SIP Feature Server Administration Guide

Appendix: Performing maintenance operations on embedded Cassandra

4/16/2025

Appendix: Performing maintenance operations on embedded Cassandra

The nodetool is a Cassandra utility for managing a Cassandra cluster. Use it for Feature Server Cassandra Maintenance.

Regular maintenance repairs inconsistencies across all data ranges, and ensures that replicated data is consistent across all nodes. You should perform maintenance on a node after a Feature Server upgrade and after new Feature Server nodes were added to the cluster. Apply additional maintenance steps after you remove a node from the cluster. See Performing maintenance after removing a node from the cluster for more details.

> **Important**
>
> The following Feature Server Cassandra Maintenance procedures are applicable only in the embedded Cassandra deployment. For information about the External Cassandra maintenance tasks, refer to the Cassandra documentation.

Performing regular maintenance

Run the nodetool command `repair` on all Feature Server hosts in a Cassandra cluster. See Running Nodetool on a Feature Server Host and Running nodetool repair on all Feature Server hosts in a Cassandra cluster for details about running nodetool and nodetool repair, respectively.

> **Important**
>
> All Feature Server hosts must be up and running to run repair. Running repair increases memory use and may not succeed if the Java heap space max limit is reached. For information on the Java heap size, see Hardware and software prerequisites.

As a best practice, schedule the regular maintenance weekly during the maintenance window (low usage hours), for example, by using Cron on Linux or as a Task Scheduler on Windows.

If you are planning for an ad hoc repair, then start the repair from the Feature Server instances that are configured as alternatevoicexml.

Contents

Running nodetool on a Feature Server host

The Nodetool utility has a command line interface. The steps below describe how to run nodetool commands on Cassandra when it is deployed on a host, as part of a Feature Server deployment.

1. Change the current directory.

In the Linux shell terminal:

```
cd Feature Server Installation Directory/work/
jetty-vms_host-http_port-fs.war-_fs-any-/webapp/WEB-INF/lib
```

In the Windows command prompt window:

```
cd Feature Server Installation Directory\work\
jetty-vms_host-http_port-fs.war-_fs-any-\webapp\WEB-INF\lib
```

where...

Feature Server Installation Directory is where Feature Server is installed in that host.

vms_host is the IP address of the vms host parameter that Feature Server started with, defined in launcher.xml or in the command line. The default is 0.0.0.0.

http_port is the port number that Feature Server started with, defined in launcher.xml. The default is 8080.

2. Run the Nodetool utility.

On Linux

Note that .jar is separated by the colon ":" punctuation mark.

```
java -cp libthrift-0.7.0.jar:cassandra-thrift-Cassandra version.jar:commons-
cli-1.1.jar:cassandra-all-Cassandra version.jar org.apache.cassandra.tools.NodeCmd -h
Cassandra host -p jmx_port nodetool command
```

On Windows

Note that .jar is separated by the semicolon ";" punctuation mark.

```
java -cp libthrift-0.7.0.jar;cassandra-thrift-Cassandra version.jar;commons-
cli-1.1.jar;cassandra-all-Cassandra version.jar org.apache.cassandra.tools.NodeCmd -h
Cassandra host -p jmx_port nodetool command
```

where...

Cassandra version could be either 1.1.6 or 1.1.12 depending on Feature Server version installed. Please check the actual cassandra-all-*.jar file name in current directory.

Cassandra host is the hostname or IP address where Feature Server is running.

jmx_port is the JMX port number that Feature Server started with, defined in launcher.xml. Default=9192.

nodetool command can be `ring [keyspace name]` or `removetoken <token>` or `repair` or `flush`.

For example, the following command line runs nodetool and sends the command `ring` to the Feature Server running on the local host:

```
java -cp libthrift-0.7.0.jar:cassandra-thrift-1.1.12.jar:commons-cli-1.1.jar:cassandra-
all-1.1.12.jar org.apache.cassandra.tools.NodeCmd -h localhost -p 9192 ring sipfs
```

Running Nodetool with Cassandra JMX Authentication

If Cassandra JMX authentication is enabled, then the nodetool command requires JMX username and password as command line arguments:

On Linux

Note that .jar is separated by the colon ":" punctuation mark.

```
java -cp libthrift-0.7.0.jar:cassandra-thrift-Cassandra version.jar:commons-
cli-1.1.jar:cassandra-all-Cassandra version.jar org.apache.cassandra.tools.NodeCmd -h
Cassandra host -p jmx_port -u username -pw password nodetool command
```

On Windows

Note that .jar is separated by the semicolon ";" punctuation mark.

```
java -cp libthrift-0.7.0.jar;cassandra-thrift-Cassandra version.jar;commons-
cli-1.1.jar;cassandra-all-Cassandra version.jar org.apache.cassandra.tools.NodeCmd -h
Cassandra host -p jmx_port -u username -pw password nodetool command
```

Running Nodetool with Secured Cassandra JMX

If Cassandra JMX port is secured using SSL/TLS, then use the following command to run the nodetool commands such as ring or repair:

> **Important**
> This command is supported in Feature Server release 8.1.202.04 and later.

On Linux

Note that .jar is separated by the colon ":" punctuation mark.

```
java -Dssl.enable=true -Dcom.sun.management.jmxremote.ssl.need.client.auth=true
-Dcom.sun.management.jmxremote.ssl=true -Dcom.sun.management.jmxremote.registry.ssl=true
-Djavax.net.ssl.keyStore=<keyStore_file>
-Djavax.net.ssl.keyStorePassword=<keyStore_password_file>
-Djavax.net.ssl.trustStore=<cert_store_file>
-Djavax.net.ssl.trustStorePassword=<cert_store_password_file> -cp
libthrift-0.7.0.jar:cassandra-thrift-1.1.12.jar:commons-cli-1.1.jar:cassandra-
all-1.1.12.jar:fs-nodetool-utility-fs-9-SNAPSHOT.jar
com.genesyslab.nodetool.utility.NodeCmdCustom -h Cassandra host -p jmx_port nodetool command
```

On Windows

Note that .jar is separated by the semicolon ";" punctuation mark.

```
java -Dssl.enable=true -Dcom.sun.management.jmxremote.ssl.need.client.auth=true
-Dcom.sun.management.jmxremote.ssl=true -Dcom.sun.management.jmxremote.registry.ssl=true
-Djavax.net.ssl.keyStore=<keyStore_file>
-Djavax.net.ssl.keyStorePassword=<keyStore_password_file>
-Djavax.net.ssl.trustStore=<cert_store_file>
-Djavax.net.ssl.trustStorePassword=<cert_store_password_file> -cp
libthrift-0.7.0.jar;cassandra-thrift-1.1.12.jar;commons-cli-1.1.jar;cassandra-
all-1.1.12.jar;fs-nodetool-utility-fs-9-SNAPSHOT.jar
com.genesyslab.nodetool.utility.NodeCmdCustom -h Cassandra host -p jmx_port nodetool command
```

Running nodetool repair on all Feature Server hosts in a Cassandra cluster

On Linux

```
cd genesys/fs/work/jetty-0.0.0.0-8080-fs.war-_fs-any-/webapp/WEB-INF/lib

java -cp libthrift-0.7.0.jar:cassandra-thrift-1.1.12.jar:commons-cli-1.1.jar:cassandra-
all-1.1.12.jar org.apache.cassandra.tools.NodeCmd -h localhost -p 9192 repair -pr
```

On Windows

```
cd genesys\fs\work\jetty-0.0.0.0-8080-fs.war-_fs-any-\webapp\WEB-INF\lib

java -cp libthrift-0.7.0.jar;cassandra-thrift-1.1.12.jar;commons-cli-1.1.jar;cassandra-
all-1.1.12.jar org.apache.cassandra.tools.NodeCmd -h localhost -p 9192 repair -pr
```

Switching Snitches

To change an **endpoint_snitch** from **SimpleSnitch** to **PropertyFileSnitch** or **GossipingPropertyFileSintch**, follow these steps:

1. Stop all SIP Feature Server instances.
2. Change **replicationOptions** and **replicationStrategyClassName** in the Cassandra section of the SIP Feature Server application, as required.
3. Create **cassandra-topology.properties** or **cassandra-rackdc.properties** file based on **PropertyFIleSnitch** or **GossipingPropertyFileSnitch** properties, respectively.
4. Copy the above-mentioned property files into the **<FS installed location>/resources** directory.
5. Update the **endpoint_snitch** value in **cassandra.yaml** file in each Cassandra node to reflect the required snitch value (**PropertyFIleSnitch** or **GossipingPropertyFileSnitch**).
6. Restart the master instance first, and then restart all other SIP Feature Server instances.
7. Run the `nodetool` repair commands. For details, refer here.

Switching Seeds

To change the master instance, you must change the seed value inside the master and non-master nodes.

> **Important**
> The seed value is the FQDN/IP address of the master instance.

Steps to switch seeds:

1. Stop all SIP Feature Server instances.
2. In the current master SIP Feature Server application, under the **Cluster** section, change the master value to `false`.
3. In the new master SIP Feature Server application, under the **Cluster** section, change the master value to `true`.
4. Edit the **cassandra.yaml** file for each SIP Feature Server instance and change the seed value. The seed value is the FQDN/IP address of the master instance.
5. Perform these changes sequentially one instance at a time.
6. Restart the master instance first, and then restart all other SIP Feature Server instances.
7. Run the `nodetool` repair commands. For details, refer here.

Performing maintenance after removing a node from the cluster

When you remove a node from the ring, you must also remove the corresponding tokens. Use these software procedures:

1. Run nodetool `ring` to obtain tokens. (See Running Nodetool on a Feature Server Host)
2. Run nodetool `removetoken` to remove nodes from the ring.
   Note: Only nodes that are down can be removed.
3. Run nodetool `ring` to validate removal.
4. Run nodetool `repair` on all Feature Server hosts in the cluster.
5. Run nodetool `ring` to validate repair.

For example, run nodetool `ring` to obtain tokens of nodes to be removed:

```
java -cp libthrift-0.7.0.jar:cassandra-thrift-1.1.12.jar:commons-cli-1.1.jar:cassandra-
all-1.1.12.jar org.apache.cassandra.tools.NodeCmd -h localhost -p 9192 ring sipfs
```

...to see the following output returned tokens of nodes which are down:

```
<fs-host-IP1> DC1 RAC1 Up Normal 1.17 MB100.00% 167086018864645871692761019448293152722
<fs-host-IP2> DC1 RAC2 Up Normal 1.29 MB 100.00% 260037876766820018229186112944720563l6
<fs-host-IP3> DC2 RAC1 Down Normal 1.15 MB 100.00% 41007983964572150951275225962045789866
<fs-host-IP4> DC2 RAC2 Down Normal 1.16 MB 100.00% 53685600614278234503162023330018045221
```

The following nodetool commands remove <fs-host-IP3> and <fs-host-IP4> nodes from the ring:

```
java -cp libthrift-0.7.0.jar:cassandra-thrift-1.1.12.jar:commons-cli-1.1.jar:cassandra-
all-1.1.12.jar org.apache.cassandra.tools.NodeCmd -h localhost -p 9192 removetoken
41007983964572150951275225962045789866

java -cp libthrift-0.7.0.jar:cassandra-thrift-1.1.12.jar:commons-cli-1.1.jar:cassandra-
all-1.1.12.jar org.apache.cassandra.tools.NodeCmd -h localhost -p 9192 removetoken
53685600614278234503162023330018045221
```

After token removal, running nodetool ring should provide the following output:

```
<fs-host-IP1> DC1 RAC1 Up Normal 1.17 MB 100.00% 167086018864645871692761019448293152722
<fs-host-IP2> DC1 RAC2 Up Normal 1.29 MB 100.00% 260037876766820018229186112944720563l6
```