PKCS11 Smart Card and TPM DNSSEC Demo Training Material

Richard Lamb and Luis Espinoza 20120927

SMARTCARD HSM UPDATE **Richard Lamb 20130819**

Please note: This site will be moving to http://www.co.tt/ri.co.cr/

We have 5 demo examples:

- Offline Smart Card KSK + Online software ZSKs
- Offline HSM KSK + Online software ZSKs using fake HSM
- Offline Smart Card KSK + Online Smart Card ZSKs
- Online Smart Card KSK + ZSKs + BIND 9.9 in-line signing
- Online TPM KSK + ZSKs + BIND 9.9 in-line signing

Note: The PKCS11 standard allows for a simplified upgrade path to HSMs. Smartcards and TPMs do on the order of 1 1024 RSA signature per second while an HSM can do greater than 1000/s. Although key backup and inialization strategies vary across devices, the C_Sign function call to generate RSA signatures is consistent across all. The examples on the demo DVD use BIND 9.9 tools with the modification of one file - bind/lib/dns/opensslrsa_link.c - to natively support PKCS11. The modified single bind-9.9.1-P2 file and the rest of the source is on the DVD.

For smart cards:

- get a USB smartcard reader (SCR331 $15)
- get a smartcard (Aventra $11)

- boot DVD and login as root password dnssec (900M ISO file for complete bootable Smartcard and TPM DVD here sha256=c5045720002064a838d8597011c81c7fb9a01a1e11525d4a1201d163f3fea0f4)
- plug in reader and insert smartcard. (card reader light, if it has one, should blink indicating pcscd daemon has recognized the card)

Note: If not using the Aventra MyEID PKI smart card 2012, replace PKCS11_LIBRARY_PATH="/opt/dccom/lib/opensc-pkcs11.so" with different pkcs11 library in various scripts such as the ones below. I have tried Athena SCS IDProtect LASER, Feitian PKI, and a few other cards and unfortunately each card vendor have very different techniques for initializing and formatting cards so all the routines will have to be customized for each vendor. The Aventra cards are easy to purchase in small quantities. However, the smallest vendor change (e.g., ATR,..) can render the OpenSC PKCS11 driver useless (this is a case in favor of proprietary driver+card like Athena SCS). So there is no guarentee that this setup will work if any element is changed.

- carderase
- cardrng
- cardsign
- genksk-sc
- genzsk-sc
- signem-sc
- signzone-99

Routines that also depend on Aventra card:

- carddel
- carderase
- cardshow
- cardwrite

Offline Smart Card KSK + Online software ZSKs

- carderase* (Use 123456 for PIN and Security Officer PIN if asked)
- export DOMAIN=yourdomain (not ending dot)
- optional: export TEST="yes" (short signature times for testing)
- cardrng (in a second terminal window. PIN from above. If you want to now test the RNG, in another window do "cat /dev/random | rngtest", wait a minute, and then ctrl-C. rngtest should return some stats.)

- genzsk
- genksk (filename like "temp")
- Stop cardrng and exit out of window
- cardwrite (CKA_LABEL like "Kdate", filename and PIN from above.)
- optional: insert another card, carderase, cardwrite to create KSK backup
- cardshow
- cardsign (Use "abc" for passphrase to encrypt keybundles, KSK CKA_LABEL and PIN from above)
- optional: signzone (Use "abc" for passphrase for keybundles. starts local nameserver and runs sample signer process to maintain signatures)

* If using Feitian PKI card, use "carderase-ft" instead of "carderase". The rest of the instructions remain the same.

See the contents of various comands found in /opt/dccom and output in demo directory /tmp/namedb for details (e.g., signemd.out). "signemd" will automatically maintain the signed zone using software ZSKs and KSK signed DNSKEY RRsets created by "cardsign" above. This parallels the pre-generated DNSKEY RRset approach used at the root. For security, do not place all pre-signed RRsets on the online signer machine.

Here is a sample DPS and Key Ceremony documentation corresponding to this demo. Other training material can be found here and here.

Doing "dig +dnssec -t soa yourdomain @127.0.0.1" should show you the signed zone SOA as it automatically gets updated.

If you would like a persistent demo, run "startx" from root prompt and use "install" icon from desktop. Study signzone and create own startup script, separating out key generation (to maintain separately), changing the /tmp directory to something more appropriate, and configuring a system to get unsigned zone updates. Your startup script "startup" might look like:

```
export PKCS11_LIBRARY_PATH="/opt/dccom/lib/opensc-pkcs11.so"
read -s -p "HSM PIN: " PKCS11_LIBRARY_PIN
echo ""
export PKCS11_LIBRARY_PIN
cd /tmp/namedb
/opt/dccom/signemd yourdomain
/opt/dccom/named -c /tmp/namedb/named.conf
```

Offline HSM KSK + Online software ZSKs using fake HSM

Follow exactly same steps as previous example but add "-n" to each command,e.g., "carderase-n" instead of "carderase"...and of course you do not have a second optional card. This example uses the software token that is included in the opencryptoki packedge. If you are curious, the demo key material is kept in /var/lib/opencryptoki/swtok.

Offline Smart Card KSK + Online Smart Card ZSKs

- carderase* (Use 123456 for PIN and Security Officer PIN if asked)
- export DOMAIN=yourdomain (not ending dot)
- optional: export TEST="yes" (short signature times for testing)
- genzsk-sc (PIN from above.)
- genksk-sc (PIN from above.)
- cardsign-sc (KSK CKA_LABEL is what genksk-sc returned t the end, e.g., Kyourdomain.+008+17118. PIN from above)
- optional: signzone-sc (PIN from above. Starts local nameserver and runs sample signer process to maintain signatures.)

* If using Feitian PKI card, use "carderase-ft" instead of "carderase". The rest of the instructions remain the same.

See the contents of various comands found in /opt/dccom and output in demo directory /tmp/namedb for details (e.g., signemd-sc.out). "signemd-sc" will automatically maintain the signed zone using smartcard ZSKs and KSK signed DNSKEY RRsets created by "cardsign-sc" above. This parallels the pre-generated DNSKEY RRset approach used at the root. For security, do not place all pre-signed RRsets on the online signer machine. Since the keys are generated inside the cards and most cards do not support key export, there is no backup.

Doing "dig +dnssec -t soa yourdomain @127.0.0.1" should show you the signed zone SOA as it automatically gets updated.

If you would like a persistent demo, run "startx" from root prompt and use "install" icon from desktop. Study signzone-sc and create own startup script, separating out key generation (to maintain separately), changing the /tmp directory to something more appropriate, and configuring a system to get unsigned zone updates. Your startup script "startup" might look like:

```
export PKCS11_LIBRARY_PATH="/opt/dccom/lib/opensc-pkcs11.so"
read -s -p "HSM PIN: " PKCS11_LIBRARY_PIN
echo ""
export PKCS11_LIBRARY_PIN
cd /tmp/namedb
```

```
/opt/dccom/signemd-sc yourdomain
/opt/dccom/named -c /tmp/namedb/named.conf
```

Online Smart Card KSK + ZSKs + BIND 9.9 in-line signing

- carderase* (Use 123456 for PIN and Security Officer PIN if asked)
- export DOMAIN=yourdomain (not ending dot)
- optional: export TEST="yes" (short signature times for testing)
- signzone-99 (PIN = 123456 from above. Generates KSK, ZSK, starts local nameserver as automated in-line signer)
- optional: monitor (a simple script using "dig" to periodically display RRSIG key tags)

* If using Feitian PKI card, use "carderase-ft" instead of "carderase". The rest of the instructions remain the same.

See the contents of /opt/dccom/signzone-99 and demo directory /tmp/namedb for details (e.g., /tmp/namedb/log/runlog). "named" with /tmp/namedb/named.conf will automatically maintain the signed zone using KSK and ZSK in the smartcard. An excellent BIND 9.9 basic example using software keys is here.

Doing "dig +dnssec -t soa yourdomain @127.0.0.1" should show you the signed zone SOA as it automatically gets updated. The "Activate:" field in the keys/*.private files indicates when named will start using the corresponding key. The SOA serial should increment then. "rndc signing -list yourdomain" shows signing status. "rndc sign yourdomain" forces a recalculation of signatures. For a key rollover, you can manually add and remove keys from the "keys/" directory after a new key has been introduced and published or could use the "Activate:" and other meta fields to effect a complete rollover.

If you would like a persistent demo, run "startx" from root prompt and use "install" icon from desktop. Study signzone-sc and create own startup script, separating out key generation (to maintain separately), changing the /tmp directory to something more appropriate, and configuring a system to get unsigned zone updates. Your startup script "startup" might look like:

```
export PKCS11_LIBRARY_PATH="/opt/dccom/lib/opensc-pkcs11.so"
read -s -p "HSM PIN: " PKCS11_LIBRARY_PIN
echo ""
export PKCS11_LIBRARY_PIN
/opt/dccom/named -c /tmp/namedb/named.conf
```

TPM Work

For TPM demo:

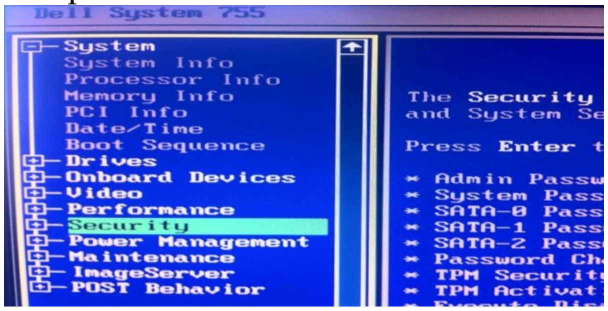
- get a machine that has a TPM. Newer ones are better than older ones. (Dell E4200, Optiplex 755, ...)
- make sure no apps are relying on it (e.g., Bitlocker, etc...)

Note: trousers, opencrptoki, tpm-tools are very finicky and building a local version for debugging requires pulling in a lot of cruft. Opencryptoki is a moving target but version 2.4 seems to have attained some stability but beware of old TPM systems that will fail after a few thousand executions of PKCS11 C_Sign. This is an excellent description of how opencryptoki implements PKCS11 using a TPM and how keys can be backed up.

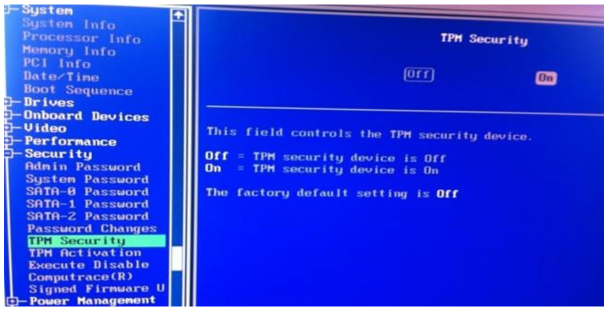
Versions for this demo on Centos 6.0:

```
tpm-tools-1.3.4-2.el6.i686.rpm
tpm-tools-pkcs11-1.3.4-2.el6.i686.rpm
opencryptoki-2.4.2-2. el6.i686.rpm
opencryptoki-libs-2.4.2-2. el6.i686.rpm
trousers-0.3.4-4. el6.i686.rpm
```

Online TPM KSK + ZSKs + BIND 9.9 in-line signing

- REBOOT MACHINE AND GO TO BIOS TO ACTIVATE AND CLEAR TPM FIRST.
  Sample BIOS screens for a Dell 755:
  

  Make sure "TPM Security" is "ON".



TPM Security

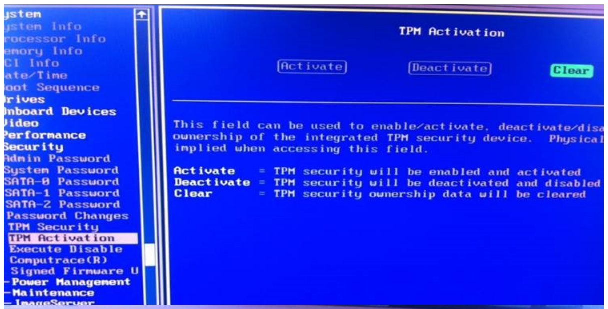
Off [✓ On]

This field controls the TPM security device.

Off = TPM security device is Off
On = TPM security device is On

The factory default setting is Off

CLEAR the TPM. CAUTION: any prior applications depending on the TPM will fail.

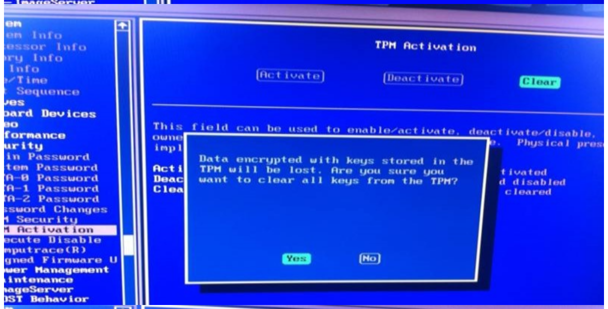


TPM Activation

Activate Deactivate [✓ Clear]

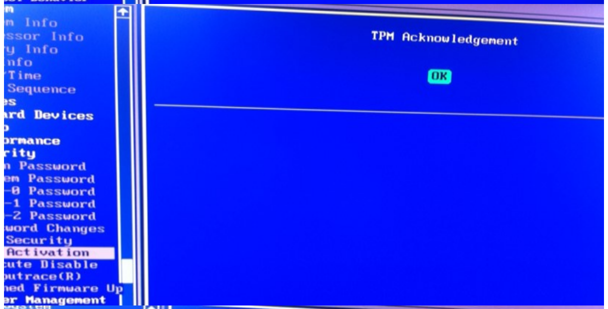
This field can be used to enable/activate, deactivate/disa ownership of the integrated TPM security device. Physical implied when accessing this field.

Activate = TPM security will be enabled and activated
Deactivate = TPM security will be deactivated and disabled
Clear = TPM security ownership data will be cleared

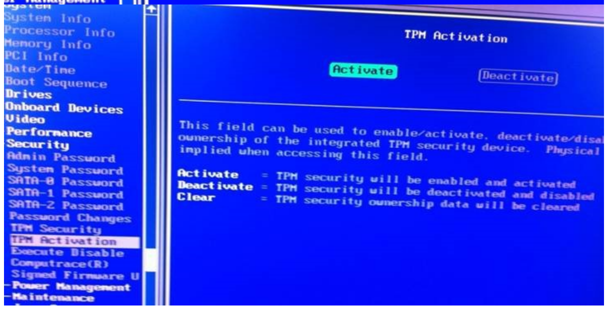


TPM Activation

Data encrypted with keys stored in the TPM will be lost. Are you sure you want to clear all keys from the TPM?



TPM Acknowledgement



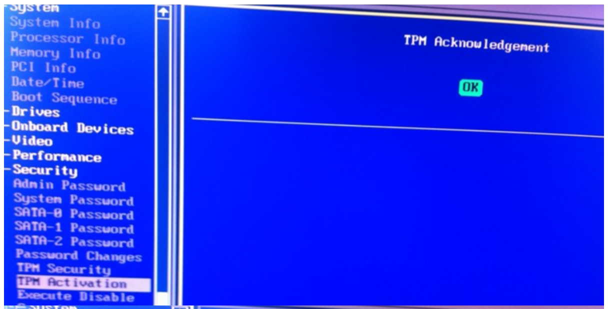
TPM Activation

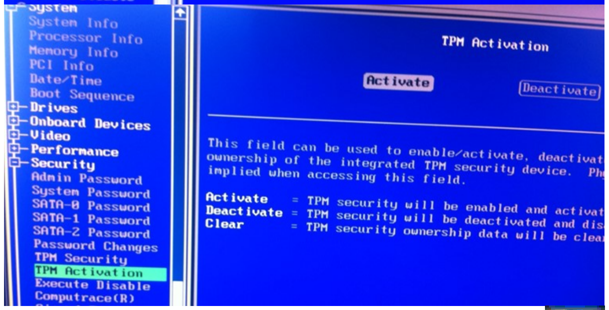
[✓ Activate] Deactivate

This field can be used to enable/activate, deactivate/disa ownership of the integrated TPM security device. Physical implied when accessing this field.

Activate = TPM security will be enabled and activated
Deactivate = TPM security will be deactivated and disabled
Clear = TPM security ownership data will be cleared

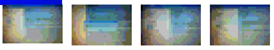
Make sure the TPM is still "Activated"

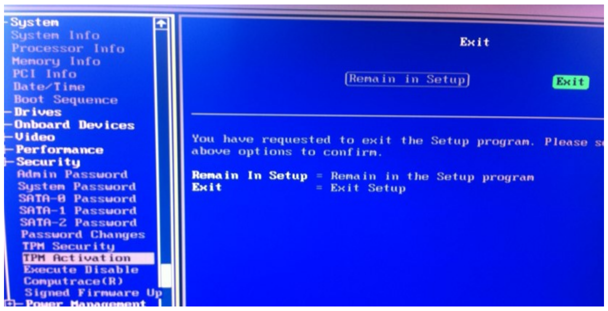




This field can be used to enable/activate, deactivat
ownership of the integrated TPM security device. Ph
implied when accessing this field.

Activate = TPM security will be enabled and activat
Deactivate = TPM security will be deactivated and dis
Clear = TPM security ownership data will be clear

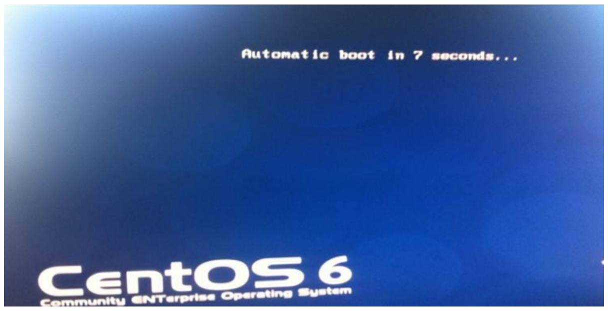
(screenshots for Dell Latitude E6400 )
Exit BIOS.



You have requested to exit the Setup program. Please s
above options to confirm.

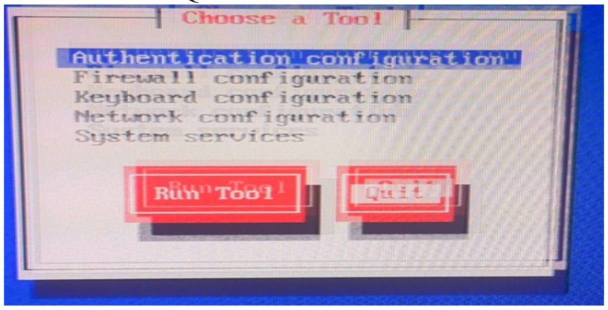
Remain In Setup = Remain in the Setup program
Exit = Exit Setup

- Boot from DVD.
  DVD boot screens:



Automatic boot in 7 seconds...

Just tab to "Quit" and ENTER



- Login as "root" password "dnssec"
- /etc/init.d/tcsd start (should return "OK")
- tpm_version (should return TPM info)
- optional: tpm_createek (this may not need to be done unless the TPM has never been initialized)
- tpm_takeownership (CR for all prompts. This will take a while generating 2048 bit SRK on the TPM. Try again or go through BIOS TPM CLEAR and ACTIVATE process again if an I/O error occurs.)
- tpm_restrictsrk -a (CR for password. For TPM error messages, see /var/log/messages)
- /etc/init.d/pkcsslotd start (should return "OK". You may have to "rm -rf /var/lib/opencryptoki/tpm" if this is not off the live dvd to clear out prior session setup. This is where encrypted keys will reside.)
- tpmtoken_init -l debug (Use 123456 for Security Officer and USER in this example. Note: do not use 87654321 for SO or 12345678 for USER. This will take a while while 2048 bit SO nd USER storage keys are generated)
- pkcsconf -t (to display results. Token #0 should show that it has been initialized,e.g., "Flags: 0x44D (RNG|LOGIN_REQUIRED|USER_PIN_INITIALIZED|CLOCK_ON_TOKEN|TOKEN_INITIALIZED)")
- export DOMAIN=yourdomain (no ending dot)
- optional: export TEST="yes" (short signature times for testing)
- signzone-tpm (PIN = 123456 from above. This will take a while. enerates KSK, ZSK, starts local nameserver as automated in-line signer)
- optional: monitor (a simple script using "dig" to periodically display RRSIG key tags)

See the contents of /opt/dccom/signzone-tpm and demo directory /tmp/namedb for details (e.g., /tmp/namedb/log/runlog and look for "C_Sign" mesages indicating PKCS11/TPM use). "named" with /tmp/namedb/named.conf will automatically maintain the signed zone using keys in /var/lib/opencryptoki/tpm/root protected by the TPM SRK.

Doing "dig +dnssec -t soa yourdomain @127.0.0.1" should show you the signed zone SOA as it automatically gets updated. The "Activate:" field in the keys/*.private files indicates when named will start using the corresponding key. The SOA serial should increment then. "rndc signing -list yourdomain" shows signing status. "rndc sign yourdomain" forces a recalculation of signatures. For a key rollover, you can manually add and remove keys from the "keys/" directory after a new key has been introduced and published and use the "Activate:" and other meta fields to effect a complete rollover. There should be plenty of BIND 9.9 documentation on how to do this.

Yes...since the demo DVD always starts fresh you need to CLEAR the TPM each time. Alternatively you may elect to install the LiveDVD onto a blank drive (or flash drive)* so that data structures created by "tpm_takeownership" are maintained across re-boots.

*Run "startx" from root prompt and use "install" icon from desktop. Before rebooting the new system, CLEAR and ACTIVATE TPM again and follow instructions from above:

- use BIOS to CLEAR and ACTIVATE TPM
- boot newly installed system.
- login as "root" password "dnssec"
- /etc/init.d/tcsd start
- tpm_version
- tpm_takeownership
- tpm_restrictsrk -a
- rm -rf /var/lib/opencryptoki/tpm
- /etc/init.d/pkcsslotd start
- tpmtoken_init -l debug
- export DOMAIN=yourdomain
- optional: export TEST="YES"
- signzone-tpm Study signzone-tpm and create own startup script, separating out key generation (to run separately), changing the /tmp directory to something more appropriate, and configuring a system to get unsigned zone updates. Your startup script might look like:
- `export PKCS11_LIBRARY_PATH="/usr/lib/opencryptoki/libopencryptoki.so"`
- `read -s -p "HSM PIN: " PKCS11_LIBRARY_PIN`
- `echo ""`
- `export PKCS11_LIBRARY_PIN`
- `/etc/init.d/tcsd start`
- `/etc/init.d/pkcsslotd start`
- `# keep /dev/random full`
- `nohup /opt/dccom/pkcs11-backup -S 0 -r -1:- >/dev/null 2>&1 &`
- `/opt/dccom/named -c /tmp/namedb/named.conf`
- done

UPDATE 19 August 2013

New work based on OpenSC 0.13.0 and Smartcard HSM by http://www.cardcontact.de/products/SmartCard-HSM_V1.0.pdf. These have many more HSM features such as the ability to securely export/import private key material between cards i.e., make backups. See this for more information. Updated DVD (1G ISO) file for complete bootable Smartcard HSM is here sha256=27cbaeb7f0aef5b7c82360ae8a410bb0d74af2231c0462d751ee11cf8f3daa79

```
NEW FILES

First card:
  hcarderase
  hmakeshares
  himportshare

Other cards:
  hcarderase
  himportshare

Gen KSK (on any card):
  hgenksk
  hwrapkey

Backup cards:
  hunwrapkey

Show contents:
  hcardshow

Delete item:
  hcarddel

Misc:
  hcardrng
  hgenzsk
  hcardsign
  hsignzone

Updated PKCS11 BIND modification (for dnssec-signzone):
  opensslrsa_link.c
```

Offline Smart Card HSM KSK + Online software ZSKs

- Create a temporary directory and make it the default: mkdir tmp; cd tmp
- hcarderase (Use 123456 for PIN and Security Officer PIN if asked)
- hmakeshares - to make two files with encrypted key shares (dkek-share-*.pbe) to be later used to make backup copies of keys. Please note passwords.
- himportshare dkek-share-1.pbe
- himportshare dkek-share-2.pbe
- export DOMAIN=yourdomain (not ending dot)
- optional: export TEST="yes" (short signature times for testing)

- In a second terminal window execute "hcardrng". PIN from above. If you want to now test the RNG, in another window do "cat /dev/random | rngtest", wait a minute, and then ctrl-C. rngtest should return some stats.)
- hgenzsk - to generate two software ZSKs using random numbers from card
- Stop hcardrng (ctrl-C) and exit out of that window
- hgenksk - to generate a KSK inside the card. Note the "label:" field.
- hcardshow - to see what is on the card. For these cards, only the private key is stored on the card. The public key is in a file suffixed with ".pub".
- optional: (See Below) make KSK backup cards
- hcardsign - will generate a bunch of pre-KSK-signed DNSKEY RRsets for future use. (Use "abc" for passphrase to encrypt keybundles, KSK CKA_LABEL and PIN from above)
- optional: hsignzone (Use "abc" for passphrase for keybundles. starts local nameserver and runs sample signer process to maintain signatures)

Making Smartcard HSM Backup cards:

With current card:

- hwrapkey - to export and encrypted (wrapped with shares) copy of the private key in a file (.wrap suffix)

Insert new card

- hcarderase
- himportshare dkek-share-1.pbe (sc-hsm-tool is the main OpenSC utility for this card. Without any arguments, it should display card status.)
- himportshare dkek-share-2.pbe
- hunwrapkey - responding with the file creates above (e.g. *.wrap)
- hcardshow - should show the key in the card
- Repeat above steps for additional cards

END

References