THALES

Red Hat Single Sign-On

INTEGRATION GUIDE
THALES LUNA HSM

Document Information

Document Part Number	007-001981-001
Revision	A
Release Date	16 August 2023

Trademarks, Copyrights, and Third-Party Software

CONTENTS

Overview....4
Certified Platforms....4
Prerequisites....5
Configure Luna HSM....5
Download Red Hat Single Sign-On Luna SPI Patch....6
Install Java Development Kit....6
Set up Red Hat Single Sign-On Server....7
Configuring Red Hat Single Sign-On to use Luna HSM....7
Configure Java for Luna Keystore....8
Generate signing key and certificate on Luna Keystore....9
Configure Red Hat Single Sign-On for Luna Plugin....11
Configure Red Hat Single Sign-On to use signing keys from Luna Keystore....15
Contacting Customer Support....20
Customer Support Portal....20
Telephone Support....20

Overview

Red Hat Single Sign-On simplifies security for web apps and RESTful web services by offering a ready-made solution for single sign-on. It streamlines the process for application developers to secure their deployed apps and services within their organization. This platform provides pre-built security features, eliminating the need for developers to create them from scratch. It adheres to open protocol standards like OpenID Connect and SAML 2.0 to ensure application security.

This guide demonstrates how to generate Red Hat Single Sign-On realm signing keys using Luna HSM. These keys are essential for signing access tokens and XML documents exchanged between the authentication server and the application. Utilizing Luna HSM for key generation brings forth significant benefits, including:

- Ensuring secure key generation, storage, and protection through FIPS 140-2 level 3 validated hardware.
- Providing full life cycle management of the keys.
- Maintaining an audit trail through HSM.
- Achieving significant performance enhancements by offloading cryptographic operations from application servers.

Certified Platforms

This integration is tested on the following platforms:

HSM Type	Platforms Tested	JDK Version	Red Hat Single Sign-On Version
Luna HSM	RHEL 8	Open JDK 11	RH-SSO 7.6

Thales Luna HSM: Thales 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. Thales Luna HSMs physically and logically secure cryptographic keys and accelerate cryptographic processing. The Luna HSM on premise offerings include the Luna Network HSM, PCIe HSM, and Luna USB HSMs.

NOTE: To seamlessly integrate with Red Hat Single Sign-On, ensure that you are using Luna Client version 10.4 or a newer release.

NOTE: You can obtain a patch from Thales Support that incorporates the Luna SPI plugin, enabling the utilization of Luna Keystore by Red Hat Single Sign-On to create and manage signing keys through Luna HSM; please be aware that despite potential references to Keycloak in the patch, it is also fully compatible with Red Hat Single Sign-On.

Prerequisites

Complete the following tasks before you begin the installation:

Configure Luna HSM

Set up and configure the Luna HSM device for your system. Refer to Thales Luna HSM Product Documentation for help.

1. Ensure that the HSM is set up, initialized, provisioned, and ready for deployment.
2. Create a partition on the HSM for use by Red Hat Single Sign-On.
3. If you are using a Thales Luna Network HSM, register a client for the system and assign the client to a partition to create an NTLS connection.
4. Initialize the Crypto Officer and Crypto User roles for the initialized partition.
5. Verify that the partition is successfully registered and configured. The command to see the registered partition is:

```
lunacm (64-bit) v10.4.0-417. Copyright (c) 2021 SafeNet. All rights
reserved.


        Available HSMs:

        Slot Id ->              0
        Label ->                TPA01
        Serial Number ->        1312109861412
        Model ->                LunaSA 7.7.1
        Firmware Version ->     7.7.1
        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 ->         Non-FM

        Current Slot Id: 0
```

NOTE: Refer to Thales Luna Network HSM Product Documentation for detailed steps about creating NTLS connection, initializing partition, and assigning user roles.

Control user access to HSM

NOTE: This section is applicable only for Linux users.

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 users to the `hsmusers` group. The users you assign to the `hsmusers` group must exist on the client workstation.

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

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

Set up Thales Luna HSM High-Availability (HA)

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

Download Red Hat Single Sign-On Luna SPI Patch

Red Hat Single Sign-On does not directly support the Luna HSM Keystore. However, it offers a way to create a plugin through the Signature Provider Interface (SPI) that works with the Luna HSM Keystore. To access this capability, contact Thales Customer Support. Thales Customer Support will provide you a patch containing the Luna plugin that you can leverage to enable Red Hat Single Sign-On with Luna Keystore and signing keys generated on Luna HSM.

> **NOTE:** README attached to the patch refers to a single supported version of a Keycloak but this guide takes precedence as to what actual versions of Red Hat Single Sign-On are supported.

Install Java Development Kit

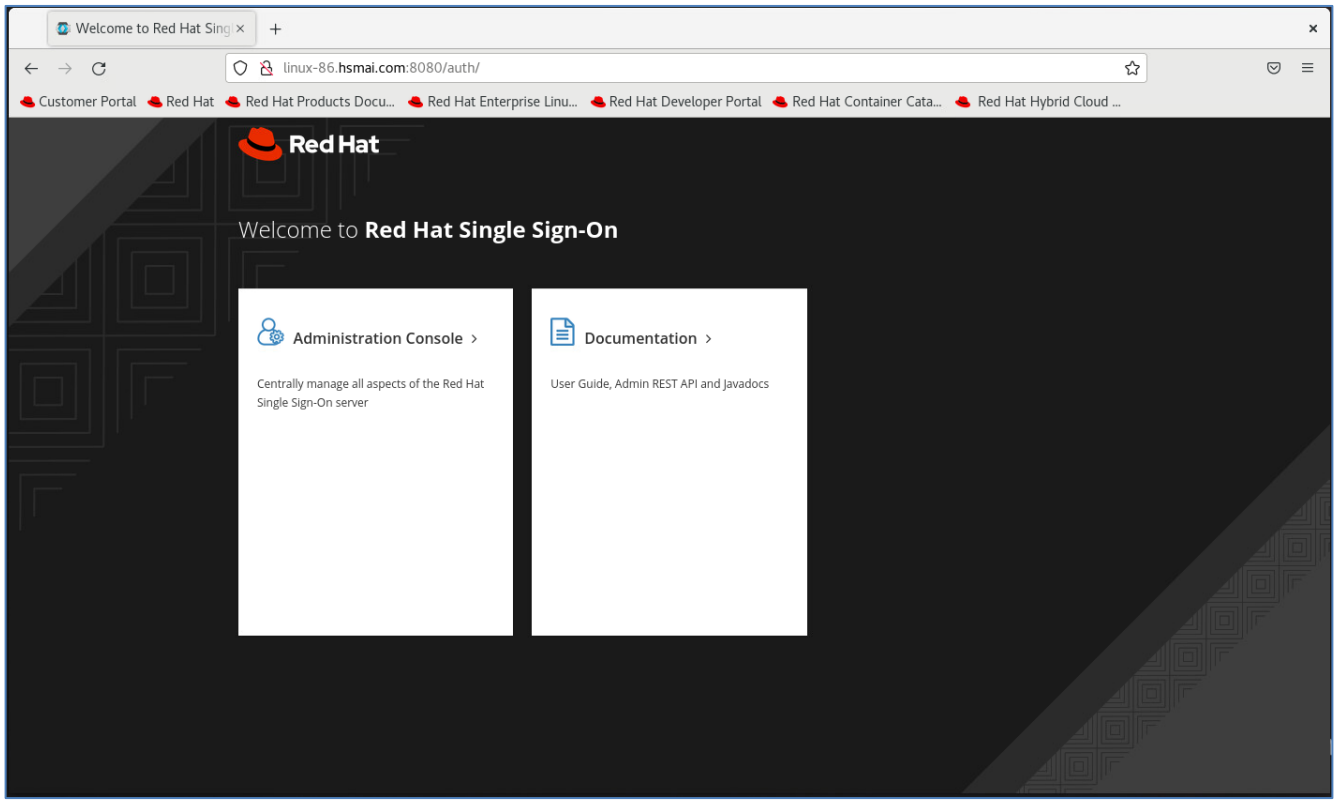
Ensure that the Java Development Kit (JDK) is installed on your server or local computer. You can run the commands provided in this guide, wherever you have the `keytool` utility available.

Set up Red Hat Single Sign-On Server

To log in to the Red Hat Single Sign-On Admin console, ensure that the Red Hat Single Sign-On server is installed and running, and you have created the initial Admin user. Consult the Red Hat Single Sign-On documentation for detailed instructions about installing and setting up an Admin account on Red Hat Single Sign-On server:

https://access.redhat.com/documentation/en-us/red_hat_single_sign-on/7.6/html/getting_started_guide/index

Click the Administration Console link on the Welcome page or go directly to http://hostname:8080/auth/admin/ (the console URL)



Configuring Red Hat Single Sign-On to use Luna HSM

This section demonstrates the steps required for generating realm signing keys and certificate on Luna HSM and configuring the Red Hat Single Sign-On to use the HSM generated key for token signing and XML document signing. Following are the steps involved in achieving Red Hat Single Sign-On and Luna HSM integration:

> Configure Java for Luna Keystore

> Generate signing keys and certificate on Luna Keystore

> Configure Red Hat Single Sign-On for Luna Plugin

> Configure Red Hat Single Sign-On to use signing keys from Luna Keystore

Configure Java for Luna Keystore

To generate the signing keys on Luna HSM through Java, configure your Java environment to use the Luna Provider.

1. Set the environment variables for `JAVA_HOME` and `PATH`.

```
# export JAVA_HOME=<JDK_installation_directory>
# export PATH=$JAVA_HOME/bin:$PATH
```

```
[root@linux-86 ~]# export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-11.0.20.0.8-3.el8.x86_64
[root@linux-86 ~]# export PATH=$JAVA_HOME/bin:$PATH
[root@linux-86 ~]#
```

2. Copy the `libLunaAPI.so` and `LunaProvider.jar` file from the `<Luna_installation_directory>/jsp/lib` directory to the `<JDK_installation_directory>/jre/lib/ext` directory.

> **NOTE:** This step is required for JDK 8, for JDK 11 skip this step.

3. Edit the Java Security Configuration file provider list, as shown below:

For JDK 11:

`$JAVA_HOME/conf/security/java.security`

```
security.provider.1=SUN
security.provider.2=SunEC
security.provider.3=SunJSSE
security.provider.4=SunJCE
security.provider.5=SunJGSS
security.provider.6=SunSASL
security.provider.7=XMLDSig
security.provider.8=SunPCSC
security.provider.9=JdkLDAP
security.provider.10=JdkSASL
security.provider.11=SunPKCS11
security.provider.12=com.safenetinc.luna.provider.LunaProvider
security.provider.13=SunRsaSign
```

For JDK 8:

`$JAVA_HOME/jre/lib/security/java.security`

```
security.provider.1=sun.security.provider.Sun
security.provider.2=sun.security.rsa.SunRsaSign
security.provider.3=sun.security.ec.SunEC
security.provider.4=com.sun.net.ssl.internal.ssl.Provider
```

```
security.provider.5=com.sun.crypto.provider.SunJCE
security.provider.6=sun.security.jgss.SunProvider
security.provider.7=com.sun.security.sasl.Provider
security.provider.8=org.jcp.xml.dsig.internal.dom.XMLDSigRI
security.provider.9=sun.security.smartcardio.SunPCSC
security.provider.10=com.safenetinc.luna.provider.LunaProvider
```

4. Save and close the `java.security` file after making the necessary changes.
5. Create a file with a custom name, such as “lunastore” (you can choose any name) and add the following entry:

   `tokenlabel:<partition_label>`

   Replace `<partition_label>` with the actual name of your Luna HSM partition.
6. Save the `lunastore` file in the current working directory. For instance, you can store it within a directory such as /opt.

Generate signing key and certificate on Luna Keystore

The signing keys and certificate will be generated on the Luna HSM using the Keytool utility from the JDK. To generate the signing keys, follow these steps:

1. Generate a signing key and certificate using the Java `keytool` utility in Luna keystore . This will generate the key pair in Luna HSM.

   For JDK 11:

   ```
   # keytool -genkeypair -alias lunakey -keyalg RSA -keysize 2048 -sigalg
   SHA256withRSA -keypass userpin1 -keystore lunastore -storetype luna -
   storepass userpin1 -providerpath
   "/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar" -providerclass
   com.safenetinc.luna.provider.LunaProvider -J-
   Djava.library.path=/usr/safenet/lunaclient/jsp/lib/ -J-cp -
   J/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar
   ```

   For JDK 8:

   ```
   # keytool -genkeypair -alias lunakey -keyalg RSA -keysize 2048 -sigalg
   SHA256withRSA -keypass userpin1 -keystore lunastore -storetype luna -
   storepass userpin1
   ```

   When the above command is run, you need to provide certificate details. A new key pair will be generated on the Luna HSM.

   > **NOTE:** The command above uses -storepass which is the partition’s Crypto Officer pin set during initializing the CO role for the partition. You need to use the Crypto Officer password in -keypass also.

2. Generate a certificate request for signing key in the Luna keystore.

   For JDK 11:

   ```
   # keytool -certreq -alias lunakey -sigalg SHA256withRSA -file certreq_file
   -keystore lunastore -storetype luna -providerpath
   ```

```
"/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar" -providerclass
com.safenetinc.luna.provider.LunaProvider -J-
Djava.library.path=/usr/safenet/lunaclient/jsp/lib/ -J-cp -
J/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar
```

For JDK 8:

```
# keytool -certreq -alias lunakey -sigalg SHA256withRSA -file certreq_file
-keystore lunastore -storetype luna
```

Enter the keystore password, when prompted. A file named `certreq_file` will be generated in the current directory.

3. Submit the CSR file to your Certification Authority (CA). The CA will authenticate the request with the Code Signing template and provide a signed certificate or a certificate chain. Save both the response and the CA root certificate in your current working directory.
4. Import the CA root certificate and signed certificate or certificate chain to your keystore.

For JDK 11:

To import the CA root certificate, execute the following command:

```
# keytool -trustcacerts -importcert -alias rootca -file root.cer -keystore
lunastore -storetype luna -providerpath
"/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar" -providerclass
com.safenetinc.luna.provider.LunaProvider -J-
Djava.library.path=/usr/safenet/lunaclient/jsp/lib/ -J-cp -
J/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar
```

To import the signed certificate reply or certificate chain, execute the following command:

```
# keytool -importcert -trustcacerts -alias lunakey -file signing.p7b -
keystore lunastore -storetype luna -providerpath
"/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar" -providerclass
com.safenetinc.luna.provider.LunaProvider -J-
Djava.library.path=/usr/safenet/lunaclient/jsp/lib/ -J-cp -
J/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar
```

For JDK 8:

To import the CA root certificate, execute the following command:

```
# keytool -trustcacerts -importcert -alias rootca -file root.cer -keystore
lunastore -storetype luna
```

To import the signed certificate reply or certificate chain, execute the following command:

```
# keytool -importcert -trustcacerts -alias lunakey -file signing.p7b -
  keystore lunastore -storetype luna
```

In the above commands, replace `root.cer` with your CA root certificate and `signing.p7b` with your signed certificate chain. Enter the keystore password, when prompted.

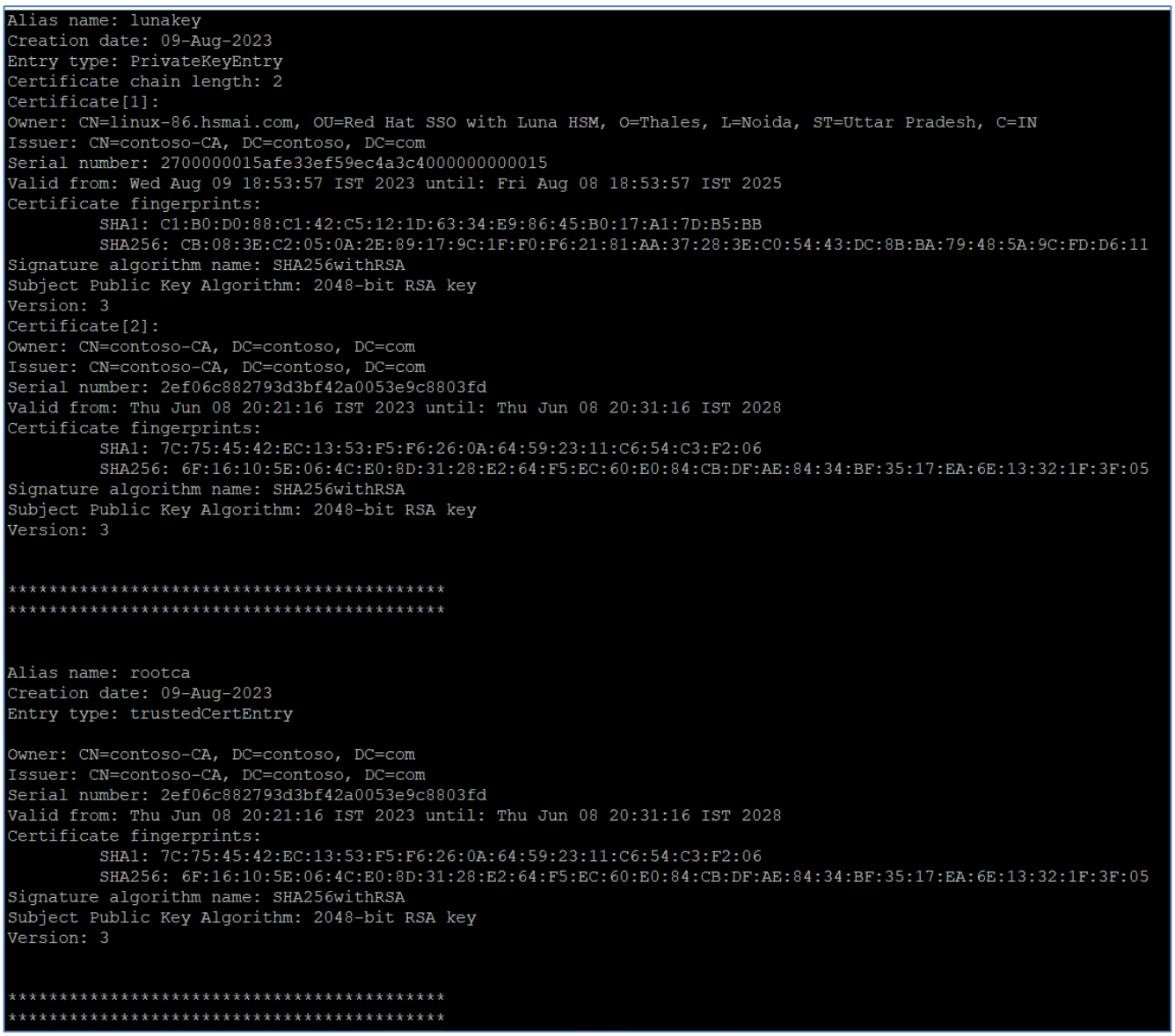
5. Verify the keystore contents in the Luna HSM.

For JDK 11:

To display the contents in `lunastore`, execute the following command:

```
# keytool -list -v -keystore lunastore -storetype luna -providerpath
"/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar" -providerclass
```

```
com.safenetinc.luna.provider.LunaProvider -J-
Djava.library.path=/usr/safenet/lunaclient/jsp/lib/ -J-cp -
J/usr/safenet/lunaclient/jsp/lib/LunaProvider.jar
```



For JDK 8:

To display the contents in `lunastore`, execute the following command:

```
# keytool -list -v -keystore lunastore -storetype luna
```

Configure Red Hat Single Sign-On for Luna Plugin

Contact Thales Customer Support to obtain a patch containing the Luna plugin that will enable Red Hat Single Sign-On to use Luna Keystore and signing keys generated on Luna HSM.

1. Extract the patch and copy the Luna Plugin zip file (extracted from the patch) to the Red Hat Single Sign-On modules directory.

```
# tar -xvf 630-000621-001_SW_Patch_keycloak_integration_Custom_Release.tar
# cp 630-000621-001_SW_Patch_keycloak_integration_Custom_Release/keycloak-spi-luna-keystore-1.1-assemblyModule.zip /opt/rh-sso/modules/
```

2. Traverse to the Red Hat Single Sign-On modules directory and extract the plugin zip file.

```
# cd /opt/rh-sso/modules/
# unzip keycloak-spi-luna-keystore-1.1-assemblyModule.zip
```

```
[root@linux-86 modules]# unzip keycloak-spi-luna-keystore-1.1-assemblyModule.zip
Archive:  keycloak-spi-luna-keystore-1.1-assemblyModule.zip
   creating: com/
   creating: com/safenetinc/
   creating: com/safenetinc/luna/
   creating: com/safenetinc/luna/keycloak/
   creating: com/safenetinc/luna/keycloak/provider/
  inflating: com/safenetinc/luna/keycloak/provider/module.xml
  inflating: com/safenetinc/luna/keycloak/provider/keycloak-spi-luna-keystore-1.1.jar
[root@linux-86 modules]#
```

3. Navigate to the `/opt/rh-sso/modules/com/safenetinc/luna` directory and create a `main/lib/linux-x86_64` folder structure underneath it:

```
# cd /opt/rh-sso/modules/com/safenetinc/luna/
# mkdir -p main/lib/linux-x86_64
```

4. Copy the `LunaProvider.jar` file into the newly created main folder within the `com\safenetinc\luna` directory.

```
# cp /usr/safenet/lunaclient/jsp/lib/LunaProvider.jar main/
```

5. Copy `libLunaAPI.so` file into the `main/lib/linux-x86_64` directory that you previously created under the `com\safenetinc\luna` directory.

```
# cp /usr/safenet/lunaclient/jsp/lib/libLunaAPI.so main/lib/linux-x86_64
```

6. Create a file `module.xml` in the `main` folder and add the following information in it.

```
# cd main
# vi module.xml
```

```
<module name="com.safenetinc.luna" xmlns="urn:jboss:module:1.9">
  <resources>
    <resource-root path="LunaProvider.jar"/>
  </resources>
  <dependencies>
   <module name="java.logging"/>
  </dependencies>
</module>
```

7. Ensure that the `LunaProvider.jar`, `module.xml`, and `lib/linux-x86_64/libLunaAPI.so` files are present in the `/opt/rh-sso/modules/com/safenetinc/luna/main` directory that you've created.

```
[root@linux-86 main]# ll /opt/rh-sso/modules/com/safenetinc/luna/main
total 652
drwxr-xr-x. 3 root root     26 Aug  9 19:36 lib
-rw-r--r--. 1 root root 660512 Aug  9 19:38 LunaProvider.jar
-rw-r--r--. 1 root root    218 Aug  9 19:39 module.xml
```

8. Traverse to the `com/safenetinc/luna/keycloak/provider` directory created in the Red Hat Single Sign-On modules directory.

```
# cd /opt/rh-sso/modules/com/safenetinc/luna/keycloak/provider/
```

9. Create a `main` folder under the `com/safenetinc/luna/keycloak/provider` directory.

```
# mkdir main
```

10. Move the `keycloak-spi-luna-keystore-1.1.jar` and `module.xml` to the main directory.

```
# mv keycloak-spi-luna-keystore-1.1.jar module.xml main/
```

11. Ensure that the `keycloak-spi-luna-keystore-1.1.jar` and `module.xml` are present in the `/opt/rh-sso/modules/com/safenetinc/luna/keycloak/provider/main` directory that you've created.

```
[root@linux-86 provider]# ll /opt/rh-sso/modules/com/safenetinc/luna/keycloak/provider/main
total 16
-rw-r--r--. 1 root root 9948 May 25  2022 keycloak-spi-luna-keystore-1.1.jar
-rw-r--r--. 1 root root  752 May 19  2022 module.xml
```

12. Ensure that the `module.xml` file has the following content, including the name of the Plugin jar file.

```
<?xml version='1.0' encoding='UTF-8'?>
<module xmlns="urn:jboss:module:1.3" name="com.safenetinc.luna.keycloak.provider" slot="main">
    <resources>
        <resource-root path="keycloak-spi-luna-keystore-1.1.jar"/>
    </resources>
    <dependencies>
        <module name="org.keycloak.keycloak-server-spi" services="import"/>
        <module name="org.keycloak.keycloak-server-spi-private" services="import"/>
        <module name="org.keycloak.keycloak-core"/>
        <module name="org.keycloak.keycloak-common"/>
        <module name="org.keycloak.keycloak-services"/>
        <module name="org.jboss.resteasy.resteasy-jaxrs"/>
        <module name="org.jboss.logging"/>
        <module name="com.safenetinc.luna"/>
    </dependencies>
</module>
```

13. Edit the Red Hat Single Sign-On configuration file `standalone.xml` to add the Provider and SPI details as follows:

```
# vi /opt/rh-sso/standalone/configuration/standalone.xml
```

```
    <provider>
        module:com.safenetinc.luna.keycloak.provider
    </provider>
```

```
.
.
.
<spi name="keys">
     <provider name="luna-keystore" enabled="true"/>
</spi>
```

In the XML file, you should include the provider and SPI details, as indicated in the screenshot provided below. Insert this information into the existing Provider and SPI sections. Specify the `luna-keystore` spi as the first SPI in the list.

```
<subsystem xmlns="urn:jboss:domain:jpa:1.1">
    <jpa default-extended-persistence-inheritance="DEEP"/>
</subsystem>
<subsystem xmlns="urn:jboss:domain:keycloak-server:1.1">
    <web-context>auth</web-context>
    <providers>
        <provider>
            classpath:${jboss.home.dir}/providers/*
        </provider>
        <provider>
            module:com.safenetinc.luna.keycloak.provider
        </provider>
    </providers>
    <master-realm-name>master</master-realm-name>
    <scheduled-task-interval>900</scheduled-task-interval>
    <theme>
        <staticMaxAge>2592000</staticMaxAge>
        <cacheThemes>true</cacheThemes>
        <cacheTemplates>true</cacheTemplates>
        <dir>${jboss.home.dir}/themes</dir>
    </theme>
    <spi name="keys">
        <provider name="luna-keystore" enabled="true"/>
    </spi>
    <spi name="eventsStore">
        <provider name="jpa" enabled="true">
            <properties>
                <property name="exclude-events" value="[&quot;REFRESH_TOKEN&quot;]"/>
            </properties>
        </provider>
    </spi>
    <spi name="userCache">
        <provider name="default" enabled="true"/>
    </spi>
    <spi name="userSessionPersister">
        <default-provider>jpa</default-provider>
    </spi>
    <spi name="timer">
        <default-provider>basic</default-provider>
    </spi>
    <spi name="connectionsHttpClient">
        <provider name="default" enabled="true"/>
    </spi>
    <spi name="connectionsJpa">
```

14. Save and exit the `standalone.xml` file after making the changes. Next, if the Red Hat Single Sign-On server is currently running, stop it. Alternatively, start the server again by executing the following command:

    ```
    # /opt/rh-sso/bin/standalone.sh
    ```

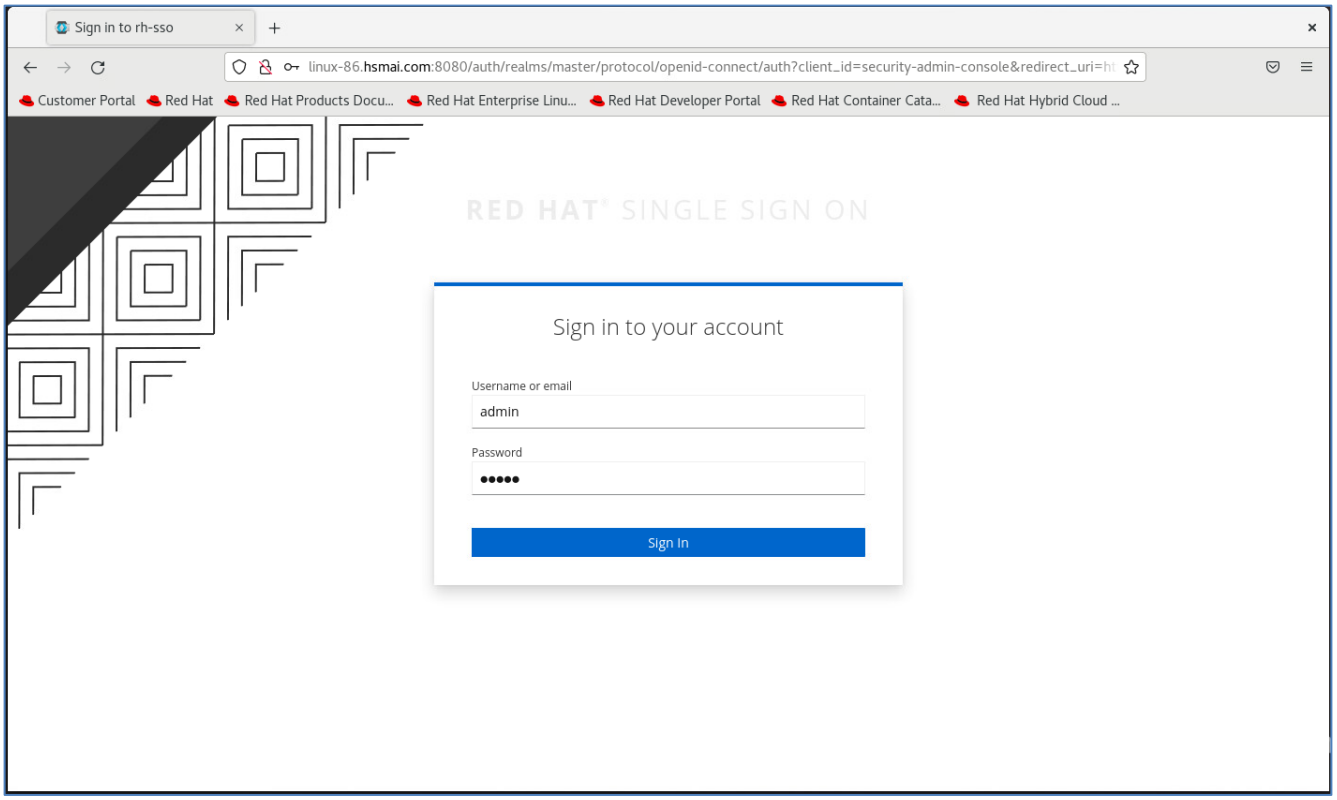
    If the Luna Plugin is enabled successfully, you will see the following information when the server starts.

```
l -- 63) Frontend: <request>, Admin: <frontend>, Backend: <request>
20:03:00,045 WARN  [org.keycloak.services] (ServerService Thread Pool -- 63) KC-SERVICES0047:
luna-keystore (com.safenetinc.luna.keycloak.provider.LunaKeystoreProviderFactory) is implement
ing the internal SPI keys. This SPI is internal and may change without notice
20:03:00,073 INFO  [org.jboss.as.clustering.infinispan] (ServerService Thread Pool -- 63) WFLY
```

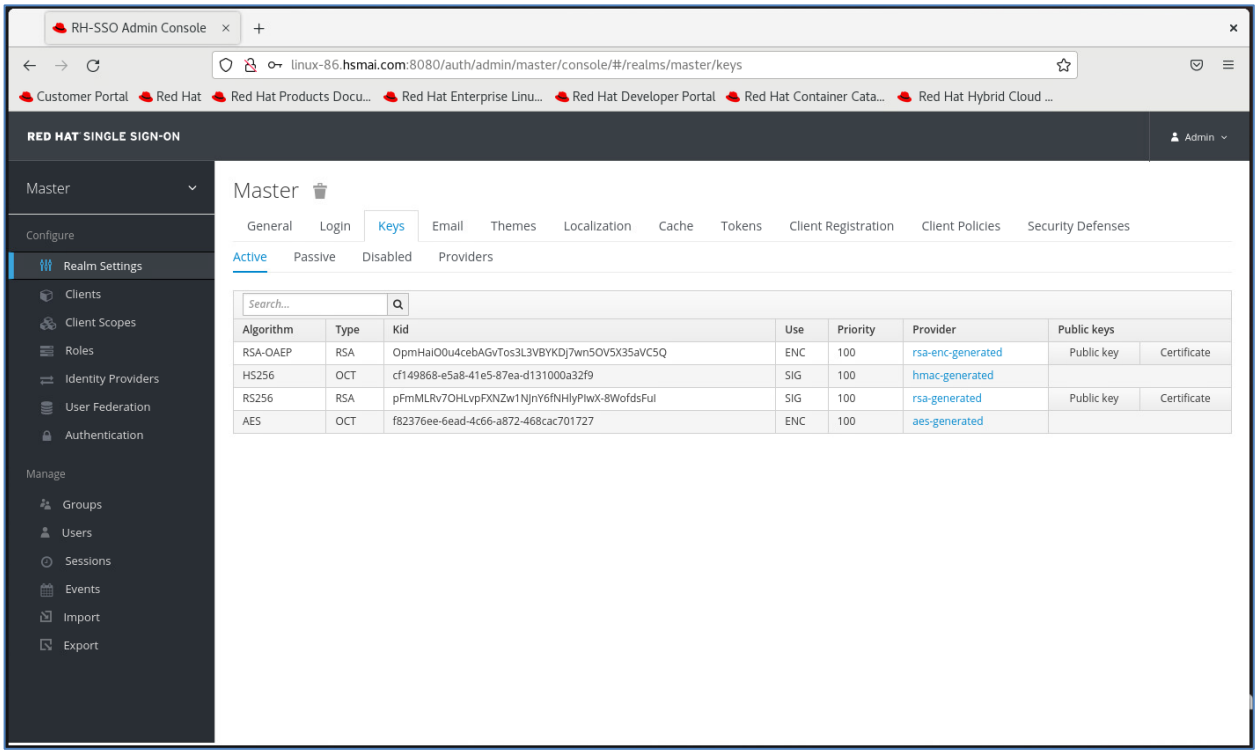
Configure Red Hat Single Sign-On to use signing keys from Luna Keystore

Log in to the Admin Console to configure Luna Keystore which will enable Red Hat Single Sign-On to use keys generated on Luna HSM.

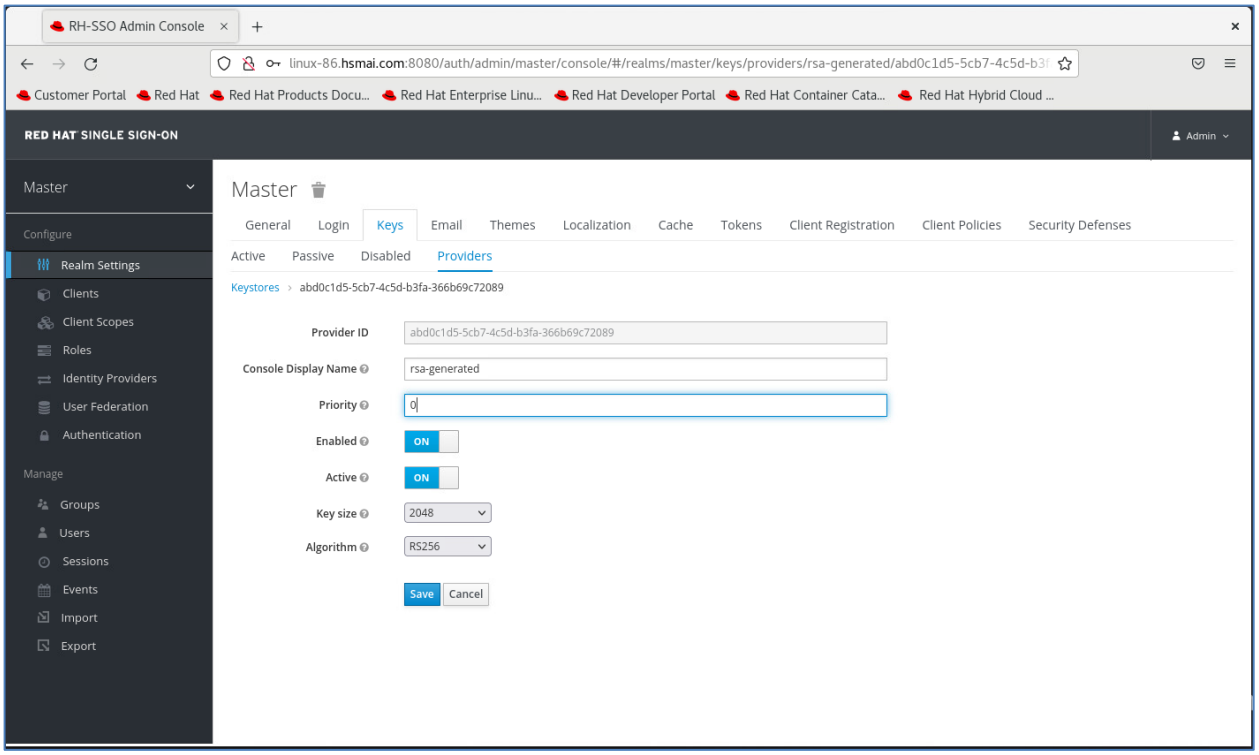
1. Log in to the Keycloak Admin Console using the URL http://hostname:8080/auth/admin.



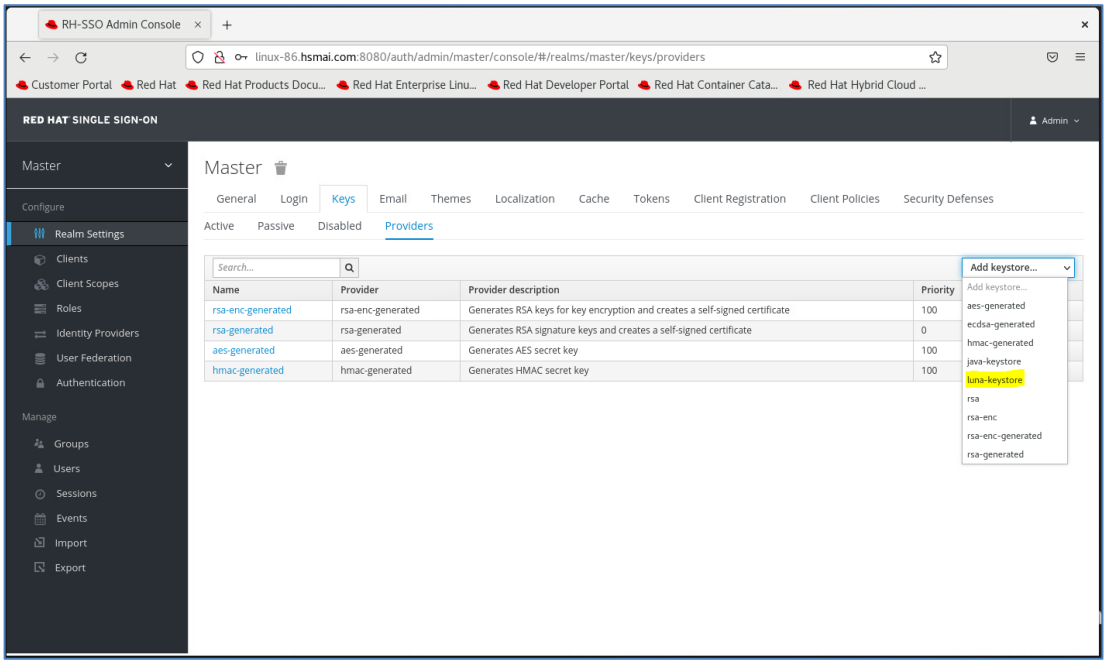
2. Provide your credentials and click on **Sign In**. Once logged-in, navigate to **Realm Settings** > **Keys**.



3. Click **rsa-generated** key in **Provider** column and change the **Priority** from 100 to 0. Click **Save**.



4. Navigate to **Realm Settings** > **Keys** > **Providers**. Click **Add keystore…** and select **luna-keystore**.



5. Enter the values for **Priority**, **Keystore**, **Keystore Password**, **Key Alias** and **Key Password**. Click **Save**.

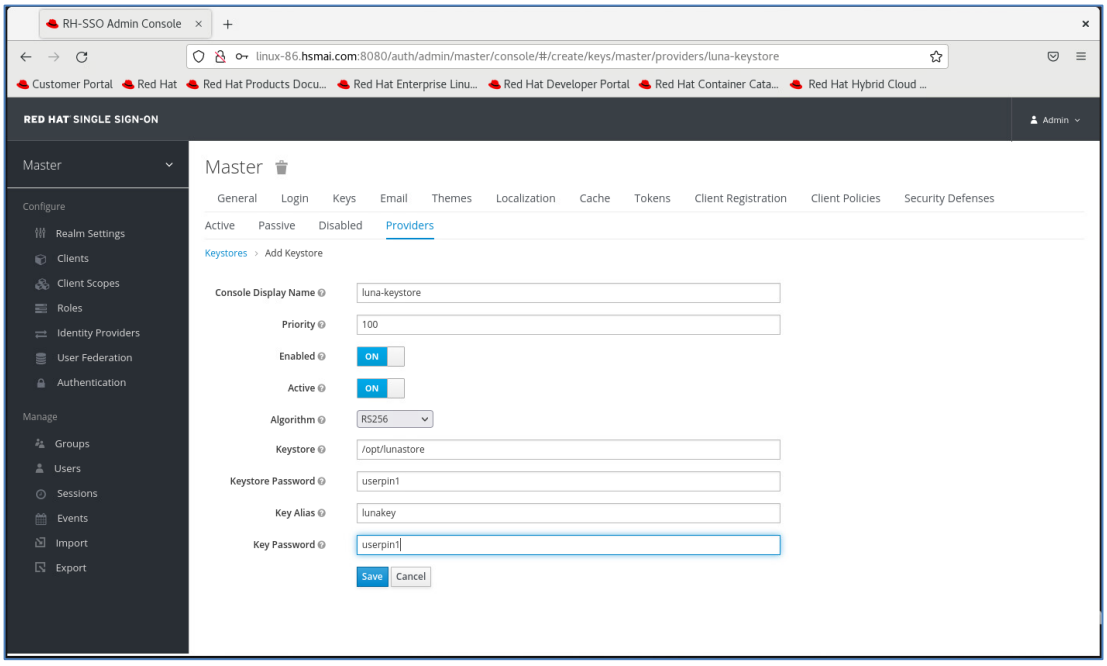
Priority = 100

Keystore = Path to lunastore file

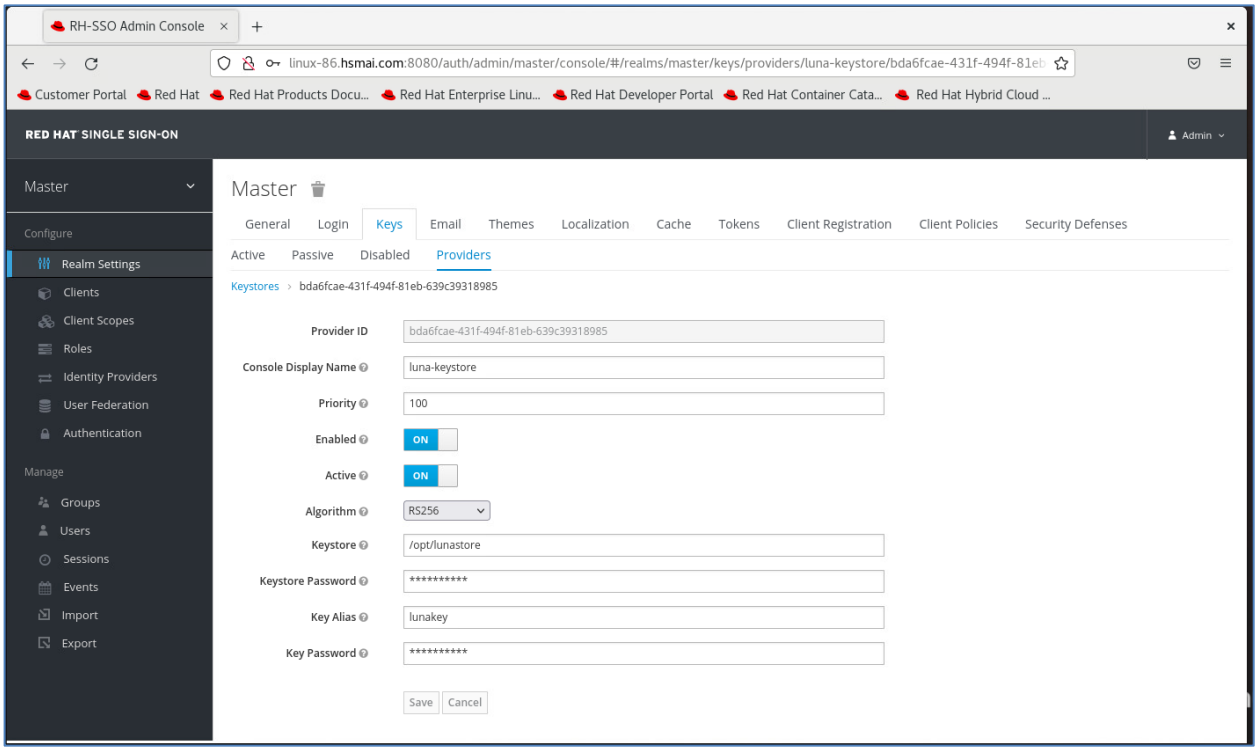
KeyStore Password = Partition CO password

Key Alias = Label of the key generated on Luna HSM

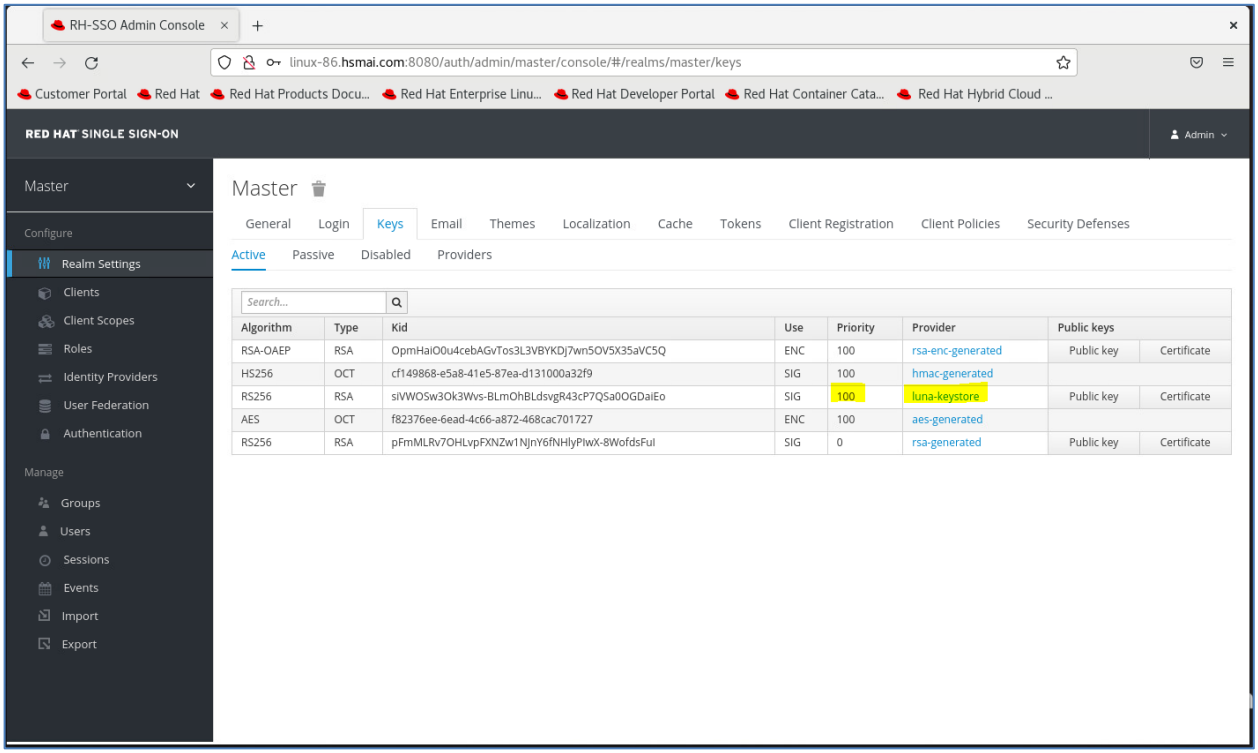
Key Password = key password set while generating the key pair



Once the provider is successfully created, a confirmation message will appear, and the associated buttons will be disabled, as depicted in this example:



6. Go to **Realm Settings** > **Keys** and make sure that **luna-keystore** Provider and a key created on Luna HSM are displayed with the highest priority (100) in the list.



7. Restart the Red Hat Single Sign-On server by running the following command and then access the admin console.

```
# /opt/rh-sso/bin/standalone.sh
```

For every login session, the token will be signed using the key generated by Luna HSM. This enables you to create User, Roles, and Client with Red Hat Single Sign-On. Any Authentication Token generated by Red Hat Single Sing-On will be signed by Realm Signing Key stored in the Luna HSM. If the Luna HSM is not available or if the NTLS is not operational, logging into the admin console will not be possible since the Red Hat Single Sign-On server will not be able to retrieve the signing keys from the Luna HSM.

This concludes the process of integrating Red Hat Single Sign-On with Luna HSM, ensuring the secure storage and utilization of signing keys and certificates on Luna HSM.

Contacting Customer Support

If you encounter a problem while installing, registering, or operating this product, 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.