Computer Science CSCI 251

Systems and Networks

Dr. Peter Walsh
Department of Computer Science
Vancouver Island University
peter.walsh@viu.ca

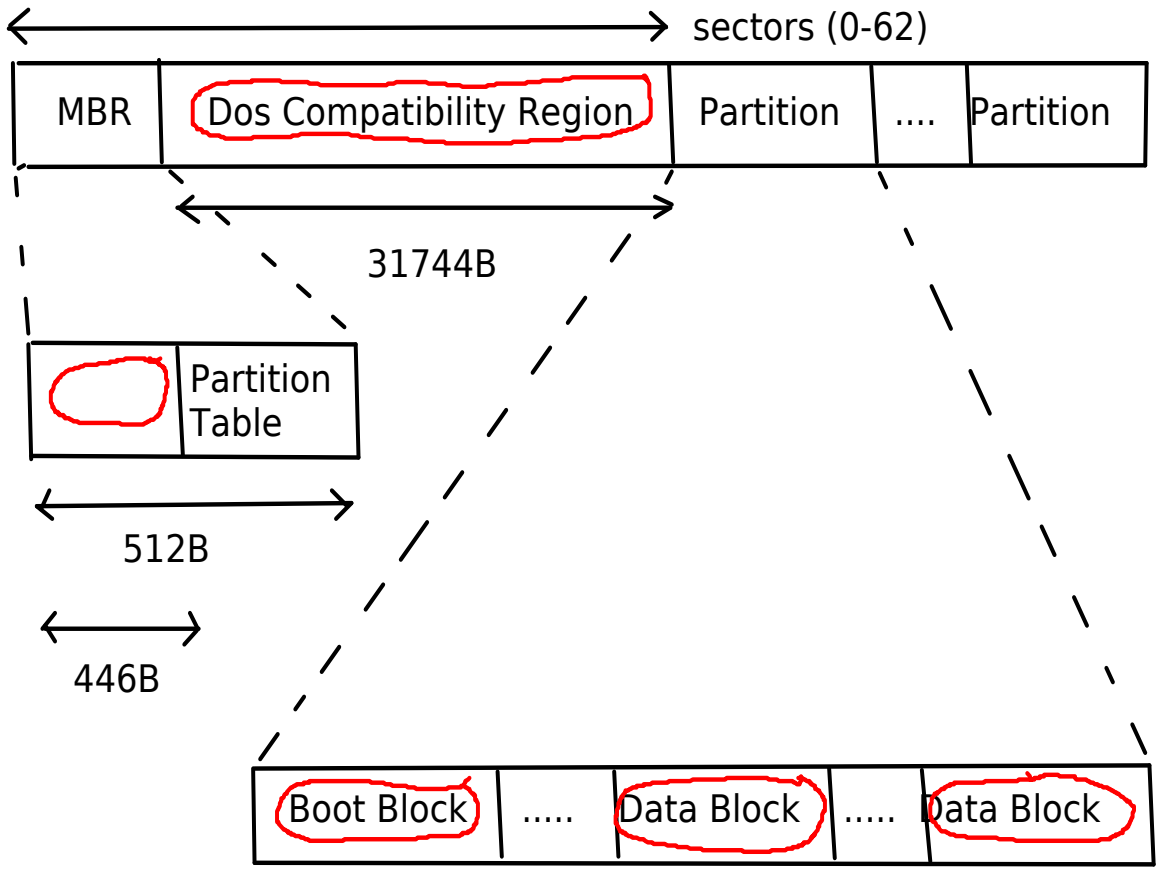
How Linux Boots (Abstraction)

- Step 1
  - firmware performs POST (power-on self test) and initiates loading the bootloader
  - choices: BIOS and UEFI
- Step 2
  - the bootloader completes loading itself into memory, (if necessary) then loads the kernel into memory and starts kernel execution
  - choices: LILO, GRUB1 and GRUB2
- Step 3
  - kernel initializes devices and memory and starts the `init` process
  - choices: SysV, Upstart and SystemD

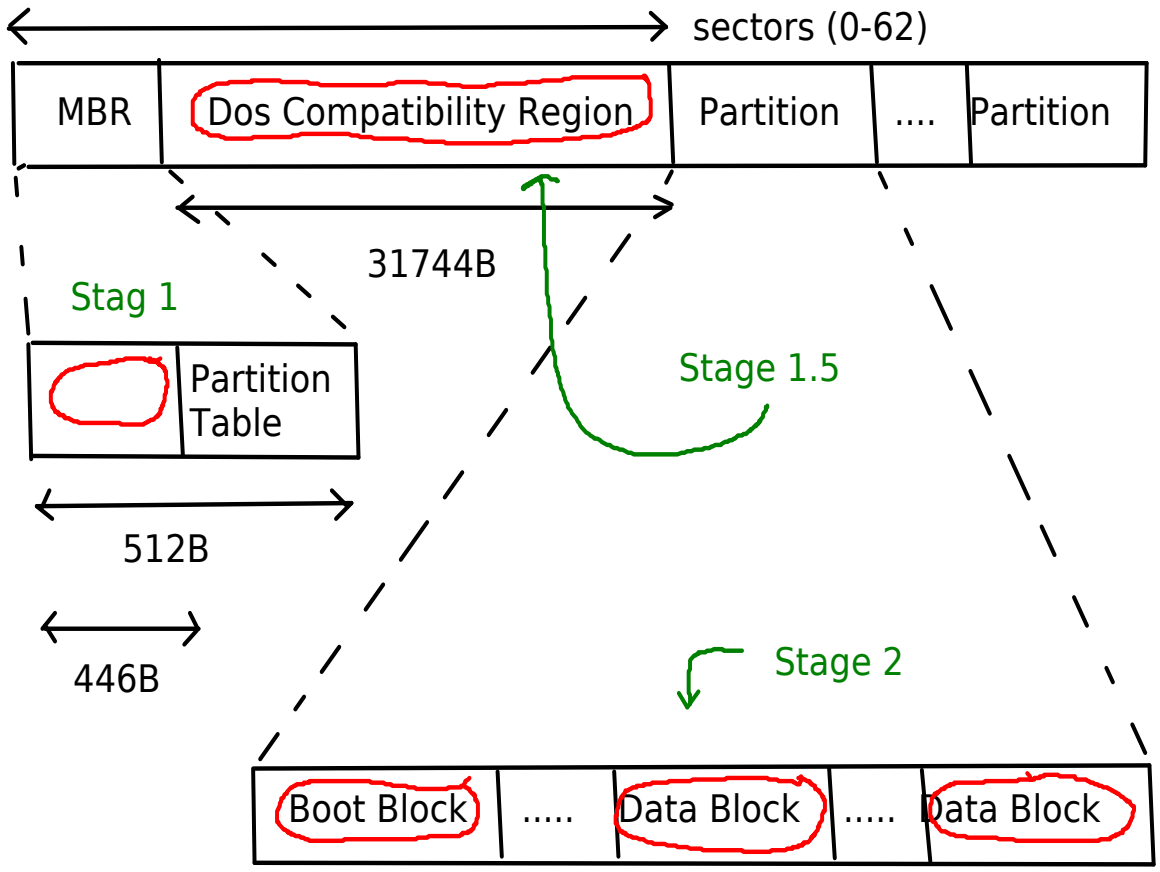
Firmware

- BIOS (Legacy)
  - de facto standard for IBM PC compatible computer
  - written mostly in assembly language
  - no knowledge of partitions or file systems
  - executes code from MBR
- UEFI
  - specification V2.8 (2020) managed by UEFI Forum
  - generally written in C
  - can read a partition table, access data and execute code contained in files within a FAT file system in an EFI partition
  - to ensure backward compatibility, UEFI on IBM PC compatible computers support BIOS booting (Intel removed support for Compatibility Support Module (CSM) in 2020)

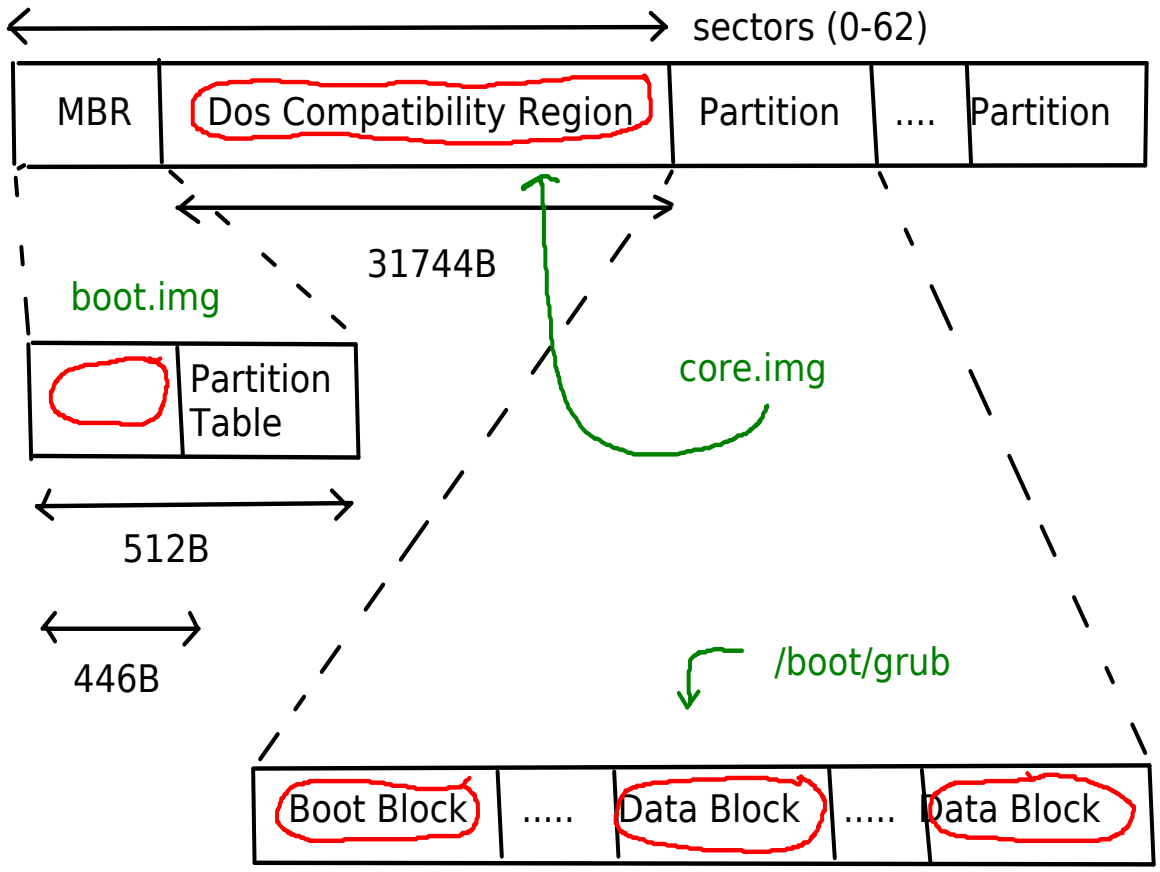
Boot Loader Installation (BIOS)



Typical GRUB1 Installation (BIOS)



Typical GRUB2 Installation (BIOS)



Boot Loader Installation (UEFI)

```
/boot/efi/EFI
    boot
        bootx64.efi
    ubuntu
        grubx64.efi
    debian
        grubx64.efi
```

GRUB1 Configuration

- edit `/boot/grub/menu.lst`

```
title          Tiny Core
root           (hd0,0)
kernel         /home/peter/TinyCore/vmlinuz root=/dev/sdb1
initrd         /home/peter/TinyCore/core.gz
```

GRUB2 Configuration

- reference `/boot/grub/grub.cfg`
- edit `/etc/grub.d/40_custom`
- execute `update-grub`

```
menuentry "Tiny Core" {
set root=(hd0,1)
        linux /home/peter/TinyCore/vmlinuz root=/dev/sdb1
        initrd /home/peter/TinyCore/core.gz
}
```