

HSM Code-Signing Journey

BIYONG SUN
7JUN 2024



PUBLIC



SECURE CONNECTIONS
FOR A SMARTER WORLD

Introduction



The Purpose

The HSM Coding-Signing is new. When we follow the instructions in Code-Signing Tool User's Guide , still has something to overcome, most of them are related to the OS.

Actually, Code-Signing Tool User's Guide can not give detail every "obvious" step.

The purpose of this document is to share the experiences on my system.

Hope those experiences can give you some clues on your system.



Environment and Setup



Environment

Debian 12.5.0(bookworm)

| | |
|--------------------------|--------------------|
| openssl | 3.0.11-1~deb12u2 |
| openc | 0.23.0-0.3+deb12u1 |
| softhsm | 2.6.1-2.1 |
| libengine-pkcs11-openssl | 0.4.12-0.1 |

Code-signing Tool 3.4.0

i.MX93 LF_v6.6.3_1.0.0 binary demo image

imx-boot-imx93evk-sd.bin-flash_singleboot

Ubuntu 22.04.3 LTS(jammy)

| | |
|--------------------------|-------------------------|
| openssl | 3.0.2-0ubuntu1.1 |
| openc | 0.22.0-1ubuntu2 |
| softhsm | 2.6.1-2ubuntu1 |
| libengine-pkcs11-openssl | 0.4.11-1ubuntu0.22.04.1 |

Setup

Following command to install the necessary software.

```
sudo apt-get install libengine-pkcs11-openssl openssl softsm
```

Creating a Token

Creating a Token

```
export SOFTHSM2_TOKEN_DIR=~/.softhsm2_token
rm -rf ${SOFTHSM2_TOKEN_DIR}
mkdir ${SOFTHSM2_TOKEN_DIR}
export SOFTHSM2_CONF=${SOFTHSM2_TOKEN_DIR}/softhsm2.conf
echo "directories.token_dir = ${SOFTHSM2_TOKEN_DIR}/" > ${SOFTHSM2_CONF}
cat ${SOFTHSM2_CONF}
export PKCS11_MODULE_PATH=/usr/lib/x86_64-linux-gnu/softhsm/libsofthsm2.so
pkcs11-tool -L --module ${PKCS11_MODULE_PATH}
pkcs11-tool --module ${PKCS11_MODULE_PATH} --init-token --init-pin --so-pin=654321 --new-pin=123456 --label="CST-HSM-DEMO" --pin=123456 --login
pkcs11-tool -L --module ${PKCS11_MODULE_PATH}
```

If successful, you will get the following message: Note: The so-pin and pin from below.

Available slots:

Slot 0 (0x1acffc18): SoftHSM slot ID 0x1acffc18

token label : CST-HSM-DEMO

token manufacturer : SoftHSM project

token model : SoftHSM v2

token flags : login required, rng, token initialized, PIN initialized, other flags=0x20

hardware version : 2.6

firmware version : 2.6

serial num : 5ca6095f1acffc18

pin min/max : 4/255

Slot 1 (0x1): SoftHSM slot ID 0x1

token state: uninitialized

You can change it and export at the very beginning.

```
# ./hsm_ahab_pki_tree.sh:96:
```

```
USR_PIN=123456
```

```
# ./Dockerfile.hsm:72:ENV SO_PIN 654321
```



Generate a PKI tree

Generate a PKI tree

cst-3.4.0/keys

```
pkcs11-tool --module ${PKCS11_MODULE_PATH} -l --pin 123456 --list-objects  
./hsm_ahab_pki_tree.sh -existing-ca n -use-ecc y -kl p384 -da sha384 -duration 10 -srk-ca n  
pkcs11-tool --module ${PKCS11_MODULE_PATH} -l --pin 123456 --list-objects
```

You are able to see the following information

Usage: encrypt, verify, wrap, derive

Access: local

Certificate Object; type = X.509 cert

label: ./SRK1_sha384_secp384r1_v3_usr

subject: DN: CN=SRK1_sha384_secp384r1_v3_usr

ID: 1001

Generating SRK Table and SRK Hash

Generating SRK Table and SRK Hash

cst-3.4.0/crts

```
../linux64/bin/srktool -a -s sha384 -t SRK1234table.bin -e SRK1234fuse.bin -f 1 -c  
SRK1_sha384_secp384r1_v3_usr crt.pem,SRK2_sha384_secp384r1_v3_usr crt.pem,SRK3_sha384_secp384r  
1_v3_usr crt.pem,SRK4_sha384_secp384r1_v3_usr crt.pem
```

csf_boot_image.txt



csf_boot_image.txt

```
[Header]
Target = AHAB
Version = 1.0

[Install SRK]
# SRK table generated by srktool
File = "SRK1234table.bin"
# Public key certificate in PEM format
#Source = "./release/crts/SRK1_sha384_secp384r1_v3_usr.crt.pem"
Source = "pkcs11:token=CST-HSM-DEMO;object=./SRK1_sha384_secp384r1_v3_usr;type=cert;pin-value=123456"
# Index of the public key certificate within the SRK table (0 .. 3)
Source index = 0
# Type of SRK set (NXP or OEM)
Source set = OEM
# bitmask of the revoked SRKs
Revocations = 0x0

[Authenticate Data]
# Binary to be signed generated by mkimage
File = "imx-boot-imx93evk-sd.bin-flash_singleboot"
# Offsets = Container header  Signature block (printed out by mkimage)
Offsets = 0x400      0x490
```

Please note the object=./SRK1_sha384_secp384r1_v3_usr should match the information by “pkcs11-tool --module \${PKCS11_MODULE_PATH} -l --pin 123456 --list-objects”

Build cst

Build cst

cst-3.4.0/code/build

```
OSTYPE=linux64 TOOLCHAIN=gcc make -f make/binaries.mk
```

If using the cst-3.4.0/linux64/bin/cst binary release, you may face the issue when you sign the image.

[ERROR] CST: **Error** loading pkcs11 engine: could not load the shared library.

Release_Notes.txt

2 KNOWN ISSUES

1. Version `GLIBC_2.xx' not found (required by cst): This package includes precompiled binaries for Linux, dynamically linked with the libc version on Ubuntu 20.04. Please note that these binaries may not function on newer Linux distributions with updated libc versions. To ensure compatibility, it is recommended to recompile the source code on your own system. Refer to the BUILD.md document for detailed build instructions.

Build cst(Cont.)

You can check the differences:

Build from source

ldd code/build/cst

```
linux-vdso.so.1 (0x00007fff1ebd0000)
libcrypto.so.3 => /lib/x86_64-linux-
gnu/libcrypto.so.3 (0x00007f5ebdc00000)
libc.so.6 => /lib/x86_64-linux-
gnu/libc.so.6 (0x00007f5ebda1f000)
/lib64/ld-linux-x86-64.so.2
(0x00007f5ebe139000)
```

Binary release

ldd linux64/bin/cst

```
linux-vdso.so.1 (0x00007ffc9d6d4000)
libdl.so.2 => /lib/x86_64-linux-
gnu/libdl.so.2 (0x00007f31804ad000)
libc.so.6 => /lib/x86_64-linux-
gnu/libc.so.6 (0x00007f317fc1f000)
/lib64/ld-linux-x86-64.so.2
(0x00007f31804c5000)
```

Sign using pkcs11 backend

Sign using pkcs11 back-end

cst-3.4.0

```
./code/build/cst -b pkcs11 -i csf_boot_image.txt -o flash.singed.bin  
CSF Processed successfully and signed image available in flash.singed.bin
```

Please note: the object=./SRK1_sha384_secp384r1_v3_usr should match the information by “pkcs11-tool --module \${PKCS11_MODULE_PATH} -I --pin 123456 --list-objects”

Reference Documents

Code-Signing Tool User's Guide Rev. 3.4.0 12/2023





SECURE CONNECTIONS
FOR A SMARTER WORLD