OML: Migration from SHA1 signed certificates to SHA2 signed certificates

Contents

Overview

The SHA-1 hashing algorithm, which is known to be weak due to advances in cryptographic attacks upon SHA-1 is being deprecated and replaced with SHA-2. Therefore it is recommended to move to SHA-2 signed certificate. And this document helps customers to migrate to SHA-2 signed certificates.
There are two ways to do this transition:-

1. Re-setup new certificate infrastructure, by completely removing existing certificates on OM server and OA nodes.
2. Use MigrateAysmKey tool available with OvSecCs components to move to SHA-2 signed certificates on OM server and re-issue certificate on all OA nodes.

Important Notes

- Please double-check that algorithm and key-lengths are same on all nodes as well as managers.
- Agent version need to be 11.15 or 12.0x or later. For agent version 11.14 hotfix HF_QA_300615_QCCR1A181891_MULTI is required. This hotfix can be obtained from Support.
- Downtime of OM is expected while migrating from SHA1 to SHA2 due to the policies deployed before migrating to SHA2 certificate requires all policies to be redeployed post migrating to SHA2 certificates.

Option #1: Completely Setup New Certificate Infrastructure

- On OM server
  - Local agent version need to be 11.15 or 12.0x or later. For agent version 11.14 hotfix HF_QA_300615_QCCR1A181891_MULTI is required. This hotfix can be obtained from Support.
  - Bring down OM services using following command
    #ovc -stop
  - Remove existing certificates using "`ovcert -remove <alias>`" utility.[For details refer to the section- **Steps to remove all certificates on the management server**]
  - It is also recommended to move to a stronger RSA key size by setting ASYMMETRIC_KEY_LENGTH configuration under sec.cm namespace.
    - For example, [sec.cm] ASYMMETRIC_KEY_LENGTH=4096.
      # ovconfchg -ns sec.cm -set ASYMMETRIC_KEY_LENGTH 4096
    - By default this configuration is set to 2048.
  - Set HASH_ALGO configuration under sec.core namespace to a supported SHA-2 algorithm.
    - For example, [sec.core]HASH_ALGO=eSHA512
      # ovconfchg -ns sec.core -set HASH_ALGO eSHA512
  - Run the MigrateAsymKey.sh script present at "/opt/OV/lbin/seccs/install" with

option – createCAcert as explained below.

#/opt/OV/lbin/seccs/install/MigrateAsymKey.sh -createCAcert

- Run the MigrateAsymKey.sh script present at "/opt/OV/lbin/seccs/install" with option - createNodecert

#/opt/OV/lbin/seccs/install/MigrateAsymKey.sh -createNodecert

- Restart the OM services using the following commands
  #ovc -start

- On OA managed node
  - It is also recommended to move to a stronger RSA key size on the OA managed nodes by setting ASYMMETRIC_KEY_LENGTH configuration under sec.cm namespace.
  - Ex . ovconfchg -ns sec.cm -set ASYMMETRIC_KEY_LENGTH 4096
  - It is also recommended to set the hash algo
    # ovconfchg -ns sec.core -set HASH_ALGO eSHA512
  - Agent version need to be 11.15 or 12.0x or later. For agent version 11.14 hotfix HF_QA_300615_QCCR1A181891_MULTI is required. This hotfix can be obtained from Support.
  - Remove existing certificates using "`ovcert -remove <alias>`" utility. [For details refer to the section- **Steps to remove all certificates on the managed node**]
  - Request for certificate using "`ovcert -certreq`"
  - Grant the certificate request from OM Server. [For details refer to the section-
  - **Granting Certificates on the management Server**]

- After this, all the certificates in the environment (both on server and managed nodes) are SHA512 signed.

Option #2: Migrate To SHA-2 Signed Certificates Using MigrateAysmKey Tool

- On OM server
  - Local agent version need to be 11.15 or 12.0x or later. For agent version 11.14 hotfix HF_QA_300615_QCCR1A181891_MULTI is required. This hotfix can be obtained from Support.
  - Ensure that OvSecCs version is 11.10.035 or above
    - At this point, all certificates would be signed using SHA1.
    - Keystore content would look like the following snap shot.

```
#/opt/OV/bin/ovcert -list
+-------------------------------------------------------------------------+
| Keystore Content                                                        |
+-------------------------------------------------------------------------+
| Certificates:                                                           |
|         6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)                        |
+-------------------------------------------------------------------------+
| Trusted Certificates:                                                   |
|         CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa                         |
|         CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048                   |
+-------------------------------------------------------------------------+

+-------------------------------------------------------------------------------+
| Keystore Content (OVRG: server)                                               |
+-------------------------------------------------------------------------------+
| Certificates:                                                                 |
|         6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)                              |
+-------------------------------------------------------------------------------+
| Trusted Certificates:                                                         |
|         CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)                           |
|         CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048 (*)                     |
+-------------------------------------------------------------------------------+
```

    - Set HASH_ALGO configuration under sec.core namespace to desired and supported hash algorithm on OM server as well as nodes

      #ovconfchg -ns sec.core -set HASH_ALGO eSHA512

    - Move to a stronger RSA key size on the OM server as well as managed nodes by setting ASYMMETRIC_KEY_LENGTH configuration under sec.cm namespace

      # ovconfchg -ns sec.cm -set ASYMMETRIC_KEY_LENGTH 4096

- A tool called MigrateAsymKey is shipped with OvSecCs 11.10.035 that take two parameters “-createCAcert” and “-createNodecert”. Run MigrateAsymkey tool with “-createCAcert” option, this creates new CA certificate for 4096 RSA key size, signed using hash algorithm configured.

#/opt/OV/lbin/seccs/install/MigrateAsymKey.sh -createCAcert

- Signing algorithm used can be validated using "openssl" utility. [For details refer to the section- **Validating the Signature Algorithm**]

```
#/opt/OV/bin/ovcert -list
 +-------------------------------------------------------+
 | Keystore Content                                      |
 +-------------------------------------------------------+
 | Certificates:                                         |
 |     6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)          |
 +-------------------------------------------------------+
 | Trusted Certificates:                                 |
 |     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa           |
 |     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048      |
 |     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_4096      |
 +-------------------------------------------------------+

 +-------------------------------------------------------+
 | Keystore Content (OVRG: server)                       |
 +-------------------------------------------------------+
 | Certificates:                                         |
 |     6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)          |
 +-------------------------------------------------------+
 | Trusted Certificates:                                 |
 |     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)       |
 |     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048 (*)  |
 |     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_4096 (*)  |
 +-------------------------------------------------------+
```

- Create new node certificate for local agent and other keystores using MigrateAsymkey tool with "-createNodecert" option.

```
#/opt/OV/lbin/seccs/install/MigrateAsymKey.sh -createNodecert
```

- Post which the certificates are for 4096 RSA key size and signed using hash algorithm configured in HASH_ALGO.

```
#/opt/OV/bin/ovcert -list
+-------------------------------------------------------+
| Keystore Content                                      |
+-------------------------------------------------------+
| Certificates:                                         |
|     6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)          |
+-------------------------------------------------------+
| Trusted Certificates:                                 |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa           |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048      |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_4096      |
+-------------------------------------------------------+

+-------------------------------------------------------+
| Keystore Content (OVRG: server)                       |
+-------------------------------------------------------+
| Certificates:                                         |
|     6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)          |
+-------------------------------------------------------+
| Trusted Certificates:                                 |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)       |
```

```
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048 (*)     |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_4096 (*)     |
+----------------------------------------------------------+
```

- Now, all the certificate on server are signed using hash algorithm configured in HASH_ALGO, in this case its SHA512 signed. Except the old CA certificate (`CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa and CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048`).

- On all OA managed nodes
  - Agent version need to be 11.15 or 12.0x or later. For agent version 11.14 hotfix HF_QA_300615_QCCR1A181891_MULTI is required. This hotfix can be obtained from Support.
  - Existing keystore content would have the CA certificate(s) of server

```
#/opt/OV/bin/ovcert -list
+----------------------------------------------------------+
| Keystore Content                                         |
+----------------------------------------------------------+
| Certificates:                                            |
| 7c2776a6-02b1-758a-1092-f01c1e986bff (*)                 |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
| CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)              |
| CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048             |
+----------------------------------------------------------+
```

  - Update trusted certificates, using “ovcert -updatetrusted” command.

```
#/opt/OV/bin/ovcert -updatetrusted
```

  - Keystore would now have newly created CA certificate imported.

```
#/opt/OV/bin/ovcert -list
+----------------------------------------------------------+
| Keystore Content |
+----------------------------------------------------------+
| Certificates: |
| 7c2776a6-02b1-758a-1092-f01c1e986bff (*) |
+----------------------------------------------------------+
| Trusted Certificates: |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa (*)          |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_2048 (*)     |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_4096 (*)     |
+----------------------------------------------------------+
```

The above step is required, to have all the nodes retain the communication for old and new policy deployment and update.

To have the nodes with only SHA512 certificates follow below steps

- Remove all existing certificates on the node using “ovcert -remove” command.
- Ensure HASH_ALGO and key length is the same as the OM Server
- Request for new certificate using “ovcert -certreq” command and grant the same from OM server.

```
#/opt/OV/bin/ovcert -certreq
```

- Grant the certificate request from OM Server.

- Now, all the certificate on node are signed using hash algorithm configured in HASH_ALGO, in this case its SHA512 signed.

```
# ovcert -list
+-------------------------------------------------------+
| Keystore Content |
+-------------------------------------------------------+
| Certificates: |
| 7c2776a6-02b1-758a-1092-f01c1e986bff (*) |
+-------------------------------------------------------+
| Trusted Certificates: |
|                                                       |
|     CA_6ea8efa2-be27-756a-0fa7-97b1adb01ffa_4096 (*)     |
+-------------------------------------------------------+
```

Once all the Nodes are migrated to SHA512 certificates, SHA1 certificates can be deleted from OM server.

NOTE: After having new certificates on the Nodes, OML setup will not be fully operational until all the policies have been redeployed. Redeployment is required to override the policies with new certificates. Refer to HPOM Administration Guide for policy deployment.

[For details refer to the section- **Granting Certificates on the management Server**]
[For details refer to the section- **Validating the Signature Algorithm**]

Steps to remove all certificates on the management server

All steps in this sub-procedure should be taken on the management server.

1. Close any open consoles.
2. Stop all OVO management server, agent and L-core processes
   - `#/opt/OV/bin/ovc -kill`
3. Ensure that all OVO and L-core processes have stopped.
   - `#/opt/OV/bin/ovc -status`
4. Remove all certificates on the management server:

   NOTE: after taking the following steps the OML setup will not be fully operational until all steps up to and including recreating the certificates and redeploying policies on ALL the agents have been completed, which implies manual steps on ALL agents and redeployment of policies to ALL agents. Refer to HPOM Administration Guide for policy deployment.

5. Complete the following steps on the management server:

a. List the installed certificates

```
#/opt/OV/bin/ovcert -list
+-------------------------------------------------------+
| Keystore Content |
+-------------------------------------------------------+
| Certificates: |
| f84f45a2-e2df-752a-0bbc-9831e8f1e9e5 (*) |
```

```
+------------------------------------------------------------+
| Trusted Certificates: |
| CA_f84f45a2-e2df-752a-0bbc-9831e8f1e9e5 |
+------------------------------------------------------------+
+------------------------------------------------------------+
| Keystore Content (OVRG: server) |
+------------------------------------------------------------+
| Certificates: |
| f84f45a2-e2df-752a-0bbc-9831e8f1e9e5 (*) |
+------------------------------------------------------------+
| Trusted Certificates: |
| CA_f84f45a2-e2df-752a-0bbc-9831e8f1e9e5 (*) |
+------------------------------------------------------------+
```

b. Remove all four of the above certificates, one by one:

```
#/opt/OV/bin/ovcert -remove f84f45a2-e2df-752a-0bbc-9831e8f1e9e5
* Do you really want to remove the certificate with
alias 'f84f45a2-e2df-752a-0bbc-9831e8f1e9e5'
(yes(y)/no(n))? y  INFO: Certificate has been
successfully removed.
#/opt/OV/bin/ovcert -remove CA_f84f45a2-e2df-752a-0bbc-9831e8f1e9e5
* Do you really want to remove the certificate with
alias 'CA_f84f45a2-e2df-752a-0bbc-9831e8f1e9e5'
(yes(y)/no(n))? y  INFO: Certificate has been
successfully removed.
#/opt/OV/bin/ovcert -remove f84f45a2-e2df-752a-0bbc-9831e8f1e9e5 -ovrg server
* Do you really want to remove the certificate with alias
f84f45a2-e2df-752a-0bbc-9831e8f1e9e5' (yes(y)/no(n))? y
INFO: Certificate has been successfully removed.
#/opt/OV/bin/ovcert -remove CA_f84f45a2-e2df-752a-0bbc-9831e8f1e9e5 -ovrg server
* Do you really want to remove the certificate with alias
'CA_f84f45a2-e2df-752a-0bbc-9831e8f1e9e5' (yes(y)/no(n))? y
INFO: Certificate has been successfully removed.
```

c. List the installed certificates again to confirm all certificates have been deleted successfully:

```
#/opt/OV/bin/ovcert -list
+------------------------------------------------------------+
| Keystore Content                                           |
+------------------------------------------------------------+
| Certificates:                                              |
+------------------------------------------------------------+
| Trusted Certificates: |
+------------------------------------------------------------+
+------------------------------------------------------------+
| Keystore Content (OVRG: server)                            |
+------------------------------------------------------------+
| Certificates: |
+------------------------------------------------------------+
| Trusted Certificates:                                      |
+------------------------------------------------------------+
```

Granting Certificates on the management Server

- Once the correct certificate request has been identified on the management server, use the console GUI to grant the certificate

Or

- use the following command line to list the request id
  `#/opt/OV/bin/ovcm -listpending -l`
- Grant the certificate using the request id using the following command
  `#/opt/OV/bin/ovcm -grant <request id>`
  For ex. `#/opt/OV/bin/ovcm -grant b833d772-79c4-758a-1e18-e7618f5e1778`

Check on the agent that the situation is back to normal:

```
#/opt/OV/bin/ovcert -list
+------------------------------------------------------------+
| Keystore Content |
+------------------------------------------------------------+
| Certificates: |
| 169f68ea-fae5-7506-0513-9ed4449eca3d (*) |
+------------------------------------------------------------+
| Trusted Certificates: |
| CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 |
+------------------------------------------------------------+
```

Migrating the AdminUI certificates from SHA-1 to SHA-2

```
#cd /opt/OV/OMU/adminUI/conf/servicemix
#/opt/OV/bin/ovc -kill
```

IMPORTANT: Take backup and delete the existing keystore and truststore.

1) Steps for creating keystore.jks and trustore.jks

```
#/opt/OV/nonOV/jre/b/bin/keytool -genkey -keyalg RSA -alias smx  -keysize 2048
-validity 6400 -dname "CN=servicemix, OU=, O=, L=, S=, C=" -keypass password -
storepass password -keystore keystore.jks

#/opt/OV/nonOV/jre/b/bin/keytool -genkey -alias jetty -keyalg RSA -keysize 2048
-validity 6400 -dname "CN=midas-webapp, OU=Unknown, O=Unknown, L=Unknown,
S=Unknown, C=Unknown" -keypass password -storepass password -keystore
keystore.jks

#/opt/OV/nonOV/jre/b/bin/keytool -genkey -alias ftpserver -keyalg RSA -keysize
2048 -validity 6400 -dname "CN=FTP Server, OU=ftpserver, O=Apache, L=City,
S=State, C=US" -keypass password -storepass password -keystore keystore.jks
```

   i. export the certificate

```
#/opt/OV/nonOV/jre/b/bin/keytool -export -alias jetty -file client.cer -
keystore keystore.jks -storepass password
```

   ii. import the certificate

```
#/opt/OV/nonOV/jre/b/bin/keytool -import -v -trustcacerts -alias jetty -
file client.cer -keystore truststore.jks
```

When prompted for "Trust this certificate? [no]: " type yes.

2) Steps for creating keystore_webapp.jks and truststore_webapp.jks

```
# /opt/OV/nonOV/jre/b/bin/keytool -genkey -alias Hostname_server_webapp -keyalg
RSA -keysize 2048 -validity 3600 -dname "CN=Hostname, OU=HPOM Administration
UI, O=Hewlett-Packard Company" -keypass password -storepass password -keystore
keystore_webapp.jks
```

   i. export the certificate

```
#/opt/OV/nonOV/jre/b/bin/keytool -export -alias Hostname_server_webapp -
file client_webapp.cer -keystore keystore_webapp.jks -storepass password
```

   ii. import the certificate

```
#/opt/OV/nonOV/jre/b/bin/keytool -import -v -trustcacerts -alias
Hostname_server_webapp -file client_webapp.cer -keystore
truststore_webapp.jks -storepass password
```

When prompted for "Trust this certificate? [no]: " type yes.

3) Steps for creating keystore_endpoint.jks and truststore_endpoint.jks

```
# /opt/OV/nonOV/jre/b/bin/keytool -genkey -alias Hostname_server   -keyalg RSA -
keysize 2048 -validity 3600 -dname "CN=Hostname, OU=HPOM Administration UI,
O=Hewlett-Packard Company" -keypass password -storepass password -keystore
keystore_endpoint.jks
```

i. export the certificate

```
#/opt/OV/nonOV/jre/b/bin/keytool -export -alias Hostname_server   -file
client_endpoint.cer -keystore keystore_endpoint.jks -storepass password
```

ii. import the certificate

```
#/opt/OV/nonOV/jre/b/bin/keytool -import -v -trustcacerts -alias
Hostname_server -file client_endpoint.cer -keystore
truststore_endpoint.jks -storepass password
```

Start the remaining processes and check that the situation is back to normal.

```
#/opt/OV/bin/ovc -start
```

Steps to remove all certificates on the managed Node

The following output illustrates the normal behavior:

```
#/opt/OV/bin/ovcoreid
169f68ea-fae5-7506-0513-9ed4449eca3d
#/opt/OV/bin/ovcert -list
+-------------------------------------------------------------+
| Keystore Content |
+-------------------------------------------------------------+
| Certificates: |
| 169f68ea-fae5-7506-0513-9ed4449eca3d (*) |
+-------------------------------------------------------------+
| Trusted Certificates: |
| CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 |
+-------------------------------------------------------------+
```

1. Remove the trusted certificate:

```
#/opt/OV/bin/ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
```

2. Remove the node certificate:

```
#/opt/OV/bin/ovcert -remove 169f68ea-fae5-7506-0513-
9ed444 9eca3d  Do you really want to remove the
certificate with alias
'169f68ea-fae5-7506-0513-9ed4449eca3d' (yes(y)/no(n))? y
INFO: Certificate has been successfully removed
```

The following output illustrates a case where the node and the trusted certificates are removed on an agent:

```
#/opt/OV/bin/ovcert -list
+-------------------------------------------------------------+
| Keystore Content |
+-------------------------------------------------------------+
| Certificates: |
+-------------------------------------------------------------+
| Trusted Certificates: |
+-------------------------------------------------------------+
```

To fix this problem, the agent must request a new node certificate and a copy of the trusted certificate from the management server. This can be done by running the command:

```
#/opt/OV/bin/ovcert -certreq
INFO: Certificate request has been successfully triggered.
```

The above command will send a certificate request to the management server where it will be seen in the list of pending certificate requests

Validating the Signature Algorithm

Install openssl on the node to verify the certificate signature.

Steps to validate the signature algorithm of the certificate from the location

```
<OvDataDir>/shared/<ov resource group>/datafiles/sec/ks
```

Ex. On a linux node.

```
/usr/bin/openssl x509 -text -noout -inform pem -in
/var/opt/OV/datafiles/sec/ks/CA_437e80e2-b410-7565-1775-a91aa8056e6a_4096_cert.pem
```

In the output look for the field `Signature Algorithm:` It represents the signature algorithm

```
Certificate: Data:
        Version: 3 (0x2) Serial Number: 0
        (0x0)
        Signature Algorithm: sha512WithRSAEncryption
        Issuer: CN=CA_437e80e2-b410-7565-1775-a91aa8056e6a_4096, L=XXXXXXXXXXXX, O=XXXXXXXXXX, OU=XXXXXX
        Validity
            Not Before: Mar 17 13:56:59 2013 GMT
            Not After : Mar 13 13:56:59 2033 GMT
        Subject: CN=CA_437e80e2-b410-7565-1775-a91aa8056e6a_2048, L=XXXXXXXXXXXXXXX,
O=XXXXXXXXXX, OU=XXXXXXX
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key:                Exponent: 3 (0x3)

----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
-------------------------
        X509v3 extensions:
            X509v3 Authority Key Identifier:
                DirName:/CN=CA_437e80e2-b410-7565-1775-
a91aa8056e6a_2048/L=XXXXXXXXXXXXX/O=XXXXXXX/OU=XXXXXXX
                serial:00
```

On an existing windows node :

```
C:\openssl\openssl x509 -text -noout -inform pem -in ”C:\ProgramData\HP\HP BTO
Software\datafiles\sec\ks\CA_437e80e2-b410-7565-1775-a91aa8056e6a_4096_cert.pem”
```

MoM Setup (Agent nodes connecting to only one of the MOM server)

1) Make sure the MoM setup is done according to the documentation

2) On the Node, and both the servers set the hash_key_algo and the key_length
   /opt/OV/bin/ovconfchg -ns sec.core -set HASH_ALGO eSHA512
   /opt/OV/bin/ovconfchg -ns sec.cm -set ASYMMETRIC_KEY_LENGTH 4096

3) Change the primary server to Node 1 using the following command

   /opt/OV/bin/OpC/opcragt -primmgr Node-name

4) To change the certificates on server 1 (primary server)

   1) In case of MoM from the secondary server 2 run the following command

      #/opt/OV/bin/OpC/opcragt -primmgr Node-name

After the above steps, send a message to confirm communication is working fine

   2) On the server 1 (Primary) run the following commands

      a) /opt/OV/lbin/seccs/install/MigrateAsymkey.sh –createCAcert
      b) /opt/OV/lbin/seccs/install/MigrateAsymkey.sh –createNodecert

   3) Shift the server to server 1 by running the following command on server 1 in case of a MoM scenario

      #/opt/OV/bin/OpC/opcragt -primmgr Node-name

In case of a server pooling scenario, follow the OS specific commands (provided above)

Reconfirm communication is intact by sending a message from the node to the server. All the messages should arrive to server 1 (Primary) .

   4) On server2 run the following commands
      /opt/OV/lbin/seccs/install/MigrateAsymkey.sh –createCAcert
      /opt/OV/lbin/seccs/install/MigrateAsymkey.sh –createNodecert

   5) Run the following command on nodes
      #/opt/OV/bin/ovcert –updatetrusted

   6) On server 1 run the following command
      #/opt/OV/bin/ovcert -exporttrusted -file /tmp/S1.cert -ovrg server
      and copy the file to server 2. On server 2 run the following commands
      a) #/opt/OV/bin/ovcert -importtrusted -file /tmp/S1.cert -ovrg server
      b) #/opt/OV/bin/ovcert -importtrusted -file /tmp/S1.cert

   7) On server2 run the following command

#/opt/OV/bin/ovcert -exporttrusted -file /tmp/S2.cert -ovrg server
and copy the file to server1 and run the following commands

a) #/opt/OV/bin/ovcert -importtrusted -file /tmp/S2.cert -ovrg server
b) #/opt/OV/bin/ovcert -importtrusted -file /tmp/S2.cert

8) run ovcert –updatetrsuted on both the Secondary and Primary and finally on the nodes

Verify by switching back the primary server to server 2 using the following command

#/opt/OV/bin/OpC/opcragt -primmgr Node-name

From both the servers delete the 2048 certificates using ovcert -remove command

NOTE: after taking the above steps redeploy policies on ALL the agent nodes. Refer to HPOM Administration Guide for policy deployment.

Migrating SHA1 to SHA2 in a Clustered environment (Basic cluster)

1) On the active server , run the following command

a) To disable the switchover
#/opt/OV/lbin/ovharg -monitor ov-server disable
c) #/opt/OV/bin/ovconfchg -ns sec.core -set HASH_ALGO eSHA512
d) #/opt/OV/bin/ovconfchg -ns sec.cm -set ASYMMETRIC_KEY_LENGTH 4096
e) #/opt/OV/lbin/seccs/install/MigrateAsymKey.sh –createCAcert
f) #opt/OV/lbin/seccs/install/MigrateAsymKey.sh –createNodecert

2) Run the below command to check if all the process are running
#/opt/OV/bin/ovc.
3) Enable the switch over using the following command

a) #/opt/OV/lbin/ovharg -monitor ov-server enable
b) Switch over to the other server.

#/opt/OV/bin/ovharg_config ov-server -switch <server 2>

c) Disable the switchover on the serve2(Active node) by running the following command

#/opt/OV/lbin/ovharg -monitor ov-server disable

4) Run the below command to check if all process are up and running
#/opt/OV/bin/ovc

Note: In case the ovcs(ov certificate server) process shows aborted state, run the below command

#/opt/OV/lbin/seccs/install/MigrateAsymKey.sh –createNodecert

#/opt/OV/bin/ovc –start ovcs

5) Enable the monitoring by running the below command

   #/opt/OV/lbin/ovharg -monitor ov-server enable

6) On all the nodes run
   #ovcert –updatetrusted
   Note: If buffering messages is observed on the node then run the command for switchover

7) On the server and nodes remove the older SHA1 certificates using ovcert -remove command

   NOTE: after taking the above steps redeploy policies on ALL the agent nodes. Refer to HPOM Administration Guide for policy deployment.