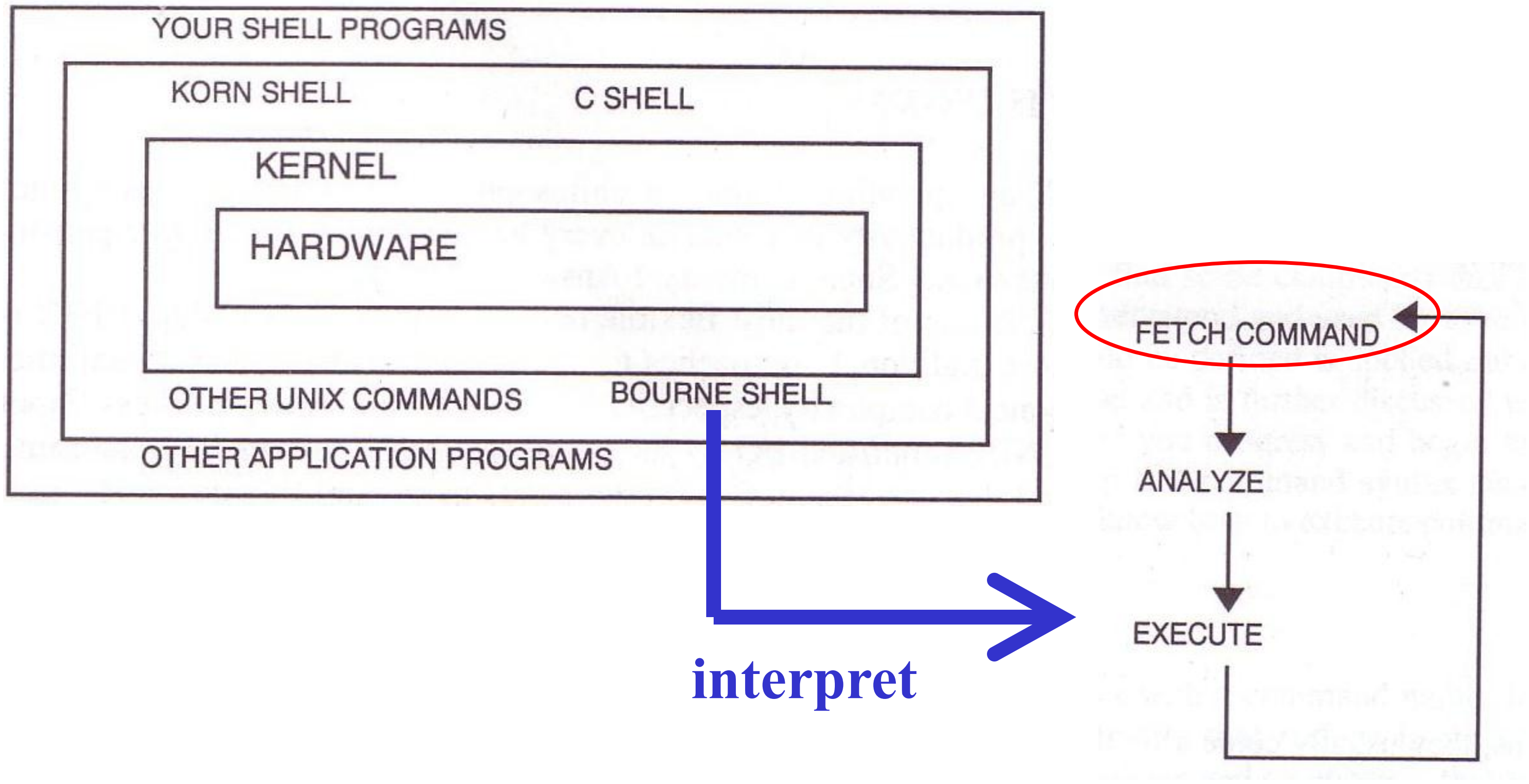
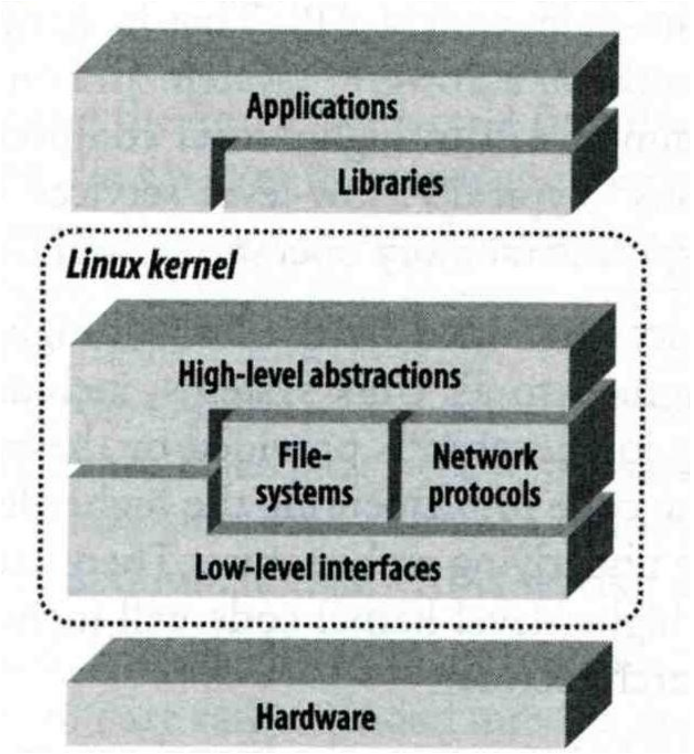
Drivers and the Kernel

Introduction – UNIX Kernel and Shell

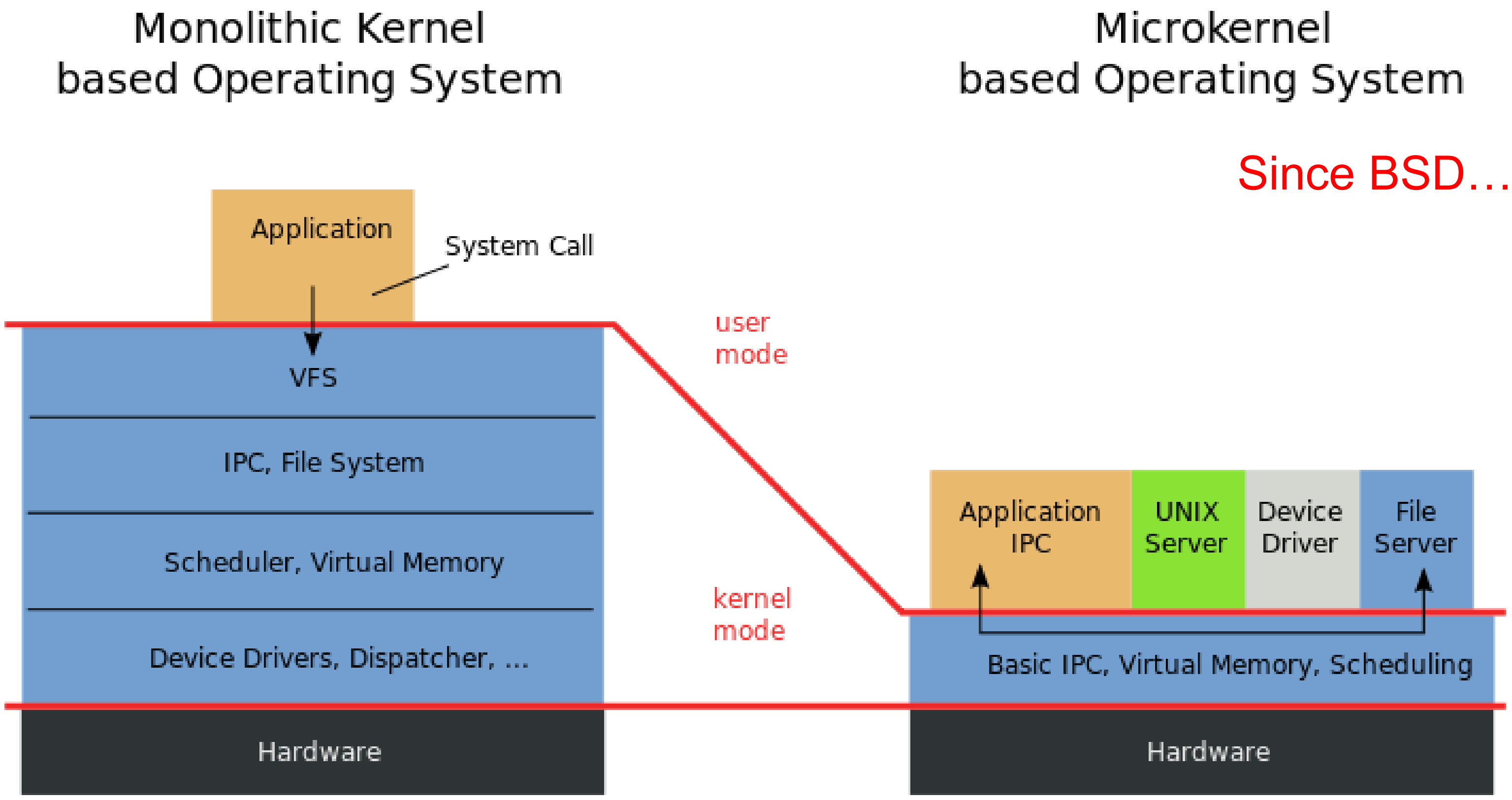


Roles of Kernel

- Components of a UNIX System
  - User-level programs
  - Kernel
  - Hardware
- Two roles of kernel (OS)
  - High-level abstractions
    - Process managements
      - Time sharing, memory protect
    - File system management
    - Memory management
    - I/O management
  - Low-level interface
    - drivers



Kernel Types



Since BSD…

<The picture is cited from wiki>

Kernel Types

Concept of being modulized…
only provides essential functionalities;
Put other sophisticated functions into user level
e.g. I/O management in the user level

- increase scalability and less difficult in maintenance
- How to communicate?
  → Message passing – less efficient

- ❑ Two extreme types
  - **Micro kernel**
    - ➢ **Provide only necessarily, compact and small functionalities**
    - ➢ **Other functions is added via well-defined interface**
  - **Monolithic kernel (龐大的kernel – e.g. unix)**
    - ➢ **Whole functionalities in one kernel**
- ❑ Modern OS
  - Solaris
    - ➢ **Completely modular kernel**
    - ➢ **Load necessarily module when it is needed**
  - BSD/Linux-derived system
    - ➢ **Much of the kernel's functionality is contained in modules**

More integrated…

"Load only when you need."

Monolithic kernel developing towards micro kernel (being more modulized),
but without IPC (message passing) problem

Kernel related directory

- Build directory and location

System	Build Directory	Kernel file
FreeBSD	/usr/src/sys	/kernel (< 4.x) *.ko(s) /boot/kernel/kernel (> 5.x)
Red Hat	/usr/src/linux	/vmlinuz or /boot/vmlinuz
Solaris	-	/kernel/unix
SunOS	/usr/kvm/sys	/vmunix

Why configure the kernel?

Generic: with various devices…, functions supported

- ❑ The native kernel is <u>often big and common</u>

kernel image → memory usage

- ❑ Tailoring kernel to match site situation
  - Purge unnecessary kernel devices and options
  - Add functionalities that you want
- ❑ OS patch
  - Remedy <u>security hole</u> of kernel implementation
- ❑ Fine-tune system performance
  - Such as <u>adjusting important system parameters</u>
- ❑ Adding device drivers
- ❑ Fast boot time
- ❑ Lower memory usage

Building a FreeBSD Kernel

- Kernel source
  - /usr/src/sys
- Kernel configuration file
  - /usr/src/sys/<ARCH>/conf
    - GENERIC, LINT (< 4.X)
    - GENERIC, "make LINT" under this dir (> 5.x)
- Steps to build a new kernel
  - Edit /usr/src/sys/<ARCH>/conf/<KERNCONF>
  - % cd /usr/src ;
  - % make buildkernel KERNCONF=SABSD
  - % make installkernel KERNCONF=SABSD

<ARCH> represents one of i386, amd64, ia64, powerpc, sparc64

LINT file: lists all options

→ To generate LINT file

SABSD: configuration file

http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig-building.html

To Build a FreeBSD Kernel…

- ❑ What to Choose?
- ❑ What to Load?
- ❑ Option Settings?
- ❑ Device Drivers?

Finding the system hardware

Listing devices from M$ windows

- Before venturing into kernel configuration
  - Get an inventory of the machine’s hardware
  - Microsoft's **Device Manager**
- dmesg
  - cat /var/run/dmesg.log

Listing devices from dmesg

```
psm0: <PS/2 Mouse> irq 12 on atkbdc0
psm0: [GIANT-LOCKED]
psm0: [ITHREAD] psm0: model Generic PS/2 mouse, device ID 0
```

- pciconf
  - man -k *Atheros*

If not originally support by GENERIC…
Ans: Use pciconf –l to list all attached devices
Than man –k [company name] to lookup usage

```
ath0@pci0:3:0:0: class=0x020000 card=0x058a1014 chip=0x1014168c
vendor = 'Atheros Communications Inc.'
device = 'AR5212 Atheros AR5212 802.11abg wireless'
class = network subclass = ethernet
```

Building a FreeBSD Kernel – Configuration file

The explanations on options and devices…

- Each line is a control phrase

  [Ref] http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig-config.html

  - Keyword + arguments e.g. device = fxp

Keyword	Function	Example
machine	Sets the machine type	i386 or amd64
cpu	Sets the CPU type	I586_CPU or HAMMER
ident	Sets the name of the kernel	SABSD
maxusers	Sets the kernel's table sizes	0
options	Sets various comiple-time options	INET or INET6
device	Declares devices	fxp

Kernel backup

Your last chance to prevent module missing…to survive!!

- ❑ Kernel file locations

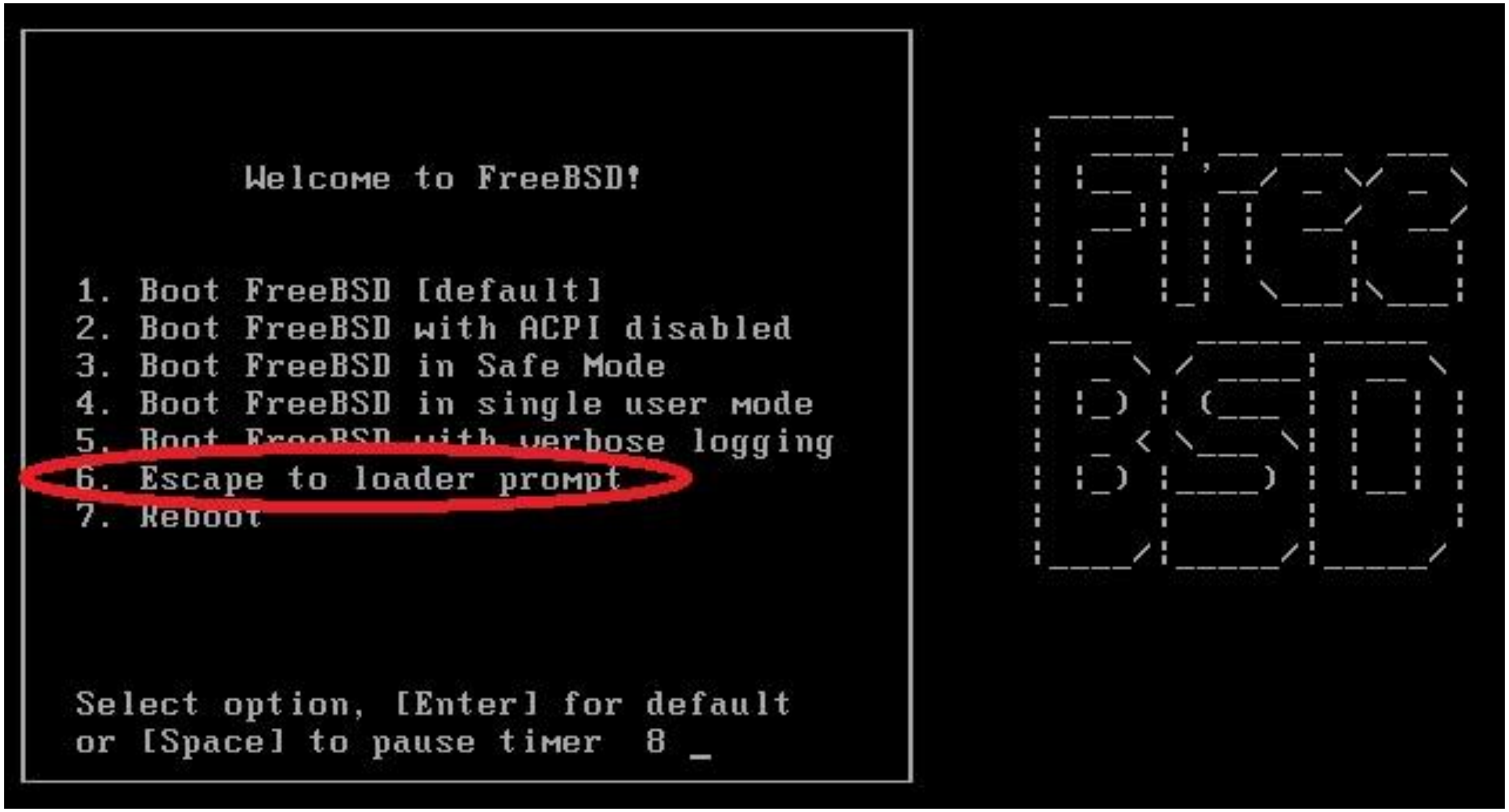
  Old kernel is automatically moved to kernel.old when you're making the new kernel

  - Put in the /boot directory
  - /boot/GENERIC/kernel, /boot/kernel.old/kernel
  - /kernel.GENERIC, /kernel.old (Freebsd 4.x)

Or just simply cp your GENERIC /boot/kernel first!

- ❑ If something goes wrong
  - ok mode !
    - ➢ unload kernel; load kernel.old/kernel
    - ➢ load kernel modules
  - mv /boot/kernel */boot/kernel.bad*

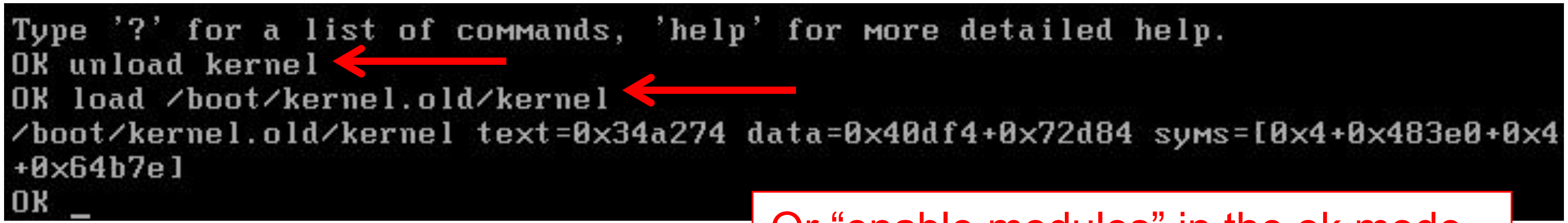
Ok mode



```
Welcome to FreeBSD!

1. Boot FreeBSD [default]
2. Boot FreeBSD with ACPI disabled
3. Boot FreeBSD in Safe Mode
4. Boot FreeBSD in single user mode
5. Boot FreeBSD with verbose logging
6. Escape to loader prompt
7. Reboot

Select option, [Enter] for default
or [Space] to pause timer  8 _
```



```
Type '?' for a list of commands, 'help' for more detailed help.
OK unload kernel
OK load /boot/kernel.old/kernel
/boot/kernel.old/kernel text=0x34a274 data=0x40df4+0x72d84 syms=[0x4+0x483e0+0x4
+0x64b7e]
OK _
```

Or “enable modules” in the ok mode..

Tuning the FreeBSD Kernel

e.g. maxusers/maxfiles and providing www service…

- sysctl command
  - Dynamically set or get kernel parameters
  - All changes made by sysctl will be lost across reboot
  - Use sysctl to tune the kernel and test it, then recompile the kernel

    The other way is to write your settings into /etc/sysctl.conf…
  - Format:

```
% sysctl [options] name[=value] …
```

Ex:

```
% sysctl –a                      list all kernel variables
% sysctl –d kern.maxfiles        print the description of the variable
% sysctl kern.maxfiles           print the value of the variable
% sudo sysctl kern.maxfiles=2048
```

Kernel modules

Module loading…
e.g. kldload if=fxp

- Kernel module location
  - /boot/kernel/*.ko → Where details can be viewed
  - /modules (Freebsd 4.x)

```
zfs[/boot/kernel] -chiahung- kldstat
Id Refs Address    Size     Name
 1   15 0xc0400000 4abd60   kernel
 2    1 0xc08ac000 13b0fc   zfs.ko
 3    2 0xc09e8000 3d5c     opensolaris.ko
 4    2 0xc09ec000 16b84    krpc.ko
 5    1 0xc0a03000 8c48     if_le.ko
```

- Load/unload kernel modules
  - kldload(8), kldunload(8)

E.g. Procedure of Loading a Device Module

❑ Loading a device module

1. pciconf –l for a device
2. man vendor name for module name in BSD
3. grep the name in /boot/kernel/*.ko
4. kldload [module name]
5. Setup permanently by
   - recompile the kernel, or
   - add [module name]_enable="YES" in /boot/loader.conf

Reference

- http://www.freebsd.org/doc/en/books/handbook/kernelconfig-config.html
- /usr/src/sys/<ARCH>/conf
  - NOTES → machine dependent kernel configuration notes.
  - LINT
  - GENERIC