ACDIS PKCS#11 Manager User Manual

Version: 1.1.0.0	
Changed on: 08/04/2024 10:05 AM	Created on: 03/01/2024
Status: FINAL	

Table of Content

1. GENERAL INFORMATION

1.1. COPYRIGHT

1.2. DOCUMENT HISTORY

Version	Date	Author	Description
1.0.0.0	08/01/2024	AUSTRIACARD, Markus Punz	First release v1.0.0.0
1.0.0.2	19/01/2024	AUSTRIACARD, Markus Punz	Added installation for Apple-Silicon (ARM) New Version due to PKCS#11 Manager Version 1.2.0.0
1.1.0.0	23/03/2024	AUSTRIACARD, Markus Punz	New Version due to PKCS#11 Manager Version 1.3.0.0

1.3. PKCS#11 MANAGER HISTORY

Version	Date	Author	Description
1.0.0.0	14/12/2023	AUSTRIACARD	First release (Windows only)
1.1.0.0	08/01/2024	AUSTRIACARD	Fixed typos in About-Dialog.

			Fix: Throw PKCS#11 error during CSR generation
1.2.0.0	19/01/2024	AUSTRIACARD	Added support for Apple-Silicon (= ARM) Windows-Deployment of PKCS#11 Library 1.5 is shipped with this version of PKCS#11 Manager (= V 1.2.0.0)
1.3.0.0	03/04/2024	AUSTRIACARD	Added support to edit Minidriver-Configuration (Generic-Mode vs. SSCD-Mode) – for details see 5.20 Added support for CA-Certificates For details see 5.10, 5.14, 5.21

1.4. DISCLAIMER OF LIABILITY

Austria Card guarantees for a period of twenty-four months from the time of delivery that the Software essentially corresponds to the program description in the accompanying written material with regard to its functionality.

Austria Card points out that the Software is qualified as a one-off service. Necessary updates can be obtained via the same channels through which the Software was obtained as long as the warranty applies.

Austria Card points out that according to the state of the art it is not possible to produce computer Software completely error-free.

If a defect occurs, the defect and its appearance must be described in such detail in a written notice of defect that a review of the defect (e.g. submission of error messages) is feasible and the exclusion of an operating error (e.g. specification of the work steps) is possible.

If the notice of defects proves to be justified, the licensee sets AUSTRIA CARD a reasonable deadline for subsequent performance. The licensee informs AUSTRIA CARD which type of supplementary performance - improvement of the delivered or delivery of a new, defect-free item - he wishes. However, AUSTRIA CARD is entitled to refuse the selected supplementary performance if this can only be carried out with disproportionate costs for it and if the other type of supplementary performance would not entail any significant disadvantages for the licensee. AUSTRIA CARD may also refuse subsequent performance altogether if it can only be carried out at disproportionate cost to it.

In order to carry out the supplementary performance, AUSTRIA CARD is entitled to two attempts for the same or directly related defect within the period set by the licensee. After the second failed attempt at subsequent performance, the licensee may withdraw from the contract or reduce the license fee. The right of withdrawal or reduction can already be exercised after the first unsuccessful attempt at subsequent performance, if a second attempt within the set period is not reasonable for the licensee. If subsequent performance has been refused under the conditions set out above, the licensee is entitled to the right of reduction or withdrawal immediately.

If the licensee has made a claim against AUSTRIA CARD under warranty, and it turns out that either there is no defect or the asserted defect does not oblige AUSTRIA CARD to provide a warranty, the licensee must reimburse all expenses incurred by AUSTRIA CARD if he is grossly negligent or intentional responsible for the use of AUSTRIA CARD

A guarantee that the Software is suitable for the purposes of the licensee and cooperates with the licensee's existing Software is excluded.

Beyond this warranty, AUSTRIA CARD shall only be liable for a period of two years from delivery of the Software in the event of intent and gross negligence in accordance with the statutory provisions. In the event of slight negligence, AUSTRIA CARD shall only be liable if an essential contractual obligation is violated or if there is a case of delay or impossibility. In the event of liability arising from slight negligence, this liability shall be limited to such damages that are foreseeable or typical. Liability for the lack of the guaranteed quality, due to fraudulent intent, for personal injury and the General Data Protection Agreement remains unaffected.

In the event of a claim against AUSTRIA CARD under warranty or liability, contributory negligence on the part of the user must be taken into account appropriately, in particular in the case of insufficient error messages or insufficient data backup. Insufficient data backup exists in particular if the licensee has failed to take precautions against external influences, in particular against computer viruses and other phenomena that can endanger individual data or an entire database, by means of appropriate, state-of-the-art security measures.

Under no circumstances shall AUSTRIA CARD or its affiliates, partners, suppliers or licensors be liable for any indirect, incidental, consequential, special or exemplary damages arising out of or in connection with your access or use of or inability to access or use the Software and any third party content and services, whether or not the damages were foreseeable and whether or not company was advised of the possibility of such damages.

2. OVERVIEW

This document is the user manual for the AUSTRIACARD ACDIS PKCS#11 Manager. This program is used to manage ACDIS smart cards via PKCS#11 interface.

For example, the following functions can be carried out:

For Generic-PKCS#11 applications:

- Change PIN
- Unlock PIN
- Change SO PIN
- Reset token
- Generate key pairs (RSA / ECC)
- Generate PKCS#10 CSRs
- Import PKCS#12 files
- Import certificates (User-certificates and CA-certificates)
- Delete key pairs
- Delete certificates

For SSCD applications:

- Activation of SSCD tokens
- Change PIN
- Unlock PIN
- Generate PKCS#10 CSRs
- Import certificates (User-certificates and CA-certificates)

Note:

- The PKCS#11 generic application can be personalized yourself. You can create and delete key pairs on your own. The private keys can be protected with different PINs.
- SSCD applications on the other hand are pre-personalized. Every SSD application has qualified signature keys that are already generated during smart card pre-personalization. This means that when the smart card is delivered,

all QES keys are already present on the card. Deleting key pairs via PKCS#11 Manager is not supported.

2.1. SUPPORTED OPERATING SYSTEMS

- Microsoft Windows 10 22H2 – 32 Bit
- Microsoft Windows 10 22H2 – 64 Bit
- Microsoft Windows 10 LTSC 2021 – 32 Bit
- Microsoft Windows 10 LTSC 2021 – 64 Bit
- Microsoft Windows 11 22H2 – 64 Bit
- Ubuntu Linux 20.04 LTS
- Ubuntu Linux 22.04 LTS
- Debian 11 / 12
- macOS Monterey (Version 12)
- macOS Ventura (Version 13)
- macOS Sonoma (Version 14)

3. INSTALLATION

Installation depends on the operating system:

3.1. WINDOWS 32-BIT / 64-BIT

The PKCS#11 Manager is supplied with its own installation program (.exe). In addition to the PKCS#11 Manager, the installer also includes the ACDIS PKCS#11 library. This means that the PKCS#11 library does not have to be installed separately.

Note:

The following dependencies will also be installed automatically if they are not present:

- Microsoft Visual C++ Redistributable for Visual Studio 2015-2022
- Microsoft .NET Runtime 6.0

3.2. LINUX

The installation is carried out using the “Homebrew” package manager.

To do this, Homebrew must be installed for the first time. Skip 3.2.1 and 3.2.2 if Homebrew is already installed.

3.2.1. Prerequisites for Homebrew

Homebrew requires the tools „curl“ and „git“.

If not available, these must be installed first.

If applicable execute:

```
sudo apt install curl
sudo apt install git
```

3.2.2. Install Homebrew

Please execute:

- /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

After succcessful install execute:

- test -d ~/.linuxbrew && eval "$(~/.linuxbrew/bin/brew shellenv)"
- test -d /home/linuxbrew/.linuxbrew && eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"
- test -r ~/.bash_profile && echo "eval \"\$($(brew --prefix)/bin/brew shellenv)\"" >> ~/.bash_profile
- echo "eval \"\$($(brew --prefix)/bin/brew shellenv)\"" >> ~/.profile

3.2.3. Install PKCS#11 Manager via Homebrew

Please execute:

- brew tap austriacard/acdislinux
- brew install acdisp11manager

Note:

- The above commands install the PKCS#11 Manager (= PKCS11Manager) in the directory /home/linuxbrew/.linuxbrew/Cellar/acdisp11manager/<<Version>>/
- During the initial installation, a number of dependencies are installed. Please note that the installation process may take some time.
- A symbolic link to the PKCS11Manager executeable is created in the directory /home/linuxbrew/.linuxbrew/bin .
- Installing the ACDIS PKCS#11 Manager also installs the ACDIS PKCS#11 library (libacdis-pkcs11.so) - if not available. This will be installed in the directory /home/linuxbrew/.linuxbrew/Cellar/acdislinux/<<Version>>/.
- A symbolic link to the libacdis-pkcs11.so file is created in the /home/linuxbrew/.linuxbrew/lib directory.

The PKCS#11 library uses PC/SC to access the card reader. The PC/SC smart card daemon must therefore be installed (if not available):

- Install PC/SC smart card daemon:

  sudo apt-get install pcscd

- Start of PC/SC smart card daemon:

  sudo service pcscd start

3.2.4. Install .NET Runtime

The ACDIS PKCS#11 Manager requires the Microsoft .NET Runtime (>= 6.0). This cannot be installed automatically.

Depending on the Linux version, please follow the instructions from Microsoft:

https://learn.microsoft.com/en-us/dotnet/core/install/linux

Note:

For example, for Debian 12, the following needs to be done:

```
wget https://packages.microsoft.com/config/debian/12/packages-microsoft-prod.deb -O packages-microsoft-prod.deb

sudo dpkg -i packages-microsoft-prod.deb

rm packages-microsoft-prod.deb

sudo apt-get install -y dotnet-runtime-8.0
```

3.2.5. Install Desktop-Shortcut for PKCS#11 Manager

So that you don't have to open a terminal window to run the PKCS11Manager program, a program icon can be created in the Applications section of your Linux-Desktop environment.

To do this, the following must be entered into the .profile file:

```
echo "export XDG_DATA_DIRS=$XDG_DATA_DIRS:$HOMEBREW_PREFIX/share" >> ~/.profile
```

Then restart Linux so that the changes are active.

3.2.6. Uninstall PKCS#11 Manager

- brew uninstall acdisp11manager
- brew untap austriacard/acdislinux

3.3. MACOS

The installation is also carried out under macOS using the “Homebrew” package manager. To do this, Homebrew must first be installed. Skip 3.3.1 if Homebrew is already installed.

3.3.1. Install Homebrew

For Intel-Mac and Apple-Silicon please execute:

- /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

Please also do the following on Apple-Silicon (ARM) – must be omitted on Intel-Mac:

- (echo; echo 'eval "$(/opt/homebrew/bin/brew shellenv)"') >> ~/.zprofile
- eval "$(/opt/homebrew/bin/brew shellenv)"

3.3.2. Install PKCS#11 Manager via Homebrew

Please execute:

- brew tap austriacard/acdismac
- brew install –cask acdisp11manager

Note:

- On Intel-Mac the above commands install the PKCS#11 Manager in the directory /usr/local/Caskroom/acdisp11manager/<<Version>>/
- On Apple-Silicon the PKCS#11 Manager is installed in the directory /opt/homebrew/Caskroom/acdisp11manager/<<Version>>/
- A program icon to the PKCS#11 Manager is created in the Applications folder of your Mac.
- Installing the ACDIS PKCS#11 Manager also installs the ACDIS PKCS#11 library (= acdis-pkcs11.dylib) - if not available. This is installed in the /usr/local/Cellar/acdismac/<<Version>>/ directory for Intel-Mac and /opt/homebrew/Cellar/acdismac/<<Version>>/ for Apple-Silicon.

- A symbolic link to the acdis-pkcs11.dylib file is created in the /usr/local/lib directory (- Intel-Mac) or /opt/homebrew/lib (- Apple-Silicon)
- The ACDIS PKCS#11 Manager automatically installs the Microsoft .NET Runtime if it is not present.

3.3.3. Uninstall PKCS#11 Manager

- brew uninstall –cask acdisp11manager
- brew untap austriacard/acdismac

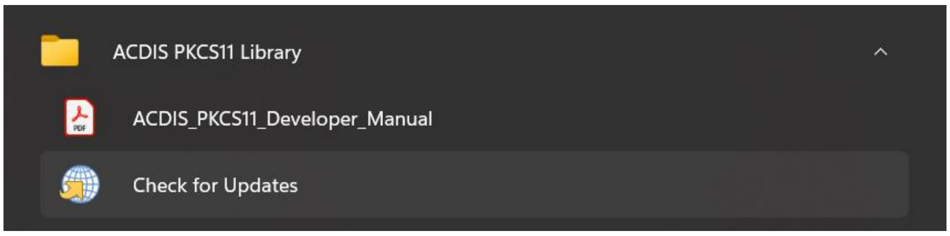
3.4. ALL OPERATING SYSTEMS

Please install the card reader driver recommended by the manufacturer for your device.

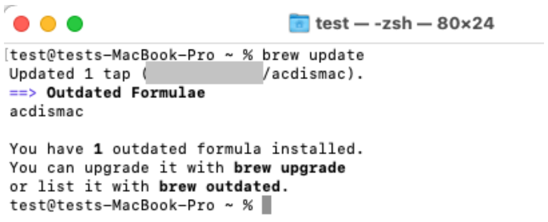
4. UPDATES

4.1. MANUALLY TRIGGERED UPDATES

To check whether there is a new version, Windows users can run the "Check for up-dates" function in the Start menu:



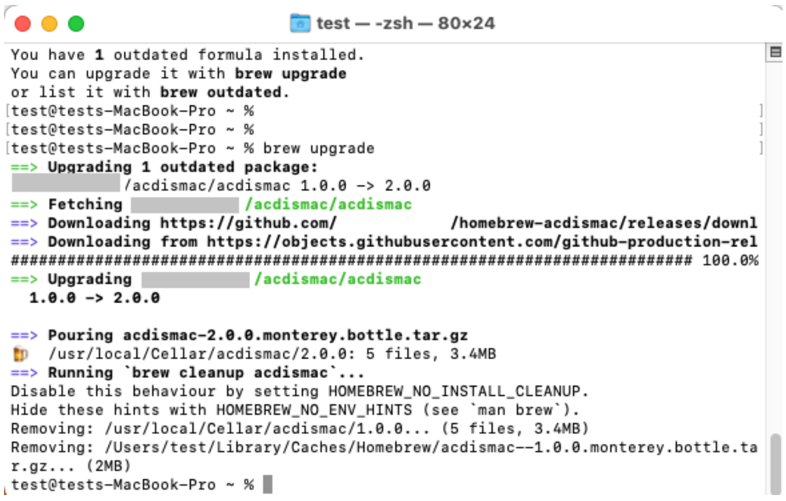
Linux/Mac users can run the "brew update" function on the command line. This checks whether there is a new version:



```
test — -zsh — 80×24
test@tests-MacBook-Pro ~ % brew update
Updated 1 tap ([redacted]/acdismac).
==> Outdated Formulae
acdismac

You have 1 outdated formula installed.
You can upgrade it with brew upgrade
or list it with brew outdated.
test@tests-MacBook-Pro ~ %
```

If a new version is found, it can be installed with "brew upgrade":



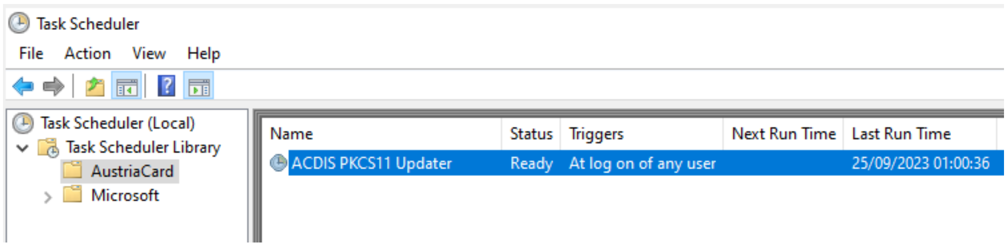
```
test — -zsh — 80×24
You have 1 outdated formula installed.
You can upgrade it with brew upgrade
or list it with brew outdated.
test@tests-MacBook-Pro ~ %
test@tests-MacBook-Pro ~ %
test@tests-MacBook-Pro ~ % brew upgrade
==> Upgrading 1 outdated package:
[redacted]/acdismac/acdismac 1.0.0 -> 2.0.0
==> Fetching [redacted]/acdismac/acdismac
==> Downloading https://github.com/[redacted]/homebrew-acdismac/releases/downl
==> Downloading from https://objects.githubusercontent.com/github-production-rel
######################################################################## 100.0%
==> Upgrading [redacted]/acdismac/acdismac
  1.0.0 -> 2.0.0

==> Pouring acdismac-2.0.0.monterey.bottle.tar.gz
🍺  /usr/local/Cellar/acdismac/2.0.0: 5 files, 3.4MB
==> Running `brew cleanup acdismac`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
Removing: /usr/local/Cellar/acdismac/1.0.0... (5 files, 3.4MB)
Removing: /Users/test/Library/Caches/Homebrew/acdismac--1.0.0.monterey.bottle.ta
r.gz... (2MB)
test@tests-MacBook-Pro ~ %
```

4.2. CONFIGURE AUTOMATIC UPDATES

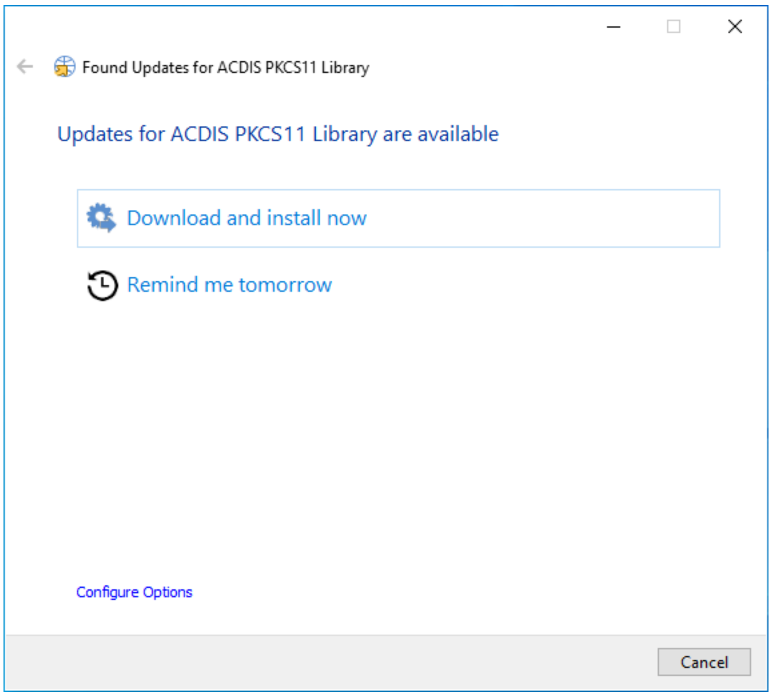
4.2.1. Windows

The ACDIS PKCS#11 Manager installer configures a task in the Windows scheduler that automatically checks after each login whether there is a new version of the program.



Name	Status	Triggers	Next Run Time	Last Run Time
ACDIS PKCS11 Updater	Ready	At log on of any user		25/09/2023 01:00:36

If so, the user can decide whether he wants to install the update or not.



Updates for ACDIS PKCS11 Library are available

4.2.2. MacOS

The Homebrew package manager can be configured to install upgrades automatically. To do this, please run the following in a terminal window:

```
brew tap homebrew/autoupdate
mkdir -p $HOME/Library/LaunchAgents
```

```
# If you want to check for updates every 24h (= default)
    brew autoupdate start --upgrade
```

```
# Alternatively: If you want to check for updates every hour (= 3.600 seconds)
# Please specify the desired update-interval in seconds
    brew autoupdate start 3600 --upgrade
```

To stop automatic updates you can do the following:

```
brew autoupdate stop
```

4.2.3. Linux

Automatic updates can be configured using shell commands that are executed automatically after login.

To do this, create a linuxbrewUpdate.sh in the user HOME with the following content:

```
#!/bin/bash
brew update
brew upgrade
```

Then run the following command in the terminal window so that the update script is started automatically after every login:

```
echo "$HOME/linuxbrewUpdate.sh &" >> ~/.profile
```

Note:

Other variants, for example in which the above script is periodically executed as a cron job, are also possible.

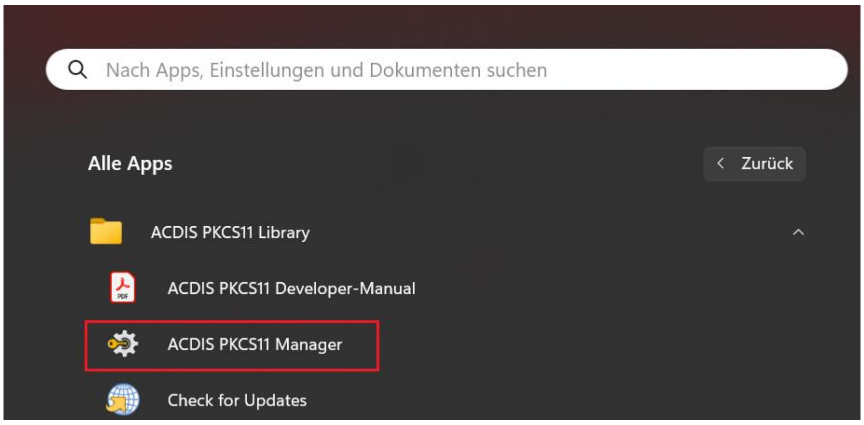
We won't go into this in more detail here.

5. PKCS#11 MANAGER OVERVIEW

5.1. STARTING PKCS#11 MANAGER

5.1.1. Windows

After installation, the ACIDS PKCS#11 Manager can be accessed via start menu:


Nach Apps, Einstellungen und Dokumenten suchen
Alle Apps
Zurück
ACDIS PKCS11 Library
ACDIS PKCS11 Developer-Manual
ACDIS PKCS11 Manager
Check for Updates

5.1.2. Linux

If a desktop shortcut has been set up according to 3.2.5, the ACDIS PKCS#11 Manager can be called from the GUI. Alternatively, the PKCS#11 Manager can also be started directly from a terminal window:

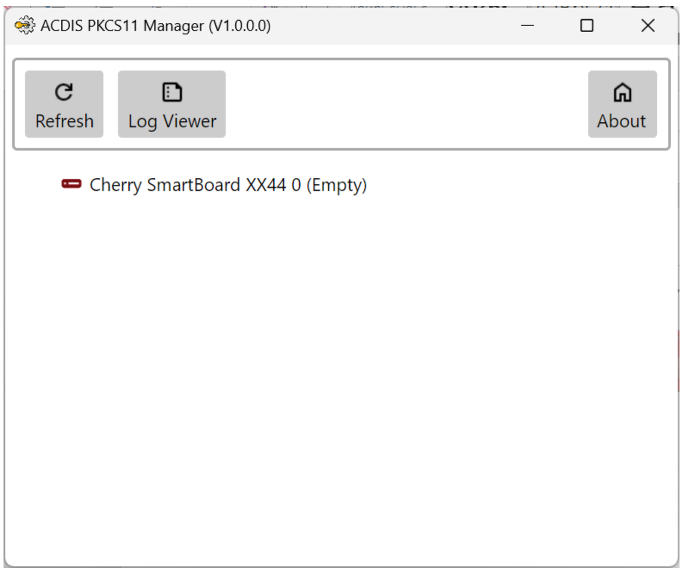
```
/home/linuxbrew/.linuxbrew/bin/PKCS11Manager
```

5.1.3. Mac

A desktop icon is installed in the Applications.

5.2. MAIN WINDOW

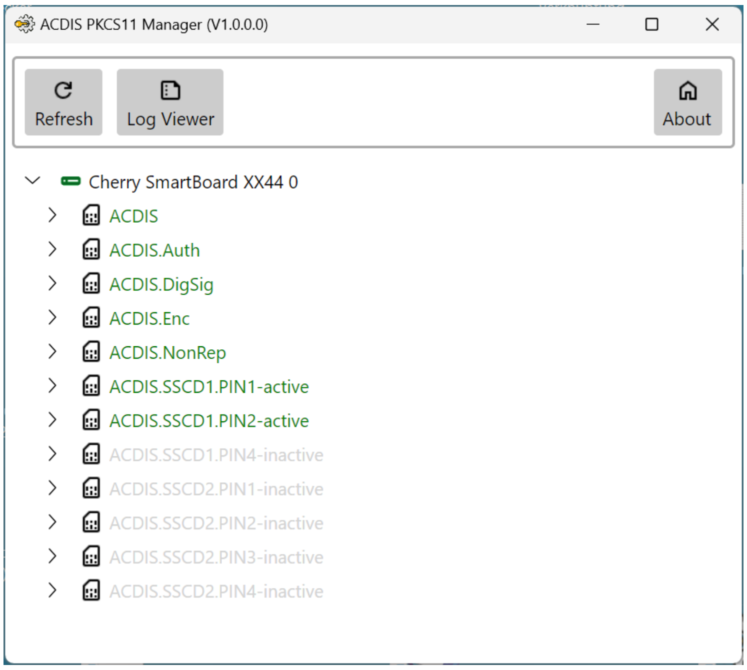
After starting PKCS#11 Manager, the main window is displayed:



Cherry SmartBoard XX44 0 (Empty)

The list contains all card readers that are found in the system. The suffix "Empty" means that the card reader does not contain an ACDIS smart card.

In this case, please insert an ACDIS smart card and click the "Refresh" button:



Cherry SmartBoard XX44 0
ACDIS
ACDIS.Auth
ACDIS.DigSig
ACDIS.Enc
ACDIS.NonRep
ACDIS.SSCD1.PIN1-active
ACDIS.SSCD1.PIN2-active
ACDIS.SSCD1.PIN4-inactive
ACDIS.SSCD2.PIN1-inactive
ACDIS.SSCD2.PIN2-inactive
ACDIS.SSCD2.PIN3-inactive
ACDIS.SSCD2.PIN4-inactive

Now all PKCS#11 tokens on the ACDIS smart card are displayed.

Depending on the personalization, ACDIS smart cards can contain tokens of different type:

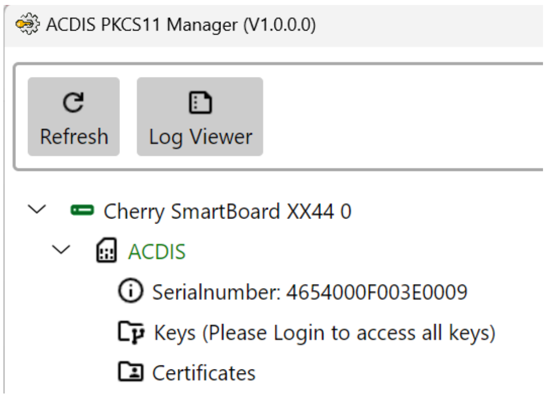
- Tokens of type "Generic PKCS#11 application":
    ACDIS
    ACDIS.Auth
    ACDIS.DigSig
    ACDIS.Enc
    ACDIS.NonRep

- Tokens of type "SSCD":
  ACDIS.SSCD<<n>>.PIN<<x>>

Each token has its own PIN.

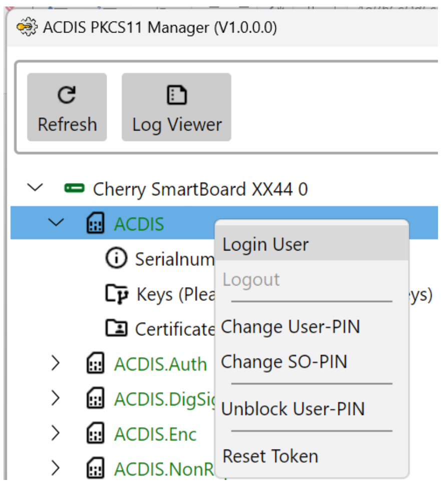
5.2.1. Display public / private token objects

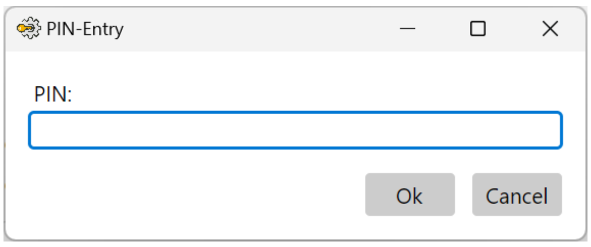
Each token has public objects and private objects. Public objects are visible without user login - for example the serialnumber or the certificates (if any are available).



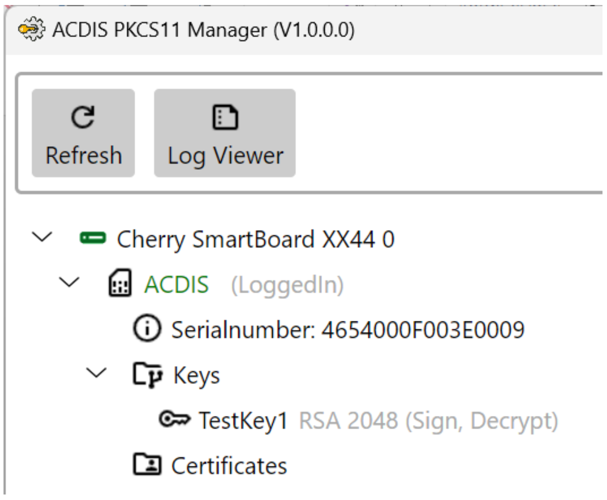
Cherry SmartBoard XX44 0
ACDIS
Serialnumber: 4654000F003E0009
Keys (Please Login to access all keys)
Certificates

"Keys", on the other hand, are private objects. In order to view this, you must log in with the PIN of the respective token.





If you are successfully logged in to the token, the suffix “LoggedIn” will be displayed to the right of the token name. Furthermore, the keys of the token are now displayed - if any are available:


ACDIS PKCS11 Manager (V1.0.0.0)
Refresh
Log Viewer
Cherry SmartBoard XX44 0
ACDIS (LoggedIn)
Serialnumber: 4654000F003E0009
Keys
TestKey1 RSA 2048 (Sign, Decrypt)
Certificates

5.2.2. Function overview

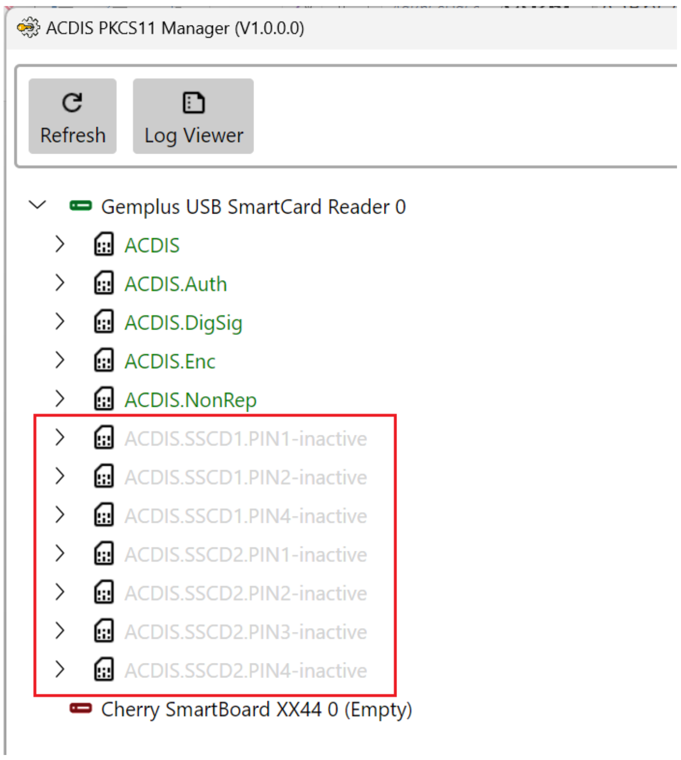
As already mentioned at the beginning, different functions can be carried out depending on the type of the token:

Function	Generic	SSCD
Activation of SSCD-Tokens Note: The PINs of SSCD tokens are inactive upon delivery and protected with a transport PIN. This means that the signatory must first change the transport PIN to a self-selected signature PIN (= activation). Only then the SSCD token can be used. (- for details see 0)	N	Y
Change PIN (- for details see 5.4)	Y	Y
Change SO-PIN Note: The SO-PIN (=PUK) of SSCD-Tokens cannot be changed. (- for details see 5.5)	Y	N
Unblock PIN Note: With Generic Tokens, a new PIN can be assigned when the PIN is unlocked. (*) With SSCD tokens, however, you can only unlock but no new PIN can be assigned. (- for details see 0)	Y	Y(*)
Generate Keypair (RSA or ECC) Note: With SSCD tokens, the key pairs are already pre-generated. (- for details see 5.7)	Y	N
Generate PKCS#10 CSR (- for details see 5.8)	Y	Y
Import Certificate	Y	Y

(- for details see 5.9)		
Import CA-Certificate (- for details see 5.10)	Y	Y
Import PKCS#12 Files (- for details see 5.11)	Y	N
Delete Keypair (- for details see 5.12)	Y	N
Delete Certificate (- for details see 5.13)	Y	N
Delete CA-Certificate (- for details see 5.14)	Y	Y
Delete (Reset) Token (- for details see 5.15)	Y	N

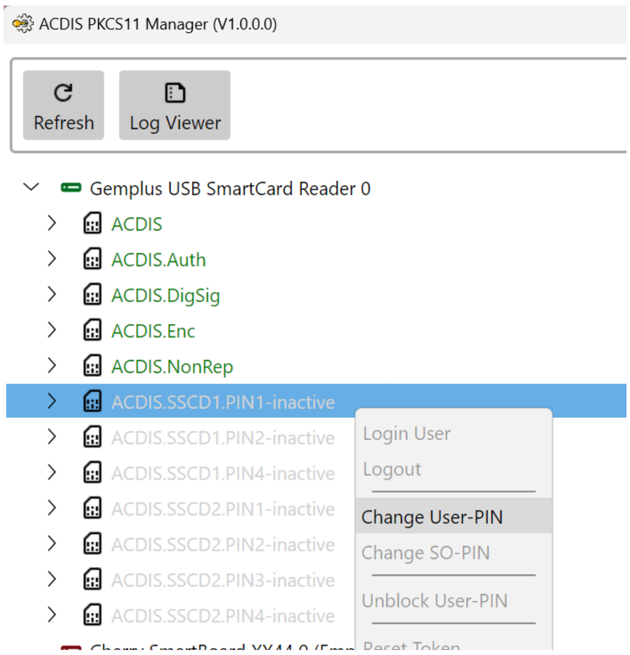
5.3. ACTIVATION OF SSCD TOKENS

The PINs of SSCD tokens are inactive upon delivery and protected with a transport PIN. The SSCD tokens are therefore displayed in grey (disabled) and have the suffix "inactive":

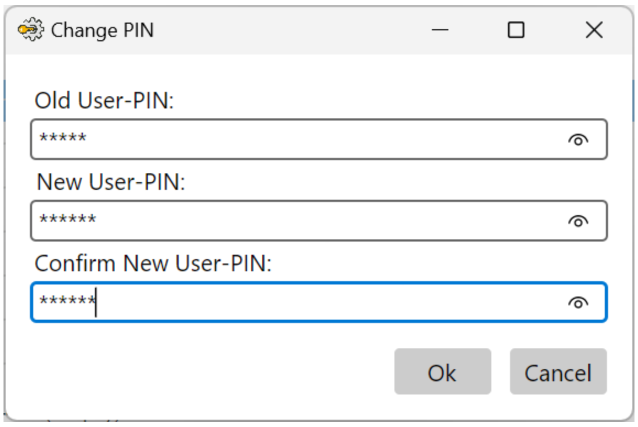


Gemplus USB SmartCard Reader 0
ACDIS
ACDIS.Auth
ACDIS.DigSig
ACDIS.Enc
ACDIS.NonRep
ACDIS.SSCD1.PIN1-inactive
ACDIS.SSCD1.PIN2-inactive
ACDIS.SSCD1.PIN4-inactive
ACDIS.SSCD2.PIN1-inactive
ACDIS.SSCD2.PIN2-inactive
ACDIS.SSCD2.PIN3-inactive
ACDIS.SSCD2.PIN4-inactive
Cherry SmartBoard XX44 0 (Empty)

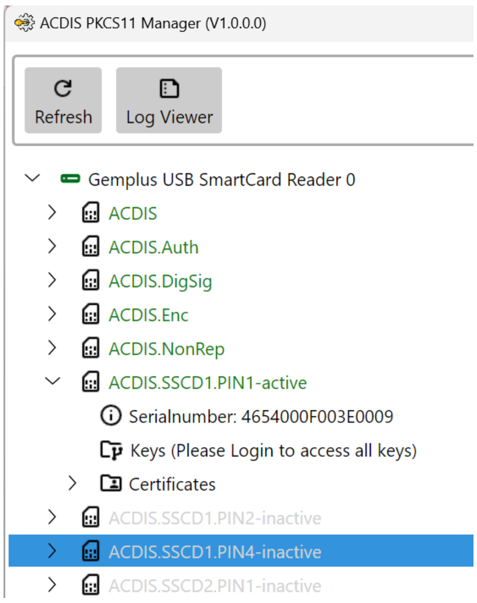
In order to be able to use the SSCD token, the transport PIN must first be changed to a self-selected user PIN. To do this, please execute the "Change User PIN" function in the context menu:


ACDIS PKCS11 Manager (V1.0.0.0)
Refresh
Log Viewer
Gemplus USB SmartCard Reader 0
ACDIS
ACDIS.Auth
ACDIS.DigSig
ACDIS.Enc
ACDIS.NonRep
ACDIS.SSCD1.PIN1-inactive
ACDIS.SSCD1.PIN2-inactive
ACDIS.SSCD1.PIN4-inactive
ACDIS.SSCD2.PIN1-inactive
ACDIS.SSCD2.PIN2-inactive
ACDIS.SSCD2.PIN3-inactive
ACDIS.SSCD2.PIN4-inactive
Login User
Logout
Change User-PIN
Change SO-PIN
Unblock User-PIN

The transport PIN must be entered as the “Old User PIN”. The self-selected user PIN follows under “New User PIN” and “Confirm New User PIN”:



Now the token is displayed “green” and has the addition “active”:

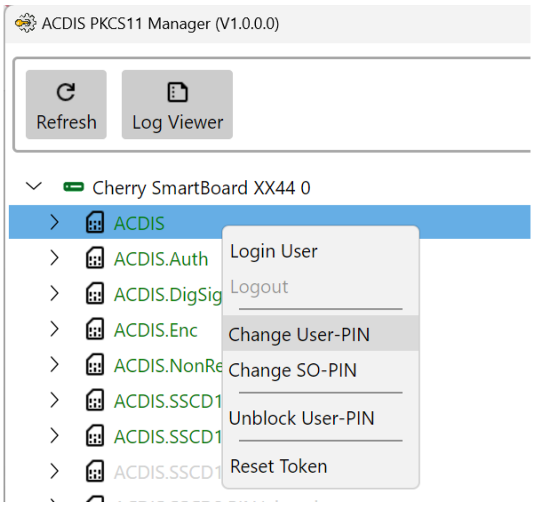


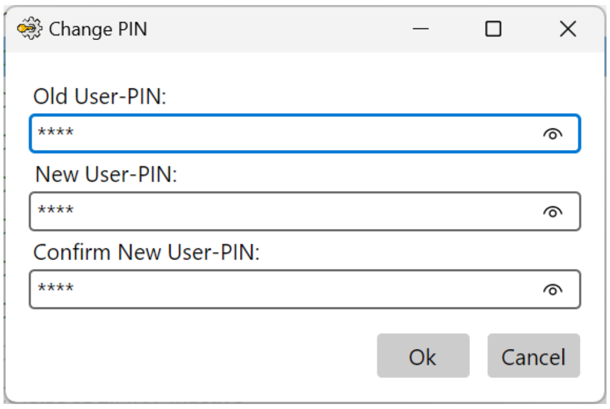
Gemplus USB SmartCard Reader 0
ACDIS
ACDIS.Auth
ACDIS.DigSig
ACDIS.Enc
ACDIS.NonRep
ACDIS.SSCD1.PIN1-active
Serialnumber: 4654000F003E0009
Keys (Please Login to access all keys)
Certificates
ACDIS.SSCD1.PIN2-inactive
ACDIS.SSCD1.PIN4-inactive
ACDIS.SSCD2.PIN1-inactive

The SSCD token is now operational and can be used for further operations.

5.4. CHANGE PIN

Right-click on desired token and select “Change User-PIN” in context menu:



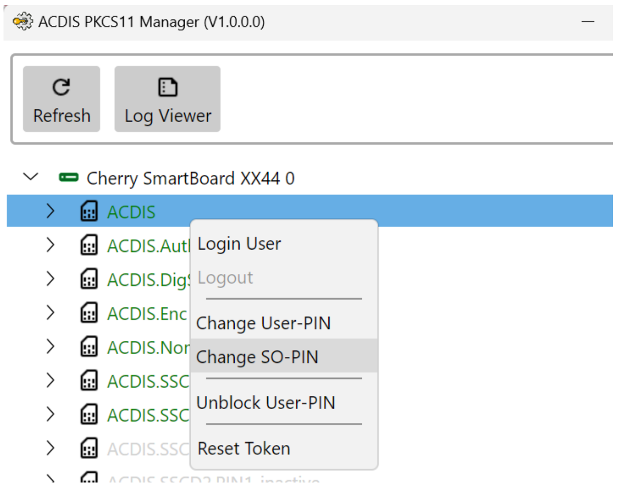


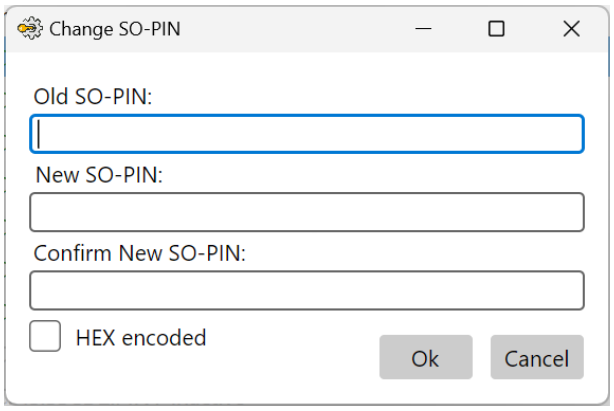
Note:

Generic PKCS#11 tokens support alphanumeric PINs. The PINs of SSCD tokens, however, must be numeric (0-9).

5.5. CHANGE SO-PIN

Right-click on desired token and select “Change SO-PIN” in context menu:

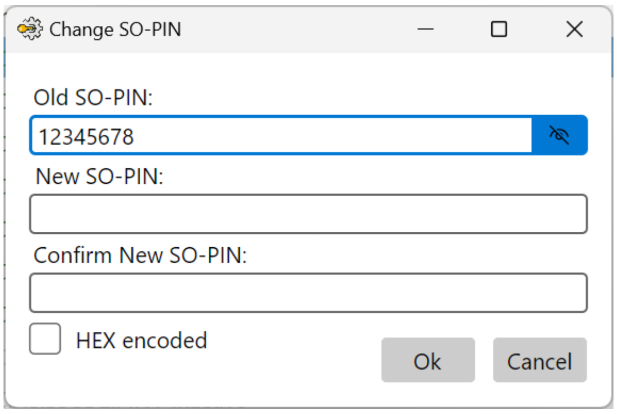




The SO-PIN of generic PKCS#11 tokens is actually a key. Since keys are often represented/interpreted as hex values, the input “Hex encoded” can also be made here.

Example:

Assuming the SO PIN has the value “12345678”. This can then be entered in ASCII as follows:


Change SO-PIN

Old SO-PIN: 12345678

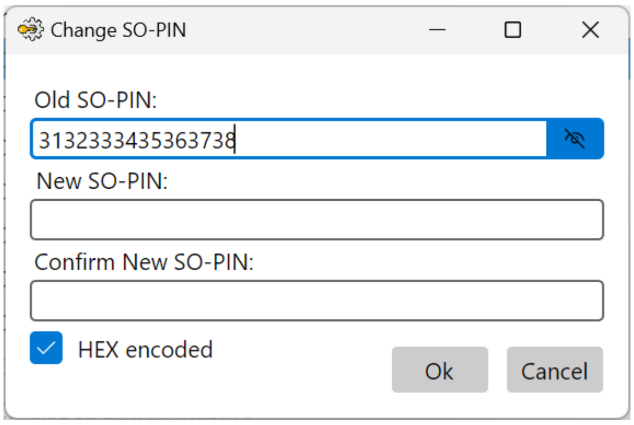
New SO-PIN:

Confirm New SO-PIN:

☐ HEX encoded

Ok Cancel

Alternatively, the same value can also be entered hex-encoded:


Change SO-PIN

Old SO-PIN: 3132333435363738

New SO-PIN:

Confirm New SO-PIN:

☑ HEX encoded

Ok Cancel

<u>SO-PIN (hex-encoded):</u> \x31\x32\x33\x34\x35\x36\x37\x38
(=ASCII “12345678”)

The selected encoding (HEX vs. ASCII) is applied to all PINs in this window - i.e. Old SO-PIN, New SO-PIN and Confirm New SO-PIN.

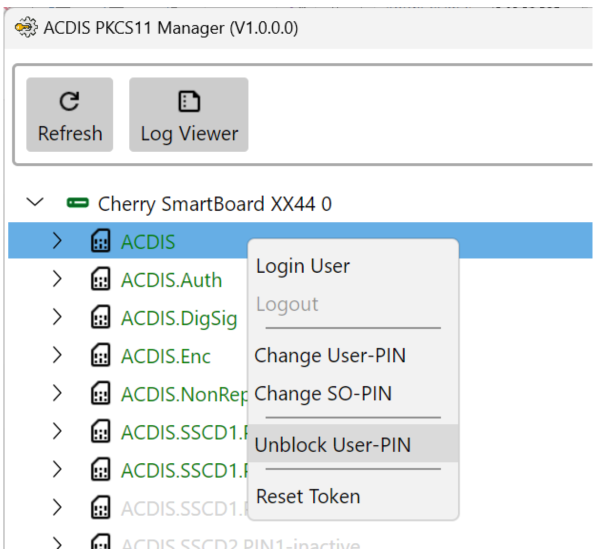
Note:

The same SO-PIN is used for all generic PKCS#11 tokens. Ie the SO-PIN for the token “ACDIS” is the same as for ACDIS.Auth, ACDIS.DigSig, ACDIS.Enc, ACDIS.NonRep. If it is changed for one token then the new SO-PIN also applies to all other tokens.

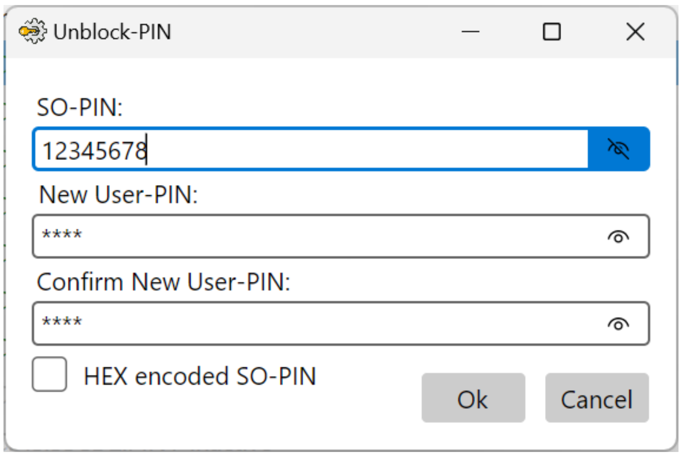
This function is not supported for SSCD Tokens.

5.6. UNBLOCK PIN

For Generic PKCS#11 Tokens:



The SO PIN must be entered followed by the desired new user PIN:



SO-PIN: 12345678

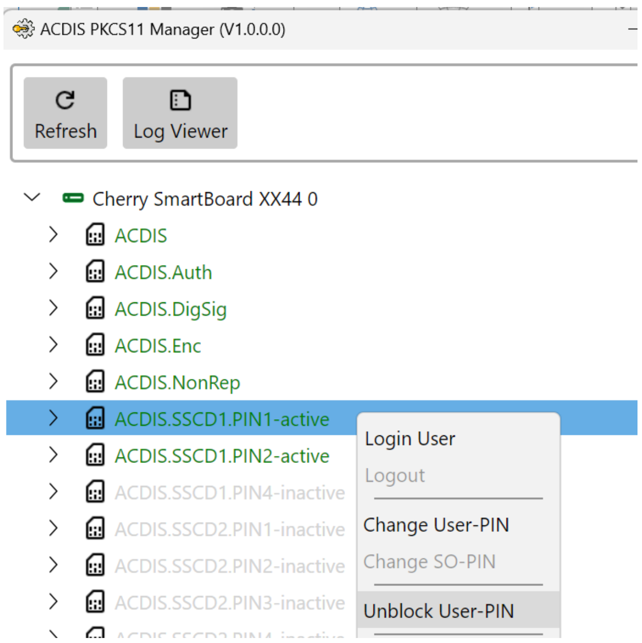
New User-PIN: ****

Confirm New User-PIN: ****

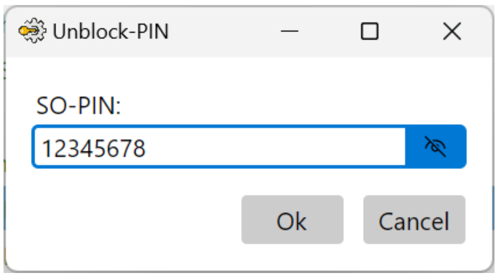
☐ HEX encoded SO-PIN

The SO-PIN can also be entered here in hex-encoded form – for details see 0. Activating this checkbox only applies to the SO-PIN. The user PIN is always interpreted as ASCII.

For SSCD Tokens:



Here the user PIN can only be unlocked. Setting an new user PIN is not supported:

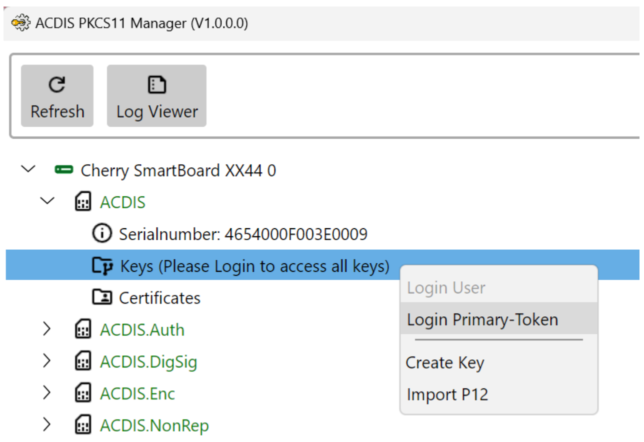


SO-PIN: 12345678

5.7. GENERATE KEYPAIR (RSA OR ECC)

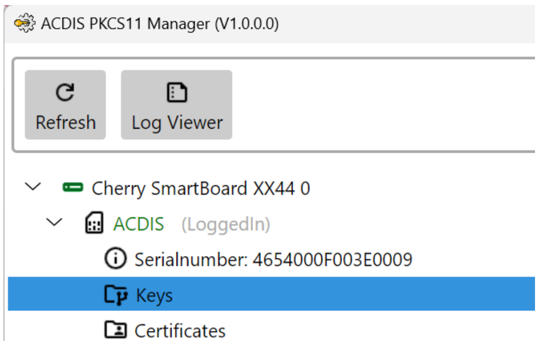
5.7.1. Creating keys on the generic token “ACDIS”

In order to create a key pair on the generic token “ACDIS” (= primary token) you must be logged in:

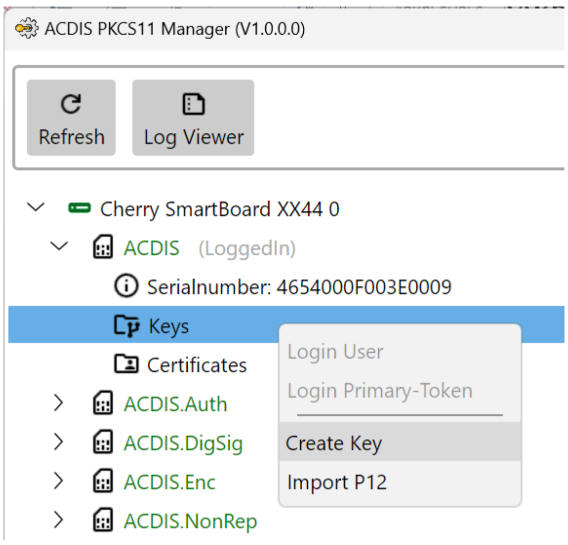


Please enter the PIN of the “ACDIS” token in this dialog window and confirm with OK.

Note: If you are already logged in, this will be indicated by the suffix “LoggedIn” to the right of the token name.



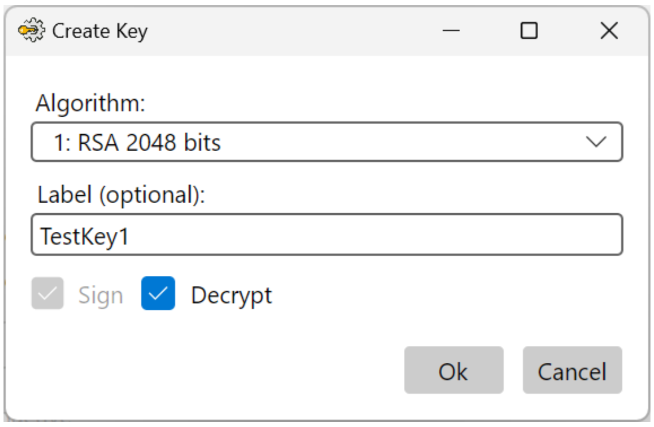
The “Create Key” function can then be called:



The desired algorithm (RSA or ECC) and the key length can now be selected here. Furthermore, a label (= name of the key) can be entered. If no name is entered, a unique name is automatically generated. The “Decrypt” checkbox can be used to select whether the key pair can also be used for encryption/decryption.

Note:

The “Sign” checkbox is always active. This indicates that the key pair can be used for signature/verification.

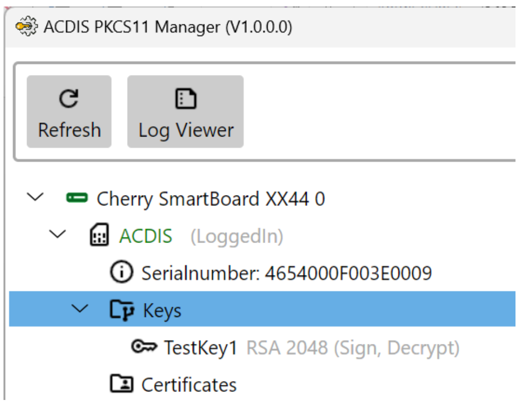


Algorithm: 1: RSA 2048 bits

Label (optional): TestKey1

[✓ Sign] [✓ Decrypt]

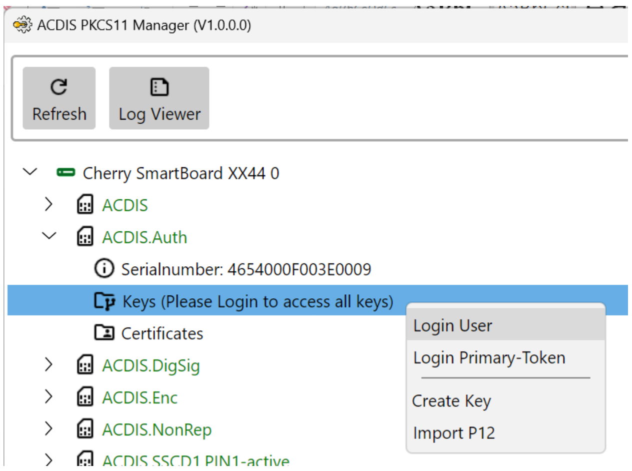
After confirming with the OK button, the key pair is generated and displayed in the overview window:


ACDIS PKCS11 Manager (V1.0.0.0)
Refresh
Log Viewer
Cherry SmartBoard XX44 0
ACDIS (LoggedIn)
Serialnumber: 4654000F003E0009
Keys
TestKey1 RSA 2048 (Sign, Decrypt)
Certificates

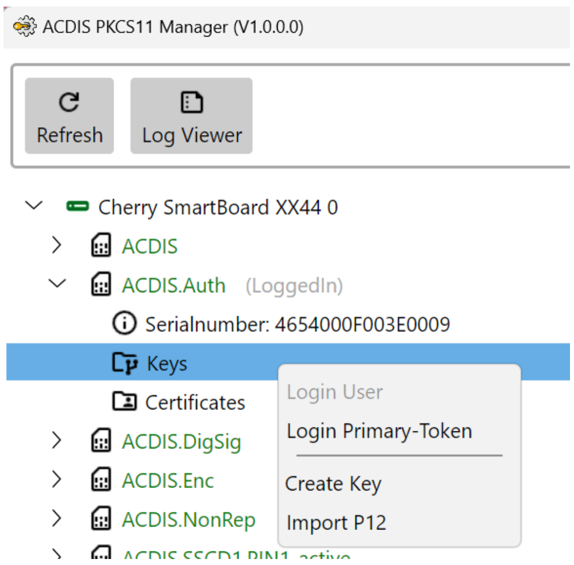
5.7.2. Creating keys on the remaining generic tokens (ACDIS.Auth, ACDIS.DigSig, etc.)

To do this, you must be logged in to the token on which the key pair is to be generated, as well as to the primary token “ACDIS”:

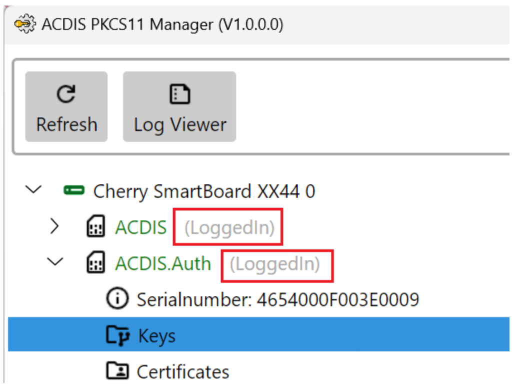
1) Login to the token on which the key pair is to be generated:



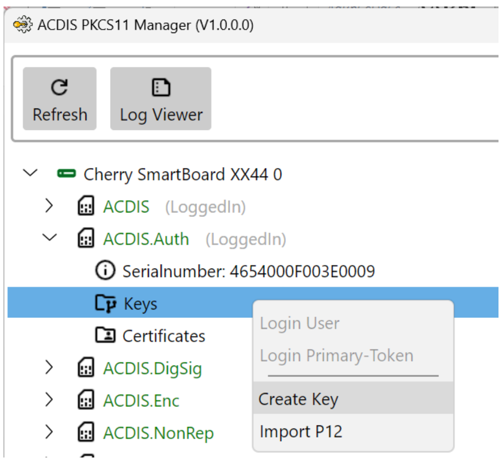
2) Login to the primary token (= “ACDIS”)



The note “LoggedIn” must therefore be displayed for both tokens:



Now the key pair can be generated:



For details see 5.7.1.

Note:

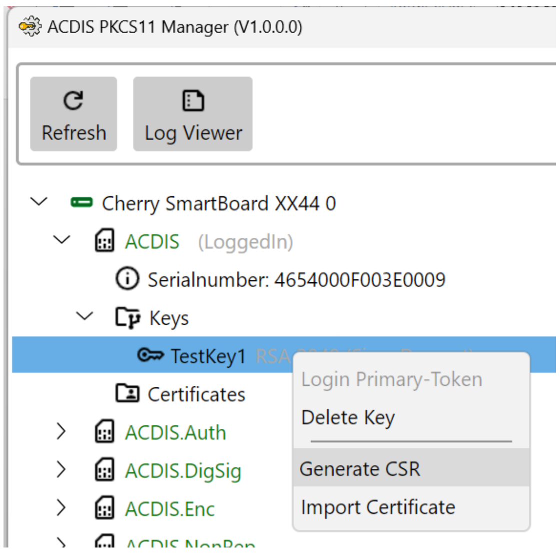
This function is not supported for SSCD Tokens. For SSCD tokens, the key pairs are already pre-generated.

5.8. GENERATE PKCS#10 CSR

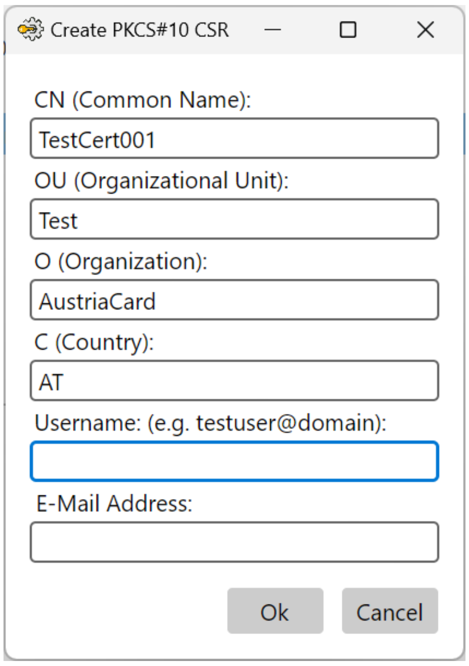
To generate a PKCS#10 CSR you must first select the desired key and select "Generate CSR" in the context menu:

Note:

The keys are only displayed if you are logged in to the token. For details see 5.2.1.



Please enter the CSR data below. Based on this data, the certificate authority issues the desired certificate:


Create PKCS#10 CSR

CN (Common Name): TestCert001

OU (Organizational Unit): Test

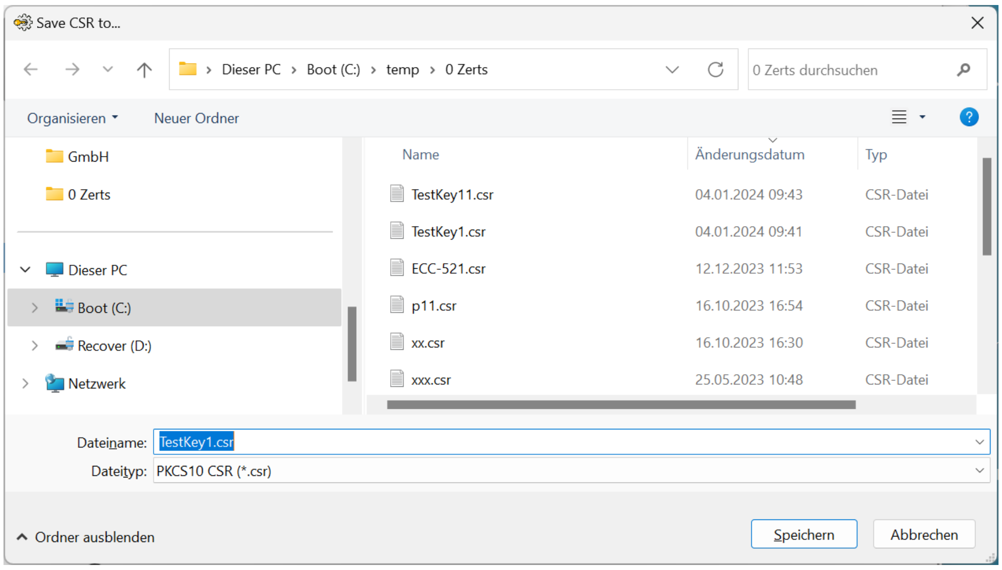
O (Organization): AustriaCard

C (Country): AT

Username: (e.g. testuser@domain):

E-Mail Address:

Finally, the PKCS#10 CSR is saved as a file. You must submit this file to your certificate authority:


Save CSR to...

Dieser PC › Boot (C:) › temp › 0 Zerts

Name	Änderungsdatum	Typ
TestKey11.csr	04.01.2024 09:43	CSR-Datei
TestKey1.csr	04.01.2024 09:41	CSR-Datei
ECC-521.csr	12.12.2023 11:53	CSR-Datei
p11.csr	16.10.2023 16:54	CSR-Datei
xx.csr	16.10.2023 16:30	CSR-Datei
xxx.csr	25.05.2023 10:48	CSR-Datei

Dateiname: TestKey1.csr

Dateityp: PKCS10 CSR (*.csr)

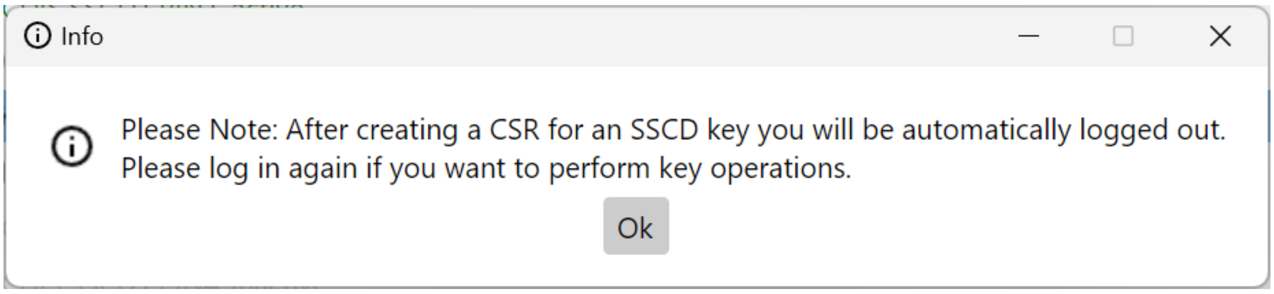
You can then import the certificate that you received from your certificate authority – see 0

Import Certificate.

Note:

If the selected key is a SSCD key, then the signature PIN is "consumed". This means that generating the CSR automatically resets the authentication state of the signature PIN.

This is indicated by the following note:



Please Note: After creating a CSR for an SSCD key you will be automatically logged out.
Please log in again if you want to perform key operations.

After the command you are no longer authenticated.

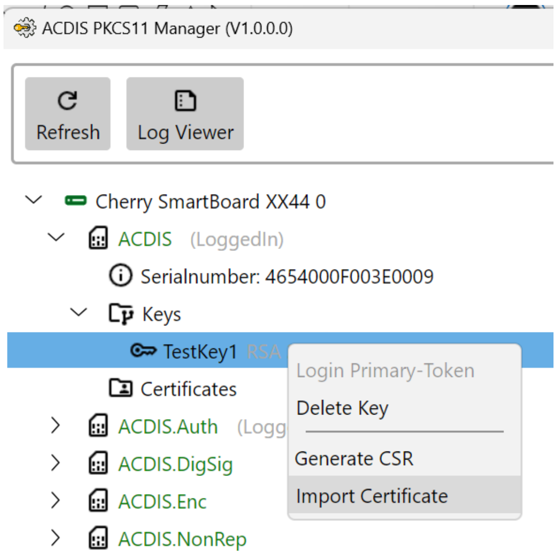
Therefore, a login must be carried out for every operation that requires the signature PIN of an SSCD-key.

5.9. IMPORT CERTIFICATE

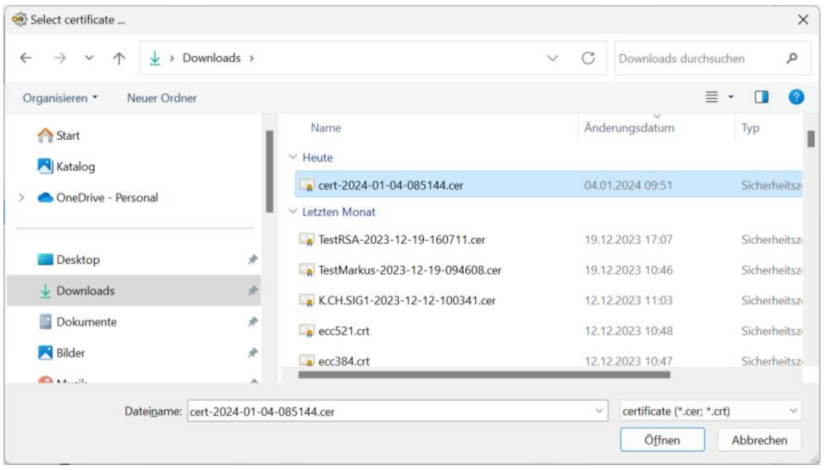
In order to import a certificate for an existing key, you must first select the desired key and select “Import Certificate” in the context menu:

Note:

The keys are only displayed if you are logged in to the token. For details see 5.2.1.



Now you have to select the certificate file that you received from the certificate authority:

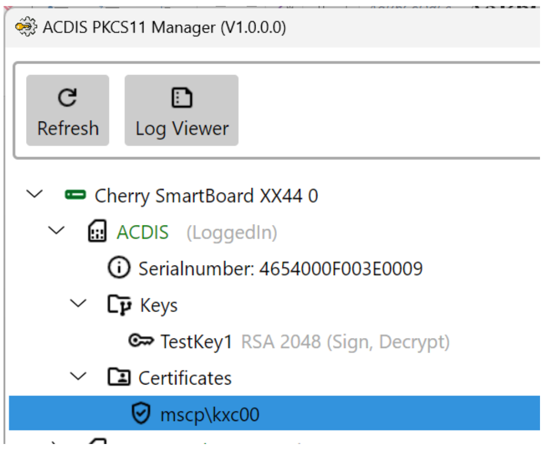


Name	Änderungsdatum	Typ
Heute		
cert-2024-01-04-085144.cer	04.01.2024 09:51	Sicherheitsz
Letzten Monat		
TestRSA-2023-12-19-160711.cer	19.12.2023 17:07	Sicherheitsz
TestMarkus-2023-12-19-094608.cer	19.12.2023 10:46	Sicherheitsz
K.CH.SIG1-2023-12-12-100341.cer	12.12.2023 11:03	Sicherheitsz
ecc521.crt	12.12.2023 10:48	Sicherheitsz
ecc384.crt	12.12.2023 10:47	Sicherheitsz

Dateiname: cert-2024-01-04-085144.cer

certificate (*.cer; *.crt)

The certificate is then imported and displayed in the overview:


ACDIS PKCS11 Manager (V1.0.0.0)

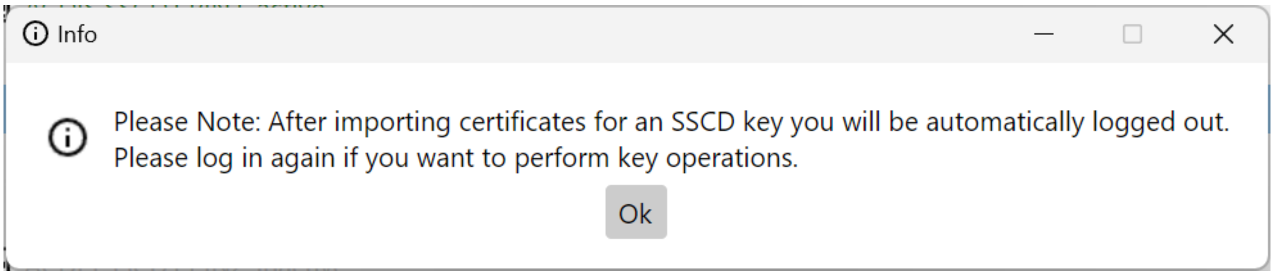
Cherry SmartBoard XX44 0
ACDIS (LoggedIn)
Serialnumber: 4654000F003E0009
Keys
TestKey1 RSA 2048 (Sign, Decrypt)
Certificates
mscp\kxc00

Note:

The names of the certificates (- in the example “mscp/kxc00”) are assigned automatically by the PKCS#11 Manager.

Note:

If the selected key is a SSCD key, then the signature PIN is "consumed". This means that importing the certificate automatically resets the authentication state of the signature PIN. This is indicated by the following note:


Info

Please Note: After importing certificates for an SSCD key you will be automatically logged out.
Please log in again if you want to perform key operations.

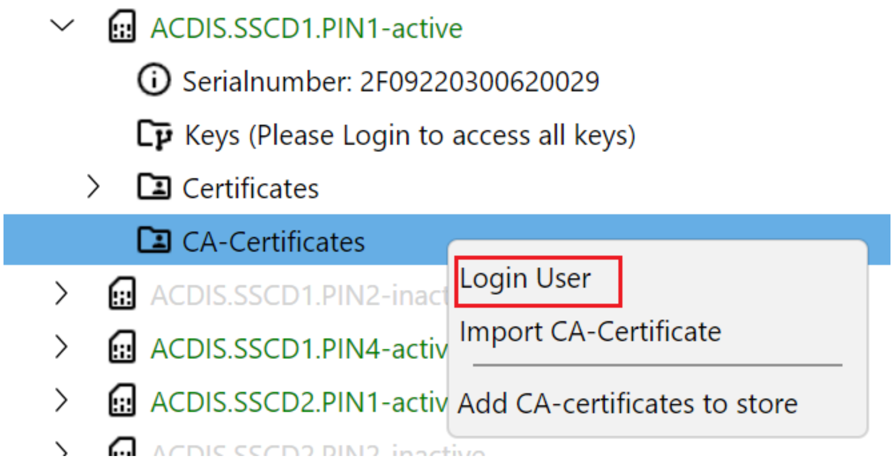
After the command you are no longer authenticated. Therefore, a login must be carried out for every operation that requires the signature PIN of an SSCD-key.

5.10. IMPORT CA-CERTIFICATES

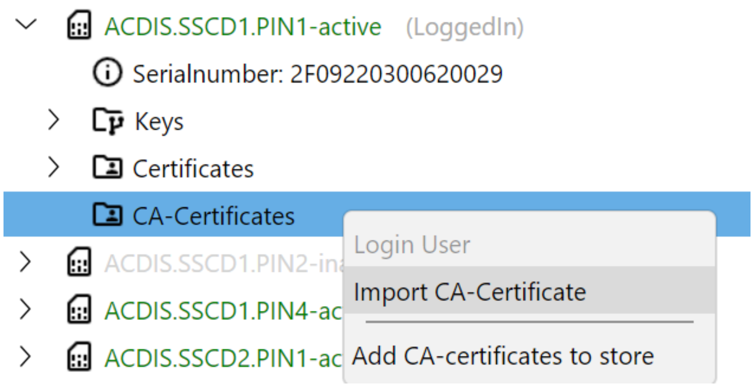
CA-Certificates are typically the parent certificates of the “normal” certificates mentioned under 5.9.

If desired, these can be saved for each token. A maximum of 3 CA certificates can be stored per SSCD token. A maximum of 6 CA certificates can be stored per Generic PKCS#11 token. This is because a Generic PKCS#11 token can contain several key pairs, which may also have different CA certificates. Therefore, a larger number is supported here.

In order to import a CA certificate, the respective token must first be logged in:



After that the “Import CA certificate” function can be called:

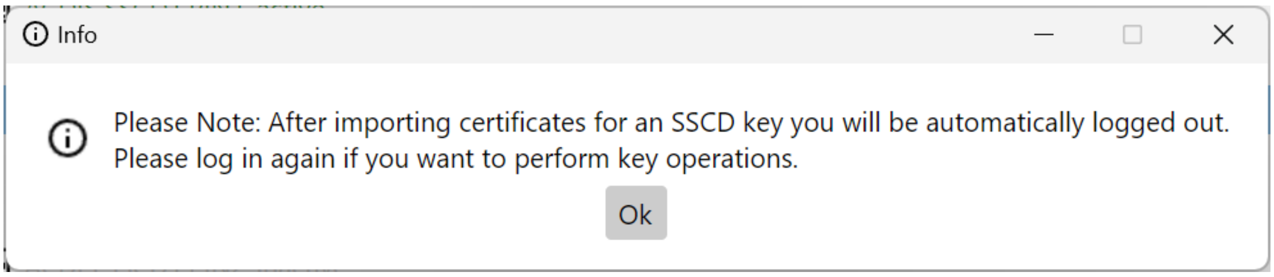


All you then have to do is select the certificate.

Note:

The selected certificate is not checked. This means that it is not checked whether it is actually a CA certificate.

If the selected token is a SSCD token, then the signature PIN is "consumed". This means that importing the CA certificate automatically resets the authentication state of the signature PIN. This is indicated by the following note:



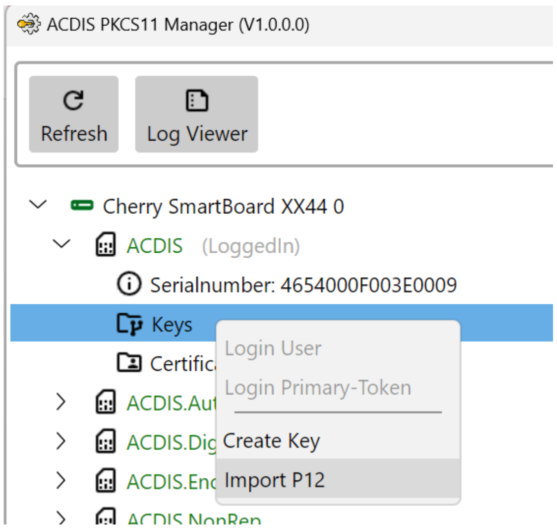
Please Note: After importing certificates for an SSCD key you will be automatically logged out. Please log in again if you want to perform key operations.

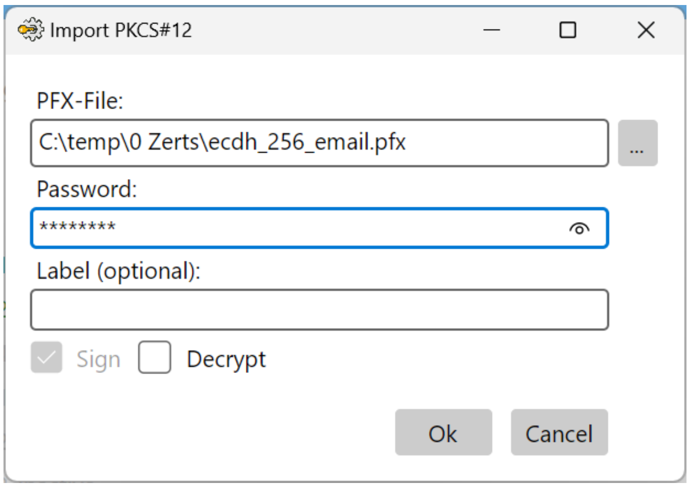
After the command you are no longer authenticated. Therefore, a login must be carried out for every operation that requires the signature PIN of an SSCD-key.

5.11. IMPORT PKCS#12 FILES

PKCS#12 files always contain a key pair (public + private key) and the associated certificate. The file is typically secured with a password.

To import such a file, you must select the entry “Import P12” in the “Keys” group in the context menu:



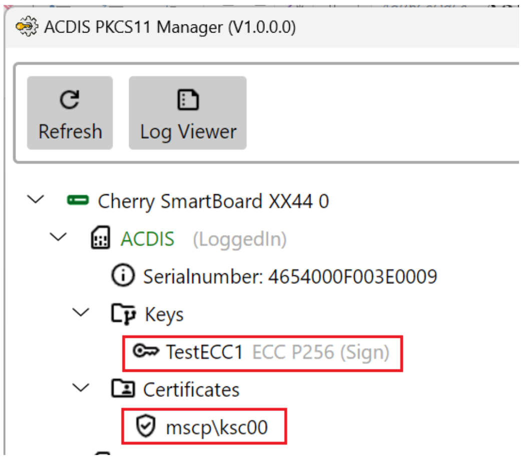


In the dialog window above you must select the PKCS#12 file (.pfx) using the “...” button. Furthermore, the password with which the file is protected must be entered. A label (= name of the key) can optionally be assigned. If not, a unique label is automatically generated. The “Decrypt” checkbox can be used to select whether the key pair can also be used for encryption/decryption.

Note:

The “Sign” checkbox is always active. This indicates that the key pair can be used for signature/verification.

The key and certificate are then imported from the PKCS#12 file:



Cherry SmartBoard XX44 0
ACDIS (LoggedIn)
Serialnumber: 4654000F003E0009
Keys
TestECC1 ECC P256 (Sign)
Certificates
mscp\ksc00

Note:

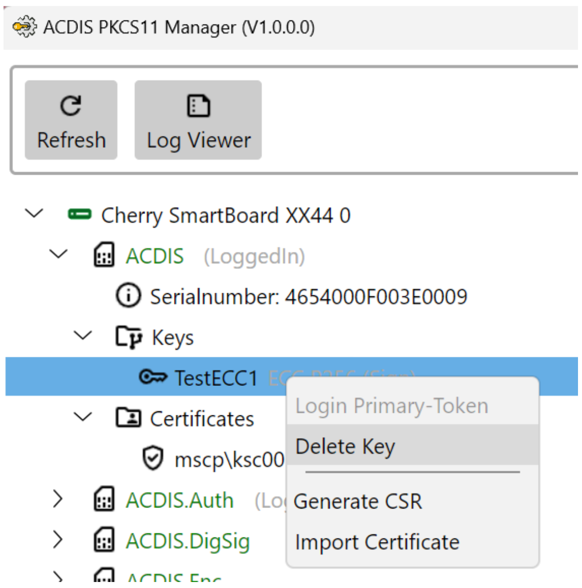
This function is not supported for SSCD Tokens.

5.12. DELETE KEYPAIR

To delete a key, you must first select the desired key and select “Delete Key” in the context menu:

Note:

The keys are only displayed if you are logged in to the token. For details see 5.2.1.



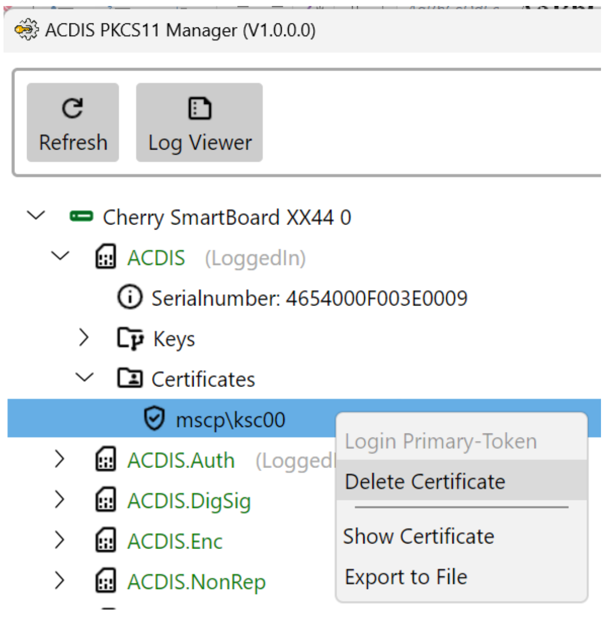
After additional confirmation by the user, the key is deleted. If there is a certificate for the key, this will also be deleted automatically.

Note:

This function is not supported for SSCD Tokens.

5.13. DELETE CERTIFICATE

To delete a certificate, you must first select the desired certificate and select "Delete Certificate" in the context menu:



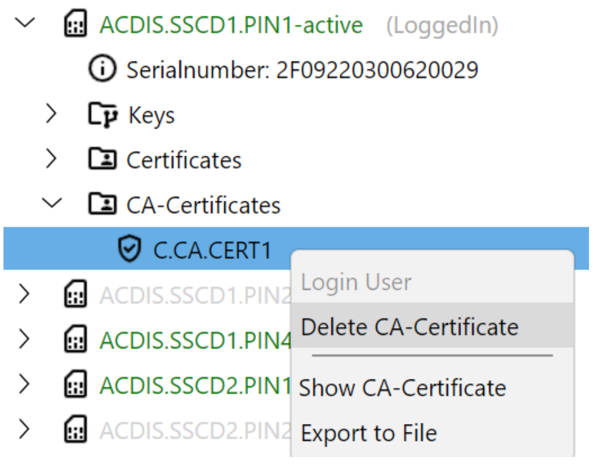
After additional confirmation by the user, the certificate is deleted. The key pair to which the certificate belongs is not deleted. This function only affects the certificate.

Note:

This function is not supported for SSCD Tokens.

5.14. DELETE CA-CERTIFICATE

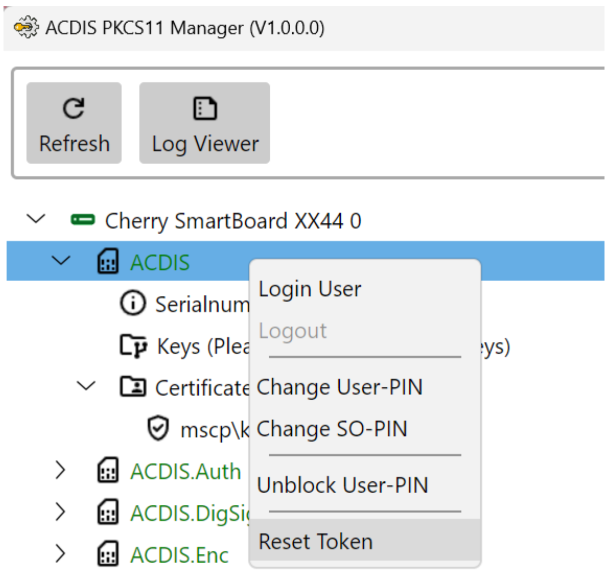
To delete a CA-certificate, you must first select the desired CA-certificate and select "Delete CA-Certificate" in the context menu:



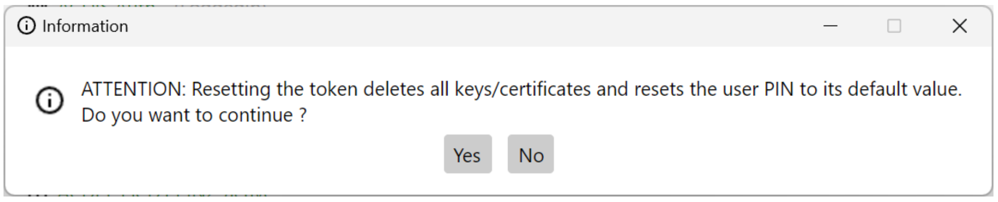
After additional confirmation by the user, the certificate is deleted.

5.15. DELETE (RESET) TOKEN

To delete all keys/certificates of a token, the “Reset Token” function can be selected in the context menu:

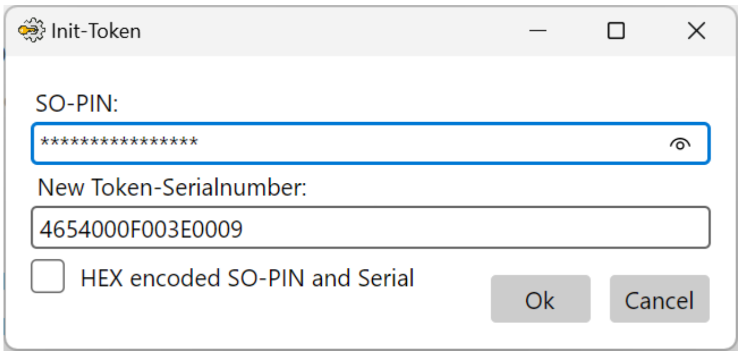


After confirmation, the operation continues:



ATTENTION: Resetting the token deletes all keys/certificates and resets the user PIN to its default value.
Do you want to continue ?

To authorize the deletion process, the SO PIN must be entered. Furthermore, a new serial number can be assigned for the token:

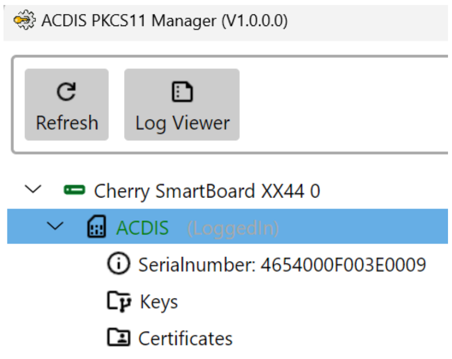


SO-PIN: ****************

New Token-Serialnumber: 4654000F003E0009

☐ HEX encoded SO-PIN and Serial

After the function has been successfully executed, all keys and certificates are deleted:


ACDIS PKCS11 Manager (V1.0.0.0)

Cherry SmartBoard XX44 0
ACDIS (LoggedIn)
Serialnumber: 4654000F003E0009
Keys
Certificates

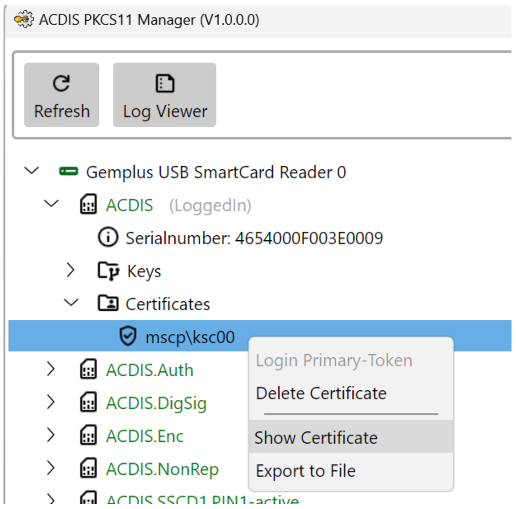
The user PIN has been reset to its default value.

Note:

This function is not supported for SSCD Tokens.

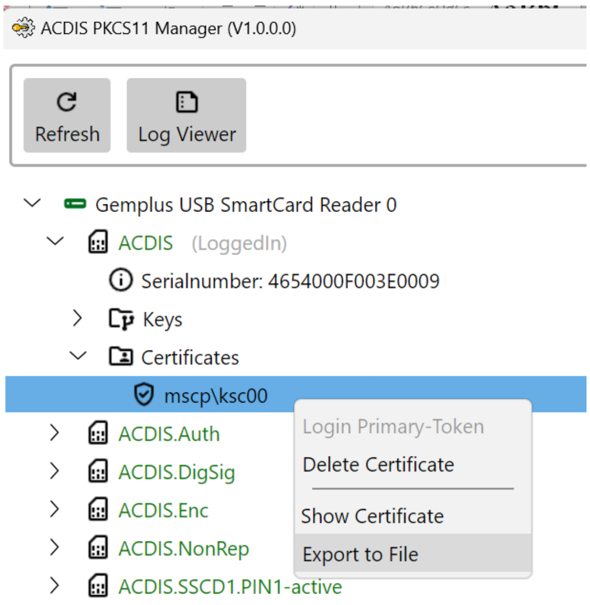
5.16. SHOW CERTIFICATE

This function displays information about the certificate:



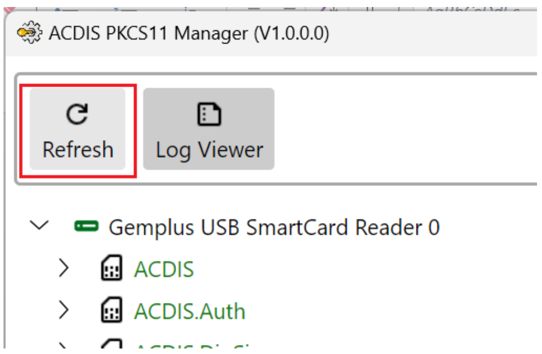
5.17. EXPORT CERTIFICATE TO FILE

This function saves the certificate to a file:



5.18. REFRESH

This function updates the display:

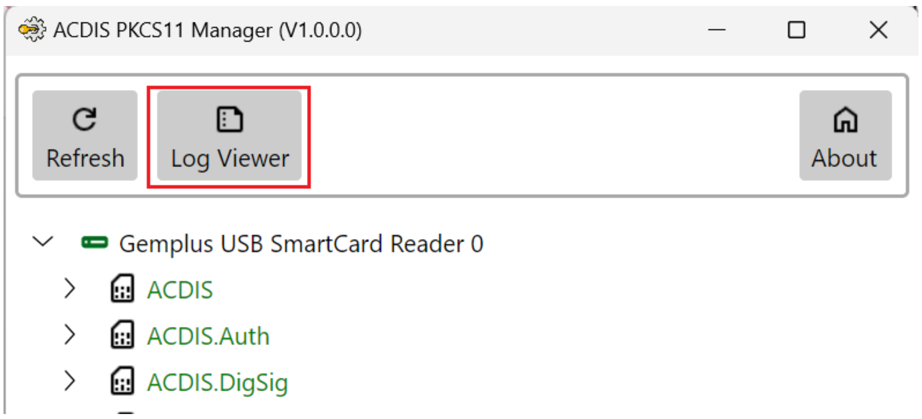

ACDIS PKCS11 Manager (V1.0.0.0)
Refresh
Log Viewer
Gemplus USB SmartCard Reader 0
ACDIS
ACDIS.Auth

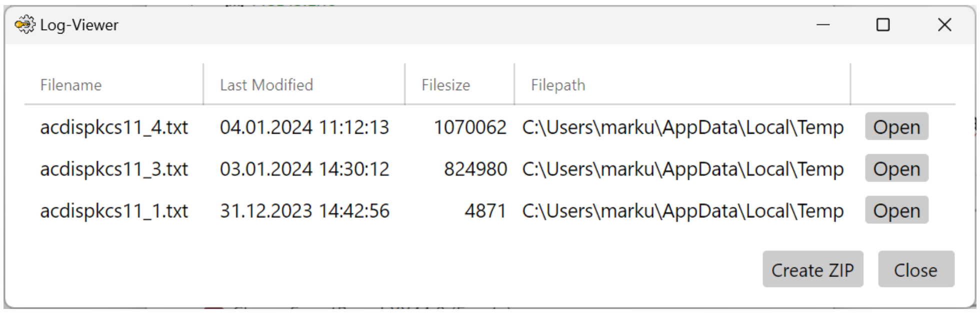
Note:

If card readers are plugged in/unplugged or ACDIS smart cards are re-plugged/re-moved/changed, this function must be carried out to update the display.

5.19. LOG-VIEWER

The log files can be displayed using the log viewer:


ACDIS PKCS11 Manager (V1.0.0.0)
Refresh
Log Viewer
About
Gemplus USB SmartCard Reader 0
ACDIS
ACDIS.Auth
ACDIS.DigSig



Filename	Last Modified	Filesize	Filepath
acdispkcs11_4.txt	04.01.2024 11:12:13	1070062	C:\Users\marku\AppData\Local\Temp
acdispkcs11_3.txt	03.01.2024 14:30:12	824980	C:\Users\marku\AppData\Local\Temp
acdispkcs11_1.txt	31.12.2023 14:42:56	4871	C:\Users\marku\AppData\Local\Temp

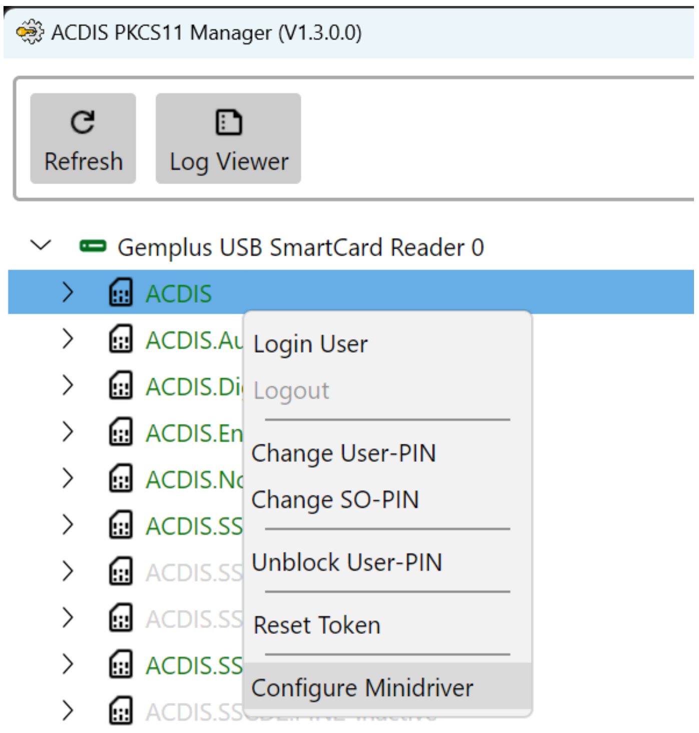
Note:

The log files are intended to enable precise analysis by the manufacturer in the event of an error. With the “Create ZIP” function, the displayed log files are packed into a ZIP file. This can be sent to the manufacturer.

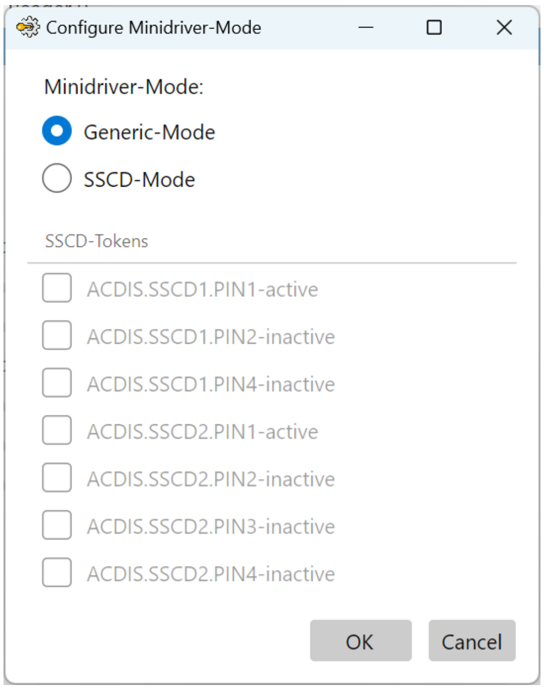
5.20. MINIDRIVER-CONFIGURATION

A Minidriver for Windows is also available for ACDIS smart cards. However, the Minidriver concept does not allow multiple applications (specifically the Generic application and the SSCD application) to be used at the same time. This means you can use the Minidriver either in Generic mode - i.e. with the Generic application or in SSCD mode with the SSCD applications.

This mode can be changed in the PKCS#11 Manager. For this purpose, there is a menu item "Configure Minidriver" for the primary token of the generic application "ACDIS".



Here you can configure the generic mode or the SSCD mode:



Configure Minidriver-Mode

Minidriver-Mode:

[✓ Generic-Mode] SSCD-Mode

SSCD-Tokens

- [ ] ACDIS.SSCD1.PIN1-active
- [ ] ACDIS.SSCD1.PIN2-inactive
- [ ] ACDIS.SSCD1.PIN4-inactive
- [ ] ACDIS.SSCD2.PIN1-active
- [ ] ACDIS.SSCD2.PIN2-inactive
- [ ] ACDIS.SSCD2.PIN3-inactive
- [ ] ACDIS.SSCD2.PIN4-inactive

If you select the “SSCD-Mode” option then you also have to select the SSCD keys that the ACDIS Minidriver provides in Windows. A maximum of 6 SSCD keys can be configured.

Note:

The restriction to a maximum of 6 SSCD keys is a technical limit of the Minidriver concept.

There is still the following to consider:

Using Microsoft CA, certificates can be generated and installed (“enrolled”) onto an ACDIS smart card via Minidriver. The prerequisite for this is that the smart card is configured in Generic-Mode.

If you switch back and forth between the Minidriver modes, there may be problems with the enrollment of these certificates.

Assuming the Minidriver is initially operated in SSCD-mode. In this mode, certificates cannot be added to the smart card via Minidriver. The Minidriver runs in READ-Only mode.

If you switch to generic mode using PKCS#11 Manager in order to apply your own certificates to the ACDIS smart card via Microsoft CA, then you may get an error during enrollment that the smart card is “read-only” - even though the Generic-Mode is configured.

The reason for this is that Windows incorrectly caches the smart card modes. This means that Windows remembers that the smart card was initially operated in SSCD-Mode (= read-only) and, due to this caching, does not notice when the smart card has been reconfigured to Generic-Mode.

This means that if an error occurs when enrolling certificates via Microsoft CA that the ACDIS smart card is “read-only” even though the Generic-Mode is selected in the Minidriver configuration, then you have to delete the smart card cache in Windows:

The corresponding registry key is:

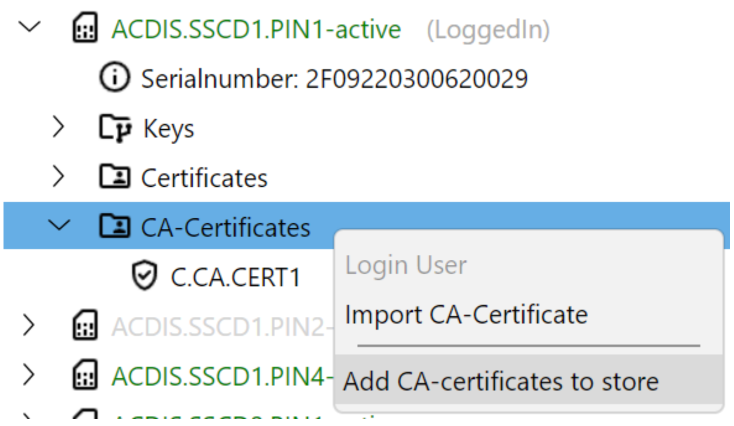
```
HKLM\SOFTWARE\Microsoft\Cryptography\Calais\Cache
```

Please delete this key if the above circumstances apply.

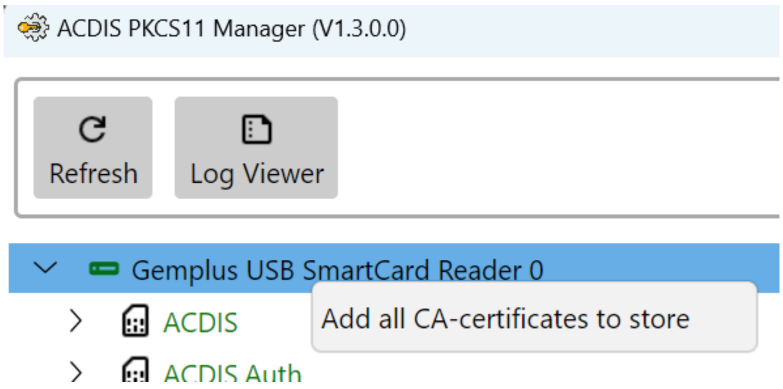
5.21. ADD CA-CERTIFICATES TO TRUST-STORE

CA certificates can be added to the Windows certificate trust store.

To do this, the “Add CA-certificates to store” function can be executed for each token that has CA certificates imported:

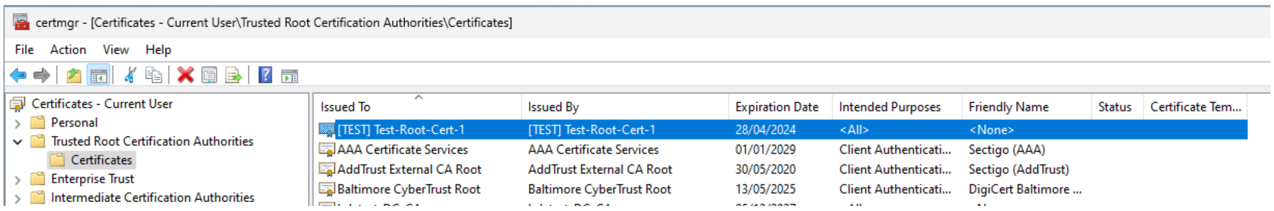


Alternatively, all CA certificates stored on a physical ACDIS smart card (i.e. the CA certificates of all tokens on an ACDIS smart card) can be added to the certificate store:


ACDIS PKCS11 Manager (V1.3.0.0)
Refresh
Log Viewer
Gemplus USB SmartCard Reader 0
ACDIS
ACDIS Auth
Add all CA-certificates to store

If the underlying tokens have CA certificates imported, then these are added to the trust store.

On Windows, the CA certificates are always imported into the CurrentUser\Root storage. Here an example:


certmgr - [Certificates - Current User\Trusted Root Certification Authorities\Certificates]

File
Action
View
Help
Certificates - Current User
Personal
Trusted Root Certification Authorities
Certificates
Enterprise Trust
Intermediate Certification Authorities

Issued To	Issued By	Expiration Date	Intended Purposes	Friendly Name	Status	Certificate Tem...
[TEST] Test-Root-Cert-1	[TEST] Test-Root-Cert-1	28/04/2024	<All>	<None>		
AAA Certificate Services	AAA Certificate Services	01/01/2029	Client Authenticati...	Sectigo (AAA)		
AddTrust External CA Root	AddTrust External CA Root	30/05/2020	Client Authenticati...	Sectigo (AddTrust)		
Baltimore CyberTrust Root	Baltimore CyberTrust Root	13/05/2025	Client Authenticati...	DigiCert Baltimore ...		

Note:

Windows explicitly asks for each certificate that should be added. It will only be added if confirmed by the user.

<u>MacOS:</u>
Importing into Root storage does not work under MacOS because there is no such store. Instead, on MacOS, the “CurrentUser\My” storage is used. This corresponds to the **login.keychain**. There is no other certificate store on MacOS.

Note:

There is the system.keychain. But this is store is for the entire system and not just the user.

<u>Linux:</u>
Here you can only import certificates to CurrentUser. These stores don't actually exist per default. They are created under

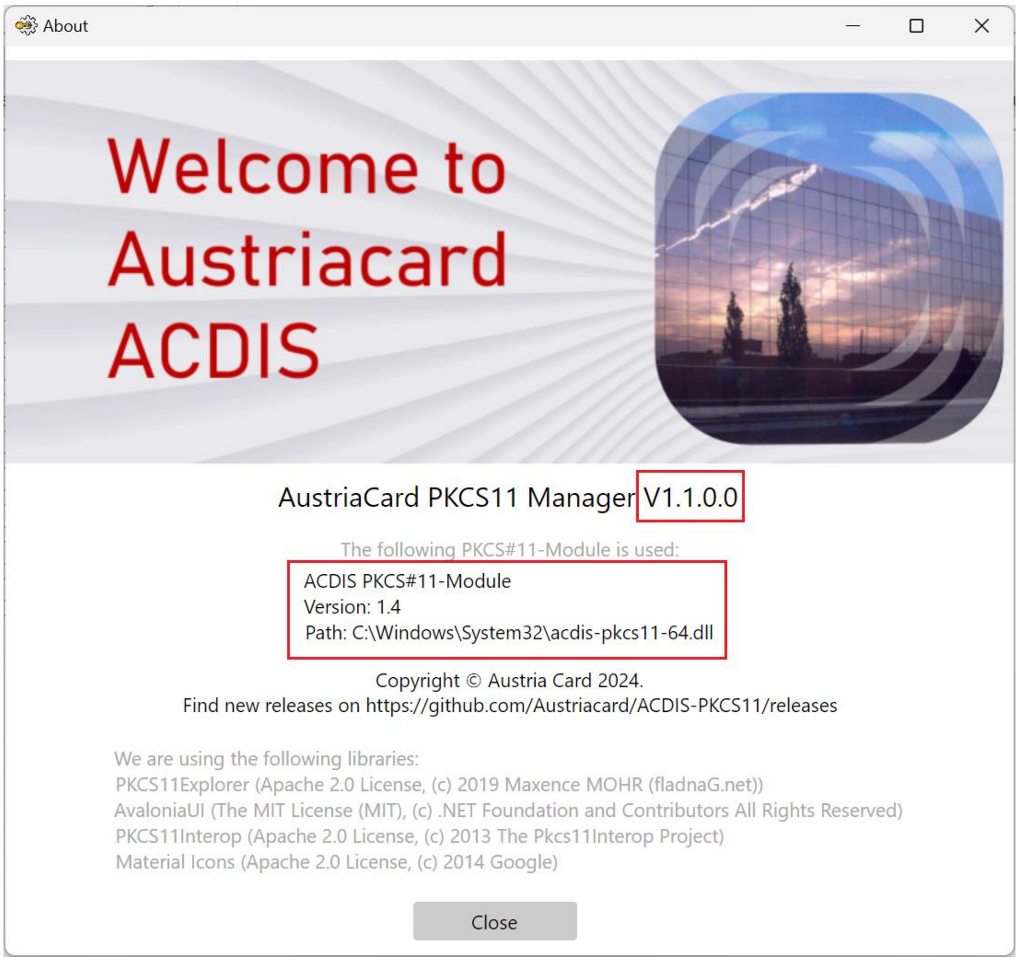
```
$HOME/.dotnet/corefx/cryptography/x509stores/
```

Dotnet applications look for certificates on this path. However, other applications probably do not.
However, there is no way to create certificates in the LocalMachine (= System) Store. Error:

```
"Unix LocalMachine X509Stores are read-only for all users."
   Only root is allowed to write under /etc/ssl/certs
```

5.22. ABOUT

The software versions are displayed in the About dialog:



AustriaCard PKCS11 Manager V1.1.0.0

The following PKCS#11-Module is used:

ACDIS PKCS#11-Module
Version: 1.4
Path: C:\Windows\System32\acdis-pkcs11-64.dll

Copyright © Austria Card 2024.
Find new releases on https://github.com/Austriacard/ACDIS-PKCS11/releases

We are using the following libraries:
PKCS11Explorer (Apache 2.0 License, (c) 2019 Maxence MOHR (fladnaG.net))
AvaloniaUI (The MIT License (MIT), (c) .NET Foundation and Contributors All Rights Reserved)
PKCS11Interop (Apache 2.0 License, (c) 2013 The Pkcs11Interop Project)
Material Icons (Apache 2.0 License, (c) 2014 Google)