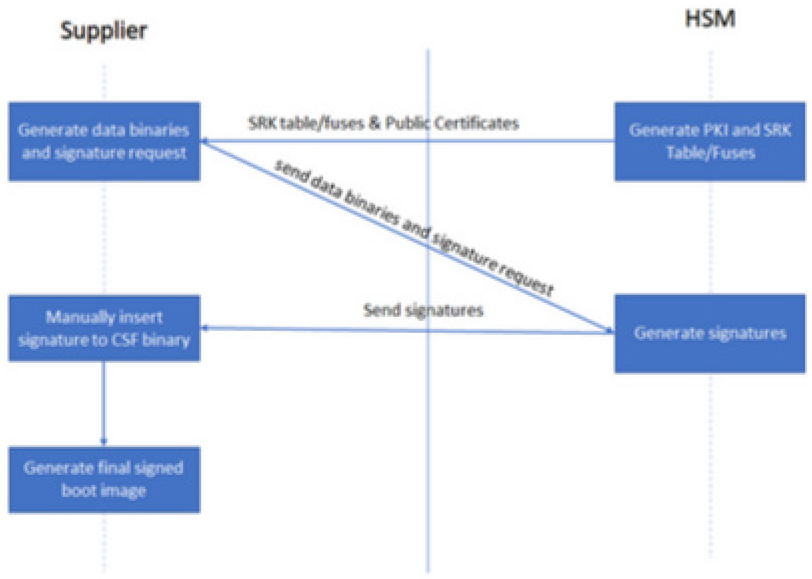
Using HSM mode for code signing in HAB devices

utkarsh_gupta:

Question

Customer want to use a HSM signing entity to sign boot artifacts. The supplier need to generate the csf command binary and image binary to be signed, and then let HSM generate the signature using their PKI. Finally the supplier inserts the signature to prepare final signed flash.bin.


Supplier
HSM
Generate data binaries and signature request
SRK table/fuses & Public Certificates
Generate PKI and SRK Table/Fuses
send data binaries and signature request
Manually insert signature to CSF binary
Send signatures
Generate signatures
Generate final signed boot image

Answer

The key to the approach is using the “Mode = HSM” in the CSF description file. This will instruct CST to prepare a “Signing request”, and output the data to be signed. The data can then be sent to OEM for signing by their HSM.

The approach can be separated to three steps:

Step 1: Prepare a 'Signing request' and output data to be signed.

1.The supplier only updates the CSF file for image signing(add “Mode = HSM” in [Header]), named as update1.csf, CSF can be:

```
[Header]
Version = 4.3
Hash Algorithm = sha256
Engine = ANY
Engine Configuration = 0
Certificate Format = X509
Signature Format = CMS
Mode = HSM

[Install SRK]
File = '../crts/SRK_1_2_3_4_table.bin'
Source index = 0

[Install NOCAK]
File = '../crts/SRK1_sha256_2048_65537_v3_usr_crt.pem'

[Authenticate CSF]

[Authenticate Data]
Verification index = 2
Blocks = 0x91ffc0 0x0 0x31e00 'flash.bin'
```

NOTE: Above example is for Fast authentication method. Same can be applied for regular authentication method.

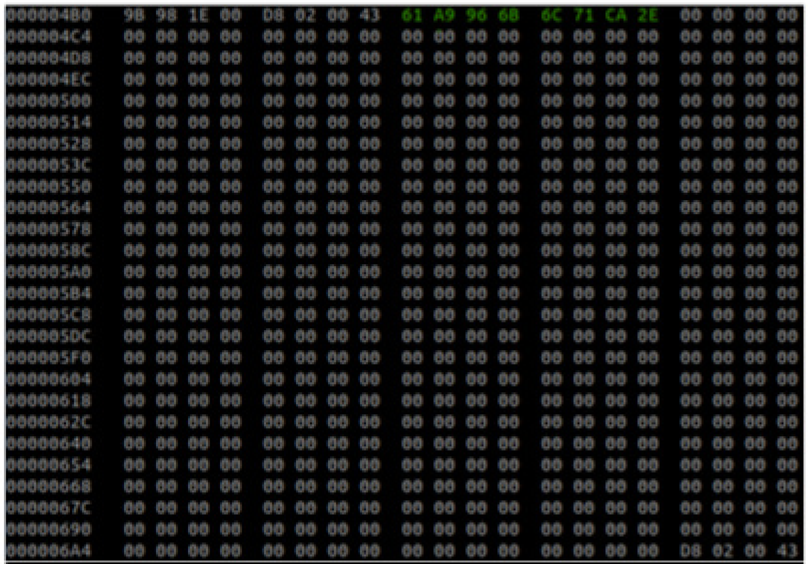
2.Use the command "../linux64/bin/cst -i updated1.csf -o csf.bin", which generates following files.

- **data_csfsig.bin** contains the CSF commands to be signed
- **data_imgsig.bin** contains the IMG data to be signe
- **csf.bin** is not the final signed flash.bin file, it misses the signature part in the CSF binary. It will be used after HSM sends the signature binary file.
- **sig_request.txt** contains a summary of the signature requests to be sent to the different HSM. The sig_request.txt can be:
  - contains the Public keys against which the private key needs to be used to sign the corresponding data_* binaries
  - The unique tag is used to match signature in the csf.bin file that needs to be updated.

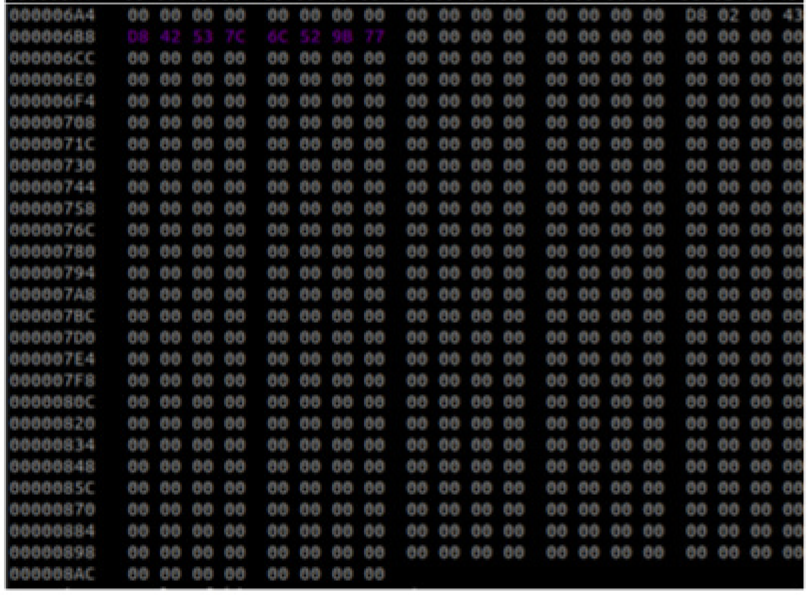
```
$ cat sig_request.txt
Signing Request:
../crts/SRK1_sha256_2048_65537_v3_usr_crt.pem
data_imgsig.bin
unique tag: 779b526c7c5342d8
Signing Request:
../crts/SRK1_sha256_2048_65537_v3_usr_crt.pem
data_csfsig.bin
unique tag: 2eca716c6b96a961
```

Note the unique tags placed in the CSF binary file below.

NOTE: The unique tags and the empty space needs to be replaced with the signature binary received from HSM.(colored green and purple in images below)



```
000004B0    9B 98 1E 00  D8 02 00 43  61 A9 96 6B  6C 71 CA 2E  00 00 00 00
000004C4    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000004D8    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000004EC    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000500    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000514    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000528    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000053C    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000550    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000564    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000578    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000058C    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000005A0    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000005B4    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000005C8    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000005DC    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000005F0    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000604    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000618    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000062C    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000640    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000654    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000668    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000067C    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000690    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000006A4    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  D8 02 00 43
```



```
000006A4    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  D8 02 00 43
000006B8    D8 42 53 7C  6C 52 9B 77  00 00 00 00  00 00 00 00  00 00 00 00
000006CC    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000006E0    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000006F4    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000708    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000071C    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000730    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000744    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000758    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000076C    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000780    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000794    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000007A8    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000007BC    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000007D0    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000007E4    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000007F8    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000080C    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000820    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000834    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000848    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
0000085C    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000870    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000884    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
00000898    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
000008AC    00 00 00 00  00 00 00 00
```

Step2: OEM generate the signature using their private key.

HSM will get the **data_csfsig.bin** and **data_imgsig.bin** and sign the two binary files with the private key to generate the signature file, which are named as **data_csfsig.sig** and **data_imgsig.sig**. Then HSM will send the signature file to the supplier, which is the signature part that should be inserted into the CSF binary. (as explained above)

Step 3: The supplier update the configuration file and generate final signed flash.bin

1. After receiving the signatures, insert them into the CSF binary file manually.

2. Generate the final signed flash.bin using the updated CSF binary depending on the SoC being used. (inserting CSF binary into flash.bin in 8M devices or appending CSF binary to the uboot image in 6/7 devices)

NOTE: The signature size is pre-calculated and determined using the knowledge of CMS signature format and the minimal data needed to construct the signature. If the signature received from HSM is bigger than the pre-calculated size, either of two things can be done.

1- Manually update the offsets of signatures in the CSF commands after the CSF binary is generated by CST before sending the data_* binaries to HSM for signing in Step 2.

(or) 2- Modify the following code in CST and re-compile CST to consider the new size of signature binary before running Step 1, the data_* binaries to HSM for signing.

```
>>> code/front_end/src/cst.c
/* Max size of buffer for signature bytes */
#define SIGNATURE_BUFFER_SIZE (1024)


>>> code/back_end-ssl/src/adapt_layer_openssl.c:
/*-------------------------
 calculate_sig_buf_size
--------------------------- */
#define SIG_BUF_FIXED_DATA_SIZE 217
```