IBM

IBM PKCS11

Integration Guide

CryptoServer HSM

Utimaco SecurityServer Software 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	2026-03-02
Status	PUBLISHED
Document No.	IG-2026-0029

Table of Contents

1 About This Guide

This guide describes how to integrate an Utimaco CryptoServer Hardware Security Module (HSM) with IBM PKCS11. Utimaco HSM securely stores the private key used by IBM PKCS11 cryptographic provider to sign the jar files.

1.1 Target Audience for This Guide

This guide is intended for IBM PKCS11 and Utimaco HSM administrators.

1.2 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.

1.3 Abbreviations

The following abbreviations are used in this guide:

Abbreviation	*Meaning*
CA	Certificate Authority
CD	Compact Disc
CMD	Command Prompt
CSADM	CryptoServer Command-line Administration Tool
CSR	Certificate Signing Request
GUI	Graphical User Interface
HSM	Hardware Security Module
IP	Internet Protocol
JCA	Java Cryptography Architecture
JCE	Java Cryptography Extension
JDK	Java Development Kit

Abbreviation	*Meaning*
LAN	Local Area Network
MBK	Master Backup Key
PCIe	PCI Express Interface
PIN	Personal Identification Number
PKCS#11	Public-Key Cryptography Standard #11
RSA	Rivest-Shamir-Adleman
SSL	Secure Socket Layer
SO	Security Officer
URL	Uniform Resource Locator

Table 2: List of abbreviations

2 Overview

2.1 IBM PKCS11 Cryptographic provider

The IBM PKCS11 Impl provider uses the Java™ Cryptography Extension (JCE) and Java Cryptography Architecture (JCA) frameworks to seamlessly add the capability to use hardware cryptography using the PKCS#11 Cryptographic Token Interface standard.

This provider takes advantage of hardware cryptography within the existing JCE architecture and gives Java programmers significant security and performance advantages of hardware cryptography with minimal changes to existing Java applications. Because the complexities of hardware cryptography are taken care of within the normal JCE, advanced security and performance using hardware cryptographic devices is made easily available.

2.2 Utimaco SecurityServer HSM

SecurityServer is a hardware security module developed by Utimaco IS GmbH. SecurityServer 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 the system environment you will be using meets the following hardware and software requirements.

3.1 Tested Versions

The integrations that have been successfully tested with the Utimaco HSM with IBMPKCS11.

Operating System	IBM JAVA	Utimaco Security Server Version	Utimaco HSM
RHEL 8	1.8.0_361	SecurityServer V4.50.0.2	CryptoServer CSe-Series/SeSeries

Table 3: List of tested versions

3.2 Software Requirements

Software	Software Requirements
HSM Interfaces	SecurityServer PKCS#11
IBM JDK 8	1.8.0_361
Host VM	Redhat 8 and above
HSM software	Utimaco SecurityServer Software 4.50.0.2
P11tool2	p11tool2 (3.1.1) from product package Utimaco SecurityServer 4.50.0.2

Table 4: List of software requirements

Here you find additional notes or supplementary information. To download IBM java: https://developer.ibm.com/languages/java/semeru-runtimes/downloads/

3.3 Hardware Requirements

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

Table 5: List of hardware requirements

Setup 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 you have installed/setup:

- SecurityServer is setup and configured. Refer the SecurityServer documentations to setup the HSM
- SecurityServer Default Admin should be replaced with a new admin user
- MBK must be created and stored onto each HSM. Refer the SecurityServer documentations to setup the MBK
- Operating system listed in Tested Versions
- SecurityServer listed in Tested Versions
- Familiarize yourself with the IBMPKCS11 documents and setup process
- Admin user for installing software on IBMPKCS11 server

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 IBMPKCS11 Server.
2. Create utimaco folder under `/opt` directory and further create 2 directories `/opt/utimaco/bin` and `/opt/utimaco/lib`.

›_ Console

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

3. Copy pkcs11 library file `libcs_pkcs11_R3.so` from Utimaco SecurityServer 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 SecurityServer 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 SecurityServer 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 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=<ask>
# ./p11tool2 slot=<slot_no> LoginSO=<ask> InitPin=<ask>
```

```
[root@IBMpkcs11 ~]# cd /opt/utimaco/bin/
[root@IBMpkcs11 bin]# ./p11tool2 slot=0 Label=ibmpkcs11 Login=ADMIN,ADMIN.key InitToken=ask
Enter SO PIN:
Repeat SO PIN:
[root@IBMpkcs11 bin]# ./p11tool2 slot=0 LoginSO=ask InitPin=ask
Enter SO PIN:
Enter normal user PIN:
Repeat normal user PIN:
[root@IBMpkcs11 bin]#
```

Figure 1 : Slot initialization output

4.4 Create pkcs11.cfg at /etc/utimaco/

Create a file `/etc/utimaco/pkcs11.cfg` and add below contents to it

pkcs11.cfg

```
name=CryptoServer library=/opt/utimaco/lib/libcs_pkcs11_R3.so slotListIndex=0
 publickeyimportonly = true attributes(*, CKO_SECRET_KEY, *) = {

   CKA_ENCRYPT=true    CKA_DECRYPT=true}

attributes(GENERATE, CKO_PRIVATE_KEY, CKK_RSA) = {

   CKA_TOKEN=true

   CKA_DECRYPT=true    CKA_UNWRAP=true}

attributes(GENERATE, CKO_PUBLIC_KEY, *) = {

   CKA_TOKEN=true    CKA_VERIFY=true}

attributes(*, CKO_PUBLIC_KEY, CKK_RSA) = {

   CKA_ENCRYPT=true    CKA_WRAP=true    CKA_VERIFY=true}

attributes(IMPORT, CKO_PRIVATE_KEY, CKK_RSA) = {

   CKA_DECRYPT=true

   CKA_UNWRAP=true    CKA_DERIVE=true}

attributes(*,CKO_PRIVATE_KEY,CKK_EC) = {

   CKA_SIGN = true

   CKA_DERIVE = true

   CKA_Token = true}
```

This file will be used by **IBMPKCS11** provider to get library and slot information and perform cryptographic operation on Utimaco HSM.

Specify correct library path and slot index.

5 IBM JAVA Configuration to use Utimaco HSM

5.1 Download and Install IBM JAVA

Download IBM JDK from https://developer.ibm.com/languages/java/semeruruntimes/downloads/ .

1. Extract the downloaded file.

›_ Console

```
# tar xf ibm-semeru-open-jdk_x.x.tar.gz
```

3. Update the PATH variable to include IBM JAVA utilities in user's bash_profile. For example, if the user is root, then add the below content in `/root/.bash_profile`.

.bash_profile

```
export PATH=<Path_to_IBM_JAVA>/bin:$PATH
```

4. Logout and login again for changes to take effect.

5.2 Update java.security file to use Utimaco HSM

1. Go to the `<JDK_Installation_directory>/jre/lib/security` directory.

›_ Console

```
# cd /opt/ibm/java-x86_64-80/jre/lib/security/java.security
```

2. Edit the java.security configuration file to add IBM PKCS11 provider as highlighted below.

›_ Console

```
security.provider.1=com.ibm.jsse2.IBMJSSEProvider2
security.provider.2=com.ibm.crypto.plus.provider.IBMJCEPlus
security.provider.3=com.ibm.crypto.provider.IBMJCE
security.provider.4=com.ibm.security.jgss.IBMJGSSProvider
security.provider.5=com.ibm.security.cert.IBMCertPath
security.provider.6=com.ibm.security.sasl.IBMSASL
security.provider.7=com.ibm.xml.crypto.IBMXMLCryptoProvider
security.provider.8=com.ibm.xml.enc.IBMXMLEncProvider
security.provider.9=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
security.provider.10=sun.security.provider.Sun
security.provider.11=com.ibm.crypto.pkcs11impl.provider.IBMPKCS11Impl /etc/
utimaco/pkcs11.cfg
```

Specify correct provider number and path for pkcs11.cfg file.

6 Jar Signing and Verification with IBM PKCS11 provider and Utimaco HSM

6.1 Using CA Signed Certificate for Jar Signing and Verification

6.1.1 With RSA Key (CA Signed Certificate)

1. Generate a keypair on Utimaco HSM with the help of keytool command.

›_ Console

```
# keytool -genkey -alias ibmrsa -keyalg RSA -keysize 2048 -keystore NONE
-storetype PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer
```

Provide information when prompted Here:

- RSA is the key algorithm
- 2048 is the key size
- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider name
- ibmrsa is the key name that will be generated on Utimaco HSM

Provide the keystore password when prompted:

```
[root@ibmpkcs11 ~]# keytool -genkey -alias ibmrsa -keyalg RSA -keysize 2048 -keystore NONE -storetype PKCS11IMPLKS -providername IBMPKCS11
Impl-CryptoServer
Enter keystore password:
What is your first and last name?
  [Unknown]:  test demo
What is the name of your organizational unit?
  [Unknown]:  security
What is the name of your organization?
  [Unknown]:  utimaco
What is the name of your City or Locality?
  [Unknown]:  pune
What is the name of your State or Province?
  [Unknown]:  MH
What is the two-letter country code for this unit?
  [Unknown]:  IN
Is CN=test demo, OU=security, O=utimaco, L=pune, ST=MH, C=IN correct? (type "yes" or "no")
  [no]:  yes

Enter key password for <ibmrsa>:
        (RETURN if same as keystore password):
```

Figure 2 : Key generation using keytool command

2. Verify the entry with same alias name is generated using keytool command.

›_ Console

```
# keytool -list -keystore NONE -storetype PKCS11IMPLKS -providername
IBMPKCS11Impl-CryptoServer
```

Here:

- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider's name

Provide the keystore password when prompted:

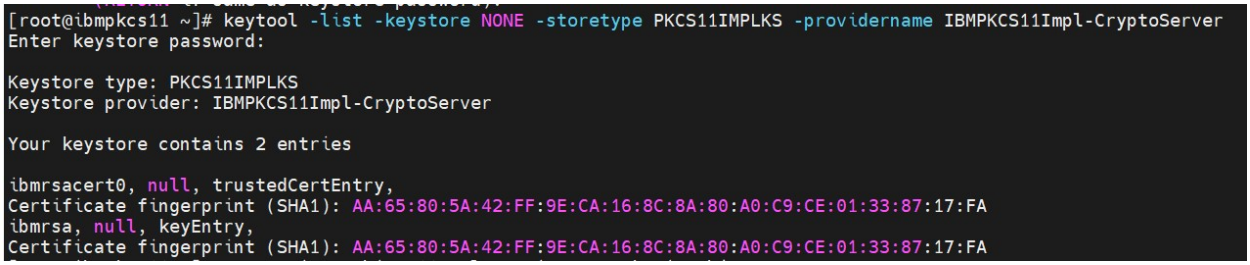

Figure 3 : Listkeys output

3. List the objects using p11tool2.

›_ Console

```
# p11tool2 Slot=0 LoginUser=ask ListObjects
```

Enter user PIN when prompted:

```
[root@ibmpkcs11 ~]# /opt/utimaco/bin/p11tool2 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_UNIQUE_ID                 = 1ADA1B6C-D378-4744-93BD-26A5B59A301F
  CKA_LABEL                     = ibmrsa
  CKA_ID                        =

  CKA_SUBJECT                   =
               0x3062310B 30090603 55040613 02494E31 |0b1 0   U    IN1|
                 0B300906 03550408 13024D48 310D300B | 0   U    MH1 0 |
                 06035504 07130470 756E6531 10300E06 |  U    pune1 0  |
                 0355040A 13077574 696D6163 6F311130 | U    utimaco1 0|
                 0F060355 040B1308 73656375 72697479 |   U    security|
                 31123010 06035504 03130974 65737420 |1 0   U    test |
                 64656D6F                            |demo            |



CKO_PUBLIC_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 371FFE92-9C8B-4D0C-951A-02F98E211095
  CKA_LABEL                     =
  CKA_ID                        =



CKO_PRIVATE_KEY:

+ 3.1
  CKA_KEY_TYPE                  = CKK_RSA
```

Figure 4 : List keys output using p11tool2

```
CKO_PUBLIC_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 371FFE92-9C8B-4D0C-951A-02F98E211095
  CKA_LABEL                     =
  CKA_ID                        =



CKO_PRIVATE_KEY:

+ 3.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 369F6543-5385-4B1C-9B77-8A9524D1BB46
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     =
  CKA_ID                        =


+ 3.2
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 3B69ECD3-8588-4100-B207-38BBA0180584
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     = ibmrsa
  CKA_ID                        =
```

4. Generate a CSR using Keytool command.

›_ Console

```
# keytool -certreq -alias ibmrsa -keystore NONE -storetype PKCS11IMPLKS
providername IBMPKCS11Impl-CryptoServer -file ibm.csr
```

Here:

- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider name
- ibmrsa is the key name
- ibm.csr is the CSR file name that will be generated

Provide keystore password when prompted

5. Get this CSR signed by CA.
6. Copy the signed certificate and root CA certificate on the IBMPKCS11 server.
7. Import Root CA certificate into HSM keystore.

›_ Console

```
#  keytool -importcert -alias rootca -file /home/LAbCA-Root.crt storetype
PKCS11IMPLKS -keystore NONE -providername IBMPKCS11ImplCryptoServer
```

```
[root@ibmpkcs11 ~]# keytool -importcert -alias rootca -file /home/LAbCA-Root.crt -storetype PKCS11IMPLKS -keystore NONE -providername IBMPKCS11Impl-Cryp
toServer
Enter keystore password:
Owner: CN=LabCA-Root, OU=IT, O=security, L=Campbell, ST=Austin, C=US
Issuer: CN=LabCA-Root, OU=IT, O=security, L=Campbell, ST=Austin, C=US
Serial number: 40f8f17a48d0bcc3
Valid from: 11/29/22 5:36 AM until: 11/29/32 5:36 AM
Certificate fingerprints:
         MD5:  80:E7:CE:91:84:D6:E9:D9:03:53:DB:F3:11:84:F3:34
         SHA1: D9:BE:FA:00:CB:52:0E:7F:7B:82:CE:D3:54:02:BB:1F:43:21:B8:A1
         SHA256: 58:E9:C6:A3:12:0D:A9:3A:97:E8:0D:03:06:9B:89:0F:05:E6:EB:1F:46:1C:E8:B1:B6:DF:DE:3E:4D:0B:EC:64
         Signature algorithm name: SHA256withRSA
         Version: 3

Extensions:

#1: ObjectId: 2.16.840.1.113730.1.1 Criticality=false
NetscapeCertType [
   SSL CA
   S/MIME CA
   Object Signing CA]

#2: ObjectId: 2.16.840.1.113730.1.13 Criticality=false
0000: 16 0e 49 6e 66 6f 73 65  63 20 4c 61 62 20 43 41  ..Infosec.Lab.CA


#3: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  Key_CertSign
  Crl_Sign
]

#4: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 85 42 28 42 84 18 55 2c  73 aa 74 dc 23 ee 74 7a  .B.B..U.s.t...tz
0010: 00 fe 2e dc                                       ....
]
]
```

Figure 5 : Importing root certificate into keystore

```
#5: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
CA:true
PathLen:2147483647
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

8. Import the signed certificate reply using the command below.

›_ Console

```
#  keytool -importcert -alias ibmrsa -file /home/test_demo.p7b -storetype
PKCS11IMPLKS -keystore NONE -providername IBMPKCS11Impl-CryptoServer
```

```
[root@ibmpkcs11 ~]# keytool -importcert -alias ibmrsa -file /home/test_demo.p7b -storetype PKCS11IMPLKS -keystore NONE -providername IBMPKCS11Impl-Crypt
oServer
Enter keystore password:
Certificate reply was installed in keystore
```

Figure 6 : Import user certificate into keystore

9. Verify that the keytool command shows the signed certificate as well as root CA certificate.

›_ Console

```
# keytool -list -keystore NONE -storetype PKCS11IMPLKS -providername
IBMPKCS11Impl-CryptoServer
```

Here:

- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider's name

Provide the keystore password when prompted:

```
[root@ibmpkcs11 ~]# keytool -list -keystore NONE -storetype PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer
Enter keystore password:

Keystore type: PKCS11IMPLKS
Keystore provider: IBMPKCS11Impl-CryptoServer

Your keystore contains 3 entries

rootca, null, trustedCertEntry,
Certificate fingerprint (SHA1): D9:BE:FA:00:CB:52:0E:7F:7B:82:CE:D3:54:02:BB:1F:43:21:B8:A1
ibmrsacert0, null, trustedCertEntry,
Certificate fingerprint (SHA1): AE:48:5C:97:A1:87:79:94:5E:5F:BF:9A:E7:C3:0E:F0:25:DF:EC:6D
ibmrsa, null, keyEntry,
Certificate fingerprint (SHA1): AE:48:5C:97:A1:87:79:94:5E:5F:BF:9A:E7:C3:0E:F0:25:DF:EC:6D
```

Figure 7 : Listkeys output showing signed certificate as well as root CA

10. Sign any sample jar file with jarsigner command.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype
PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer -signedjar
hello_worldoutput.jar HelloWorld-0.7.0.jar ibmrsa
```

Here

- http://timestamp.digicert.com is URL of timestamp server
- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider's name

- hello_worldoutput.jar is the new output signed jar file that will be generated
- HelloWorld-0.7.0.jar is the jar file to be signed
- ibmrsa is the RSA key used for jar signing

Provide the keystore password when prompted:

```
[root@ibmpkcs11 ~]# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer -signed
jar hello_worldoutput.jar HelloWorld-0.7.0.jar ibmrsa
Enter Passphrase for keystore:
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
jar signed.

The timestamp will expire on 2031-11-09.
```

Figure 8 : Signing the jar using jarsigner command

11. Verify the signed jar.

›_ Console

```
# jarsigner -verify hello_worldoutput.jar -keystore NONE -storetype PKCS11IMPLKS
-storepass 12345678 -providername IBMPKCS11Impl-CryptoServer
```

Here hello_worldoutput.jar is the newly generated signed jar file:

```
[root@ibmpkcs11 ~]# jarsigner -verify hello_worldoutput.jar -keystore NONE -storetype PKCS11IMPLKS -storepass 12345678 -providername IBMPKCS11Impl-Crypt
oServer
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp
 Use com.xxx.xx.cpcp

jar verified.
```

Figure 9 : Verifying signed jar

6.1.2 With EC Key (CA Signed Certificate)

1. Generate an EC keypair on Utimaco HSM.

›_ Console

```
# keytool -genkey -alias ibmec -keyalg EC -keystore NONE -storetype PKCS11IMPLKS
-providername IBMPKCS11Impl-CryptoServer
```

Provide information when prompted Here:

- EC is the key algorithm
- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider name
- ibmec is the key name that will be generated on Utimaco HSM

Provide the keystore password when prompted

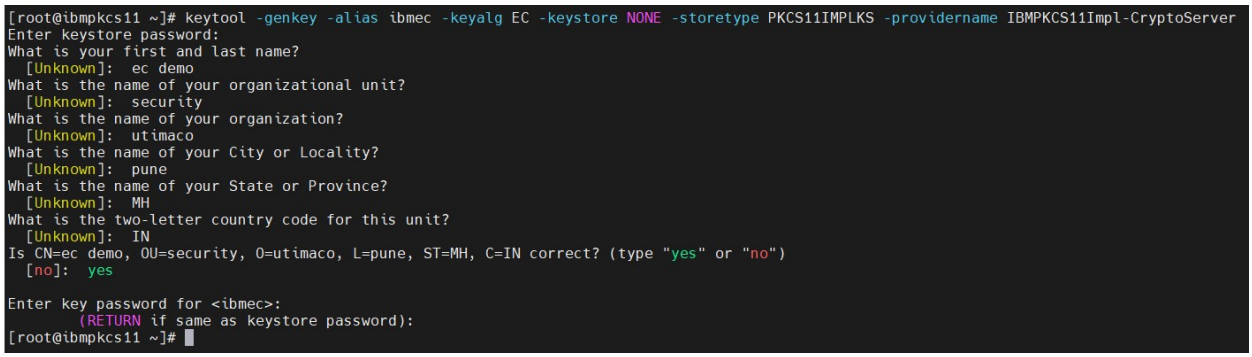

Figure 10 : Key generation using keytool command

2. Verify the entry with same alias name is generated using keytool command.

›_ Console

```
# keytool -list -keystore NONE -storetype PKCS11IMPLKS -providername
IBMPKCS11Impl-CryptoServer
```

Here:

- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider name

Provide the keystore password when prompted

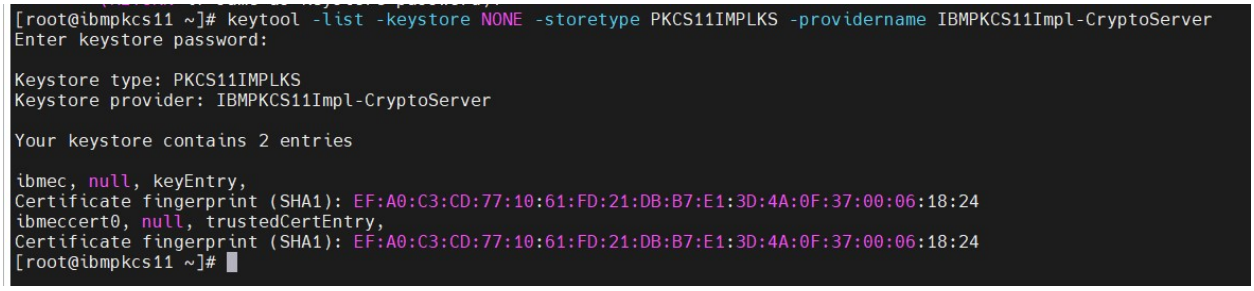

Figure 11 : Listkeys output

3. List the objects using p11tool2.

›_ Console

```
# p11tool2 Slot=0 LoginUser=ask ListObjects
```

Enter user PIN when prompted

```
[root@ibmpkcs11 ~]# /opt/utimaco/bin/p11tool2 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE           = CKC_X_509
  CKA_UNIQUE_ID                  = F0DA2E6A-78FD-4788-859C-9C8C6642A468
  CKA_LABEL                      = ibmec
  CKA_ID                         =

  CKA_SUBJECT                    =
               0x3060310B 30090603 55040613 02494E31 |0`1 0   U    IN1|
                 0B300906 03550408 13024D48 310D300B | 0   U    MH1 0 |
                 06035504 07130470 756E6531 10300E06 |  U    pune1 0  |
                 0355040A 13077574 696D6163 6F311130 | U    utimaco1 0|
                 0F060355 040B1308 73656375 72697479 |   U    security|
                 3110300E 06035504 03130765 63206465 |1 0   U    ec de|
                 6D6F                               |mo             |



CKO_PUBLIC_KEY:

+ 2.1
  CKA_KEY_TYPE                   = CKK_ECDSA
  CKA_UNIQUE_ID                  = 3A266A96-E3C7-4F6C-B3AC-7091B962C6A4
  CKA_LABEL                      =
  CKA_ID                         =



CKO_PRIVATE_KEY:

+ 3.1
  CKA_KEY_TYPE                   = CKK_ECDSA
  CKA_UNIQUE_ID                  = ECC3A35D-02E5-4F36-AC09-8A0E0DD70F86
  CKA_SENSITIVE                  = CK_TRUE
  CKA_EXTRACTABLE                = CK_FALSE
  CKA_LABEL                      =
```

Figure 12 : List keys output using p11tool2

```
  CKA_LABEL                      =
  CKA_ID                         =


+ 3.2
  CKA_KEY_TYPE                   = CKK_ECDSA
  CKA_UNIQUE_ID                  = 182E0EEF-36A0-4EB5-9757-901D5CA72233
  CKA_SENSITIVE                  = CK_TRUE
  CKA_EXTRACTABLE                = CK_FALSE
  CKA_LABEL                      = ibmec
  CKA_ID                         =

[root@ibmpkcs11 ~]#
```

4. Generate a CSR using Keytool command.

›_ Console

```
# keytool -certreq -alias ibmec -keystore NONE -storetype PKCS11IMPLKS
providername IBMPKCS11Impl-CryptoServer -file ec.csr
```

Here:

- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- Provide the keystore password when prompted
- IBMPKCS11Impl-CryptoServer is the provider name
- ibmec is the key name
- ec.csr is the CSR file name that will be generated

Provide the keystore password when prompted

5. Get this CSR signed by CA.
6. Copy the signed certificate and root CA certificate on the IBMPKCS11 server.
7. Import Root CA certificate into HSM keystore.

›_ Console

```
#  keytool -importcert -alias rootca -file /home/LAbCA-Root.crt storetype
PKCS11IMPLKS -keystore NONE -providername IBMPKCS11ImplCryptoServer
```

```
[root@ibmpkcs11 ~]# keytool -importcert -alias rootca -file /home/LAbCA-Root.crt -storetype PKCS11IMPLKS -keystore NONE -providername IBMPKCS11Impl-Cryp
toServer
Enter keystore password:
Owner: CN=LabCA-Root, OU=IT, O=security, L=Campbell, ST=Austin, C=US
Issuer: CN=LabCA-Root, OU=IT, O=security, L=Campbell, ST=Austin, C=US
Serial number: 40f8f17a48d0bcc3
Valid from: 11/29/22 5:36 AM until: 11/29/32 5:36 AM
Certificate fingerprints:
         MD5:  80:E7:CE:91:84:D6:E9:D9:03:53:DB:F3:11:84:F3:34
         SHA1: D9:BE:FA:00:CB:52:0E:7F:7B:82:CE:D3:54:02:BB:1F:43:21:B8:A1
         SHA256: 58:E9:C6:A3:12:0D:A9:3A:97:E8:0D:03:06:9B:89:0F:05:E6:EB:1F:46:1C:E8:B1:B6:DF:DE:3E:4D:0B:EC:64
         Signature algorithm name: SHA256withRSA
         Version: 3

Extensions:

#1: ObjectId: 2.16.840.1.113730.1.1 Criticality=false
NetscapeCertType [
   SSL CA
   S/MIME CA
   Object Signing CA]

#2: ObjectId: 2.16.840.1.113730.1.13 Criticality=false
0000: 16 0e 49 6e 66 6f 73 65  63 20 4c 61 62 20 43 41  ..Infosec.Lab.CA


#3: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  Key_CertSign
  Crl_Sign
]

#4: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 85 42 28 42 84 18 55 2c  73 aa 74 dc 23 ee 74 7a  .B.B..U.s.t...tz
0010: 00 fe 2e dc                                       ....
]
]
```

Figure 13 : Importing root certificate into keystore

```
#5: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
CA:true
PathLen:2147483647
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
[root@ibmpkcs11 ~]#
```

8. Import the signed certificate reply using the command below.

›_ Console

```
#  keytool -importcert -alias ibmec -file /home/ec_demo.p7b -storetype
PKCS11IMPLKS -keystore NONE -providername IBMPKCS11Impl-CryptoServer
```

```
[root@ibmpkcs11 ~]# keytool -importcert -alias ibmec -file /home/ec_demo.p7b -storetype PKCS11IMPLKS -keystore NONE -providername IBMPKCS11Impl-CryptoSe
rver
Enter keystore password:
Certificate reply was installed in keystore
[root@ibmpkcs11 ~]#
```

Figure 14 : Import User Certificate into Keystore

9. Verify that the keytool command shows the signed certificate as well as root CA certificate.

›_ Console

```
# keytool -list -keystore NONE -storetype PKCS11IMPLKS -providername
IBMPKCS11Impl-CryptoServer
```

Here:

- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider's name

Provide the keystore password when prompted

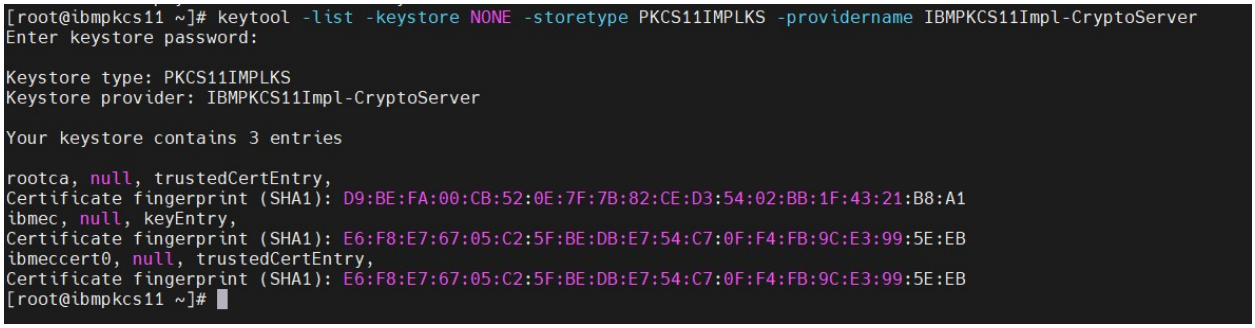

Figure 15 : Listkeys output showing signed certificate as well as root CA 10. Sign any sample jar file using jarsigner tool

10. Sign any sample jar file using jarsigner tool.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype
PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer -signedjar
sample_output.jar samples.jar ibmec
```

Here:

- http://timestamp.digicert.com is URL of timestamp server
- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype

- IBMPKCS11Impl-CryptoServer is the provider's name
- sample_output.jar is the new output signed jar file that will be generated
- samples.jar is the Jar file to be signed
- ibmec is the key name that will be used for signing

Provide the keystore password when prompted

Figure 16 : Signing the jar using jarsigner command

11. Verify the signed jar.

›_ Console

```
# jarsigner -verify sample_output.jar -keystore NONE -storetype PKCS11IMPLKS
-storepass 12345678 -providername IBMPKCS11Impl-CryptoServer
```

Here sample_output.jar is the newly generated signed jar file

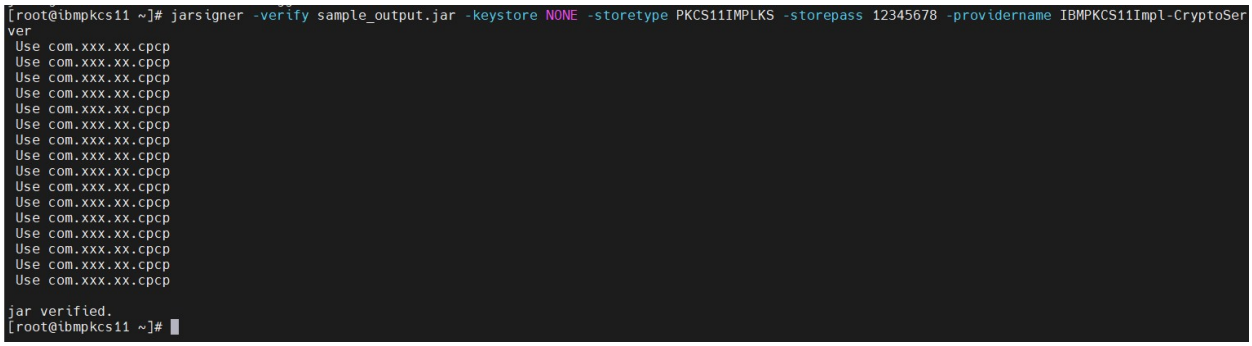

Figure 17 : Verifying signed jar

6.2 Using Self Signed Certificate for Jar Signing and Verification

6.2.1 With RSA key (Self Signed Certificate)

1. Generate a keypair on Utimaco HSM.

›_ Console

```
# keytool -genkey -alias ibmrsa -keyalg RSA -keysize 2048 -keystore NONE
-storetype PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer
```

Provide information when prompted Here:

- RSA is the key algorithm
- 2048 is the key size
- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider name
- ibmrsa is the key name that will be generated on Utimaco HSM

Provide the keystore password when prompted

```
[root@ibmpkcs11 ~]# keytool -genkey -alias ibmrsa -keyalg RSA -keysize 2048 -keystore NONE -storetype PKCS11IMPLKS -providerna
me IBMPKCS11Impl-CryptoServer
Enter keystore password:
What is your first and last name?
  [Unknown]:  test demo
What is the name of your organizational unit?
  [Unknown]:  security
What is the name of your organization?
  [Unknown]:  utimaco
What is the name of your City or Locality?
  [Unknown]:  pune
What is the name of your State or Province?
  [Unknown]:  MH
What is the two-letter country code for this unit?
  [Unknown]:  IN
Is CN=test demo, OU=security, O=utimaco, L=pune, ST=MH, C=IN correct? (type "yes" or "no")
  [no]:  yes

Enter key password for <ibmrsa>:
        (RETURN if same as keystore password):
```

Figure 18 : Key generation using keytool command

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

2. Verify the entry with same alias name is generated using keytool command.

›_ Console

```
# keytool -list -keystore NONE -storetype PKCS11IMPLKS -providername
IBMPKCS11Impl-CryptoServer
```

Here:

- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider's name

Provide the keystore password when prompted

```
[root@ibmpkcs11 ~]# keytool -list -keystore NONE -storetype PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer
Enter keystore password:

Keystore type: PKCS11IMPLKS
Keystore provider: IBMPKCS11Impl-CryptoServer

Your keystore contains 2 entries

ibmrsacert0, null, trustedCertEntry,
Certificate fingerprint (SHA1): B9:62:6B:05:5D:84:52:C4:74:8D:9D:B7:C5:6F:74:FA:C6:5B:67:5D
ibmrsa, null, keyEntry,
Certificate fingerprint (SHA1): B9:62:6B:05:5D:84:52:C4:74:8D:9D:B7:C5:6F:74:FA:C6:5B:67:5D
```

Figure 19 : Listkeys output

3. List the objects using p11tool2.

›_ Console

```
# p11tool2 Slot=0 LoginUser=ask ListObjects
```

Enter user PIN when prompted

```
[root@ibmpkcs11 ~]# /opt/utimaco/bin/p11tool2 Slot=0 LoginUser=12345678 ListObjects


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE           = CKC_X_509
  CKA_UNIQUE_ID                  = C532E8AB-C8C7-447F-BB94-ACA61138C6FE
  CKA_LABEL                      = ibmrsa
  CKA_ID                         =

  CKA_SUBJECT                    =
               0x3062310B 30090603 55040613 02494E31 |0b1 0   U    IN1|
                 0B300906 03550408 13024D48 310D300B | 0   U    MH1 0 |
                 06035504 07130470 756E6531 10300E06 |  U    pune1 0  |
                 0355040A 13077574 696D6163 6F311130 | U    utimaco1 0|
                 0F060355 040B1308 73656375 72697479 |   U    security|
                 31123010 06035504 03130974 65737420 |1 0   U    test |
                 64656D6F                           |demo           |



CKO_PUBLIC_KEY:

+ 2.1
  CKA_KEY_TYPE                   = CKK_RSA
  CKA_UNIQUE_ID                  = A5CD007A-4D24-4462-B7E2-2FFB9B707FFF
  CKA_LABEL                      =
  CKA_ID                         =



CKO_PRIVATE_KEY:

+ 3.1
  CKA_KEY_TYPE                   = CKK_RSA
  CKA_UNIQUE_ID                  = DA2FCE5F-D82F-4E81-A1AD-67D051CACECB
  CKA_SENSITIVE                  = CK_TRUE
  CKA_EXTRACTABLE                = CK_FALSE
  CKA_LABEL                      =
  CKA_ID                         =
```

Figure 20 : List keys output using p11tool2

4. Sign any sample jar file with jarsigner command.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype
PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer -signedjar
hello_worldoutput.jar HelloWorld-0.7.0.jar ibmrsa
```

Here:

- http://timestamp.digicert.com is URL of timestamp server
- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype

- IBMPKCS11Impl-CryptoServer is the provider's name
- hello_worldoutput.jar is the new output signed jar file that will be generated
- HelloWorld-0.7.0.jar is the Jar file to be signed
- ibmrsa is the RSA key used for jar signing

Provide the keystore password when prompted

Figure 21 : Signing the jar using jarsigner command

5. Verify the signed jar.

›_ Console

```
# jarsigner -verify hello_worldoutput.jar -keystore NONE -storetype PKCS11IMPLKS
-storepass 12345678 -providername IBMPKCS11Impl-CryptoServer
```

Here hello_worldoutput.jar is the newly generated signed jar file

Figure 22 : Verifying signed jar

6.2.2 With EC key (Self Signed Certificate)

1. Generate an EC keypair on Utimaco HSM.

›_ Console

```
# keytool -genkey -alias ibmec -keyalg EC -keystore NONE -storetype PKCS11IMPLKS
-providername IBMPKCS11Impl-CryptoServer
```

Provide information when prompted Here:

- EC is the key algorithm
- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider name
- ibmec is the key name that will be generated on Utimaco HSM

Provide the keystore password when prompted

```
[root@ibmpkcs11 ~]# keytool -genkey -alias ibmec -keyalg EC -keystore NONE -storetype PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServe
r
Enter keystore password:
What is your first and last name?
  [Unknown]:  ec demo
What is the name of your organizational unit?
  [Unknown]:  security
What is the name of your organization?
  [Unknown]:  utimaco
What is the name of your City or Locality?
  [Unknown]:  pune
What is the name of your State or Province?
  [Unknown]:  MH
What is the two-letter country code for this unit?
  [Unknown]:  IN
Is CN=ec demo, OU=security, O=utimaco, L=pune, ST=MH, C=IN correct? (type "yes" or "no")
  [no]:  yes

Enter key password for <ibmec>:
        (RETURN if same as keystore password):
[root@ibmpkcs11 ~]#
```

Figure 23 : Keytool command to generate keys

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

2. Verify the entry with same alias name is generated.

›_ Console

```
# keytool -list -keystore NONE -storetype PKCS11IMPLKS -providername
IBMPKCS11Impl-CryptoServer
```

Here:

- NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider's name

Provide the keystore password when prompted

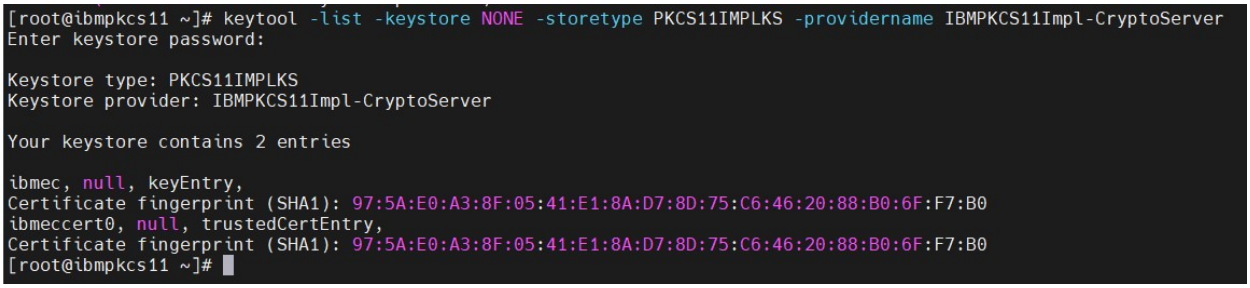

Figure 24 : Listkeys output

3. List the objects using p11tool2.

›_ Console

```
# p11tool2 Slot=0 LoginUser=ask ListObjects
```

Enter user PIN when prompted

```
[root@ibmpkcs11 ~]# /opt/utimaco/bin/p11tool2 Slot=0 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_UNIQUE_ID                 = D596011A-4A9B-410C-B98E-FEF4541F6605
  CKA_LABEL                     = ibmec
  CKA_ID                        =

  CKA_SUBJECT                   =
               0x3060310B 30090603 55040613 02494E31 |0`1 0   U    IN1|
                 0B300906 03550408 13024D48 310D300B | 0   U    MH1 0 |
                 06035504 07130470 756E6531 10300E06 |  U    pune1 0  |
                 0355040A 13077574 696D6163 6F311130 | U    utimaco1 0|
                 0F060355 040B1308 73656375 72697479 |   U    security|
                 3110300E 06035504 03130765 63206465 |1 0   U    ec de|
                 6D6F                                |mo             |



CKO_PUBLIC_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = 6CE31288-F739-4482-9A8B-598BD5F20039
  CKA_LABEL                     =
  CKA_ID                        =



CKO_PRIVATE_KEY:

+ 3.1
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = 8176972B-8B52-4F12-A87C-9E4828EDE3E5
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     =
```

Figure 25 : List keys output using p11tool2

```
+ 3.2
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = F38E9AB1-C2DD-4A6F-A471-06344035A84C
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     = ibmec
  CKA_ID                        =

[root@ibmpkcs11 ~]#
```

4. Sign any sample jar file using jarsigner tool.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype
PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer -signedjar
sample_output.jar samples.jar ibmec
```

Here:

- http://timestamp.digicert.com is URL of timestamp server
- Here NONE is the keystore for HSM
- PKCS11IMPLKS is the storetype
- IBMPKCS11Impl-CryptoServer is the provider's name
- sample_output.jar is the new output signed jar file that will be generated
- samples.jar is the jar file to be signed

```
[root@ibmpkcs11 ~]# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11IMPLKS -providername IBMPKCS11Impl-CryptoServer -signed
jar sample_output.jar samples.jar ibmec
Enter Passphrase for keystore:
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
jar signed.

Warning:
The signer's certificate is self-signed.

The timestamp will expire on 2031-11-09.
[root@ibmpkcs11 ~]#
```

Figure 26 : Signing the jar using jarsigner command

5. Verify the signed jar.

›_ Console

```
# jarsigner -verify sample_output.jar -keystore NONE -storetype PKCS11IMPLKS
-storepass 12345678 -providername IBMPKCS11Impl-CryptoServer
```

Here sample_output.jar is the newly generated signed jar file

```
[root@ibmpkcs11 ~]# jarsigner -verify sample_output.jar -keystore NONE -storetype PKCS11IMPLKS -storepass 12345678 -providername IBMPKCS11Impl-CryptoSer
ver
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp
  Use com.xxx.xx.cpcp

jar verified.
[root@ibmpkcs11 ~]#
```

Figure 27 : Verifying signed jar

This completes the Integration for IBM PKCS11 with Utimaco SecurityServer.

7 Troubleshooting

Error	Diagnosis
LoginUser= failed: 05.12.2021 23:45:45 src/p11adm_R2.c[429] p11_login: C_Login [type=1] returned Error 0x00000102 (CKR_USER_PIN_NOT_INITIALIZED)	PKCS#11 Slot is not initialized.
The CryptoServer PKCS#11 Library R3 is not initialized. Error CKR_CRYPTOKI_NOT_INITIALIZED occurred	PKCS#11 Slot is not initialized.

Table 6: List of error and its diagnosis

8 Further Information

This document forms a part of the information and support which is provided by the 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