Encryption disk

Full disk and hidden OS in EFI

v1.2

Revisions

N	Date	Name	Comment
1.2	28-mar-17	kavsrf	Hidden OS. Simplified
1.1	28-feb-17	kavsrf	Hidden OS. Developer way.
1.0	28-jan-17	kavsrf	Started.

1 Full disk encryption

1.1 Boot from local hard disk

1.1.1 Final disk structure

GPT	S62	Part1	Part2	…	PartN	VeraCrypt loader part	GPT
Open		Encrypted				Open	

1.1.2 Installation scenario (It is proposal).

Create separate volume for VeraCrypt boot loader only. MS Windows loader and others starts from encrypted volume.

Create and save S62 with master keys. (test phase)

Boot from rescue USB
Request authorization
Install DCS loader to VeraCrypt part (recovery volume)
Modify S62 with entire disk range. (Check encryption range! Disk has to be decrypted! EncryptionLength=0)
DcsBoot is installed to VeraCrypt part (in boot menu and binaries copy)
DcsBoot will start bootmgfw.efi from ESP instead of VeraCrypt part. (ESP is encrypted)
Restart, authorize and encryption of range selected in OS booted

1.1.3 Developer way. To test PoC.

After boot from rescue USB use rescue disk with shell.efi.
Copy EFI\VeraCrypt from rescue USB to VeraCrypt part
Install boot menu to execute DcsBoot from VeraCrypt part
Start Shell.efi
1. List drives
> EFI\VeraCrypt\DcsCfg.dcs –dl d
2. List partitions to select correct range
> EFI\VeraCrypt\DcsCfg.dcs –ds <driveN> -pl
Modify sector 62 with range selected
> EFI\VeraCrypt\DcsCfg.dcs –vcp <drive> -rnd 2 -aa

2 Hidden OS installation

2.1 Installation in addition to already encrypted OS

2.1.1 Prepare disk state

GPT	S62	ESP	OS	…	Outer start	H_ESP	H_OS	Outer end	…	GPT
Open			Encrypted	Open						

2.1.2 Final disk state

H_OS [gpt_hos]	GPT	S62	ESP	OS	…	Outer start	H_ESP	H_OS	Outer end	…	GPT
OS [gpt_os]						Outer					
comment	Open			Enc.		Enc.	Encrypted		Enc.	`	

boot GPT for OS and boot GPT for H_OS

2.1.3 Installation scenario

Boot from rescue USB for OS

1. Protect ESP (optional step)

Convert ESP type to recovery volume to protect installation of ordinary OS (if it is installed)

2. Create partitions for hidden OS

Create 4 sequential partitions
Via diskpart or any other tool
1. Outer start – partition for fake data (outer volume will be mounted as ordinary volume)
2. ESP - EFI system partition of hidden OS (for MS boot loader)
3. H_OS - partition for hidden OS (for OS)
4. Outer end – partition for fake data (outer volume will be mounted as hidden volume)

3. Install H_OS and VeraCrypt in H_OS
4. Start System encryption
5. Modify encryption range to include outer volumes

Boot from rescue USB with EFI shell
Execute
> EFI\VeraCrypt\DcsCfg.dcs -oshideprep
It wipes Outer volumes, modifies sectors range to be encrypted, create GPT hidden and GPT OS.

6. Create authorization USB for OS and H_OS

> EFI\VeraCrypt\DcsCfg.dcs –srw <N> –ds <usbN>
> EFI\VeraCrypt\DcsCfg.dcs –srm <N> –ds <usbN>
> EFI\VeraCrypt\DcsCfg.dcs -pf gpt_hos –aa -pe
> EFI\VeraCrypt\DcsCfg.dcs -pf gpt_hos –sra 0 –ds <usbN>
(optional add keys of ordinary OS)
> EFI\VeraCrypt\DcsCfg.dcs -pf key_of_os –sra 1 –ds <usbN>
Edit DcsProp config keys
SecRegionSearch – is “1” to search USB marked with keys
DcsBootForce – is 0 to avoid asking password if the USB marked is not connected.

7. Boot H_OS and encrypt
8. Boot from rescue USB for H_OS

Update GPT to hide H_OS and H_ESP
> EFI\VeraCrypt\DcsCfg -pf gpt_os -pe <ESP_N> -ps
> EFI\VeraCrypt\DcsCfg -ds <driveN> -pf gpt_os -pa

9. Restore ESP from recovery type

2.1.4 Developer way

Create disk GPT with all OSs (install Linux or others if needed) and partitions: Outer, H_ESP, H_OS
Format H_ESP (FAT32), H_OS(NTFS)

Boot rescue USB for OS. Shell.
Save partition info of OS to “parts_os”
> EFI\VeraCrypt\DcsCfg –dl d
> EFI\VeraCrypt\DcsCfg –ds <driveN> -pf parts_os -ps
Copy part_os part_hos_prep

Edit ESP to hide (part type wre)
> EFI\VeraCrypt\DcsCfg -pf parts_os_prep –pl
> EFI\VeraCrypt\DcsCfg -pf parts_os_prep –pe <espN> -ps

Edit H_ESP (part type efi)
> EFI\VeraCrypt\DcsCfg -pf parts_os_prep –pe <h_espN> -ps
Apply new GPT to disk
> EFI\VeraCrypt\DcsCfg –ds <driveN> -pf parts_os_prep -pa

Install Windows on to H_ESP and H_OS
Start VeraCrypt encryption

Reboot to rescue USB
List partitions to select correct range
> EFI\VeraCrypt\DcsCfg.dcs –ds <driveN> -pl
Modify sector 62 with range of Outer, H_ESP, H_OS
> EFI\VeraCrypt\DcsCfg.dcs –vcp <drive> -rnd 2 –aa

Boot Windows and encrypt
Note: It is possible to check H_OS encrypted boot after encryption. (start VeraCrypt from ESP(rename bootmgfw.efi before)

Modify sector 62 with range of H_ESP, H_OS (exclude Outer)
> EFI\VeraCrypt\DcsCfg.dcs –vcp <drive> -rnd 2 –aa

Save partition info of H_OS to “parts_h_os”
> EFI\VeraCrypt\DcsCfg –dl d
> EFI\VeraCrypt\DcsCfg –ds <driveN> -pf parts_h_os -ps

Copy part_os part_os_final

Create Outer partition with H_ESP, H_OS inside
> EFI\VeraCrypt\DcsCfg -pf parts_os_final –phide <start,end> -pnt <outerN> -ps
Note: To hide from disk manager select partition type msr for Outer
> EFI\VeraCrypt\DcsCfg -pf parts_os_final -pe <outerN> -ps

Apply GPT to disk
> EFI\VeraCrypt\DcsCfg -pf parts_os_final –ds <driven> -pa

Create hidden volume header for Outer volume (it is possible from OS also)
> EFI\VeraCrypt\DcsCfg –dl
> EFI\VeraCrypt\DcsCfg -ps <OuterN> -ach -aa
Note: Two sectors with headers. Header sector 0 for full Outer volume, sector 128 for protective

Create key USB
Select USB disk
> EFI\VeraCrypt\DcsCfg –dl d
Wipe security region data. srN – number of headers. Size of the region 128K*srN. It starts from 61. 61 sector contains mark. The mark is depends of computer and USB serial number.
Note/todo: it writes random. It is better to create several fake headers.
> EFI\VeraCrypt\DcsCfg -ds <usbN> -srw <srN>
> EFI\VeraCrypt\DcsCfg -ds <usbN> -srm <srN>

Prepare security region data for OS
Copy parts_os_final to sr_os
> EFI\VeraCrypt\DcsCfg -pf sr_os –pz
Encrypt security region
> EFI\VeraCrypt\DcsCfg -pf sr_os –pe –ps –aa

Prepare security region data for H_OS
Copy pf parts_h_os to sr_h_os
Edit module path (bootmgfw) to be executed after authorization success.
> EFI\VeraCrypt\DcsCfg -pf sr_h_os –pexec –ps
Edit password cache (to mount system favorites)
> EFI\VeraCrypt\DcsCfg -pf sr_h_os –pwdcache –ps
Edit rnd parameters (or via DcsProp random key)
> EFI\VeraCrypt\DcsCfg -pf sr_h_os –rnd <type, init data> –prndsave –ps
Encrypt security region
> EFI\VeraCrypt\DcsCfg -pf sr_h_os –pe –ps -aa

Save security regions to USB
> EFI\VeraCrypt\DcsCfg -pf sr_h_os –sra 0 –ds <usbN>
> EFI\VeraCrypt\DcsCfg -pf sr_os –sra 1 –ds <usbN>
Edit DcsProp config keys
SecRegionSearch – is “1” to search USB marked with keys
DcsBootForce – is 0 to avoid asking password if the USB marked is not connected.

Note:
Random – random type 2 – RDRAND, 5 – TPM12, 3 – HMAC SHA512, 4 - OpenSSL.
DcsDriver – is 0/1 (install DcsBoot as UEFI driver)

3 Exit actions

DcsProp config keys to control authorization process

ActionNotFound – string to execute if USB marked is not found
ActionFailed– string to execute if authorization is failed
ActionSuccess– string to execute if authorization is OK

3.1 Action string rules

Exit – simple exit (default)
Status(code) – select exit status code (0 is OK)
File(path) – path to file to be executed
Guid(xxx-x..) – GUID of partition with file to be executed
Printinfo – print guid, file and status.
Message(msg) – message to display for the action
Postexec – send loader path to DcsBoot to execute after exit
Exec –execute module
Halt – CPU halt
Delay(N) – delay boot

E.g.
<config key="ActionNotFound"> exit status(1)</config>

4 Platform and/or TPM locked

Select random (via DcsProp Random key or via security region)
In loader enter password and press F2 to change
Enter new password
F7 to lock platform, F8 to lock TPM
Confirm new password
F7 to lock platform, F8 to lock TPM
TO DO: F9 is reserved to lock password with smart card.