

Execute Health Check Job Providing Export URI

OPERATING INSTRUCTIONS

Copyright

© Ericsson AB 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



Execute Health Check Job Providing Export URI



1 Introduction

This document describes how to manually execute a health check job previously created in the Managed Object Model (MOM) storing the related report file in a local or remote user-defined location. After successful job execution, the Managed Element (ME) health status is provided in the MOM as condensed result. More details are provided in a report file that is stored in the user-defined location.

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 name of the health check job to execute is known. For instructions on how to create the job, refer to *Create Health Check Job*.
- No other health check job execution is in progress.
- The Uniform Resource Identifier (URI) of the local or remote destination where the report file is to be stored is known. In particular, in case of export to remote destination, the IP address/name of the target machine, are known.
- If the URI requires a password, the password is known.
- An Ericsson Command-Line Interface (ECLI) session in Exec mode is in progress.





2 Procedure

To execute a health check job manually, providing user defined location URI for storing the report file:

1. Navigate to the *HealthCheckM* MO:

```
>dn ManagedElement=<node_name>,SystemFunctions=1,HealthCheckM=1
```

Note: The string *node_name* is specific for the ME.

2. Navigate to the *HcJob* MO representing the health check job to be executed, for example:

```
(HealthCheckM=1) >HcJob=jobName
```

3. Start the job execution providing the user-defined location URI where the report file has to be stored:

```
(HcJob=jobName) >execute --exportUri "<userDefinedURI>"  
[--exportPassword "<password>"]
```

Note:

In case of user-defined local URI:

- The *userDefinedURI* string can be any of the following:
 - The local absolute directory path where the report file and compressed archive file containing logs are to be stored.
 - The local relative path, starting from path specified by attribute *localFileStorePath*, where the report file and compressed archive file containing logs are to be stored.
- The *exportPassword* parameter is not needed. If provided, it is ignored

In case of user-defined remote URI:

- The *userDefinedURI* string is the URI of the remote destination where the report file has to be stored it is composed according to the generic URI syntax defined by RFC 3986. Currently only the SFTP protocol is supported according to the following URI format:
sftp:// [<user>@] <host> [:<port>] [/path].
- The *password* string is the password to be used for authentication on the remote destination.



A compressed archive file, containing logs used for the rules evaluation, is also provided by HCF, it is stored in the same location as the report file. Hence both the report file and the compressed archive are stored in the user-defined location.

The user-defined destination for report file and compressed archive is also applied to the job to be triggered, if any. For instructions about how to set a job to trigger, refer to *Modify Health Check Job To Trigger*.

4. Check the job execution progress:

```
(HcJob=jobName) >show progressReport
```

The following are example outputs:

```
actionName="EXECUTE"
progressInfo="Job Execution ongoing"
progressPercentage=33
result=NOT_AVAILABLE
resultInfo=""
state=RUNNING
timeActionStarted="2015-04-23T16:08:58"
timeOfLastStatusUpdate="2015-04-23T16:10:07"
```

```
actionName="EXECUTE"
progressInfo="Job Execution ongoing"
progressPercentage=72
result=NOT_AVAILABLE
resultInfo=""
state=RUNNING
timeActionStarted="2015-04-23T16:08:58"
timeOfLastStatusUpdate="2015-04-23T16:10:08"
```

Note: The current job execution percentage is available in the `progressPercentage` attribute. Once the job completes execution, `progressPercentage=100` is shown.

The progress of job execution is available in the `progressInfo` attribute. When the job execution is complete, it shows the `Job Execution completed` value.

The result of job execution, in terms of success or failure, is available in `result` attribute. It shows value `NOT_AVAILABLE` until job completion. Once the job is executed, it shows `SUCCESS` if the job accomplished its execution without problem or `FAILURE` if the job execution terminated because of an error.

The current state of the job is available in the `state` attribute. While the job is executing `state=RUNNING` is shown. When the job terminates its execution successfully `state=FINISHED` is shown whereas if the job ends its execution because of an error `state=CANCELLED` and `result=FAILURE` are the provided outputs.



5. Check the job execution progress until completion:

```
(HcJob=jobName) >show progressReport
```

The following is an example output in case of success:

```
actionName="EXECUTE"
progressInfo="Job Execution completed"
progressPercentage=100
result=SUCCESS
resultInfo="Job correctly executed"
state=FINISHED
timeActionCompleted="2015-04-23T15:10:07"
timeActionStarted="2015-04-23T15:09:57"
timeOfLastStatusUpdate="2015-04-23T15:10:07"
```

The following is an example output in case of failure:

```
actionName="EXECUTE"
progressInfo="Job Execution completed"
progressPercentage=100
result=FAILURE
resultInfo="The provided output folder doesn't exist."
state=CANCELLED
timeActionCompleted="2015-04-23T15:46:21"
timeActionStarted="2015-04-23T15:46:21"
timeOfLastStatusUpdate="2015-04-23T15:46:21"
```

6. Is the job execution ended with success?

Yes: Continue with the next step.

No: Proceed with Step 3.



Note: Job execution failure can depend on any of the following:

- A rule set file containing rules not correct from the semantic perspective. The rule set file must be checked.
- In case `<userDefinedURI>` is a user-defined local location : it was not possible to write the report file in the output directory. Check the output directory on the file system.
- In case `<userDefinedURI>` is a user-defined remote location: problems were detected exporting the report file and compressed archive to the remote destination. Possible causes are:
 - The URI is not correctly formed. Check the format of the remote destination URI.
 - The remote destination is not available. Check the availability via SFTP of the remote destination then the correctness of the provided remote destination address/path.
 - Authentication on the remote destination fails. Check the credentials provided for authentication.
 - General problems during file transfer. Check the connection stability.
 - Time-out expiration during file transfer. If the predefined time-out for job execution expires before both report file and compressed archive are transferred to the remote location, the job fails. The value of time-out for job execution is set to 5 minutes. Huge file dimension or connection slowness are possible causes. Check the connection speed. The HCF job time-out expiration does not interrupt the file transfer if it is ongoing.

Retry executing the job. If the problem persists, consult the next level of maintenance support, this procedure ends.

7. Check the ME health status:

```
(HcJob=jobName) >show status
```

The health status of the ME, as computed by the job execution, is shown.

The following is an example output in case all rules check executed by the job were successful:

```
status=HEALTHY
```

The following is an example output in case the check of at least one rule whose severity is `CRITICAL` failed:

```
status=NOT_HEALTHY
```



Under the user-defined `<userDefinedURI>`, the report file is available with details about job execution, computed ME health status, and information about successful and failed rules. Moreover, in the same directory, a compressed archive file containing logs used for the rules evaluation is stored.

8. Is the ME health status different from `HEALTHY`?

Yes: Continue with the next step.

No: This procedure ends.

9. Check the list of failed rules:

```
(HcJob=jobName) >show failedRules
```

The list of rules, executed by the health check for which the check result was not as expected, is shown, for example:

```
failedRules
  hcRule="hcRuleId=PROVIDER2_006"
  reason="The CPU usage percentage is Greater than the defined threshold value."
  severity=CRITICAL
failedRules
  hcRule="hcRuleId=PROVIDER2_007"
  reason="TThere is at least one software item not used. =>
    Check it with cmw-repository-list Command"
  severity=WARNING
failedRules
  hcRule="hcRuleId=PROVIDER_0015"
  reason="Core Dumps present."
  severity=CRITICAL
```

Note: The attribute `hcRule` specifies the relative distinguished name of the failed rule. It is expressed in the form of `"hcRuleId=<rule_id>"`.

10. For each failed rule, retrieve the recommended action. Navigate to the *HcRule* MO representing each rule:

```
(HcJob=jobName) >dn ManagedElement=1, SystemFunctions=1, HealthCheckM=1, HcRule=<rule_id>
```

```
(HcRule=<rule_id>) >show recommendedAction
```

Note: The string `rule_id` is the value component of each failed rule retrieved in Step 9.

The following is an example output:

```
(HcRule=PROVIDER_0015) >show recommendedAction
recommendedAction="Collect all the needed information according to =>
  the OPI and contact the next level of support"
```

The suggested action must be performed to get related rule successfully executed.