

CUDB Troubleshooting Guide

CUDB TROUBLESHOOTING GUIDE

Copyright

© Ericsson AB 2016-2018. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

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

Trademark List

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



Contents

1	Introduction	1
1.1	Purpose and Scope	1
1.2	Revision Information	1
1.3	Target Groups	3
1.4	Prerequisites	3
1.5	Related Information	3
2	Tools	5
2.1	CLI Commands	5
2.2	UDC Cockpit Tool	10
3	Troubleshooting Procedure	11
4	Trouble Reporting	13
	Glossary	15
	Reference List	17





1 Introduction

This document provides troubleshooting information for CUDB nodes. However, since a CUDB system is made up of a set of CUDB nodes, the document is also valid for troubleshooting system-level problems and failures.

1.1 Purpose and Scope

The purpose of this document is to provide the instructions and tools needed to recover a CUDB node in case of abnormal behavior or failures.

The document does not contain information on the maintenance tasks and configuration procedures, refer to *CUDB Node Preventive Maintenance*, Reference [1] for more information on these topics.

For more information on the user names and passwords used in this document, refer to *CUDB Users and Passwords*, Reference [2].

Note: In case the procedures described in this document do not fix the experienced faults, contact the next level of Ericsson support.

1.2 Revision Information

Rev. A

This document is based on 1/1553-HDA 104 03/9 with the following changes:

- Updated hardware information.
- Virtualization terminology update all throughout the document.

Rev. B

Other than editorial changes, this document has been revised as follows:

- Terminology updated throughout the document.
- Section 2.1.4 on page 7: The `cudbMpstat` command is available for Ericsson personnel only.

Rev. C

Other than editorial changes, this document has been revised as follows:

- Updated Ericsson personnel information.



Rev. D

Other than editorial changes, this document has been revised as follows:

- Updated Ericsson personnel information.

Rev. E

Other than editorial changes, this document has been revised as follows:

- Updated Ericsson personnel information.

Rev. F

Other than editorial changes, this document has been revised as follows:

- Section 2.1.3 on page 7: Updated the note regarding the `cudbCollectInfo` command.
- Section 2.1.5 on page 7: Updated Step 2 with `-v` option and added differentiation between ESA FM Agent and ESA Master Agent.

Rev. G

Other than editorial changes, this document has been revised as follows:

- Updated Ericsson personnel information.

Rev. H

Other than editorial changes, this document has been updated as follows:

- Updated Ericsson personnel information.

Rev. J

Other than editorial changes, this document has been updated as follows:

- Updated Ericsson personnel information.

Rev. K

Other than editorial changes, this document has been updated as follows:

- Updated Ericsson personnel information.



1.3 Target Groups

This document is intended for personnel working with the CUDB system.

1.4 Prerequisites

This document provides troubleshooting information only for properly installed and configured CUDB nodes.

Before starting the troubleshooting procedure, ensure the following:

- The CUDB node was installed and configured appropriately before the abnormal behavior or failure occurred.
- This document is available.
- All the documents listed under References are available. For the list of documents, see Reference List on page 17.

Attention!

Do not activate trace or log without prior consultation with Ericsson, as it can affect traffic throughput. Certain troubleshooting activities can have an impact on node performance.

1.5 Related Information

Definition and explanation of acronyms and terminology, trademark information, and typographic conventions can be found in the following documents:

- *CUDB Glossary of Terms and Acronyms*, Reference [3]
- *Trademark Information*, Reference [4]
- *Typographic Conventions*, Reference [5]





2 Tools

This section describes the tools that can be used to troubleshoot the CUDB system.

2.1 CLI Commands

This section describes the tools and resources that can be used to help troubleshoot CUDB through the command line interface of CUDB Nodes.

2.1.1 Checking the Active System Controller

Several troubleshooting resources (such as the `cudbGetLogs` and `cudbAnalyser` scripts, or the CoreMW console) can be executed only on the active System Controller (SC). If needed, use the following command to check which SC is the active one:

```
# cudbHaState | grep COM | grep ACTIVE
```

The expected output must be similar to the below example:

```
COM is assigned as ACTIVE in controller SC-1.
```

2.1.2 CUDB Logchecker

CUDB offers a troubleshooting tool called Logchecker, a set of scripts that aim to improve the in-service performance of CUDB, and also help troubleshooting by means of log collection and automatic log analysis.

The CUDB Logchecker consists of the following two scripts:

- `cudbGetLogs`

This script collects preventive maintenance logs for log analysis. Log collection takes approximately 4-5 minutes, and the resulting log is saved in the following folder:

```
/home/cudb/monitoring/preventiveMaintenance/
```

The name of the log file contains the node ID and a timestamp in the following format:

```
<CUDB_node_id>_<YYYYmmddHHMM>.log
```

For example, a log file for a CUDB node with ID 99 looks as follows:



```
CUDB_99_201304090025.log
```

For more details about the command, refer to *CUDB Node Commands and Parameters*, Reference [6].

- `cudbAnalyser`

This script analyzes the logs collected by `cudbGetLogs`.

Note: The `cudbGetLogs` and `cudbAnalyser` scripts can be executed only on active SCs. See Section 2.1.1 on page 5 for more information on how to check the active SC.

By default, Logchecker performs scheduled log analysis at 00:50 and 12:50, and saves the detailed result in the following location:

```
/home/cudb/monitoring/preventiveMaintenance/cron_analysis.<SC_NAME>.log
```

In the above path, the `<SC_NAME>` variable can be `SC_2_1` or `SC_2_2`.

The automatic log analysis raises or clears alarms according to the analysis result. The severity of the alarms depends on the severity of the detected faults. For further information, refer to *Preventive Maintenance, Logchecker Found Error(s)*, Reference [7].

For more information about the CUDB Logchecker, refer to *CUDB Logchecker*, Reference [8] and *CUDB Node Commands and Parameters*, Reference [6].

2.1.2.1 Manual Log Collection

The CUDB Logchecker logs can be collected manually with the `cudbGetLogs` command. An example output of the command is shown in Example 1.

```
CUDB107 SC_2_1# cudbGetLogs
Starting /opt/ericsson/cudb/OAM/bin/cudbGetLogs ...
Grepping logs and creating /home/cudb/ \
monitoring/preventiveMaintenance/ \
CUDB_107_201304091136.log ...
The log file is saved as : /home/cudb/\
monitoring/preventiveMaintenance/ \
CUDB_107_201304091136.log
CUDB107 SC_2_1#
```

Example 1 Manual Log Collection

Use this content to provide fault information to the next level of Ericsson support.

2.1.2.2 Log Analysis

Two options are available to trigger unscheduled log analysis:



- **cudbAnalyser -a**: This option is used to check the last two log files automatically.
- **cudbAnalyser --logfile <newer_logfile> --previous-logfile <older_logfile>**: This option is used to check and compare two log files defined with the <newer_logfile> and <older_logfile> parameters.

Note: The timestamp of <newer_logfile> must be newer than that of <older_logfile> in order to perform a correct analysis.

Below is an example of how to compare two log files:

```
cudbAnalyser --logfile /home/cudb/monitoring/preventive
Maintenance/CUDB_28_201304090300.log --previous-logfile
/home/cudb/monitoring/preventiveMaintenance/CUDB_28_2
01304080300.log
```

2.1.3 CUDB Logs Collection

The **cudbCollectInfo** command creates a compressed archive from the CUDB logs. Use the output of this command to provide fault information to the next level of Ericsson support.

For further information about this command, execute **cudbCollectInfo -h**, or refer to *CUDB Node Commands and Parameters*, Reference [6].

Note: **cudbCollectInfo** is a system-level command, executed on all nodes automatically. Therefore, only one system-wide **cudbCollectInfo** command can be executed in the CUDB system at a time.

The command can take from 5 to 20 minutes, depending of the amount of information to be collected.

2.1.4 LDE Status

The status of the Linux Distribution Extension (LDE) platform can be checked with the **cudbHaState** command. To use this command, at least one SC must be in ACTIVE state.

Execute the command on an SC as follows (in the example below, **SC_2_1** is used):

```
SC_2_1# cudbHaState
```

Execute the following command to check the alarm status on all clusters:

```
SC_2_1# cluster alarm --status --all
```

For more details about the commands, refer to *CUDB Node Commands and Parameters*, Reference [6] and *Data Collection Guideline for CUDB*, Reference [9].



2.1.5 ESA Processes

The status of the ESA processes can be checked by performing the following steps:

1. Check that the ESA agents are running on both SC. Execute the following command on both SCs:

```
esa status
```

The expected output must look similar to the following:

```
[info] ESA Sub Agent is running.
[info] ESA Master Agent is running.
[info] ESA PM Agent is running.
```

- 2.

- a. Check the ESA FM Agent cluster status with the following command on any of the SC blades:

```
esaclusterstatus
```

The expected output must look similar to the example below. One SC blade must be in M state, while the other in (M) state:

```
M * OAM1 10.22.0.1
(M) OAM2 10.22.0.2
```

- b. Check the ESA Master Agent cluster status with the `-v` option:

```
esaclusterstatus -v
```

The expected output must look similar to the example below:

```
M * esama esafma OAM1 10.22.0.1
(M) esama esafma OAM2 10.22.0.2
```

Where ESA in the first host listed is the ESA Master Agent Active Master.

The description of the different states are:

M	ESA Master is located on that SC.
(M)	ESA Slave is located on that SC.
*	The SC from where the command was sent.
esama	ESA Master Agent is running on that SC.



esafma	ESA FM Agent is running on that SC.
Inactive	Cluster (ESAFM or ESAMA) is inactive for ESA on that SC.
Unknown	Cluster (ESAFM or ESAMA) is configured as active for ESA on that SC, but it is not connected to the cluster.

If a host is not shown, we can assume that either the node is down or only the ESA is down in that host.

In case the ESA cluster mode is inactive, the output will simply indicate it as follows:

```
Cluster mode inactive.
```

For more details about the command, refer to the *Command: Cluster Status* section of *ESA Setup and Configuration*, Reference [10].

2.1.6 Check Database Consistency between Database Clusters

Use the `cudbCheckConsistency` command to check database consistency in a lightweight manner between database clusters (that is, between the master PLDB or DSG replicas and their slaves) as follows:

```
SC_2_1# cudbCheckConsistency
```

For more details about the command, refer to *CUDB Node Commands and Parameters*, Reference [6].

2.1.7 cudbConsistencyMgr

Use the `cudbConsistencyMgr` command to check database consistency between database clusters (that is, between the master PLDB or DSG replicas and their slaves) as follows:

```
SC_2_1# cudbConsistencyMgr --order ms --node <nodeid>
{--dsg <dsgid> | --pl}
```

For more details about the command, refer to *CUDB Node Commands and Parameters*, Reference [6].

2.1.8 Check Replication Channels Behavior

Replication channels on CUDB system level can be checked by generating dummy transactions, and then verifying that those transactions are replicated properly for each DSG and PLDB slave replica.

Use the `cudbCheckReplication` command as follows to generate dummy transactions:



```
SC_2_1# cudbCheckReplication
```

For more details about the command, refer to *CUDB Node Commands and Parameters*, Reference [6].

2.2 UDC Cockpit Tool

To follow present and recall earlier system status and performance information of CUDB nodes, use the UDC Cockpit. This is a monitoring application, which presents collected data on a single, web-based GUI.



3 Troubleshooting Procedure

A troubleshooting workflow is shown in Figure 1.

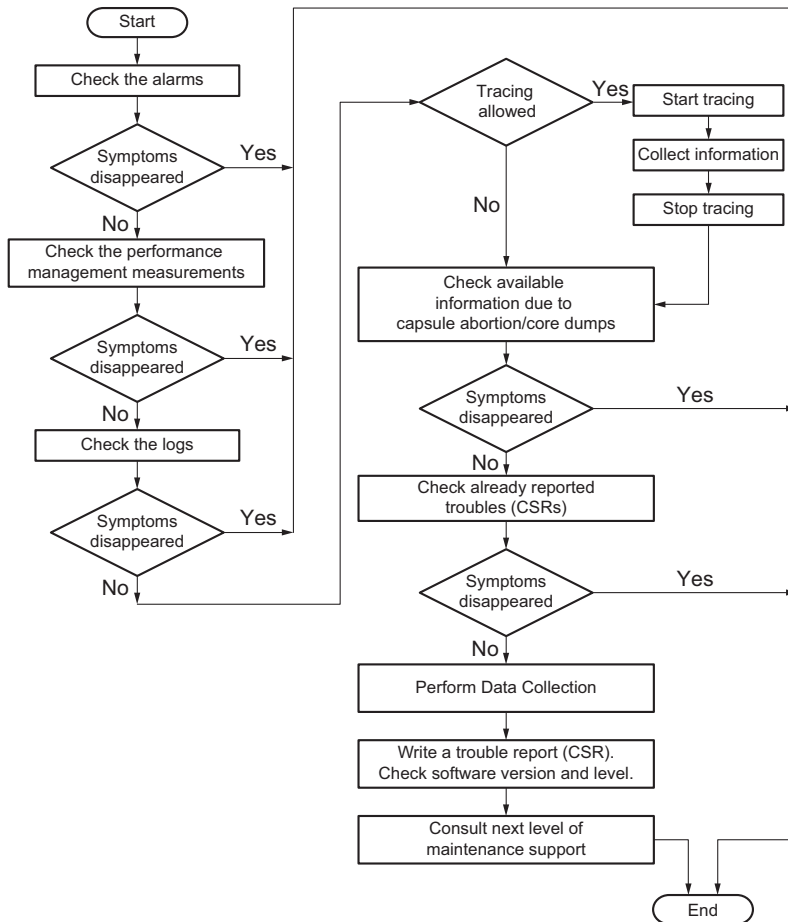


Figure 1 Troubleshooting Workflow





4 Trouble Reporting

Problems identified that cannot be solved by using this document must be reported to the next level of maintenance support through a Customer Service Report (CSR).

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

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

For more information on how to collect information, refer to *Data Collection Guideline for CUDB*, Reference [9].

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





Glossary

For the terms, definitions, acronyms and abbreviations used in this document, refer to *CUDB Glossary of Terms and Acronyms*, Reference [3].





Reference List

CUDB Documents

- [1] *CUDB Node Preventive Maintenance*
- [2] *CUDB Users and Passwords*, 3/00651-HDA 104 03/10
- [3] *CUDB Glossary of Terms and Acronyms*
- [4] *Trademark Information*
- [5] *Typographic Conventions*
- [6] *CUDB Node Commands and Parameters*
- [7] *Preventive Maintenance, Logchecker Found Error(s)*
- [8] *CUDB Logchecker*
- [9] *Data Collection Guideline for CUDB*

Other Ericsson Documents

- [10] *ESA Setup and Configuration*