USB KEY BUILDING FOR CHIPSEC AND SECUREBOOT CHECKS v0.1 (04/2019)

`A Help to build your own ChipSec and SecureBoot USB keys`

Final USB keys

- **USB KEY 1** : live Debian distribution to launch **ChipSec** from the computer to analyze
- **USB KEY 2** : contains **SecureBoot** keys to import, tool to import your own trust keys and to check importation

Linux Tools to install before generating the USB keys

```
sudo apt-get install debootstrap
sudo apt-get install sbsigntool
sudo apt-get install efitools
```

Tool to build the usb keys : create-keys.sh

Note: Some sub scripts require access to sudo commands.

Build USB KEY 1

Plug a new usb key (attached on /dev/sdc in this case).

`./create-keys.sh live /dev/sdc`

Unplug the usb key.

Build USB KEY 2

Plug a new usb key (attached on /dev/sdc in this case).

`./create-keys.sh shell /dev/sdc`

Unplug the usb key.

Boot on keys

Plug one of keys, start the computer.

USB KEY 1

1. boot on the USB key, then at the bootloader prompt start linux live
2. when finished booting, login as root (no password)
3. from the root terminal, launch ChipSec with “chipsec_main.py”.

USB KEY 2 :

1. Go the BIOS/Firmware configuration and set the platform to SecureBoot enabled and reset to Setup Mode.
2. either:
   - boot on USB key and launch EFI binaries from EFI shell (Shell.efi is automaticaly started).
   - OR interrupt the normal boot to select a shell EFI from boot configuration and launch EFI binaries from EFI shell.
3. execute binaries from EFI shell :
   - identify the USB key letter storing the binaries with commmands "fs0" or "fs1" or fsX ... then "dir"
   - launch "KeyTool.efi" to import trust keys.
4. replace (in that order) db, KEK and PK using files from keytool folder
5. importing the PK will set the platform to User mode
6. restart the platform:
   - the shell should run since it's signed with trust anchor to the PK
   - HelloWorld.efi should not run since it's unsigned