UEFI

Unified Extensible Firmware Interface

Ver.2 del 26/10/2014

Avvertenza

Non sono un esperto di UEFI, ho installato alcuni sistemi con UEFI con Secure Boot e mi sono chiesto che cosa sto facendo?

Questo è il risultato della “ricerca”.

Bibliografia

- http://en.wikipedia.org/wiki/BIOS
- http://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface
- http://www.rodsbooks.com/linux-uefi (consigliato)
- https://www.happyassassin.net/2014/01/25/uefi-boot-how-does-that-actually-work-then/ (consigliato)
- http://www.uefi.org ed in particolare

  UEFI Specification v2.4 (2296 pagine)

Bibliografia Secure Boot

- https://www.suse.com/communities/conversations/uefi-secure-boot-overview/
- https://www.suse.com/communities/conversations/uefi-secure-boot-plan/
- https://www.suse.com/communities/conversations/uefi-secure-boot-details/
- http://www.rodsbooks.com/efi-bootloaders/secureboot.html

Cronologia

- 1975: Basic I/O System in CP/M OS
- 1998: Intel Boot Initiative a causa dei sistemi Itanium => poi rinominato EFI
- 2005: Intel interrompe EFI e contribuisce alla nascita di Unified EFI Forum
- 2011: Consumer motherboard con UEFI
- 2012: Micro$oft Windows 8 richiede che I computer certificati W8 abbiano secure boot
- 2012: shim

Come si “accende” un computer?

- Se non ha la batteria carica si collega alla presa elettrica
- Si schiaccia il pulsante di accensione

….. Un po' più in dettaglio...

BIOS

- Il processore riceve la corrente con certi altri segnali (CPU reset)
- Comincia ad eseguire del codice macchina ad una certa locazione fissa e prevedibile
- Nei processori x86 in real mode (ossia con gli indirizzi a 20bit) e quindi nei primi (2^10)^2=1MiB, in particolare al fondo dove si trova quindi un jump al BIOS vero e proprio

BIOS

Il BIOS esegue i seguenti passi

- POST (Power On Self Test)

  controllo ed inizializzazione CPU,RAM, HD, controllori DMA, video, interrputs ecc ecc

- Ricerca moduli ROM (firmware)

  anche questi sono nel primo MiB e testano ed inizializzano l' hw che controllano (esempio scheda video, controlles SCSI)

- Int 19h (boot)

  SE il modulo ROM ritorna ALLORA

  ricerca il boot loader sui 'boot device' in ordine ed inizia il bootstrapping (=tirarsi su dai lacci) ossia booting

BIOS e MBR

- Int 19h

  la chiamata all'interrput 19 ricerca il boot loader nei “boot device' che sono dischi o chiavette ecc che vengono individuati dalle informazioni di configurazione in CMOS, EEPROM.

  Per ogni device prova a capire se il primo settore (boot sector) è un loader caricandolo in memoria.

  Se non lo legge assume che non ci sia nulla, se lo legge assume che sia il loader e gli traferisce il controllo.

- MBR

  Il Master Boot Record è il boot loader più le informazioni sul partizionamento del disco

MBR

- Introdotto nel 1983

  per HD da 10MB (non è un errore tipografico)
- Piccolo (512b= 1 blocco) quindi il loader deve trasferire il controllo al second stage
- La partition table del disco lo limita a 2TB (2^32*512b) e ha la vecchia struttuta di al massimo 4 partizioni (quindi l'invenzione della partizione logica)

OSSERVAZIONI

- Limiti sulla gestione del HD
- Problemi di sicurezza => per esempio nel 1999 virus CIH o Chernobyl virus che cancellava la flash con ROM BIOS
- Setup utility per scegliere le impostazioni

  ma anche overclocking
- Riprogrammabile (i più nuovi)
- Fornisce dei servizi (non più usati perché troppo primitivi)

UEFI è differente!

- Differentemente da BIOS che è legato al real mode di 20bits degli x86 UEFI è agnostico

  “UEFI also overcame the hardware scaling limitations that the IBM PC design assumed, allowing its broad deployment across high-end enterprise servers modern PC’s, and embedded devices. UEFI is “processor architecture-agnostic,” supporting x86, x64, ARM and Intel® Itanium.”

  da The Uefi Primer

UEFI e driver

- Il processore non usa il real mode (nel caso degli x86)
- C'è l'*UEFI Driver Model* che descrive come scrivere i driver per poter accedere ai boot devices nel preboot
- UEFI deve esser compatibile con le vecchie Option ROM

UEFI e boot manager (1)

- Le specifiche definisco un UEFI boot manager che carica driver e programmi come

  “The UEFI boot manager is a firmware policy engine that can be configured by modifying architecturally defined global NVRAM variables. The boot manager will attempt to load UEFI drivers and UEFI applications (including UEFI OS boot loaders) in an order defined by the global NVRAM variables.”

- Può esser gestito da linux con *efibootmgr*

UEFI e boot manager (2)

La sequenza di boot del boot manager

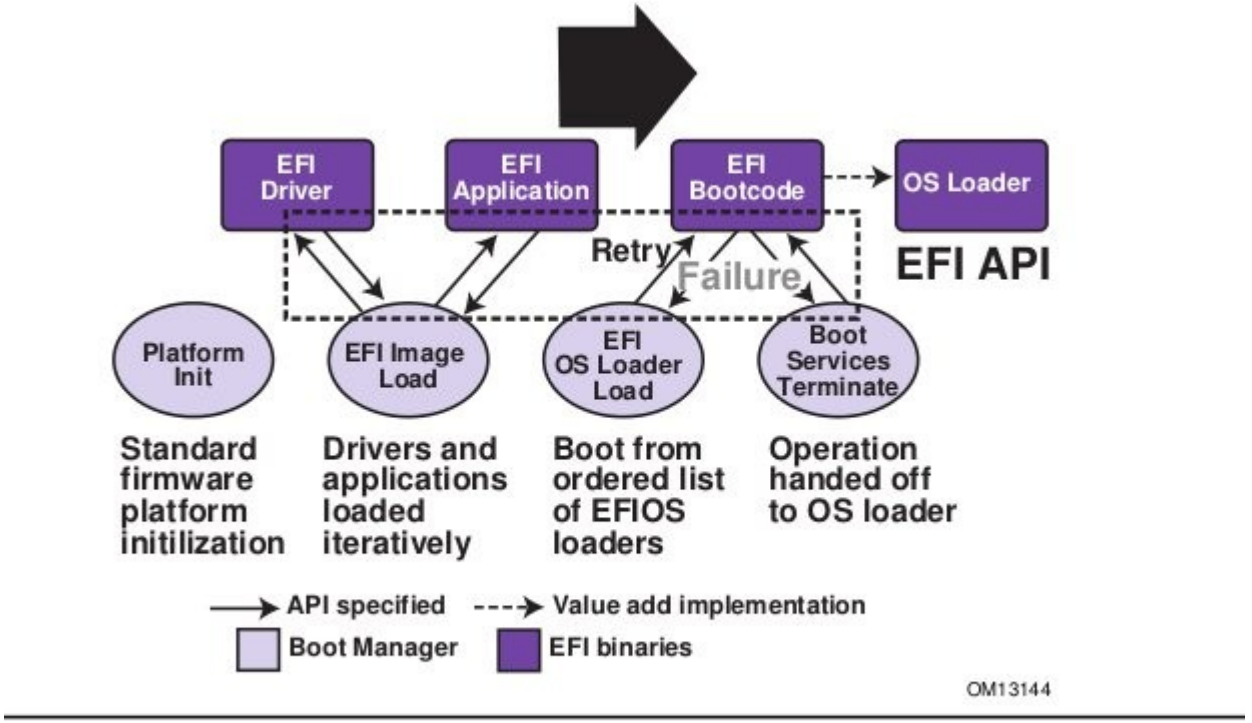

Figure 2. Booting Sequence

UEFI ha una “virtual machine”

- Per poter scrivere e far funzionare i device driver in maniera indipendente dalla piattaforma c'è

  EFI Byte Code Virtual Machine

- Anche i boot loaders sono byte compiled!
- I driver per HD non possono ovviamente risiedere sull HD stesso ma devon esser in qualche rom accessibile a UEFI

UEFI e GPT

- UEFI può accedere direttamente al disco da cui legge i programmi
- Il disco partizionato può esser descritto con la *GUID Partition Table* GPT che permette dischi grossi e praticamente tante partizioni quante se ne vogliono

  IMPLICAZIONE
- Si devono usare gparted, gdisk ma non fdisk
- Può succeder di dover partizionare il HD a mano

UEFI e ESP (1)

- UEFI è in pratica più di un “MS-DOS” come capacità
- Può far partire programmi grossi (loader, setup utilities...)
- Fornisce Services (tipo ora ecc)

  QUINDI
- Ha *almeno* una sua partizione di nome ESP (EFI System Partition) in FAT16 o FAT32 dedicata su cui risiedono I programmi fra cui i boot loaders

UEFI e ESP (2)

- La sua partizione di nome ESP (EFI System Partition) è obbligatoria per qualsiasi boot device

  IMPLICAZIONE

- Le chiavette usb per l'installazione di linux non devo esser più riformattate ma solo partizionate

UEFI e BIOS

- Ha un modo di compatibilità con il BIOS che permette di usare il MBR però questo non significa disabilitarlo (non è possibile)

  IMPLICAZIONE
- In particolare l'interfaccia di setup è quella di UEFI e non del BIOS

UEFI e efibootmgr

- Efibootmgr pemette di gestire da linux i parametri di boot del boot manager

```
[root@system directory]#

[root@system directory]# efibootmgr -v
BootCurrent: 0002
Timeout: 3 seconds
BootOrder: 0003,0002,0000,0004
Boot0000* CD/DVD Drive  BIOS(3,0,00)
Boot0001* Hard Drive    HD(2,0,00)
Boot0002* Fedora        HD(1,800,61800,6d98f360-cb3e-4727-8fed-5ce0c040365d)File(\EFI\fedora\grubx64.efi)
Boot0003* opensuse      HD(1,800,61800,6d98f360-cb3e-4727-8fed-5ce0c040365d)File(\EFI\opensuse\grubx64.efi)
Boot0004* Hard Drive    BIOS(2,0,00)P0: ST1500DM003-9YN16G       .
[root@system directory]#

[root@system directory]# efibootmgr -v
```

- Il loader di default è \EFI\BOOT\Bootx64.EFI
- Qui si vedono i loader di fedora e opensuse

UEFI e Secure Boot

- Quando Secure Boot è attivato
- UEFI controlla che i programmi siano firmati prima di eseguirli

  PROBLEMA
- Windows 8+ richiede Secure Boot

  => firma Micro$oft in tutte the schede madri

  => GNU/linux firmato da Micro$oft

Secure Boot e shim

SOLUZIONE al problema precedente

- Shim che è firmato da M$ ed è un loader di loader a cui si possono aggiungere facilmente delle chiavi
- Però il kernel deve esser firmato

- In realtà è possibile per ogni utente aggiungere la propria chiave a UEFI e firmare il proprio SW!

  Questo dicenta necessario se si ricompila il kernel.

Ulteriori note

- UEFI ha una shell come grub
- I programmi di UEFI si scrivono in C con un SDK della intel

UEFI in pratica (1)

Aggiornato con l'esperienza del LIP e basato su Ubuntu 14.04. Per le altre distro dovrebbe funzionare tutto mutatis mutande

- Ubuntu 14.04 è perfettamente compatibile con UEFI e Secure Boot => non necessita disabilitarli dal “BIOS”
- Se Windows è in modalità UEFI anche Linux deve esser in modalità UEFI, viceversa se è in modalità BIOS anche linux deve esserlo

UEFI in pratica (2)

- Far partire linux parte in modalità prova (ricordarsi di nomodeset se problemi col video nero)
- Controllare che Windows sia installato in modalita' UEFI (che dovrebbe sempre esser per W8+), se sì si deve usare UEFI

  Come? Se *ls /sys/firmware/efi* (o *efibootmgr -v* purtroppo questo comando non c'è nell'immagine di boot) fa vedere qlc si è sicuramente UEFI, altrimenti bisognerebbe usare gparted o gdisk per vedere se c'è la partizione ESP (che può anche chiamarsi in maniera diversa...)

UEFI in pratica (2.1)

- Vediamo in dettaglio. Aprire un terminale, sudo -i per diventare root, *gdisk -l /dev/sda* mostra se sul disco c'è MBR e/o GPT, se c'è GPT allora probabilmente è UEFI. Esempio di puro MBR

```
root@MobilIgor5a:~# gdisk -l /dev/sda
GPT fdisk (gdisk) version 0.8.8

Partition table scan:
  MBR: MBR only
  BSD: not present
  APM: not present
  GPT: not present


****************************************************************************
Found invalid GPT and valid MBR; converting MBR to GPT format
in memory.
****************************************************************************

Disk /dev/sda: 625142448 sectors, 298.1 GiB
Logical sector size: 512 bytes
Disk identifier (GUID): 6C624660-AFA8-45CC-AC2D-B2B1FB73827B
```

UEFI in pratica (2.2)

- Per assicurarsi che una partizione, diciamo per esempio /dev/sda1 sia la partizione ESP entriamo in gdisk con *gdisk /dev/sda,* al prompt digitiamo *i* e poi alla domanda Partition number, gli diamo 1.

  Se il GUID è C12A7328-F81F-11D2-BA4B-00A0C93EC93 allora quella è la partizione ESP

```
Command (? for help): i
Partition number (1-6): 1
Partition GUID code: EBD0A0A2-B9E5-4433-87C0-68B6B72699C7 (Microsoft basic data)
Partition unique GUID: B00897C5-5293-445C-84AF-075E56A94030
First sector: 2048 (at 1024.0 KiB)
Last sector: 409599 (at 200.0 MiB)
Partition size: 407552 sectors (199.0 MiB)
Attribute flags: 0000000000000000
Partition name: 'Microsoft basic data'
```

UEFI in pratica (3)

- disabilitare fastboot di W8 (che è una specie di ibernazione) http://www.eightforums.com/tutorials/6320-fast-startup-turn-off-windows-8-a.html

UEFI in pratica (4)

- Se volete sentirvi sicuri, fare bck della partizione UEFI con un semplice cp -R

  Come?

  Vedere se è montata /boot/efi, se no da terminale con gparted o gdisk vedere quale partizione è flaggata boot (per compatibilità col BIOS) o seguite quanto scritto prima per capire se il sistema è UEFI.

  Quindi montarla e vedere se c'è dentro la dir EFI, fare una copia su una chiavetta

  Poi smontarla altrimenti interferisce coll'installazione

UEFI in pratica (5)

- Se volete sentirvi sicuri, fate anche il backup della partition table con gdisk, ossia come root *gdisk /dev/sda,* al prompt usare il comando *b*

UEFI in pratica (6)

- Usando gparted shrinkare la partizione Windows
- Usando gparted fare almeno due partizioni una swap ed una ext4 (partizione che monterete come /)

  Queste due sono le partizioni minime per il funzionamento.

UEFI in pratica (7)

- Avviavare l'installazione linux, quando chiede dove installarlo dire che si vuole fare a mano e scegliere la extr4 come / e dire di installare grub nella partizione ESP dove è già installato il bootloader di Windows.

  Ossia grub non deve esser installato nel disco generico /dev/sda ma in una partizione come /dev/sda1 dove c'è il bootlaoder di Windows, ossia nella partizione ESP (Non abbiate paura non sovrascrivete il loader di Windows)

UEFI in pratica (8)

- Finito fatte il reboot
- Se linux non parte e parte direttamente W => controllare l'ordine di boot.

  Come?

  O dal setup del “BIOS” (se si può)

  O se non si può, entrare in Windows, scaricare EasyUEFI ed aggiornare l'ordine (può capitare che faccia vedere due loader di nome ubuntu e Ubuntu, quello che dovete metter per primo è quello associato al loader di nome shimx64.efi)