資訊系統原理

郭大維教授
臺灣大學資訊工程系

1

Contents

- Computer Systems Overview
  - Operating Systems Concept
  - UNIX
  - Other System Services
- Unified Modeling Language
  - UML Introduction
  - System Development Process
  - Use Cases, Class Diagrams, etc.

2

UNIX

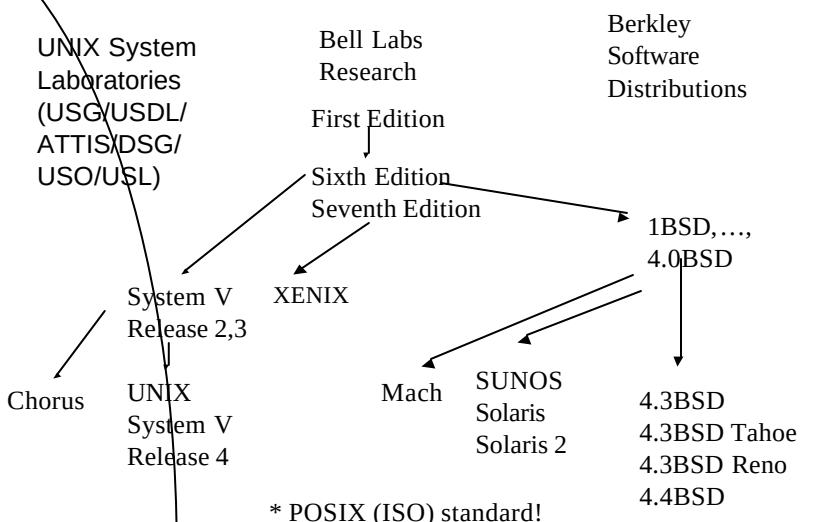
- Introduction
- Programmer Interface
- User Interface
- Process Management
- Memory Management
- File System
- I/O System
- Interprocess Communication

* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 647-693.

UNIX

- Created by Ken Thompson & Dennis Ritchie at Bell Laboratories in 1969 & on PDP-7.
  - ACM Turing award winners for the design of UNIX in 1983.
  - C programming language inventor: Dennis Ritchie.
- Major Contributors:
  - Bell Laboratories, Computer Systems Research Group (CSRG) of the University of California at Berkley (released in BSD), UNIX System Laboratories (USG/USDL/ATTIS/DSG/USO/USL), etc.

UNIX



* POSIX (ISO) standard!

5

UNIX

- Influence
  - fork() from Berkley's GINIE, 4.2 BSD file-mapping virtual memory interface from TENEX/TOPS-20, 4.4BSD virtual memory interface from MACH. fcntl system call from System V. Disk quotas and 4.3 BSD time-zone-handling package from the user community.
  - 4BSD job control, reliable signals, multiple file-access permission groups, and file system interface were adopted by AT&T UNIX System V, IEEE POSIX.1 standard, etc. 4BSD socket ported to AT&T System III. 4BSD implementation of TCP/IP networking protocol suite widely adopted!

6

UNIX - Design Goals

- 4.2BSD - 1983
  - DARPA (Defense Advanced Research Projects Agency) wanted a standard research operating systems for the VAX.
  - Networking support - remote login, file transfer (ftp), etc. Support for a wide range of hardware devices, e.g., 10Mbps Ethernet.
  - Higher-speed file system.
  - Revised virtual memory to support processes with large sparse address space (not part of the release).
  - Inter-process-communication facilities.

7

UNIX - Design Goals

- 4.3 BSD - 1986
  - Improvement of 4.2 BSD
    - Loss of performance because of many new facilities in 4.2 BSD.
    - Bug fixing, e.g., TCP/IP implementation.
    - New facilities such as TCP/IP subnet and routing support.
  - Backward compatibility with 4.2 BSD.
  - Second Version - 4.3 BSD Tahoe
    - support machines beside VAX
  - Third Version - 4.3 BSD Reno
    - freely redistributable implementation of NFS, etc.

8

UNIX - Design Goals

- 4.4 BSD - 1993
  - POSIX compatibility
  - Deficiencies remedy of 4.3 BSD
    - Support for numerous architectures such as 68K, SPARC, MIPS, PC.
    - New virtual memory better for large memory and less dependent on VAX architecture - Mach.
    - TCP/IP performance improvement and implementation of new network protocols.
    - Support of an object-oriented interface for numerous filesystem types, e.g., SUN NFS.

9

UNIX - Major UCB CSRG Distributions

- Major new facilities:
  - 3BSD, 4.0BSD, 4.2BSD, 4.4 BSD
- Bug fixes and efficiency improvement:
  - 4.1 BSD, 4.3BSD

10

UNIX

- Distinguishing Features
  - Written nearly completely in a high-level language, i.e., C.
    - High portability!
  - Distributed in source form.
    - Contributions and bug fixing from everywhere!
  - Provide powerful primitives and functions such as concurrent processes.

11

Design Principles

- Simple Algorithms for Implementation
- Replaceable Standard User Interface
  - Shell
- Time-Sharing
  - Simple Priority-Driven CPU Scheduling
- Demand-Paging Virtual Memory (4.3BSD)
  - Swapping
- Similar treatments of disk files and I/O devices

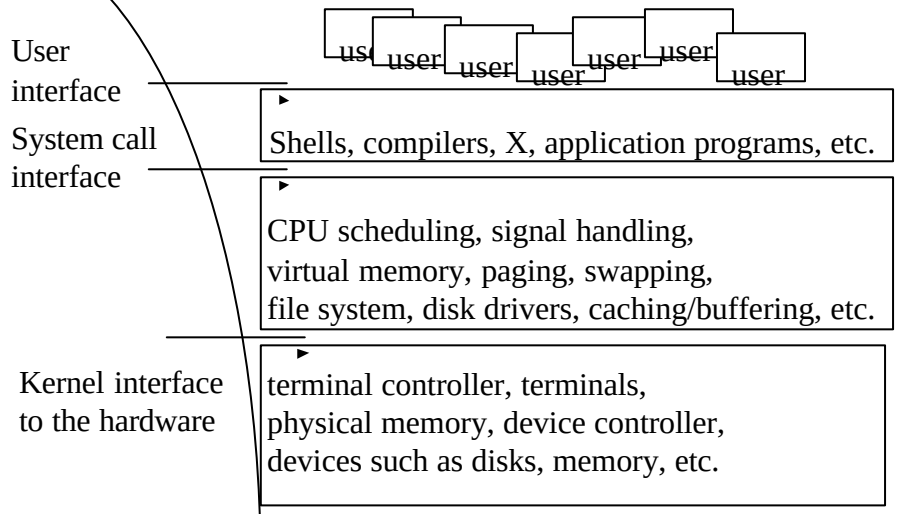
12

UNIX

- Introduction
- Programmer Interface
- User Interface
- Process Management
- Memory Management
- File System
- I/O System
- Interprocess Communication

13

UNIX Architecture



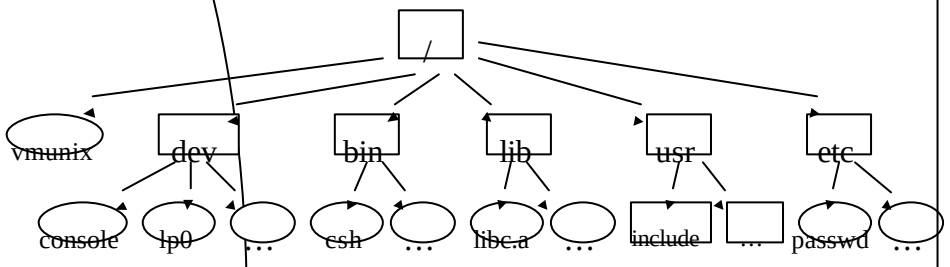
UNIX Architecture

- A Layer Architecture
  - System Calls
    - Programmer Interface to UNIX
    - Trap 40 – VAX 4.2BSD
    - R0 – error code
  - Categories
    - File Manipulation
      - Devices are special files under “/dev”!
    - Process Control
    - Information Manipulation

15

System Calls - File Manipulation

- File
  - A sequence of bytes
- Directory
  - A file that includes info on how to find other files.



16

System Calls - File Manipulation

- Path name
  - Absolute path name
    - Start at the root / of the file system
    - /user/john/fileA
  - Relative path name
    - Start at the "current directory" which is an attribute of the process accessing the path name.
    - ./dirA/fileB
- Links
  - Symbolic Link – 4.3BSD
    - A file containing the path name of another file can across file-system boundaries.
  - Hard Link
    - . or ..

17

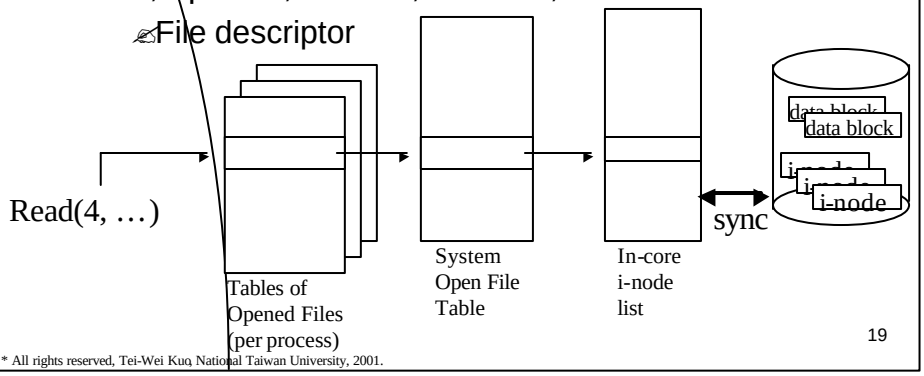
System Calls - File Manipulation

- Directories
  - /vmunix - binary root image of UNIX
  - /dev - device special files, e.g., /dev/console
  - /bin - binaries of UNIX system programs
    - /usr/ucb - written by Berkley instead of AT&T
    - /usr/local/bin - written at the local site
  - /lib - library files, e.g., those for C
  - /user - directories for users, e.g., /user/john
  - /etc - administrative files and programs, e.g., passwd
  - /tmp - temporary files

18

System Calls - File Manipulation

- Operations
  - open, close, read, write, trunc, lseek, dup, rename, chmod, chown, fcntl, ioctl, mkdir, cd, opendir, readdir, closedir, etc.
    - File descriptor

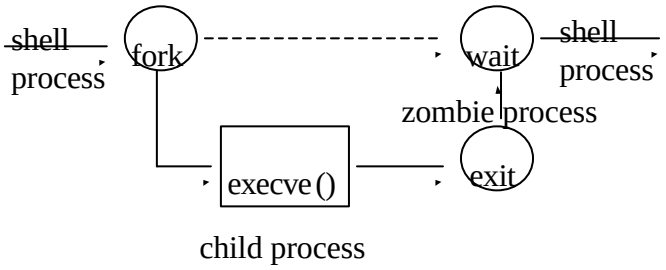


19

System Calls – Process Control

- A process
  - A program in execution

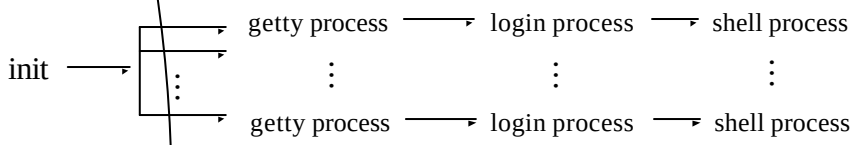
```
if (pid = fork()) {
…
wait();
} else {
execve("a.out");
}
```



20

System Calls – Process Control

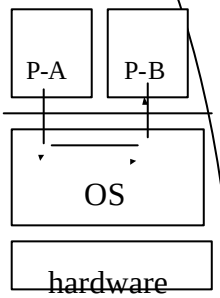
- User ID
  - Effective UID, real UID
    - With setuid bit in the inode of a file!
- Operations
  - setuid, getuid, geteuid, getgid, getegid, getgroups, etc.



21

System Calls - Signals

- Signal – no priority!
  - A facility for handling exceptional conditions similar to software interrupts
    - Generated by errors, keyboard, events, etc, e.g., kill -9 pid
  - ~ 20 signals, such as SIGSEGV, SIGINT (^C) – Multiple instances could be lost!
    - POSIX – reliable signals
  - User-level signal handlers, masking, etc.
    - Except SIGKILL, SIGINT, SIGQUIT, which is for the killing of run-away processes.



22

System Calls – Process Control

- Job (or process group)
  - A group of processes frequently cooperate to accomplish a common goal, e.g., ps | lpr.
  - setpgrp()
  - One job can use a terminal I/O at any time – foreground job!
- SIGINT, SIGSTOP, or SIGTTOU vs SIGCONT
  - freeze vs resume
- Useful for X – each window as a terminal
  - SIGWINCH

23

System Calls – Information Manipulation & Library Routine

- Information Manipulation
  - gettimeofday/settimeofday (us), gethostname, getpid, getgid, etc.
- Library routines
  - With header files, e.g., stdio.h, math.h, additional program support is provided.
  - Over 150 system calls for 4.3BSD, over 300 library functions for C
    - Eventually result in system calls!
      - getchar() -> read() if the file buffer is empty.

24

UNIX

- Introduction
- Programmer Interface
- User Interface
- Process Management
- Memory Management
- File System
- I/O System
- Interprocess Communication

25

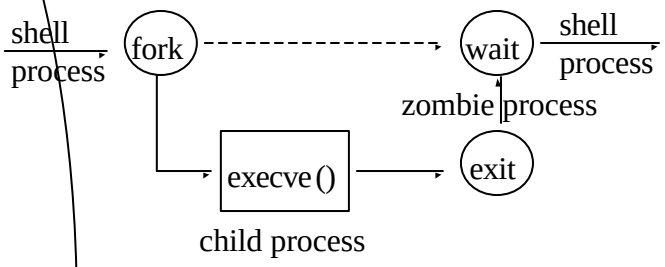
User Interface

- Interface of system programs
  - File/directory-oriented
    - Mkdir, rmdir, pwd, cd, ls, cp, mv, rm, cat, more, head, tail, diff, grep, etc.
  - Editors
    - Emacs, vi, ed, etc.
  - Compilers
    - C, Pascal, FORTRAN, etc.
  - Others
    - Mail, X, sort, etc.

26

User Interface – Shells and Commands

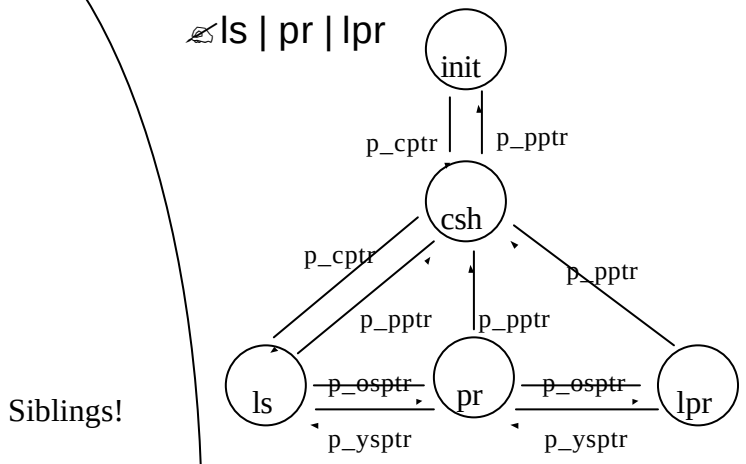
- Shell – commander interpreter
  - /bin/sh (Bourne shell by Steve Bourne), /bin/csh (C shell by Bill Joy), etc.



27

User Interface – Shells

- ls | pr | lpr



Siblings!

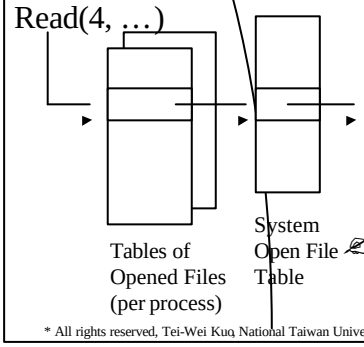
28

User Interface – Shells and Commands

- Commands
  - Shell-built-in commands, e.g., cd, etc.
  - System programs, e.g., ls, ps, etc.
    - Search Path (e.g. those in .cshrc or .login)
      - set path=(/usr/sbin /usr/ucb /usr/bin /usr/local/bin /etc /usr/etc .)
      - setenv LD_LIBRARY_PATH /usr/local/lib:/usr/lib:/usr/openwin/lib
- Foreground vs Background Cmds
  - ps& vs fg

User Interface – Standard I/O

- Standard file descriptors
  - 0, 1, 2 for stdin, stdout, stderr
- I/O redirection
  - pipe: carry data from one process to another!
    - ls | pr > filea
    - lpr < filea
    - make program >& err
- Shell script



UNIX

- Introduction
- Programmer Interface
- User Interface
- Process Management
- Memory Management
- File System
- I/O System
- Interprocess Communication

31

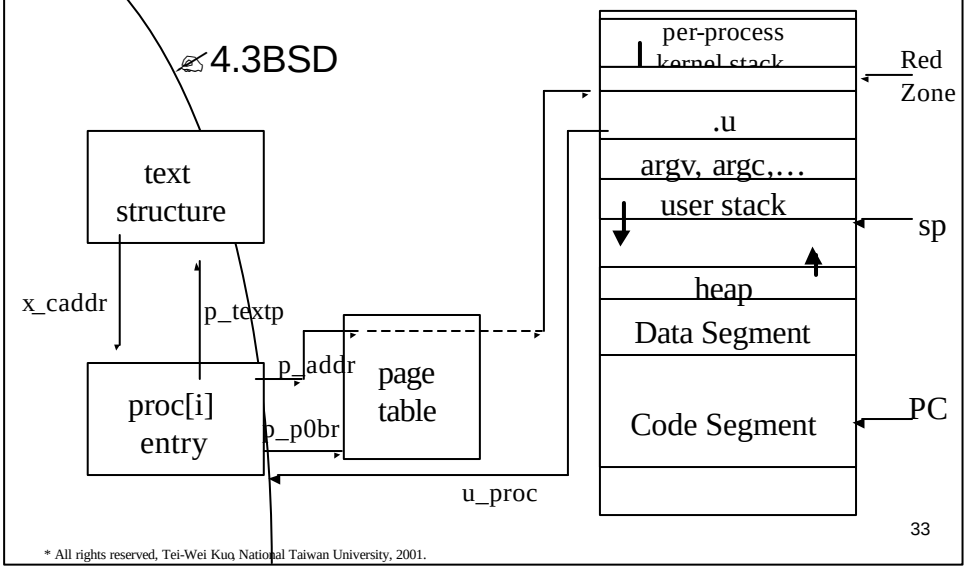
Process Management

- How to represent a process for
  - Process control
  - CPU scheduling
- Process Control Block (PCB)
  - proc[i]
    - Everything the system must know when the process is swapped out.
      - pid, priority, state, timer counters, etc.
  - .u
    - Things the system should know when process is running
      - signal disposition, statistics accounting, files[], etc.

32

Process Management - DS

- 4.3BSD



33

Process Management - DS

- proc[I] entry
  - pid, ppid
  - user_priority, system_priority
  - state, e.g., SRUN, SSLEEP,, ZOMBIE, etc.
  - signal mask, signal state
  - timer counters
  - Etc
- text structure (memory resident)
  - A list to all processes sharing the text segment – a counter is maintained!

34

Process Management - DS

- Virtual memory address space – user space
  - Text segment
    - Read-only except when debuggers' checkpoints must be set up (rw).
  - Data and stack segments
    - RW mode!
  - .u called user structure
    - System call parameters and return values, table of opened files, etc.

35

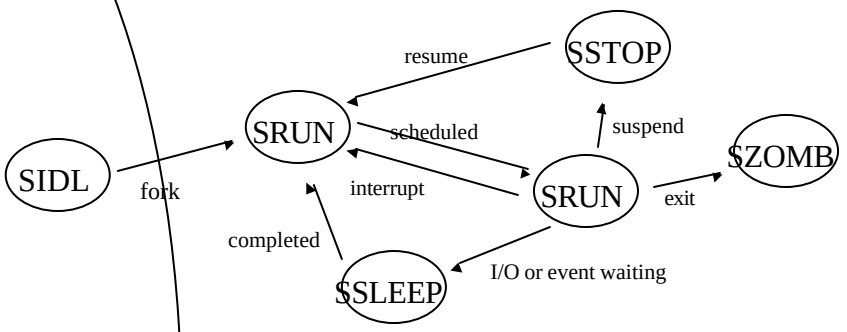
Process Management - DS

- Resources for the process in the kernel space
  - A page table per process
  - per-process kernel stack
    - For the process running in the kernel mode, e.g., for interrupt stacking.
    - System data segment = .u + per-process kernel stack
- Other resources
  - PC, CPU registers, opened files, etc.

36

Process Management – Life Cycle

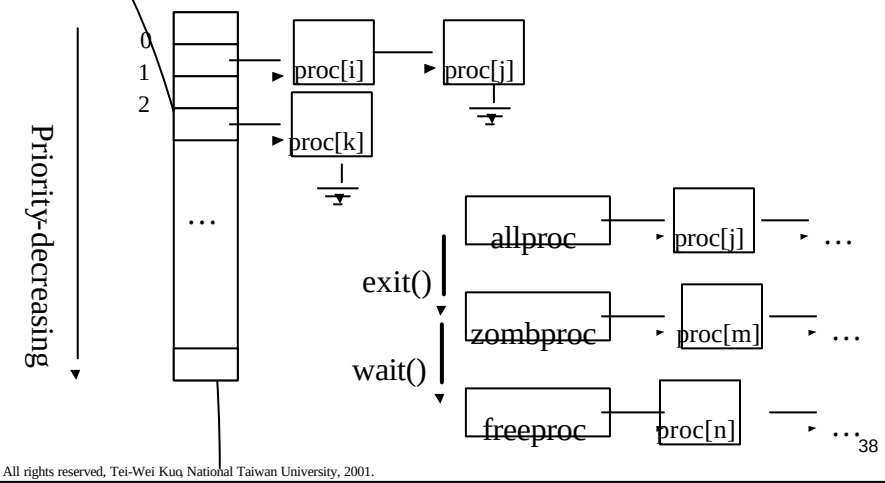
- Process State



37

Process Management - Lists

- Ready Queue



38

Process Management – fork

```
if (pid = fork()) {
…
wait();
} else {
execve("a.out");
}
```

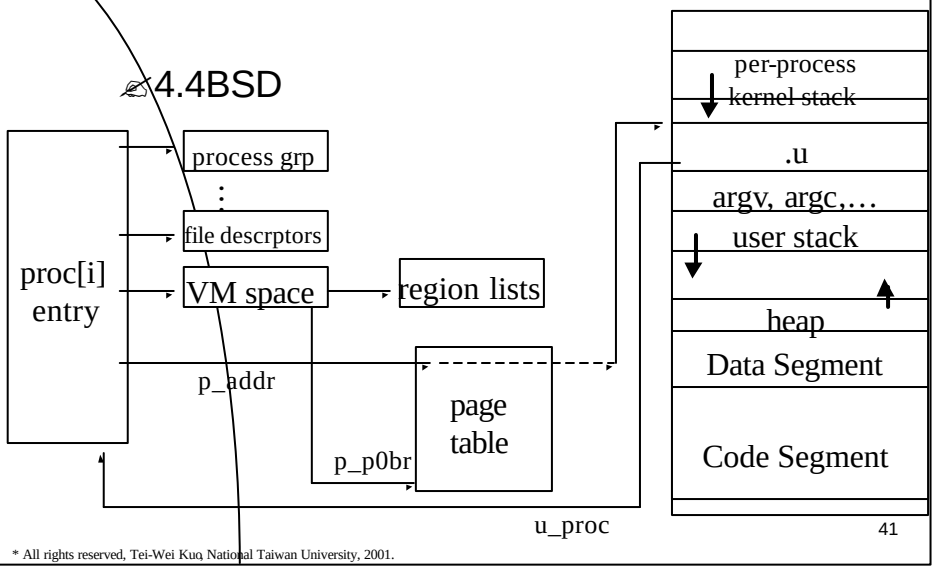
- fork()
  1. Allocate a new proc entry
  2. Register the "text structure"
  3. Allocate memory for data and stack segments
  4. Copy the data and stack segments of its parent to those of the process.
  5. Build a new page table by copying from the page table of its parent!
  6. Copy .u
     - Open file descriptors, usr/grp identifiers, signal handling, etc.

Process Management – execve/vfork

- execve()
  - Discard text, data, and stack segments of the process and reset its page table
  - Load the executable and rebuild text, data, and stack segments and its page table
  - Reset signal handling routines, etc.
- vfork()
  - Borrow segments of its parent
  - Implementation Concerns
    - Suspend the parent until the process terminates or call execve()
    - Or
      - duplicate the page table of its parent
      - Do not create pages unless Copy-on-write pages

Process Management

- 4.4BSD

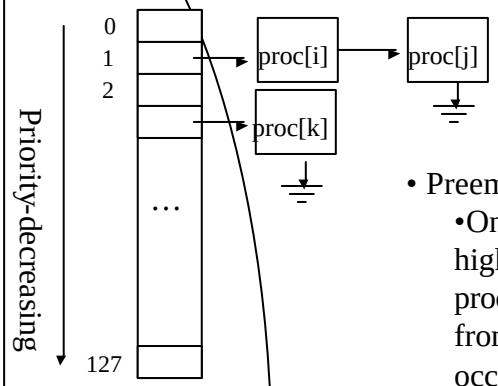


Process Management - Scheduling

- Scheduling Priority
  - User-mode: p_usrpri 50~127
  - Kernel-mode: p_priority 0 ~ 49
    - For the waiting of any event in the kernel mode.
    - Processes with p_priority between (PZERO, PUSER) (i.e., 22 and 50) would be waken up by a signal. (in 4.3BSD, PZERO = 25)
- CPU Scheduling
  - round-robin priority-driven scheduling
    - quantum = 100ms

Process Management - Scheduling

- Ready Queue



- Preemptive Scheduling Policy
  - Once a process arrives with a higher priority while the running process is in the user mode or exits from a system call, a context switch occurs immediately!

Process Management - Scheduling

- Scheduling Priority
  - Raising Priority
    - Longer period being blocked, e.g., one sec
    - Blocking in the kernel mode
    - nice() ~ -20
  - Lowering priority
    - Recent CPU usage
    - Exit from the kernel mode
    - nice() ~ +20

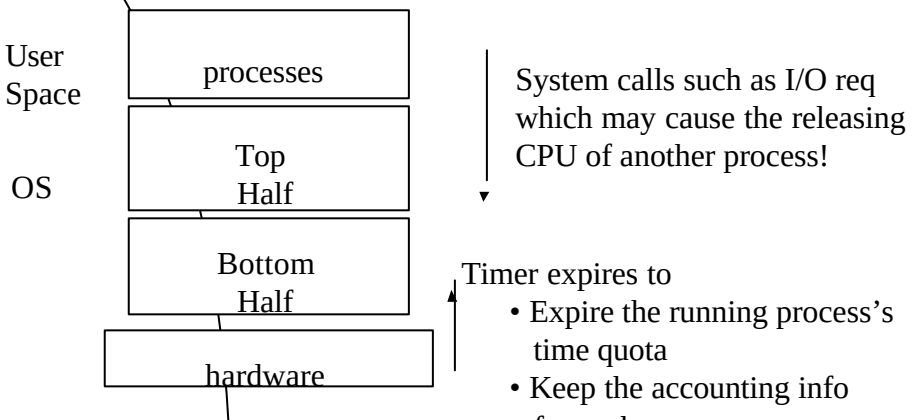
p_usrpri = PUSER + ceiling(p_cpu/4) + 2p_nice ; every tick
p_cpu = [2load/(2load+1)]*p_cpu+p_nice ; every second
p_cpu = p_cpu [2load/(2load+1)]$^{p\_slptime}$; once the process is awaken.

Process Management - Scheduling

- Context Switch
  - Synchronous CS
    - Voluntary
      - Call system calls and then sleep
    - Involuntary
      - Time quantum is up!
  - Asynchronous CS
    - Device interrupts

45

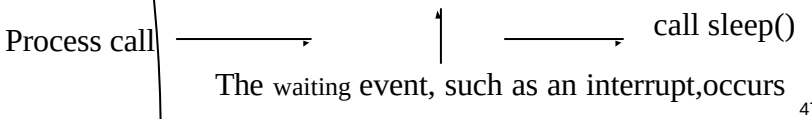
CPU Scheduling - Revisiting



46

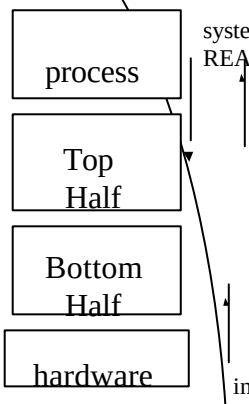
Process Management - Scheduling

- Synchronous Voluntary Context Switch
  - A system call finally invoke sleep(&wchan)!
    - wchan – address of some data struture
      - Lbolt: wait for one second
      - proc: wait for child process
      - u: wait for a signal
      - buffer header: wait for I/O operations
        - File reading, block flushing, page fault, etc.
  - Race Condition



47

Process Management - Scheduling



- A general solution in UNIX for resolving race conditions!
  - Raise hardware processor priority
    - e.g., mask interrupts
  - Single-thread kernel
- An obstacle for multi-CPU UNIX!

48

Process Management

- scheduler (pid = 0)
  - CPU scheduling
- init (pid = 1)
  - Create daemons, login processes, etc.
- pagedaemon (pid = 2)
  - Swapper – mid-term scheduler

49

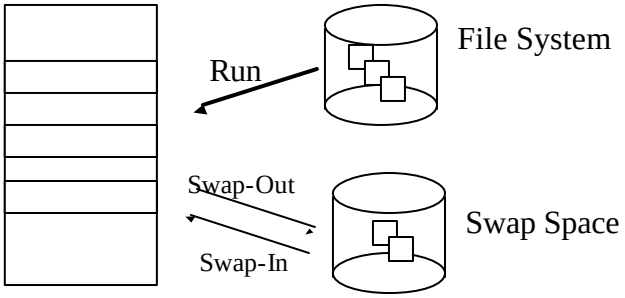
UNIX

- Introduction
- Programmer Interface
- User Interface
- Process Management
- Memory Management
- File System
- I/O System
- Interprocess Communication

50

Memory Management

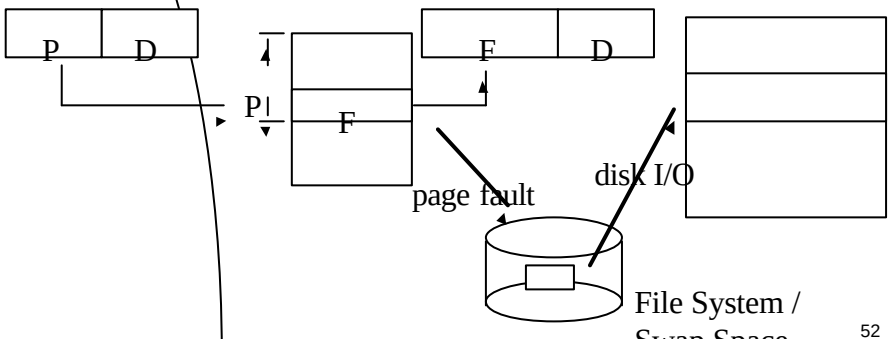
- Virtual Memory – Demand paging



51

Memory Management

- Demand Paging
  - Page fault -> disk I/O -> modify the page table -> rerun the instruction!



52

Memory Management

- 4.3BSD
  - Reasons for page fault
    - Initialized data and text
      - Corresponding page-table entries are marked as fetch-on-demand (fod)
    - Un-initialized data
      - Corresponding page-table entries are marked as invalid -> zero-filled

53

Memory Management

- Page Replacement
  - When the number of free pages is under a threshold, some pages are paged out to release space so that the needed pages can be moved in from disks.
  - Done by pageout() – system process pagedaemon is waked up when the number of free memory is less than *lotsfree*, e.g., ¼ MM size!
  - Approximate Least-Recently-Used (LRU) policy

54

Memory Management

- Working Set Model
  - Each process in memory should be allocated with at least those pages in the working set to prevent trashing.
- Swapping
  - It is invoked only when paging is unable to keep up with memory needs.
  - pagedaemon – waked up when free memory is under *minfree* ,e.g., 1/16 MM size.

55

Memory Management

- 3BSD
  - Virtual memory – demand paging
  - Page replacement – approx. LRU
  - Pre-allocated swap area
- 4.1BSD
  - Logical page – cluster
  - pre-paging
  - Caching recently used text pages
- 4.3BSD
  - Text images and page tables retained in cache after exit

56

UNIX

- Introduction
- Programmer Interface
- User Interface
- Process Management
- Memory Management
- File System
- I/O System
- Interprocess Communication

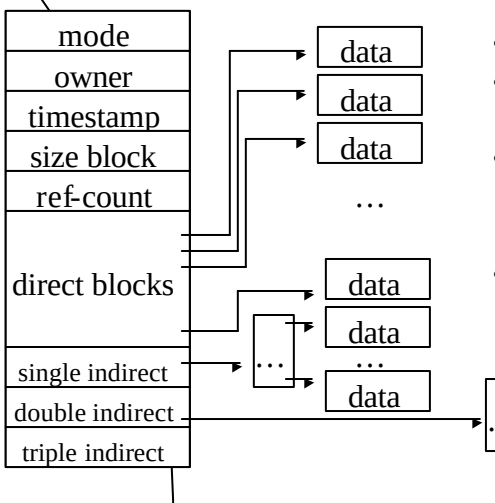
57

UNIX File System

- Main Objects
  - Files and Directories
- Data Blocks
  - Physical Blocks
    - Sector – 512bytes
  - Logical Blocks
    - 4.1BSD – 1KB
    - 4.2BSD – block size, e.g., 4KB, and fragment size, e.g., 512B, initialized during file-system creation.

58

UNIX File System – i-node



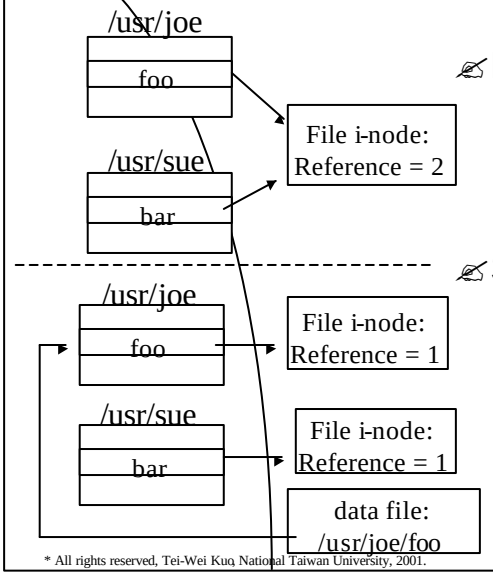
- 4KB block size
- 12 direct pointers
  - 48KB
- 1 single indirect
  - 4-byte block ptr
  - 1K * 4KB = 4MB
- >> 4GB for the largest file! (offset = 32 bits, 4G=2^{32})

* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 380.

UNIX File System – Directories

- Directory
  - A special file distinguished by its i-node type.
  - Since 4.2BSD, file names can be up to 255bytes – a variable length for each directory entry.
  - Linear search of empty directory entry!
  - Given a path name, each directory file is opened and searched for the next node in the path until the desired i-node is returned, illegal access is found, or error occurs.

Sharing of Files - Revisiting



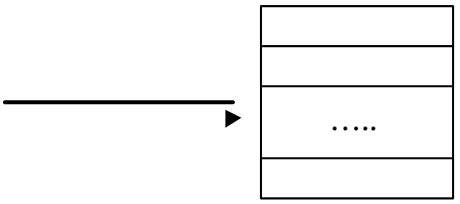
- Hard Link
  - Each directory entry creates a hard link of a filename to the i-node that describes the file's contents.
- Symbolic Link (Soft Link)
  - It is implemented as a file that contains a pathname.
  - Example: Shortcut on Windows

* Problem – infinite loop in tracing a path name with symbolic links – 4.3BSD, no 8 passings of soft links
* Dangling pointers

61

File Systems Revisiting

File: a sequence of bytes

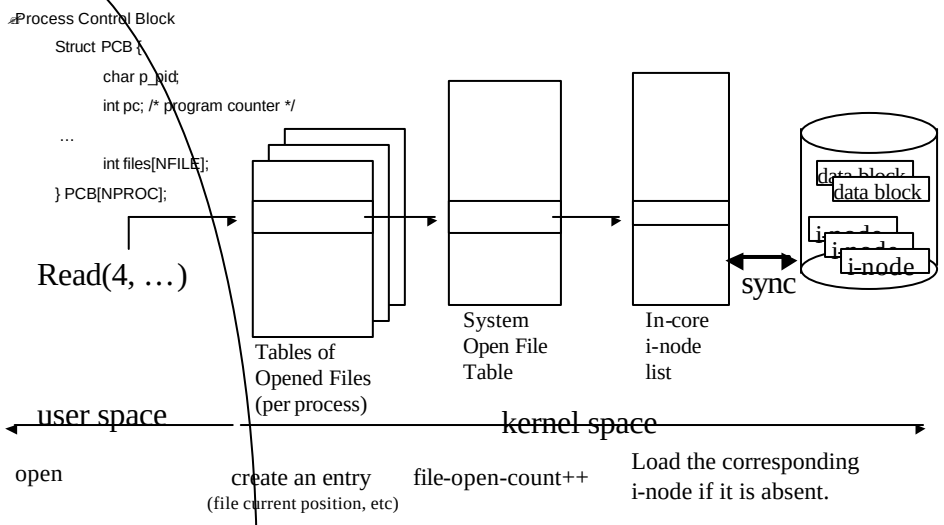


a sequence of (logical) blocks

- File Methods
  - Sequential Access
    - Basic Operations
      - READ, WRITE + file pointers
  - Direct Access
    - Basic Operations
      - READ N or Write N, where N is the relative block number.

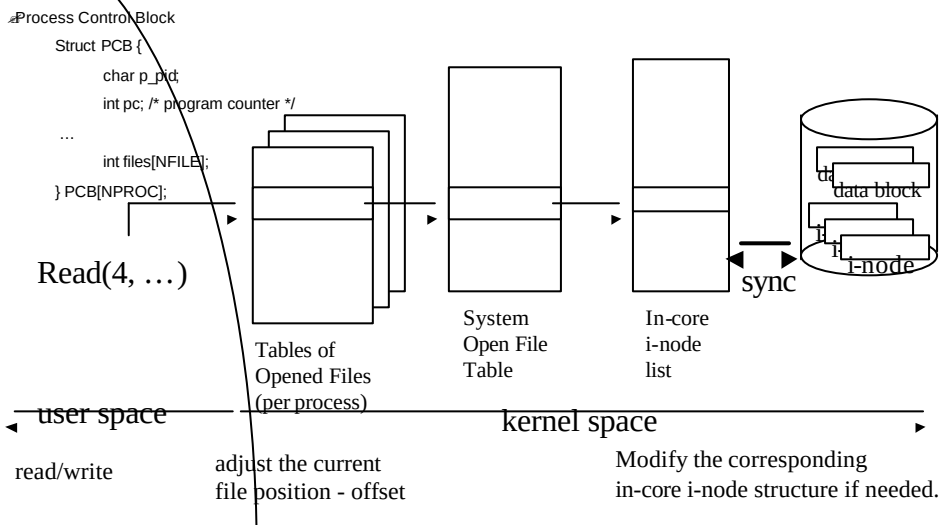
62

UNIX File System



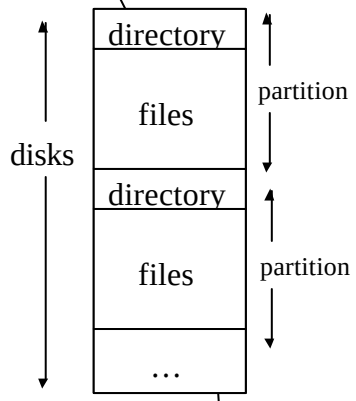
* The i-node structure of a file includes info regarding the disk location of the file.

UNIX File System



* When the file is closed, and file-open-count = 0, then the disk-resident i-node should be modified.

File System – Directory Structure

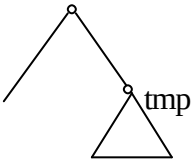


- Partition (/Volume):
  - a low level structure in which files and directories reside.
- Directory:
  - Records info for "all" files on a partition.

* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 349,354-358.

UNIX File System - Mounting

- (name of the device, mount point)



Use an appropriate device driver to read the device directory and verify the format => mount!

- Mount point: the location within the file structure at which to attach the file system.
- A bit in the i-node indicates that whether a file system is mounted on it! -> find the i-node of the root of the mounted file system.

UNIX: manual mounting

UNIX File System - Mounting

- When a mounting-point node in a path name is reached, the mounting table is searched.
  - (major number, minor number)
    - major number identifies the right driver.
    - minor number identifies the device
- Boot block
  - First sector of a file system – primary bootstrap program
  - Call a secondary bootstrap program residing in the next 7.5K.

67

UNIX File System

- 4.3BSD
  - Cylinder Group
    - Goal: Localization of disk movements
    - Several consecutive cylinders
      - A superblock
        - Size of the group, block/fragment size, etc.
      - An array of inodes
      - Data blocks

68

TCP/IP for Linux

郭大維教授
ktw@csie.ntu.edu.tw
國立台灣大學 資訊工程系

69

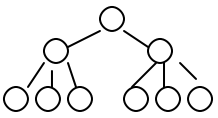
Network Topology

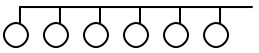
- Why distributed systems?
  - Resource and information sharing
  - Reliability
- Issues:
  - Basic Cost
  - Communication Time and Predictability
  - Reliability

70

Network Topology

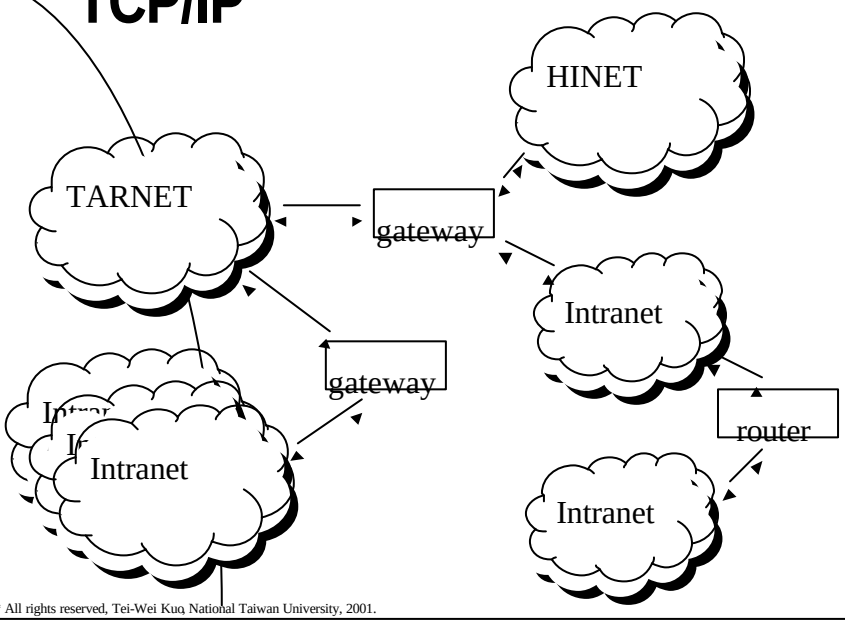
- Topologies
  - Fully Conncted Networks
    - Switches
  - Hierarchical Networks
    - GSM
  - Ring Networks
    - FDDI
  - Multiaccess Bus Networks
    - Ethernet





71

TCP/IP



72

Network Contention

- CSMA/CD (carrier sense with multiple access / collision detection)
  - Listen before transmission
  - Try again after some random time
  - Example – Ethernet (IEEE802.3)
- Token Passing
  - Transmit msgs when the token arrives.
  - FDDI
- Msg Slots
  - Circulate slots
  - Cambridge Digital Communication Ring

73

ISO Protocol Layers

- Application layer
  - Interacting with users, e.g., remote login
- Presentation layer
  - Resolve formats, etc.
- Session layer
  - Maintain session
- Transport layer
  - In-order msg transfer, etc, e.g., TCP
- Network layer
  - Routing packages, etc.
- Data-link layer
  - Framing, error detection, etc.
- Physical layer
  - Electrical details of the transmission of the bit streams.

74

Internet

- Origin:
  - ARPANET (1970's - 1988)
  - NSF NET (no longer exist?) and Internet.
- World Wide Web (WWW)
  - Utilize TCP/IP over ARPANET/Internet.
  - Linux adopts Unix 4.3BSD sockets and supports TCP/IP.

• Definition of "Intranet": roughly speaking for any network under one authorization, e.g., a company or a school.
  • Often in a Local Area Network (LAN), or connected LAN's.
  • Having one (or several) gateway with the outside world.
  • In general, it has a higher bandwidth because of a LAN.

75

TCP/IP

- IP Address:
  - 140.123.101.1
    - 256*256*256*256 combinations
      - 140.123 -> Network Address
      - 101.1 -> Host Address
    - Subnet:
      - 140.123.101 and 140.123.102
  - Mapping of IP addresses and host names
    - Static assignments: /etc/hosts
    - Dynamic acquisition: DNS (Domain Name Server)
      - /etc/resolv.confg
    - If /etc/hosts is out-of-date, re-check it up with DNS!
  - Domain name: cs.ccu.edu.tw as a domain name for 140.123.100, 140.123. 101, and 140.123.103

76

Name Resolution in TCP/IP Network

- Name Resolution
  - A hierarchical host name
    - bob.csie.ntu.edu.tw
  - A 32-bit Internet Number (host id)
    - 140.112.101.32
- How it works?
  - The sending system checks its routing table to locate a router. The routers use the network part of the host-id to transfer the packet to the destination network.

77

Name Resolution in TCP/IP Network

- Mapping of Ethernet (IEEE 802.3) physical addresses and IP addresses
  - Each Ethernet card has a built-in Ethernet physical address, e.g., 08-01-2b-00-50-A6.
  - Ethernet cards only recognize frames with their physical addresses.
  - Linux uses ARP (Address Resolution Protocol) to know and maintain the mapping.
    - Periodically broadcast requests over Ethernet for IP address resolution over ARP.
      - A UDP packet with the host-id and Ethernet address
    - Machines with the indicated IP addresses reply with their Ethernet physical addresses.

78

Name Resolution in TCP/IP Network

- Within a network,
  - Each host caches the info in its ARP cache with aging.
  - When a process specifies a host to communicate.
    - The kernel determine the host's in using domain name server (DNS) lookup.
    - Pass all layers with Ethernet address in the packet.
    - The Host receives the packets and pass all layers.

79

Name Resolution over Internet

- Domain Name Service defines the structure of the host names and name-to-address resolution!
  - A process on a host A communicates with bob.cs.utexas.edu:
    1. The kernel of A issues a req to the name server of the "edu" domain to ask for the name server's host address for "utexas.edu".
    2. The kernel of A issues a req to the name server of the "utexas.edu" domain to ask for the name server's host address for "cs.utexas.edu".
    3. The kernel of A issues a req to the name server of the "cs.utexas.edu" domain to ask for the name server's host address for "bob.cs.utexas.edu".

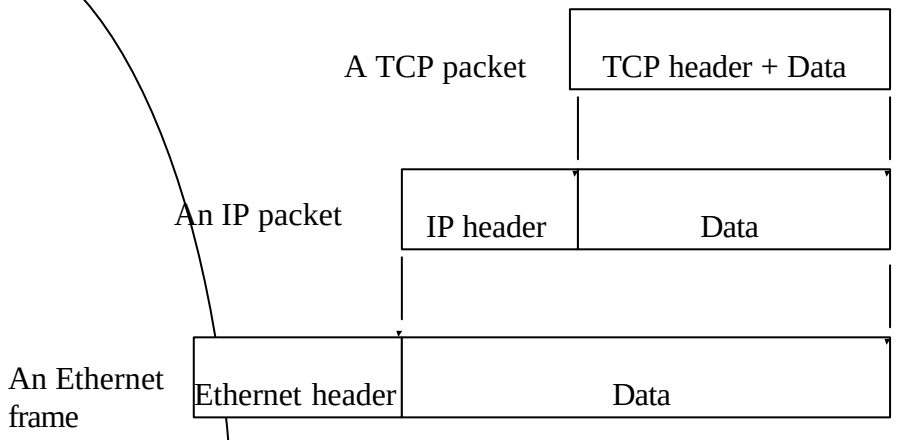
80

Name Resolution over Internet

- How to reduce the inefficiency of the protocol?
  - Use local cache!
- How to resolve the crash problem of name servers?
  - A duplicate!
- Name resolution
  - Autonomous at all sites!
  - The kernel at the destination host is responsible to passing the info to the right process!

81

TCP/IP



- Each IP packet has an indicator of which protocol used, e.g., TCP or UDP

82

Routing

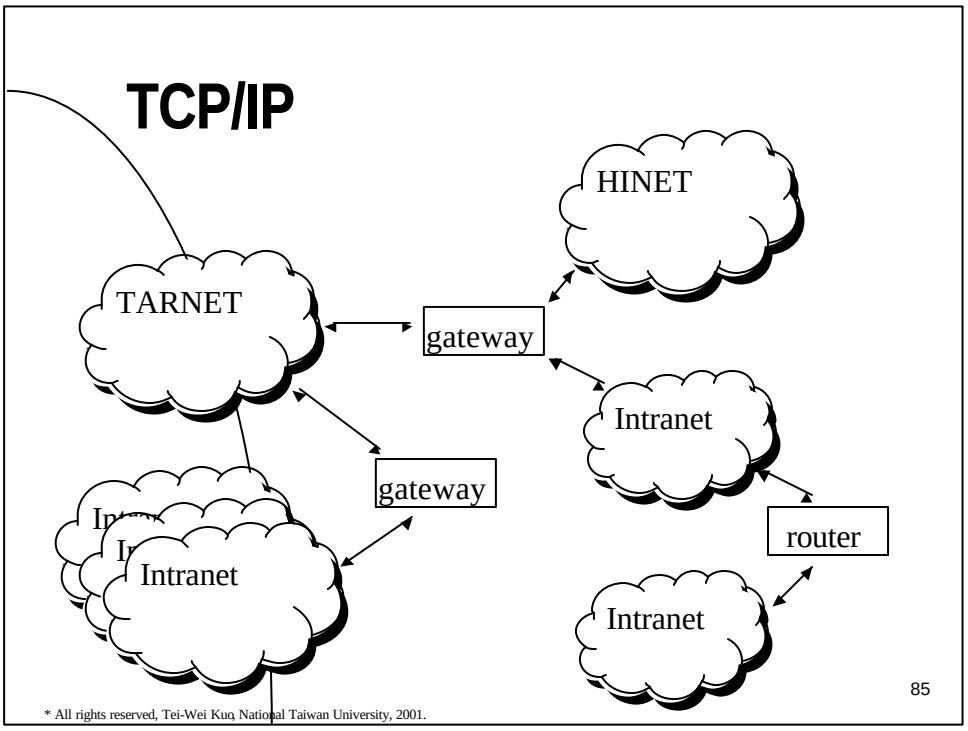
- Def:
  - How are msgs sent through network?
- Common routing scheme
  - Fixed routing
    - A path is specified in advance & does not change!
  - Virtual routing
    - A path is fixed for a session, e.g., a file transfer or remote login!
  - Dynamic routing
    - A path is chosen only when a msg is sent.
    - Wrong sequence order!

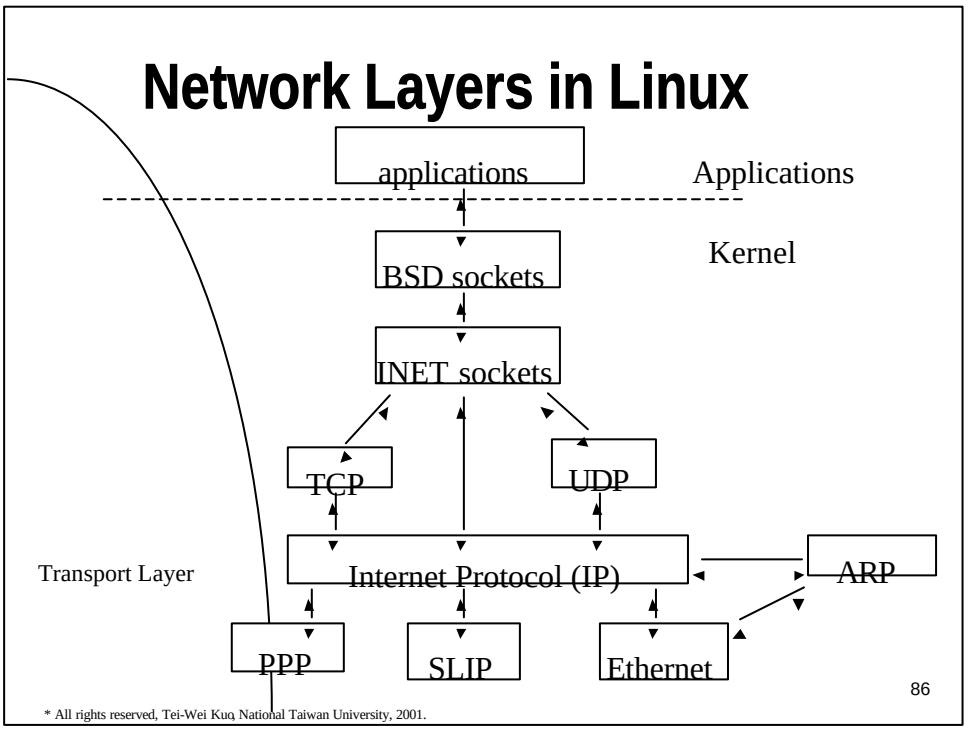
83

TCP/IP

- Router
  - With a Routing table
    - Use some routing protocol, e.g., to maintain network topology by broadcasting.
  - Connecting several subnets (of the same IP- or-higher-layer protocols) for forwarding packets to proper subnets.
- Gateway
  - Functionality containing that of routers.
  - Connecting several subnets (of different or the same networks, e.g., Bitnet and Internet)for forwarding packets to proper subnets.

84


TCP/IP
HINET
TARNET
gateway
Intranet
gateway
router
Intranet
Intranet
85
* All rights reserved, Tei-Wei Kuo, National Taiwan University, 2001.


Network Layers in Linux
applications
Applications
Kernel
BSD sockets
INET sockets
TCP
UDP
Transport Layer
Internet Protocol (IP)
ARP
PPP
SLIP
Ethernet
86
* All rights reserved, Tei-Wei Kuo, National Taiwan University, 2001.

TCP/IP

- Transmission Control Protocol (TCP)
  - Reliable point-to-point packet transmissions.
  - Applications which communicate over TCP/IP with each another must provide IP addresses and port numbers.
    - /etc/services
      - Port# 80 for a web server.
- User Datagram Protocol (UDP)
  - Unreliable point-to-point services.
- Both are over IP.

87