## **DataDemics**

## **SSL configuration**

1.Create the directory for the certificates:

|  |
| --- |
| $ mkdir -p /opt/cloudera/security/x509/ /opt/cloudera/security/jks/  $ cd /opt/cloudera/security/jks  Use chmod/chown to change ownership of the /opt/cloudera/security/jks directory to give Cloudera Manager access to the directory.  $chown -R cloudera-scm:cloudera-scm /opt/cloudera/security/jks |

2.Generate the key pair and self-signed certificate, and store these in the keystore (example.keystore). Set -keypass to the same value as -storepass (Cloudera Manager does not support separate values for keypass and storepass.)

For CN, use the FQDN of the Cloudera Manager Server host to avoid a java.io.IOException: HTTPS hostname wrong exception.

|  |
| --- |
| $keytool -genkeypair -keystore example.keystore -keyalg RSA -alias cmhost \  -dname "CN=ip-10-0-0-190.ec2.internal" -storepass password -keypass password |

3.Copy the default Java truststore (cacerts) to the alternate system truststore (jssecacerts). Self-signed certificates are appended to jssecacerts without modifying the default cacerts file.

|  |
| --- |
| $ cp /usr/java/jdk1.7.0\_67-cloudera/jre/lib/security/cacerts /usr/java/jdk1.7.0\_67-cloudera/jre/lib/security/jssecacerts |

4.Export the certificate from the keystore (example.keystore).

|  |
| --- |
| $ keytool -export -alias cmhost -keystore example.keystore -rfc -file selfsigned.cer |

5.Copy the self-signed certificate (selfsigned.cer) to the /opt/cloudera/security/x509/ directory

|  |
| --- |
| $ cp selfsigned.cer /opt/cloudera/security/x509/cmhost.pem |

6.Import the public key into the alternate system truststore (jssecacerts), so that any process that runs with Java on this machine will trust the key. The default password for the Java truststore is changeit.

|  |
| --- |
| $ keytool -import -alias cmhost -file /opt/cloudera/security/jks/selfsigned.cer \  -keystore /usr/java/jdk1.7.0\_67-cloudera/jre/lib/security/jssecacerts -storepass changeit |

Important: Repeat this process on each machine in the cluster

7.Rename the keystore, such as from example.keystore to cmhost-keystore.jks:

|  |
| --- |
| $ mv /opt/cloudera/security/jks/example.keystore /opt/cloudera/security/jks/cmhost-keystore.jks |

**The self-signed certificate set up is complete.**

https://www.cloudera.com/documentation/enterprise/5-6-x/topics/sg\_self\_signed\_tls.html

CM side configurations:

Enable HTTPS for the Cloudera Manager Admin Console and Specify Server Keystore Properties

1.Log into the Cloudera Manager Admin Console.

2.Select Administration > Settings.

3.Click the Security category.

4.Configure the following TLS settings:

i)Path to TLS Keystore File ====:The complete path to the keystore file. In the example, this path would be:

/opt/cloudera/security/jks/cmhost-keystore.jks

ii)Keystore Password: The password for keystore: password

iii)Use TLS Encryption for Admin Console :Check this box to enable TLS encryption for Cloudera Manager.

iv)save changes

CMS side configurations:

Specify SSL Truststore Properties for Cloudera Management Services

note:When enabling TLS for the Cloudera Manager UI, you must set the Java truststore location and password in the Cloudera Management Services configuration. If this is not done, roles such as the Host Monitor and Service Monitor will be unable to connect to Cloudera Manager and will not start.

i)TLS/SSL Client Truststore File Location:/usr/java/jdk1.7.0\_67-cloudera/jre/lib/security/jssecacerts

ii)TLS/SSL Client Truststore File Password:changeit

iii)restart CMS

iv)service cloudera-scm-server restart

level1:Configuring TLS Encryption for Cloudera Manager Agents

Level 2: Configuring TLS Verification of Cloudera Manager Server by the Agents

Level 3: Configuring TLS Authentication of Agents to the Cloudera Manager Server