, the CM model is only synchronized to support inventory data management needed for Link Management.

— For MSC/HLR, there is no CM Sync support (CM is over WinFIOL). Node status `UNSYNCHRONIZED` is valid and correct for these nodes.

2.2.1 Node Credentials Update on Router6000 Series Nodes

The **CM Supervision** of `Router6000` series nodes added in ENM must be disabled before updating the node credentials on the node and in ENM.

2.2.2 Node Credentials Update on CCRC, CCDM, and CCSM Nodes

Before updating the node credentials on the node and in ENM, CM Supervision for CCRC, CCDM, and CCSM nodes added in ENM must be disabled.

2.2.3 Node Credentials Update on RNC EVO 8200, Media Gateway, and RNC EVO 8300 Nodes

This section describes how to update node credentials on RNC EVO 8200, Media Gateway (MGW, MRS and so on), and RNC EVO 8300 nodes.

Whenever the node is being added to the ENM, create node credentials in the ENM using `secadm` command. Before creating the credentials, test them manually and then update in ENM.

RNC EVO 8200 or Media Gateway Nodes

Following is the procedure to create credentials in ENM:

1. Connect to the node using the username and password provided by the node team. Check whether `passwd` file is present or not as follows:

   ```
   [PNTC-5356@ieatlms4898 ~]$ ssh mrs1@131.160.83.13
   mrs1@131.160.83.13's password:
   Welcome to OSE Shell OSE5.8.
   $ ls -ltr  /c/security/passwd
   -rw-rw-rw-   0       211  22 Oct 2020 10:12:56 /c/security/passwd
   ```

   If the `passwd` file is present, then proceed with *Step 2*, otherwise proceed with *Step 3*.

2. Execute the COLI commands and try to connect through SFTP with the provided credentials on the node as follows:

   a. COLI commands.

   ```
   [PNTC-5356@ieatlms4898 ~]$ ssh mrs1@131.160.83.13
   mrs1@131.160.83.13's password:
   Welcome to OSE Shell OSE5.8.
   $ secmode -s

   Security configuration settings:
   Access method           Current security mode
   ------------------------------------------------------------------
   TelnetFtpServers        secure, node internal Telnet and FTP servers ar →
   e OFF.
   TargetMonitor           secure, node Target Monitor is OFF.
   DbgServerUdpLnh         secure, Debug server and UDP Linkhandler are OF →
   F.
   FileXferClient          secure, node internal file transfer client uses →
    SFTP.
   CorbaSecurity           secure, corba security is ON.
   WebServer               unsecure, HTTP server is ON.
   UserAAPolicy            basic, One Node Password is used in SL1 and SL2
   OperationalSecLevel     level 2
   ConfiguredSecLevel      level 2

   --End settings----------------------------------------------
   $ exit
   ```

   b. Check SFTP connection.

```
[PNTC-5356@ieatlms4898 ~]$ sftp mrs1@131.160.83.13
Connecting to 131.160.83.13...
```

```
mrs1@131.160.83.13's password:
sftp> ls
>                                                    configuration           →
                                cv.bak
cv.ptr                                               deletetmp.txt           →
                                loadmodules
logfiles                                             mrs11-oam-65cc647ca1f74 →
b7b-1583333057068.p12           rollback.cnt
rollback.ini                                         rollback.lst            →
                                rollbacklst.bak
systemfiles                                          te                      →
                                tmp
usr
sftp> exit
```

c. If the COLI commands execution and SFTP connection are successful, proceed with *Step 4* and if it fails, then proceed with *Step 3*.

3. Execute the following steps to update the node password:

a. Check whether the SSH and SFTP connections fail with the following error:

```
COLI COMMAND :

[PNTC-5356@ieatlms4898 ~]$ ssh  mrs1@131.160.83.13
mrs1@131.160.83.13's password:
Welcome to OSE Shell OSE5.8.
$ secmode -l 2
illegal command 'secmode', status: Command not allowed. Node password m  →
ust be set using passwd command!
$ exit

SFTP CONNECTION :

[PNTC-5356@ieatlms4898 ~]$ sftp mrs1@131.160.83.13
Connecting to 131.160.83.13...
mrs1@131.160.83.13's password:
subsystem request failed on channel 0
```

b. Set a new password to the node as follows using `passwd` command.

```
[PNTC-5356@ieatlms4898 ~]$ ssh  mrs1@131.160.83.13
mrs1@131.160.83.13's password:
Welcome to OSE Shell OSE5.8.
$ passwd
Enter new password:
Re-enter new password:
$ exit
```

c. Check the COLI commands and SFTP connection with the newly updated credentials.

4. Create the credentials in ENM using the `Secadm` command with updated working credentials.

```
secadm credentials create --rootusername <username> --rootuserpassword "<nod →
e-password>" --secureusername <username> --secureuserpassword "<node-passwor →
```

```
d>" --normalusername <username> --normaluserpassword "<node-password>" -n  < →
node-name>
```

RNC EVO 8300 Nodes

Following is the procedure to create credentials in ENM:

Steps

1. Connect to the node using the username and password provided by the node team. Check whether COLI commands and SFTP connections are working or not as follows:

    a. COLI command

    ```
    [root@svc-1-mscmce cloud-user]# ssh  user@10.74.58.20
    user@10.74.58.20's password: Ericsson_rnc20
    Welcome to Shell on Evo8300
    $ secmode -s

    Security configuration settings:
    Access method           Current security mode
    ------------------------------------------------------------
    TelnetFtpServers        secure, node internal Telnet and FTP server →
    s are OFF.
    TargetMonitor           secure, node Target Monitor is OFF.
    DbgServerUdpLnh         secure, Debug server and UDP Linkhandler ar →
    e OFF.
    FileXferClient          secure, node internal file transfer client  →
    uses SFTP.
    CorbaSecurity           secure, corba security is ON.
    WebServer               unsecure, HTTP server is ON.
    UserAAPolicy            basic, Node Users with user specific passwo →
    rd are used in SL1 and SL2
    OperationalSecLevel     level 2
    ConfiguredSecLevel      level 2

    --End settings--------------------------------------------------
    ```

    b. Check SFTP connection

    ```
    [root@svc-1-mscmce cloud-user]# sftp user@10.74.58.20
    Connecting to 10.74.58.20...
    user@10.74.58.20's password:
    sftp> ls
    configuration cv.bak cv.ptr loadmodules lost+found rollback.cnt rol →
    lback.ini rollback.lst rollbacklst.bak
    systemfiles tftp usr
    sftp>
    ```

    c. If the execution of COLI command and SFTP connection is successful, then proceed with *Step 3* and if it fails, then proceed with *Step 2*.

2. Execute the following steps to update the password:

    a. Check whether the SSH and SFTP connections fail with the following error.

    ```
    COLI COMMAND :
    [root@svc-1-mscmce cloud-user]# ssh  user1@10.74.58.20
    ```

```
user1@10.74.58.20's password: <password>
You are required to change your password immediately
Welcome to Shell on Evo8300
WARNING: Your password has expired.
You must change your password now and login again!


SFTP CONNECTION :
[root@svc-1-mscmce cloud-user]# sftp dummy@10.74.58.20
Connecting to 10.74.58.20...
dummy@10.74.58.20's password:
Connection closed
```

b. Update the password while logging in with SSH with an automatic request for change of password. Check the COLI commands and SFTP connection after the credentials are updated.

```
[root@svc-1-mscmce cloud-user]# ssh  user1@10.74.58.20
user1@10.74.58.20's password: <login-password>
You are required to change your password immediately
Welcome to Shell on Evo8300
WARNING: Your password has expired.
You must change your password now and login again!
Enter login password: <login-password>
Changing password for user1.
Enter new password: <desired-password>
Retype new password: <desired-password>
passwd: all authentication tokens are updated successfully
Connection to 10.74.58.20 closed.

Check the credentials :
[root@svc-1-mscmce cloud-user]# ssh  user1@10.74.58.20
user1@10.74.58.20's password: <desired-password>
Welcome to Shell on Evo8300
$ secmode -s

Security configuration settings:
Access method           Current security mode
------------------------------------------------------------
TelnetFtpServers        secure, node internal Telnet and FTP server →
s are OFF.
TargetMonitor           secure, node Target Monitor is OFF.
DbgServerUdpLnh         secure, Debug server and UDP Linkhandler ar →
e OFF.
FileXferClient          secure, node internal file transfer client  →
uses SFTP.
CorbaSecurity           secure, corba security is ON.
WebServer               unsecure, HTTP server is ON.
UserAAPolicy            basic, Node Users with user specific passwo →
rd are used in SL1 and SL2
OperationalSecLevel     level 2
ConfiguredSecLevel      level 2

--End settings----------------------------------------------

Sftp Connection :
[root@svc-1-mscmce cloud-user]# sftp user@10.74.58.20
Connecting to 10.74.58.20...
user@10.74.58.20's password:
sftp> ls
configuration cv.bak cv.ptr loadmodules lost+found rollback.cnt rol →
lback.ini rollback.lst rollbacklst.bak
systemfiles tftp usr
sftp>
```

3. Create the credentials in ENM using the `Secadm` command with updated working credentials.

```
secadm credentials create --rootusername <username> --rootuserpassword "<pas →
sword>" --secureusername <username> --secureuserpassword "<password>" --norm →
alusername <username> --normaluserpassword "<password>" -n  <node-name>
```

2.3 Node Version and the ossModelIdentity

> Historically, from a management perspective, node versions were judged atomically, as they published a single data-model.

For example, the version of nodes based on CPP like an ERBS was easily recognizable by its MIM Version.

However, ECIM nodes use Component Based Architecture. A node release overall model is composed of a set of independent data models. These are known as fragments. Each fragment has its own identity. This means that for ECIM nodes the concept of a single version is less obvious.

To address this, ENM has introduced the concept of a Model Identity. It is used internally in ENM to identify node versions. There are two fields to note:

ossModelIdentity:
represents the version that ENM sees the node as

nodeModelIdentity:
represents the actual version of the node

These two fields are stored as attributes on the `NetworkElement` MO. They typically contain the same value as each other except if **`Treat-as`**.

They can be viewed, managed, and searched for in the ENM CLI in the same way as any other modeled attribute of any MOs. Both the `nodeModelIdentity` and the `ossModelIdentity` are populated by ENM when CM Supervision is enabled. For CPP and ECIM-based nodes, ENM auto calculates the `ossModelIdentity` with an algorithm that searches for the best-match between the models on the node and those present in ENM. It then decides the version to `treat-as` the node.

For all the other NetworkElement types, the `ossModelIdentity` is normally populated by ENM when CM Supervision is enabled. It must only be modified by the system. It must be explicitly set in the following scenarios:

- If the node is used in ENM without enabling CM Supervision and waiting for the node to Sync, then the `ossModelIdentity` must be supplied.
- There are two scenarios involving Treat-as when the `ossModelIdentity` must be supplied. These are described in Treat-As on page 14.

The possible values for the `ossModelIdentity` can be retrieved as follows:

```
>> cmedit describe -netype <neType>
```

This is described in the *ENM CLI online help*.

For example, for a Router6672 this would be the following:

```
>> cmedit describe --netype Router6672
```

The `ossModelIdentity` represents the version that the node is to be managed as. If multiple Model Identities are available for the given MIM Version, any can be chosen, as they are all functionally equivalent.

2.4 Node Version Matching for MINI-LINK Family Nodes

> Automatically selecting the model identity for MINI-LINK nodes is supported, but there are some differences.

MINI-LINK software versions are identified with a single comparable revision number. The models in ENM refer to a product identity and a revision number. To find a suitable match, the product identity read from the node must be equal to the product identity supplied with the model, and the revision from the node is equal or higher than the one from the model. The highest revision of such models is considered the best match to the node, and the `ossModelIdentity` attribute is populated with the identity of that model. The `nodeModelIdentity` attribute cannot be calculated, it is only populated when the node revision is considered an exact match for the model.

2.5 Treat-As

> Treat-As means that ENM can manage a node as a different version to the actual version of that node. An unknown node version is managed (or treated) as a known node version.

Treat-As can be required where nodes are upgraded to an unsupported version, or added to ENM with an unsupported version. It must be used when the node is on a later version than the node model in ENM.

ENM Treat-As must be used for backwardly compatible changes in the node. Do not use it to manage a node where there are backwardly incompatible changes between the treat-as version, and the actual node version.

If there are backwardly incompatible changes between the treat-as version and the node version, then it is not possible to synchronize ENM with the node, and the sync operation fails.

The `nodeModelIdentity` provides information on the actual node version. The `ossModelIdentity` provides information on the version it is being managed with by ENM. If both values are the same, the node is fully supported, otherwise it is in treat-as mode.

2.5.1 Automatic Treat-As (CPP and ECIM-Based Nodes)

For CPP and ECIM-based nodes, the `nodeModelIdentity` and the `ossModelIdentity` are populated by ENM when CM Supervision is enabled. If a node is upgraded to a supported version, ENM updates the values of both `nodeModelIdentity` and `ossModelIdentity`. If a node is upgraded to an unsupported version, the `nodeModelIdentity` is updated according to its actual node version, and the `ossModelIdentity` is updated, if needed, verifying it is set to the most appropriate treat-as version. This is achieved by an automated algorithm which looks for the best matching between the models on the node and those known by ENM. This algorithm is described in Manual Treat-As (All the Other Network Element Types) on page 15.

For CPP nodes, the model identity identifies the MIM version and the PmEvent model. The model identity of a CPP node of a MIM version supported in ENM, can result as unsupported if the version of `PmEvent` model is different from the one included in the model version supported by ENM. ECIM-based nodes use multiple MIMs, so the comparison has to work on all MIMs.

ENM automatically verifies the "best possible match" at the occurrence of the following events:

— At every full synchronization of the node, either manual or system-driven, using the algorithm described in Manual Treat-As (All the Other Network Element Types) on page 15;

— After an execution of the Release Independence Manager “Add Support” use case that adds new supported node versions;

— After an execution of the Release Independence Manager “Remove Support” use case that removes supported node versions;

— After an ENM Upgrade which can both add and remove some node versions.

If the `ossModelIdentity` cannot be automatically set by ENM, it is not possible to manage the node.

Nodes in "treat-as" are those that have the `ossModelIdentity` different from the `nodeModeIdentity`.

2.5.2 Manual Treat-As (All the Other Network Element Types)

For all the other Network Element types, if a node is upgraded to an unsupported version, the `nodeModelIdentity` is updated by ENM but the `ossModelIdentity` is not. The `ossModelIdentity` must be supplied in the following cases:

— If the node is being added to ENM in Treat-as mode.

— If the node is being upgraded to a version that ENM does not support, the `ossModelIdentity` is not updated, and the node is managed, or treated, as with the version it was previously managed as. Sometimes this is not

appropriate as there is a better match that fulfills the backwardly compatible criteria previously discussed. In this case, the `ossModelIdentity` must be updated to match this.

— If the PM Initiation and Collection is using such a node without or before enabling CM Supervision.

For the manual Treat-As case, consider the following example:

	Versions in ENM					Network element
	v.1	v.2	v.3	v.4	v.5	
MIM A	1.0.0	1.0.0	1.1.0	1.2.0	1.2.0	1.1.0
MIM B	2.1.0	2.1.1	2.1.1	2.1.1	3.0.0	2.1.1
MIM C	1.3.0	1.3.0	2.0.0	2.1.0	2.1.0	1.3.1

All cells in yellow are identified as having a MIM version higher than what the Network element has. This means that versions 3, 4, and 5 are discarded for selection.

The remaining versions of the individual MIMs are then compared. The green cell is found, indicating that the MIM version here is lower than in another version. Hence, version 1 is discarded for selection.

v.2 remains, so this is the best possible match for Treat-As.

If the `ossModelIdentity` is not set, it is not possible to manage the node. The value is set by using the following command in the ENM CLI:

```
cmedit set <network element FDN> ossModelIdentity=<value>
```

Note: This command is not intended to be used while the `syncStatus` of the node is in transition state (for example: PENDING, DELTA). Wait until the node is stable and then run the command.

If there are synchronization problems after setting `ossModelIdentity`, use the command:

```
cmedit action <network element FDN>,CmFunction=1 sync
```

2.5.3 Treat-As Support for BSC

BSC and APG models are bundled together as a single OSS model identity. ENM selects the equivalent OSS model identity for Treat-As support based on the revisions of both components. ENM Treat-As selects the supported OSS model version with a revision that is equal to, or lower than, the model version on NE.

For a description of the support for **BSC OSS Model Identities** see ENM Supported Network Elements, 3/1029-AOM 901 151.

2.5.4 Treat-As Support for Router6000 Nodes

`Router6672`, `Router6675`, `Router6274`, `Router6273`, and `Router6x71` nodes do not support `Automatic Treat-As`. Only `Manual Treat-As` is supported for these nodes.

2.5.5 Treat-As Support for CCRC, CCDM, and CCSM Nodes

CCRC, CCDM, and CCSM nodes do not support Automatic Treat-As. Only Manual Treat-As is supported for these nodes. OSS Model Identity must be supplied when adding the node to ENM.

2.6 Treat-As for MINI-LINK Family Nodes

> Although the user provides an **ossModelIdentity** before synchronizing the node, the **ossModelIdentity** is always updated if there is a better match available.

The highest such model identity is selected where the revision of the node is equal to or higher than the one supplied with the model. If the version of the node is above the model, we consider that situation as an automatic treat-as (for example, treating a maintenance release as the GA release preceding it). If there is no suitable match for the revision of the node, the user has to supply a model identity (for example, product identity does not match any models, or revision is older than the oldest supported), we consider this as manual treat-as.

2.7 Replication Activities for Newly Added Nodes

In GEO-R Deployments, Replication Activities are required on newly added network elements to connect to the Secondary ENM after switchover where there is a Primary Disaster.

See *Replication Activities* in the ENM Geographical Redundancy User Guide, 1553-CNA 403 3466 Uen.

3 Network Element Integration Configuration Overview

> ENM provides DHCP, DNS, and NTP services for the integration of network elements.

These services are available on the **ITServices** VM located on the SVC cluster. The following sections outline the steps involved in the configuration and setup of the available services.

3.1 DHCP Configuration for Network Element Autointegration

`Network Elements Autointegration` requires a Dynamic Host Configuration Protocol (DHCP) server. The following content gives information on how to enable and configure the `ITServices DHCP` server provided within ENM. The functionality described here covers both IPv4 and IPv6.

If a DHCP server is already available, follow the configuration instructions in *DHCP Configuration for Autointegration using Customer Infrastructure* in the *ENM Network Integration Guide*.

Prerequisites

— The user has access to the ENM CLI as an authorized user.

— This user has `DHCP_Operator`, `DHCP_Service_Operator`, or `DHCP_Administrator` role assigned.

  See *User Management* in the *ENM Identity and Access Management Programmers Guide* for user and role configuration in ENM.

Note: The DHCP service can change between hosts at `failover`, `upgrade`, or `rollback`. To ensure the availability of the DHCP relays, the relay agents in the network elements must be configured to forward DHCP requests to the master DHCP server.

The following tasks can be executed by `DHCP_Operator` user:

— View all networks.

— View client configuration.

— Validate networks and network ranges.

— View DHCP service status.

The previous and the following tasks can be executed by `DHCP_Service_Operator` user:

— Add and delete networks.

— Add ranges to networks.

— Delete ranges from networks.

— Update networks.

— Add and delete client configuration.

The previous and the following tasks can be executed by `DHCP_Administrator` user:

— Start DHCP service.

— Stop DHCP service.

— Configure DHCP service configuration.

3.1.1 DHCP Server Management using ENM Command Line Interface

3.1.1.1 Start DHCP Service

Before starting the DHCP service, the DHCP service must be disabled.

Command

To start the DHCP Service, use the following command:

```
dhcp start attributes [options]
```

Attributes

The following parameters are required to start the DHCP service with the selected IP version.

`--ipv4, -4, --ipv6, -6`	Specifies IP version.

Note: The start DHCP service command attributes must be in the following order:

```
dhcp start --ipv4|-4|--ipv6|-6 [option]
```

Options

The following parameters are optional:

`--interface`	Specifies DHCP server interface.
`--macfilter`	Specifies Ericsson MAC address filtering.

Note: When configuring Ericsson MAC address filtering:

- The Ericsson MAC address filtering configuration is only supported for DHCP4 service.
- With MAC filtering enabled, ENM DHCP v4 server only serves requests from Ericsson nodes with Ericsson MAC addresses. Requests from non-Ericsson nodes are ignored.

Examples

DHCP4

```
»dhcp start --ipv4 --interface eth1 --macfilter enable
DHCP over IP version 4 started successfully.
```

DHCP6

```
»dhcp start --ipv6 --interface eth2
DHCP over IP version 6 started successfully.
```

3.1.1.2 Stop DHCP Service

Before stopping the DHCP service, the DHCP service must be enabled.

Command

To stop the DHCP Service, use the following command:

```
dhcp stop attributes
```

Attributes

The following parameters are required to stop the DHCP service with the selected IP version.

`--ipv4, -4, --ipv6, -6`	Specifies IP version.

Examples

DHCP4

```
»dhcp stop --ipv4
DHCP4 Server is disabled.
```

DHCP6

```
»dhcp stop --ipv6
DHCP6 Server is disabled.
```

3.1.1.3 View DHCP Service Status

Command

To view DHCP service status, use the following command:

```
dhcp status attributes
```

Attributes

The following parameters are required to view the DHCP service status with selected IP version:

`--ipv4, -4, --ipv6, -6`	Specifies IP version.

Examples

DHCP4

```
»dhcp status --ipv4
Server: DHCP4
Status: Running
Interface: eth1
MAC Address Filtering: Enabled
Last Update Time: 2019-08-05 12:03:20
```

DHCP6

```
»dhcp status --ipv6
Server: DHCP6
Status: Running
Interface: eth2
Last Update Time: 2019-08-05 12:04:01
```

3.1.2 DHCP Server Network Configuration using ENM Command Line Interface

3.1.2.1 Add DHCP Network

Command

To create a DHCP network, use the following command:

```
dhcp network add attributes [options]
```

Attributes

The following parameters are required to create a DHCP network.

`-s | --subnet | no parameter name`	Specifies subnet IP address
`-n | --netmask`	Specifies netmask IP address

Options

The following parameters are optional when creating a DHCP network.

`-d | --defaultrouter`	Specify default router.
`-b | --broadcast`	Specify broadcast.
`-a | --aiws`	Specify AIWS.
`--ntp`	Specify NTP IP address list, for example, <IP1>,<IP2>.
`--dns`	Specify DNS IP address list, for example, <IP1>,<IP2>.
`--dn | --domainname`	Specify domain name.
`-r | --range`	Specify list of IP ranges, for example, <IP1>-<IP2>,<IP3>-<IP4>.
`-g | --securitygateway`	Specify Security Gateway.
`--uto`	Specify universal time offset.
`--noreload`	Specify no reload DHCP configuration.

Note:
- In case AIWS parameter not specified, AIWS value is set `default`.
- Multiple NTP IP addresses, multiple DNS IP addresses, and multiple IP ranges are separated by comma (",").

Example

```
»dhcp network add -s 193.11.1.0 -n 255.255.255.0 -d 193.11.1.4 -b 193.11.1.255 - →
a 192.168.0.186 --ntp 10.10.10.10,10.10.10.11 --dns 4.4.4.4,6.6.6.6,8.8.8.8 --dn →
 example.com -r 193.11.1.1-193.11.1.8 -g 193.11.1.1 --uto 3600
```

```
Network added successfully.

Subnet: 193.11.1.0
Netmask: 255.255.255.0
Broadcast address: 193.11.1.255
Domain name servers: 4.4.4.4, 6.6.6.6, 8.8.8.8
Domain name: example.com
NTP servers: 10.10.10.10, 10.10.10.11
Default router: 193.11.1.4
IP ranges: 193.11.1.1-193.11.1.8
Security gateway: 193.11.1.1
AIWS: 192.168.0.186
Time offset: 3600
```

3.1.2.2 Update DHCP Network

Command

To update a DHCP network, use the following command:

```
dhcp network update attributes [options]
```

Attributes

The following parameters are required to update a DHCP network.

`-s | --subnet | no parameter name`	Specifies subnet IP address
	At least one optional parameter

Options

The following parameters are optional when updating a DHCP network.

`-d | --defaultrouter`	Specify default router.
`-b | --broadcast`	Specify broadcast.
`-a | --aiws`	Specify AIWS.
`--ntp`	Specify NTP IP address list, for example, <IP1>,<IP2>.
`--dns`	Specify DNS IP address list, for example, <IP1>,<IP2>.
`--dn | --domainname`	Specify domain name.
`-r | --range`	Specify list of IP ranges, for example, <IP1>-<IP2>,<IP3>-<IP4>.
`-g | --securitygateway`	Specify Security Gateway.
`--uto`	Specify universal time offset.
`--noreload`	Specify no reload DHCP configuration.

Note: — All specified optional parameters values are overwritten.

— Multiple NTP IP addresses, multiple DNS IP addresses, and multiple IP ranges are separated by comma (",").

Example

```
»dhcp network update --subnet 193.11.1.0 --netmask 255.255.255.52


Network updated successfully.

Subnet: 193.11.1.0
Netmask: 255.255.255.52
Broadcast address: 193.11.1.255
Domain name servers: 4.4.4.4, 6.6.6.6, 8.8.8.8
Domain name: example.com
NTP servers: 10.10.10.10, 10.10.10.11
Default router: 193.11.1.4
IP ranges: 193.11.1.1-193.11.1.8
Security gateway: 193.11.1.1
AIWS: 192.168.0.186
Time offset: 3600
```

3.1.2.3 View DHCP Networks

View All DHCP Networks

Command

To view all DHCP networks, use the following command:

```
dhcp network  view
```

Options

The following parameters are optional to view all DHCP networks.

`-t | --table`	Output viewed as table.

Examples

```
»dhcp network view

Network(s) configuration:

Subnet: 193.11.1.0
Netmask: 255.255.255.0
Broadcast address: 193.11.1.255
Domain name servers: 4.4.4.4, 6.6.6.6, 8.8.8.8
Domain name: example.com
NTP servers: 10.10.10.10, 10.10.10.11
Default router: 193.11.1.4
IP ranges: 193.11.1.1-193.11.1.8
Security gateway: 193.11.1.1
AIWS: 192.168.0.186
Time offset: 3600

Subnet: 193.11.0.0
Netmask: 255.255.0.0
Broadcast address: 193.11.1.255
```

```
Domain name servers: 4.4.4.4, 6.6.6.6, 8.8.8.8
Domain name: example.com
NTP servers: 10.10.10.10, 10.10.10.11
Default router: 193.11.1.4
IP ranges: 193.11.1.1-193.11.1.8
Security gateway: 193.11.1.1
AIWS: 192.168.0.186
Time offset: 3600

2 item(s)
```

```
»dhcp network view --table
```

View DHCP Network by Subnet

Command

To view DHCP network for a specific subnet, use the following command:

```
dhcp network view  attributes
```

Attributes

The following parameters are required to view DHCP network-specific subnet.

`-s | --subnet`	Specifies subnet IP address

Example

```
»dhcp network view --subnet 193.11.1.0


Network(s) configuration:

Subnet: 193.11.1.0
Netmask: 255.255.255.52
Broadcast address: 193.11.1.255
Domain name servers: 4.4.4.4, 6.6.6.6, 8.8.8.8
Domain name: example.com
NTP servers: 10.10.10.10, 10.10.10.11
Default router: 193.11.1.4
IP ranges: 193.11.1.1-193.11.1.8
Security gateway: 193.11.1.1
AIWS: 192.168.0.186
Time offset: 3600
```

3.1.2.4 Delete DHCP Network

Command

To delete a DHCP network with a specific subnet, use the following command:

```
dhcp network delete attributes
```

Attributes

The following parameters are required to delete a DHCP network with a specific subnet.

-s | --subnet | no parameter name	Specifies subnet IP address.

Example

```
»dhcp network delete --subnet 193.11.1.0


Network deleted successfully.

Subnet: 193.11.1.0
Netmask: 255.255.255.0
Broadcast address:
Domain name servers:
Domain name:
NTP servers:
Default router:
IP ranges:
Security gateway:
AIWS: 192.168.0.186
Time offset:
```

3.1.2.5 Add Range to DHCP Network

Command

To add a range to a DHCP network with a specific subnet, use the following command:

```
dhcp range add attributes
```

Attributes

The following parameters are required to add a range to a DHCP network.

-s | --subnet	Specifies subnet IP address.
-r | --range	Specifies range to be added.

Example

```
»dhcp range add --subnet 193.11.1.0 --range 193.11.1.10-193.11.1.14


Network updated successfully.

Subnet: 193.11.1.0
Netmask: 255.255.255.52
Broadcast address: 193.11.1.255
Domain name servers: 4.4.4.4, 6.6.6.6, 8.8.8.8
Domain name: example.com
```

```
NTP servers: 10.10.10.10, 10.10.10.11
Default router: 193.11.1.4
IP ranges: 193.11.1.1-193.11.1.8,193.11.1.10-193.11.1.14
Security gateway: 193.11.1.1
AIWS: 192.168.0.186
Time offset: 3600
```

3.1.2.6 Delete Range from DHCP Network

Command

To remove a range from a DHCP network with specific subnet, use the following command:

```
dhcp range delete attributes
```

Attributes

The following parameters are required to delete a range from a DHCP network.

`-s | --subnet`	Specifies subnet IP address.
`-r | --range`	Specifies range to be deleted.

Example

```
»dhcp range delete --subnet 193.11.1.0 --range 193.11.1.10-193.11.1.14


Network updated successfully.

Subnet: 193.11.1.0
Netmask: 255.255.255.52
Broadcast address: 193.11.1.255
Domain name servers: 4.4.4.4, 6.6.6.6, 8.8.8.8
Domain name: example.com
NTP servers: 10.10.10.10, 10.10.10.11
Default router: 193.11.1.4
IP ranges: 193.11.1.1-193.11.1.8
Security gateway: 193.11.1.1
AIWS: 192.168.0.186
Time offset: 3600
```

3.1.3 DHCP Server Client Configuration using ENM Command Line Interface

3.1.3.1 Add New DHCP Client

To bind a node with fixed IP address from client ID, the user can add a client definition to DHCP records. The fixed IP must be in the network the DHCP client belongs to. To avoid IP address conflicts, the fixed IP must not be in an IP range that is reserved in this network.

Command

To add a new DHCP client, use the following command:

```
dhcp client add attributes [options]
```

Attributes

The following parameters are required to create a DHCP client.

`--client, -c`	Specifies client identifier.
`--host, -h`	Specifies hostname.
`--ip, -i`	Specifies fixed IP.
`--subnet, -s`	Specifies subnet.

Note: When adding a new DHCP client, the specific order of attributes for the `dhcp client add` command attributes must be followed:

```
dhcp client add [--client|-c] <clientIdentifier> --host|-h
<hostname> --ip|-i <ipAddress> --subnet|-s <subnet>
[options]
```

Options

The following parameters are optional when creating a DHCP client.

`--aiws, -a`	Specifies the Auto Integration Web Servers. Only one AIWS is supported.
`--ntp`	Specifies the NTP IP address.
`--uto, -u`	Specifies the Universal Time Offset. It is only applicable for IPv4.
`--noreload`	Specifies no reload DHCP configuration.

Note:

1. If optional AIWS parameter is not configured, the default AIWS value is automatically configured as the one from `subnet` that the client belongs to. If AIWS parameter is also not configured in the subnet, the default AIWS value for client is automatically configured as the `svc_CM_vip` address from `/ericsson/tor/data/global.properties` file.

2. If optional `--noreload` parameter is not used when adding new DHCP client, the DHCP configuration is automatically reloaded after the `dhcp client add command` is executed. If optional `--noreload` parameter is used instead, the operator must execute `dhcp services stop` and then `dhcp services start` commands manually for the specified IP version. This reloads DHCP configuration and makes the DHCP configuration update effective after the `dhcp client add` command is executed. Start or Stop DHCP Server requires the user to have `DHCP_Administrator` role assigned.

3. If configuring multiple clients in same IP version, to avoid the server reload after each configuration changes, use `--noreload` in the commands except for the final one. The DHCP server reloads only once after all configurations are completed.

4. Multiple NTP IP addresses are separated by comma (",").

Example

IPv4

```
»dhcp client add --client D16Q698818 --host D16Q698818 --ip 192.168.1.20 --subne →
t 192.168.0.0 --aiws 192.168.1.100 --ntp 192.168.1.101,192.168.1.102 --uto 8 --n →
oreload
Client added successfully.

Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 192.168.1.20
Subnet: 192.168.0.0
AIWS: 192.168.1.100
NTP servers: 192.168.1.101, 192.168.1.102
Time offset: 8
```

IPv6

```
»dhcp client add --client D16Q698818 --host D16Q698818 --ip 20:1b:82:13b::4f --s →
ubnet 20:1b:82:13b:: --aiws 20:1b:82:13b::1111 --ntp 20:1b:82:13b::1fff,20:1b:82 →
:13b::1eee --noreload
Client added successfully.
Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 20:1b:82:13b:0:0:0:4f
Subnet: 20:1b:82:13b:0:0:0:0
AIWS: 20:1b:82:13b:0:0:0:1111
NTP servers: 20:1b:82:13b:0:0:0:1fff, 20:1b:82:13b::1eee
Time offset:
```

3.1.3.2 Delete DHCP Client

If the user deletes a client from DHCP records, the client must exist. The client is deleted from the DHCP records after the client passes validation checks.

Command

To delete a DHCP client, use the following command:

```
dhcp client delete attribute [option]
```

Attribute

The following parameter is required to delete a DHCP client.

`--client, -c`	Specifies client identifier.

Options

The following parameter is optional when deleting a DHCP client.

`--noreload`	Specifies no reload DHCP configuration.

Note: If optional `--noreload` parameter is not used when deleting the DHCP client, the DHCP configuration is automatically reloaded after the `dhcp client delete` command is executed. If optional `--noreload` parameter is used instead, execute `dhcp services stop` and `dhcp services start` commands manually for the specified IP version. This reloads DHCP configuration, and makes the DHCP configuration update effective after the `dhcp client delete` command is executed. Start or Stop DHCP Server requires the user with DHCP_Administrator role.

Example

Delete a DHCP client without optional parameter:

```
»dhcp client delete --client D16Q698818
Client deleted successfully.

Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 192.168.1.20
Subnet: 192.168.0.0
AIWS: 192.168.1.100
NTP servers: 192.168.1.101
Time offset: 8
```

Delete a DHCP client with optional parameter:

```
»dhcp client delete -c D16Q698818 --noreload
Client deleted successfully.

Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 20:1b:82:13b:0:0:0:4f
```

```
Subnet: 20:1b:82:13b:0:0:0:0
AIWS: 20:1b:82:13b:0:0:0:1111
NTP servers: 20:1b:82:13b:0:0:0:1fff
Time offset:
```

3.1.3.3 View DHCP Client Configuration

Command

To view DHCP client configuration, use the following command:

```
dhcp client view attributes [option]
```

Attributes

The following parameters are required to view DHCP client configuration.

`--client, -c`	Specifies client identifier.
`--host, -h`	Specifies host name.
`--ip, -i`	Specifies fixed IP.
`--subnet, -s`	Specifies subnet.

Note: When viewing DHCP clients, the attribute order of `dhcp client view` command must be followed:

```
dhcp client view [--client|-c] <clientIdentifier>
dhcp client view --host|-h <hostname>
dhcp client view --ip|-i <ipAddress>
dhcp client view --subnet|-s <subnet> [option]
```

Options

The following parameter is optional when viewing DHCP client configuration:

`--table, -t`	Specifies table view for client.

Examples

View a DHCP client configuration:

```
»dhcp client view --client D16Q698818
Client(s) configuration:
Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 192.168.1.20
Subnet: 192.168.0.0
AIWS: 192.168.1.100
NTP servers: 192.168.1.101
Time offset: 8
```

View a DHCP client configuration with specified client hostname:

```
»dhcp client view --host D16Q698818
Client(s) configuration:
```

```
Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 192.168.1.20
Subnet: 192.168.0.0
AIWS: 192.168.1.100
NTP servers: 192.168.1.101
Time offset: 8
```

View a DHCP client configuration with specified client IP address:

```
»dhcp client view --ip 192.168.1.20
Client(s) configuration:
Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 192.168.1.20
Subnet: 192.168.0.0
AIWS: 192.168.1.100
NTP servers: 192.168.1.101
Time offset: 8
```

View multiple DHCP clients configurations with specified subnet:

```
»dhcp client view --subnet 192.168.0.0
Client(s) configuration:
Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 192.168.1.20
Subnet: 192.168.0.0
AIWS: 192.168.1.100
NTP servers: 192.168.1.101
Time offset: 8

Identifier: D16Q698821
Hostname: D16Q698821
Fixed IP: 192.168.1.118
Subnet: 192.168.0.0
AIWS: 192.168.1.100
NTP servers: 192.168.1.101
Time offset: 6

2 item(s)
```

View multiple DHCP clients configurations not using table view:

```
»dhcp client view
Client(s) configuration:
Identifier: D16Q698818
Hostname: D16Q698818
Fixed IP: 192.168.1.20
Subnet: 192.168.0.0
AIWS: 192.168.1.100
NTP servers: 192.168.1.101
Time offset: 8

Identifier: D16Q698821
Hostname: D16Q698821
Fixed IP: 20:1b:82:13b:0:0:0:4f
Subnet: 20:1b:82:13b:0:0:0:0
AIWS: 20:1b:82:13b:0:0:0:1111
NTP servers: 20:1b:82:13b:0:0:0:1fff
Time offset:

2 item(s)
```

View multiple DHCP clients configurations using table view:

```
» dhcp client view --table
```

Client(s) configuration:

Identifier	Hostname	Fixed IP	Subnet	AIWS	NTP servers	Time offset
D16Q698818	D16Q698818	192.168.1.20	192.168.0.0	192.168.1.100	192.168.1.101	8
D16Q698821	D16Q698821	20:1b:82:13b:0:0:0:4f	20:1b:82:13b:0:0:0:0	20:1b:82:13b:0:0:0:1111	20:1b:82:13b:0:0:0:1fff	

2 item(s)

3.1.4 DHCP Server Configuration using ENM Command Line Interface

3.1.4.1 Configure Ericsson MAC Address Filtering

Command

To configure the Ericsson MAC address filtering configuration command for DHCP4 service only, use the following command:

```
dhcp configure attributes [options]
```

Attributes

The following parameters are required to configure Ericsson MAC address filtering with selected IPv4.

`--ipv4, -4`	Specifies IP version.
`--macfilter`	Specifies Ericsson MAC address filtering.

Note: When configuring Ericsson MAC address filtering:

- With MAC filtering enabled, ENM DHCP v4 server only serves requests from Ericsson nodes with Ericsson MAC addresses. Requests from non-Ericsson nodes are ignored.
- The Ericsson MAC address filtering configuration command attribute must follow the order shown:

  ```
  dhcp configure --ipv4|-4 --macfilter <enable|disable>
  [option]
  ```

Options

The following parameters are optional:

`--noreload`	Specifies DHCP configuration change is not automatically reloaded to the running or live DHCP.

Note:

— If the optional `--noreload` parameter is not used when configuring Ericsson MAC address filtering, the DHCP configuration is automatically reloaded after the `dhcp configure` command is executed.

— If the optional `--noreload` parameter is used when configuring Ericsson MAC address filtering, an operator with `DHCP_Administrator` role must execute `dhcp stop` and `dhcp start` command manually for the specified IP version. This reloads the DHCP configuration to make the DHCP configuration update effective after the `dhcp configure` command is executed.

Example

Enable Ericsson MAC Address Filtering

```
»dhcp configure --ipv4 --macfilter enable
DHCP over IP version 4 configured successfully.
Server: DHCP4
MAC Address Filtering: Enabled
```

Disable Ericsson MAC Address Filtering

```
»dhcp configure --ipv4 --macfilter disable --noreload
DHCP over IP version 4 configured successfully.
Server: DHCP4
MAC Address Filtering: Disabled
```

3.1.4.2 Configure DHCP Server Interface

Command

To configure the DHCP server interface configuration command, use the following command:

```
dhcp configure attributes [options]
```

Attributes

The following parameters are required to configure DHCP server interface with selected IP version.

`--ipv4, -4, --ipv6, -6`	Specifies IP version.
`--interface`	Specifies DHCP server interface.

Note: When configuring DHCP server interface, the DHCP server interface configuration command attributes must follow the order shown:

```
dhcp configure --ipv4|-4|--ipv6|-6 --interface <interface>
[option]
```

Options

The following parameters are optional:

`--noreload`	Specifies DHCP configuration change is not automatically reloaded to the running or live DHCP.

Note:
- If the optional `--noreload` parameter is not used when configuring DHCP server interface, the DHCP configuration is automatically reloaded after the `dhcp configure` command is executed.
- If the optional `--noreload` parameter is used when configuring DHCP server interface, an operator with `DHCP_Administrator` role must execute `dhcp services stop` command and `dhcp services start` command manually for the specified IP version, to reload DHCP configuration to make the DHCP configuration update effective after the `dhcp configure` command is executed.

Examples

Configure DHCP4 Server Interface

```
»dhcp configure --ipv4 --interface eth1
DHCP over IP version 4 configured successfully.
Server: DHCP4
Interface: eth1
```

Configure DHCP6 Server Interface

```
»dhcp configure --ipv6 --interface eth2
DHCP over IP version 6 configured successfully.
Server: DHCP6
Interface: eth2
```

3.2 DNS Configuration for Network Element Integration

The integration of Network Elements (NE) requires a Domain Name System Server (DNS). This section describes how to enable and configure the **ITServices DNS** server provided within ENM.

If a DNS server is already available, follow the configuration instructions in *DNS Configuration using Customer Infrastructure* in the *ENM Network Integration Guide*.

Prerequisites

— The user has access to the `ITServices VM` as authorized user.

— The user has `Scripting_Operator` role assigned.

See *User Management in the Identity and Access Management Programmers Guide* - for user and role configuration in ENM.

The following tasks can be executed by an authorized user where DNS is required to manage a NE:

— Get, Set, and Delete Domain values.

— List DNS records & Forwarder addresses.

— Add and Delete DNS entries.

— Bulk Add and Delete DNS entries.

— Forward and Reverse Lookup.

— Add and Remove Forwarder Addresses.

Steps

1. Enable DNS service:

   To log on to the **ITServices** VM, see *VM Security Tasks and Connect to a Virtual Machine* in the *ENM System Administrator Guide*, and enable the `dns` service using the **ITServices** CLI.

   ```
   # ssh <authorized_user>@svc-<x>-itservices
   <authorized_user>@svc-<x>-itservices's password:

   # /opt/ericsson/itservice/bin/its.py --service enable services=dns
   Service MANAGE_NAMED successfully enabled
   ```

   The usage of the **ITServices** CLI for the services is:

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --service -h

            Usage:     python its.py --service [option] [parameter/s]
            Option:                  Parameter:
                       enable                      services=<service>,...,<serv →
   ice>
                       disable                     services=<service>,...,<serv →
   ice>
                       list
   ```

```
                    -h | --help
          Supported services: dns, ntp, dhcp4, dhcp6
```

The **ITServices** CLI information for operation and configuration of DNS is:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns -h

          Usage:      python its.py --dns [option] [--norestart] [-h]
          Option:                         Parameter:
                      add                       record_type=<record_type>
                                                ip=<ip_address>
                                                fqdn=<domain_name>
                      delete                    Mandatory for A, AAAA records:
                                                record_type=<record_type>
                                                ip=<ip_address>
                                                Mandatory for PTR records:
                                                record_type=<record_type>
                                                fqdn=<domain_name>
                      bulk_add                  csv=<csv_file>
                      bulk_delete               csv=<csv_file>
                      add_fwd_addr              Mandatory:
                                                ips=<ip>,...<ip>
                                                Optional:
                                                zone=<zone_domain>
                      del_fwd_addr              Mandatory:
                                                ips=<ip>,...<ip>
                                                Optional:
                                                zone=<zone_domain>
                      fwd_lookup                ip=<ip_address>
                      rev_lookup                fqdn=<domain_name>
                      get_domain
                      set_domain                domain=<domain_name>
                      del_domain
                      list
                      list_fwd_addr
                      -h | --help
```

When DNS data is manipulated using the **ITServices** CLI, the changes are stored in the database and the `dns` service is automatically restarted, so that the latest configuration can be fetched. The automatic restart can be overridden by using the `--norestart` argument. This allows multiple configuration changes, and the service to be restarted once all the changes have been done.

2. Configure the Domain:

   To get, set, or delete the domain, run the commands:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns get_domain
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns set_domain <domain_name> [--norestart]
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns del_domain [--norestart]
```

   Example

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns set_domain domain=example.com
DNS domain name set to: example.com
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns get_domain
DNS Domain Name: example.com
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns del_domain
```

```
DNS domain name set to: null
[<authorized_user>@svc-<x>-itservices ~]#
```

3. View all DNS Entries:

   To list all records currently defined within DNS, run the command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns list
```

   Result: The expected output is similar to the following, showing details regarding:

   — Record type.

   — IP Address.

   — Fully Qualified Domain Name.

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns list
 RECORD_TYPE |  IP_ADDRESS  |         FQDN
-------------+--------------+----------------------
 A           | 10.45.13.123 | foo.example.com
1 records
[<authorized_user>@svc-<x>-itservices ~]#
```

4. View All Forwarder Addresses currently defined within DNS by running the command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns list_fwd_addr
```

   Result: The expected output is similar to the following, showing details regarding:

   — Zone.

   — Forwarder IP Address.

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns list_fwd_addr
     zone    | forwarder_ip
-------------+---------------
 example.com | 1.2.3.7
 example2.com| 192.168.1.1
(2 rows)
[<authorized_user>@svc-<x>-itservices ~]#
```

5. Add a DNS Entry by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns add record_type=<record_type> ip=<ip_address> fqdn=<domain_name> [--n →
orestart]
```

   where:

record_type is a mandatory parameter and the value is one of the following: A, AAAA, PTR.

ip is a mandatory parameter and represents the IP address to be added.

fqdn is a mandatory parameter and represents the Fully Qualified Domain Name for the specified IP address.

Note: The `fqdn` must be a subdomain of the DNS domain.

An output example of the command is:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns add record_type=A ip=10.45.13.123 fqdn=foo.example.com
DNS Record Type: A IP: 10.45.13.123 FQDN: foo.example.com successfully added →
 to the DNS configuration
Restarting service: named
Service restarted: named
[<authorized_user>@svc-<x>-itservices ~]#

[<authorized_user>@svc-1-itservices ~]# /opt/ericsson/itservice/bin/its.py - →
-dns list
 record_type |     ip_address     |         fqdn
-------------+--------------------+----------------------
 A           | 10.45.13.123       | foo.example.com
1 records
[<authorized_user>@svc-<x>-itservices ~]#
```

6. Bulk Add of DNS Entries:

   Run the `bulk_add` option to add multiple entries into the DNS table:

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --dns bulk_add csv=<csv_file> [--norestart]
   ```

   where:

   csv is a mandatory parameter and represents the file containing the information. The file has to contain entries in the following sequence: "record_type,IP_address,FQDN".

   Example

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --dns bulk_add csv=/var/tmp/file.csv
   Number of records successfully inserted: 3/3
   Number of records that failed to insert: 0/3

   Restarting service: named
   Service restarted: named
   [<authorized_user>@svc-<x>-itservices ~]#
   [<authorized_user>@svc-<x>-itservices ~]# more /var/tmp/file.csv
   A,11.12.13.14,foo.example.COM
   PTR,11.12.13.14,bar.example.COM
   AaAa,3333::3333:ff:fef8,foo.example.COM
   [<authorized_user>@svc-<x>-itservices ~]#
   ```

7. Forwarder Lookup:

To check the domain fqdn using IP Forward lookup, run the command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itser →
vice/bin/its.py --dns fwd_lookup ip=<ip_address>
```

where:

ip is a mandatory parameter, and represents the IP address of the entry in the database for which the FQDN is displayed.

Expected output:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns fwd_lookup ip=31.32.33.34

Matching record(s):
     Record Type:    A
     IP Address:     31.32.33.34
     FQDN:           foo.example.com


[<authorized_user>@svc-<x>-itservices ~]#
```

8. Reverse lookup:

   The determination of an IP address associated with a given FQDN.

   For a reverse lookup, run the command:

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --dns rev_lookup fqdn=<domain_name>
   ```

   where:

   fqdn is a mandatory parameter and represents the known Fully Qualified Domain Name for the requested IP address

   Example

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --dns rev_lookup fqdn=bar.example.com

   Matching record(s):
        Record Type:    PTR
        IP Address:     11.12.13.14
        FQDN:           bar.example.com
   [<authorized_user>@svc-<x>-itservices ~]#
   ```

9. Delete DNS Records by running the following commands:

   Command used to delete ALIAS records:

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --dns delete record_type=<record_type> ip=<ip_address> [--norestart]
   ```

Command used to delete PTR records:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --dns delete record_type=<record_type> fqdn=<domain_name> [--norestart]
```

Example

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --dns delete record_type=A ip=10.45.13.123
DNS Record Type: A IP: 10.45.13.123 successfully deleted from the DNS config
uration
Restarting service: named
Service restarted: named
[<authorized_user>@svc-<x>-itservices ~]#

[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --dns delete record_type=PTR fqdn=foo.example.com
DNS Record Type: PTR FQDN: foo.example.com successfully deleted from the DNS
 configuration
Restarting service: named
Service restarted: named
[<authorized_user>@svc-<x>-itservices ~]#
```

10. Bulk Delete DNS records by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# python /opt/ericsson/itservice/bin
/its.py --bulk_delete csv=<csv_file> [--norestart]
```

where:

csv is a mandatory parameter and represents the file containing the information to be deleted. The file has to contain entries in the following sequence: "record_type,IP_address,FQDN".

Example

```
[<authorized_user>@svc-<x>-itservices ~]# vim file.csv_deleted
A,11.12.13.14,FOO.EXAMPLE.COM
PTR,11.12.13.14,BAR.EXAMPLE.CoM
AaaA,1111::1111:ff:fef8,FOOBAR.Example.com
[<authorized_user>@svc-<x>-itservices ~]#

[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --dns bulk_delete csv=/var/tmp/file.csv_deleted
Number of records that successfully deleted: 3/3
Number of records that failed to delete: 0/3

Restarting service: named
Service restarted: named
[<authorized_user>@svc-<x>-itservices ~]#
```

11. Add a Forwarder Address by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --dns add_fwd_addr ips=<ip_address>[,<ip_address>] [zone=<zone_domain>] [--
norestart]
```

where:

ips is a mandatory parameter used to add one or multiple forwarder addresses to the zone

Example

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns add_fwd_addr ips=10.15.128.55
DNS forwarder address(es) '10.15.128.55' added to zone 'global'
Restarting service: named
Service restarted: named
[<authorized_user>@svc-<x>-itservices ~]#
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns list_fwd_addr
         zone          |             forwarder_ip
-----------------------+----------------------------------------
 example.com           | 11.22.33.44,55.66.66.88,99.99.99.99
 example2.com          | 7.7.7.7
 global                | 192.168.1.99,10.15.128.55
(3 rows)
[<authorized_user>@svc-<x>-itservices ~]#
```

12. Delete a Forwarder Address by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns del_fwd_addr ips=<ip_address>[,<ip_address>] [zone=<zone_domain>] [-- →
norestart]
```

where:

ips is a mandatory parameter used to delete one or multiple forwarder addresses from the zone

Example

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns del_fwd_addr ips=10.15.128.55
DNS Forwarder address(es) ['10.15.128.55'] removed from zone 'global'
Restarting service: named
Service restarted: named
[<authorized_user>@svc-<x>-itservices ~]#
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --dns list_fwd_addr
        zone           |              forwarder_ip
-----------------------+----------------------------------------
 example.com           | 11.22.33.44,55.66.66.88,99.99.99.99
 example2.com          | 7.7.7.7
 global                | 192.168.1.99
(3 rows)
[<authorized_user>@svc-<x>-itservices ~]#
```

Results

The DNS server is configured according to the requirements.

3.3 NTP Configuration for Network Element Integration

The integration of Network Elements requires a Network Time Protocol (NTP) server. The following content gives information on how to enable and configure the **ITServices NTP** server provided within ENM.

If an NTP server is already available, follow the configuration instructions in NTP Configuration using Customer Infrastructure in the ENM Network Integration Guide.

Prerequisites

— The user has access to the **ITServices** VM as authorized user.

— This user needs the `Scripting_Operator` role assigned.

See *User Management* in the *Identity and Access Management Programmers Guide* - for user and role configuration in ENM.

The following tasks can be executed by an authorized user for NTP configuration:

— List external NTP servers.

— Add external NTP server.

— Update the IP address of an external NTP server.

— Delete an external NTP server.

— Enable NTP default restrictions.

— Disable NTP default restrictions.

— List NTP default restrictions.

Steps

1. Enable NTP service:

   To log on to the **ITServices** VM, see *VM Security Tasks and Connect to a Virtual Machine* in the *ENM System Administrator Guide*, and enable the `ntp` service using the **ITServices** CLI.

   ```
   # ssh <authorized_user>@svc-<x>-itservices
   <authorized_user>@svc-<x>-itservices's password:

   # /opt/ericsson/itservice/bin/its.py --service enable services=ntp
   Service MANAGE_NTPD successfully enabled
   ```

   The usage of the **ITServices** CLI for the services is:

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --service -h

           Usage:     python its.py --service [option] [parameter/s]
           Option:                   Parameter:
                      enable                      services=<service>,...,<serv →
   ice>
                      disable                     services=<service>,...,<serv →
   ice>
                      list
                      -h | --help
           Supported services: dns, ntp, dhcp4, dhcp6
   ```

   The CLI information for operation and configuration of NTP is:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --ntp -h

         Usage:     python its.py --ntp [option] [--norestart] [parameter/s]
         Option:                  Parameter:
                    add                          ips=<server_ip>,...,<server_ →
ip>
                    update                       old_ip=<old_server_ip>       →

                                                 new_ip=<new_server_ip>
                    delete                       ips=<server_ip>,...,<server_ →
ip>
                    enable                       restrictions=<option>,...,<o →
ption>
                      disable                         restrictions=<option>,. →
..,<option>
                    list
                    list_default_restrictions
                    -h | --help
```

2. View NTP records:

   To list all records currently defined within NTP, run the following command:

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --ntp list
   ```

   The expected output is an IP addresses list of the external NTP servers.

   Example

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --ntp list
    option_name |        value
   -------------+-------------------
    timeserver  | server 10.15.24.145
    timeserver  | server 10.15.24.167
    (2 rows)
   [<authorized_user>@svc-<x>-itservices ~]#
   ```

   The previous list shows the IP addresses of external NTP servers, if already configured. If the list is empty, the local **ITServices VM** onboard system time serves the Network Elements.

   The table can be populated by adding external NTP servers through the `add` option of the CLI.

3. Add NTP Entries by running the following command:

   ```
   [<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
    --ntp add ips=<ip_address>[,<ip_address>] [--norestart]
   ```

   where:

   ips is a mandatory parameter and represents the IP address of one or several servers to be added.

   Comma-separated values can be provided for multiple entries.

Example

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --ntp add ips=10.12.13.16,10.12.13.18
Server 10.12.13.16 successfully added to the NTP configuration
Server 10.12.13.18 successfully added to the NTP configuration
Restarting service: named
Service restarted: named
[<authorized_user>@svc-<x>-itservices ~]#
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --ntp list
 option_name |       value
-------------+-------------------
 timeserver  | server 10.12.13.16
 timeserver  | server 10.12.13.18
 timeserver  | server 10.15.24.145
 timeserver  | server 10.15.24.167
(4 rows)
[<authorized_user>@svc-<x>-itservices ~]#
```

4. Update NTP Entries by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --ntp update old_ip=<old_server_ip> new_ip=<new_server_ip> [--norestart]
```

where:

old_ip is a mandatory parameter and represents the existing IP address of the NTP server

new_ip is a mandatory parameter and represents the new IP address to assign

Example

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --ntp update old_ip=10.12.13.16 new_ip=10.12.13.20
Server updated from 10.12.13.16 to 10.12.13.20 within the NTP configuration
Restarting service: ntpd
Service restarted: ntpd
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --ntp list
 option_name |       value
-------------+-------------------
 timeserver  | server 10.12.13.18
 timeserver  | server 10.12.13.20
 timeserver  | server 10.15.24.145
 timeserver  | server 10.15.24.167
(4 rows)
[<authorized_user>@svc-<x>-itservices ~]#
```

5. Delete NTP Entries by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --ntp delete ips=<ip_address>[,<ip_address>] [--norestart]
```

where:

ips is a mandatory parameter and represents the IP address of the server to be deleted. Comma separated values can be provided for multiple entries.

Example

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --ntp delete ips=10.12.13.20
Server 10.12.13.20 successfully deleted from the NTP configuration
Restarting service: ntpd
Service restarted: ntpd
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --ntp list
 option_name |       value
-------------+-------------------
 timeserver  | server 10.12.13.18
 timeserver  | server 10.15.24.145
 timeserver  | server 10.15.24.167
(3 rows)
[<authorized_user>@svc-<x>-itservices ~]#
```

6. List NTP default restrictions by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --ntp list_default_restrictions
  DEFAULT    |   STATUS
---------------------------
 nopeer      |   enabled
 nomodify    |   enabled
 noquery     |   enabled
 kod         |   enabled
 notrap      |   enabled
---------------------------
[<authorized_user>@svc-<x>-itservices ~]#
```

7. Enable NTP default restrictions by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --ntp enable restrictions=kod,notrap
Default kod successfully enabled
Default notrap successfully enabled
Restarting service: ntpd
[<authorized_user>@svc-<x>-itservices ~]#
```

8. Disable NTP default restrictions by running the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --ntp disable restrictions=kod,notrap
Default kod successfully disabled
Default notrap successfully disabled
Restarting service: ntpd
[<authorized_user>@svc-<x>-itservices ~]#
```

Results

The NTP server is configured according to the requirements.

Note: Certain ENM Application and Network Element functions require that the Network Element NTP clock is synchronized with that of the ENM.

For more information on the NTP configuration for the relevant Network Element, see the specific CPI content for the Network Element.

3.4 Trusted NTP Server Configuration

3.4.1 Trusted NTP Server Configuration

ENM configures reliable time and date information on the node through NTP Server. Considering current security threats, It is imperative to introduce authentication and integrity mechanisms between node and NTP server while time synchronization. To achieve this, NTP Server must be configured with security parameters (`Secret keys` and `Digest Algorithm`) to become trustworthy. This section describes the steps to configure trusted NTP Server in ENM.

Prerequisites

— User has access to the `ITServices VM` as authorized user.

— User has `Scripting_Operator` role assigned.

See *User Management* in the *ENM Identity and Access Management System Administrator Guide, 2/1543-aom 901 151-1 Uen* - for user and role configuration in ENM.

The following operations are performed by an authorized user for Trusted NTP Server Configuration:

1. Generate NTP keys.
2. List NTP keys.
3. Disable NTP default restrictions.
4. Disable NTP service.
5. Enable NTP service.
6. Restart NTP service.

Note: All the previous steps must be performed in only one of the `ITServices VM`. Execute Steps 5 and 6 in all the remaining `ITService VMs` .

Command Syntax

```
# ssh <authorized_user>@svc-<x>-itservices
<authorized_user>@svc-<x>-itservices's password:

[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp -h

         Usage:     python its.py --ntp [option] [--norestart] [parameter/s]

         Option:                  Parameter:                              descript →
ion:

                    add                          ips=<server_ip>,...,<server_ip>   →

                    update                       old_ip=<old_server_ip>            →
                                                 new_ip=<new_server_ip>            →
```

```
                    delete                      ips=<server_ip>,...,<server_ip>
                    list
                    enable                      restrictions=<option>,...,<optio →
n>
                    disable                     restrictions=<option>,...,<optio →
n>
                    list_default_restrictions
                    generate_keys
                    list_key_data               -all
                                                -mapped
                                                node_fdn=<node_fdn>,...,<node_fd →
n>
                                                key_id=<key_id>,...,<key_id>
                                                key_id_file=<filepath>
                                                node_fdn_file=<filepath>
                    remove_key_map               -all
                                                node_fdn=<node_fdn>,...,<node_fd →
n>
                                                key_id=<key_id>,...,<key_id>
                                                key_id_file=<filepath>
                                                node_fdn_file=<filepath>
                    renew_keys                  -all
                                                node_fdn=<node_fdn>,...,<node_fd →
n>
                                                key_id=<key_id>,...,<key_id>
                                                key_id_file=<filepath>
                                                node_fdn_file=<filepath>
                    delete_ntp_keys
                    -h | --help
```

3.4.1.1 Generate NTP Keys

> Describes the procedure to generate NTP keys for configuration of trusted NTP Server in `ITService VM`.

To generate the NTP keys, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp generate_keys
```

The following is the output for the NTP key generation if it is performed for the first time:

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n
tp generate_keys
Please enter the number of keys has to be generated
Number of keys ? 40000


PLEASE WAIT, DO NOT PRESS ANY KEY UNTIL ALL KEYS ARE GENERATED SUCCESSFULLY.....
..

Key generation has been started for 40000 key (s)....
Key generation has been successfully completed, Please use list option to view a
vailable keys
[<authorized_user>@svc-<x>-itservices ~]
```

The following input confirmation is displayed where NTP key data is already present in the `ITServices DB`.

```
A (Add more keys to existing keys)
R (Replace all the keys)
```

— **A (Add more keys to existing keys):**

  This adds new key data to the existing key data where the number of required NTP keys to be entered.

  Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --ntp generate_keys
The number of keys which are available : 40000
The number of keys which are available to map :40000
The number of keys which are mapped with the nodes :0
Note:Since the key data exists in the server. To Continue, Enter one of the
following options:
To add more keys    : A
To replace all keys : R

Your Option ? A
Please enter the number of keys has to be generated
Number of keys ? 1000


PLEASE WAIT, DO NOT PRESS ANY KEY UNTIL ALL KEYS ARE GENERATED SUCCESSFULLY.
......

Key generation has been started for 1000 key (s)....
1000 key(s) data shall be added to existing key data
Addition of keys has been successfully completed, Please use list command to
 view available keys.
```

— **R (Replace all the keys)**

  This replaces all the existing NTP keys with newly generated keys where number of required NTP keys to be entered.

  Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py
 --ntp generate_keys

The number of keys which are available : 41000
The number of keys which are available to map :41000
The number of keys which are mapped with the nodes :0
Note:Since the key data exists in the server. To Continue, Enter one of the
following options:
```

```
To add more keys    : A
To replace all keys : R

Your Option ? R
All the keys will be replaced as Replace option given by user
Please enter the number of keys has to be generated
Number of keys ? 450
PLEASE WAIT, DO NOT PRESS ANY KEY UNTIL ALL KEYS ARE GENERATED SUCCESSFULLY. →
......

Replacing of keys has been started for 450 key(s)...
Key data have been successfully replaced, Please use list command to view av →
ailable keys.
```

When the operator is trying to replace the key data by giving option R, where some of the keys are already mapped with network elements, then the operator must provide confirmation whether to proceed or not. The following input confirmation is displayed when existing keys are already mapped with the network elements.

```
Note : Some of the existing keys are already mapped with the nodes, If you p →
roceed with yes, then all keys mappings with nodes will be removed and all t →
he old keys will be replaced with new keys.
Would you like to continue (yes/no) ?
```

If user answers `yes`, then execution continues, if the answer is `no`, then execution terminates.

The following is the example for the confirmation value `yes`:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py →
 --ntp generate_keys

The number of keys which are available : 40000
The number of keys which are available to map :40000
The number of keys which are mapped with the nodes :0
Note:Since the key data exists in the server. To Continue, Enter one of the  →
following options:
To add more keys    : A
To replace all keys : R

Your Option ? R
All the keys will be replaced as Replace option given by user

Note : Some of the existing keys are already mapped with the nodes, If you p →
roceed with yes, then all keys mappings with nodes will be removed and all t →
he old keys will be replaced with new keys.
Would you like to continue (yes/no) ?
yes
Replacing of NTP keys has been started as user has given 'YES' as input
Please enter the number of keys has to be generated
No of keys ? 40000
PLEASE WAIT, DO NOT PRESS ANY KEY UNTIL ALL KEYS ARE GENERATED SUCCESSFULLY. →
......

Key generation has been started for 40000 keys....
Key generation has been successfully completed, Please use list option to vi →
ew available keys
```

The following is the example for the confirmation value 'no':

```
[<authorized_user>@svc-<x>-itservices ~]#/opt/ericsson/itservice/bin/its.py  →
--ntp generate_keys

The number of keys which are available : 40000
The number of keys which are available to map :40000
The number of keys which are mapped with the nodes :0
```

```
Note:Since the key data exists in the server. To Continue, Enter one of the
following options:
To add more keys    : A
To replace all keys : R

Your Option ? R
All the keys will be replaced as Replace option given by user

Note : Some of the existing keys are already mapped with the nodes, If you p
roceed with yes, then all keys mappings with nodes will be removed and all t
he old keys will be replaced with new keys.
Would you like to continue (yes/no) ?
no
Replacing of NTP keys has been terminated as user has given 'NO' as input.
```

3.4.1.2 List NTP Keys

> Describes the procedure to List NTP key data.

The following are the options to `list` command to display the generated key data:

- `-all`
- `-mapped`
- `key_id`
- `node_fdn`
- `node_fdn_file`
- `key_id_file`

Option (-all):

To list all key data, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n
tp list_key_data  -all
```

The expected output is a list of `key_ids`, network element `node_fdns`, and `is_renew_key_distributed`.

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]#  /opt/ericsson/itservice/bin/its.py --
ntp list_key_data  -all

key_id                          node_fdn                    is_renew_key_distrib
uted
1                                NODE0001
2                                NODE0002                   NA
3                                NODE0003                   No
4                                NODE0004                   Yes
```

— `is_renew_key_distributed` attribute value is empty, when keys are generated in the `itservices` and are not configured on the node.

— `is_renew_key_distributed` attribute value is NA, when keys are generated in the `itservices` and are configured on the node.

— `is_renew_key_distributed` attribute value is No, when keys are renewed in the `itservices` and node is not configured with the renewed key.

— `is_renew_key_distributed` attribute value is set to Yes, when keys are renewed in the `itservices` and node is configured with the renewed key.

Option (-mapped):

To list key data for `key_id` for which mapping with `node_fdn` is present, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp list_key_data  -mapped
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]#  /opt/ericsson/itservice/bin/its.py -- →
ntp list_key_data  -mapped

key_id                          node_fdn                    is_renew_key_distrib →
uted
1                               NODE0001
2                               NODE0002
```

Option (node_fdn):

To list key data based on `node_fdn`, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp list_key_data node_fdn=<NODE_FDN1>,...,<NODE_FDNN>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]#  /opt/ericsson/itservice/bin/its.py -- →
ntp list_key_data node_fdn=NODE0001,NODE0002

key_id                          node_fdn                is_renew_key_distributed →

1                               NODE0001
2                               NODE0002
```

Option (key_id):

To list key data based on `key_id`, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp list_key_data key_id=<key_id>,...,<key_id>
```

Example Output>

```
[<authorized_user>@svc-<x>-itservices ~]#  /opt/ericsson/itservice/bin/its.py --
ntp list_key_data key_id=1,2

key_id                          node_fdn                is_renew_key_distributed

1                               NODE0001
2                               NODE0002
```

Option (node_fdn_file):

To list key data for many Network Elements, a file input can be given that contains `node_fdn`:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n
tp list_key_data node_fdn_file=<filepath>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]#  /opt/ericsson/itservice/bin/its.py --
ntp list_key_data node_fdn_file=/var/tmp/inputFile.txt

key_id                          node_fdn                is_renew_key_distributed

1                               NODE0001
2                               NODE0002
```

Sample Input File

```
NODE0001
NODE0002
NODE0003
.
.
.
.
<NODE_FDN_n>
```

Option (key_id_file):

To list key data for many `key_ids`, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n
tp list_key_data key_id_file=<filepath>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]#  /opt/ericsson/itservice/bin/its.py --
ntp list_key_data  key_id_file=/var/tmp/inputFile.txt

key_id                          node_fdn                is_renew_key_distributed

1                               NODE0001
2                               NODE0002
```

Sample Input File

```
1
2
3
.
.
.
.
<n>
```

Note: If a large amount of key data is found in DB, instead of displaying key data on console, all the key data is copied to `/var/tmp/list_keydata.txt` file.

3.4.1.3 Disable NTP Default Restrictions

> Describes the procedure to disable default restrictions in NTP server.

To list all the default restrictions, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp list_default_restrictions
  DEFAULT    |  STATUS
---------------------------
 kod         |  enabled
 nomodify    |  enabled
 notrap      |  enabled
 nopeer      |  enabled
 noquery     |  enabled
---------------------------
```

Note: Run the following command, where any of the previously listed restrictions is in an `enabled` state.

To disable default restrictions, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp disable restrictions=kod,nomodify,notrap,nopeer,noquery
Default restriction kod successfully disabled
Default restriction nomodify successfully disabled
Default restriction notrap successfully disabled
Default restriction nopeer successfully disabled
Default restriction noquery successfully disabled
Restarting service: ntpd
Service restarted: ntpd
```

3.4.1.4 Disable the NTP Service

> Describes the procedure to disable NTP service.

To disable the NTP service, execute the following command:

```
# /opt/ericsson/itservice/bin/its.py --service disable services=ntp
Service MANAGE_NTPD successfully disabled with ITServices
Service ntp may not be stopped on the server. Use the RedHat command 'service <s →
ervice_name> status' to double check
```

3.4.1.5 Enable the NTP Service

> Describes the procedure to enabling NTP service.

To enable the NTP service, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --service enable services=ntp
Please wait, while NTP key data is being updated in /etc/ntp/keys and /etc/ntp.conf files.
Service MANAGE_NTPD successfully enabled within ITServices
Service ntp may not be started on the server. Use the RedHat command 'service <service_name> status' to double check
```

3.4.1.6 Restart the NTP Service

> Describes the procedure to restart NTP service.

After successful execution of all the previous steps, restart the NTP service using the following command:

```
# ssh <authorized_user>@svc-<x>-itservices
<authorized_user>@svc-<x>-itservices's password:
#sudo service ntpd restart
Shutting down ntpd:                                        [  OK  ]
Starting ntpd:                                             [  OK  ]
```

Note: To verify successful configuration, check `/etc/ntp.conf` and `/etc/ntp/keys` files whether all the generated keys data updated or not.

Example Output

```
ssh <authorized_user>@svc-<x>-itservices
<authorized_user>@svc-<x>-itservices's password:
#cat /etc/ntp/keys
1 MD5 F966192C25ED519E285556CDDAAF422FC5413F2E85378105
2 MD5 90BF7D2BDA9FEA4327C6175F3157BF4A8AF8BD5CF58598C2
3 MD5 F9B8996F01F6156414776C332E89607C853983B813AF65C3
4 MD5 61E211FC7D410ADF730D20C58EDAAE74DC68DBA8C02A94C9
5 MD5 3C84EDCBE722DCE0DE99606CF9464E68601426F3E9E472A1
.
.
.
40000 MD5 3CCD55D3C14A67A27F4FF3764B1919CBFFB32B4A02B7CCA7
ssh <authorized_user>@svc-<x>-itservices
<authorized_user>@svc-<x>-itservices's password:
#cat /etc/ntp.conf
#DO NOT EDIT THIS FILE MANUALLY, ALL CHANGES WILL BE LOST AFTER RESTART OF ITSERVICE DAEMON!
restrict 127.0.0.1
restrict -6 ::1
server 10.186.140.98
server  127.127.1.0
server -6 ::1
driftfile /var/lib/ntp/drift
keys /etc/ntp/keys
#END_OF_FILE
trustedkey 1 2 3 4 5.....40000
```

```
controlkey 1 2 3 4 5.....40000
requestkey 1 2 3 4 5.....40000
```

3.4.2 Renew NTP Keys

> Describes the procedure to renew key data. Only keys data is renewed, and any existing network element mappings remains the same.

The following operations must be performed by an authorized user for `Trusted NTP Server Configuration`:

1. Renew NTP keys.
2. Disable NTP default restrictions.
3. Disable NTP service.
4. Enable NTP service.
5. Restart the NTP service.

Note: All the previous steps must be performed in only one of the `ITServices VMs`. Execute Steps 4 and 5 in all the remaining `ITService VM`.

The operator must delete key data on the configured nodes before executing renew action. See Renew Key Data on page 56.

3.4.2.1 Renew Key Data

Options to Renew Key Data:

- `-all`
- `key_id`
- `node_fdn`
- `node_fdn_file`
- `key_id_file`

Option (-all):

To renew all the key data, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp renew_keys -all
```

The following output can be expected if there are no key mappings with the Network Elements.

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]#  /opt/ericsson/itservice/bin/its.py --ntp renew_keys -all

PLEASE WAIT, DO NOT PRESS ANY KEY UNTIL ALL KEYS ARE RENEWED SUCCESSFULLY.......

Renew of all keys has been successfully updated in DB with the new keys.
Please wait, while NTP key data is being updated in /etc/ntp/keys and /etc/ntp.conf files.
Renew key(s) action has been completed successfully.
```

When the operator is trying to renew all the key data, where some of the keys are mapped with Network Elements, then the operator must enter confirmation for the following message:

```
Note : Some of the existing keys are already mapped with the nodes, If you proceed with yes, then all the old keys will be replaced with new keys.
Would you like to continue (yes/no) ?
```

If user answers `yes`, then execution continues, if the answer is `no`, execution terminates.

The following is the example for the confirmation value `yes`:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --ntp renew_keys -all
Note : Some of the existing keys are already mapped with the nodes, If you proceed with yes, then all the old keys will be replaced with new keys.
Would you like to continue (yes/no) ?

yes

PLEASE WAIT, DO NOT PRESS ANY KEY UNTIL ALL KEYS ARE RENEWED SUCCESSFULLY.......

Renew of all keys has been successfully updated in DB with the new keys.
Please wait, while NTP key data is being updated in /etc/ntp/keys and /etc/ntp.conf files.
Renew key(s) action has been completed successfully.
```

The following is the example for the confirmation value `no`:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --ntp renew_keys -all
Note : Some of the existing keys are already mapped with the nodes, If you proceed with yes, then all the old keys will be replaced with new keys.
Would you like to continue (yes/no) ?


no

Renew of NTP keys has been terminated as user has given 'NO' as input.
```

Option (key_id):

To renew key data based on `key_ids`, execute the following commands:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp renew_keys key_id=<Key_id>,...,<key_id>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp renew_keys key_id=2,5,6
Keys have been successfully renewed in DB.
Key data has been renewed for the given below key_id (s)
['2', '5', '6']
Renew key(s) action has been completed successfully.
```

Option (node_fdn):

To renew key data based on node_fdns, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp renew_keys node_fdn=<node_fdn>,...,<node_fdn>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp renew_keys node_fdn=NODE0001,NODE0002,NODE0003
Keys have been successfully renewed in DB.
Key data has been renewed for the given below node_fdn (s)
['NODE0001', 'NODE0002', 'NODE0003']
Renew key(s) action has been completed successfully.
```

Option (node_fdn_file):

To renew key data based on node_fdns, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp renew_keys node_fdn_file=<input_file>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp renew_keys node_fdn_file=inputnodefdns.txt
Keys have been successfully renewed in DB.
Key data has been renewed for the given below node_fdn (s)
['NODE0001', 'NODE0002', 'NODE0003'..... 'NODE000_n']
Renew key(s) action has been completed successfully.
```

Sample Input file

```
NODE0001
NODE0002
NODE0003
.
.
.
.
<NODE_FDN_n>
```

Option (key_id_file):

To renew key data based on `key_ids`, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n
tp renew_keys key_id_file=<input_file>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n
tp renew_keys key_id_file=inputkeyids.txtKeys have been successfully renewed in
DB.
Key data has been renewed for the given below key_id (s)
['2', '8', '15']
Renew key(s) action has been completed successfully.
```

Sample Input File

```
2
8
15
.
.
.
.
<n>
```

Once key data is renewed in the `ITServices DB` to reflect renewed key data in `/etc/ntp/keys` file, complete the following steps:

1. List NTP Keys on page 51
2. Disable NTP Default Restrictions on page 54
3. Disable the NTP Service on page 54
4. Enable the NTP Service on page 55
5. Restart the NTP Service on page 55

3.4.3 Remove Mapping between Key and Node

> Describes the procedure to remove mapping between key id and network element in `ITService DB`.

To remove key mapping with the Network Elements, the operator can perform with the following options:

— `-all`

— `key_id`

— `node_fdn`

— node_fdn_file

— key_id_file

Option (-all):

To remove all NTP keys mapping, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map -all
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map -all
Mapping is removed successfully for all node_fdns
```

Option (key_id):

To remove all NTP keys mapping for key_ids, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map key_id=<key_id>,...,<key_id>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map key_id=1,2
Mapping has been successfully removed for key_id (s)['1','2']
```

Option (node_fdn):

To remove all NTP keys mapping for node_fdns, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map node_fdn=<node_fdn>,...,<node_fdn>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map node_fdn=NODE0001,NODE0002
Mapping has been successfully removed for node_fdn (s)['NODE0001','NODE0002']
```

Option (node_fdn_file):

To remove all NTP keys mapping by providing the node_fdns in a file as an input parameter, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map node_fdn_file=<input_file>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map node_fdn_file=inputnodefdns.txt
Mapping has been successfully removed for node_fdn (s)['NODE0001','NODE0002']
```

Option (key_id_file):

To remove all NTP keys mapping by providing the `key_ids` in a file as an input parameter, execute the following command:

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map key_id_file=<input_file>
```

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp remove_key_map key_id_file=inputkeyids.txt
Mapping has been successfully removed for key_id 1Mapping has been successfully →
removed for key_id (s)['1','2']
```

3.4.4 Delete NTP Keys

> Describes the procedure for deletion of key data from `ITServices DB`.

The operator must delete key data on the configured nodes. See *Remove NTP Server Details* in *ENM Network Security Configuration System Administrator Guide*, 2/1543-AOM 901 151-2.

After successful execution of this operation, all key data will be removed from `ntp_keys` table and `/etc/ntp/keys` file.

Example Output

```
[<authorized_user>@svc-<x>-itservices ~]# /opt/ericsson/itservice/bin/its.py --n →
tp  delete_ntp_keys
Keys are deleted from ntp_keys table and /etc/ntp/keys file. To reflect these ch →
anges, disable and then enable the ntp service
```

3.4.5 Verify that ENM is Configured with Trusted NTP Server

Steps

1. Where ENM version is less than 19.15 sprint, ENM is configured with untrusted NTP Server.
2. Where ENM version is 19.15 sprint or above, then use the following steps to know if `Trusted NTP Server` is configured in the ENM:

a. Connect to ITServices VM, view /etc/ntp.conf file. It contains trustedkey, controlkey, and requestkey with keyid values. To view /etc/ntp.conf file, execute the following command:

```
vim /etc/ntp.conf
```

Example Output

```
[cloud-user@svc-3-itservices ~]$ vim /etc/ntp.conf
#DO NOT EDIT THIS FILE MANUALLY, ALL CHANGES WILL BE LOST AFTER RES →
TART OF ITSERVICE DAEMON!
restrict 127.0.0.1
restrict -6 ::1
server  127.127.1.0
server -6 ::1
driftfile /var/lib/ntp/drift
keys /etc/ntp/keys
#END_OF_FILE
trustedkey 1 2 3 4 5
controlkey 1 2 3 4 5
requestkey 1 2 3 4 5
```

b. Connect to ITServices VM, keys file in the path /etc/ntp. It contains Algorithm Type, keys, and associated keyids, listed in ntp.conf file from previous step. To view /etc/ntp/keys file, execute the following command in ITServices VM.

```
vim /etc/ntp/keys
```

Example Output

If the previously mentioned information is met, then Trusted NTP Server is configured in ENM.

```
[root@svc-3-itservices cloud-user]# vi /etc/ntp/keys
1 MD5 55545485d7c1be8d7ce8cf76b86575e534ce4e84
2 MD5 7535138ecd55c48457418df8e5be567414e5715f
3 MD5 dd4c8c6d8c3bd1b15d5ebc6bfcc3f1d5dec7415e
4 MD5 de66d43ccbd48b81cdb38f6c7356b34b66ddc755
5 MD5 b6d6c55cbebedcf5ed5dcb4dc6e8d3656ee55713
```

3.5 E2E Offline Enrollment Procedure for Baseband Radio Node

> This section describes the set of steps to perform offline enrollment of Radio Node.

Enrollment is a procedure by which a node gets its credential and a set of trusted credentials. Enrollment described here is called an offline enrollment procedure. The operator has to generate the node credential and trusted credentials manually, and install them on the node, in an offline manner.

The procedure described here is applicable for `OAM Enrollment` only. Do not use it for `IPsec Enrollment`.

Detailed description of operations to be performed on the node are available in *Manage Security, 18/1553-LZA 701 6014*.

Target Groups

The intended target groups for this procedure are the following:

— Security Administrator.

— System Administrator.

— ENM Operator.

— Installation Engineer.

— Radio Network Engineer.

Overview

The offline procedure is based on three main steps:

1. End Entity creation and Credential generation from ENM.
2. LDAP Configuration from ENM.
3. Node configuration.

Details for each step are described.

After these steps, the Radio Node is ready to be added in ENM.

At the end of the procedure, the Radio Node is added in ENM and in SYNC with TLS.

3.5.1 End Entity Creation and Credential Generation for E2E Offline Enrollment for Baseband Radio Node

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System. They use it for communication with other ENM systems. End Entities must be created in the PKI system.

To generate credentials, each End Entity is mapped to an Entity Profile (EP) that defines the Certificate Authority (CA).

For information about Entity and Profiles, see *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide* 2/1543-AOM 901 151-3.

To access the web-based ENM application as ADMINISTRATOR, open the cliapp, then execute the following commands:

Prerequisites

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system, and is familiar with its commands.

See *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3.

— The operator has root access privileges to log in to the management server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has ADMINISTRATOR role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The Radio Node SW. version is Rel16A IP12 or higher.

Steps

1. Check Entity Profile:

   List all the Entity Profiles already present in ENM PKI system:

   ```
   pkiadm pfm --list -type entity
   ```

   The highlighted profile must be available in the command output:

```
SCEPRA_IPSec_EP                          ENTITY_PROFILE
Alarm_Monitor_Client_Email_EP            ENTITY_PROFILE
DUSGen2OAM_CHAIN_EP                      ENTITY_PROFILE
MacroRBSGen1OAM_CHAIN_EP                 ENTITY_PROFILE
MicroRBSOAM_CHAIN_EP                     ENTITY_PROFILE
PicoRBSOAM_CHAIN_EP                      ENTITY_PROFILE
DUSGen2IPSec_CHAIN_EP                    ENTITY_PROFILE
MacroRBSGen1IPSec_CHAIN_EP               ENTITY_PROFILE
MicroRBSIPSec_CHAIN_EP                   ENTITY_PROFILE
PicoRBSPISec_CHAIN_EP                    ENTITY_PROFILE
```

The highlighted profile is the default profile used when OAM Enrollment is performed for Radio Nodes.

2. Prepare the XML File for the End Entity Creation:

   A different End Entity must be created for each Radio Node.

   The End Entities are created from an XML file. The template for the XML follows:

   `Template.xml`

```
<Entities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamesp →
aceSchemaLocation="EntitiesSchema.xsd">
  <Entity>
    <PublishCertificatetoTDPS>true</PublishCertificatetoTDPS>
    <EntityProfile Name="DUSGen2OAM_CHAIN_EP"/>
    <KeyGenerationAlgorithm>
      <Name>RSA</Name>
      <KeySize>2048</KeySize>
    </KeyGenerationAlgorithm>
    <Category>
      <Modifiable>true</Modifiable>
      <Name>NODE-OAM</Name>
    </Category>
    <EntityInfo>
      <Name>NetworkElementID-oam</Name>
      <Subject>
        <SubjectField>
          <Type>ORGANIZATION</Type>
          <Value>ERICSSON</Value>
        </SubjectField>
        <SubjectField>
          <Type>ORGANIZATION_UNIT</Type>
          <Value>BUCI DUAC NAM</Value>
        </SubjectField>
        <SubjectField>
          <Type>COUNTRY_NAME</Type>
          <Value>SE</Value>
        </SubjectField>
```

```
        <SubjectField>
          <Type>COMMON_NAME</Type>
          <Value>NetworkElementID-oam</Value>
        </SubjectField>
      </Subject>
    </EntityInfo>
  </Entity>
</Entities>
```

In XML creation, the following rules must be followed:

— In the `<EntityInfo>` tag, the `<name>` must be `Network Element ID-oam`.

  The `NetworkElementID` is the identifier that the user must use at the end of this procedure when the Radio Node is added in ENM.

— In the `<EntityInfo><SubjectField>` tag, the `<Value>` must be `NetworkElementID-oam`.

  The `NetworkElementID` is the identifier that the user must use at the end of this procedure when the Radio Node is added in ENM.

The file is an example of the XML used to generate the End Entity for the `RadioNode` with `Network Element ID=G2RBS_27`.

`EE-G2RBS_27-oam.xml`

```
<Entities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamesp →
aceSchemaLocation="EntitiesSchema.xsd">
  <Entity>
    <PublishCertificatetoTDPS>true</PublishCertificatetoTDPS>
    <EntityProfile Name="DUSGen2OAM_CHAIN_EP"/>
    <KeyGenerationAlgorithm>
      <Name>RSA</Name>
      <KeySize>2048</KeySize>
    </KeyGenerationAlgorithm>
    <Category>
      <Modifiable>true</Modifiable>
      <Name>NODE-OAM</Name>
    </Category>
    <EntityInfo>
      <Name>G2RBS_27-oam</Name>
      <Subject>
        <SubjectField>
          <Type>ORGANIZATION</Type>
          <Value>ERICSSON</Value>
        </SubjectField>
        <SubjectField>
          <Type>ORGANIZATION_UNIT</Type>
          <Value>BUCI DUAC NAM</Value>
        </SubjectField>
        <SubjectField>
          <Type>COUNTRY_NAME</Type>
          <Value>SE</Value>
        </SubjectField>
        <SubjectField>
          <Type>COMMON_NAME</Type>
          <Value>G2RBS_27-oam</Value>
        </SubjectField>
      </Subject>
    </EntityInfo>
  </Entity>
</Entities>
```

3. Save the XML file:

Suggested name for the file is `EE_Network Element ID-oam.xml`.

4. Create the End Entity:

   Drag and drop the XML file created in Step 2 into the ENM CLI app and run the following command to create the End Entity:

   ```
   pkiadm etm -c -xf file:EE_NetworkElementID-oam.xml
   ```

5. Verify the End Entity Creation:

   Verify that the End Entity has been created by listing all the End Entities in the ENM PKI system with the command:

   ```
   pkiadm etm -l -type ee
   ```

   The End Entity must be present in the list of End Entity.

   The End Entity is created, with name `Network Element ID-oam`, and its status is `NEW`. The following example shows an EE related to `Radio Node G2RBS_27`.

   Example

   ```
   91  CREDMAPIieatrcxb5553                    ENTITY          ACTIVE
   93  CREDMAPIieatrcxb4717                    ENTITY          ACTIVE
   95  G2RBS_26-ipsec                          ENTITY          ACTIVE
   94  G2RBS_26-oam                            ENTITY          ACTIVE
   96  G2RBS_27-oam                            ENTITY          NEW
   ```

6. End Entity Credential Generation:

   The following command generates the End Entity credential, and packages it in a `p12` file. The `p12` file also contains the corresponding private key of the End Entity.

   ```
   pkiadm ctm EECert -gen -nocsr -en NetworkElementID-oam -f P12
   ```

   In the previous command `NetworkElementID-oam` is the name of the entity created in Step 4.

7. Download of Trusted Credentials:

   For OAM connectivity with `netconf` over TLS, the following trusted CA credentials are needed on node:

   a. `ENM_OAM_CA`.

   b. `ENM_Infrastructure_CA`.

c. ENM_PKI_Root_CA.

The ENM_PKI_Root_CA credential is already included in the p12 file generated in Step 6. The other two credentials must be downloaded manually.

Run the command to download ENM_OAM_CA and ENM_PKI_Root_CA in .pem format.

```
pkiadm ctm CACert -expcert -en ENM_OAM_CA -f PEM
pkiadm ctm CACert -expcert -en ENM_Infrastructure_CA -f PEM
pkiadm ctm CACert -expcert -en ENM_PKI_Root_CA -f PEM
```

3.5.2 LDAP Configuration for E2E Offline Enrollment for Baseband Radio Node

Prerequisites

The following prerequisites must be fulfilled before progressing with this procedure:

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system and is familiar with its commands.

See *Public Key Infrastructure System* of the *ENM Security System Administrator Guide 2/1543-AOM 901 151*.

— The operator has root access privileges to log in to the management server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has ADMINISTRATOR role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The Radio Node SW version is Re116A IP12 or higher.

Steps

1. Create Data for LDAP Configuration:

   To gather the ldap configuration information from the ENM system, use the following command:

   ```
   secadm ldap configure --manual
   ```

   A sample output of running the command is as follows:

```
»secadm ldap configure --manual

PROPERTY                        VALUE
fallbackLdapIpv6Address
ldapIpv6Address
tlsPort                         1389
ldapIpv4Address                 141.137.211.41
fallbackLdapIpv4Address         141.137.211.42
ldapsPort                       1636
bindDn                          cn=ProxyAccount_1,ou=proxyagent,ou=com,dc=ieatclvmlms904,dc=com
baseDn                          dc=ieatclvmlms904,dc=com
bindPassword                    wuo08qpz

Command Executed Successfully
```

2. Gather the LDAP configuration data for Radio node:

   This data is used to populate the MO Attributes on the node in the last step of this procedure.

   With the following indications, collect the value for each MO Attribute listed in the first column:

MO Attribute	Value	Description
`Ldap->baseDn`	get from Step 1	Base `dn` for which the client binds.
`Ldap->bindDn`	get from Step 1	Proxy account for bind.
`Ldap->bindPassword`	get from Step 1	Password for proxy account bind.
`Ldap->ldapIpAddress`	get from Step 1 (1)	The IP address of the primary LDAP server.
`Ldap->fallbackLdapIpAddress`	get from Step 1 (1)	The IP address of the fallback LDAP server. The fallback server is used when the primary server is inaccessible.
`Ldap->serverPort`	1636	Ldap->tlsMode server port number to access the primary and secondary LDAP servers.
	LDAPS	Toggles the TLS establishment mode to access the primary and secondary LDAP servers. Takes effect when useTls is true.
`Ldap->useTls`	true	Ldap-Toggle to enable TLS access to primary and secondary LDAP targets.
`Ldap->userLabel`	ENM Server	Extra descriptive text.
`Ldap->profileFilter`	ERICSSON_FILTER	Selects an LDAP filter to determine the user authorization profile. Authorization profiles are used by authorization methods. For `ERICSSON_FILTER`, the contained EricssonFilter MO must exist.
`EricssonFilter`	1	
`Filter`	1	

(1) If the node is IPv4 type, get the IPv4 `ldap` and `fallbackldap` IP addresses. If the node is IPv6 type get the IPv6 `ldap` and `fallbackldap` IP addresses.

3.5.3 Node Configuration for E2E Offline Enrollment for Baseband Radio Node

Prerequisites

The following prerequisites must be fulfilled before progressing with this procedure:

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system and is familiar with its commands.

See *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide* 2/1543-AOM 901 151-3.

— The operator has root access privileges to log in to the management server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has `ADMINISTRATOR` role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The Radio Node SW version is Rel16A IP12 or higher.

Installation of Credentials and LDAP Configuration on the Node.

According to procedures described in the document *18/1553-LZA 701 6014, Manage Security*, Node credential and LDAP must be configured on the Radio Node by following these steps:

Steps

1. Configure Node Credentials:

   Described in *Node Credential Enrollment - PKCS#12 Based Offline Enrollment*

   `p12` file provided at step 6 of: End Entity Creation and Credential Generation for E2E Offline Enrollment for Baseband Radio Node on page 63.

2. Install trusted CA credential onto the node:

   Described in *Trusted Certificates Management*, including `TrustCategory MO` that must reference the TrustedCertificate MOs.

pem files provided at step 7 of: End Entity Creation and Credential Generation for E2E Offline Enrollment for Baseband Radio Node on page 63

3. Configure LDAP:

   Described in *Configure LDAP:*

   LDAP parameters provided at step 2 of: LDAP Configuration for E2E Offline Enrollment for Baseband Radio Node on page 68

4. Configure Transport Layer Security (TLS):

   Described in *Configure NETCONF over TLS*

3.6 User Exposure READ ONLY for FM Model

Some parameters of `FmFunction` and `FmAlarmSupervision` cannot be changed from the **Topology Browser** application. They can only be changed from the web CLI application (alarm).

Steps

1. Log in to ENM.
2. From **Application Launcher**, click on **Topology Browser**.
3. Search for an Object.
4. Enter **`FmFunction`** or **`FmAlarmSupervision`**.
5. In **Results** table, the attributes can be read only.

Results

The attributes listed in Table 1 are `READ ONLY` on Topology Browser:

Table 1 Results Table Attributes

Class Name	Attribute Name	By Topology Browser
`FmAlarmSupervision`	`heartbeatTimeout`	READ ONLY
	`Heartbeatinterval`	READ ONLY
	`failoverCount`	READ ONLY
`FmFunction`	`currentServiceState`	READ ONLY
	`lastUpdated`	READ ONLY
	`lastUpdatedTimeStamp`	READ ONLY
	`alarmSuppressedState`	READ ONLY
	`technicianPresentState`	READ ONLY

3.7 Online Certificate Enrollment on RadioNode

Describes the steps to perform online enrollment of Radio Node.

Enrollment is a procedure that assigns to a node its credential and a set of trusted credentials. Enrollment here described is called `Online Enrollment Procedure`, as the operator starts the node enrollment through the `CMPv2` protocol, obtaining certificate from `PKI-RA`.

Target Groups

The target groups for this procedure are:

— Security Administrator.

— System Administrator.

— ENM Operator.

— Installation Engineer.

— Radio Network Engineer.

This procedure is only applicable for `OAM Enrollment`. Do not use for `IPSEC Enrollment`.

Details of the operations are available in *Manage Security, 18/1553-LZA 701 6014 Uen*.

Overview

The online procedure is based on three main steps:

— End Entity creation and Credential generation from ENM.

— LDAP Configuration from ENM.

— Node configuration.

Details for each step follow.

At the end of the procedure, the Radio Node is added in ENM and is synchronized with TLS.

3.7.1 End Entity Creation and Credential Generation for Online Certificate Enrollment on RadioNode

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System. They use it for communication with other ENM systems. End Entities must be created in the PKI system.

To generate credentials each End Entity is mapped to an Entity Profile (EP) which defines the Certificate Authority (CA).

For more information about Entity and Profiles, see *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3 Uen.

Prerequisites

The following prerequisites must be fulfilled before progressing with this procedure:

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system and is familiar with its commands.

See *Public Key Infrastructure System* of *ENM Public Key Infrastructure System Administrator Guide* 2/1543-AOM 901 151-3.

— The operator has root access privileges to log in to the management server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has ADMINISTRATOR role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The Radio Node SW version is Rel16A IP12 or higher.

Steps

1. Check entity profile:

   List all the Entity Profiles already present in ENM PKI system:

   ```
   pkiadm pfm --list -type entity
   ```

   The following highlighted profile must be available in the command output:

```
SCEPRA_IPSec_EP                          ENTITY_PROFILE
Alarm_Monitor_Client_Email_EP            ENTITY_PROFILE
DUSGen2OAM_CHAIN_EP                      ENTITY_PROFILE
MacroRBSGen1OAM_CHAIN_EP                 ENTITY_PROFILE
MicroRBSOAM_CHAIN_EP                     ENTITY_PROFILE
PicoRBSOAM_CHAIN_EP                      ENTITY_PROFILE
DUSGen2IPSec_CHAIN_EP                    ENTITY_PROFILE
MacroRBSGen1IPSec_CHAIN_EP               ENTITY_PROFILE
MicroRBSIPSec_CHAIN_EP                   ENTITY_PROFILE
PicoRBSPISec_CHAIN_EP                    ENTITY_PROFILE
```

The highlighted profile is the default profile used when `OAM Enrollment` is performed for Radio Nodes.

2. Prepare the XML File for the End Entity Creation:

   A different End Entity must be created for each Radio Node.

   The End Entities are created starting from an `*.xml` file. The following is the template for the `*.xml` file:

   Template.xml

```
<Entities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamesp →
aceSchemaLocation="EntitiesSchema.xsd">
  <Entity>
    <PublishCertificatetoTDPS>true</PublishCertificatetoTDPS>
    <EntityProfile Name="DUSGen2OAM_CHAIN_EP"/>
    <KeyGenerationAlgorithm>
      <Name>RSA</Name>
      <KeySize>2048</KeySize>
    </KeyGenerationAlgorithm>
    <Category>
      <Modifiable>true</Modifiable>
      <Name>NODE-OAM</Name>
    </Category>
    <EntityInfo>
      <Name>NetworkElementID-oam</Name>
      <Subject>
        <SubjectField>
          <Type>ORGANIZATION</Type>
          <Value>ERICSSON</Value>
        </SubjectField>
        <SubjectField>
          <Type>ORGANIZATION_UNIT</Type>
          <Value>BUCI DUAC NAM</Value>
        </SubjectField>
        <SubjectField>
          <Type>COUNTRY_NAME</Type>
          <Value>SE</Value>
        </SubjectField>
```

```
        <SubjectField>
          <Type>COMMON_NAME</Type>
          <Value>NetworkElementID-oam</Value>
        </SubjectField>
      </Subject>
    </EntityInfo>
  </Entity>
</Entities>
```

During the `*.xml` creation, the following rules must be respected:

— In the `<EntityInfo>` tag, the `<name>` must be `Network Element ID-oam`.

  The `NetworkElementID` is the identifier used at the end of this procedure, when the Radio Node is added in ENM.

— In the `<EntityInfo><SubjectField>` tag, the `<Value>` must be `NetworkElementID-oam`.

  The `NetworkElementID` is the identifier that the user must use at the end of this procedure when the Radio Node is added in ENM.

  The following file is an example of `*.xml` file used to generate the End Entity for the RadioNode with Network Element `ID=G2RBS_27`.

  EE-G2RBS_27-oam.xml

```
<Entities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noN →
amespaceSchemaLocation="EntitiesSchema.xsd">
  <Entity>
    <PublishCertificatetoTDPS>true</PublishCertificatetoTDPS>
    <EntityProfile Name="DUSGen2OAM_CHAIN_EP"/>
    <KeyGenerationAlgorithm>
      <Name>RSA</Name>
      <KeySize>2048</KeySize>
    </KeyGenerationAlgorithm>
    <Category>
      <Modifiable>true</Modifiable>
      <Name>NODE-OAM</Name>
    </Category>
    <EntityInfo>
      <Name>G2RBS_27-oam</Name>
      <Subject>
        <SubjectField>
          <Type>ORGANIZATION</Type>
          <Value>ERICSSON</Value>
        </SubjectField>
        <SubjectField>
          <Type>ORGANIZATION_UNIT</Type>
          <Value>BUCI DUAC NAM</Value>
        </SubjectField>
        <SubjectField>
          <Type>COUNTRY_NAME</Type>
          <Value>SE</Value>
        </SubjectField>
        <SubjectField>
          <Type>COMMON_NAME</Type>
          <Value>G2RBS_27-oam</Value>
        </SubjectField>
      </Subject>
    </EntityInfo>
  </Entity>
</Entities>
```

3. Save the `*.xml` file:

Name the file as `EE_Network Element ID-oam.xml`.

4. Create the End Entity:

   Drag and drop the `*.xml` file created in Step 2 into the ENM CLI app. To create the End Entity, run the following command:

   ```
   pkiadm etm -c -xf file :EE_NetworkElementID-oam.xml
   ```

5. Verify End Entity Creation:

   Verify that the End Entity has been created. List all the End Entities in the ENM PKI system with the following command:

   ```
   pkiadm etm -l - type ee
   ```

   The End Entity must be present in the list of End Entity.

   The End Entity must be created with name `NetworkElementID-oam`, and status is `NEW`. The following example shows the EE related to `Radio Node G2RBS_27`.

   Example

   ```
   91  CREDMAPIieatrcxb5553                    ENTITY          ACTIVE
   93  CREDMAPIieatrcxb4717                    ENTITY          ACTIVE
   95  G2RBS_26-ipsec                          ENTITY          ACTIVE
   94  G2RBS_26-oam                            ENTITY          ACTIVE
   96  G2RBS_27-oam                            ENTITY          NEW
   ```

6. Retrieve End Entity Subject DN:

   From `EE_NetworkElementID-oam.xml` :

   ```
   EE Subject DN:
   CN=<EntityInfo.Subject.COMMON_NAME.value>,C=<EntityInfo.Subject.COUNTRY_NAME →
   .value>,O=<EntityInfo.Subject.ORGANIZATION.value>,OU=<EntityInfo.Subject.ORG →
   ANIZATION_UNIT.value>,
   ```

   For instance, from `EE-G2RBS_27-oam.xml`:

   ```
   EE Subject DN:

   CN=G2RBS_27-oam,C=SE,O=ERICSSON,OU=BUCI DUAC NAM
   ```

7. Get Enrollment URI:

   For global-properties get the IP address of PKI RA Service (HAProxy South Bound):

```
[root@ieatclvmlms908-1 ~]# cat /ericsson/tor/data/global.properties | grep h
aproxysb
haproxysb=141.137.211.135
haproxysb_ipv6=2001:1b70:82a1:138:0:2313:5249:4a
```

Choose IPv4 or IPv6, according to Node IP stack.

8. Get Root CA Finger Print:

    a. Retrieve ENM_PKI_Root_CA active Certificate from WebCLI:

    ```
    pkiadm certmgmt CACert --exportcert --entityname ENM_PKI_Root_CA --
    format PEM
    ```

    This command downloads ENM_PKI_Root_CA active certificate in PEM format (for example, ENM_PKI_Root_CA.pem).

    b. Retrieve Issuer of ENM_PKI_Root_CA:

    ```
    openssl x509 -in ENM_PKI_Root_CA.pem -noout -issuer
    issuer= /CN=ENM_PKI_Root_CA/OU=BUCI_DUAC_NAM/C=SE/O=ERICSSON
    ```

If ENM_PKI_Root_CA is self-signed:

— Calculate SHA1 FingerPrint of ENM_PKI_Root_CA Using Openssl:

```
>openssl x509 -in ENM_PKI_Root_CA.pem -noout -fingerprint
SHA1 Fingerprint=57:FD:2A:59:36:D5:18:76:34:4D:FB:B7:98:FC:5B:15:BF:68:
19:E8
```

otherwise (case with External CA Imported), retrieve the Root CA:

— Calculate SHA1 FingerPrint of External Root CA.

Assuming, for instance, ENM_PKI_Root_CA has been signed by "O=TCS, C=IN, OU=DLF, CN=PrimeTowerIntermediateCA". - Retrieve External CA List: From WebCLI:

```
>pkiadm extcalist
```

List of Certificate(s)

External CA Name	Subject	Issuer
O=TCS, C=IN, OU=DLF, CN=PrimeTowerRootCA	O=TCS, C=IN, OU=DLF, CN=PrimeTowerRootCA	O=TCS, C=IN, OU=DLF, CN=PrimeTowerRootCA
CN=ExternalRootCA, O=TCS, L=Hyd, ST=Telangana, C=IN	CN=ExternalRootCA, O=TCS, L=Hyd, ST=Telangana, C=IN	CN=ExternalRootCA, O=TCS, L=Hyd, ST=Telangana, C=IN
CN=ExternalIntermediateCA, O=TCS, ST=Telangana, C=IN	CN=ExternalIntermediateCA, O=TCS, ST=Telangana, C=IN	CN=ExternalRootCA, O=TCS, L=Hyd, ST=Telangana, C=IN
VC_Root_CA_A1	CN=VC_Root_CA_A1, O=Ericsson, C=SE	CN=VC_Root_CA_A1, O=Ericsson, C=SE
O=TCS, C=IN, OU=DLF, CN=PrimeTowerIntermediateCA	O=TCS, C=IN, OU=DLF, CN=PrimeTowerIntermediateCA	O=TCS, C=IN, OU=DLF, CN=PrimeTowerRootCA

From this table is it possible to retrieve the ExtRoot CA. For example, if from step b the ENM_PKI_Root_CA has been signed by O=TCS, C=IN, OU=DLF, CN=PrimeTowerIntermediateCA, then the ExtRoot CA name is O=TCS, C=IN, OU=DLF, CN=PrimeTowerRootCA - Retrieve ExtCA Certificate From WebCLI:

```
>pkiadm extcaexport -n "O=TCS, C=IN, OU=DLF, CN=PrimeTowerRootCA"
```

This command downloads `PrimeTowerRootCA` active certificate in PEM format (for example PrimeTowerRootCA.pem) - calculates `SHA1 FingerPrint` of External Root CA Using `Openssl`:

```
>openssl x509 -in PrimeTowerRootCA.pem -noout -fingerprint
SHA1 Fingerprint=54:8B:D7:B9:81:E9:7D:D5:6E:3D:2D:B4:C5:A9:63:89:E9:9E:B2:26
```

3.7.2 LDAP Configuration for Online Certificate Enrollment on RadioNode

Setting LDAP Configuration and Establishing LDAP with Start TLS

Connectivity

Prerequisites

The following prerequisites must be fulfilled before progressing with this procedure:

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system and is familiar with its commands.

See *Public Key Infrastructure System* of *ENM Security System Administration Guide 2/1543-AOM 901 151*.

— The operator has root access privileges to log in to the management server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has `ADMINISTRATOR` role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The Radio Node SW version is Rel16A IP12 or higher.

Steps

1. Get data for LDAP configuration:

   To gather the `ldap` configuration information from the ENM system, use the following command:

   ```
   secadm ldap configure --manual
   ```

   A sample output by running the above command follows:

```
»secadm ldap configure --manual

PROPERTY                    VALUE
ldapIpv4Address             10.44.29.114
fallbackLdapIpv4Address     10.44.29.115
tlsPort                     1389
ldapsPort                   1636
ldapIpv6Address             2001:1b70:82a1:140:0:423:5372:28
fallbackLdapIpv6Address     2001:1b70:82a1:140:0:423:5372:29
bindPassword                bmq38gfb
bindDn                      cn=ProxyAccount_2,ou=Profiles,dc=enmms1,dc=com
baseDn                      dc=enmms1,dc=com
```

2. Gather the LDAP configuration data for Radio Node:

   This data is used to populate the MO Attributes on the radio node in the last step of this procedure.

   Following the indications reported below, collect the value for each MO Attribute listed in the first column:

MO Attribute	Value	Description
`Ldap->baseDn`	get from Step 1	Base `dn` that the client binds.
`Ldap->bindDn`	get from Step 1	Proxy account for bind.
`Ldap->bindPassword`	get from Step 1	Password for proxy account bind.
`Ldap->ldapIpAddress`	get from Step 1 (1)	The IP address of the primary LDAP server.
`Ldap->fallbackLdapIpAddress`	get from Step 1 (1)	The IP address of the fallback LDAP server. The fallback server is used when the primary server is inaccessible.
`Ldap->serverPort`	1636	Server port number to access the primary and secondary LDAP servers.
`Ldap->tlsMode`	LDAPS	Toggles the TLS establishment mode to access the primary and secondary LDAP servers. Takes effect when `useTls` is true.
`Ldap->useTls`	true	Toggle to enable TLS access to primary and secondary LDAP targets.
`Ldap->userLabel`	ENM Server	Extra descriptive text.
`Ldap->profileFilter`	ERICSSON_FILTER	Selects an LDAP filter to determine the user authorization profile. Authorization profiles are used by authorization methods. For `ERICSSON_FILTER`, the contained EricssonFilter MO must exist.
`EricssonFilter`	1	
`Filter`	1	

(1) If the node is IPv4 type, get the IPv4 `ldap` and `fallbackldap` IP addresses. If the node is IPv6 type, get the IPv6 `ldap` and `fallbackldap` IP addresses.

3.7.3 Node Configuration for Online Certificate Enrollment for RadioNode

Prerequisites

The following prerequisites must be fulfilled before progressing with this procedure:

- The operator knows about ENM server deployment structure.
- The operator knows about the PKI system and is familiar with its commands.

  Refer to the section *Public Key Infrastructure System* of *ENM Public Key Infrastructure System Administrator Guide* 2/1543-AOM 901 151-3.
- The operator has root access privileges to log in to the management server and the virtual machines.
- The operator has root access privileges to log in to the DB nodes.
- The operator has ADMINISTRATOR role to access the CLI in ENM.
- The operator knows about Configuring MOs on the Node.
- The Radio Node SW version is `Re116A IP12` or higher.

Refer to procedures in *Manage Security, 18/1553-LZA 701 6014 Uen*. Node credential and LDAP must be configured on the Radio Node by following these steps:

Steps

1. Configure CertM MOs:

   Described in *Online Enrollment*

   a. `EnrollmentAuthority MO`:

      - enrollmentCaFingerprint=<From Step 1 of LDAP Configuration for Online Certificate Enrollment on RadioNode on page 78 >.
      - enrollmentAuthorityName=" X.501 distinguished name of the issuing CA: NE_OAM_CA ".

      Example

      ```
      enrollmentCaFingerprint=57 :FD:2A: 59 : 36 :D5: 18 : 76 : 34 :4D:FB →
      :B7: 98 :FC:5B: 15 :BF: 68 : 19 :E8
      ```

```
enrollmentAuthorityName="C=SE,OU=BUCI_DUAC_NAM,O=ERICSSON,CN=NE_OAM_CA"
```

b. `EnrollmentServer MO`:

protocol=CMP URI=<From Step 7 of End Entity Creation and Credential Generation for Online Certificate Enrollment on RadioNode on page 73 >

Example

in case Node is configured with IPv4:

```
uri=http://141.137.211.135:8091/pkira-cmp/NE_OAM_CA/synch
```

Example

In case Node is configured with IPv6:

```
uri=http://2001:1b70:82a1:138:0:2313:5249:4a:8091/pkira-cmp/NE_OAM_CA/synch
```

c. `NodeCredential MO`:

SubjectName=<From Step 6 of End Entity Creation and Credential Generation for Online Certificate Enrollment on RadioNode on page 73 >.

Example

```
SubjectName=CN=G2RBS_27-oam,C=SE,O=ERICSSON,OU=BUCI DUAC NAM
```

2. Configure LDAP:

   Described in *Configure LDAP*.

   LDAP parameters provided at step 1 of LDAP Configuration for Online Certificate Enrollment on RadioNode on page 78: "Get required data for LDAP configuration"

3. Configure Transport Layer Security (TLS):

   Described in *Configure NETCONF over TLS*.

3.8 Online Certificate Enrollment on Router 6000 Family

3.8.1 OAM Enrollment

Describes the steps to perform online enrollment of Ericsson Router 6000 Series Family Nodes. Enrollment is a procedure that assigns to a node its credential and

a set of trusted credentials. Enrollment here described is called Online Enrollment Procedure, as the operator starts the node enrollment through the CMPv2 protocol, obtaining certificate from PKI-RA.

Target Groups

The target groups for this procedure are:

— Security Administrator.

— System Administrator.

— ENM Operator.

— Installation Engineer.

— Radio Network Engineer.

The procedure described here is only applicable for OAM Enrollment.

Details of operations to be performed are available in *Manage Security, 18/1553-LZA 701 6014 Uen*.

At the end of the procedure, the Router 6000 series node is added in ENM and is synchronized with TLS.

3.8.1.1 End Entity Creation and Credential Generation for Online Certificate Enrollment on Router 6000

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System. They use it for communication with other ENM systems. End Entities must be created in the PKI system.

To generate credentials each End Entity is mapped to an Entity Profile (EP) that defines the Certificate Authority (CA).

For more information about Entity and Profiles, see *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3 Uen.

Details of operations are available in the *Manage Security*, 18/1553-LZA 701 6014 Uen.

Prerequisites

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system and is familiar with its commands.

See *Public Key Infrastructure System* of *ENM Public Key Infrastructure System Administrator Guide* 2/1543-AOM 901 151-3.

— The operator has root access privileges to log in to the Management Server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has ADMINISTRATOR role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The Router6672 node type SW version is Release 17B or higher.

— The Router6675 node type SW version is Release 18A or higher.

— The Router6675 node type SW version is Release 18Q2 or higher.

— The Router6274 node type SW version is Release 18Q2 or higher.

— The Router6273 node type SW version is Release 20Q1 or higher.

— COMUser user configured preliminary to enable TLS on the node.

  See *Router 6672 Preliminary Configuration*, 1/127 01-CRA 119 2183 Uen, for additional details on Node Configurations.

— The node is SYNCHRONIZED with SSH protocol.

— Vendor credential MO is present in the node.

— The external cert supported by that VC is imported to ENM.

  See *Router6000 external CA Import Procedure* in the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3.

Steps

1. To enable TLS on the node, access the node through Node CLI, and execute the following commands in config mode:

```
SF151[config]#netconf tls server admin-state enabled
SF151[config]#default tls cipher-filter
SF151[config]#commit
```

2. Access the web-based ENM application as ADMINISTRATOR, and open the cliapp to execute the following commands:

3. List all the Entity Profiles present in ENM PKI system:

```
pkiadm pfm --list -type entity
```

Result: Verify that DUSGen2OAM_CHAIN_EP profile is listed in the command output. This is the default profile used when OAM Enrollment is performed for Router6672 Nodes.

4. Create a new XML File for each Router6000 series Node:

```
<Nodes>
    <Node>
        <NodeFdn>NetworkElement-FDN</NodeFdn>
        <EntityProfileName>DUSGen2OAM_CHAIN_EP</EntityProfileName>
        <EnrollmentMode>CMPv2_VC</EnrollmentMode>
        <KeySize>RSA_2048</KeySize>
    </Node>
</Nodes>
```

Note: The <NodeFdn> value must be NetworkElement-Fdn.

5. Import the XML file to ENM web CLI.

6. Perform online enrollment for the node:

```
secadm cert issue -ct OAM -xf file:<End_Entity_xml File>
```

Result: A link displays.

7. Click the link generated as result of step Step 6 to check the enrollment job status.

Result: If Job Status is COMPLETED and Workflow Status is SUCCESS, then the enrollment of the node is done and you can continue to the next step. If not, contact Ericsson Local Support.

8. When the node has successfully enrolled, switch it from SSH to TLS.

    a. Turn off CM supervision:

```
cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 a →
ctive=false
```

    b. Set transportProtocol in Er6000ConnectivityInformation MO to TLS:

```
cmedit set NetworkElement=<nodeName>,Er6000ConnectivityInformation= →
1 transportProtocol="TLS"
```

    c. Turn on CM supervision:

```
cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 a →
ctive=true
```

    d. Check the syncStatus:

```
cmedit get NetworkElement=<nodeName>,CmFunction=1
```

Result: The `syncStatus` attribute of `CmFunction MO` must be `SYNCHRONIZED`.

3.8.2 IPSec Enrollment

This procedure is only applicable to `IPSec Enrollment`.

Details of operations are available in *Manage Security*, *18/1553-LZA 701 6014 Uen*.

3.8.2.1 End Entity Creation and Credential Generation for Online Certificate Enrollment on Router 6000

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System. They use it for communication with other ENM systems. End Entities must be created in the PKI system.

To generate credentials each End Entity is mapped to an Entity Profile (EP) that defines the Certificate Authority (CA).

For more information about Entity and Profiles, see *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3 Uen.

Details of operations to be performed are available in: *Manage Security*, *18/1553-LZA 701 6014 Uen*.

Prerequisites

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system and is familiar with its commands.

  See *Public Key Infrastructure System* of *ENM Security System Administrator Guide*, *2/1543-AOM 901 151*.

— The operator has root access privileges to log in to the Management Server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has `ADMINISTRATOR` role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The Router6672 node type SW version is Release 17B or higher.

— The Router6675 node type SW version is Release 18A or higher.

— The Router6x71 node type SW version is Release 18A or higher.

— The Router6675 node type SW version is Release 18Q2 or higher.

— The Router6274 node type SW version is Release 18Q2 or higher.

— The Router6273 node type SW version is Release 20Q1 or higher.

— The node is SYNCHRONIZED with SSH protocol.

— Vendor credential MO is present in the node.

— The external cert supported by that VC is imported to ENM.

See *Router6000 external CA Import Procedure* in the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3.

Steps

1. Access the web-based ENM application as ADMINISTRATOR, and open the cliapp to execute the following commands:
2. Create an XML File for each Router6000 series Node:

```
<?xml version="1.0" encoding="UTF-8"?>
<Nodes>
    <Node>
       <NodeFdn>SpitFire02</NodeFdn>
       <SubjectAltName>127.0.0.1</SubjectAltName>
       <SubjectAltNameType>IP_ADDRESS</SubjectAltNameType>
       <EnrollmentMode>CMPv2_VC</EnrollmentMode>
       <KeySize>RSA_2048</KeySize>
   </Node>
</Nodes>
```

Note: The <NodeFdn> value must equal NetworkElement-Fdn.

3. Import the XML file to ENM web CLI.
4. Perform online enrollment for the node:

```
ecadm cert issue -ct IPSEC -xf file:<End Entity XML file>
```

A link displays.

5. Click the link generated as result of previous step to check the enrollment job status. If Job Status is COMPLETED, and Workflow Status is SUCCESS, then the enrollment of the node is done. If not, contact Ericsson Local Support.

3.9 E2E Offline Certificate Enrollment on Router 6000 Family

Enrollment is the procedure by which a node gets its credential, and a set of trusted credentials. Enrollment as described here is an offline enrollment procedure. The operator generates the node credential and trusted credentials manually, and installs them on the node in an offline manner.

3.9.1 OAM Enrollment

This procedure is only for `OAM Enrollment`.

At the end of the procedure, the Router 6000 series node is added in ENM and is synchronized with TLS.

Target Groups

The target groups for this procedure are:

— Security Administrator.

— System Administrator.

— ENM Operator.

— Installation Engineer.

End Entity Creation and Credential Generation for E2E Offline Enrollment for Router 6000 Family

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System. They use it for communication with other ENM systems. End Entities must be created in the PKI system.

To generate credentials each End Entity is mapped to an Entity Profile (EP) that defines the Certificate Authority (CA).

For more information about Entity and Profiles, see *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3 Uen.

Prerequisites

The following prerequisites must be fulfilled before starting this procedure:

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system and is familiar with its commands.

See *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3.

— The operator has root access privileges to log in to the management server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has `ADMINISTRATOR` role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The node is configured with the required configuration before adding the node in ENM. See *Router 6672 Preliminary Configuration* - 1/127 01-CRA 119 2183 Uen.

— The Router6672 node type SW version is Release 17B or higher.

— The Router6675 node type SW version is Release 18A or higher.

— The Router6x71 node type SW version is Release 18A or higher.

— The Router6274 node type SW version is Release 18Q2 or higher.

— The Router6273 node type SW version is Release 20Q1 or higher.

— The node is added in ENM and synchronized with SSH.

PKI Configuration Clean up on the Node

If the node already has PKI configuration, it can be cleaned up by using the following commands:

1. Check the existing configuration:

```
[local]router6000#show pki

number of node-credential: 1
name: oamNodeCredential

number of trusted-certificate: 4
name: 1
subject-name: CN=NE_OAM_CA, C=SE, O=ERICSSON, OU=BUCI_DUAC_NAM
serial number: 033EDC6ED9BE693D
name: 2
subject-name: CN=ENM_PKI_Root_CA, C=SE, O=ERICSSON, OU=BUCI_DUAC_NAM
serial number: 2C3F10C3A2527865
name: 3
subject-name: CN=ENM_Infrastructure_CA, C=SE, O=ERICSSON, OU=BUCI_DUAC_NAM
serial number: 74F4A95A1B066F7D
name: 4
subject-name: CN=ENM_OAM_CA, C=SE, O=ERICSSON, OU=BUCI_DUAC_NAM
serial number: 2F740B4641200115

number of capabilities: 1
name: 1

number of vendor-credential: 1
name: 1
```

```
number of trust-category: 1
name: oamTrustCategory

number of enrollment-authority: 1
name: 1

number of enrollment-server-group: 1
name: 1
```

2. Perform the clean up:

```
[local]router6000# config
[local]router6000(config)# no netconf tls server node-credential
[local]router6000(config)# no netconf tls server trust-category oamTrustCate →
gory
[local]router6000(config)# commit
[local]router6000(config)# no pki node-credential oamNodeCredential
[local]router6000(config)# no pki trust-category oamTrustCategory
[local]router6000(config)# no pki enrollment-authority 1
[local]router6000(config)# no pki enrollment-server-group 1
[local]router6000(config)# commit
[local]router6000(config)# end

[local]router6000# pki remove-trusted-cert 1
[local]router6000# pki remove-trusted-cert 2
[local]router6000# pki remove-trusted-cert 3
[local]router6000# pki remove-trusted-cert 4
```

Steps

1. Check Entity Profile:

   List all the Entity Profiles already present in ENM PKI system:

```
pkiadm pfm --list -type entity
```

   The highlighted profile must be available in the command output:

   The highlighted profile is the default profile used when OAM Enrollment is performed for Router 6000 nodes.

Figure 1 Default Entity Profile

2. Prepare the XML file for the End Entity Creation:

   A different End Entity must be created for each Router 6000 node in ENM.

   The End Entities are created starting from an `xml` file. The template for the `xml` file follows:

```
<Entities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="Ent →
itiesSchema.xsd">
     <Entity>
        <PublishCertificatetoTDPS>true</PublishCertificatetoTDPS>
        <EntityProfile Name="DUSGen2OAM_CHAIN_EP"/>
        <KeyGenerationAlgorithm>
            <Name>RSA</Name>
            <KeySize>2048</KeySize>
        </KeyGenerationAlgorithm>
        <Category>
             <Modifiable>true<Modifiable>
             <Name>NODE-OAM</Name>
        </Category>
        <EntityInfo>
             <Name>NetworkElementID-oam</Name>
             <Subject>
                  <SubjectField>
                       <Type>ORGANIZATION</Type>
                       <Value>ERICSSON</Value>
                  </SubjectField>
                  <SubjectField>
                       <Type>ORGANIZATION_UNIT</Type>
                       <Value>BUCI DUAC NAM</Value>
                  </SubjectField>
                  <SubjectField>
                       <Type>COUNTRY_NAME</Type>
                       <Value>SE</Value>
                  <SubjectField>
                  <SubjectField>
                       <Type>COMMON_NAME</Type>
                       <Value>NetworkElementID-oam</Value>
                  </SubjectField>
             </Subject>
         </EntityInfo>
```

```
     </Entity>
</Entities>
```

In the xml creation, the following rules must be respected:

— In the <EntityInfo> tag, the <name> must be NetworkElementID-oam.

— In the <EntityInfo><SubjectField> tag, the <Value> must be NetworkElementID-oamIn the <EntityInfo> tag, the <name> must be NetworkElementID-oam.

Note: The NetworkElementID is the identifier that the user uses at the end of this procedure, when the Router 6000 Node is added in ENM.

The following is an example of the xml used to generate the End Entity for the Router6000 with NetworkElementID=RouterOfflineEnrollment.

```
<Entities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="Ent →
itiesSchema.xsd">
    <Entity>
        <PublishCertificatetoTDPS>true</PublishCertificatetoTDPS>
        <EntityProfile Name="DUSGen2OAM_CHAIN_EP"/>
        <KeyGenerationAlgorithm>
            <Name>RSA</Name>
            <KeySize>2048</KeySize>
        </KeyGenerationAlgorithm>
        <Category>
             <Modifiable>true</Modifiable>
             <Name>NODE-OAM</Name>
        </Category>
        <EntityInfo>
        <Name>RouterOfflineEnroll-oam</Name>
             <Subject>
                 <SubjectField>
                     <Type>ORGANIZATION</Type>
                     <Value>ERICSSON</Value>
                 </SubjectField>
                 <SubjectField>
                     <Type>ORGANIZATION_UNIT</Type>
                     <Value>BUCI DUAC NAM</Value>
                 </SubjectField>
                 <SubjectField>
                     <Type>COUNTRY_NAME</Type>
                     <Value>SE</Value>
                 </SubjectField>
                 <SubjectField>
                     <Type>COMMON_NAME</Type>
                     <Value>RouterOfflineEnroll-oam</Value>
                 </SubjectField>
              </Subject>
        </EntityInfo>
     </Entity>
</Entities>
```

3. Save the xml file:

   Suggested name for the file is EE_NetworkElementID-oam.xml.

4. Create the End Entity:

   Drag and drop the xmll file created in Step 2 into the ENM CLI app, and run the following command to create the End Entity:

```
pkiadm etm -c -xf file:EE_NetworkElementID-oam.xml
```

5. Verify the End Entity Creation:

   Verify through ENM CM CLI that the End Entity has been created. List all the End Entities in the ENM PKI system with the command:

```
pkiadm etm -l -type ee
```

   The End Entity must be present in the list of End Entities. The End Entity must be created with name `NetworkElementID-oam` and its status is `NEW`. In the following example, EE is related to Router6000 node with ID `RouterOfflineEnrollment`.

Id	Entity Name	Entity Type	Entity Status
55	ENM_NETCONF_MEDIATION_svc-2-mspmip	ENTITY	ACTIVE
22	NODEDISCOVERYSERVICEsvc-2-msap	ENTITY	ACTIVE
97	RouterOfflineEnroll-oam	ENTITY	NEW

Figure 2 End Entity

6. Download of Trusted Credentials:

   For OAM connectivity with `netconf` over `TLS`, the following trusted CA credentials must be on node:

   — `NE_OAM_CA`.

   — `ENM_OAM_CA`.

   — `ENM_Infrastructure_CA`.

   — `ENM_PKI_Root_CA`.

   Run the following commands in ENM CLI to download the credentials in `.pem` format:

```
pkiadm ctm CACert -expcert -en NE_OAM_CA -f PEM
pkiadm ctm CACert -expcert -en ENM_OAM_CA -f PEM
pkiadm ctm CACert -expcert -en ENM_Infrastructure_CA -f PEM
pkiadm ctm CACert -expcert -en ENM_PKI_Root_CA -f PEM
```

7. Install the trusted credentials on the node:

   The PEM files must be placed on a server (in the following example: 10.170.118.41) and installed on the node.

```
[local]Ericsson#pki install-trusted-cert sftp://XXX@10.170.118.41/md/NE_OAM_CA.pem password XXXXXX
[local]Ericsson#pki install-trusted-cert sftp://XXX@10.170.118.41/md/ENM_PKI_Root_CA.pem password →
XXXXXX
[local]Ericsson#pki install-trusted-cert sftp://XXX@10.170.118.41/md/ENM_Infrastructure_CA.pem pas →
sword XXXXXX
[local]Ericsson#pki install-trusted-cert sftp://XXX@10.170.118.41/md/ENM_OAM_CA.pem password XXXXX →
```

```
X
```

8. Create the node credential:

   Node credential must be created on the node.

```
[local]Ericsson#config
[local]Ericsson(config)#pki node-credential RouterOfflineEnroll-oam
[local]Ericsson(pki-node-credential)#subject-name 'CN=RouterOfflineEnroll-oam, O=ERICSSON, C=SE, O
U=BUCI_DUAC_NAM'
[local]Ericsson(pki-node-credential)#key-info rsa_2048
[local]Ericsson(pki-node-credential)#commit
[local]Ericsson(pki-node-credential)#end
```

   The `subject-name` and `key-info` must be the same as the ENM entity created in Step 2, Step 3, and Step 4.

9. Create CSR to a URI:

   CSR file must be generated on the node. The CSR file is uploaded to a server (10.10.10.1 in the following example).

```
[local]Ericsson#pki node-credential RouterOfflineEnroll-oam enroll offline sftp://XXX@10.10.10.1//
home/user1/pki/pem/p1s_225_0105-1.csr password XXXXXX
```

10. Verify the enrollment progress:

    Verify that the enrollment is 60% done after the CSR file is uploaded.

```
[local]router6000#show pki node-credential Router149-oam enroll-progress
Action ID : 0
Action Name : startOfflineCsrEnrollment
Additional Info : offline enrollment
Progress Info : waiting for certificate installation
Progress Percentage : 60%
Result : not available
Result Info :
State : running
Started : 2018-01-19T08:25:30
Completed : 1900-01-01T00:00:00
Last Update : 2018-01-19T08:25:30
```

11. End Entity Credential Generation:

    Download the previous CSR file and drag and drop into the ENM CLI app and then execute the following command to generate the End Entity credential in PEM format.

```
pkiadm ctm EECert -gen -en NetworkElementID-oam -csr file:<csr-file> -f PEM
```

    In the command, `NetworkElementID-oam` is the name of the entity created in Step 4.

The command downloads a PEM file with the name `NetworkElementID-oam.pem`.

12. Install the node certificate after the CSR was signed by CA:

    The PEM file downloaded at Step 10 must be uploaded to a server (10.10.10.1 in the following example) and installed on the node using the following command.

```
[local]Ericsson#pki node-credential RouterOfflineEnroll-oam install-cert sftp://XXX@10.10.10.1/hom →
e/user1/pki/pem/RouterOfflineEnroll-oam.pem password XXXXXX
```

13. Verify the enrollment progress on the node:

    Verify that the enrollment is 100% done after the certificate is installed.

```
[local]router6000#show pki node-credential RouterOfflineEnroll-oam enroll-progress
Action ID : 0
Action Name : startOfflineCsrEnrollment
Additional Info : offline enrollment
Progress Info : notifying event
Progress Percentage : 100%
Result : success
Result Info : success
State : finished
Started : 2018-01-19T08:25:30
Completed : 2018-01-19T08:28:06
Last Update : 2018-01-19T08:28:06
```

14. Configure `COMUser` on the node (if not configured):

    `COMUser` must be configured on the node.

```
[local]router6000#configure
[local]router6000(config)#context local
[local]router6000(config-ctx)#administrator COMUser password ********
[local]router6000(config-administrator)#role NetconfPlatformAdministrator
[local]router6000(config-administrator)#commit
```

15. Configure `trust-category` on the node (if not existing):

    Configure `trust-category` on the node.

```
[local]router6000#config
[local]router6000(config)# pki trust-category oamTrustCategory trusted-certificate 1,2,3,4
[local]router6000(config)#commit
Transaction committed.
[local]router6000(config)#end
```

16. Configure TLS on the node:

```
[local]router6000#config
[local]router6000(config)#netconf tls server admin-state enabled
[local]router6000(config)#netconf tls server trust-category oamTrustCategory
[local]router6000(config)#netconf tls server node-credential RouterOfflineEnroll-oam
```

```
[local]router6000(config)#commit
Transaction committed.
[local]router6000(config)#end
```

17. Switch the node to TLS in ENM:

    As the node is successfully enrolled offline, switch it from SSH to TLS:

    — Turn off CM supervision:

    ```
    cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 activ →
    e=false
    ```

    — Set `transportProtocol` in `Er6000ConnectivityInformation MO` to TLS:

    ```
    cmedit set NetworkElement=<nodeName>,Er6000ConnectivityInformation=1 tr →
    ansportProtocol="TLS"
    ```

    — Turn on CM supervision:

    ```
    cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 activ →
    e=true
    ```

    — Check the `syncStatus` of `CmFunction MO` is `SYNCHRONIZED`:

    ```
    cmedit get NetworkElement=<nodeName>,CmFunction=1
    ```

3.9.2 IPSec Enrollment

This procedure is only for `IPSec Enrollment`.

Target Groups

The target groups for this procedure are:

— Security Administrator.

— System Administrator.

— ENM Operator.

— Installation Engineer.

End Entity Creation and Credential Generation for E2E Offline Enrollment for Router 6000 Family

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System. They use it for communication with other ENM systems. End Entities must be created in the PKI system.

To generate credentials each End Entity is mapped to an Entity Profile (EP) that defines the Certificate Authority (CA).

For more information about Entity and Profiles, see *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3 Uen.

Prerequisites

The following prerequisites must be fulfilled before progressing with this procedure:

— The operator knows about ENM server deployment structure.

— The operator knows about the PKI system and is familiar with its commands.

  See *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide*, 2/1543-AOM 901 151-3.

— The operator has root access privileges to log in to the management server and the virtual machines.

— The operator has root access privileges to log in to the DB nodes.

— The operator has `ADMINISTRATOR` role to access the CLI in ENM.

— The operator knows about Configuring MOs on the Node.

— The node has the required configuration before adding the node in ENM, see: *Router 6672 Preliminary Configuration* - 1/127 01-CRA 119 2183 Uen.

— The Router6672 node type SW version is Release 17B or higher.

— The Router6675 node type SW version is Release 18A or higher.

— The Router6x71 node type SW version is Release 18A or higher.

— The Router6273 node type SW version is Release 20Q1 or higher.

— The node is added in ENM and synchronized with SSH.

Steps

1. Check Entity Profile:

Using ENM CLI, list the Entity Profiles already present in ENM PKI system:

```
pkiadm pfm --list -type entity
```

The highlighted profile must be available in the command output:

Id	Profile Name
2	ENM_PKI_Root_CA_EP
5	NE_CA_EP
6	ENM_CA_EP
8	ENM_System_CA_EP
41	ENM_System_SBI_COM_TPFC_EP
42	ENM_External_Server_EP
43	Alarm_Monitor_Client_Email_EP
44	DUSGen2IPSec_SAN_CHAIN_EP

Figure 3 Entity Profile

The highlighted profile is the default profile used when `OAM Enrollment` is performed for Router 6000 nodes.

2. Prepare the XML file for End Entity Creation:

   A different End Entity must be created for each Router 6000 node.

   The End Entities are created starting from an `xml` file. The following is the template for the `xml` file:

```
<Entities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamesp →
aceSchemaLocation="EntitiesSchema.xsd">
   <Entity>
       <PublishCertificatetoTDPS>true</PublishCertificatetoTDPS>
           <EntityProfile Name="DUSGen2IPSec_SAN_CHAIN_EP"/>
           <KeyGenerationAlgorithm>
                 <Name>RSA</Name>
                 <KeySize>2048</KeySize>
           </KeyGenerationAlgorithm>
           <Category>
                 <Modifiable>true</Modifiable>
                 <Name>NODE-IPSEC</Name>
           </Category>
           <EntityInfo>
                 <Name>NetworkElementID-ipsec</Name>
                 <Subject>
                      <SubjectField>
                           <Type>ORGANIZATION</Type>
                           <Value>ERICSSON</Value>
                      </SubjectField>
                      <SubjectField>
                           <Type>ORGANIZATION_UNIT</Type>
                           <Value>BUCI DUAC NAM</Value>
                      </SubjectField>
                      <SubjectField>
                           <Type>COUNTRY_NAME</Type>
                           <Value>SE</Value>
                      </SubjectField>
```

```
                    <SubjectField>
                         <Type>COMMON_NAME</Type>
                         <Value>NetworkElementID-ipsec</Value>
                    </SubjectField>
               </Subject>
       </EntityInfo>
   </Entity>
</Entities>
```

In XML creation, the following rules must be respected:

— In the <EntityInfo><SubjectField> tag, the <Value> must be NetworkElementID-ipsec

Following is an example of XML used to generate the End Entity for the Router6000 with NetworkElementID=RouterOfflineEnrollment:

```
In the
<Entities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamesp →
aceSchemaLocation="EntitiesSchema.xsd">
    <Entity>
        <PublishCertificatetoTDPS>true</PublishCertificatetoTDPS>
              <EntityProfile Name="DUSGen2IPSec_SAN_CHAIN_EP"/>
              <KeyGenerationAlgorithm>
                   <Name>RSA</Name>
                   <KeySize>2048</KeySize>
              </KeyGenerationAlgorithm>
              <Category>
                   <Modifiable>true</Modifiable>
                   <Name>NODE-OAM</Name>
              </Category>
              <EntityInfo>
                    <Name>RouterOfflineEnroll-ipsec</Name>
                    <Subject>
                          <SubjectField>
                              <Type>ORGANIZATION</Type>
                              <Value>ERICSSON</Value>
                          </SubjectField>
                          <SubjectField>
                               <Type>ORGANIZATION_UNIT</Type>
                               <Value>BUCI DUAC NAM</Value>
                          </SubjectField>
                          <SubjectField>
                                <Type>COUNTRY_NAME</Type>
                                <Value>SE</Value>
                          </SubjectField>
                          <SubjectField>
                                <Type>COMMON_NAME</Type>
                                <Value>RouterOfflineEnroll-ipsec</Value>
                          </SubjectField>
                    </Subject>
         </EntityInfo>
    </Entity>
</Entities>
```

3. Save the XML file:

   Suggested name for the file is EE_NetworkElementID-ipsec.xml.

4. Create the End Entity:

   Drag and drop the XML file created in Step 2 into the ENM CLI app, and run the following command:

```
pkiadm etm -c -xf file:EE_NetworkElementID-ipsec.xml
```

5. Verify End Entity Creation:

   Verify that the End Entity has been created by listing all the End Entities in the ENM PKI system with the following command:

   ```
   pkiadm etm -l -type ee
   ```

   The End Entity must be present in the list of End Entities. The End Entity must be created with name `NetworkElementID-ipsec` and its `status` is `NEW`. In the following example, EE related to Router6000 node with ID `RouterOfflineEnrollment`.

98	p1s97-oam	ENTITY	ACTIVE
101	RouterOfflineEnroll-ipsec	ENTITY	ACTIVE
102	p1s97-ipsec	ENTITY	ACTIVE

   Figure 4 End Entity

6. Download Trusted Credentials:

   For OAM connectivity with `netconf` over TLS, the following trusted CA credentials must be on node:

   — `NE_IPsec_CA`.

   — `ENM_Infrastructure_CA`.

   — `ENM_PKI_Root_CA`.

   Run the following commands in ENM CLI to download the credentials in `.pem` format:

   ```
   pkiadm ctm CACert -expcert -en NE_IPsec_CA -f PEM
   pkiadm ctm CACert -expcert -en ENM_Infrastructure_CA -f PEM
   pkiadm ctm CACert -expcert -en ENM_PKI_Root_CA -f PEM
   ```

7. Install the trusted credentials on the node:

   The PEM files must be placed on a server (in the following example: 10.170.118.41) and installed on the node.

   ```
   [local]Ericsson#pki install-trusted-cert sftp://XXX@10.170.118.41/md/NE_IPse →
   c_CA.pem password XXXXXX
   [local]Ericsson#pki install-trusted-cert sftp://XXX@10.170.118.41/md/ENM_PKI →
   _Root_CA.pem password XXXXXX
   [local]Ericsson#pki install-trusted-cert sftp://XXX@10.170.118.41/md/ENM_Inf →
   rastructure_CA.pem password XXXXXX
   ```

8. Create the node credential:

   Node credential must be created on the node.

```
[local]Ericsson#config
[local]Ericsson(config)#pki node-credential ipsecNodeCredential
[local]Ericsson(pki-node-credential)#subject-name 'CN=RouterOfflineEnroll-ip →
sec, O=ERICSSON, C=SE, OU=BUCI_DUAC_NAM'
[local]Ericsson(pki-node-credential)#key-info rsa_2048
[local]Ericsson(pki-node-credential)#commit
[local]Ericsson(pki-node-credential)#end
```

The `subject-name` and `key-info` must be the same as the ENM entity created in Step 2, Step 3, and Step 4.

9. Create CSR to a URI:

   CSR file must be generated on the node. The CSR file is uploaded to a server (10.10.10.1 in the following example).

```
[local]Ericsson#pki node-credential ipsecNodeCredential enroll offline sftp: →
//XXX@10.10.10.1//home/user1/pki/pem/p1s_225_0105-1.csr password XXXXXX
```

10. Verify the enrollment progress:

    Verify that the enrollment is 60% done after the CSR file is uploaded.

```
[local]router6000#show pki node-credential ipsecNodeCredential enroll-progre →
ss
Action ID : 0
Action Name : startOfflineCsrEnrollment
Additional Info : offline enrollment
Progress Info : waiting for certificate installation
Progress Percentage : 60%
Result : not available
Result Info :
State : running
Started : 2018-01-19T08:25:30
Completed : 1900-01-01T00:00:00
Last Update : 2018-01-19T08:25:30
```

11. End Entity Credential Generation:

    Download the CSR file, and drag and drop it into the ENM CLI app. Then execute the following command to generate the End Entity credential in `PEM` format.

```
pkiadm ctm EECert -gen -en NetworkElementID-ipsec -csr file:<csr-file> -f PE →
M
```

    `NetworkElementID-ipsec` is the name of the entity created in Step 4.

    The command downloads a `PEM` file with the name `NetworkElementID-ipsec.pem`.

12. Install the node certificate after the CSR is signed by `CA`:

The PEM file downloaded at Step 10 must be uploaded to a server (10.10.10.1 in the following example) and be installed on the node using the following command:

```
[local]Ericsson#pki node-credential ipsecNodeCredential install-cert sftp://
XXX@10.10.10.1/home/user1/pki/pem/RouterOfflineEnroll-ipsec.pem password XXX
XXX
```

13. Verify the enrollment progress:

    Verify that the enrollment is 100% done after the certificate is installed:

```
[local]router6000#show pki node-credential ipsecNodeCredential enroll-progre
ss
Action ID : 0
Action Name : startOfflineCsrEnrollment
Additional Info : offline enrollment
Progress Info : notifying event
Progress Percentage : 100%
Result : success
Result Info : success
State : finished
Started : 2018-01-19T08:25:30
Completed : 2018-01-19T08:28:06
Last Update : 2018-01-19T08:28:06
```

14. The IPSec enrollment is successful.

3.10 LDAP Configuration for Router 6000 Family

Prerequisites

- Router6672, Router6675, Router6x71 node SW version must be 18A or higher.
- Router6274 node S/W version must be 18Q2 or higher.
- Router6273 node S/W version must be 20Q1 or higher.
- `NetworkElementSecurity` MO contains empty `ldapAuthenticatioUserNamel` and `ldapAuthenticationUserPassword`.
- Router6000 series nodes are in SYNC state.
- Online / Offline Security enrollment must be completed.

To perform online enrollment, see: *Online Certificate Enrollment on Router 6000 Family*, 1/1553-AOM 901 151 Uen.

To perform offline enrollment, see: *E2E Offline Certificate Enrollment on Router 6000 Family*, 1/1553-AOM 901 151 Uen.

Steps

1. Prerequisite verification:

    a. Verify that the following MO contains empty `ldapAuthenticatioUserName` and `ldapAuthenticationUserPassword`:

    ```
    »cmedit get MeContext=<NodeName>,ManagedElement=1,SystemFunctions=1 →
    ,SecM=1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1
    FDN : MeContext=NodeName,ManagedElement=NodeName,SystemFunctions=1, →
    SecM=1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1
    baseDn :
    bindDn : null
    bindPassword : {password=, cleartext=true}
    fallbackLdapIpAddress : null
    ldapId : 1
    ldapIpAddress :
    nodeCredential : null
    profileFilter : null
    serverPort : null
    tlsMode : LDAPS
    trustCategory : null
    useReferrals : false
    userLabel : null
    useTls : true
    ```

    b. Verify that LDAP authentication is not active:

    ```
    »cmedit get MeContext=<NodeName>,ManagedElement=1,SystemFunctions=1 →
    ,SecM=1,UserManagement=1,LdapAuthenticationMethod=1
    FDN : MeContext=RouterLDAP,ManagedElement=1,SystemFunctions=1,SecM= →
    1,UserManagement=1,LdapAuthenticationMethod=1
    administrativeState : LOCKED
    ldapAuthenticationMethodId : 1
    ```

    c. Verify that the node is in Sync:

    ```
    »cmedit get NetworkElement=<NodeName>,CmFunction=1
    ```

2. Configure LDAP:

    a. Create an XML file (`NodeName_ldapConfig_LDAPS.xml`), using the following template, to configure LDAP on the node from ENM:

    ```
    <Nodes>
       <Node>
          <nodeFdn>NetworkElement=NodeName</nodeFdn>
          <tlsMode>LDAPS</tlsMode>
          <userLabel>ENM</userLabel>
          <useTls>true</useTls>
       </Node>
    </Nodes>
    ```

    b. Drag and drop the XML file created above into the ENM CLI app and run the following command to configure LDAP:

```
secadm ldap configure --xmlfile file:"NodeName_ldapConfig_LDAPS.xml
"
```

c. Verify that LDAP is correctly configured using the following command:

```
»cmedit get MeContext=<NodeName>,ManagedElement=1,SystemFunctions=1
,SecM=1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1
```

d. Verify that the value for the `profileFilter` attribute is set to `ERICSSON_FILTER`. If not, then set it using the below command:

```
»cmedit set MeContext=<NodeName>,ManagedElement=1,SystemFunctions=1
,SecM=1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1 profileF
ilter=ERICSSON_FILTER
```

e. Configure `trustCategory` and `nodeCredential` on LDAP MO:

Note: Only if offline enrollment is performed.

```
»cmedit set MeContext=<NodeName>,ManagedElement=1,SystemFunctions=1
,SecM=1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1 trustCat
egory=<fdn of trustCategory MO>

»cmedit set MeContext=<NodeName>,ManagedElement=1,SystemFunctions=1
,SecM=1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1 nodeCred
ential=<fdn of node-credential MO>
```

f. Set the administrative state of `LdapAuthenticationMethod` MO to `UNLOCKED` to activate LDAP authentication on the node.

```
»cmedit set MeContext=<NodeName>,ManagedElement=1,SystemFunctions=1
,SecM=1,UserManagement=1,LdapAuthenticationMethod=1 administrativeS
tate=UNLOCKED
```

3. Enable LDAP user and toggle CM Supervision in ENM:

a. Stop CM Supervison:

```
»cmedit set NetworkElement=<NodeName>,CmNodeHeartbeatSupervision=1a
ctive=false
```

b. Verify that the node is in `UNSYNCRONIZED` state:

```
»cmedit get NetworkElement=<NodeName>,CmFunction=1
```

c. Enable `ldapUser` for the node so that ENM will use `ldapApplicationUser` for authentication purposes:

```
»secadm credentials update --ldapuser enable --nodelist <NodeName>
```

d. Verify that the ldapApplicationUserName and ldapApplicationUserPassword fields are compiled:

```
»cmedit get NetworkElement=<NodeName>,SecurityFunction=1,NetworkEle →
mentSecurity=1
```

e. Start CM Supervison:

```
»cmedit set NetworkElement=<NodeName>,CmNodeHeartbeatSupervision=1a →
ctive=true
```

f. Verify that the node is in SYNC status so that LDAP authentication on the node is successful:

```
»cmedit get NetworkElement=<NodeName>,CmFunction=1
```

3.11 Configure LDAP on BSC Node in SSH

In the following examples, <nodename> is the name of the node, and <ossprefix> is the configured OSS prefix of the node, as read with the following ENM web CLI command:

```
cmedit get <nodename> NetworkElement.ossPrefix
FDN : NetworkElement=<nodename>
ossPrefix : <ossprefix>
```

Prerequisites

— BSC node is SYNCHRONIZED.

— BSC node credentials created under ENM with Secure user equal to BSC node Troubleshooter (TS) user and empty ldapApplication user.

— The operator must have NodeSecurity_Administrator role and Cmedit_Administrator role to access the ENM web CLI.

Steps

1. Verify Prerequisites (ENM web CLI):

    a. Verify that node is SYNCHRONIZED:

    ```
    cmedit get <nodename> CmFunction.syncStatus
    ```

    b. Verify that node credentials have been created under ENM, with Secure user equal to node Troubleshooter (TS) user with the command:

```
secadm credentials get --plaintext show --nodelist <nodename>
```

`secureUserName` and `secureUserPassword` must match the node TS `username` and `password` configured locally into the node.

Verify `NetworkElementSecurity` MO contains empty `ldapApplicationUserName` and `ldapApplicationUserPassword` with the command:

```
cmedit get <nodename> NetworkElementSecurity.(ldapApplicationUserNa
me, ldapApplicationUserPassword)
FDN : NetworkElement=<nodename>,SecurityFunction=1,NetworkElementSe
curity=1
ldapApplicationUserName :
ldapApplicationUserPassword :
```

Note: In above case if Prerequisite Verification is not true, `secureUserName`, `secureUserPassword`, `ldapAuthenticatioUserName`, and `ldapAuthenticationUserPassword` must be set to the correct values with the following commands:

Disable CM Heartbeat Supervision:

```
cmedit set NetworkElement=<nodename>,CmNodeHeartbeatSupervision=1 a
ctive=false
```

Verify that node is UNSYNCHRONIZED:

```
cmedit get <nodename> CmFunction.syncStatus
```

Update Secure user and ldapApplication user credentials:

```
secadm credentials update --secureusername <ts_username> --secureus
erpassword <ts_password> --ldapuser disable --nodelist <nodename>
```

Enable CM Heartbeat Supervision:

```
cmedit set NetworkElement=<nodename>,CmNodeHeartbeatSupervision=1 a
ctive=true
```

Verify that node is SYNCHRONIZED:

```
cmedit get <nodename> CmFunction.syncStatus
```

2. OAM Certificate Issue (ENM web CLI):

    a. Use the following file to issue the OAM certificate on BSC node (put the correct <nodename> inside the file).

    Filename: <nodename>-OAM.xml

Template.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Nodes>
 <Node>
 <NodeFdn><nodename></NodeFdn>
 </Node>
</Nodes>
```

b. Drag and drop the <nodename>-OAM.xml file onto the ENM web CLI, then perform the following command:

```
»secadm certificate issue --certtype OAM --xmlfile file:"<nodename> →
-OAM.xml"
```

The following output must be displayed:

```
Successfully started a job to issue certificates for n →
odes. Perform 'secadm job get -j <jobId number>' to ge →
t progress info.
```

As described by the output of the command, run the suggested command to monitor the workflow progress until the "Job Status" is `COMPLETED`.

When the "Job Status" is `COMPLETED`, verify that, in the same command output, the "Workflow Status" is `SUCCESS`.

3. Configure LDAP and set its Administrative State to `UNLOCKED` (ENM web CLI):

   a. Get `Ldap.bindDn` attribute value:

```
cmedit get <nodename> Ldap.bindDn
FDN : SubNetwork=<ossprefix>,ManagedElement=<nodename>,SystemFuncti →
ons=1,SecM=1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1
bindDn : <ldapBindDn>
```

The `Ldap.bindDn` value is used for further checks on LDAP configuration result.

   b. Use the following file to configure LDAP (put the correct <nodename> inside the file).

Filename: <nodename>_LDAPS.xml

Template.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Nodes>
    <Node>
        <nodeFdn>NetworkElement=<nodename></nodeFdn>
        <tlsMode>LDAPS</tlsMode>
        <userLabel>ENM</userLabel>
        <useTls>true</useTls>
```

```
    </Node>
</Nodes>
```

c. Drag and drop the <nodename>-LDAPS.xml file onto the ENM web CLI, then perform the following command:

```
secadm ldap configure --xmlfile file:"<nodename>_LDAPS.xml"
```

d. Verify that the LDAP is correctly configured:

```
cmedit get <nodename> Ldap.bindDn
```

The current `Ldap.bindDn` value must be different from the previous one.

e. Get the LDAP administrative status:

```
cmedit get <nodename> LdapAuthenticationMethod.administrativeState
```

If the administrativeState is `LOCKED`, set it to `UNLOCKED`:

```
cmedit set <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1, →
SecM=1,UserManagement=1,LdapAuthenticationMethod=1 administrativeSt →
ate=UNLOCKED
```

4. Disable CM Heartbeat Supervision (ENM web CLI):

a. Stop the CM heartbeat supervision for the node:

```
cmedit set NetworkElement=<nodename>,CmNodeHeartbeatSupervision=1 a →
ctive=false
```

b. Verify that the node is `UNSYNCHRONIZED`:

```
cmedit get <nodename> CmFunction.syncStatus
```

5. Update Node Credentials (ENM web CLI):

a. Create (if not existing) a User Role named `CpRole0` with Role Type `COM Role`.

b. Create (if not existing) an ENM User to be used as Secure user:

The ENM User can be defined once in ENM and used as Secure user for multiple BSC nodes.

The ENM User must have the following roles:

— SystemAdministrator.

— SystemSecurityAdministrator.

— EricssonSupport.

— BscApplicationAdministrator.

— SystemReadOnly.

— CpRole0.

For all the roles, the Target Group must be set to ALL.

The ENM User must have the Password Aging disabled.

The ENM User must have the Force Password Change disabled.

c. Update node credentials with ldapApplication user enabled and with ENM User credentials as Secure user credentials:

```
secadm credentials update --secureusername <ENM User username> --se →
cureuserpassword %lt;ENM User password> --ldapuser enable --nodelis →
t <nodename>
```

d. Verify that node credentials have been updated under ENM with Secure user equal to ENM User with the command:

```
secadm credentials get --plaintext show --nodelist <nodename>
```

secureUserName and secureUserPassword must match the ENM User username and password.

e. Verify that the ldapApplicationUserName and ldapApplicationUserPassword fields are not empty:

```
cmedit get <nodename> NetworkElementSecurity.(ldapApplicationUserNa →
me, ldapApplicationUserPassword)
FDN : NetworkElement=<nodename>,SecurityFunction=1,NetworkElementSe →
curity=1
ldapApplicationUserName : <ldapApplication username>
ldapApplicationUserPassword : <ldapApplication password>
```

6. Enable CM Heartbeat Supervision (ENM web CLI):

a. Start the CM heartbeat supervision for the node:

```
cmedit set NetworkElement=<nodename>,CmNodeHeartbeatSupervision=1 a →
ctive=true
```

b. Verify that the node is SYNCHRONIZED:

```
cmedit get <nodename> CmFunction.syncStatus
```

3.12 Launch EM with SSO for Generic ESA

This section describes the procedure to launch EM with SSO for Generic ESA node.

The generic ESA target type is used for adding any type of ESA node in ENM. All the ESA type of nodes might support or might not support EM GUI and even if EM GUI is supported, SSO might be supported or might not be supported.

Prerequisites

- Configure users to Access AMOS and Element Manager towards SL2 or TLS Enabled nodes.
- The user must have `ssu` to launch an AMOS or EM session towards a Security Level 2 (SL2), for CPP nodes, or Transport Layer Security (TLS), and for nodes supporting ECIM. The `ssu` credentials are generated when the user logs on to AMOS for the first time. For the `ssu` credentials to be generated, an entity for each user must be created in PKI. See *section 2.3.9* in ENM Security Administrator Guide for more information.
- User must have Scripting operator role.
- User must have EM operator role.
- Install a certificate on the node to obtain a proper handshake between ENM and node.

Following is the procedure for certificate generation:

Steps

1. To create Entity, navigate to **PKI Entity Management**. Create a new **END** entity with the following details and save the Entity.

Option	Description
Entity name	<Node server hostname>
Entity Category	NODE-OAM
Entity Profile	DUSGen2OAM_CHAIN_EP
Key Generation Algorithm	RSA-2048
ORGANIZATION_UNIT	BUCI DUAC NAM
COMMON_NAME	<Entity name>
COUNTRY_NAME	SE
ORGANIZATION	ERICSSON
Password	<password>

Option	Description
Re-Type Password	<password>

2. Select the created Entity and click **Issue**. Select the format which needs to be downloaded.
3. Provide the credentials and click **Issue**.

   A pop-up window appears asking for the file path to be saved. After selecting the path, the End Entity Certificate along with Chain is downloaded.See *Node CPI* for installation of the generated certificate on node (which is updated once it is available from the node team).

   Note: EM option is shown for all ESA type of nodes irrespective of whether it is supported from node side or not.

3.13 Configure HLR/MSC with TLS

Describes the procedure to configure HLR/MSC with TLS as transport protocol, when following nodes are not added as part of migration, but are added into ENM using GUI or CLI.

Supported Variants:

— Compact CTC

— Compact IP-STP

— Compact MSC-S

— Compact vCTC

— Compact vIP-STP

— Compact vMSC-S

— CTC BC (BSP)

— CTC BC (IS)

— CTC DB

— HLR-FE (APZ)

— HLR-FE (BSP)

— HLR-FE (IS)

— IP-STP

— MSC-S BC (BSP)

— MSC-S BC (IS)

— MSC-S DB

— vCTC HC

— vHLR-FE

— vMSC-S HC

Prerequisites

— The user has the following COM roles to access or modify specific MOs under Node HLP/MSC.

  • SystemAdministrator.

  • SystemSecurityAdministrator.

  • EricssonSupport.

  • SystemReadOnly.

  • CpRole0.

— Ensure that Assign Target Groups (which contains all or specific nodes) is set to all COM Roles, and must be configured first with `targetType` under `UserManagement=1` MO of node HLR/MSC.

Steps

1. Add Node with TLS in ENM:

   To add a node in ENM, launch the ENM CLI application for "HLR-FE" or "MSC" node types. See the appropriate online help under section *Add Node* for respective node type.

   Or

   The Add Node application can be used from ENM GUI to add the node.

   Note: Even if `port` and `transportProtocol` are not mentioned in the CLI commands, the node is added with default port and TLS as `transportProtocol`.

2. OAM Certificate Enrollment and LDAP Configuration (Node CLI):

   To perform Certificate enrollment and LDAP configuration see the appropriate Offline Enrollment Procedure section for the node in question in

the ENM Network Security Configuration System Administrator Guide 2/1543-AOM 901 151-2.

3. Put CliTls `administrativeState=UNLOCKED` (Node CLI):

   Enable the TLS communication using the provided certificates.

   This is achieved by setting the attribute `administrativeState` in MOs CliTls=1 to value `UNLOCKED`.

   If the attribute is reset to `LOCKED`, all related opened sessions - AP session (MML session) are automatically closed.

   a. Get the correct 'nodeCredential' and 'trustCategory' references from LDAP MO as configured in Step 2:

      Navigate to `Ldap=1` MO of the Node by tabbing out each comma-separated MO.

```
>dn -m Ldap
ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,
LdapAuthenticationMethod=1,Ldap=1
(Ldap=1)>show -v
Ldap=1
baseDn=" dc=ieatlms5589,dc=com"
bindDn="cn=ProxyAccount_19,ou=proxyagent,ou=com,dc=ieatlms5589,dc=c
om"
nodeCredential="ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,C
ertM=1,NodeCredential=1"
profileFilter=ERICSSON_FILTER
serverPort=1636
tlsMode=LDAPS
trustCategory="ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,Ce
rtM=1,TrustCategory=1"
useReferrals=false <default>
...
```

   b. Navigate to the `CliTls=1` MO of the Node by tabbing out each comma-separated MO and update `nodeCredential` MO, `TrustCategory MO, and administrativeState`:

```
>dn -m CliTls
ManagedElement =<NE_NAME>,SystemFunctions=1,SysM=1,CliTls=1
(CliTls=1)>configure
(config-CliTls=1)>show -v
CliTls=1
administrativeState=LOCKED
cliTlsId="1"
nodeCredential=[] <empty>
trustCategory=[] <empty>
(config-CliTls=1)>nodeCredential="ManagedElement=<NodeName>,SystemF
unctions=1,SecM=1,CertM=1,NodeCredential=1"
(config-CliTls=1)>trustCategory="ManagedElement=<NodeName>,SystemFu
nctions=1,SecM=1,CertM=1,TrustCategory=1"
(config-CliTls=1)>administrativeState=UNLOCKED
(config-CliTls=1)>commit -s
(config-CliTls=1)>show
CliTls=1
administrativeState=UNLOCKED
nodeCredential="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,
CertM=1,NodeCredential=1"
trustCategory="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,C
ertM=1,TrustCategory=1"
(config-CliTls=1)>up
```

4. Update the Credentials (ENM web CLI):

    a. Create, if not existing, a User Role named `CpRole0` with Role Type of `COM Role`:

    b. Create (if not existing) an ENM User to be used as Secure user:

       The ENM User can be defined once in ENM and used as Secure user for multiple HLR/MSC nodes.

       The ENM User must have the following roles:

       — SystemAdministrator.

       — SystemSecurityAdministrator.

       — EricssonSupport.

       — SystemReadOnly.

       — CpRole0.

       For all these roles, the Target Group must be set to`ALL` .

       The ENM User must have `Password Aging` disabled.

       The ENM User must have `Force Password Change` disabled.

    c. Update node credentials with `ldapApplication` user enabled and with ENM User credentials as Secure user credentials:

       ```
       »secadm credentials update --secureusername <ENM User username> --s →
       ecureuserpassword <ENM User password> --ldapuser enable --nodelist →
       <nodename>
       ```

    d. Verify that node credentials have been updated under ENM with Secure user equal to ENM User with the command:

       ```
       »secadm credentials get --plaintext show --nodelist <nodename>
       ```

       `secureUserName` and `secureUserPassword` must match the ENM User `username` and `password`.

    e. Verify that the `ldapApplicationUserName` and `ldapApplicationUserPassword` fields are not empty:

       ```
       »cmedit get <nodename> NetworkElementSecurity.(ldapApplicationUserN →
       ame, ldapApplicationUserPassword)
       ```

       FDN:
       NetworkElement=<nodename>,SecurityFunction=1,NetworkElementSecurity=1

ldapApplicationUserName : <ldapApplication username>

ldapApplicationUserPassword : <ldapApplication password>

3.14 Move BSC Node from SSH to TLS

In the following examples, <nodename> is the name of the node, and <ossprefix> is the configured OSS prefix of the node, as read with the following ENM web CLI command:

```
cmedit get <nodename> NetworkElement.ossPrefix
FDN : NetworkElement=<nodename>
ossPrefix : <ossprefix>
```

Prerequisites

— BSC node is `SYNCHRONIZED`.

— BSC node credentials created under ENM with Secure user equal to BSC node Troubleshooter (TS) user and empty `ldapApplication` user.

— The operator must know the credentials of BSC node Troubleshooter (TS) user.

— The operator must know the credentials of BSC node Root user.

— The operator must have `NodeSecurity_Administrator` Role and `Cmedit_Administrator` Role to access the ENM web CLI.

Steps

1. Verify Prerequisites (ENM web CLI):

    a. Verify that node is `SYNCHRONIZED`:

    ```
    cmedit get <nodename> CmFunction.syncStatus
    ```

    b. Verify that node credentials have been created under ENM with Secure user equal to node Troubleshooter (TS) user with the command:

    ```
    secadm credentials get --plaintext show --nodelist <nodename>
    ```

    `secureUserName` and `secureUserPassword` must match the node TS `username` and `password` configured locally into the node.

    Verify `NetworkElementSecurity` MO contains empty `ldapApplicationUserName` and `ldapApplicationUserPassword` with the command:

```
cmedit get <nodename> NetworkElementSecurity.(ldapApplicationUserNa
me, ldapApplicationUserPassword)
FDN : NetworkElement=<nodename>,SecurityFunction=1,NetworkElementSe
curity=1
ldapApplicationUserName :
ldapApplicationUserPassword :
```

Note: In above case if Prerequisite Verification is not true, `secureUserName`, `secureUserPassword`, `ldapAuthenticatioUserName`, and `ldapAuthenticationUserPassword` must be set to the correct values with the following commands.

Disable CM Heartbeat Supervision:

```
cmedit set NetworkElement=<nodename>,CmNodeHeartbeatSupervision=1 a
ctive=false
```

Verify that node is UNSYNCHRONIZED:

```
cmedit get <nodename> CmFunction.syncStatus
```

Update Secure user and ldapApplication user credentials:

```
secadm credentials update --secureusername <ts_username> --secureus
erpassword <ts_password> --ldapuser disable --nodelist <nodename>
```

Enable CM Heartbeat Supervision:

```
cmedit set NetworkElement=<nodename>,CmNodeHeartbeatSupervision=1 a
ctive=true
```

Verify that node is SYNCHRONIZED:

```
cmedit get <nodename> CmFunction.syncStatus
```

2. Add `LocalCOMUser` user (on APG43L):

    a. Create specific user for the standard Linux OS (on APG43L) by logging on to one of the BSC system controllers as TS user and then acquiring root privileges:

    Note: Execute commands under BSC node System.

```
ssh <ts_username>@<node IP address> -p 4422

BSC115-SC-2-2: sudo su
root's password:

BSC115-SC-2-2: id LocalCOMUser

uid=508(LocalCOMUser) gid=100(users) groups=1001(com-emergency),10
0(users)
```

b. If `LocalCOMUser` is not present, create a user account:

```
BSC115-SC-2-2: sudo useradd LocalCOMUser
```

A new account is created according to the defaults of `/etc/default/useradd`.

c. Define `LocalCOMUser` globally for active and passive sides of node:

Note: Following steps to be executed in active side of node with root access.

i. Check the SC state of node:

```
BSC115-SC-2-2: sudo su

BSC115-SC-2-2: prcstate -l

active node is up and working
passive node is up and working
```

ii. Define user globally for active and passive sides of node:

```
BSC115-SC-2-2: lde-global-user --user LocalCOMUser
```

iii. Check `LocalCOMuser` added for passive side of node:

```
BSC115-SC-2-2:  ssh SC-2-1

BSC115-SC-2-1: id LocalCOMUser
uid=508(LocalCOMUser) gid=100(users) groups=100(users)
```

d. If the user account does not belong to the `com-emergency` group for the standard Linux OS (on APG43L):

```
BSC115-SC-2-2: usermod -G "com-emergency" LocalCOMUser
```

The account is added to the `com-emergency` group.

e. Exit from the system controller:

```
BSC115-SC-2-2: exit
```

Note: On ENM, the CM Bulk Synch Flow with BSC node user `LocalCOMUser` is controlled by PIB parameter `useLocalCOMUser`, described in ENM Parameter List 1/19059-AOM 901 151.

3. OAM Certificate Issue (ENM web CLI):

a. Verify `NetconfTls` and `LDAP` MOs configuration.

Get `LDAP` MO configuration:

```
cmedit get <nodename> Ldap.(nodeCredential,trustCategory)
FDN : <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,SecM=
1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1
nodeCredential : <ldapNodeCredentialFdn>
trustCategory: <ldapTrustCategoryFdn>
```

Get `NetconfTls` MO configuration:

```
cmedit get <nodename> NetconfTls.(nodeCredential,trustCategory)
FDN : <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,SysM=
1,NetconfTls=1
nodeCredential : <netconfTlsNodeCredentialFdn>
trustCategory : <netconfTlsTrustCategoryFdn>
```

If `NetconfTls.nodeCredential` value is different from `Ldap.nodeCredential` value OR if `NetconfTls.trustCategory` value is different from `Ldap.trustCategory` value, the `NetconfTls` MO must be aligned to `LDAP` MO with the following commands:

```
cmedit set <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,
SysM=1,NetconfTls=1 nodeCredential="<LdapNodeCredentialFdn>"
cmedit set <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,
SysM=1,NetconfTls=1 trustCategory="<LdapTrustCategoryFdn>"
```

b. Use the following file to issue the OAM certificate on BSC node (put the correct <nodename> inside the file).

   Filename: `<nodename>-OAM.xml`

   Template.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<Nodes>
 <Node>
 <NodeFdn><nodename></NodeFdn>
 </Node>
</Nodes>
```

c. Drag and drop the `<nodename>-OAM.xml` file onto the ENM web CLI and execute the following command:

```
secadm certificate issue --certtype OAM --xmlfile file:"<nodename>-
OAM.xml"
```

   The following output must be displayed:

```
Successfully started a job to issue certificates for nodes. Perform
 'secadm job get -j <jobId number>' to get progress info.
```

   As described by the output of the command, run the suggested command to monitor the workflow progress until the "Job Status" is `COMPLETED`.

   When the "Job Status" is `COMPLETED`, verify that, in the same command output, the "Workflow Status" is `SUCCESS`.

4. Configure LDAP and set its Administrative State to UNLOCKED (ENM web CLI):

    a. Get Ldap.bindDn attribute value:

    ```
    cmedit get <nodename> Ldap.bindDn
    FDN : SubNetwork=<ossprefix>,ManagedElement=<nodename>,SystemFuncti →
    ons=1,SecM=1,UserManagement=1,LdapAuthenticationMethod=1,Ldap=1
    bindDn : <ldapBindDn>
    ```

    The Ldap.bindDn value is used for further checks on LDAP configuration result.

    b. Use the following file to configure LDAP (put the correct <nodename> inside the file).

    Filename: <nodename>_LDAPS.xml

    Template.xml

    ```
    <?xml version="1.0" encoding="UTF-8"?>
    <Nodes>
        <Node>
            <nodeFdn>NetworkElement=<nodename></nodeFdn>
            <tlsMode>LDAPS</tlsMode>
            <userLabel>ENM</userLabel>
            <useTls>true</useTls>
        </Node>
    </Nodes>
    ```

    c. Drag and drop the <nodename>-LDAPS.xml file onto the ENM web CLI and execute the following command:

    ```
    secadm ldap configure --xmlfile file:"<nodename>_LDAPS.xml"
    ```

    d. Verify that the LDAP is correctly configured:

    ```
    cmedit get <nodename> Ldap.bindDn
    ```

    The current Ldap.bindDn value must be different from the previous one.

    e. Get the LDAP administrative status:

    ```
    cmedit get <nodename> LdapAuthenticationMethod.administrativeState
    ```

    If the administrativeState is LOCKED, set it to UNLOCKED:

    ```
    cmedit set <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1, →
    SecM=1,UserManagement=1,LdapAuthenticationMethod=1 administrativeSt →
    ate=UNLOCKED
    ```

5. Update CliTls (ENM web CLI):

a. Get the correct `nodeCredential` and `trustCategory` references from NetconfTls MO:

```
cmedit get <nodename> NetconfTls.(nodeCredential,trustCategory)
FDN : <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,SysM=
1,NetconfTls=1
nodeCredential : <netconfTlsNodeCredentialFdn>
trustCategory : <netconfTlsTrustCategoryFdn>
```

b. Update `nodeCredential` and `trustCategory` attributes on CliTls accordingly:

```
cmedit set <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,
SysM=1,CliTls=1 nodeCredential="<netconfTlsNodeCredentialFdn>"
cmedit set <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,
SysM=1,CliTls=1 trustCategory="<netconfTlsTrustCategoryFdn>"
```

c. Enable CliTls by setting its administrative state to `UNLOCKED`:

```
cmedit set <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,
SysM=1,CliTls=1 administrativeState=UNLOCKED
```

6. Set NetconfTls Administrative State to `UNLOCKED` (ENM web CLI):

a. Get the NetconfTls administrative status:

```
cmedit get <nodename> NetconfTls.administrativeState
```

b. if the administrativeState is `LOCKED`, set it to `UNLOCKED`:

```
cmedit set <ossprefix>,ManagedElement=<nodename>,SystemFunctions=1,
SysM=1,NetconfTls=1 administrativeState=UNLOCKED
```

7. Disable CM Heartbeat Supervision (ENM web CLI):

a. Stop the CM heartbeat supervision for the node:

```
cmedit set NetworkElement=<nodename>,CmNodeHeartbeatSupervision=1 a
ctive=false
```

b. Verify that the node is `UNSYNCHRONIZED`:

```
cmedit get <nodename> CmFunction.syncStatus
```

8. Update Node Credentials (ENM web CLI).

a. Create (if not existing) a User Role named `CpRole0` with Role Type of `COM Role`.

b. Create (if not existing) an ENM User to be used as Secure user.

The ENM User can be defined once in ENM and used as Secure user for multiple BSC nodes.

The ENM User must have the following roles:

— SystemAdministrator.

— SystemSecurityAdministrator.

— EricssonSupport.

— BscApplicationAdministrator.

— SystemReadOnly.

— CpRole0.

For all the roles, the Target Group must be set to `ALL`.

The ENM User must have the Password Aging disabled.

The ENM User must have the Force Password Change disabled.

c. Update node credentials with `ldapApplication` user enabled and with ENM User credentials as Secure user credentials:

```
secadm credentials update --secureusername <ENM User username> --se →
cureuserpassword <ENM User password> --ldapuser enable --nodelist < →
nodename>
```

d. Verify that node credentials have been updated under ENM with Secure user equal to ENM User with the command:

```
secadm credentials get --plaintext show --nodelist <nodename>
```

`secureUserName` and `secureUserPassword` must match the ENM User `username` and `password`.

e. Verify that the `ldapApplicationUserName` and `ldapApplicationUserPassword` fields are not empty:

```
cmedit get <nodename> NetworkElementSecurity.(ldapApplicationUserNa →
me, ldapApplicationUserPassword)
FDN : NetworkElement=<nodename>,SecurityFunction=1,NetworkElementSe →
curity=1
ldapApplicationUserName : <ldapApplication username>
ldapApplicationUserPassword : <ldapApplication password>
```

9. Move node from SSH to TLS (ENM web CLI):

a. Read the current configuration in `BscConnectivityInformation` MO:

```
cmedit get <nodename> BscConnectivityInformation.(transportProtocol →
,port)
FDN : NetworkElement=<nodename>,BscConnectivityInformation=1
port : 830
transportProtocol : SSH
```

b. Change the transport protocol from SSH to TLS in `BscConnectivityInformation` MO:

```
cmedit set NetworkElement=<nodename>,BscConnectivityInformation=1 t →
ransportProtocol=TLS
```

c. Change the port number in `BscConnectivityInformation` MO:

```
cmedit set NetworkElement=<nodename>,BscConnectivityInformation=1 p →
ort=6513
```

10. Enable CM Heartbeat Supervision (ENM web CLI):

a. Start the CM heartbeat supervision for the node:

```
cmedit set NetworkElement=<nodename>,CmNodeHeartbeatSupervision=1 a →
ctive=true
```

b. Verify that the node is `SYNCHRONIZED`:

```
cmedit get <nodename> CmFunction.syncStatus
```

3.14.1 Configure BSC with TLS

This section describes the procedure to add BSC with default TLS protocol.

Prerequisites

— BSC node configured with credentials created with secureuser equal to TS user

— `NetworkElementSecurity` MO contains empty `ldapAuthenticatioUserName` and `ldapAuthenticationUserPassword`.

— The operator has `NodeSecurity_Administrator` Role and `Cmedit_Administrator` Role to access the CLI in ENM.

— The user has the following COM roles to access or modify specific MOs under Node BSC.

  — `SystemAdministrator`

  — `SystemSecurityAdministrator`

  — `EricssonSupport`

  — `SystemReadOnly`

  — `CpRole0`

  — `BscApplicationCellCnAdministrator`

- BscApplicationBtsAdministrator
- BscApplicationAdministrator
- BscApplicationReadOnly

— Ensure that Assign Target Groups (which contains all or specific nodes) is set to all COM Roles, and must be configured first with targetType under UserManagement=1 MO of node BSC.

Steps

1. Add Node with TLS in ENM

   To add a node in ENM, launch the ENM CLI application for BSC node type. See the appropriate online help under section *Add Node* for respective node type.

   OR

   The Add Node application can be used from ENM GUI to add the node.

   Note: Even if port and transportProtocol are not mentioned in the CLI commands, the node is added with default port and TLS as transportProtocol.

2. OAM Certificate Enrollment and LDAP Configuration (Node CLI)

   To perform Certificate enrollment and LDAP configuration, see the appropriate *Offline Enrollment Procedure* section for the node in question in the ENM Network Security Configuration System Administrator Guide

3. Put NetconfTls and CliTls administrativeState=UNLOCKED (Node CLI)

   Enable the TLS communication using the provided certificates.

   Do this by setting the attribute administrativeState in both MOs CliTls=1, NetconfTls=1 to value UNLOCKED.

   If the attribute is reset to LOCKED, all related opened sessions - AP session (MML session) are automatically closed.

   a. Get the correct nodeCredential and trustCategory references from LDAP MO as configured in Step 2:

      Navigate to Ldap=1 MO of the Node by tabbing out each comma-separated MO.

```
>dn -m Ldap
ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1, →
LdapAuthenticationMethod=1,Ldap=1
(Ldap=1)>show -v
Ldap=1
baseDn=" dc=ieatlms5589,dc=com"
bindDn="cn=ProxyAccount_19,ou=proxyagent,ou=com,dc=ieatlms5589,dc=c →
```

```
om"
nodeCredential="ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,C
ertM=1,NodeCredential=1"
profileFilter=ERICSSON_FILTER
serverPort=1636
tlsMode=LDAPS
trustCategory="ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,Ce
rtM=1,TrustCategory=1"
useReferrals=false <default>
...
```

b. Navigate to the `CliTls=1` MO of the Node by tabbing out each comma-separated MO and update `nodeCredential` MO, `TrustCategory` MO, and `administrativeState`:

```
>dn -m CliTls
ManagedElement =<NE_NAME>,SystemFunctions=1,SysM=1,CliTls=1
(CliTls=1)>configure
(config-CliTls=1)>show -v
CliTls=1
administrativeState=LOCKED
cliTlsId="1"
nodeCredential=[] <empty>
trustCategory=[] <empty>
(config-CliTls=1)>nodeCredential="ManagedElement=<NodeName>,SystemF
unctions=1,SecM=1,CertM=1,NodeCredential=1"
(config-CliTls=1)>trustCategory="ManagedElement=<NodeName>,SystemFu
nctions=1,SecM=1,CertM=1,TrustCategory=1"
(config-CliTls=1)>administrativeState=UNLOCKED
(config-CliTls=1)>commit -s
(config-CliTls=1)>show
CliTls=1
administrativeState=UNLOCKED
nodeCredential="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,
CertM=1,NodeCredential=1"
trustCategory="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,C
ertM=1,TrustCategory=1"
(config-CliTls=1)>up
```

c. Navigate to the `NetconfTls=1` MO of the Node by tabbing out each comma-separated MO and update `nodeCredential` MO, `TrustCategory` MO, and `administrativeState`:

```
>dn -m NetconfTls
ManagedElement=<NE_NAME>,SystemFunctions=1,SysM=1,NetconfTls=1
(NetconfTls=1)>show -v
NetconfTls=1
administrativeState=LOCKED
netconfTlsId="1"
nodeCredential=[] <empty>
trustCategory=[] <empty>
(NetconfTls=1)>configure
(config-NetconfTls=1)nodeCredential="ManagedElement=<NodeName>,Syst
emFunctions=1,SecM=1,CertM=1,NodeCredential=1"
(config-NetconfTls=1)>trustCategory="ManagedElement=<NodeName>,Syst
emFunctions=1,SecM=1,CertM=1,TrustCategory=1"
(config-NetconfTls=1)>administrativeState=UNLOCKED
(config-NetconfTls=1)>commit -s
(config-NetconfTls=1)>show
NetconfTls=1
administrativeState=UNLOCKED
nodeCredential="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,
CertM=1,NodeCredential=1"
trustCategory="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,C
ertM=1,TrustCategory=1"
(config-NetconfTls=1)>
```

4. Update the Credentials (ENM web CLI).

a. Create a User Role (if it does not exist), named `CpRole0` with Role Type, `COM` Role.

b. Create (if it does not exist) an ENM User to be used as Secure user.

   The ENM User can be defined once in ENM and used as Secure user for multiple BSC nodes. The ENM User must have the following roles:

   — `SystemAdministrator`

   — `SystemSecurityAdministrator`

   — `EricssonSupport`

   — `SystemReadOnly`

   — `CpRole0`

   — `BscApplicationCellCnAdministrator`

   — `BscApplicationBtsAdministrator`

   — `BscApplicationAdministrator`

   — `BscApplicationReadOnly`

   For all these roles, the Target Group must be set to `ALL`.

   The ENM User must have `Password Aging` and `Force Password Change`disabled.

c. Update node credentials with `ldapApplication` user enabled and with ENM User credentials as Secure user credentials:

```
»secadm credentials update --secureusername <ENM User username> --s →
ecureuserpassword <ENM User password> --ldapuser enable --nodelist →
<nodename>
```

d. Verify that node credentials have been updated under ENM with Secure user equal to ENM User with the command:

```
»secadm credentials get --plaintext show --nodelist <nodename>
```

   `secureUserName` and `secureUserPassword` must match the ENM User username and password.

e. Verify that the `ldapApplicationUserName` and `ldapApplicationUserPassword` fields are not empty:

```
»cmedit get <nodename> NetworkElementSecurity.(ldapApplicationUserN →
ame, ldapApplicationUserPassword)
```

FDN: NetworkElement <nodename>, SecurityFunction=1, NetworkElementSecurity=1

ldapApplicationUserName: <ldapApplication username>

ldapApplicationUserPassword: <ldapApplication password>

Results

At the end of this procedure, the BSC node must be in SYNC with TLS transport protocol.

3.15 Move HLR Node from SSH to TLS

> Describes the procedure to move HLR node from SSH to TLS.

At the end of this procedure, the HLR node is configured with TLS transport protocol.

Prerequisites

— HLR node configured with credentials created with secureuser equal to TS user and No Ldap Options.

— NetworkElementSecurity MO contains empty ldapAuthenticatioUserName and ldapAuthenticationUserPassword.

3.15.1 Install OAM Certificate on Node

Steps

1. Obtain the Trust Distribution Point Service (TDPS) URL for each of the three ENM CA certificates, by launching the ENM CLI, and executing the following commands:

   These are used later to download the ENM CA certificates to the Node.

   ```
   pkiadm trustmgmt --list --entitytype ca --entityname ENM_PKI_Root_CA pkiadm trustmgmt --list --entitytype ca --entityname ENM_Infrastructure_CA pkiadm trustmgmt --list --entitytype ca --entityname ENM_OAM_CA pkiadm trustmgmt --list --entitytype ca --entityname NE_OAM_CA
   ```

2. Log on to the ENM server and use WinFiol Application to access the Node ECLI:

   ```
   ssh <TS_USER_NAME>@<NE_IP_ADDRESS>
   ```

3. Navigate to the CertM MO of the Node:
4. For each of the three ENM CA certs, install the certificate by executing the `installTrustedCertFromUri` action on the Node:

   This action requires three parameters:

   — TDPS URL of the ENM CA cert to be downloaded.

   — URL password (if no password is required, the NULL string can be provided).

   — CA fingerprint (NULL string can be provided).

```
(CertM=1)>installTrustedCertFromUri <TDPS URL> NULL NULL
   OR

Trust certificates can be downloaded and sftp to node and installed using fi →
le:// or sftp.
In case of offline enrollment procedure. The file cacert.pem is expected to  →
be into folder /certificates on Node
file system.

>configure
(config)>dn -m CertM
ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,CertM=1
(config-CertM=1)>installTrustedCertFromUri cacert.pem NULL NULL
true
(config-CertM=1)>show
localFileStorePath="certificates"
actionName="installTrustedCertFromUri"
additionalInfo
"TrustedCertificate=1"
progressPercentage=100
result=SUCCESS
resultInfo="installed from the certificate file"
state=FINISHED
(config-CertM=1)>TrustCategory=1
(config-TrustCategory=1)>trustedCertificates="ManagedElement=CEMSS07,SystemF →
unctions=1,SecM=1,Cer
(config-TrustCategory=1)>commit -s
```

5. Execute the show command on the `CertM` MO to display the installed Trusted Certificates on the NE.
6. Navigate to the `TrustCategory=1` MO and add the ENM CA certificates installed in Step 5.
7. Commit the changes to the `TrustCategory=1` MO to add the ENM CA certificates.
8. Verify that the Trust Category has been updated correctly. Navigate to `TrustCategory` and execute the `show` command to verify that all the certificates are added.

3.15.2 End Entity Creation and Credential Generation

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System and use it for communication with other ENM systems. End Entities are created in the PKI system.

To generate credentials each End Entity is mapped to an Entity Profile (EP) which defines the Certificate Authority (CA).

For more information about Entity and Profiles see *Public Key Infrastructure System* in the ENM Security System Administrator Guide, 2/1543-AOM 901 151 Uen.

Log on to the OSS-RC server and use the `WinFiol` Application to access the Node ECLI:

```
ssh<TS_USER_NAME>@<NE_IP_ADDRESS>

....input TS User password
```

Create NodeCredential MO

Steps

1. Navigate to the `NodeCredential=1` of the Node by tabbing out each comma-separated MO.

   ```
   ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,CertM=1,NodeCredential=1
   ```

2. Execute the following command to create MO `NodeCredential=2` and install new P12 certificates:

   ```
   (NodeCredential=1)>installCredentialFromUri sftp://root@LMS_IP/tmp/NE_NAME-o →
   am.p12 <LMS Password> <Password of P12 file> NULL
    true
    (NodeCredential=1)>show enrollmentProgress
    enrollmentProgress
       actionId=0
       actionName="installCredentialFromUri"
       progressInfo=""
       progressPercentage=100
       result=SUCCESS
       resultInfo="installed from the container file"
       state=FINISHED
       timeActionCompleted="2017-08-05T12:37:36Z"
       timeActionStarted="2017-08-05T12:37:36Z"
       timeOfLastStatusUpdate="2017-08-05T12:37:36Z"
    (NodeCredential=1)>


   OR



   (CertM=1)>NodeCredential=2 (NodeCredential=1)>configure (config-NodeCredenti →
   al=2)>installCredentialFromUri NE_NAME-oam.p12 NULL <Password of P12 file> N →
   ULL true (config-NodeCredential=2)>show enrollmentProgress
   enrollmentProgress
       actionId=0
       actionName="installCredentialFromUri"
       progressInfo=""
       progressPercentage=100
       result=SUCCESS
       resultInfo="installed from the container file"
       state=FINISHED
   ```

Note: The password of the P12 file, is the same password as was used when creating the P12 file.

Make sure that the enrollment progress is 100 percent completed. To verify the same, the command **show enrollmentProgress** can be used.

3.15.3 Configure LDAP and Set administrativeState=UNLOCKED

> Describes the procedure to reconfigure LDAP credentials on the HLR-FE to use ENM LDAP server.

Execute the following command in ENM CLI app to retrieve LDAP configuration details:

Steps

1. Retrieve values from ENM:

```
secadm ldap configure --manual
```

2. Navigate to the Ldap=1 MO of the Node by tabbing out each comma-separated MO.

```
ManagedElement =<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuth →
enticationMethod=1,Ldap=1
```

Note: Before navigating to Ldap=1 MO of the node, make sure that the attribute administrativeState must be UNLOCKED under LdapAuthenticationMethod=1 MO.

```
LdapAuthenticationMethod=1)>show
LdapAuthenticationMethod=1
administrativeState=UNLOCKED
   Ldap=1
```

3. Configure the LDAP MO details as follows, using the values retrieved from Step Step 1:

```
(Ldap=1)>dn -m Ldap

ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuthe →
nticationMethod=1,Ldap=

(Ldap=1)>configure
(config-Ldap=1)>baseDn="dc=ieatlms5589,dc=com"
(config-Ldap=1)>bindDn="cn=ProxyAccount_19,ou=proxyagent,ou=com,dc=ieatlms55 →
89,dc=com"
(config-Ldap=1)>fallbackLdapIpAddress="131.160.129.59"
(config-Ldap=1)>ldapIpAddress="131.160.129.60"
(config-Ldap=1)>bindPassword="gfu94ncy" cleartext
(config-Ldap=1)>nodeCredential="ManagedElement=<NE_NAME>,SystemFunctions=1,S →
ecM=1,CertM=1,NodeCredential=2"
(config-Ldap=1)>commit
```

```
(Ldap=1)>show -v
Ldap=1
   baseDn=" dc=ieatlms5589,dc=com"
   bindDn="cn=ProxyAccount_19,ou=proxyagent,ou=com,dc=ieatlms5589,dc=com"
   bindPassword="1:36OpqTK3HCo9zjV46dm1HmnQcIrXImn"
   fallbackLdapIpAddress="131.160.129.59"
   ldapId="1"
   ldapIpAddress="131.160.129.60"
   nodeCredential="ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,CertM=1 →
,NodeCredential=2"
   profileFilter=ERICSSON_FILTER
   serverPort=1636
   tlsMode=LDAPS
   trustCategory="ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,CertM=1, →
TrustCategory=2"
   useReferrals=false <default>
   userLabel="LDAP based login authentication"
   useTls=true
   EricssonFilter=1
   Filter=1
(Ldap=1)>
```

4. Set `ERICSSON_FILTER`:

   Navigate to the `Ldap=1` MO of the Node by tabbing out each comma-separated MO.

   Example:

```
ManagedElement =<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuth →
enticationMethod=1,Ldap=1

>dn -m Ldap

ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuthe →
nticationMethod=1,Ldap=

(Ldap=1)>configure

(config-Ldap=1)>profileFilter=ERICSSON_FILTER

(config-Ldap=1)>commit
```

5. Set the administrative status of `LdapAuthenticationMethod=UNLOCKED`:

   Navigate to the `LdapAuthenticationMethod=1` MO of the Node by tabbing out each comma-separated MO.

   Example:

```
ManagedElement =<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuth →
enticationMethod=1

>dn -m LdapAuthenticationMethod

ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuthe →
nticationMethod=1

(config-UserManagement=1)>LdapAuthenticationMethod=1

(config-LdapAuthenticationMethod=1)>administrativeState=UNLOCKED

(config-UserManagement=1)>commit
```

3.15.4 Set netconfTls administrativeState=UNLOCKED

Steps

1. Verify that the `netconf` for the node under test is still SSH:

```
»cmedit get NetworkElement=<NodeName>,HlrFeConnectivityInformation=1
FDN : NetworkElement=<NodeName>,HlrFeConnectivityInformation=1

HlrFeConnectivityInformationId : 1
apnodeAIpAddress : 10.198.97.68
apnodeBIpAddress : 10.198.97.69
apSftpPort : 22
apSshPort : 22
httpsPort : 8181
ipAddress : 10.198.97.70
isManagerIpAddress : 10.198.97.71
port : 830
snmpAgentPort : 161
snmpReadCommunity : public
snmpSecurityLevel : null
snmpSecurityName : null
snmpVersion : SNMP_V2C
snmpWriteCommunity : public
transportProtocol : SSH
```

2. Set the attribute `administrativeState` in one or both MOs `CliTls=1`, `NetconfTls=1`, to value `UNLOCKED`.

   Note: This enables the TLS communication using the provided certificates. If the attribute is reset to `LOCKED`, all related opened sessions, AP session, or NETCONF session are automatically closed.

   Example:

   The following example shows how to configure on APG the support for both AP session and NETCONF session over TLS, using, for both, the CA trusted certificate belonging to the set of trusted certificates represented by MO `TrustedCategory=1`, and the APG node credential represented by MO.

   Navigate to the `CliTls=1` MO of the Node by tabbing out each comma-separated MO.

```
>dn -m CliTls
ManagedElement =<NE_NAME>,SystemFunctions=1,SysM=1,CliTls=1
(CliTls=1)>configure
(config-CliTls=1)>show -v
CliTls=1
administrativeState=LOCKED
cliTlsId="1"
nodeCredential=[] <empty>
trustCategory=[] <empty>
(config-CliTls=1)>nodeCredential="ManagedElement=<NodeName>,SystemFunctions= →
1,SecM=1,CertM=1,NodeC
(config-CliTls=1)>trustCategory="ManagedElement=<NodeName>,SystemFunctions=1 →
,SecM=1,CertM=1,TrustC
(config-CliTls=1)>administrativeState=UNLOCKED
(config-CliTls=1)>commit -s
(config-CliTls=1)>show
CliTls=1
administrativeState=UNLOCKED
nodeCredential="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,CertM=1,N →
odeCredential=1"
trustCategory="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,CertM=1,Tr →
ustCategory=1"
(config-CliTls=1)>up
(config-SwM=1)>NetconfTls=1
(config-NetconfTls=1)>show -v
```

```
NetconfTls=1
administrativeState=LOCKED
netconfTlsId="1"
nodeCredential=[] <empty>
trustCategory=[] <empty>
(config-NetconfTls=1)nodeCredential="ManagedElement=CEMSS07,SystemFunctions=
1,SecM=1,CertM=1,No
(config-NetconfTls=1)>trustCategory="ManagedElement=CEMSS07,SystemFunctions=
1,SecM=1,CertM=1,Tr
(config-NetconfTls=1)>administrativeState=UNLOCKED
(config-NetconfTls=1)>commit -s
(config-NetconfTls=1)>show
NetconfTls=1
administrativeState=UNLOCKED
nodeCredential="ManagedElement=CEMSS07,SystemFunctions=1,SecM=1,CertM=1,Node
Credential=1"
trustCategory="ManagedElement=CEMSS07,SystemFunctions=1,SecM=1,CertM=1,Trust
Category=1"
(config-NetconfTls=1)>
```

3.15.5 Credential Update with LDAP Enabled

Steps

1. Update the node credential with `ldapuser Enabled`:

```
»secadm credentials update --ldapuser enable --nodelist <NodeName>
```

2. Verify that the `ldapApplicationUserName` and `LdapApplicationUserPassword` fields are compiled:

```
»cmedit get NetworkElement=<NodeName,SecurityFunction=1,NetworkElementSecuri
ty=1
```

3.15.6 Update the HlrFeConnectivityInformation MO to Use TLS on Port 6513

Steps

1. Change the `transportProtocol` from SSH to TLS:

```
»cmedit set NetworkElement=<NodeName>,HlrFeConnectivityInformation=1 transpo
rtProtocol=TLS
```

2. Change the port number:

```
»cmedit set NetworkElement=<NodeName>,HlrFeConnectivityInformation=1 port=65
13
```

3.16 Move MSC/HLR Variants from SSH to TLS

This section describes the procedure to move MSC/HLR variants from SSH to TLS.

Supported variants are listed as follows:

— Compact CTC

— Compact IP-STP

— Compact MSC-S

— Compact vCTC

— Compact vIP-STP

— Compact vMSC-S

— CTC BC (BSP)

— CTC BC (IS)

— CTC DB

— HLR-FE (APZ)

— HLR-FE (IS)

— IP-STP

— LR-FE (BSP)

— MSC-S BC (BSP)

— MSC-S BC (IS)

— MSC-S DB

— vCTC HC

— vHLR-FE

— vMSC-S HC

Prerequisites

— Node is configured with credentials created with `secureuser` equal to `TS` user and `No Ldap Options`.

— `NetworkElementSecurity` MO contains empty `ldapAuthenticatioUserName` and `ldapAuthenticationUserPassword`.

Expected Results

The node is configured with the TLS transport protocol.

3.16.1 OAM Certificate to Be Installed on Node

To obtain the Trust Distribution Point Service (TDPS) URL for each of the three ENM CA certificates, launch the ENM CLI, and execute the following commands. These are used later to download the ENM CA certificates to the Node.

```
pkiadm trustmgmt --list --entitytype ca --entityname ENM_PKI_Root_CA pkiadm trus →
tmgmt --list --entitytype ca --entityname ENM_Infrastructure_CA pkiadm trustmgmt →
 --list --entitytype ca --entityname ENM_OAM_CA pkiadm trustmgmt --list --entity →
type ca --entityname NE_OAM_CA
```

Steps

1. Log on to the ENM server and use `WinFiol` Application to access the Node CLI:

   ```
   ssh <TS_USER_NAME>@<NE_IP_ADDRESS>
   ```

2. Navigate to the `CertM` MO of the Node.

3. For each CA certs, install the certificates by executing the `installTrustedCertFromUri` action on the Node.

   This action requires three parameters:

   a. `TDPS URL` of the ENM CA cert to be downloaded.

   b. `URL` password (if no password is required, the NULL string can be provided).

   c. CA fingerprint (NULL string can be provided).

   OR

   d. Trust certificates can be downloaded and SFTP to the node and installed using file:// or SFTP.

   In offline enrollment procedure. The file `cacert.pem` is expected to be placed in folder /certificates on Node file system.

   ```
   (CertM=1)>installTrustedCertFromUri <TDPS URL> NULL NULL

   OR

   >configure
   (config)>dn -m CertM
   ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,CertM=1
   ```

```
(config-CertM=1)>installTrustedCertFromUri cacert.pem NULL NULL
true
(config-CertM=1)>show
localFileStorePath="certificates"
actionName="installTrustedCertFromUri"
additionalInfo
"TrustedCertificate=1"
progressPercentage=100
result=SUCCESS
resultInfo="installed from the certificate file"
state=FINISHED
(config-CertM=1)>TrustCategory=1
(config-TrustCategory=1)>trustedCertificates="ManagedElement=CEMSS07,SystemF →
unctions=1,SecM=1,Cer
(config-TrustCategory=1)>commit -s
```

4. Execute the show command on the `CertM` MO to display the installed Trusted Certificates on the NE.
5. Navigate to the `TrustCategory=1` MO and add the ENM CA certificates installed in Step 3.
6. Commit the changes to the `TrustCategory=1` MO to add the ENM CA certificates.
7. Verify that the Trust Category has been updated correctly. Navigate to `TrustCategory` and execute the `show` command to verify that all the certificates are added.

3.16.2 End Entity Creation and Credential Generation

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System. They use it for communication with other ENM systems. End Entities must be created in the PKI system.

To generate credentials each End Entity is mapped to an Entity Profile (EP) which defines the Certificate Authority (CA).

For more information about `Entity and Profiles` see *Public Key Infrastructure System* in the ENM Security System Administration Guide, 2/1543-AOM 901 151 Uen.

Log on to the OSS-RC server and use the `WinFiol` Application to access the Node ECLI

```
ssh<TS_USER_NAME>@<NE_IP_ADDRESS>
....input TS User password
```

3.16.2.1 Create NodeCredential MO

Steps

1. Navigate to the `NodeCredential=1` of the Node by tabbing out each comma-separated MO.

```
ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,CertM=1,NodeCredential=1
```

2. Use the following command to create MO `NodeCredential=2` and install new P12 certificates.

```
(config-NodeCredentail=1)>commit
```

3. The following command installs new P12 certificates in `NodeCredential=1`:

   OR

4. Download and SFTP node credential certificates to the node. Install them using `file://` or SFTP. The new node credential must be uploaded on the node file system in the folder `/certificates`:

```
(NodeCredential=1)>installCredentialFromUri sftp://root@LMS_IP/tmp/NE_NAME-o →
am.p12 <LMS Password> <Password of P12 file> NULL
 true
 (NodeCredential=1)>show enrollmentProgress
 enrollmentProgress
    actionId=0
    actionName="installCredentialFromUri"
    progressInfo=""
    progressPercentage=100
    result=SUCCESS
    resultInfo="installed from the container file"
    state=FINISHED
    timeActionCompleted="2017-08-05T12:37:36Z"
    timeActionStarted="2017-08-05T12:37:36Z"
    timeOfLastStatusUpdate="2017-08-05T12:37:36Z"
 (NodeCredential=1)>


OR



(CertM=1)>NodeCredential=2 (NodeCredential=1)>configure (config-NodeCredenti →
al=2)>installCredentialFromUri NE_NAME-oam.p12 NULL <Password of P12 file> N →
ULL true (config-NodeCredential=2)>show enrollmentProgress
enrollmentProgress
    actionId=0
    actionName="installCredentialFromUri"
    progressInfo=""
    progressPercentage=100
    result=SUCCESS
    resultInfo="installed from the container file"
    state=FINISHED
```

Note: The password of the P12 file is the same as was used to create the P12 file.

Make sure that the enrollment progress is 100 percent completed. To verify the same, the command `show enrolmentProgress` is used.

3.16.3 Configure LDAP and Put It with administrativeState=UNLOCKED

> Describes the procedure to reconfigure LDAP credentials on the node to use ENM LDAP server.

Execute the following commands in ENM CLI app to retrieve LDAP configuration details.

Steps

1. Retrieve values from ENM:

```
secadm ldap configure --manual
```

2. Navigate to the `Ldap=1` MO of the Node by tabbing out each comma-separated MO.

```
ManagedElement =<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuth →
enticationMethod=1,Ldap=1
```

Note: Before navigating to `Ldap=1` MO of the node, make sure that the attribute `administrativeState` is `UNLOCKED` under `LdapAuthenticationMethod=1` MO.

```
LdapAuthenticationMethod=1)>show
LdapAuthenticationMethod=1
administrativeState=UNLOCKED
   Ldap=1
```

3. Configure the LDAP MO details as follows, using the values retrieved from Step 1:

```
(Ldap=1)>dn -m Ldap

ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuthe →
nticationMethod=1,Ldap=

(Ldap=1)>configure
(config-Ldap=1)>baseDn="dc=ieatlms5589,dc=com"
(config-Ldap=1)>bindDn="cn=ProxyAccount_19,ou=proxyagent,ou=com,dc=ieatlms55 →
89,dc=com"
(config-Ldap=1)>fallbackLdapIpAddress="131.160.129.59"
(config-Ldap=1)>ldapIpAddress="131.160.129.60"
(config-Ldap=1)>bindPassword=”gfu94ncy" cleartext
(config-Ldap=1)>nodeCredential="ManagedElement=<NE_NAME>,SystemFunctions=1,S →
ecM=1,CertM=1,NodeCredential=2"
(config-Ldap=1)>commit
(Ldap=1)>show -v
Ldap=1
   baseDn=" dc=ieatlms5589,dc=com"
   bindDn="cn=ProxyAccount_19,ou=proxyagent,ou=com,dc=ieatlms5589,dc=com"
   bindPassword="1:36OpqTK3HCo9zjV46dm1HmnQcIrXImn"
   fallbackLdapIpAddress="131.160.129.59"
   ldapId="1"
   ldapIpAddress="131.160.129.60"
   nodeCredential="ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,CertM=1 →
,NodeCredential=2"
   profileFilter=ERICSSON_FILTER
   serverPort=1636
```

```
   tlsMode=LDAPS
   trustCategory="ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,CertM=1,
TrustCategory=2"
   useReferrals=false <default>
   userLabel="LDAP based login authentication"
   useTls=true
   EricssonFilter=1
   Filter=1
(Ldap=1)>
```

4. Set ERICSSON_FILTER:

   Navigate to the `Ldap=1` MO of the Node by tabbing out each comma-separated MO.

   Example

```
ManagedElement =<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuth
enticationMethod=1,Ldap=1

>dn -m Ldap

ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuthe
nticationMethod=1,Ldap=

(Ldap=1)>configure

(config-Ldap=1)>profileFilter=ERICSSON_FILTER

(config-Ldap=1)>commit
```

5. Set the administrative status of `LdapAuthenticationMethod=UNLOCKED`:

   Navigate to the `dapAuthenticationMethod=1L` MO of the Node by tabbing out each comma-separated MO.

   Example

```
ManagedElement =<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuth
enticationMethod=1

>dn -m LdapAuthenticationMethod

ManagedElement=<NE_NAME>,SystemFunctions=1,SecM=1,UserManagement=1,LdapAuthe
nticationMethod=1

(config-UserManagement=1)>LdapAuthenticationMethod=1

(config-LdapAuthenticationMethod=1)>administrativeState=UNLOCKED

(config-UserManagement=1)>commit
```

3.16.4 Put NetconfTls administrativeState=UNLOCKED.

Steps

1. Verify that the `NETCONF` for the node under test is `SSH`:

   Example

```
»cmedit get NetworkElement=<NodeName>,MscConnectivityInformation=1
FDN : NetworkElement=<NodeName>,MscConnectivityInformation=1
```

```
MscConnectivityInformationId : 1
apnodeAIpAddress : 10.198.97.68
apnodeBIpAddress : 10.198.97.69
apSftpPort : 22
apSshPort : 22
httpsPort : 8181
ipAddress : 10.198.97.70
isManagerIpAddress : 10.198.97.71
port : 830
snmpAgentPort : 161
snmpReadCommunity : public
snmpSecurityLevel : null
snmpSecurityName : null
snmpVersion : SNMP_V2C
snmpWriteCommunity : public
transportProtocol : SSH
```

Note: For HLR variants connectivity Information can be checked with the following command:

```
»cmedit get NetworkElement=<NodeName>,HlrFeConnectivityInformation=
1
```

2. Set `NetconfTls` to `UNLOCKED`:

   Enable the TLS communication using the provided certificates. This is done by setting the attribute `administrativeState` in one or both MOs `CliTls=1`, `NetconfTls=1`, to value `UNLOCKED`.

   If the attribute is reset to `LOCKED`, all related opened sessions, AP session, or NETCONF session, are automatically closed.

   This example shows how to configure on APG the support for both AP session and NETCONF session over TLS by using, for both, the CA trusted certificate belonging to the set of trusted certificates represented by MO `TrustedCategory=1`, and the APG node credential represented by MO.

   Navigate to the `CliTls=1` MO of the Node by tabbing out each comma-separated MO.

```
>dn -m CliTls
ManagedElement =<NE_NAME>,SystemFunctions=1,SysM=1,CliTls=1
(CliTls=1)>configure
(config-CliTls=1)>show -v
CliTls=1
administrativeState=LOCKED
cliTlsId="1"
nodeCredential=[] <empty>
trustCategory=[] <empty>
(config-CliTls=1)>nodeCredential="ManagedElement=<NodeName>,SystemFunctions=
1,SecM=1,CertM=1,NodeC
(config-CliTls=1)>trustCategory="ManagedElement=<NodeName>,SystemFunctions=1
,SecM=1,CertM=1,TrustC
(config-CliTls=1)>administrativeState=UNLOCKED
(config-CliTls=1)>commit -s
(config-CliTls=1)>show
CliTls=1
administrativeState=UNLOCKED
nodeCredential="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,CertM=1,N
odeCredential=1"
trustCategory="ManagedElement=<NodeName>,SystemFunctions=1,SecM=1,CertM=1,Tr
ustCategory=1"
(config-CliTls=1)>up
(config-SwM=1)>NetconfTls=1
(config-NetconfTls=1)>show -v
```

```
NetconfTls=1
administrativeState=LOCKED
netconfTlsId="1"
nodeCredential=[] <empty>
trustCategory=[] <empty>
(config-NetconfTls=1)nodeCredential="ManagedElement=CEMSS07,SystemFunctions=
1,SecM=1,CertM=1,No
(config-NetconfTls=1)>trustCategory="ManagedElement=CEMSS07,SystemFunctions=
1,SecM=1,CertM=1,Tr
(config-NetconfTls=1)>administrativeState=UNLOCKED
(config-NetconfTls=1)>commit -s
(config-NetconfTls=1)>show
NetconfTls=1
administrativeState=UNLOCKED
nodeCredential="ManagedElement=CEMSS07,SystemFunctions=1,SecM=1,CertM=1,Node
Credential=1"
trustCategory="ManagedElement=CEMSS07,SystemFunctions=1,SecM=1,CertM=1,Trust
Category=1"
(config-NetconfTls=1)>
```

3.16.5 Update the Credentials with LDAP Enabled.

Steps

1. Update the node credential with `ldapuser Enabled`:

```
»secadm credentials update --ldapuser enable --nodelist <NodeName>
```

2. Verify that the `ldapApplicationUserName` and `LdapApplicationUserPassword` fields are compiled:

```
»cmedit get NetworkElement=<NodeName,SecurityFunction=1,NetworkElementSecuri
ty=1
```

3.16.6 Update the MscConnectivityInformation MO/HlrFeConnectivityInformation MO to Use TLS on Port 6513

Steps

1. Change the `transportProtocol` from `SSH` to `TLS`:

   Example

```
»cmedit set NetworkElement=<NodeName>,MscConnectivityInformation=1 transport
Protocol=TLS
```

2. Change the port number:

   Example

```
»cmedit set NetworkElement=<NodeName>,MscConnectivityInformation=1 port=6513
```

3.17 Install Certificate to Launch SBG(IS) Element Manager

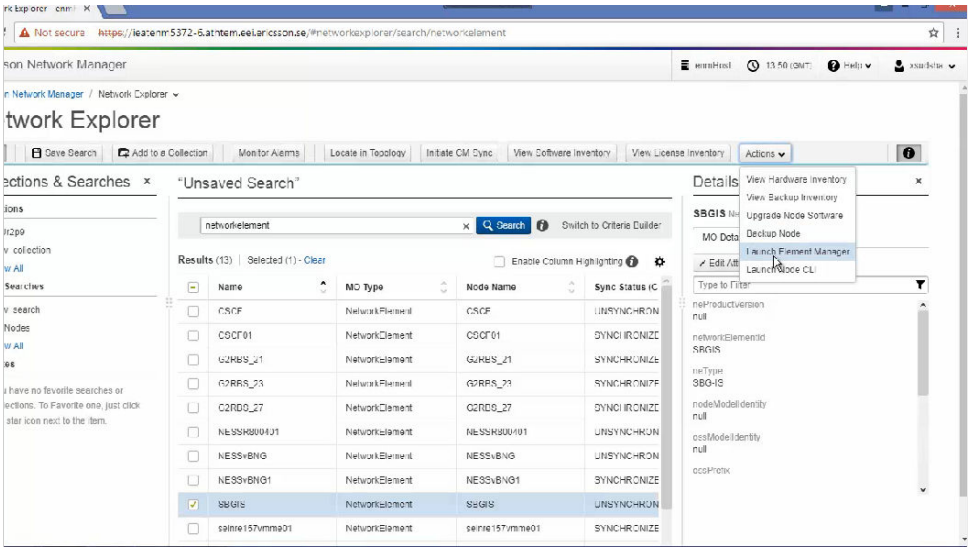
Install SBG(IS) (SIS server certificate) certificate to the management host and launch SBG-IS Element Manager. If port 443 is not open for HTTPS, the URL is redirected to HTTP port 80 on the node.

Prerequisites

`Network_Explorer` and `Element_Manager_Operator` role or `Element Manager Capabilities` roles.

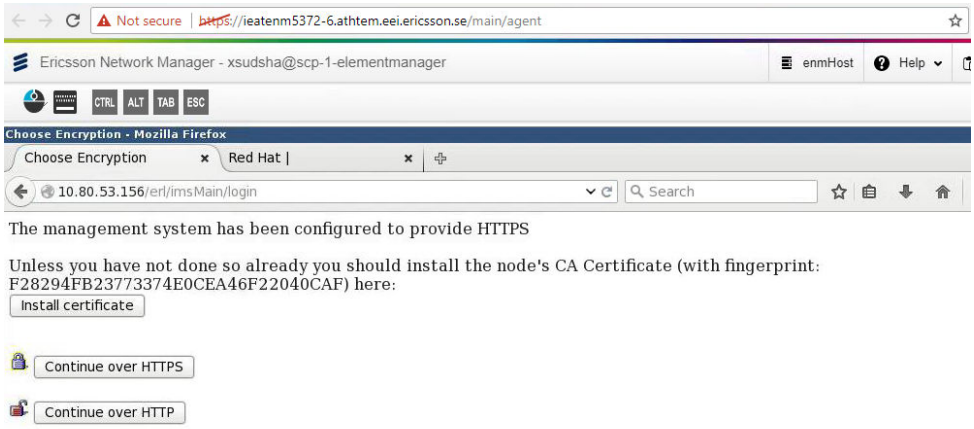
Steps

1. Launch Network Explorer.
2. Select a node, and from the **Actions** drop-down list, select **Launch Element Manager**.



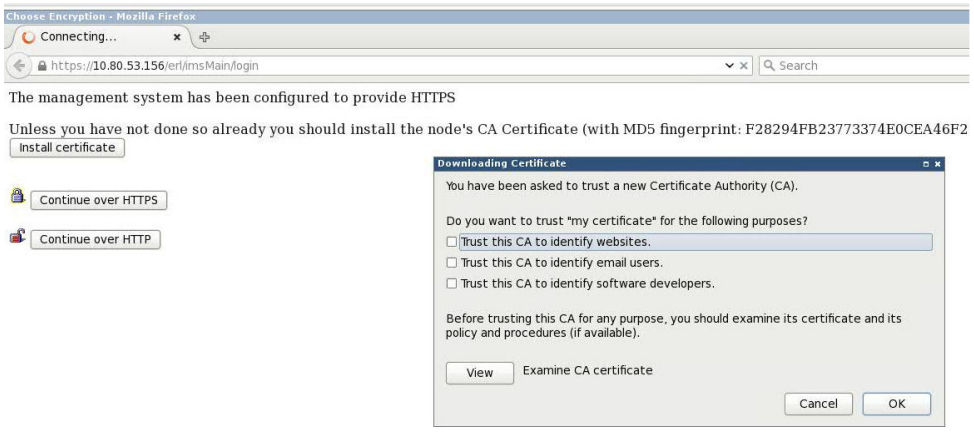
Result: Element Manager is launched in **Cendio Thinlinc** desktop.

3. If connecting for the first time, click **Install certificate** and continue to Step 5, otherwise skip to Step 5.



Result: The **Download Certificate** pop-up displays.

4. Select all check boxes in the pop-up menu, and click **OK**.

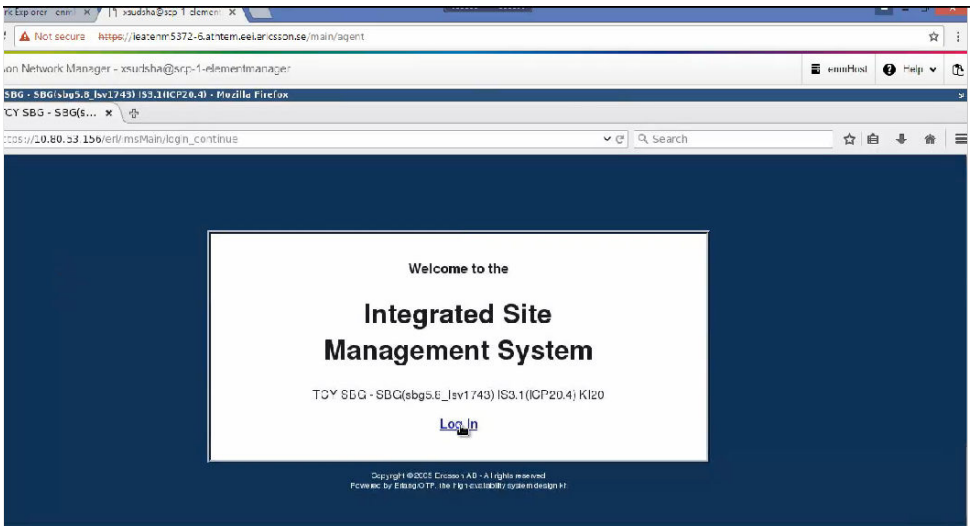


Result: SIS server certificate is installed.

5. Click **Continue over HTTPS** to go to the **ISM GUI Welcome** page.

Result:

ISM GUI Welcome page displays.



3.18 Command to Set Release Attribute for ENIQ

Describes the command to set the release attribute value on the `NetworkElement` MO, that gets populated into Node version attribute in ENIQ topology file.

Prerequisites

— `Network_Explorer` and `Cmedit_Operator` role.

Or

— `Cmedit Capabilities` role.

Steps

1. Execute the following command from ENM CLI. Command sets the release attribute on `NetworkElement` MO for WMG, vWMG, HSS-FE, vHSS-FE, vWCG, SBG-IS, SAPC, vSAPC, BSP, along with add node commands.

   ```
   cmedit set NetworkElement=<NodeName> release=<node version>
   ```

   Example

   ```
   cmedit set NetworkElement=CORE29vWCG002 release=1.0
   ```

   Note: For `<node version>` value in Step 1 `cmedit` command, see *Node Version* in *Supported Network Elements - ENIQ-S, 2/1029-AOM 901 163*

3.19 D-SC Dashboard Launch

Describes the steps to launch the D-SC Dashboard (Element Manager).

Prerequisites

The user needs the following roles assigned:

— `Network_Explorer`.

— `SECURITY_ADMIN`.

— `Element_Manager_Operator`.

— `Scripting_Operator`.

— `Cmedit_Operator` or `Cmedit Capabilities`.

Node Configuration:

For detailed steps on D-SC Dashboard prerequisites and configurations, see *Dashboard GUI (23/1553-CSA 121 20/6)*, available in the D-SC CPI.

3.20 Install Certificate to Launch MSC-BC-IS Element Manager

Install `MSC-BC-IS` (SIS server certificate) certificate to the management host and launch **MSC-BC-IS Element Manager**.

Prerequisites

— `Network_Explorer` and `Element_Manager_Operator` role.

Or

— `Element Manager Capabilities` role.

Steps

1. Launch **Network Explorer**.
2. Select a node.
3. From the **Actions** drop-down list, select **Launch Element Manager**.

Note: If the node CA certificate is not installed, install it before performing the next step.

To install the certificate, click **Install Certificate**, and in the **Download Certificate** pop-up, select all the check boxes, then click **OK**.

4. Click **Continue over HTTPS** button to go to the **ISM GUI Welcome** page.

3.21 Install Certificate to Launch HLR-FE (IS) Element Manager

Install `HLR-FE (IS)` (SIS server certificate) certificate to the management host and launch **HLR-FE (IS) Element Manager**.

Prerequisites

— `Network_Explorer`.

And

— `Element_Manager_Operator` roles

Or

— `Element Manager Capabilities` roles

Steps

1. Launch **Network Explorer**.
2. Select a node, then from the **Actions** drop-down list, select **Launch Element Manger**.
3. If connecting for the first time, click **Install certificate** button.
4. Select all check boxes, then click **OK**.
5. Click **Continue over HTTPS** to go to the **ISM GUI Welcome** page.

3.22 Overview of GSM and AXE Node Support

Management of AXE-based nodes (BSC, MSC and HLR-FE families) in ENM.

3.22.1 Node Topology

BSC

BSC nodes are added to ENM in the normal way.

The neType BSC must be used to create such a node.

MSC

MSC nodes are added to ENM in the normal way.

For the variants of MSC, ENM has several neTypes to be used as appropriate: MSC-DB, MSC-DB-BSP, vMSC, MSC-BC-IS, MSC-BC-BSP, vMSC-HC, IP-STP, IP-STP-BSP, vIP-STP. For more detail, see the *Supported Network Element* document.

HLR-FE

HLR-FE nodes are added to ENM in the normal way.

For the variants of HLR-FE, ENM has several neTypes to be used as appropriate: HLR-FE, HLR-FE-BSP, HLR-FE-IS, vHLR-FE. For more detail, see the *Supported Network Element* document.

Base Station

— DU RadioNodes (BTSs) are not modeled as nodes in ENM. They do not require any manual configuration when adding BSCs. They are represented as MOs in the BSC data model.

  The following areas of support are available for BTSs: FM, CM, and SHM. This functionality is available by selecting the controlling BSC.

— Baseband RadioNodes are modeled as nodes in ENM, whether they have GSM, WCDMA or LTE functions on them.

LAN Switch

— NWI-E LAN Switches are not modeled as nodes in ENM. Connectivity information for them must be configured in extra attributes of the BSC's NetworkElement definition. 0, 1 or 2 NWI-E LAN Switches can be configured for each BSC, and can be defined when adding the node, or later.

  The following areas of support are available for NWI-E LAN Switches: FM and Node CLI. This functionality is available by selecting the controlling BSC.

— Note, other types of LAN Switch are not supported at all in ENM.

APG

APGs are not modeled as nodes in ENM. Instead, connectivity information for an APG must be configured in extra attributes of the BSC/MSC/HLR-FE NetworkElement definition. An APG must be configured for each BSC/MSC/HLR-FE.

Some node types support the option of Dual APGs. As well as the mandatory connectivity attributes for one APG, the relevant NetworkElement also has optional connectivity attributes for defining a second APG. A second APG can be defined when adding a node, or later.

The following area of support is available for APGs: SHM. This functionality is available by selecting the relevant BSC/MSC/HLR-FE.

Note: For AXE Nodes (MSC, HLR and BSC) nodes, TLS is supported from APG43L 3.6 and on-wards. For nodes which had lower APG43L version use SSH.

3.22.2 Authorization

Setup

OSS-RC has application TSS, that handles user authorization, and TBAC for AXE-based node, that are the MML commands a user sends to each node.

When managed by ENM, the nodes perform these checks, as needed, at runtime.

The setup process for authorization is as follows:

1. COM roles are defined on the BSC/MSC/HLR-FE nodes - these can be user-defined, or predefined on the nodes. Each COM role grants a certain amount of access on the node.
2. The same COM roles are defined in ENM Role Management.
3. Each user can then be assigned 1 or more of these COM roles as appropriate by ENM User Management, and so this data is synced to LDAP.

The setup process for node-based TBAC is as follows:

1. Configure the node to turn on TBAC.
2. Target groups are defined in ENM - each containing some nodes.
3. For each COM role assigned to a user, target groups can be assigned - this data is also synced to LDAP.

At runtime, when a user accesses a node, the node checks LDAP to see what COM roles the user has on that particular node.

Note: The node-based TBAC is different from ENM TBAC support for applications that is detailed in the Identity and Access Management System Administrator Guide.

Runtime Access from WinFIOL, OPS, and SHM Applications

For BSC/MSC/HLR-FE nodes configured for TLS, when a user performs an action towards one of these nodes from WinFIOL, OPS or SHM (upgrade use case), then the real ENM user-id is used in communication with the node. The node checks LDAP to see what COM roles the user has for this node, and rejects or handles the request as appropriate.

Runtime Access from Model-Based CM Tools.

These are the tools that give CM support of the BSC modeled data - for example, **Import**, **ENM CLI**, **Topology Browser**.

When a user performs an action towards one of these nodes from these applications, the real ENM user-id is not used - instead COM-user is used in communication with the node. Thus the real ENM user COM roles are disregarded in this scenario; the COM-user is permitted to make all changes.

Note:
- These Model-Based CM Tools have this same behavior for all other node types.
- Auditing in ENM will log the real ENM user-id and the action performed.

Runtime Access from Cell Management

When Cell Management writes BSC data, it uses the modeled data replicated in ENM. Thus the COM-user is used towards the node, as per the previous Model-based CM case.

When Cell Management writes MSC data configured for TLS, it uses the real ENM user-id towards the node, as in the previous WinFIOL case.

3.22.3 Fault Management

FM for BSC/MSC/HLR-FE nodes is supported in the suite of FM applications, like **Alarm Overview**, **Alarm Monitor**, **Alarm Search**.

Alarms for DU RadioNodes (BTSs) and NWI-E LAN Switches are available by the controlling BSC.

3.22.4 Performance Management

Setup

The PMIC application has subscriptions to support various PM data types from BSC/MSC/HLR-FE nodes. However the subscriptions only control collection of data. Activation or deactivation of a subscription does not initiate or terminate data production on the node.

Initiation or termination of PM Statistics (STS) and BSC Recordings (BAR, RIR, MRR, CER, CTR) must be done manually on the nodes - for details on these procedures see the *Performance Management System Administrator Guide*.

File Formats

PMIC makes STS files available NB in the same format (ASN.1) that is available from OSS-RC - not in 3GPP XML format.

For BAR, RIR, MRR, CER, CTR, and MTR files PMIC provides an option to store decoded versions of these files. Based on customer settings, the original encoded files collected from nodes, and the decoded files, are available in the **PM NBI**. See the *Performance Management System Administrator Guide* for details.

3.22.5 Configuration Management

OSS-RC has applications like CNA and BSM to help configure BSC nodes. With the introduction the Managed Object Model (MOM), ENM supports configuration of this BSC data through ENM Model-based CM Tools, for example, **Import**, **ENM CLI**, **Topology Browser**.

ENM has WinFIOL and OPS applications which support sending commands direct to the BSC, MSC and HLR-FE nodes. These applications behave similarly to OSS-RC.

Note: On ENM, the BSC CM Bulk Synch Flow with BSC node user LocalCOMUser is controlled by PIB parameter useLocalCOMUser, described in *ENM Parameter List 1/190 59-AOM 901 151*. However, if all BSC nodes managed by ENM are with APG43L 3.7 and higher, then this PIB parameter can be set to FALSE.

3.22.6 Software and Hardware Management

ENM SHM application supports the following use cases per node family:

Use Case	BSC	MSC	HLR-FE	BTS	Comment
Upgrade	Yes	Yes	Yes	Yes	
Backup	Yes	Yes	Yes	n/a	
SW Inventory	Yes	Yes	Yes	Yes	

Use Case	BSC	MSC	HLR-FE	BTS	Comment
HW Inventory	Yes	Yes	Yes	n/a(1)	(2) This data is available in the BSC MOM.
License Management	Yes	Yes	Yes	n/a	

(1) This data is available in the BSC MOM.

3.23 PM Job Name and File Path for CUDB/vCUDB Nodes

For CUDB/vCUDB nodes, the following is the default path for PM file collection on the node:

```
/home/cudb/oam/performanceMgmt/output
```

For ENM and ENIQ-S integration of CUDB nodes, PM jobs in CUDB must be with name as counter group, unlike in OSS-RC where OSS-RC/SGW application created PM jobs with job appended to counter group.

Default PM jobs in CUDB are created with job name as counter group names.

Example:

Previous format on the node when integrating with OSS:

```
A20190313.1145-1200-REG23AUCSUBSjob_CUDB2.xml
```

Expected format on the node for ENIQ-S integration with ENM:

```
A20190313.1145-1200-REG23AUCSUBS_CUDB2.xml
```

3.23.1 Centralized User Management for vSAPC Node with LDAP

Note: An operator can access the PM file collection on node side in the directory /home/eir/oam/Performancemanagement/output only with LDAP username and password.

LDAP must be configured on the node before testing the features in ENM. User management must be supported on the node. This procedure must be verified on the node when integrated with ENM.

A workaround configuration on node is needed to set up CustomRole with a rule to allow access to ManagedElement attributes including dnPrefix for ENM CM sync to work.

Follow these steps to verify LDAP functionality from ENM side:

Steps

1. Create an LDAP user in User Management UI.
2. Add at least one of the following two COM roles or both for NETCONF or SSH to work.
   - `SapcSystemAdministrator`
   - `SapcSystemSecurityAdministrator`
3. Set username and password to the user and roles are assigned to that particular user.

   ```
   __ secadm credentials create --secureusername <username> --secureuserpasswor →
   d <password> -n <nodename>
   ```

   Credentials are created successfully.
4. Log on to ENM with the created user and test the Offline enrollment feature.

   This feature is supported for SAPC or vSAPC from ENM, once LDAP is configured on the node. Online enrollment is not supported for this node from ENM.
5. Retrieve the LDAP Configuration details using the command `secadm ldap configure` with manual option (`--manual | -ml`) on WEBCLI:

   ```
   secadm ldap configure --manual
   ```

3.24 LDAP Support for EIR-FE/vEIR-FE (ESA-Based Node)

Note: The PM file collection on node side, in the directory `/home/eir/oam/Performancemanagement/output`, can only be accessed using the `LDAP` username and password.

`LDAP` must be configured on the node before testing the features in ENM. User management must be supported on the node.

For `EIR-FE/vEIR-FE`, to configure `LDAP` on node see *LDAP Authentication Configuration* in the *EIR-FE System Administration Guide - 5/1553-CSA 101 25/2-V1*.

This procedure must be verified on the node when integrated with ENM.

Use the following procedure to verify `LDAP` functionality from ENM side.

1. Create a `new user` in `User Management` UI.
2. Add all the COM-Based roles to the `new user`:

Roles: Admin, Amos Admin, All COM roles (COM target all).

3. Set `username` and `password` for the new user using the following command:

```
__secadm credentials create --secureusername <username> --secureuserpassword
 <password> -n <nodename>
```

4. Log in to ENM as the `new user` to test the EIR-FE/vEIR_FE features, when `LDAP` is configured on node.

5. Retrieve the LDAP Configuration details using the following command on WEBCLI:

```
secadm ldap configure --manual
```

3.25 Auto-Configure SNMP Security for Baseband RadioNode and VNF

The security level of the SNMP communication channel between ENM and the Baseband RadioNode Network Elements Autointegration requires a secured SNMP communication channel between node and ENM.

There are two ENM level parameter settings that may be used to automate the SNMP configurations with the wanted values when adding the node to the network.

1. NODE_SNMP_SECURITY

   Used to auto-configure OAM IP Address change and FM SNMP communication channels.

2. NODE_SNMP_INIT_SECURITY

   Used to auto-configure the snmp channel for nodeup communication during initial stages of autointegration process. Once the nodeup hello message is successfully consumed by ENM, these SNMP parameter settings are no longer used for this node.

All Baseband RadioNodes being added to the network using AutoProvisioning will have their SNMP configured as per the above parameters. If the parameters are modified then all subsequent Baseband RadioNode integrations will take on the new parameter values.

For details of these parameters, see *ENM Parameter List*, 1/190 59-AOM 901 151.

The following content gives information on how to view and configure these parameters.

3.25.1 View SNMP Security Configuration

Prerequisites

— ENM CLI access.

— One of the following roles:

- CM_Application_Administrator
- Custom role with the capability: "Resource: cm_app_param, Operation: update"

See *User Management* in the *ENM Identity and Access Management Programmers Guide* for user and role configuration in ENM.

Command

To view specified SNMP Security Configuration, use the following command:

```
admin parameter view <attribute>
```

Attribute

The following attribute is required to view specified SNMP Security Configuration.

--name	Specifies SNMP Security Parameter name

Examples

`NODE_SNMP_SECURITY`

```
>>admin parameter view --name NODE_SNMP_SECURITY
NODE_SNMP_SECURITY: {securityLevel:NO_AUTH_NO_PRIV,authProtocol:NONE,authPasswor →
d:,privProtocol:NONE,privPassword:,user:defaultsnmpuser}
```

`NODE_SNMP_INIT_SECURITY`

```
>>admin parameter view --name NODE_SNMP_INIT_SECURITY
NODE_SNMP_INIT_SECURITY: {securityLevel:NO_AUTH_NO_PRIV,authProtocol:NONE,authPa →
ssword:,privProtocol:NONE,privPassword:,user:defaultsnmpuser}
```

3.25.2 Modify SNMP Security Configuration

Prerequisites

— ENM CLI access.

— One of the following roles:

  - CM_Application_Administrator
  - Custom role with the capability: "Resource: cm_app_param, Operation: update"

See *User Management* in the *ENM Identity and Access Management Programmers Guide* for user and role configuration in ENM.

Command

To modify specified SNMP Security Configuration, use the following command:

```
admin parameter modify <attributes>
```

Attributes

The following attributes are required to modify specified SNMP Security Configuration.

--name	Specifies SNMP Security Parameter name
--value	Specifies SNMP Security Parameter value

Examples

NODE_SNMP_SECURITY

```
>>admin parameter modify --name NODE_SNMP_SECURITY --value {securityLevel:AUTH_P →
RIV,authProtocol:SHA1,authPassword:onlytest,privProtocol:AES128,privPassword:nod →
eSnmpPassword,user:nodeSnmpUser}
Parameter NODE_SNMP_SECURITY updated successfully.
```

NODE_SNMP_INIT_SECURITY

```
>>admin parameter modify --name NODE_SNMP_INIT_SECURITY --value {securityLevel:A →
UTH_PRIV,authProtocol:SHA1,authPassword:onlytest,privProtocol:AES128,privPasswor →
d:nodeUpHelloSnmpPassword,user:nodeUpSnmpUser}
Parameter NODE_SNMP_INIT_SECURITY updated successfully.
```

Note: This table displays the allowed values for NODE_SNMP_SECURITY and NODE_SNMP_INIT_SECURITY parameters, and the node MOs that are subsequently auto-configured during autointegration.

Field Name	Field Value	Default Value	Configure Node in Integration
securityLevel	NO_AUTH_NO_PRIV AUTH_NO_PRIV AUTH_PRIV	NO_AUTH_NO_PRIV	— SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=1 snmpSecurityLevel — SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=ENMFM snmpSecurityLevel
authProtocol	NONE MD5 SHA1	NONE	— SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=1 authProtocol — SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=ENMFM authProtocol
authPassword	String, can not contain space and the character "," can not be used.	Empty String	— SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=1 authKey — SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=ENMFM authKey With encrypted value.
privProtocol	NONE DES AES128	NONE	— SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=1 privProtocol — SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=ENMFM privProtocol
privPassword	String, can not contain space and the character "," can not be used.	String, can not contain space and the character "," can not be used.	— SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=1 privKey — SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=ENMFM privKey With encrypted value.
user	String[1..20], can not contain space, the characters "," and ":" can not be used.	defaultsnmpuser	— SystemFunctions=1,SysM=1,Snmp=1,SnmpTargetV3=1 user — set to 'ECIMUser', when securityLevel is NO_AUTH_NO_PRIV — set to 'user' value with a unique hash suffix appended when securityLevel is AUTH_NO_PRIV / AUTH_PRIV, e.g. for user=snmpuser, SnmpTargetV3.user=snmpuser17d0ce32 — SystemFuncons=1,SysM=1,Snmp=1,SnmpTargetV3=ENMFM user — set to 'user' value

3.26 Performance Management for GenericESA-based Nodes

Activation of a subscription schedules file collection tasks. Collection tasks are based on the set of nodes included in the subscription, and the reporting output period (ROP) supported by the node. For a `GenericESA` type of node, a 15-minute ROP interval is supported. PM jobs must be initiated on node. There are no counters available on ENM to select, and the corresponding file collection is initiated in PMIC.

— For `GenericESA` type of nodes, the default path for PM file collection on the node is:

`/var/log/esa/pm3gpXml`

— When initiating `PM Statistical Subscription`, when the node is added to the subscription, the counter tab is disabled. This is because there is not any selection of counters.

3.27 Accessing Element Manager GUI for SCEF Node

Prerequisites

ENM user must be assigned the following `SCEF Portal GUI` specific roles to launch `SCEF Portal GUI` or `Element Manager`:

Table 2 SCEF Portal GUI Specific Roles

Role Name	Web Console	Configuration Manager	Central Logging (Kibana)	Metrics GUI (Performance Management)
`generic_viewer`	Read Only	Read Only	Read and Create Index	Read Only
`generic_admin`	Read and Write	Read and Write	Read and Create Index	Read Only
`viewer_cm_pm_wc`	Read Only	Read Only	No access right(1)	Read Only
`admin_cm_pm_wc`	Read and Write	Read and Write	No access right(1)	Read Only

(1) ENM user with this role cannot access Kibana from SCEF Administrator Portal.

Steps

1. Log in to ENM as administrator.
2. Create custom roles, named as in Prerequisites, in `Portal GUI Role Name`, with the appropriate capabilities.

3. Create User in User Management UI, and assign with appropriate node specific roles.
4. Log in to ENM as `new User`.
5. Check if SSO is enabled for node.

   If SSO not enabled, run the following commands:

   ```
   cmedit get NetworkElement=<Node Name>,EsaConnectivityInformation=1

   cmedit set NetworkElement=<Node Name>,EsaConnectivityInformation=1 enableSSO →
   =True
   ```

6. Launch Network Explorer.
7. Select a node, and from the `Actions` drop-down, select `Launch Element Manager`
8. If the User has node specific roles, Element Manager is launched.

   If the User does not have the node specific roles, a pop-up message is displayed, stating `"Failed to launch. User is not Authorised to launch ENM GUI"`

3.28 Online Certificate Enrollment on Fronthaul 6020

> How to perform the online enrollment of Ericsson Fronthaul 6020.

Enrollment assigns node credentials. This process is called Online Enrollment Procedure. The node is enrolled through the CMPv2 protocol, obtaining certificate from PKI-RA.

Target Groups

— Security Administrators

— System Administrators

— ENM Operators

— Installation Engineers

This procedure is only applicable for OAM Enrollment. Do not use for IPSEC Enrollment.

Details of the operations are available in *Manage Security, 18/1553-LZA 701 6014*.

Upon enrollment the Fronthaul node is added in ENM and synchronized with TLS.

3.28.1 End Entity Creation and Credential Generation

End Entities (EE) of the PKI System are the end users who get credentials from the ENM PKI System. They use it for communication with other ENM systems. End Entities must be created in the PKI system.

To generate credentials each End Entity is mapped to an Entity Profile (EP) that defines the Certificate Authority (CA).

For more information about Entity and Profiles, see *Public Key Infrastructure System* of the *ENM Public Key Infrastructure System Administrator Guide, 2/1543-AOM 901 151-3*.

Details of operations are available in the *Manage Security, 18/1553-LZA 7016014*.

Prerequisites

- The operators' knowledge/access to the following,
  - ENM server deployment structure.
  - The PKI system and its commands.
  - Root access privileges to log in to the Management Server and virtual machines.
  - Root access privileges to log in to the DB nodes.
  - ADMINISTRATOR role to access the CLI in ENM.
  - Configuration of MOs on the Node.
- Release 20.Q2.1 or higher version of the FH6020 node, type SW.
- The node is SYNCHRONIZED with SSH protocol.

Steps

1. Access the node through Node CLI and execute the following commands in config mode to enable TLS on the node:

```
[config]#netconf-tls-edit netconf-tls-port 6513 admin-status UNLOCKED
[config]#commit
```

2. Create a new XML File for each FH6020 series Node:

```
<?xml version="1.0" encoding="UTF-8"?>
<Nodes>
<Node>
<NodeFdn>NetworkElement-FDN</NodeFdn>
<EnrollmentMode>CMPv2_INITIAL</EnrollmentMode>
```

```
</Node>
</Nodes>
```

Note: The <NodeFdn> value must be NetworkElement-Fdn.

3. Import the XML file to ENM web CLI.
4. Perform online enrollment for the node:

```
secadm cert issue -ct OAM -xf file:<End_Entity_xml File>
```

Result: A link displays.

5. Click the link generated as result of the preceding step to check the enrollment job status.

Result: If Job Status is COMPLETED and Workflow Status is SUCCESS, the enrollment of the node is done and you can continue to the next step. If not, contact Ericsson Local Support.

6. When the node has successfully enrolled, switch it from SSH to TLS.
    a. Turn off CM supervision:

```
cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 a →
ctive=false
```

    b. Set transportProtocol in FrontHaul6000ConnectivityInformationMO to TLS:

```
cmedit set NetworkElement=nodename,FrontHaul6000ConnectivityInforma →
tion=1 port=6513

cmedit set NetworkElement=nodename,FrontHaul6000ConnectivityInforma →
tion=1 transportProtocol=TLS
```

    c. Turn on CM supervision:

```
cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 a →
ctive=true
```

    d. Check the syncStatus:

```
cmedit get NetworkElement=<nodeName>,CmFunction=1
```

Result: The syncStatus attribute of CmFunction MO must be SYNCHRONIZED.

4 Software and Hardware Manager Overview

> The **Software Hardware Manager (SHM)** is a web-based application for managing the software, hardware, and licenses of Network Elements (NEs).

SHM provides the following functions for the management of the NEs:

— Hardware Administration.

— Software Administration.

— Backup Administration.

— License Administration.

SHM application uses the job concept for functionality that includes node operations such as upgrade, backup, restore, and license installation. A job comprises one or more activities to be performed on one or more nodes. A job is created for a collection or list of nodes and it contains Network Element (NE) level job for each node. Each NE level job can contain one or more activities. With **SHM** web application, user can supervise multiple jobs towards multiple Managed Elements (MEs) from a single client. **SHM** application enables user to view and analyze the jobs created by users.

SHM does not apply restrictions on the number of jobs executed in parallel. The recommendation is to execute a maximum of five different SHM jobs with 1000 nodes in parallel. To avoid resource usage issues, do not execute more than five large SHM jobs in parallel.

The following operations can be performed in **SHM** application:

— Upgrade Nodes - The Software Upgrade feature enables user to install a software package on a node/NE. Software Upgrade is used to add new functions and improve performance and correct faults in the existing software installed on a node.

— Backup Nodes - This feature is used to create a backup job on a selected set of nodes. The created backup job has the combination of software, configuration, and other vital information stored in a file on the node.

— Install License Keys - Install License feature enables user to install the selected license key file on one or more nodes.

— Restore Backup - This feature is used to create a restore backup job on selected nodes. This function makes it possible to restore the node to a configuration defined in a previously downloaded backup.

— Backup Housekeeping - This feature is used to create a backup housekeeping job on selected nodes. This function makes it possible to clean backups on the specified nodes.

— Backup Administration - The Backup Administration page enables user to view the stored backups (Configuration Versions if CPP) available on specified nodes.

— Hardware Administration - The Hardware Administration page enables user to view the hardware information of the nodes, this includes Hardware Type, Serial number, Unit Location and so on.

— License Administration - The license administration page enables user to view and manage the license information on the node/NE. License Inventory displays the summary of the Licenses active on the node. This includes: Finger Print, Sequence Number, installation date and so on.

— Software Administration - The Software Administration page enables the user to view and analyze the software running in the network. SHM fetches software inventory information about software versions and software items from each node on the network to display the information to the user. It also provides details about the upgrade packages available on the node to enable the user to select the unwanted upgrade packages and delete them.

— Import License Key Files - This feature enables the user to import a zip file containing a set of license key files and license data files into Ericsson Network Manager (ENM).

  These imported license files are installed to the respective nodes based on the requirement.

— Generate License Request File (LRF) - This feature enables user to generate License Request File to request for required features capacity on the node based on actual node configuration.

— Import Software Packages - This feature enables user to import software packages into Ericsson Network Manager (ENM). These imported software packages are installed to the respective nodes based on the requirement.

— Onboard Software Packages - This feature enables user to onboard VNF software packages from Ericsson Network Manager (ENM) to Network Function Virtual Orchestrator (NFVO).

Prerequisite

SHM uses web push to execute many of the use cases. To make sure that the web push works properly, it is necessary for the URL to have the required certificates on browser. To install browser certificates, see *ENM Identity and Access Management System Administrator Guide*, 2/1543-AOM 901 151-1 Uen.

Note: SHM jobs depend on services such as workflows and DPS. SHM incorporates retries to handle ENM upgrade scenarios. However, the jobs might hang or fail if the dependent VM is lost (for example, because of a reboot of the VM, or during an ENM upgrade) even after performing retries.

4.1 Node Software Upgrade using Software and Hardware Manager

This section describes the following:

— SHM Node Software Upgrade Workflow.

— SHM 'Treat-As' Node Software Upgrade.

4.1.1 SHM Node Software Upgrade Workflow

> SHM enables you to upgrade the software running on the nodes in your network.

To upgrade the software, create a Software Upgrade job. This job specifies the nodes to upgrade, the software to upgrade them with, and the relevant upgrade activities to perform. The upgrade activities depend on the node types specified in the job. Upgrade activities for CPP-based nodes (such as ERBS, MGW) are different to the upgrade activities for ECIM-based nodes (such as baseband radio node, SGSN-MME), and for AXE-based nodes, such as BSC, MSC, and HLR-FE. The activities are fetched on-demand from the selected software package. You can schedule when the job, and the related upgrade activities, are performed.

— Use the SHM Job scheduling to reduce the impact of the upgrade on the network.

  For example, create a job that performs the upgrade activities that do not interrupt traffic (such as, Install, Prepare, Verify). Then it performs the traffic affecting upgrade activities at a later scheduled time (during a maintenance window). Or create two separate Upgrade jobs, the first performing the non-traffic-affecting upgrade activities, and the second performing the traffic-affecting upgrade activities.

— Some of the upgrade job activities, such as the Upgrade and Confirm activities (for CPP-based nodes) or the Activate and Confirm activities (for ECIM-based nodes), and activities for AXE-based nodes, might impact network traffic, as they require a node restart. Perform these activities during an appropriate window.

Before you can upgrade the nodes, you must import the relevant software packages into SHM. This is done using the **SHM Import Software Package** functionality.

In the **Software Hardware Manager** page, use the **Jobs** table to monitor the overall progress of an Upgrade job. The **Progress** column displays the job progress in terms of percentage. The upgrade job progress is updated according to the completion of upgrade activities on the nodes specified in the job.

You can also monitor the job progress at individual node level specified in the job using **Job Details** page and also at individual node activities level using **Node Activities Panel**.

In the **Jobs** table, select the relevant **job** > **View Job Details** > **select the relevant node from the Node Progress table** > **Node Activities**. In the **Node Activities** panel, you can see the progress of the job activities performed on the node.

SHM performs internal validation before actually starting the desired activity on the node to ensure that node is in right state to perform the activity.

For CPP-based nodes, for each job activity in the **Node Activities** panel, the progress bar and progress percentage increases whenever a validation is completed, an action on a node is started, and an activity is completed

For ECIM-based nodes, for each job activity in the **Node Activities** panel, progress bar and progress percentage increases according to progress percentage notifications received from node.

For AXE-based nodes, validation is not applicable, but you can monitor Script Execution response from OPS. To launch **OPS GUI**, in **Node Progress** table select the relevant NE job, right click, then select **Launch OPS GUI**.

Node Selection in Upgrade Job Configuration

— You can create a single upgrade job that upgrades software for different node types (such as ECIM-based, CPP-based nodes, or AXE-based).

— When you create an Upgrade Job, specify the nodes you want to upgrade. You can specify individual nodes, or use collections or saved searches, to specify the nodes. The selected nodes are evaluated during the job execution time.

— For AXE-based nodes, while creating an Upgrade Job, there are Sub Components for some nodes such as MSC-BC-BSP, MSC-BC-IS. By default Cluster component is selected, and you can select other desired components from **Node components** tab.

SHM does not apply restrictions on the number of nodes in a job. However, the recommended maximum number is 1000 nodes in a single job.

To avoid resource usage issues, do not execute large SHM jobs in parallel.

Note: In SHM, you cannot install a software package on a node that is already running with the same software package. To ensure this, SHM checks the product numbers and revisions of the software installed on the node with those of the software package to be installed.

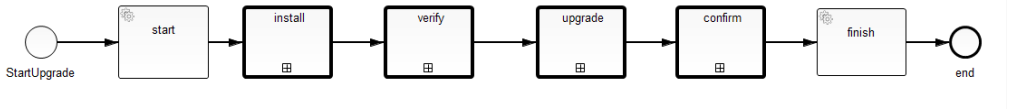
In the **Create Upgrade Job** wizard, if you select the **Validate** option (in the **Select Upgrade Package** step), nodes that are running with the software package you want to install are skipped and are not included in the upgrade job. If you do not select the **Validate** option, SHM performs a check when executing the upgrade job, and skips any nodes running with the software package you want to install.

This behavior can be modified using the configuration parameters mentioned in *SHM: Configuration parameter for Node Software Upgrade* under *Software and Hardware Management Administration Tasks* section of *ENM System Administrator Guide*.

Note: For AXE-based nodes, Software Package validation is not applicable.

4.1.1.1 Upgrade Job Workflow for CPP-Based Nodes

> Describes the upgrade activities that are executed sequentially on each node when an Upgrade Job executes for CPP-based nodes.



Install Activity

This activity downloads the software package from ENM to node. If the software package is already available on the node, this activity is skipped.

Note: The `Install` activity requires an SFTP connection between ENM and the node. This activity might fail if the SFTP connection between the node and the relevant ENM VM is lost (for example, because of a reboot of the VM, or during an ENM upgrade).

Verify Activity

This activity verifies if the downloaded software package can be used to upgrade the node.

Upgrade Activity

This activity upgrades the node with the software package that is downloaded and verified.

Confirm Activity

This activity makes the upgraded node software permanent. If the confirm activity is not selected as part of upgrade procedure, and the confirm is not issued

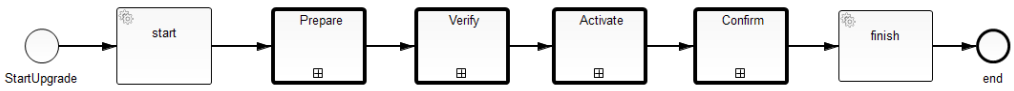
within the time-out interval, the node rolls back to the previous version. For default time-outs related to different node types, see the node documentation.

Note:
— The `Install` and `Verify` activities do not cause interruption to network traffic. These activities can be performed at any time.

— The `Upgrade` and `Confirm` activities might impact network traffic, by causing the nodes to restart. This might result in a full sync of CM data). Perform the `Upgrade` and `Confirm` activities during the maintenance windows.

4.1.1.2 Upgrade Job Workflow for ECIM-Based Nodes

> Describes the upgrade activities that are executed sequentially on each node when an Upgrade Job executes for ECIM-based nodes.



Prepare Activity

This activity downloads the software package from ENM to node. If the software package is already available on the node, this activity is skipped.

Note: The `Prepare` activity requires an SFTP connection between ENM and the node. This activity might fail if the SFTP connection between the node and the relevant ENM VM is lost (for example, because of a reboot of the VM or during an ENM upgrade).

Verify Activity

This activity verifies if the downloaded software package can be used to upgrade the node.

Activate Activity

This activity activates the upgrade package that is downloaded and verified. It involves a node restart based on the software package that is used for the upgrade.

Confirm Activity

This activity makes the upgraded node software permanent. If the confirm activity is not selected as part of upgrade procedure, and the confirm is not issued within the time-out interval, the node rolls back to the previous version. For default time-outs related to different node types, see the node documentation.

Note:

— The `Prepare` and `Verify` activities do not interrupt network traffic. These activities can be performed at any time.

— The `Activate` and `Confirm` activities might impact network traffic, by causing the nodes to restart. This might result in a full sync of CM data).

Perform the `Activate` and `Confirm` activities during maintenance windows.

4.1.1.3 Upgrade Job Workflow for MINI-LINK Indoor Nodes

> Describes the list of activities that are executed sequentially on each node when an upgrade job executes for MINI-LINK Indoor nodes.

Download Activity

This activity downloads the software package from ENM to node.

Note: The `Download` activity requires an SFTP connection between ENM and the node. This activity might fail if the SFTP connection between the node, and the relevant ENM VM is lost (for example, because of a reboot of the VM or during an ENM upgrade).

Activate Activity

This activity activates the software package that is downloaded. It involves a node restart based on the software package that is used for node software upgrade.

Confirm Activity

This activity makes the upgraded node software permanent. If the confirm activity is not selected as part of upgrade procedure, and is not issued within the time-out interval, the node rolls back to the previous node version. For default time-outs related to different node types, refer to node documentation.

Note: The `Activate` and `Confirm` activities might impact network traffic, by causing the nodes to restart. This might result in a full sync of CM data. Perform the `Activate` and `Confirm` activities during maintenance windows.

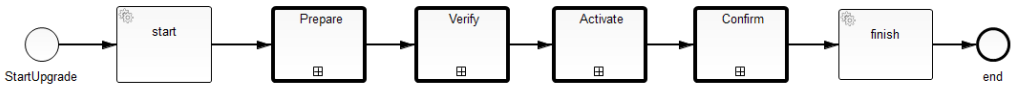
4.1.1.4 Upgrade Job Workflow for vRAN Nodes

> Describes the list of activities that are executed sequentially per node during an upgrade job execution that involves vRAN nodes.

SHM interacts with RAN-VNFM using a proprietary ep-LCM-em REST interface to do software upgrade for vRAN nodes. (For example, vPP, vSD).

VNF software package must be onboarded to NFVO before performing the upgrade on vRAN nodes.

Internally, RAN-VNFM controls the detailed steps with a dedicated workflow, delivered as part of the VNF Package.



Prepare Activity

The prepare activity involves the creation of an upgrade job in RAN-VNFM. Onboard the package to the NFVO configured for the RAN-VNFM to create an upgrade job.

Once the job is created, a unique job id is returned to SHM. This id is used to perform other activities like prepare, verify, activate, confirm.

SHM starts the prepare activity after creation of the upgrade job.

RAN-VNFM performs all non-traffic affecting checks during prepare phase.

Verify Activity

The verify activity involves evaluation of probabilities by RAN-VNFM to succeed with upgrade activation.

The verification activity is optional.

Activate Activity

The activate activity involves creation of a new parallel VNF using NFVO. After instantiating the new VNF, the traffic is stopped in from-state VNF, and then started in to-state VNF.

This activity involves IP address change of the VNF that is intimated to ENM through SNMP trap.

Confirm Activity

The confirm activity involves the cleanup of resources like `from-state` VNF using NFVO. If the confirm activity is not selected as part of upgrade procedure, and the confirm is not issued within the rollback interval, then node rolls back to previous node version.

Note: `SHM` has an in-built mechanism of load balancing the maximum number of activate activities that can be executed in parallel, based on the network size, and deployment layout.

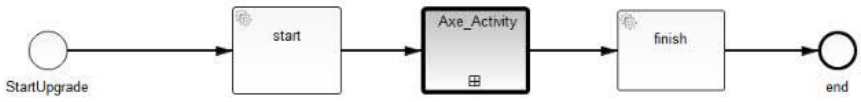
Usually they activate, and confirm that activities are configured to run during the maintenance window. These can cause the node to restart causing an interruption to traffic. This might also cause a full sync of CM data after node restart.

`Cancel` is supported for vRAN upgrade at all activities.

4.1.1.5 Upgrade Job Workflow for AXE-Based Nodes

> Describes the workflow of upgrade activity execution on each node when an Upgrade Job executes for AXE-based nodes.

The following diagram.



For `AXE-based` nodes, activities are rendered dynamically based on the selected upgrade package. Selected activities are executed sequentially for each `AXE-based` node.

For `APG` upgrade, the operator must select `APG-specific` upgrade software package. Then all the `APG` components available on the node are upgraded, irrespective of node and node sub component selection in SHM UI.

`NWI-E` can be upgraded by a manual procedure. See the *RAN CPI* for more details.

4.1.1.5.1 Downloading and Editing the OPS Script and OPS Script Execution Log

To download and edit the ops script, and script execution logs from SMRS, complete the following steps. They list the files under the software package, and download to local (any SFTP client). The user can edit the downloaded script, and copy it to the software package using SFTP.

OPS script in software package, and Logs files created in SMRS, can be downloaded to a local file system and uploaded:

Note: Any modification in software packages post software import in ENM is not recommended.

Prerequisites

1. Secure a user profile with role `FIELD_TECHNICIAN` from the system administrator.
2. Get the `svc_CM_vip_ipaddress` value from your System Administrator:

```
Example to read the svc_CM_vip_ipaddress

grep svc_CM_vip_ipaddress /ericsson/tor/data/global.properties
```

Note: `CM_vip public` IP address used to open the SFTP connection to `smrs` service in ENM.

Steps

1. Establish an SFTP connection to the path `smrsroot/software/axe/` using any SFTP client to the IP address received in the Prerequisite Step 2 and using the user credentials specified in Prerequisite Step 1:

```
sftp <user>@<IP Address>:/smrsroot/software/axe/
```

2. Navigate to software package to list the ops script, or LOG directory to list the logs that are available under LOG directory. Script execution logs are found in node subdirectory under LOG:

   Example

```
/smrsroot/software/axe/<software package>/LOG/<nodeName>/
```

3. Download the log file as listed:

```
cd /<softwarePackage>/
ls -l
cd LOG/<nodeName>/
ls -l
get <log file name>
```

4. Download the script file from software package. Navigate to the software package directory and list the ops script to download:

```
cd <softwarePackage>/
ls -l
get <ScriptName>
exit
```

5. Edit the downloaded script or log using file editor, then **Save** and close the editor.
6. Follow the Step 1 to establish SFTP connection, and navigate to specific directory.
7. Upload the script or log file to software package under SMRS and close the SFTP session:

```
put <file name>
exit
```

4.1.1.6 Deletion of Non-Active Software for BSP Node before SHM Upgrade

> This section describes how to delete a non-active software version on the node.

New software cannot be downloaded if a non-active BSP software version is already present in the system. Hence, before proceeding for SHM Upgrade, it is mandatory to remove the existing non-active software version on BSP node manually. Execute the following steps after connecting to the node through NodeCLI:

Steps

1. Check if there is any non-active software stored:

```
>dn ManagedElement=1,SystemFunctions=1,SwInventory=1
               (SwInventory=1)>show
```

Example

```
>dn ManagedElement=1,SystemFunctions=1,SwInventory=1
(SwInventory=1)>show
SwInventory=1
active
    "ManagedElement=1,SystemFunctions=1,SwInventory=1,SwVersion=CXP9033913-R →
5B_1.01"
SwItem=CXC1060320-P13A01
SwItem=CXC1060320-R12A01
SwItem=CXC1372110-P9A06
SwItem=CXC1372110-R8A06
[...]
SwVersion=CXP9033913-P6B_1.01
SwVersion=CXP9033913-R5B_1.01
```

2. Is there only one `SwVersion` reference at the end of the printout?

   Yes: There is no non-active software to be deleted.

   No: Go to next step.

3. Delete the non-active software. It is the `SwVersion` which is not stated inside the quotes:

```
>dn ManagedElement=1,SystemFunctions=1,SwM=1

(SwM)>removeSoftwareVersion ManagedElement=1,SystemFunctions =1,SwInventory=
1,SwVersion=<sw-version>
```

   Example

```
 >dn ManagedElement=1,SystemFunctions=1,SwM=1
(SwM=1)>removeSoftwareVersion ManagedElement=1,SystemFunctions=1,SwInventory
=1,SwVersion=CXP9033913-P6B_1.01
false
```

4. If the response to the `removeSoftwareVersion` is `true`, the non-active software is deleted successfully. Execute the command in Step 1 to check whether the non-active software is deleted.

   If the response to the `removeSoftwareVersion` is `false` (as in this example), check if there is a configuration backup stored on BSP, that depends on this software and delete it.

   a. Identify the dependent backups:

```
>show ManagedElement=1,SystemFunctions=1,SwM=1,reportProgress
```

      Example

```
>show ManagedElement=1,SystemFunctions=1,SwM=1,reportProgress
reportProgress
    actionId=0
    actionName="removeSoftwareVersion"
    progressInfo=""
    progressPercentage=100
    result=FAILURE
    resultInfo="At least one backup that depends on this SW exists.
 Export and remove the backup(s) and try again. Dependent backups:
1, 2"
    state=FINISHED
    timeActionCompleted="2020-05-19T12:11:59Z"
    timeActionStarted="2020-05-19T12:11:13Z"
    timeOfLastStatusUpdate="2020-05-19T12:11:59Z"
```

   b. Check the backup name for each dependent backup (in the example, backup IDs 1 and 2):

```
>show-table ManagedElement=1,SystemFunctions=1,BrM=1,BrmBackupManag
er=1 -m BrmBackup -p backupName,brmBackupId
```

      Example

```
>show-table ManagedElement=1,SystemFunctions=1,BrM=1,BrmBackupManag
er=1 -m BrmBackup -p backupName,brmBackupId
=======================================================
| brmBackupId | backupName                              |
=======================================================
| 1           | InitialJsBackup_CXP9033913-P6B_1.01     |
| 2           | RstHwBackup_P6B_1.01                    |
=======================================================
```

c. Delete each dependent backup separately:

```
>ManagedElement=1,SystemFunctions=1,BrM=1,BrmBackupManager=1,delete
Backup --name <backup-name>
```

Example

```
>ManagedElement=1,SystemFunctions=1,BrM=1,BrmBackupManager=1,delete
Backup --name InitialJsBackup_CXP9033913-P6B_1.01
0
```

d. Delete the non-active software:

```
>dn ManagedElement=1,SystemFunctions=1,SwM=1
(SwM)>removeSoftwareVersion ManagedElement=1,SystemFunc tions=1,SwI
nventory=1,SwVersion=<sw-version>
```

Example

```
>dn ManagedElement=1,SystemFunctions=1,SwM=1
(SwM=1)>removeSoftwareVersion ManagedElement=1,SystemFunctions=1,Sw
Inventory=1,SwVersion=CXP9033913-P6B_1.01
true
```

`true` indicates that the non-active software is deleted successfully. Execute the command in Step 1 to confirm deletion.

For more information, see Upgrade Instruction with SMX and Upgrade Instruction

4.1.1.7 Upgrade Procedure for BSP Node

> This section describes the upgrade activities that are executed sequentially on BSP node when an Upgrade Job is initiated.

UpgradePackage MOs handle all the actions involved in a software upgrade. The following sequence applies to the actions of a BSP UpgradePackage:

1. Create upgrade package
2. Prepare upgrade package
3. Verify upgrade package
4. Activate step 1
5. Activate step 2
6. Activate step 3

Create Upgrade Package

The operator can either 'Import Software Package' through **Software Administration** UI in ENM or can create the package on node before triggering the upgrade. To create upgrade package on BSP, see the section *Create Upgrade Package* in BSP Software Management

Prepare Upgrade Package

The prepare action downloads the software to be used in an upgrade package from an external SFTP server and distributes it to the switch blades. New software cannot be downloaded if an inactive BSP software version is already present in the system. For information on how to remove the inactive software version, see section, *Deletion of non-active software for BSP node before SHM Upgrade.*

When the software has been successfully downloaded and distributed, the state of the software upgrade package changes to PREPARE_COMPLETED. If the prepare action fails, for example, if the software to be downloaded is not of higher revision state than the active software, the upgrade package state returns to INITIALIZED.

Verify Upgrade Package

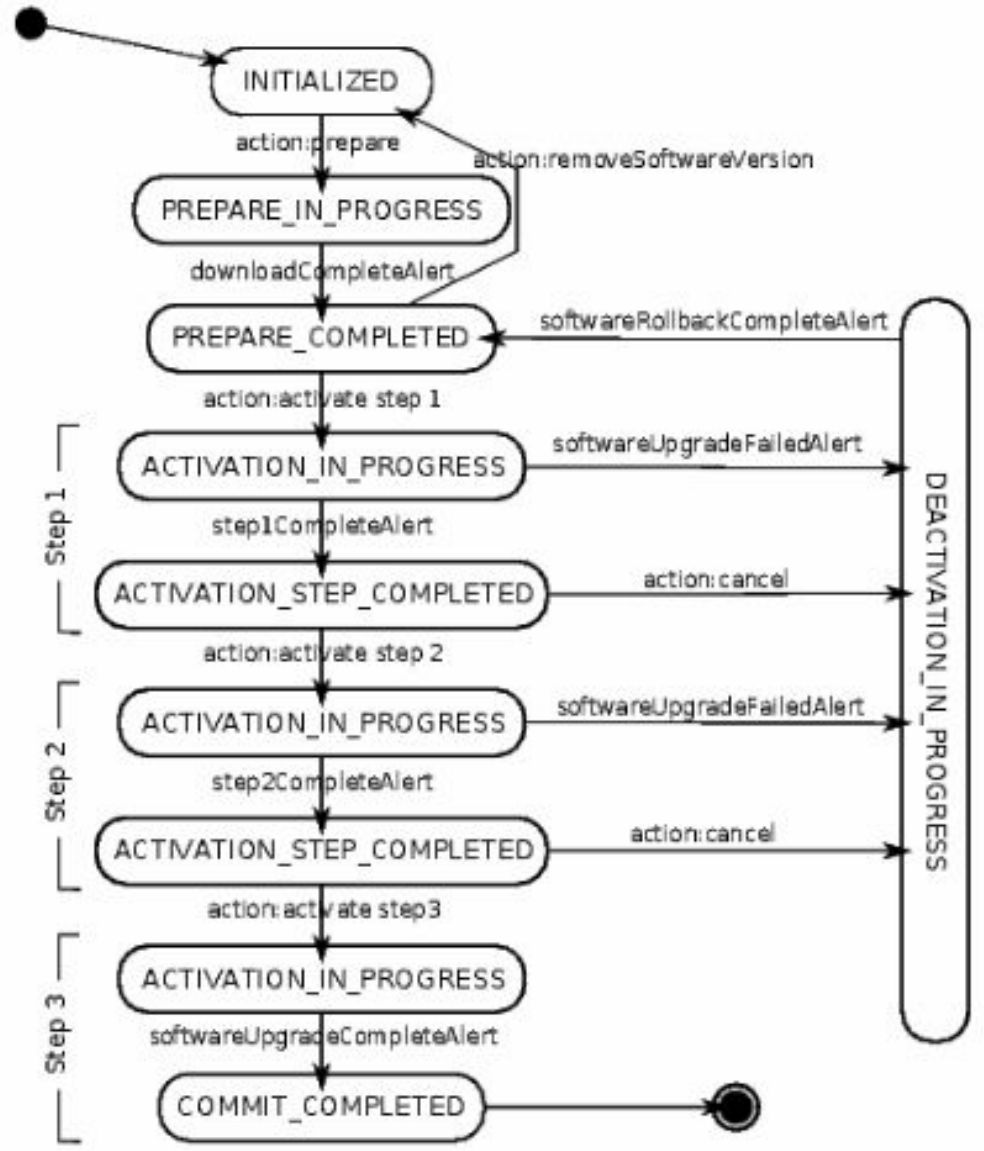
Before activating an upgrade package, it is recommended to verify that the system is in such a state that the upgrade package activation can be started. In this step, verification of upgrade package takes place.

Activate Upgrade Package

Since Activation of new Upgrade Package is performed in three steps, the operator must select **Step-by-Step** in **Select Upgrade Activities** UI while triggering upgrade. As per 'Step-by-Step' Activation procedure in SHM, user input is prompted for initiating each activation step. This enables the operator to do some checks manually before proceeding for next step activation. The job status is in `WAIT_FOR_USER_INPUT` until the operator manually initiates activation of next step. Activation of each step can be done by selecting the job and clicking on **Continue** in **Job Details** UI.

After Activation of each step is completed, the upgrade package state changes to `ACTIVATION_STEP_COMPLETED`. Since BSP does not have a separate confirm action, the third activation step acts as a confirm action to commit the software version on the node. When activation of step 3 is completed, the upgrade package state changes to `COMMIT_COMPLETED` and after this step, it is no longer possible to cancel the upgrade.

For more information, see the following BSP upgrade package state diagram.



Note: There is a limitation for BSP that, after SHM upgrade is successful, information about active software version is not updated in SwInventory. Hence the Software Administration web page still reflects old software version as active software version.

Information about active software version is reflected in ENM after manual resync of BSP node using the following commands in ENM CLI:

```
>> cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 ac →
tive=false

>> cmedit set NetworkElement=<nodeName>,CmNodeHeartbeatSupervision=1 ac →
tive=true
```

4.1.2 SHM 'Treat-As' Node Software Upgrade

ENM releases support specific Node software versions. ENM uses the following `ModelIdentity` attributes to identify different node software versions. Both of these attributes are stored on the `NetworkElement MOC`.

nodeModelIdentity
: This attribute specifies the actual software version running on a node.

ossModelIdentity
: This attribute specifies the software version that ENM has identified as running on the node.

Typically, the `nodeModelIdentity` and `ossModelIdentity` attributes are the same (have the same value). When you upgrade the software on the node, both values change to the same value. If these values differ, ENM treats the node as though it is running the software version that is specified in the `ossModelIdentity` attribute. In this case, the node is in `Treat-As` mode.

For example, if you update the node software to a newer version that is not currently supported by ENM, the `nodeModelIdentity` and `ossModelIdentity` have different values. ENM treats the node in `Treat-As` mode as if it is running the older-ENM supported software version specified in the `ossModelIdentity` attribute.

The concept of model identities, and how these model identities are used in the context of add node and sync node use cases, are explained in Configuration Management Overview on page 2.

Overview

— If you upgrade a node to a software version that is not supported by ENM, the node restarts and CM software synchronization updates the node model identity, but not the `ossModelIdentity`. ENM treats the node as though it is still running the pre-upgrade software version. This behavior is called `Treat-As`.

— If you upgrade a node to an ENM-supported software version, the node restarts and CM software synchronization updates the `ossModelIdentity` to the known supported `ossModelIdentity`. During this process, a full synchronization occurs on CM data, and the data is updated in ENM against new model version of the node that is supported in ENM.

`Treat-As` lets ENM manage a node that is running an unknown node version. It does this by treating it as a node running a supported node version, as long as the unknown node version is backward compatible with the known node version. `Treat-As` is used where nodes are upgraded to an unsupported version, or added to ENM with an unsupported version.

If a more accurate version of the node model becomes available in ENM, you might want to update the `ossModelIdentity` to refer to this newer model. In this

case, ENM still provides Treat-As support for the upgraded node. But the ENM model is more similar to the node model (this improves the backward compatibility of the ENM support).

The following steps describe how to identify and set the ossModelIdentity.

Note: Only set the ossModelIdentity across many nodes after all the required nodes at a particular software level are upgraded to the relevant node software level.

Steps

1. Identify the ossModelIdentity for the node model software:

```
cmedit describe --netype <netype>
```

2. Set the ossModelIdentity.
3. Update the ossModelIdentity to the node model version to treat the node:

```
cmedit set <network element FDN> ossModelIdentity=<value>
```

4. Update ossModelIdentity across multiple nodes:

```
cmedit set * NetworkElement.(nodeModelIdentity==<old_ossModelIdentity_value> →
) ossModelIdentity=<new_ossModelIdentity_value>
```

```
cmedit set * NetworkElement.(nodeModelIdentity==<old_ossModelIdentity_value> →
) ossModelIdentity=<new_ossModelIdentity_value>
```

5. Update the ossModelIdentity across large number of nodes that were at different node model versions before the upgrade:

```
cmedit set * NetworkElement.(nodeModelIdentity==<old_ossModelIdentity_value1 →
>, nodeModelIdentity==<old_ossModelIdentity_value2>) ossModelIdentity=<new_o →
ssModelIdentity_value>
```

4.2 Node Software Upgrade Using Automated Software Upgrade Flow

Node software upgrade can be performed through Automated Software Upgrade (ASU), which is designed with a sequence of activities. These are executed in an end-to-end NE Software Upgrade Roll Out process.

The Automated Software Upgrade flow supports:

— DU Radio nodes.

— Baseband Radio nodes.

— MINI-LINK nodes

Execute the ASU flow from **Flow Automation**, and select the activities to be executed as part of this flow. Once the flow execution starts, the selected activities are executed. The status can be monitored at **Flow Automation UI**.

ASU supports the execution of the following activities. The activities are selected as part of `Custom Input`, or by `File Input` directly as a JSON file.

— Delete Upgrade Package Job - (Optional).

— Delete Backup Housekeeping Job - (Optional).

— Backup Jobs - (Optional).

— Install License Job - (Optional).

— Execute Script Files - (Optional).

— Health Check Jobs - (Optional).

— Upgrade Package Validation - (Mandatory when only activation phase is executed).

— Enable or Disable FM Supervision - (Optional).

— Upgrade Job - (Mandatory).

Select the optional jobs to execute as part of an ASU. By default, only `Upgrade Job` is selected.

An ASU consists of two Stages:

— Setup

— Execute

Setup

During the Setup stage the user provides the required inputs:

— Setup Type

1. Interactive: User provides inputs by using ASU provided **User Tasks**.
2. File Input: User uploads a JSON file with the required inputs, ASU further validates these inputs.

— Upgrade Method:

1. Upgrade in Parallel: User performs the software upgrade in parallel of a batch of selected DU and Baseband Radio nodes based on system resources.

2. Topology Aware: User performs the software upgrade of MINI-LINK nodes based on their actual physical links network topology.

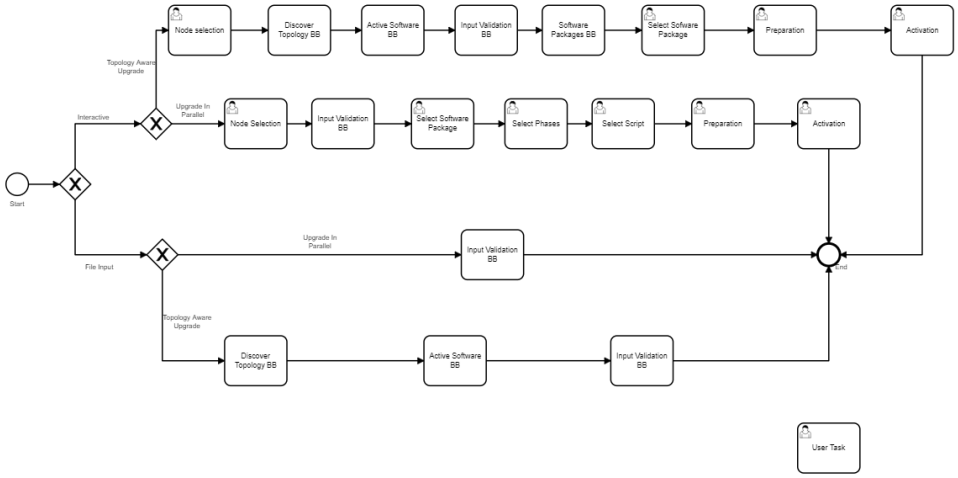


Figure 5 Flow Diagram for Setup Phase of ASU

Note: In this phase, snapshot of the nodes in `Collection` or `Saved Search` are taken, validated, and converted to a list of Nodes.

Execute

When the `Setup` stage is done, and the user confirms all the inputs, the `Execute` stage starts. The `Execute` stage has three phases:

1. **Initialization**: This phase creates the required jobs for `ASU` in `SHM` and `NHC`.

2. **Preparation**: This phase prepares the nodes for upgrade by executing the required activities.

3. **Activation**: This phase performs the actual upgrade on the node, and all the post-upgrade steps required or selected by the user.

Note: **Activation** phase is the only phase that contains steps that might affect network service.

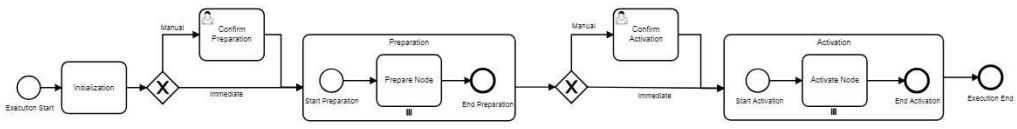


Figure 6 Workflow for Execute Stage of ASU

Initialization Phase

This phase creates all the required jobs or reports for ASU in SHM and NHC.

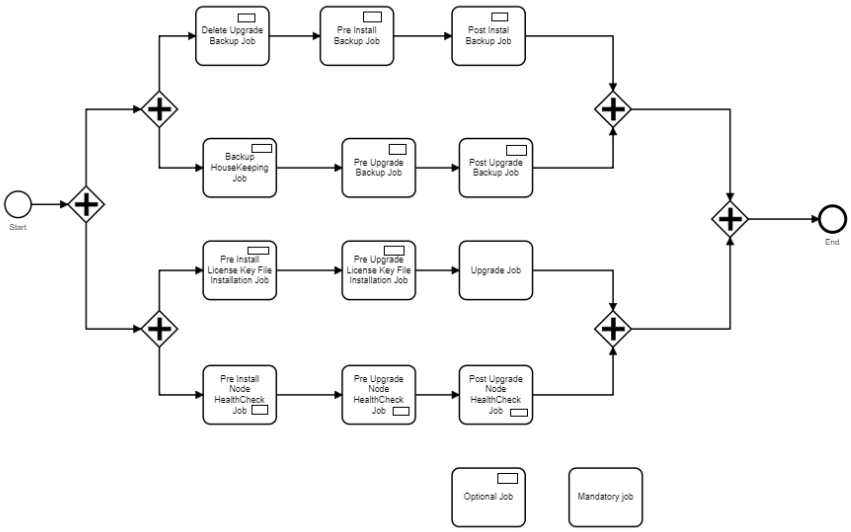


Figure 7 Workflow for Initialization Phase of ASU

Preparation Phase

Preparation phase contains all the activities required to prepare a node for software upgrade. On failure of any of the **Building Blocks**, other than **NHC Building Block**, in this phase, ASU flow is terminated for that particular instance of nodes.

The following activities execute in the preparation phase:

— Housekeeping of Upgrade packages.

— Housekeeping of Backups.

— Pre-Install Healthcheck Jobs.

— Pre-Install Script execution

— Pre-Install Node Backup.

— Pre-Install Installation of License Key File.

— Download Software on Node.

— Verify Software Package compatibility on Node.

— Post-Install Node Backup.

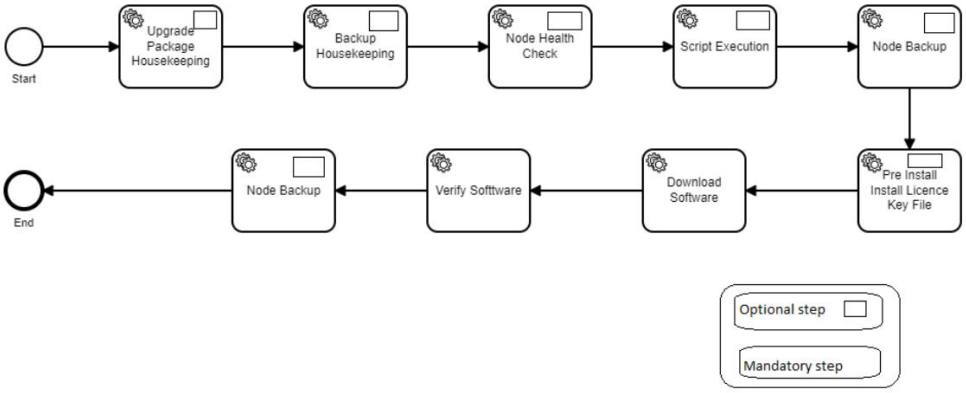


Figure 8 Preparation Phase of ASU with All the Activities

Activation Phase

Activation Phase is responsible for upgrading the node software and running all the post-install Jobs. The activation phase is scheduled to run during a maintenance window as it impacts the network or Node. On failure of any of the activities other than NHC in this phase, ASU flow terminates for that instance of nodes.

The following are the activities involved in ASU activation phase:

— Upgrade Package Validation (performed if Activation Phase only is selected for execution).

— Pre-Upgrade Node Healthcheck Job.

— Pre-Upgrade Script Execution.

— Pre-Upgrade Installation of License Key File.

— Pre-Upgrade Backup Job.

— Disable FM Supervision.

— Activate Node Software.

— Confirm Node Software.

— Post-Upgrade Script Execution.

— Enable FM Supervision.

— Post-Upgrade Node Healthcheck.

— Post-Upgrade backup.

— Cleanup Script.

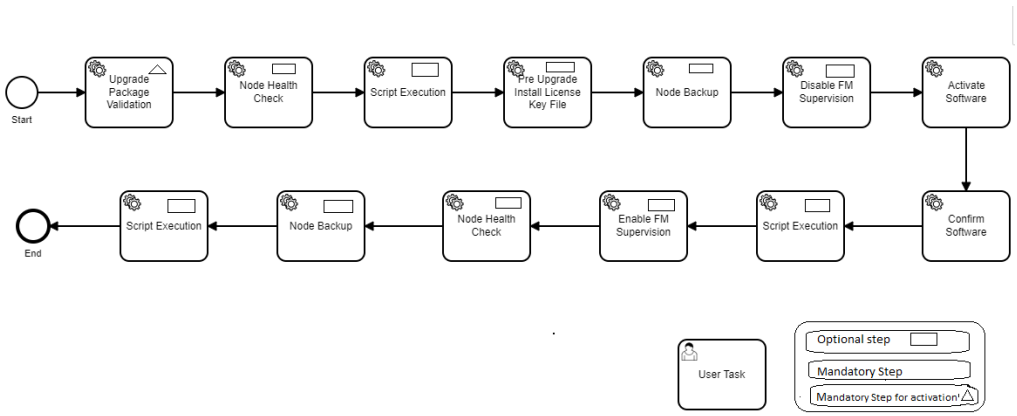


Figure 9 Workflow Diagram for Activation Phase

The previously mentioned activities are managed by the `Building Blocks of` ASU.

Note: Both `Disable` and `Enable FM Supervision` activities use `Change FM Supervision BB`.

For all the optional activities in the `Preparation` and `Activation` phases, the following activity bpmn is called:

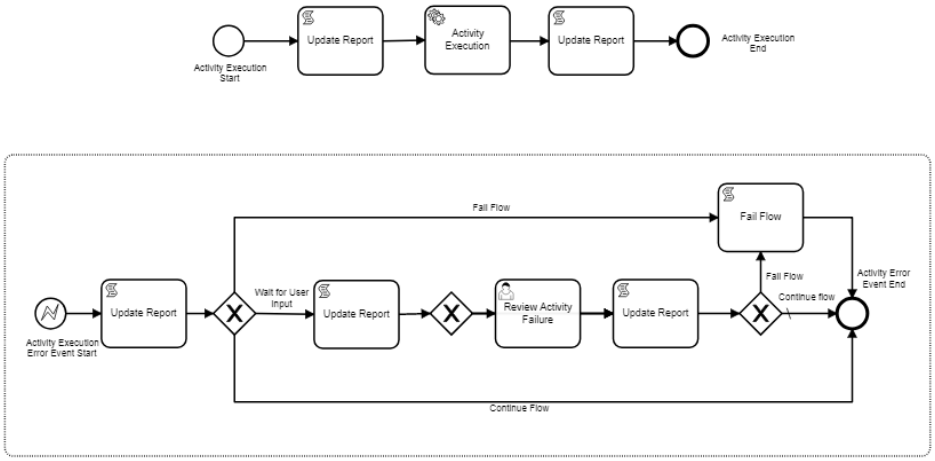


Figure 10 Activity bpmn

4.2.1 Building Blocks

To perform the tasks on nodes, Building Blocks are configured for each type of activity. Building Blocks are responsible for managing the activity on the Nodes or Network Elements (NEs). ASU provides the functionality in terms of building blocks. The following are the available Building Blocks:

4.2.1.1 Validation Building Block

Validation BB is responsible for performing all the validations for the **ASU** flow. This block takes all the input parameters given as part of `File Input` or `Custom User` tasks, and validates them. The responsibilities of this Building Block are:

— To verify the collections and Saved Searches, and resolve them to **Ne Names**.

— To perform TBAC Validation.

This Block is mandatory for the ASU Flow.

4.2.1.2 Backup Job Building Block

This BB creates a backup job on a selected set of nodes. The created backup job has a combination of software, configuration, and other vital information stored in a file on the node.

Backup Job is an optional block for the user. It has the following activities:

— Create backup.

— Upload backup.

As **ASU** performs **Node Software Upgrade**, which can impact the node configuration, **Backup Job** provides an option to perform **Pre-Install Backup**, **Pre-Upgrade**, and **Post-Upgrade Backup**.

4.2.1.3 Backup Housekeeping Building Block

This BB creates a **Backup Housekeeping Job** that manages and defines the number of ENM-created Backup files that can be stored on a node. This helps manage the disk space available on the node. This job has only one activity, **Delete Backup Activity**.

4.2.1.4 Delete Upgrade Package Building Block

This BB creates a **Delete Upgrade Package Job** that manages the **Upgrade Packages** present on a node. All inactive upgrade packages that can, are deleted using this block. This helps manage the disk space available on the node.

Upgrade Package Housekeeping is an optional block for the user. This job has only one activity, **Delete Upgrade Package Activity**.

4.2.1.5 Change FM Supervision Building Block

Change FM Supervision BB enables users to disable **FM Supervision** before activation of an upgrade package on the node, and enable it after upgrade has completed. **Change FM Supervision** BB is an optional block in the flow.

4.2.1.6 Install License Key Building Block

Install License Key BB creates an **Install License Key File** job. This BB validates the latest available license key file on ENM onto the node. It then installs the latest available license key file on ENM onto the node.

Install license Key Files BB is an optional block for the user. This job has only two activities, **Validate License Activity** and **Install License Activity**.

4.2.1.7 Node Health Check Building Block

Node Health Check (NHC) BB creates NHC reports for the nodes in the network to check NE health.

ASU performs upgrades that might impact network traffic. NHC can be executed at different stages of the flow execution. They include **Pre-Install Health Check**, **Pre-Upgrade Health Check**, and **Post-Upgrade Health Check**. NHC BB is optional.

4.2.1.8 Upgrade Building Block

This BB creates upgrade jobs. **Software Upgrade** enables user to install a software package on a node or NE. **Software Upgrade** is used to add new functions, improve performance, and correct faults in the existing software installed on a node.

Upgrade BB is mandatory for ASU Flow.

Upgrade BB has four activities:

— Download Software.

— Verify Software.

— Activate Software.

— Confirm Software.

4.2.1.9 SHM Continue Building Block

SHM Continue BB continues **SHM** jobs created using the previous BBs. This BB is common to all **SHM** jobs.

SHM Continue BB is executed as part of **Preparation** and **Activation** phases.

4.2.1.10 NHC Continue Building Block

NHC Continue BB continues the NHC reports created using **Node Health Check** BB.

NHC Continue BB is executed as part of `Preparation` and `Activation` phases.

4.2.1.11 SHM Cancel Building Block

On failure in the execution of flows or jobs, **SHM Cancel** BB is triggered. It cancels any remaining **SHM** Jobs created as part of the flow.

SHM Cancel BB is present in **Preparation** and **Activation** phases.

4.2.1.12 NHC Cancel Building Block

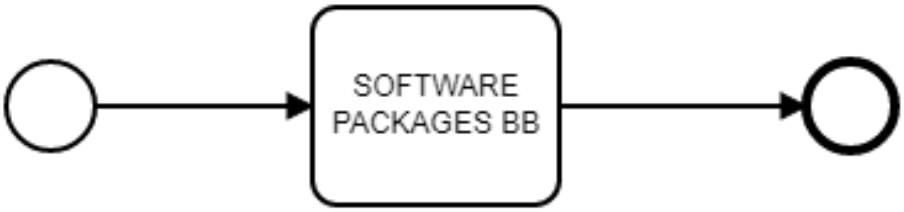
On failure in the execution of flows or jobs, **NHC Cancel** BB is triggered. It cancels any remaining NHC reports created as part of the flow.

NHC Cancel BB is present in **Preparation** and **Activation** phases.

4.2.1.13 Fetch Software Package Building Block

This BB fetches the available software packages for respective node types. This Building Block takes as input, the list of `neNames`, `CollectionIds`, or `SavedSearch Names`.

The output of this BB is a list of software packages supported by the nodes, based on node type. When a user provides inputs to the ASU flow using **User Tasks**, **Fetch Software Package** BB presents as one of the user tasks. It shows all the available packages for each node type. The user can select the requirements, and proceed with the flow.



4.2.1.14 Scripting Building Block

Scripting BB executes scripts configured by the user.

The user selects script from a list of scripts present in ENM. Scripts can execute at four phases during ASU execution.

Scripting BB is part of **Preparation** and **Activation** phases. It executes `pre-install`, `pre-upgrade`, `post-upgrade`, and `cleanup` scripts.

4.2.1.15 Discover Topology Building Block

Discover Topology Building Block is responsible for discovering the subtended network topology through the node links.

Discover Topology Building Block is a part of the Setup phase for `Topology Aware Upgrade`.

4.2.1.16 Active Software Building Block

Active Software Building Block is responsible for retrieving the details of the software running on the nodes.

Active Software Building Block is part of the `Setup` phase for `Topology Aware Upgrade`. It is executed to get the software active details for the discovered nodes.

4.2.1.17 Upgrade Package Validation Building Block

Upgrade Package Validation Building block is used to validate the upgrade package state on node.

If upgrade package is not in valid state, then user task will be presented to user to take action.

If upgrade package is already active on node, then the flow will be canceled for that node.

Upgrade Package Validation BB is a Mandatory block for ASU flow when Activation Phase only is selected for execution.

4.3 Cancel Job Using Software and Hardware Manager

Describes how the cancel job option operates with various node types.

Cancel Job allows the operator to cancel a job that is in `running` or `waiting for the operator` input state. Cancel can be done at main job or node level job. When a job is canceled, the current running activity on the node is canceled if the node supports the cancel option for that activity. If cancel is supported on the node for that activity, SHM waits until the cancel operation completes on the node. If node does not support the cancel option, then that activity continues, and SHM does not proceed with any further activities in the job.

If there are AXE nodes, **Cancel Job** functionality allows the operator to cancel a job that is in a `running`, `waiting`, or `Interrupted for the operator` input state. If the cancel operation fails, manually revert the node to bring it back to its initial stage. This procedure is performed outside of ENM.

4.3.1 Job and Activity Cancel Support on the Node

The following table lists for node types those activities for which the cancel operation is supported.

Node Type	Backup				Delete Backup	License	Restore					Upgrade			
	Create Backup	Set the CV as Startable	Set the CV first in RollBack List	Upload Backup	Delete Backup	Install License	Download	Verify	Install Upgrade Package	Restore	Confirm	Install / Prepare / Download	Verify	Upgrade / Activate	Confirm
eNodeB DU Radio Node	×	×	×	×	×	×	×	×	✓	✓	×	✓	×	✓	×
MGw	×	×	×	×	×	×	×	×	✓	✓	×	✓	×	✓	×
eNodeB Baseband Radio Node	✓	NA	NA	✓	✓	×	✓	NA	✓	✓	×	✓	✓	✓	×
SGSN-MME	✓	NA	NA	✓	✓	NA	✓	NA	✓	✓	×	✓	✓	✓	×
vMME	✓	NA	NA	✓	✓	NA	✓	NA	✓	✓	×	✓	✓	✓	×
MINI-LINK Indoor	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	×	NA	×	×

If a cancel operation is active for a backup job, SHM cancels the upload option for ECIM-based nodes. It carries on with executing the upload option for node types `eNodeB DU Radio` nodes and `MGW`, as there is no support for cancel operation on these node types.

4.4 Create and Manage Node Backups Using SHM Backup Job Functionality

> How the backup of node configuration or software is performed and managed in Software and Hardware Manager (SHM).

Backup Management in General

SHM enables you to create a **Backup** file for the nodes in your network. The backup file is used to roll back a node to an earlier working version. For example, if you encounter an error during, after upgrading, or activating software on a node. Depending upon the node type, a backup file contains software and configuration information details of a node.

To create a **Backup** file, create a **Backup Job**. This job specifies the nodes that you want to create a backup file for. A backup job performs all required backup activities. Examples are creating the backup file, uploading the backup file to ENM, and so on. You can schedule when the job and the related backup activities are performed. You can also create a backup job that runs periodically - this creates backup files at regular intervals.

The actual backup activities performed by a **Backup Job** depend on the node type that you are creating the backup file for. For example, the backup activities for CPP-based nodes, such as `MGW` nodes, are different to the backup activities for ECIM-based nodes, such as `eNodeB Baseband Radio Nodes`.

From the **Software Hardware Manager** page, use the `Jobs` table to monitor the overall progress of a **Backup** job. The **Progress** column displays the job progress in percentages. The backup job progress is updated as **Backup** activities on the nodes specified in the job complete.

Monitor the job progress at individual node level specified in the job by using the **Job Details** page. Monitor at individual node activities level by using the **Node Activities Panel**. In the **Jobs** table, select the **relevant job** > **View Job Details** > **select the relevant node from the Node Progress table** > **Node Activities**. In the **Node Activities** panel, you can see the **Progress** of the job activities performed on the node.

SHM performs internal validation before starting the desired activity on the node, to ensure that node is in right state to perform the activity.

For CPP-based nodes, for each job activity in the Node Activities panel, the progress bar and progress percentage increases whenever a validation is completed, an action on a node is started, or an activity is completed.

For ECIM-based nodes, for each job activity in the Node Activities panel, the progress bar and progress percentage increases as the activity progresses on the node.

Node Selection in Backup Job Configuration

— You can create a single backup job that creates **Backup** files for different node types, such as ECIM-based, or CPP-based nodes, and so on.

— When you create a `Backup Job`, specify the nodes you want to create backup files for. You can specify individual nodes, collections, or saved searches to specify the nodes. The added nodes are evaluated during the job execution time.

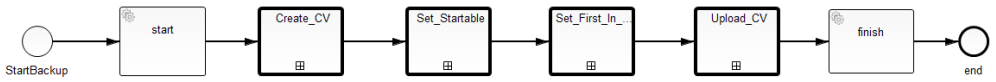
— For AXE-based nodes you can select sub-components (Cluster, CP, APG43L) of nodes. The added components are evaluated during job execution time.

SHM does not apply restrictions on the number of nodes in a job. However, the recommended maximum number is `5000` nodes in a single job.

To avoid resource usage issues, do not execute large SHM jobs in parallel.

4.4.1 Backup Job Workflow for CPP-Based Nodes

> Describes the backup activities that are executed sequentially on each node when a Backup job executes for CPP-based nodes.



Backup file is a generic term. For a CPP-based Node, the backup file created is referred to as a `Configuration Version` (CV).

Create CV Activity

This activity creates a backup CV of the current node configuration. The new CV is stored persistently in the node file store (flash file system).

Set as Startable Activity

This activity sets the created CV as a `Startable` CV, which means the node uses this CV the next time the node restarts.

Note: If this configuration does not work, the node automatically reverts to a CV that is known to work. That is included in the rollback list.

Set First in the Rollback List Activity

This activity adds the created CV as the first CV in the rollback list.

Upload Activity

This activity uploads the created configuration version from the node file store to ENM.

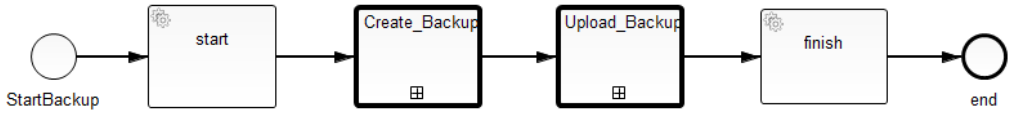
Note:

— The `Set_Startable` and `Set_First_in_Rollback_List` activities are only applicable to CPP-based Nodes, such as, `eNodeB DU Radio` and `MGW` nodes.

— You can also perform `Set_Startable`, `Set_First_in_the_Rollback_List` and `Upload` activities on an existing Configuration Version using the `Manage Backups` functionality.

4.4.2 Backup Job Workflow for ECIM-Based Nodes

Describes the backup activities that are executed sequentially on each node, when a Backup job executes for ECIM-based nodes.



Create Backup Activity

This activity (implicit activity) creates a backup of the node software.

Upload Activity

This activity uploads the created backup from the node file store to ENM.

Note: You can perform an **Upload** activity on an existing backup using the Manage Backups option.

For CPP-based nodes, for each job activity in the Node Activities panel, the progress bar and progress percentage increases whenever a validation is completed, an action on a node is started, or an activity is completed.

For ECIM-based nodes, for each job activity in the Node Activities panel, progress bar and progress percentage increases according to progress percentage notifications received from node.

SHM performs internal validation before starting the desired activity on the node, to ensure that node is in right state to perform the activity, or to ensure that the activity is performed properly on the node.

Downloading Backup File from SMRS

Backup files created in SHM can be downloaded to a local file system, using the following steps:

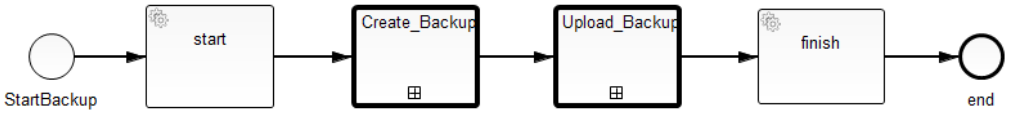
Steps

1. Secure a user profile with role `FIELD_TECHNICIAN` from the system administrator.
2. Get the `svc_CM_vip_ipaddress` value from your System Administrator.
3. Establish an SFTP connection to the path `smrsroot/backup/<node_type>/<node_name>` using any SFTP client to the IP Address received in the Step 2 and using the user credentials specified in Step 1.

4. Download the backup file as specified in **SHM > Backup Administration**.

4.4.3 Backup Job Workflow for AXE-Based Nodes

> Describes the backup activities that are executed sequentially on each node, when a Backup job executes for AXE-based nodes.



Create Activity

This activity (implicit activity) creates a backup of the node software.

The following are the options to create backup:

— **Rotate Backup**: Rotate Backup moves the backup file with the highest number in sequence to `RELFSW0` and increment all other backups by one. This is applicable only for CP (CP, SPX, Cluster) components.

— **Overwrite Existing Backup**: Overwrites the backup with the highest number in sequence. This is applicable only for CP (CP, SPX, Cluster) components.

  If the previous options are not selected, by default (Create backup) a new backup is created with the name `RELFSW(x)`, where `x` is one higher than the highest backup number in sequence.

— **Secure Backup**: To create a secure backup, the user has to provide `Password` and `UserLabel`. This is applicable only for `APG43L` components with version `3.7.0` or later.

Upload Activity

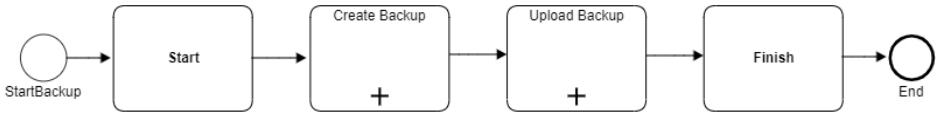
This activity uploads the created backup from the node file store to ENM.

Note: You can perform an `Upload` activity on an existing backup using the **Manage Backups** option.

For AXE-based nodes, for each job activity in the **Node Activities** panel, progress bar and progress percentage increases according to progress percentage responses received from `Winfiol`.

4.4.4 Backup Job Workflow for MINI-LINK-Indoor Based Nodes

Describes the backup activities that are executed sequentially on each node, when a Backup job executes, for MINI-LINK-Indoor based nodes.



Create Backup Activity

This activity creates a backup of the node software.

Upload Activity

This activity uploads the created backup from the node file store to ENM location `/home/smrs/MINI-LINK/MINI-LINK-Indoor/tn_backup_configuration`.

Downloading Backup File from SMRS

Backup files created in SHM can be downloaded to a local file system, using the following steps:

1. Create an SHM Backup profile job on MINI-LINK-Indoor node.
2. Log in to the server and connect to `mssnmpcm` service group.
3. Navigate to the path `/home/smrs/MINI-LINK/MINI-LINK-Indoor/tn_backup_configuration/<node_name>`.
4. Download the required backup file as specified in `SHM > Backup Administration` in ENM GUI.

4.5 Restoring a Node Backup in Software and Hardware Manager

This section describes how the backup of node configuration or software is restored in Software and Hardware Manager (SHM).

Generic Restoring of Node Backup Flow

If you encounter problems with nodes after upgrading or activating software on them, you might need to restore the nodes to an earlier working configuration.

SHM lets you restore the nodes using previously created backup files. To restore a backup on node, create a `Restore Job`.

An SHM `Restore Job` specifies the nodes that you want to restore, the backups to be used, and the related restore activities that you want to perform. For example, to select a backup stored on the node, or the one stored in ENM. You can also schedule when the `Restore Job` and the related activities are performed.

The restore activities that you can perform depend on the node type being restored. For example, the restore activities for CPP-based nodes, such as ERBS or MGW nodes, are different to the backup activities for ECIM-based nodes, such as, `eNodeB Baseband Radio Nodes` or `SGSN-MME` nodes.

You can create a single `Restore Job` that restores backup files for different node types, such as ECIM-based or CPP-based nodes.

In the **Software Hardware Manager** page, use the **Jobs** table to monitor the overall progress of a `Restore` job. The **Progress** column displays job progress in percentages. The **Restore** job progress is updated according to the completion of `Restore` activities on the nodes specified in the job.

You can monitor the job progress at individual node level specified in the job using the **Job Details** page, and at individual node activities level using the **Node Activities Panel**. In the Jobs table, select the **relevant job** > **View Job Details** > **select the relevant node from the Node Progress table** > **Node Activities**. In the **Node Activities** panel, you can see the **Progress** of the job activities performed on the node.

SHM performs internal validation before starting the activity on the node to ensure that the node is in the right state to perform the activity.

For CPP-based nodes, for each job activity in the **Node Activities** panel, the progress bar and progress percentage increases whenever a validation is completed, an action on a node is started, or an activity is completed.

For ECIM-based nodes, for each job activity in the **Node Activities** panel, progress bar and progress percentage increases as the activity progresses on the node.

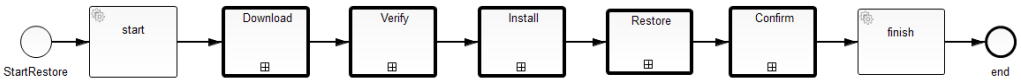
Node Selection in Restore Job Configuration

— When you create a **Restore Job**, specify the nodes that you want to restore backup files to. You can specify individual nodes, use collections, or saved searches to specify the nodes. The selected nodes are evaluated during the job execution time.

Note: SHM does not apply restrictions on the number of nodes in a job. The recommended maximum number is 200 nodes in a single job. To avoid resource usage issues, do not execute large SHM jobs in parallel.

4.5.1 Restore Job Workflow for CPP-Based Nodes

> Describes the restore activities that are executed sequentially on each node, when a Restore Job executes for CPP-based nodes.



Backup file is a generic term. For a CPP-based Node, the backup file created is referred to as a `Configuration Version` (CV).

Download Activity

This activity downloads a CV from ENM to the node.

Verify Activity

This activity verifies the consistency of the CV downloaded from ENM. It verifies whether there are any missing or corrupted upgrade packages that must be installed to restore this CV on the node during `Install` activity phase. It also verifies whether forced restore is possible.

Note: A CV is a locally stored snapshot of a node configuration at a given time. The CV consists of a database backup and some configuration files. A CV includes only a reference to the UP and so does not include UP themselves.

Install Activity

This activity installs the Upgrade Packages and also provides the following options while configuring the job:

Install Missing Upgrade Packages:
Installs any missing upgrade packages that are identified in the Verify activity.

Replace Corrupted Upgrade Packages:
Replaces any corrupted upgrade packages that are identified in the Verify activity.

Restore Activity

This activity restores the selected CV on the node and also has the following options while configuring the job:

Auto Configuration
This enables you to choose auto configuration for a node on a selected backup.

Forced Restore This activity restores the node to a configuration defined in the previously downloaded CV.

Confirm Activity

This activity confirms a restore permanently. A final CV is created and is set as `startable`.

The Download and Install activities require an SFTP connection between ENM and the node. These activities may fail if the SFTP connection between the node and the relevant VM is lost (for example, because of a reboot of the VM or during an ENM upgrade).

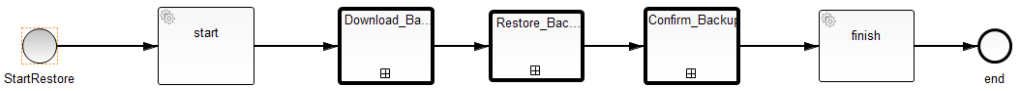
Note: If a restore operation is not confirmed within the confirmation time-out determined by the node, the node rolls back to the previous configuration version.

It is possible to restore Backups present on ENM or Node.

However, CVs present on the node with type other than `DOWNLOADED` are restored when the CV is set as `startable` and then by restarting the node. This information is available in job logs under Restore Activity. During this activity, if the node restart fails, the job is failed and the CV remains in set as `startable` state.

4.5.2 Restore Job Workflow for ECIM-Based Nodes

> Describes the restore backup activities that are executed sequentially on each node, when a Restore Job executes for ECIM-based nodes:



Download Activity

This activity downloads a backup from ENM on to the node.

Restore Activity

This activity restores the selected backup on the node.

Confirm Activity

This activity confirms a restore permanently.

Note:

— The Download activity requires an SFTP connection between ENM and the node.

— This activity might fail if the SFTP connection between the node and the relevant VM is lost, for example, because of a reboot of the VM or during an ENM upgrade.

— If a restore operation is not confirmed within the confirmation time-out interval determined by the node, the system is restored to the state it was in before the last restore operation was started.

4.6 Housekeeping Backups on Nodes Using Software Hardware Manager

> Describes the SHM housekeeping functionality which enables the management of backups stored on the nodes.

Housekeeping of Backups on Node Flow in General

Use the SHM housekeeping functionality to remove backups that are no longer required, from the nodes on which they are stored. Backup file is a generic term. For a CPP-based Node, the backup files are referred to as Configuration Versions (CVs).

To remove the backups, create a `Backup Housekeeping Job`. This job specifies the nodes to perform housekeeping on, and also the related housekeeping activities to perform. You can schedule when the job and the related backup housekeeping activities are performed. You can also create a `Backup Housekeeping Job` which runs periodically - this removes backup files at regular intervals.

The backup housekeeping activities depend on the node type for which you are removing the backups. The backup housekeeping activities for CPP-based nodes, such as ERBS or MGW nodes, are different to the backup housekeeping activities for ECIM-based nodes, such as baseband radio node or SGSN-MME.

You can create a single backup housekeeping job that removes backup files from different node types, such as ECIM-based or CPP-based nodes.

In the `Software Hardware Manager` page, use the `Jobs` table to monitor the overall progress of the backup housekeeping job. The `Progress` column displays the job progress in terms of percentage. The backup housekeeping job progress is updated according to the completion of backup houseKeeping activities on the nodes specified in the job.

You can also monitor the job progress at individual node level specified in the job using `Job Details` page, and also at individual node activities level using `Node Activities Panel`. In the Jobs table, select the **relevant job** > **View Job Details** > **select the relevant node from the Node Progress table** > **Node Activities**. In the

Node Activities panel, you can see the Progress of the job activities performed on the node.

SHM performs internal validation before starting the desired activity on the node to ensure that node is in right state to perform the activity.

For CPP-based nodes, for each job activity in the Node Activities panel, the progress bar and progress percentage increases whenever a validation is completed, an action on a node is started, and an activity is completed.

For ECIM-based nodes, for each job activity in the Node Activities panel, progress bar and progress percentage increases as the activity progresses on the node.

1. When you execute a backup housekeeping job, backups, or CVs for the CPP-based nodes, are only deleted from the node, if the number of backups on the node exceeds the `Number of Backups` to keep, or the `Number of CVs` to keep values specified in the relevant job.

2. The Backup Housekeeping Job deletes backups (or CVs) from the nodes starting with the oldest backup.

   For ECIM-based nodes, Backups that are `SYSTEM_CREATED` are not considered for Backup Housekeeping. For CPP-based nodes, the `startableConfigurationVersion` and `currentLoadedConfigurationVersion` are not considered from deletion in Backup Housekeeping.

3. The Backup Housekeeping Job result is failed if any backup fails to delete. For example, if you want to keep five backups on a node which contains 20 backups, the job deletes 15 backups starting with the oldest. If any of the 15 backups fail to delete, the job result is failed.

4. If a backup fails to delete, the Backup Housekeeping Job does not attempt to delete the next oldest job. For example, if you want to keep two backups on node which contains 10 backups, the eight backups starting with the oldest get deleted. If one of the eight backups fails to delete, the job does not attempt to delete the ninth oldest backup.

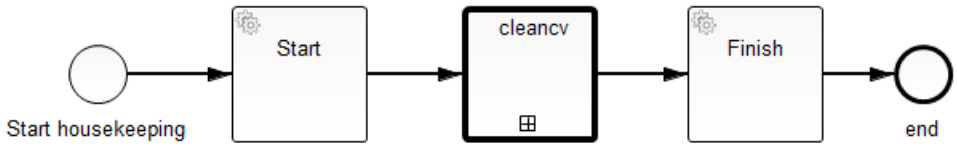
Node Selection in Backup Housekeeping Job Configuration

— When you create a Backup Housekeeping Job, specify the nodes from which you want to remove backup files. You can specify individual nodes or use collections or saved searches to specify the nodes. The selected nodes are evaluated during the job execution time.

Note: SHM does not restrict on the number of nodes in a job. However, the recommended maximum number is 5000 nodes in a single job. To avoid resource usage issues, do not execute large SHM jobs in parallel.

4.6.1 Housekeeping of Backups on Node Workflow for CPP-Based Nodes

Describes the backup housekeeping activities that are executed sequentially on each node when a Backup Housekeeping Job executes for CPP-based nodes.



Clean CV Activity

This activity deletes the CVs on the node starting from the oldest CV. In the Create Backup Housekeeping Job wizard, use the following options to specify the number of CVs to keep on the CPP-based node:

— Clear to roll back list and delete all possible CVs.

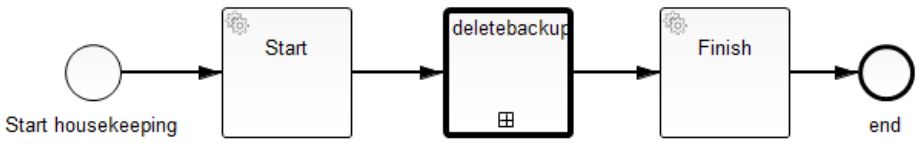
  This option enables you to delete all possible CVs on the node and from rollback list.

— Specify rollback list size or number of CVs to keep.

  This option enables you to specify the number of CVs to keep on node and in rollback list.

4.6.2 Housekeeping of Backups on Node Workflow for ECIM-Based Nodes

Describes the backup housekeeping activities that are executed sequentially on each node when a Backup job executes for ECIM-based nodes:



Delete Backup Activity

This activity deletes the backups on the node, starting with the oldest backup, leaving the specified number of backups to keep on node.

4.7 Install License Key in Software and Hardware Manager

> Using Software and Hardware Manager (SHM) to install license keys on the nodes in the network.

License Key Installation on Nodes

SHM enables you to install `License Key Files` on the nodes in your network to enable features and increase capacity on the nodes.

To install license key files on nodes, create an `Install License Key Job`. This job specifies the license key files to install and the nodes on which to install them. The job also specifies when the license key installation executes. The install license key job functionality supports license key files validation when creating the job. For example, warnings are displayed if you try to install a license key file with a lower Sequence Number than the current Sequence Number.

In the `Software Hardware Manager` page, use the `Jobs` table to monitor the overall progress of the install license key job. The `Progress` column displays the job progress in terms of percentage. The install license job progress is updated according to the completion of license activities on the nodes specified in the job.

You can also monitor the job progress at individual node level specified in the job using `Job Details` page and also at individual node activities level using `Node Activities Panel`. In the Jobs table, select the **relevant job** > **View Job Details** > **select the relevant node from the Node Progress table** > **Node Activities**. In the `Node Activities` panel, you can see the progress of the job activities performed on the node.

SHM performs internal validation before starting the desired activity on the node to ensure that node is in right state to perform the activity.

For CPP-based nodes, for each job activity in the node activities panel, the progress bar and progress percentage increases whenever a validation is completed, an action on a node is started, and an activity is completed.

For ECIM-based nodes, for each job activity in the node activities panel, progress bar and progress percentage as the activity progresses on the node.

Node Selection in License Job Configuration

— You can create a single `Install License Key Job` that installs the key files for different node types, such as ECIM-based or CPP-based nodes.

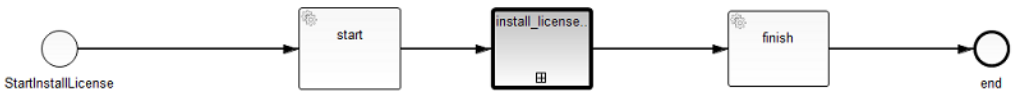
— When you create an `Install License Key Job`, specify the nodes on which you want to install license key files. You can specify individual nodes or use collections or saved searches to specify the nodes. The added nodes are evaluated during the job execution time.

Note: SHM does not apply restrictions on the number of nodes in a job. However, the recommended maximum number is `1000` nodes in a single job. To avoid resource usage issues, do not execute large SHM jobs in parallel.

Install license key files during an appropriate maintenance window.

4.7.1 Install License Key Job Workflow for CPP-Based Nodes

> Describes the install activities that are executed sequentially on each node, when an Install License Key job executes for CPP-based nodes.



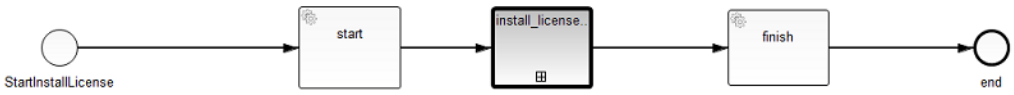
Install License Activity

This activity installs the `License Key File` on the node.

Note: The `Install License` activity requires an SFTP connection between ENM and the Node. This activity might fail if the SFTP connection between the Node and the relevant ENM VM is lost. For example, because of a reboot of the VM, or during an ENM upgrade.

4.7.2 Install License Key Job Workflow for ECIM-Based Nodes

> Describes the install activities that are executed sequentially on each node when an Install License Key job executes for ECIM-based nodes.



Fail-Safe Backup Feature for Baseband Radio Node License Installation

Before executing the `Install License Key Job`, the SHM fail-safe backup mechanism creates an internal backup on the relevant nodes. Once the License

Key Files are successfully installed on the nodes, SHM issues a deactivate call to the nodes, that deletes the fail-safe backup.

The fail-safe backup mechanism lets you restore the node to a working state if an issue occurs while executing the `Install License Key` job. For example, if the job fails because of connectivity issues.

Note: The fail-safe backup mechanism is enabled by default. To disable the fail-safe backup mechanism, set the `SHM_PERFORM_FAILSAFE_BACKUP` to `false`.

See *SHM: Controlling Failsafe Backup during Radio Node License Install* under *Software and Hardware Management Administration Tasks* section of *ENM Configuration System Administrator Guide*.

The fail-safe backup mechanism requires the MO `BrmFailsafeBackup`.

This MO is delivered in the baseband radio node BRM fragment version 3.4. If baseband radio node BRM fragment is an earlier version (that does not contain the MO), SHM does not perform fail-safe backup.

4.8 Upgrade Package State in Software Inventory

> Describes the mapping between the state on the node and in the SHM Software Administration page.

Software Administration

The `Software Administration` page provides access to SHM functionality to manage the software running on the nodes in the network.

For example, you can import software into ENM using the `Import Software Packages` functionality, and then install and activate this software on specified nodes using the `Create Upgrade Job` wizard. The software administration page provides inventory information about the software running on the nodes in the network.

The software administration page displays the software inventory on the nodes as `Software Versions` and `Software Items`. `Software Versions` are the upgrade packages that are available for the selected nodes. `Software Items` are the Load Modules associated with a `Software Version`.

Upgrade Package States

To upgrade the software running on selected nodes, import the relevant software version (UP), and then install and activate on the nodes using the `Create Upgrade Job` wizard. Depending on the progress of the upgrade job, the state of the software on the node can vary. This is displayed in the inventory as the `State` column in the `Software Versions` table. The software version state on the node

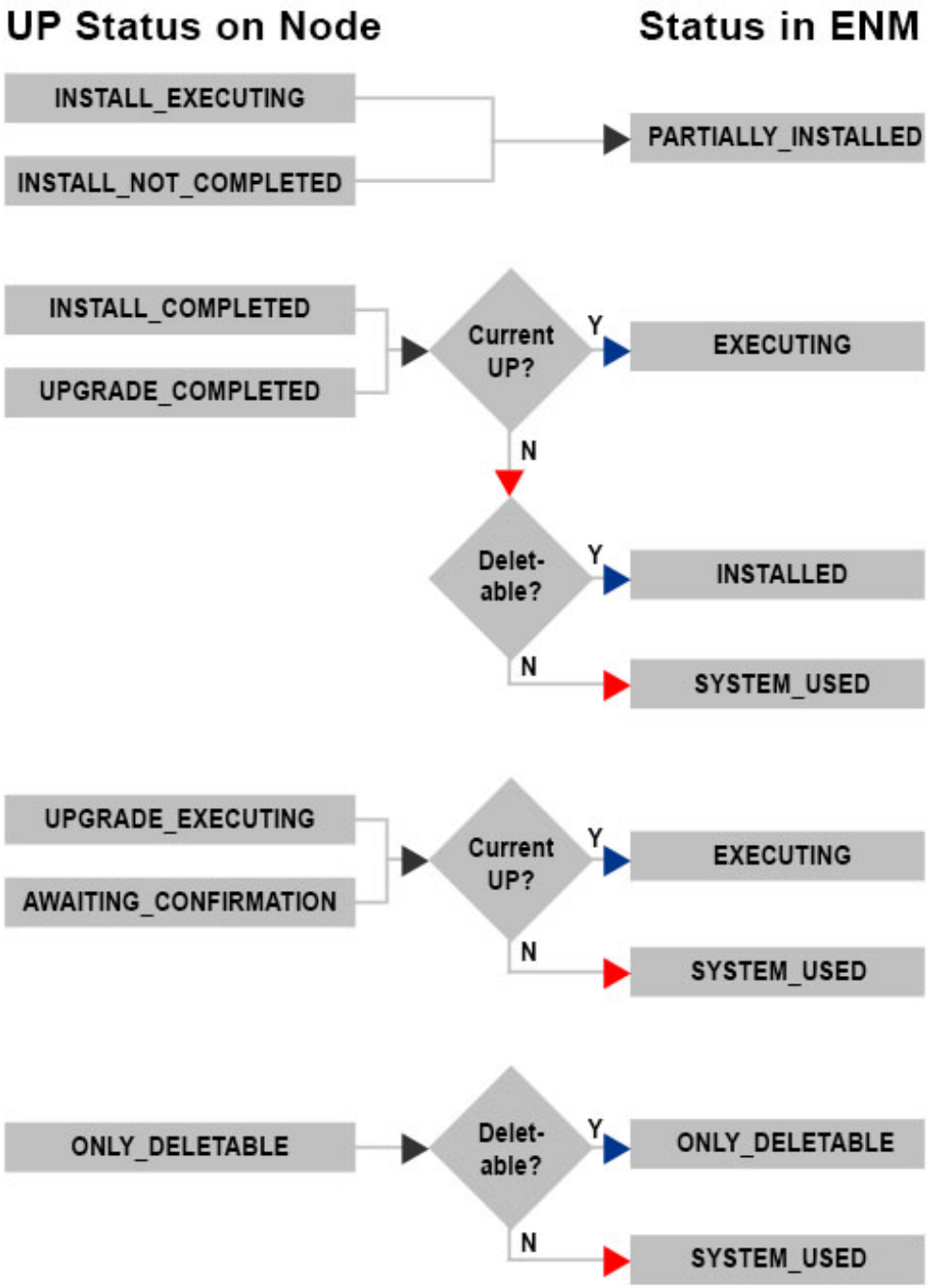
is available from `UpgradePackage MO` (UP MO) state attribute by using the following command:

```
cmedit get <NodeName> UpgradePackage.*
```

For ECIM-based nodes, the state value in the inventory is same as the value of state attribute on the node.

For CPP based nodes, the state value in the inventory sometimes differs from the value of state attribute on the node. For example, when the UP MO state attribute value is `INSTALL_EXECUTING`, or `INSTALL_NOT_COMPLETED`, it is displayed as `PARTIALLY_INSTALLED` in the inventory.

The following flowchart depicts how the states on the CPP based nodes are mapped to the values displayed in the inventory:



4.8.1 States on Node

State	Description
INSTALL_EXECUTING	Installation of the UpgradePackage or PiuType MO is executing.
INSTALL_NOT_COMPLETED	The upgrade package is not fully installed, that is, only parts of it have been installed.

State	Description
INSTALL_COMPLETED	The upgrade package has been installed.
UPGRADE_COMPLETED	The upgrade (the action update, upgrade, or rebootNodeUpgrade) has been successfully executed. If this was the current state when an action was started. it is also the end state.
UPGRADE_EXECUTING	Upgrade (the action update, upgrade, or rebootNodeUpgrade) is executing.
AWAITING_CONFIRMATION	The execution of upgrade needs confirmation before it proceeds.
ONLY_DELETEABLE	In this state, the only allowed operations are delete and action updateFTPServerData. If any other type of action is started, an exception ActionNotAllowedException is thrown.

4.8.2 States in ENM

State	Description
PARTIALLY_INSTALLED	This indicates that the software is being installed on the node, but it is not yet activated.
EXECUTING	This indicates that the software has been installed, is active on the node, but has not been permanently activated.
INSTALLED	Software is installed on the node, it is not currently active, and it can be deleted.
ONLY_DELETEABLE	Node has identified that this software can be deleted.
SYSTEM_USED	Software is installed, but it is not currently active on the node. The software cannot be deleted. For example, because dependencies exist between this software and other active software.

4.9 Delete Upgrade Packages Using Software and Hardware Manager

> Describes the deletion of upgrade packages and its associated backups for both CPP based nodes and ECIM-based nodes.

Delete Upgrade Packages in General.

SHM enables the deletion of an upgrade package, and all its associated backups, from nodes. This is done to free up disk space before, or after an upgrade of the nodes. The use case is supported for both CPP based and ECIM-based nodes.

The `Delete Upgrade Package` activities are the same for both CPP-based nodes (such as ERBS, MGW) and ECIM-based nodes (such as baseband radio node, SGSN-MME).

The `Software Administration` page provides inventory information about the software running on the nodes in the network. It is displayed under the **Software Information** tab; and information about all the upgrade packages on the nodes are listed under **Upgrade Packages on Node** tab.

To delete an `Upgrade Package`, select the required node in **Network Elements** panel then click the **Upgrade Packages on nodes** tab. Select the package in

Upgrade Packages table then click **Delete Upgrade Packages** from the action bar. A dialog box is displayed. Select the required `Additional Options` and click **Delete** to delete the `Upgrade Package`.

In the `Software Hardware Manager` page, use the `Jobs` table to monitor the overall progress of a delete upgrade package job. The `Progress` column displays the job progress in terms of percentage. The `Delete Upgrade Package` job progress is updated according to the completion of `Delete Upgrade Package` activities on the nodes specified in the job.

You can also monitor the job progress at individual node level specified in the job using `Job Details` page, and also at individual node activities level using `Node Activities Panel`. In the Jobs table, select the **relevant job** > **View Job Details** > **select the relevant node from the Node Progress table** > **Node Activities**. In the `Node Activities` panel, you can see the progress of the job at activity level.

For CPP-based nodes, for each job activity on the `Node Activities` panel, the progress bar and progress percentage increases as the `Delete Upgrade Package` operation progresses on the node. The activity is marked as completed when it is finished on the node.

For ECIM-based nodes, for each job activity on the `Node Activities` panel, the progress bar and progress percentage increases according to progress percentage notifications received for each upgrade package deletion.

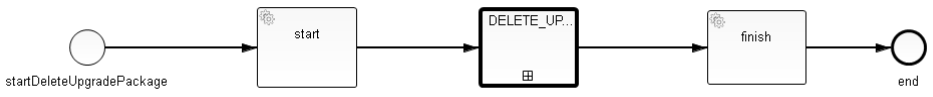
Node Selection in Delete Upgrade Package Job Configuration

It is possible to create a single `Delete Upgrade Package` job that deletes upgrades packages from different node types, such as ECIM-based or CPP-based nodes, and so on.

SHM does not apply restrictions on the number of nodes in a job. However, the recommended maximum number is `1000` nodes in a single job.

4.9.1 Delete Upgrade Package Job Flow for CPP Flow

> Describes the Delete Upgrade Package activities that are executed on each node.



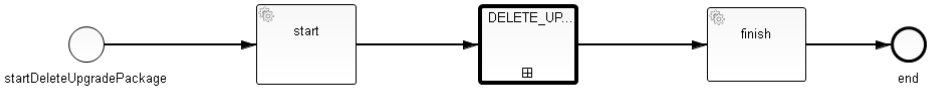
Delete Upgrade Package Activity.

This activity deletes the upgrade package on the node only when `deletePreventingCVs` and `deletePreventingUPs` are deleted.

SHM enables `Delete Upgrade Package` operation on a node which has `deletePreventingCvs` and `deletePreventingUPs`, by providing the `Delete from rollback list` and `Delta referred upgrade packages` option.

4.9.2 Delete Upgrade Package Job Flow for ECIM Flow

Describes the Delete Upgrade Package activities that are executed on each node.



Delete Upgrade Package Activity.

This activity deletes the upgrade packages on the node only when all the backups referred to the selected upgrade package are deleted.

SHM enables `Delete Upgrade Package` operation on a node which has backups, by providing the `Delete referred backups` option.

4.10 Onboard and Delete VNF Software Packages

VNF software packages for vRAN nodes can be managed using SHM.

SHM provides import, on-board, and delete functionality for VNF software packages.

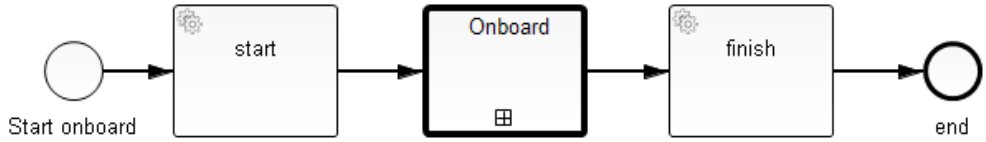
On-board and delete VNF software packages functions are executed as SHM jobs.

4.10.1 Onboard VNF Package Job Workflow

Onboarding is the process where VNF software package available in ENM is made available in NFV Orchestrator (NFVO).

Onboard job allows the operator to onboard the software packages to NFV orchestrator using SHM GUI.

The following workflow diagram depicts the list of activities that are executed sequentially per NFVO during an onboard job execution.



Onboard Activity

SHM onboards the VNF software package to NFVO using VNF package management rest interface provided by NFVO. This interface is part of `os-ma-nfvo` and `or-vnfm` reference points.

Once package is onboarded, the corresponding software package details is updated in SHM.

Cancel is not supported for onboard activity.

4.10.2 Delete VNF Package Job Workflow

VNF software packages available in NFVO can be deleted using SHM. Delete package job allows the operator to delete a software package from ENM or NFVO.

The following workflow diagram depicts the list of activities that are executed sequentially per NFVO during a delete job execution.

Delete Software Package Activity

SHM deletes the VNF software package from NFVO using VNF package management rest interface provided by NFVO. This interface is part of `os-ma-nfvo` and `or-vnfm` reference points.

Based on the operator selection, the software package is deleted from ENM SMRS as well. If a package is in use, NFVO marks the package as `delete pending`, and rejects the deletion.

4.11 Automatic Import of Software Packages from Network Element SoftwareStore (CAS-C)

Note: To configure Automatic Import of Software Packages from Network Element SoftwareStore (CAS-C), see *Configure Automatic Import of Software Packages from Network Element Software Store (CAS-C)* in the ENM Configuration System Administrator Guide 1/1543-AOM 901 151-1.

If `Automatic Import` is active, ENM polls the Network Element Software Store (CAS-C) to check for new node software packages. The parameter `NE_SOFTWARE_STORE_SCHEDULER_INTERVAL_IN_MINUTES` controls the frequency of this polling. For details on this parameter, see *Parameter Table - Configuration Management Service for the Network Element Software Store (CAS-C)* of the ENM Configuration System Administrator Guide 1/1543-AOM 901 151-1. For details on how to update this parameter see *Update Scheduler Interval Parameter for Polling to the Network Element Software Store (CAS-C)* of the ENM Configuration System Administrator Guide 1/1543-AOM 901 151-1.

ENM imports all the node software packages from Network Element Software Store (CAS-C) for a respective node type except the software packages which are already imported into ENM or software packages that were deleted from ENM by housekeeping functionality during Automatic Import process.

There is also a Housekeeping functionality built into the Automatic Import process. This is controlled by the parameter `TOTAL_NUMBER_OF_IMPORTED_SOFTWARE_PACKAGES_TO_RETENTION`. For details on this parameter, see *Parameter Table - Configuration Management Service for the Network Element Software Store (CAS-C)* of the *ENM Configuration System Administrator Guide 1/1543-AOM 901 151-1*. The parameter controls the number of node software packages to be retained after automatic import has run for a node type. This value is applicable to each individual node type that is to be automatically imported. The housekeeping functionality continues to delete the oldest node software package for a respective node type. This continues until the number of node software packages is equal to the value set by the parameter `TOTAL_NUMBER_OF_IMPORTED_SOFTWARE_PACKAGES_TO_RETENTION`. The oldest node software package is decided based on the "`Release Date`" of a node software package. If a node type does not have any node software packages marked for automatic import, then no housekeeping is performed for this node type.

4.11.1 Example Automatic Import Scenarios

Scenario 1 - Where No Automatic Import Happens:

ENM already has eight node software packages imported; four packages for ERBS, two packages for RadioNode and two packages for SGSN-MME.

Two latest packages become "Available for Import" from the Network Element Software Store (CAS-C); one package for ERBS and one package for RadioNode.

The following parameters are set:

— NE_SOFTWARE_STORE_SCHEDULER_INTERVAL_IN_MINUTES = 60

— TOTAL_NUMBER_OF_IMPORTED_SOFTWARE_PACKAGES_TO_RETENTION = 2

— AUTOMATIC_SWPKG_IMPORT_FROM_NE_SOFTWARE_STORE = false

Result:

ENM polls the Network Element Software Store (CAS-C), how often this happens is controlled by NE_SOFTWARE_STORE_SCHEDULER_INTERVAL_IN_MINUTES.

Since automatic import is disabled (AUTOMATIC_SWPKG_IMPORT_FROM_NE_SOFTWARE_STORE = false), no automatic import occurs.

ENM updates the Available for Import table in SHM with the latest data from the poll of Network Element Software Store (CAS-C).

Scenario 2 - Where ENM Housekeeping Deletes Node Software Packages, Then Automatically Imports Node Software Packages for Different Nodes:

ENM already has eight node software packages imported; four packages for ERBS, two packages for RadioNode and two packages for SGSN-MME.

Two latest packages become "Available for Import" from the Network Element Software Store (CAS-C); one package for ERBS and one package for RadioNode.

The following parameters are set:

— NE_SOFTWARE_STORE_SCHEDULER_INTERVAL_IN_MINUTES = 60

— TOTAL_NUMBER_OF_IMPORTED_SOFTWARE_PACKAGES_TO_RETENTION = 2

— AUTOMATIC_SWPKG_IMPORT_FROM_NE_SOFTWARE_STORE = true

Result:

ENM polls the Network Element Software Store (CAS-C), how often this happens is controlled by NE_SOFTWARE_STORE_SCHEDULER_INTERVAL_IN_MINUTES.

ENM calculates that there are two nodes that have packages to be automatically imported; ERBS and RadioNode.

Two ERBS packages (the oldest two) are deleted from ENM, based on the TOTAL_NUMBER_OF_IMPORTED_SOFTWARE_PACKAGES_TO_RETENTION parameter.

Once the packages have been deleted from ENM, one ERBS and one RadioNode package that became available at the start of the scenario are now automatically imported.

The SGSN-MME packages are unaffected, as there are no packages to be automatically imported for SGSN-MME.

Scenario 3 - Where ENM Automatically Imports Node Software Packages for Different Nodes, and No Housekeeping Is Required:

ENM already has six node software packages imported, two packages for ERBS, two packages for RadioNode and two packages for SGSN-MME.

Two latest packages become "Available for Import" from the Network Element Software Store (CAS-C), one package for ERBS and one package for RadioNode.

The following parameters are set:

— `NE_SOFTWARE_STORE_SCHEDULER_INTERVAL_IN_MINUTES` = `60`

— `TOTAL_NUMBER_OF_IMPORTED_SOFTWARE_PACKAGES_TO_RETENTION` = `12`

— `AUTOMATIC_SWPKG_IMPORT_FROM_NE_SOFTWARE_STORE` = true

Result:

ENM polls the Network Element Software Store (CAS-C), how often this happens is controlled by `NE_SOFTWARE_STORE_SCHEDULER_INTERVAL_IN_MINUTES`.

ENM calculates that there are two nodes that have packages to be automatically imported; ERBS and RadioNode.

No packages are deleted, as the number of packages for ERBS is less than the `TOTAL_NUMBER_OF_IMPORTED_SOFTWARE_PACKAGES_TO_RETENTION` parameter value and the number of packages for RadioNode is less than the `TOTAL_NUMBER_OF_IMPORTED_SOFTWARE_PACKAGES_TO_RETENTION` parameter value.

The two latest packages that became available at the start of the scenario are now automatically imported and ENM now has three ERBS packages "Imported" and three RadioNode packages "Imported".

The SGSN-MME packages are unaffected, as there is no packages to be automatically imported for SGSN-MME.

4.12 SHM Load Controller

SHM is an application in ENM where operators perform various types of jobs. The jobs supported by SHM include:

— Backup

— Upgrade

— License

— Restore

— Backup House Keeping

— Node Restart

To distribute the load and manage resources, SHM uses the `Load Controller`.

SHM defines the load controller value for each activity of a job. This is the maximum number of instances of that activity that can run concurrently.

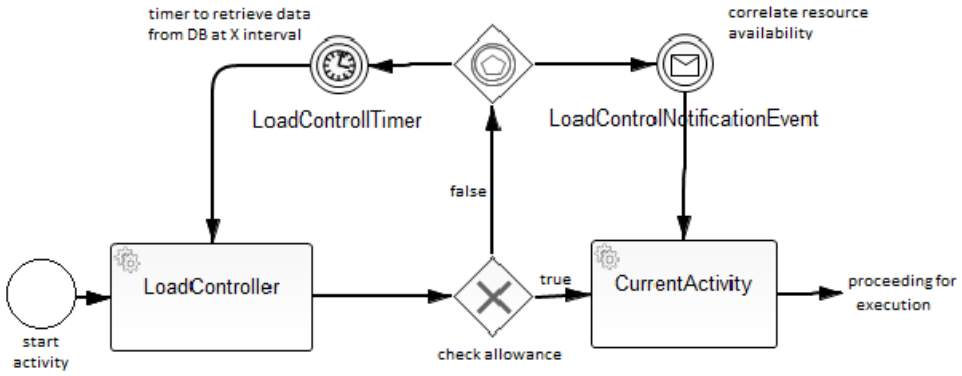
The SHM operator triggers upgrade jobs on multiple ECIM nodes, and for each node, `PREPARE`, `VERIFY`, `ACTIVATE`, or `CONFIRM` activities are selected. Based on the activity type, the load controller divides them into smaller batches to execute one at a time.

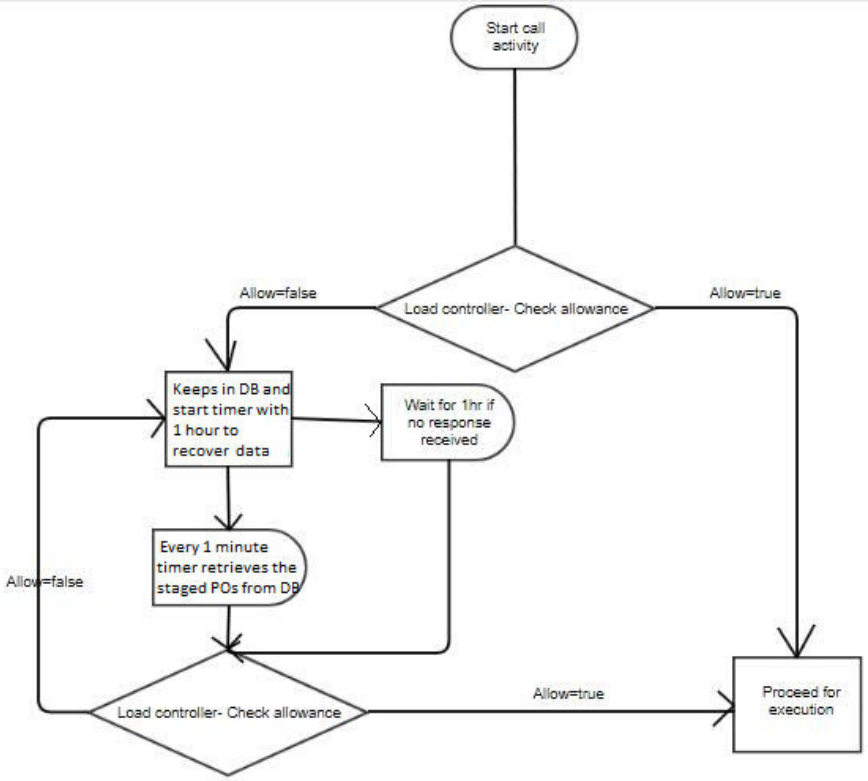
Load controller operates in the same way for SHM jobs being executed from CLI or UI. Based on the execution start time, jobs are picked up.

Example:

The load controller value of ECIM Upgrade **ACTIVATE** activity is `1000`, and an upgrade job is triggered on `2000` nodes. In this scenario, only `1000` activate activity jobs can run at a time in SHM, irrespective of how many activity jobs are triggered.

The remaining activities are stored in DB and are executed as one of the current executing activities completes.


timer to retrieve data
from DB at X interval
correlate resource
availability
LoadControllTimer
LoadControlNotificationEvent
false
LoadController
true
CurrentActivity
proceeding for
execution
start
activity
check allowance


Start call
activity
Allow=false
Load controller- Check allowance
Allow=true
Keeps in DB and
start timer with
1 hour to
recover data
Wait for 1hr if
no response
received
Every 1 minute
timer retrieves the
staged POs from DB
Allow=false
Load controller- Check allowance
Allow=true
Proceed for
execution

4.13 Instantaneous Licensing Functionality - LKF Request Job

Describes the instantaneous licensing functionality support in SHM.

`Instantaneous Licensing` function is an alternative to `Manual Licensing`. The `Instantaneous Licensing` function automates and speeds up licensing actions.

Instantaneous license functionality is implemented as license request job in ENM.

License request functionality has the following use cases. For all the use cases, SHM creates a license request job.

License Key Refresh from Node

In this use case, operator triggers the license key request activity action directly on node, without creating a job in SHM.

Operator can use node cli or ENM CM CLI to trigger refreshKeyFile action on InstantaneousLicensing MO.

Once this MO action is performed node sends LKF Refresh Request to ENM.

After receiving the instantaneous license request, SHM creates the job automatically.

SHM requests license key files from ELIS, and if install activity is selected, received license key files are installed on the node.

License Key Refresh from SHM UI

In this use case, operator creates `Refresh License Keys` job in using ENM SHM UI.

ENM SHM performs `refreshKeyFiles` action on node, and receives refresh requests from node. Then SHM requests license key files from ELIS and, if install activity is selected, the received license key files are installed on the node.

License Key File for Software Upgrade

In this use case, operator creates `Upgrade License Keys` job in ENM SHM by selecting list of nodes and software packages. ENM SHM requests License Key Files from ELIS for the selected software package using software release information present in software package. If install activity is selected, received license key files are installed on the node.

License Key File for Node Integration

In this use case, operator creates `Integrate License Keys` job in ENM SHM, by providing details as comma separated (CSV) file.

ENM SHM requests license key files from ELIS based on the user information. SHM receives license key files, and license key files are imported into SHM.

License Key File for Capacity Increase

In this use case, node triggers the license key requests for additional capacity when operator performs resource configuration or reconfiguration on node.

After receiving the instantaneous license request, SHM creates the job automatically.

SHM requests license key files from ELIS, and received license key files with new capacity are installed on the node.

Note: For License Key Refresh from node and License Key File for Capacity Increase, usecases jobs will be created automatically within ENM SHM with username "administrator". License key files received from ELIS will be imported into ENM using the same username.

Activities Applicable for Different Jobs:

— License Key Refresh from Node:

  - Request
  - Install

— License Key Refresh from SHM UI:

  - Refresh
  - Request
  - Install

— Upgrade Licensing Keys:

  - Request
  - Install

— Integrate Licensing Keys:

  - Request

— License Key File for Capacity Increase

  - Request

- Install

Prerequisites

— ENM configured and connected to CAS-C.

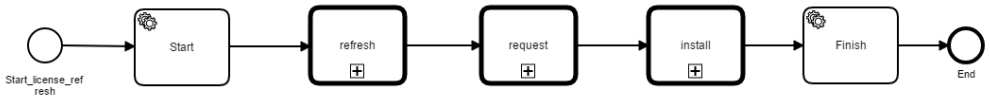
— Node has instantaneous licensing enabled.

Instantaneous licensing functionality can be enabled on node by setting `administrativeState` attribute of `InstantaneousLicensing` MO as `UNLOCKED`.

Whenever `administrativeState` is changed to `UNLOCKED` on node, ENM SHM configures the rest URI Path on node, to which node sends instantaneous licensing request.

All use cases of instantaneous licensing functionality are supported in SHM, using LKF request job.

The following workflow diagram depicts the list of activities that are executed as part of license request job:



Refresh Activity:

As part of this activity ENM SHM performs `refreshKeyFile` action on `InstantaneousLicensing` MO. This is an asynchronous action, and node sends the progress information.

Node sends the instantaneous license request to ENM. This request is received as HTTPS request on interface hosted in SMRS.

If `refreshKeyFile` action is directly performed on node without ENM intervention, then ENM SHM automatically creates job for such requests, and marks this activity as skipped.

Request Activity:

As part of this activity ENM SHM requests ELIS licensing system for the licenses. ELIS generates the licenses, and transfers the license key files to CAS-C drop box.

SHM retrieves the license key files, and imports them into SHM database.

Install Activity:

As part of this activity, ENM SHM installs the license key files received from ELIS on node.

5 Node Health Check Overview

Node Health Check (NHC) is a web-based application used to perform health checks on nodes and Network Elements (NEs) in your network. NHC allows you to perform the Setup and Run health check report.

The following operations can be performed in NHC Application:

— Create health check profiles, by customizing health checks, and its input parameters.

— View detailed health check profile information.

— Delete health check profiles.

— Execute health check reports, by selecting predefined sets of rules with default parameters, or by selecting health check profiles with customized rule parameters.

— View detailed information in health check reports about the health status of nodes.

— Compare health check reports.

Prerequisites

— NHC uses web push to execute many of the use cases. For the web push to work properly, the URL must to have the required certificates on the browser. To install browser certificates, see *ENM Identity and Access Management System Administration Guide*, 2/1543-AOM 901 151-1 Uen.

— Operator must have the nhc_administrator role to create reports or profiles, and nhc_operator role to view reports or profiles.

— Operator must have Target Based Access Control rights to perform actions on the required nodes.

— Profile must be created to execute health check report with profiles, see Create Health Check Profile on page 222.

— To create Profile, import the supported software packages into ENM for Baseband Radio nodes. This is done using the SHM Import Software Package functionality.

— Ensure the profile count is not at its maximum limit. The default maximum allowed profile count is 100. You are not allowed to create a profile if the profile count is at the maximum limit. To create a new profile, delete unused profiles and try again.

5.1 Create Node Health Check Report

NHC lets you create a Node Health Check Report for the nodes in your network.

To perform a health check on nodes, set up a health check report using the NHC wizard.

To create a Node Health Check Report, in the Node Health Check page, click on `Setup & Run Health Check Report`, select the required nodes in the Network Elements panel, then select `Health Check Configuration`. You can `Select Schedule` to run the Health Check Report immediately, and also to run periodically.

In the Health Check Reports page, use the reports table to monitor the overall progress of a Node Health Check Report. The Progress column displays the report progress in terms of percentage.

The Node Health Check Report progress is updated as the Node Health Check Activities complete on the nodes specified in the report.

Node Selection in Create Node Health Check Report

— You can create a single Health Check Report for different node types.

— When you create a Health Check Report, you can specify individual nodes, or use collections or saved searches, to specify the nodes. The selected nodes are evaluated during the Health Check Report execution time.

NHC does not apply restrictions on the number of nodes in a Health Check report. However, the recommended maximum number is `1000` nodes in a single report.

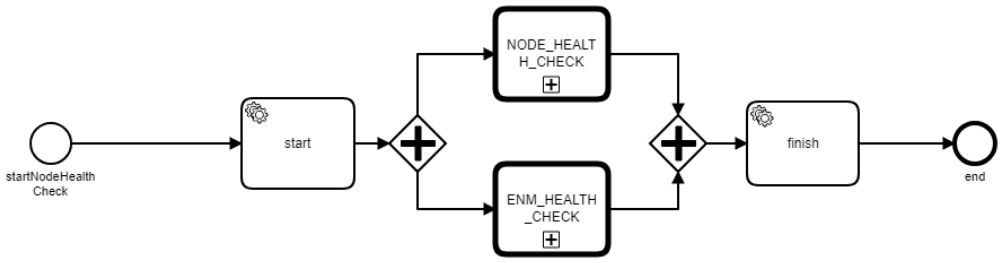


Figure 11 Health Check Report Workflow

5.1.1 Node Health Check for ECIM Nodes

1. `Predefined Health Check Configuration`:

   Health check report can be created by selecting any of the following options:

   - `Categories`

Any one of the following `Categories` can be selected for a set of `Baseband Radio` nodes (L18 Q3 release onwards):

- PRE_INSTALL
- PRE_UPGRADE
- POST_UPGRADE
- TROUBLESHOOT
- SITE_ACCEPTANCE

- `Additional Health Check`

  Any of the following health checks can be selected for ECIM platform nodes:

  - Critical Alarm Check.
  - Major Alarm Check.
  - Configuration Management Sync Status.

2. `Custom Health Check Configuration`:

   Profile creation is currently supported for `Baseband Radio` nodes. Support is from 20.Q2 release onwards.

   A profile is selected in this health check report configuration by customizing `node health check` and `additional health check`. To create a profile, see Create Health Check Profile on page 222.

5.1.2 Node Health Check for CPP Nodes

Predefined Health Check Configuration

Health check report can be created by selecting any of the following options:

- `Categories`

  Any one of the following `Categories` can be selected for a set of `eNodeB DU Radio` nodes:

  - PRE_INSTALL
  - PRE_UPGRADE
  - POST_UPGRADE

- `Additional Health Check`

Any of the following health checks can be selected for CPP platform nodes:

- Critical Alarm Check.
- Major Alarm Check.
- Configuration Management Sync Status.

Custom Health Check Configuration

A profile is selected in this health check report configuration by customizing `node health check` and `additional health check`. To create a profile, see Create Health Check Profile on page 222.

5.2 Create Health Check Profile

NHC enables you to create a Health Check Profile for a node type in your network.

Use NHC Create Health Check Profile page for profile creation.

To create a Health Check Profile, in the `Create Node Health Check` page, provide profile `Name`, `Description`, and `Labels`. Then select `Node Type`, and its supported `Software Version` (applicable for Baseband Radio nodes), to customize the health check parameters. Customization of health check includes changing severity, changing inputs parameter values, and including or excluding a particular health check.

View detailed health check Profile information, such as included health check, parameters, and overview of the profile, in `Health Check Profiles` page.

The default maximum profile count is `100`. You are not allowed to create a profile if the profile count is at the maximum limit. To create a new profile, delete unused profiles and try again.

6 Fault Management Overview

> The ENM fault management system receives alarms from objects in the supervised telecommunications networks. It translates the alarm information to a standard format, enriches or handles alarms with FMX, forwards alarms to NMS, and logs the alarms.

The ENM fault management application shows the alarm situation for one or more Network Elements in the network, and for internal ENM alarms.

The following features give an overview of the most important information:

— The `Alarm Monitor` displays open alarms, based on the applied Network Elements and alarm attributes.

— The `Related Alarm Monitor` displays open alarms, based on the alarm correlation UUID they belong to, and alarm attributes.

— The `Alarm Monitor` enables the user to import Network Elements from the `Network Explorer` application.

— The `Alarm Supervision Status` displays information related to node status. It allows the user to enable, disable, and synchronize supervision on single or multiple nodes.

— The `Alarm Overview` displays the nodes information. This is in descending order, based on the number of alarms, or alarm type present on the Network Elements.

— The `Alarm Overview` provides a graphical representation of the different severity or alarm types present on the network element.

— The `Alarm Search` displays both open and historical alarm details. The alarms are searchable, on event time, specific attributes, or specific nodes.

The following actions are supported:

— `Acknowledge`, `clear`, `unacknowledge`, and `attach` comments to alarms.

— `Acknowledge`, `clear`, `unacknowledge` alarms, and on request, all related alarms belonging to the same correlation UUID.

— `Enable` or `Disable` supervision on a node.

— `Manual` or `Automatic` synchronization between ENM alarm list and the network element alarm list.

— Create alarm `auto-acknowledge` routes on Network Elements with specified alarm attributes. Alarms that match the attributes in the route are filtered and acknowledged.

— Create alarm `Save to File` routes on Network Elements with specified alarm attributes. Alarms that match the attributes in the route are sent to `File`.

— Export alarm information to files.

— Access available alarm information. For alarms belonging to a correlation UUID, it accesses alarm information of related alarms.

— View alarm handling instructions, that are Operational Instructions (OPI) for an alarm, in an internal document viewer.

— Forward alarms to NMS through FM NBIs (CORBA, SNMP, BNSI).

— Heartbeat check, to detect connection failures between ENM and a node.

— `Automatic Alarm Overload Protection`, to reduce alarms towards Operator and NMS when alarms exceed a threshold.

— Create alarm `email` routes on Network Elements with specified alarm attributes. Alarms that match the attributes in the route are sent to an email address (disabled by default).

— Change parameters in PIB, to govern FM behaviors. See *ENM Parameter List* and *ENM Monitoring System Administrator Guide*.

Note: Throughout this Chapter, a reference to `alarm` also refers to `alarm and error`, unless the context states otherwise.

6.1 Fault Management Concepts

The Fault Manager consists of various components. The following figure illustrates the high-level components.

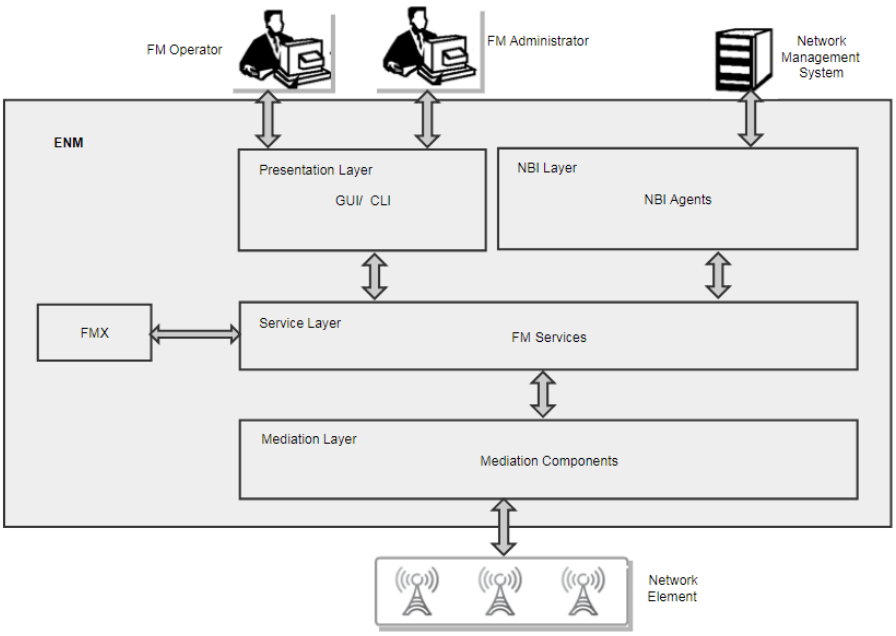


Figure 12 Fault Manager Components

`Fault Management` (FM) in ENM performs the following:

— Communicate alarms and error messages coming from Network Elements to FM users.

— Perform alarm correlation. This correlates new alarms and error messages with existing alarms and error messages.

— Elaborate correlation information to:

  - Distinguish primary alarms from secondary alarms.
  - Associate primary and secondary alarms that have the same alarm correlation UUID.
  - Associate alarms that come from the same compute resource, based on the common system UUID.
  - Identify the role of the compute resource based on the compute name.

— Automatically acknowledge alarms.

— Forward received alarms and error messages to presentation functions.

— Forward received alarms and error messages to the user. They view and work with the alarms and error messages using the Graphical User Interface (GUI) or Command Line Interface (CLI).

— Forward alarms to NBI agents. They send received alarms to other products, such as Network Management Systems (NMS).

— Process and send alarms to FMX, according to the rules configured. FMX triggers different events on alarms sent from FM. The Fault Manager Expert automates some functionality, for example, `Acknowledge`, `Clear`, `Show`, and `Hide` alarms.

— FM Service Layer uses two databases:

Neo4j This database contains alarm information of open alarms.

SOLR This database contains alarm information, and the actions performed on different alarms.

Note: For nodes where ENM sets the SNMP target automatically, the operator does not need to create targets manually on the node. For list of such nodes, see *ENM Product Description, 1/1551-AOM 901 151 Uen*.

6.1.1 Node Supervision

> To handle alarms from a Network Element, an entry for the Network Element must exist in the ENM database. Alarms from a Network Element are related to the Network Element sending the alarm, or to an associated object, for example, equipment or a connection object.

Once the Network Element is added into the system, FM alarm supervision can be turned on.

`Supervision Status` and `Subscription Status` represent the `Administrative` and `Operational State` of the Network Element.

Table 3 Network Elements Supervision Status

Supervision Status	Subscription Status	Node Status	Allowed Action
Off	false	Supervision Disabled	Enable Supervision.
Off	true	Disabling supervision	Wait while disabling.
On	false	Enabling supervision	Wait while enabling.
On	true	Supervision Enabled	Disable Supervision.

Change in FM Specific Attributes

The following characteristics can be changed for a Network Element using ENM CLI commands:

— `Alarm Supervision`: `On` or `Off`. Also includes `Heart Beat Supervision`: `On` or `Off`.

— `Heart Beat Timeout and Interval`: See Heartbeat Management on page 229.

— `Automatic Synchronization`: `On` or `Off`. Alarm synchronization is performed:

  • When an alarm overload protection ceases; the synchronization happens against the nodes whose alarms were disregarded during protection.

  • When clearing `Alarm Rate Threshold Crossed Alarm` for a node.

— `Automatic Synchronization`: `On` or `Off`. If set to `On`, alarm synchronization is performed in the following scenarios:

  — When supervision is turned `On`;

  — When the Heartbeat Failure alarm is cleared by FM Mediation on reconnection with the Node;

  — When an `alarmListReBuilt` is received from the Node;

  — When ENM, monitoring the alarm reporting, detects an incorrect sequence of alarms;

6.1.1.1 Rebalancing of Nodes

> After a rolling upgrade, it is expected that only (N-1) out of N FM Mediation Instances are available to supervise the network post upgrade.

FM automatically rebalances nodes among the available Mediation Instances. This reload-balancing is done only on a subset of nodes to avoid redistribution of the supervision on large number of nodes. This would cause a large amount of alarm synchronization towards the Node, and a temporary supervision loss on those nodes.

The reload-balancing is done based on the following equation:

```
Maximum Allowed Load With Threshold = ( Normal Allowed Load + (Reload Threshold Percentage * Normal Allowed Load) / 100 )
```

Where:

Maximum Allowed Load With Threshold
: Maximum number of nodes allowed to be supervised by each of the mediation instance after the reload balancing.

Normal Allowed Load
: Node Count (total number of Nodes being supervised) / MS Instance Count (total number of mediation instances available).

Reload Threshold Percentage

% of nodes allowed to be supervised beyond the `Normal Allowed Load`, a configurable parameter with a default value of 5.

At the end of the reload-balancing, each of the Mediation Service (MS), except the final upgraded MS, must be supervising `Maximum Allowed Load With Threshold` number of nodes. The final upgraded MS supervises the rest of the nodes. That is `Node Count` - `Maximum Allowed Load With Threshold`.

This is performed eight hours after the first FM MS instance is upgraded in the ENM upgrade procedure. This duration is a configurable parameter, and can be changed if necessary.

Note: It is not advisable to change this parameter. Contact the `System Administrator` before changing it.

To reemphasize that alarm supervision is balanced properly among all MS instances, rebalancing of nodes happens only after a successful upgrade of ENM. It is required once after each upgrade.

Immediately after an upgrade, load is balanced only on (N-1) MS instances for already supervised nodes. After an upgrade, if any node is newly supervised, this node is mediated by the least loaded MS instance that is an Nth MS.

6.1.2 Alarm Synchronization

At synchronization, all the alarms in the FM open alarm list are checked against the alarms fetched from the Network Element alarm list. If an alarm from the Network Element alarm list is not found in the FM alarm list, the new alarm is inserted. Alarms present in FM alarm list, but not in the Network Element alarm list, are cleared in FM.

ENM performs node alarm synchronization in the following cases:

— Request by user;

— Request by NBI;

— Request by APS (for example, for DB failure or `orphan clear issue`);

— At Supervision `On`;

— At Heartbeat Alarm `Off`;

— Request by node through `Alarm List Rebuilt`;

— `Out of synch` (alarm received out correct sequence numbering);

— When `Alarm Overload Protection` ceases, for nodes whose alarms were eliminated during protection;

— When clearing `Alarm Rate Threshold Crossed Alarm` for a node.

6.1.3 Heartbeat Management

> FM Heartbeat for FM checks the communication between the node and the ENM system. Where connectivity issues are detected, Heartbeat Failure alarm is reported on the Alarm Monitor.

Heartbeat Supervision

The regular monitoring of managed element responses is called Heartbeat Supervision. The `Mediation Component` performs this service, by receiving or polling the network element for heartbeat signals, at regular predefined time intervals. If no heartbeat signal is received within the time interval specified, a `Heartbeat Failure Alarm` is generated. This alarm is cleared when FM detects that the node is responding again.

Heartbeat Supervision ON

Using the GUI, or the web-based ENM CLI alarm command, it is possible to enable FM Node Supervision to monitor the connection between ENM and the node.

The Heartbeat mechanism can supervise nodes according to the following different approaches of the supervised nodes:

— PULL mode.

In PULL mode, ENM has an active role, while the supervised node is passive. ENM periodically monitors the status of the connection and interrogates the node by a request sent through the FM specific protocol.

Supervision is made according to the configured heartbeat frequency. The heartbeat monitoring for nodes supporting the PULL mode is implemented by a periodical check to test the status of the notification subscription on the NE. The notification subscription can be checked as long as a connection to the node is set in place first, so if the check fails ENM attempts to reconnect to the node.

When supervision is set to `ON`, ENM configures the FM mediation with the `heartbeatinterval` attribute of the `FmAlarmSupervision` MO (by default `100` seconds). This attribute is used to configure the frequency used by the ENM FM Mediation to check the connectivity to the node. The `heartbeatTimeout` attribute of the `FmAlarmSupervision` MO (by default, `300` seconds) is the size of the time window used by the ENM FM Mediation for raising an alarm when the connectivity is broken.

— PUSH mode

In PUSH mode, the node is responsible for sending a regular heartbeat notification to ENM, based on the configured heartbeat frequency.

When supervision is set to ON, ENM configures the heartbeat interval on the node to generate the notification for FM (`heartbeatinterval` attribute of `FmAlarmSupervision` MO, by default, `100` seconds). ENM mediation performs a periodical check (that is the `heartbeatinterval`) to test the heartbeat notification. The `heartbeatTimeout` attribute of the `FmAlarmSupervision` MO (by default, `300` seconds) is the size of the time window used by the ENM FM Mediation to declare if the connectivity is broken.

Note: PUSH is the preferred mode if the supervised node type supports it.

Example of Heartbeat Support on ENM-Managed Node Types

On CPP nodes (CORBA FM protocol), the PULL mode is used. ENM checks the status of the connection with the node at regular `heartbeatTimeout`. If no heartbeat response is received after the `heartbeatTimeout`, a Heartbeat Failure Alarm is generated. This alarm is cleared when FM detects that the node is responding again.

On MINI-LINK Indoor nodes (SNMP FM protocol), the PULL mode is used. ENM checks the status of the connection with the node at any `heartbeatInterval` with an SNMP GET. If no heartbeat responses are received within the `heartbeatTimeout`, a Heartbeat Failure Alarm is generated. This alarm is cleared when FM detects that the node is responding again.

On Baseband Radio Nodes (SNMP FM protocol), the PUSH mode is used. Nodes are configured to send an SNMP heartbeat trap at every `heartbeatinterval` when supervision is enabled. At every `heartbeatinterval`, ENM checks if the heartbeat trap is received. If no traps are received within the `heartbeatTimeout`, a Heartbeat Failure Alarm is generated. This alarm is cleared when FM detects that the node is sending heartbeat traps again.

On SGSN-MME and SGSN-vMME nodes (SNMP FM protocol):

— Releases supporting the FM ECIM model (greater than or equal to 15B-CP01) use the PUSH mode also.

Heartbeat Supervision OFF

If Node Supervision is set to `OFF` using the UI or the web-based ENM CLI, the heartbeat check is disabled. As a result, the Heartbeat Failure Alarm, if present, is automatically cleared by ENM.

FmAlarmSupervision MO Heartbeat Attributes and Suggested Configuration

heartbeatinterval

The time frequency used for checking the status of the FM connection to a node.

— Default value = 100

— Time units = seconds

— Constraint: use a value greater than 0

Note: Exercise caution when configuring the heartbeatInterval attribute as the heartbeat check is a resource-consuming operation for ENM.

heartbeatTimeout

The time window used by ENM for generating the Heartbeat Failure Alarm.

— Default value = 300

— Time units = seconds

— Constraints: use a value greater than 0.

Note: As a rule, the heartbeat timeout must be longer than the heartbeat interval (by default, the heartbeat timeout is three times longer than the heartbeat interval).

Prevent the node goes into timeout (heatbeatFailure), setting the values of Heartbeat Timeout and Heartbeat Interval supported by the node.

In general, set the Heartbeat Timeout and Interval values not too low compared to the default values.

For example, the following are the lowest values supported by a Baseband Radio Node G2 in TLS:

HB timeout = 90 and HB Interval = 30.

For example, the following are the lowest values supported by a TCU02 and SIU02 Nodes:

HB timeout = 360 and HB Interval = 120.

The HB Interval and Timeout must be multiple of 60 for TCU02 and SIU02 nodes.

Reducing heartbeatTimeout and heartbeatInterval values must be taken with care particularly on large networks. As an example with 20000 nodes using the default values there are already 200 messages / second only to manage heartbeat.

The heartbeat mechanism notifies the loss of connection after one heartbeatTimeout plus one heartbeatInterval.

By sampling the asynchronous indications of connection every heartbeatInterval, an additional delay between 0 and less than one heartbeatInterval may apply.

Change Heartbeat Timeout and Heartbeat Interval

To change `heartbeattimeout` and `hearbeatinterval` values, see the *alarm command set* in the ENM CLI online help.

Note: The values of the `heartbeatTimeout` and `heartbeatInterval` attributes cannot be changed using the UI.

A Heartbeat alarm is raised immediately, for AXE nodes (BSC, Compact MSC-S, and HLR-FE node types), when there is a failure in alarm supervision.

For further information on the Heartbeat timeout settings, see *Fault Management Heartbeat Timeout and Automatic Synchronization Configuration* section of *ENM System Administrator Guide 1/1543-AOM 901 151*.

6.1.4 Overload Protection

By default an alarm overload protection mechanism is enabled on ENM.

If FM crosses the `alarmOverloadProtectionThreshold`, an alert is raised by ENM to signal that an overload is detected.

Where:

`alarmOverloadProtectionThreshold` is the maximum number of alarms allowed within the alarm rate check interval of 5 minutes.

During the overload, alarms and alerts are processed if their `PerceivedSeverity` and their `RecordType` are the following:

PerceivedSeverity

— CRITICAL

— MAJOR

— INDETERMINATE

— CLEARED

RecordType

— ALARM

— REPEATED_ALARM

— SYNCHRONIZATION_ALARM

— HEARTBEAT_ALARM

— NON_SYNCHABLE_ALARM

— REPEATED_NON_SYNCHABLE

— ERROR_MESSAGE

— REPEATED_ERROR_MESSAGE

otherwise they are discarded.

The `CurrentServiceState` of the Network Elements with at least one discarded alarm is put in `ALARM_SUPPRESSED`.

When FM reaches the `alarmOverloadProtectionLowerThreshold` the alert previously raised is acknowledged, all the Network Elements in `ALARM_SUPPRESSED` are resynchronized, all alarms and alerts are processed.

Where:

`alarmOverloadProtectionLowerThreshold` is the percentage of alarm protection threshold for ceasing alarm protection on the whole network.

6.1.5 Configuration of Alarm Type Severity

ENM operator can change the alarm type severity. This is done in two ways:

Severity change done on the nodes

ENM operator inputs a proper `cmedit` command, that is reflected in the remote nodes. Then, the affected alarms, as specified in the command, is sent by the nodes with the new severity.

Note: ENM does not apply any severity changes to such alarms.

Severity change done in ENM

ENM operator inputs a proper `fmedit` command, that is reflected locally to ENM. Then, the severity change is applied by ENM to the affected alarms, as specified in the command.

Note: The node does not apply any severity changes to the alarms, while such changes are applied by ENM.

6.1.5.1 Configuration of Alarm Type Severity on the Nodes

> The alarm type severity for FM on ECIM nodes is configured directly on the node. After a configuration change, the new alarm generated by the node has the new severity.

To modify severity of all alarms, every alarm has to be changed separately. Already-active alarms are not affected by the change. The aim of the feature is to read, configure, update (if already configured) and clear (unset) the severity provided by the alarming object. A typical scenario is to lower the severity for an alarm type caused by transient network disturbances. All new and updated alarms of this type are generated with the configured severity. Currently active alarms of this type are not affected. As a precondition, CM Supervision must be enabled, and the node must be synchronized in ENM. For more information, see *CM Supervision* in *ENM Product Description*. The configuration of the alarm type severity does not require FM supervision to be enabled. To receive alarms from the node with the new severity, FM Supervision must be enabled.

FmAlarmType Managed Object Attributes for Setting Alarm Severity

The ECIM FmAlarmType managed object includes the following attributes:

defaultSeverity This is a read-only attribute. It specifies the default severity of the alarm type. When set, the value indicates the severity for all alarms of this type. No value indicates that there is no fixed severity, and that the severity is decided by the alarming object, depending on the fault.

configuredSeverity This is a read-write attribute. It gives the possibility to configure the severity for a certain alarm type. When set, it overrides the severity provided by the alarming object. Setting this attribute impacts new and updated alarms only, it has no impact on existing alarms. When unsetting (clearing) this attribute, new alarms are generated using the severity value provided by the `defaultSeverity` attribute.

When the alarm type severity is set on the `defaultSeverity` or `configuredSeverity` attributes, the possible values are aligned with the `ERICSSON-ALARM-MIB` numbering as follows:

— CRITICAL (3),

— MAJOR (4),

— MINOR (5),

— WARNING (6)

Note: All the possible use cases (get, set, and update of the alarm type severity) are CM use cases, but they also affect the FM use cases on the alarm severity reporting. For further details, see the *ENM CLI tutorial* section of *online help*.

6.1.5.2 Configuration of Alarm Type Severity in ENM

The alarm type severity on nodes is configured in ENM. After a configuration change, for the new alarm generated by the node, ENM changes the alarm severity. The new alarm generated by the node has the new severity in ENM.

A typical scenario is to lower the severity for an alarm type caused by transient network disturbances. All new and updated alarms of this type are generated with the configured severity. Currently active alarms of this type are not affected.

With a `fmedit` command, through ENM CLI, it is possible to modify alarm type severity by specifying `neType`, `specificProblem`, and `perceivedSeverity`. For example, to modify severity of all alarms, every alarm type has to be changed separately.

In addition, it is possible to read and clear alarm type severity.

Note: All the possible use cases (get, set, and delete of the alarm severity) are FM use cases. For further details, see *ENM CLI tutorial* section of online help.

6.2 Fault Management Flows

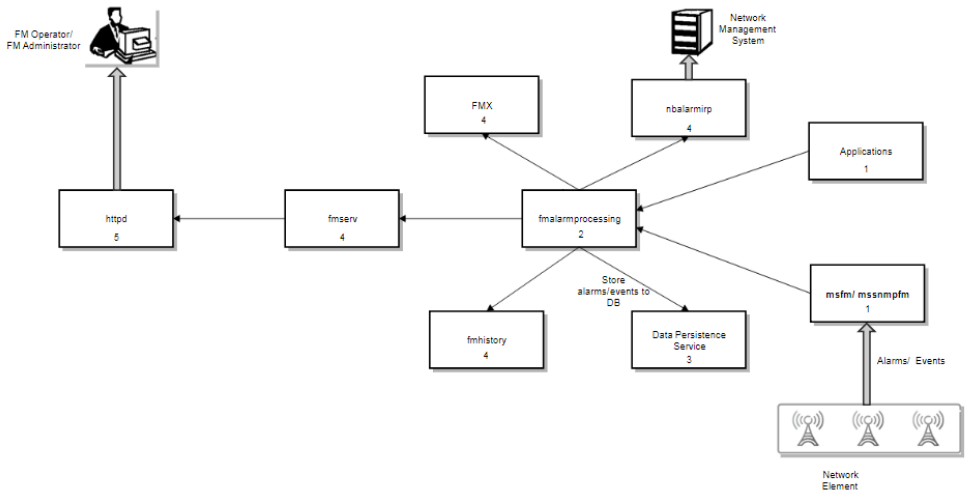


Figure 13 FM Alarm Flow

Flow Description:

— `Alarm Information` sent from the Network Elements is received by mediation components in FM. Mediation forwards the alarm information received to service component `fmalarmprocessing`.

— Other alarms (for example, ENM internal alarms generated by Internal Alarm Service) are received by service component `fmalarmprocessing`.

— Service component `fmalarmprocessing` processes, stores, and forwards the alarm to all the `fmalarmprocessing` subscribers like `fmserv`, `FM NBIs`, `FMX`, and `fmhistory`.

— Alarms, forwarded to NBI components, are further sent to NMSs.

— Alarms, forwarded to service component `fmserv`, are further displayed in presentation function FM Graphical User Interface.

— Alarms, forwarded to service component `fmhistory`, are further processed to `auto-acknowledge`, `Save to File`, or `Route to email`, if `Alarm Route` is configured.

Note: It is not possible to implement any FMX Rule based on the correlation attributes. For a description of correlation attributes, see *ENM Fault Management Interfaces Specification* (1/155 19-CRA 119 2176 available from Local Ericsson Support).

6.3 Fault Management Processing

> Mediation components receive alarm information from Network Elements. The alarm information represents a new alarm or new information about an existing alarm (for instance, clear alarm, or a change in severity). The mediation components then forward the alarm information to Service Layer.

`Alarm Processing` component in FM processes the alarms, and correlates them with existing open alarms present in Neo4j database. See Alarm Correlation on page 237.

When processing is completed, FM Service adds a new `OpenAlarm`, or updates an existing `OpenAlarm`, in the Neo4j database. A new `HistoryAlarm` is added into the SOLR database. The `FmFunction` in Neo4j is also updated with the status of the corresponding managed object when a notification that affects the node status is received. For instance, when heartbeat failure or node suspended are received.

HistoryAlarm and OpenAlarm

The `HistoryAlarm` table in SOLR database contains the list of new alarms, and alarm updates, received by FM Service Layer. It includes actions performed on alarms. Housekeeping functions are in place to keep the size of SOLR database under control. It does this by deleting the oldest records in database; the deletion policy can be tailored by the operator, by changing parameters in PIB.

The Neo4j database contains the list of alarms which are not acknowledged and not cleared. When an alarm occurs, the operator has to acknowledge the alarm to

show that it has been observed. Alarms that have not yet been cleared are called active alarms.

An alarm is removed from the `OpenAlarm` table when it has been cleared and acknowledged. The alarm can be cleared by the operator, by the node, or BNSI NBI. The alarm can be acknowledged by the operator, BNSI NBI, or automatically in ENM.

Error messages do not need to be cleared. When as an error message is acknowledged, it is removed from the `OpenAlarm` table, and removed from Neo4j database.

Presentation functions are notified about the alarms that have to be presented to the operator. The operator, through the presentation functions, can send a request back, for example: `acknowledgement` or `manual clearing`.

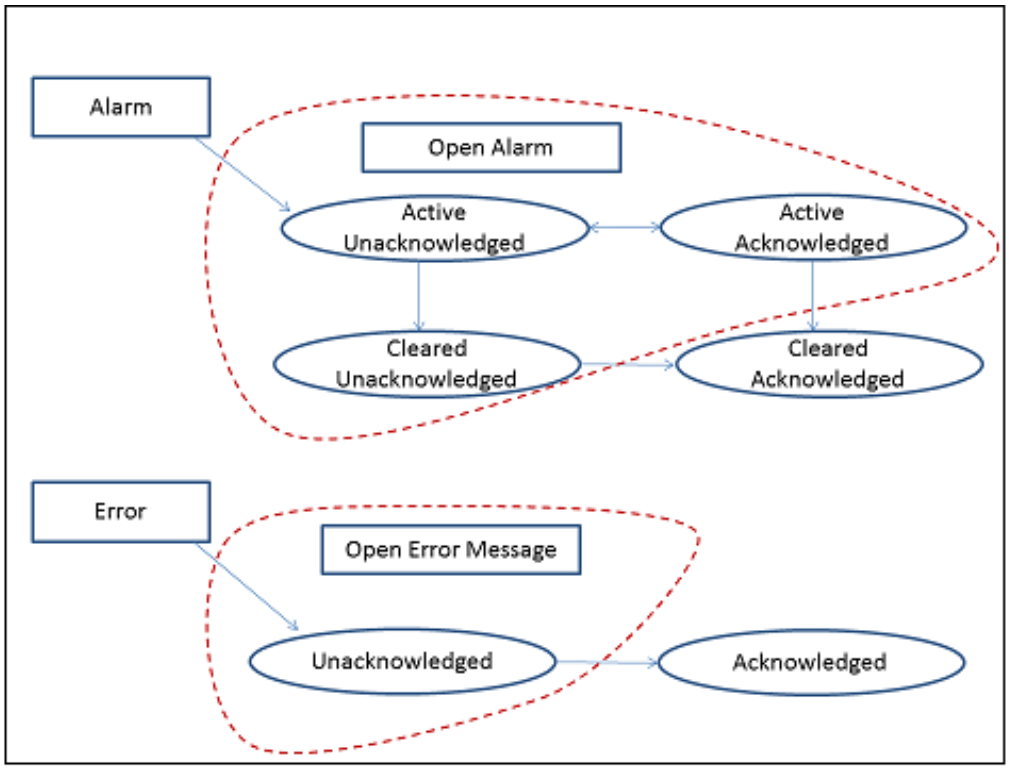


Figure 14 Alarms and Error Messages

6.3.1 Alarm Correlation

`Alarm Processor` correlates alarms, based on the alarm number attribute in each alarm record.

If an alarm number exists, the combination of alarm number and `Object Of Reference` (OOR) makes the alarm unique, and correlation is done based on that.

If an alarm number is not present, the correlation is done based on the combination of the following alarm record attributes:

— Object of Reference.

— Event Type.

— Probable Cause.

— Specific Problem.

6.3.2 Alarm Correlation Based on Additional Information

`Alarm Processor` elaborates additional information in each alarm record to extract correlation UUID and, if present, it correlates alarms.

An alarm correlation UUID is generated on a node when a primary alarm is raised. The same correlation UUID is included in any secondary alarms that are raised as a result of the primary alarm. Because of this, alarms that have the same root cause can be correlated, based on having a common alarm correlation UUID.

With the alarm correlation information, the operator can perform the following actions:

— Distinguish primary alarms from secondary alarms.

— Associate primary and secondary alarms that have the same alarm correlation UUID.

— Associate alarms that come from the same compute resource based on the common system UUID.

— Identify the role of the compute resource based on the compute name.

6.3.3 Alarm Clearing

A unique ID, `eventPoId` is assigned to an alarm when inserted in database by data persistence wrapper. If alarm is repeated, `Alarm Correlation` is performed against the alarms present in the database, and the `eventPoId` of the correlated alarm is retrieved to update with the newly received attributes. When an alarm clearing event, `Perceived Severity set to Cleared`, is received, FM searches the `Open Alarm` table to find the original alarm that is to be cleared.

When a `clear alarm` is received before its original alarm, it is added to `clear alarm cache`, and if the original alarm is not received within two minutes, ENM initiates an `alarm sync` on a node.

6.3.4 Repeated Alarm

An alarm or error notification might be a repetition of an active alarm, or an error. This happens when a new notification is received with the same attributes, or with an updated attribute. For example, indicating a change in the alarm severity, other than `Cleared`. FM searches the open alarm list to find the original alarm or error message that is repeated, and if found:

— The `Record Type` attribute is set to `REPEATED_ALARM`, `REPEATED_ERROR_MESSAGE`, or `REPEATED_NON_SYNCHABLE`.

— `EventTime` is updated.

— Any other changed attributes, if any, are updated.

— If parameter `updateInsertTime` is set to `TRUE`, then `Insert Time` is updated.

— The correlation between the original alarm or error, and the repeated alarm or error is performed as in Alarm Correlation part of Alarm Processing paragraph.

Note: The repeated alarm or error is sent to NMS, and saved in alarm history with the same `eventPOID` of the original one.

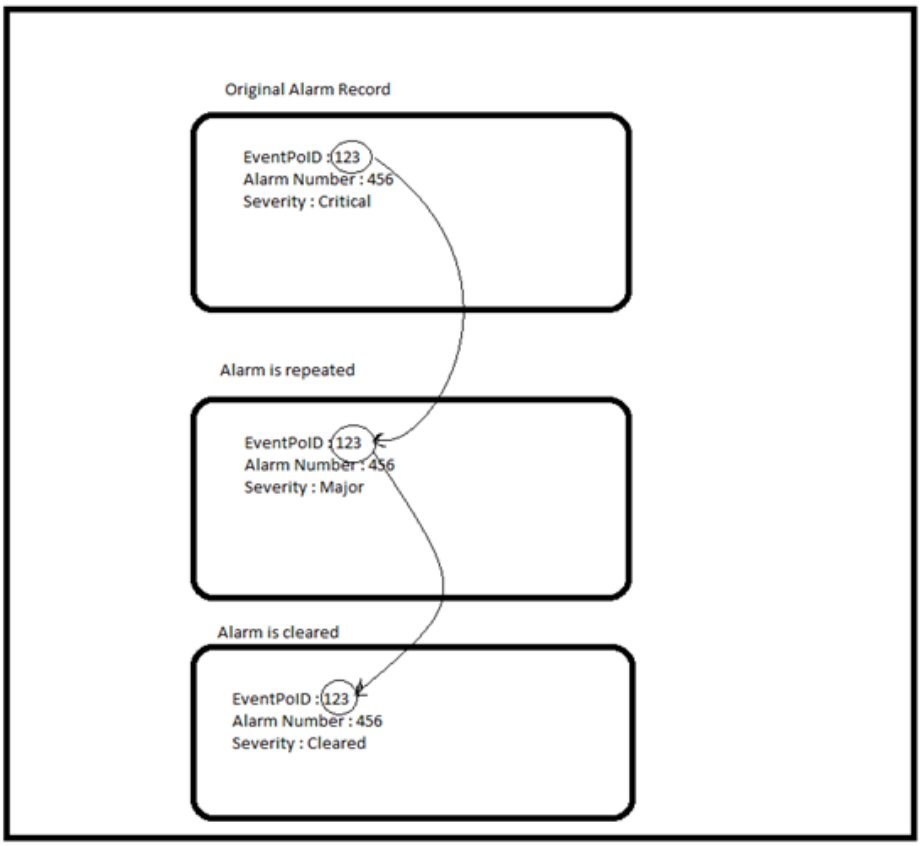


Figure 15 Examples of Correlated Alarm Record

6.3.5 Oscillating Alarm

Oscillating alarms occur when an alarm is sent and cleared, and then is sent again. This causes unnecessary alarms to appear in the alarm list. To reduce these alarms the `Service Layer` is configured to correlate against alarms that are cleared, and remove them from the alarm list. If a cleared alarm reoccurs, the original alarm is displayed.

This feature is enabled by default. To change this, modify the `oscillationAlarmCorrelation` PIB parameter. See *Enable and Disable Fault Management Oscillation-Correlation* in the ENM Monitoring System Administrator Guide 1/1543-AOM 901 151-2.

For new alarms that correlate against a cleared alarm, the `RecordType` is set to `REPEATED_ALARM`, and the Oscillation count is incremented.

6.3.6 Alarm Auto-Acknowledgement

Due to the number of alarms and errors from certain nodes, it is sometimes necessary to automate the acknowledgment of alarms and errors, to remove them from the FM system.

This function acknowledges ceased alarms and errors, after a configured time has elapsed since the alarm was ceased, or the error arrived. This gives the possibility to notice the alarm or error before it is removed. This function simplifies alarm synchronization from NMS, as the number of alarms or errors to receive is reduced.

This function is applied to all the managed network. It is enabled by default, and is governed by parameters in PIB. Please refer to *Enable and Disable Fault Management Delayed Acknowledgment Functionality* in the *ENM Monitoring System Administrator Guide*, 1/1543-AOM 901 151-2.

In addition, it is possible to automatically acknowledge the alarms and errors for identified nodes. This is done by creating an alarm rule in `Alarm Routing` application in ENM. Automatic Acknowledge can also be done by creating a specific rule in FMX.

6.3.7 Forward Unprocessed Alarm to NMS

Unprocessed Alarms are forwarded by ENM FM NBIs (SNMP, BNSI, CORBA) to subscribed NMSs. There are three scenarios where unprocessed alarms are forwarded to NMS:

— Database is unavailable:

  Based on DPS notification, FM maintains a Flag. It is set to `false` when the database is unavailable. In such cases, unprocessed alarms are forwarded to NMS.

— Read, Write, or both operations to database are failing:

  When the database is available, but read, write, or both operations to database are failing, each alarm is retried for 5 times, before forwarding the unprocessed alarms to NMS.

— Slow processing of alarms:

  If the database is available, and the alarm processing rate is less than the expected, unprocessed alarms are forwarded to NMS.

  In such cases, alarms are sent within a maximum period of 20 - 30 seconds.

Alarm synchronization is performed once every two minutes, on all the nodes for which the unprocessed alarms are sent.

Note: The unique `eventPoId` is updated when the alarm is persisted successfully. In the previous scenarios, as alarms are not persisted so unprocessed alarms are forwarded to NMS, with default `eventPoId` "**-2**".

6.3.8 Alarm from Unknown Node

When an alarm arrives to the `Fault Manager`, the name of the object is examined to verify if the alarm is sent from a Network Element that FM supervises. The complete name of node is called `Full Distinguished Name` (FDN).

If no Network Element exists in the database with this specific FDN, the alarm is discarded. A discarded alarm is not stored in any database.

6.3.9 Alarm Hashing

ENM FM application does not use the alarm number provided by APG (for MSC and HLR BC Nodes) to identify a cluster level alarm or blade level alarm for these BC nodes. It generates an internal alarm number, by applying a hashing function to the alarm printout content. `Specific Problem` (SP), `Probable Cause` (PC), `Event Time` (ET), `OOR`, and `Problem` text in `ALLIP` alarm printout are hashed for generating unique alarm number. This ENM FM generated alarm number is unique for the cluster and blade level alarms, and is not changed as long as the alarm is active. The ENM generated alarm number is used by the rest of ENM FM applications (if ENM FM is used for Fault Management) and it is sent north-bound to any other Network Management application if ENM FM is used as alarm mediation system. ENM FM application keeps the initially stored alarm time stamp where the alarm time-stamp received during an alarm synchronization activity is different to the one stored when the alarm was first received.

6.3.10 FMX Processing

Types of FM alarm processing with FMX subscriptions:

— `NormalProc`:

- When an incoming alarm is matched with a `NormalProc` subscription, it is in a hidden state by default.
- Alarm is persisted in the DB with visibility attribute value `false`, and then forwarded to FMX.
- The hidden alarm is not sent to the NorthBound (UI/NBI).
- The hidden alarm can not be acknowledged.

- The alarm is sent to NorthBound (UI/NBI) after the reception of a show request from FMX.
- If an alarm is in hidden state because it is matching with a NormalProc subscription, and the subscription is deactivated, further notifications for the same alarm are forwarded to NorthBound (UI/NBI).
- For repeated alarms, it is mandatory to add the relevant FMX Repeated Block into the Rule. Otherwise the visibility is set to `false` for repeated alarms, and remains `false` even if there are further notifications. Alarm is removed if a `clear` is received from node, and never forwarded to UI/NBI.
- The `Alarm Record Type` is changed from `SYNCHRONIZATION_ALARM` to `ALARM` when a synchronization alarm is matched to FMX `NormalProc` rule.

— `PostProc`:

- When an incoming alarm is matched with a `PostProc` subscription, it is in a show state by default.
- Alarm is persisted in the DB with visibility attribute value `true` and then forwarded to FMX.
- This show alarm is sent to the NorthBound (UI/NBI).
- This show alarm is `Auto-Acknowledged` if `Alarm Route` is enabled.

In the following use cases, the open alarm is be deleted from database:

— For stateless alarms: If it matches with a `NormalProc` subscription the alarm is directly deleted.

— When a `hide` request from FMX is received on a cleared alarm.

— When a `clear` is received from the Node or FMX on a hidden alarm.

There is heartbeat mechanism between FM and FMX. When FM determines that FMX is not available due to heartbeat failure, the following actions are performed:

— FM expects FMX to subscribe afresh once it is available.

— FM removes all the FMX subscriptions.

— FM sends these alarms to Northbound (ATR,UI and NBI).

— FM unhides all hidden alarms in open alarm database.

Processing of Alarms coming from FMX Event Simulator

Alarms coming from FMX Event Simulator against the Network Element are processed only if alarm supervision state of the Network Element is enabled.

If the Network Element created with dummy IP address for testing FMX generated events then it is expected to see heartbeat alarm for this Network Element.

Even if the node with dummy IP Address shows heartbeat failure alarm, it is possible to see FMX Generated alarms in FM UI for that node.

6.4 Fault Management Formats

Describes the FM data conversion and the different data formats.

Format Conversion

FM does the following data type conversions at different layers of solution in ENM. Different types of Network Elements generate alarm data in different formats.

FM Mediation translates different types of alarm from Network Elements into a standard `EventNotification` format before further processing.

The `EventNotification` requires conversion to a `ProcessedAlarmEvent` (presentation format) before being displayed by the FM Presentation Functions.

Converting Network Element Alarm to an EventNotification

The Alarm Mediation converts an alarm from a network element to the `EventNotification` format, using configuration data and translation rules as input. The `EventNotification` has a defined layout with attributes that are set according to network element-specific translation rules.

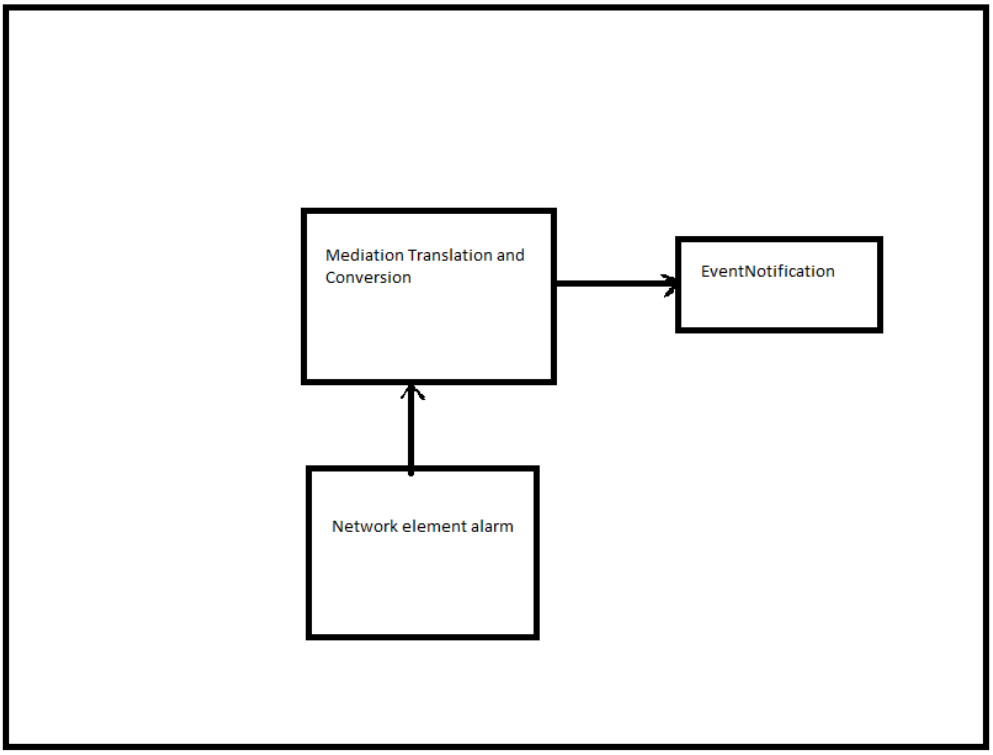


Figure 16 Converting Network Element Alarm Printouts to EventNotification Format

Converting EventNotification to Presentation Format

The Alarm Processor, one of the components of FM Service Layer converts the `EventNotification` to a `ProcessedAlarmEvent` format, inserts the record into the database, and sends the event to NBI Agents, Presentation, FMX, and Alarm Routing subscribers.

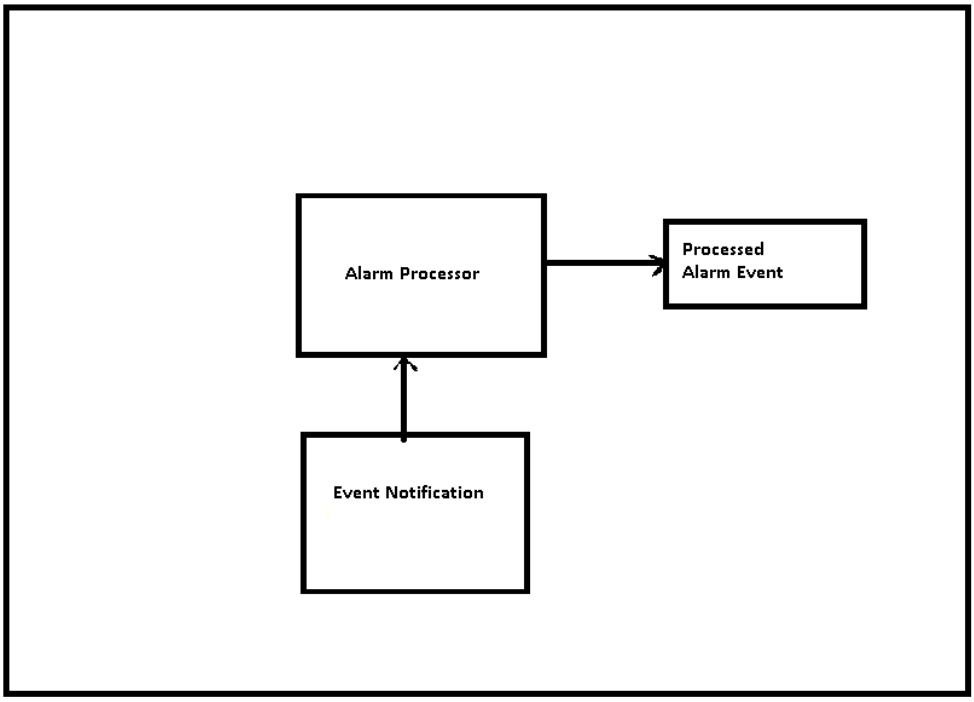


Figure 17 Converting EventNotification to Presentation Format

`Alarm Record` is a class used by FM components like AQS, UI, and CLI, to get alarm information from `OpenAlarm DB`. A more detailed description of the Alarm Record attributes follows:

EventType

This attribute specifies the type of the alarm. Each Alarm Manager sets this attribute in accordance with what is specified for the network element supervised.

EventTime

This attribute indicates the date and time when the alarm occurred in the network element.

RecordType

This attribute identifies the type of alarm. This is used to indicate if it is an alarm or an error message, or if the alarm came spontaneously or through a synchronization. There are also some record types which indicate special messages that are not really alarms. The attribute can have one of the following values for alarm messages as in Table 6.

ManagedObjectInstance

This attribute specifies the instance of the Managed Object, which represents the network element, issuing the alarm.

PerceivedSeverity

This attribute indicates the severity of the alarm. Possible values for this are:

— CRITICAL

— MAJOR

— MINOR

— INDETERMINATE

— WARNING

— CLEARED

When the Managed Object indicates a cessation of the alarm, the severity is set to `CLEARED`.

ProbableCause

This attribute gives a hint of the general problem causing the alarm. Each Alarm Manager sets this attribute in accordance with the specifications given for the network element supervised. It is specific for each type of network element.

SpecificProblem

`SpecificProblem` is used to give a further refinement to the cause of the alarm. Each `Alarm Manager` sets this attribute in accordance with the specifications given for the network element supervised.

Root

The attribute indicates the correlation type of the alarm. `NOT_APPLICABLE` means that the alarm does not support correlation UUID.

Possible values are:

— PRIMARY

— SECONDARY

— NOT_APPLICABLE

AdditionalAttributes

AdditionalAttributes holds information of various parameters as follows::

— `additionalText`

This attribute consists of a free-form textual description of the problem.

— `backedUpStatus`

If the alarm is of such a type that a backup object can take over from the failing Network Element, it is indicated using this attribute. If the switch to the backup object was successful, the attribute `Backup Object Instance` is filled in with the `Distinguished Name` of the backup object.

The attribute values are listed in the following table:

Table 4 backedup Status Attribute Values

Attribute	Backup Status
`0`	No success
1	Success

— `trendIndication`

This attribute is used for repeated alarms to indicate a change in the `PerceivedSeverity` attribute of a previous alarm.

The attribute values are listed in the following table.

Table 5 Trend Indication Attribute Values

Attribute Value	Trend Indication
`0`	lessSevere
1	noChange
2	moreSevere

— `generatedAlarmId`

If used, this number must be unique among active alarms for a particular object of reference. It can be used to identify an alarm for a clearing or for repeated alarms.

— `targetAdditionalInformation`

This attribute includes the correlation UUID (if any) and other target-specific info. For further details, see *Target Additional Information Format* in the ENM FM BNSI/SNMP/CORBA Northbound Interface Integration Programmers Guide.

Table 6 RecordType Attribute Values

RecordType	Description
ALARM	State-full alarms received in FM.
ALARM_SUPPRESSED_ALARM	An alarm indicating that no alarms except the critical alarms will be forwarded from the Network Element due to maintenance activity. Synchronization does not change this Record Type
CLEARALL	An error message raised in response to network element alarm list rebuilt notification and synchronization is initiated on that node.
ERROR_MESSAGE	Stateless alarms received in FM.
HEARTBEAT_ALARM	An alarm indicating a loss of connection with a node; the node is no longer responding or reachable from FM.
NODE_SUSPENDED	An alarm indicating that the alarm flow from node has been suspended because of a too high alarm rate.
NON_SYNCHABLE_ALARM	Alarm that is not provided by the node in alarm synchronization procedure.
OSCILLATORY_HB_ALARM	The repeated Heartbeat alarms when oscillation correlation is enabled.
OUT_OF_SYNC	An alarm indicating that ENM alarm List and Node alarm list are not in synchronization.
REPEATED_ALARM	A duplicate alarm received in FM is considered a repeated alarm. The alarm can increase or decrease in severity, change additional attributes, or stay unchanged. The possible change in severity is indicated by the trendIndication attribute.
REPEATED_ERROR_MESSAGE	An error message that is a repetition of a previous error message.
REPEATED_NON_SYNCHABLE	An alarm that is a repetition of a NON_SYNCHABLE_ALARM.
SYNCHRONIZATION_ABORTED	An error message indicating the abort of an alarm synchronization. The alarms received during the synchronization are logged, but no clearing of alarms is generated.
SYNCHRONIZATION_ALARM	An alarm that has been sent from the list of currently active alarms at node during a synchronization.
TECHNICIAN_PRESENT	An alarm indicating that field technician is present at the Network Element to perform maintenance activity. Synchronization does not change this Record Type.
UNKNOWN_RECORD_TYPE	Alarm or error message that does not match with any of the other RecordTypes.
UPDATE	A message indicating that an extra attribute has been created or changed.

6.4.1 EventNotification

> The EventNotification contains the adapted alarm information, some of which comes from the managed object and some which is set on its way through the system by the Mediation.

The following table lists the `EventNotification` alarm attributes:

Table 7 EventNotification Attributes

Attribute	Data Type	Description
EventType	String	Type of the event (alarm).
EventTime	String	Time when the alarm occurred in the object.
RecordType	String	Type of the Alarm Record.
ManagedObjectInstance	String	Distinguished Name of the object generating the alarm.
PerceivedSeverity	String	Severity of the alarm.
ProbableCause	String	General problem causing the alarm.
SpecificProblem	String	A refinement of the ProbableCauseattribute.
AdditionalAttributes	map<String,String>	Container for additional information (additional attributes) that may be passed between different parts of FM.
EventAgentId	String	
TimeZone	String	Time Zone of the node.
ExternalEventId	String	
SourceType	String	
TranslateResult	String	
ProcessingType	String	Indication of how alarm should be processed for FMX.
FmxGenerated	String	Indication if alarm was generated by FMX.
DiscriminatorIdList	List<String>	
Visibility	boolean	Show or hide the Alarm Record in Alarm Monitor.
Operator	String	
Acknowledged	boolean	
AckTime	String	Time when alarm is acknowledged.

6.4.2 ProcessedAlarmEvent

> The ProcessedAlarmEvent contains the adapted alarm information, some of which comes from the managed object and some which is set on its way through the system by the FM Service Layer.

The table below lists the `ProcessedAlarmEvent` attributes.

Table 8 ProcessedAlarmEvent Attributes

Attribute	Data Type	Description
eventPoId	Long	A unique identifier for the event (alarm).
eventType	String	Type of the event (alarm).
insertTime	Date	Time when the alarm is inserted in database.

Attribute	Data Type	Description
eventTime	Date	Time when the alarm occurred in the object.
lastUpdatedTime	Date	Time when the alarm is lastUpdated in the database.
recordType	FMProcessedEventType	Type of the Alarm Record.
objectOfReference	String	Distinguished Name of the object generating the alarm.
presentSeverity	ProcessedEventSeverity	Severity of the alarm.
probableCause	String	General problem causing the alarm.
specificProblem	String	A refinement of the ProbableCauseattribute.
alarmNumber	long	A number to uniquely identify active alarms.
backUpObjectInstance	String	Distinguished Name of an object that backs up the object generating this alarm.
backUpStatus	boolean	Status of the backup object.
trendIndication	ProcessedEventTrendIndication	For a repeated alarm, an indication of an increase or decrease in the severity, or no change.
previousSeverity	ProcessedEventSeverity	Previous PerceivedSeverity, if changed.
proposedRepairAction	String	Action to take to fix this alarm.
alarmState	ProcessedEventState	For internal use by FM.
ceaseTime	Date	Time when the clear alarm is generated.
commentText	String	Text containing the comment.
problemText	String	Text describing the alarm.
correlatedPoId	long	EventPoId of correlated alarm .
problemDetail	String	Additional information about the alarm.
ceaseOperator	String	Information about the operator performed cease operation on the alarm.
fdn	String	Full Distinguished Name of corresponding node.
lastAlarmOperation	ProcessedLastAlarmOperation	Indicated the operation last performed on the alarm.
additionalInformation	map<String,String>	Container for additional information (additional attributes) that may be passed between different parts of FM.
timeZone	String	Time Zone of the node.
processingType	String	Indication of how alarm should be processed for FMX.
fmxGenerated	String	Indication if alarm was generated by FMX.
discriminatorIdList	List<String>	
visibility	boolean	Show or hide the Alarm Record in Alarm Monitor.
ackOperator	String	Information about the operator performed acknowledge operation on the alarm.
ackTime	Date	Time when the alarm is acknowledged.
managedObject	String	
alarmingObject	String	
alarmId	Long	
historyPOId	Long	
correlatedVisibility	Boolean	

Note: If the acknowledge action is performed directly on the Node for NeTypes CCN-TSP,CSCF-TSP,HSS-FE-TSP,MTAS-TSP,VPN-TSP,cSAPC-TSP, the relevant nodeAcknowledgeTime and nodeAcknowledger are reported as additional information and they aren't reflected into the Ack and Acktime attributes.

7 Automatic Alarm Handling (FMX) Overview

Automatic Alarm Handling (FMX) applications are provided in ENM in the form of two Kernel Virtual Machines (KVMs).

The KVMs run Red Hat Enterprise Linux. Each KVM has a private IP address. There is no direct way to log on from outside ENM. Users must log in first into ENM using SSH and then connect to either of the FMX KVMs using an SSH or SFTP client. There is no direct way to log on to the FMX KVMs from the UI. Access to the FMX service requires that the user is assigned one of the POSIX user roles.

7.1 OSS-RC FMX vs ENM FMX: Compatibility and Delta functionality notes

Presents the delta notes for users migrating and managing Rule Modules from legacy systems (OSSRC) into ENM.

Module Management

The OSS-RC FMX Module Admin has been redesigned using the common UI SDK as a web app. Now it is called FMX Module Management UI. FMX Module Management UI includes Module Management, Module Parameters, Module Triggers, Module Statistics and Event Simulator apps.

Activation of FMX rule modules is changed in the 17A release of FMX

FMX allows a user to activate an FMX Rule Module either on the whole network or on a part of the network. To activate an FMX Rule Module on a part of the network, the user selects the desired part of the network. FMX then decides, for each incoming alarm, if the Rule Module should be triggered on that alarm. In the ENM 16B release, the module activation was distinguished by the alarm's ObjectOfReference attribute. In the ENM 17A release, the FDN alarm attribute is used instead. The FDN alarm attribute is used to determine if there is any FMX Rule Module active which should process the alarm.

Any FMX rule module which is activated before an upgrade will no longer be activated for the same objects (or network elements) after the upgrade. The module will act as if it was not activated. To remedy this situation, the FMX rule module must be deactivated, and the activate-for action must be re-done by selecting the desired FDNs.

7.1.1 Modules Compatibility

Rule Modules

ENM FMX supports backwards compatibility within OSS-RC FMX rule modules as long as those rules are using building blocks from FMX product standard utility modules.

Rule modules from OSS-RC FMX can be uploaded and imported into ENM FMX. After imported, the rule module can be managed as usual from Module Management UI or modified from Rule Editor UI.

Note: If ENM is not using the same platform as OSS-RC. Therefore, any rule using custom scripts or commands might not work because of Operating System, OSS, or 3PP dependencies. For example, migration rules using specific OSS-RC commands or Perl scripts with module dependencies need to be studied as part of LCM.

Utility Modules

Utility Modules from OSS-RC FMX cannot be imported into ENM FMX.

All the functionality and blocks provided by standard Utility Modules delivered as part of OSS-RC FMX product are available as well in the Utility Modules delivered as part of ENM FMX product.

Non-Standard Utility Modules for interaction with other entities (nodes, data sources, management systems) can be delivered on request.

7.1.2 Rule Engine

SourceType Vs SystemType

The OSS-RC attribute `SystemType` is deprecated in ENM. When importing a legacy Rule Module, FMX translates the `SystemType` attribute values into `SourceType` attribute values. The translations are specified in the following table.

The translation ensures that imported legacy Rule Modules are still triggered on the appropriate network element alarm after a migration from a legacy system. However, it is possible that the translation results in "false positive" hits, that is, the rules trigger on alarms from Network Elements in ENM that they were not triggering on, in the legacy system. When migrating rule modules, the rule developer must check the resulting `SourceType` values. Edit the affected trigger blocks in the FMX Rule Editor, and only keep the values that are applicable.

SystemType (Legacy)	SourceType (ENM)	Notes
`LRAN`	RadioTNode, ERBS, RadioNode, MSRBS_V1	(1)
`WPP`	SGSN-MME	

SystemType (Legacy)	SourceType (ENM)	Notes
COM	RadioTNode, RadioNode, MSRBS_V1, SAPC	(1)
URAN	RNC, RBS	(1)
CELLO	MGW	
JUNIPER	EPG, vEPG	(1)
SSR	EPG, vEPG	(1)
JAMBALA	SAPC	
ENIQ	RadioTNode, ERBS, RadioNode, MSRBS_V1, RNC, RBS	(1), ENIQ sends alarms with the SourceType of the original network element, that is, the one where a threshold has been crossed.
AXE	BSC, MSC-BC-BSP, MSC-BC-IS, MSC-S DB, MSC-DB-BSP, vMSC, vMSC-HC, IP-STP, IP-STP-BSP, HLR-FE (APZ), HLR-FE (BSP), vHLR-FE, HLR-FE (IS)	(1)

(1) The listed neTypes are all set, as a list, in the SourceType attribute on the trigger block within the rules.

7.1.3 Release Changes

> In the ENM 16B release, the `SourceType` attribute on each alarm is set by ENM alarm mediation.

In the ENM 17A release, the `SourceType` attribute on each alarm is set centrally by FM, and always matches the `neType` of the network element that sent the alarm. For virtual nodes, the `SourceType` is set to the `TargetType` of the Network Element.

Notable changes:

— eNodeB G1 uses SourceType = ERBS.

— eNodeB G2 uses SourceType = RadioNode.

Record Type

Record Type is using predefined strings instead of number codes in OSS-RC. See FM documentation for a completed list of supported Record Types.

replace-block vs update-block

In OSS-RC FMX there are two different blocks, `Update-alarm` and `Replace-alarm`. On import to ENM, they are upgraded to ENM `Update-Alarm` block. They have different settings depending on the type of block in OSS:

— `Update-alarm` block has in ENM the default setting `Display Status=Original`.

— `Replace-alarm` block has in ENM the default setting `Display Status=Show`.

The migrate rule behaves in the same way as in the legacy system, but the display status can be modified.

event-used-in-other-module block

The FMX block `event-used-in-other-module` in OSS-RC is a part of a utility module called `event-used-in-other-module`. This block is included in `baseblocks-module`. This might result in a dependency issue when loading legacy rule modules.

To resolve dependency issues, follow these steps:

1. Create a rule module called `event-used-in-other-module` and check it in. It can be empty, but must exist on the system.
2. Load the legacy rule module, create a version, then check in the new version.
3. Now the dependency to `event-exists-in-other-module` is gone.

7.1.4 Rule Editor

> The Rule Editor application has not been ported to HTML using the UI SDK and remains as a thick java client application. It is launched using the ENM remote desktop server.

The user rule modules are stored in the user space under `$HOME/.fmx/` (shared file system). Loaded modules are stored in `/ericsson/tor/data/fmx/modules`. This folder is referred within the FMX nodes by using the variable `$FMX_HOME`.

Note: The `/ericsson/tor/data/fmx/` is in the Shared File System, and accessible from all the ENM nodes. Within the FMX nodes, this folder is linked as `/var/opt/ericsson/fmx/`.

7.1.5 Rule Trace

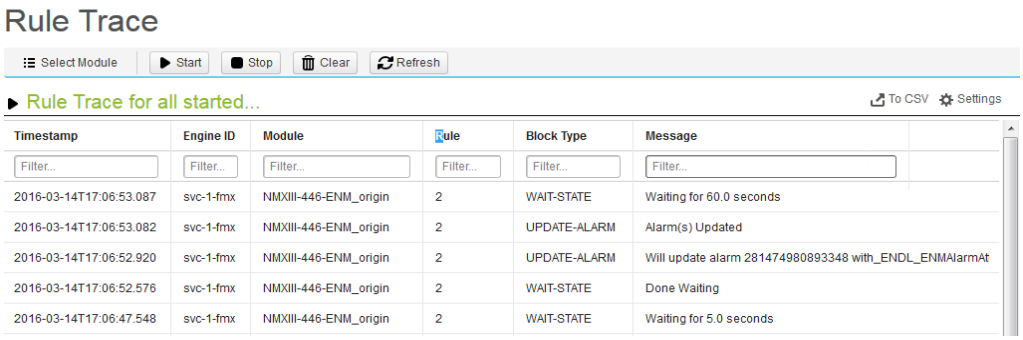
> Because of both replacing CORBA communication with MQ, and the new distributed FMX engine architecture, Rule Trace works differently.

In OSS-RC FMX, the Rule Trace pane within the Rule Editor "tailed" the trace text file in the local file system. That is not possible any longer because of the distributed architecture. There are multiple FMX nodes, and these are stateless (can be reinstalled/upgraded at any time). In ENM FMX, the Rule Trace apps, both in Rule Editor pane, and the stand-alone Rule Trace UI web app, "subscribe" to the rule trace messages channel in `RabbitMQ`, running as cluster in the FMX nodes.

The option of "save trace to file" has been deprecated from the Rule Editor pane. If you use the **Rule Trace** UI, there is an option to save the trace as a CSV file.

The Rule Trace log is in the local file system of any of the FMX servers: `/var/log/fmx/fmxie_ruletrace.log`.

To change configuration for Rule Trace log file destination folder, log level, and retention policies, see *System Administrator Guide*.



Module Repository

The module repository for Rule Modules version control is based on `git`, providing finer granularity for change control, and better accountability. The Rule Editor and Module Management operations, load, unload, check in, check out, archive, and so on, hide the internal repository management complexity from the end user.

7.1.6 Actions

AMOS (CPP)

In OSS-RC FMX rules, it is frequent the use of amos (Advanced Moshell) commands within the rules to interact with LTE and WCDMA nodes.

This can be done within the `Action&Response` block or with the specialized blocks provided by the cpp utility module. ENM provides a new version of the `cpp-block-module` utility module with same functionality plus the possibility to write custom amos commands within the block.

Configuration Server (cstest)

In OSS-RC FMX rules, the unsupported `cstest` command is frequently used within rules, by using the `Action&Response` block. This command fetches node data from the OSS-RC Configuration Server.

ENM provides a more powerful tool: the ENM CLI. The ENM CLI functionality is available from the web UI and a python module is available for building custom scripts.

Interaction with ENM CLI can be done within the FMX rules in two different ways:

— If the rule developer wants to use same command syntax as with ENM CLI web UI → use the ENM CLI block, provided by the `enmcli-blocks-module` utility module.

— If the rule developer wants to use custom scripts (for example, using the ENM CLI python module) → use the Command Action & Response block.

Create a specific system user for using the ENM CLI, then assign the relevant roles depending on the tasks required of that user. See *Create System User for Automatic Alarm Handling (FMX) Interaction with ENM CLI* in *ENM System Administrator Guide*.

Time Periods

The `Time Periods` format used in OSS-RC FMX has been deprecated.

Instead, `Time Periods` uses the iCalendar standard (https://icalendar.org/RFC-Specifications/iCalendar-RFC-5545/), an open standard for exchanging calendar and scheduling information between users and computers, from based rule triggering based and time conditions (timeperiods). The iCalendar standard promotes the filename extension of `.ics` to be used for files containing calendaring and scheduling information. This extension is supported by FMX Module Triggers application.

An FMX Time Period container is a iCalender `ics` file. FMX supports only the basic time event (VEVENT). `VEVENT` describes an event, which has a scheduled amount of time on a calendar. Such events have a `DTSTART` which sets a starting time, and a `DTEND` which sets an ending time. If the calendar event is recurring, `DTSTART` sets up the start of the first event.

See https://tools.ietf.org/html/rfc5545 for more information on iCalendar standard.

If importing a rule with Time Periods blocks from OSS-RC into ENM, verify the Time Periods blocks configuration. The needed Time Periods must be defined from Time Period application.

7.1.7 Module Statistics

In OSS-RC, `FMX Module Statistics` report all the `Blocks` present in the Rules, regardless if they have been processed by the `Rule Engine` or not (count equals to `0`).

In ENM, `Trigger Blocks` are always reported, while other `Block` types are reported if processed by the `Rule Engine` (count greater than `0`).

7.1.8 Other

Obsolete Features

Preprocessing

In previous FMX on legacy systems, there was a `Preprocessing` option, that has been deprecated in ENM. In Preprocessing, the `Alarm Records` are sent from the `Alarm Managers` directly to FMX instead of to the FM Kernel. This was useful for Alarm Managers handling high volume of alarm-related data from the network, where FMX processing flexibility is used to decide which data can be forwarded to the FM Kernel as Alarm Records. Preprocessed alarms that are not presented to the operator, are not logged in the FM Alarm Log database. This use case is no longer needed in ENM. See *Function Specification* for description of other types of processing.

Fmxcli

The following `fmxcli` commands have been deprecated:

— `addobj`. Add objects a module is activated for.

— `removeobj`. Remove objects a module is activated for.

EDD

The Event Discriminator Descriptor (EDD) file containing the list of FMX currently active triggers, and their configuration details, has been deprecated. Instead, the Module Triggers application provides the same information.

Functions

The following rule engine function is not supported in ENM:

— #clear_im_object_recursive

Not Implemented Legacy Features

The following legacy features are planned, but not yet ported to ENM:

— Debugging of rule execution in Rule Editor.

— Simulation of action and response blocks execution.

8 Scripting Support Overview

ENM provides a scripting service in the form of two Kernel Virtual Machines (KVMs) allocated to allow the use of scripts.

The scripts run interactively and are scheduled through a cron service. The KVMs run Red Hat Enterprise Linux, a POSIX-compliant scripting environment. During upgrade scenarios, the scripting service availability is limited. The scripting KVMs are restarted sequentially, ending any sessions that are currently active. Cron jobs scheduled on a VM during the restart are not run. Each KVM has a public IP address that allows incoming connections. Users choose one of the KVMs, and connect using an SSH or SFTP client. There is no direct way to log on to the scripting KVMs from the UI. To access to the scripting service, the user has to be assigned one of the POSIX user roles.

8.1 Log on to a Scripting KVM

Steps

— To log on to the scripting KVMs, choose a machine, and use its public address as the `hostname` parameter. The `username` and `password` are the `ENM username` and `ENM password`.

Note: SSH uses the ENM `username` and `password`. It is case-sensitive: use the same case used to create the ENM `username` in the scripting service environment.

Example

A user, created as `ScriptingOp1`, logging on to a scripting KVM named `scp-2-scripting.enm1234.ericsson.se` from the command line of a different Linux machine, the command is:

```
another.linux.machine$ ssh ScriptingOp1@scp-2-scripting.enm1234.ericsson.se
        Password:
        scp-2-scripting$
```

The exact procedure to log on depends on the SSH/SFTP client used.

Certain usernames are not available for POSIX roles, as they are reserved for use by the RHEL system or services. These names are outlined in the *User Management* section of the *ENM Identity and Access Management System Administrator Guide*, 2/1543-AOM 901 151-1.

When the scripting KVMs are recreated, for example, during upgrade, the SSL certificates are regenerated. This can cause the SSH client to generate warning messages, which can be ignored.

8.2 Running Scripts

The scripting VMs run Advanced MO Shell (AMOS) scripts, either interactively or through the cron service.

Other Scripts

PM Decoding and Client Scripting use the scripting service. Refer to ENM documentation to learn more about these products.

Prerequisites

AMOS scripts are programmed according to the guidelines in the *MOShell User Guide*, available in the scripting cluster through the following path: `/opt/ericsson/amos/moshell/UserGuide.pdf`

Steps

1. Start AMOS by typing `amos <nodename>` or `amos <node IP>` at the command prompt.
2. Exit AMOS by typing **`quit`**, or use **`Ctrl-D`**.

   `Ctrl-Z` does not recover the resources used, and its use is not recommended.

8.3 Reschedule Failover Cron Services

To schedule jobs in failover mode, create the job in all Scripting VMs. Configure the job using the '`prefer <HOSTNAME>`' command, and then the actual job to run.

Although the job only runs on the preferred VM, it has to exist on the other VMs for failing over to them if the preferred VM goes down.

If the '`prefer <HOSTNAME>`' is invalid, the job does not run on any of the VMs, and the error is logged in the `/var/log/messages` directory.

Error example:

```
Feb 24 00:55:02 scp-1-scripting CRON_FAILOVER: ERROR ( /usr/bin/prefer ): Invali →
d preferred host, scheduled cron job won't be run
```

The jobs running at the time of the failing over do not rerun on the newly selected VMs in the cluster.

Known Limitations

All the limitations for the normal Cron Scheduling task apply to the `failover`.

Prerequisites

— Knowledge of Linux operation and Cron schedule.

— Scripting_Operator role or Cron Capability.

— A plan for distributing cron load among the available scripting KVMs.

Steps

1. Follow the instructions in the *Connect to a Virtual Machine* section, in the *ENM System Administrator Guide*, then log on to the scripting VM.
2. Connect to the scripting VM as an administrator.

   In this example, the username is `administrator`.

   ```
   ssh administrator@<scripting VM hostname>
   administrator@<scripting VM hostname> password:
   ```

3. Create a Cron job.

   ```
   [administrator@scp-1-scripting ~]$ crontab -e
   ```

   Entering `crontab -e` opens `vi` editor to allow the user to edit cron entries.

   Example

   ```
   [administrator@scp-1-scripting ~]$ crontab -l
   * * * * * logger "The local job is running on this scp-1-scripting"
   * * * * * prefer scp-1-scripting logger "The failover job is running on this →
    vm"
   ```

   In this example, we are logging a message every minute.

4. When editing is complete, type `:wq` to save and quit.

   ```
   [administrator@scp-1-scripting ~]$ crontab -e
   crontab: installing new crontab
   administrator@scp-1-scripting ~]$
   ```

Results

These steps schedule the cron job in the local VM automatically. It saves a copy of that cron file on the SFS, under `/ericsson/vmcrons/$HOSTNAME/` to be able to recover it after upgrade or if there is a failure in the VM.

8.4 Load Balancing for AMOS and General Scripting VMs

A load balanced port is available for both scripting and AMOS interactive SSH connections.

— Amos VMs can be accessed through port 5022:

  Using command as:

  ```
  ssh -p 5022 username@enm-haproxy-address
  ```

— Scripting VMs can be accessed through port 5023:

  Using command as:

  ```
  ssh -p 5023 username@enm-haproxy-address
  ```

— When the load balancing selects a new VM, a certificate warning may be produced.

  This warning can be ignored.

  When a VM switch happens, the terminal client may popup an alert warning. Ignore this alert, and click **Yes** to remain connected.

— To disable this alert warning, refer to the *ENM Installation Instructions, 5/1531-AOM 901 151*, Section: 'Prevent an SSH Key Mismatch Issue after ENM Initial Install'.

8.5 Automated House Keeping on All Scripting VMs

> The purpose of the cleanup script is to remove old unneeded files from certain directories on all scripting VMs (AMOS, Element Manager, and Scripting). This script is triggered every hour using cron.

Criteria for Deletion

1. File was not accessed or modified for the last two days.
2. Created by ENM user.

3. File exists in one of these folders:

    a. `/tmp`

    b. `/var/tmp`

    c. `/usr/tmp`

Note: Files created by root user are not deleted.

Useful Info

— Script location: `/ericsson/ERICpamopenam_CXP9032008/scripts/tmpcleanup.sh`

— Triggering Frequency: Hourly

8.6 Copy and Paste Long Lines in AMOS and General Scripting Shell Terminal

This section illustrates the supported behavior of copy and paste functionality in AMOS or Scripting UI shell terminal.

— Allows users to copy a long command (which has multiple lines) from either the terminal or an external editor and execute it as a single command. If the command is to be executed as a single command, then the user must use two exclamations (!!) before pasting the command into AMOS or Scripting shell terminal.

— Allows users to copy and paste multiple lines commands directly from the terminal or an external text editor into AMOS or Scripting shell terminal. If the commands are to be executed as individual commands, then do not use two exclamations (!!) but copy and paste them into terminal.

Note: A combination of the preceding two scenarios is not supported.

9 Automatic Link Discovery (ALD)

Beginning with ENM 19.4, this functionality provides automatic discovery of links between all NEs supported by `Transport Common Information Model Normalization`.

9.1 Automatic Discovery for Radio Link

MINI-LINK node model for Radio interfaces includes information about the remote terminal-id. This allows matching interfaces on the two nodes to be found. A link is created if the relevant attributes are properly configured.

They are:

— carrier-id

— node-name

— terminal-id

— far-end-node-name

— far-end- terminal-id

The attribute `carrier-id` must be the first attribute configured, otherwise no radio links are discovered.

Example

To discover a Radio Link between NodeA and NodeZ the parameters must have the following values:

NodeA	NodeZ
Interface=CT-1/3/1, carrier-id=1	Interface=CT-1/3/1, carrier-id=1
Interface=CT-1/3/2, carrier-id=5	Interface=CT-1/3/2, carrier-id=5
node-name: MiniLink1	node-name: MiniLink2
terminal-id: `001`	terminal-id: `002`
far-end-node-name: MiniLink2	far-end-node-name: MiniLink1
far-end- terminal-id: `002`	far-end- terminal-id: `001`

9.2 Automatic Discovery for Ethernet Link

When LLDP information is managed by node models, and mirrored into ENM DPS, the TCIM updates the relevant attributes, and automatic discovery of Ethernet links is allowed.

The following parameters must be set in a proper way, and in the following sequence, to allow the automatic discovery of Ethernet links:

1. local-port-id
2. remote-port-id
3. remote-sys-name
4. lldp- sys-name

Example

To discover a Radio Link between NodeA and NodeZ, the previous parameters must have the following values:

NodeA	NodeZ
lldp-sys-name: MiniLink1	lldp-sys-name: MiniLink2
local-port-id: 001	terminal-id: 002
remote-sys-name: MiniLink2	remote-sys-name: MiniLink1
remote-port-id: 002	remote-port-id: 001

Ethernet Links on WAN ports are discovered only if the four radio links are already created and discovered.