Troubleshoot Tools for FND

Contents

Introduction

This document describes how to use troubleshooting tools which are available as part of Field Network Director (FND) deployment.The FND solution is broad in scope and encompasses numerous diverse technologies and specializations. Due to this, there are numerous scripts and command line tools which can help verify behavior in a certain situation or diagnose a complex issue.

Contributed by Ryan Bowman, Cisco TAC Engineer.

Prerequisites

Requirements

Cisco recommends that you have a fully operational production or lab environment with registered Head End Router (HER), Field Area Router and Connected Grid Endpoint (CGE). In order to verify CoAP Simple Management Protocol (CSMP) statistics with **getStats.sh** you must have at least one CGE that generates CSMP traffic.

In order to utilize the files located in the /opt/cgms-tools/ directory, the **cgms-tools** RPM package must be installed on your application server.

Components Used

The information in this document was all gathered with the use of FND version 3.0.1-36 with all Linux servers installed on virtual machines running RHEL 6.5.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Command Line Tools

This section covers CLI utilities available as part of both the **cgms** and **cgms-tools** packages. The default installation path for the **cgms** RPM is **/opt/cgms/**and the default installation path for **cgms-**

tools is **/opt/cgms-tools/**.

setupCgms.sh (/opt/cgms/bin/setupCgms.sh):

After first-time installation of the FND this script should be run order to configure necessary application variables. Once production has started, you can still use this utility to change critical configuration parameters. Before you execute this script, you must stop the cgms service, navigate to the **/opt/cgms/bin/** directory and execute the **./setupCgms** command.

```
[root@fnd bin]#./setupCgms.sh
Are you sure you want to setup IoT-FND (y/n)? n
Do you wish to configure another database server for this IoT-FND ? (y/n)? n
Do you want to change the database password (y/n)? n
Do you want to change the keystore password (y/n)? n
Do you want to change the web application 'root' user password (y/n)? n
Do you want to change the FTP settings (y/n)? n
Do you want to change router CGDM protocol settings (y/n)? n
Do you want to change log file settings)? (y/n)? n
```

getstats.sh (/opt/cgms/bin/getstats.sh):

This script is intended to be executed while the application is up and running. It is extremely helpful when you analyze performance with load balanced clusters and active/standby DB pairs. Each performance metric is beyond the scope of this article but here is a sample output when we run the script.

When you troubleshoot FND deployments in a cluster, execute this script on each server in order to verify that load balancing works correctly. If one of the app servers has a much higher CSMP processing rate than the others, then load balancing is probably not configured incorrectly. Furthermore, when you analyze this output, if you see your queue sizes increase then you know there is a bottleneck process somewhere.

```
[root@fnd bin]# ./getstats.sh
Current Time: 2017-03-08 01:06
============ events statistics ==============
ElapsedTimePrepareForRules (ms):..................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimeBatchCommit (ms):......................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
Events request rate:..............................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimeSendToSyslog (ms):.....................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
Batch Commit Size :...............................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
==============================================
============ metric statistics ==============
ElapsedTimePersistBatch (ms):.....................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimePersistNetElementMetrics (ms):.........................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimeBatchCommit (ms):......................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
Incoming message rate to Metric Server:...........................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimeFindCurrentMetric (ms):................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimefindCurrentMetricsForNetObject (ms):...................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
sendMetricEvents:.................................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimePersistNetElementMetric (ms):...................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimeAddMetricWithoutPropagation (ms):...............................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
Rate of message drop at the metric server:.................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
Batch Commit Size :........................................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimeAddMetricsInBulkWithoutPropagation (ms):........................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
==============================================
============ issues statistics ==============
Issues Incoming Rate:......................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
UpdateEventAndIssues (ms):.................................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimeBatchCommit (ms):...............................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
Batch Commit Size :........................................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
Issues Processing Rate:....................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
==============================================
============ label statistics ==============
Label drop rate:...........................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimePersistBatch (ms):..............................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
Label processing rate:.....................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
ElapsedTimeBatchCommit (ms):...............................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
Label request rate:........................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
Batch Commit Size :........................................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
==============================================
============ csmp statistics ==============
csmpConNotificationRate:...................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
csmpNonNotificationRate:...................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
csmpNonQueueSize:..........................................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
csmpNotificationRate:......................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
csmpDropRate:..............................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
csmpProcessingRate:........................................................[ val: 0/s over 1 min
] [ avg: 0.0 ] [ counter: 0 ] [ lastUpdate: never ]
csmpConQueueSize:..........................................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
==============================================
============ database connection pool statistics ==============
dbConFlushCount:...........................................................[ val: 0 ] [ avg: 0.0
] [ counter: 0 ] [ lastUpdate: never ]
ActiveCount:...............................................................[ val: 13 ]
InUseCount:................................................................[ val: 7 ]
AvailableCount:............................................................[ val: 243 ]
CreatedCount:..............................................................[ val: 13 ]
DestroyedCount:............................................................[ val: 0 ]
```

keytool (/opt/cgms/jre/bin/keytool):

It is important to know that the FND installation comes with Java. You need to use the **keytool** utility in order to create and manage the cgms_keystore which has to be appropriately configured on both the FND and the Tunnel Provisioning Server (TPS).

In some environments, the server already has Java installed and the **keytool** command will be available for any user through use of the $PATH environment variable. If you use the **keytool** command and find this error then there is another solution for you:

```
[root@fnd]# keytool
-bash: keytool: command not found
```

You can navigate to the **/opt/cgms/jre/bin/** directory and invoke the keytool utility in this directory for example:

```
[root@fnd ~]# keytool -v -list -keystore /opt/cgms/server/cgms/conf/cgms_keystore
-bash: keytool: command not found
[root@fnd ~]# cd /opt/cgms/jre/bin/
[root@fnd bin]# ./keytool -v -list -keystore /opt/cgms/server/cgms/conf/cgms_keystore
Enter keystore password:
```

cgdm-client.sh (/opt/cgms-tools/bin/cgdm-client.sh):

> **Note**: This script and other scripts in the /opt/cgms-tools/ directory are bundled in the **cgms-tools** RPM package.

FND uses Netconf over HTTPS in order to access and communicate with Field Area Router (FAR). Netconf uses XML formatted messages in order to provide a service which is not only reliable and dependable but which can also be easily broken down and sent to a database. There is a CLI tool called **cgdm-client** which will open a manual Connected Grid Device Manager (CGDM) session to a FAR of your choice, execute a remote command and send the XML received in response from FAR to stdout in BASH.

If you execute the script without options, you will be presented with usage guidelines:

```
[root@fnd bin]# ./cgdm-client
ERROR: Please specify an IP address and a command
usage: cgdm-client <cgr ip address> <cgdm CLI command>
-c <arg> Conf and keystore directory path, default =
/opt/cgms/server/cgms/conf
-v Verbose mode
```

For example, let's say you want to verify that time is fully synchronized on one of your routers where the management IP (the 'IP' value in your .csv file) is 192.0.2.1. From a terminal session on your FND application server, you could query the time on a CGR with the **show clock** command:

```
[root@fnd bin]# ./cgdm-client 192.0.2.1 show clock
<?xml version="1.0" encoding="ISO-8859-1"?>
<nf:rpc-reply xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns="http://www.cisco.com/nxos:1.0" message-id="1">
<nf:data>15:44:58.092 CST Mon Mar 13 2017
</nf:data>
</nf:rpc-reply>
```

Optionally, you can specify verbose output with the use of the 'v' flag in your command. Note that the verbose output comes from the Java and Cisco software processes and syntax. You will not see any additional network or device information in this output:

```
[root@fnd bin]# ./cgdm-client -v 192.0.2.1 show clock
< output omitted >
```

csmp-request.sh (/opt/cgms-tools/bin/csmp-request.sh):

In addition to the **cgdm-client** FAR tool, there is a tool for endpoints called **csmp-request**. Similarly to the **cgdm-client** script, this script will allow you to query information from your CGEs using CSMP. You just need to specify the IPv6 address of the mesh endpoint and the TLV (Type Length Value) that you are querying on the device. The full list of TLV codes is out of scope in this article but a few well-known examples will be shown below. Syntax for the script is:

./csmp-request -r [] TLV-Value

1. Query CGE firmware version on a meter with IP 2001:db8::1/32

```
[root@fnd bin]# ./csmp-request -r [2001:db8:0:0:0:0:0:1] 75
```

2. Query uptime on a meter with IP 2001:db8::1/32

```
[root@fnd bin]# ./csmp-request -r [2001:db8:0:0:0:0:0:1] 22
```

signature-tool (/opt/cgms-tools/bin/signature-tool):

The Signature Tool is a Java utility which will allow you to encrypt clear-test passwords, decrypt encrypted passwords or strings and print the SSM_CSMP certificates in clear text. This tool should be used to generate encrypted password strings for your .csv files so that they do not contain administrator passwords in clear-text.

To view command syntax, execute the script without options:

```
[root@fnd bin]# ./signature-tool
usage:
signature-tool print
signature-tool export <binary|base64> <filename>
signature-tool decrypt <keystore> <filename>
signature-tool encrypt <keystore> <filename>
```

To print the SSM_CSMP certificate use:

```
[root@fnd bin]# ./signature-tool print
```

To encrypt a clear-text admin password:

1. Navigate to the **/opt/cgms-tools/bin** directory:
   ```
   [root@fnd ~]# cd /opt/cgms-tools/bin
   [root@fnd bin]# pwd
   /opt/cgms-tools/bin
   ```
2. Create a new text file which contains only the relevant string/password in clear-text:

1.
   ```
   [root@fnd bin]# echo AdminPassword > clear-text-password.txt
   [root@fnd bin]# cat clear-text-password.txt
   AdminPassword
   ```
2. Execute the Signature Tool script with the use of the 'encrypt' option, and specify the exact path of the cgms_keystore file and the name of the file you just created which has the clear-text password in it. When prompted for the alias, use 'cgms' as only the certificate in the cgms_keystore file with the 'cgms' alias is used by the FND application to authenticate with your CA:

```
[root@fnd bin]# ./signature-tool encrypt /opt/cgms/server/cgms/conf/cgms_keystore clear-text-
```

```
password.txt
Enter alias: cgms
Enter password:
pXHcF+YxyoJarz4YAqvFVMrLT2I//caHLddiJfrb7k65RmceIJUNlDd2dUPhGyGZTeEfz8beh8tWSGZ4lc66rhAQ9mYNaw2X
SPaL8psoK+U0wzHgY068tnc7q17t05CZ5HQh8tWSGZ4lc66rhAQ9mOivj1B3XRKFmkpSXo4ZubeKRJ4NNaGAKFV8cjBJQDWs
h7NAXL3x5D62/7w4Mhmftf2XiGlqeWlc66rhAQF+YxyoJarz4YAqvFVMrLT2I//caHLIDYoKoeTVB2SLQXtSZR+dwxYjQsE0
hCmBpHv0lDD/l4gg==
```

To decrypt an encrypted string:

1. Create a new .txt file in the **/opt/cgms-tools/bin/** directory using the encrypted string:

```
[root@fnd bin]# echo
pXHcF+YxyoJarz4YAqvFVMrLT2I//caHLddiJfrb7k65RmceIJUNlDd2dUPhGyGZTeEfz8beh8tWSGZ4lc66rhAQ9mYNaw2X
SPaL8psoK+U0wzHgY068tnc7q17t05CZ5HQh8tWSGZ4lc66rhAQ9mOivj1B3XRKFmkpSXo4ZubeKRJ4NNaGAKFV8cjBJQDWs
h7NAXL3x5D62/7w4Mhmftf2XiGlqeWlc66rhAQF+YxyoJarz4YAqvFVMrLT2I//caHLIDYoKoeTVB2SLQXtSZR+dwxYjQsE0
hCmBpHv0lDD/l4gg== > encrypted-password.txt
```

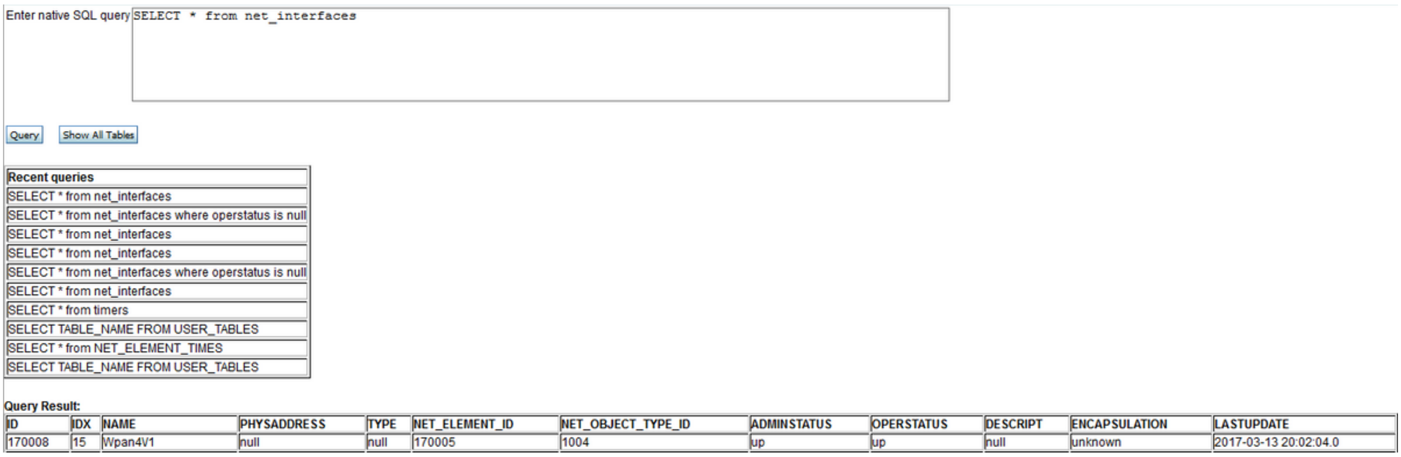
2.Execute the Signature Tool with the use of the **decrypt** option, and once again specify the exact path of the keystore file as well as the name of the .txt file which has the encrypted password stored in it.

```
[root@fnd bin]# ./signature-tool decrypt /opt/cgms/server/cgms/conf/cgms_keystore encrypted-
password.txt
Enter alias: cgms
Enter password:
AdminPassword
```

Database Tools

Just like the robust set of command line tools/utilities, the FND contains a nice suite of GUI based tools which can help you analyze and diagnose troubles with the database. To access the DB tools, log into the main Dashboard of your FND deployment and then paste /pages/diag/db.seam after the .com portion of your URL.

This area has three tabs: DB Query, DB Info and Log Viewer. The DB Query tab will let you run custom queries and will provide a list of all tables if you click on **Show All Tables** to the right of the **Query** button. For example, to view the layer 1 and 2 status for all device interfaces, type **SELECT * FROM NET_INTERFACES** in the SQL query box and then click on the **Query** button. You will be provided with a list of all HER and FAR interfaces, their MAC addresses, Administrative layer 1 status and layer 2 link status for each interface.



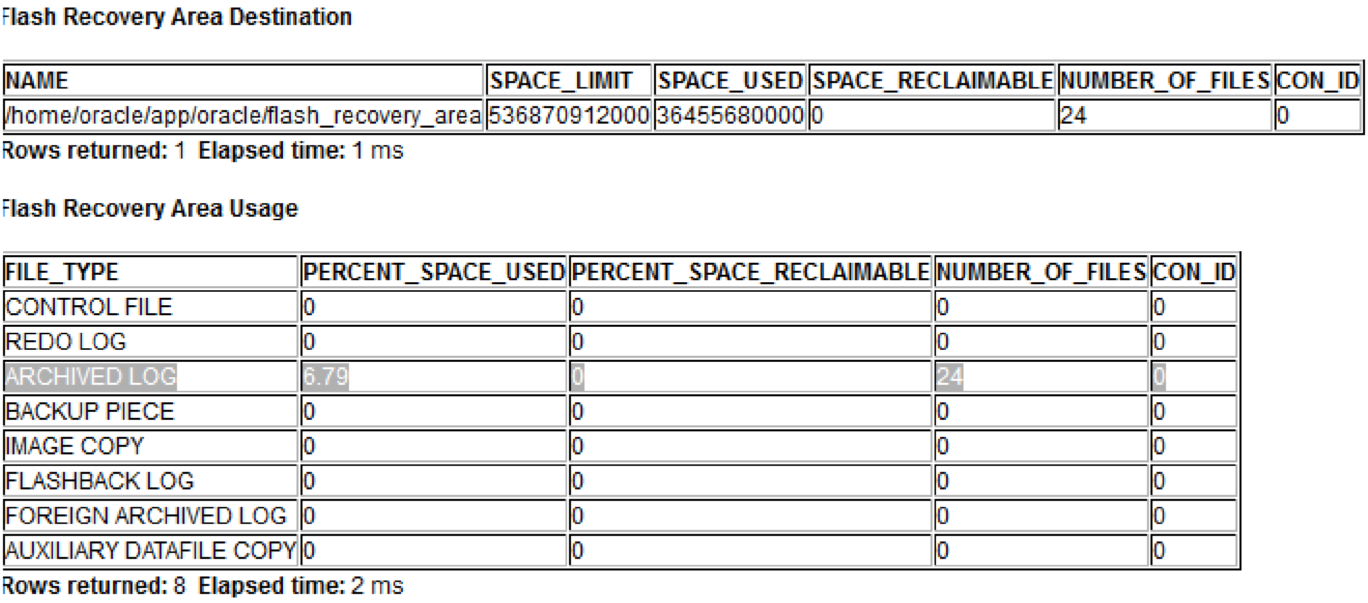
```
SELECT * from net_interfaces
```

Recent queries
SELECT * from net_interfaces
SELECT * from net_interfaces where operstatus is null
SELECT * from net_interfaces
SELECT * from net_interfaces
SELECT * from net_interfaces where operstatus is null
SELECT * from net_interfaces
SELECT * from timers
SELECT TABLE_NAME FROM USER_TABLES
SELECT * from NET_ELEMENT_TIMES
SELECT TABLE_NAME FROM USER_TABLES

Query Result:

ID	IDX	NAME	PHYSADDRESS	TYPE	NET_ELEMENT_ID	NET_OBJECT_TYPE_ID	ADMINSTATUS	OPERSTATUS	DESCRIPT	ENCAPSULATION	LASTUPDATE
170008	15	Wpan4V1	null	null	170005	1004	up	up	null	unknown	2017-03-13 20:02:04.0

If you want to verify database connection settings, click on the **DB Info** tab of the db.seam page. Here, you will have read-only access to numerous database variables such as connection URL, database username, Oracle version, port number, SID and size of each table. Also listed on this page Flash Recovery Area (FRA) information such as space used by each type of file stored on FRA and how much space is reclaimable.



Flash Recovery Area Destination

NAME	SPACE_LIMIT	SPACE_USED	SPACE_RECLAIMABLE	NUMBER_OF_FILES	CON_ID
/home/oracle/app/oracle/flash_recovery_area	536870912000	36455680000	0	24	0

Rows returned: 1 **Elapsed time:** 1 ms

Flash Recovery Area Usage

FILE_TYPE	PERCENT_SPACE_USED	PERCENT_SPACE_RECLAIMABLE	NUMBER_OF_FILES	CON_ID
CONTROL FILE	0	0	0	0
REDO LOG	0	0	0	0
ARCHIVED LOG	6.79	0	24	0
BACKUP PIECE	0	0	0	0
IMAGE COPY	0	0	0	0
FLASHBACK LOG	0	0	0	0
FOREIGN ARCHIVED LOG	0	0	0	0
AUXILIARY DATAFILE COPY	0	0	0	0

Rows returned: 8 **Elapsed time:** 2 ms