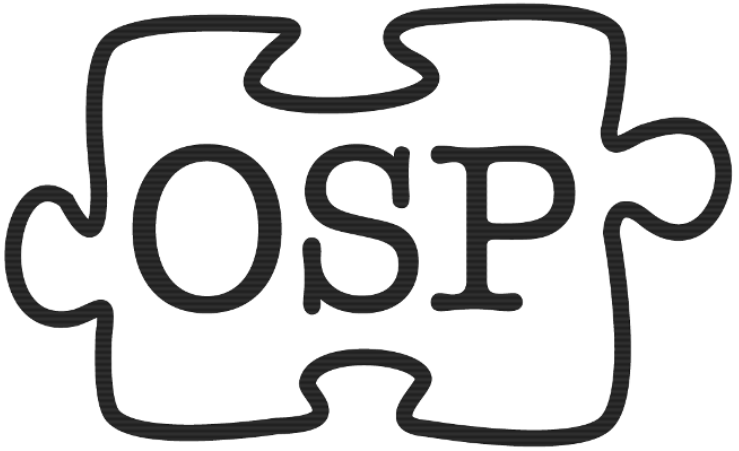
Unified Extensible Firmware Interface

- UEFI -



Open Source Party

2015

UEFI - Unified Extensible Firmware Interface:

"Software zum starten des Rechners und des Betriebssystems"

Aussprache: wie "Unify" ohne "n", oft auch nur kurz EFI

UEFI ist eine erweiterbare Firmware zum starten des Rechners und ist der Nachfolger des BIOS. Alle modernen Rechner (besonders 64Bit) sind damit ausgestattet.

Ursprüngliche Entwicklung von Intel als Ersatz für BIOS, heute spezifiziert durch das "Unified EFI Forum" (www.uefi.org).

BIOS

Aufgaben des BIOS (grob):

Hardware des Rechners soweit initialisieren, sodass ein Bootloader gestartet werden kann, der dann das Betriebssystem startet.

Initialisieren der Hardware:

- Selbsttest (CPU, RAM, ...)
- Initialisierung der wichtigsten Eingabegeräte (Tastatur, Maus)
- Initialisierung der Grafikkarte
- Initialisierung von Datenträgern zum Booten (HDD, CD, ...)
- Laden des Bootsektors

BIOS

Zum starten des Betriebssystems wird der erste Datenblock eines Boot-Mediums (HDD, CD, ...) geladen und der darin enthaltene Bootcode (Bootloader) gestartet. Der Bootcode kann dann das Betriebssystem starten, einen Bootmanager anzeigen oder andere Dinge enthalten (z.B. auch einen Virus).

Bootsektor:

- erster Sektor auf der Festplatte
- bei PCs üblicherweise der "Master Boot Record" (MBR)
- hat eine Größe von 512 Byte
- beinhaltet die Partitionstabelle und den Bootcode zum starten des Betriebssystems
- für den Bootcode stehen nur 440 Byte zur Verfügung!

UEFI

UEFI hat im Wesentlichen die gleichen Aufgaben wie BIOS, ist jedoch für moderne Hardware konzipiert.

Wesentliche Eigenschaften:

- CPU unabhängige Architektu (z.B. direkter Start im 64 Bit Modus (BIOS ist 32Bit)
- Hardware kann über ladbare Module verfügbar gemacht werden (Grafikkarte, Netzwerk, Controller, ...)
- Treiber können so systemunabhängig sein
- Netzwerkbootfähigkeit (Preboot Execution Environment)
- Shell
- Bietet einen Bootmanager (nicht alle)
- Unterstützung für GPT (GUID Partition Table)
- BIOS Verhalten für alte Betriebssysteme "Compatibility Support Module" (CSM), nicht von jeder Firmware unterstützt

MBR Partitionen

Master Boot Record – MBR:

- MBR hat nur 512 Byte (erster Sektor)
- 64 Byte davon sind für die Partitionstabelle vorgesehen
- max. 4 Partitionen (sog. Primärpartitionen)
- eine Primärpartition kann als "Erweiterte Partition" definiert werden
- "Erweiterte Partition" kann weitere Partitionen enthalten

Probleme:

- Partitionstabelle zu klein
- max. Partitionsgröße ca. 2TB
- max. Festplattengröße ca. 4TB
- Bootcode nur 440Byte

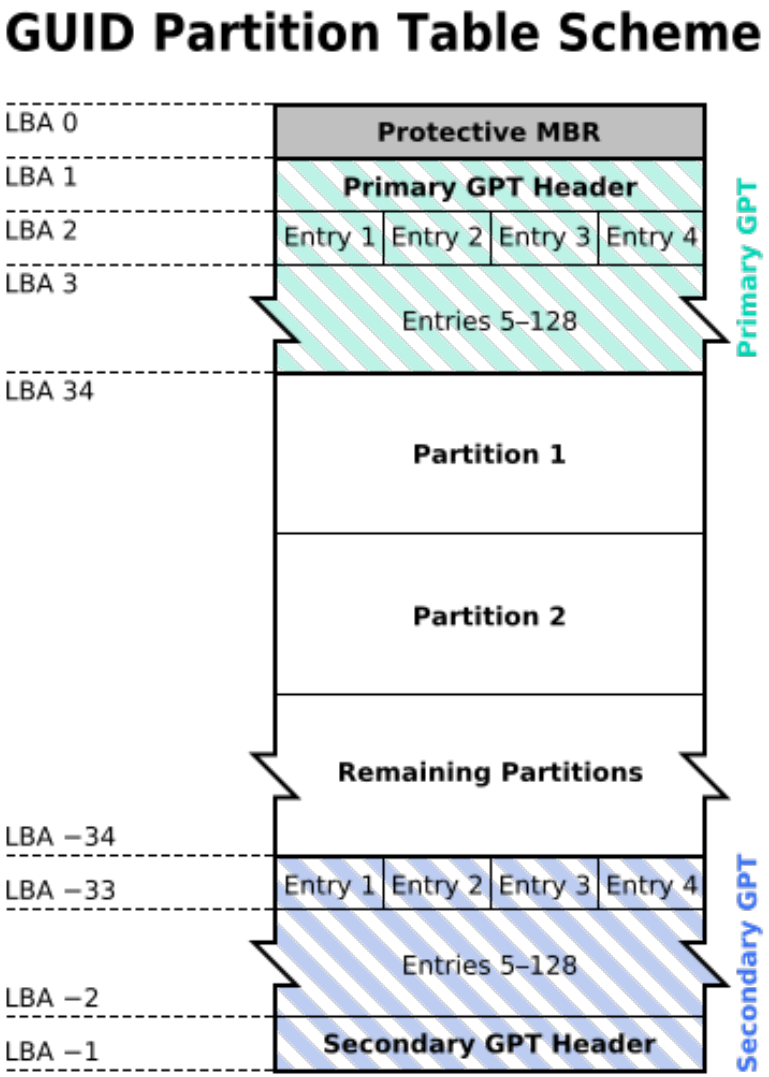
GPT Partitionen

GUID Partition Table – GPT:

- teilw. abwärtskompatibel zu MBR
  - MBR enthält dann eine GPT-Partition (Protective MBR)
- pro Partitionseintrag stehen 128 Byte zur Verfügung
- für die GPT Tabelle stehen 32 Blöcke (LBA) zur Verfügung
  - bei 512 Byte Blöcken sind das (512 / 128) * 32 = 128 Partitionen
- GPT Tabelle wird am Anfang und am Ende der Festplatte redundant mit CRC-Prüfsumme eingetragen
- kann daher im Fehlerfall wieder hergestellt werden

Mischung von GPT und MBR ist möglich jedoch nicht empfohlen.

GPT Partitionen



Quelle: Wikipedia

Startvorgang mit UEFI:

- benötigt keinen Bootsektor mit Bootcode
- es wird nach der "EFI System Partition" (ESP) gesucht
- in der ESP befinden sich die Bootloader der Betriebssysteme
- gestartet wird nach einer Bootpriorität, welche im NVRAM gespeichert wird
- ein Bootmenu kann aufgerufen werden (wird nicht von jeder Firmware unterstützt)

UEFI kann die Bootloader in der ESP automatisch erkennen und somit beliebige Datenträger (z.B. USB-Sticks) booten.

EFI System Partition – ESP

Aufbau der ESP:

- muss im FAT32 Dateisystem formatiert sein (auch FAT16 für Wechseldatenträger)
- Größe 100MB - 200MB ausreichend
- Partitionstyp MBR: `0xEF` – GPT: `0xEF00`
  - GUID für GPT: `C12A7328-F81F-11D2-BA4B-00A0C93EC93B`

Der Default-Bootloader befindet sich in:

```
/EFI/Boot/bootx64.efi
```

Genauer:

```
<ESP_PARTITION>/BOOT/BOOT<MACHINE_TYPE_SHORT_NAME>.EFI
```

Für ein 64Bit X86-64 System entspricht dies dem o.g. Namen.

ESP Dateistruktur

ESP Dateistruktur:

`/EFI/Boot/bootx64.efi` - Default Bootloader

`/EFI/beliebig/beliebig.efi` - andere Bootloader

`/EFI/beliebige/bootloader.efi ...`

Beispiele:

Microsoft Windows: `/EFI/Microsoft/Boot/Bootmgfw.efi`

Linux GRUB2 (Ubuntu): `/EFI/ubuntu/grubx64.efi`

Durch Kopieren von `/EFI/ubuntu/grubx64.efi` nach `/EFI/Boot/bootx64.efi` kann so z.B. der GRUB2 Bootloader als Default-Bootloader eingerichtet werden.

Für einen bootfähigen Wechseldatenträger (z.B. USB-Stick) genügt FAT32 mit einem Verzeichnis "`EFI`", in der sich die ESP Dateistruktur befindet.

UEFI Booteinträge

Booteinträge im NVRAM:

UEFI speichert bootfähige Geräte/Dateträger/Bootloader in Variablen in einem nichtflüchtigen Speicher (NVRAM). Diese Einträge können entweder in der Firmware (sofern unterstützt) oder mit Hilfe von EFI-Bootmanagern bearbeitet werden:

- "EasyBCD" für Windows
- "efibootmgr" für Linux

Beispiel NVRAM:

```
BootCurrent: 0003
BootOrder: 0003,0000,0001,0002
Boot0000  Setup
Boot0001  Diagnostic Programm
Boot0002  Microsoft Boot Manager
Boot0003* Linux
```

Linux efibootmgr

Der efibootmgr ist ein Kommandozeilentool, mit dem die Booteinträge bearbeitet werden können:

Auflistung der Booteinträge:

```
efibootmgr
```

Bootreihenfolge ändern:

```
efibootmgr --bootorder 0002,0003,0000,0001
```

Einmalig anderen Eintrag booten:

```
efibootmgr --bootnext 0002
```

Booteintrag löschen:

```
efibootmgr -b 0003 -B
```

Neuen Booteintrag erstellen (auf Windows Notation achten!):

```
efibootmgr --c -d /dev/sda -p 2 -l "PCLinuxOS" -l \\EFI\\pclinuxos\\grubx64.efi
```

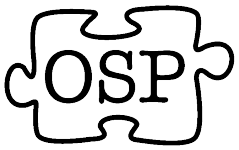
Linux UEFI Bootmanager

GRUB2:

Um Linux von UEFI starten zu können, ist ein spezieller UEFI-Bootloader nötig, der weit verbreitet GRUB geht nicht. Für GRUB2 gibt es eine UEFI-Erweiterung (grubx64.efi), dieser kann somit von UEFI gestartet werden.

Viele aktuelle Linux-Distributionen (noch nicht alle) tragen sich jedoch schon mit GRUB2 bei der Installation in UEFI ein.

Andere Distribution haben meist zumindest Pakete von GRUB2 mit GRUB-EFI enthalten und können mit ein bisschen Aufwand auf UEFI umgestellt werden.



Linux UEFI Bootmanager

Gummiboot:

Gummiboot ist ein einfacher Bootmanager, der (anders als GRUB2) in der Lage ist, den Linux-Kernel direkt zu starten. Der Kernel muss hierfür allerdings vorbereitet sein (EFISTUB). Die Aufwendige Struktur von GRUB2 ist hier nicht nötig.

Weiterhin kann gummiboot andere UEFI Bootmanager starten (teilweise auch automatisch erkennen) und stellt ein Bootmenu bereit.

Gummiboot in der ESP:

```
/EFI/gummiboot/gummibootx64
```

Die Konfigurationsdateien für gummiboot befinden sich dann unter:

```
/EFI/loader/
```

Linux UEFI Bootmanager

Von gummiboot wird automatisch erkannt:

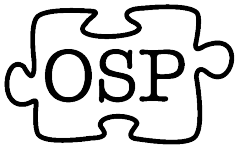
`/EFI/Boot/bootx64.efi` - EFI Default Loader

`/EFI/Microsoft/Boot/Bootmgfw.efi` - Windows Boot Manager

`/EFI/shellx64.efi` - EFI Shell

Die Booteinträge für gummiboot befinden sich dann im Verzeichnis:

```
/EFI/loader/loader.conf
/EFI/loader/entries/linux.conf
/EFI/loader/entries/ubuntu.conf
/EFI/loader/entries/unknown.conf
```



Linux UEFI Bootmanager

Beispielkonfigurationen für gummioot:

loader.conf:

```
default  linux
timeout  10
```

entries/linux.conf:

```
title    PCLinuxOS
linux    /vmlinuz
initrd   /initrd.img
options  root=UUID=dde6f8be-c4e0-0464-971e-d44ccb4a9d8b
```

UEFI Secureboot

Für die Zertifizierung von Windows 8 Rechnern ist *Secureboot* durch Microsoft vorgeschrieben.

Durch Secureboot können durch UEFI nur signierte Bootmanager, Treiber usw. ausgeführt werden. Dies soll die Systemsicherheit erhöhen, da Schadsoftware mehr so leicht eingeschleust werden kann.

Für Linux bedeutet dies, das zumindest der Bootloader (GRUB2) oder sogar der ganze Kernel kostenpflichtig signiert werden muss.

Es gibt z.Z. auch einen signierten GRUB2 Bootloader, einige Distributionen wie Ubuntu, Fedora oder SuSE nutzen diesen bereits.

Secureboot lässt sich in den meisten Fällen in der Firmware deaktivieren.

Testen von UEFI

USB-Stick:

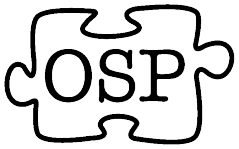
Eine einfache Möglichkeit UEFI zu testen und Erfahrungen zu sammeln besteht darin, z.B. einen USB-Stick UEFI-Bootfähig zu machen (Rechner mit UEFI vorausgesetzt).

Virtuelle Maschine:

In VirtualBox und VMware (Workstation/Player) kann ein UEFI Modus aktiviert werden:

VirtualBox: Settings->System->Enable-EFI aktivieren

VMware: in der `.vmx` Datei `firmware = "efi"` eintragen



Links

Links:

http://uefi.org/

http://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface

https://wiki.archlinux.org/index.php/Unified_Extensible_Firmware_Interface

http://wiki.ubuntuusers.de/EFI_Grundlagen