**How to Configure TLS Encryption for Cloudera Manager**

When you configure authentication and authorization on a cluster, Cloudera Manager Server sends sensitive information over the network to cluster hosts, such as Kerberos keytabs and configuration files that contain passwords. To secure this transfer, you must configure TLS encryption between Cloudera Manager Server and all cluster hosts.

TLS encryption is also used to secure client connections to the Cloudera Manager Admin Interface, using HTTPS.

Cloudera Manager also supports TLS authentication. Without certificate authentication, a malicious user can add a host to Cloudera Manager by installing the Cloudera Manager Agent software and configuring it to communicate with Cloudera Manager Server. To prevent this, you must install certificates on each agent host and configure Cloudera Manager Server to trust those certificates.

This guide shows how to configure and enable TLS encryption and certificate authentication for Cloudera Manager. The provided examples use an internal certificate authority (CA) to sign all TLS certificates, so this guide also shows you how to establish trust with the CA. (For certificates signed by a trusted public CA, establishing trust is not necessary, because the Java Development Kit (JDK) already trusts them.)

## Generate TLS Certificates

**Important:**

* You must use the Oracle JDK keytool utility. Do not use other JDK (such as OpenJDK) command line tools for this procedure. If you have multiple JDKs, set the PATH variable such that the Oracle JDK is first. For example:

$ export JAVA\_HOME=/usr/java/jdk1.7.0\_67-cloudera

$ export PATH=$JAVA\_HOME/bin:$PATH

* Use the same password for the -keypass and -storepass values. Cloudera Manager does not support using different passwords for the key and keystore.

Before configuring Cloudera Manager Server and all Cloudera Manager Agents to use TLS encryption, generate the server and agent certificates:

### Generate the Cloudera Manager Server Certificate

The following procedure assumes that an internal certificate authority (CA) is used, and shows how to establish trust for that internal CA. If you are using a trusted public CA (such as Symantec, GeoTrust, Comodo, and others), you do not need to explicitly establish trust for the issued certificates, unless you are using an older JDK and a newer public CA. Older JDKs might not trust newer public CAs by default.

1. On the Cloudera Manager Server host, create the /opt/cloudera/security/pki directory:

$ sudo mkdir -p /opt/cloudera/security/pki

If you choose to use a different directory, make sure you use the same directory on all cluster hosts to simplify management and maintenance.

1. On the Cloudera Manager Server host, use the keytool utility to generate a Java keystore and certificate signing request (CSR). Replace the OU, O, L, ST, and C entries with the values for your environment. Use the same password for the -keypass and -storepass values. Cloudera Manager does not support using different passwords for the key and keystore.

Note: please set hostname before running this command, hostname should be fully qualified domain name (FQDN) like mac127.cybage.com

|  |
| --- |
| [root@mac127 pki]# hostname mac127.cybage.com  [root@mac127 pki]# hostname -f  mac127.cybage.com |

|  |
| --- |
| $ sudo keytool -genkeypair -alias $(hostname -f)-server -keyalg RSA -keystore \  /opt/cloudera/security/pki/$(hostname -f)-server.jks -keysize 2048 -dname \  "CN=$(hostname -f),OU=Engineering,O=Cybage,L=Pune,ST=MAHARASHTRA,C=IN" \  -storepass password -keypass password  $ sudo keytool -certreq -alias $(hostname -f)-server \  -keystore /opt/cloudera/security/pki/$(hostname -f)-server.jks \  -file /opt/cloudera/security/pki/$(hostname -f)-server.csr -storepass password \  -keypass password |

Check if csr and jks files have been created under **/opt/cloudera/security/pki**

|  |
| --- |
| [root@mac127 pki]# ls -l  total 8  -rw-r--r-- 1 root root 1109 May 23 09:31 mac127.cybage.com-server.csr  -rw-r--r-- 1 root root 2279 May 23 09:31 mac127.cybage.com-server.jks |

1. Submit the CSR file (for example, mac127.cyabage.com-server.csr) to your certificate authority to obtain a server certificate

Follow attached doc for new certificate request



1. On the Cloudera Manager Server host, copy the JDK cacerts file to jssecacerts:

|  |
| --- |
| sudo cp $JAVA\_HOME/jre/lib/security/cacerts $JAVA\_HOME/jre/lib/security/jssecacerts |

*Note: The Oracle JDK uses the jssecacerts file for its default truststore if it exists. Otherwise, it uses the cacerts file. Creating the jssecacerts file allows you to trust an internal CA without modifying the cacerts file that is included with the JDK.*

1. Import the root CA certificate into the JDK truststore. If you do not have the $JAVA\_HOME variable set, replace it with the path to the Oracle JDK.

|  |
| --- |
| sudo keytool -importcert -alias rootca -keystore $JAVA\_HOME/jre/lib/security/jssecacerts \  -file /opt/cloudera/security/pki/rootca.cert.pem -storepass changeit |

|  |
| --- |
| [root@mac127 pki]# sudo keytool -importcert -alias rootca -keystore $JAVA\_HOME/jre/lib/security/jssecacerts \  > -file /opt/cloudera/security/pki/rootca.cert.pem -storepass changeit  Owner: CN=VeriSign Trial Secure Server Root CA - G2, OU="For Test Purposes Only. No assurances.", O="VeriSign, Inc.", C=US  Issuer: CN=VeriSign Trial Secure Server Root CA - G2, OU="For Test Purposes Only. No assurances.", O="VeriSign, Inc.", C=US  Serial number: 168164a428ca12dfab12f19fb1b93554  Valid from: Wed Apr 01 05:30:00 IST 2009 until: Sun Apr 01 05:29:59 IST 2029  Certificate fingerprints:  MD5: E0:19:F5:FC:C0:9A:13:0E:38:B7:BF:0D:02:40:D3:C2  SHA1: 51:51:B8:63:8A:4C:1F:15:54:56:ED:37:C9:10:35:CA:D3:01:B9:36  SHA256: 89:DD:5C:3D:FE:28:13:87:45:1F:A3:A0:F7:8C:1A:B6:77:DB:18:63:9E:71:72:AD:B2:52:91:CF:BE:F7:8D:19  Signature algorithm name: SHA1withRSA  Version: 3  Extensions:  #1: ObjectId: 1.3.6.1.5.5.7.1.12 Criticality=false  0000: 30 5F A1 5D A0 5B 30 59 30 57 30 55 16 09 69 6D 0\_.].[0Y0W0U..im  0010: 61 67 65 2F 67 69 66 30 21 30 1F 30 07 06 05 2B age/gif0!0.0...+  0020: 0E 03 02 1A 04 14 8F E5 D3 1A 86 AC 8D 8E 6B C3 ..............k.  0030: CF 80 6A D4 48 18 2C 7B 19 2E 30 25 16 23 68 74 ..j.H.,...0%.#ht  0040: 74 70 3A 2F 2F 6C 6F 67 6F 2E 76 65 72 69 73 69 tp://logo.verisi  0050: 67 6E 2E 63 6F 6D 2F 76 73 6C 6F 67 6F 2E 67 69 gn.com/vslogo.gi  0060: 66 f  #2: ObjectId: 2.5.29.19 Criticality=true  BasicConstraints:[  CA:true  PathLen:2147483647  ]  #3: ObjectId: 2.5.29.15 Criticality=true  KeyUsage [  Key\_CertSign  Crl\_Sign  ]  #4: ObjectId: 2.5.29.14 Criticality=false  SubjectKeyIdentifier [  KeyIdentifier [  0000: 48 19 E7 92 6F 92 9D 34 63 99 C0 F0 99 C8 D6 A5 H...o..4c.......  0010: 8C 8C 7F 65 ...e  ]  ]  Trust this certificate? [no]: yes  Certificate was added to keystore |

1. Copy the **jssecacerts** file from the Cloudera Manager Server host to all other cluster hosts. Make sure you copy the file to the correct location ($JAVA\_HOME/jre/lib/security/jssecacerts), because the Oracle JDK expects it there
2. On the Cloudera Manager Server host, append the intermediate CA certificate to the signed server certificate, and then import it into the keystore. Make sure that you use the append operator (>>) and not the overwrite operator (>):

|  |
| --- |
| sudo cat /opt/cloudera/security/pki/intca.cert.pem >> /opt/cloudera/security/pki/$(hostname -f)-server.cert.pem |

|  |
| --- |
| $ sudo keytool -importcert -alias $(hostname -f)-server \  -file /opt/cloudera/security/pki/$(hostname -f)-server.cert.pem \  -keystore /opt/cloudera/security/pki/$(hostname -f)-server.jks |

Here Password is password

|  |
| --- |
| [root@mac127 pki]# sudo keytool -importcert -alias $(hostname -f)-server \  > -file /opt/cloudera/security/pki/$(hostname -f)-server.cert.pem \  > -keystore /opt/cloudera/security/pki/$(hostname -f)-server.jks  Enter keystore password:  Top-level certificate in reply:  Owner: CN=Symantec Trial Secure Server CA - G3, OU="For Test Purposes Only. No assurances.", O=Symantec Corporation, C=US  Issuer: CN=VeriSign Trial Secure Server Root CA - G2, OU="For Test Purposes Only. No assurances.", O="VeriSign, Inc.", C=US  Serial number: 12880ba5c77745a8d5a9158c954286b9  Valid from: Fri Dec 05 05:30:00 IST 2014 until: Sat Dec 07 05:29:59 IST 2024  Certificate fingerprints:  MD5: E9:BB:80:E5:5A:18:54:F6:98:1F:F0:3D:0E:1B:5D:77  SHA1: A2:5B:0C:6A:F8:9E:8C:B6:07:21:D6:84:EA:FB:E9:8C:28:48:83:C0  SHA256: 98:4C:41:1C:EC:D5:08:6F:7B:EE:E2:CC:1F:5F:10:87:BC:F3:D6:93:E5:EC:65:47:C4:1E:8D:44:F7:0E:AA:84  Signature algorithm name: SHA256withRSA  Version: 3  Extensions:  #1: ObjectId: 1.3.6.1.5.5.7.1.1 Criticality=false  AuthorityInfoAccess [  [  accessMethod: ocsp  accessLocation: URIName: http://s2.symcb.com  ]  ]  #2: ObjectId: 2.5.29.35 Criticality=false  AuthorityKeyIdentifier [  KeyIdentifier [  0000: 48 19 E7 92 6F 92 9D 34 63 99 C0 F0 99 C8 D6 A5 H...o..4c.......  0010: 8C 8C 7F 65 ...e  ]  ]  #3: ObjectId: 2.5.29.19 Criticality=true  BasicConstraints:[  CA:true  PathLen:0  ]  #4: ObjectId: 2.5.29.31 Criticality=false  CRLDistributionPoints [  [DistributionPoint:  [URIName: http://s.symcb.com/SVRTrialRootG2.crl]  ]]  #5: ObjectId: 2.5.29.32 Criticality=false  CertificatePolicies [  [CertificatePolicyId: [2.16.840.1.113733.1.7.21]  [PolicyQualifierInfo: [  qualifierID: 1.3.6.1.5.5.7.2.1  qualifier: 0000: 16 17 68 74 74 70 73 3A 2F 2F 64 2E 73 79 6D 63 ..https://d.symc  0010: 62 2E 63 6F 6D 2F 63 70 73 b.com/cps  ], PolicyQualifierInfo: [  qualifierID: 1.3.6.1.5.5.7.2.2  qualifier: 0000: 30 19 1A 17 68 74 74 70 73 3A 2F 2F 64 2E 73 79 0...https://d.sy  0010: 6D 63 62 2E 63 6F 6D 2F 72 70 61 mcb.com/rpa  ]] ]  ]  #6: ObjectId: 2.5.29.15 Criticality=true  KeyUsage [  Key\_CertSign  Crl\_Sign  ]  #7: ObjectId: 2.5.29.14 Criticality=false  SubjectKeyIdentifier [  KeyIdentifier [  0000: A9 F0 46 01 FA AF 3D 89 40 E7 16 21 68 71 44 35 ..F...=.@..!hqD5  0010: 5D 2A D0 BE ]\*..  ]  ]  ... is not trusted. Install reply anyway? [no]: yes  Certificate reply was installed in keystore  [root@mac127 pki]# |

### Generate the Cloudera Manager Agent Certificates

Complete the following procedure on each Cloudera Manager Agent host. The provided examples continue to use an internal certificate authority (CA) to sign the agent certificates.

1. On all Cloudera Manager Agent hosts, create the **/opt/cloudera/security/pki** directory:
2. On all Cloudera Manager Agent hosts, create a Java Keystore and private key as follows:

|  |
| --- |
| $ keytool -genkeypair -alias $(hostname -f)-agent -keyalg RSA -keystore \  /opt/cloudera/security/pki/$(hostname -f)-agent.jks -keysize 2048 -dname \  "CN=$(hostname -f),OU=Engineering,O=Cybage,L=Pune,ST=MAHARASHTRA,C=IN" \  -storepass password -keypass password |

Use the same password for the -keypass and -storepass values. Cloudera Manager does not support using different passwords for the key and keystore

1. On all Cloudera Manager Agent hosts, generate the certificate signing request (CSR) and submit it to a CA.

Follow attached document for CSR



1. Use the keytool extended attributes to specify both serverAuth and clientAuth options:

|  |
| --- |
| $ keytool -certreq -alias $(hostname -f)-agent \  -keystore /opt/cloudera/security/pki/$(hostname -f)-agent.jks \  -file /opt/cloudera/security/pki/$(hostname -f)-agent.csr \  -ext EKU=serverAuth,clientAuth \  -storepass password -keypass password |

1. Inspect the certificates to verify that both server and client authentication options are present:

$ openssl x509 -in /opt/cloudera/security/pki/$(hostname -f)-agent.cert.pem -noout -text

Look for output similar to the following:

X509v3 Extended Key Usage:

TLS Web Server Authentication, TLS Web Client Authentication

If the certificate does not have both TLS Web Server Authentication and TLS Web Client Authentication listed in the X509v3 Extended Key Usage section, re-submit the CSR to the CA, and request that they generate a certificate that can be used for both server and client authentication.

|  |
| --- |
| [root@mac127 pki]# openssl x509 -in /opt/cloudera/security/pki/$(hostname -f)-agent.cert.pem -noout -text  Certificate:  Data:  Version: 3 (0x2)  Serial Number:  5f:de:bf:e7:23:43:c9:66:e8:8c:78:3b:d2:11:ab:6c  Signature Algorithm: sha256WithRSAEncryption  Issuer: C=US, O=Symantec Corporation, OU=For Test Purposes Only. No assurances., CN=Symantec Trial Secure Server CA - G3  Validity  Not Before: May 23 00:00:00 2017 GMT  Not After : Jun 22 23:59:59 2017 GMT  Subject: C=IN, ST=MAHARASHTRA, L=Pune, O=Cybage, OU=Engineering, CN=mac127.cybage.com  Subject Public Key Info:  Public Key Algorithm: rsaEncryption  Public-Key: (2048 bit)  Modulus:  00:c6:c2:1a:b4:9b:da:05:e4:0b:17:19:59:bc:e5:  26:ab:cf:c6:9b:c4:84:6b:59:55:9a:7e:a5:2d:e2:  b5:29:37:5d:3e:9b:c6:3b:ef:55:f3:c8:3c:3f:e2:  97:20:fd:a2:cc:29:d3:1a:a2:13:4c:9e:d6:ef:87:  0f:1b:ce:10:5f:98:68:d2:0f:e0:3a:3b:46:d6:78:  5f:1d:a7:43:08:32:07:ce:c2:95:e8:9f:8a:f3:aa:  3d:42:05:c9:a0:26:cd:fb:0c:38:7a:27:8a:71:b8:  81:80:52:a2:87:f3:e4:ca:fa:a2:be:b3:87:2a:89:  01:79:47:da:d0:85:0f:39:a5:a3:e9:eb:58:8e:39:  57:ba:51:a5:21:ff:69:53:18:e5:3a:59:ee:f6:c7:  5e:6a:f7:64:97:95:55:ee:78:ba:b5:6d:13:a8:33:  b1:02:b1:7d:29:90:3e:58:6d:02:1b:32:11:e4:1b:  ad:87:9f:d0:05:24:7c:b4:5a:8c:dd:f5:97:05:70:  2e:af:e2:9b:2e:42:2e:7a:3b:b1:7a:a1:7a:ef:ff:  8c:c5:6b:72:81:8f:87:c7:ff:60:ab:03:be:82:7c:  fb:16:b3:13:9c:1d:09:aa:d3:46:47:47:de:b3:c1:  e6:17:59:8d:bd:92:c7:c5:d8:53:8f:01:be:c8:70:  27:d7  Exponent: 65537 (0x10001)  X509v3 extensions:  X509v3 Subject Alternative Name:  DNS:mac127.cybage.com  X509v3 Basic Constraints:  CA:FALSE  X509v3 Key Usage: critical  Digital Signature, Key Encipherment  X509v3 CRL Distribution Points:  Full Name:  URI:http://re.symcb.com/re.crl  X509v3 Certificate Policies:  Policy: 2.16.840.1.113733.1.7.21  CPS: https://d.symcb.com/cps  User Notice:  Explicit Text: https://d.symcb.com/rpa  X509v3 Extended Key Usage:  TLS Web Server Authentication, TLS Web Client Authentication  X509v3 Authority Key Identifier:  keyid:A9:F0:46:01:FA:AF:3D:89:40:E7:16:21:68:71:44:35:5D:2A:D0:BE  Authority Information Access:  CA Issuers - URI:http://re.symcb.com/re.crt  Signature Algorithm: sha256WithRSAEncryption  43:4e:98:38:2c:f4:ac:36:8c:a4:8a:1c:dc:46:fb:60:df:75:  bd:ed:41:dc:01:b2:ff:01:77:31:4e:24:0a:e7:fa:8d:08:bc:  05:3c:eb:c4:8e:35:a3:97:d9:2e:42:19:23:ab:b8:e2:1f:42:  14:8a:dc:f9:d8:bc:69:42:6f:a2:fd:dd:1a:3d:cf:9b:41:76:  42:d4:11:2d:8d:8f:c3:75:f3:40:d2:8e:92:ae:96:00:4a:f8:  11:f1:28:3f:b4:50:36:ee:b0:8d:7f:f1:c3:98:11:ca:d6:db:  85:9f:9d:01:0d:79:9c:ea:01:c0:96:66:7a:02:f3:4d:b7:01:  8e:05:2f:11:ef:d2:67:33:19:12:a2:47:ee:e6:10:38:c9:6e:  2a:1a:ad:e4:eb:46:e0:0a:f1:f3:aa:75:68:60:80:f7:63:af:  78:90:a4:09:fb:9d:fb:27:94:57:b2:e9:43:2c:98:9c:93:11:  b0:06:d1:39:fd:48:74:ef:5d:17:45:73:cc:14:af:cc:a9:63:  e6:0c:82:76:18:2a:a3:33:2e:3b:b0:68:44:e3:f2:26:5f:58:  45:96:14:be:cf:10:76:29:78:b0:20:89:65:bb:8f:3a:1c:65:  3e:60:41:14:e6:ab:34:65:23:0c:d9:10:82:c3:ea:4b:2c:6b:  48:38:cf:91 |

1. On each Cloudera Manager Agent host, append the intermediate CA certificate to the signed certificate, and then import it into the keystore. Make sure that you use the append operator (>>) and not the overwrite operator (>):

|  |
| --- |
| $ sudo cat /opt/cloudera/security/pki/intca.agent.cert.pem >> /opt/cloudera/security/pki/$(hostname -f)-agent.cert.pem |

|  |
| --- |
| $ sudo keytool -importcert -alias $(hostname -f)-agent \  -file /opt/cloudera/security/pki/$(hostname -f)-agent.cert.pem \  -keystore /opt/cloudera/security/pki/$(hostname -f)-agent.jks |

If you see a message like the following, enter yes to continue:

... is not trusted. Install reply anyway? [no]: yes

You must see the following response verifying that the certificate has been properly imported against its private key.

Certificate reply was installed in keystore

Password is password

|  |
| --- |
| [root@mac127 pki]# sudo keytool -importcert -alias $(hostname -f)-agent \  > -file /opt/cloudera/security/pki/$(hostname -f)-agent.cert.pem \  > -keystore /opt/cloudera/security/pki/$(hostname -f)-agent.jks  Enter keystore password:  Top-level certificate in reply:  Owner: CN=Symantec Trial Secure Server CA - G3, OU="For Test Purposes Only. No assurances.", O=Symantec Corporation, C=US  Issuer: CN=VeriSign Trial Secure Server Root CA - G2, OU="For Test Purposes Only. No assurances.", O="VeriSign, Inc.", C=US  Serial number: 12880ba5c77745a8d5a9158c954286b9  Valid from: Fri Dec 05 05:30:00 IST 2014 until: Sat Dec 07 05:29:59 IST 2024  Certificate fingerprints:  MD5: E9:BB:80:E5:5A:18:54:F6:98:1F:F0:3D:0E:1B:5D:77  SHA1: A2:5B:0C:6A:F8:9E:8C:B6:07:21:D6:84:EA:FB:E9:8C:28:48:83:C0  SHA256: 98:4C:41:1C:EC:D5:08:6F:7B:EE:E2:CC:1F:5F:10:87:BC:F3:D6:93:E5:EC:65:47:C4:1E:8D:44:F7:0E:AA:84  Signature algorithm name: SHA256withRSA  Version: 3  Extensions:  #1: ObjectId: 1.3.6.1.5.5.7.1.1 Criticality=false  AuthorityInfoAccess [  [  accessMethod: ocsp  accessLocation: URIName: http://s2.symcb.com  ]  ]  #2: ObjectId: 2.5.29.35 Criticality=false  AuthorityKeyIdentifier [  KeyIdentifier [  0000: 48 19 E7 92 6F 92 9D 34 63 99 C0 F0 99 C8 D6 A5 H...o..4c.......  0010: 8C 8C 7F 65 ...e  ]  ]  #3: ObjectId: 2.5.29.19 Criticality=true  BasicConstraints:[  CA:true  PathLen:0  ]  #4: ObjectId: 2.5.29.31 Criticality=false  CRLDistributionPoints [  [DistributionPoint:  [URIName: http://s.symcb.com/SVRTrialRootG2.crl]  ]]  #5: ObjectId: 2.5.29.32 Criticality=false  CertificatePolicies [  [CertificatePolicyId: [2.16.840.1.113733.1.7.21]  [PolicyQualifierInfo: [  qualifierID: 1.3.6.1.5.5.7.2.1  qualifier: 0000: 16 17 68 74 74 70 73 3A 2F 2F 64 2E 73 79 6D 63 ..https://d.symc  0010: 62 2E 63 6F 6D 2F 63 70 73 b.com/cps  ], PolicyQualifierInfo: [  qualifierID: 1.3.6.1.5.5.7.2.2  qualifier: 0000: 30 19 1A 17 68 74 74 70 73 3A 2F 2F 64 2E 73 79 0...https://d.sy  0010: 6D 63 62 2E 63 6F 6D 2F 72 70 61 mcb.com/rpa  ]] ]  ]  #6: ObjectId: 2.5.29.15 Criticality=true  KeyUsage [  Key\_CertSign  Crl\_Sign  ]  #7: ObjectId: 2.5.29.14 Criticality=false  SubjectKeyIdentifier [  KeyIdentifier [  0000: A9 F0 46 01 FA AF 3D 89 40 E7 16 21 68 71 44 35 ..F...=.@..!hqD5  0010: 5D 2A D0 BE ]\*..  ]  ]  ... is not trusted. Install reply anyway? [no]: yes  Certificate reply was installed in keystore |

1. On each Cloudera Manager Agent host, create symbolic links (symlink) for the certificate and keystore files:

|  |
| --- |
| $ ln -s /opt/cloudera/security/pki/$(hostname -f)-agent.cert.pem /opt/cloudera/security/pki/agent.cert.pem  $ ln -s /opt/cloudera/security/pki/$(hostname -f)-agent.jks /opt/cloudera/security/pki/agent.jks |

This allows you to use the same **/etc/cloudera-scm-agent/config.ini** file on all agent hosts rather than maintaining a file for each agent

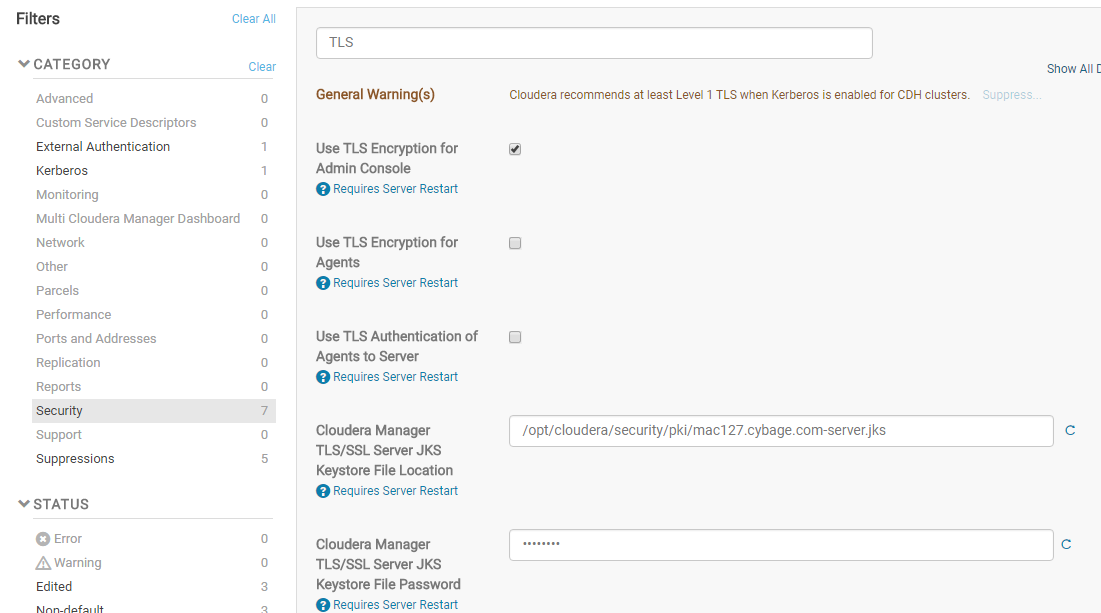
## Configuring TLS Encryption for the Cloudera Manager Admin Console

### Step 1: Enable HTTPS for the Cloudera Manager Admin Console

1. Log in to the Cloudera Manager Admin Console.
2. Select **Administration** > **Settings**.
3. Select the **Security** category.
4. Configure the following TLS settings:

| **Property** | **Description** |
| --- | --- |
| Cloudera Manager TLS/SSL Server JKS Keystore File Location | The complete path to the keystore file. In this example, the path is **/opt/cloudera/security/pki/*mac127.cybage.com*-server.jks** |
| Cloudera Manager TLS/SSL Server JKS Keystore File Password | The password for the **/opt/cloudera/security/jks/ *mac127.cybage.com*-server.jks**keystore. Which we set as password in script |
| Use TLS Encryption for Admin Console | Check this box to enable TLS encryption for Cloudera Manager. |

1. Click **Save Changes** to save the settings.



### Step 2: Specify SSL Truststore Properties for Cloudera Management Services

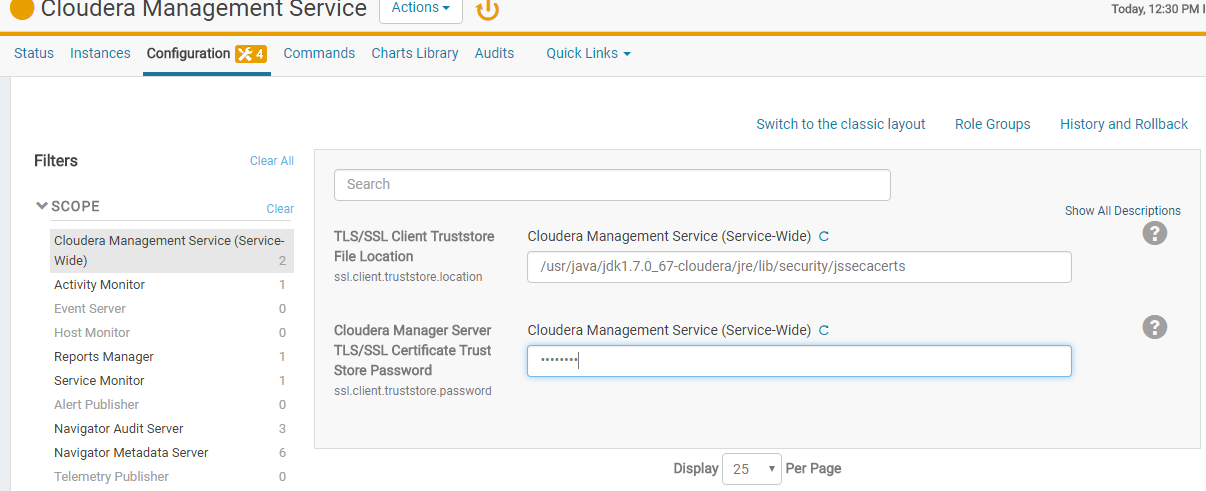
When enabling TLS for the Cloudera Manager Server admin interface, you must set the Java truststore location and password in the Cloudera Management Services configuration. Otherwise, roles such as Host Monitor and Service Monitor cannot connect to Cloudera Manager Server and will not start.

Configure the path and password for the $JAVA\_HOME/jre/lib/security/jssecacerts truststore that you created earlier. Make sure that you copied this file to all cluster hosts, including the Cloudera Management Service hosts.

1. Open the Cloudera Manager Administration Console and go to the **Cloudera Management Service** service.
2. Click the **Configuration** tab.
3. Select **Scope** > **Cloudera Management Service (Service-Wide)**.
4. Select **Category** > **Security**.
5. Edit the following TLS/SSL properties according to your cluster configuration.

| **Property** | **Description** |
| --- | --- |
| **TLS/SSL Client Truststore File Location** | The path to the client truststore file used in HTTPS communication. This truststore contains certificates of trusted servers, or of Certificate Authorities trusted to identify servers. For this example, set the value to $JAVA\_HOME/jre/lib/security/jssecacerts. Replace $JAVA\_HOME with the path to the Oracle JDK. |
| **TLS/SSL Client Truststore File Password** | The password for the truststore file. Here the password is changeit |

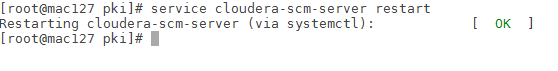
1. Click **Save Changes** to commit the changes.



### Step 3: Restart Cloudera Manager and Services

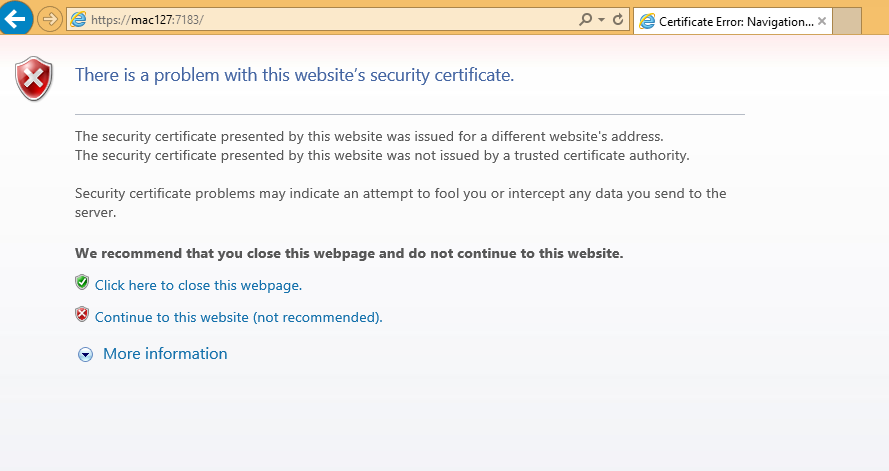
You must restart both Cloudera Manager Server and the Cloudera Management Service for TLS encryption to work. Otherwise, the Cloudera Management Services (such as Host Monitor and Service Monitor) cannot communicate with Cloudera Manager Server.

1. Restart the Cloudera Manager Server by running **service** **cloudera-scm-server restart** on the Cloudera Manager Server host.



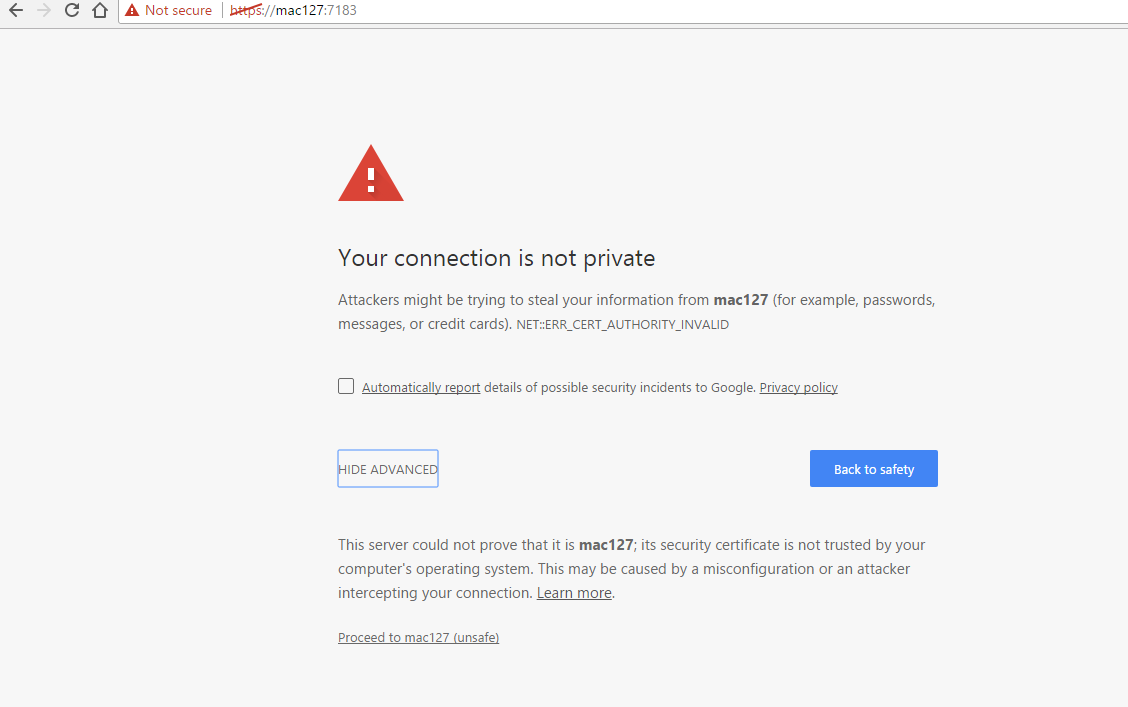
1. After the restart completes, connect to the Cloudera Manager Admin Console using the HTTPS URL (for example: https://mac127.cybage.com:7183). If you used an internal CA-signed certificate, you must configure your browser to trust the certificate. Otherwise, you will see a warning in your browser any time you access the Cloudera Manager Administration Console. By default, certificates issued by public commercial CAs are trusted by most browsers, and no additional configuration is necessary if your certificate is signed by one of them.

Using Internet Explorer



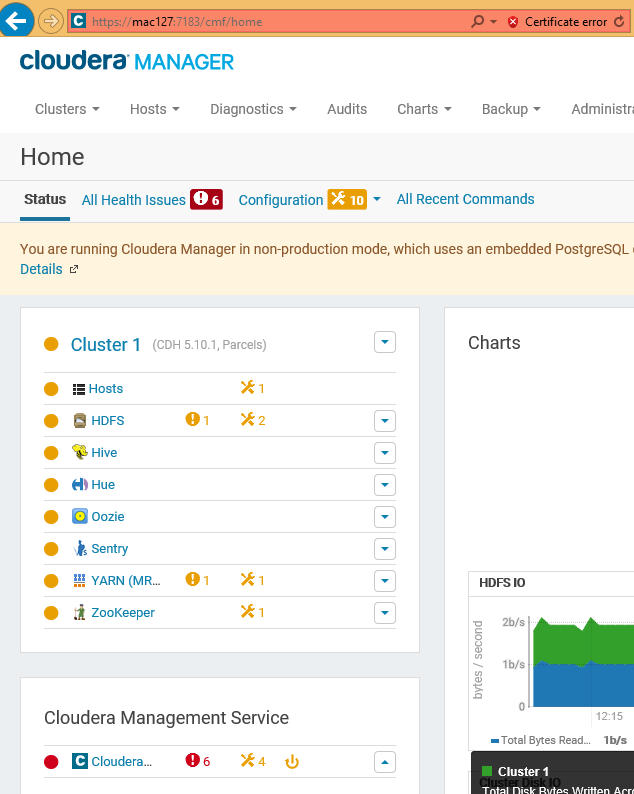
Click on Continue

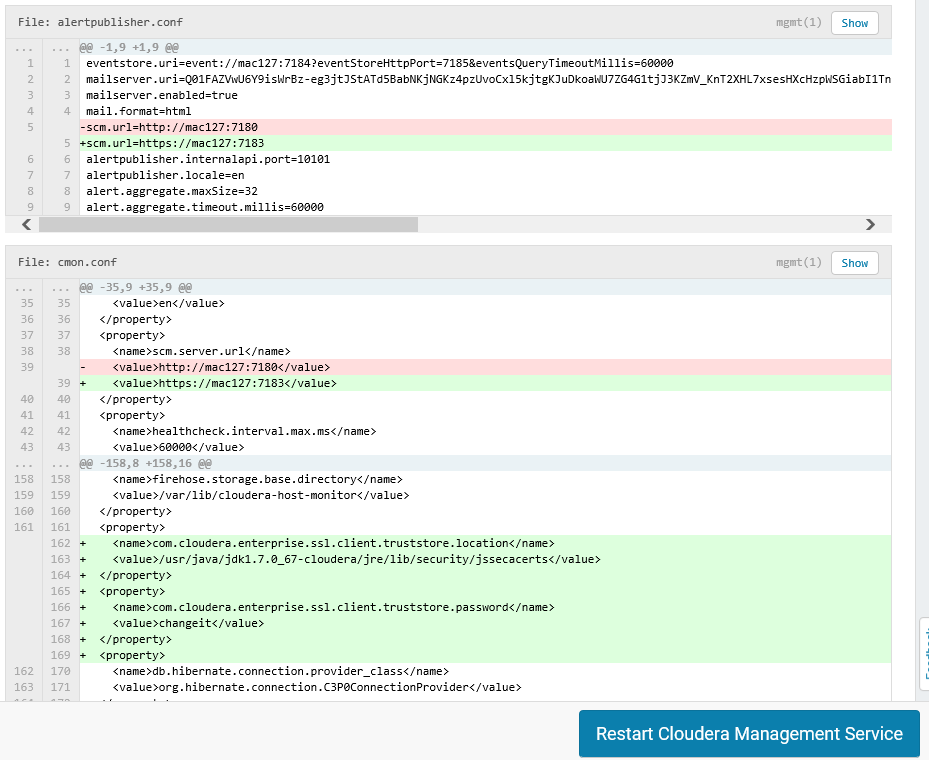
Using Google Chrome

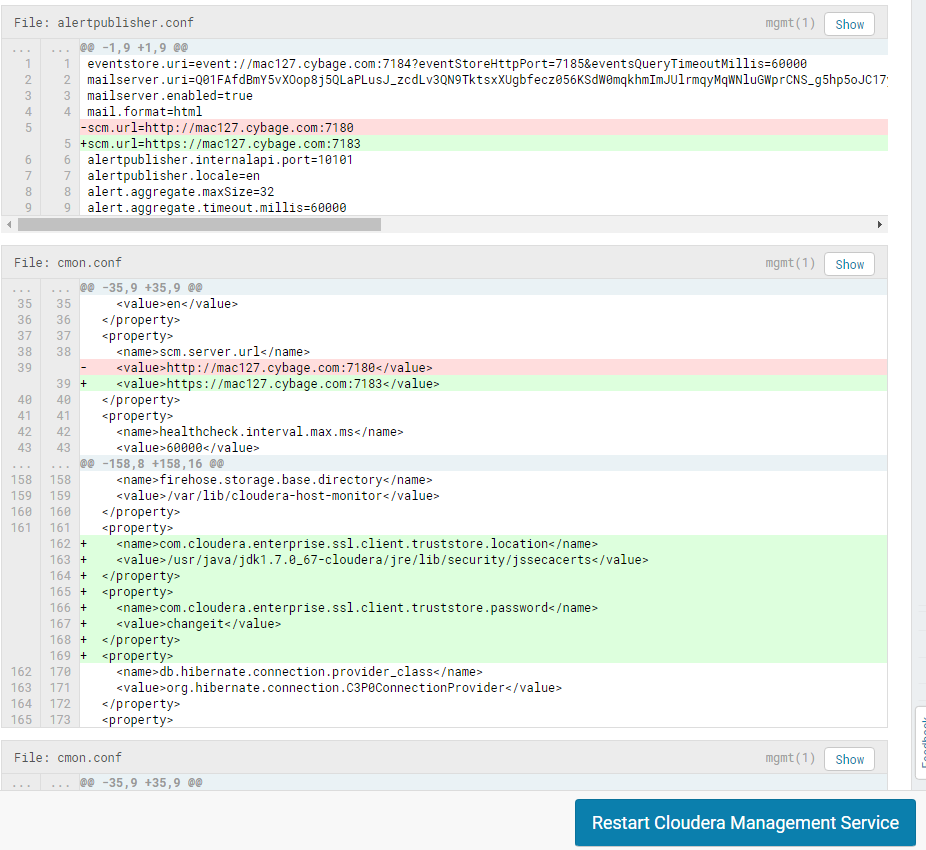


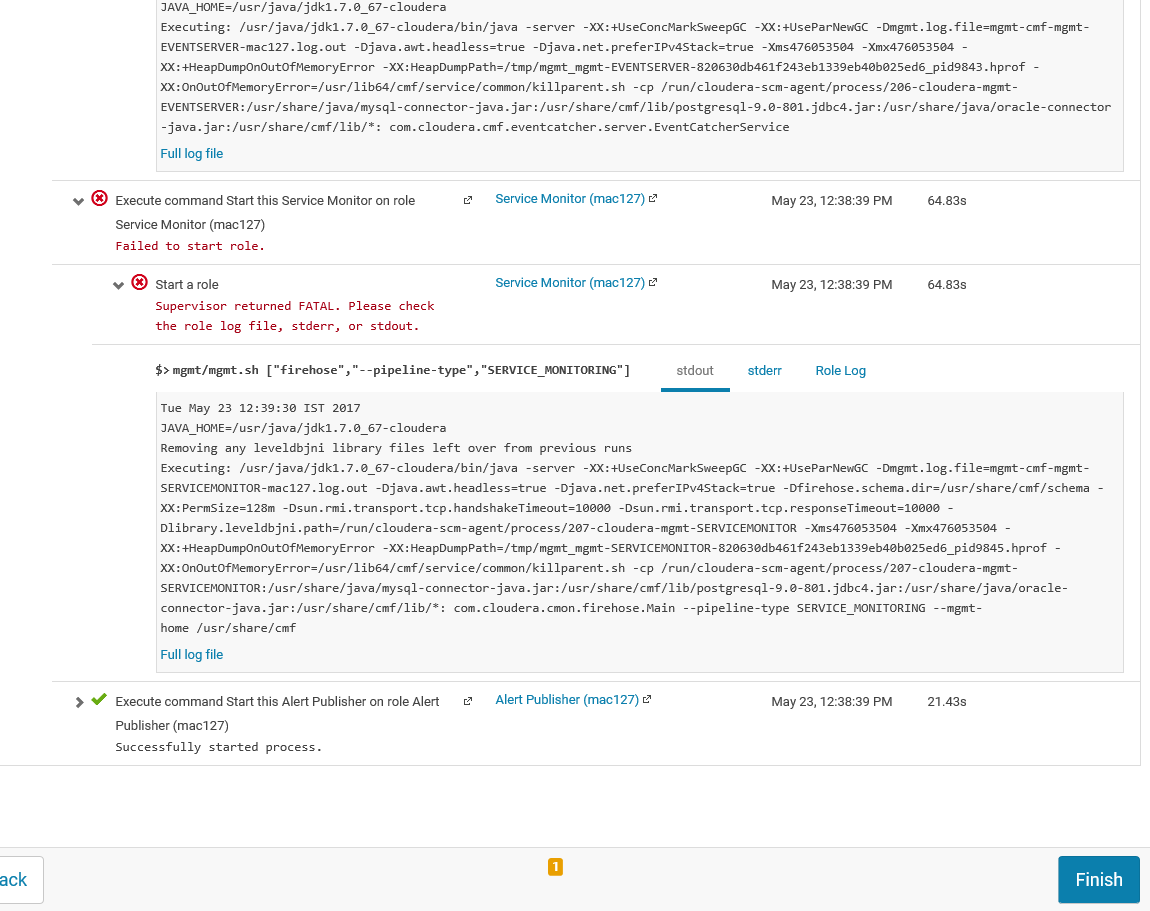
Click on proceed to mac127

1. Restart the Cloudera Management Service (**Cloudera Management Service** > **Actions** > **Restart**).









|  |
| --- |
| SimpleEvent{attributes={STACKTRACE=[java.io.IOException: HTTPS hostname wrong: should be <mac127>  at sun.net.www.protocol.https.HttpsClient.checkURLSpoofing(HttpsClient.java:652)  at sun.net.www.protocol.https.HttpsClient.afterConnect(HttpsClient.java:577)  at sun.net.www.protocol.https.AbstractDelegateHttpsURLConnection.connect(AbstractDelegateHttpsURLConnection.java:185)  at sun.net.www.protocol.http.HttpURLConnection.getOutputStream(HttpURLConnection.java:1091)  at sun.net.www.protocol.https.HttpsURLConnectionImpl.getOutputStream(HttpsURLConnectionImpl.java:250)  at com.cloudera.cmf.BasicScmProxy.authenticate(BasicScmProxy.java:265)  at com.cloudera.cmf.BasicScmProxy.fetch(BasicScmProxy.java:561)  at com.cloudera.cmf.BasicScmProxy.getFragmentAndHash(BasicScmProxy.java:651)  at com.cloudera.cmf.DescriptorAndFragments.newDescriptorAndFragments(DescriptorAndFragments.java:64)  at com.cloudera.cmon.firehose.Main.main(Main.java:378) |

This is because we have used two different hostname, while installing Cloudera manager we used mac127 and during signed certificate request it is mandatory to use fully qualified domain name (FQDN) so we used mac127.cybage.com, ideally at the time of Cloudera deployment we should use FQDN

Once you setup the Cloudera with hostname as FQDN , it will work

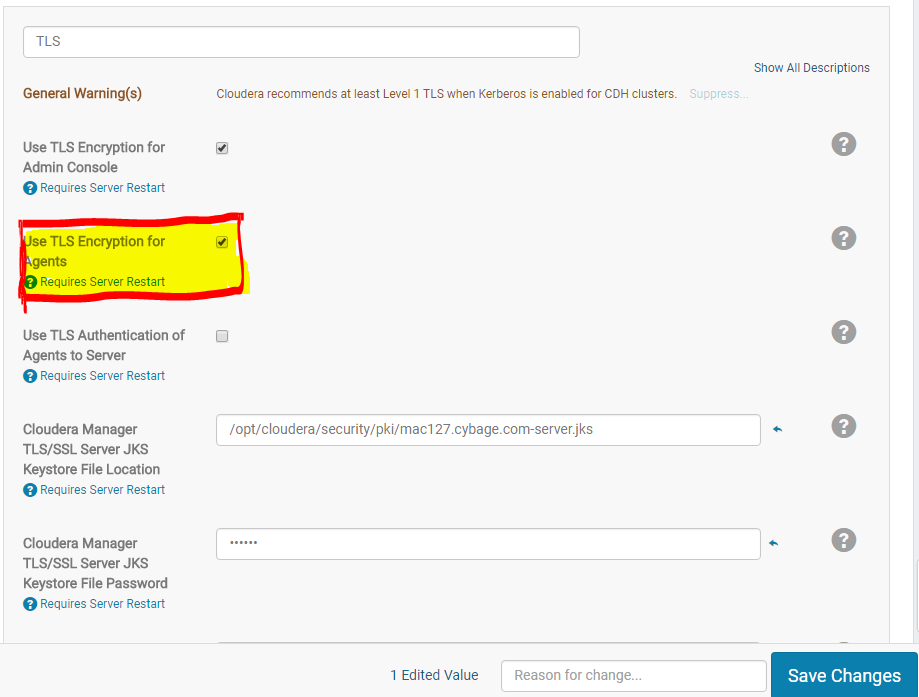
## Configuring TLS Encryption for Cloudera Manager Agents

Use the following procedure to encrypt the communication between Cloudera Manager Server and Cloudera Manager Agents:

### Step 1: Enable TLS Encryption for Agents in Cloudera Manager

Configure the TLS properties for Cloudera Manager Agents.

1. Log in to the Cloudera Manager Admin Console.
2. Select **Administration** > **Settings**.
3. Select the **Security** category.
4. Select the **Use TLS Encryption for Agents** option.
5. Click Save Changes.



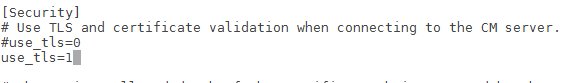
### Step 2: Enable TLS on Cloudera Manager Agent Hosts

To enable TLS between the Cloudera Manager agents and Cloudera Manager, you must specify values for the TLS properties in the /etc/cloudera-scm-agent/config.ini configuration file on all agent hosts.

1. On each agent host, open the /etc/cloudera-scm-agent/config.ini configuration file and set the use\_tlsparameter in the [Security] section as follows:

use\_tls=1

Alternatively, you can edit the config.ini file on one host, and then copy it to the other hosts because this file by default does not contain host-specific information. If you have modified properties such as listening\_hostname or listening\_ip address in config.ini, you must edit the file individually on each host.



### Step 3: Restart Cloudera Manager Server and Agents

Restart the Cloudera Manager Server with the following command to activate the TLS configuration settings.

$ sudo service cloudera-scm-server restart

On each agent host, restart the Cloudera Manager agent service:

$ sudo service cloudera-scm-agent restart

### Step 4: Verify that the Cloudera Manager Server and Agents are Communicating

In the Cloudera Manager Admin Console, go to **Hosts** > **All Hosts**. If you see successful heartbeats reported in the **Last Heartbeat** column after restarting the agents, TLS encryption is working properly.

**Issue found**

|  |  |
| --- | --- |
| Not getting Last heartbeat, restarted Cloudera management service    Got below issue under **/var/log/cloudera-scm-agent/cloudera-scm-agent.log file**   |  | | --- | | WrongHost: Peer certificate subjectAltName does not match host, expected 172.27.155.127, got DNS:mac127.cybage.com |   To resolve this issue you need to change the server host value from IP to hostname on each Cloudera agent  Update /etc/cloudera-scm-agent/config.ini file    Restart the Cloudera Manager Server with the following command to activate the TLS configuration settings.  $ sudo service cloudera-scm-server restart  On each agent host, restart the Cloudera Manager agent service:  $ sudo service cloudera-scm-agent restart  Now able to see last heartbeat |

## Enabling Server Certificate Verification on Cloudera Manager Agents

**Minimum Required Role:** [**Cluster Administrator**](https://www.cloudera.com/documentation/enterprise/5-9-x/topics/cm_sg_user_roles.html#concept_wfh_tvy_qp) (also provided by **Full Administrator**)

If you have completed the previous sections, communication between Cloudera Manager server and the agents is encrypted, but the certificate authenticity is not verified. For full security, you must configure the agents to verify the Cloudera Manager server certificate. If you are using a server certificate signed by an internal certificate authority (CA), you must configure the agents to trust that CA:

1. On each agent host, open the /etc/cloudera-scm-agent/config.ini configuration file, and then uncomment and set the following property:

verify\_cert\_file=/opt/cloudera/security/pki/rootca.cert.pem

Alternatively, you can edit the config.ini file on one host, and then copy it to the other hosts because this file by default does not contain host-specific information. If you have modified properties such as listening\_hostname or listening\_ip address in config.ini, you must edit the file individually on each host.

1. Restart the Cloudera Manager agents. On each agent host, run the following command:

$ sudo service cloudera-scm-agent restart

1. Restart the Cloudera Management Service. On the **Home** > **Status** tab, click  to the right of the Cloudera Management Service and select **Restart**.
2. Verify that the Cloudera Manager server and agents are communicating. In the Cloudera Manager Admin Console, go to **Hosts** > **All Hosts**. If you see successful heartbeats reported in the **Last Heartbeat** column after restarting the agents and management service, TLS verification is working properly. If not, check the agent log (/var/log/cloudera-scm-agent/cloudera-scm-agent.log) for errors.

## Configuring Agent Certificate Authentication

**Important:** Repeat this procedure on each agent host.

Without certificate authentication, a malicious user can add a host to Cloudera Manager by installing the Cloudera Manager agent software and configuring it to communicate with Cloudera Manager Server. To prevent this, you must configure Cloudera Manager to trust the agent certificates.

### Step 1: Export the Private Key to a File

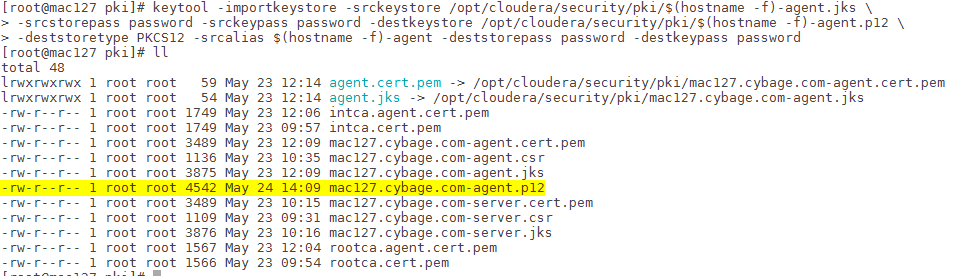
On each Cloudera Manager Agent host, use the keytool utility to export the private key and certificate to a PKCS12 file, which can then be split up into individual key and certificate files using the openssl command:

1. Export the private key and certificate:

$ keytool -importkeystore -srckeystore /opt/cloudera/security/pki/$(hostname -f)-agent.jks \

-srcstorepass password -srckeypass password -destkeystore /opt/cloudera/security/pki/$(hostname -f)-agent.p12 \

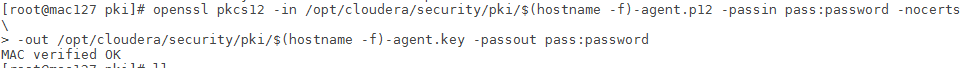
-deststoretype PKCS12 -srcalias $(hostname -f)-agent -deststorepass password -destkeypass password



1. Use the openssl command to export the private key into its own file:

$ openssl pkcs12 -in /opt/cloudera/security/pki/$(hostname -f)-agent.p12 -passin pass:password -nocerts \

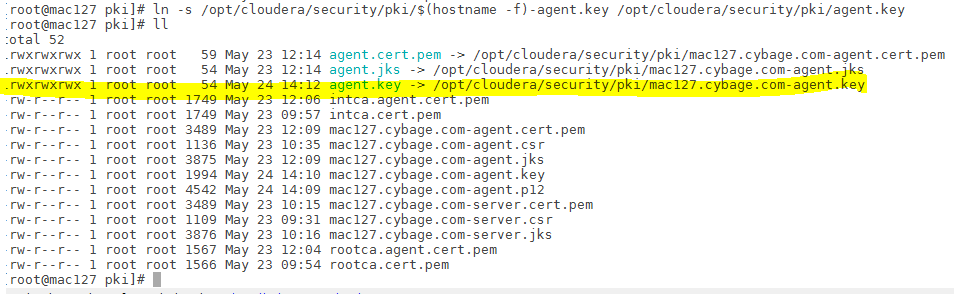
-out /opt/cloudera/security/pki/$(hostname -f)-agent.key -passout pass:password



1. Create a symbolic link for the .key file:

$ ln -s /opt/cloudera/security/pki/$(hostname -f)-agent.key /opt/cloudera/security/pki/agent.key

This allows you to use the same /etc/cloudera-scm-agent/config.ini file on all agent hosts rather than maintaining a file for each agent.



### Step 2: Create a Password File

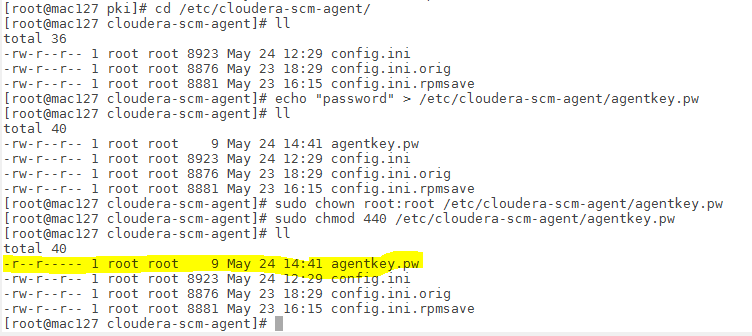
The Cloudera Manager agent obtains the password from a text file, not from a command line parameter or environment variable. The password file allows you to use file permissions to protect the password. For example, run the following commands on each Cloudera Manager Agent host, or run them on one host and copy the file to the other hosts:

$ echo "password" > /etc/cloudera-scm-agent/agentkey.pw

$ sudo chown root:root /etc/cloudera-scm-agent/agentkey.pw

$ sudo chmod 440 /etc/cloudera-scm-agent/agentkey.pw

Replace password with the password you created in [Step 1: Export the Private Key to a File.](#_Step_1:_Export)

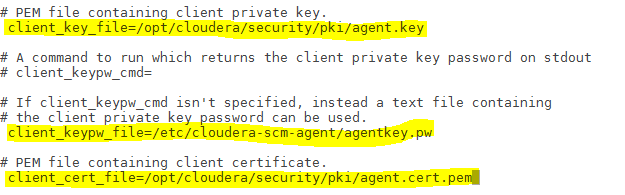
[](#_Step_1:_Export)

### Step 3: Configure the Agent to Use Private Keys and Certificates

On a Cloudera Manager Agent, open the /etc/cloudera-scm-agent/config.ini configuration file and edit the following properties.

| **Property** | **Example Value** | **Description** |
| --- | --- | --- |
| client\_key\_file | /opt/cloudera/security/pki/agent.key | Path to the private key file. |
| client\_keypw\_file | /etc/cloudera-scm-agent/agentkey.pw | Path to the private key password file. |
| client\_cert\_file | /opt/cloudera/security/pki/agent.cert.pem | Path to the client certificate file. |

Copy the file to all other cluster hosts. If you have modified properties such as listening\_hostname or listening\_ip address in config.ini, you must edit the file individually on each host.

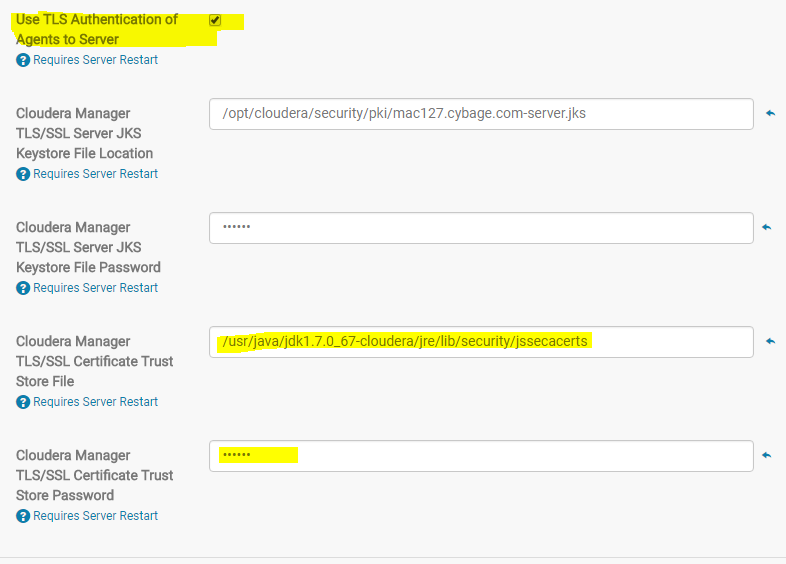


### Step 4: Enable Agent Certificate Authentication

1. Log in to the Cloudera Manager Admin Console.
2. Select **Administration** > **Settings**.
3. Click the **Security** category.
4. Configure the following TLS settings:

| **Setting** | **Description** |
| --- | --- |
| **Use TLS Authentication of Agents to Server** | Select option to enable TLS authentication of agents to the server. |
| **Cloudera Manager TLS/SSL Certificate Trust Store File** | Specify the full filesystem path to the jssecacerts file located on the Cloudera Manager Server host. For example, /usr/java/jdk1.7.0\_67-cloudera/jre/lib/security/jssecacerts. |
| **Cloudera Manager TLS/SSL Certificate Trust Store Password** | Specify the password for the jssecacerts truststore.  Here password is changeit |

1. Click **Save Changes** to save the settings.



### Step 5: Restart Cloudera Manager Server and Agents

1. On the Cloudera Manager server host, restart the Cloudera Manager server:

$ sudo service cloudera-scm-server restart

1. On every agent host, restart the Cloudera Manager agent:

$ sudo service cloudera-scm-agent restart

### Step 6: Verify that Cloudera Manager Server and Agents are Communicating

In the Cloudera Manager Admin Console, go to **Hosts** > **All Hosts**. If you see successful heartbeats reported in the **Last Heartbeat** column after restarting the agents and server, TLS certificate authentication is working properly. If not, check the agent log (/var/log/cloudera-scm-agent/cloudera-scm-agent.log) for errors.