Configuring Kerberos with DataStax Enterprise

Updated: 2019-01-19-08:00

Table of Contents

Configuring Kerberos with DataStax Enterprise

This tutorial is intended for anyone interested in enabling Kerberos authentication in DataStax Enterprise and OpsCenter. It provides step-by-step instructions on configuring DataStax Enterprise as Kerberos clients.

Goals of the tutorial

At the completion of this tutorial you will have a DataStax Enterprise cluster that authenticates principals against a Kerberos realm. You will install the correct client libraries on each node and configure DataStax Enterprise to connect to the Kerberos server to perform authentication.

Before you start this tutorial

You must:

- Have a basic understanding of how Kerberos works
- Have a running Kerberos server and the proper administration permissions
- Know your Kerberos realm name, and the fully-qualified domain names of all the nodes in your DataStax Enterprise cluster
- Have the ability to install software on the machines on which you want to use Kerberos and DataStax Enterprise
- Be familiar with running commands from a Unix terminal
- Have the following software installed:
  # DataStax Enterprise 4.7 or later.
  # A Kerberos 5 server configured for your Kerberos realm.
  # NTP configured on each node of your cluster.
  # DNS configured on each node of your cluster.

  You will install the following software while completing the steps in this tutorial:

  # Kerberos 5 client libraries for your operating system.

Resources for setting up a Kerberos realm

This tutorial doesn't document how to set up a Kerberos realm. If you need to set up a Kerberos realm, see the following resources:

- Setting Up MIT Kerberos 5 for Debian and Ubuntu
- Setting up Kerberos on Red Hat Enterprise Linux for Red Hat and other compatible distributions that use Yum

Setting up your environment

Each node in your cluster requires DNS to be working properly, NTP to be enabled and the system time set, and the Kerberos client libraries installed.

Prerequisites:

You must have installed the required software as described in Before you start this tutorial *(page 4)*.

1. On each node, confirm DNS is working.

```
$ hostname
```

```
node1.example.com
```

2. On each node, confirm NTP is configured and running.

```
$ ntpq -p
```

```
     remote           refid      st t when poll reach   delay   offset
 jitter
==============================================================================
*li506-17.member 209.51.161.238   2 u  331 1024  377   80.289    1.384
  1.842
-tock.eoni.com   216.228.192.69   2 u  410 1024  377   53.812    1.706
 34.692
+time01.muskegon 64.113.32.5      2 u  402 1024  377   59.378   -1.635
  1.840
-time-a.nist.gov .ACTS.           1 u  746 1024  151  132.832   26.931
 55.018
+golem.canonical 131.188.3.220    2 u  994 1024  377  144.080   -1.732
 20.072
```

3. If you are using Oracle Java, make sure the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files are installed on each node.

   By default Kerberos uses the AES-256 cypher. Oracle Java does not include the AES-256 cypher by default due to export restrictions to certain countries. OpenJDK includes AES-256 because it is not export restricted.

   For general instructions on installing the JCE Unlimited Strength Policy Files for Oracle Java, see the DataStax Enterprise documentation *(page)*.

   If you are using Debian/Ubuntu and are using the webupd8 PPA repository to manage your Oracle Java 8 installations, install the Unlimited Strength Policy Files:

```
$ sudo apt-get install oracle-java8-unlimited-jce-policy
```

4. If you are using RedHat or CentOS, install the EPEL repository.

   ```
   $ sudo yum install epel-release
   ```

5. Install the Kerberos client packages on each node.

   RedHat and CentOS

   ```
   $ sudo yum install krb5-workstation krb5-libs krb5-pkinit-openssl
   ```

   Debian and Ubuntu

   ```
   $ sudo apt-get install krb5-user krb5-config krb5-pkinit
   ```

6. Copy the `krb5.conf` file from the Kerberos server to each node.

   The `krb5.conf` file contains configuration information for your Kerberos domain.

   From the Kerberos server or Kerberos Domain Controller (KDC):

   ```
   $ scp /etc/krb5.conf node1.example.com:/etc/
   ```

   Repeat these steps for each node in your cluster.

Adding Kerberos service principals for each node in a cluster

Add Kerberos principals for each node's Cassandra service and an additional principal for HTTP communication.

On the Kerberos Domain Controller (KDC), add the principals from each node using the `kadmin` command.

Prerequisites:

You must have:

- An existing Kerberos domain set up.
- An existing KDC running.
- Admin rights to the KDC.
- Installed and verified the software as described in Setting up your environment *(page 5)*.

1. On each node, note the fully qualified domain name (FQDN) of the machine.

```
$ hostname --fqdn
```

```
node1.example.com
```

2. On the KDC, run the `kadmin` command and then enter the Cassandra and HTTP users for each node, using the FQDN of each machine, to the domain using the `addprinc` command within `kadmin`.

   In this example of a 3-node cluster, the default Cassandra username of `cassandra` is used. The Kerberos domain name is `EXAMPLE.COM`

```
$ kadmin
addprinc -randkey cassandra/node1.example.com
addprinc -randkey HTTP/node1.example.com
addprinc -randkey cassandra/node2.example.com
addprinc -randkey HTTP/node2.example.com
addprinc -randkey cassandra/node3.example.com
addprinc -randkey HTTP/node3.example.com
```

   To verify that the principals have been added, run the `listprincs` command within `kadmin`:

```
listprincs
HTTP/node1.example.com@EXAMPLE.COM
HTTP/node2.example.com@EXAMPLE.COM
HTTP/node3.example.com@EXAMPLE.COM
cassandra/node1.example.com@EXAMPLE.COM
```

```
cassandra/node2.example.com@EXAMPLE.COM
cassandra/node3.example.com@EXAMPLE.COM
kadmin/admin@EXAMPLE.COM
```

3. Create a keytab file for each node, with the principals keys for that node, using the `ktadd` command in `kadmin`.

   The keytab file is used to store Kerberos principal keys. You must create the keytab file during the same `kadmin` session in which you created the service principals.

```
ktadd -k /tmp/node1.keytab cassandra/node1.example.com
ktadd -k /tmp/node1.keytab HTTP/node1.example.com
ktadd -k /tmp/node2.keytab cassandra/node2.example.com
ktadd -k /tmp/node2.keytab HTTP/node2.example.com
ktadd -k /tmp/node3.keytab cassandra/node3.example.com
ktadd -k /tmp/node3.keytab HTTP/node3.example.com
quit
```

4. Copy the node-specific keytab files from the KDC machine to the nodes.

```
$ scp /tmp/node1.keytab cassandra@node1.example.com:/etc/dse/
$ scp /tmp/node2.keytab cassandra@node2.example.com:/etc/dse/
$ scp /tmp/node3.keytab cassandra@node3.example.com:/etc/dse/
```

5. On each node, change the name of the keytab file to `dse.keytab`.

   Make the file names the same across all the nodes for consistency, and so that the entry in each node's dse.yaml is the same.

   The location of the dse.yaml *(page)* file depends on the type of installation:

Installer-Services	`/etc/dse/dse.yaml`
Package installations	`/etc/dse/dse.yaml`
Installer-No Services	*`install_location`*`/resources/dse/conf/dse.yaml`
Tarball installations	*`install_location`*`/resources/dse/conf/dse.yaml`

```
$ hostname --fqdn
node1.example.com
$ mv /etc/dse/node1.keytab /etc/dse/dse.keytab
```

6. Change the permissions on `dse.keytab` so that only the `cassandra` user can read and write to the keytab file.

```
$ sudo chown cassandra:cassandra /etc/dse/dse.keytab
```

```
$ sudo chmod 600 /etc/dse/dse.keytab
```

Configuring DataStax Enterprise for Kerberos authentication

The cassandra.yaml and dse.yaml files must be edited on each node to enable Kerberos authentication. Add the Kerberos authenticator to `cassandra.yaml` and add the Kerberos options to `dse.yaml`.

The location of the dse.yaml *(page)* file depends on the type of installation:

Installer-Services	`/etc/dse/dse.yaml`
Package installations	`/etc/dse/dse.yaml`
Installer-No Services	*`install_location`*`/resources/dse/conf/dse.yaml`
Tarball installations	*`install_location`*`/resources/dse/conf/dse.yaml`

The location of the cassandra.yaml *(page)* file depends on the type of installation:

Package installations	`/etc/dse/cassandra/cassandra.yaml`
Tarball installations	*`install_location`*`/resources/cassandra/conf/cassandra.yaml`

1. On each node, edit the `cassandra.yaml` file to set the authenticator to `com.datastax.bdp.cassandra.auth.KerberosAuthenticator`.

   ```
   authenticator: com.datastax.bdp.cassandra.auth.KerberosAuthenticator
   ```

2. Make sure the `rpc_address` and `listen_address` options in `cassandra.yaml` are set to the IP address or hostname that matches the hostname in DNS, not localhost.

   ```
   rpc_address: 1.2.3.4
   listen_address: 1.2.3.4
   ```

3. On each node, edit the `dse.yaml` file and enter the correct Kerberos options to enable authentication.

   The options are located in the `kerberos_options` section.

   ```
   kerberos_options:
      keytab: /etc/dse/dse.keytab
      service_principal: cassandra/_HOST@EXAMPLE.COM
      http_principal: HTTP/_HOST@EXAMPLE.COM
      qop: auth
   ```

The `_HOST` variable is used in `dse.yaml`, and will be replaced correctly by DSE.

What's next:

To test your configuration, configure `cqlsh` to use Kerberos authentication *(page)* as described in the DataStax Enterprise documentation and connect to your cluster.

Test Kerberos authentication with cqlsh

Use the `cqlsh` tool to authenticate to DataStax Enterprise using a Kerberos principal. To use Kerberos authentication with `cqlsh`, create a `cqlshrc` file and configure the options for your Kerberos realm.

1. On the KDC server, add the user principals in `kadmin` using the `addprinc` command.

   ```
   $ kadmin
   addprinc jane
   ```

   In Kerberos, there's a difference between a service principal and a user principal. Typically, user principals have the form *`username@Kerberos realm name`*, while service principals have the form *`servicename/hostname@Kerberos realm name`*. For example, a user principal is `jane@EXAMPLE.COM`, while a service principal is `cassandra/node1.example.com@EXAMPLE.COM`.

   Do not confuse the default `cassandra` database superuser with the `cassandra` Unix user that corresponds with the `cassandra` Kerberos service principals (for example `cassandra/node1.example.com@EXAMPLE.COM`) used in this tutorial.

2. On the DataStax Enterprise node where you will run `cqlsh`, add the user principals to the `system_auth.users` table.

   a. Temporarily disable Kerberos authentication in cassandra.yaml and restart the node.

      In `cassandra.yaml`:

      ```
      # authenticator:
       com.datastax.bdp.cassandra.auth.KerberosAuthenticator
      authenticator: PasswordAuthenticator
      authorizer: CassandraAuthorizer
      ```

      Restart the node:

      ```
      $ sudo service dse restart
      ```

   b. Create a new superuser with the same name as the user principal.

      ```
      $ cqlsh
      cqlsh> create user 'jane@EXAMPLE.COM' SUPERUSER;
      ```

      The Cassandra username must match the full user principal name, including the Kerberos realm.

   c. Re-enable the Kerberos authenticator in `cassandra.yaml`.

```
authenticator:
 com.datastax.bdp.cassandra.auth.KerberosAuthenticator
```

d. Restart the node.

```
$ sudo service dse restart
```

3. On the DataStax Enterprise node where you will run `cqlsh`, install the Python dependencies for `cqlsh` Kerberos authentication.

RedHat and CentOS

```
$ sudo yum install python-pip
$ sudo pip install pure-sasl
$ sudo yum install python27-kerberos
```

Note: You must use the `python27-kerberos` package from the DataStax RPM repository. The `python-kerberos` package from the main RPM repositories will not work with `cqlsh`.

Debian and Ubuntu

```
$ sudo apt-get install python-pip
$ sudo pip install pure-sasl
$ sudo apt-get install python-kerberos
```

4. Create a `cqlshrc` file based on the sample file included with DataStax Enterprise.

Package installs

```
$ mkdir ~/.cassandra
$ cp /usr/share/doc/dse-libcassandra-4.8.3/cqlshrc.sample.kerberos
 ~/.cassandra/cqlshrc
```

Tarball installs

```
$ mkdir ~/.cassandra
$ cp DSE_HOME/resources/cassandra/conf/cqlshrc.sample.kerberos
 ~/.cassandra/cqlshrc
```

5. Edit `cqlshrc` and set the options according to your cluster and Kerberos realm.

Set the `hostname` option in the `[connection]` section to the hostname of the node. In the `[kerberos]` section set the `hostname` option to the hostname of the node and set the `principal` option to the name of the user principal you created.

```
[connection]
hostname = node1.example.com
port = 9042

[kerberos]
hostname = node1.example.com
service = cassandra
; optional
principal = jane@EXAMPLE.COM
```

6. Get a Kerberos ticket for your user principal.

```
$ kinit jane
Password for jane@EXAMPLE.COM:
$ klist
Ticket cache: FILE:/tmp/krb5cc_1000
Default principal: jane@EXAMPLE.COM

Valid starting       Expires              Service principal
12/14/2015 19:18:36  12/15/2015 05:18:36  krbtgt/
EXAMPLE.COM@EXAMPLE.COM
 renew until 12/21/2015 19:18:34
```

7. Start `cqlsh`.

```
$ cqlsh
Connected to Test Cluster at node1.example.com:9042.
[cqlsh 5.0.1 | Cassandra 2.1.11.969 | DSE 4.8.3 | CQL spec 3.2.1 |
 Native protocol v3]
Use HELP for help.
cqlsh>
```

Configuring OpsCenter for Kerberos authentication

After configuring DSE for Kerberos authentication, complete the procedure for configuring OpsCenter for Kerberos authentication *(page)*.

When configuring OpsCenter for Kerberos authentication, create and configure the OpsCenter principals first, and then add the cluster to OpsCenter.