Oracle

Jarsigner

JDK v.1.8

Integration Guide

CryptoServer HSM

4.45.3.0

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	2026-05-20
Status	PUBLISHED
Document No.	IG-2026-0045

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 web site at https://utimaco.com/.

1.1 About This Guide

This guide describes how to enable HSM integration with Oracle Jarsigner. The instructions in this document have been thoroughly tested and provide a straightforward integration process. There may be other untested ways to achieve interoperability.

This document may not cover every step in the process of setting up all the software. We assume you have read the HSM documentation and that you are familiar with the documentation and setup process.

1.1.1 Target Audience for This Guide

This guide is intended for Oracle Jar signing-in and HSM administrators.

1.1.2 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`

Convention	Use	Example
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.

1.1.3 Abbreviations

The following abbreviations are used in this guide:

Abbreviation	Meaning
CA	Certificate Authority
CSR	Certificate Signing Request
CSADM	CryptoServer Command-line Administration Tool
CSAR	Cloud Service Architecture
GUI	Graphical User Interface

Abbreviation	Meaning
HSM	Hardware Security Module
JAR	Java ARchive
JDK	Java Development Kit
LAN	Local Area Network
MBK	Master Backup Key
PKCS#11	Public-Key Cryptography Standard #11
RSA	Rivest-Shamir-Adleman
URL	Uniform Resource Locator

Table 2: List of Abbreviations

2 Overview

2.1 Oracle Jarsigner

Jarsigner signs and verifies Java Archive (JAR) files. The JAR feature enables the packaging of class files, images, sounds, and other digital data in a single file for faster and easier distribution. The Jarsigner tool is used to sign Java Archive (JAR) files, and also to verify the integrity of the signature on a JAR file. To generate an entity's signature for a file, the entity must first have a public/private key pair associated with it and one or more certificates that authenticate its public key. A certificate is a digitally signed statement from one trusted entity that says that the public key of another entity has a particular value.

The Jarsigner command uses key and certificate information from a keystore to generate digital signatures for JAR files. A keystore is a database of private keys and their associated X.509 certificate chains that authenticate the corresponding public keys. The `keytool` command is used to create and administer keystores. Jarsigner uses an entity's private key to generate a signature, which is then attached to the JAR file. The signed JAR file contains, among other things, a copy of the certificate from the keystore for the public key corresponding to the private key used to sign the file. It can also be used to verify the digital signature of the signed JAR file using the certificate inside it (in its signature block file).

The Jarsigner command can generate signatures that include a time stamp that lets a system or deployer (including Java Plug-in) to check whether the JAR file was signed while the signing certificate was still valid. In addition, APIs allow applications to obtain the timestamp information.

This integration guide covers all the necessary information to install, configure, and integrate Oracle Jarsigner with Utimaco Hardware Security Modules (HSM).

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 Oracle Jarsigner.

Operating System	JDK	Utimaco Security Server Version	Utimaco HSM
CentOS 8.2 Windows 2019	JDK 1.8 JDK 11 JDK 17	SecurityServer V4.45.3.0	CryptoServer CSe-Series/Se-Series u.trust Anchor Se*k and u.trust Anchor CSAR

Table 3: List of Software Requirements

3.2 Software Requirements

Software	Version
HSM Interfaces	CryptoServer PKCS 11 configured
Java SE 8	Oracle Java JDK 1.8.0_321
Java SE 11	Oracle Java JDK 11.0.13
Java SE 17	Oracle Java JDK 17.0.2

Table 4: List of Software Requirements

3.3 Hardware Requirements

Hardware	Hardware Requirements
Utimaco Support Software V4.45.3.0	Utimaco Support Software V4.45.3.0 u.trust Anchor Se*k and u.trust Anchor CSAR with firmware 4.46 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

Please ensure that:

- The operating system used is listed in Tested Versions.
- The SecurityServer used is listed in Tested Versions.
- The CryptoServer Default Admin is replaced with a new admin user with dual-control ("4-eyes") administrators for user and system admin.
- The CryptoServer is set up and configured. Refer to the CryptoServer documentation to set up the HSM.
- You have installed the required version of JDK 1.8/JDK11/JDK17.
- You have the CryptoServer (PCIe or LAN) with MBK loaded.
- The PKCS#11 library is set up and configured as per your environment. Refer to the CryptoServer documentation to set up and configure the PKCS#11 library.
- The `JAVA_HOME` environment variable is set.
- You familiarize yourself with the Oracle Jarsigner documentation and setup process, and have the Utimaco documentation available.

4 PKCS#11 Configuration

4.1 On Linux

1. Create the directory `/etc/utimaco`. We will copy the Utimaco PKCS#11 configuration file `cs_pkcs11_R3.cfg` into this directory. It is located in your CryptoServer-V4.45.3.0 installer directory, `Linux/x86-64/Crypto_APIs/PKCS11_R3/sample`.

›_ 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 located at `/etc/utimaco/` and make the appropriate changes to the file.

As a best practice, cs_pkcs11_R3.cfg should be owned by the user or service group, not by root.

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

4. Copy `pkcs11` library file `libcs_pkcs11_R3.so` from Utimaco CryptoServer software to the `/opt/utimaco/lib` directory and make the file executable.

›_ Console

```
# mkdir -p /opt/utimaco/bin && mkdir /opt/utimaco/lib
# chmod +x /opt/utimaco/lib/libcs_pkcs11_R3.so
```

5. 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
# chmod +x /opt/utimaco/bin/p11tool2
```

6. Create `pkcs11.cfg` at location `/etc/utimaco/`.

›_ Console

```
# touch /etc/utimaco/pkcs11.cfg
```

7. Add the content below to the `pkcs11.cfg`.

›_ pkcs11.cfg

```
name=CryptoServer
library=/opt/utimaco/lib/libcs_pkcs11_R3.so
slot=0
attributes=compatibility
attributes(*,*,*) = {
CKA_TOKEN = true
}
```

4.2 On Windows

On Windows, as part of CryptoServer software installation, `cs_pkcs11_R3.cfg` will get automatically created and will be available under `C:\ProgramData\Utimaco\PKCS11_R3` folder.

Create `pkcs11.cfg` file at location `C:\ProgramData\Utimaco\PKCS11_R3` and add the contents as listed below.

›_ pkcs11.cfg

```
name=CryptoServer
library=C:/Program Files/Utimaco/SecurityServer/Lib/cs_pkcs11_R3.dll slot=0
attributes=compatibility attributes(*,*,*) = { CKA_TOKEN = true
}
```

5 Update Java.security

Update `java.security` file to use Utimaco Security Provider for appropriate Java versions.

The sample content shown may not be true in all cases/installs, as the `java.security` file is site, version and installation dependent.

The pattern is to add a new `security.provider.<index>`, where <index> is one greater than the last found in the file.

5.1 Update java.security file for JDK 8

1. Edit the Java security configuration file `java.security` located in the directory `<JDK_Installation_directory>/Jre/lib/Security`.
2. Update `java.security` file to use Utimaco security provider for JDK 8 as mentioned below.

›_ sample content from java.security

```
security.provider.1=sun.security.provider.Sun
security.provider.2=sun.security.rsa.SunRsaSign
security.provider.3=sun.security.ec.SunEC
security.provider.4=com.sun.net.ssl.internal.ssl.Provider
security.provider.5=com.sun.crypto.provider.SunJCE
security.provider.6=sun.security.jgss.SunProvider
security.provider.7=com.sun.security.sasl.Provider
security.provider.8=org.jcp.xml.dsig.internal.dom.XMLDSigRI
security.provider.9=sun.security.smartcardio.SunPCSC

security.provider.10=sun.security.pkcs11.SunPKCS11

/etc/utimaco/pkcs11.cfg
```

5.2 Update java.security file for JDK11/JDK 17

1. Edit the Java security configuration file `java.security` located in the following directory: `<JDK_Installation_directory>/Conf/Security`.

2. For Java 11, update the `java.security` file to use Utimaco security provider as mentioned below.

›_ sample content from java.security

```
security.provider.1=SUN
security.provider.2=SunRsaSign
security.provider.3=SunEC
security.provider.4=SunJSSE
security.provider.5=SunJCE
security.provider.6=SunJGSS
security.provider.7=SunSASL
security.provider.8=XMLDSig
security.provider.9=SunPCSC
security.provider.10=JdkLDAP
security.provider.11=JdkSASL
security.provider.12=SunPKCS11 /etc/utimaco/pkcs11.cfg
```

For Windows, update `security.provider.12=SunPKCS11 /etc/utimaco/pkcs11.cfg` to `security.provider.12=SunPKCS11 C:/ProgramData/Utimaco/PKCS11_R3/pkcs11.cfg` .

Note that the directory markers are forward slashes, as this will be interpreted by Java, not by Windows.

6 Generate Signing Key and Certificate on Utimaco Keystore

1. Generate a signing key and certificate using Java `keytool` utility. This will generate a key pair on Utimaco HSM.

›_ Console

```
# keytool -genkeypair -alias utimacokey -keyalg RSA -keysize 2048 -sigalg
SHA256withRSA -keystore NONE -storetype PKCS11 -storepass 123456 - providername
SunPKCS11-CryptoServer -dname "CN=Java Code Signing, OU=IT, O=Utimaco, L=Aachen,
ST=NRW, C=DE"
```

2. Verify the private keys onto the Utimaco HSM.

›_ Console

```
# keytool -list -v -keystore NONE -storetype PKCS11 -providername SunPKCS11-
CryptoServer -storepass 123456
Keystore type: PKCS11
Keystore provider: SunPKCS11-CryptoServer Your keystore contains 1 entry
Alias name: utimacokey Entry type: PrivateKeyEntry Certificate chain length: 1
Certificate[1]:
Owner: CN=Java Code Signing, OU=IT, O=Utimaco, L= Aachen, ST= NRW, C=DE
Issuer: CN=Java Code Signing, OU=IT, O=Utimaco, L= Aachen, ST= NRW, C= DE
Serial number: 1a81f667
Valid from: Fri Apr 08 05:28:28 UTC 2022 until: Thu Jul 07 05:28:28
UTC 2022
...
...
```

3. Generate a CSR.

›_ Console

```
# keytool -certreq -alias utimacokey -keystore NONE -sigalg SHA256withRSA
-storetype PKCS11 -storepass 123456 -providername SunPKCS11-CryptoServer
-file certreq.csr
```

4. Submit this CSR file to your Certificate Authority (CA). The CA will provide the signed certificate or certificate chain. Save the file onto the server, in an appropriate location.

5. Optionally import CA certificate into `cacerts` store.

›_ Console

```
# keytool -trustcacerts -importcert -alias rootca -keystore
<JDK_Installation_directory>/Jre/lib/Security/cacerts -file ROOTCA.cer
Enter keystore password:
Certificate already exists in keystore under alias <base> Do you still want to
add it? [no]: yes
Certificate was added to keystore
```

Table 6: For JDK8

›_ Console

```
# keytool -trustcacerts -importcert -alias rootca -keystore
<JDK_Installation_directory>/lib/Security/cacerts -file ROOTCA.cer
Warning: use -cacerts option to access cacerts keystore Enter keystore password:
Certificate already exists in keystore under alias <rootca> Do you still want to
add it? [no]: yes
Certificate was added to keystore
```

Table 7: For JDK11/17

6. Import the Signed Certificate reply.

›_ Console

```
# keytool -import -alias utimacokey -keystore NONE -storetype PKCS11 -
providername SunPKCS11-CryptoServer -storepass 123456 -file
Java_Code_Signing.p7b
Top-level certificate in reply:
Owner: CN=Utimaco-RootCA, DC=utimaco, DC=local Issuer: CN=Utimaco-RootCA,
DC=utimaco, DC=local Serial number: 48064f50a86092854178e482a270b6d3
Valid from: Sun Oct 17 14:39:24 UTC 2021 until: Sat Oct 17 14:49:22
UTC 2026
...
...
... is not trusted. Install reply anyway? [no]: yes Certificate reply was
installed in keystore
```

7. Verify the `keystore` contents.

›_ Console

```
# keytool -list -v -keystore NONE -storetype PKCS11 -providername SunPKCS11-
CryptoServer -storepass 123456
Keystore type: PKCS11
Keystore provider: SunPKCS11-CryptoServer Your keystore contains 1 entry
Alias name: utimacokey Entry type: PrivateKeyEntry Certificate chain length: 2
Certificate[1]:
Owner: CN=Java Code Signing, OU=IT, O=Utimaco, L= Aachen, ST= NRW, C=DE
Issuer: CN=Utimaco-RootCA, DC=utimaco, DC=local Serial number:
2d0000004aeb29a7a265cc264700000000004a
Valid from: Fri Apr 08 05:29:43 UTC 2022 until: Sat Apr 08 05:29:43
UTC 2023
...
...
...
```

7 Sign and Verify the Sample Jar File

1. Sign the sample `jar` file using Utimaco HSM `keystore`.

›_ Console

```
# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storepass 123456
-storetype PKCS11 -providername SunPKCS11-CryptoServer -signedjar

<name_of_signedjar_to_be_generated> <jar_to_be_signed> utimacokey Example

# jarsigner -tsa http://timestamp.digicert.com -keystore NONE -storepass 123456
-storetype PKCS11 -providername SunPKCS11-CryptoServer -signedjar
sample_signed.jar sample.jar utimacokey

jar signed.

The signer certificate will expire on 2023-04-08. The timestamp will expire on
2033-03-14.
```

2. Verify the signed `jar` file.

›_ Console

```
# jarsigner -verify sample_signed.jar -verbose

s   1908 Fri Apr 08 06:07:58 UTC 2022 META-INF/MANIFEST.MF

1989 Fri Apr 08 06:07:58 UTC 2022 META-INF/UTIMACOK.SF

7355 Fri Apr 08 06:07:58 UTC 2022 META-INF/UTIMACOK.RSA

0 Wed Mar 23 18:18:34 UTC 2022 META-INF/

0 Wed Mar 23 18:18:34 UTC 2022 META-INF/maven/

0 Wed Mar 23 18:18:34 UTC 2022 META-INF/maven/com.utimaco/

0 Wed Mar 23 18:18:34 UTC 2022 META-

INF/maven/com.utimaco/utimaco-commons/

0 Wed Mar 23 18:18:32 UTC 2022 com/

0 Wed Mar 23 18:18:32 UTC 2022 com/utimaco/

sm  1426 Thu Mar 03 13:38:18 UTC 2022 META-

INF/maven/com.utimaco/utimaco-commons/pom.xml

sm  64 Wed Mar 23 18:18:34 UTC 2022 META-

INF/maven/com.utimaco/utimaco-commons/pom.properties

s = signature was verified

m = entry is listed in manifest

k = at least one certificate was found in keystore

i = at least one certificate was found in identity scope
- Signed by "CN=Java Code Signing, OU=IT, O=Utimaco, L= Aachen, ST= NRW, C=DE "

Digest algorithm: SHA-256

Signature algorithm: SHA256withRSA, 2048-bit key

Timestamped by "CN=DigiCert Timestamp 2022 - 2, O="DigiCert, Inc.", C=US" on Fri
Apr 08 06:07:58 UTC 2022

Timestamp digest algorithm: SHA-256
```

›_ Console

```
Timestamp signature algorithm: SHA256withRSA, 4096-bit key jar verified.

The signer certificate will expire on 2023-04-08. The timestamp will expire on
2033-03-14.
```

8 Troubleshooting

Error	Diagnosis
Error:NO_DEVICE_AVAILABLE	1. Ensure that the environment variable CS_PKCS11_R3_CFG correctly points to a valid `cs_pkcs11_R3.cfg` file. 2. Verify that the `cs_pkcs11_R3.cfg` file has a valid Device entry, and points to existing CryptoServer instances.
Error: Failed to attach external HSM client library. Please check if you specified the vendor provided PKCS#11 library path correctly	1. Verify whether the correct path to PKCS#11 library path is specified. 2. Verify if the `cs_pkcs11_R3.cfg` file is available under `/etc/utimaco` folder. 3. Verify if the `cs_pkcs11_R3.cfg` file configurations are correct.
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.
Error:Slot 0000 0000: p11cat.P11.getAuthState(Native Method): CS_GetSessionInfo returned Error 0x00000030 (CKR_DEVICE_ERROR)	1. Verify HSM services are up and running. 2. Check the `cs_pkcs11_R3.cfg` file has correct IP entry.

Error	Diagnosis
Keystore generation error keytool error: java.io.IOException: load failed	Verify if `pkcs11.cfg` has been added proper slot entry which we have initialized.
Keytool command thrown ProviderException : Initialization failed keytool error: java.security.ProviderException: Initialization failed :	The ProviderException means that the linking configuration file (`pkcs11.cfg`) is wrong in some way. (`.so/.dll` not found or not accessible, etc). If the environment configuration file (pointed to by `CS_PKCS11_R3_CFG`) has an invalid Device line, (NO_DEVICE_AVAILABLE).

keytool command shows exception as java.security.NoSuchProviderException: no such provider: SunPKCS11-CryptoServer.	1. Make sure `java.security` file located at
	
	`<JDK_Installation_directory>/Jre/lib/Security` has entry for utimaco provider entry: `security.provider.<index>=sun.security.pkcs11.SunPKCS11`
	
	`<path to pkcs11.cfg>`
	
	2. Check that the `pkcs11.cfg` is correctly formatted and syntactically correct.
	
	3. Verify that the `pkcs11.cfg` **name** field is set to "CryptoServer".

Table 8: List of Errors and their Diagnoses

9 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/.

10 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 9: References

11 Contact

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.