SIP Feature Server Deployment Guide

Deploying Cassandra

5/4/2025

Contents

Deploying Cassandra

The Apache Cassandra database is an open source NoSQL database, which is easily scalable and provides high availability without compromising performance.

> **Important**
>
> Instructions provided in this chapter are complimentary to installation guidelines of the Cassandra database. If you are using a commercial version then you must follow the instructions given in official Cassandra documentation. If you're using the community version, you can follow the steps described here.
> To use Cassandra 4.X, you must install SIP Feature Server version 8.1.203 or later.

Ensure that you have a minimum of two nodes per data center and the clocks on all Cassandra nodes are synchronized.

Prerequisites to use Cassandra

1. The external Cassandra cluster deployed can be co-located/shared with other Genesys components. However, the SIP Feature Server keyspaces (global and regional) should not be shared with other components.
2. The minimum disk space required for SIP Feature Server keyspaces can be computed using the disk sizing tool mentioned in the Hardware and software prerequisites page.
3. While sharing Cassandra with multiple components, disk requirement is computed by summing the minimum required space from all components.

Selecting a Seed node

Seed node is a comma-delimited list of IP addresses used by gossip for bootstrapping new nodes joining a cluster. In multiple data-center clusters, the seed list must include at least one node from each data center (replication group). More than a single seed node per data center is recommended for fault tolerance. Otherwise, gossip has to communicate with another data center when bootstrapping a node. Making every node a seed node is not recommended because of increased maintenance and reduced gossip performance. Gossip optimization is not critical, but Genesys recommends that you use a small seed list. Usually, the first node installed in each data center is considered as a seed node. For more information on seed nodes and gossip, see Internode communications (gossip).

Deploying Cassandra

The following steps show how to deploy each Cassandra node in the co-located/external Cassandra cluster:

1. Download the latest version in Apache Cassandra 2.2, 3.x, or 4.x from either the Cassandra archive

index page or from the Downloading Cassandra page.

2. Extract the downloaded zip file to any desired directory (Cassandra installed directory).
3. Edit the **<Cassandra installed directory>\conf\cassandra.yaml** file and configure the parameters:
   - `cluster_name` : `FeatureServerCluster`
   - `start_rpc` : `true`
   - `listen_address` : IP or FQDN of the node
   - `rpc_address` : IP or FQDN of the node
   - `seeds` : comma separated IP or FQDN of the seed nodes
   - `storage_port` : 7000 (default value)
   - `ssl_storage_port` : 7001 (default value)
   - `native_transport_port`: 9042 (default value)
   - `endpoint_snitch` : `PropertyFileSnitch`
4. Edit the **<Cassandra installed directory>\conf\cassandra-topology.properties** file and configure the data centers as follows:
   - The following example shows a single data center with two Cassandra nodes:
     - `Cassandra node 1 IP or FQDN 1=data center 1 name:RAC1`
     - `Cassandra node 2 IP or FQDN 2=data center 1 name:RAC2`
   - The following example shows a multi data center with four Cassandra nodes with two nodes per data centre:
     - `Cassandra node 1 IP or FQDN 1=data center 1 name:RAC1`
     - `Cassandra node 2 IP or FQDN 2=data center 1 name:RAC2`
     - `Cassandra node 3 IP or FQDN 3=data center 2 name:RAC1`
     - `Cassandra node 4 IP or FQDN 4=data center 2 name:RAC2`

> **Important**
>
> If you want to connect SIP Feature Server to Cassandra 2.2.X or 3.X using the legacy **thrift** protocol, enable the port **rpc_port 9160** during your Cassandra installation.

Configuring Cassandra logging

By default, the Cassandra logs are generated under **<Cassandra installed directory>\logs**. To change the log file directory:

- On Linux, edit the **<Cassandra installed directory>\bin\cassandra** file and update the parameter:
  - `-Dcassandra.logdir=$CASSANDRA_HOME/logs`
- On Windows, edit the **<Cassandra installed directory>\bin\cassandra.bat** file and update the parameter:

- `Dcassandra.logdir="%CASSANDRA_HOME%\logs`

To configure Cassandra logging, see Configuring logging.

Configuring Cassandra Authentication and Authorization

1. On each Cassandra node, edit the **<Cassandra installed directory>\conf\cassandra.yaml** file and set the follwoing parameters and restart the nodes:
   - `authenticator` :`PasswordAuthenticator`
   - `authorizer` :`CassandraAuthorizer`
2. On the Master Cassandra node, navigate to **<Cassandra installed directory>\bin** and run the CQL client as follows:
   - On Linux
     - `./cqlsh <IP or FQDN of the Cassandra node configured in Cassandra.yaml> <CQL Port>`
   - On Windows
     - `cqlsh <IP or FQDN of the Cassandra node configured in Cassandra.yaml> <CQL Port>`
   - By default, the CQL Port is 9042.
3. Increase the replication factor of the “system_auth”(Pre-defined keyspace in Cassandra 2.2 or higher) keyspace by using the CQL query:
   - `alter keyspace system_auth with replication = {'class': 'NetworkTopologyStrategy', <replication factor>};`
     For example, if Cassandra cluster is configured with two Data centers (DC1 & DC2) and each data center is configured with two nodes, then set the replication factor of `system_auth` keyspace as `DC1:2, DC2:2` by using the cql query as follows:
     - `alter keyspace system_auth with replication = {'class': 'NetworkTopologyStrategy', 'DC1': 2, 'DC2': 2};`
4. Create a user using the following cql query:
   - create user <user_name> with password <password>;
5. Create a role by using the following cql query:
   - create role <role_name>;
6. SIP Feature Server must be authorized to create and access keyspace, hence grant all permissions to the role name
   - grant all permissions on all keyspaces to <role_name>;
7. Grant access to the created user.
   - grant <role_name> to <user_name>;
8. If the Cassandra cluster is used for any other Genesys components, revoke the access rights after master SIP Feature Server is started for the first time, and grant all permissions only to the SIP Feature Server keyspace (sipfs) by using the following query:
   - revoke all permissions on all keyspaces from <role_name>;

- grant all permissions on keyspace sipfs to <role_name>;

9. If regional keyspace is created, then grant permissions to regional keyspace as well.
   - grant all permissions on keyspace <regional keyspace name> to <role_name>;

Configuring Cassandra SSL

The following steps show how to enable secure connection (SSL) for each Cassandra node:

1. Generate server certificates and keystore. For more information on certificate generation and installation, see Genesys Security Deployment Guide.
2. Copy the keystore file generated during certificate installation to the **<Cassandra Installed directory>\bin**.
3. Update the **Client-to-node** option in the **Cassandra.yaml** file as follows:

```
client_encryption_options
        enabled: true

        optional: false

        keystore: <keystore_name>

        keystore_password: <keystore_password>
```

4. Update the **node-to-node** option in the **Cassandra.yaml** file as follows:

```
internode_encryption: all
        keystore: <keystore name>

        keystore_password: <keystore_password>

        truststore: <truststore name>

        truststore_password: <truststore_password>

        require_client_auth: true|false

        protocol: (Default: TLS)
```

5. Restart Cassandra node.

Monitoring Cassandra

Linux

Configuring Cassandra as a Service

1. Create the /etc/init.d/cassandra startup script.
2. Edit the contents of the file:

```
#!/bin/sh
 #
```

```
# chkconfig: - 80 45
# description: Starts and stops Cassandra
# update daemon path to point to the cassandra executable
DAEMON=<Cassandra installed directory>/bin/cassandra
start() {
        echo -n "Starting Cassandra... "
        $DAEMON -p /var/run/cassandra.pid
        echo "OK"
        return 0
}
stop() {
        echo -n "Stopping Cassandra... "
        kill $(cat /var/run/cassandra.pid)
        echo "OK"
        return 0
}
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart)
        stop
        start
        ;;
  *)
        echo $"Usage: $0 {start|stop|restart}"
        exit 1
esac
exit $?
```

3. Make the file executable:

   ```
   sudo chmod +x /etc/init.d/cassandra
   ```

4. Add the new service to the list:

   ```
   sudo chkconfig --add cassandra
   ```

5. Now you can manage the service from the command line:

   ```
   sudo /etc/init.d/cassandra start
    sudo /etc/init.d/cassandra stop
   ```

Start/Stop Cassandra

Start the seed node and then start the rest of the Cassandra nodes sequentially:

```
Start: sudo service cassandra start
Stop: sudo service cassandra stop
```

Verifying Cassandra

After you have deployed Cassandra Cluster, you may want to verify that all the nodes can communicate with each other. To do this, execute the following command on any Cassandra host:

```
cd <Cassandra installed directory>/bin
./nodetool status
```

Windows

Monitoring Cassandra as a service

Currently this is supported if Cassandra is deployed in a windows machine. First we need configure Cassandra to run as a service and then configure the service as third party application in GA to control using SCI.

Configuring Cassandra as a Service

1. Download the latest apache commons daemon from Apache Commons Project Distributions.
2. Extract the commons daemon in **<Cassandra installed directory>\bin**.
3. Rename the extracted folder as daemon.
4. Add `<Cassandra installed directory>` as `CASSANDRA_HOME` in windows environment variable.
5. Edit the **cassandra.yaml** file in **<Cassandra installed directory>\conf** and uncomment the `data_file_directories`, `commitlog_directory`, `saved_cache_directory` and set the absolute paths.
6. Edit **cassandra.bat** in **<Cassandra installed directory>\bin** and replace the value for the `PATH_PRUNSRV` as follows:
   - for 32 bit windows, set `PATH_PRUNSRV=%CASSANDRA_HOME%\bin\daemon\`
   - for 64 bit windows, set `PATH_PRUNSRV=%CASSANDRA_HOME%\bin\daemon\amd64\`
7. Change the service name for `SERVICE_JVM` as required in **cassandra.bat**.
   - For example, setting `SERVICE_JVM="cassandra_228"` creates the windows service in the `cassandra_228` name.
8. With administrator privileges, run **cassandra.bat install** from command prompt.

This creates a Windows Service.

Configure Cassandra Service in GA

1. Create a new application object with application template **ThirdPartyServer**.
2. Under Server Info, provide the host in which the Cassandra node is running.
3. Under Start Info, provide the working directory as:

- for 32 bit windows, set `<Cassandra installed directory>\bin\daemon\`
- for 64 bit windows, set `<Cassandra installed directory>\bin\daemon\amd64\`

4. Under Start info, provide the command line as:
   - `prunsrv.exe`
5. Under Start info, provide the command line arguments as:
   - `//RS//<service name>`
6. In the **Annex** tab, create a new section called **start_stop**.
7. In the **start_stop** section, create an option **start_command** with value `net start <cassandra service>`.
8. In the **start_stop** section, create an option **stop_command** with value `net stop <cassandra service>`.

Now, monitor (start/stop) Cassandra from SCI.

Start/Stop Cassandra

Start the seed node first, followed by rest of the Cassandra nodes sequentially.

- Use SCI to start/stop the Cassandra nodes as mentioned in the Monitoring Cassandra as a service on Windows section earlier.

Verifying Cassandra

After you have deployed the Cassandra Cluster, you can verify that all the nodes communicate with each other. To do this, run the following command on any Cassandra host:

```
cd <Cassandra installed directory>\bin
nodetool status
```