Linux Unified Key Setup (LUKS)

Integration Guide

CryptoServer HSM

SecurityServer 4.50.0.2

utimaco®

Imprint

Copyright 2026	Utimaco IS GmbH
	Krefelder Straße 220
	52070 Aachen
	Germany
Phone	AMERICAS +1-844-UTIMACO (+1 844-884-6226)
	EMEA +49 800-627-3081
	APAC +81 800-919-1301
Internet	https://support.hsm.utimaco.com/
e-mail	support@utimaco.com
Document Version	1.0.0
Date	2026-07-21
Status	PUBLISHED
Document No.	IG-2026-0078

Table of Contents

1 Introduction

This guide is part of the information and support provided by Utimaco. Additional documentation produced to support your Utimaco SecurityServer product can be found in the document directory of the Utimaco SecurityServer product bundle. All Utimaco SecurityServer product documentation is available from Utimaco's website at *https://utimaco.com/*.

1.1 About This Guide

This guide describes how to enable HSM integration with LUKS. Utimaco HSM securely stores the Keys and Certificates used by LUKS to encrypt the volume.

1.2 Target Audience for This Guide

This guide is intended for LUKS and HSM administrators.

1.3 Document Conventions

The following conventions are used in this guide:

Convention	Use	Example
Bold	Items of the Graphical User Interface (GUI), e.g., menu options	Select **Details** and click on **Properties** button
`Monospaced`	Code that is given for explanation or as an example, file paths	`certreq.exe -new request.inf IISCertRequest.csr`
Italic	References and important terms	Operating system listed in *Tested Versions*

Table 1: Document conventions

We use special icons to highlight the most important notes and information.

Here you will find important safety information that should be followed.

Here you will find additional notes or supplementary information.

This message marks the result expected after the successful execution of an instruction.

1.4 Abbreviations

The following abbreviations are used in this guide:

Abbreviation	Meaning
CD	Compact Disc
CSADM	CryptoServer Command-line Administration Tool
GUI	Graphical User Interface
HSM	Hardware Security Module
IP	Internet Protocol
LAN	Local Area Network
MBK	Master Backup Key
PCIe	PCI Express Interface

Abbreviation	Meaning
PIN	Personal Identification number
PKCS#11	Public-Key Cryptography Standard #11
SSL	Secure Socket Layer
SO	Security Officer
URL	Uniform Resource Locator

Table 2: List of abbreviations

2 Overview

2.1 LUKS Integration

LUKS is used to encrypt a block device. The contents of the encrypted device are arbitrary, and therefore any filesystem can be encrypted, including swap partitions. The presence of this header is a major difference between LUKS and plain dm-crypt, since the header allows multiple different passphrases to be used, with the ability to change and remove them with ease. However, if the header is lost or corrupted, the device will no longer be decryptable.

2.2 Utimaco CryptoServer HSM

CryptoServer is a hardware security module developed by Utimaco IS GmbH. CryptoServer is a physically protected specialized computer unit designed to perform sensitive cryptographic tasks and to securely manage, as well as store, cryptographic keys and data. It can be used as a universal, independent security component for heterogeneous computer systems.

3 Integration Requirements and Prerequisites

Ensure that the system environment you will be using meets the following hardware and software requirements.

This guide assumes that the user has already installed and configured required software.

3.1 Tested Versions

The integrations that have been successfully tested with the Utimaco HSM with LUKS.

Operating System	Open SSL Version	Systemd-cryptenroll Version	Cryptsetup Version	Utimaco Security Server Version	Utimaco HSM
Fedora 37 Fedora 36	Open SSL 3.0.5	systemd 250 (250-8)	cryptsetup 2.4.3	SecurityServer 4.50.0.2	CryptoServer CSeSeries/Se-Series

Table 3: List of tested versions

LUKS integration with Utimaco HSM is not supported with RHEL 9 as RHEL 9 does not have PKCS11 library (libcryptsetup-token-systemd-pkcs11.so) for cryptsetup.

3.2 Software Requirements

Software	Software Requirements
HSM Interfaces	PKCS11
OpenSSL	3.0.5
Libp11	libp11-0.4.12

Software	Software Requirements
System-cryptenroll	systemd 250 (250-8)
Cryptsetup Version	cryptsetup 2.4.3

Table 4: List of software requirements

3.3 Hardware Requirements

Hardware	Hardware Requirements
Utimaco LAN HSM	CryptoServer CSe-Series/Se-Series LAN with firmware SecurityServer 4.50.0.2 or higher
Utimaco PCI-e HSM	CryptoServer CSe-Series/Se-Series PCI-e with firmware SecurityServer 4.50.0.2 or higher

Table 5: List of hardware requirements

Set up an account on the Utimaco support portal and request download access at the following URL https://support.hsm.utimaco.com/ .

3.4 Prerequisites

Before you begin, please ensure that:

- The CryptoServer is set up and configured. Refer to the CryptoServer documentation to set up the HSM.
- The MBK has been created and stored onto each HSM. Refer to the CryptoServer documentation to set up the MBK.
- The CryptoServer Default Admin has been replaced with a new admin user.
- The operating system is listed in *Tested Versions*.

- The SecurityServer is listed in *Tested Version*.
- There is an admin user set up as it is required for installing software.

4 Installing and Configuring Utimaco SecurityServer Software

4.1 Download and Install Utimaco Software

If you have not already done so, please create and request an Utimaco Support Portal Account. This will allow you to download the software components needed for this installation.

1. Copy the downloaded software at the appropriate location on the Luks Server.
2. Create a utimaco folder under the `/opt` directory and further create 2 directories `/opt/utimaco/bin` and `/opt/utimaco/lib`.

›_ Console

```
# mkdir -p /opt/utimaco/bin

# mkdir -p /opt/utimaco/lib
```

3. Copy the pkcs11 library file `libcs_pkcs11_R3.so` from Utimaco CryptoServer software to the `/opt/utimaco/lib` directory.

›_ Console

```
# cp ~/path_to_application_folder/lib/libcs_pkcs11_R3.so /opt/utimaco/lib
```

4. Copy the csadm and p11tool2 files from Utimaco CryptoServer software to `/opt/utimaco/bin` directory and make both the files executable.

›_ Console

```
# cd ~/path_to_application_folder

# cp csadm p11tool2 /opt/utimaco/bin

# chmod +x /opt/utimaco/bin/csadm /opt/utimaco/bin/p11tool2
```

4.2 CryptoServer PKCS#11 Configuration

1. Create the directory `/etc/utimaco`. Locate the Utimaco PKCS#11 configuration file in your SecurityServer directory, `Linux/x86-64/Crypto_APIS/PKCS11_R3/sample`. Copy the Utimaco PKCS#11 configuration file `cs_pkcs11_R3.cfg` into `/etc/utimaco` directory.

›_ Console

```
# mkdir /etc/utimaco

# cd <install directory>/Software/Linux/x86-

64/Crypto_APIs/PKCS11_R3/sample

# cp cs_pkcs11_R3.cfg /etc/utimaco

# cd /etc/utimaco
```

2. Edit the `cs_pkcs11_R3.cfg` file and make the appropriate changes to the file.

cs_pkcs11_R3.cfg

```
[Global]
# For unix:
Logpath = /tmp
# Loglevel (0 = NONE; 1 = ERROR; 2 = WARNING; 3 = INFO; 4 = TRACE)
Logging = 1
Keepalive = true
# Set the Device to connect with
[CryptoServer]
# Device specifier
Device = <HSM_IP>
```

For more information regarding the commands and command parameters please check the Utimaco CryptoServer documentation. The device may be a CryptoServer (PCIe or LAN) device. The device line will follow one of these patterns, based on the HSM form-factor: Device = 288@<HSM IP address> Hardware (LAN) HSM

OR

Device = /dev/cs2.0 Hardware (PCIe) HSM

To make your testing easier, it would be good to enable the PKCS#11 log file. That can be enabled by editing the **Logging Loglevel**. Set the **LogPath** and **Logging Loglevel** to 1. For testing you may want to increase it to 4.

The added **LogPath** points to a writable directory, not to a file.

If you encounter problems, check the log file named `cs_pkcs11_R3.log` in the **LogPath** defined directory. When you are done testing, you should change Logging to 1 or 2. This will limit the logging to only critical and important messages.

4.3 Create SO User and Initialize a Slot

You must initialize a slot with a custom label using p11tool2.

First, using p11tool2 create the SO or Security Officer and then using the p11tool2 command initialize the slot that you want to use, and the slot user as shown below.

›_ Console

```
# ./p11tool2 slot=<slot_no> Label=<token_label> Login=ADMIN,ADMIN.key
InitToken=<SO_PIN>

# ./p11tool2 slot=<slot_no> LoginSO=<SO_PIN> InitPin=<CryptoUser_PIN>
```

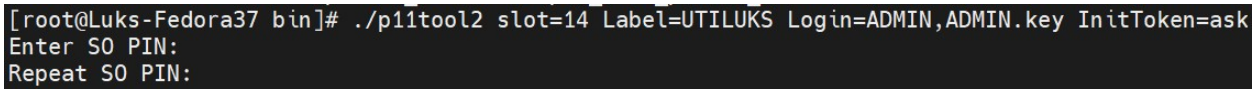

Figure 1 : Slot initialization output

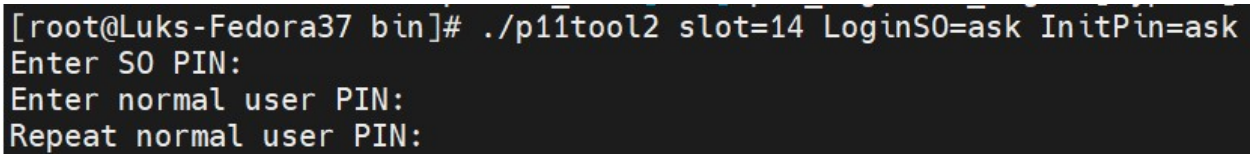

Figure 2 : Slot initialization output

5 Installing PKCS11 Engine

5.1 Installing Libp11and Opensc

Use the steps below to install Libp11.

1. Download the latest libp11 package from Releases·

›_ Console

```
# dnf install libp11-devel
```

2. Download the latest Opensc package from Releases.

›_ Console

```
# dnf install opensc gnutls-utils
```

If you want to install the OpenSSL and Libp11 from source code, then follow OpenSSL Integration Guide with Utimaco HSM.

5.2 Setting up Utimaco CryptoServer Library in OpenSSL Configuration File

1. Open the file `/etc/pki/tls/openssl.cnf` and enter the following line in the first line of the file.

›_ Console

```
# openssl_conf=openssl_init
```

2. Enter the following lines under last section of `openssl.cnf` file.

›_ Console

```
# [openssl_init] engines=engine_section


[engine_section] pkcs11 = pkcs11_section


[pkcs11_section] engine_id = pkcs11

dynamic_path = /usr/lib64/engines-3/pkcs11.so MODULE_PATH = /opt/utimaco/lib/
libcs_pkcs11_R3.so init = 0
```

5.3 Verify PKCS#11 Engine

Run the command below to verify if the OpenSSL Engine is available or not.

›_ Console

```
# openssl engine pkcs11 -t
```

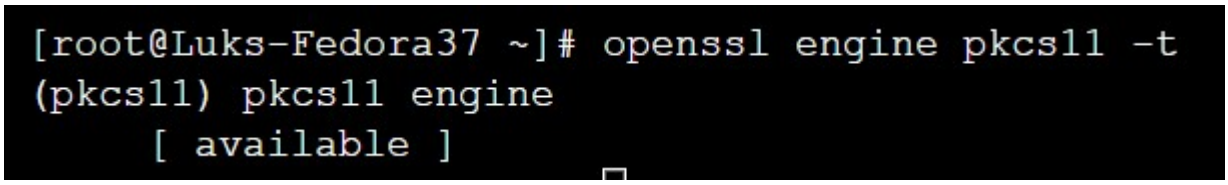

Figure 3 : Openssl pkcs11 engine status

6 Integrating LUKS with Utimaco HSM

6.1 Generating the encryption Key and Certificate on Utimaco HSM

1. Generate the RSA key using p11tool2.

›_ Console

```
# p11tool2 Slot=14 LoginUser=ask

PubKeyAttr=CKA_LABEL="LUKSPublicKey",CKA_ID=0x45

PrvKeyAttr=CKA_LABEL="LUKSPrivateKey",CKA_ID=0x45 GenerateKeyPair=RSA
```

Figure 4 : Key generation output

2. Verify that the keys are generated onto the HSM using following command.

›_ Console

```
# p11tool2 slot=<no> LoginUser=ask ListObjects
```

Example

```
[root@Luks-Fedora37 bin]# ./p11tool2 slot=14 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_PUBLIC_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 0C537604-AE6F-4B8E-A74D-616A91DAA189
  CKA_LABEL                     = LUKSPublicKey
  CKA_ID                        = 0x45 (E)
+ 2.4
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 3FC70BF4-91A0-4216-9998-D5480BD2C433
  CKA_LABEL                     = LUKSPrivateKey
  CKA_ID                        = 0x45 (E)
  CKA_SUBJECT                   =
               0x301B3119 30170603 5504030C 10746573 |0 1 0   U    tes
                 742E7574 696D6163 6F2E636F 6D       |t.utimaco.com
```

Figure 5 : Key list using p11tool2

3. Generate a self-signed certificate.

›_ Console

```
# openssl req -x509 -engine pkcs11 -keyform engine -new -key

"pkcs11:token=UTILUKS;object=LUKSPrivateKey" -sha256 -out lukscert.pem subj "/
CN=test.utimaco.com"
```

Here LUKS is the token label and LUKSPrivateKey is the key on the HSM. Provide Cryptouser PIN when prompted.

```
[root@Luks-Fedora37 bin]# openssl req -x509 -engine pkcs11 -keyform engine -new -key "pkcs11:token=UTILUKS;object=LUKSPrivateKey" -
sha256 -out lukscert.pem -subj "/CN=test.utimaco.com"
Engine "pkcs11" set.
Enter PKCS#11 token PIN for UTILUKS:
```

Figure 6 : Self sign certificate generation

It is recommended to use CA signed certificate for production environment.

4. Convert `lukscert.pem` to `lukscert.der` format.

›_ Console

```
# openssl x509 -outform der -in lukscert.pem -out lukscert.der
```

```
[root@Luks-Fedora37 bin]# openssl x509 -outform der -in lukscert.pem -out lukscert.der
```

Figure 7 : Certificate type change output

5. Import the certificate.

›_ Console

```
# p11tool2 slot=14 LoginUser=ask

CertAttr=CKA_LABEL="LUKSCERT",CKA_ID=0x45

PubKeyAttr=CKA_LABEL="LUKSPrivateKey",CKA_ID=0x45

ImportCert=/opt/utimaco/bin/lukscert.der
```

```
[root@Luks-Fedora37 bin]# ./p11tool2 slot=14 LoginUser=ask CertAttr=CKA_LABEL="LUKSCERT",CKA_ID=0x45 PubKeyAttr=CKA_LABEL="LUKSPriv
ateKey",CKA_ID=0x45 ImportCert=/opt/utimaco/bin/lukscert.der
Enter normal user PIN:
```

Figure 8 : Certificate import using p11tool2

6. Check the certificate on HSM.

›_ Console

```
# p11tool2 slot=<no> LoginUser=ask ListObjects
```

```
[root@Luks-Fedora37 bin]# ./p11tool2 slot=14 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_UNIQUE_ID                 = D91408D2-5EDB-4880-9B58-9FB674BFC841
  CKA_LABEL                     = LUKSCERT
  CKA_ID                        = 0x45 (E)
  CKA_SUBJECT                   =
                0x301B3119 30170603 5504030C 10746573 |0 1 0   U    tes|
                  742E7574 696D6163 6F2E636F 6D       |t.utimaco.com   |
```

Figure 9 : Certificate list output

6.2 Create a volume for LUKS Encryption

1. Create a partition.

›_ Console

```
# fdisk /dev/sdc
```

```
[root@Luks-Fedora37 bin]# fdisk /dev/sdc

Welcome to fdisk (util-linux 2.38.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x66df9134.

Command (m for help):
```

Figure 10 : fdisk command prompt

2. Type n for new partition and p for primary partition type.

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1):
```

Figure 11 : New partition creation

3. Press ENTER to use up all the available space after your first sector or specify the size for your partition.

```
Partition number (1-4, default 1):
First sector (2048-67108863, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-67108863, default 67108863):

Created a new partition 1 of type 'Linux' and of size 32 GiB.

Command (m for help):
```

Figure 12 : Specifying size of disk

4. Type w to write the partition table.

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

Figure 13 : Saving changes to partition table

5. Verify the newly added partition using fdisk -l command.

›_ Console

```
# fdisk -l
```

```
Disk /dev/sdc: 32 GiB, 34359738368 bytes, 67108864 sectors
Disk model: Virtual Disk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: dos
Disk identifier: 0x085df4eb

Device     Boot Start      End  Sectors Size Id Type
/dev/sdc1        2048 67108863 67106816  32G 83 Linux
```

Figure 14 : Disk list

6. Format the newly added partition.

›_ Console

```
# cryptsetup -y -v luksFormat /dev/sdc1
```

```
[root@Luks-Fedora37 bin]# cryptsetup -y -v luksFormat /dev/sdc1

WARNING!
========
This will overwrite data on /dev/sdc1 irrevocably.

Are you sure? (Type 'yes' in capital letters): YES
Enter passphrase for /dev/sdc1:
Verify passphrase:
Key slot 0 created.
Command successful.
```

Figure 15 : Formatting the partition

6.3 Configuring LUKS partition to use encryption key on Utimaco HSM

1. List all the slots using the command below.

›_ Console

```
# p11tool --provider /opt/utimaco/lib/libcs_pkcs11_R3.so --list-all
```

```
[root@Luks-Fedora37 bin]# p11tool --provider /opt/utimaco/lib/libcs_pkcs11_R3.so --list-all
warning: no token URL was provided for this operation; the available tokens are:

pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0000;token=CryptoServer%20PKCS11%20Token
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0001;token=SKLMDemo
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0002;token=LUKS
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0003;token=CryptoServer%20PKCS11%20Token
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0004;token=Nginx
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0005;token=OpensslSlot
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0006;token=HAproxy
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0007;token=LUKS
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0008;token=CryptoServer%20PKCS11%20Token
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0009;token=Openshift
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0010;token=LUKS
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0011;token=PindIDSSO
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0012;token=LUKS1
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0013;token=
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0015;token=
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0016;token=
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0017;token=
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0018;token=
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0019;token=
```

Figure 16 : Slot information

2. List the objects on the slot and note down the URL of the certificate and private key.

›_ Console

```
# p11tool --provider /opt/utimaco/lib/libcs_pkcs11_R3.so --login --listall
pkcs11:token=UTILUKS
```

```
[root@Luks-Fedora37 bin]# p11tool --provider /opt/utimaco/lib/libcs_pkcs11_R3.so --login --list-all pkcs11:token=UTILUKS
Token 'UTILUKS' with URL 'pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS' requires u
ser PIN
Enter PIN:
Object 0:
        URL: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%45;object=LUKSPublic
Key;type=public
        Type: Public key (RSA-2048)
        Label: LUKSPublicKey
        Flags: CKA_WRAP/UNWRAP; CKA_PRIVATE; CKA_SENSITIVE;
        ID: 45

Object 1:
        URL: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%45;object=LUKSCERT;t
ype=cert
        Type: X.509 Certificate (RSA-2048)
        Expires: Sat Apr  8 19:22:19 2023
        Label: LUKSCERT
        Flags: CKA_PRIVATE; CKA_SENSITIVE;
        ID: 45
```

Figure 17 : Slot objects list

3. Add pkcs11 module in `/etc/pkcs11/modules/utpkcs11.module` file.

›_ utpkcs11.module File

```
module: /opt/utimaco/lib/libcs_pkcs11_R3.so
```

```
[root@Luks-Fedora37 ~]# cat /etc/pkcs11/modules/utpkcs11.module
module: /opt/utimaco/lib/libcs_pkcs11_R3.so
```

Figure 18 : Content of utpkcs11.module

4. Enroll the certificate on LUKS drive using `systemd-cryptenroll` command.

›_ Console

```
# systemd-cryptenroll --pkcs11-token-

uri="pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;
serial=SI003001_0000;token=UTILUKS;id=%45;object=LUKSCERT;type=cert" /dev/sdc1
```

```
[root@Luks-Fedora37 bin]# systemd-cryptenroll --pkcs11-token-uri="pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial
=SI003001_0000;token=UTILUKS;id=%45;object=LUKSCERT;type=cert" /dev/sdc1
🔐 Please enter current passphrase for disk /dev/sdc1: ****************
```

Figure 19 : systemd-cryptenroll command output

Provide slot PIN when prompted.

5. Verify that the pkcs11 url is added to the luks partition in the tokens section.

›_ Console

```
# cryptsetup luksDump /dev/sdc1
```

```
[root@Luks-Fedora37 bin]# cryptsetup luksDump /dev/sdc1
LUKS header information
Version:        2
Epoch:          5
Metadata area:  16384 [bytes]
Keyslots area:  16744448 [bytes]
UUID:           0a9571f8-11f5-4e35-acc6-c4c35c5390d1
Label:          (no label)
Subsystem:      (no subsystem)
Flags:          (no flags)

Data segments:
  0: crypt
        offset: 16777216 [bytes]
        length: (whole device)
        cipher: aes-xts-plain64
        sector: 4096 [bytes]

Keyslots:
  0: luks2
        Key:        512 bits
        Priority:   normal
        Cipher:     aes-xts-plain64
        Cipher key: 512 bits
        PBKDF:      argon2id
        Time cost:  4
        Memory:     813818
        Threads:    2
        Salt:       81 77 9b ad 86 fd 77 75 73 f7 3c 72 5a 93 44 9b
                    32 2f d8 5d 2b 35 7e af 73 8b 24 6f 48 86 02 16
        AF stripes: 4000
        AF hash:    sha256
        Area offset:32768 [bytes]
        Area length:258048 [bytes]
        Digest ID:  0
  1: luks2
        Key:        512 bits
```

Figure 20 : Encrypted volume information

```
  1: luks2
        Key:        512 bits
        Priority:   normal
        Cipher:     aes-xts-plain64
        Cipher key: 512 bits
        PBKDF:      pbkdf2
        Hash:       sha512
        Iterations: 1000
        Salt:       3b ce be 3f 44 b7 f7 00 5d ba 77 8a 18 a5 db c4
                    af 8f 9f b0 da 18 69 ea c9 68 e2 27 59 e7 97 98
        AF stripes: 4000
        AF hash:    sha512
        Area offset:290816 [bytes]
        Area length:258048 [bytes]
        Digest ID:  0
Tokens:
  0: systemd-pkcs11
        pkcs11-uri: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%45;object=LUKSPrivateKey;
type=cert
        pkcs11-key: 14 eb 4e 54 63 94 33 ec f6 10 95 7e 4b 12 58 06
                    ff f4 5f 8a 24 57 c0 ce e0 4b b2 10 d4 7a c6 68
                    15 1e 7c dc bd 5b 0f df 05 bd 5f a2 9b 3f db 1e
                    86 2a 8b e1 e0 b2 9b 8a 6f fc 00 c1 1a 54 12 8b
                    27 fb 3f 66 90 de 3c 88 65 a2 c7 9b 7a d1 06 22
                    9c 20 17 73 60 48 17 eb 0f 10 34 b2 3d a8 50 b0
                    d7 ac 99 b7 20 a8 d0 34 9c 6a d0 fc 54 b3 1d 25
                    6a 2c d9 bb 9c bf 05 eb 20 dc a6 25 62 e9 e8 c3
                    49 4e 1f e2 a1 3e f4 e3 7f c3 b4 cb d9 06 6e e2
                    f0 2e cf ef 3b ed 25 82 81 e8 38 23 e1 56 ec 42
                    07 47 f3 1c 80 3f c1 cc 90 30 1f 81 87 28 53 1b
                    b3 e0 d0 42 10 6b a0 61 d9 ea 46 09 57 d5 43 64
                    2f 2d 52 24 52 93 2e 93 58 12 a9 ad ac 82 df 93
                    fc 5e 4a 5e 6b 3b fb 5a ef ef b3 c4 de 51 41 c7
                    57 73 ee 28 2a 6b 56 94 c7 d7 3c c5 5d 0d 72 ba
                    8c 5d 55 0a 8f 9b fa 86 44 45 cd 32 f6 ee a5 8b
        Keyslot:    1
```

Figure 21 : Encrypted volume information

6. Open the luks volume as mybackup disk.

›_ Console

```
# cryptsetup luksOpen /dev/sdc1 mybackup
```

```
[root@Luks-Fedora37 bin]# cryptsetup luksOpen /dev/sdc1 mybackup
Enter passphrase for /dev/sdc1:
```

Figure 22 : luksopen command output

7. Export token using the command below to `header.json` file.

›_ Console

```
# cryptsetup token export --token-id 0 /dev/sdc1 > header.json
```

```
# cryptsetup token export --token-id 0 /dev/sdc1 > header.json
```

Figure 23 : Exporting token id information

8. Open `header.json` file and replace the `pkcs-uri` with type=private url.

Example

›_ header.json file

```
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS7108
43_0014;token=UTILUKS;id=%45;object=LUKSPrivateKey;type=private
```

9. Import the `header.json` file.

›_ Console

```
# cryptsetup token import --json-file header.json --token-id 1 /dev/sdc1
```

```
]# cryptsetup token import --json-file header.json --token-id 1 /dev/sdc1
```

Figure 24 : Importing token id information

10. Verify token id 1 using below command.

›_ Console

```
# cryptsetup luksDump /dev/sdc1
```

```
[root@Luks-Fedora37 bin]# cryptsetup luksDump /dev/sdc1
LUKS header information
Version:        2
Epoch:          6
Metadata area:  16384 [bytes]
Keyslots area:  16744448 [bytes]
UUID:           0a9571f8-11f5-4e35-acc6-c4c35c5390d1
Label:          (no label)
Subsystem:      (no subsystem)
Flags:          (no flags)

Data segments:
  0: crypt
        offset: 16777216 [bytes]
        length: (whole device)
        cipher: aes-xts-plain64
        sector: 4096 [bytes]

Keyslots:
  0: luks2
        Key:        512 bits
        Priority:   normal
        Cipher:     aes-xts-plain64
        Cipher key: 512 bits
        PBKDF:      argon2id
        Time cost:  4
        Memory:     813818
        Threads:    2
        Salt:       81 77 9b ad 86 fd 77 75 73 f7 3c 72 5a 93 44 9b
                    32 2f d8 5d 2b 35 7e af 73 8b 24 6f 48 86 02 16
        AF stripes: 4000
        AF hash:    sha256
        Area offset:32768 [bytes]
        Area length:258048 [bytes]
```

Figure 25 : LuksDump output

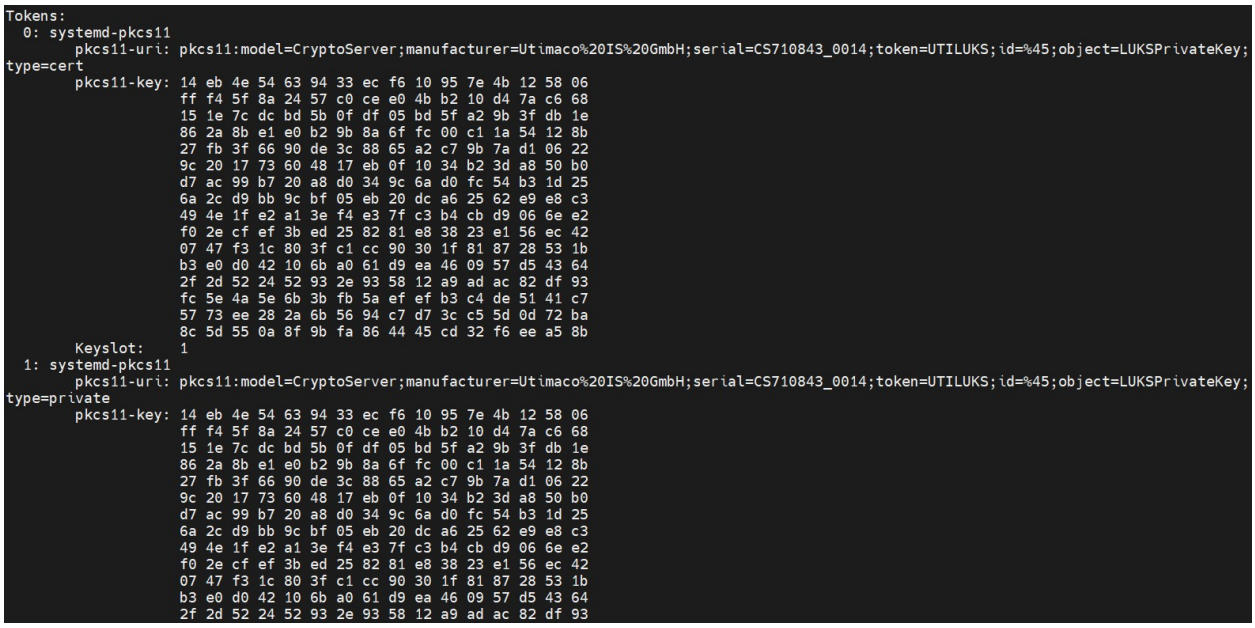

Figure 26 : LuksDump output

11. Remove token id 0 and verify with below command.

›_ Console

```
# cryptsetup token remove --token-id 0 /dev/sdc1

# cryptsetup luksDump /dev/sdc1
```

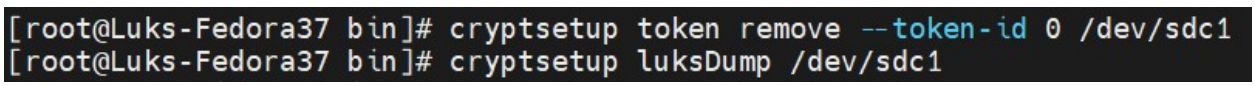

Figure 27 : Removing old token id

```
Tokens:
  1: systemd-pkcs11
        pkcs11-uri: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%45;object=LUKSPrivateKey;
type=private
        pkcs11-key: 14 eb 4e 54 63 94 33 ec f6 10 95 7e 4b 12 58 06
                    ff f4 5f 8a 24 57 c0 ce e0 4b b2 10 d4 7a c6 68
                    15 1e 7c dc bd 5b 0f df 05 bd 5f a2 9b 3f db 1e
                    86 2a 8b e1 e0 b2 9b 8a 6f fc 00 c1 1a 54 12 8b
                    27 fb 3f 66 90 de 3c 88 65 a2 c7 9b 7a d1 06 22
                    9c 20 17 73 60 48 17 eb 0f 10 34 b2 3d a8 50 b0
                    d7 ac 99 b7 20 a8 d0 34 9c 6a d0 fc 54 b3 1d 25
                    6a 2c d9 bb 9c bf 05 eb 20 dc a6 25 62 e9 e8 c3
                    49 4e 1f e2 a1 3e f4 e3 7f c3 b4 cb d9 06 6e e2
                    f0 2e cf ef 3b ed 25 82 81 e8 38 23 e1 56 ec 42
                    07 47 f3 1c 80 3f c1 cc 90 30 1f 81 87 28 53 1b
                    b3 e0 d0 42 10 6b a0 61 d9 ea 46 09 57 d5 43 64
                    2f 2d 52 24 52 93 2e 93 58 12 a9 ad ac 82 df 93
                    fc 5e 4a 5e 6b 3b fb 5a ef ef b3 c4 de 51 41 c7
                    57 73 ee 28 2a 6b 56 94 c7 d7 3c c5 5d 0d 72 ba
                    8c 5d 55 0a 8f 9b fa 86 44 45 cd 32 f6 ee a5 8b
        Keyslot:    1
```

Figure 28 : Removing old token id

12. Close the LUKS volume.

›_ Console

```
# cryptsetup luksClose mybackup
```

13. Attach the LUKS volume with `systemd-cryptsetup`.

›_ Console

```
# /usr/lib/systemd/systemd-cryptsetup attach luks /dev/sdc1 - pkcs11uri="auto"
```

```
[root@Luks-Fedora37 bin]# /usr/lib/systemd/systemd-cryptsetup attach luks /dev/sdc1 - pkcs11-uri="auto"
Set cipher aes, mode xts-plain64, key size 512 bits for device /dev/sdc1.
🔐 Please enter PIN for security token 'UTILUKS' in order to unlock Virtual_Disk (luks): ***************
Successfully logged into security token 'UTILUKS'.
Successfully decrypted key with security token.
```

Figure 29 : Attaching LUKS volume

Provide slot PIN when prompted.

6.4 Auto-Attaching and Auto-Mounting LUKS Volume During Boot

1. Open `/etc/crypttab` and add below content.

›_ crypttab file

```
mydisk UUID=0a9571f8-11f5-4e35-acc6-c4c35c5390d1 - pkcs11-uri=auto
```

2. Use the command below to start the network service before running `system-cryptsetup` service.

›_ Console

```
# grubby --update-kernel=ALL --args="rd.neednet=1"
```

3. Reboot the system.
4. There is a password prompt during the boot for attaching LUKS volume. Provide the slot PIN when prompted.

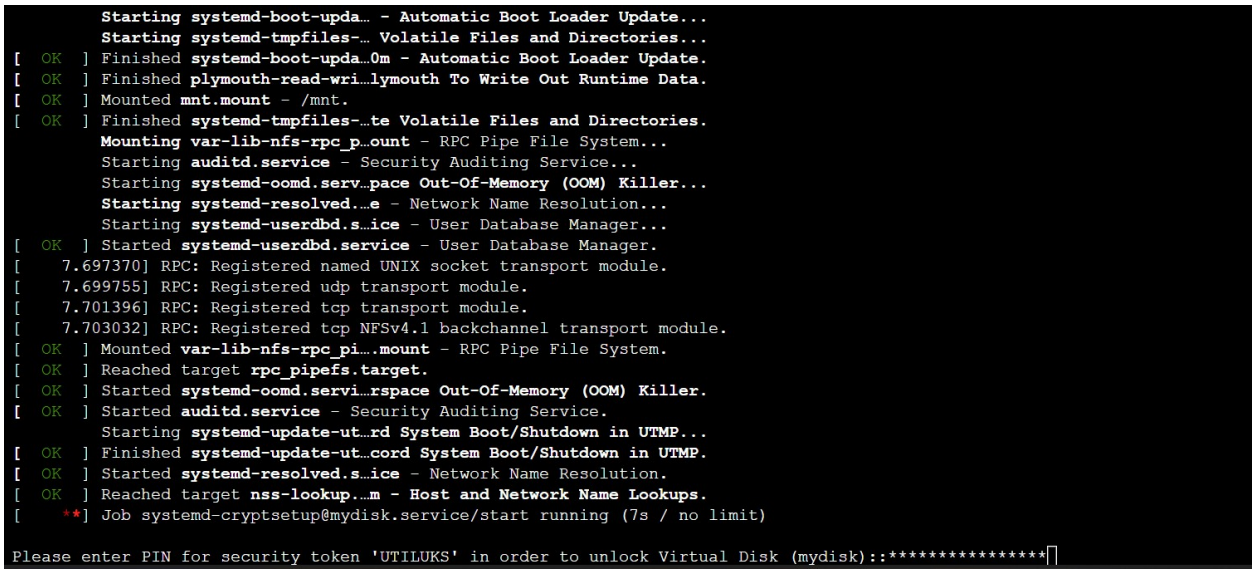

Figure 30 : Boot window

5. Once the system is up, verify that status of systemd-cryptsetup service.

›_ Console

```
# systemctl status systemd-cryptsetup@mydisk.service
```

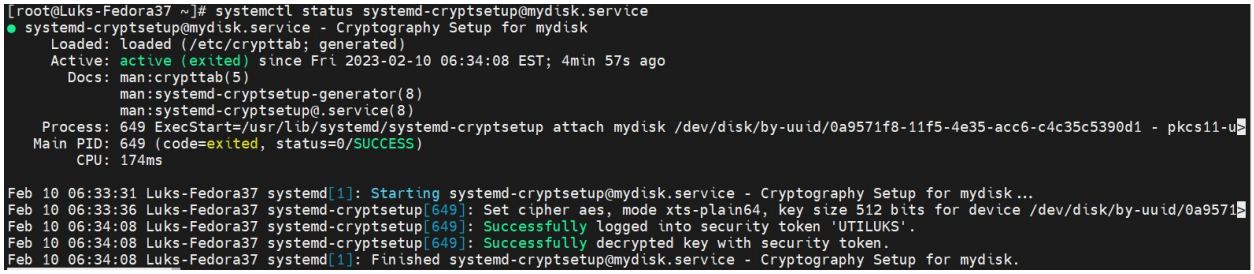

Figure 31 : systemd-cryptsetup service status

6. Format the volume with File System.

›_ Console

```
# mkfs.xfs /dev/mapper/mydisk
```

7. Create one directory for mounting the volume.

›_ Console

```
# mkdir /mnt/hdd
```

8. Get UUID of `/dev/mapper/mydisk`.

›_ Console

```
# blkid
```

9. Add below entry in `/etc/fstab`.

›_ fstab file

```
UUID=7f514a18-c3e7-4e35-b975-d68ceb624bcf   /mnt/hdd xfs    defaults        0 0
```

Here UUID is for `/dev/mapper/mydisk`.

10. Run below command to mount the encrypted volume.

›_ Console

```
# mount -a
```

```
[root@Luks-Fedora37 ~]# mount -a
mount: (hint) your fstab has been modified, but systemd still uses
       the old version; use 'systemctl daemon-reload' to reload.
```

Figure 32 : Mounting LUKS volume

If the above command does not work, troubleshoot accordingly.

11. Next, reboot the system and check if the reboot halts waiting for LUKS passphrase to mount the encrypted device.

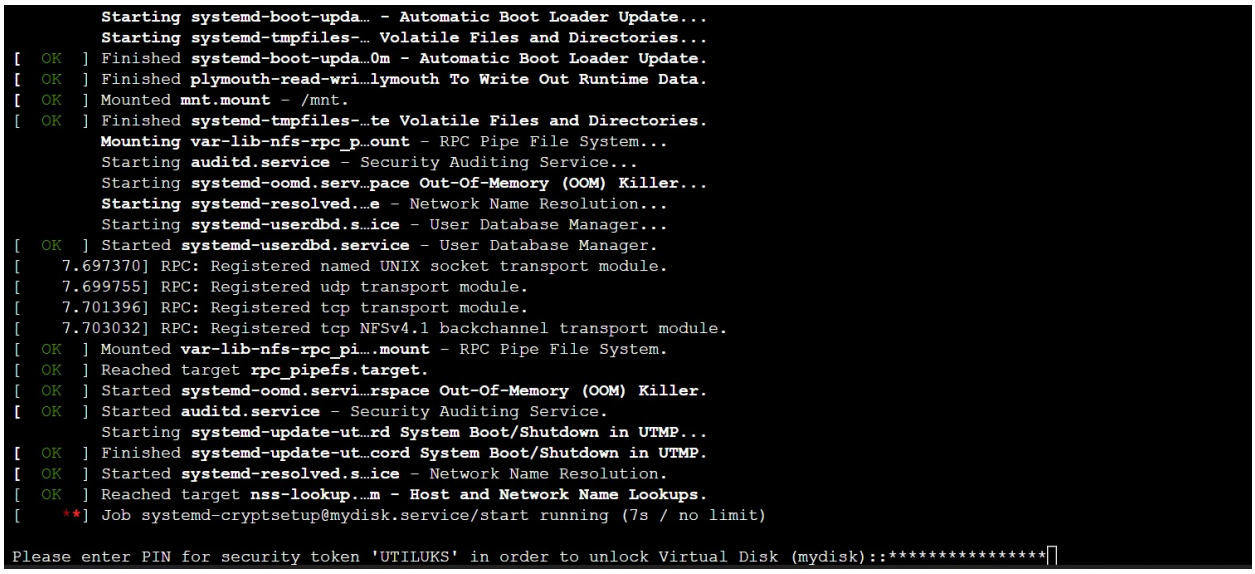

Figure 33 : Boot window

Provide slot PIN when prompted.

12. When the system is up and running, verify that the encrypted volume is mounted automatically.

›_ Console

```
# df -Th
```

```
[root@Luks-Fedora37 ~]# df -Th
Filesystem          Type      Size  Used Avail Use% Mounted on
devtmpfs            devtmpfs  4.0M     0  4.0M   0% /dev
tmpfs               tmpfs     3.9G     0  3.9G   0% /dev/shm
tmpfs               tmpfs     1.6G  1.1M  1.6G   1% /run
/dev/sda1           ext4      8.8G  2.1G  6.3G  26% /
tmpfs               tmpfs     3.9G     0  3.9G   0% /tmp
/dev/sda2           ext4      974M  246M  661M  28% /boot
/dev/sdc1           fuseblk    16G   45M   16G   1% /mnt
/dev/mapper/mydisk  xfs        32G  261M   32G   1% /mnt/hdd
tmpfs               tmpfs     795M     0  795M   0% /run/user/1000
[root@Luks-Fedora37 ~]#
```

Figure 34 : Listing mounted volumes

6.5 Rekeying LUKS volume

It is recommended to take backup of your data to some external storage before performing rekey operation.

1. Unmount the mounted volume from the mountpoint.

›_ Console

```
# umount /mnt/hdd
```

```
[root@Luks-Fedora37 bin]# umount /mnt/hdd
```

Figure 35 : Unmounting the mounted volume

2. Detach the attached LUKS volume.

›_ Console

```
# /usr/lib/systemd/systemd-cryptsetup detach mydisk /dev/sdc1 - pkcs11uri="auto"
```

```
[root@Luks-Fedora37 bin]# ./p11tool2 /usr/lib/systemd/systemd-cryptsetup detach mydisk /dev/sdc1 - pkcs11-uri="auto"
```

Figure 36 : Detach the attached LUKS volume

3. Generate a new RSA key using p11tool2.

›_ Console

```
# p11tool2 Slot=<no> LoginUser=ask

PubKeyAttr=CKA_LABEL="NEWPublicKey",CKA_ID=0x46

PrvKeyAttr=CKA_LABEL="NEWPrivateKey",CKA_ID=0x46 GenerateKeyPair=RSA
```

```
[root@Luks-Fedora37 bin]# ./p11tool2 slot=14 LoginUser=ask PubKeyAttr=CKA_LABEL="NEWPublicKey",CKA_ID=0x46 PrvKeyAttr=CKA_LABEL="NE
WPrivateKey",CKA_ID=0x46 GenerateKeyPair=RSA
Enter normal user PIN:
```

Figure 37 : Key generation output

4. Verify that the keys are generated onto the HSM using following command.

›_ Console

```
# p11tool2 Slot=<no> LoginUser=ask ListObjects
```

```
[root@Luks-Fedora37 bin]# ./p11tool2 slot=14 LoginUser=ask ListObjects
Enter normal user PIN:

+ 2.2
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 9C59AFA9-9974-43F1-8F98-E6F197693B79
  CKA_LABEL                     = NEWPublicKey
  CKA_ID                        = 0x46 (F)
+ 3.2
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 4F8096D0-859B-4EC8-86F7-BB805E1A7E43
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     = NEWPrivateKey
  CKA_ID                        = 0x46 (F)
```

Figure 38 : Key list using p11tool2

5. Generate a self-signed certificate with the generated key.

›_ Console

```
# openssl req -x509 -engine pkcs11 -keyform engine -new -key

"pkcs11:token=UTILUKS;object=NEWPrivateKey" -sha256 -out newcert.pem subj "/
CN=localhost"
```

```
[root@Luks-Fedora37 bin]# openssl req -x509 -engine pkcs11 -keyform engine -new -key "pkcs11:token=UTILUKS;object=N
EWPrivateKey" -sha256 -out newcert.pem -subj "/CN=localhost"
Engine "pkcs11" set.
Enter PKCS#11 token PIN for UTILUKS:
```

Figure 39 : self-sign certificate generation

6. Convert newcert.pem to `newcert.der` format.

›_ Console

```
# openssl x509 -outform der -in newcert.pem -out newcert.der
```

```
[root@Luks-Fedora37 bin]# openssl x509 -outform der -in newcert.pem -out newcert.der
```

Figure 40 : Certificate type change output

7. Import the certificate onto HSM.

›_ Console

```
# p11tool2 slot=<no> LoginUser=ask

CertAttr=CKA_LABEL="NEWCERT",CKA_ID=0x46

PubKeyAttr=CKA_LABEL="NEWPublicKey",CKA_ID=0x46

ImportCert=/opt/utimaco/bin/newcert.der
```

```
[root@Luks-Fedora37 bin]# p11tool2 slot=14 LoginUser=ask CertAttr=CKA_LABEL="NEWCERT",CKA_ID=0x46 PubKeyAttr=CKA_LABEL="NEWPublicKe
y",CKA_ID=0x46 ImportCert=/opt/utimaco/bin/newcert.der
```

Figure 41 : Certificate import using p11tool2

8. Check that the certificate is imported successfully on HSM.

›_ Console

```
# p11tool2 Slot=<no> LoginUser=ask ListObjects
```

```
[root@Luks-Fedora37 bin]# ./p11tool2 slot=14 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_UNIQUE_ID                 = D91408D2-5EDB-4880-9B58-9FB674BFC841
  CKA_LABEL                     = LUKSCERT
  CKA_ID                        = 0x45 (E)
  CKA_SUBJECT                   =
                0x301B3119 30170603 5504030C 10746573 |0 1 0   U    tes|
                  742E7574 696D6163 6F2E636F 6D       |t.utimaco.com   |


+ 1.2
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_UNIQUE_ID                 = A11C70FE-DDDF-4FB1-9DF8-F6C1A2FEACC5
  CKA_LABEL                     = NEWCERT
  CKA_ID                        = 0x46 (F)
  CKA_SUBJECT                   =
                0x30143112 30100603 5504030C 096C6F63 |0 1 0   U    loc|
                  616C686F 7374                       |alhost          |
```

Figure 42 : Certificate list output

9. List the objects on the slot and note down the URL of the new certificate and new private key.

›_ Console

```
# p11tool --provider /opt/utimaco/lib/libcs_pkcs11_R3.so --login --listall
pkcs11:token=UTILUKS
```

```
[root@Luks-Fedora37 bin]# p11tool --provider /opt/utimaco/lib/libcs_pkcs11_R3.so --login --list-all pkcs11:token=UTILUKS
Token 'UTILUKS' with URL 'pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS' requires u
ser PIN
Enter PIN:
Object 2:
        URL: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%46;object=NEWPublicK
ey;type=public
        Type: Public key (RSA-2048)
        Label: NEWPublicKey
        Flags: CKA_WRAP/UNWRAP; CKA_PRIVATE; CKA_SENSITIVE;
        ID: 46
Object 4:
        URL: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%46;object=NEWCERT;ty
pe=cert
        Type: X.509 Certificate (RSA-2048)
        Expires: Sun Apr  9 10:18:48 2023
        Label: NEWCERT
        Flags: CKA_PRIVATE; CKA_SENSITIVE;
        ID: 46

Object 6:
        URL: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%46;object=NEWPrivate
Key;type=private
        Type: Private key (RSA-2048)
        Label: NEWPrivateKey
        Flags: CKA_WRAP/UNWRAP; CKA_PRIVATE; CKA_NEVER_EXTRACTABLE; CKA_SENSITIVE;
        ID: 46
```

Figure 43 : Slot objects list

10. Enroll the certificate on LUKS drive using `systemd-cryptenroll` command.

›_ Console

```
# systemd-cryptenroll --pkcs11-token-

uri="pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;
serial=CS710843_0014;token=UTILUKS;id=%46;object=NEWCERT;type=cert" /dev/sdc1
```

```
[root@Luks-Fedora37 bin]# systemd-cryptenroll --pkcs11-token-uri="pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial
=CS710843_0014;token=UTILUKS;id=%46;object=NEWCERT;type=cert" /dev/sdc1
🔐 Please enter current passphrase for disk /dev/sdc1: ****************
🔐 Please enter PIN for security token 'UTILUKS' in order to unlock volume enrollment operation: ****************
Successfully logged into security token 'UTILUKS'.
New PKCS#11 token enrolled as key slot 2.
```

Figure 44 : systemd-cryptenroll command output

11. Verify that the pkcs11 url of new certificate is added to the luks partition in the token section.

›_ Console

```
# cryptsetup luksDump /dev/sdc1
```

```
[root@Luks-Fedora37 bin]# cryptsetup luksDump /dev/sdc1
LUKS header information
Version:        2
Epoch:          9
Metadata area:  16384 [bytes]
Keyslots area:  16744448 [bytes]
UUID:           23011cfc-3f1e-4c6f-9e9e-5d8c9cbe10d6
Label:          (no label)
Subsystem:      (no subsystem)
Flags:          (no flags)

Data segments:
  0: crypt
        offset: 16777216 [bytes]
        length: (whole device)
        cipher: aes-xts-plain64
        sector: 4096 [bytes]

Keyslots:
  0: luks2
        Key:        512 bits
        Priority:   normal
        Cipher:     aes-xts-plain64
        Cipher key: 512 bits
        PBKDF:      argon2id
        Time cost:  4
```

Figure 45 : Encrypted volume information

```
Tokens:
  0: systemd-pkcs11
        pkcs11-uri: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%46;object=NEW
CERT;type=cert
        pkcs11-key: 9c 77 d3 55 f7 34 b0 8c d6 18 9c 9f 92 84 29 5d
                    08 43 b0 c2 3a 80 18 74 69 d0 aa 7b 3f 56 65 a8
                    95 1b 49 00 b1 0e ec 58 32 4e e1 ee e7 77 7e 63
                    5c 3e 82 43 0c 37 c1 51 0c 45 bd 62 d5 f5 b7 44
                    aa 4f 6e ae 38 dc e5 ec 5e 71 42 f4 1a 22 4c 60
                    12 41 4a 79 67 57 df ba 68 ca 60 eb 2e 26 6e 4f
                    60 86 e4 c9 3a 50 ac b1 da c5 9a d6 7b dd c8 a0
                    3e fb fa b9 df 96 c0 3b c2 4f 79 b2 a5 05 64 2f
                    47 de 42 80 6a 7b e2 45 88 d9 d6 06 7e fd c2 17
                    d5 29 e9 0d d9 af b4 21 73 78 21 21 3e 82 04 b5
                    49 12 8b 53 be 43 9b 6d 15 6c cd e0 c2 da 83 7c
                    e8 7f 24 e5 e7 d8 f9 ba 88 d5 c8 68 06 9b 3d ed
                    67 6f 17 e7 96 0a 45 f5 39 e0 06 18 e7 05 b1 dd
                    d2 79 8d 4a a8 c4 4a 42 61 91 af 39 29 ba 0d 90
                    f5 65 84 ca 90 74 4f 54 64 0c d5 b2 db 55 5f 9e
                    da 77 b3 16 88 49 14 bc aa 34 c0 fb ac 67 1d 9c
        Keyslot:    2
```

Figure 46 : Encrypted volume information

12. Open the luks volume as newbackup disk.

›_ Console

```
# cryptsetup luksOpen /dev/sdc1 newbackup
```

```
[root@Luks-Fedora37 ~]# cryptsetup luksOpen /dev/sdc1 newbackup
Enter passphrase for /dev/sdc1:
```

Figure 47 : luksopen command output

13. Export token using below command to `header.json` file

›_ Console

```
# cryptsetup token export --token-id 0 /dev/sdc1 > header.json
```

```
[root@Luks-Fedora37 bin]# cryptsetup token export --token-id 0 /dev/sdc1 > header.json
```

Figure 48 : Exporting token id information

14. Open `header.json` file and replace the `pkcs-uri` with type=private url of new key. Example

›_ header.json file

```
pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS71084
3_0014; token=UTILUKS;id=%46;object=NEWPrivateKey;type=private
```

15. Import the `header.json` file

›_ Console

```
# cryptsetup token import --json-file header.json --token-id 2 /dev/sdc1
```

Figure 49 : Importing token id information

16. Verify token id 2 is showing new key information using the command below.

›_ Console

```
# cryptsetup luksDump /dev/sdc1
```

```
[root@Luks-Fedora37 bin]# cryptsetup luksDump /dev/sdc1
LUKS header information
Version:        2
Epoch:          15
Metadata area:  16384 [bytes]
Keyslots area:  16744448 [bytes]
UUID:           0a6d0b22-7501-45e0-a6cf-8281ef0be458
Label:          (no label)
Subsystem:      (no subsystem)
Flags:          (no flags)

Data segments:
  0: crypt
        offset: 16777216 [bytes]
        length: (whole device)
        cipher: aes-xts-plain64
        sector: 4096 [bytes]

Keyslots:
  0: luks2
        Key:        512 bits
        Priority:   normal
        Cipher:     aes-xts-plain64
        Cipher key: 512 bits
        PBKDF:      argon2id
        Time cost:  4
        Memory:     875624
        Threads:    2
        Salt:       24 96 6b 4b d9 83 f0 3b 00 03 3f 99 e8 d5 6d ad
                    ae 66 00 e4 2c ed b9 37 22 6a 9b d4 aa 46 e2 39
        AF stripes: 4000
        AF hash:    sha256
        Area offset:32768 [bytes]
```

Figure 50 : LuksDump output

```
  2: systemd-pkcs11
        pkcs11-uri: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%46;object=NEW
PrivateKey;type=private
        pkcs11-key: 9c 77 d3 55 f7 34 b0 8c d6 18 9c 9f 92 84 29 5d
                    08 43 b0 c2 3a 80 18 74 69 d0 aa 7b 3f 56 65 a8
                    95 1b 49 00 b1 0e ec 58 32 4e e1 ee e7 77 7e 63
                    5c 3e 82 43 0c 37 c1 51 0c 45 bd 62 d5 f5 b7 44
                    aa 4f 6e ae 38 dc e5 ec 5e 71 42 f4 1a 22 4c 60
                    12 41 4a 79 67 57 df ba 68 ca 60 eb 2e 26 6e 4f
                    60 86 e4 c9 3a 50 ac b1 da c5 9a d6 7b dd c8 a0
                    3e fb fa b9 df 96 c0 3b c2 4f 79 b2 a5 05 64 2f
                    47 de 42 80 6a 7b e2 45 88 d9 d6 06 7e fd c2 17
                    d5 29 e9 0d d9 af b4 21 73 78 21 21 3e 82 04 b5
                    49 12 8b 53 be 43 9b 6d 15 6c cd e0 c2 da 83 7c
                    e8 7f 24 e5 e7 d8 f9 ba 88 d5 c8 68 06 9b 3d ed
                    67 6f 17 e7 96 0a 45 f5 39 e0 06 18 e7 05 b1 dd
                    d2 79 8d 4a a8 c4 4a 42 61 91 af 39 29 ba 0d 90
                    f5 65 84 ca 90 74 4f 54 64 0c d5 b2 db 55 5f 9e
                    da 77 b3 16 88 49 14 bc aa 34 c0 fb ac 67 1d 9c
        Keyslot:    2
```

Figure 51 : LuksDump output

17. Remove token id 0 and 1 and verify with the command below.

›_ Console

```
# cryptsetup token remove --token-id 0 /dev/sdc1

# cryptsetup token remove --token-id 1 /dev/sdc1
```

```
[root@Luks-Fedora37 bin]# cryptsetup token remove --token-id 0 /dev/sdc1
[root@Luks-Fedora37 bin]# cryptsetup token remove --token-id 1 /dev/sdc1
[root@Luks-Fedora37 bin]# cryptsetup luksDump /dev/sdc1
```

Figure 52 : Removing old token id

```
Tokens:
  2: systemd-pkcs11
        pkcs11-uri: pkcs11:model=CryptoServer;manufacturer=Utimaco%20IS%20GmbH;serial=CS710843_0014;token=UTILUKS;id=%46;object=NEW
PrivateKey;type=private
        pkcs11-key: 9c 77 d3 55 f7 34 b0 8c d6 18 9c 9f 92 84 29 5d
                    08 43 b0 c2 3a 80 18 74 69 d0 aa 7b 3f 56 65 a8
                    95 1b 49 00 b1 0e ec 58 32 4e e1 ee e7 77 7e 63
                    5c 3e 82 43 0c 37 c1 51 0c 45 bd 62 d5 f5 b7 44
                    aa 4f 6e ae 38 dc e5 ec 5e 71 42 f4 1a 22 4c 60
                    12 41 4a 79 67 57 df ba 68 ca 60 eb 2e 26 6e 4f
                    60 86 e4 c9 3a 50 ac b1 da c5 9a d6 7b dd c8 a0
                    3e fb fa b9 df 96 c0 3b c2 4f 79 b2 a5 05 64 2f
                    47 de 42 80 6a 7b e2 45 88 d9 d6 06 7e fd c2 17
                    d5 29 e9 0d d9 af b4 21 73 78 21 21 3e 82 04 b5
                    49 12 8b 53 be 43 9b 6d 15 6c cd e0 c2 da 83 7c
                    e8 7f 24 e5 e7 d8 f9 ba 88 d5 c8 68 06 9b 3d ed
                    67 6f 17 e7 96 0a 45 f5 39 e0 06 18 e7 05 b1 dd
                    d2 79 8d 4a a8 c4 4a 42 61 91 af 39 29 ba 0d 90
                    f5 65 84 ca 90 74 4f 54 64 0c d5 b2 db 55 5f 9e
                    da 77 b3 16 88 49 14 bc aa 34 c0 fb ac 67 1d 9c
        Keyslot:    2
```

Figure 53 : Removing old token id

18. Close the LUKS volume.

›_ Console

```
# cryptsetup luksClose newbackup
```

Figure 54 : lukscolse output

19. Attach the LUKS volume with `systemd-cryptsetup`.

›_ Console

```
# /usr/lib/systemd/systemd-cryptsetup attach luks /dev/sdc1 - pkcs11uri="auto"
```

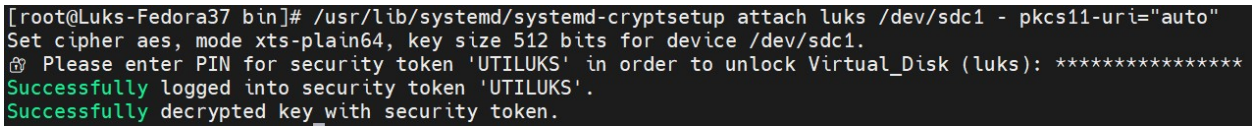

Figure 55 : Attaching luks volume

For Auto-Atttachig and Auto-Mounting LUKS volume refer to section *Auto-Attaching and AutoMounting LUKS Volume during Boot*.

This completes the Integration for Luks with Utimaco SecurityServer.

7 Troubleshooting

Error	Diagnosis
Failed to load LUKS superblock on device /dev/sdc1: Invalid argument	Check the partitions info using df -h command and attach proper disk
Cannot use device /dev/sdc1 which is in use (already mapped or mounted). Failed to activate with specified passphrase: Device or resource busy	Check the partition status, unmount it or reboot the system if needed.

Table 6: List of errors and their diagnoses

8 Further Information

This document forms a part of the information and support which is provided by Utimaco IS GmbH. Additional documentation can be found on the product CD in the Documentation directory.

All CryptoServer product documentation is also available at the Utimaco IS GmbH website: *https://utimaco.com/*.

9 References

Reference	Title/Company	Document No.
[CSADMIN]	CryptoServer – csadm Manual/Utimaco IS GmbH	2009-0003
[CSTrSh]	CryptoServer Troubleshooting/Utimaco IS GmbH	M011-0008-en
[CSADMIN2]	CryptoServer_csadm_Manual_Systemadministrators.pdf	2009-0003
[CSP11Tool2]	CryptoServer_p11tool2_Manual.pdf	2012-0004
[CSPKCSM]	CryptoServer - PKCS#11 P11CAT Manual	M013-0001-en
[CSLAN5]	CryptoServerLAN_Manual_Systemadministrators.pdf	2018-0004

Table 7: References

10 Contact and Support Information

You can reach us from Monday to Friday, 09.00 a.m. to 05.00 p.m., Central European Time (CET).

Utimaco IS GmbH
Krefelder Str. 220
52070 Aachen
Germany

RMA Query

If you need to send the device back to Utimaco IS GmbH, please open a new RMA case (Return Merchandise Authorization). We request that you use the following web address. RMA cases cannot be opened by email or phone.

https://support.hsm.utimaco.com/support/rma/new

Other Support Queries

- Mail (preferred contact method)
  support@utimaco.com
  Attach the diagnostic information to your email.
- Web portal
  https://support.hsm.utimaco.com/support/cases/new/
  The diagnostic information will be requested in our response if necessary.
- By phone
  AMERICAS +1-844-UTIMACO (+1 844-884-6226)
  EMEA +49 800-627-3081
  APAC +81 800-919-1301
  The diagnostic information will be requested in our response if necessary.