THALES

NGINX: Integration Guide

THALES LUNA HSM AND DPOD LUNA CLOUD HSM

Document Information

Document Part Number	007-013662-001
Revision	D
Release Date	20 January 2021

Trademarks, Copyrights, and Third-Party Software

CONTENTS

Overview

This document guides you through the steps of integrating Luna HSM and Luna Cloud HSM with NGINX. NGINX is a free, open-source, high-performance HTTP server and reverse proxy, as well as an IMAP/POP3 proxy server. NGINX is known for its high performance, stability, rich feature set, simple configuration, and low resource consumption. You can integrate Luna HSMs and Luna Cloud HSM services with NGINX to generate 2048-bit RSA key pairs for SSL and protect the private keys within a FIPS 140-2 certified hardware security module. The benefits of integrating Luna HSMs and Luna Cloud HSMs with NGINX include:

- Secure generation, storage, and protection of SSL keys on FIPS 140-2 level 3 validated hardware.
- Complete life cycle management of the keys.
- Access to HSM audit trail*.
- Significant performance improvements by off-loading cryptographic operations from servers.

*Luna Cloud HSM services do not have access to the secure audit trail.

Certified Platforms

This integration is certified on the following platforms:

HSM Type	Operating System
Luna HSM	RHEL 7.x Ubuntu

NOTE: This integration is tested with Luna HSM clients in FIPS and HA Mode.

Luna HSM: Luna HSM appliances are purposefully designed to provide a balance of security, high performance, and usability that makes them an ideal choice for enterprise, financial, and government organizations. Luna HSMs physically and logically secure cryptographic keys and accelerate cryptographic processing. Luna HSM on premise offerings include the Luna Network HSM, Luna PCIe HSM, and Luna USB HSMs. Luna HSMs are also available for access as an offering from cloud service providers such as IBM cloud HSM and AWS cloud HSM classic.

HSM Type	Operating System
Luna Cloud HSM	Ubuntu

Luna Cloud HSM: Luna Cloud HSM provides on-demand HSM and Key Management services through a simple graphical user interface. With Luna Cloud HSM, security is simple, cost effective and easy to manage because there is no hardware to buy, deploy and maintain. As an Application Owner, you click and deploy services, generate usage reports and maintain just the services you need.

Prerequisites

Before you begin this integration, complete the following tasks:

Configure Luna HSM

If you are using Luna HSM:

1. Verify the HSM is set up, initialized, provisioned and ready for deployment. Refer to the Luna HSM documentation for more information.
2. Create a partition that will be later used by NGINX.
3. If using a Luna Network HSM, register a client for the system and assign the client to the partition to create an NTLS connection. Initialize the Crypto Officer and Crypto User roles for the registered partition.
4. Ensure that the partition is successfully registered and configured. The command to view the registered partitions is:

```
# /usr/safenet/lunaclient/bin/lunacm
lunacm (64-bit) v10.3.0-140. Copyright (c) 2020 SafeNet. All rights reserved.
Available HSMs:
Slot Id ->          0
Label ->            NGINX
Serial Number ->    1385675017779
Model ->            LunaSA 7.7.0
Firmware Version ->  7.7.0
Bootloader Version ->  1.1.2
Configuration ->    Luna User Partition With SO (PW) Key Export With Cloning Mode
Slot Description ->  Net Token Slot
FM HW Status ->     FM Ready
```

5. For PED-authenticated HSM, enable partition policies 22 and 23 to allow activation and auto-activation.

> **NOTE:** Refer to Luna HSM documentation for detailed steps to create NTLS connection and initialize the partitions and various user roles.

Configure Luna HSM HA (High-Availability)

Refer to Luna HSM documentation for HA steps and details regarding configuring and setting up two or more HSM appliances on Windows and UNIX systems. You must enable the HAOnly setting in HA for failover to work so that if primary stop functioning for some reason, all calls automatically routed to secondary till primary start functioning again.

Use Luna HSM in FIPS mode

Under FIPS 186-3/4, the RSA methods permitted for generating keys are 186-3 with primes and 186-3 with aux primes. This means that RSA PKCS and X9.31 key generation is no longer approved for operation in a FIPS-

compliant HSM. If you are using the Luna HSM in FIPS mode, you have to make the following change in configuration file:

```
Misc = {
RSAKeyGenMechRemap = 1;
}
```

The above setting redirects the older calling mechanism to a new approved mechanism when Luna HSM is in FIPS mode.

> **NOTE:** The above setting is not required for the Universal Client. This setting is applicable only for Luna Clients 6.x and 7.x.

Control user access to the HSM

By default, only the root user has access to the HSM. You can specify a set of non-root users that are permitted to access the HSM by adding them to the hsmusers group. The client software installation automatically creates the hsmusers group. The hsmusers group is retained when you uninstall the client software, allowing you to upgrade the software while retaining your hsmusers group configuration.

Add a user to hsmusers group

To allow non-root users or applications access to the HSM, assign the user to the hsmusers group. The users you assign to the hsmusers group must exist on the client workstation. To add a user to hsmusers group:

1. Ensure that you have sudo privileges on the client workstation.
2. Add a user to the hsmusers group.

   ```
   sudo gpasswd --add <username> hsmusers
   ```

   Where `<username>` is the name of the user you want to add to the hsmusers group.

Remove a user from hsmusers group

To remove a user from hsmusers group:

1. Ensure that you have sudo privileges on the client workstation.
2. Remove a user from the hsmusers group.

   ```
   sudo gpasswd -d <username> hsmusers
   ```

   Where `<username>` is the name of the user you want to remove from the hsmusers group. You must log in again to see the change.

> **NOTE:** The user you delete will continue to have access to the HSM until you reboot the client workstation.

Configure Luna Cloud HSM service

You can configure Luna Cloud HSM Service in the following ways:

> Standalone Cloud HSM service using minimum client package

> Standalone Cloud HSM service using full Luna client package

> Luna HSM and Luna Cloud HSM service in hybrid mode

NOTE: Luna Client v10.x or higher is required for configuring Luna HSM device and Luna Cloud HSM service in hybrid mode.

Standalone Cloud HSM service using minimum client package

To configure Luna Cloud HSM service using minimum client package:

1. Transfer the downloaded .zip file to your Client workstation using pscp, scp, or other secure means.
2. Extract the .zip file into a directory on your client workstation.
3. Extract or untar the appropriate client package for your operating system. Do not extract to a new subdirectory; place the files in the client install directory.

   ```
   [Windows]
   cvclient-min.zip
   [Linux]
   cvclient-min.tar
   # tar -xvf cvclient-min.tar
   ```

4. Run the setenv script to create a new configuration file containing information required by the Luna Cloud HSM service.

   ```
   [Windows]
   Right-click setenv.cmd and select Run as Administrator.
   [Linux]
   Source the setenv script.
   # source ./setenv
   ```

5. Run the LunaCM utility and verify the Cloud HSM service is listed.

Standalone Cloud HSM service using full Luna client package

To configure Luna Cloud HSM service using full Luna client package:

1. Transfer the downloaded .zip file to your Client workstation using pscp, scp, or other secure means.
2. Extract the .zip file into a directory on your client workstation.
3. Extract or untar the appropriate client package for your operating system. Do not extract to a new subdirectory; place the files in the client install directory.

   ```
   [Windows]
   cvclient-min.zip
   [Linux]
   cvclient-min.tar
   ```

```
# tar -xvf cvclient-min.tar
```

4. Run the setenv script to create a new configuration file containing information required by the Luna Cloud HSM service.

```
[Windows]
Right-click setenv.cmd and select Run as Administrator.
[Linux]
Source the setenv script.
# source ./setenv
```

5. Copy the server and partition certificates from the Cloud HSM service client directory to Luna client certificates directory:

Cloud HSM Certificates:

```
server-certificate.pem
partition-ca-certificate.pem
partition-certificate.pem
```

LunaClient Certificate Directory:

```
[Windows default location for Luna Client]
C:\Program Files\Safenet\Lunaclient\cert\
[Linux default location for Luna Client]
/usr/safenet/lunaclient/cert/
```

> **NOTE:** Skip this step for Luna Client v10.2 or higher.

6. Open the configuration file from the Cloud HSM service client directory and copy the XTC and REST section.

```
[Windows]
crystoki.ini
[Linux]
Chrystoki.conf
```

7. Edit the Luna Client configuration file and add the XTC and REST sections copied from Cloud HSM service client configuration file.
8. Change server and partition certificates path from step 5 in XTC and REST sections. Do not change any other entries provided in these sections.

[XTC]

```
. . .
PartitionCAPath=<LunaClient_cert_directory>\partition-ca-certificate.pem
PartitionCertPath00=<LunaClient_cert_directory>\partition-certificate.pem
. . .

[REST]
. . .
SSLClientSideVerifyFile=<LunaClient_cert_directory>\server-certificate.pem
```

```
. . .
```

NOTE: Skip this step for Luna Client v10.2 or higher.

9. Edit the following entry from the Misc section and update the correct path for the plugins directory:

```
Misc]
PluginModuleDir=<LunaClient_plugins_directory>

[Windows Default]

C:\Program Files\Safenet\Lunaclient\plugins\

[Linux Default]

/usr/safenet/lunaclient/plugins/
```

10. Save the configuration file. If you wish, you can now safely delete the extracted Cloud HSM service client directory.
11. Reset the ChrystokiConfigurationPath environment variable and point back to the location of the Luna Client configuration file.

    Windows

    In the Control Panel, search for "environment" and select Edit the system environment variables. Click Environment Variables. In both list boxes for the current user and system variables, edit ChrystokiConfigurationPath and point to the crystoki.ini file in the Luna client install directory.

    Linux

    Either open a new shell session, or export the environment variable for the current session pointing to the location of the Chrystoki.conf file:

    ```
    # export ChrystokiConfigurationPath=/etc/
    ```

12. Run the LunaCM utility and verify that the Cloud HSM service is listed. In hybrid mode, both Luna and Cloud HSM service will be listed.

NOTE: Follow the Luna Cloud HSM documentation for detailed steps for creating service, client, and initializing various user roles.

Luna HSM and Luna Cloud HSM service in hybrid mode

To configure Luna HSM and Luna Cloud HSM service in hybrid mode, follow the steps mentioned under the Standalone Cloud HSM service using full Luna client package section above.

NOTE: Luna Client v10.x or higher is required for configuring Luna HSM device and Luna Cloud HSM service in hybrid mode.

To use Luna Cloud HSM Service in FIPS mode

Cloud HSM service operates in both FIPS and non-FIPS mode. If your organization requires non-FIPS algorithms for your operations, enable the Allow non-FIPS approved algorithms check box when configuring your Cloud HSM service. The FIPS mode is enabled by default. Refer to the Mechanism List in the SDK Reference Guide for more information about available FIPS and non-FIPS algorithms.

Set up NGINX

NGINX server must be installed on the target machines to carry on with the integration process. For a detailed installation procedure, refer to the NGINX documentation: https://docs.nginx.com/nginx/.

> **NOTE:** If you are using HSM in FIPS mode, the NGINX must be compiled and installed with OpenSSL in FIPS mode. Currently FIPS support for OpenSSL is available only till OpenSSL v1.0.2. OpenSSL v1.1.1 does not have FIPS support available.

Integrating Luna HSM with NGINX

Integration of Luna HSM with NGINX involves two use cases:

> Integrate Luna HSM with NGINX by generating new SSL keys

> Integrate Luna HSM with NGINX by migrating existing SSL keys

Integrate Luna HSM with NGINX by generating new SSL keys

To integrate Luna HSM with NGINX by generating new SSL keys, you need to complete the following tasks:

> Configure OpenSSL to enable GemEngine

> Configure SSL for NGINX using OpenSSL

Configure OpenSSL to enable GemEngine.

You can configure the OpenSSL installed by default in the system or you can install another version and configure it. To configure GemEngine for OpenSSL:

1. Log on to the NGINX server as root user or as any other user with administrative privileges.
2. Download the OpenSSL toolkit from Thales support portal, extract it on your system, and go to the directory where GemEngine is extracted to locate the `gembuild` utility.

   Example:

   ```
   # cd /home/gemengine-1.2
   ```

   > **NOTE:** NGINX uses OpenSSL for SSL/TLS support. OpenSSL includes a component called ENGINE to store keys on hardware devices. Thales provides the OpenSSL toolkit having support of GemEngine that is used to communicate with the Luna HSM. OpenSSL toolkit can be download from the Thales Support Portal. It is recommended that you should familiarize yourself with OpenSSL. Refer to the appropriate documents for OpenSSL commands at the following location: http://www.openssl.org/docs/

3. Locate the OpenSSL engines directory using the gembuild command.

   ```
   # ./gembuild locate-engines
   ```

   The directory displays the engines directory for the OpenSSL that is in PATH.

4. Copy the **libgem.so** to the engines directory displayed in the previous command.

   Example:

   ```
   # cp builds/linux/rhel/64/1.0.2/libgem.so /usr/lib64/openssl/engines/
   ```

NOTE: You can also build and install Gem Engine, SAUTIL, and OpenSSL using the OpenSSL Toolkit downloaded from Thales Portal. Refer the README files provided with the Toolkit for detailed instructions.

5. Verify that GemEngine is present and supported by OpenSSL.

```
# openssl engine gem -v
(gem) Gem engine support
enginearg, openSession, closeSession, login, logout, engineinit,
CONF_PATH, ENGINE_INIT, ENGINE2_INIT, engine2init, DisableCheckFinalize,
SO_PATH, GET_HA_STATE, SET_FINALIZE_PENDING, SKIP_C_INITIALIZE,
IntermediateProcesses
```

6. Create a passfile containing partition CO pin and edit the /etc/Chrystoki.conf file to add support for GemEngine. Update the Chrystoki.conf as follows:

```
GemEngine = {
LibPath = /usr/safenet/lunaclient/lib/libCryptoki2.so;
LibPath64 = /usr/safenet/lunaclient/lib/libCryptoki2_64.so;
EnableDsaGenKeyPair = 1;
EnableRsaGenKeyPair = 1;
DisablePublicCrypto = 1;
EnableRsaSignVerify = 1;
EnableLoadPubKey = 1;
EnableLoadPrivKey = 1;
DisableCheckFinalize = 1;
DisableEcdsa = 1;
DisableDsa = 0;
DisableRand = 0;
EngineInit = "<myTokenLabel>":0:0:passfile=</path/to/my/passfile>;
EnableLoginInit = 1;
}
```

Where `myTokenLabel` is the partition label and `passfile` is the path to file containing the partition CO pin.

NOTE: There are other methods available to enable login via GemEngine, if you do not want to save the partition CO pin in a file. Refer the README files provided with the Toolkit for detail instructions using other methods.

7. After making all the above changes, verify that OpenSSL is configured successfully to use Luna HSM using GemEngine.

```
# openssl engine gem -t
(gem) Gem engine support
[ available ]
```

This completes the OpenSSL configuration for GemEngine support. The next section describes how to configure SSL for NGINX.

Configure SSL for NGINX using OpenSSL

NGNIX server utilizes OpenSSL to generate the SSL keys and certificates needed for SSL communication. You need to generate certificate and keys for SSL using OpenSSL that generates the key on HSM as GemEngine is already configured for this purpose. After that, update the NGINX configuration file to support SSL communication.

> Generate certificates

> Update NGINX to support SSL

Generate certificates

To configure SSL, you need to generate the certificate that can either be self-signed or signed by renowned CA. In both cases, the certificate private key will be secured in Luna HSM. Below are steps to generate CA signed certificate and self-signed certificate, respectively.

> **NOTE:** It is recommended to use the CA signed certificate in production environment. Self-Signed certificate is suitable for test environment only.

To generate the CA-signed SSL certificate:

1. Execute the command below to generate the keys on Luna HSM and save the certificate request and key reference.

   ```
   # openssl req -engine gem -new -newkey rsa:2048 -nodes -sha256 -keyout server.key -out server.csr
   ```

   Two keys (Public and Private) will be generated onto HSM and the Private Key reference generated on HSM will be saved in **server.key** file. This is required later in the procedure and Certificate Request (CSR) will be saved in **server.csr** file.

2. Run the **cmu** list to verify the generated key pair on Luna HSM.

   ```
   # /usr/safenet/lunaclient/bin/cmu list

   Certificate Management Utility (64-bit) v10.3.0-140. Copyright (c) 2020 SafeNet. All rights reserved.

   Please enter password for token in slot 0 : ********

   handle=47       label=rsa-private-826041657831821103 68d32e9873dbcff8c7e3eb

   handle=35       label=rsa-public-826041657831821103 68d32e9873dbcff8c7e3eb
   ```

3. Submit the CSR file to a CA such as VeriSign or Entrust. The CA authenticates the request and returns a signed certificate or a certificate chain. Save the CA-signed certificate in the system directory.

Generate self-signed SSL certificate

1. Execute the command below to generate the keys on Luna HSM and save the key reference.

   ```
   # openssl genrsa -engine gem -out server.key 2048
   ```

   The server.key is the key reference to Private Key Generated on HSM, this is required later in the procedure.

2. To generate a self-signed certificate, which is good for test purpose, issue the following command.

```
# openssl req -new -engine gem -x509 -key server.key -sha256 -out server.pem
```

Where the server.pem is the self-signed certificate in PEM format.

Update NGINX to support SSL

For updating NGINX to support SSL:

1. Open the NGINX configuration file <NGINX installation directory>/nginx.conf file and update the nginx.conf file as follows to enable the SSL support at the end of http section.

```
server {
listen       443 ssl;
server_name  <Server Hostname or IP Address>;
ssl_certificate      <path to CA signed or self-signed certificate>;
ssl_certificate_key  <path to the private key file>;
ssl_session_cache    shared:SSL:1m;
ssl_session_timeout  5m;
ssl_protocols   TLSv1.2 TLSv1.3;
ssl_ciphers  HIGH:!aNULL:!MD5;
ssl_prefer_server_ciphers  on;
location / {
root   html;
index  index.html index.htm;
}
}
```

2. Replace the server_name, ssl_certificate and ssl_certificate_key value with actual values in your environment.

Where:

- ssl_certificate is the self-signed or CA signed certificate
- ssl_certificate_key is the location of reference to private key generated on HSM in PEM format.

> **NOTE:** TLSv1.3 support is available in OpenSSL v1.1.1 only. Older OpenSSL versions do not supports TLSv1.3.

3. Open the NGINX configuration file <NGINX installation directory>/nginx.conf file and update the nginx.conf file as follows to enable the Gem Engine support before the beginning of http section.

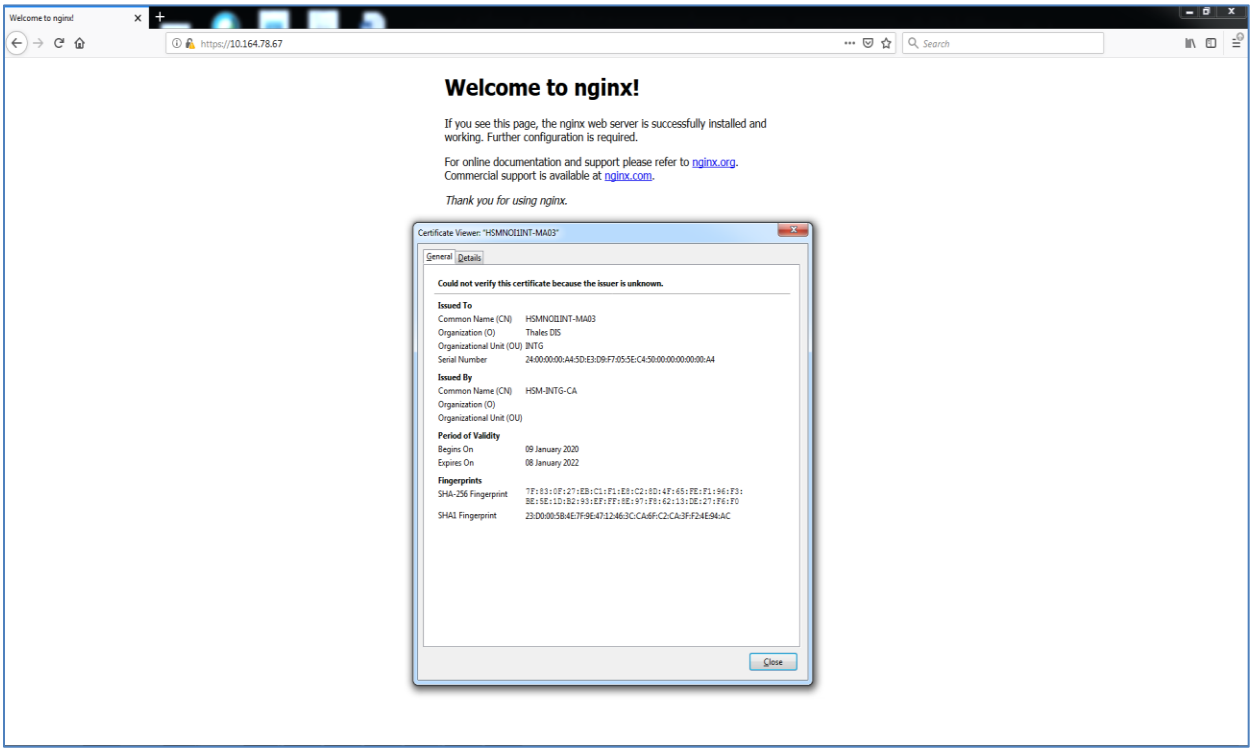
```
ssl_engine gem;
```

4. Run the following command under the <NGINX installation directory> to verify there is no error due to configuration changes in nginx.conf file.

```
# ./nginx -t
nginx: the configuration file /usr/local/nginx/nginx.conf syntax is ok
```

```
nginx: configuration file /usr/local/nginx/nginx.conf test is successful
```

5. Restart the NGINX server.
6. Open any web browser and access the NGINX server. Verify the certificate.



This completes the NGINX integration with Luna HSM. The NGINX server's SSL private key is now secured on HSM partition. The SSL page will be displayed only if HSM partition is accessible and available to NGINX Server.

Integrate Luna HSM with NGINX by migrating existing SSL keys

It is assumed that the NGINX server is already configured and running on SSL where SSL certificate and keys are generated by OpenSSL and saved somewhere in the directory.

To migrate existing SSL keys:

1. Configure OpenSSL to use GemEngine by executing the steps mentioned in the Configure OpenSSL to Enable GemEngine.section.
2. Locate the directory where the SSL Private Key and Certificate are located.
3. Extract the Certificate Public key using the command below.

   ```
   # openssl rsa -in server.key -pubout -out pubkey.pem
   ```

4. Extract the Private Key in PKCS#8 format using the below command.

   ```
   # openssl pkcs8 -in server.key -topk8 -nocrypt -out privatekey.pem
   ```

5. Using the CMU utility provided with Luna Client, import the public key and private key to the HSM.

   For Public Key:

   ```
   # /usr/safenet/lunaclient/bin/cmu import -inputFile pubkey.pem -label
   nginx_public_key -pubkey=rsa
   ```

   For Private Key:

```
# /usr/safenet/lunaclient/bin/cmu importkey -PKCS8 -in privatekey.pem -keyalg
RSA
```

6. Verify that the keys are generated on Luna HSM partition and note the private key handle that will be used later.

```
# /usr/safenet/lunaclient/bin/cmu list

Certificate Management Utility (64-bit) v10.1.0-32. Copyright (c) 2019 SafeNet.
All rights reserved.

Please enter password for token in slot 0 : ********

handle=37       label=CMU Unwrapped RSA Private Key

handle=36       label=nginx_public_key
```

7. In case you have multiple keys, you can recognize the private key by providing it a label. Use the following command to provide a label to the private key:

```
# /usr/safenet/lunaclient/bin/cmu setattribute -handle=37 -
label=nginx_private_key
```

8. Verify that the private key label matches the label of public key.

```
# /usr/safenet/lunaclient/bin/cmu list

Certificate Management Utility (64-bit) v10.3.0-140. Copyright (c) 2020
SafeNet. All rights reserved.

Please enter password for token in slot 0 : ********

handle=37       label=nginx_private_key

handle=36       label=nginx_public_key
```

9. Copy the SAUTIL utility provided with OpenSSL toolkit to create the Private Key reference in software.

```
# cp /home/gemengine-1.2/builds/linux/rhel/64/1.0.2/sautil /usr/bin/
```

10. Run the **sautil** utility to create Private Key Reference to actual private key imported in Luna HSM.

```
# sautil -v -s 0 -i 0:0 -a 0:RSA -f HSMKey_ref.pem -o -q –c
```

Provide the HSM partition CO password and key handle when prompted. After successful execution of the `sautil` command, HSMKey_ref.pem will be generated and you need to specify that in the SSL setting in nginx.conf file.

11. Remove the Private Key generated by OpenSSL that was used before importing the key in to Luna HSM along with the PKCS#8 format key generated in step 4.

```
# rm -rf /usr/localnginx/server.key /usr/local/nginx/privatekey.pem
```

12. Edit the nginx.conf file and update the ssl_certificate_key location with the HSM Key Reference generated in step 10.

```
server {

listen       443 ssl;

server_name  <Server Hostname or IP Address>;

ssl_certificate      /usr/local/nginx/server.pem;

ssl_certificate_key  /usr/local/nginx/HSMKey_ref.pem;
```

```
ssl_session_cache    shared:SSL:1m;
ssl_session_timeout  5m;
ssl_protocols    TLSv1.2 TLSv1.3;
ssl_ciphers  HIGH:!aNULL:!MD5;
ssl_prefer_server_ciphers  on;
location / {
root   html;
index  index.html index.htm;
}
}
```

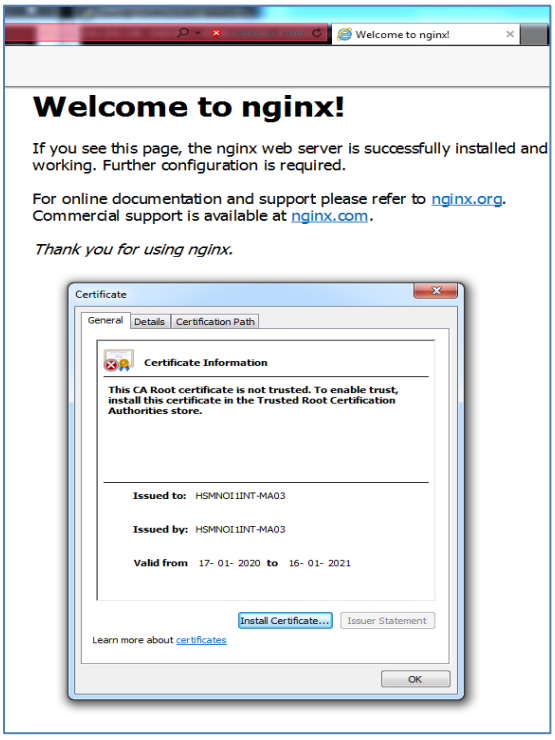
13. Replace the server_name, ssl_certificate, and ssl_certificate_key value with actual values in your environment. Here, ssl_certificate is the self-signed or CA signed certificate and ssl_certificate_key is the location of reference to Private Key Generated on HSM in PEM format.

> **NOTE:** TLSv1.3 support is available in OpenSSL v1.1.1 only. Older OpenSSL versions do not support TLSv1.3.

14. Open the NGINX configuration file <NGINX installation directory>/nginx.conf and update it as follows to enable the Gem Engine support before the beginning of http section.

```
ssl_engine gem;
```

15. Restart the NGINX server.
16. Open any browser and access the NGINX Server. Verify that server is accessible and is using the private key migrated to HSM.



Contacting Customer Support

If you encounter a problem during this integration, refer to the documentation. If you cannot resolve the issue, contact your supplier or Thales Customer Support. Thales Customer Support operates 24 hours a day, 7 days a week. Your level of access to this service is governed by the support plan arrangements made between Thales and your organization. Please consult this support plan for further information about your entitlements, including the hours when telephone support is available to you.

Customer Support Portal

The Customer Support Portal, at https://supportportal.thalesgroup.com, is a database where you can find solutions for most common problems. The Customer Support Portal is a comprehensive, fully searchable repository of support resources, including software and firmware downloads, release notes listing known problems and workarounds, a knowledge base, FAQs, product documentation, technical notes, and more. You can also use the portal to create and manage support cases.

> **NOTE:** You require an account to access the Customer Support Portal. To create a new account, go to the portal and click on the **REGISTER** link.

Telephone Support

If you have an urgent problem, or cannot access the Customer Support Portal, you can contact Thales Customer Support by telephone at +1 410-931-7520. Additional local telephone support numbers are listed on the support portal.

Email Support

You can also contact technical support by email at technical.support.DIS@thalesgroup.com.