Booten
mit EFI

Prof. Dipl.-Ing. Klaus Knopper
<knopper@knopper.net>

Der Bootvorgang (classic)

1. *BIOS*
Durchsucht alle angeschlossenen Geräte auf Bootfähigkeit

2. *Bootrecord* (Verweis auf Bootloader)
Befindet sich entweder auf dem ersten Sektor der Festplatte/ROM, oder auf einer Partition, auf die
Von dort verwiesen wird. Kann auch das Boot-ROM einer Netzwerkkarte sein.

3. *Bootlader* zeigt Bootmenü oder geht direkt zum nächsten Schritt

4. *Kernel* wird (mit Hilfe von Realmode-BIOS-Routinen) in den Hauptspeicher geladen.

5. Der Kernel startet und lädt „Treiber“ bzw. „Module“ für den Hardwarezugriff, um weiter mit dem Datenträger arbeiten zu können und übergibt dann die Kontrolle an *`init`*.

Der Bootvorgang (EFI)

Unified Extensible Firmware Interface (die volle Wahrheit unter http://de.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface)

- **Soll (Argument) den „Real Mode“ mit Maschinencode und Konfigurationsmenüs im BIOS ablösen, und das Booten vereinheitlichen,**
- **In Wahrheit kommt stattdessen ein Dateisystem (FAT32) sowie architekturabhängige „Treiber-“ und „Programmstarter“ hinzu.**
- **Der „classic“-Bootmodus *soll* laut Spezifikation weiter als Option aktivierbar sein, heißt dann aber „Compatibility Support Module“ (Abkürzung *CSM*!)**
- **Im einfachsten Fall wird ein prozessorspezifischer Bootlader von Festplatte gelesen und gestartet, aber auch der Direktstart eines Linux-Kernels als „EFI-Programm“ ist möglich (`CONFIG_EFI_STUB`).**

Die EFI-Partition

- **EFI unterstützt laut Spezifikation als Dateisystem FAT32 auf der ersten primären Partition.**

- **Ein EFI-Menü (wenn vorhanden) kann direkt auf Dateien und Verzeichnisse der EFI-Partition zugreifen.**

- **Standard Pfad für das „Default-Bootprogramm“ ist:**
  ***efi/boot/bootx64.efi* (64bit) bzw.**
  ***efi/boot/bootx32.efi* (32bit)**
- **Falsches *-bit Instruktionsset führt zu Absturz oder Reset. Einige Hersteller scheinen den Default auch nicht zu kennen und zeigen trotzdem ein Dateimenü an.**

Vorteile von EFI

- *.efi-Programme können Treiber beinhalten, die vom OS benutzt werden können.
- Es ist bereits vor dem Start des OS hochauflösende Grafik möglich für Bootloader etc.
- Man könnte ein eigenes Menüsystem oder eigene Startroutinen schreiben.
- Auch Spiele etc. im EFI-Format wären denkbar, die ohne OS laufen.
- Das Betriebssystem kann im EFI-Variablenspeicher Informationen ablegen, z.B. Optionen für den nächsten Start, Details zu aufgetretenen Problemen, Logs.

Nachteile von EFI

- Verkompliziert den Bootvorgang und macht ihn durch Erhöhung des Komplexität fehleranfällig.
- Fehlerhafte Implementierung kann zu Totalschaden führen (s.a. http://www.heise.de/ct/artikel/Ausgeknipst-1798592.html)
- Das eigentlich vorgeschriebene CSM zur Start per MBR fehlt bei einigen Implementierungen (Apfel).
- Es werden weder Ressourcen für das alte BIOS eingespart (für Booten von CD/DVD und initiale Hardwarekonfiguration notwendig), noch wirkt sich EFI positiv auf Stabilität oder Performance aus.

Secure Boot

- Im EFI-Bios sind „public Keys“ hinterlegt, deren „secret keys“ Softwarehersteller zum Unterschreiben der Prüfsummen ihrer Programme verwenden können. Nur EFI-Programme mit gültiger Signatur werden gestartet.
- Laut Standard sollen Benutzer auch weitere Schlüssel für eigene Programme hinterlegen können.
- Laut Standard soll Secure Boot auch immer abschaltbar sein.
- Interessanter Ansatz für Schadsoftware-Programmierer: Secure Boot verwenden, um Benutzer auszusperren, Lösegeld kassieren, um wieder freizuschalten.

Beispiel Knoppix auf Flash

- Knoppix benutzt traditionell den `syslinux`-Bootloader.
- Neu: EFI-Verzeichnis auf bereits vorhandener FAT32-Partition enthält `syslinux.efi` (Version 6 beta mit `ldlinux`-Bibliothek) in 32bit (`bootx32.efi`, `ldlinux.e32`) und 64bit (`bootx64.efi`, `ldlinux.e64`).
- Patch: Auch unterschiedliche Konfigurationsdatei, um sofort den richtigen Kernel zu starten: `boot/syslinux/syslinux.cfg` (32), `boot/syslinux/syslnx64.cfg` (64 bit).
- Bugs: Bootgrafik funktioniert nicht, funktioniert nicht überall, `memdisk` (für DOS-Diskimages) funktioniert systembedingt nicht, weil kein Real-Modus mehr existiert.