SSL Configuration for Release Auto >= 5.0.x, using self-signed certificates(rough draft).

SSL Configuration for Release Auto >= 5.0.x, using self-signed certificates(rough draft).

If you are looking to configure versions below 5.0.x, please see 'SSL Configuration for NES/NAGs using self-signed certificates' (v4.7 and older).

Assumptions made in this guide:

1. We are running Linux.
2. Absolute base directory for our installation is default for 5.0.x(/usr/local/ReleaseAutomationServer).
3. All commands, unless specified otherwise, are run from aforementioned base directory.
4. Keystore with our custom keypairs will be located at %home%/conf/custom-keystore.jks
5. Trust store with our trusted certs will be located at %home%/conf/custom-truststore.jks

There are **three** sections to this document, each with a specific purpose. I would recommend running through all three to sufficiently familiarize yourself with java+SSL config implementation and how it is utilized by RA:

Sec 1. Configuration of Web UI/components to use custom SSL certificate.

Sec 2. Configuration of SSL between NAC<-->NES (ActiveMQ)

Sec 3. Configuration of SSL between NES<-->NAG (NiMi/TLS)

**** Section 1, Configuration of Web UI/Components to use custom SSL certificate. ****

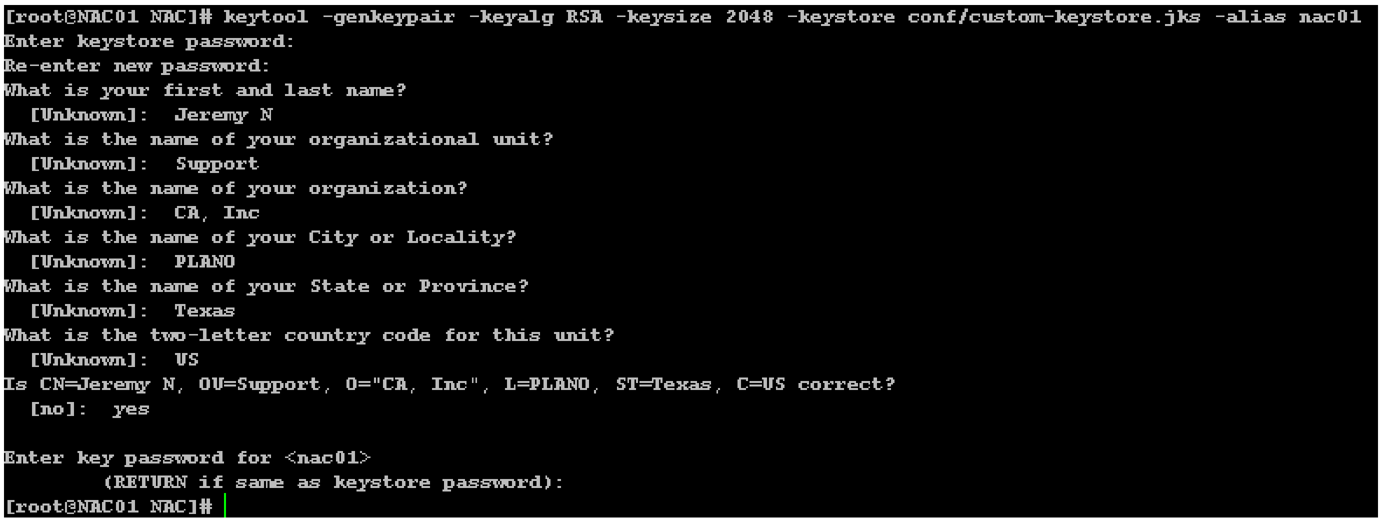
a. Generate a new keypair and new custom keystore using keytool.

First, we will want to generate a new keypair and new custom key store with the following command:

```
keytool -genkeypair -keyalg RSA -keysize 2048 -keystore conf/custom-keystore.jks -alias nac01
```

***** Note: You should then be prompted for a password, this is arbitrary, however do not forget the password as it goes to this keystore only. When prompted for alias password, just press ENTER to use the keystore password.***

<u>Output should be similar to:</u>



```
[root@NAC01 NAC]# keytool -genkeypair -keyalg RSA -keysize 2048 -keystore conf/custom-keystore.jks -alias nac01
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  Jeremy N
What is the name of your organizational unit?
  [Unknown]:  Support
What is the name of your organization?
  [Unknown]:  CA, Inc
What is the name of your City or Locality?
  [Unknown]:  PLANO
What is the name of your State or Province?
  [Unknown]:  Texas
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Jeremy N, OU=Support, O="CA, Inc", L=PLANO, ST=Texas, C=US correct?
  [no]:  yes

Enter key password for <nac01>
        (RETURN if same as keystore password):
[root@NAC01 NAC]#
```

b. (Optional), Verify contents of newly created custom key store.

***** Note: This is optional, but you can verify the contents of your key store by executing the following:***

> *keytool -list -v -keystore conf/custom-keystore.jks*

c. Export newly created public key for use in trust store.

Now we will want to export the public key to a certificate that can be used by other key stores/trust stores:

> *keytool -exportcert -alias nac01 -file nac01.crt -keystore conf/custom-keystore.jks -v -rfc*

***** Note: You will be prompted for the password that you have just initially set on this keystore. Upon successful authentication, the public certificate will be written to a file 'nac01.crt' in the CWD.***

d. Create custom trust store on server.

Next, we want to create a custom trust store, and our first certificate in the store will be the one we just exported 'nac01.crt'

Execute the following commands:

> *keytool -importcert -alias nac01 -file nac01.crt -keystore conf/custom-truststore.jks -v -rfc*

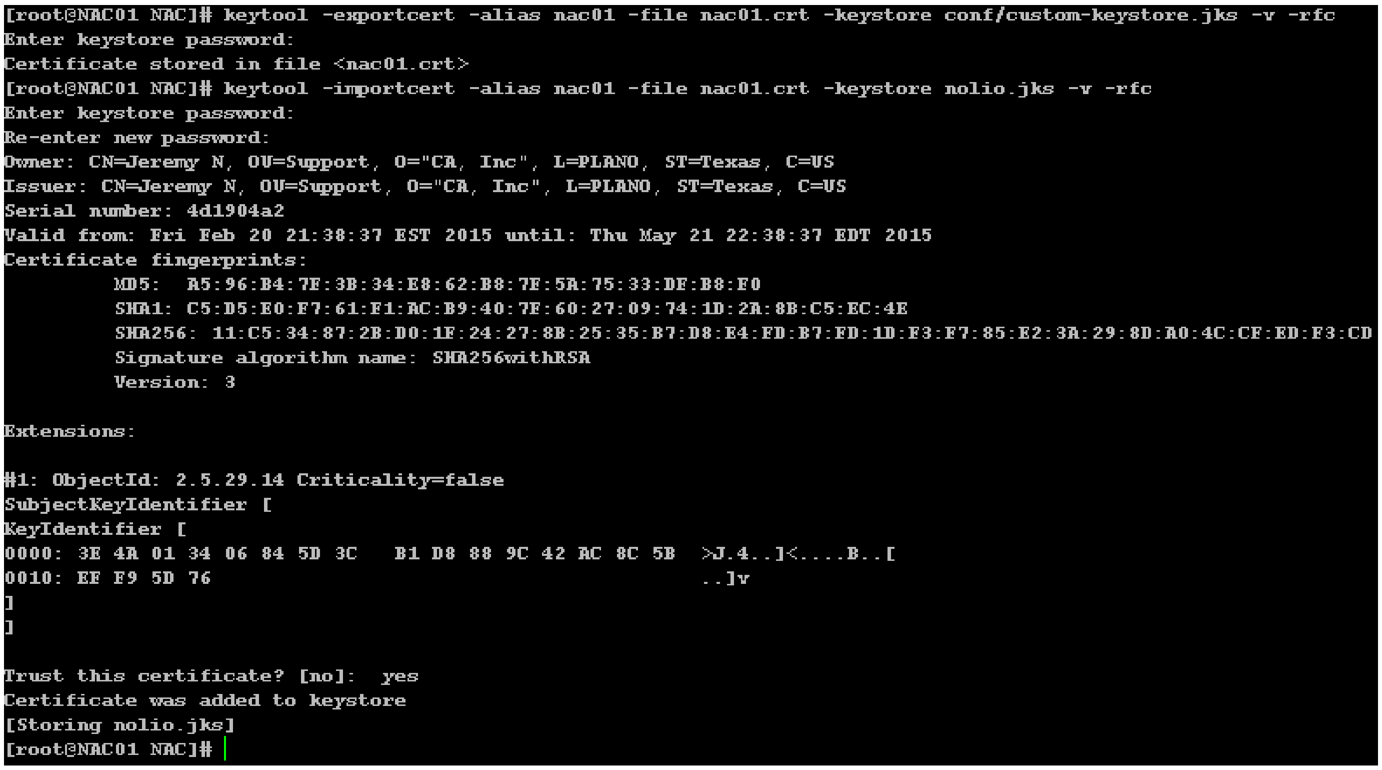
***** Note: This creates the initial trust store on this host, you should then be prompted for a password, this is arbitrary, however do not forget the password as it goes to this store only.***

> *keytool -importcert -alias nac01 -file nac01.crt -keystore nolio.jks -v -rfc*

__** Note: This creates the initial trust store that will only contain the certificate we wish to use with Web UI Components. We will need to sign to update the SSL certificate used by default. (See Section E.).__

__** Note: You should then be prompted for a password, this is arbitrary, however do not forget the password as it goes to this store only.__

<u>Output should be similar to:</u>

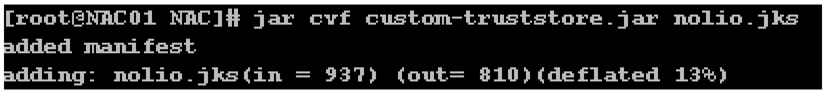


```
[root@NAC01 NAC]# keytool -exportcert -alias nac01 -file nac01.crt -keystore conf/custom-keystore.jks -v -rfc
Enter keystore password:
Certificate stored in file <nac01.crt>
[root@NAC01 NAC]# keytool -importcert -alias nac01 -file nac01.crt -keystore nolio.jks -v -rfc
Enter keystore password:
Re-enter new password:
Owner: CN=Jeremy N, OU=Support, O="CA, Inc", L=PLANO, ST=Texas, C=US
Issuer: CN=Jeremy N, OU=Support, O="CA, Inc", L=PLANO, ST=Texas, C=US
Serial number: 4d1904a2
Valid from: Fri Feb 20 21:38:37 EST 2015 until: Thu May 21 22:38:37 EDT 2015
Certificate fingerprints:
         MD5:  A5:96:B4:7E:3B:34:E8:62:B8:7E:5A:75:33:DF:B8:F0
         SHA1: C5:D5:E0:F7:61:F1:AC:B9:40:7E:60:27:09:74:1D:2A:8B:C5:EC:4E
         SHA256: 11:C5:34:87:2B:D0:1F:24:27:8B:25:35:B7:D8:E4:FD:B7:FD:1D:F3:F7:85:E2:3A:29:8D:A0:4C:CF:ED:F3:CD
         Signature algorithm name: SHA256withRSA
         Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 3E 4A 01 34 06 84 5D 3C   B1 D8 88 9C 42 AC 8C 5B  >J.4..]<....B..[
0010: EF F9 5D 76                                        ..]v
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
[Storing nolio.jks]
[root@NAC01 NAC]#
```

e. Create jar with nolio.jks, sign, and move to proper directory.

> *jar cvf custom-truststore.jar nolio.jks*

<u>Output should be similar to:</u>

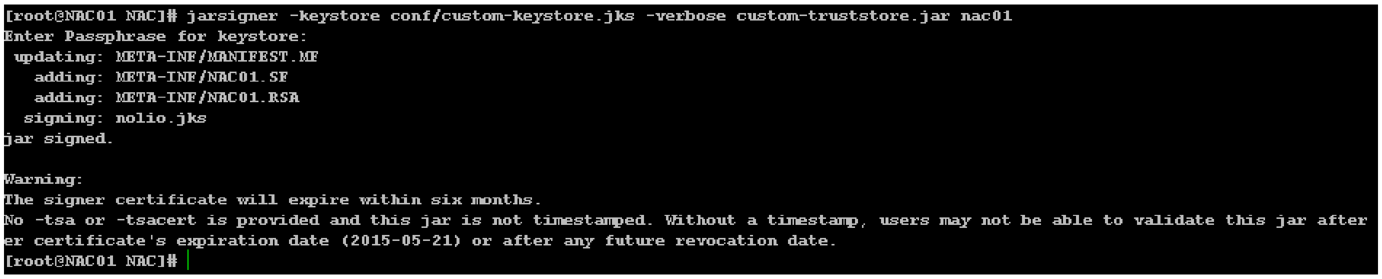


```
[root@NAC01 NAC]# jar cvf custom-truststore.jar nolio.jks
added manifest
adding: nolio.jks(in = 937) (out= 810)(deflated 13%)
```

1e. Sign newly created jar file

> *jarsigner -keystore conf/custom-keystore.jks -verbose custom-truststore.jar nac01*

<u>Output should be similar to:</u>



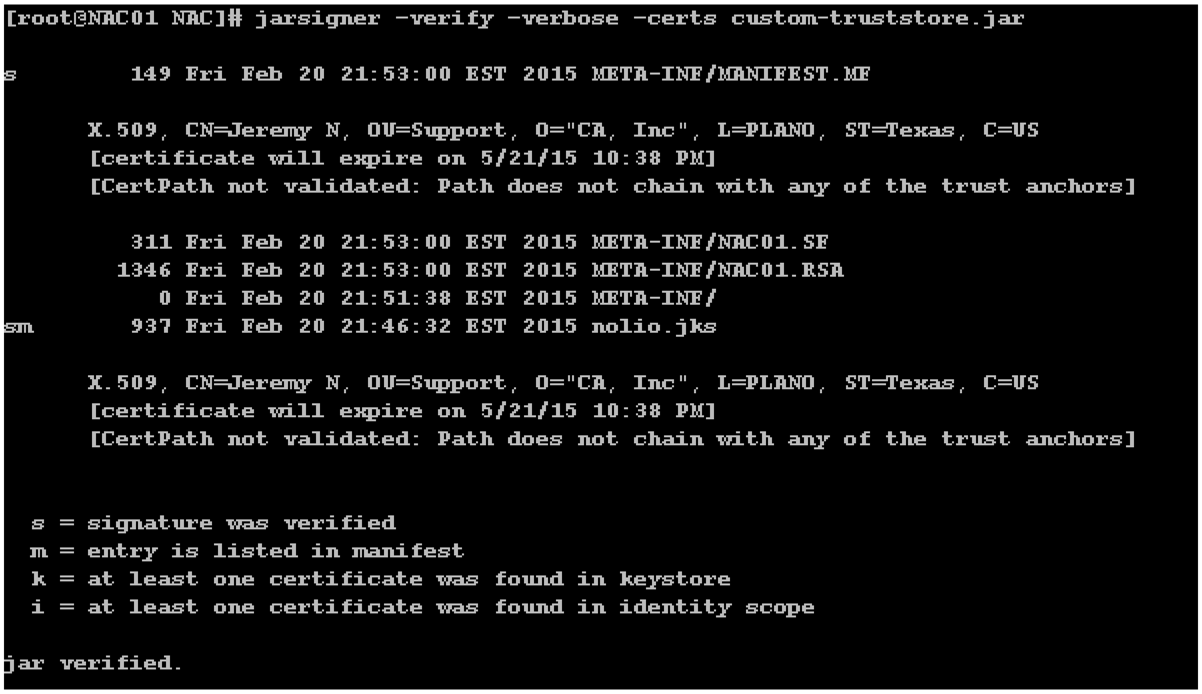
```
[root@NAC01 NAC]# jarsigner -keystore conf/custom-keystore.jks -verbose custom-truststore.jar nac01
Enter Passphrase for keystore:
 updating: META-INF/MANIFEST.MF
   adding: META-INF/NAC01.SF
   adding: META-INF/NAC01.RSA
  signing: nolio.jks
jar signed.

Warning:
The signer certificate will expire within six months.
No -tsa or -tsacert is provided and this jar is not timestamped. Without a timestamp, users may not be able to validate this jar after
er certificate's expiration date (2015-05-21) or after any future revocation date.
[root@NAC01 NAC]#
```

2e. (Optional) Verify signature, move jar to proper location.

(Optional) Verify proper signature on jar:

> *jarsigner –verify –verbose –certs custom-truststore.jar*

Output should be similar to:



```
[root@NAC01 NAC]# jarsigner -verify -verbose -certs custom-truststore.jar

s        149 Fri Feb 20 21:53:00 EST 2015 META-INF/MANIFEST.MF

      X.509, CN=Jeremy N, OU=Support, O="CA, Inc", L=PLANO, ST=Texas, C=US
      [certificate will expire on 5/21/15 10:38 PM]
      [CertPath not validated: Path does not chain with any of the trust anchors]

         311 Fri Feb 20 21:53:00 EST 2015 META-INF/NAC01.SF
        1346 Fri Feb 20 21:53:00 EST 2015 META-INF/NAC01.RSA
           0 Fri Feb 20 21:51:38 EST 2015 META-INF/
sm       937 Fri Feb 20 21:46:32 EST 2015 nolio.jks

      X.509, CN=Jeremy N, OU=Support, O="CA, Inc", L=PLANO, ST=Texas, C=US
      [certificate will expire on 5/21/15 10:38 PM]
      [CertPath not validated: Path does not chain with any of the trust anchors]


  s = signature was verified
  m = entry is listed in manifest
  k = at least one certificate was found in keystore
  i = at least one certificate was found in identity scope

jar verified.
```

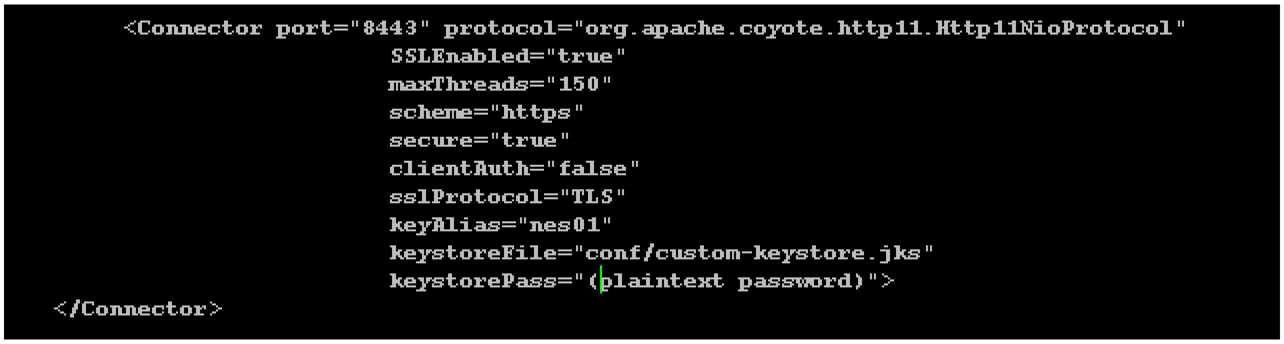
3e. Now we want to move the jar containing our verified ceritifcate to the proper location:

> *mv custom-truststore.jar webapps/nolio-app/apps/v2.0.0/lib*

f. Create a new file, conf/security-customization.properties - with the following contents:

```
ui.trustStorePassword=<plaintext password for keystore generated for custom-truststore.jar>
javax.net.ssl.trustStore=conf/custom-truststore.jks
javax.net.ssl.trustStorePassword=<plaintext password for conf/custom-truststore.jks>
```

g. Open/Edit conf/server.xml, update connector to reflect new custom information:

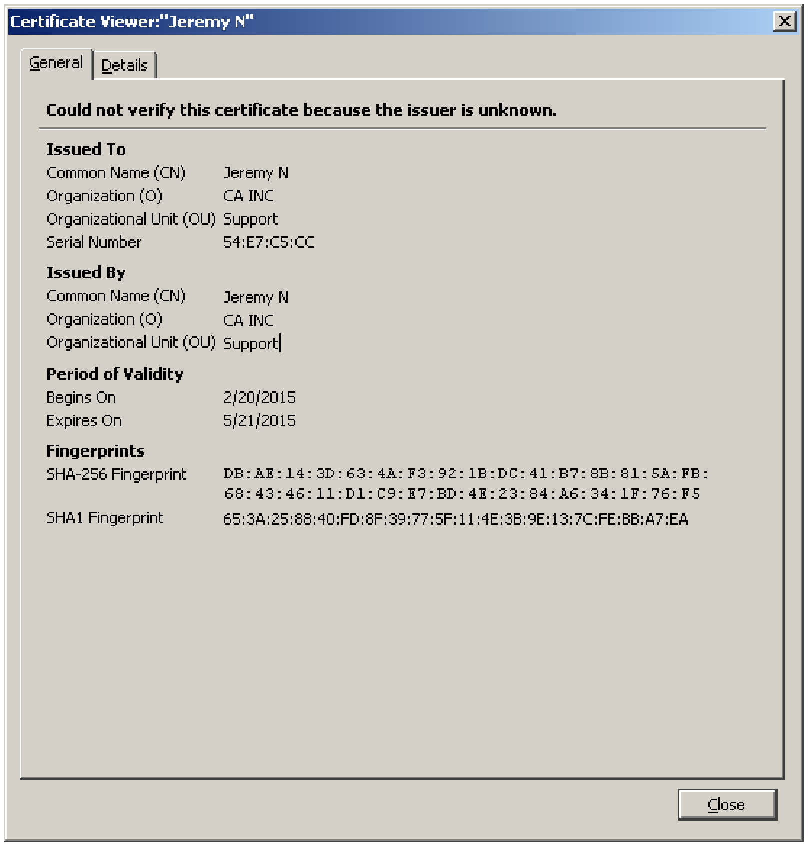


```
        <Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
                          SSLEnabled="true"
                          maxThreads="150"
                          scheme="https"
                          secure="true"
                          clientAuth="false"
                          sslProtocol="TLS"
                          keyAlias="nes01"
                          keystoreFile="conf/custom-keystore.jks"
                          keystorePass="(plaintext password)">
   </Connector>
```

h. Restart NAC/Management server for new settings to take effect, and clear the Java WS cache(remove old settings):

```
javaws -uninstall
```

Test the new settings by simply opening an HTTPS connection to NAC(https://nac-hostname:8443/) - If you view the https certificate, it should reflect the certificate we configured, such as our case:



Could not verify this certificate because the issuer is unknown.

Issued To

Common Name (CN)	Jeremy N
Organization (O)	CA INC
Organizational Unit (OU)	Support
Serial Number	54:E7:C5:CC

Issued By

Common Name (CN)	Jeremy N
Organization (O)	CA INC
Organizational Unit (OU)	Support

Period of Validity

Begins On	2/20/2015
Expires On	5/21/2015

Fingerprints

SHA-256 Fingerprint	DB:AE:14:3D:63:4A:F3:92:1B:DC:41:B7:8B:81:5A:FB:68:43:46:11:D1:C9:E7:BD:4E:23:84:A6:34:1F:76:F5
SHA1 Fingerprint	65:3A:25:88:40:FD:8F:39:77:5F:11:4E:3B:9E:13:7C:FE:BB:A7:EA

***** Note: If you are only configuring the Web UI component(s) you are done. If you are configuring the entire system, continue on to Section 2.***

**** Section 2. Configuration of SSL between NAC<-->NES (ActiveMQ) ****

a. Configure ActiveMQ on execution server to use SSL only:

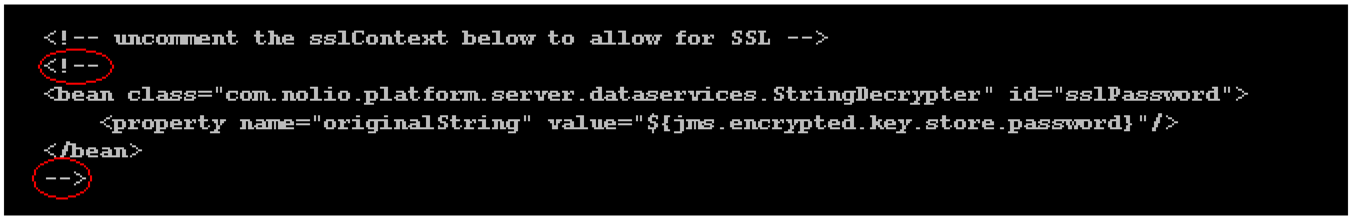
On the Execution server, open the following file in your favorite text editor(consider backing up first):

> *%home%/webapps/execution/WEB-INF/activemq-broker-nes.xml*

Very carefully, uncomment the comments for the *"amq:sslContext"* sections(2 of them):

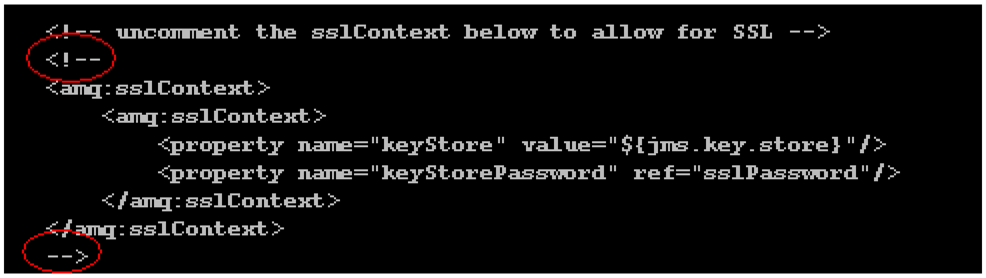
*** Examples(remove areas in red)*

1a.



```
<!-- uncomment the sslContext below to allow for SSL -->
<!--
<bean class="com.nolio.platform.server.dataservices.StringDecrypter" id="sslPassword">
    <property name="originalString" value="${jms.encrypted.key.store.password}"/>
</bean>
-->
```

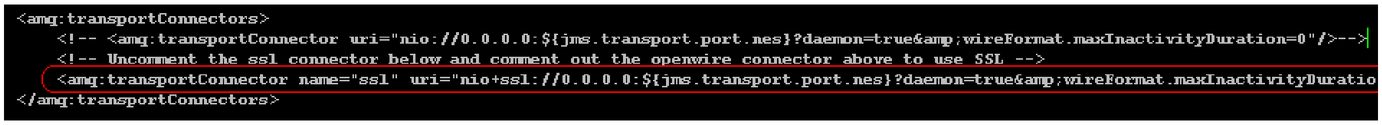
2a.



```
<!-- uncomment the sslContext below to allow for SSL -->
<!--
<amq:sslContext>
    <amq:sslContext>
        <property name="keyStore" value="${jms.key.store}"/>
        <property name="keyStorePassword" ref="sslPassword"/>
    </amq:sslContext>
</amq:sslContext>
-->
```

Next, we need to replace the transport connector with the SSL enabled configuration:

3a. Comment out the current amq:transportConnector(result should look like), enable 'ssl' transportConnector:



```
<amq:transportConnectors>
    <!-- <amq:transportConnector uri="nio://0.0.0.0:${jms.transport.port.nes}?daemon=true&amp;wireFormat.maxInactivityDuration=0"/>-->
    <!-- Uncomment the ssl connector below and comment out the openwire connector above to use SSL -->
    <amq:transportConnector name="ssl" uri="nio+ssl://0.0.0.0:${jms.transport.port.nes}?daemon=true&amp;wireFormat.maxInactivityDuratio
</amq:transportConnectors>
```

4a. Restart the Execution Server, SSL(HTTPS) is now enabled using the default nolio key store/trust store, keystore.conf and truststore.conf respectively.

b. Configure NAC and NES ActiveMQ brokers to use custom key store/trust store.

** Note: Default, RA/Nolio uses conf/keystore.jks and conf/nolio.jks for key store, and trust store, respectively. Do not modify these.

1b. If you do not have a custom truststore as generated in Sec 1, we will need to go ahead and create these stores on our Management(NAC) and Execution Servers(NES) If you already have, you can skip this step and just copy your custom-truststore.jar to the conf/ directory.

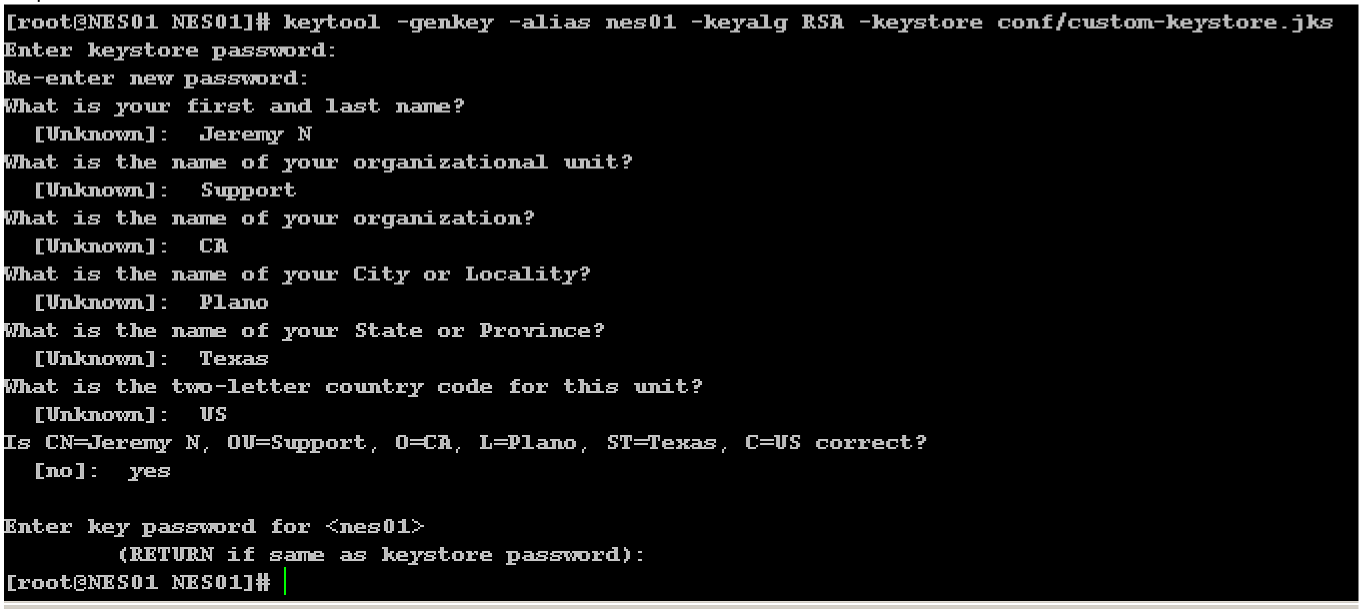
From Execution(NES) Server:

> *keytool -genkey -alias nes01 -keyalg RSA -keystore conf/custom-keystore.jks*

This creates a keystore with the alias 'nes01'.

** Note, the alias defined is arbitrary.

Output should be similar to:



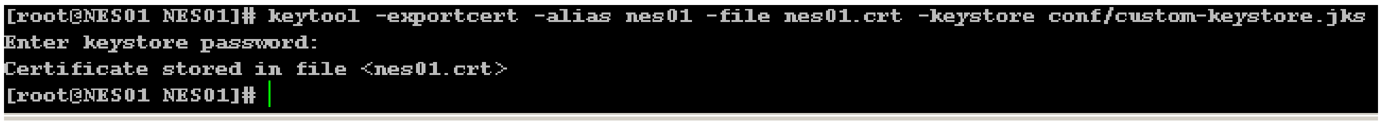
```
[root@NES01 NES01]# keytool -genkey -alias nes01 -keyalg RSA -keystore conf/custom-keystore.jks
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  Jeremy N
What is the name of your organizational unit?
  [Unknown]:  Support
What is the name of your organization?
  [Unknown]:  CA
What is the name of your City or Locality?
  [Unknown]:  Plano
What is the name of your State or Province?
  [Unknown]:  Texas
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Jeremy N, OU=Support, O=CA, L=Plano, ST=Texas, C=US correct?
  [no]:  yes

Enter key password for <nes01>
        (RETURN if same as keystore password):
[root@NES01 NES01]#
```

2b. Export NES certificate for addition to system wide trust store:

Execute the following on NES:

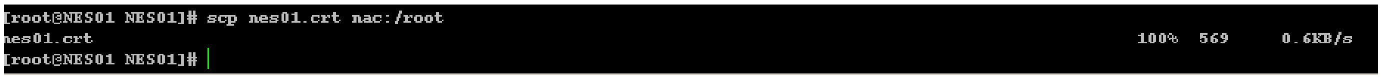
keytool -exportcert -alias nes01 -file nes01.crt -keystore conf/custom-keystore.jks

Output should be similar to:



```
[root@NES01 NES01]# keytool -exportcert -alias nes01 -file nes01.crt -keystore conf/custom-keystore.jks
Enter keystore password:
Certificate stored in file <nes01.crt>
[root@NES01 NES01]#
```

3b. Copy the newly created 'nes01.crt' file to a location on your NAC(Management) server, eg(linux):

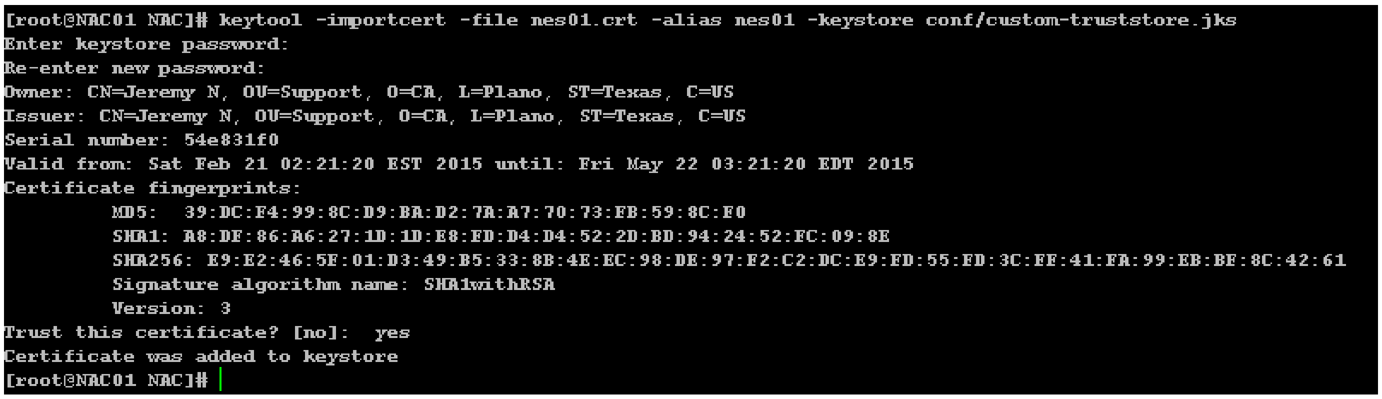


```
[root@NES01 NES01]# scp nes01.crt nac:/root
nes01.crt                                                                  100%  569     0.6KB/s
[root@NES01 NES01]#
```

4b. Import Certificate to NAC trust store:

> *keytool -importcert -file /root/nes01.crt -alias nes01 -keystore conf/custom-truststore.jks*

Output should be similar to:



```
[root@NAC01 NAC]# keytool -importcert -file nes01.crt -alias nes01 -keystore conf/custom-truststore.jks
Enter keystore password:
Re-enter new password:
Owner: CN=Jeremy N, OU=Support, O=CA, L=Plano, ST=Texas, C=US
Issuer: CN=Jeremy N, OU=Support, O=CA, L=Plano, ST=Texas, C=US
Serial number: 54e831f0
Valid from: Sat Feb 21 02:21:20 EST 2015 until: Fri May 22 03:21:20 EDT 2015
Certificate fingerprints:
         MD5:  39:DC:F4:99:8C:D9:BA:D2:7A:A7:70:73:FB:59:8C:E0
         SHA1: A8:DF:86:A6:27:1D:1D:E8:FD:D4:D4:52:2D:BD:94:24:52:FC:09:8E
         SHA256: E9:E2:46:5F:01:D3:49:B5:33:8B:4E:EC:98:DE:97:F2:C2:DC:E9:FD:55:FD:3C:FF:41:FA:99:EB:BF:8C:42:61
         Signature algorithm name: SHA1withRSA
         Version: 3
Trust this certificate? [no]:  yes
Certificate was added to keystore
[root@NAC01 NAC]#
```

5b. Update server.xml with new keystore information, just as the procedure is done in (Sec 1, G):

Open/Edit conf/server.xml, update connector information:

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
                      SSLEnabled="true"
                      maxThreads="150"
                      scheme="https"
                      secure="true"
                      clientAuth="false"
                      sslProtocol="TLS"
                      keyAlias="nac01"
                      keystoreFile="conf/custom-keystore.jks"
                      keystorePass="[plaintext password for custom-keystore]">
  </Connector>
```

6b. Redistribute the custom-truststore.jks we just updated from NAC to all NES/Agents as needed.

c. Edit jms.properties

Edit the file(NAC, NES):

> *%home%/webapps/datamanagement/WEB-INF/jms.properties*
>
> *%home%/webapps/execution/WEB-INF/jms.properties*

```
#Thu Feb 12 14:08:52 EST 2015
jms.transport.port.nes=61616
http.to.nac=false
jms.trust.store=conf/custom-truststore.jks *
jms.encrypted.key.store.password=B3BF5CD9D770DA575DE81A4D74D76249 **
jms.encrypted.trust.store.password=B3BF5CD9D770DA575DE81A4D74D76249 ***
jms.transport.port.nac=61617
jms.activate.broker=true
jms.key.store=conf/custom-keystore.jks ****
```

*   - updated path to new trust store(conf/custom-truststore.jks in our current reality).
**   - 'encrypted' version of the password for our custom key store^.

*** - 'encrypted' version of the password for our custom trust store, please reference above for further instructions.
**** - updated path to new key store(conf/custom-keystore.conf if you are following this document verbatim).

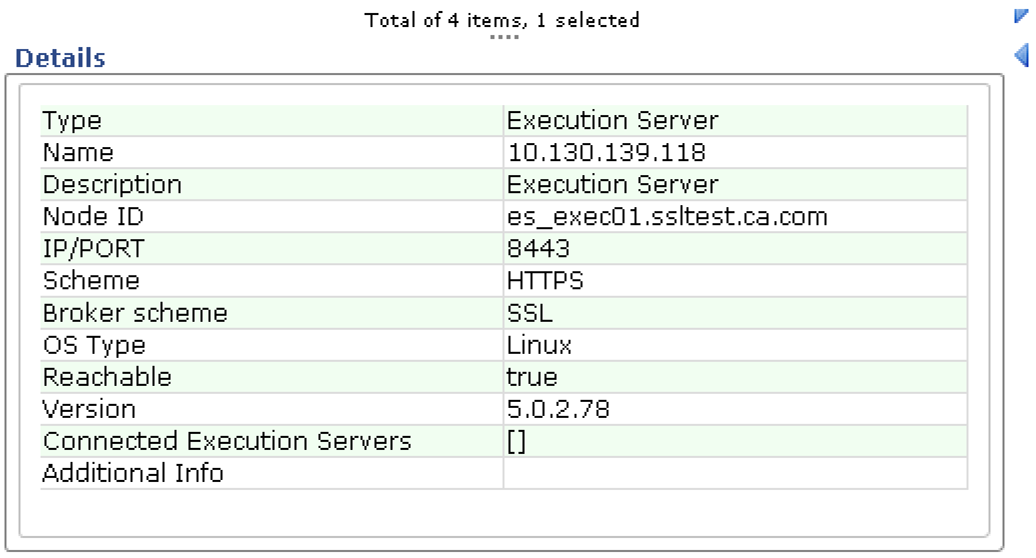
^ This password can be generated on the NAC by running:

> *scripts/encrypt_password.sh <password>*

***** Note: Due to JRE location, it is easiest to execute this from base install directory.***

d. NAC/NES are now configured to use custom/self-signed SSL cert(s).

Once all NAC/NES components are configured, further user configuration is also required. At this point you will need to restart both components if you have not. Once online, within 'Agent Management' (ASAP), you will need to add the servers as using HTTPS, port 8443. When done correctly, the information for the execution servers should look as so:



Total of 4 items, 1 selected

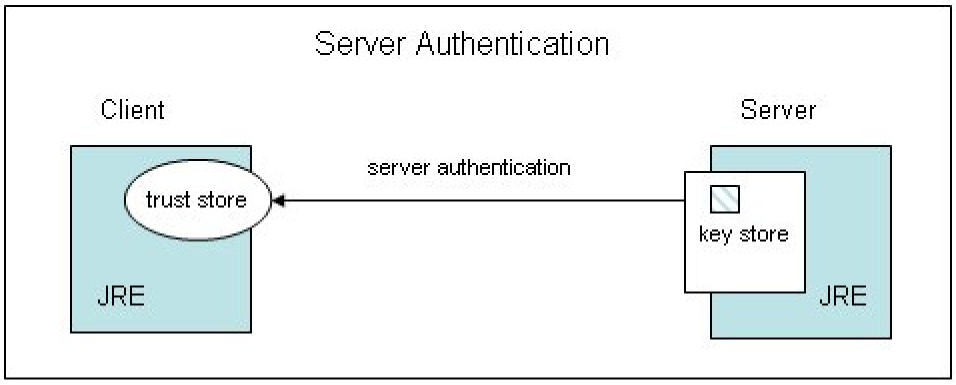
Details

Type	Execution Server
Name	10.130.139.118
Description	Execution Server
Node ID	es_exec01.ssltest.ca.com
IP/PORT	8443
Scheme	HTTPS
Broker scheme	SSL
OS Type	Linux
Reachable	true
Version	5.0.2.78
Connected Execution Servers	[]
Additional Info	

Enjoy(hopefully)!

******* Section 3. Configuration of SSL/TLS between NES<-->NAG (NiMi/TLS) *******

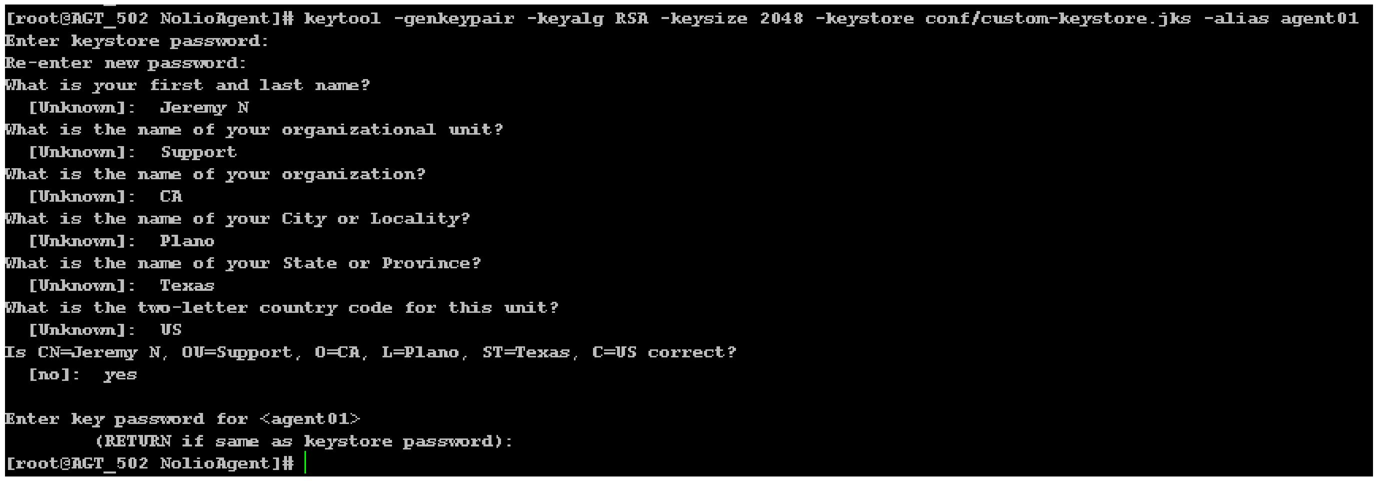
<Execution> <-NiMi-> <Deployer/Agent(NAG)>



NOTE: This section assumes /usr/local/NolioAgent as the installation root.

a. Create a custom keystore for the Nolio Agent, and generate a private/public keypair with a unique alias for the specific agent.

```
keytool -genkeypair -keyalg RSA -keysize 2048 -keystore conf/custom-keystore.jks -alias agent01
```



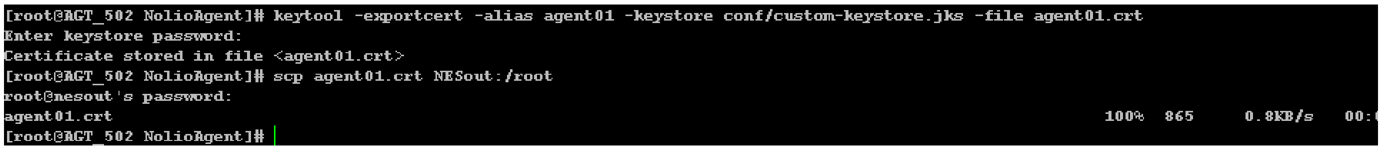
```
[root@AGT_502 NolioAgent]# keytool -genkeypair -keyalg RSA -keysize 2048 -keystore conf/custom-keystore.jks -alias agent01
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  Jeremy N
What is the name of your organizational unit?
  [Unknown]:  Support
What is the name of your organization?
  [Unknown]:  CA
What is the name of your City or Locality?
  [Unknown]:  Plano
What is the name of your State or Province?
  [Unknown]:  Texas
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Jeremy N, OU=Support, O=CA, L=Plano, ST=Texas, C=US correct?
  [no]:  yes

Enter key password for <agent01>
        (RETURN if same as keystore password):
[root@AGT_502 NolioAgent]#
```

This will create the agents new keystore, please take note of the key store password.

b. Export the agent(s) new key into a public certificate/file(agent01.crt), and copy/move to Agent NES/Execution Server:

> *keytool -exportcert -alias agent01 -file agent01.crt -keystore conf/custom-keystore.jks -v -rfc*
>
> *scp agent01.crt <NES>:/path/to/nolio*

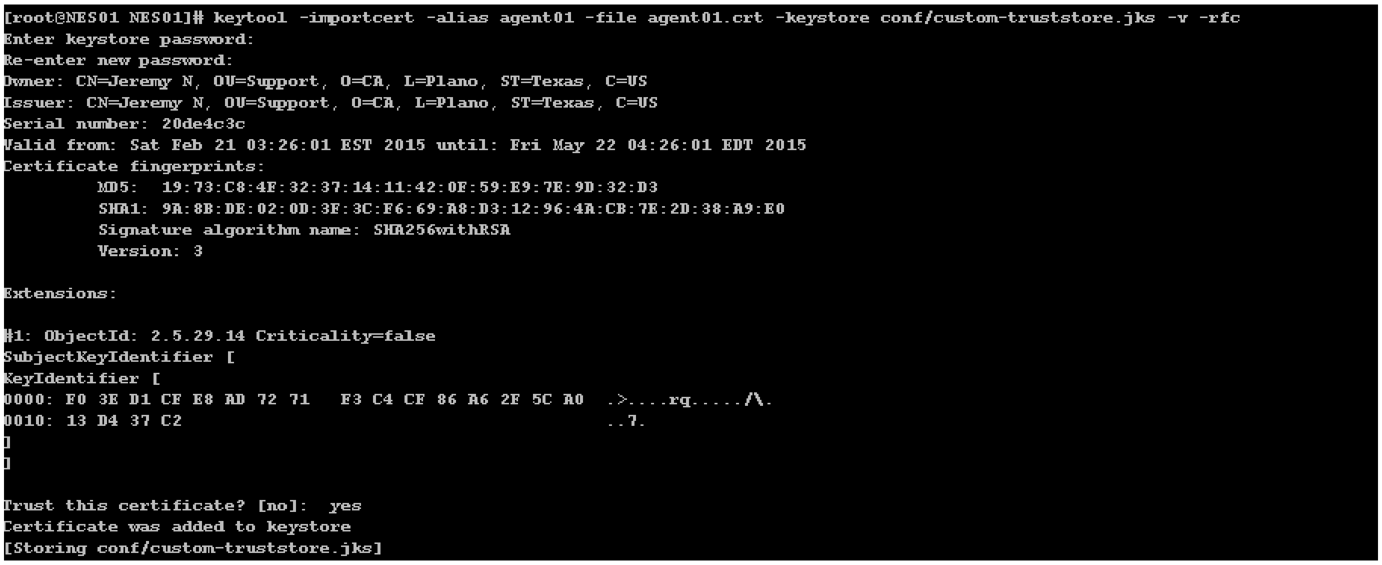
***** Note: You will be prompted for the newly created NAG custom-keystore.jks password.***



```
[root@AGT_502 NolioAgent]# keytool -exportcert -alias agent01 -keystore conf/custom-keystore.jks -file agent01.crt
Enter keystore password:
Certificate stored in file <agent01.crt>
[root@AGT_502 NolioAgent]# scp agent01.crt NESout:/root
root@nesout's password:
agent01.crt                                                                                   100%  865     0.8KB/s   00:
[root@AGT_502 NolioAgent]#
```

d. Update the NES custom-truststore.jks:

> *keytool -importcert -alias agent01 -file agent01.crt -keystore conf/custom-truststore.jks -v -rfc*



```
[root@NES01 NES01]# keytool -importcert -alias agent01 -file agent01.crt -keystore conf/custom-truststore.jks -v -rfc
Enter keystore password:
Re-enter new password:
Owner: CN=Jeremy N, OU=Support, O=CA, L=Plano, ST=Texas, C=US
Issuer: CN=Jeremy N, OU=Support, O=CA, L=Plano, ST=Texas, C=US
Serial number: 20de4c3c
Valid from: Sat Feb 21 03:26:01 EST 2015 until: Fri May 22 04:26:01 EDT 2015
Certificate fingerprints:
         MD5:  19:73:C8:4F:32:37:14:11:42:0F:59:E9:7E:9D:32:D3
         SHA1: 9A:8B:DE:02:0D:3F:3C:F6:69:A8:D3:12:96:4A:CB:7E:2D:38:A9:E0
         Signature algorithm name: SHA256withRSA
         Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: F0 3E D1 CF E8 AD 72 71   F3 C4 CF 86 A6 2F 5C A0  .>....rq...../\.
0010: 13 D4 37 C2                                        ..7.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
[Storing conf/custom-truststore.jks]
```

e. Distribution of updated trust store to other Agents(and NES in some scenarios):

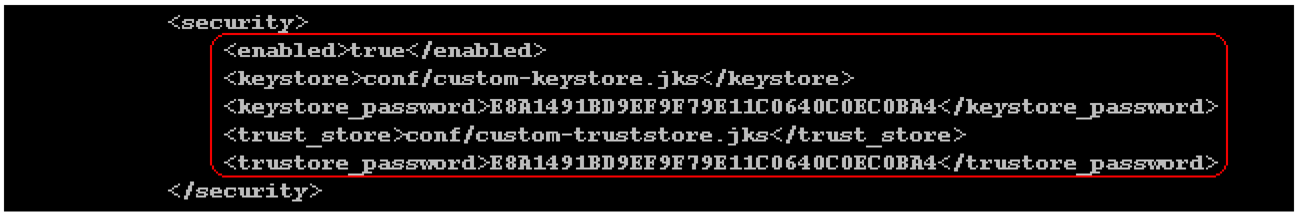
Once you have imported the certificate file into the custom-truststore.jks of the NES, you will want to update any and all NAG(Agents) custom-truststore.jks that you wish to have the ability to communicate via SSL/TLS to NES, or directly(p2p).

> - *cp custom-truststore.jks <NOLIO_AGENT_PATH/conf>*
> - *scp custom-truststore.jks agt1:/usr/local/NolioAgent/conf*

f. On the NAG and its NES, update nimi_config.xml to reflect custom settings:

Open the file 'nimi_config.xml':

> *%home%/conf/nimi_config.xml*



```
        <security>
            <enabled>true</enabled>
            <keystore>conf/custom-keystore.jks</keystore>
            <keystore_password>E8A1491BD9EF9F79E11C0640C0EC0BA4</keystore_password>
            <trust_store>conf/custom-truststore.jks</trust_store>
            <trustore_password>E8A1491BD9EF9F79E11C0640C0EC0BA4</trustore_password>
        </security>
```

Update the selected area to reflect custom settings(above).

This password(trustore_password) can be generated on the NAC by running:

> *scripts/encrypt_nimi_password.sh <password>*

***** Note: Due to JRE location, it is easiest to execute this from base install directory.***

Restart all components that have updated key and/or trust stores, and SSL/TLS NiMi is enabled between NES<->NAG.

2/21/2015 -nelje05@ca.com