

# Booting Up and Shutting Down

lctseng (2019-2021, CC BY-SA)

? (1996-2018)

國立陽明交通大學資工系資訊中心

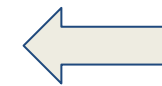
Computer Center, Department of Computer Science, NYCU

# Handbook and Manual pages

- Complete guide and be found at
  - <https://www.freebsd.org/doc/en/books/handbook/boot.html>
  - [https://www.freebsd.org/doc/zh\\_TW/books/handbook/boot.html](https://www.freebsd.org/doc/zh_TW/books/handbook/boot.html)

# Booting Overview - After Powering On

- BIOS (Basic Input/Output System) - stored on motherboard
  - Find MBR in the bootable media (disk, cd, usb stick, ...)
- MBR (Master Boot Record) - stored on the first sector of disk/media
  - Record partition information of the disk
  - Load boot loader in Boot Sector (prompt if multiple choices available)
- Boot Sector - stored in the each slice (outside of usual file system)
  - Recognize FreeBSD file system. Find kernel loader under /boot
- Kernel loader - stored in main file system (all below)
  - Show booting prompt and load selected kernel
- OS Kernel
  - Initialize hardware/drivers
- Init
  - Mount filesystem, acquire DHCP, start shell, ...



# MBR – Master Boot Record

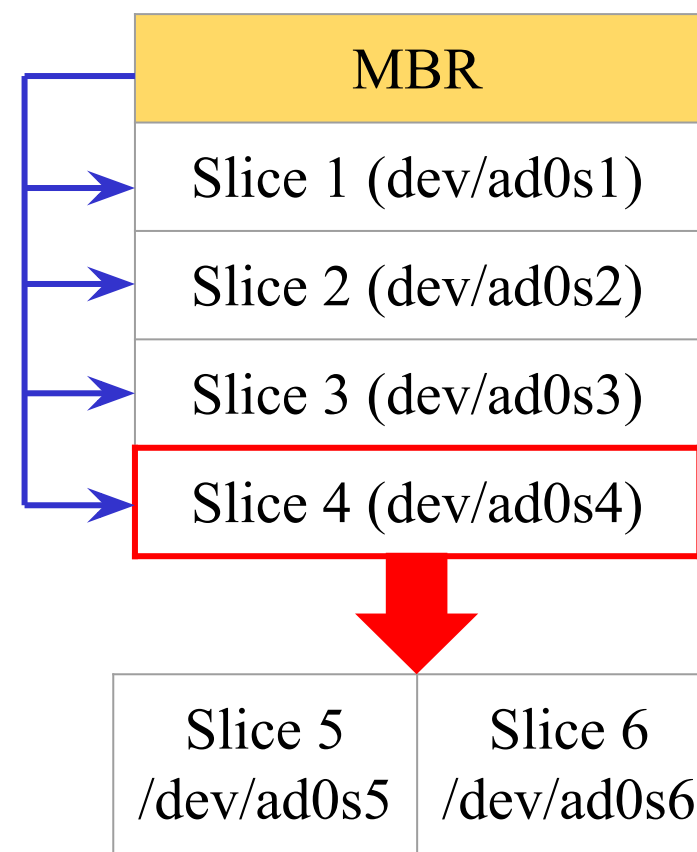
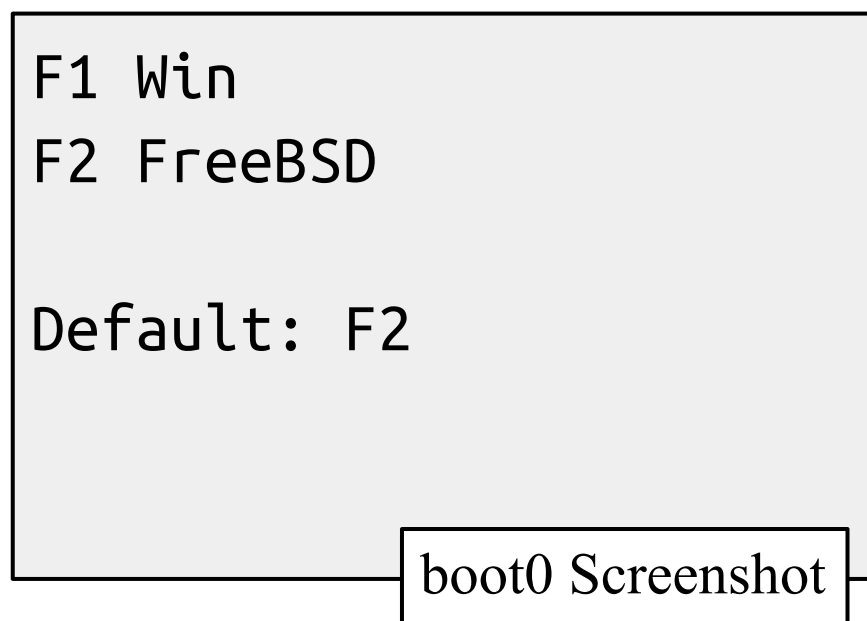
- First 512 bytes of disk, outside the FreeBSD filesystem
  - Last 2 Bytes are 0x55AA
  - Corresponding copy in FreeBSD is **/boot/boot0** or **/boot/mbr**

```
nctucs [~] -lctseng- ls -l /boot/boot0
-r--r--r--  1 root  wheel  512 Nov 12  2014 /boot/boot0
nctucs [~] -lctseng- ls -l /boot/mbr
-r--r--r--  1 root  wheel  512 Nov 12  2014 /boot/mbr
```

```
nctucs [~] -lctseng- xxd /boot/mbr
00000000: fc31 c08e c08e d88e d0bc 007c be1a 7cbf  .1.....|..|.
00000010: 1a06 b9e6 01f3 a4e9 008a 31f6 bbbe 07b1  .....1.....
...
000001d0: 0000 0000 0000 0000 0000 0000 0000 0000  .....
000001e0: 0000 0000 0000 0000 0000 0000 0000 0000  .....
000001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  .....U.
```

# MBR – Master Boot Record

- 466 bytes - code for booting
- 64 bytes - partition table
- Responsible to find the boot code on the boot sector of bootable slice.



# FreeBSD Booting Stages

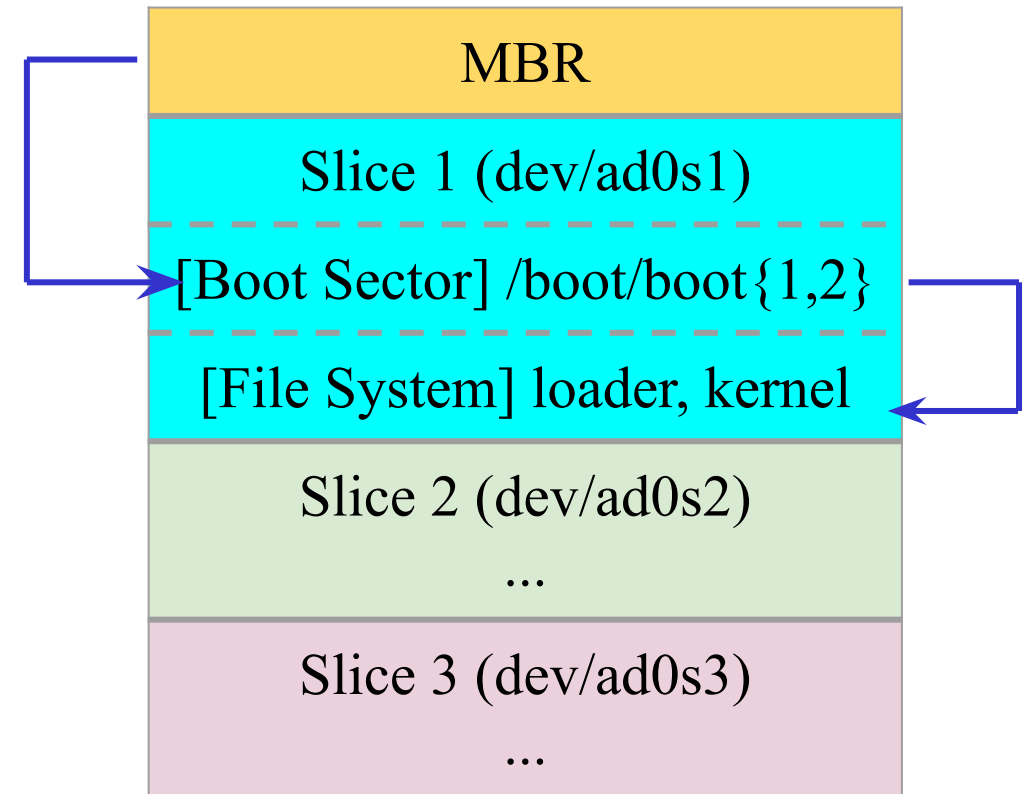
- Stage 0 (MBR)
  - /boot/mbr or /boot/boot0
  - Finds bootable partitions
- Stage 1 (Boot Sector)
  - /boot/boot1
  - Limited to 512 byte. Only recognize bsdlablel.
  - Find /boot/boot2 in somewhere of Boot Sector
- Stage 2 (Boot Sector)
  - /boot/boot2
  - Find /boot/loader or load kernel directly
- Stage 3 (BSD file system)
  - /boot/loader
  - Show prompt and load kernel

# Boot Stage One and Stage Two

- boot1 and boot2 ( $/boot/boot1 + /boot/boot2 = /boot/boot$ )
  - In boot sector of given partition (outside of FreeBSD file system)
  - They belong to the same program, but divided into two parts because of space constraint
    - Copied from `/boot/boot`
  - Used to run the loader.

```
>> FreeBSD/i386 BOOT
Default: 0:ad(0,a)/boot/loader
boot:
```

boot2 Screenshot





# Boot Stage Three



- Boot Stage Three: The loader
  - Provide a user-friendly interface to configure booting choice.
  - /boot/loader
    - Wait for 10 seconds then autoboot
    - Configuration
      - Kernel options, kernel modules, boot delay, ...

Default loader behavior	User-defined loader behavior
/boot/defaults/loader.conf	/boot/loader.conf

```
autoboot_delay="10"  
zfs_load="YES"
```

```
/boot/loader.conf
```



# Files in /boot/

- /boot/mbr (Standard)
  - Simplified version of boot0, blindly boot the partition marked active
- /boot/boot0 (BootMgr)
  - bootmanager
- /boot/boot{1,2} = /boot/boot
  - boot1 is very simple, since it can only be 512 bytes in size, and knows just enough about the FreeBSD **bsdlabel**, which stores information about the slice, to find and execute boot2.
  - boot2 is slightly more sophisticated, and **understands the FreeBSD file system enough to find files on it**, and can provide a simple interface to choose the kernel or loader to run /boot/loader
- /boot/loader
  - load the kernel from disk
- /boot/kernel/kernel

# MBR recover

- If MBR is overwritten by MS (or others), and you want to replace it with FreeBSD MBR:
  - Boot with CD or Floppy
  - `$ fdisk -B -b /boot/boot0 ad0 or boot0cfg -B /dev/ad0`
    - -B means reinitialize the boot code contained in sector 0 of the disk
    - -b is used to specify the boot code
- If you want to replace it with MS MBR
  - Boot with DOS CD/floppy
  - `C:\fdisk /mbr`

# Single User Mode

- Similar to Windows "Safe Mode"
- Repair system
  - Inconsistent file system
  - Error in a boot configuration
- Reset lost root password
  - Entering single user mode **requires no password**
- Full access to local file and configuration (root permission)
- No network access

# Boot in single user mode

OS	Command
FreeBSD	Interrupt the boot loader and type "boot -s" Or type "2" in the menu
Linux	Lilo: linux single Grub: append 'single' in the boot menu
Solaris	Press "STOP" and "a" to enter the boot PROM and Press "boot -s"

# Insecure single user mode

- Single user mode requires **no password** by default
  - When the physical security to the console is considerable, set console to be insecure in `/etc/ttys`

```
# name  getty                type  status
comments
#
# If console is marked "insecure", then init will ask
# for the root password
# when going to single-user mode.
# console none                unknown off secure
console none                  unknown off insecure
```

# Using Single User Mode

- Only the root partition is mounted and mounted as read only
  - `mount -u /`
    - Indicates that the status of an already mounted file system should be changed
  - `mount -a -t ufs (or other external types)`
    - Mount all file systems with specific type
  - `swapon -a`
    - Enable all swap

# Multibooting (1)

- FreeBSD
  - FreeBSD boot loader will try to detect bootable partitions
  - You can also declare the bootable partitions explicitly with `boot0cfg`
    - `$ boot0cfg -B -m 0x7 ad0`
      - `-m` means mask, Specify slices to be enabled/disabled,
    - E.g. `0x7` means `0111`, boot menu will detect slice 1~3 to show the options (and slice 4 is disabled)



# Multibooting (2)

- Linux
  - Using lilo or grub

```
default 0
timeout 30
fallback 1

# For booting GNU/Linux
title GNU/Linux
kernel (hd1,0)/vmlinuz root=/dev/hdb1

# For booting FreeBSD
title FreeBSD
root (hd0,2,a)
kernel /boot/loader

# For booting Windows NT or Windows95
title Windows NT / Windows 95 boot menu
root (hd0,0)
makeactive
chainloader +1
```

# Steps in the boot process

- Loading and initialization of the kernel
- Device detection and configuration
- Creation of spontaneous system processes
- Execution of system startup scripts
- Multiuser operation

# Steps in the boot process – Kernel initialization

- Get kernel image into memory to be executed
- Perform memory test
  - Allocate internal data structures of kernel

OS	Kernel Image Path
FreeBSD	/boot/kernel/kernel
Linux	/boot/vmlinuz
Solaris	/kernel/genunix
SunOS	/vmunix

# Steps in the boot process – Hardware configuration

- Devices specified in kernel configuration file
  - Kernel will try to locate and initialize it
- Devices not specified in kernel configuration file
  - Kernel tries to determine other information by probing the bus
    - If the driver is missing or not responsible to the probe, device is disabled
  - We can load kernel module to support this device.
    - `kldload`, `kldstat`, `kldunload`
    - `/boot/kernel/*.ko`

```
if_em_load="YES"  
vboxdrv_load="YES"  
vboxnet_enable="YES"
```

`/boot/loader.conf`

# Steps in the boot process – System Processes

- Spontaneous process
- Not created by the normal UNIX fork mechanism

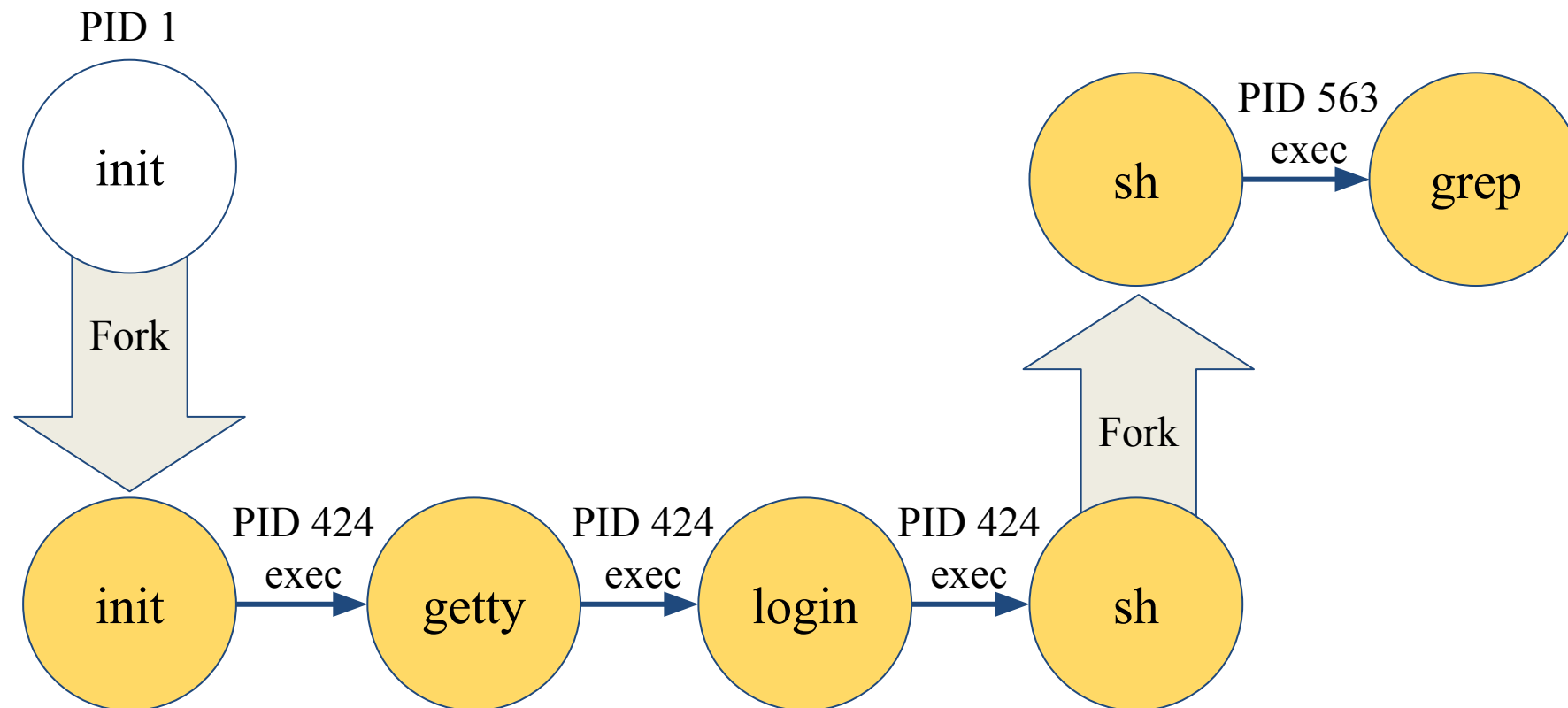
OS	Pid 0	Pid 1	Pid 2 and more
FreeBSD	kernel	init	g_event
Linux	-	init	kthreadd, kflushed, kupdate kpiod, kswapd
SunOS	sched	init	pageout

# Steps in the boot process – Execution of startup scripts

- The startup scripts are selected and run by init
- Typical works are:
  - Setting the name of the computer
  - Setting the time zone
  - Checking the disk with fsck
  - Mounting the system's disks
  - Removing files from /tmp directory
  - Configuring network interface
  - Starting up daemons and network services

# Steps in the boot process – Multiuser operation

- From now on, the system is fully operational, but no one can login
  - init will spawn getty processes to listen for login





# FreeBSD startup scripts

- The BSD-style booting
- init will run `/etc/rc`
- `/etc/rc` will reads the following configuration
  - `/etc/defaults/rc.conf`
  - `/etc/rc.conf`
  - `/etc/rc.d`
- [rc\(8\)](#)

# Ways to shutdown or reboot

- Turn off the power ← Please Don't
- Use the [shutdown\(8\)](#) command
  - Or using the halt and reboot command
    - halt = shutdown -h
    - reboot = shutdown -r

# Ways to shutdown or reboot – shutdown command

OS	Pathname	Time	Reboot	Halt	Single User Mode	Skip Fscck
FreeBSD	/sbin/shutdown	time	-r	-h		
Linux	/sbin/shutdown	time	-r	-h		
Solaris	/usr/sbin/shutdown	-gsecs	-i6	-i0	-is	
SunOS	/usr/sbin/shutdown	+mins	-r	-h		-f

- Time format can be
  - +m
  - hh:mm => Linux
  - yymmddhhmm => FreeBSD

# Halt ? Poweroff ?

- Halt
  - Terminate all processes, write data back to disks
  - When everything is ready, tell user to turn off the power
    - Or reboot by press any key
  - In older systems, you need to manually do so

```
The operating system has halted.  
Please press any key to reboot.
```

It's now safe to turn off  
your computer.

您現在可以放心關機

# Poweroff

- Halt + Turn off the power
- ACPI / APM
  - Advanced Configuration and Power Interface
  - Advanced Power Management
- In FreeBSD
  1. Try “shutdown -p now”
  2. Compile this into kernel In case it does not work...
    - device apm0 at nexus?flag 0x20
  3. Rebuild the kernel
  4. Edit /etc/rc.conf
    - apm\_enable="YES"
    - apmd\_enable="YES"
  5. Reboot
  6. Try “shtudown -p now” (or poweroff)

# Other Booting Manager

- Besides BSD-style booting, another line is System-V
  - Used by many Linux distributions
    - Solaris, Debian

# System-V Startup Scripts

- Run-level
  - `/etc/inittab`
  - `init` follow the `inittab` from level 0 to level k
- Example: `inittab` in `sun1`

Run Level	Startup scripts	Meaning
0	<code>/etc/rc.d/rc0.d/</code>	Halt
1	<code>/etc/rc.d/rc1.d/</code>	Single user mode
2	<code>/etc/rc.d/rc2.d/</code>	Multiuser without NFS
3	<code>/etc/rc.d/rc3.d/</code>	Full multiuser mode
4	<code>/etc/rc.d/rc4.d/</code>	User defined
5	<code>/etc/rc.d/rc5.d/</code>	Multiuser with graphical interface
6	<code>/etc/rc.d/rc6.d/</code>	Reboot



# Ways to shutdown or reboot – telinit

- Only for SystemV systems (and Systemd)
- telinit: change run-level
- Halt/poweroff
  - `$ telinit 0`
- Reboot
  - `$ telinit 6`
- Single user mode
  - `$ telinit 1`

# Systemd

- Modern system/service manager for many Linux distributions
  - Ubuntu, Debian, ...
- Evolved from System-V
  - Another booting manager beside BSD
  - Used by older versions of Linux distributions
  - Debian < 8.0 : System-V
  - Debian >= 8.0 : Systemd
- Similar to System-V, but faster and easier to use

# Systemd

- Use 'targets' to replace run-levels

SysV Run Level	Systemd targets	Meaning
0	runlevel0.target, poweroff.target	Poweroff
1	runlevel1.target, rescue.target	Single user mode
2	runlevel2.target, multi-user.target	User defined. Default: same as level 3
3	runlevel3.target, multi-user.target	Multiuser mode
4	runlevel4.target, multi-user.target	User defined. Default: same as level 3
5	runlevel5.target, graphical.target	Multiuser with graphical interface
6	runlevel6.target, reboot.target	Reboot