
RedHat OpenShift Luna HSM Operator: Integration Guide

THALES LUNA HSM

Document Information

Document Part Number	007- 001866 -001
Revision	A
Release Date	24 January 2023

Trademarks, Copyrights, and Third-Party Software

Copyright © 2023 Thales Group. All rights reserved. Thales and the Thales logo are trademarks and service marks of Thales Group and/or its subsidiaries and are registered in certain countries. All other trademarks and service marks, whether registered or not in specific countries, are the property of their respective owners.

CONTENTS

Overview	4
Certified Platforms	4
Prerequisites	4
Configure Luna HSM	4
Configure OpenShift Container Platform web console	6
Create secrets and Luna configuration file for container/pod	6
Integrating RedHat OpenShift Luna HSM Operator with Luna HSM	12
Install Thales_Luna_Operator in OpenShift Container Platform	12
Create instance and set up Luna partition using Thales_Luna_Operator	14
Verify the accessibility of Luna partition	17
Contacting Customer Support.....	19
Customer Support Portal	19
Telephone Support	19

Overview

OpenShift Container Platform offers a Kubernetes environment for managing the lifecycle of container-based applications and their dependencies on various computing platforms, such as bare metal, virtualized, on-premise, and in cloud. OpenShift Container Platform utilizes a number of computing resources, known as nodes. A node has a lightweight, secure operating system based on Red Hat Enterprise Linux (RHEL), known as Red Hat Enterprise Linux CoreOS (RHCOS).

Luna HSM service provides strong physical protection of secure assets, including keys, and should be considered a best practice when working with containerized applications in the OpenShift Container Platform environment.

Certified Platforms

This integration is certified on the following platforms:

HSM Type	Platforms Certified
Luna HSM	RedHat OpenShift Container Platform 4.9

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 CloudHSM Classic.

Prerequisites

Before you proceed with the any of the integrations described in this document, complete the following tasks:

Configure Luna HSM

To configure Luna HSM:

1. Verify that the HSM is set up, initialized, provisioned, and ready for deployment.
2. Create a partition on the HSM that will be later used by the container/pod in the OpenShift environment.
3. If you are 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 see the registered partition is:

```
# /usr/safenet/lunaclient/bin/lunacm
lunacm (64-bit) v10.4.0-417. Copyright (c) 2021 SafeNet. All rights reserved.

Available HSMs:

Slot Id ->          0
Label ->           LunaOperator
Serial Number ->   1312109862209
Model ->          LunaSA 7.7.1
Firmware Version -> 7.7.1
Bootloader Version -> 1.1.2
Configuration ->   Luna UserPartitionWithSO(PW)Key Export With Cloning Mode
Slot Description -> Net Token Slot
FM HW Status ->   Non-FM

Current Slot Id: 0
```

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 on creating NTLS connection, initializing the partitions, and assigning various user roles.

Set up Luna HSM High-Availability Group

Refer to the [Luna HSM documentation](#) for HA steps and details regarding configuring and setting up two or more HSM boxes on host systems. You must enable the HAOnly setting in HA for failover to work so that if the primary goes down due to any reason, all calls get automatically routed to the secondary until the primary recovers and starts up.

Set up Luna HSM in FIPS Mode

NOTE: This setting is not required for Luna HSM Universal Client. This setting is applicable only for Luna HSM Client 7.x.

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 Luna HSM in FIPS mode, you have to make the following change in the 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.

Configure OpenShift Container Platform web console

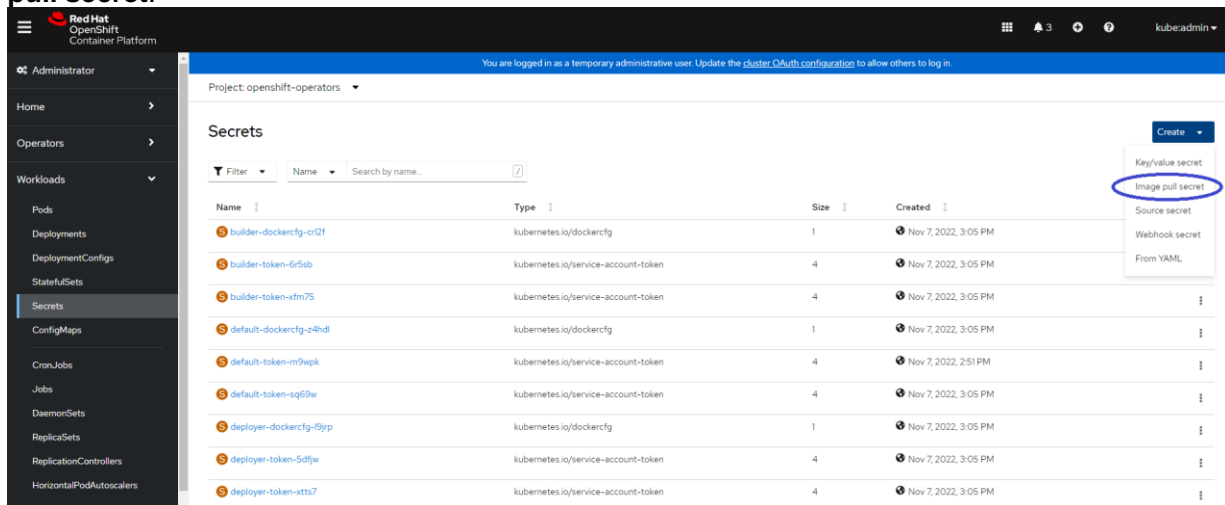
In order to execute this integration, an existing setup of OpenShift Container Platform is a must. Ensure that you have:

- Access to OpenShift Container Platform web console.
- Necessary permissions to create secrets and resources in **openshift-operators** project.
- Necessary permissions to install an operator from the Operator Hub in your OpenShift Container Platform.
- Write access to the nodes of the OpenShift cluster.

Create secrets and Luna configuration file for container/pod

To create the required secrets and Luna configuration file for the container/pod on which you will be accessing the Luna partition:

1. Login to the OpenShift Container Platform web console and go to the **Administrator** panel on the left hand side.
2. Go to **Workloads -> Secrets**. Select **openshift-operators** as **Project** value. Click **Create -> Image pull secret**.



The screenshot shows the OpenShift web console interface. The left sidebar is expanded to 'Workloads' and then 'Secrets'. The main content area shows the 'Secrets' page for the 'openshift-operators' project. A 'Create' button is visible in the top right, with a dropdown menu open showing options: 'Key/Value secret', 'Image pull secret' (highlighted with a red circle), 'Source secret', 'Webhook secret', and 'From YAML'. Below the table, there is a list of existing secrets with columns for Name, Type, Size, and Created.

Name	Type	Size	Created
builder-dockercfg-0r2f	kubernetes.io/dockercfg	1	Nov 7, 2022, 3:05 PM
builder-token-6r5ab	kubernetes.io/service-account-token	4	Nov 7, 2022, 3:05 PM
builder-token-xfm75	kubernetes.io/service-account-token	4	Nov 7, 2022, 3:05 PM
default-dockercfg-z4hd	kubernetes.io/dockercfg	1	Nov 7, 2022, 3:05 PM
default-token-m9apk	kubernetes.io/service-account-token	4	Nov 7, 2022, 2:51 PM
default-token-sq69w	kubernetes.io/service-account-token	4	Nov 7, 2022, 3:05 PM
deployer-dockercfg-9ryp	kubernetes.io/dockercfg	1	Nov 7, 2022, 3:05 PM
deployer-token-5dfje	kubernetes.io/service-account-token	4	Nov 7, 2022, 3:05 PM
deployer-token-xtts7	kubernetes.io/service-account-token	4	Nov 7, 2022, 3:05 PM

- Enter Secret name as **regcred**. Select **Authentication type** as **Image registry credentials**. Enter **Registry server address** as **registry.connect.redhat.com**. Enter your **Username**, **Password** and **Email** for RedHat repository in the respective fields and click **Create**. This secret will allow you to pull the required image from RedHat repository.

Project: openshift-operators

Create image pull secret

Image pull secrets let you authenticate against a private image registry.

Secret name *

regcred

Unique name of the new secret.

Authentication type

Image registry credentials

Registry server address *

registry.connect.redhat.com

For example quay.io or docker.io

Username *

redhatuser

Password *

.....

Email

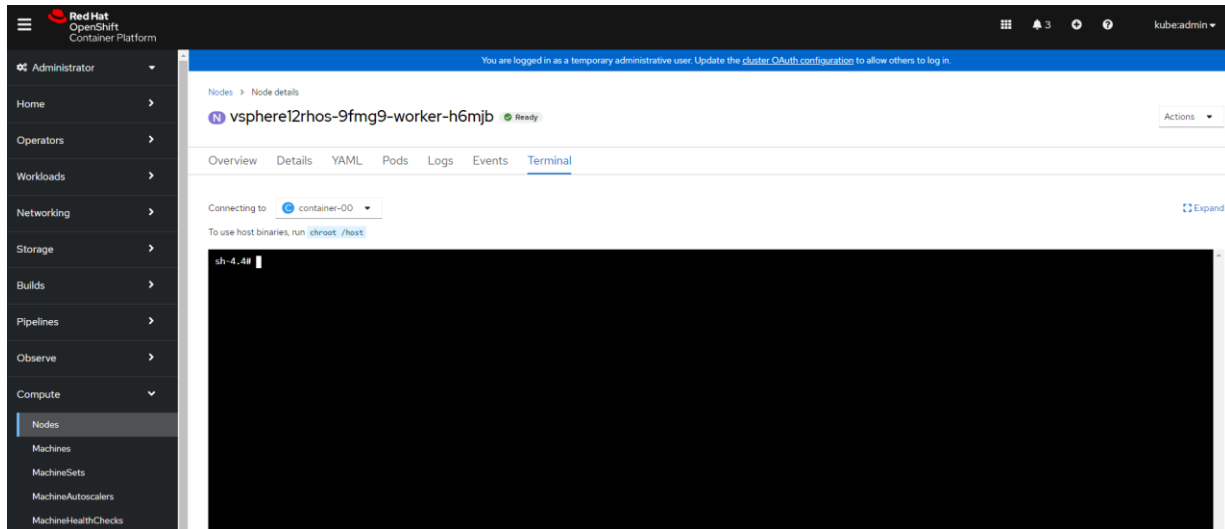
[Add credentials](#)

[Create](#) [Cancel](#)

- Go to **Compute -> Nodes**. Select a node.

Name	Status	Role	Pods	Memory	CPU	Filesystem	Created	Instance type
vsphere2rhos-9fmg9-master-0	Ready	master	34	7.08 GiB / 15.65 GiB	0.514 cores / 4 cores	17.09 GiB / 119.8 GiB	Nov 7, 2022, 2:58 PM	-
vsphere2rhos-9fmg9-master-1	Ready	master	37	7.7 GiB / 15.65 GiB	0.603 cores / 4 cores	17.36 GiB / 119.8 GiB	Nov 7, 2022, 2:59 PM	-
vsphere2rhos-9fmg9-master-2	Ready	master	56	9.86 GiB / 15.65 GiB	0.838 cores / 4 cores	16.93 GiB / 119.8 GiB	Nov 7, 2022, 2:59 PM	-
vsphere2rhos-9fmg9-worker-h5mjb	Ready	worker	26	5.09 GiB / 7.77 GiB	0.631 cores / 2 cores	44.51 GiB / 119.8 GiB	Nov 7, 2022, 3:16 PM	-
vsphere2rhos-9fmg9-worker-lhkxp	Ready	worker	24	2.5 GiB / 7.77 GiB	0.134 cores / 2 cores	45.31 GiB / 119.8 GiB	Nov 7, 2022, 3:16 PM	-
vsphere2rhos-9fmg9-worker-w259h	Ready	worker	23	4.46 GiB / 7.77 GiB	0.511 cores / 2 cores	39.87 GiB / 119.8 GiB	Nov 7, 2022, 3:16 PM	-

5. Go to the Terminal tab.



6. Create a shell script file by the name **OperatorPreReqs.sh**. Copy the following shell script to the file and give it executable permissions.

```
#!/bin/bash

export LC_ALL=C; unset LANGUAGE
export PWD=`pwd`

if [ -f /etc/Chrystoki.conf ]
then
    cp /etc/Chrystoki.conf "$PWD/Chrystoki.conf"
fi

if [ ! -f "$PWD/Chrystoki.conf" ]
then
    printf "\nLUNACLIENT CONFIGURATION FILE [Chrystoki.conf] NOT PRESENT AT [$PWD]
LOCATION AS [$PWD/Chrystoki.conf]. COPY THE LUNACLIENT CONFIGURATION FILE
[Chrystoki.conf] AT [$PWD] LOCATION. ABORTING SCRIPT EXECUTION ! \n"
    exit 1
fi

chmod 777 "$PWD/Chrystoki.conf"

export Chrystoki_ClientPrivKeyFile_Path=`cat "$PWD/Chrystoki.conf" | grep
"ClientPrivKeyFile =" | awk 'BEGIN{FS="="}{print $NF}' | sed 's/^ *///g' | sed 's/
*$///g' | sed 's/;/;/g'`
export Chrystoki_ClientPrivKeyFile_Name=`cat "$PWD/Chrystoki.conf" | grep
"ClientPrivKeyFile =" | sed 's/^ *///g' | sed 's/ *$///g' | awk 'BEGIN{FS="/" }{print
$NF}' | sed 's/;/;/g'`

export Chrystoki_ClientCertFile_Path=`cat "$PWD/Chrystoki.conf" | grep "ClientCertFile
=" | awk 'BEGIN{FS="="}{print $NF}' | sed 's/^ *///g' | sed 's/ *$///g' | sed 's/;/;/g'`
export Chrystoki_ClientCertFile_Name=`cat "$PWD/Chrystoki.conf" | grep "ClientCertFile
=" | sed 's/^ *///g' | sed 's/ *$///g' | awk 'BEGIN{FS="/" }{print $NF}' | sed 's/;/;/g'`

export Chrystoki_ServerCAFile_Path=`cat "$PWD/Chrystoki.conf" | grep "ServerCAFile ="
| awk 'BEGIN{FS="="}{print $NF}' | sed 's/^ *///g' | sed 's/ *$///g' | sed 's/;/;/g'`
export Chrystoki_ServerCAFile_Name=`cat "$PWD/Chrystoki.conf" | grep "ServerCAFile ="
| sed 's/^ *///g' | sed 's/ *$///g' | awk 'BEGIN{FS="/" }{print $NF}' | sed 's/;/;/g'`
```



```

if [ ! -f "$Chrystoki_ClientPrivKeyFile_Path" ] && [ ! -f
"$Chrystoki_ClientCertFile_Path" ]
then
    if [ ! -f "$PWD/$Chrystoki_ClientPrivKeyFile_Name" ] && [ ! -f
"$PWD/$Chrystoki_ClientCertFile_Name" ]
    then
        printf "\nCLIENT'S PRIVATE KEY FILE
[$Chrystoki_ClientPrivKeyFile_Name] AND CLIENT'S CERTIFICATE FILE
[$Chrystoki_ClientCertFile_Name] NOT PRESENT AT [$PWD] LOCATION. IF LUNACLIENT IS NOT
INSTALLED ON THIS HOST OR NTLS IS NOT SETUP ON THIS NODE THEN COPY THE FILE
[$Chrystoki_ClientPrivKeyFile_Name] AND [$Chrystoki_ClientCertFile_Name] AT [$PWD]
LOCATION. ABORTING SCRIPT EXECUTION ! \n
        exit 1
    else
        chmod 777 "$PWD/$Chrystoki_ClientCertFile_Name"
        chmod 777 "$PWD/$Chrystoki_ClientPrivKeyFile_Name"
        printf "\nCREATING SECRET FOR CLIENT'S PRIVATE KEY AND CERTIFICATE ...
\n"
        kubectl create secret generic operator-luna-client-secret --from-
file="$PWD/$Chrystoki_ClientCertFile_Name" --from-
file="$PWD/$Chrystoki_ClientPrivKeyFile_Name" --namespace=openshift-operators
        fi
    else
        chmod 777 "$Chrystoki_ClientCertFile_Path"
        chmod 777 "$Chrystoki_ClientPrivKeyFile_Path"
        printf "\nCREATING SECRET FOR CLIENT'S PRIVATE KEY AND CERTIFICATE ... \n"
        kubectl create secret generic operator-luna-client-secret --from-
file="$Chrystoki_ClientCertFile_Path" --from-file="$Chrystoki_ClientPrivKeyFile_Path"
--namespace=openshift-operators
        fi
if [ ! -f "$Chrystoki_ServerCAFile_Path" ]
then
    if [ ! -f "$PWD/$Chrystoki_ServerCAFile_Name" ]
    then
        printf "\n[$Chrystoki_ServerCAFile_Name] NOT PRESENT AT
[$PWD/$Chrystoki_ServerCAFile_Name]. IF LUNACLIENT IS NOT INSTALLED ON THIS HOST OR
NTLS IS NOT SETUP ON THIS NODE THEN COPY THE FILE [$Chrystoki_ServerCAFile_Name] AT
[$PWD] LOCATION. ABORTING SCRIPT EXECUTION ! \n"
        exit 1
    else
        chmod 777 "$PWD/$Chrystoki_ServerCAFile_Name"
        printf "\nCREATING SECRET FOR SERVER CA FILE ... \n"
        kubectl create secret generic operator-luna-cafile-secret --from-
file="$PWD/$Chrystoki_ServerCAFile_Name" --namespace=openshift-operators
        fi
    else
        chmod 777 "$Chrystoki_ServerCAFile_Path"
        printf "\nCREATING SECRET FOR SERVER CA FILE ... \n"
        kubectl create secret generic operator-luna-cafile-secret --from-
file="$Chrystoki_ServerCAFile_Path" --namespace=openshift-operators
        fi
export Chrystoki_LibUNIX=`cat "$PWD/Chrystoki.conf" | grep "LibUNIX =" | sed 's/^
*//g' | sed 's/ *$//g'`
export New_Chrystoki_LibUNIX='LibUNIX = /var/usrlocal/luna/libs/64/libCryptoki2.so;'

export Chrystoki_LibUNIX64=`cat "$PWD/Chrystoki.conf" | grep "LibUNIX64 =" | sed 's/^
*//g' | sed 's/ *$//g'`
export New_Chrystoki_LibUNIX64='LibUNIX64 =
/var/usrlocal/luna/libs/64/libCryptoki2_64.so;'

export Chrystoki SSLConfigFile=`cat "$PWD/Chrystoki.conf" | grep "SSLConfigFile =" |

```

```

sed 's/^ *//g' | sed 's/ *$//g'`
export New_Chrystoki_SSLConfigFile='SSLConfigFile = /var/usrlocal/luna/openssl.cnf;'

export Chrystoki_ClientPrivKeyFile=`cat "$PWD/Chrystoki.conf" | grep
"ClientPrivKeyFile =" | sed 's/^ *//g' | sed 's/ *$//g'`
export Chrystoki_ClientPrivKeyFile_Name=`cat "$PWD/Chrystoki.conf" | grep
"ClientPrivKeyFile =" | sed 's/^ *//g' | sed 's/ *$//g' | awk 'BEGIN{FS="/"}{print
$NF}' | sed 's/;//g'`
export New_Chrystoki_ClientPrivKeyFile="ClientPrivKeyFile =
/var/usrlocal/luna/cert/client/$Chrystoki_ClientPrivKeyFile_Name;"

export Chrystoki_ClientCertFile=`cat "$PWD/Chrystoki.conf" | grep "ClientCertFile =" |
sed 's/^ *//g' | sed 's/ *$//g'`
export Chrystoki_ClientCertFile_Name=`cat "$PWD/Chrystoki.conf" | grep "ClientCertFile
=" | sed 's/^ *//g' | sed 's/ *$//g' | awk 'BEGIN{FS="/"}{print $NF}' | sed 's/;//g'`
export New_Chrystoki_ClientCertFile="ClientCertFile =
/var/usrlocal/luna/cert/client/$Chrystoki_ClientCertFile_Name;"

export Chrystoki_ServerCAFile=`cat "$PWD/Chrystoki.conf" | grep "ServerCAFile =" | sed
's/^ *//g' | sed 's/ *$//g'`
export New_Chrystoki_ServerCAFile="ServerCAFile =
/var/usrlocal/luna/cert/server/CAFile.pem;"

export Chrystoki_ToolsDir=`cat "$PWD/Chrystoki.conf" | grep "ToolsDir =" | sed 's/^
*//g' | sed 's/ *$//g'`
export New_Chrystoki_ToolsDir="ToolsDir = /var/usrlocal/luna/bin/64;"

export Chrystoki_PartitionPolicyTemplatePath=`cat "$PWD/Chrystoki.conf" | grep
"PartitionPolicyTemplatePath =" | sed 's/^ *//g' | sed 's/ *$//g'`
export New_Chrystoki_PartitionPolicyTemplatePath="PartitionPolicyTemplatePath =
/var/usrlocal/luna/ppt/partition_policy_templates;"

export Chrystoki_ClientTokenLib=`cat "$PWD/Chrystoki.conf" | grep "ClientTokenLib =" |
sed 's/^ *//g' | sed 's/ *$//g'`
export New_Chrystoki_ClientTokenLib="ServerCAFile =
/var/usrlocal/luna/libs/64/libSoftToken.so;"

export Chrystoki_SoftTokenDir=`cat "$PWD/Chrystoki.conf" | grep "SoftTokenDir =" | sed
's/^ *//g' | sed 's/ *$//g'`
export New_Chrystoki_SoftTokenDir="SoftTokenDir = /var/usrlocal/luna/stc/token;"

export Chrystoki_ClientIdentitiesDir=`cat "$PWD/Chrystoki.conf" | grep
"ClientIdentitiesDir =" | sed 's/^ *//g' | sed 's/ *$//g'`
export New_Chrystoki_ClientIdentitiesDir="ClientIdentitiesDir =
/var/usrlocal/luna/stc/client_identities;"

export Chrystoki_PartitionIdentitiesDir=`cat "$PWD/Chrystoki.conf" | grep
"PartitionIdentitiesDir =" | sed 's/^ *//g' | sed 's/ *$//g'`
export New_Chrystoki_PartitionIdentitiesDir="PartitionIdentitiesDir =
/var/usrlocal/luna/stc/partition_identities;"

printf "\nCREATING LUNACLIENT CONFIG FILE FOR POD/CONTAINER FOR K8S/OPENSIFT
ENVIRONMENT ... \n"
sed -i "s|${Chrystoki_LibUNIX}|${New_Chrystoki_LibUNIX}|g" "$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_LibUNIX64}|${New_Chrystoki_LibUNIX64}|g" "$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_SSLConfigFile}|${New_Chrystoki_SSLConfigFile}|g"
"$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_ClientPrivKeyFile}|${New_Chrystoki_ClientPrivKeyFile}|g"
"$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_ClientCertFile}|${New_Chrystoki_ClientCertFile}|g"
"$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_ServerCAFile}|${New_Chrystoki_ServerCAFile}|g" "$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_ToolsDir}|${New_Chrystoki_ToolsDir}|g" "$PWD/Chrystoki.conf"

```

```

sed -i
"s|${Chrystoki_PartitionPolicyTemplatePath}|${New_Chrystoki_PartitionPolicyTemplatePath}|g
" "$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_ClientTokenLib}|${New_Chrystoki_ClientTokenLib}|g"
"$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_SoftTokenDir}|${New_Chrystoki_SoftTokenDir}|g" "$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_ClientIdentitiesDir}|${New_Chrystoki_ClientIdentitiesDir}|g"
"$PWD/Chrystoki.conf"
sed -i "s|${Chrystoki_PartitionIdentitiesDir}|${New_Chrystoki_PartitionIdentitiesDir}|g"
"$PWD/Chrystoki.conf"

printf "\nCREATING SECRET FOR LUNACLIENT CONFIGURATION FILE [Chrystoki.conf] ... \n"
kubectl create secret generic operator-luna-config-secret --from-
file="$PWD/Chrystoki.conf" --namespace=openshift-operators

```

7. Run the shell script. This will create additional secrets and Luna configuration file required for the container.

NOTE: If your Luna partition is configured on a client/node other than the cluster node on which you have created this script, then you need to copy **Luna configuration file Chrystoki.conf**, **node's/client's certificate** and **private key file** and **server CA file CAFile.pem** from the other client/node to the current node, at the location where you have created this script.

Nodes > Node details

N vsphere12rhos-9fmg9-worker-h6mjb ✔ Ready

Overview Details YAML Pods Logs Events Terminal

Connecting to container-00

To use host binaries, run `chroot /host`

```

sh-4.4# chroot /host
sh-4.4#
sh-4.4# cd /root
sh-4.4#
sh-4.4# vi OperatorPreReqs.sh
sh-4.4#
sh-4.4# chmod +x OperatorPreReqs.sh
sh-4.4#
sh-4.4# ls -l
total 16
-rwxrwxrwx. 1 root root 13071 Jan  6 12:46 OperatorPreReqs.sh
sh-4.4#
sh-4.4# ./OperatorPreReqs.sh

CREATING SECRET FOR CLIENT'S PRIVATE KEY AND CERTIFICATE ...
secret/operator-luna-client-secret created

CREATING SECRET FOR SERVER CA FILE ...
secret/operator-luna-cafile-secret created

CREATING LUNACLIENT CONFIG FILE FOR POD/CONTAINER FOR K8S/OPENSIFT ENVIRONMENT ...

CREATING SECRET FOR LUNACLIENT CONFIGURATION FILE [Chrystoki.conf] ...
secret/operator-luna-config-secret created

sh-4.4#

```

Integrating RedHat OpenShift Luna HSM Operator with Luna HSM

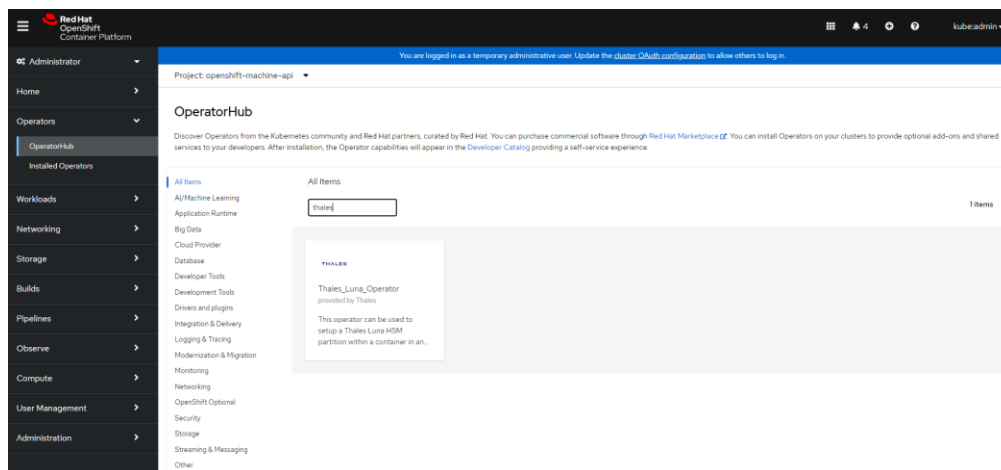
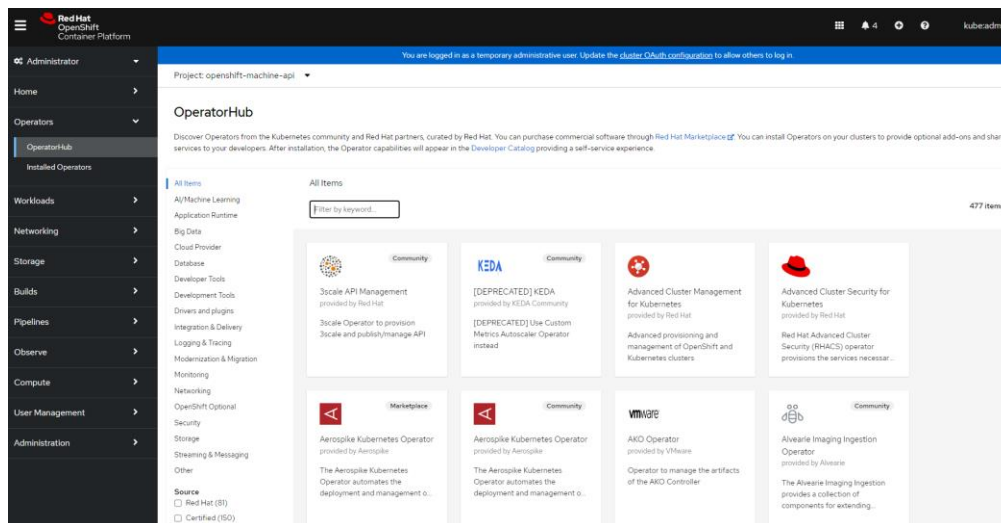
To integrate Luna HSM Operator with Luna HSM using RedHat OpenShift Container Platform, complete the following tasks:

- > [Install Thales_Luna_Operator in OpenShift Container Platform](#)
- > [Create Instance and set up a Luna partition using Thales_Luna_Operator](#)
- > [Verify the accessibility of Luna partition](#)

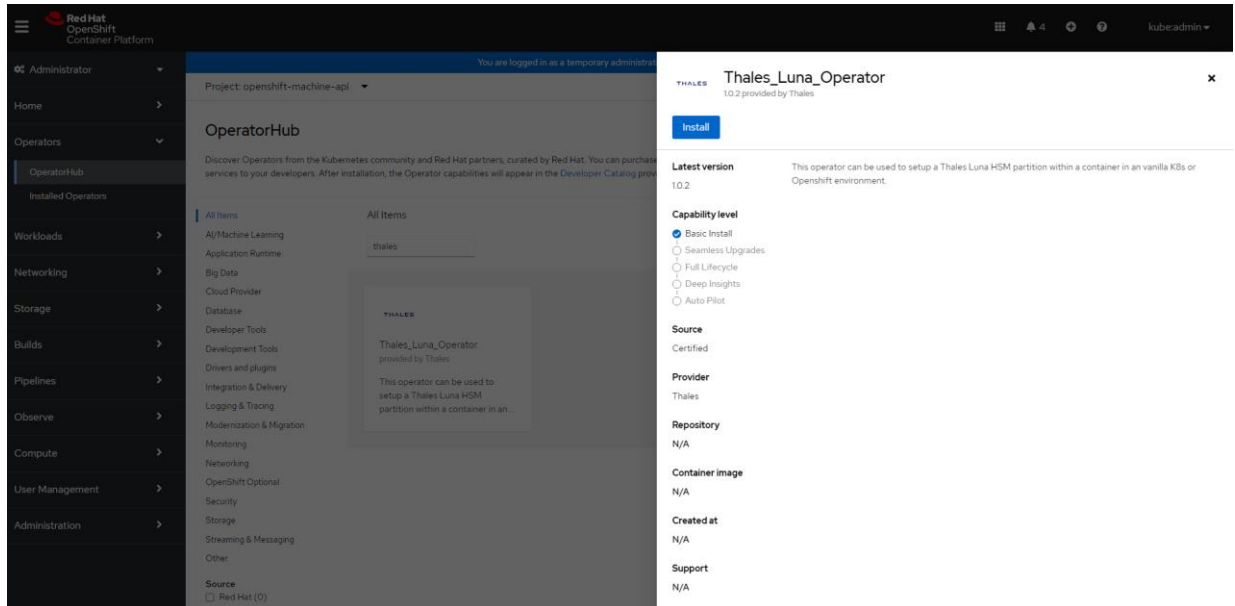
Install Thales_Luna_Operator in OpenShift Container Platform

Follow these steps to install the Thales_Luna_Operator in OpenShift Container Platform environment:

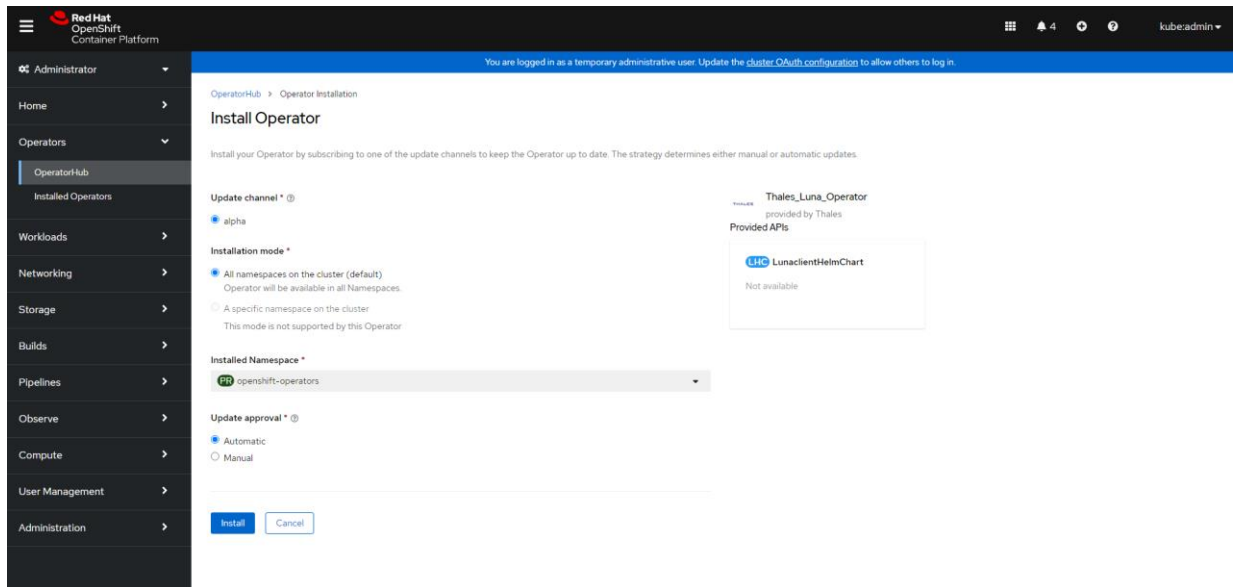
1. From **Administrator** panel, go to **perators-> OperatorHub**. Enter the keyword **thales** in the search box.



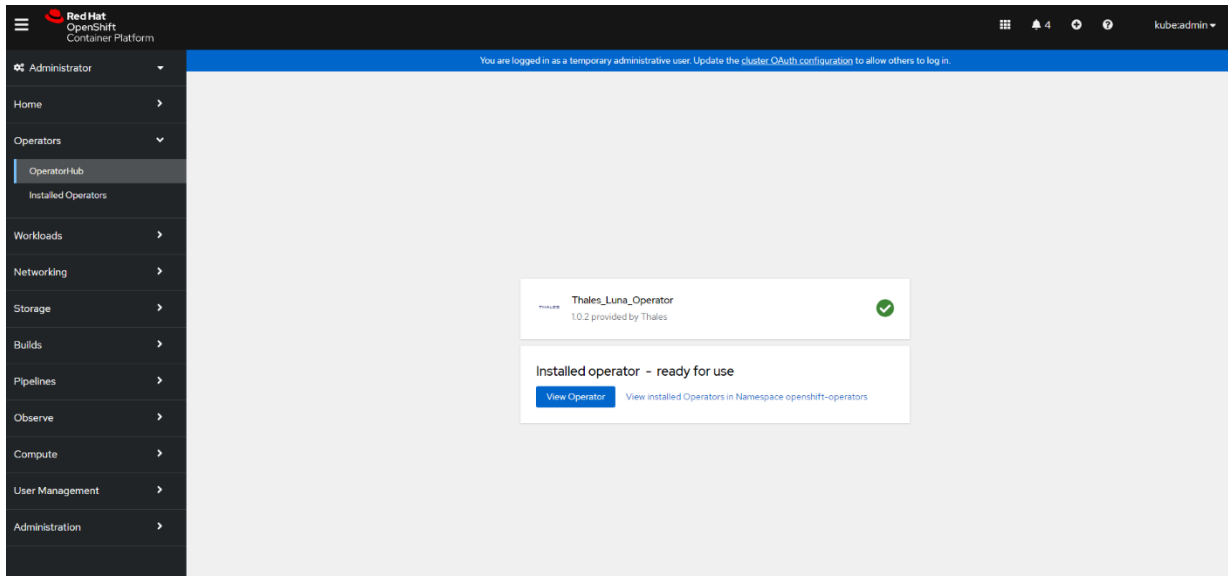
- Click the tile **Thales_Luna_Operator**, and then click **Install**.



- A new installation page will open. Click again on **Install**.



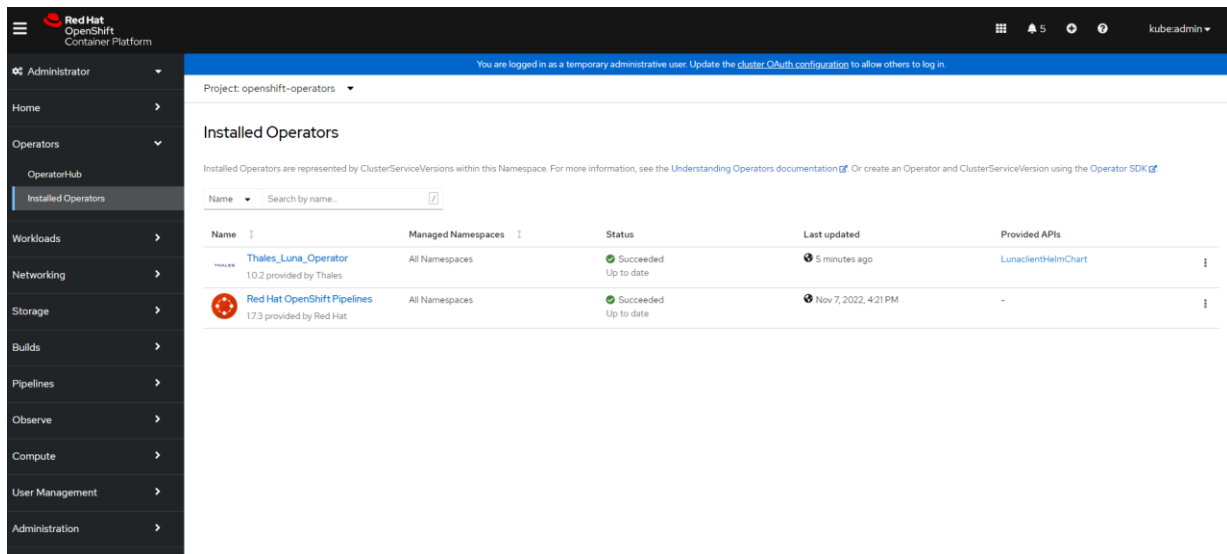
- Click **View Operator** when you see that the operator is installed and is ready for use.



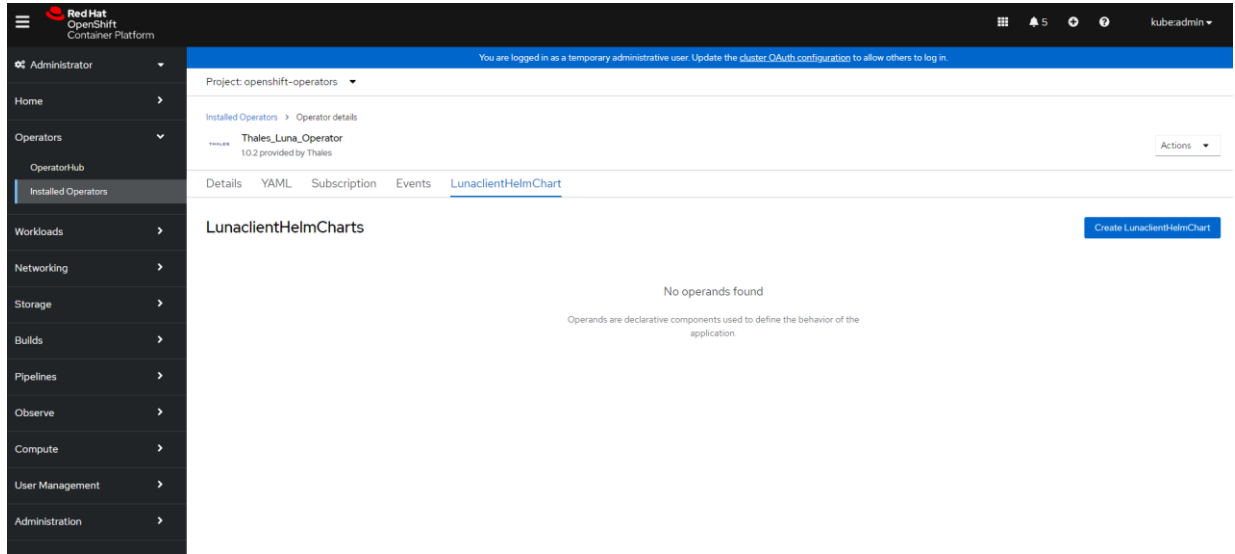
Create instance and set up Luna partition using Thales_Luna_Operator

To create an instance and set up a Luna partition:

- In the **Administrator** panel, go to **Operators** -> **Installed Operators**. You'll see **Thales_Luna_Operator** listed on the **Installed Operators** page.



- Click **Thales_Luna_Operator** to open the **Operator detail** page. Go to the **LunaclientHelmChart** tab.



- Click **Create LunaclientHelmChart**, fill out the required details such as **Name** and **Labels**, and then click **Create**.

Project: openshift-operators

Thales_Luna_Operator > Create LunaclientHelmChart

Create LunaclientHelmChart

Create by completing the form. Default values may be provided by the Operator authors.

Configure via: Form view YAML view

Note: Some fields may not be represented in this form. Please select "YAML View" for full control of object creation.

Name *

Labels

Project: openshift-operators

Installed Operators > Operator details

Thales_Luna_Operator
1.0.2 provided by Thales

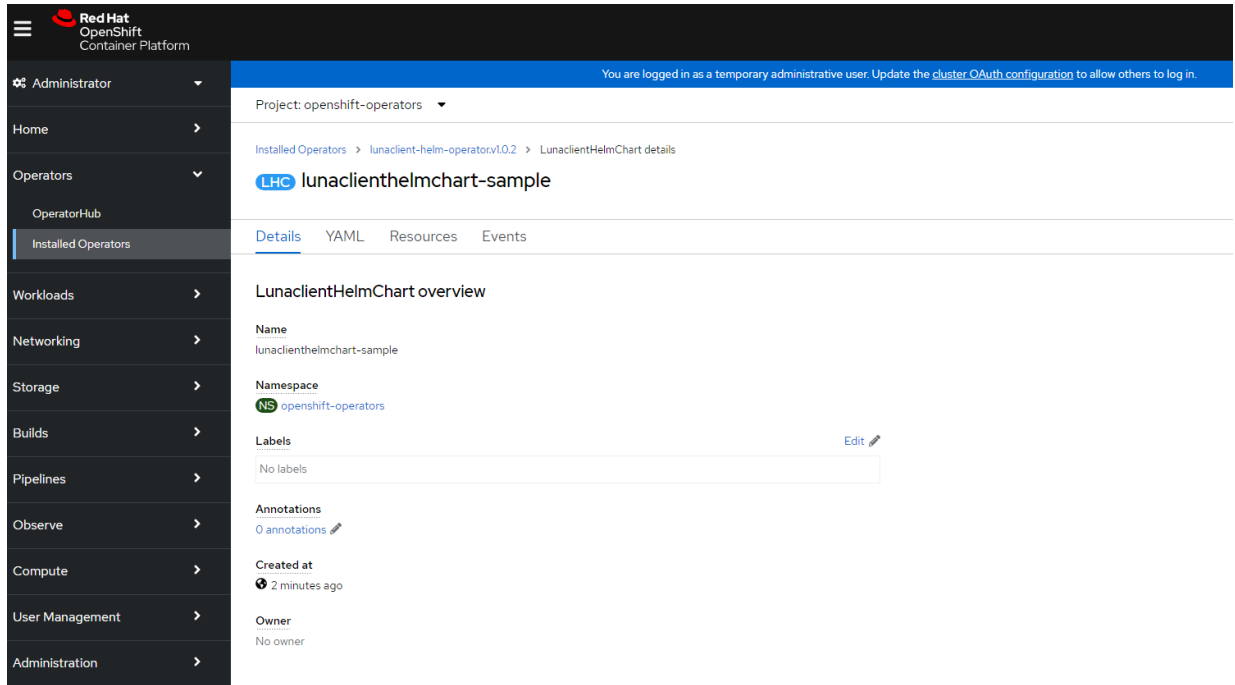
Details | YAML | Subscription | Events | LunaclientHelmChart

LunaclientHelmCharts

Name

Name	Kind	Status	Labels	Last updated
LHC lunaclienthelmchart-sample	LunaclientHelmChart	Conditions: Initialized, Deployed	No labels	Just now

4. Wait for the **Status** to become **Initialized, Deployed**. Click **lunaclienthelmchart-sample** link and go to the **Resources** tab.



Project: openshift-operators

Installed Operators > lunaclient-helm-operatorv1.0.2 > LunaclientHelmChart details

LHC lunaclienthelmchart-sample

Details YAML Resources Events

LunaclientHelmChart overview

Name
lunaclienthelmchart-sample

Namespace
NS openshift-operators

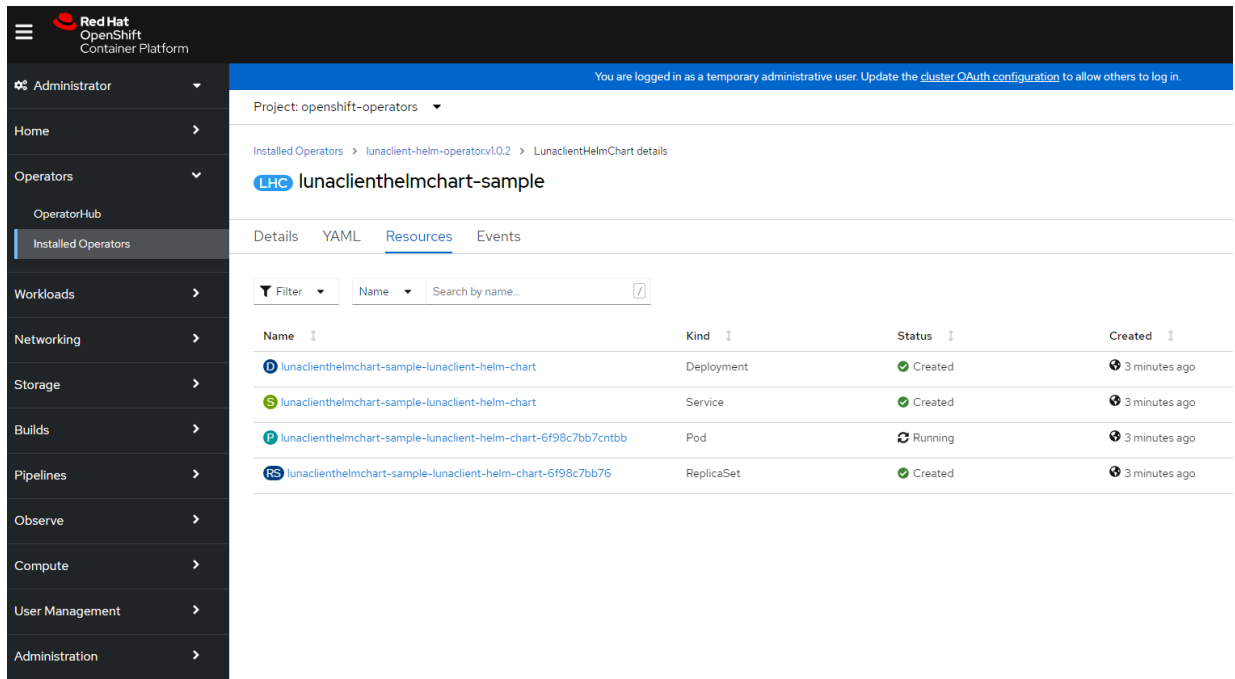
Labels [Edit](#)

No labels

Annotations
0 annotations [+](#)

Created at
2 minutes ago

Owner
No owner



Project: openshift-operators

Installed Operators > lunaclient-helm-operatorv1.0.2 > LunaclientHelmChart details

LHC lunaclienthelmchart-sample

Details YAML Resources Events

Filter Name Search by name...

Name	Kind	Status	Created
lunaclienthelmchart-sample-lunaclient-helm-chart	Deployment	Created	3 minutes ago
lunaclienthelmchart-sample-lunaclient-helm-chart	Service	Created	3 minutes ago
lunaclienthelmchart-sample-lunaclient-helm-chart-6f98c7bb7cntbb	Pod	Running	3 minutes ago
lunaclienthelmchart-sample-lunaclient-helm-chart-6f98c7bb76	ReplicaSet	Created	3 minutes ago

5. Validate that the status for Kind – Deployment, Kind – Service and Kind – ReplicaSet is **Created** and the status for the Kind – Pod is **Running**.

Verify the accessibility of Luna partition

To verify whether the Luna partition can be accessed by the new Pod:

1. Click the resource with **Kind – Pod**. Here the resource name is **lunaclienthelmchart-sample-lunaclient-helm-chart-6f98c7bb7cntbb**.

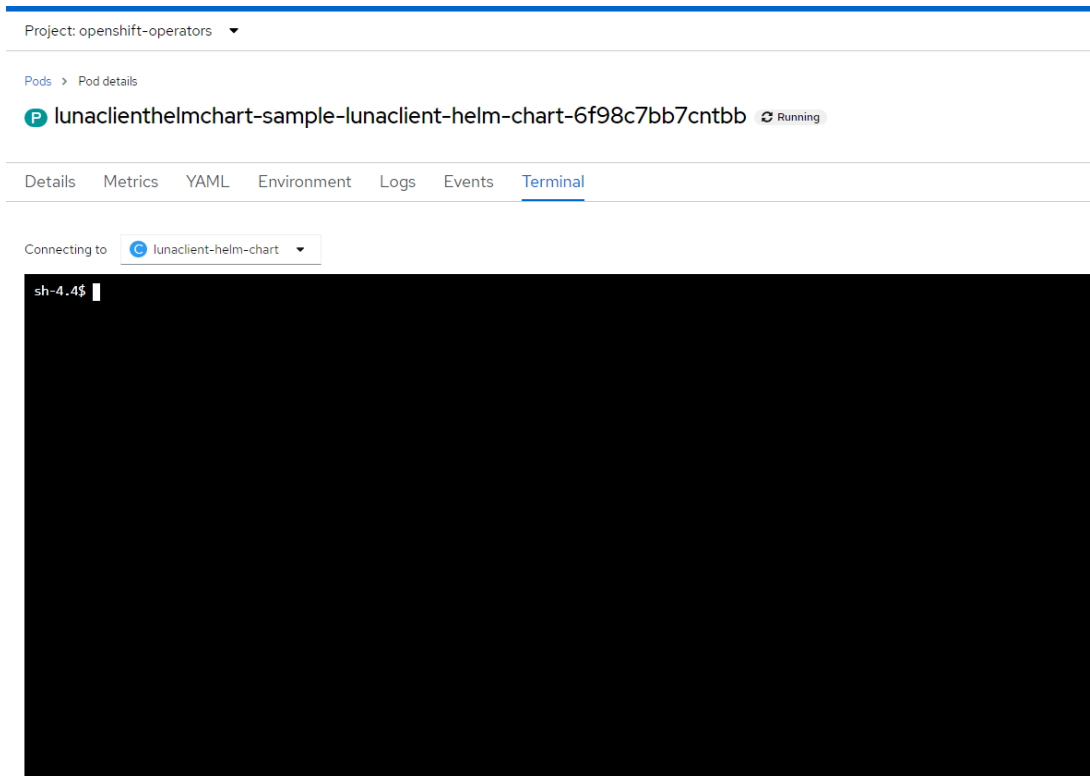
The screenshot shows the OpenShift console interface. On the left is a navigation sidebar with categories like Administrator, Home, Operators, Workloads, etc. The main content area shows the 'Resources' tab for the 'lunaclienthelmchart-sample' operator. A table lists the following resources:

Name	Kind	Status	Created
lunaclienthelmchart-sample-lunaclient-helm-chart	Deployment	Created	3 minutes ago
lunaclienthelmchart-sample-lunaclient-helm-chart	Service	Created	3 minutes ago
lunaclienthelmchart-sample-lunaclient-helm-chart-6f98c7bb7cntbb	Pod	Running	3 minutes ago
lunaclienthelmchart-sample-lunaclient-helm-chart-6f98c7bb76	ReplicaSet	Created	3 minutes ago

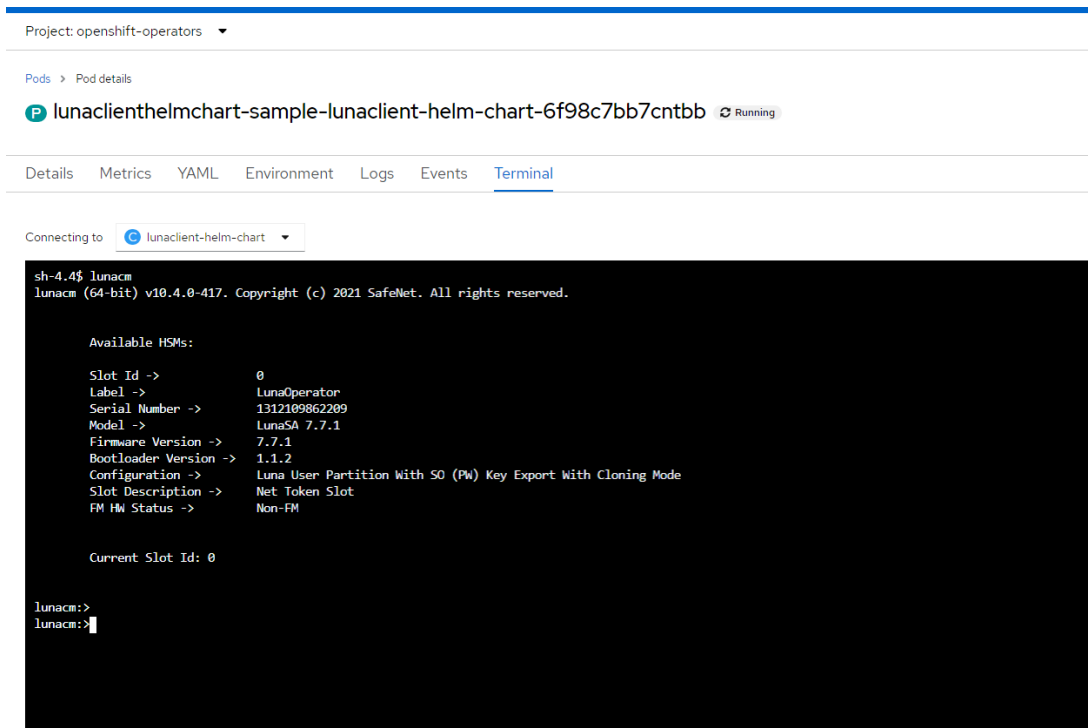
2. Go to the **Terminal** tab on the **Pod details** page.

The screenshot shows the 'Pod details' page for the pod 'lunaclienthelmchart-sample-lunaclient-helm-chart-6f98c7bb7cntbb'. The pod is in a 'Running' state. The 'Terminal' tab is selected. The page displays various details about the pod:

- Name:** lunaclienthelmchart-sample-lunaclient-helm-chart-6f98c7bb7cntbb
- Status:** Running
- Namespace:** openshift-operators
- Restart policy:** Always restart
- Labels:** app.kubernetes.io/instance=lunaclienthelmchart-sample, app.kubernetes.io/name=lunaclient-helm-chart, pod-template-hash=6f98c7bb76
- Active deadline seconds:** Not configured
- Pod IP:** 10.131.1164
- Node selector:** No selector
- Node:** vsphere12rhos-9fmg9-worker-lhkwp
- Tolerations:** 2 tolerations
- Annotations:** 3 annotations
- Created at:** 9 minutes ago
- Owner:** lunaclienthelmchart-sample-lunaclient-helm-chart-6f98c7bb76



3. Check and verify your Luna partition by running the **lunacm** utility.



This completes the integration of RedHat OpenShift Luna HSM Operator with Thales Luna HSM.

Contacting Customer Support

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