The OV Certificates Cookbook

OVO/Unix 8.x
Version 1.0
2004/12/09

Known issues with this version:

- Procedures that are specific to the OVO management server running on a cluster as a package or resource group have not been tested
- No procedures worked out for Flexible Management Setups

Feedback to thierry.ledent@hp.com

Index

1 Certificate problems on the OVO agent

1.1 Normal situation

The following output illustrates a normal situation:

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

This output confirms that the node certificate and the trusted certificate are installed. The star next to the node certificate indicates that the node’s private key is available. Note that the node certificate is named after the node’s coreid.

The trusted certificate is named after the coreid of the trusted certificate authority, with the prefix “CA_”. In the case of OVO, the management server takes the role of trusted certificate authority.

In a flexible management environment, the OVO agent may be configured to report to multiple OVO management servers. In this case the node will still have only one node certificate but it will need a copy of the trusted certificate of all OVO management servers. The procedures in this version of the cookbook do not yet account for this case.

1.2 Missing node certificate

The following output illustrates a case where the node certificate is missing:

```
agent# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993             |
+---------------------------------------------------------+
```

To fix this problem, first remove the trusted certificate:

```
agent# ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
         'CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
```

```
INFO:    Certificate has been successfully removed.
```

Now continue with the procedure described in Missing node and trusted certificate.

1.3 Missing trusted certificate

The following output illustrates a case where the trusted certificate is missing on an agent:

```
agent# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
|     169f68ea-fae5-7506-0513-9ed4449eca3d (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
+---------------------------------------------------------+
```

1.3.1 Solution 1

To fix this problem, the agent can import the trusted certificate from the management server or another agent managed by the same management server.

First, on the management server or on another agent managed by the same management server, export the trusted certificate:

```
agent# ovcert -exporttrusted -file /tmp/trustedcertif
INFO:    Trusted certificates have been successfully exported to file '/tmp/
         trustedcertif'.
```

Next, copy the file /tmp/trustedcertif to the problem agent and import the trusted certificate:

```
agent# ovcert -importtrusted -file /tmp/trustedcertif
INFO:    Import operation was successful.
```

Finally, check on the problem agent that the situation is back to normal.

1.3.2 Solution 2

Alternatively, it is possible to remove the node certificate and then proceed with procedure Missing node and trusted certificate.

To remove the node certificate:

```
agent# ovcert -remove 169f68ea-fae5-7506-0513-9ed4449eca3d
       * 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.
```

Now continue with procedure Missing node and trusted certificate.

1.4 Missing node and trusted certificate

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

```
agent# 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.

First stop all OVO agent and L-core processes:

```
agent# ovc –kill
```

It is quite common that some processes will not stop or that “ovc” will report an error. This is due to the fact that some processes communicate locally through HTTPS and you are currently resolving a problem with certificates, which may adversely affect HTTPS communication. You will have to kill these processes manually, for instance:

```
agent# ps –ef | grep –i ov
    root 17952 17951  0 10:49:20 ?         0:01 /opt/OV/bin/ovbbccb -nodaemon
    root 17951     1  0 10:49:20 ?         1:48 /opt/OV/bin/ovcd
agent# kill 17952
agent# kill 17951
agent# ps –ef | grep –i ov
agent# ps –ef | grep –i opc
agent# ps –ef | grep –o coda
```

After killing a process, verify that it was indeed stopped. If necessary, use “`kill -9`”.

Now restart the control daemon and communication broker only:

```
agent# ovc –start CORE
```

You should see something similar to:

```
agent# ovc
ovcd        OV Control                          CORE         (17834)  Running
ovbbccb     OV Communication Broker             CORE         (17835)  Running
ovconfd     OV Config and Deploy                COREXT                Stopped
coda        OV Performance Core                 AGENT,CODA            Stopped
opcmsga     OVO Message Agent                   AGENT,EA              Stopped
```

```
opcacta     OVO Action Agent                    AGENT,EA              Stopped
opcmsgi     OVO Message Interceptor             AGENT,EA              Stopped
```

The agent should now automatically have sent a certificate request to the OV management server. To verify this, first check the coreid on the agent:

```
agent# ovcoreid
169f68ea-fae5-7506-0513-9ed4449eca3d
```

On the management server, verify that there is a pending certificate request for the agent:

```
mgmtsv# ovcm -listpending -l

RequestID:    0a878f5c-8b52-7508-0776-f107499f74c2
Context:
CN:           169f68ea-fae5-7506-0513-9ed4449eca3d
Nodename:     mcsc-sy1.bel.hp.com
IPAddress:    16.56.172.161
Platform:     HP-UX 11.11, CPU: PARisc
InstallType:  Manual
TimeReceived: 11/25/04 03:51:26 PM MET
```

Check if there is a pending certificate request where the CN field corresponds to the agent’s coreid and verify that the TimeReceived field corresponds to the time when the control daemon was restarted. If not, you may need to manually generate a certificate request on the agent:

```
agent# ovcert -certreq
INFO:    Certificate request has been successfully triggered.
```

Once you have identified the correct certificate request on the management server, use the RequestID to grant the certificate:

```
mgmtsv# ovcm -grant 0a878f5c-8b52-7508-0776-f107499f74c2
```

On the agent, you should now see the certificates:

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

Finally, you may start the remaining agent processes and check that the situation is back to normal.

```
agent# ovc -start
```

Note that there are other methods to install the node certificate and trusted certificate on an agent. The methods are described in the “HTTPS Agent Concepts and Configuration Guide”.

1.5 Missing node private key

The following output illustrates a case where the node private key is missing:

```
agent# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
|     169f68ea-fae5-7506-0513-9ed4449eca3d                |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993             |
+---------------------------------------------------------+
```

Comparing this to the output in Normal situation, you will note that the star is missing next to the node certificate name.

It is not possible to recover a lost private key. To fix this problem, the agent must request a new node certificate. First remove the current certificates:

```
agent# ovcert -remove 169f68ea-fae5-7506-0513-9ed4449eca3d
       * 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.
agent# ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
         'CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
```

Now proceed with the procedure document in Missing node and trusted certificate.

1.6 How to recreate certificates

Check what certificates are currently installed on the node:

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

Remove each installed certificate. For instance, in above example, the node certificate and trusted certificate are currently installed, so remove them with:

```
agent# ovcert -remove 169f68ea-fae5-7506-0513-9ed4449eca3d
       * 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.
agent# ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
         'CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
```

Now proceed with the procedure document in Missing node and trusted certificate.

2 Certificate problems on the management server

2.1 Normal situation

The following output illustrates a normal situation:

```
mgmtsv# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993             |
+---------------------------------------------------------+

+---------------------------------------------------------+
| Keystore Content (OVRG: server)                         |
+---------------------------------------------------------+
| Certificates:                                           |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)         |
+---------------------------------------------------------+
```

Note that the node (dcd0c94c-cb7d-7506-079a-9cc1b0282993) and server (dcd0c94c-cb7d-7506-079a-9cc1b0282993) certificates are two instances of the same certificate. The stars indicate that the private key corresponding to this certificate is available to both the node and the server.

The node certificate will be used by any OV application, such as the OVO agent, that is not registered as a separate OV resource group (OVRG).

The OVO management server registers as OV resource group “server” and therefore will use the server certificate. If the server is running on a cluster as a package or resource group, the node and server certificates are different, but the node and server should still each have access to their own private key.

An instance of the same trusted certificate is installed on the node and server, but the private key corresponding to the trusted certificate is only available to the server.

In a flexible management environment, the OVO agent may be configured to report to multiple OVO management servers. In this case the node will still have only one node certificate but it will need a copy of the trusted certificate of all OVO management servers. The procedures in this version of the cookbook do not yet account for this case.

2.2 Missing node certificate

The following output illustrates a case where the node certificate is missing:

```
mgmtsv# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993             |
+---------------------------------------------------------+

+---------------------------------------------------------+
| Keystore Content (OVRG: server)                         |
+---------------------------------------------------------+
| Certificates:                                           |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)         |
+---------------------------------------------------------+
```

The procedure to fix this problem depends on whether the OVO management server runs standalone or as a package or resource group on a cluster.

2.2.1 The OVO management server is standalone

To fix this problem, export the server certificate and import it on the node:

```
mgmtsv# ovcert -exportcert -file /tmp/certif -pass mypass -ovrg server
INFO:    Certificate has been successfully exported to file '/tmp/certif'.
mgmtsv# ovcert -importcert -file /tmp/certif -pass mypass
INFO:    Import operation was successful.
mgmtsv# rm /tmp/certif
```

2.2.2 The OVO management server runs on a cluster as a package or resource group

NOTE: this procedure was not tested.

To fix this problem, you must issue and import a new node certificate.

First put the package or resource group in maintenance mode to prevent a switch. Then stop all OVO agent and L-core processes:
:

```
mgmtsv# ovc –kill
```

It is quite common that some processes will not stop or that “ovc” will report an error. This is due to the fact that some processes communicate locally through HTTPS and you are currently resolving a problem with certificates, which may adversely affect HTTPS communication. You will have to kill these processes manually, for instance:

```
mgmtsv# ps –ef | grep ov
```

```
    root 17952 17951  0 10:49:20 ?         0:01 /opt/OV/bin/ovbbccb -nodaemon
    root 17951     1  0 10:49:20 ?         1:48 /opt/OV/bin/ovcd
    root    18 27141  0 17:55:24 pts/1    0:00 grep ov
    root 29902 29886  0 17:53:25 ?        0:00 ovtopmd -O
    root  1088     1  0   Dec 03 ?        0:01 /opt/OV/lbin/xpl/trc/ovtrcd
    root 29887 29886  0 17:53:21 ?        0:00 ovsessionmgr
    root 29886     1  0 17:53:20 ?        0:00 ovspmd -U
    root 29888 29886  0 17:53:21 ?        0:01 ovwdb -O
     bin 29892 29886  0 17:53:21 ?        0:02 ovrequestd -s
    root 29903 29886  0 17:53:25 ?        0:00 ovtrapd
     bin 29904 29886  0 17:53:25 ?        0:00 ovactiond
    root 29918 29886  0 17:53:36 ?        0:07 ovas
    root 29922 29919  0 17:53:37 ?        0:00 ovoareqhdlr
    root 29905 29886  0 17:53:25 ?        0:01 ovalarmsrv
    root 29907 29886  0 17:53:25 ?        0:00 ovdbcheck -ovspmd
    root 29919 29886  0 17:53:36 ?        0:01 ovoareqsdr -start
    root 29911 29886  0 17:53:26 ?        0:00 ovuispmd -O
    root 29910     1  0 17:53:25 ?        0:00 /opt/OV/bin/ovdbrun -c
/var/opt/OV/share/databases/analysis/default
mgmtsv# kill 17952
mgmtsv# kill 17951
mgmtsv# ps –ef | grep opc
ovec1@/etc/opt/OV/share/conf: ps -ef | grep opc
    root    35 27141  0 17:57:39 pts/1    0:00 grep opc
    root 29946 29923  0 17:53:42 ?        0:01 opccsad
    root 29938 29923  0 17:53:42 ?        0:05 opcdispm
    root 29950 29923  0 17:53:43 ?        0:01 opcsvcm
    root 29941 29923  0 17:53:42 ?        0:00 opcdistm
    root 29949 29923  0 17:53:43 ?        0:00 opcbbcdist
    root 29935 29923  1 17:53:41 ?        1:42 opcmsgm
    root 29923 29886  0 17:53:37 ?        0:04 opcctlm -start
    root 29936 29923  0 17:53:41 ?        0:00 opcttnsm
    root 29937 29923  0 17:53:42 ?        0:00 opcforwm
    root 29925 29919  0 17:53:38 ?        0:00 opcmsgrd
    root 29934 29923  0 17:53:41 ?        0:00 opcactm
    root 29924 29919  0 17:53:38 ?        0:28 opcmsgrb
mgmtsv# ps –ef | grep coda
```

After killing a process, verify that it was indeed stopped. If necessary, use "`kill -9`". Take good care not to kill NNM or OVO management server processes.

You can now issue and import the new node certificate:

```
mgmtsv# ovcm -issue -file /tmp/certif -name $(hostname) -pass mypass -coreid
$(ovcoreid)
INFO:    Issued certificate was written to file '/tmp/certif'.
mgmtsv# ovcert -importcert -file /tmp/certif -pass mypass
INFO:    Import operation was successful.
mgmtsv# rm /tmp/certif
```

Finally restart the agent processes and check that the situation is back to normal.

```
mgmtsv# ovc –start
```

Remember to turn off maintenance mode for the package or resource group.

2.3 Missing server certificate

The following output illustrates a case where the server certificate is missing:

```
mgmtsv# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993             |
+---------------------------------------------------------+

+---------------------------------------------------------+
| Keystore Content (OVRG: server)                         |
+---------------------------------------------------------+
| Certificates:                                           |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)         |
+---------------------------------------------------------+
```

The procedure to fix this problem depends on whether the OVO management server runs standalone or as a package or resource group on a cluster.

2.3.1 The OVO management server is standalone

To fix this problem, export the node certificate and import it on the server:

```
mgmtsv# ovcert -exportcert -file /tmp/certif -pass mypass
INFO:    Certificate has been successfully exported to file '/tmp/certif'.
mgmtsv# ovcert -importcert -file /tmp/certif -pass mypass -ovrg server
INFO:    Import operation was successful.
mgmtsv# rm /tmp/certif
```

2.3.2 The OVO management server runs on a cluster as a package or resource group

NOTE: this procedure was not tested.

To fix this problem, issue and import a new server certificate:

```
mgmtsv# ovcm -issue -file /tmp/certif -name $(hostname) -pass mypass -coreid
$(ovcoreid –ovrg server)
INFO:    Issued certificate was written to file '/tmp/certif'.
mgmtsv# ovcert -importcert -file /tmp/certif -pass mypass –ovrg server
INFO:    Import operation was successful.
mgmtsv# rm /tmp/certif
```

Upon deployment of policies, the management server signs and caches the policies. Since the management server’s private key has now changed, it is recommend to manually clear the cache in order to ensure that policies will be signed with the new server private key:

To clear the policy cache:

```
mgmtsv# cd /etc/opt/OV/share/conf/OpC/mgmt_sv/templates
mgmtsv# rm */*/*
```

Next, clear any pending distributions:

```
mgmtsv$ rm /var/opt/OV/share/tmp/OpC/distrib/*
```

Now, restart all OVO and L-core processes to ensure that they pick up the new certificate:

```
mgmtsv# ovstop opc ovoacomm
mgmtsv# ovc –kill
mgmtsv# ps –ef
```

Ensure that all processes have stopped. It is quite common that some processes will not stop or that “ovc –kill” will report an error. This is due to the fact that some processes communicate locally through HTTPS and not all processes have picked up the new certificate yet. You will have to kill these processes manually.

```
mgmtsv# ovc -start
mgmtsv# ovstart ovoacomm opc
```

2.4 Missing trusted certificate

2.4.1 Missing trusted certificate on the node side

The following output illustrates a case where the trusted certificate is missing from the node side:

```
mgmtsv# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
+---------------------------------------------------------+

+---------------------------------------------------------+
| Keystore Content (OVRG: server)                         |
+---------------------------------------------------------+
| Certificates:                                           |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)         |
+---------------------------------------------------------+
```

To fix this problem, export the trusted certificate from the server and import it on the node:

```
mgmtsv# ovcert -exporttrusted -file /tmp/certif -ovrg server
INFO:    Trusted certificates have been successfully exported to file '/tmp/
         certif'.
```

```
mgmtsv# ovcert -importtrusted -file /tmp/certif
INFO:    Import operation was successful.
```

2.4.2 Missing trusted certificate on the server side

The following example illustrates a case where the trusted certificate is missing on the server side:

```
mgmtsv# ovcert -list
+----------------------------------------------------------+
| Keystore Content                                         |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993              |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                          |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
+----------------------------------------------------------+
```

To fix this problem restore the server certificates from the backup.

First, stop all OVO and L-core processes:

```
mgmtsv# ovstop opc ovoacomm
mgmtsv# ovc –kill
mgmtsv# ps –ef | grep ov
mgmtsv# ps –ef | grep opc
mgmtsv# ps –ef | grep coda
```

Ensure that all processes have stopped. It is quite common that some processes will not stop or that “ovc –kill” will report an error. This is due to the fact that some processes communicate locally through HTTPS and you are currently resolving a problem with certificates, which may adversely affect HTTPS communication. You will have to kill these processes manually.

Next, remove all remaining certificates, for instance in the case of above example:

```
mgmtsv# ovcert -remove dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
         'dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
mgmtsv# ovcert -remove dcd0c94c-cb7d-7506-079a-9cc1b0282993 -ovrg server
       * Do you really want to remove the certificate with alias
         'dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
mgmtsv# ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
```

```
         'CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
```

Now, import the certificates from the certificates backup file (assuming /tmp/svr_certificates.bkp):

```
mgmtsv# opcsvcertbackup -force -restore -passwd mypwd -file
/tmp/svr_certificates.bkp
Info: Performing restore of OVO Server certificate data.
      Archive is /tmp/svr_certificates.bkp.
Info: Determining core IDs ...
Info: Local system is not member of a HA cluster.
(ctrl-111) Ovcd is not yet started.
Info: ovc running. Killing due to -force ...
Info: Unpacking archived certificate files from /tmp/svr_certificates.bkp ...
x /tmp/dcd0c94c-cb7d-7506-079a-9cc1b0282993.phys.cert, 2066 bytes, 5 tape
blocks
x /tmp/dcd0c94c-cb7d-7506-079a-9cc1b0282993.log.cert, 2066 bytes, 5 tape blocks
x /tmp/trusted.phys.cert, 1229 bytes, 3 tape blocks
x /tmp/trusted.log.cert, 1229 bytes, 3 tape blocks
x /tmp/CA.cert, 2073 bytes, 5 tape blocks
x /tmp/opcsvcertbackup.20041126_092105.txt, 250 bytes, 1 tape blocks
Info: OVO Certificate backup archive
      Created on:         Fri Nov 26 09:21:06 MET 2004
      Hostname:           ovec1
      Physical Core ID:   dcd0c94c-cb7d-7506-079a-9cc1b0282993
      Logical HA Core ID: dcd0c94c-cb7d-7506-079a-9cc1b0282993
Info: Validating core ID in archive ...
Info: Core ID in archive matches local core ID.
Info: Importing server certificates ...
INFO:    Import operation was successful.
INFO:    Import operation was successful.
Info: Importing trusted certificates ...
INFO:    Import operation was successful.
INFO:    Import operation was successful.
Info: Importing CA certificate ...
INFO:    Import operation was successful.
Info: All done. Exiting.
```

Finally, restart all OVO and L-core processes:

```
mgmtsv# ovc -start
mgmtsv# ovstart ovoacomm opc
```

Refer to Certificates best practices for explanations on how to backup certificates on the OVO management server.

If no backup is available or the password of the backup file is lost, you are in trouble. You must then proceed with the steps described in How to remove and recreate all certificates.

2.4.3 Missing trusted certificate on both sides

The following example illustrates a case where the trusted certificate is missing on the node and the server side:

```
mgmtsv# ovcert -list
```

```
+----------------------------------------------------------+
| Keystore Content                                         |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                          |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
+----------------------------------------------------------+
```

To fix this problem, proceed with the same steps as in Missing trusted certificate on the server side.

2.5 Missing server certificate and trusted certificate on the server side

The following example illustrates a case where the server certificate is missing and the trusted certificate is missing on the server side:

```
mgmtsv# ovcert -list
+----------------------------------------------------------+
| Keystore Content                                         |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993              |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                          |
+----------------------------------------------------------+
| Certificates:                                            |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
+----------------------------------------------------------+
```

To fix this problem, proceed with the same steps as in Missing trusted certificate on the server side.

2.6 Missing node private key

The following output illustrates a case where the node private key is missing:

```
mgmtsv# ovcert -list
+----------------------------------------------------------+
| Keystore Content                                         |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993                  |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993              |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                          |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)          |
+----------------------------------------------------------+
```

To fix this problem first remove the node certificate:

```
mgmtsv# ovcert -remove dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
         'dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
```

Then proceed with the same steps as in Missing node certificate.

2.7 Missing server private key

The following output illustrates a case where the node private key is missing:

```
mgmtsv# ovcert -list
+----------------------------------------------------------+
| Keystore Content                                         |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993              |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                          |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993                 |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)          |
+----------------------------------------------------------+
```

To fix this problem, first remove the server certificate:

```
mgmtsv# ovcert -remove dcd0c94c-cb7d-7506-079a-9cc1b0282993 -ovrg server
       * Do you really want to remove the certificate with alias
         'dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
```

Then proceed with the same steps as in Missing server certificate.

2.8 Missing trusted authority’s private key

The following example illustrates a case where the trusted authority’s private key is missing:

```
mgmtsv# ovcert -list
+----------------------------------------------------------+
| Keystore Content                                         |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993              |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                          |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993              |
+----------------------------------------------------------+
```

To fix this problem, first remove the trusted certificate from the server side:

```
mgmtsv# ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 –ovrg server
       * Do you really want to remove the certificate with alias
         'CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
```

Then proceed with the same steps as in Missing trusted certificate on the server side.

3 How to identify and remove invalid or corrupt certificates

The fields ‘Issuer CN’ and ‘Valid from’ in the trusted certificate on the server are used as a reference to determine if other certificates are valid.

On the management server, printout the trusted certificate as follows:

```
mgmtsv# ovcert -list
+----------------------------------------------------------+
| Keystore Content                                         |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993              |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                          |
+----------------------------------------------------------+
| Certificates:                                            |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)             |
+----------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)          |
+----------------------------------------------------------+

mgmtsv# ovcert -certinfo CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 –ovrg server

Type       :  X509Certificate
Subject CN :  CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Subject DN :  L:  ovec1.bel.hp.com
              O:  Hewlett-Packard
              OU: OpenView
              CN: CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Issuer CN  :  CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Issuer DN  :  L:  ovec1.bel.hp.com
              O:  Hewlett-Packard
              OU: OpenView
              CN: CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Serial no. :  00
Valid from :  11/24/04 04:15:56 PM GMT
Valid to   :  11/20/24 04:15:56 PM GMT
Hash (SHA1):  EE:31:26:17:7B:82:DD:A5:3B:13:C1:96:B8:2C:22:7D:74:D0:BF:BC
```

On all agents, including the agent running on the management server, check that the same trusted certificate is installed, for instance:

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

agent# ovcert -certinfo CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993

Type       :  X509Certificate
Subject CN :  CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Subject DN :  L:  ovec1.bel.hp.com
              O:  Hewlett-Packard
              OU: OpenView
              CN: CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Issuer CN  :  CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Issuer DN  :  L:  ovec1.bel.hp.com
              O:  Hewlett-Packard
              OU: OpenView
              CN: CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Serial no. :  00
Valid from :  09/28/04 15:23:43 GMT
Valid to   :  09/24/24 15:23:43 GMT
Hash (SHA1):  EA:94:F9:14:17:58:1D:D1:CC:69:18:30:65:B7:0A:E1:92:20:29:E8
```

To ensure that the trusted certificates are the same on all agents, compare at least the fields ‘Issuer CN’ and ‘Valid from’. These fields should be identical. In the above example, the agent’s trusted certificate is different from the server’s trusted certificate. The agent’s instance of the trusted certificate is therefore invalid and should be removed.

All other certificates on the server and agents should have been issued by the same trusted authority and after the current trusted certificate was installed on the server. To ensure this, compare again the field ‘Issuer CN’ of the tested certificate against the trusted certificate on the server and verify that the ‘Valid from’ date is more recent than the trusted certificate’s ‘Valid from’ date. In our example, the following node certificate is valid:

```
agent# ovcert -certinfo 169f68ea-fae5-7506-0513-9ed4449eca3d

Type       :  X509Certificate
Subject CN :  169f68ea-fae5-7506-0513-9ed4449eca3d
Subject DN :  L:  mcsc-sy1.bel.hp.com
              O:  Hewlett-Packard
              OU: OpenView
              CN: 169f68ea-fae5-7506-0513-9ed4449eca3d
Issuer CN  :  CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Issuer DN  :  L:  ovec1.bel.hp.com
              O:  Hewlett-Packard
              OU: OpenView
              CN: CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
Serial no. :  09
Valid from :  11/24/04 22:08:54 GMT
Valid to   :  11/20/24 22:08:54 GMT
Hash (SHA1):  7B:4C:9F:7C:67:7D:C6:47:52:D2:1E:D5:ED:DC:65:EF:41:63:4E:4F
```

Any certificate that does not satisfy the above described criteria is invalid and should be removed. After removing all invalid certificates, proceed with the appropriate steps from previous chapters to recreate valid certificates.

These criteria are however not sufficient to guarantee that certificates are valid. If in doubt, consider removing suspect certificates and recreating them according to the procedures described in earlier sections.

NOTE: never remove or recreate the trusted certificate of the server, unless you have no other choice and know what you are doing.

4 How to remove and recreate all certificates

This procedure is very long and involves manual steps on all agents and redeployment of policies to all agents. It should only be used in last resort when no other option is available. For instance, this procedure may be considered if the private key of the certificate authority has been lost or compromised.

This procedure consists of several subprocedures:

Remove all certificates on the management server
Recreate the trusted certificate on the management server
Recreate the server and node certificate on the management server
Backup the certificates and private keys on the management server
Prepare the management server for certificate and policy deployment
Redeploy policies to the management server
Redeploy policies to the agent on the management server
Recreate the certificates and redeploy policies on all the agents

These subprocedures are designed to be run in sequence. It is not safe to jump directly to a subprocedure until you have completed all previous subprocedures. Once you have started with the first subprocedure, you must complete all subprocedures to recover a fully operational OVO setup.

4.1 *Remove all certificates on the management server*

All steps in this subprocess should be taken on the management server.

If the OVO management server runs on a cluster as a package or resource group, first put the package or resource group into maintenance mode to avoid it from switching to another node.

Stop all OVO management server, agent and L-core processes:

```
mgmtsv# ovstop opc ovoacomm
mgmtsv# ovc -kill
mgmtsv# ps -ef | grep ov
mgmtsv# ps -ef | grep opc
mgmtsv# ps -ef | grep coda
```

Ensure that all OVO and L-core processes have stopped. It is quite common that some processes will not stop or that “ovc” will report an error. This is due to the fact that some processes communicate locally through HTTPS and you are currently resolving a problem with certificates that may adversely affect HTTPS communication. You will have to kill these processes manually. Use “kill -9” if necessary.

Now remove all certificates on the management server:

NOTE: after taking the following steps the OVO setup will not be fully operational until you proceed with all steps up to and including Recreate the certificates and redeploy policies on all the agents, which implies manual steps on all agents and redeployment of policies to all agents.

```
mgmtsv# ovcert -list
+----------------------------------------------------------+
| Keystore Content                                          |
+----------------------------------------------------------+
| Certificates:                                             |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)              |
+----------------------------------------------------------+
| Trusted Certificates:                                     |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993               |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                           |
+----------------------------------------------------------+
| Certificates:                                             |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)              |
+----------------------------------------------------------+
| Trusted Certificates:                                     |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)           |
+----------------------------------------------------------+
mgmtsv#: ovcert -remove dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
         'dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
mgmtsv# ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
         'CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
mgmtsv# ovcert -remove dcd0c94c-cb7d-7506-079a-9cc1b0282993 -ovrg server
       * Do you really want to remove the certificate with alias
         'dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
mgmtsv# ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 -ovrg server
       * Do you really want to remove the certificate with alias
         'CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
```

You should now see the following:

```
mgmtsv# ovcert -list
+----------------------------------------------------------+
| Keystore Content                                          |
+----------------------------------------------------------+
| Certificates:                                             |
+----------------------------------------------------------+
| Trusted Certificates:                                     |
+----------------------------------------------------------+

+----------------------------------------------------------+
| Keystore Content (OVRG: server)                           |
+----------------------------------------------------------+
| Certificates:                                             |
+----------------------------------------------------------+
| Trusted Certificates:                                     |
+---------------------------------------------------------+
```

You must now proceed with step Recreate the trusted certificate on the management server.

4.2 Recreate the trusted certificate on the management server

All steps in this subprocess should be taken on the management server.

Since all generated certificates must be signed by the certificate authority, as a first step we must recreate the trusted certificate, also referred to as the root certificate or the CA certificate.

To recreate the trusted certificate on the server:

```
mgmstv# ovcm -newcacert
INFO:    Generating a new CA key pair...
INFO:    Installing...
INFO:    Installation was successful.
```

You should now see the following:

```
mgmtsv# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
+---------------------------------------------------------+

+---------------------------------------------------------+
| Keystore Content (OVRG: server)                         |
+---------------------------------------------------------+
| Certificates:                                           |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)         |
+---------------------------------------------------------+
```

Now you can export the trusted certificate from the server side and import it on the node side:

```
mgmtsv# ovcert -exporttrusted -file /tmp/trustedcertif -ovrg server
INFO:    Trusted certificates have been successfully exported to file '/tmp/
         trustedcertif'.
mgmtsv# ovcert -importtrusted -file /tmp/trustedcertif
INFO:    Import operation was successful.
```

You should now see the following:

```
mgmtsv# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993             |
+---------------------------------------------------------+

+---------------------------------------------------------+
| Keystore Content (OVRG: server)                         |
+---------------------------------------------------------+
| Certificates:                                           |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)         |
+---------------------------------------------------------+
```

You must now proceed with step Recreate the server and node certificate on the management server.

4.3 Recreate the server and node certificate on the management server

All steps in this subprocess should be taken on the management server, but they depend on whether the OVO management server runs standalone or as a package or resource group on a cluster.

4.3.1 The OVO management server is standalone

Issue a new certificate for the management server and local agent, then import it on the management server and local agent:

```
mgmtsv# ovcm -issue -file /tmp/certif -name $(hostname) -pass mypwd -coreid
$(ovcoreid)
INFO:    Issued certificate was written to file '/tmp/certif'.
mgmtsv# ovcert -importcert -file /tmp/certif -pass mypwd -ovrg server
INFO:    Import operation was successful.
mgmtsv# ovcert -importcert -file /tmp/certif -pass mypwd
INFO:    Import operation was successful.
mgmtsv# rm /tmp/certif
```

You should now see the following:

```
mgmtsv# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993             |
+---------------------------------------------------------+

+---------------------------------------------------------+
| Keystore Content (OVRG: server)                         |
+---------------------------------------------------------+
| Certificates:                                           |
|     dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                    |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993 (*)          |
+----------------------------------------------------------+
```

You must now proceed with step Backup the certificates and private keys on the management server.

4.3.2 The OVO management server runs on a cluster as a package or resource group

NOTE: This case has not been tested.

Issue and import a new server certificate:

```
mgmtsv# ovcm -issue -file /tmp/certif -name $(hostname) -pass mypwd -coreid
$(ovcoreid –ovrg server)
INFO:    Issued certificate was written to file '/tmp/certif'.
mgmtsv# ovcert -importcert -file /tmp/certif -pass mypwd -ovrg server
INFO:    Import operation was successful.
mgmtsv# rm /tmp/certif
```

Issue and import a new node certificate:

```
mgmtsv# ovcm -issue -file /tmp/certif -name $(hostname) -pass mypwd -coreid
$(ovcoreid)
INFO:    Issued certificate was written to file '/tmp/certif'.
mgmtsv# ovcert -importcert -file /tmp/certif -pass mypwd
INFO:    Import operation was successful.
mgmtsv# rm /tmp/certif
```

You must now proceed with step Backup the certificates and private keys on the management server.

4.4 Backup the certificates and private keys on the management server

All steps in this subprocess should be taken on the management server.

To backup the certificates and private key on the management server:

```
mgmtsv# opcsvcertbackup -backup -passwd mypwd –file /tmp/svr_certificates.bkp
Info: Performing backup of OVO Server certificate data.
      Archive is /var/opt/OV/share/tmp/server_certificates.bkp.
Info: Determining core IDs ...
Info: Local system is not member of a HA cluster.
Info: Extracting server certificates ...
INFO:    Certificate has been successfully exported to file '/tmp/
         dcd0c94c-cb7d-7506-079a-9cc1b0282993.phys.cert'.
INFO:    Certificate has been successfully exported to file '/tmp/
         dcd0c94c-cb7d-7506-079a-9cc1b0282993.log.cert'.
Info: Extracting trusted certificates ...
INFO:    Trusted certificates have been successfully exported to file '/tmp/
         trusted.phys.cert'.
```

```
INFO:    Trusted certificates have been successfully exported to file '/tmp/
         trusted.log.cert'.
Info: Extracting CA certificate ...
INFO:    CA certificate was successfully exported to file '/tmp/CA.cert'.
Info: Archiving export files into /var/opt/OV/share/tmp/server_certificates.bkp
...
a /tmp/dcd0c94c-cb7d-7506-079a-9cc1b0282993.phys.cert 3K
a /tmp/dcd0c94c-cb7d-7506-079a-9cc1b0282993.log.cert 3K
a /tmp/trusted.phys.cert 2K
a /tmp/trusted.log.cert 2K
a /tmp/CA.cert 3K
a /tmp/opcsvcertbackup.20041125_175244.txt 1K
Info: All done. Exiting.
```

Make sure to move the file /tmp/svr_certificates.bkp to a secured location and to remember the password since you will need this if you must later restore the certificates. Note that this file contains the private keys of the management server and the certificate authority. **Unauthorized access to this information will compromise your entire OVO setup.**

You must now proceed with step Prepare the management server for certificate and policy deployment.

4.5 Prepare the management server for certificate and policy deployment

All steps in this subprocess should be taken on the management server.

Upon deployment of policies, the management server signs and caches the policies. When the management server's certificate changes the policies must be signed again, but the cache is not automatically updated. Therefore, the policy cache must be cleared manually.

To clear the policy cache:

```
mgmtsv# cd /etc/opt/OV/share/conf/OpC/mgmt_sv/templates
mgmtsv# rm */*/*
```

Next, clear any pending distributions:

```
mgmtsv$ rm /var/opt/OV/share/tmp/OpC/distrib/*
```

To facilitate granting of certificates in the next subprocedures, first ensure that all pending certificate requests have been deleted from the management server. This requires to start the control daemon, the communication broker and the certificate server:

```
mgmtsv# ovc –start CORE SERVER
```

You should now see something like:

```
mgmtsv# ovc
```

```
ovcd        OV Control                          CORE         (3610)   Running
ovbbccb     OV Communication Broker             CORE         (3611)   Running
opcle       OVO Logfile Encapsulator            AGENT,EA              Stopped
opcecaas    ECS Annotate Server                 AGENT,EA              Stopped
opcacta     OVO Action Agent                    AGENT,EA              Stopped
ovconfd     OV Config and Deploy                COREXT                Stopped
ovcs        OV Certificate Server               SERVER       (3616)   Running
coda        OV Performance Core                 AGENT,CODA            Stopped
opcmsga     OVO Message Agent                   AGENT,EA              Stopped
opcmona     OVO Monitor Agent                   AGENT,EA              Stopped
opcmsgi     OVO Message Interceptor             AGENT,EA              Stopped
opctrapi    OVO SNMP Trap Interceptor           AGENT,EA              Stopped
opceca      OVO Event Correlation               AGENT,EA              Stopped
```

Now delete any pending certificate request:

```
mgmtsv# for id in $(ovcm –listpending)
> do
> ovcm –remove $id
> done
```

You must now proceed with step Redeploy policies to the management server.

4.6 Redeploy policies to the management server

The policies currently installed on the server were signed using the old trusted certificate. Since we have now installed the new trusted certificate, we must redeploy all server policies.

First, start the Config and Deploy daemon and the OVO management server processes.

```
mgmtsv# ovc –start COREXT
mgmtsv# ovstart ovoacomm opc
```

You should see something like:

```
mgmtsv# ovc
ovcd        OV Control                          CORE         (3610)   Running
ovbbccb     OV Communication Broker             CORE         (3611)   Running
opcle       OVO Logfile Encapsulator            AGENT,EA              Stopped
opcecaas    ECS Annotate Server                 AGENT,EA              Stopped
opcacta     OVO Action Agent                    AGENT,EA              Stopped
ovconfd     OV Config and Deploy                COREXT       (5147)   Running
ovcs        OV Certificate Server               SERVER       (3616)   Running
coda        OV Performance Core                 AGENT,CODA            Stopped
opcmsga     OVO Message Agent                   AGENT,EA              Stopped
opcmona     OVO Monitor Agent                   AGENT,EA              Stopped
opcmsgi     OVO Message Interceptor             AGENT,EA              Stopped
opctrapi    OVO SNMP Trap Interceptor           AGENT,EA              Stopped
opceca      OVO Event Correlation               AGENT,EA              Stopped

mgmtsv# opcsv
OVO Management Server status:
-----------------------------
Control Manager           opcctlm     (5157) is running
Action Manager            opcactm     (5168) is running
Message Manager           opcmsgm     (5169) is running
TT & Notify Mgr           opcttnsm    (5170) is running
```

```
Forward Manager           opcforwm    (5171) is running
Service Engine            opcsvcm     (5176) is running
Cert. Srv Adapter         opccsad     (5174) is running
BBC config adapter        opcbbcdist  (5175) is running
Display Manager           opcdispm    (5172) is running
Distrib. Manager          opcdistm    (5173) is running

Open Agent Management status:
-----------------------------
Request Sender            ovoareqsdr  (5153) is running
Request Handler           ovoareqhdlr (5156) is running
Message Receiver (HTTPS)  opcmsgrb    (5158) is running
Message Receiver (DCE)    opcmsgrd    (5159) is running

OV Control Core components status:
----------------------------------
OV Control                ovcd        (5145) is running
OV Communication Broker   ovbbccb     (5146) is running
OV Certificate Server     ovcs        (5149) is running
```

Log into the Motif GUI as opc_adm and redeploy the server policies. Since there is no force option to redeploy server policies, you must first deassign and redeploy server policies, then reassign and redeploy the server policies:

In the Node Bank:

- Actions->Server->Assign Templates…
- Memorize current template assignements
- Remove all template assignments
- Actions->Server->Install / Update Server Templates
- Actions->Server->Assign Templates..
- Reassign templates
- Actions->Server->Install / Update Server Templates

You must now proceed with step Redeploy policies to the agent on the management server.

4.7 Redeploy policies to the agent on the management server

All steps in this subprocess should be taken on the management server.

The policies currently installed on all agents were signed using the old trusted certificate. Since we have now installed the new trusted certificate on the management server's agent, we must redeploy all policies.

To redeploy all policies to the management server's agent:

```
mgmtsv# opcragt -distrib -force $(hostname)
Node ovec1.bel.hp.com:
Create distribution data and inform agent...Done.
```

Since all policies need now to be read from the database and signed before they are stored in the cache, the above command may take longer than usual.

The deployment should succeed eventually. To confirm, use the following commands:

```
mgmtsv# ll /var/opt/OV/share/tmp/OpC/distrib
```

The above directory should be empty after a few minutes.

Then start the OVO agent and check that all processes are running correctly. You should see something like:

```
mgmtsv# ovc -start AGENT
mgmtsv# ovc
ovcd        OV Control                          CORE          (5145)   Running
ovbbccb     OV Communication Broker             CORE          (5146)   Running
opcle       OVO Logfile Encapsulator            AGENT,EA      (5324)   Running
opcacta     OVO Action Agent                    AGENT,EA      (5326)   Running
ovconfd     OV Config and Deploy                COREXT        (5147)   Running
ovcs        OV Certificate Server               SERVER        (5149)   Running
coda        OV Performance Core                 AGENT,CODA    (5309)   Running
opcmsga     OVO Message Agent                   AGENT,EA      (5325)   Running
opcmona     OVO Monitor Agent                   AGENT,EA      (5327)   Running
opcmsgi     OVO Message Interceptor             AGENT,EA      (5328)   Running
opctrapi    OVO SNMP Trap Interceptor           AGENT,EA      (5329)   Running
```

There should be no aborted processes and the following command should list the deployed policies:

```
mgmtsv# ovpolicy –list
```

This would also be the right time to check that the agent running on the management server is fully operational. You can check further deployments to this agent, sending test messages to the browser, running actions on this agent, etc… Take care that you may now see old buffered error messages coming into the browser that may no longer be relevant.

If the agent is not fully operational, something may have gone wrong in the previous steps. The problem should be understood before continuing with the next steps to avoid that the next steps need to be repeated multiple times on all agents.

You must now proceed with step Recreate the certificates and redeploy policies on all the agents.

4.8 Recreate the certificates and redeploy policies on all the agents

For each OVO agent, except the agent running on the management server, remove the certificates and install new certificates.

Login on the agent and stop the OVO agent and L-core processes:

```
agent# ovc –kill
```

It is quite common that some processes will not stop or that “ovc” will report an error. This is due to the fact that some processes communicate locally through HTTPS and you are currently resolving a problem with certificates, which may adversely affect HTTPS communication. You will have to kill these processes manually, for instance:

```
agent# ps –ef | grep ov
    root 17952 17951  0 10:49:20 ?         0:01 /opt/OV/bin/ovbbccb -nodaemon
    root 17951     1  0 10:49:20 ?         1:48 /opt/OV/bin/ovcd
agent# kill 17952
agent# kill 17951
agent# ps –ef | grep ov
agent# ps –ef | grep opc
agent# ps –ef | grep coda
```

After killing a process, verify that it was indeed stopped. If necessary, use “`kill -9`”.

Then remove the certificates:

```
agent# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
|     169f68ea-fae5-7506-0513-9ed4449eca3d (*)            |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
|     CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993             |
+---------------------------------------------------------+
agent# ovcert -remove 169f68ea-fae5-7506-0513-9ed4449eca3d
       * 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.
agent# ovcert -remove CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993
       * Do you really want to remove the certificate with alias
         'CA_dcd0c94c-cb7d-7506-079a-9cc1b0282993' (yes(y)/no(n))? y
INFO:    Certificate has been successfully removed.
```

You should now see the following:

```
agent# ovcert -list
+---------------------------------------------------------+
| Keystore Content                                        |
+---------------------------------------------------------+
| Certificates:                                           |
+---------------------------------------------------------+
| Trusted Certificates:                                   |
+---------------------------------------------------------+
```

Start the control daemon and communication broker:

```
agent# ovc –start CORE
```

You should see something similar to:

```
agent# ovc
```

```
ovcd        OV Control                          CORE         (17834)  Running
ovbbccb     OV Communication Broker             CORE         (17835)  Running
ovconfd     OV Config and Deploy                COREXT                Stopped
coda        OV Performance Core                 AGENT,CODA            Stopped
opcmsga     OVO Message Agent                   AGENT,EA              Stopped
opcacta     OVO Action Agent                    AGENT,EA              Stopped
opcmsgi     OVO Message Interceptor             AGENT,EA              Stopped
```

The agent should now automatically have sent a certificate request to the OV management server. To verify this, first check the coreid on the agent:

```
agent# ovcoreid
169f68ea-fae5-7506-0513-9ed4449eca3d
```

On the management server, verify that there is a pending certificate request for the agent:

```
mgmtsv# ovcm -listpending -l

RequestID:    0a878f5c-8b52-7508-0776-f107499f74c2
Context:
CN:           169f68ea-fae5-7506-0513-9ed4449eca3d
Nodename:     mcsc-sy1.bel.hp.com
IPAddress:    16.56.172.161
Platform:     HP-UX 11.11, CPU: PARisc
InstallType:  Manual
TimeReceived: 11/25/04 03:51:26 PM MET
```

Check if there is a pending certificate request where the CN field corresponds to the agent's coreid and verify that the TimeReceived field corresponds to the time when the control daemon was restarted. If not, you may need to manually generate a certificate request on the agent:

```
agent# ovcert -certreq
INFO:    Certificate request has been successfully triggered.
```

Once you have identified the correct certificate request on the management server, use the RequestID to grant the certificate:

```
mgmtsv# ovcm -grant 0a878f5c-8b52-7508-0776-f107499f74c2
```

On the agent, you should now see the certificates:

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

Start the Config and Deploy daemon on the agent:

```
agent# ovc –start COREXT
```

You should see something like:

```
agent# ovc
ovcd        OV Control                          CORE         (17834)  Running
ovbbccb     OV Communication Broker             CORE         (17835)  Running
ovconfd     OV Config and Deploy                COREXT       (18009)  Running
coda        OV Performance Core                 AGENT,CODA            Stopped
opcmsga     OVO Message Agent                   AGENT,EA              Stopped
opcacta     OVO Action Agent                    AGENT,EA              Stopped
opcmsgi     OVO Message Interceptor             AGENT,EA              Stopped
opcle       OVO Logfile Encapsulator            AGENT,EA              Stopped
opcmona     OVO Monitor Agent                   AGENT,EA              Stopped
opctrapi    OVO SNMP Trap Interceptor           AGENT,EA              Stopped
opceca      OVO Event Correlation               AGENT,EA              Stopped
opcecaas    ECS Annotate Server                 AGENT,EA              Stopped
```

On the management server, redeploy policies to this node:

```
mgmtsv# opcragt –distrib –force <agent_FQDN>
```

To confirm that the deployment succeeds, verify that the distrib directory is empty after a few minutes:

```
mgmtsv# ll /var/opt/OV/share/tmp/OpC/distrib
```

Now on the agent, start the remaining OVO agent processes and check that all processes are running correctly. You should see something like:

```
agent# ovc –start AGENT
agent# ovc
ovcd        OV Control                          CORE         (17834)  Running
ovbbccb     OV Communication Broker             CORE         (17835)  Running
ovconfd     OV Config and Deploy                COREXT       (18009)  Running
coda        OV Performance Core                 AGENT,CODA   (18836)  Running
opcmsga     OVO Message Agent                   AGENT,EA     (18977)  Running
opcacta     OVO Action Agent                    AGENT,EA     (18978)  Running
opcmsgi     OVO Message Interceptor             AGENT,EA     (18979)  Running
opcle       OVO Logfile Encapsulator            AGENT,EA     (18852)  Running
opcmona     OVO Monitor Agent                   AGENT,EA     (18853)  Running
```

There should be no aborted processes and the following command should list the deployed policies:

```
agent# ovpolicy –list
```

You can check further deployments to this agent, sending test messages to the browser, running actions on this agent, etc… Take care that you may now see old buffered error messages coming into the browser that may no longer be relevant.

Repeat all steps in this subprocedure for all OVO agents.

5 Certificates best practices

With the exception of the private key of the certificate server, all other keys and certificates can be regenerated with simple procedures. Therefore, keeping a backup of this private key is recommended. To achieve this, use the following command which will actually backup all private keys and certificates on the management server:

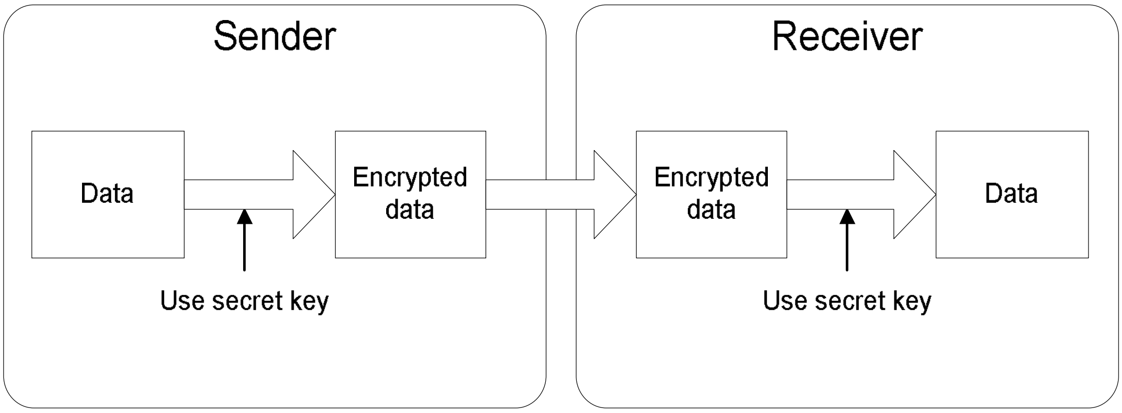
```
mgmtsv# opcsvcertbackup -backup -passwd mypwd –file /tmp/svr_certificates.bkp
Info: Performing backup of OVO Server certificate data.
      Archive is /var/opt/OV/share/tmp/server_certificates.bkp.
Info: Determining core IDs ...
Info: Local system is not member of a HA cluster.
Info: Extracting server certificates ...
INFO:    Certificate has been successfully exported to file '/tmp/
         dcd0c94c-cb7d-7506-079a-9cc1b0282993.phys.cert'.
INFO:    Certificate has been successfully exported to file '/tmp/
         dcd0c94c-cb7d-7506-079a-9cc1b0282993.log.cert'.
Info: Extracting trusted certificates ...
INFO:    Trusted certificates have been successfully exported to file '/tmp/
         trusted.phys.cert'.
INFO:    Trusted certificates have been successfully exported to file '/tmp/
         trusted.log.cert'.
Info: Extracting CA certificate ...
INFO:    CA certificate was successfully exported to file '/tmp/CA.cert'.
Info: Archiving export files into /var/opt/OV/share/tmp/server_certificates.bkp
...
a /tmp/dcd0c94c-cb7d-7506-079a-9cc1b0282993.phys.cert 3K
a /tmp/dcd0c94c-cb7d-7506-079a-9cc1b0282993.log.cert 3K
a /tmp/trusted.phys.cert 2K
a /tmp/trusted.log.cert 2K
a /tmp/CA.cert 3K
a /tmp/opcsvcertbackup.20041125_175244.txt 1K
Info: All done. Exiting.
```

Make sure to move the file /tmp/svr_certificates.bkp to a secured location and to remember the password since you will need this if you must later restore the certificates. Note that this file contains the private keys of the management server and the certificate authority. **Unauthorized access to this information will compromise your entire OVO setup.**

6 Background information

6.1 Secret keys and symmetric encryption

Symmetric encryption is the oldest and best-known technique. A secret key, which can be a number, a word, or just a string of random letters, is applied to the text of a message to change the content in a particular way. This might be as simple as shifting each letter by a number of places in the alphabet. As long as both sender and recipient know the secret key, they can encrypt and decrypt all messages that use this key.



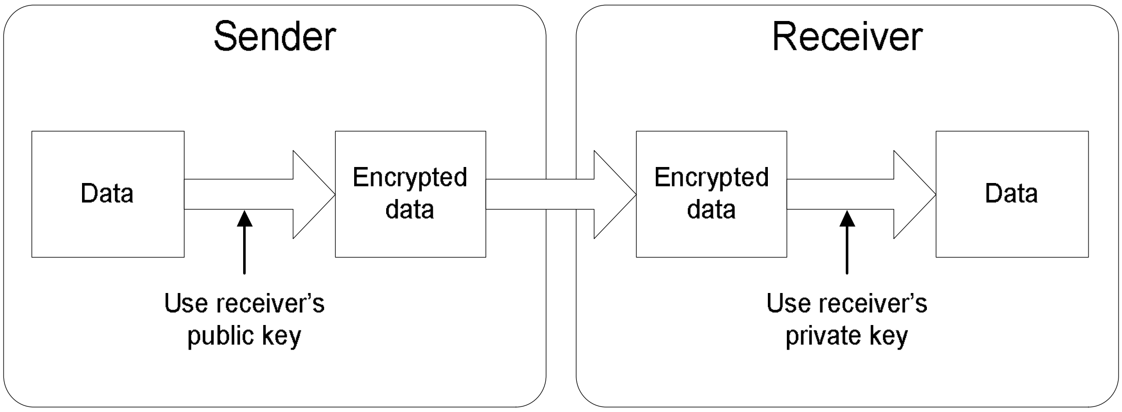
OVO uses symmetric encryption to request and grant new certificates.

6.2 Private/Public key pairs and symmetric encryption

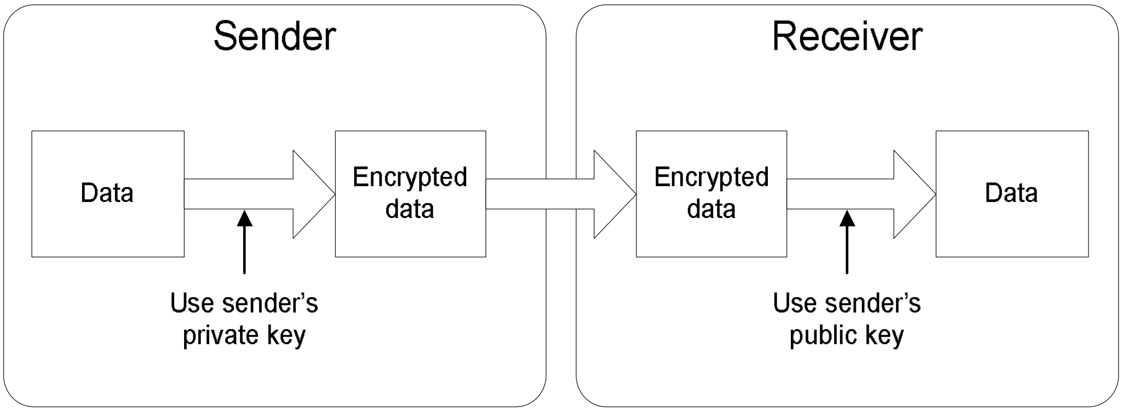
The problem with secret keys is exchanging them over the Internet or a large network while preventing them from falling into the wrong hands. Anyone who knows the secret key can decrypt the message. One answer is asymmetric encryption, in which there are two related keys, a key pair. A public key is made freely available to anyone who might want to send you a message. A second, private key is kept secret, so that only the owner knows it. The whole concept of SSL security relies on this private key remaining secret.

Any message (text, binary files, or documents) that are encrypted by using the public key can only be decrypted by applying the same algorithm, but by using the matching private key.

It is not practically possible to derive the private key from the public key. This means that you do not have to worry about passing public keys over the Internet (the keys are supposed to be public). By providing your public key on the internet, you enable others to encrypt data for you, that you alone will be able to decrypt, using your private key. Encrypting data with a receiver’s public key ensures that only the receiver can decrypt it.



Any message that is encrypted by using the private key can only be decrypted by using the matching public key. As explained later, encryption using one's private key provides a way to sign data, and thus guarantee its authenticity.



A problem with asymmetric encryption, however, is that it is slower than symmetric encryption. It requires far more processing power to both encrypt and decrypt the content of the message.

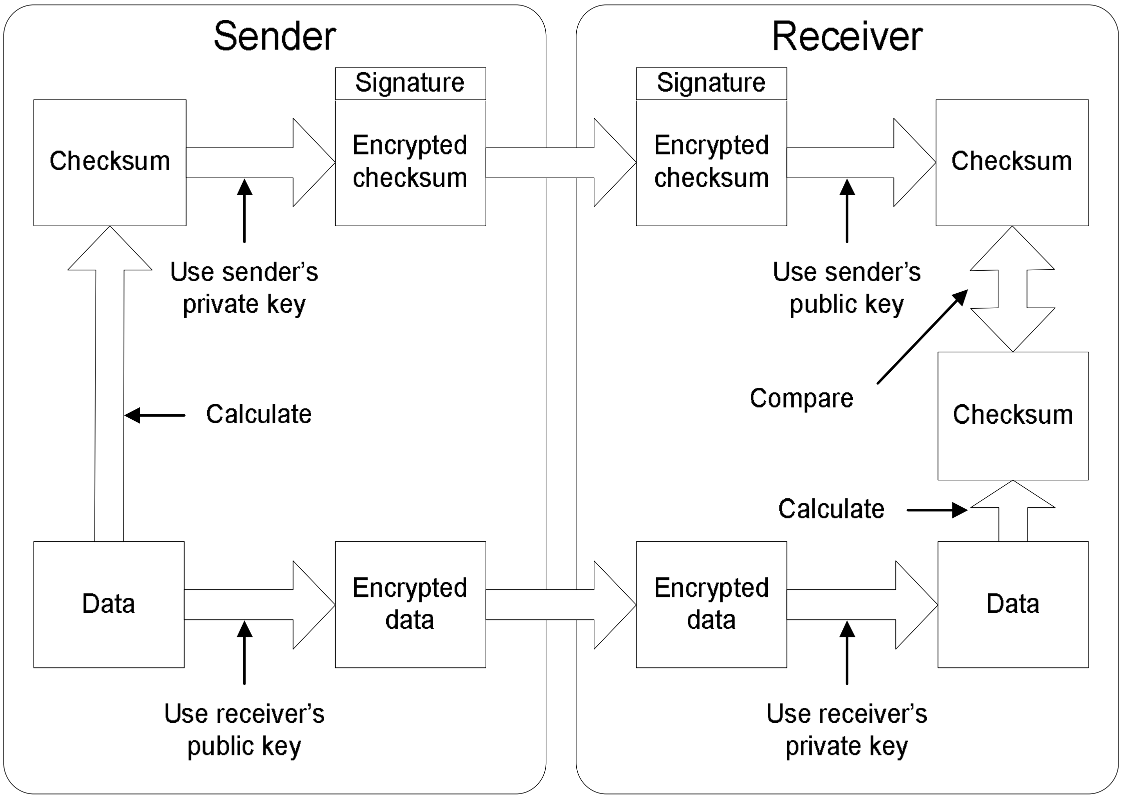
OVO uses asymmetric encryption to establish an SSL communication between an HTTPS node and the management server.

6.3 Signatures

Signatures are used to verify that transferred data has not been altered. Although the sender will encrypt the data with the receivers public key, which guarantees that only the receiver can decrypt it (using its private key), the data could still be altered intentionally or unintentionally.

To enable the receiver to validate the authenticity of the data, the sender will calculate a checksum on this data which it will encrypt with its own private key and send to the receiver along with the data. Using the private key to encrypt the checksum ensures that nobody but the sender could have generated this checksum. Hence, this encrypted checksum acts as a signature on the data.

After decrypting the data with its private key, the receiver can then calculate the same checksum. It then decrypts the senders checksum using the senders public key, and compares it against its own calculated checksum. If they are the same, the receiver is assured that the data has not been altered.



On top of signing certificates (see below), OVO signs action requests and policies to ensure that they cannot be altered.

6.4 Certificates

To use asymmetric encryption, there must be a way for people to discover other public keys. The typical technique is to use digital certificates, also known simply as certificates. A certificate is a package of information that identifies an entity (a user, a client, a server…), and contains information such as this entity's name and public key. In the case of OVO, the entities are nodes identified by their coreid.

Certificates are signed by a commonly trusted certificate authority. By doing this, we ensure that nobody can alter certificates or generate fake certificates. The certificate guarantees that the public key contained in it belongs to the entity that is mentioned in the certificate. Hence, when using this public key to encrypt a message, you are assured that only the entity mentioned in the certificate will be able to decrypt it.

On OVO, all certificates are generated by the certificate server running on the management server.

6.5 Trusted certificate and trusted certificate authority

The trusted certificate authority is a commonly trusted authority that is solely empowered to generate certificates. The trusted certificate authority possesses its own public and private key pair. It uses its private key to sign the certificates that it generates, thereby ensuring that no one else can generate certificates.

Whenever an entity is presented with a certificate from another party, it uses the trusted certificate authority's public key to verify the signature in the other party's certificate, thereby validating the authenticity of the certificate.

To make its public key available, the certificate authority generates its own certificate, often called trusted certificate, or root certificate, or CA certificate. Just like any other certificate, the trusted certificate contains an identification of the entity (in this case the trusted certificate authority) and its public key.

This explains why the trusted certificate must be installed on all the nodes of an OVO setup.

The certificate server running on the OVO management server acts as trusted certificate authority. Currently OVO does not support external trusted certificate authorities.

6.6 SSL handshake

When a server and client require a secure encrypted communication, they send a query over the network to the other party, which sends back a copy of its certificate. Each party can then extract the other party's public key from the certificate.

This enables each party to authenticate the other party. Indeed, to achieve this, all it takes is to encrypt a random piece of data with the other party's public key and verify that the other party can send it back unencrypted. If that is the case, the other party does indeed possess the private key corresponding to the public key contained in the certificate. Since the certificate cannot be faked, we know that the other party is the indeed the entity mentioned in the certificate.

At this point, the two parties can communicate with each other using asymmetric encryption to negotiate a secret key that will be used to symmetrically encrypt data that will be exchanged over this connection. This ensures faster, but still secure transfer of data once the SSL handshake has taken place. When this connection is closed, the whole SSL handshake process needs to be restarted, including the negotiation of a new secret key.

On OVO, an SSL connection automatically expires after 5 minutes or when there is no more data to transfer. The 5 minutes limit ensures that a new secret key will be negotiated on regular intervals.