

Prepare Upgrade Package

OPERATING INSTRUCTIONS

Copyright

© Ericsson AB 2014, 2015. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

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

Trademark List

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



Contents

| | | |
|----------|---------------------|----------|
| 1 | Introduction | 1 |
| 1.1 | Prerequisites | 1 |
| 2 | Procedure | 3 |





1 Introduction

This document describes the preparation phase of a software upgrade.

A software upgrade has a limited time window during which some planned service impact is allowed. The preparation phase has no service impact. It is therefore recommended to perform it outside the upgrade time window during normal “traffic hours”. Use the upgrade time window only for the execution phase of the software upgrade. The execution phase is described in *Upgrade Using One-Step Activation* and *Upgrade Using Step-by-Step Activation*.

This preparation phase consists of the following:

- Creates the upgrade package, by making the upgrade package visible in the Managed Object Model.
- Prepares the upgrade package, by performing the relevant software upgrade package extraction activities or integrity checks. The integrity check performs checksum checks or equivalent (oblivious hashing, check and guard system, active or passive tamper resistance) and is needed to secure that the correct upgrade package has been fetched.
- Verifies the Management Element (ME) ability to activate the upgrade package.

The procedure in this document is illustrated by an example where a software version with product name `ERIC-COREMW_RUNTIME`, product number `CXP9020355_1`, and product revision `R7E01` is running in the system. A software upgrade package `ERIC_UP-CXP9020355_1-R7F01`, which is designed to upgrade this software version to product revision `R7F01`, is going through the preparation phase.

1.1 Prerequisites

This section describes the prerequisites, which must be fulfilled before using the procedure.

1.1.1 Conditions

The following conditions must apply:

- The ME has passed a health check routine.
- The uncompressed software upgrade package file is present under a known directory on a remote repository. In this document, `/node/software/upgrade/R7F01` is used as a directory example.



- The required SSH File Transfer Protocol (SFTP) user and password are known.
- An Ericsson Command-Line Interface (ECLI) session in Exec mode is in progress.
- There are no black patches on the site.
- A manual backup must be done before the upgrade.



2 Procedure

To prepare an upgrade package:

1. Navigate to the *SwM* Managed Object (MO), for example:

```
>dn ManagedElement=NODE06ST, SystemFunctions=1, SwM=1
```

2. Create an upgrade package MO by importing the software upgrade package file from a remote repository, for example:

```
(SwM=1) >createUpgradePackage --uri sftp://user@192.0.2.10://node/software/upgrade/R7F01 --password 26538813
```

The system returns `invocation Id (actionId)`, which is 2 in this example. This indicates a successful operation. If the operation is unsuccessful, the system returns 0. The returned `actionId` is also presented in the associated `reportProgress` structure as an attribute.

2

3. Verify the result for a successfully triggered operation:

```
(SwM=1) >show -v
```

The following example output shows the final result. If `state=RUNNING`, the creation is not yet completed. Result `result=FAILURE` means that the creation failed.

```
SwM=1
[...]
  reportProgress <read-only>
    actionId=2 <read-only>
  [...]
    progressPercentage=100 <read-only>
    result=SUCCESS <read-only>
    state=FINISHED <read-only>
  [...]
```

4. Navigate to the *UpgradePackage* MO:

```
(SwM=1) >UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
```

5. Prepare the upgrade package:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >prepare
```

The system returns output `true` or `false`.

6. Verify the preparation:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >show -v
```



The following is an example output, which shows the final result. If a transient result is shown, where `state=RUNNING`, the operation is not yet completed.

```
UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
  activationFallbackTimer=600 <read-only>
[...]
  state=PREPARE_COMPLETED <read-only>
[...]
  reportProgress <read-only>
    actionId=5 <read-only>
    actionName="Prepare" <read-only>
    additionalInfo[]
      "" <read-only>
    progressInfo="Prepare UpgradePackage" <read-only>
    progressPercentage=100 <read-only>
    result=SUCCESS <read-only>
    resultInfo="" <read-only>
    state=FINISHED <read-only>
    step=1 <read-only>
    stepProgressPercentage=0 <read-only>
    timeActionCompleted="2013-12-18T04:43:08" <read-only>
    timeActionStarted="2013-12-18T04:43:03" <read-only>
    timeOfLastStatusUpdate="2013-12-18T04:43:08" <read-only>
```

7. Verify the upgrade package:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >verify
```

The system returns output `true` or `false`.

8. Verify the result:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >show -v
```

The following is an example output:

```
UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
[...]
  state=PREPARE_COMPLETED
[...]
  reportProgress
    actionName="Verify"
    result=SUCCESS
    state=FINISHED
[...]
```

This output shows the final result. If `state=RUNNING` is shown, the operation is not yet completed. Result `result=FAILURE` means that the creation failed.