Genesys Knowledge Center Deployment Guide

Cassandra Security

12/17/2025

Contents

Cassandra Security

Unauthorized access to Cassandra data is possible at several points:

- Direct access via "standard" interfaces: Thrift and CQL
- Access to data traveling through the network
- Access to data files that Cassandra stores on hard drives

Cassandra's default configuration provides mechanisms to secure direct interfaces (through authentication and authorization) and network traffic (through the use of TLS). The data stored on hard drives can be secured either by third-party commercial offerings or with some development investments.

Securing access interfaces

You can secure your access interfaces based on an authentication and authorization scheme. In other words, Cassandra needs to know:

- Who is trying to access the system
- Whether they are allowed to access the system at all
- If so, which data they should have access to

With the default setup, anybody is allowed to access all the data.

Authentication

Authentication (who) is managed by the authenticator parameter in the `cassandra.yaml` file.

Procedure

Start

1. Locate Cassandra configuration file *Cassandra installation directory/conf/cassandra.yaml*.
2. Change the authenticator option in the cassandra.yaml file to PasswordAuthenticator.

By default, the authenticator option is set to AllowAllAuthenticator.
`authenticator: PasswordAuthenticator`

1. Increase the replication factor for the system_auth keyspace to N (number of nodes).

If you use the default, 1, and the node with the lone replica goes down, you will not be able to log into the cluster because the system_auth keyspace was not replicated.

1. Restart the Cassandra client.

2. Start cqlsh using the superuser name and password.

```
./cqlsh -u cassandra -p cassandra
```

1. Create another superuser, not named cassandra. This step is optional but highly recommended.
2. Log in as that new superuser.
3. Change the cassandra user password to something long and incomprehensible, and then forget about it. It won't be used again.
4. Take away the cassandra user's superuser status.
5. Use the CQL statements listed previously to set up user accounts and then grant permissions to access the database objects.
6. Set the new user name and password to the values of the cassandra-keyspace **userName** and **password** options for the Knowledge Center Cluster application.

End

For more information about permissions see:

- Apache Cassandra Authentication
- Apache Cassandra Authorization

Knowledge Center Cluster Configuration

Prerequisites

The Knowledge Center Cluster applications are created and configured

Procedure

Start

For all Cassandra Resource Access Points:

1. Open the Knowledge Center Cluster configuration option, cassandra-keyspace section.
2. Set the userName option to the name of an already-created user.
3. Set the password option to the user's password.

End

Securing Network Traffic

The client-to-node and node-to-node traffic in your Cassandra deployment may require protection. They can both be secured by using SSL (Secure Sockets Layer) encryption.

Client-to-Node Encryption

Client-to-node encryption uses SSL to protect data that is traveling from client machines (Knowledge Center CMS nodes) to a database cluster. It does this by establishing a secure channel between the client and the coordinator node.

Prerequisites

- You must install Java Cryptography Extension (to enable 256-bit encryption).
- All nodes must have all of the relevant SSL certificates. See Preparing server certificates (Cassandra documentation).

> **Important**
>
> The Oracle Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 6 must be installed when enabling client-to-node encryption.
>
> 1. Download the JCE:
>
> - JAVA 8
> - JAVA 7
> - JAVA 6
>
> - Unzip the downloaded file
> - Place the two jars from the zip file into `<java_jre_install_dir>/lib/security/` if running the jre or `<java_jdk_install_dir>/jre/lib/security` if running the jdk

Start

1. Configuring Cassandra nodes:
    a. On each Cassandra node edit Cassandra installation directory/conf/cassandra.yaml
    b. Set the following options under the client_encryption_options section:

```
# enable or disable client/server encryption.
 client_encryption_options:
     enabled: true
     optional: false
     keystore: <path to your JKS keystore, for example  c:\genesys\keystore.jks >
     keystore_password:  <password for JKS keystore>
     require_client_auth: false
     truststore: <path to your JKS trustore, for example  c:\genesys\truststore.jks>
     truststore_password: <password for JKS truststore>
     protocol: TLS
     algorithm: SunX509
     store_type: JKS

 cipher_suites:[TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA,TLS_DHE_RSA_WITH_AES_128_CBC_SHA,TLS_DHE_RSA_WITH_AES_256_CBC_SHA,TLS_ECDHE_
```

3. Configuring Genesys Knowledge Center CMS:
    a. Navigate to the Application in Genesys Administrator
    b. Open Genesys Knowledge Center Cluster object
    c. Navigate to options tab
    d. Set the following options in the cassandra-security section:
        - enable-ssl = true
        - truststore-path = <path to your JKS trustore, for example c:\genesys\truststore.jks>
        - truststore-password = <password for JKS truststore>

> **Important**
>
> You can define truststore-path and truststore-password options in the Genesys Knowledge Center CMS application options in case you use different paths and passwords on every host.

    e. Open Cassandra Resource Access Point application object
    f. Open the properties for the port with an ID of default.
    g. Set this port to secured.

End

Node-to-Node Encryption

Node-to-node encryption uses SSL to protect data being transferred between cluster nodes. This includes node-to-node gossip communication.

Prerequisites

- You must install Java Cryptography Extension (to enable 256-bit encryption).
- All nodes must have all of the relevant SSL certificates. See Preparing server certificates.

Procedure

Start

1. On each Cassandra node edit Cassandra installation directory/conf/cassandra.yaml
2. Set the following options under the server_encryption_options section:

```
server_encryption_options:
    internode_encryption: all
                          # all—Cassandra encrypts all internodal traffic
                          # dc—Cassandra encrypts all traffic between datacenters
                          # rack—Cassandra encrypts all traffic between racks
    keystore:<path to your JKS keystore, for example  c:\genesys\keystore.jks>
    keystore_password:<password for JKS keystore>
    truststore:<path to your JKS trustore, for example  c:\genesys\truststore.jks >
    truststore_password:<password for JKS truststore>
    protocol: TLS
    algorithm: SunX509
    store_type: JKS
    cipher_suites: [TLS_DHE_DSS_WITH_AES_128_CBC_SHA, TLS_DHE_RSA_WITH_AES_128_CBC_SHA, TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA]
    require_client_auth: false
```

End