THALES

# HAProxy: Integration Guide

## THALES LUNA HSM

## Document Information

| | |
|---|---|
| **Document Part Number** | 007-000297-001 |
| **Revision** | E |
| **Release Date** | 1 March 2021 |

## Trademarks, Copyrights, and Third-Party Software

# CONTENTS

# Overview

HAProxy functions as a load balancer and integrates with the Thales Luna HSM to provide secure SSL termination. This allows for secure transport of the load balancing traffic to the backend servers ensuring the integrity of the system. This integration describes how to secure the keys used for SSL Termination in HAProxy on a Thales HSM using the OpenSSL toolkit with GemEngine support. Using Luna HSMs to secure the HAProxy SSL Keys provides the following benefits:

- Secure generation, storage, and protection of the Identity signing private key on FIPS 140-2 level 3 validated hardware.
- Full life cycle management of the keys.
- HSM audit trail.
- Significant performance improvements by off-loading cryptographic operations from application servers.

# Certified Platforms

This integration is tested/verified with Luna HSM on the following operating systems:

| Platforms Tested | HAProxy Version | GemEngine Version | OpenSSL Version |
|---|---|---|---|
| RHEL 8 | 2.2 | 1.3 | 1.0.2 FIPS |
| Ubuntu 18.04 | 1.8 | 1.3 | 1.1.1 Non FIPS<br>1.0.2 FIPS |
| RHEL 7 | 1.8 | 1.2 | 1.0.2 FIPS |
| Centos 7 | 1.8 | 1.2 | 1.0.2 FIPS |

**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. The 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.

# Prerequisites

Before you proceed with the integration, complete the following tasks:

## Configure Luna HSM

To configure Luna HSM:

1. Ensure that the HSM is set up, initialized, provisioned and ready for deployment. Refer to the HSM product documentation for help.
2. Create a partition that will be later used by HAProxy.
3. Create and exchange certificate between the Luna Network HSM and Client system. Register client and assign partition to create an NTLS connection. Initialize Crypto Officer and Crypto User roles for the registered partition.
4. Ensure that the partition is successfully registered and configured. The command to see the registered partitions is:

```
# /usr/safenet/lunaclient/bin/lunacm

lunacm (64-bit) v7.3.0-165. Copyright (c) 2018 SafeNet. All rights reserved.

  Available HSMs:

  Slot Id ->              0
  Label ->                HAProxy
  Serial Number ->        1280780175865
  Model ->                LunaSA 7.3.0
  Firmware Version ->     7.3.0
  Configuration ->        Luna User Partition With SO (PW) Key Export With
   Cloning Mode
  Slot Description ->     Net Token Slot

  Current Slot Id: 0
```

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

> **NOTE:** Follow the Luna Network Luna HSM documentation for detailed steps about creating NTLS connection, initializing the partitions, and assigning various user roles.

### To 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. RSA PKCS and X9.31 key generation are no longer approved for operation in a FIPS-compliant HSM. If you are using Luna HSM in FIPS mode, make the following change in configuration file:

```
[Misc]

RSAKeyGenMechRemap=1
```

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

> **NOTE:** For Luna Client 10.x onwards, this setting is not needed. It is applicable for Luna Client 7.x only.

**To configure Luna HSM HA (High-Availability)**

Please refer to the 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 starts functioning again.

> **NOTE:** This integration is tested in both HA and FIPS mode.

## Download GemEngine Toolkit

Acquire the OpenSSL toolkit with GemEngine support from Thales Customer Support.

> **NOTE:** You require a Customer Support Portal account to download the GemEngine. The doc IDs for downloading the GemEngine v1.2 from support portal is KB0016309. The Doc ID for downloading the GemEngine v1.3 from support portal is KB0017806.

## Set up HAProxy

For the purpose of this integration, it's recommended that you use an Apache webserver to serve a static webpage. In this integration guide, the following three machines have been used:

- One primary machine that operates as the Load Balancer and has HAProxy installed. For detailed information about installing HAProxy, Refer the *HAProxy Documentation*.

  > **NOTE:** If you want to install HAProxy with Custom OpenSSL, refer to the Appendix section of this document.
  >
  > **NOTE:** If you are using Luna HSM in FIPS mode, then HAProxy should also be built with OpenSSL with FIPS module supported.

- Two secondary machines that operate as the backend server with FQDN and IP enabled running Apache and serving static content.

  > **NOTE:** You can modify this configuration according to your requirements.

# Integrating Luna HSM with HAProxy

To integrate Luna HSM with HAProxy, you need to perform the following tasks:

- Configure OpenSSL to use GemEngine
- Configure HAProxy for SSL Termination

## Configure OpenSSL to use GemEngine

To configure OpenSSL to use GemEngine:

1. Copy the GemEngine toolkit to any directory. Locate the OpenSSL engines directory using gembuild. Gembuild is available in the directory where you extracted the GemEngine toolkit.

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

This gives the location of the directory of the `libgem.so`. By default, the OpenSSL engines directory is located at `/usr/lib64/openssl/engines`.

2. Copy the libgem.so to the engines directory for your OpenSSL version. For example:

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

3. Verify that the GemEngine is loading. Execute:

```
# 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
```

> **NOTE:** If you are installing OpenSSL from source, follow the *README-GEMBUILD* text file in the **/docs** directory. The document provides further details on compiling and building with GemEngine.

4. Create a passfile and store the partition password in it.

```
# echo <partition_password> > /tmp/passfile
```

5. Open the /etc/Chrystoki.conf file and add the following GemEngine section.

```
GemEngine = {
LibPath = /usr/safenet/lunaclient/lib/libCryptoki2_64.so;
LibPath64 = /usr/safenet/lunaclient/lib/libCryptoki2_64.so;
EnableDsaGenKeyPair = 1;
EnableRsaGenKeyPair = 1;
DisablePublicCrypto = 1;
EnableRsaSignVerify = 1;
EnableLoadPubKey = 1;
EnableLoadPrivKey = 1;
DisableCheckFinalize = 0;
IntermediateProcesses = 0;
DisableEcdsa = 1;
DisableDsa = 0;
DisableRand = 0;
EngineInit = <slot_id>:0:0:passfile=<path_to_passfile>;
EnableLoginInit = 1;
}
```

## Configure HAProxy for SSL Termination

To configure HAProxy for SSL termination:

1. Create a private key.

```
# openssl genrsa -engine gem -out key.pem
```

2. Create a self-signed certificate using the above private key.

```
# openssl req -engine gem -new -x509 -days 365 -key key.pem -out cert.cer
```

> **NOTE:** This integration uses self-signed certificates in a test environment only. For a production environment, we recommend using a more secure method such as a certificate authority to issue the certificate.

3. Copy the private key reference and certificate to a new file at `/etc/pki/tls/certs/haproxy.pem` using the following command:

```
# cat key.pem cert.cer > /etc/pki/tls/certs/haproxy.pem
```

4. Open the `/etc/opt/rh/rh-haproxy18/haproxy/haproxy.cfg` file in a text editor and add the following code:

```
global
   log         127.0.0.1 local2
   pidfile     /var/run/rh-haproxy18-haproxy.pid
   maxconn     4000
   pidfile     /var/run/haproxy.pid
   user        haproxy
   group       hsmusers
   daemon
   ssl-engine gem
   tune.ssl.default-dh-param 2048


defaults
   mode                    http
   log                     global
   option                  httplog
   option                  dontlognull
   retries                 3
   timeout http-request    10s
   timeout queue           1m
   timeout connect         10s
   timeout client          1m
   timeout server          1m
```

```
    timeout http-keep-alive 10s
    timeout check           10s

frontend  https-in
    bind *:80
    use_backend static
    option forwardfor
    bind *:443 ssl crt /etc/pki/tls/certs/haproxy.pem

backend static
    balance     roundrobin
    server server1.example.com 10.164.76.117:80 check
    server server2.example.com 10.164.78.118:80 check
```

> **NOTE:** The actual path for **haproxy.cfg** may vary depending on the HAProxy Installation. Also the fields in **haproxy.cfg** may vary according to your requirement. Follow the *HAProxy Documentation* for detailed information of each field. Ensure you add **ssl-engine gem** to the **global** section so that OpenSSL can use GemEngine.

5. Check the SELinux setting. Set it to permissive mode by running the following command:

   ```
   # setenforce 0
   ```

   This will temporarily set SELinux to permissive mode. To make a permanent change, go to `/etc/selinux/config` and edit the following line:

   ```
   SELINUX= permissive
   ```

   Reboot the system for the changes to occur.

   ```
   # reboot
   ```

   > **NOTE:** If you are using Ubuntu then skip this step.

6. Start the HAProxy service.

   ```
   # systemctl start haproxy.service
   ```

   Ensure you have added the HAProxy service to the firewall daemon to allow traffic through the firewall.

7. Check the server status and verify that it starts without any error.

   ```
   # systemctl status haproxy.service  -l
   rh-haproxy18-haproxy.service - HAProxy Load Balancer
      Loaded: loaded (/usr/lib/systemd/system/rh-haproxy18-haproxy.service;
   disabled; vendor preset: disabled)
      Active: active (running) since Fri 2019-03-01 12:08:44 IST; 3s ago
   ```

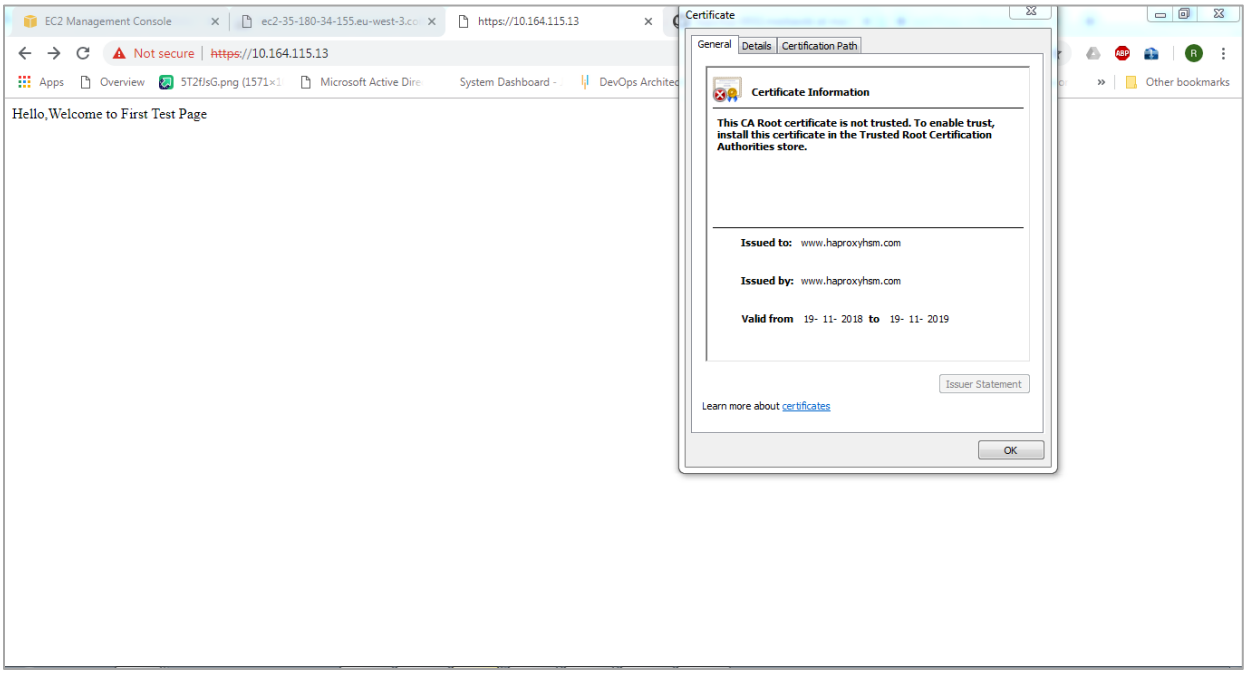
```
  Process: 17181 ExecStartPre=/opt/rh/rh-haproxy18/root/usr/sbin/haproxy -f
$CONFIG -c -q (code=exited, status=0/SUCCESS)
 Main PID: 17189 (haproxy)
    Tasks: 2
   CGroup: /system.slice/rh-haproxy18-haproxy.service
           +-17189 /opt/rh/rh-haproxy18/root/usr/sbin/haproxy -Ws -f
/etc/opt/rh/rh-haproxy18/haproxy/haproxy.cfg -p /run/rh-haproxy18-haproxy.pid
           +-17197 /opt/rh/rh-haproxy18/root/usr/sbin/haproxy -Ws -f
/etc/opt/rh/rh-haproxy18/haproxy/haproxy.cfg -p /run/rh-haproxy18-haproxy.pid

Mar 01 12:08:42 localhost.localdomain systemd[1]: Starting HAProxy Load
Balancer...
Mar 01 12:08:43 localhost.localdomain haproxy[17181]: STC client identity not
configured
Mar 01 12:08:44 localhost.localdomain haproxy[17189]: STC client identity not
configured
Mar 01 12:08:44 localhost.localdomain systemd[1]: Started HAProxy Load
Balancer.
```

8. Open any browser and access the HAProxy load balancer.

   `https://<HostName or IP Address>:443`

9. Accept the certificate and verify its details.




This completes the integration of HAProxy with the Thales Luna HSM.

# Integrating Luna HSM with HAProxy by migrating existing SSL keys

To integrate Luna HSM with HAProxy by migrating existing SSL keys, the HAProxy server should already be 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 use 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
   ```

   Here, `server.key` is the software key.
4. Extract the private key in PKCS#8 format using the following command.

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

   Here, `server.key` is the software key.
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
   haproxy_public_key  -pubkey=rsa
   ```

   **For private key:**

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

   Provide the partition password when prompted.
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) v10.3.0-275. Copyright (c) 2020
   SafeNet. All rights reserved.

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

   handle=2000001  label=CMU Unwrapped RSA Private Key

   handle=2000002  label=haproxy_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=2000001 -
   label=haproxy_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) v10.3.0-275. Copyright (c) 2020
SafeNet. All rights reserved.

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

handle=2000001  label=haproxy_private_key

handle=2000002  label=haproxy_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 password and key handle when prompted. After successful execution of the `sautil` command, HSMKey_ref.pem will be generated.

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 server.key privatekey.pem
    ```

12. Copy the private key reference and certificate to a new file at `/etc/pki/tls/certs/haproxy.pem` using the following command:

    ```
    # cat HSMKey_ref.pem softcert.cer > /etc/pki/tls/certs/haproxy.pem
    ```

    Here, `softcert.cer` is the existing certificate that was generated earlier using the software key.

13. Open the `/etc/opt/rh/rh-haproxy18/haproxy/haproxy.cfg` file in a text editor and add the following code:

    ```
    global
       log         127.0.0.1 local2
       pidfile     /var/run/rh-haproxy18-haproxy.pid
       maxconn     4000
       pidfile     /var/run/haproxy.pid
       user        haproxy
       group       hsmusers
       daemon
       ssl-engine gem
       tune.ssl.default-dh-param 2048


    defaults
       mode                    http
       log                     global
       option                  httplog
       option                  dontlognull
       retries                 3
    ```

```
    timeout http-request    10s
    timeout queue           1m
    timeout connect         10s
    timeout client          1m
    timeout server          1m
    timeout http-keep-alive 10s
    timeout check           10s

frontend  https-in
    bind *:80
    use_backend static
    option forwardfor
    bind *:443 ssl crt /etc/pki/tls/certs/haproxy.pem

backend static
    balance     roundrobin
    server server1.example.com 10.164.76.117:80 check
    server server2.example.com 10.164.78.118:80 check
```

> **NOTE:** The actual path for **haproxy.cfg** may vary, depending on the HAProxy Installation. In addition, the fields in **haproxy.cfg** may vary according to your requirement. Follow the *HAProxy Documentation* for detailed information of each field. You must add **ssl-engine gem** to the **global** section so that OpenSSL can use GemEngine and provide the correct certificate location.

14. Check the SELinux setting. Set it to permissive mode by running the following command:

    ```
    # setenforce 0
    ```

    This will temporarily set SELinux to permissive mode. To make a permanent change, go to `/etc/selinux/config` and edit the following line:

    ```
    SELINUX= permissive
    ```

    Reboot the system.

    ```
    # reboot
    ```

    > **NOTE:** If you are using Ubuntu then skip this step.

15. Restart the HAProxy service.

    ```
    # systemctl restart haproxy.service
    ```

    Ensure that you have added the HAProxy service to the firewall daemon to allow traffic through the firewall.

16. Check the server status and verify that it starts without any error.

```
# systemctl status haproxy.service
```

```
haproxy.service - HAProxy Load Balancer
   Loaded: loaded (/usr/lib/systemd/system/haproxy.service; disabled; vendor
   preset: disabled)
   Active: active (running) since Tue 2021-02-23 14:40:21 IST; 1h 35min ago
     Docs: man:haproxy(1)
           file:/usr/share/doc/haproxy/configuration.txt.gz
  Process: 164765 ExecStart=/usr/sbin/haproxy -W -f $CONFIG -p $PIDFILE
$EXTRAOPTS (  code=exited, status=0/SUCCESS)
   Process: 164754 ExecStartPre=/usr/sbin/haproxy -f $CONFIG -c -q $EXTRAOPTS
(code=exited, status=0/SUCCESS)
 Main PID: 164775 (haproxy)
    Tasks: 3 (limit: 23814)
   Memory: 24.9M
   CGroup: /system.slice/haproxy.service
           ├─164775 /usr/sbin/haproxy -W -f /etc/haproxy/haproxy.cfg -p
      /run/haproxy.pid
           └─164776 /usr/sbin/haproxy -W -f /etc/haproxy/haproxy.cfg -p
      /run/haproxy.pid

Feb 23 14:40:21 localhost.localdomain haproxy[164765]: [NOTICE] 053/144021
(164775) : New worker #1 (164776) forked
Feb 23 14:40:21 localhost.localdomain   systemd[1]: Started HAProxy Load
Balancer.
```

**17.** Open any browser and access the HAProxy load balancer.

```
https://<HostName or IP Address>:443
```

**18.** Accept the certificate and verify its details.

This completes the integration of Luna HSM with HAProxy by migrating existing SSL keys.

# Appendix

This section contains detailed instructions and procedures for:

> Installing HAProxy with Custom OpenSSL

> Configuring Backend Servers for HAProxy

> Configuring HAProxy Server to run in chroot

## Installing HAProxy with Custom OpenSSL

To install HAProxy with custom OpenSSL:

**1.** Install the following packages.

```
# apt install make gcc perl libpcre3-dev  zlib1g-dev -y
```

**2.** Download the HAProxy source file.

```
# wget https://www.haproxy.org/download/1.8/src/haproxy-1.8.8.tar.gz
```

**3.** Extract it to any location.

```
# tar xzf haproxy-1.8.8.tar.gz
```

4. Change the directory where you have extracted the zipped file.

```
# cd haproxy-1.8.8
```

5. Run the `make` command with following Flags:

```
# make TARGET=generic  USE_OPENSSL=1 SSL_INC=/usr/local/ssl/include
SSL_LIB=/usr/local/ssl/lib/ USE_PCRE=1 USE_ZLIB=1 USE_GETADDRINFO=1
USE_REGPARM=1   USE_PCRE_JIT=1 USE_NS=1
```

> **NOTE:** Make sure to provide correct include and lib path in SSL_INC and SSL_LIB respectively for Custom OpenSSL that you have installed.

6. Install the HAProxy.

```
# make install
```

7. Copy HAProxy binary file to /usr/sbin/haproxy.

```
# cp /usr/local/sbin/haproxy  /usr/sbin/haproxy
```

8. Verify that it was installed correctly with desired Custom OpenSSL version.

```
# haproxy -vv
```

9. Create Service file /lib/systemd/system/haproxy.service and add the following lines:

```
[Unit]
Description=HAProxy Load Balancer
Documentation=man:haproxy(1)
Documentation=file:/usr/share/doc/haproxy/configuration.txt.gz
After=network.target rsyslog.service
[Service]
EnvironmentFile=-/etc/default/haproxy
Environment="CONFIG=/etc/haproxy/haproxy.cfg" "PIDFILE=/run/haproxy.pid"
Environment="LD_LIBRARY_PATH=/usr/local/ssl/ssl/lib/"
ExecStartPre=/usr/sbin/haproxy -f $CONFIG -c -q $EXTRAOPTS
ExecStart=/usr/sbin/haproxy -W -f $CONFIG -p $PIDFILE $EXTRAOPTS
ExecReload=/usr/sbin/haproxy -f $CONFIG -c -q $EXTRAOPTS
ExecReload=/bin/kill -USR2 $MAINPID
KillMode=mixed
Restart=always
SuccessExitStatus=143
Type=forking
[Install]
WantedBy=multi-user.target
```

> **NOTE:** Make sure to point the Environment="LD_LIBRARY_PATH" to custom openssl lib folder.

10. Create haproxy config file **/etc/haproxy/haproxy.cfg** and add the following lines:

```
global
        log /dev/log    local0
        log /dev/log    local1 notice
        stats socket /run/haproxy/admin.sock mode 660 level admin expose-fd
listeners
        stats timeout 30s
        user haproxy
        group haproxy
        daemon
defaults
        log     global
        mode    http
        option  httplog
        option  dontlognull
```

11. Add haproxy user:

```
# id -u haproxy &> /dev/null || useradd -s /usr/sbin/nologin -r haproxy
```

12. Create a directory /run/haproxy.

```
# mkdir -p /run/haproxy
```

13. Reload the daemon and restart the haproxy service.

```
# systemctl daemon-reload
# systemctl start haproxy.service
```

Ensure that HAProxy service start successfully and then follow the instructions provided in the section "Integrating Luna HSM with HAProxy"

## Configuring Backend Servers for HAProxy

To configure backend servers for HAProxy:

1. Install apache server.

```
# apt install httpd -y
```

2. Modify your webpage accordingly in /var/www/html/index.html file.
3. Start the httpd service.

```
# systemctl start httpd
```

## Configuring HAProxy Server to run in chroot

To configure the HAProxy Server to run in chroot:

**NOTE:** This guide assumes that chroot is done in /var/lib/haproxy. If you are using some other directory, you need to change the path accordingly.

1. Complete the steps mentioned in the Integrating Luna HSM with HAProxy section.
2. Stop the haproxy service if it is running.

   ```
   # systemctl stop haproxy.service
   ```
3. Create the directory in which you want to chroot.

   ```
   # mkdir -p /var/lib/haproxy
   ```
4. Create /var/lib/haproxy/usr/safenet directory.

   ```
   # mkdir -p /var/lib/haproxy/usr/safenet
   ```
5. Mount /usr/safenet/ to /var/lib/haproxy/usr/safenet.

   ```
   # mount --bind /usr/safenet/ /var/lib/haproxy/usr/safenet/
   ```
6. Create /var/lib/haproxy/etc directory and copy /etc/Chrystoki.conf file to it.

   ```
   # mkdir /var/lib/haproxy/etc
   # cp /etc/Chrystoki.conf /var/lib/haproxy/etc/
   ```
7. Provide haproxy user permission to the /var/lib/haproxy/etc/Chrystoki.conf file.

   ```
   # setfacl  -m u:haproxy:rwx /var/lib/haproxy/etc/Chrystoki.conf
   ```
8. Create /var/lib/haproxy/tmp directory.

   ```
   # mkdir /var/lib/haproxy/tmp
   ```
9. Mount the /tmp to /var/lib/haproxy/tmp/ directory.

   ```
   # mount --bind /tmp/ /var/lib/haproxy/tmp/
   ```
10. Add the following to the global section in the /etc/haproxy/haproxy.cfg file.

    ```
    chroot /var/lib/haproxy
    ```
11. Reload the daemon and start the haproxy service.

    ```
    # systemctl daemon-reload
    # systemctl start haproxy.service
    ```

**NOTE:** All the mounts in above steps are lost after reboot. To avoid this, add these mounts in the /etc/fstab file.

# 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.