Keyfactor SignServer

Integration Guide

CryptoServer HSM

SecurityServer 4.30.0

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	2026-07-29
Status	PUBLISHED
Document No.	IG-2026-0089

Table of Contents

1 Introduction

Thank you for purchasing our CryptoServer security system. We hope you are satisfied with our product. Please do not hesitate to contact us if you have any complaints or comments.

1.1 About this Guide

This guide describes how to enable HSM integration with PrimeKey SignServer, including PrimeKey SignServer installation.

1.2 Target Audience for this Guide

This guide is intended for PrimeKey SignServer administrators and HSM administrators.

1.3 Document Conventions

The following conventions are used in this guide:

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

Table 1: Document conventions

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

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

Here you will find additional notes or supplementary information.

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

1.4 Abbreviations

We use the following abbreviations in this guide:

Abbreviation	Meaning
HSM	Hardware Security Module
PKI	Public Key Infrastructure
TSP	Time-Stamp Protocol
CSP	Cryptographic Service Provider
CNG	Cryptography API: Next Generation
API	Application programming interface
MBK	Master Backup Key
CXI	Cryptographic eXtended Interface

Abbreviation	Meaning
CAT	CryptoServer Administration Tool
TLS	Transport Layer Security
TS-SO	Security Officer for the TimestampServer
CLI	Command Line Interface
CSR	Certificate Signing Request
CA	Certificate Authority

Table 2: List of abbreviations

2 Overview

SignServer is a server-side application, capable of creating digital signatures. It supports different types and formats of digital signatures through extendable architecture. Operations regarding/ related to ALI Cryptographical operations cryptography are typically carried out in a Hardware Security Module (HSM) or using software key stores. The organization can keep track of all usages of the signing keys.

CryptoServer is a hardware security module, developed by Utimaco IS GmbH. It 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 Prerequisites and Requirements

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

3.1 Software Requirements

Software	Software Requirements
Application Server	JBoss EAP 7.0-7.2 WildFly 9, 10, 11, 14
Deployment tool	Apache Ant 1.9 or later
Java	OpenJDK 8
Database	MariaDB 5.5/10 MySQL 5.5 PostgreSQL 9 Oracle Database 10/11g

Table 3: List of software requirements

3.2 Hardware Requirements

Hardware	Hardware Requirements
Utimaco LAN HSM	CryptoServer CSe-Series/Se-Series LAN with firmware SecurityServer 4.30.0

Hardware	Hardware Requirements
Utimaco PCI-e HSM	CryptoServer CSe-Series/Se-Series PCI-e with firmware SecurityServer 4.30.0

Table 4: List of hardware requirements

4 Installing SignServer Prerequisites

4.1 Installing OpenJDK 8

The command below will download and install OpenJDK 8.

›_ Console

```
# sudo yum install java-1.8.0-openjdk-devel
```

4.2 Installing and Setting up Apache Ant

Use the one provided by your Linux distribution or download it from the Apache Ant website.

The command below will download and install the Apache Ant 1.9.x application.

›_ Console

```
# sudo yum install ant
```

4.3 Setting up a Database

To install MariaDB in CentOS, run the following:

›_ Console

```
# sudo yum install mariadb mariadb-server
```

4.3.1 Creating a Database Account

Create a database and a database user for a SignServer. Example:

›_ Console

```
# mysql -u root

CREATE DATABASE signserver;

GRANT ALL PRIVILEGES ON signserver.* TO signserver@localhost IDENTIFIED BY
'signserver';
```

4.3.2 Creating Tables and Indexes

The application server will attempt to create tables during the startup of a SignServer and if the database user does not have the permissions to create tables, the tables should be created manually. For information on database schemas for different databases, refer to `doc/sql-scripts/create-tables-signserver-*.sql`.

To improve the performance, the appropriate database indexes should also be applied. This is extremely important for the AuditRecordData table, otherwise the SignServer is much slower at the start. Refer to the `doc/sql-scripts/create-index-signserver.sql`.

4.3.3 Configuring MariaDB Binlog Format

For MariaDB, set the `binlog_format` to "row", for instance in the `/etc/mysql/my.cnf, /etc/my.cnf`, `/etc/my.cnf.d/mariadb-server.cnf` set:

›_ Console

```
# binlog_format=row
```

4.4 Installing Application Server

To install WildFly 14 in CentOS and start it, run the following command:

›_ Console

```
# unzip wildfly-14.0.1.Final.zip cd wildfly-14.0.1.Final ./bin/standalone.sh
```

4.4.1 Configure Web Server Keystores

To configure the Web Server TLS keystore, copy the keystore and truststore files to your application server keystore folder. The following displays the WildFly keystore folder location:

`WILDFLY_HOME/standalone/configuration/keystore/keystore.jks`

`WILDFLY_HOME/standalone/configuration/keystore/truststore.jks`

For testing purposes, you can use the provided test keystore (`dss10_demo-tls.jks`) and truststore (`dss10_truststore.jks`) provided in the `res/test/dss10/` folder in the SignServer release.

4.4.2 Configuring TLS and HTTP

The port 8080 is used for a regular HTTP traffic, the port 8442 for HTTPS with server authentication and the port 8443 for HTTPS with both client and server authentication.

1. Start JBoss with the CLI command:

›_ Console

```
# APPSRV_HOME/bin/jboss-cli.sh -c
```

2. To remove any existing TLS and HTTP configuration and enable configuration of the port 8443, run the following commands:

›_ Console

```
# /subsystem=undertow/server=default-server/http-listener=default:remove
# /subsystem=undertow/server=default-server/https-listener=https:remove
# /socket-binding-group=standard-sockets/socket-binding=http:remove
# /socket-binding-group=standard-sockets/socket-binding=https:remove # :reload
```

3. Configure interfaces by using the appropriate bind-address: e.g. this example uses bindaddress to makeaccess available for anyone:

›_ Console

```
# /interface=http:add(inet-address="0.0.0.0")

# /interface=httpspub:add(inet-address="0.0.0.0")

# /interface=httpspriv:add(inet-address="0.0.0.0")
```

Note that WildFlys default size for an HTTP post size limit is 10 MB. To allow signing larger files, increase the limits on the HTTP/HTTPS listeners by using the max-post-size attribute in the following code examples.

For Wildfly 10: If a larger limit than the default value of 10 MB is specified for the HTTPS listeners, check that the max-post-size value is updated in the standalone.xml file after running the CLI command. If the value was not updated in the XML file, stop the application server and update the max-post-size value in the standalone.xml file manually, before starting the application server again.

4. Configure the HTTPS `httpspriv` listener and set up a private port, that requires a client certificate. Use the appropriate values for the key-alias (hostname), password (keystore password), ca-certificate-password (trust store password) and the supported protocols.

Use the enable-http2="false" to avoid error messages in the log.

›_ Console

```
# /core-service=management/security-realm=SSLRealm:add()

#/core-service=management/security-realm=SSLRealm/server-
identity=ssl:add(keystorepath="keystore/keystore.jks", keystore-relative-
to="jboss.server.config.dir", keystorepassword="serverpwd", alias="localhost")
:reload
#/core-service=management/security-realm=SSLRealm/
authentication=truststore:add(keystore
path="keystore/truststore.jks", keystore-relative-to="jboss.server.config.dir",
keystore
password="changeit")
:reload
#/socket-binding-group=standard-sockets/socket-
binding=httpspriv:add(port="8443",interface="httpspriv")
#/subsystem=undertow/server=default-server/https-listener=httpspriv:add(socket-
binding="httpspriv",
security-realm="SSLRealm", verify-client=REQUIRED, max-post-size="10485760",
enable-http2="true")
```

5. Configure the default HTTP listener:

›_ Console

```
# /socket-binding-group=standard-sockets/socket-
binding=http:add(port="8080",interface="http")

#/subsystem=undertow/server=default-server/http-listener=default:add(socket-
binding=http, max-postsize="10485760", enable-http2="true")

#/subsystem=undertow/server=default-server/http-listener=default:write-
attribute(name=redirect-socket, value="httpspriv")

:reload
```

6. Configure the HTTPS `httpspub` listener and set up the public SSL port that does not require a client certificate:

›_ Console

```
# /socket-binding-group=standard-sockets/socket-
binding=httpspub:add(port="8442",interface="httpspub")

# /subsystem=undertow/server=default-server/https-listener=httpspub:add(socket-
binding="httpspub", security-realm="SSLRealm", max-post-size="10485760", enable-
http2="true") # :reload
```

7. Configure the remoting HTTP `listener` and secure the CLI by preventing the HTTP-remotingconnector from using the HTTP port and use a separate port 4447 instead:

›_ Console

```
# /subsystem=remoting/http-connector=http-remoting-connector:remove

#/subsystem=remoting/http-connector=http-remoting-connector:add(connector-
ref="remoting",securityrealm="ApplicationRealm")

#/socket-binding-group=standard-sockets/socket-binding=remoting:add(port="4447")
#/subsystem=undertow/server=default-server/http-listener=remoting:add(socket-
binding=remoting, maxpost-size="10485760", enable-http2="true")
```

4.4.3 WSDL Location

To configure the WSDL location, run:

›_ Console

```
#/subsystem=webservices:write-attribute(name=wsdl-host,
value=jbossws.undefined.host)

#/subsystem=webservices:write-attribute(name=modify-wsdl-address,
value=true) :reload
```

4.4.4 URL Encoding

Run the following commands:

›_ Console

```
# /system-property=org.apache.catalina.connector.URI_ENCODING:remove()

# /system-property=org.apache.catalina.connector.URI_ENCODING:add(value=UTF-8)

# /system-
property=org.apache.catalina.connector.USE_BODY_ENCODING_FOR_QUERY_STRING:remove
()

# /system-

property=org.apache.catalina.connector.USE_BODY_ENCODING_FOR_QUERY_STRING:add(va
lue=true) # :reload
```

4.5 Configuring the Database

This step is required only if the SingServer runs with a database management system.

4.5.1 Configuring the Database Driver

Add the MariaDB database driver by hot-deploying it into the deployment directory. The driver will be picked up by WildFly and deployed, allowing the creation of the data source in the next step. You can use a generic name, without a version number to get the generic `driver-name` for the data source command.

›_ Console

```
# cp mariadb-java-client-2.1.0.jar APPSRV_HOME/standalone/deployments/mariadb-
java-client.jar
```

4.5.2 Configuring the Data Source

To add a data source for SignServer to use, run the command below with JBoss CLI (by using `APPSRV_HOME/bin/jboss-cli.sh`).

›_ Console

```
#data-source add --name=signserverds --driver-name="mariadb-java-client.jar" --
connection-

url="jdbc:mysql://127.0.0.1:3306/signserver" --jndi-name="java:/SignServerDS" --
use-ccm=true --driverclass="org.mariadb.jdbc.Driver" --user-name="signserver" --
password="signserver" --validate-onmatch=true --background-validation=false --
prepared-statements-cache-size=50 --share-preparedstatements=true --min-pool-
size=5 --max-pool-size=150 --pool-prefill=true --transaction-

isolation=TRANSACTION_READ_COMMITTED --check-valid-connection-sql="select 1;" --
enabled=true :reload
```

The MariaDB data source example configuration uses the driver deployed in the previous step *Configuring the Database Driver*. The `--jindi-name` above is the default database.properties value.

5 Installing SignServer

Please, follow instructions in this chapter for the server-side installation of SignServer.

5.1 Download and Unpack SignServer

Download and unzip the latest SignServer Enterprise Edition from your PrimeKey download area or use the latest *SignServer Community Edition* release archive from SourceForge.

The available distributions are:

- signserver-5.x.y-bin.zip (recommended binary distribution).
- signserver-5.x.y.zip (mixed distribution that contains the sources and the required libraries. Build SignServer before the deployment).
- signserver-5.x.y-src.tar.gz: This is a source-only tarball distribution. First, gather all the dependencies and then build it.

Compare the checksums as provided on the website: *https://signserver.org/download.html* or from the download site that PrimeKey provided.

›_ Console

```
# sha256sum signserver-5.x.y-bin.zip

# unzip signserver-5.x.y-bin.zip
```

Optionally, to build SingServer please refer to the *SignServer Installation*.

5.2 Set Environment Variables

Set `APPSRV_HOME` to point to your application server installation. For example:

›_ Console

```
# export APPSRV_HOME=/opt/wildfly-14.0.1.Final
```

The `APPSRV_HOME` variable is used when deploying to the application server and could, for example, be set in your `.bashrc` or similar file, or be provided every time the deploy command is executed.

To set `SIGNSERVER_NODEID` to a unique ID for the server, use:

›_ Console

```
# export SIGNSERVER_NODEID=node1
```

The `SIGNSERVER_NODEID` variable should be available to the application server and might need to be set in the `/etc/environment` or similar. The variable is generally not mandatory. If not set, warnings are printed in the log.

5.3 Configuring SignServer Deployment

The file `signserver_deploy.properties` includes configuration settings for the application, database and for the web services.

Copy the `conf/signserver_deploy.properties.sample` to the `conf/signserver_deploy.properties` and open it for editing in a text editor.

›_ Console

```
# cp conf/signserver_deploy.properties.sample conf/signserver_deploy.properties
```

5.4 Deploying SignServer

To build the configuration and deploy it to the selected application server, execute the command below:

›_ Console

```
# bin/ant deploy
```

5.5 Verifying Installation and Accessing SingServer

For verifications, access one of the available user interfaces.

5.5.1 Administration CLI

To test the access to the server and print the deployed version, run the following Admin CLI command:

›_ Console

```
# bin/signserver getstatus brief all
```

Example of the command output:

Current version of server is: SignServer EE 4.0.0

5.5.2 Web Interface

Point your web browser to http://localhost:8080/signserver for demo web pages, administration and local documentation.

To access the SignServer Admin Web, point your web browser tohttp://localhost:8080/signserver/adminweb.

To temporarily allow all valid client certificates to administer the Administration Web, run the following:

›_ Console

```
# bin/signserver wsadmins -allowany
```

6 Configuring PKCS#11

6.1 Introduction and Prerequisites

Before the PKCS#11 interface and library can be used, some manual actions must be performed. Follow the steps below to configure the PKCS#11 library and initialize a PKCS#11 slot. For further information about the PKCS#11 setup, please refer to [CSPKCSDG].

6.2 Installing PKCS#11 on the Workstation

The installation file for the PKCS#11 is located on the Product CD. The installer creates an environment variable called `CS_PKCS11_R2_CFG`. It contains the path to Utimaco's PKCS#11 configuration file. By default, we must copy the file to `/etc/utimaco` in Linux. We must set the `CS_PKCS11_R2_CFG` environment variable to point to this location.

To be able to access the HSM via PKCS#11, the configuration file needs to be modified.

1. Set the path to the logfile and set the desired log level.

```
[Global]
# For unix:
Logpath = /tmp
# For windows:
# Logpath = C:/ProgramData/Utimaco/PKCS11_R2
# Loglevel (0 = NONE; 1 = ERROR; 2 = WARNING; 3 = INFO; 4 = TRACE)
Logging = 4
```

2. Set the IP address of the HSM.

```
[CryptoServer]
# Device specifier (here: CryptoServer is CSLAN with IP address 127.0.0.1)
Device = 127.0.0.1
```

3. Optionally, make additional modifications to the configuration file, such as setting up an external store as described in [CSPKCSM]. We suggest modifying the PKCS#11 config file to `KeepAlive` flag active.

›_ Console

```
[Global]

# Prevents expiring session after inactivity of 15 minutes
KeepAlive = true
```

6.3 Generating an MBK

One of the steps of the HSM initialization is generating a new MBK, which can be used for creating backups, for using an external storage and for synchronizing HSM clusters. By default, MBK is an AES256 key, though it is also possible to generate a DES MBK by using the csadm tool for backward compatibility reasons.

It is required to generate an MBK for the HSM to become operational. Without an MBK no cryptographic operations can be run.

To generate an MBK:

1. Open the Crypto Administration Tool.
2. Achieve the permission level of at least **02000000**.
3. Click **Manage MBK** to access the **Remote Master Backup Key Management** window and select the **Generate** tab.
4. Type the name of the MBK in the MBK Name section.
5. Select the backup mode of the MBK shares as either XOR or m out of n.
   - If m out of n was selected it is necessary to select the number of m (shares) and n (shares) by using the drop-down menus, set by default as 2 and 3.
6. In case this MBK also needs to be imported at the same time into the HSM, select the Automatic MBK Import option.

7. Click **Generate**.
8. Select whether the MBK shares should be saved on smartcards or as keyfiles by selecting either the Smartcard Token or the Keyfile Token option.

- If you chose to export the MBK shares on smartcards, follow the instructions on the smart card reader to export all of the m parts.

6.4 Importing the MBK

In case the MBK was not imported, when it was generated, or if you want to upload it to another HSM, you need to import it from the keyfiles or smartcards that carry its parts. The MBK was divided into multiple parts by using Shamir's Secret Sharing or XOR and can be restored by using the "m out of n" or XOR principle.

1. Make sure that at least m out of n smart cards/keyfiles are available.
2. Open the CryptoServer Administration Tool.
3. Achieve the permission level of at least **02000000**.
4. Click **Manage MBK** to access the **Remote Master Backup Key Management** window and select the **Import** tab.
5. Select the type of MBK that will be imported and the value m.
6. Click **Import**.
7. Select whether to save the MBK parts on smartcards or as keyfiles by selecting either **Smartcard Token** or **Keyfile Token** option.
8. Follow the instructions to import all the m parts.

6.5 Initializing PKCS#11 on the HSM

In addition to PKCS#11, the PKCS#11 graphical interface tool (P11CAT) and the PKCS#11 command line interface (p11tool2) are installed as well. This chapter shows how to use the P11CAT to initialize the PKCS#11 Slot 0. There are 10 active PKCS#11 slots by default. The number of PKCS#11 slots can be modified in the PKCS#11 configuration file.

1. Make sure that the PKCS#11 configuration file contains the IP address of your HSM and that the HSM is running.

2. Open the P11CAT tool on your workstation. When opening the tool for the first time, the slots should be as shown in Figure 3.

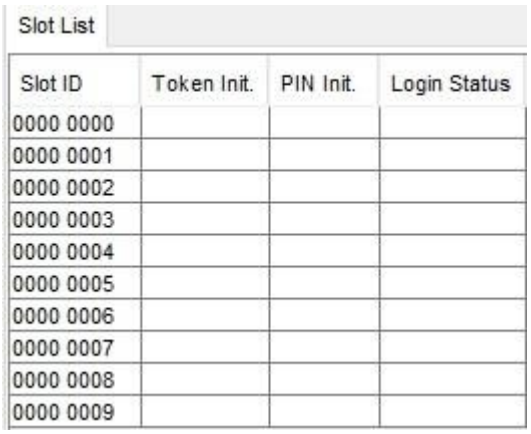

Figure 1 : PKCS#11 slots before initialization

3. Select the row **0000 0000** under the **Slot ID** on the top left-hand side in the **Slot List**.
4. Click on **Login/Logou**t.
5. Click on **Login Generic**.
6. Log in as a user with the permission mask at least **20000000** (User Manager permission).
7. Click on **Slot Management**.
8. Create a Security Officer (SO) for Slot 0. Click on **Init Token**. Write the **Token Label**. Set the **SO PIN**. Confirm the **SO PIN**. Click on **Init Token**. Observe the changed **Token Init**. status for Slot 0.
9. Logout the ADMIN user. Click on **Login/Logout**. Click on **Logout All**.
10. Login as the **SO**. Click on **Login/Logout**. Click on **Login SO**. Enter the **SO PIN**. Click on **Login**.
11. Click on **Slot Management**.
12. Create the User for Slot 0. Select **Init PIN**. Enter the **Normal User PIN**. Confirm the **Normal User PIN**. Click on **Init PIN**. Observe the changed PIN Init. status for Slot 0.
13. Logout the SO. Click on **Login/Logout**. Click on **Logout All**.

The Slot 0 is now initialized. An application or a user can now connect to the PKCS#11 Slot 0 and create or store objects on the slot. Find further information on creating or deleting objects and users in [CSPKCSM] and [LPKCSHD].

7 Setting up SignServer Workers

7.1 Setting up a Sample HSM Crypto Token

To create an HSM (PKCS#11) Crypto Worker, please follow the steps below:

1. Open the *SignServer Administration Web*.

Figure 2 : SingServer web

2. Click on **Workers** in the top menu.

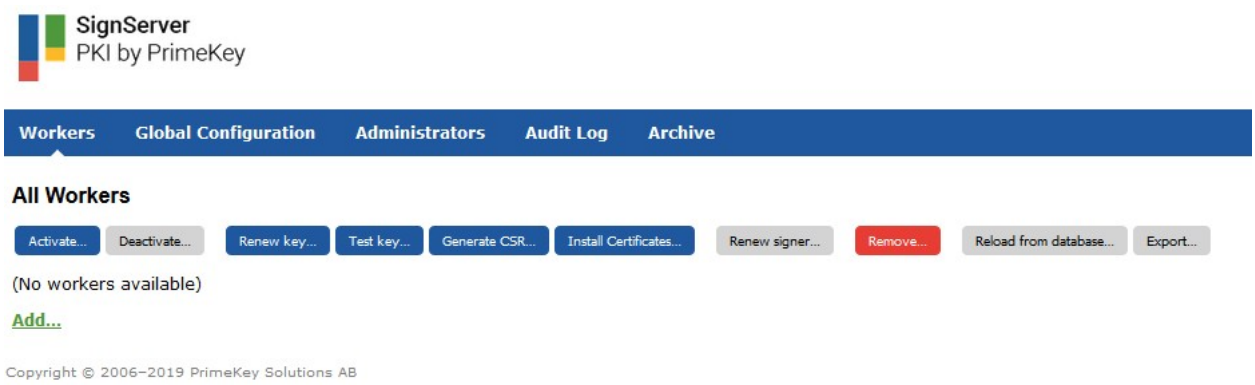

Figure 3 : Signserver Workers page

3. Click on **Add**… below the workers list.
4. Click on **From Template**.

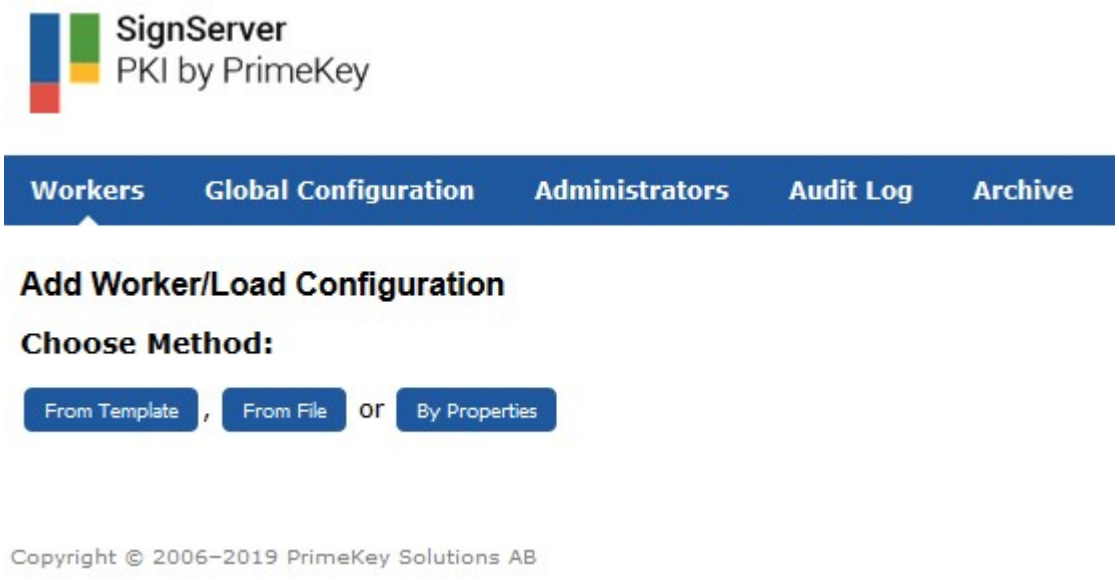

Figure 4 : Signserver Add Worker

5. Select the pkcs11-crypto.proprties template and click on **Next**.

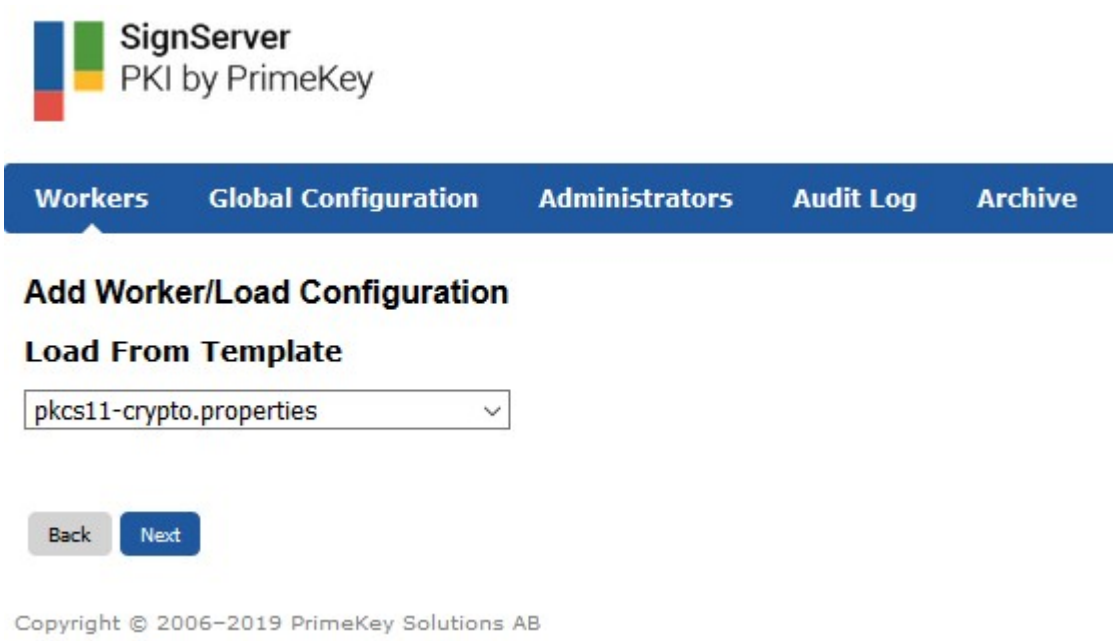

Figure 5 : Signserver selecting a template

6. In the configuration text area, modify the property `WORKERGENID1.LIBRARYNAME` to `WORKERGENID1.SHAREDLIBRARYNAME=Utimaco`. If required, the values for slot numbers can be edited to correspond to the configured slot in the HSM.

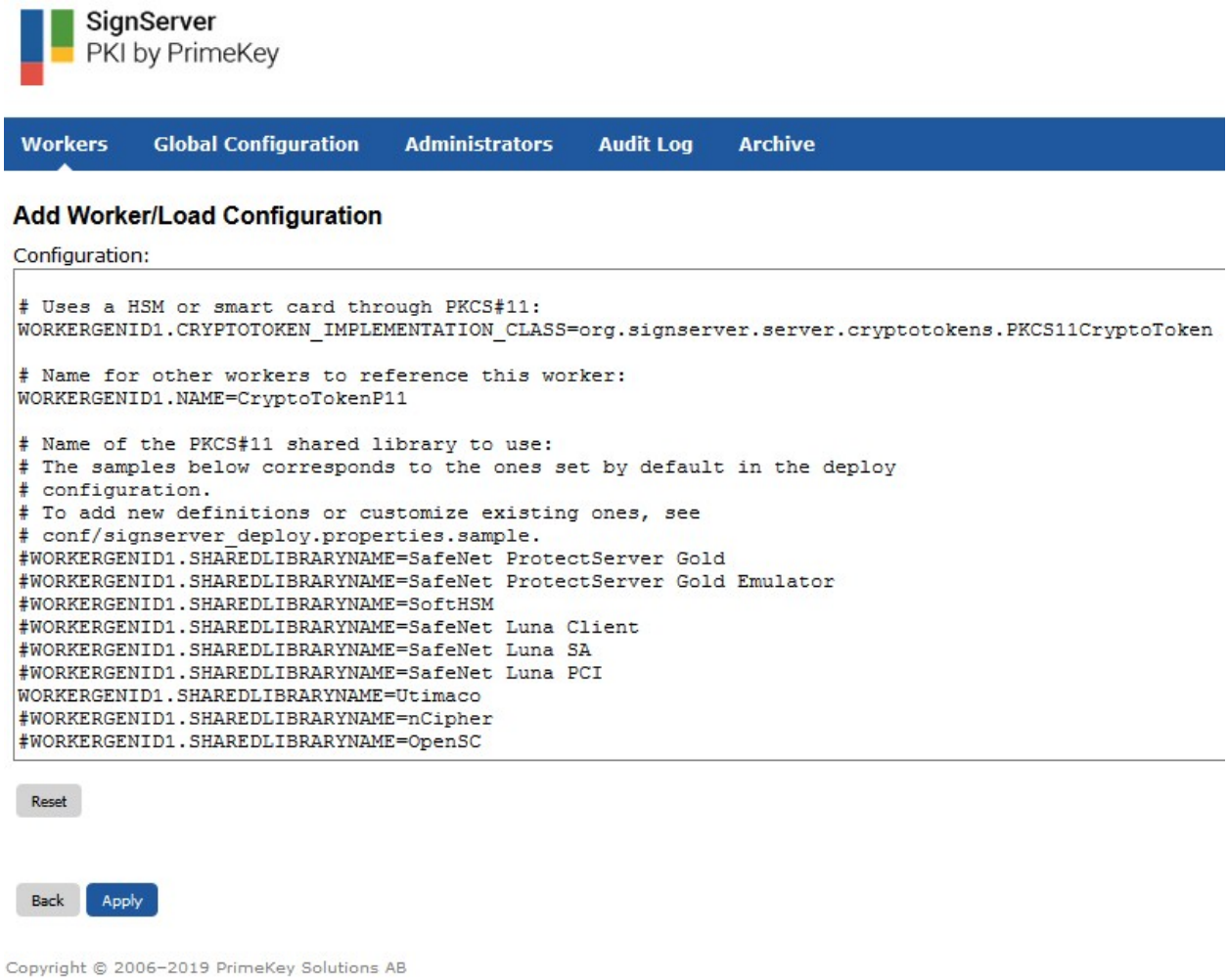

Figure 6 : Signserver Workers configuration example

7. Click on **Apply**.

7.1.1 Generating a New Key-pair

1. Click **Renew key...**

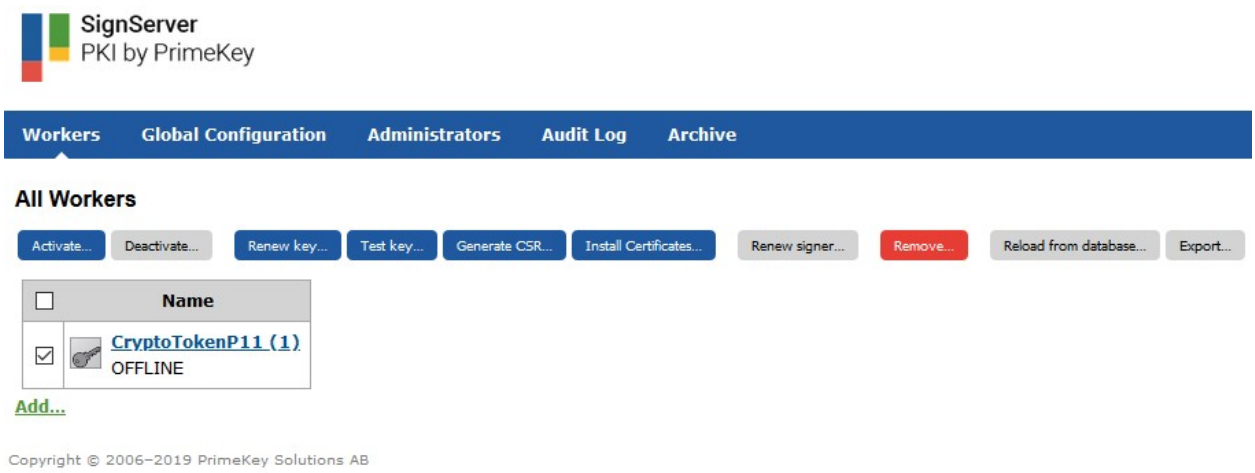

Figure 7 : Selecting a worker

2. Enter the key algorithms (for example RSA or ECDSA and a suitable key specification, i.e. 2048 for RSA, or prime256v1 for ECDSA) and a new key alias for the key.

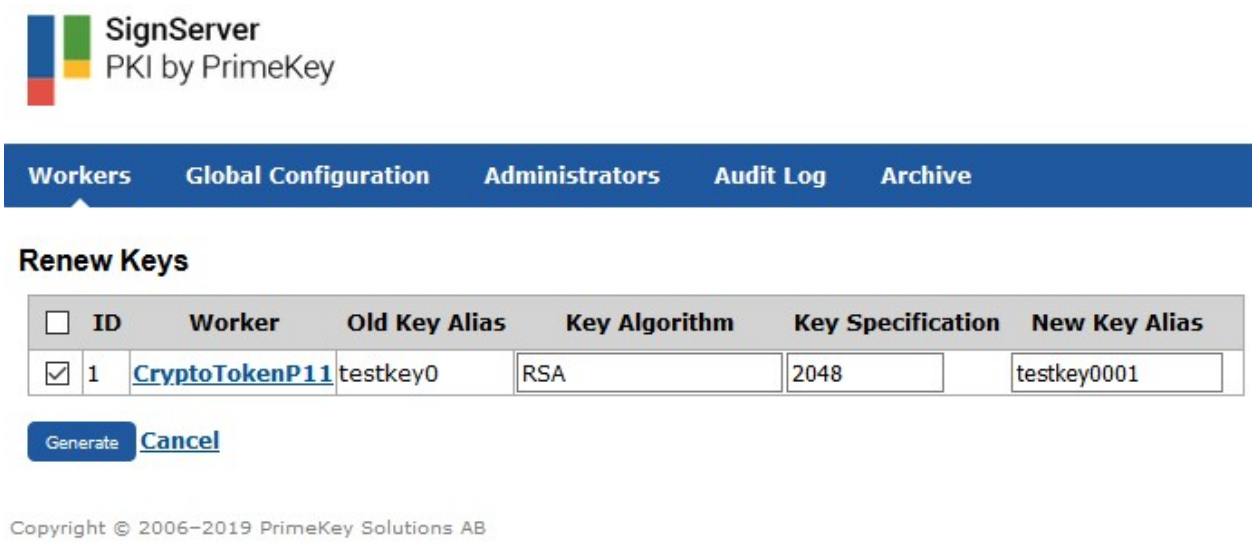

Figure 8 : Signserver renew keys

3. Click on **Generate**.

7.1.2 Generating Certificate Signing Request (CSR) for a Signer

1. Click **Generate CSR** and enter the key alias of the newly generated key (See Figure 7).
2. Click the **<** button to enter the key alias.

3. Enter the signature algorithm, for example "SHA256withRSA" and a distinguished name (DN) for signing certificate (for example `CN=testsigner`).

4. Click **Generate**, click **Download** below the result and then save the resulting CSR as a `p10` file.

Figure 9 : Signserver Generate CSR

5. Issue a signer certificate for your new signer by using your CA and this CSR.

Ensure you to issue an appropriate certificate when setting up a time-stamp signer or code signer by using the correct certificate extensions.

7.2 Set up a Sample Signer Using an HSM Crypto Token

To set up a sample signer by using an HSM crypto token, do the following:

1. Click on **Add** below the workers' list (See Figure 3).

2. Click on **From Template**, select pdfsigner.properties in the list menu, and click on **Next**.

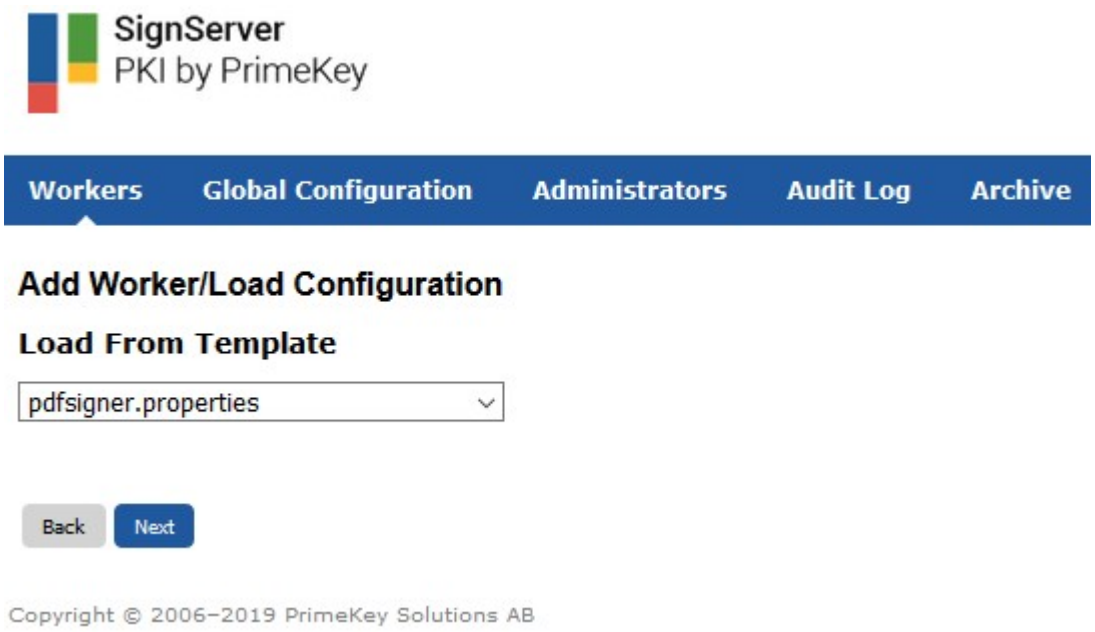

Figure 10 : PdfSigner Template example

3. Edit the signer settings in the configuration text area and change the `WORKERGENID1.CRYPTOTOKEN` setting to use the commented-out sample by using the PKCS#11 crypto token, `CryptoTokenP11`, to match the crypto token set up.

4. Click on **Apply**.

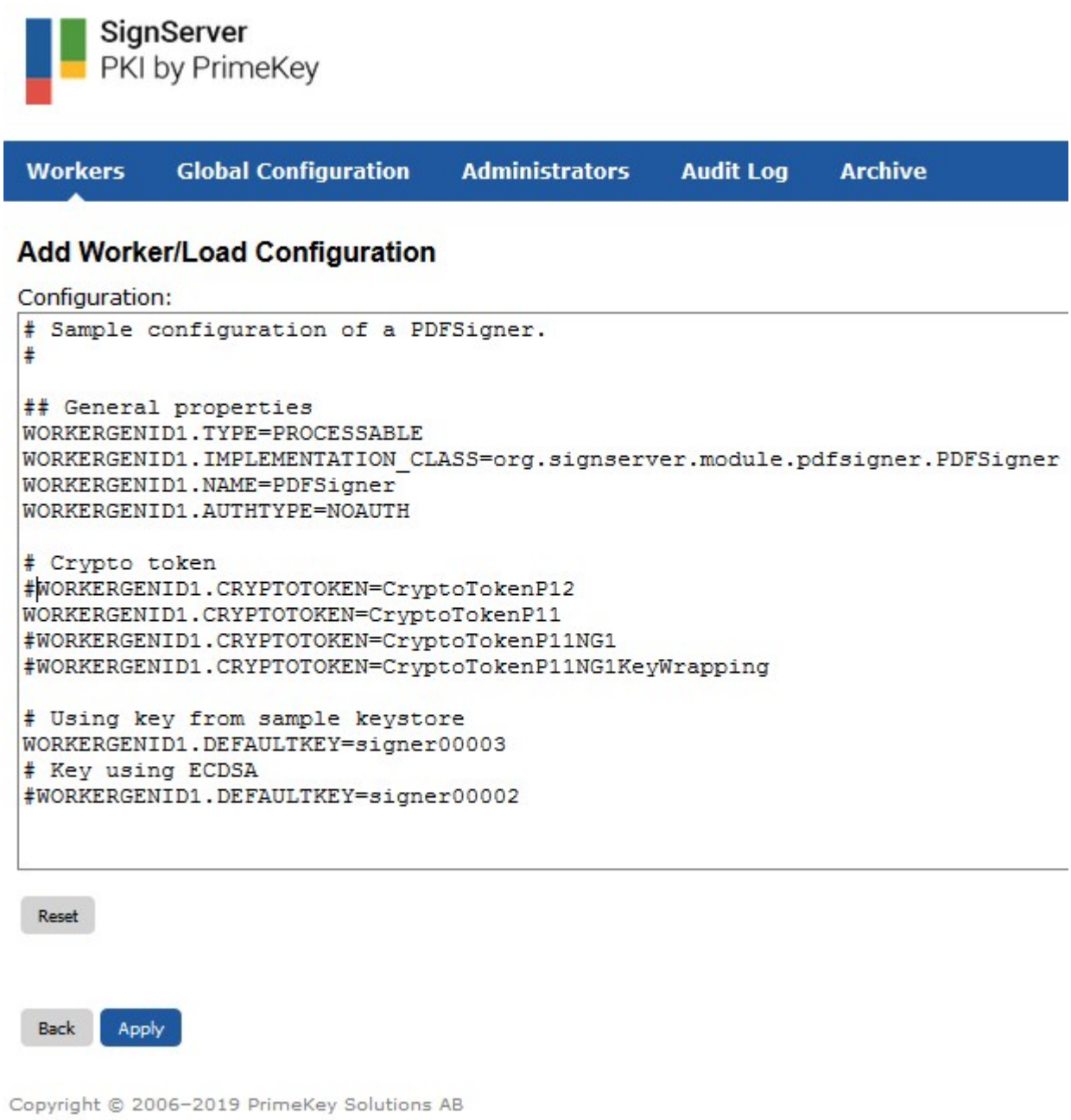

Figure 11 : Signserver PdfSigner Configuration example

5. Set the `DEFAULTKEY` worker property by clicking on the new signer, click on **Configuration** and then click the **Edit** link in the table row for the `DEFAULTKEY` property.

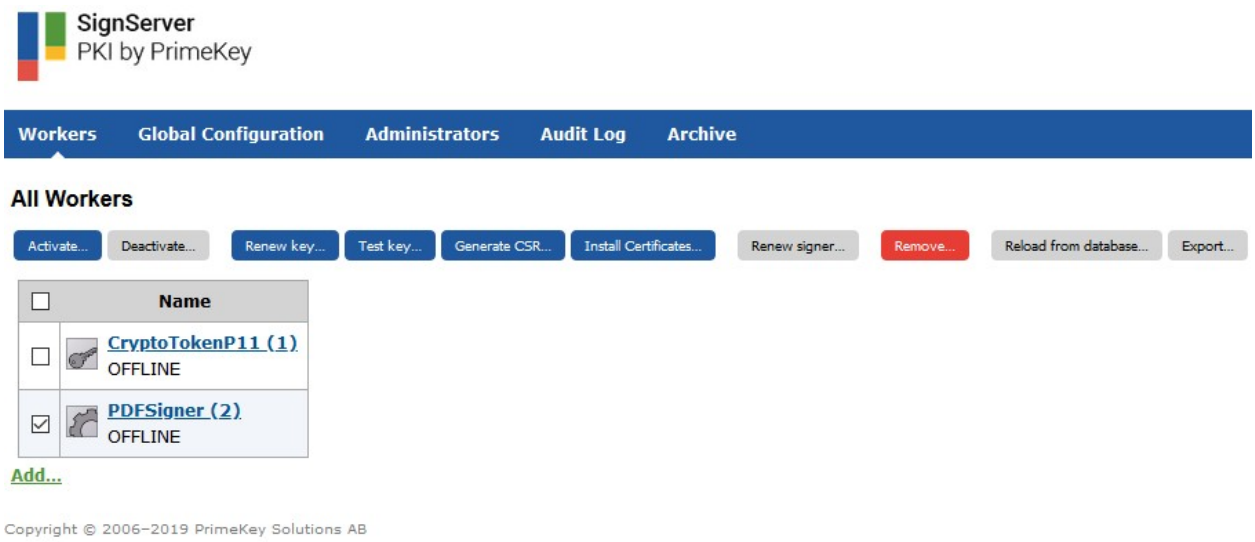

Figure 12 : Signserver select PdfSigner

6. Enter the key alias for the newly generated key in the HSM into the Value text area and click on **Submit**.

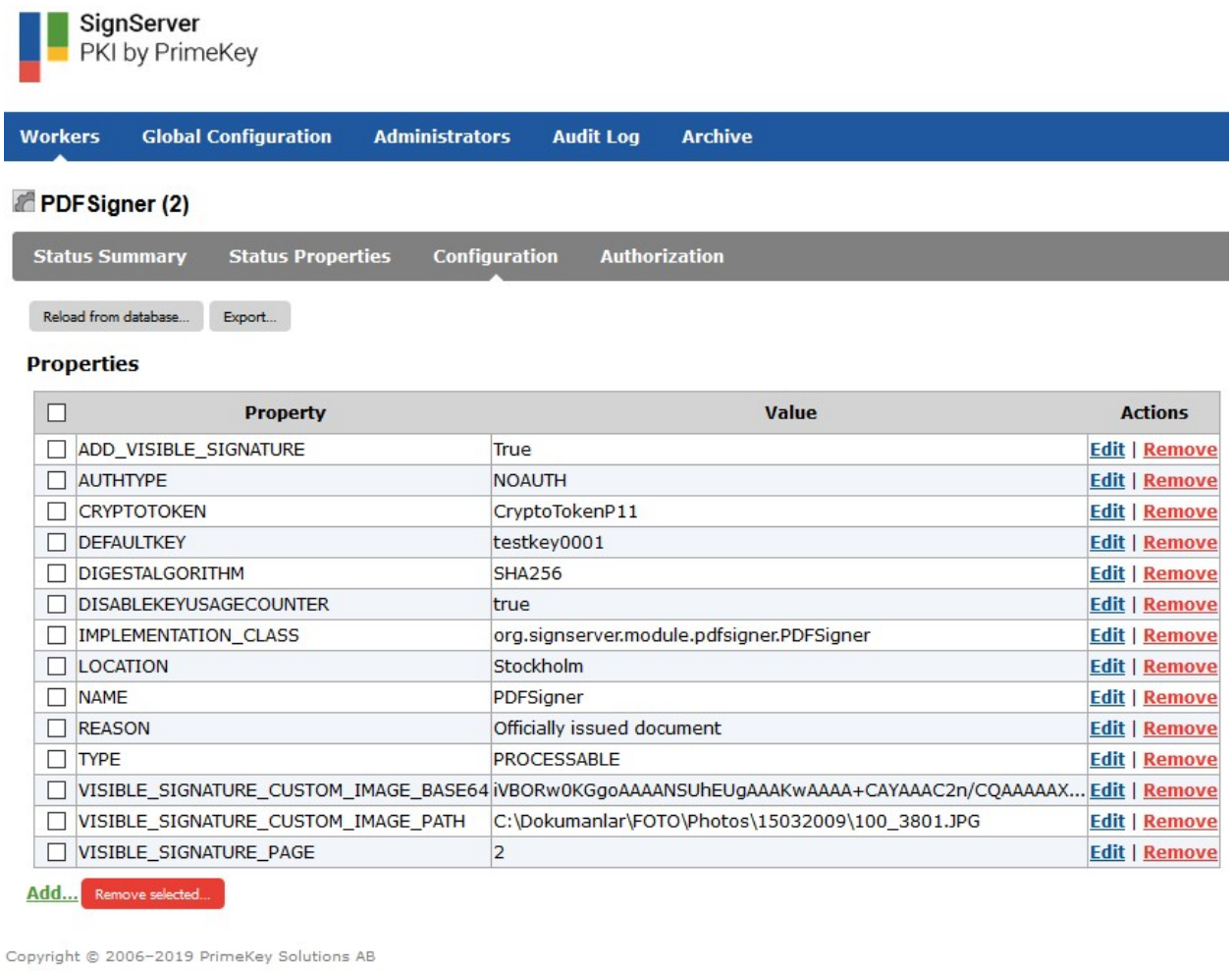

Figure 13 : PdfSigner configuration example

7. Install the signer certificate chain as issued by your CA:
    a. Click the link to your PKCS#11 crypto worker in the workers' list and click on **Install certificates**.
8. Click the **>** button to select your key generated previously.
9. Click on **Browse** and select your issued certificate chain.
10. Select **Install** in token and click on **Install**.

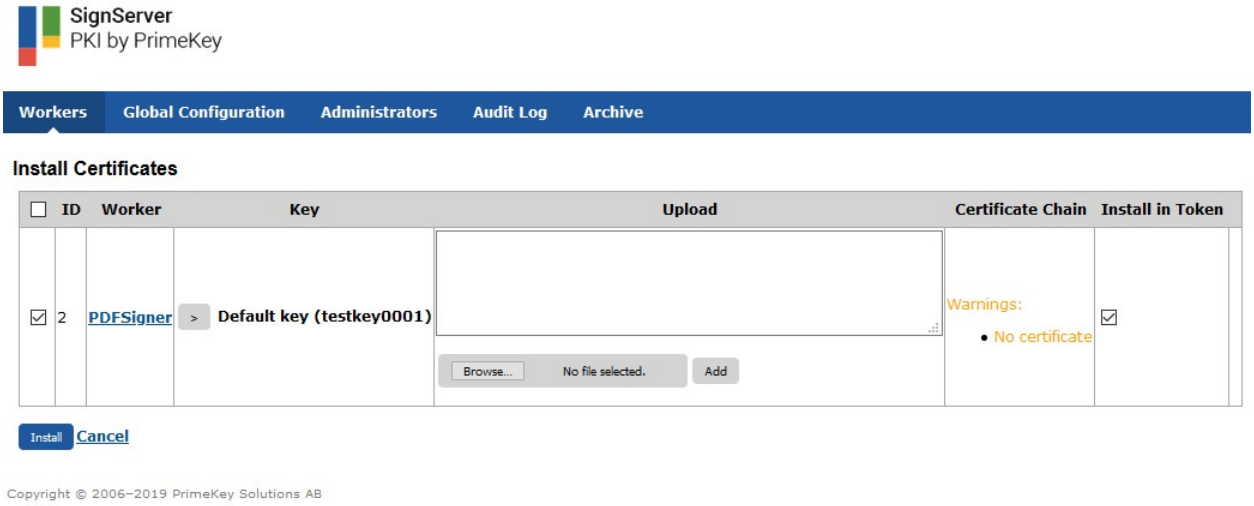

Figure 14 : Signserver PdfSigner certificate install

11. To activate the new signer, select the link to the new signer in the workers list, and click on **Activate**.
12. Enter the HSM slot PIN and click on **Activate**.

8 Back Up and Restore

The Utimaco HSM enables different ways of making backups of either the entire database of keys or just groups of keys. All the backups are encrypted by using the Master Backup Key (MBK), generated by the system administrator, when setting up the HSM. More information about the MBK can be found in [CSADMIN].

8.1 Backing up and Restoring Key Database

All the keys inside the HSM are stored in a CXI database and it is possible to back up the entire database at once. In the same way, the user database can be backed up as well. In FIPS mode this feature is not supported.

1. Open the Crypto Administration Tool.
2. Make sure that the HSM is connected and in the operational mode.
3. Click on **Login/Logoff** to open the **Login/Logoff User** window.
4. Log in the appropriate users to achieve the permission level of at least **22000000**.
5. Click on **Backup/Restore** to open the **CryptoServer Database Backup/Restore Wizard** window.

    a. In the Command section select either:

- Backup databases from source CryptoServer to backup directory.
- Restore databases from backup directory to target CryptoServer.
- Copy databases from source CryptoServer to target CryptoServer.

    b. In the **Settings** section:

- Select the appropriate Source CryptoServer.
- If available, select the appropriate Target CryptoServer.
- In the **Backup** directory section type the appropriate backup directory path (set to `C:\Program Files\Utimaco\CryptoServer\Administration` as default) or click … to browse for the appropriate directory.

6. Select the databases to backup or restore.
7. Click **Execute**.

8. A confirmation window appears.

For a FIPS backup, please use the P11CAT or the P11tool2 tools.

8.2 Backing up and Restoring a Key Database with P11CAT

It is possible to back up separate PKCS#11 slots by using either the P11CAT or the p11tool2. In this guide we use the P11CAT for the backup procedure. Please refer to [CSP11TOOL2] for the backup procedure with the p11tool2.

1. Open the PKCS#11 CryptoServer Administration (P11CAT).
2. Log in to the slot you wish to back up as the Cryptographic User (achieve the permission level of at least **00000002**).
3. Click on **Backup/Restore**.
4. Click on **Backup/Restore Keys**.
5. Select one of the 4 options. Click the one that corresponds to your case. The possibilities are to perform either:

- Backup Internal Keys.
- Backup External Keys.
- Restore Key Backup to Internal Key Store.
- Restore Key Backup to External Key Store.

6. A pop-up window opens.
7. a. Select the directory where the key database will be backed up and type the name of the key database in the section File name.
   b. If you chose to restore a key backup to an internal or an external key store, select the directory where your backup is located.
   c. To confirm your choice, click on **Save**.

7. In the **Status** window a log of the performed action is displayed.

9 FIPS Requirements

All the steps are identical for HSM in FIPS 140-2 approved mode. The only difference is that the backup of the entire key database is not possible. In this case the P11CAT or the p11tool2 are used for backing up the keys.

Note that although the integration does not require extra steps, the HSM running in FIPS mode will accept ONLY FIPS compliant parameters. Be careful to select FIPS compliant algorithms, key lengths and elliptic curves when generating new keys. For more information about the FIPS compliant algorithms please refer to the CryptoServer User and Administration Guides.

10 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*.

11 References

Reference	Title/Company	Document No.
[CSADMIN]	CryptoServer – csadm Manual/Utimaco IS GmbH	2009-0003
[CSPKCSM]	CryptoServer - PKCS#11 P11CAT Manual	M013-0001-en
[LPKCSHD]	Learning PKCS#11 in Half a Day	2015-0008
[CSPKCSDG]	CryptoServer PKCS#11 R2 Developer Guide	2012-0007

Table 5: List of references

12 Contact and Support Information

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

Utimaco IS GmbH
Krefelder Str. 220
52070 Aachen
Germany

RMA Query

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

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

Other Support Queries

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