UEFI & EFI

- Objet : Boot d'une distribution non compatible avec EFI & UEFI.
- Niveau requis :
  débutant, avisé
- Commentaires : *Ce tuto concerne les personnes disposant d'une carte mère intégrant des firmwares EFI & UEFI.*
- Débutant, à savoir : Utiliser GNU/Linux en ligne de commande, tout commence là ! 😀
- Suivi :
  à-tester
  - Création par mytux le 03/09/2013
  - Testé par ... le ...
- Commentaires sur le forum : c'est ici [1)]

Introduction

Toutes les versions 64bits des PC qui exécutent Windows certifié par le programme de certification Windows utilisent l'UEFI à la place du BIOS.

Le BIOS

A l'aube de l'ère du premier PC en 1981, IBM livrait son **P**ersonal **C**omputer 5150, avec un firmware connu sous le nom de BIOS (**B**asic **I**nput/**O**utput **S**ystem).

Le BIOS est l'interface logicielle entre le matériel et le logiciel à un niveau très basique. Il fournit le niveau d'interface le plus bas entre les pilotes de périphériques et le matériel.

Les routines du BIOS sont inscrites dans une mémoire de type EEPROM (Electric Erasable Programmable Read-Only Memory) implantée sur la carte mère. Ce type de mémoire est donc re-programmable ce qui permet de ré-inscrire son contenu pour effectuer une mise à jour de ces routines ("Flashage" du BIOS).

Une zone mémoire de type CMOS associée à un circuit d'horloge (RTC), est alimentée par une batterie de manière à pouvoir garder, quand le système est hors tension, les informations sur la configuration matérielle de la machine ainsi que l'heure et la date (RTC) qui sera utilisée par le système. Ces informations seront utilisées par le POST pour l'initialisation du matériel et sont aussi accessibles et modifiables par une interface dont le programme (BIOS Setup) est inscrit dans la mémoire BIOS.

Quand l' ordinateur est mis sous tension, ou lors d'un *reset*, le signal (Power Good) est transmis au microprocesseur qui va alors lire et exécuter les instructions inscrites dans la mémoire BIOS.

Le programme du BIOS effectue alors une séquence d'auto-test appelée le POST (**P**ower **O**n **S**elf **T**est) puis recherche, parmi les périphériques de mémoire de masse qui lui sont accessibles, ceux dont le premier secteur est un MBR.

Si, dans ce MBR, il ne trouve pas de routine d'amorçage, le programme du BIOS va alors lire le premier secteur de chaque partition pour y chercher un PBR (ou VBR).

MBR (Master Boot Record) ou zone d'amorçage. D' une taille de 512 octets il contient dans ces 446 premiers octets une routine (un programme) d' amorçage destiné soit à démarrer le système d' exploitation sur la partition active, soit un chargeur de démarrage (bootloader). Les 6 derniers octets de ce bloc peuvent contenir une signature numérique optionnelle sur 4 octets et 2 octets nuls. Les 64 octets suivant contiennent la table des quatre partitions primaires. Le tout finit par une signature 0xAA55 sur 2 octets.

PBR (Partiton Boot Record) aussi appellé VBR (Volume Boot Record) ou Partition Boot Sector, est le premier secteur de chaque partition primaire ou logique. Il peut contenir une routine de démarrage d'un système d'exploitation, un chargeur de démarrage, voire rien du tout si la partition n'a pas vocation à être bootée. Quand le BIOS ne contient pas de routine de boot, le BIOS tente de démarrer et d'exécuter la routine de boot inscrite dans le PBR de la partition marquée active.

EFI et UEFI

J'utilise le terme EFI pour se référer à EFI 1.X et l'UEFI.

L'EFI (Extensible Firmware Interface) et sa variante plus récente UEFI (Unified EXtensible Firmware Interface), sont des conceptions de firmwares destinés à remplacer le BIOS.

EFI, a été développé par Intel pour son nouveau processeur Intel-HP Itanium.

Les capacités de l'EFI sont bien plus grandes que celles du BIOS.

En effet un EFI peut lire une table de partitions et accéder aux systèmes de fichiers, il ne prend pas en charge le multitâche mais peut exécuter des applications en C/C++, pour les versions les plus récentes tel que l' UEFI, des applications graphiques avec prise en charge de la souris.

A l'instar du bios qui utilisait les fichiers cachées *IBMIO.SYS, IBMDOS.SYS (ou IO.SYS, MSDOS.SYS)*. L'exécution du programme au standard UEFI lit le système de fichiers à la recherche d'exécutable UEFI.

Il est aussi possible de compiler un noyau Linux de sorte à se passer d'un chargeur de démarrage. On pourra ensuite créer une entrée dans la NVRAM de notre carte mère à l'aide de **efibootmgr**, du **Shell EFI**, ou d'utiliser un boot manager comme **rEFIND**.

Voir EFI stub loader sur ArchWiki

Une autre caractéristique de l'EFI est le Secure Boot. Cette fonctionnalité est destinée à améliorer la sécurité, en veillant à ce que seuls les chargeurs de démarrage signés avec une clé de chiffrement puissent s'exécuter.

Structure de disque EFI

Selon la version et le constructeur de votre fimware, ces règles peuvent varier.
En voici un aperçu :

- Tous les périphériques de stockage EFI recherchent un répertoire EFI et listent un point amorce pour chaque fichier ressemblant à *x64.efi *ia32.efi.
- Deux formats de tables sont pris en charges, MBR et GPT, cependant je vous recommande le partitionnement sur une table GPT, elle accepte jusqu'à 128 partitions primaires.
- Pour qu'un amorçage UEFI ait lieu sur un disque fixe, ce dernier doit comporter une partition spéciale ESP (EFI File System), cela peut être contourné en utilisant les pilotes de systèmes de fichier EFI.
- L'ESP devrait officiellement utiliser un système de fichier FAT32, bien que de nombreuses distributions Linux utilisent un système de fichier FAT16, quelquefois un FAT12.

GPT et FAT32 sont un bon compromis.

- Des utilisateurs ont constaté que certains EFI ont des bugs avec certains boot loaders,qui causent des problèmes avec les ESP inférieures à 512MiB (537MB) où des fichiers ne peuvent pas être lus par l'EFI, donc je vous recommande la création d'une ESP d'au moins 550MiB.
- Chaque chargeur de démarrage EFI doit être stocké dans un sous-répertoire du répertoire EFI sur l'ESP.
- Ces répertoires sont généralement nommés d'après le système d'exploitation qui les a créés. Par exemple:
- Debian met ses fichiers dans EFI/debian
- Ubuntu met ses fichiers EFI dans EFI/ubuntu
- Red Hat met ses fichiers dans EFI/redhat

Une fois Linux installé, l'ESP est monté dans **/boot/efi** de sorte que l'arborescence soit :

```
/boot/efi/EFI/ubuntu
```

```
/boot/efi/EFI/redhat
```

```
/boot/efi/EFI/debian
```

Si vous avez besoin d'installer un chargeur de démarrage indépendamment de votre système d'exploitation, vous pouvez créer votre propre répertoire.
Exemple :

```
 EFI/refind pour Refind.
```

rEFIND, EFI boot manager

La plupart des ordinateurs avec une architecture x86-64 et un firmware EFI intègrent un module de compatibilité appellé CSM (Compatibility Support Module), qui est un mode d'émulation du BIOS. D'autres firmwares EFI sont construits au-dessus d' un bios traditionnel, ainsi il est possible de garder les capacités du bios et booter un ordinateur EFI, en mode legacy.

Cependant il peut arriver que certaines cartes mères n'offrent pas ces options ou que vous souhaitiez

par soucis de facilité, utiliser un boot loader plus polyvalent et pouvoir lancer une image disque d'une distribution Linux n'offrant pas le support pour l' UEFI, en mode EFI.

Il existe de nombreux chargeurs de démarrage EFI, tel que :

- **Elilo** : Elilo est le plus ancien boot loader stable EFI.
- **Grub legacy** : La version officiel de Grub legacy ne supporte pas le démarrage EFI,cependant Fedora(>=17) utilise une version qui inclut ce support.
- **Grub 2** : Grub2, peut chaîner des d'autres chargeurs de démarrage EFI, peut lire de nombreux systèmes de fichiers, permet de passé des options au noyau avant le démarrage, détecte et configure d'autres systèmes d'exploitation et fournit un Shell.
- **Efi stub loader** : Fin 2011, les développeurs du noyau Linux, transforme le kernel en application EFI, ainsi il est possible de passer la main au kernel sans utiliser de chargeur de démarrage. (A partir du kernel 3.3.0)
- **Refit** : Refit n'est pas un gestionnaire de démarrage, il ne peut charger le noyau Linux ni tout autre OS, son seul but est de charger un boot loader.
- **Gumniboot** :Gumniboot, est un gestionnaire de démarrage, il est très simple, seulement en mode texte.
- **Refind**: Refind est le couteau Suisse UEFI, c'est un gestionnaire de démarrage, il détecte automatiquement les systèmes et chargeur de démarrage présents sur vos disques.

Refind est un un fork REFIT (l'auteur de Refit a abandonné le projet, il n'a pas été mis a jour depuis plus de trois ans), c'est un boot manager, pour les ordinateurs implémentés par l'Extensible Firmware Interface (EFI) et Unified EFI (UEFI). Il faut bien faire la distinction entre un boot loader tel que GRUB, qui charge l'initial RAM disk et le noyau en mémoire et un boot mangager qui charge un boot loader. rEFIND est indépendant du systèmes d'exploitation, c'est une application UEFI qui accède aux mêmes interfaces que le gestionnaire d'amorçage intégré dans le micrologiciel de votre carte mère.

Caractéristiques :

- Auto-détection des chargeurs de démarrage EFI et Bios.
- Auto-détection de l'initial RAM disk et du kernel.
- Lancement des chargeurs de démarrahe EFI et Bios.
- Lancement des utilitaires EFI Shell et Gptsync.
- Charge les pilotes EFI pour les systèmes de fichier ou périphériques non supportés nativement par votre firmware.

Mais aussi :

- La possibilité d'avoir le choix entre une interface graphique ou texte seulement.
- Graphique configurable, en ajoutant des polices, fond d'écrans et icônes personnalisés.
- Réglage de la résolution d'écran.
- Economisateur d'écran.

Il peut être utile d'avoir une clé USB de secours avec une partition réservée juste pour rEFIND car en cas de soucis au boot vous pourrez toujours démarrer votre OS indépendamment et ainsi débuguer votre EFI.

Site officiel de rEFIND

Installation sur disque amovible

Configuration

On approche de la fin, et on entame la partie la plus marrante.

Toutes la configuration se fait dans le fichier `refind.conf`.

Jetez un œil sur la page de Roderick W. Smith, Configuring the Boot manager.

Et aussi, le fichier `example.conf` qu'on a renommé tout à l'heure.

```
nano refind.conf
```

Mon fichier refind.conf:

refind.conf

```
timeout 10

icons_dir EFI/boot/icons

textonly

use_graphics_for linux

showtools shell, about, reboot, exit

scan_driver_dirs EFI/boot/drivers_x64

scanfor internal,external,optical,manual

scan_delay 1

scan_all_linux_kernels

dont_scan_dirs EFI/linuxmint,boot

default_selection Crunch_Live

menuentry Crunch_Live {
  icon /EFI/boot/icons/os_debian.icns
  volume 1:
  loader /live/vmlinuz
  initrd /live/initrd.img
  options "ro root=UUID=C847-7244 add_efi_memmap config boot=live"
  osype Linux
  graphics off
}
```

```
menuentry Crunch_Install {
  icon /EFI/boot/icons/os_debian.icns
  volume 1:
  loader /install/vmlinuz
  initrd /install/initrd.gz
  options "ro root=UUID=C847-7244 add_efi_memmap
file=/cdrom/install/crunchbang.cfg"
  osype Linux
  graphics off
}
```

rEfind est assez capricieux, votre fichier refind.conf ne doit pas avoir un pet de travers, aussi quelques fois un simple reboot permet de lancer votre kernel tranquillement. C'est pour cela qu'il n'y a pas de commentaire dans le fichier ci-dessus.

Pour vous aider vous aurez besoin du fichier `syslinux.cfg` généré par Unetbootin :

```
cd /media/usb1
```

```
cat syslinux.cfg
```

retour de la commande

```
default menu.c32
prompt 0
menu title UNetbootin
timeout 100

label unetbootindefault
menu label Default
kernel /ubnkern
append initrd=/ubninit boot=live config quiet

label ubnentry0
menu label Live Session
kernel /live/vmlinuz
append initrd=/live/initrd.img boot=live config quiet

label ubnentry1
menu label Install
kernel /install/gtk/vmlinuz
append initrd=/install/gtk/initrd.gz video=vesa:ywrap,mtrr vga=788
quiet file=/cdrom/install/crunchbang.cfg

label ubnentry2
menu label Memory Test
kernel /live/memtest
```

```
append initrd=/ubninit
```

UEFI Shell

C'est le moment de vérité, on peut redémarrer.
Dans l'interface de votre EFI, vous choisissez votre clé USB, la première partition si vous avez le choix.
Là normalement rEFIND se lance, il doit tout d'abord scanner vos disques.
En mode texte vous devriez avoir le choix entre votre OS principal et ensuite les menuentry que l'on a ajouté dans notre `refind.conf`.

Essayons Crunch_Live ...

Si vous avez de la chance, il se lance, sinon comme je vous l'ai dit plus haut, il est très capricieux, il ne doit pas y avoir un pet de travers dans votre fichier de configuration.

Enfin pas de panique, il y a le **shell UEFI** pour éditer ce fichier sans avoir besoin de redémarrer, je l'ai mis dans les tools, un peu plus haut.

Si vous voulez éditer vos fichiers, démarrer linux en ligne de commande ou ajouter une nouvelle entrée dans votre NVRAM, c'est là que ça se passe.

Le Shell EFI est en Qwerty, entraînez-vous un peu avant ! Il faut que vous repériez sur votre clavier où sont les caractères spéciaux !

Dans le menu de rEFIND, vous choisissez Shell EFI, là du texte jaune sur un fond noir c'est le Shell EFI.

La syntaxe ressemble beaucoup à DOS.

Les systèmes de fichiers sont représentés par des fs0, fs1, fs3 etc... , fs0: étant la première partition de votre clef USB, si vous lancez ce Shell depuis celle-ci.
L'antislash est de rigueur.

Exemple :

```
shell> cd fs0:\EFI\boot
fs0:\EFI\boot\> edit refind.conf
```

Quelques commandes utiles :

Commande	Description
`cd`	Se déplacer.
`rm`	Supprimer un fichier.
`mkdir`	Créer un répertoire.
`map`	Pour lister vos périphériques.
`map -t hd`	Pour filtrer la sortie, sur vos disques de USB et SATA.

Commande	Description
`edit`	Éditeur de texte, F2 pour enregistrer et F3 pour quitter.
`exit`	Quitter le shell.
`reset`	Reboot.

Références

- Linux et EFI (en)

[1)]
N'hésitez pas à y faire part de vos remarques, succès, améliorations ou échecs !