SmartCard-HSM
XKEK Key Domain HowTo

Author	Andreas Schwier
Version	1.0
Date	2020-09-04

Table of Contents

1 Introduction

This document is a step-by-step description for configuring XKEK Key Domains with a SmartCard-HSM 4K.

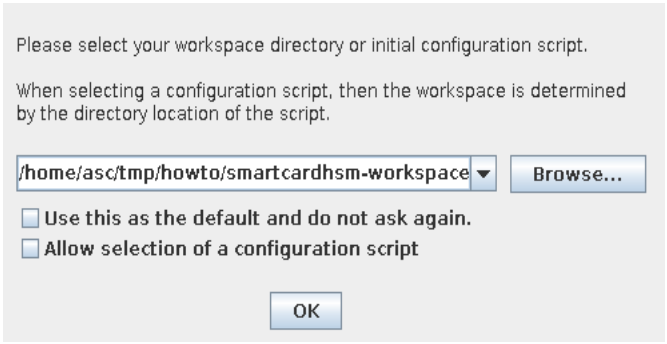
2 Prerequisite

2.1 Installing Software

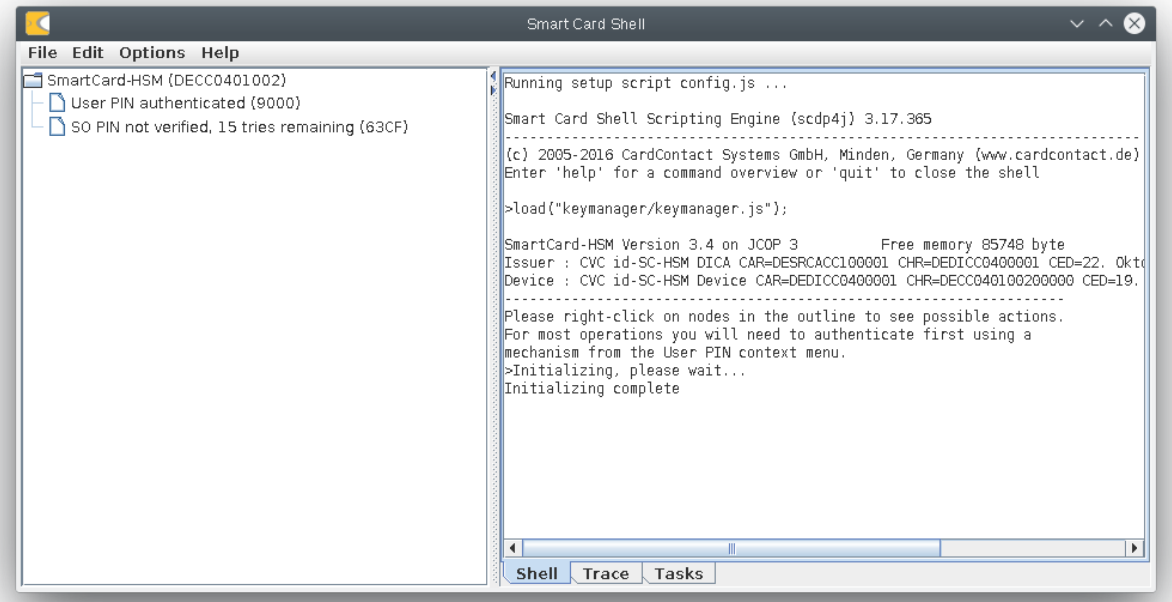
You need to install the Smart Card Shell at least in version 3.16 available from www.openscdp.org/scsh3.

2.2 Starting the Smart Card Shell

Start the Smart Card Shell and select a workspace directory. This can be any folder on your system, for example the workspace folder from the SmartCard-HSM Starterkit.



Run the KeyManager with "File/Key Manager (CTRL+M)".

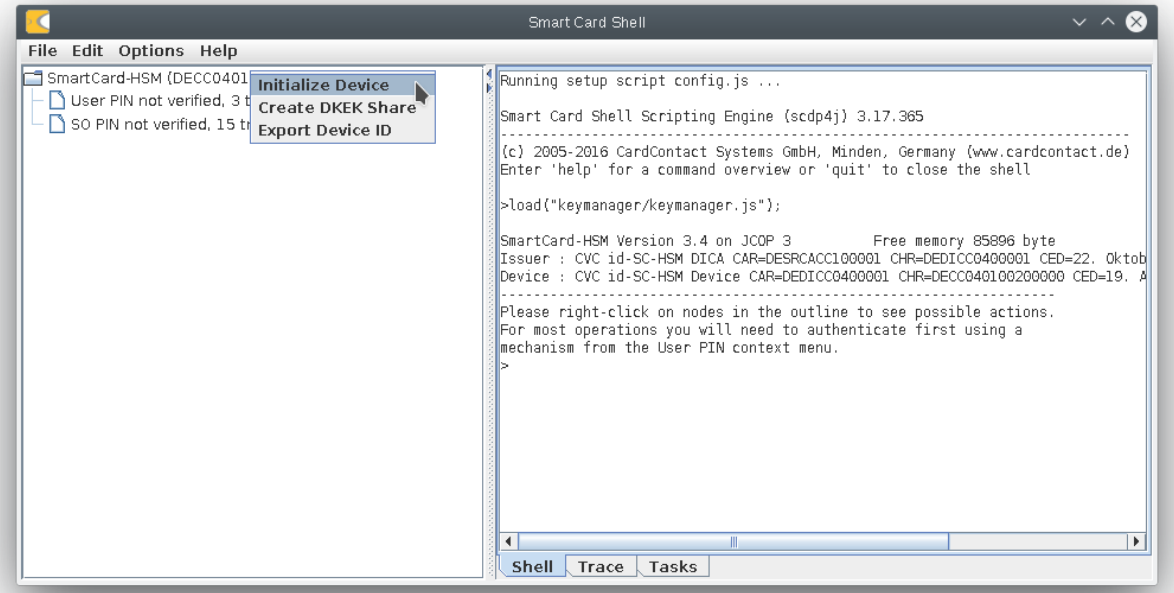


The shell shows some status information of the device, in particular the device id (here DECC0401002).

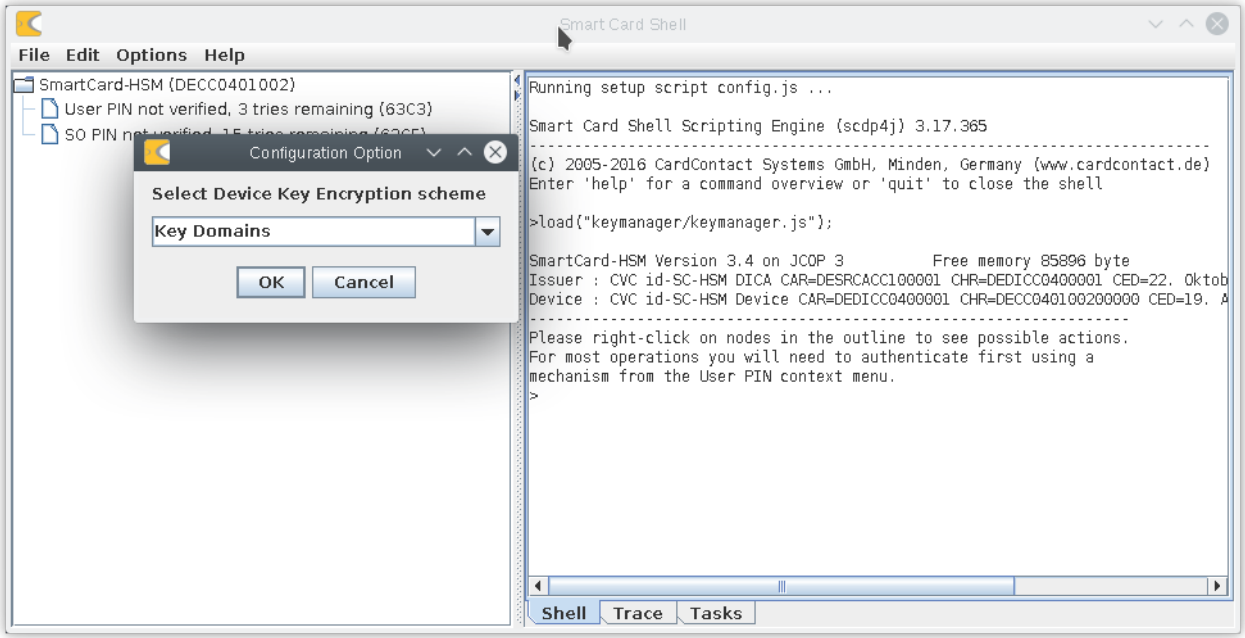
3 Setting up a XKEK Key Domain

3.1 Initialize Device for using Key Domains

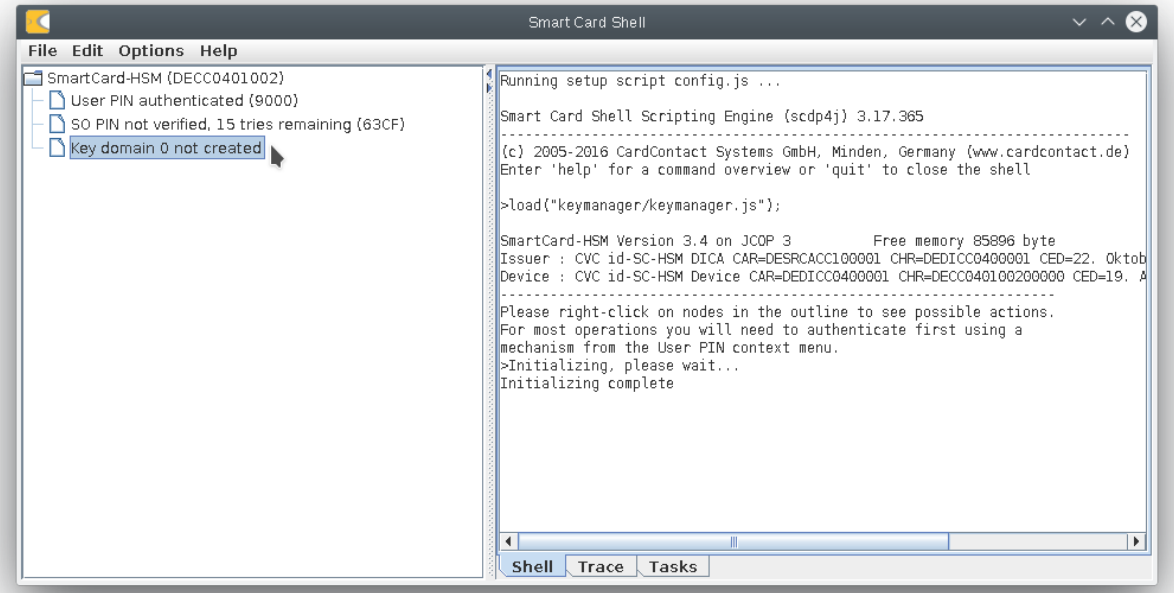
During SmartCard-HSM initialization you will need to specify how many key domains you want to configure. Select "Initialize Device" from the context menu attached to the "SmartCard-HSM" node in the outline.



Continue to the "Select Device Key Encryption Scheme" and select "Key Domains.



On the next dialog confirm 1 key domain. The device is initialized and shows one key domain slot that is not yet created.

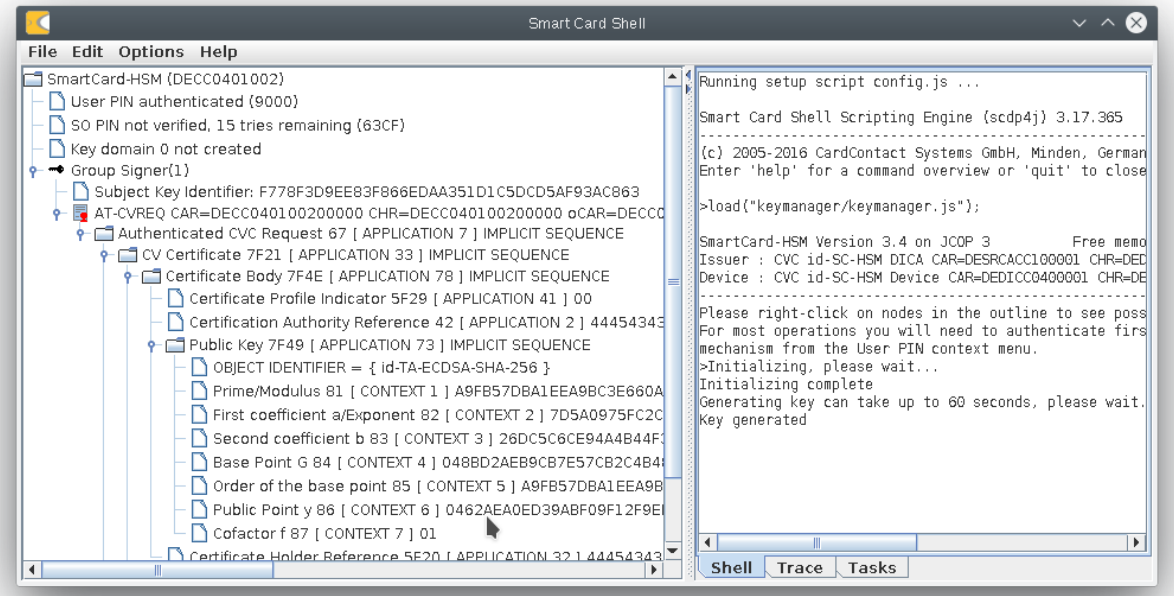


You can create up to 255 key domains and each one can be either a DKEK or XKEK key domain.

3.2 Create the Group Signer

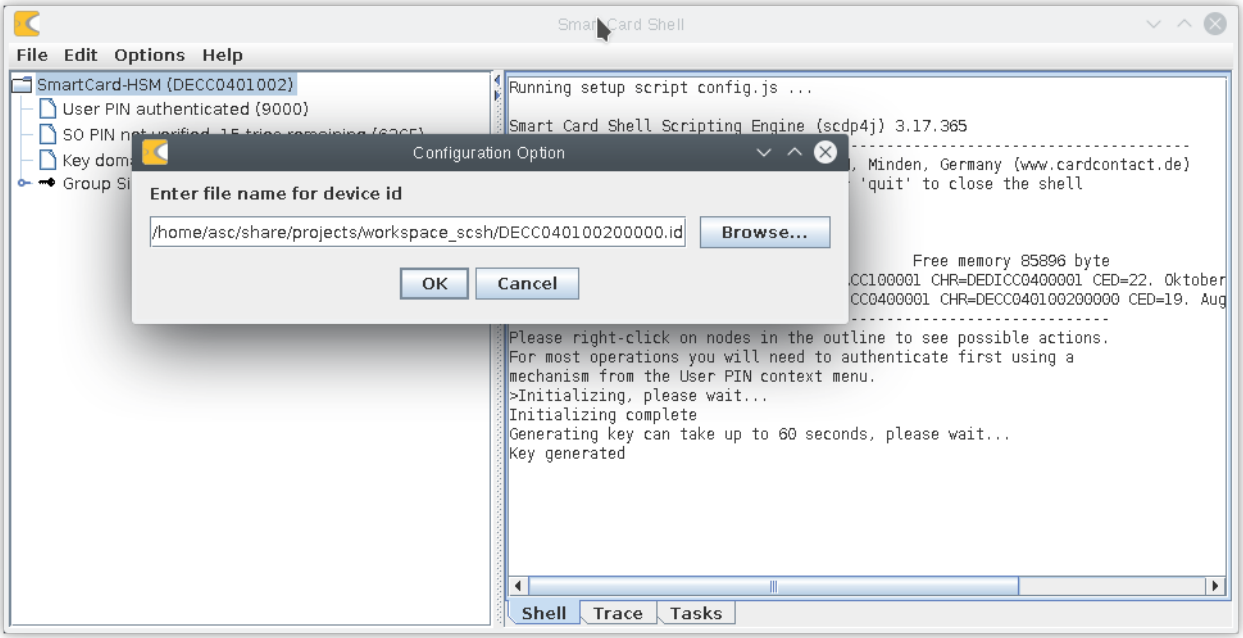
The Group Signer is the certification instance that issues Key Domain Membership certificates for members of the key domain group. The signer is an ECDSA key on the brainpoolP256r1 curve. It can be created on any SmartCard-HSM. For demonstration purpose we create the key on the same device, using "Generate ECC key" from the SmartCard-HSM context menu. Choose "brainpoolP256r1" and a label "Group Signer". Leave algorithms empty or enter "70" for ECDSA.

If you now open the AT-CVREQ structure with the newly generated public key, then you can see the Public Point y 86, which starts with '0462AEA0'. The 32 bytes following the '04' (the x-coordinate of the public point) are the key domain UID that uniquely identifies the group signer and a key domain instance on a SmartCard-HSM.

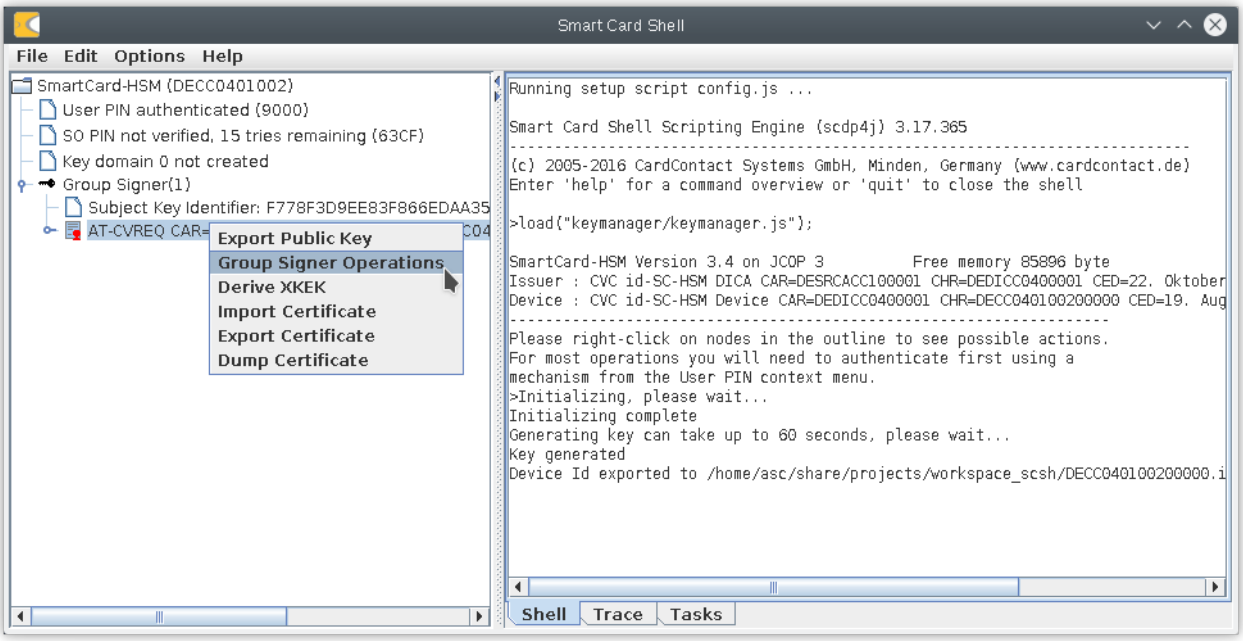


3.3 Adding a Device to the Group

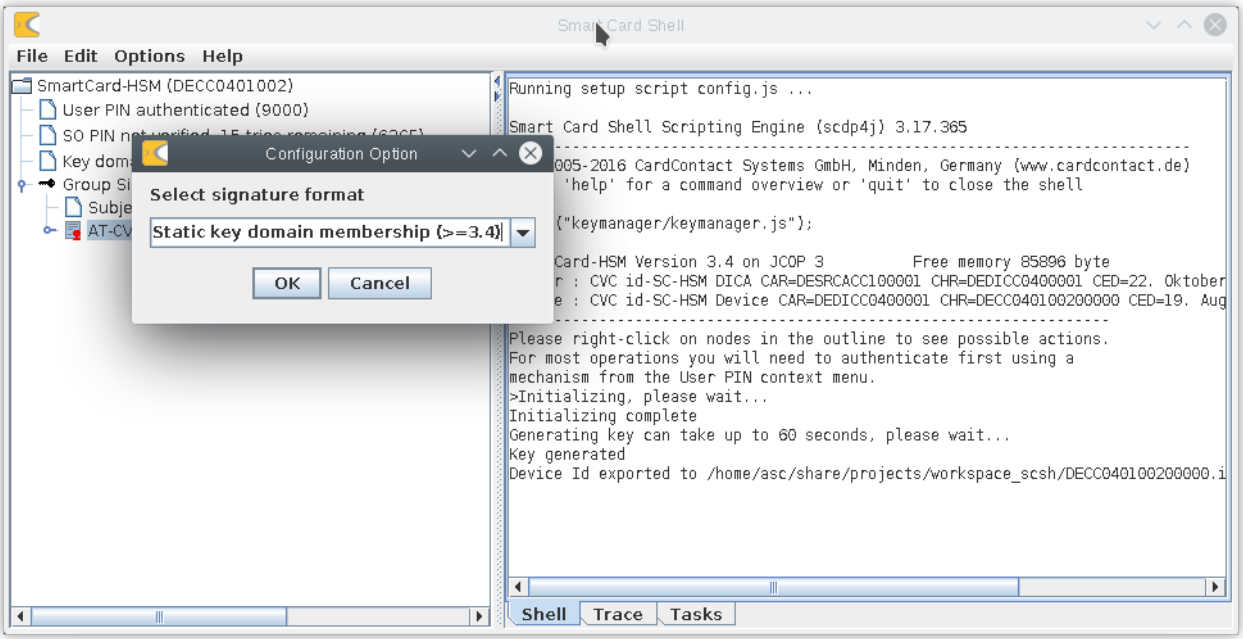
To add a SmartCard-HSM to the group controlled by the group signer, the device authentication public key must be certified in a Key Domain Membership (KDM) certificate. In order to do that, you will first need to export the Device ID of the SmartCard-HSM you want to add. The data written to the file is actually a concatenation of the device certificate and the device issuer's CA certificate. Choose "Export Device ID" from the context menu and save with the file name proposed in the dialog.



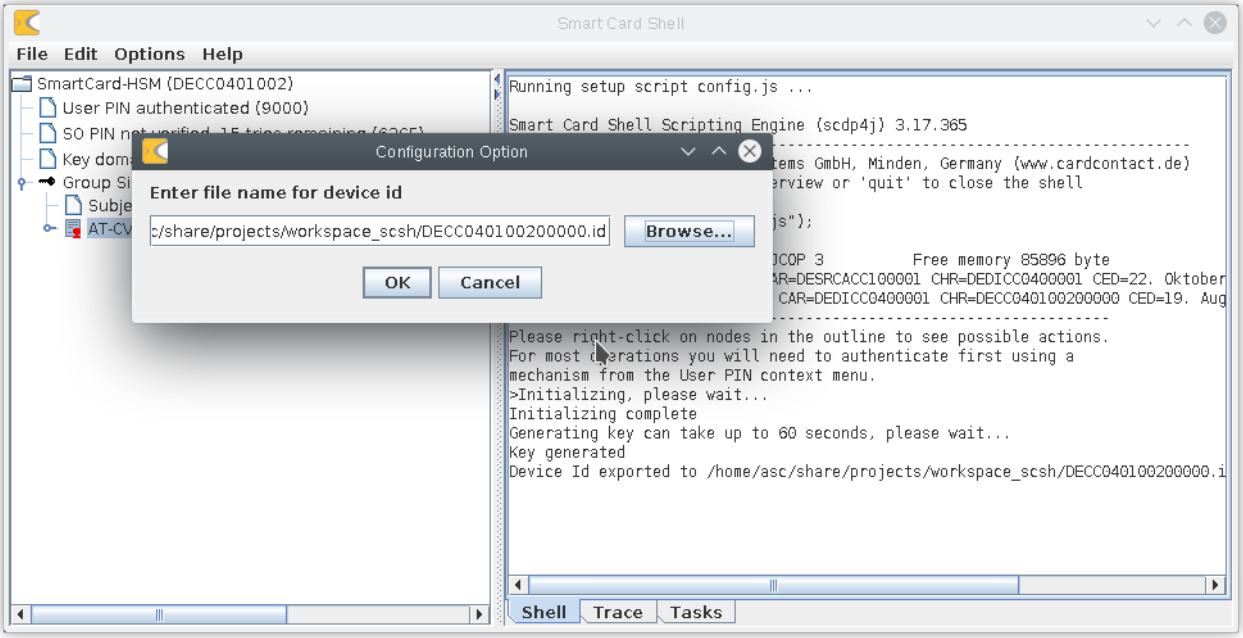
Next you will need to issue the KDM for the device. Right-click on the AT-CVREQ node under the group signer and select "Group Signer Operations".



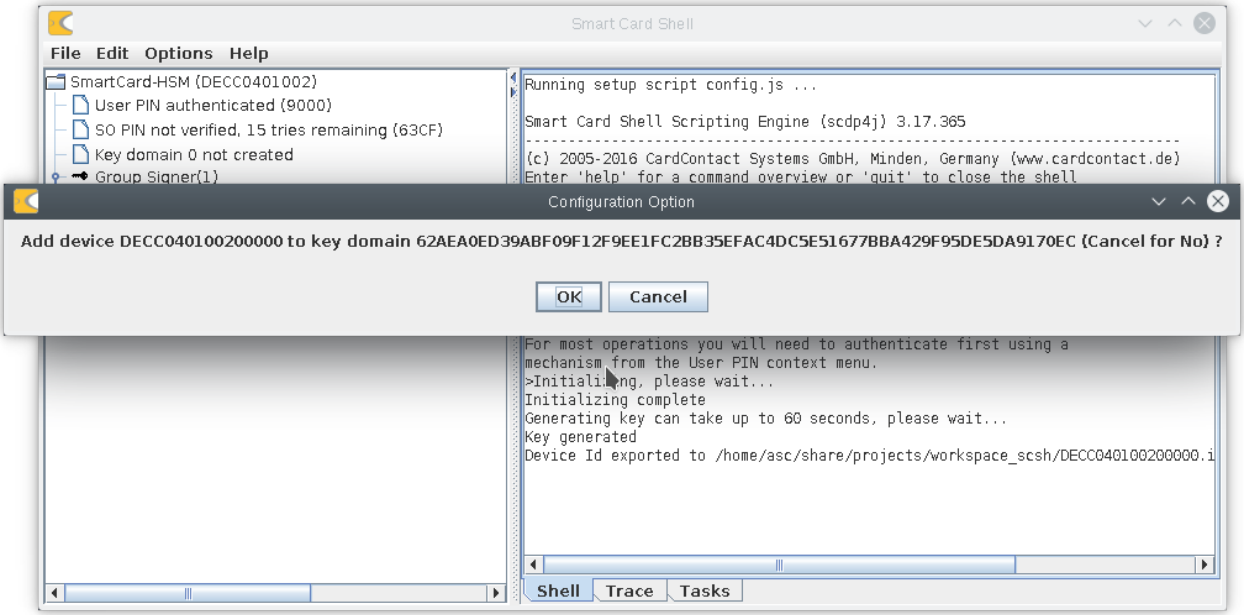
You can now select, if you want to issue a KDM for a device with version less than 3.4 or above.



After "OK" you are prompted for the file containing the device id of the device you want to add.

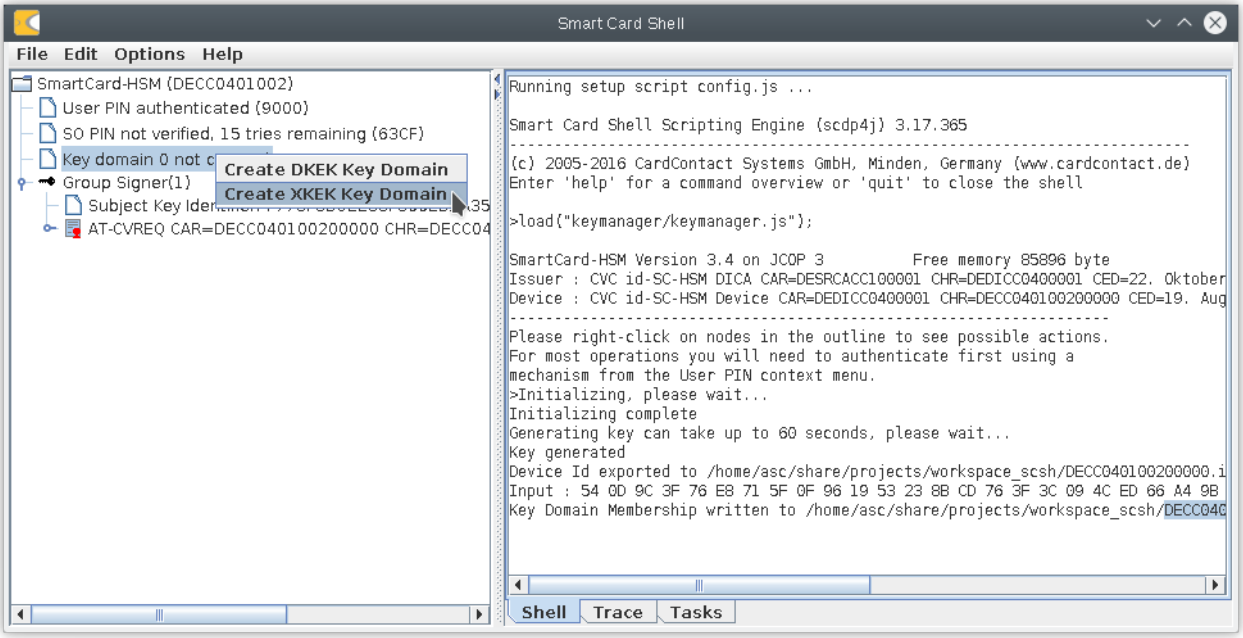


The script will now ask, if you really want to add that device to the key domain controlled by the selected group signer. You can see that the key domain UID matches the public key of the group signer.

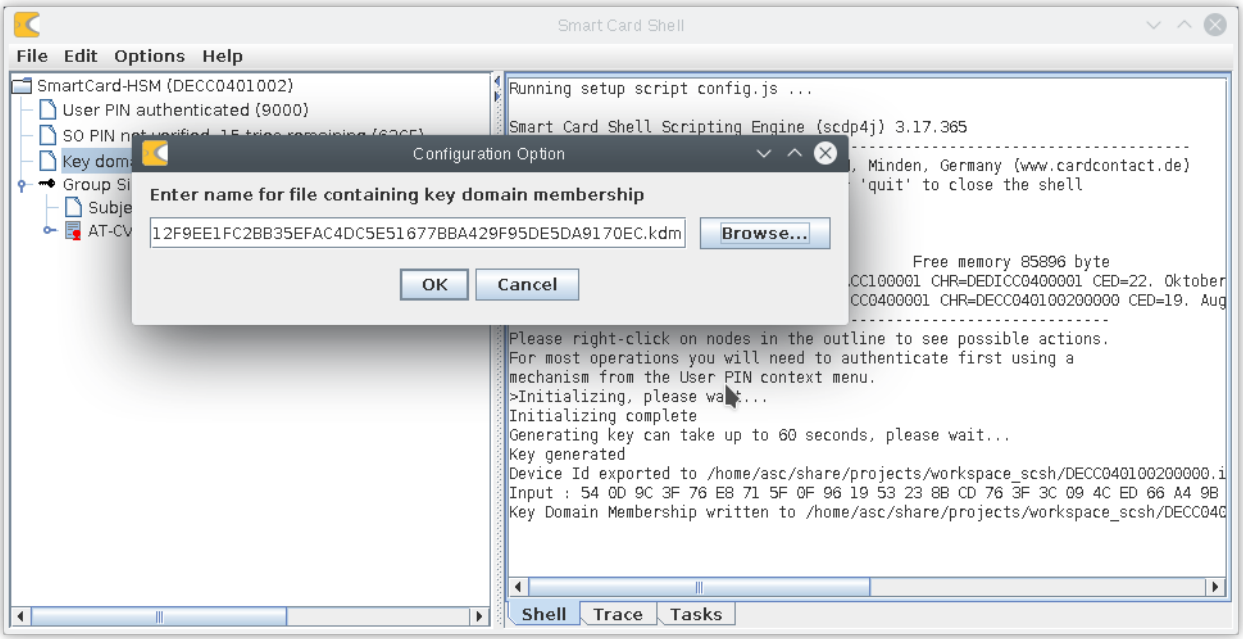


The script automatically writes a file with extension ".kdm" in the workspace (here DECC040100200000-62AEA0ED39ABF09F12F9EE1FC2BB35EFAC4DC5E51677BBA429F95DE5DA9170EC.kdm).

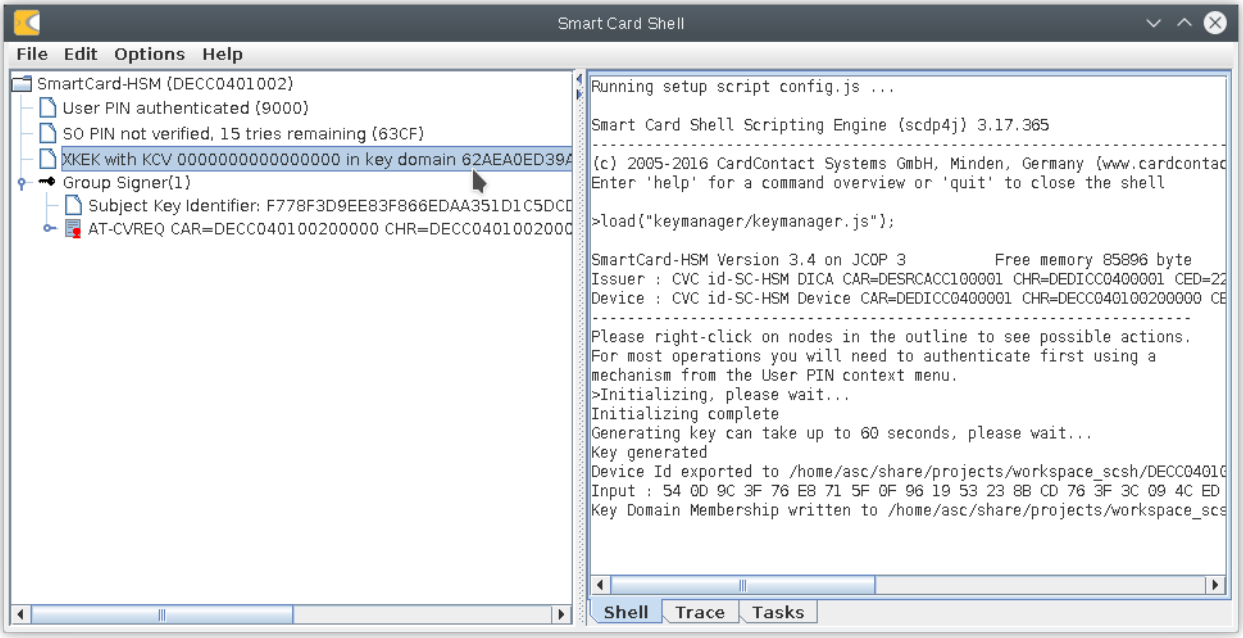
Now you can create the key domain instance for that device in the key domain slot allocated during initialization. Right click on the key domain entry and select "Create XKEK Key Domain".



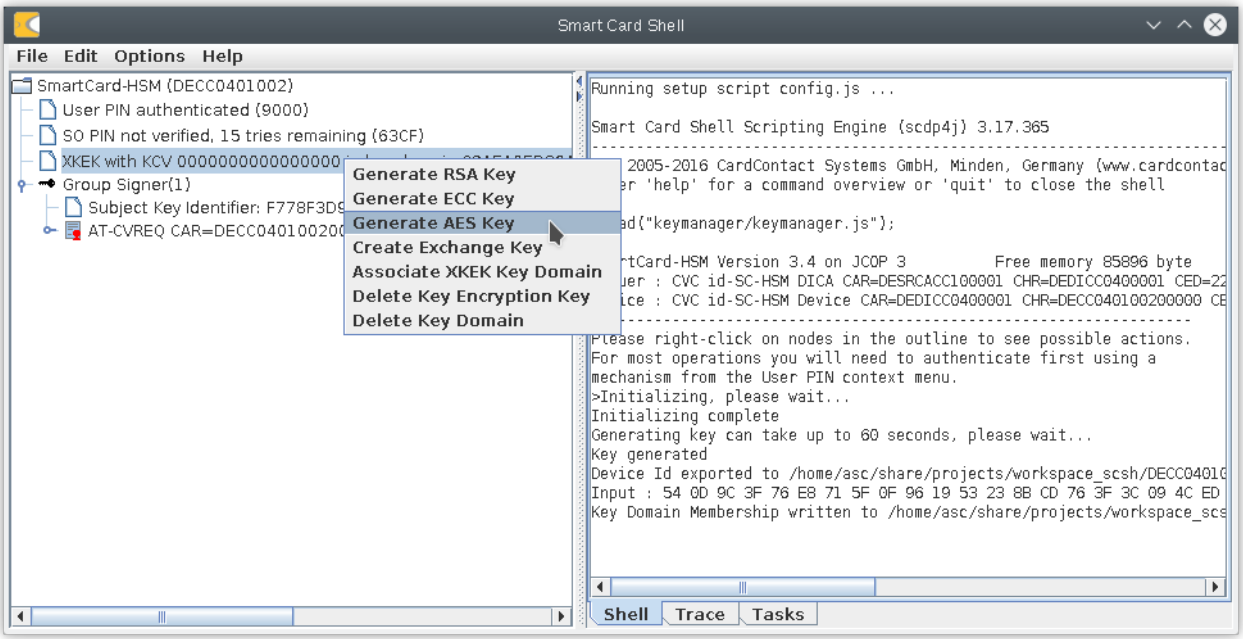
In the next step you need to select the .kdm file that contains the key domain membership certificate issued by the group signer.



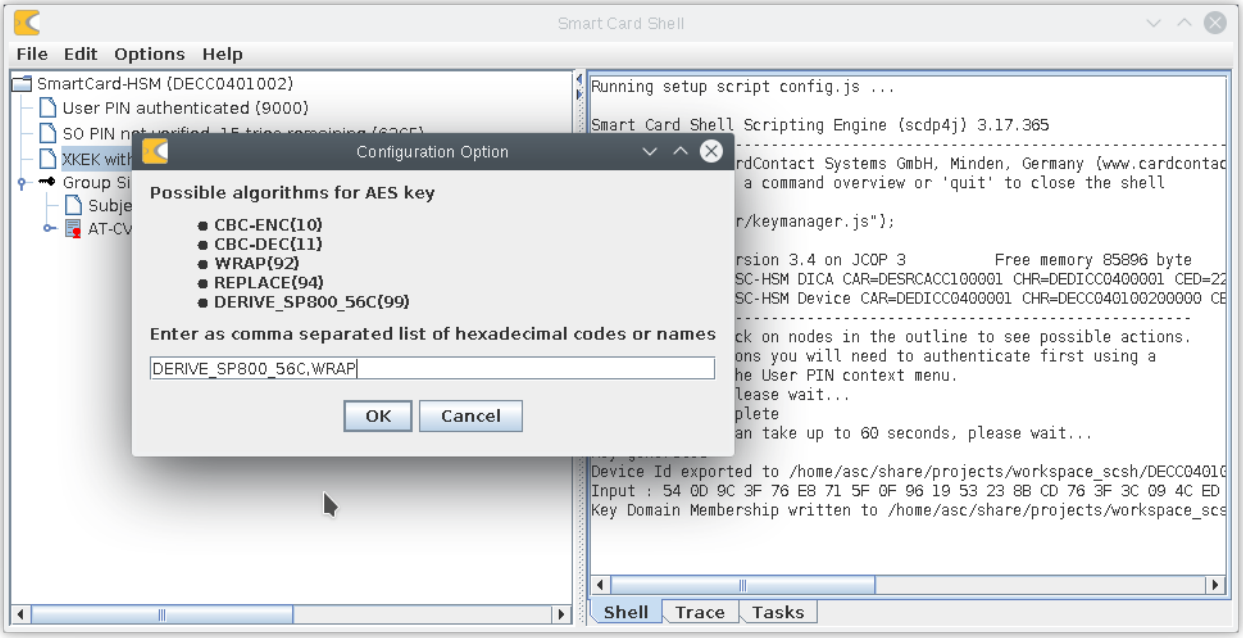
After pressing "OK" the key domain instance is created. You can again see the key domain UID shown on the node.



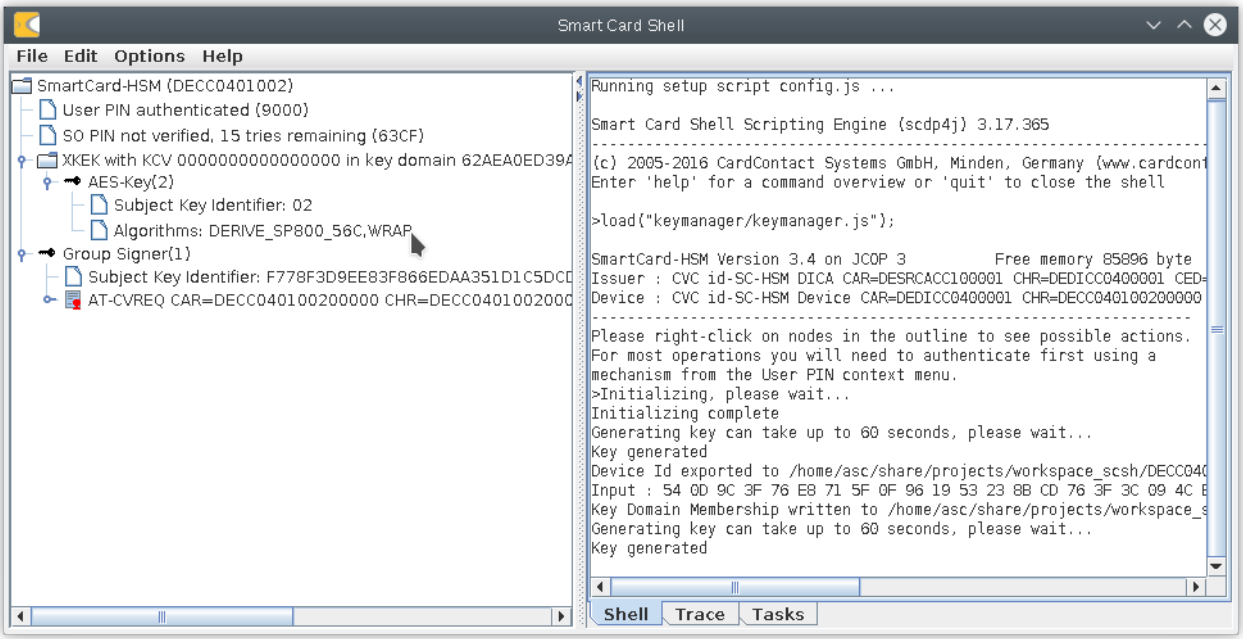
Now that we have created a XKEK Key Domain, we can start generating keys in it. Let's right-click on the key domain and select "Generate AES Key".



For the new key we select "WRAP" to allow exporting the key from the key domain.



The new key is shown as a member of the key domain. It can be used just like any other key. It shows up normally in PKCS#11 or the Java Key Store.



The KCV of 00..00 indicates that currently no XKEK is present to wrap or unwrap keys.

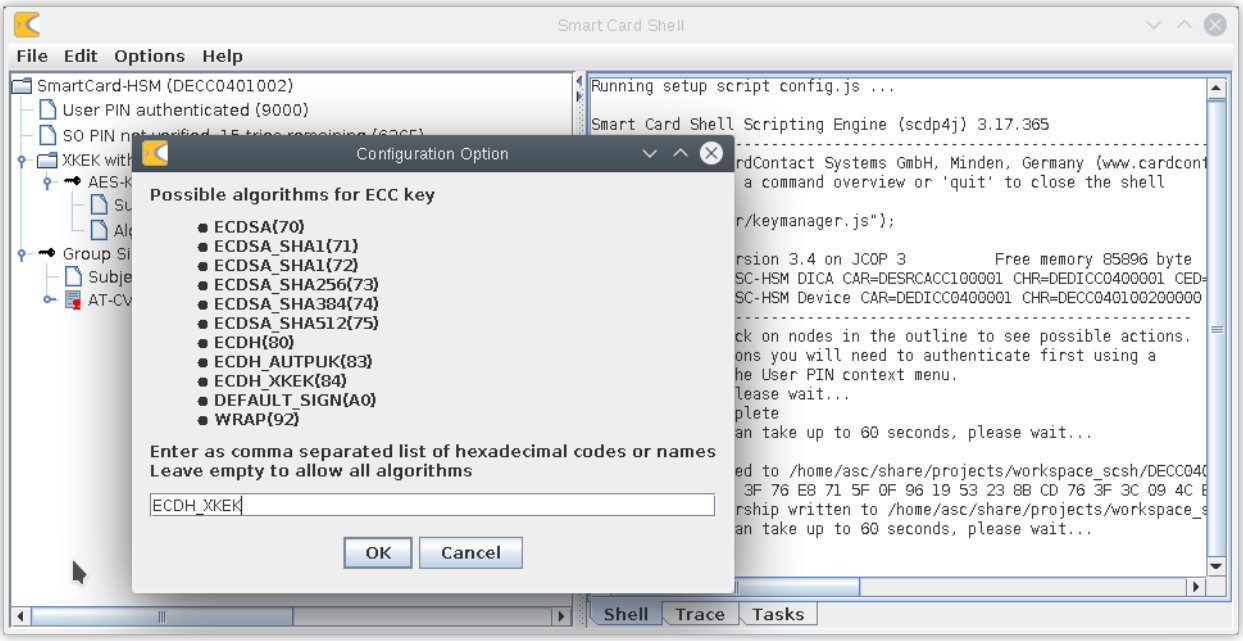
4 Migrating Keys in a XKEK Key Domain

Now that we have created a XKEK Key Domain and a contained AES key, we want to migrate that key to another SmartCard-HSM that is also part of the key domain. You will need to repeat step 3.3 for each device you would like to add.

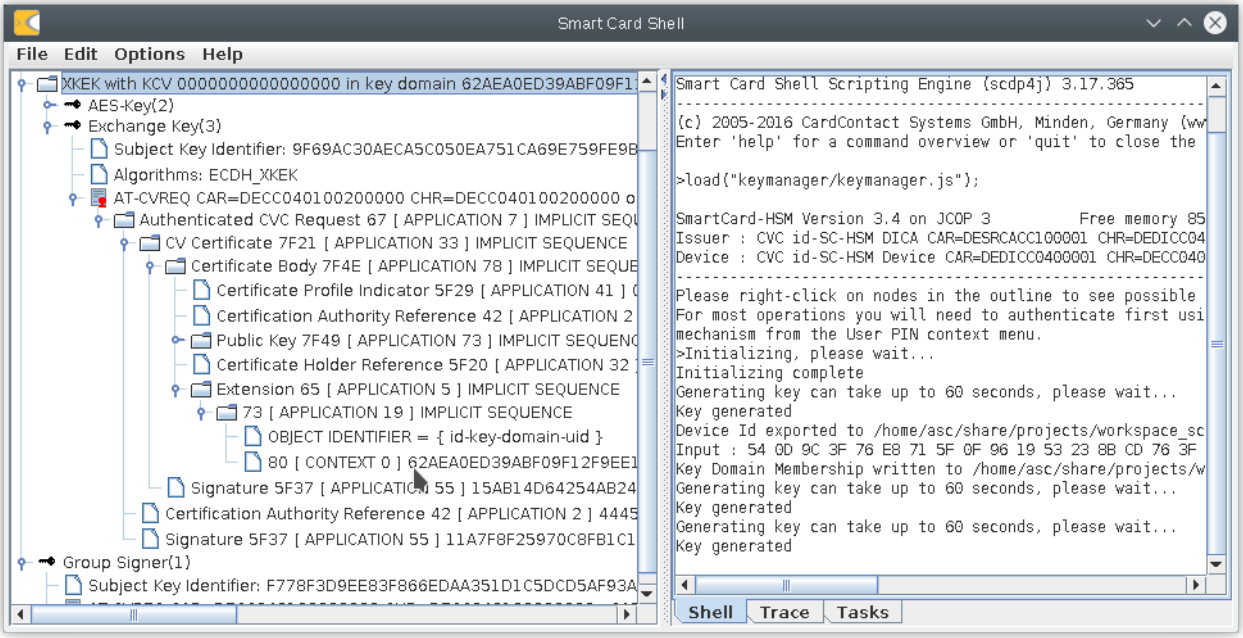
For the purpose of this demonstration, I will use a single device. The steps are identical if you export and import into the different devices.

For exporting a key from the key domain we first need to have a Key Encryption Key (KEK). Unlike with DKEKs that are imported using key shares, a XKEK is dynamically created as the result of performing a key agreement with EC Diffie-Hellman. Both, sender and receiver must generate an EC key pair and make the public key available to the other side. Each side then performs an ECDH operation with it's private key and the other side's public key. The result is a shared XKEK on both side. This XKEK is then used to wrap keys for export or to unwrap key after import.

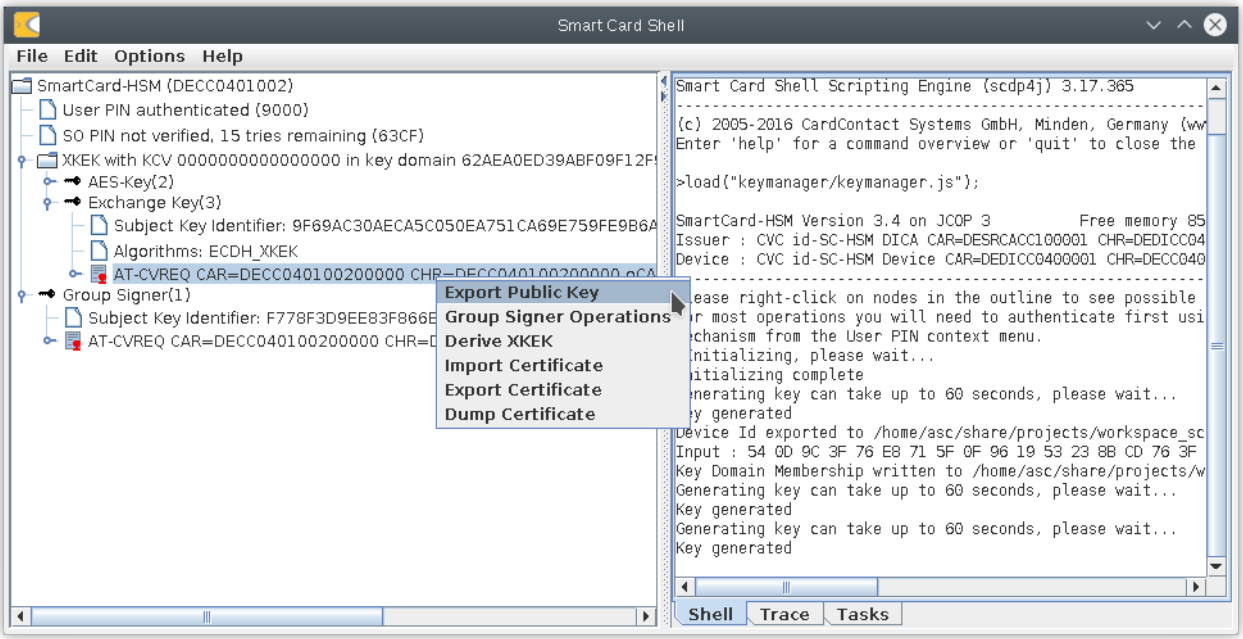
The first step is to create the EC key pair for the key agreement operation. Choose "Generate ECC Key" from the key domain's context menu. It is important to select the "ECDH_XKEK" algorithm (and only this algorithm).



If you now look into the AT-CVREQ of the newly generated public key, then you can see that it contains an extension id-key-domain-uid that indicates the key domain in which this key was generated. The id-key-domain-uid extension is only added for ECC keys with ECDH_XKEK set in the algorithm list.

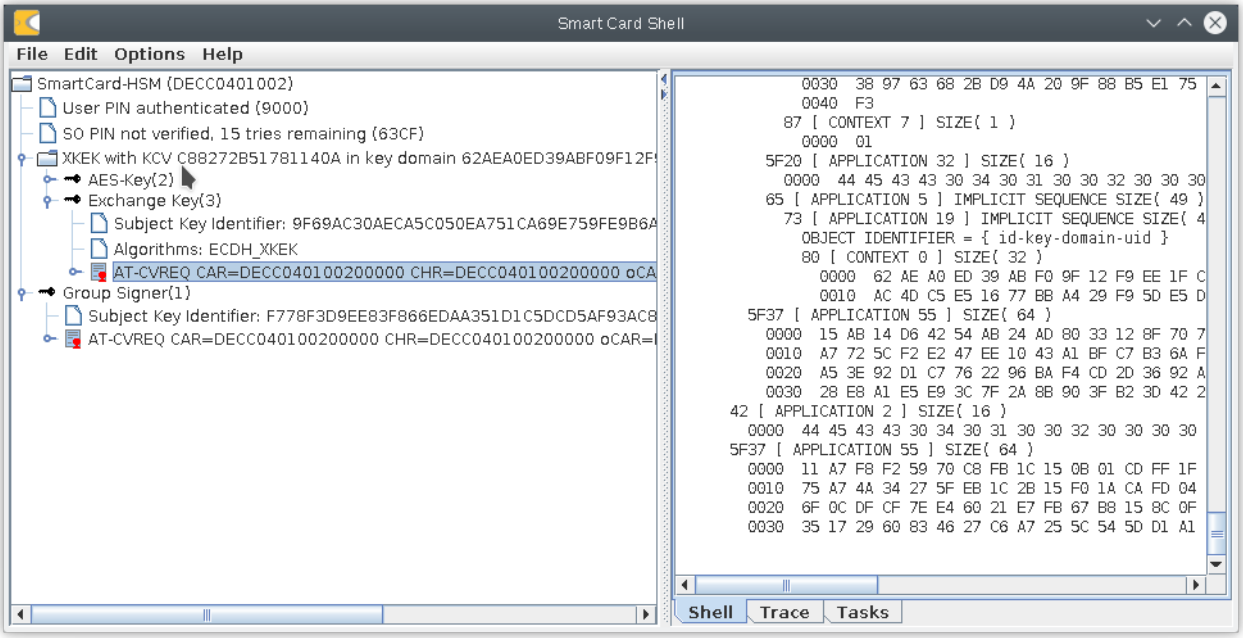


We save the AT-CVREQ and it's certificate chain into a file using "Export Public Key".

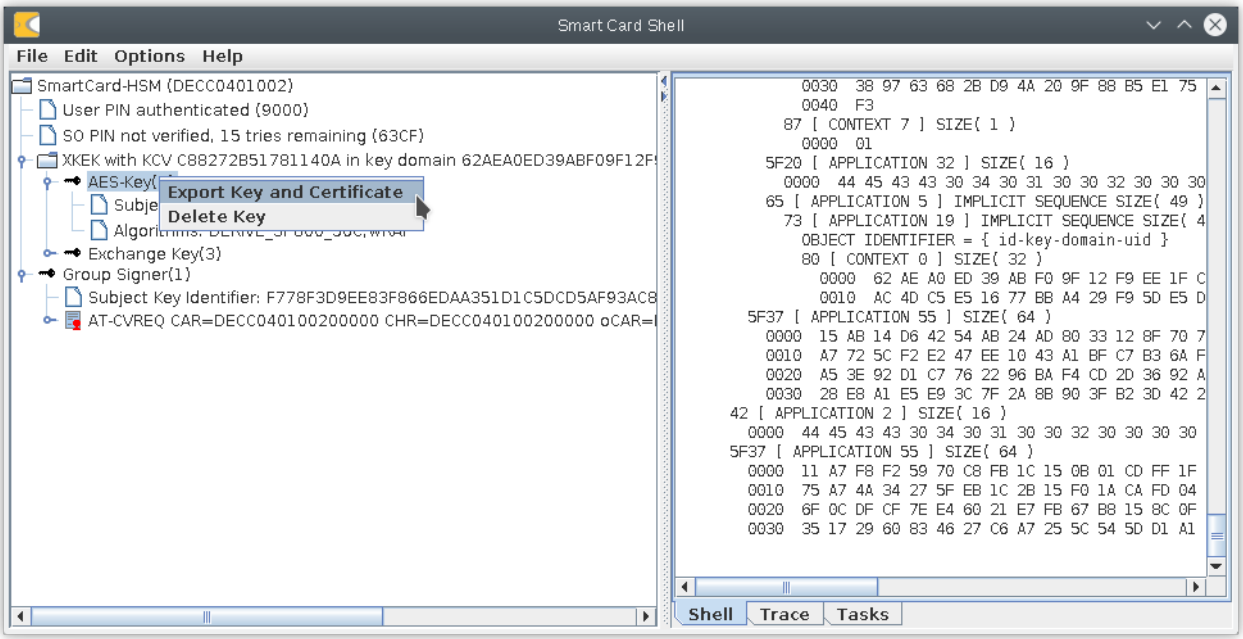


Having done that on both sides, we are ready to agree a XKEK. For that we choose "Derive XKEK" and select the file containing the public key of the other side (our own public key will work as well).

As a result the XKEK node shows the Key Check Value (KCV) of the freshly agreed XKEK.

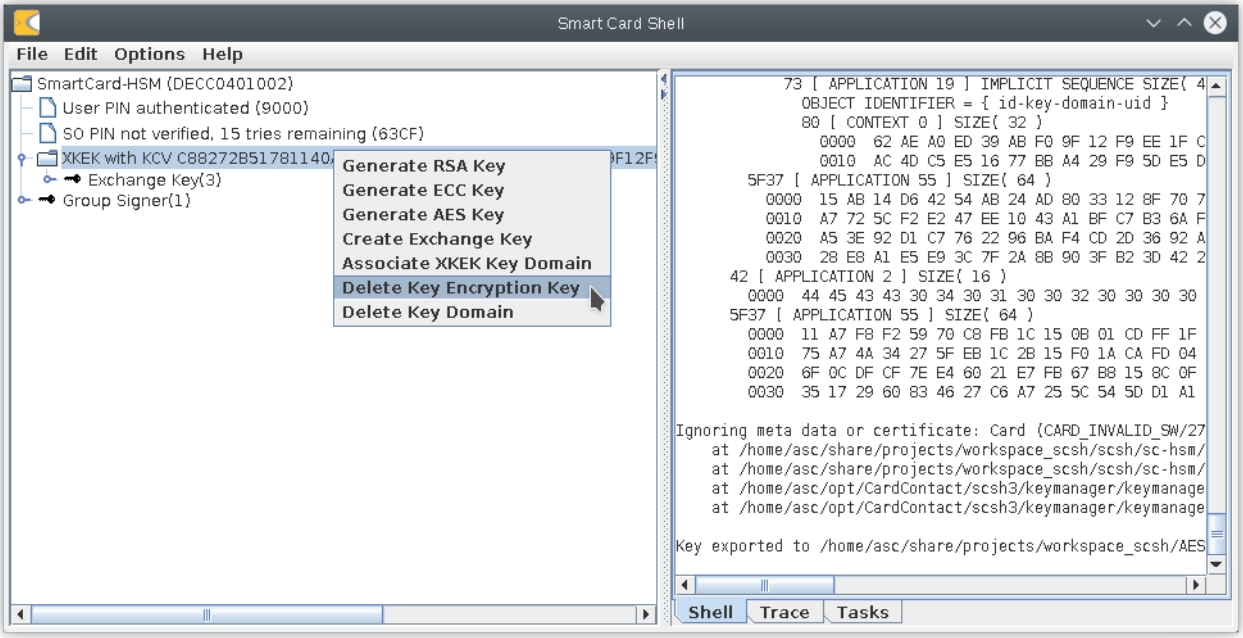


Now that we have a XKEK, we can wrap and export our AES-Key. Select "Export Key and Certificate" on the AES-Key node (AES keys of course don't have a certificate, so you can ignore the warning).

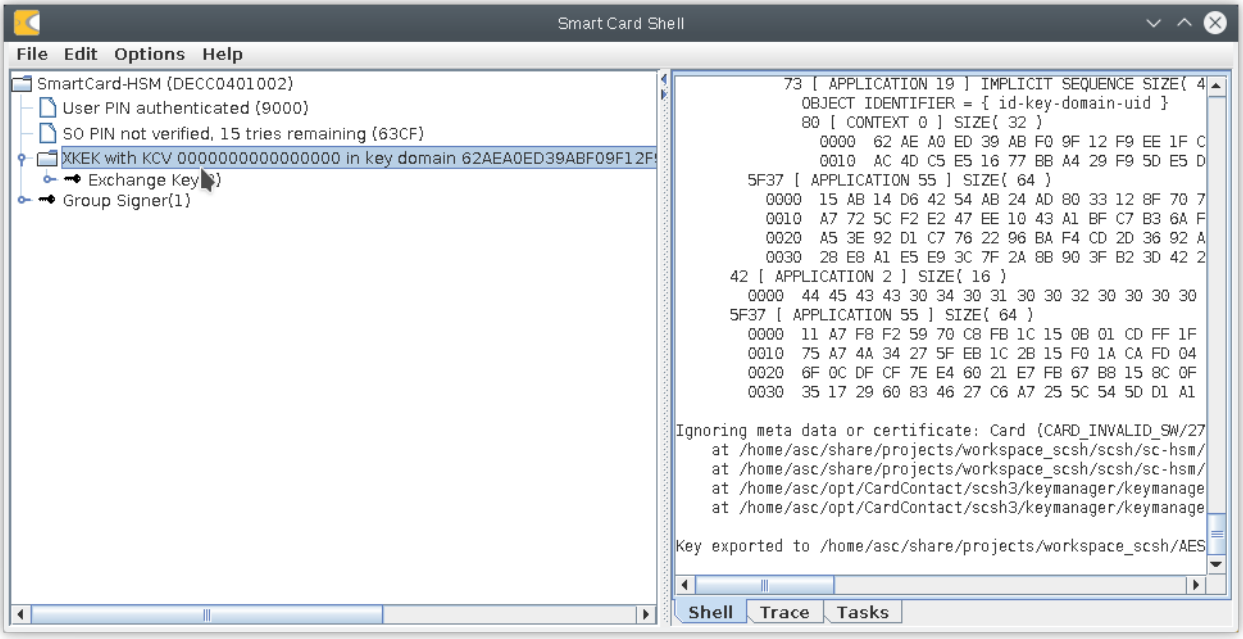


After selecting the file name, the key is wrapped under the XKEK, exported from the device and written into the file.

We can now delete the AES key and clear the XKEK with "Delete Key Encryption Key" from the key domain's context menu.

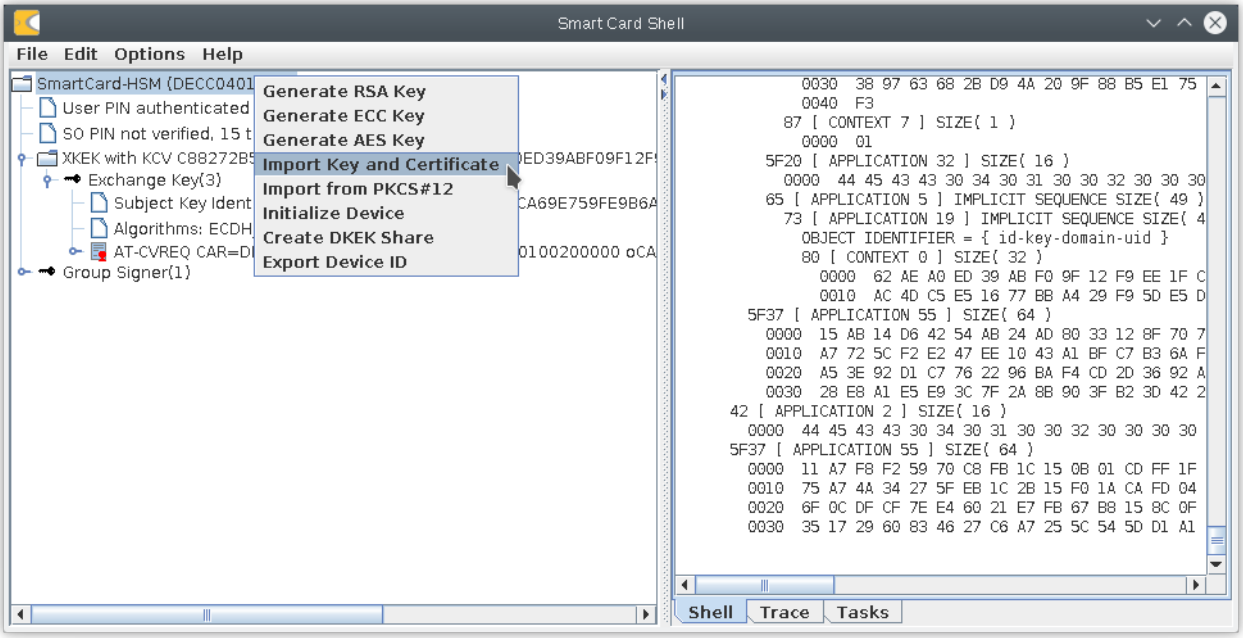


The XKEK is cleared and the KCV is reset to 00..00

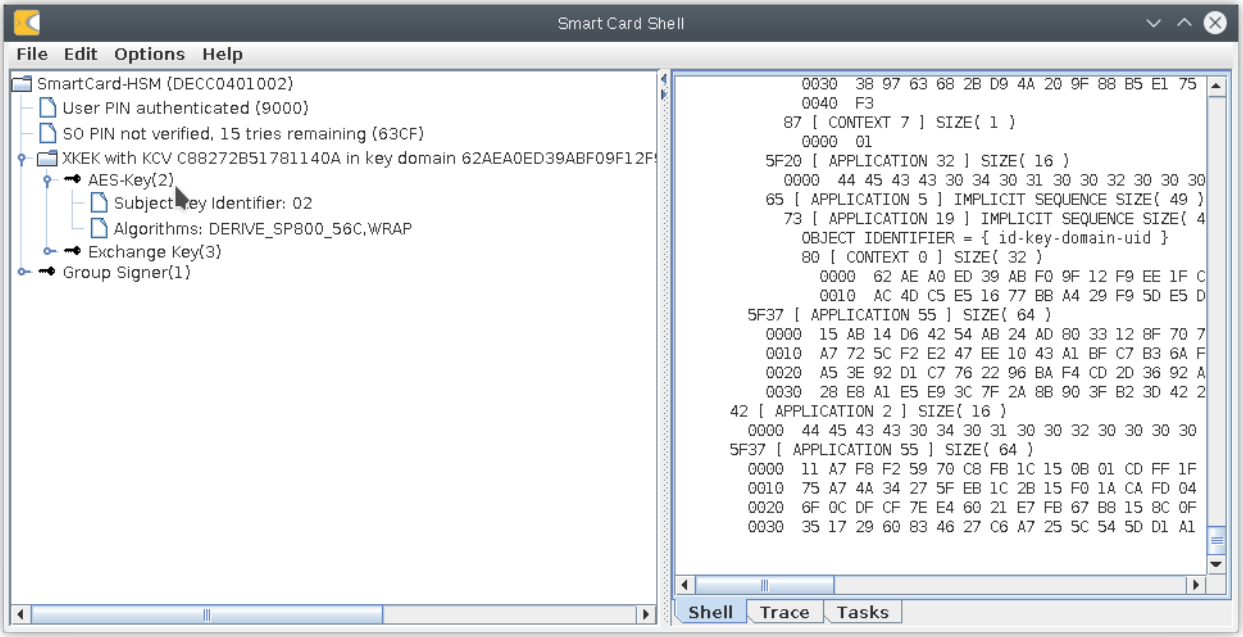


To import the key we need to repeat the XKEK key agreement using the public key of the sender in the "Derive XKEK" function. That generates the XKEK, whose KCV is show at the node.

For importing the AES-Key you need to select "Import Key and Certificate" from the SmartCard-HSM node. No need to select the right key domain – that is done automatically, as the wrapped key contains the KCV of the XKEK in it's meta data.



Et voila, the key is back in the device.



5 Summary

Key Domains are a powerful feature of the SmartCard-HSM to implement secure key management.

While DKEK key domains with imported key shares are easy to set-up, they put a high burden on the organizational overhead to securely generate and control DKEK shares. If you loose control over your DKEK shares, then confidentially of your keys is at risk. DKEK key shares allow to recover the Key Encryption Key

that wraps your key material exported from the SmartCard-HSM.

XKEK Key Domains are different and there is a guarantee, that a key can never leave a XKEK key domain. The Key Encryption Key is always the result of an ECDH operation and the private key can not leave the key domain either. Effectively, there is no way to establish the XKEK outside of the SmartCard-HSM, so it is impossible to decrypt the wrapped key material.

Implementing key backup with a XKEK Key Domain means adding a SmartCard-HSM to the key domain and migration keys into that device. This is controlled by the group signer that you need to keep under control. If you need to replace a broken device, just add a new device to the group and you are ready to migration key material into that device.

As you've seen from the steps above, managing a XKEK manually is a complex task. Our suggestion is to write key management scripts using the SmartCard-HSM. You can take a look at the key manager script (keymanager/keymanager.js in the Smart Card Shell installation) to learn how that can be done.