UEFI Basics

Chris Irwin
KWLug
04 March 2019

What is UEFI

- System firmware
- Initializes devices
  - Graphics
  - Add-on Cards
- Starting your operating system
- Can perform other functions

Why UEFI?

- Allows more functionality without an OS booted
  - Hardware diagnostics
  - Remote OS installs
- Allows more pre-boot functionality
  - Signature checking (“Secure Boot”)
- Allows more boot options
  - Multiple OS selection

Why Not OpenFirmware/etc

- Intel made EFI (for Itanium)
- EFI became UEFI (now managed by foundation instead of licensed by Intel)
- Intel drives their own platform
  - Except that one time with AMD killed Itanium’s market with AMD64

Why Not BIOS

- BIOS was old
  - 16-bit real mode
  - Did a bunch of unnecessary things that the OS has to do again anyway
    - Your OS is not 16-bit real mode
  - Problems with large disks
  - Problems with bootloader size/location
- There is no BIOS standard, just documented examples of “How IBM did it” + some changes that don’t break things.

I'm Sticking With BIOS!

- Your computer will be old
  - All Computers shipped with Windows 8 or newer are UEFI-based
    - That was 2012
  - UEFI was on some systems for years before that
    - All Intel Macs (~2006)
- "My PC has the option to boot with plain BIOS"
  - Your computer actually boots a UEFI program that emulates a BIOS
  - That is going away in 2020

Boot-time problems

- What gets booted on BIOS?
  - Boot first disk?
  - What if your hard drive is plugged into SATA2, and you plug in a new disk in SATA1?
  - What if your first boot drive is SATA1, you install your OS to SATA2, then later remove SATA1?
- Installing a new OS clobbers the bootloader for any other OS
  - Install Windows first, then Linux

UEFI Bootloaders

- EFI System Partition
  - Fat32 (except Macs)
  - Contains bootloaders
- When an OS is installed, it puts a bootloader on the EFI System partition, then tells the system about it
  - Including a name
- This is why you see “Fedora” or “Windows” when you select a boot device at boot time.

UEFI Fallback boot

- Search for UEFI System partitions
- Boot “efi/boot/bootx64.efi”
  - a64 for Itanium
  - a32 for 32-bit x86
  - arm for 32-bit arm*
  - aa64 for 64-bit arm*
  - I think Apple does some slightly different things

efibootmgr

$ sudo efibootmgr

BootCurrent: 0000

Timeout: 0 seconds

BootOrder:
0000,0001,0017,0018,0019,001A,001B,001C

Boot0000* Fedora

Boot0001* rEFInd Boot Manager

Boot0010 Setup

Boot0011 Boot Menu

Boot0012 Diagnostic Splash Screen

Boot0013 Lenovo Diagnostics

Boot0014 Startup Interrupt Menu

Boot0015 Rescue and Recovery

Boot0016 MEBx Hot Key

Boot0017* USB CD

Boot0018* USB FDD

Boot0019* NVMe0

Boot001A* ATA HDD0

Boot001B* USB HDD

Boot001C* PCI LAN

Boot001D* IDER BOOT CDROM

Boot001E* IDER BOOT Floppy

Boot001F* ATA HDD

Boot0020* ATAPI CD

efibootmgr -v

$ sudo efibootmgr -v

BootCurrent: **0000**

Timeout: 0 seconds

BootOrder: **0000**,0001,0017,0018,0019,001A,001B,001C

Boot**0000*** Fedora HD(1,GPT,ef95b73b-397c-4c13-bf49-b07b2234ada9,0x800,0x79800)/File(\EFI\fedora\shimx64.efi)

Boot0001* rEFInd Boot Manager HD(1,GPT,ef95b73b-397c-4c13-bf49-b07b2234ada9,0x800,0x79800)/File(\EFI\refind\refind_x64.efi)

[...]

efibootmgr -v

Boot0000* Fedora HD(1,GPT,**ef95b73b-397c-4c13-bf49-b07b2234ada9**,0x800,0x79800)/File(\EFI\fedora\shimx64.efi)

- Looks Familiar:

$ readlink -f /dev/disk/by-partuuid/**ef95b73b-397c-4c13-bf49-b07b2234ada9**

/dev/nvme0n1p1

$ mount | grep /boot/efi

/dev/nvme0n1p1 on /boot/efi type vfat

- I’m using an NVMe drive. Consider nvme0n1p1 to be sda1

efibootmgr -v

Boot0000* Fedora HD(1,GPT,ef95b73b-397c-4c13-bf49-b07b2234ada9,0x800,0x79800)/File(**\EFI\fedora\shimx64.efi**)

$ ls -l /boot/efi/EFI/fedora | grep efi

-rwx------. 1 root root 65392 Jan 4 06:30 fwupdx64.efi
-rwx------. 1 root root 65824 Aug 8 2018 fwupia32.efi
-rwx------. 1 root root 77496 Aug 8 2018 fwupx64.efi
-rwx------. 1 root root 1744712 Oct 4 17:18 grubx64.efi
-rwx------. 1 root root 1159560 Oct 2 14:00 mmx64.efi
-rwx------. 1 root root 1210776 Oct 2 14:00 shim.efi
-rwx------. 1 root root 1210776 Oct 2 14:00 shimx64.efi
-rwx------. 1 root root 1204496 Oct 2 14:00 shimx64-fedora.efi

Adding memtest86 UEFI

- Download memtest86
- Doesn’t offer a bare UEFI file download, only a bootable USB image
- I restored the image to a USB disk
- I then copied the memtest86 files to my EFI System Partition

Memtest86 files

$ sudo cp -r /run/media/cirwin/1787-0D85/EFI/BOOT /boot/efi/EFI/memtest86

$ sudo ls /boot/efi/**EFI/memtest86/**

Benchmark

blacklist.cfg

BOOTIA32.efi

BOOTX64.efi

mt86.png

unifont.bin

Add Memtest86 to boot order

$ sudo efibootmgr --create --disk /dev/nvme0n1p1 --label 'Memtest86 for KWLug' --loader '\EFI\memtest86\BOOTx64.efi'

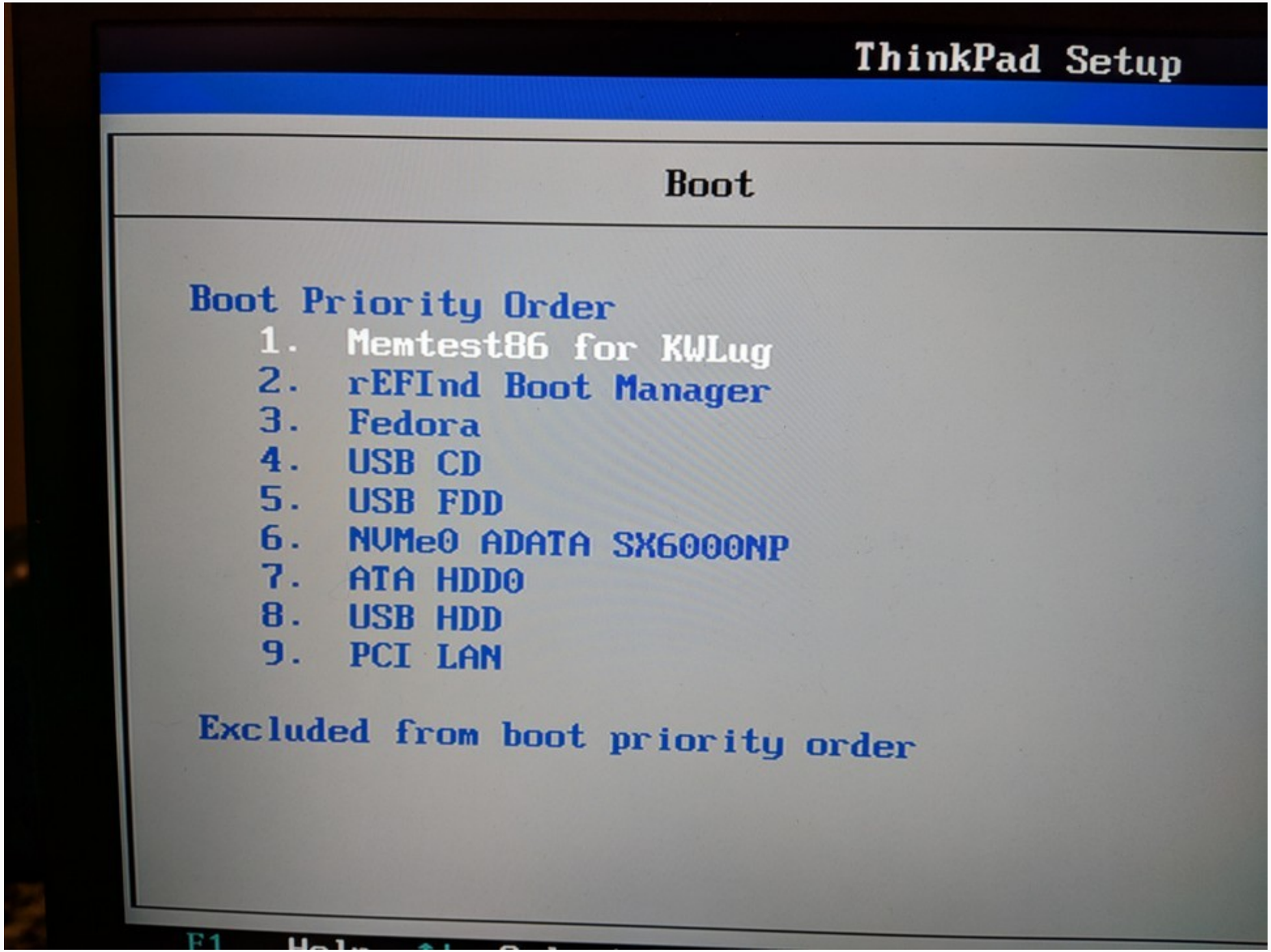
- Check results:

$ sudo efibootmgr -v | grep Memtest

Boot0002* Memtest86 for KWLug	HD(1,GPT,ef95b73b-397c-4c13-bf49-b07b2234ada9,0x800,0x79800)/File(\EFI\memtest86\BOOTx64.efi)


Boot Menu
App Menu
Memtest86 for KWLug
rEFInd Boot Manager
Fedora
NVMe0: ADATA SX6000NP


Please select an icon to continue
Use the Left / Right arrow keys, or mouse
PASSMARK
MemTest86™
Exit
Config
Lenovo


ThinkPad Setup
Boot
Boot Priority Order
1. Memtest86 for KWLug
2. rEFInd Boot Manager
3. Fedora
4. USB CD
5. USB FDD
6. NVMe0 ADATA SX6000NP
7. ATA HDD0
8. USB HDD
9. PCI LAN
Excluded from boot priority order

Fixing boot order

```
$ sudo efibootmgr -v | grep BootOrder
BootOrder:
0002,0000,0001,0017,0018,0019,001A,001B,001C


$ sudo efibootmgr --bootorder 0000,0001
[...]
BootOrder: 0000,0001
[...]
```

References

- Matthew Garrett
- Adam Williamson
- man efibootmgr
- Unified Extensible Firmware Interface - Wikipedia
- rEFInd