IBM

IBM PKCS11

Integration Guide

u.trust GP HSM

SecurityServer 6.4.0.0

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	2026-08-05
Status	PUBLISHED
Document No.	IG-2026-0064

All rights reserved	No part of this documentation may be reproduced in any form (printing, photocopy or according to any other process) without the written approval of Utimaco IS GmbH or be processed, reproduced or distributed using electronic systems. Utimaco IS GmbH reserves the right to modify or amend the documentation at any time without prior notice. Utimaco IS GmbH assumes no liability for typographical errors and damages incurred due to them. All trademarks and registered trademarks are the property of their respective owners.

Table of Contents

1 About This Guide 4
1.1 Target Audience for This Guide 4
1.2 Document Conventions 4
1.3 Abbreviations 5
2 Overview 7
2.1 PKCS11 Cryptographic Provider 7
2.2 Utimaco u.trust GP HSM 7
2.3 Joint Value Proposition 7
3 Integration Requirements and Prerequisites 9
3.1 Tested Versions 9
3.2 Software Requirements 9
3.3 Hardware Requirements 10
3.4 Prerequisites 10
4 Installing and Configuring Utimaco u.trust GP HSM Software 11
4.1 Download and Install Utimaco Software 11
4.2 SecurityServer PKCS#11 Configuration 12
4.3 Create SO User and Initialize a Slot 13
4.4 Create pkcs11.cfg at /etc/utimaco/ 14
5 IBM JAVA Configuration to Use Utimaco HSM 15
5.1 Download and Install IBM JAVA 15
6 Verification with IBM PKCS11 Provider and Utimaco HSM 16
6.1 Using CA Signed Certificate for Jar Signing and Verification 16
6.1.1 With RSA Key (CA Signed Certificate) 16
6.1.2 With EC Key (CA Signed Certificate) 21
6.2 Using Self Signed Certificate for Jar Signing and Verification 27
6.2.1 With RSA Key (Self Signed Certificate) 27
6.2.2 With EC Key (Self Signed Certificate) 31
7 Troubleshooting 36
8 Contact and Support Information 37
9 References 38
10 Command Summary 39

1 About This Guide

This guide describes how to integrate an Utimaco u.trust General Purpose (GP) 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 PKCS11 Cryptographic Provider

The IBM Semeru JDK (version 11 and above) includes the standard SunPKCS11 provider by default, enabling integration with PKCS#11-compliant cryptographic devices without requiring an IBM-specific PKCS#11 provider.

The SunPKCS11 provider integrates with the Java Cryptography Architecture (JCA) and Java Cryptography Extension (JCE) frameworks to enable hardware-based cryptographic operations using the PKCS#11 standard. It allows Java applications to securely access external cryptographic devices, such as Hardware Security Modules (HSMs), through standard Java APIs without requiring application-level changes.

In this integration, the SunPKCS11 provider acts as a bridge between the Java runtime and the Utimaco SecurityServer HSM via the PKCS#11 library. Cryptographic operations such as key generation and digital signing are performed within the HSM, ensuring that private keys remain protected and are never exposed to the application.

2.2 Utimaco u.trust GP HSM

u.trust GP HSM is a hardware security module developed by Utimaco IS GmbH. u.trust GP HSM 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.

2.3 Joint Value Proposition

Enhanced Security: Private keys are securely stored and used within the Utimaco HSM, never exposed to applications.

Seamless Integration: SunPKCS11 enables Java applications to use HSM-backed cryptography with minimal changes.

Hardware-Based Protection: Cryptographic operations are performed inside the HSM, improving security over software-based approaches.

Compliance Ready: Supports regulatory requirements such as PCI-DSS and FIPS with centralized key management.

Standard-Based & Scalable: Uses PKCS#11 for interoperability and supports enterprise-scale workloads across JDK 11, 17, and 21.

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 SunPKCS11.

Operating System	IBM JAVA	Utimaco Security Server Version	Utimaco HSM
RHEL 9	21.0.10 17.0.18 11.0.30	SecurityServer v6.4.0.0	CryptoServer CSe-Series/SeSeries

Table 3: List of tested versions

3.2 Software Requirements

Software	Software Requirements
HSM Interfaces	SecurityServer PKCS#11
IBM JDK 21	21.0.10.1
Host VM	Redhat 9 and above
HSM software	Utimaco SecurityServer Software v6.4.0.0
P11tool2	p11tool2 (3.1.1) from product package Utimaco SecurityServer v6.4.0.0

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 v6.4.0.0 or higher
Utimaco PCI-e HSM	CryptoServer CSe-Series/Se-Series PCI-e with firmware SecurityServer v6.4.0.0 or 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:

- u.trust GP HSM is set up and configured. Refer to the u.trust GP HSM documentations to set up the HSM.
- u.trust GP HSM Default Admin should be replaced with a new admin user.
- MBK must be created and stored onto each HSM. Refer the u.trust GP HSM 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 u.trust GP HSM 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 u.trust GP HSM 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 u.trust GP HSM 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 u.trust GP HSM documentation. The device may be a u.trust GP HSM (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

Whenever the keystore prompts for a password, enter the HSM slot PIN that you set during slot initialization.

4.4 Create pkcs11.cfg at /etc/utimaco/

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

pkcs11.cfg

```
name=Utimaco
library=/opt/utimaco/lib/libcs_pkcs11_R3.so
slotListIndex=0
```

This file will be used by PKCS11 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/semeru-runtimes/downloads/?license=IBM

1. Extract the downloaded file.

›_ Console

```
# tar xf ibm-semeru-certified-jdk_x64_linux_x.x.tar.gz
```

2. 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
```

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

6 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 -genkeypair -alias utimacoRSAKey -keyalg RSA -keysize 2048 -keystore
NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11
-providerArg /etc/utimaco/pkcs11-java.cfg
```

Provide information when prompted Here:

- RSA is the key algorithm
- 2048 is the key size
- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class
- utimacoRSAKey is the key name that will be generated on Utimaco HSM

Provide the keystore password when prompted:

```
[root@localhost ibm]# keytool -genkeypair -alias utimacoRSAKey -keyalg RSA -keysize 2048 -keystore NONE -storetype PKCS11 -providerClas
s sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg
Enter keystore password:
Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the default value in braces
.
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]:  Campbell
What is the name of your State or Province?
  [Unknown]:  California
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=test demo, OU=Security, O=Utimaco, L=Campbell, ST=California, C=US correct?
  [no]:  yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA384withRSA) with a validity of 90 days
        for: CN=test demo, OU=Security, O=Utimaco, L=Campbell, ST=California, C=US
```

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 PKCS11 -providerClass
sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg
```

Here:

- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider's class

Provide the keystore password when prompted:

```
[root@localhost ibm]# keytool -list -keystore NONE -storetype PKCS11 -providerclass sun.security.pkcs11.SunPKCS11 -providerArg /etc/uti
maco/pkcs11-java.cfg
Enter keystore password:
Keystore type: PKCS11
Keystore provider: SunPKCS11-Utimaco

Your keystore contains 1 entry

utimacoRSAKey, PrivateKeyEntry,
Certificate fingerprint (SHA-256): 3D:72:A0:84:00:04:8A:DA:18:6F:0B:F1:28:A3:E3:BB:70:0F:CD:ED:CC:4B:06:6D:4B:20:AC:2A:B4:8B:4B:23
```

Figure 3 : Listkeys output

3. List the objects using p11tool2.

›_ Console

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

Enter user PIN when prompted:

```
[root@localhost ibm]# cd /opt/utimaco/bin/
[root@localhost bin]# ./p11tool2 Slot=0 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE           = CKC_X_509
  CKA_UNIQUE_ID                  = E40F7E8B-70AE-420E-9139-0B2EDC14A43F
  CKA_LABEL                      = utimacoRSAKey
  CKA_ID                         =
               0x7574696D 61636F52 53414B65 79       |utimacoRSAKey   |

  CKA_SUBJECT                    =
               0x306E310B 30090603 55040613 02555331 |0n1 0   U    US1|
                 13301106 03550408 130A4361 6C69666F | 0   U    Califo|
                 726E6961 3111300F 06035504 07130843 |rnia1 0   U    C|
                 616D7062 656C6C31 10300E06 0355040A |ampbell1 0   U  |
                 13075574 696D6163 6F311130 0F060355 |  Utimaco1 0   U|
                 040B1308 53656375 72697479 31123010 |    Security1 0 |
                 06035504 03130974 65737420 64656D6F |  U    test demo|



CKO_PRIVATE_KEY:

+ 2.1
  CKA_KEY_TYPE                   = CKK_RSA
  CKA_UNIQUE_ID                  = 13B810F6-4FFD-4138-8EA7-B10FC395EECC
  CKA_SENSITIVE                  = CK_TRUE
  CKA_EXTRACTABLE                = CK_FALSE
  CKA_LABEL                      =
  CKA_ID                         =
               0x7574696D 61636F52 53414B65 79       |utimacoRSAKey   |
```

Figure 4 : List keys output using p11tool2

4. Generate a CSR using Keytool command.

›_ Console

```
# keytool -certreq -alias utimacoRSAKey -keystore NONE -storetype PKCS11
-providerClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-
java.cfg -file utimacoRSA.csr
```

Here:

- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class
- utimacoRSAKey is the key name
- utimacoRSA.csr is the CSR file name that will be generated

Provide keystore password when prompted

5. Get this CSR signed by CA.

›_ Console

```
# openssl x509 -req -in utimacoRSA.csr -CA ca.crt -CAkey ca.key -CAcreateserial
-out serverRSA.crt -days 365 -sha256
```

6. Copy the signed certificate and the root CA certificate and combine them into a single full-chain certificate file.

›_ Console

```
# cat serverRSA.crt ca.crt > fullchainRSA.crt
```

7. Import converted full chain certificate into HSM keystore.

›_ Console

```
# keytool -importcert -alias utimacoRSAKey -file fullchainRSA.crt -keystore NONE
-storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg /
etc/utimaco/pkcs11-java.cfg
```

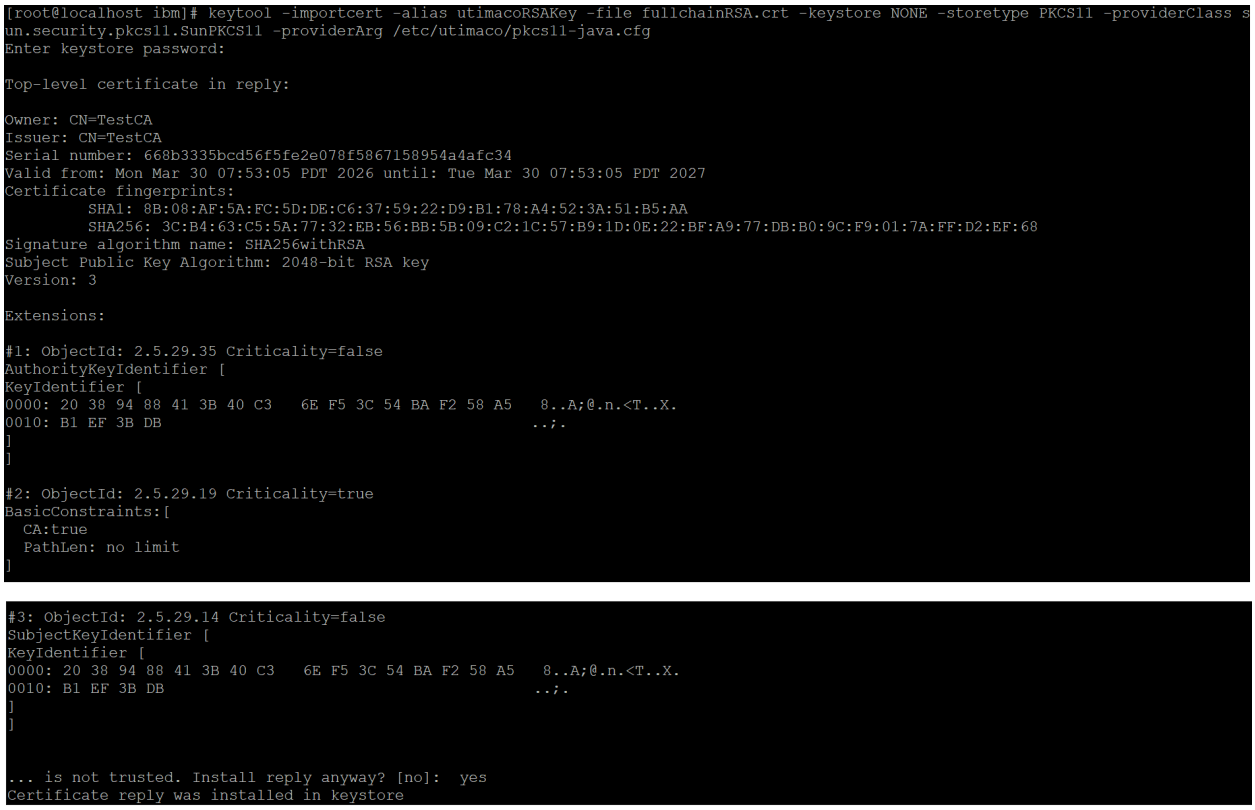


Figure 5 : Importing full chain certificate

8. Sign any sample jar file with jarsigner command.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11
-providerClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-
java.cfg -signedjar HelloWorldRSASigned.jar HelloWorld.jar utimacoRSAKey
```

Here

- http://timestamp.digicert.com is URL of timestamp server
- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class
- HelloWorldRSASigned.jar is the new output signed jar file that will be generated
- HelloWorld.jar is the jar file to be signed
- utimacoRSAKey is the RSA key used for jar signing

Provide the keystore password when prompted:

```
[root@localhost ibm]# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.
SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg -signedjar HelloWorldRSASigned.jar HelloWorld.jar utimacoRSAKey
Enter Passphrase for keystore:
jar signed.
```

Figure 6 : Signing the jar using jarsigner command

9. Verify the signed jar.

›_ Console

```
# jarsigner -verify HelloWorldRSASigned.jar
```

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

```
[root@localhost ibm]# jarsigner -verify HelloWorldRSASigned.jar

jar verified.

Warning:
This jar contains entries whose certificate chain is invalid. Reason: PKIX path building failed: sun.security.provider.certpath.SunCert
PathBuilderException: unable to find valid certification path to requested target

Re-run with the -verbose and -certs options for more details.
```

Figure 7 : Verifying signed jar

6.1.2 With EC Key (CA Signed Certificate)

1. Generate an EC keypair on Utimaco HSM.

›_ Console

```
# keytool -genkeypair -alias utimacoECKey -keyalg EC -groupname secp256r1
-keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11
-providerArg /etc/utimaco/pkcs11-java.cfg
```

Provide information when prompted Here:

- EC is the key algorithm
- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class
- utimacoECKey is the key name that will be generated on Utimaco HSM

Provide the keystore password when prompted

```
[root@localhost ibm]# keytool -genkeypair -alias utimacoECKey -keyalg EC -groupname secp256r1 -keystore NONE -storetype PKCS11 -provide
rClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg
Enter keystore password:
Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the default value in braces
.
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]:  Campbell
What is the name of your State or Province?
  [Unknown]:  California
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=test demo, OU=Security, O=Utimaco, L=Campbell, ST=California, C=US correct?
  [no]:  yes

Generating 256 bit EC key pair and self-signed certificate (SHA384withECDSA) with a validity of 90 days
       for: CN=test demo, OU=Security, O=Utimaco, L=Campbell, ST=California, C=US
```

Figure 8 : Key generation using keytool command

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

›_ Console

```
# keytool -list -keystore NONE -storetype PKCS11 -providerClass
sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg
```

Here:

- NONE is the keystore for HSM
- PKCS11 is the storetype
- PKCS11-CryptoServer is the provider name

Provide the keystore password when prompted

```
[root@localhost ibm]# keytool -list -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/uti
maco/pkcs11-java.cfg
Enter keystore password:
Keystore type: PKCS11
Keystore provider: SunPKCS11-Utimaco

Your keystore contains 1 entry

utimacoECKey, PrivateKeyEntry,
Certificate fingerprint (SHA-256): 51:96:B0:7F:63:51:4E:A4:71:2E:F1:25:D1:AB:2E:CC:85:B4:47:59:0A:2A:26:D6:37:87:3D:B7:92:24:E7:42
```

Figure 9 : Listkeys output

3. List the objects using p11tool2.

›_ Console

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

Enter user PIN when prompted

```
[root@localhost ibm]# cd /opt/utimaco/bin/
[root@localhost bin]# ./p11tool2 Slot=0 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_UNIQUE_ID                 = 659FAB92-FD08-4469-867D-E33EA761D69A
  CKA_LABEL                     = utimacoECKey
  CKA_ID                        =
                0x7574696D 61636F45 434B6579          |utimacoECKey    |

  CKA_SUBJECT                   =
                0x306E310B 30090603 55040613 02555331 |0n1 0   U    US1|
                  13301106 03550408 130A4361 6C69666F | 0   U    Califo|
                  726E6961 3111300F 06035504 07130843 |rnia1 0   U    C|
                  616D7062 656C6C31 10300E06 0355040A |ampbell1 0   U  |
                  13075574 696D6163 6F311130 0F060355 |  Utimaco1 0   U|
                  040B1308 53656375 72697479 31123010 |    Security1 0 |
                  06035504 03130974 65737420 64656D6F |  U    test demo|



CKO_PRIVATE_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = B1996B17-D7BA-44DD-8C43-B6C60A75A7CD
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     =
  CKA_ID                        =
                0x7574696D 61636F45 434B6579          |utimacoECKey    |
```

Figure 10 : List keys output using p11tool2

4. Generate a CSR using Keytool command.

›_ Console

```
# keytool -certreq -alias utimacoECKey -keystore NONE -storetype PKCS11
-providerClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-
java.cfg -file utimacoEC.csr
```

Here:

- NONE is the keystore for HSM
- PKCS11 is the storetype

- Provide the keystore password when prompted
- sun.security.pkcs11.SunPKCS11 is the provider class
- utimacoECKey is the key name
- utimacoEC.csr is the CSR file name that will be generated

Provide the keystore password when prompted

5. Get this CSR signed by CA.

›_ Console

```
# openssl x509 -req -in utimacoEC.csr -CA ca.crt -CAkey ca.key -CAcreateserial
-out serverEC.crt -days 365 -sha256
```

6. Copy the signed certificate and the root CA certificate and combine them into a single full-chain certificate file.

›_ Console

```
# cat serverEC.crt ca.crt > fullchainEC.crt
```

7. Import converted full chain certificate into HSM keystore.

›_ Console

```
#  keytool -importcert -alias utimacoECKey -file fullchainEC.crt -keystore NONE
-storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg /
etc/utimaco/pkcs11-java.cfg
```

```
[root@localhost ibm]# keytool -importcert -alias utimacoECKey -file fullchainEC.crt -keystore NONE -storetype PKCS11 -providerClass sun
.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg
Enter keystore password:

Top-level certificate in reply:

Owner: CN=TestCA
Issuer: CN=TestCA
Serial number: 668b3335bcd56f5fe2e078f5867158954a4afc34
Valid from: Mon Mar 30 07:53:05 PDT 2026 until: Tue Mar 30 07:53:05 PDT 2027
Certificate fingerprints:
         SHA1: 8B:08:AF:5A:FC:5D:DE:C6:37:59:22:D9:B1:78:A4:52:3A:51:B5:AA
         SHA256: 3C:B4:63:C5:5A:77:32:EB:56:BB:5B:09:C2:1C:57:B9:1D:0E:22:BF:A9:77:DB:B0:9C:F9:01:7A:FF:D2:EF:68
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 20 38 94 88 41 3B 40 C3   6E F5 3C 54 BA F2 58 A5   8..A;@.n.<T..X.
0010: B1 EF 3B DB                                        ..;.
]
]

#2: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen: no limit
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 20 38 94 88 41 3B 40 C3   6E F5 3C 54 BA F2 58 A5   8..A;@.n.<T..X.
0010: B1 EF 3B DB                                        ..;.
]
]


... is not trusted. Install reply anyway? [no]:  yes
Certificate reply was installed in keystore
```

Figure 11 : Importing full chain certificate

8. Sign any sample jar file using jarsigner tool.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11
-providerClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-
java.cfg -signedjar HelloWorldECSigned.jar HelloWorld.jar utimacoECKey
```

Here:

- http://timestamp.digicert.com is URL of timestamp server
- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class
- HelloWorldECSigned.jar is the new output signed jar file that will be generated

- HelloWorld.jar is the Jar file to be signed
- utimacoECKey is the key name that will be used for signing

Provide the keystore password when prompted

```
[root@localhost ibm]# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.
SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg -signedjar HelloWorldECSigned.jar HelloWorld.jar utimacoECKey
Enter Passphrase for keystore:
jar signed.

Warning:
The signer's certificate chain is invalid. Reason: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderExceptio
n: unable to find valid certification path to requested target

The signer certificate will expire on 2027-03-30.
The timestamp will expire on 2031-11-09.
```

Figure 12 : Signing the jar using jarsigner command

9. Verify the signed jar.

›_ Console

```
# jarsigner -verify HelloWorldECSigned.jar
```

Here sample_output.jar is the newly generated signed jar file

```
[root@localhost ibm]# jarsigner -verify HelloWorldECSigned.jar

jar verified.

Warning:
This jar contains entries whose certificate chain is invalid. Reason: PKIX path building failed: sun.security.provider.certpath.SunCert
PathBuilderException: unable to find valid certification path to requested target

Re-run with the -verbose and -certs options for more details.
```

Figure 13 : 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 -genkeypair -alias utimacoRSAKey -keyalg RSA -keysize 2048 -keystore
NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11
-providerArg /etc/utimaco/pkcs11-java.cfg
```

Provide information when prompted Here:

- RSA is the key algorithm
- 2048 is the key size
- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class
- utimacoRSAKey is the key name that will be generated on Utimaco HSM

Provide the keystore password when prompted

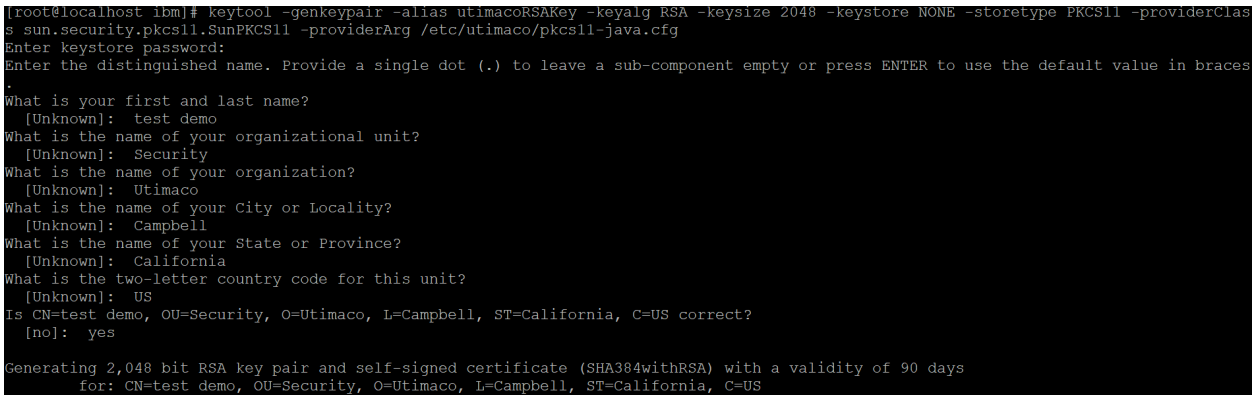


Figure 14 : Key generation using keytool command



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 PKCS11 -providerClass
sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg
```

Here:

- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class

Provide the keystore password when prompted.

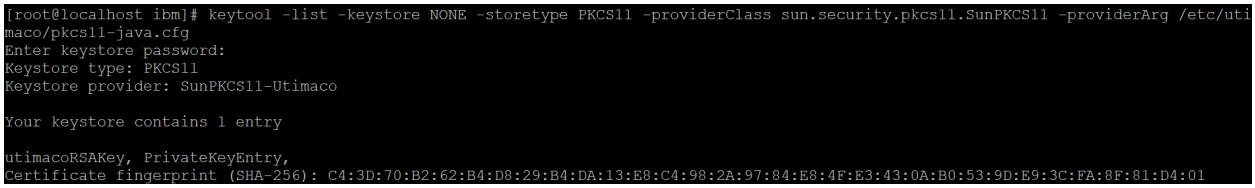


Figure 15 : Listkeys output

3. List the objects using p11tool2.

›_ Console

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

Enter user PIN when prompted.

```
[root@localhost ibm]# cd /opt/utimaco/bin/
[root@localhost bin]# ./p11tool2 Slot=0 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_UNIQUE_ID                 = E40F7E8B-70AE-420E-9139-0B2EDC14A43F
  CKA_LABEL                     = utimacoRSAKey
  CKA_ID                        =
               0x7574696D 61636F52 53414B65 79       |utimacoRSAKey   |

  CKA_SUBJECT                   =
               0x306E310B 30090603 55040613 02555331 |0n1 0   U    US1|
                 13301106 03550408 130A4361 6C69666F | 0   U    Califo|
                 726E6961 3111300F 06035504 07130843 |rnia1 0   U    C|
                 616D7062 656C6C31 10300E06 0355040A |ampbell1 0   U  |
                 13075574 696D6163 6F311130 0F060355 |  Utimaco1 0   U|
                 040B1308 53656375 72697479 31123010 |    Security1 0 |
                 06035504 03130974 65737420 64656D6F |  U    test demo|



CKO_PRIVATE_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 13B810F6-4FFD-4138-8EA7-B10FC395EECC
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     =
  CKA_ID                        =
               0x7574696D 61636F52 53414B65 79       |utimacoRSAKey   |
```

Figure 16 : List keys output using p11tool2

4. Sign any sample jar file with jarsigner command.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11
-providerClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-
java.cfg -signedjar HelloWorldRSASigned.jar HelloWorld.jar utimacoRSAKey
```

Here:

- http://timestamp.digicert.com is URL of timestamp server

- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class
- HelloWorldRSASigned.jar is the new output signed jar file that will be generated
- HelloWorld.jar is the Jar file to be signed
- utimacoRSAKey is the RSA key used for jar signing

Provide the keystore password when prompted.

```
[root@localhost ibm]# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.
SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg -signedjar HelloWorldRSASigned.jar HelloWorld.jar utimacoRSAKey
Enter Passphrase for keystore:
jar signed.

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

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

Figure 17 : Signing the jar using jarsigner command

5. Verify the signed jar.

›_ Console

```
# jarsigner -verify HelloWorldRSASigned.jar
```

Here HelloWorldRSASigned.jar is the newly generated signed jar file.

```
[root@localhost ibm]# jarsigner -verify HelloWorldRSASigned.jar

jar verified.

Warning:
This jar contains entries whose certificate chain is invalid. Reason: PKIX path building failed: sun.security.provider.certpath.SunCert
PathBuilderException: unable to find valid certification path to requested target
This jar contains entries whose signer certificate is self-signed.

Re-run with the -verbose and -certs options for more details.
```

Figure 18 : Verifying signed jar

6.2.2 With EC Key (Self Signed Certificate)

1. Generate an EC keypair on Utimaco HSM.

›_ Console

```
keytool -genkeypair -alias utimacoECKey -keyalg EC -groupname secp256r1
-keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11
-providerArg /etc/utimaco/pkcs11-java.cfg
```

Provide information when prompted Here:

- EC is the key algorithm
- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class
- utimacoECKey is the key name that will be generated on Utimaco HSM

Provide the keystore password when prompted.

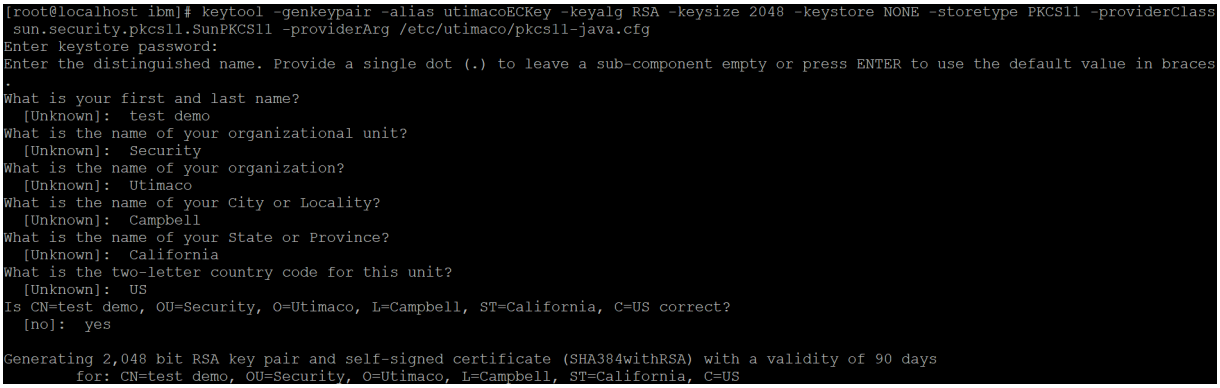


Figure 19 : Keytool command to generate keys



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 PKCS11 -providerClass
sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg
```

Here:

- NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class

Provide the keystore password when prompted.



Figure 20 : Listkeys output

3. List the objects using p11tool2.

›_ Console

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

Enter user PIN when prompted.

```
[root@localhost ibm]# cd /opt/utimaco/bin/
[root@localhost bin]# ./p11tool2 Slot=0 LoginUser=ask ListObjects
Enter normal user PIN:


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_UNIQUE_ID                 = 659FAB92-FD08-4469-867D-E33EA761D69A
  CKA_LABEL                     = utimacoECKey
  CKA_ID                        =
               0x7574696D 61636F45 434B6579          |utimacoECKey    |

  CKA_SUBJECT                   =
               0x306E310B 30090603 55040613 02555331 |0n1 0   U    US1|
                 13301106 03550408 130A4361 6C69666F | 0   U    Califo|
                 726E6961 3111300F 06035504 07130843 |rnia1 0   U    C|
                 616D7062 656C6C31 10300E06 0355040A |ampbell1 0   U  |
                 13075574 696D6163 6F311130 0F060355 |  Utimaco1 0   U|
                 040B1308 53656375 72697479 31123010 |    Security1 0 |
                 06035504 03130974 65737420 64656D6F |  U    test demo|



CKO_PRIVATE_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = B1996B17-D7BA-44DD-8C43-B6C60A75A7CD
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     =
  CKA_ID                        =
               0x7574696D 61636F45 434B6579          |utimacoECKey    |
```

Figure 21 : List keys output using p11tool2

4. Sign any sample jar file using jarsigner tool.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11
-providerClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-
java.cfg -signedjar HelloWorldECSigned.jar HelloWorld.jar utimacoECKey
```

Here:

- http://timestamp.digicert.com is URL of timestamp server
- Here NONE is the keystore for HSM
- PKCS11 is the storetype
- sun.security.pkcs11.SunPKCS11 is the provider class

- HelloWorldECSigned.jar is the new output signed jar file that will be generated
- HelloWorld.jar is the jar file to be signed

```
[root@localhost ibm]# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.
SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg -signedjar HelloWorldECSigned.jar HelloWorld.jar utimacoECKey
Enter Passphrase for keystore:
jar signed.

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

Figure 22 : Signing the jar using jarsigner command

5. Verify the signed jar.

›_ Console

```
# jarsigner -verify HelloWorldECSigned.jar
```

Here HelloWorldECSigned.jar is the newly generated signed jar file.

```
[root@localhost ibm]# jarsigner -verify HelloWorldECSigned.jar

jar verified.

Warning:
This jar contains entries whose certificate chain is invalid. Reason: PKIX path building failed: sun.security.provider.certpath.SunCert
PathBuilderException: unable to find valid certification path to requested target
This jar contains entries whose signer certificate is self-signed.

Re-run with the -verbose and -certs options for more details.
```

Figure 23 : Verifying signed jar

This completes the Integration for 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 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 Straße 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.

9 References

Title/Company	Document No.
utrust_Anchor_LAN_V5_Operating_Manual	2021-0039
utrust_Anchor_PCIe_Operating_Manual	2020-0042

Table 7: References

For more information on PKCS#11 provider, refer the links below:

https://www.ibm.com/docs/en/semeru-runtime-ce-z/11.0.0?topic=security-pkcs11-provider-differences

https://www.ibm.com/support/pages/ibm-semeru-runtime-certified-edition-zos-version-11-general-avail

10 Command Summary

Command	Purpose
`keytool -genkeypair -alias <key name> -keyalg RSA -keysize 2048 -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg / etc/utimaco/pkcs11-java.cfg`	Generate an RSA key pair directly inside the Utimaco HSM using the PKCS#11 provider.
`keytool -genkeypair -alias <key name> -keyalg EC -groupname secp256r1 -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg / etc/utimaco/pkcs11-java.cfg`	Generate an EC key pair directly inside the Utimaco HSM using the PKCS#11 provider.
`keytool -list -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg / etc/utimaco/pkcs11-java.cfg`	List all key entries available in the HSM that are accessible via Java.
`./p11tool2 Slot=0 LoginUser=ask ListObjects`	Verify and list all objects (keys, certificates) present in the HSM.
`keytool -certreq -alias <key name> -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg / etc/utimaco/pkcs11-java.cfg -file <file name>.csr`	Generate a Certificate Signing Request (CSR) for the key stored in the HSM.
`openssl x509 -req -in <file name>.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out serverR<file name>SA.crt -days 365 -sha256`	Sign the CSR using a Certificate Authority (CA) to generate a signed certificate.

Command	Purpose
`cat <file name>.crt ca.crt > <full chain>.crt`	Combine the signed certificate and CA certificate into a full chain certificate file.
`keytool -importcert -alias <key name> -file <full chain>.crt -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg / etc/utimaco/pkcs11-java.cfg`	Import the signed certificate (and chain) into the HSM and associate it with the existing key.
`jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11 -providerArg /etc/utimaco/pkcs11-java.cfg -signedjar <output jar name>.jar <input jar name>.jar <key name>`	Sign the JAR file using the HSM-based private key with timestamping support.
`jarsigner -verify <output jar name>.jar`	Verify the integrity and signature of the signed JAR file.

Table 8: Commands