INTERNET SECURITY - 3

Ayush Sondhiya
- CSE/OT

LEC-1

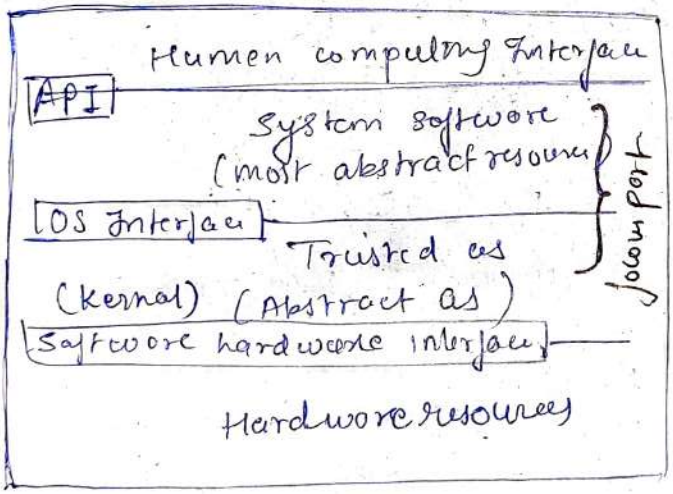
SYSTEM SOFTWARE

OS

Independent of an individual application but common to all of them

Ex = → C Binary function
→ A window system
→ DBMS
→ Resource management function
→ OS

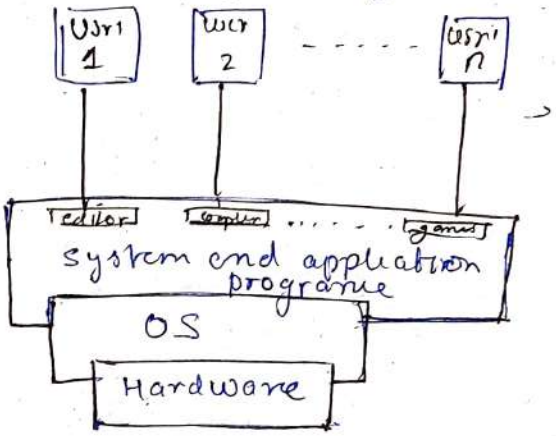
One can say that OS is agent a system software but lets see how misleading it can be although statement is true



* Components of Computer

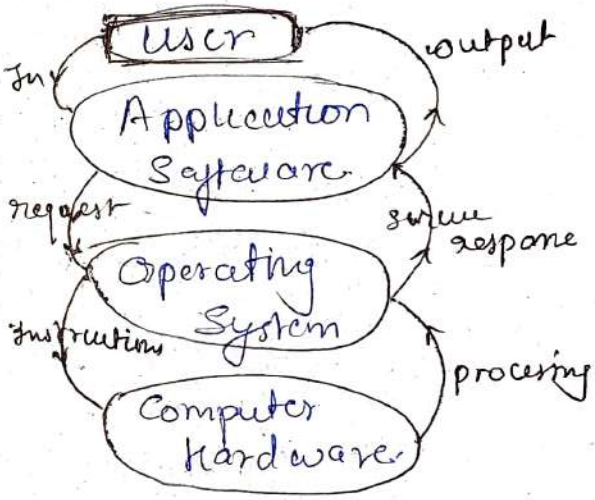
- Hardware - basic computing resources
  - CPU, Memory, IO
- OS
  - control & cordinator use of hardware among various applications and user
  - → Application programs define ways in which programs use's system's resources

- word processor, compiler, Database systems, games etc



* Components in action

An operating system is a crucial link to put the all computer in action



Different type of OS are there How your end-Application is

* Types of OS

- Single user OS (single task) designed to manage system of only one user can do one thing at a time (palm os)
- Single user multi task allow one user to perform more than one task at a time

Multi-user: it allows many user to take advantage of computer resources.
eg. UNIX / LINUX servers

Real time OS: an os That manage resources of computer such that a particular operation executes precisely at the same amount of time every time it's given

Distributed OS: manage group of individual computer and makes them looks as one to the system software

Embedded OS: phone, smart watch, etc

* Roles of OS

- Resource Allocator
- controller of programme.

Defination & Gates

- it's Dont have any universally accepted def.
- Kernal - heart of os (cannot be called as OS)

How Dose OS start

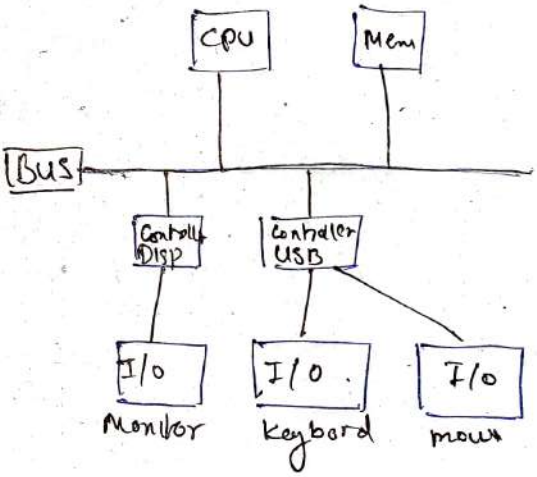
- a boot strap program is loaded
  - Stored in ROM or EPROM known as firmware
  - initiates all aspects of system
  - loads OS kernals & start execution

* COA / CSO

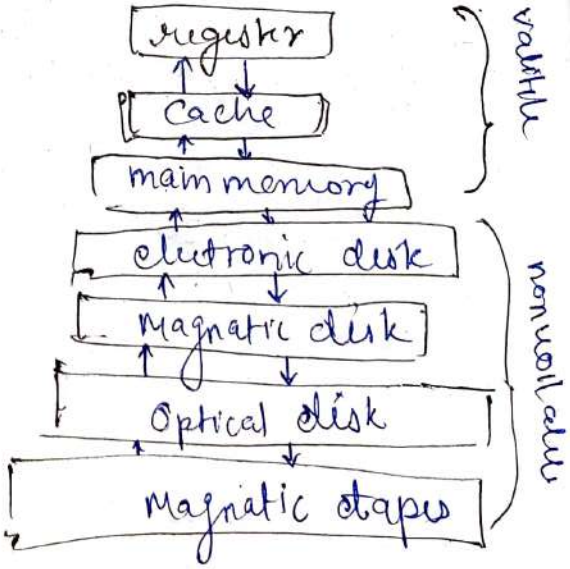
- one or more CPUs device controllers connects through common bus
- IO Device and CPU can execute concurrently

→ IO Device sends and interrupt to CPU and asks for operation after way

DMA (Direct Memory Acess) it will bypass the CPU it will take to memory



* Storage



Hirarchy
→ Speed
→ cost
→ Volatility

Caching :- copying info into faster acess times

* I/O In a computer

→ Syschronous I/O
- after IO starts, controls returns to user to program only upon I/o completion
- wait instructions Idels
- wait loop
- at most 1 I/o at a time.

→ Asyschronous I/o
after I/O starts control return to usr without waiting for I/O completion
- System call — allows usr to wait for I/O completion
- Devie Status tables for each I/o type address state

* Interrupt (main link)

it's a syscronised signal. indicates need of attention or a syschronous event in software indication need for change in execution
- an interrupt is a request to cpu for a service
→ can be genarid by hardware
→ can be generated by software

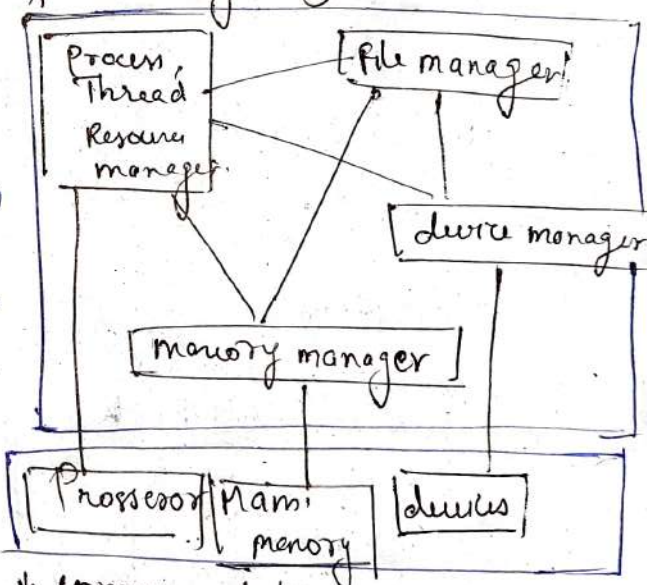
* Trap ?

An exception is a request for servic from cpu.
— by hardware - sytem bus (interrupt)
— by software - system call for IO
Trap for divid by 0
all OS are Interrupts driven

* Some basic difinations

- Process. Sequential program in execution
Resource - thing process can request
File special case of resource (Byte stream)

* OS organization.



* component / Service of OS

- Process management
- main memory management
- File management
- I/O management
- Secondary management
- Networking
- Protocol System
- Command - interpreter system

LEC-3

* Process management

- A process is a program in execution
  - → eg: Batch Job (super computer)
    : system task.
    : time shared user program
- A process needs certain resources including CPU time memory files and I/O devices
- An OS is responsible for following activities in a connection with management
  - → process creation & deletion
  - → Suspention & resumption mechanism for.
    : Process sync.
    : Process communication
- A Process (active) is a
- programme (passive) in execution

  Process needs resources to accomplish the task
  - CPU, memory, I/O

  process termination requires reclam of usable resources
- Single threaded process has one program counter for specifing location of next instruction of to execute
- multi threaded process has one PC per thread.
  - many process run concurrently by MUX CPU

Process management responsible for
- create & delete both user and system process
- Suspending & resuming the process
- providing mech for process sync
- — " — process communication
- — " — deadlock handling

* Memory Management

- its large array of words bytes each with its own address
- repository for quickly accessing data shared by CPU and I/O
  - its main memory (volatile)
  - loose contents in sys failure
- The OS is responsible from
  - Keep track of memory is used by whom
  - decide which process loads when memory available
  - allocate & de-allocate spaces needed

* file management

- most visible an component of
- it can be stored on different disks / systems

for a convinet use of computer system OS provides a uniform local logical view of storage
- → creating & deleting file
- → — " — directory
- mapping file on secondary storage

OS responsible for →

- Free space management
- Storage allocation
- Disk scheduling

* I/O System management →

I/O system consist of

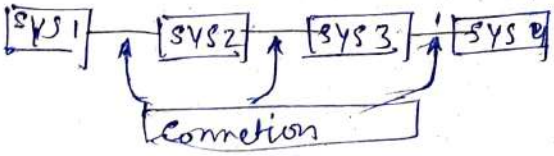
- Buffer caching system
- device driver interface
- Drive for specific hardware

Buffer (storing data temporarily)
Cache (Storing part of data in faster performance)
Spooling (overlapping output of one job)

- generic device driver Interface

* Networking (Distributed System)

- A distributed system is a collection processor that don't share memory or clock
- each processor has local own memory
- computational speed up



LEC-4

* How Does OS protect itself

Dual mode operation allows OS to protect itself and other components

- OS must not get corrupted

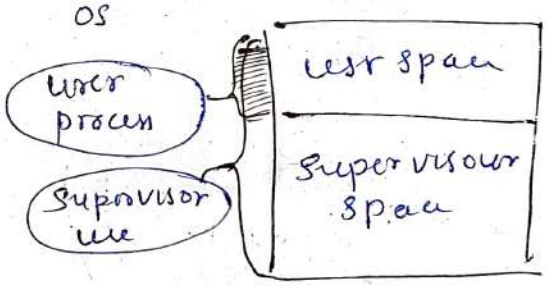
USER mode & KERNAL mode

- mode bit provided by hardware

Supervisor Kernal mode	user mode
• execute all machine instruction	• execute subset of the instructions.
• can reference all memory locations	• can reference subset of memory locations.

* Kernal and Memory →

- part of os is critical to correct operation
- execute in supervisor mode
- Trap instruction is used to switch user to enter the OS



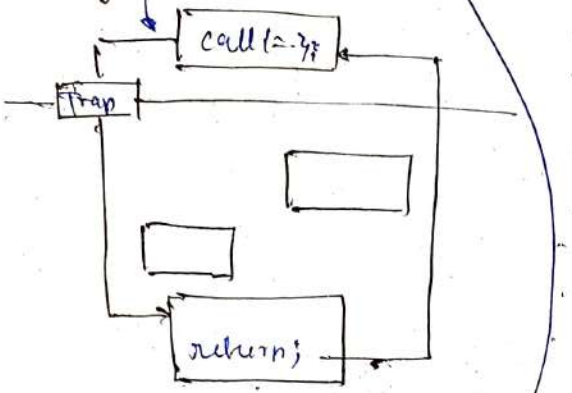
* Transition from user to Kernal

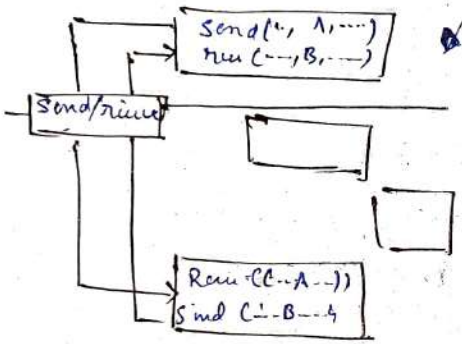
Timer to prevent infinite loop or hogging resource
- set interrupt after some time
(Round Robin scheduling)

* How do we request Services

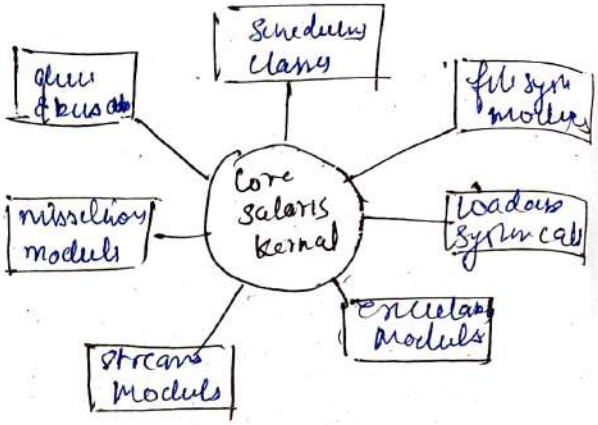
- system call
  - Interrupts calls
  - process traps to as interrupt handler
- message passing

* System call & API

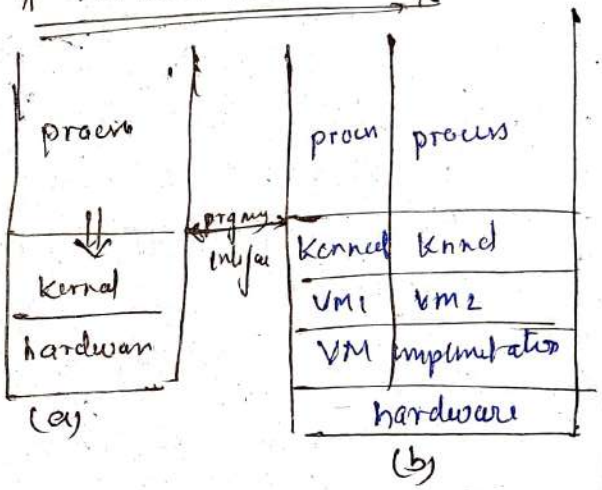




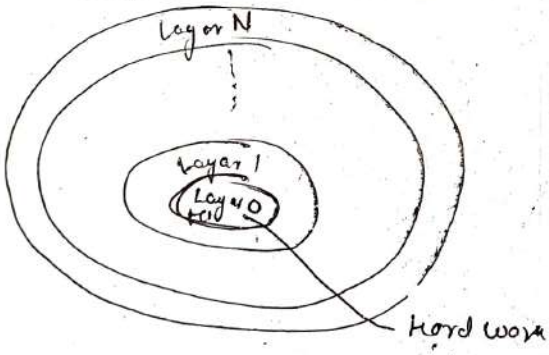
* Modular OS structures



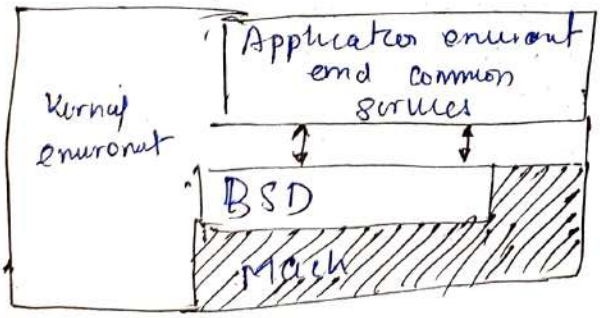
* Virtual machine



* Layered OS structure



* Micro kernal OS structure



* Process VS threads LEC-5

Similarities—

- Like a processor share CPU only one thread active at a time
- Like process threads within a processor executes. thread within process executes subsequently.
- and like process if one thread is blocked, another thread is run..

Differences—

- unlike process threads are not independent of one another.
- all threads can access every address in task
- thread are designed to assist one another

* Adv of Threads

- context switching
  - Threads are in expensive to create and destroy they are in expensive to represent
    - → if they require space to store the PC SP general purpose registers but don't require space to share memory
  - with little context it is so much faster to switch b/w threads.
- Sharing resources is also very easy compared to in process

Dis Adv

Blocking - kernal is single threaded. System call of one of the thread will block whole process and CPU may be idle during that process

Security - due to extensive sharing

** Kernal (during context switching) (PCB level)

Before a process can be switched its process control block (PCB) must be saved by the OS

- The PCB consists of following info
  - process state, PC, value of different registers, scheduling
  - memory accounting info
  - status info of process

* Thread library

- Thread library provides programmer with API for creating threads & managing them
- Two ways of Implementing
  - Library entirely in user space
  - Kernal level library supported by OS

P-Threads

- may be provides either as user level or kernal level
- A POSIX standard (IEEE 1003) api for thread creation and synchronization
- API specifies behavior of Thread library, implementation of library
- common in unix system (Solaris, Linux etc)

Thread cancellation

Terminate a thread before it has finished.

- Two approaches
  - → Asynchronous Immidiatly Terminate threads
  - → Synchronous allow target thread to periodically check if it should be cancelled.

Thread Pool

- Lot of threads are already created and waiting for work

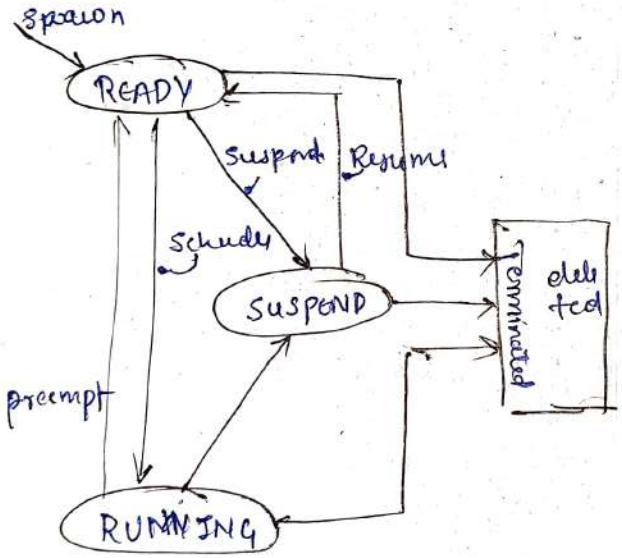
Thread specific data

allow each thread to have its own copy of data

Thread state

Primary - Running
Ready
Blocked

Operations
Spawn - creat a thread provide register content and stack
Block - event wait
Save user registers PC & stack pointer
Unblock - move to ready state
Finish deallocate register content & stacks



LEC -6

* Procs scheduling

- how OS assigns physical processor to processes
- allows processor to accomplish the work
- Scheduler - looks at que and find out which need to schedule
- dispatcher - put it into process

general rules of scheduling
- keeps CPU busy
- major source of I/O
- many program have characteristics
- many programe have characteristics CPU-I/O burst cycle alternately
phase of cpu activity and I/O inactivity
- cpu bound programs have few long CPU burst
- I/O bound have more shorter I/O burst

scheduling mechanisms

Long term : Job scheduler
(≥ 10 min)
medium term :
Real time
Short term : invoke in milliseconds
CPU scheduler
Select from among the proc that are ready to execute allocate the CPU to one of them
- Switch from Running to waiting state
- Switch from running to ready state
- Switches from waiting to ready
- Terminate

Preemtive - we can't interrupts process
Non Premeptive we can interrupts process

* Dispatcher

- Dispatcher is an imp. component involved in CPU scheduling
- the OS code that takes away ~~from~~ CPU from current process and hand it over to newly scheduled process known as dispatcher
- Dispatcher gives control of CPU to process selected by short term scheduler

it performs —
→ Switch context
→ Switch to user mode
→ jumping to proper location

How does process relenquish
- first two when a program can relinquish

lev-1 cooperative scheduling
- process goes from running to waiting state
- process terminates

lev-2 preemptive scheduling
- Interrupt cause a process to go from waiting to ready state
- key word "Interrupt"

Waiting time —
non preemptive = start time - arrival time
Preemptive = finish time - Arrival - CPU Time

Turnaround time
Finish time - Arrival time
amount of time to execute a particular process.

Criteria for optimization
Max CPU
Max throughput
Min turnaround
Min waiting time
Min response time

LEC-7

* SCHEDULING ALGO

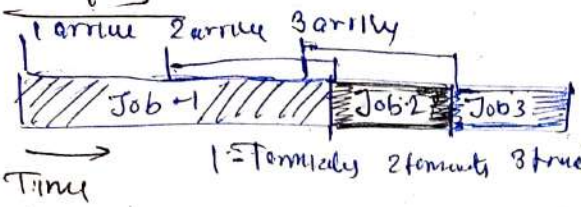
major jobs
- multiprogramming to improve the responsiveness
- Improve hardware utilization
- app. use / system computation
- with scheduling algo both responsiveness & improve hardware

Response time avg time submitting an job until it terminates

ALGO's

- First come first serve (FCFS)
- Shortest job first (SJF)
- Shortest remaining time (SRT)
- non preemptive priority scheduling
- preemptive priority scheduling
- Round-Robin scheduling
- Highest response ratio (HRRN)
- multi level feedback queue
- multilevel queue scheduling

FCFS



- Job 2 is ahead of job 3. So when job 1 terminates job 2 starts so Job 3 must wait till the termination of job-2
- Reason for multiprogramming is for user to wait less
- Jobs are serviced in order they arrive in FCFS scheme

→ Non preemtive
— CPU first is allowed the CPU first
— Implemented FIFO queue PCB linked on tail of queue

Limitation
- FCFS favours long process compared to short ones
- Avg waiting time is quite long
- order significantly affects avg waiting time.

IDEAL CASE
- when there are k jobs in the systems they all run simultaneously but at rate of 1/k
- short jobs dont have to wait for job 2.

SJF and SRTF
when CPU is available it assigns to the process that smallest next CPU burst
- if two process have same length of next CPU burst FCFS scheduling used to break the tie

Shortest Job first (SJF)
— Non preemtive

Shortest remaing time first (SRTF)
— Preemtive

LEC – 8

* Process Synchronization
multiple resource compete for a common resources
- at a time they may cooprate to sync some activity or share a common resources
- concurrent access to shared data may result in data inconsistency if no controlled access
- maintaing resource regularity ensures orderly execution of process.

— this process can be affected or affect other process

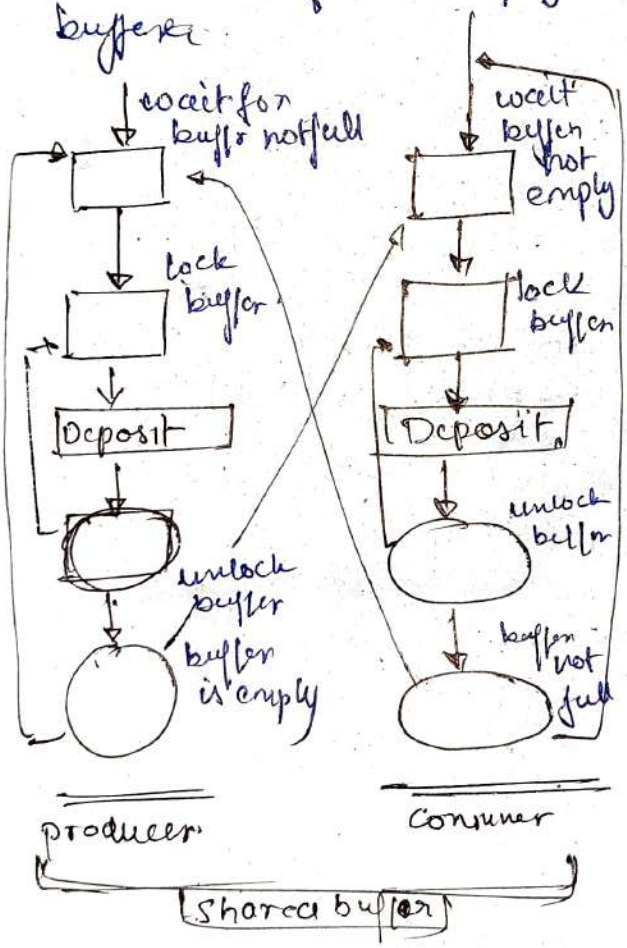
when process cooprate →
- assumes we have one process that produces info and other consumes info and share a common fixed size buffer to store this info.

we can have two types of buffers

Unbound places no practical limit on size of buffer

bound assume that has a fixed size

problem is that to make sheere that procdeer dont make excess data into buffer & consumer dont try to extract more data from Empty buffers



Solution

mutal exclution →

- a sysn mech to avoid race conditions by ensuring exclusion execution of critical sections
- This proces is only one process doing certain things at one time

Critical section

a critical code which reads or writes shared data

From OS POV

Utilize semophore or monitor

-) sync counting variable
-) an integer var value that cannot go below 0
- a smaphore comprises -
  - an intiger value
  - two opricalor P() and V()
  - P() decrement value while value == 0 sleep)
  - V() incriment value if any thread waiting for value to become non zero wake at least 1 thread.

P == ? lock acuire

V == ? Lock Release

Critical Section

1) entray section
2) critical section
3) exit section
4) remainder section

* Reader Writers problem

dataset shared among hemb of ~~concurrent programmers~~. Concurrent proc

Reader : only reads the data

Writer : can both read & write

Problem: allows multiple readers to read at same time ; only writer can access shared data at the same time

Shared data
- Data set
- Semaphore mutex initialized to 1
- Semaphore wrt initialized to 1
- Integer readcount initialized to 0

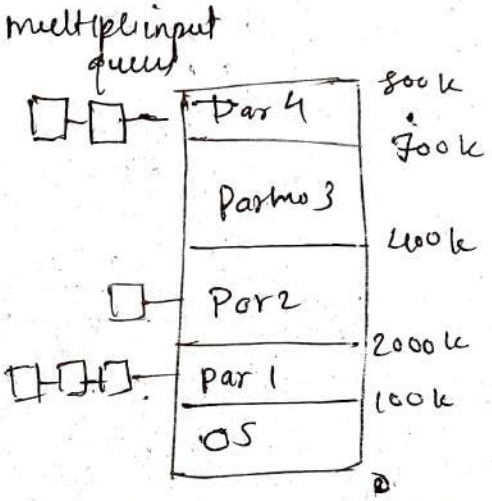
* When will deadlock occur

- resource can only hold at most 1 process
  a process can't deadlock on shareable process (mutual exclusion)
- Hold and wait
  process can hold resources and block for other processes
- no preemption
- circular wait
  when process A has resource which A has on a waiting for which B has.

* Memory Mangement

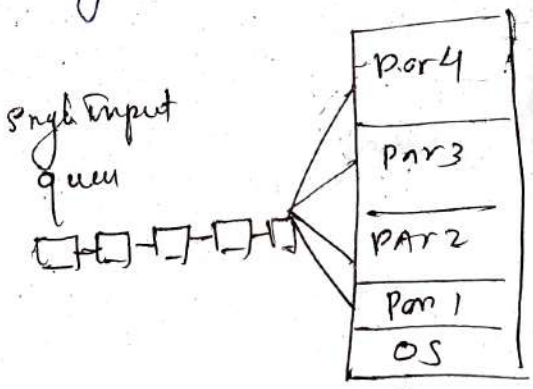
- cache } volatile & fast
- RAM }
- Disks } non volatile & slow

* multiprogramming with fixed partitions (Main memory)



a

fixed memory partition
- Separate input queue for each partition
- Single input queue



b

Allocation : Program need to allocate some space

- swapping, fragmentation compaction
- Programs move from main memory as it terminates

- when it moves it create empty space (holes)
- new process that may needs occupy it
- there is a probability that main systems have too many small holes
- it may need to situation that need process cont fill then
- This means memory is fragmented
- OS - reallocate existing programme in continues region creates large enough free space for allocation to new process [called compaction]
- compacting is done by collecting garbage (holes) after the dynamic program terminates.

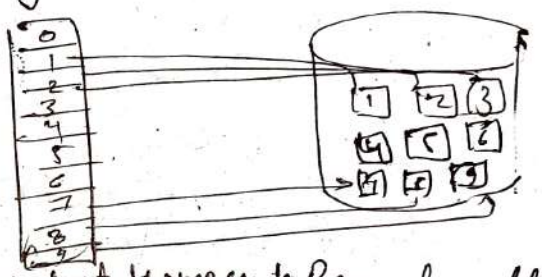
Virtual memory
when processor sees a large logical storage space (a virtual storage space) though actual main memory may not be that large
- I/O support most of block oriented I/O called as spulized files

Summing up
main memory
- large array of words or bytes
- each word byte has own address
- for execution it must be address to exact location

OS must
- keep track of which part of memory currently used by system & process
- allocate & deallocate

Virtual memory meth

Demand paging -



What Happen to Page fault
- Trap to kernel
- find wat to do
- Terminate user (illegal access)
- Swap from disk
- find victim frame to replace with faulting page
- get faulty page content from disk
- update kernal tables
- Return user mod Retry instruction

Page rep
If page replacement algo is slow the sys/ programe will be slow

FIFO
oldest address page

LRU replace least recently used of them

FILE SYSTEM

LEC-11

→ File structure
  - Logical storage unit
  - Collection of related info
→ File system organised into layers
→ File system resides on secondary storage (disk)
→ File control block storage structure consists of info
→ Device driver

Layered file system

application program
⇓
Logical file system
⇓
file organization system
⇓
basic file system
⇓
I/O control
⇓
Devices

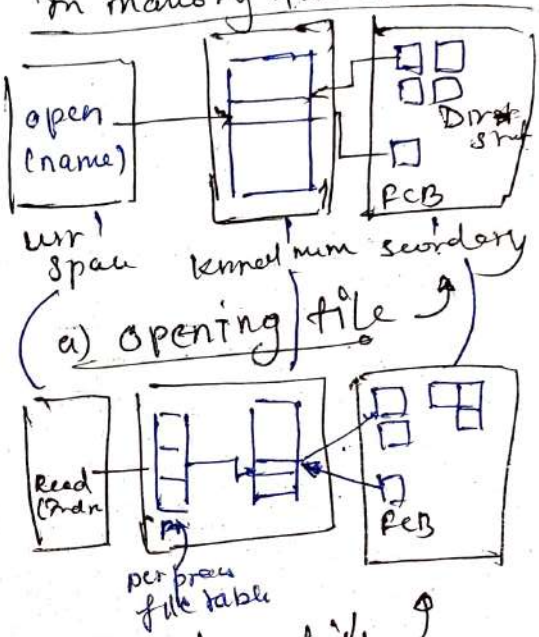
Implementation

- Boot control block
- Volume control block (contains partition/volume)
- per file file control block (FCB) contains details about file

File control block

→ file permissions
→ file dates
→ file owners
→ file size
→ file data block/pointers

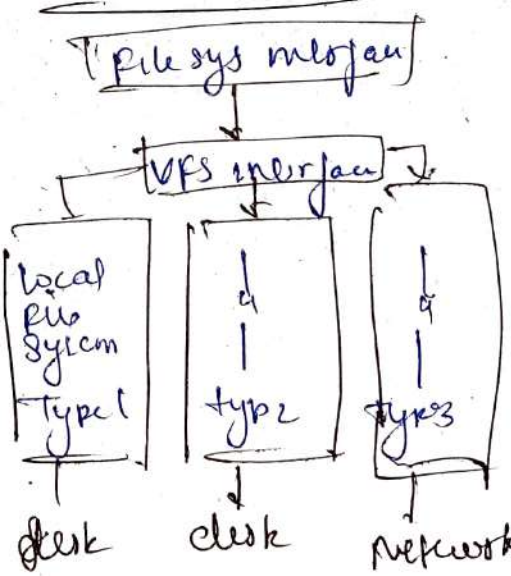
In memory file structure



Virtual file system

- VFS provide object oriented way of implmt file sys
- VFS allows API for used for diff or system
- API is to VFS interface rather than any file sys

Schematic view



Directory implimitation

- Linear list – scan 1 by 1
  - slow
- Hashtable – linear with hash data structure
  - faster
  - collision : where two file name has same location
  - fixed size

LEC 12

Allocation methods

- contagious
- linked
- Indexed.

contagious –

- each file occupies contagious blocks on disk
- simply
- wasteful space
- Random access
- file cant grow

~~Linked~~ –
mapping logical to physical

LA 512 → Q
R

Block to be accessed = Q + base address

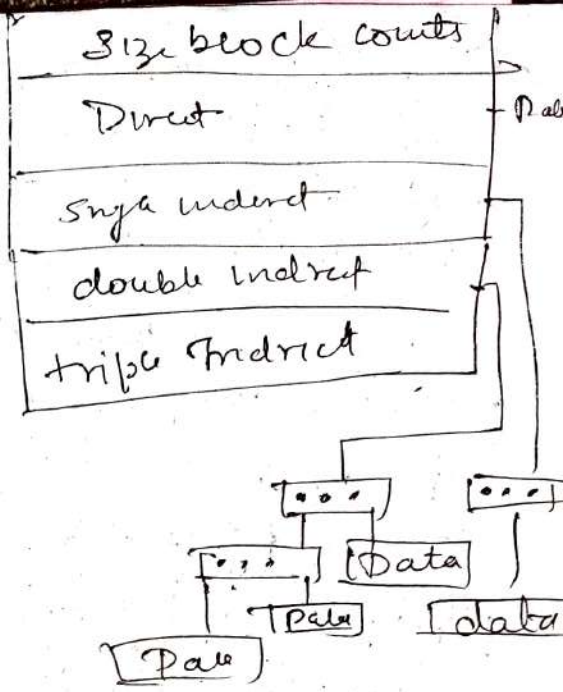
→ Displacing into block R.

Linked –

- file can grow
- complex
- ~~vulnerable~~ vulnerable

Indexed –

- file can grow
- depends on control node
- complex



UNIX UFS (4K byte sys)

mode
owners (2)
time stamps (3)

* UNIX - File system [LEC-13]

Every thing is a file

- Regular files
- Directories: (folders) or file for list of files
- Symbolic link (shortcut)
- Device and peripherals
- Pipes use to cascade

cat * log | grep error

[output of 1st = input of 2nd]

- Socket: Inter process communication

File names →

- case sensitive
- no obvious length limits
- can contain any of characters, except (/)
- file name extension are not mandate

etc README .bashrc index.htm Windows but not

* LINUX File System Structure

/ - Root directory

/bin/ - basic essential system commands

/boot/ - kernal image

/dev/ - Files represent devices

/home/ - User directory

/lib/ - System shared library

/lost+found - Corrupt files are placed

/media - Removable media

/mnt/ - mount point for temporary mounted

/opt/ - Specific tools installed by the system

/proc/ - access to the system info

/root/ - root user home directory

/sbin/ - admin commands

/sys/ - system & device controls (eg clock etc)

/tmp/ - temporary files

/usr/ - user based tools.

/usr/local/ - specific software installed by sysadmin

/var/ - data used by sys or system servers.

* File path

path is a sequence of the nested directories with a file or directory in the end.

- Relative path
  - documents/fun/ms.html.
- Absoulute path.
  /home/bills/bugs/crash.1101

⇒ /: root directory
Start of absoulute path.

TLEC-14

* Shell & File Handeling

Command line Interpretor

- Shell: tools to execute user commands
- called shell because it hide details on underlying operations
- commands are input in text terminal
- Results are also displayed on terminal.
- Shells can be scripted.

well known shells

- most famous & popular shells
  - → sh: the bourne shell (obsolete)
  - → csh: (the c shell)
  - → tcsh: the TC shell (still very popular)
  - → bash: bourne again shell (most popular.)

x ls command

ls -a (all) list all the files (including * file)

ls -l (long) type, data size, owner.)

ls -t (time) list most recent file

ls -s (~~small~~ (size) test the biggest files

ls -r (reverse) reverse the sort order

ls -ltr. long lasting most recent files at end

file name pattern substitution

ex—

- ls *.txt : the shell replaces *.txt by all file directory names ending by .txt (including .txt) except those starting with, and then the ls command line.
- ls -d .* list all the files and directories -d tells ls not to display content of director.
- cat ?.log Display all the file which name start by and by .log
- ./ this current directory same = /readme.txt & readme.txt
- ../ - parent of current directory usage - cd
- ~/ = home directory of current user.
- ~sydy/ = want to go in home directory of particular user (sydy user)

* File basic Command

- cd <dir> = change directory command
- cd - previous current directory
- pwd - Display current directory

cp command

- cp <source file> <target file> copies source file to target
- cp file1 file2 file3 ..... dir copies files to target directory
- cp -i (interactive) asks for user confirmation
- cp -r <source_dir> <target_dir> copies whole directory

mv and rm commands

- mv <old filename> <new file name> (move - rename the file
- rm file1 file2 ..... (remove) remove given file
  rm -(interactive) (recursively)
  rm - dir1 dir2 .....
- Mkdir dir1 dir2 (make dir)
- Rmdir dir1 dir2 (remove dir) only works when directory is empty

display contents

- cat
- more (page by page display
- less (searching or display lines) or specific thing by (?) command

Head & tail cmd

- head [-n] <files>
  first n lines of some dir
- tail [-n] <file>
  last n line of some dir
- tail [-f] <file>
  display last 10 lines by default of given file

Grep cmd

grep <pattern> <files>
Scan the given files

- grep error*.log
  display all files contains erros in .log files
- grep -i
  Same but case sensitive
- grep -ri error
  Same but recursive
- grep -r info*.log
  output all lines plus in in subdirectories excpt those containg info.

Sort cmd

- Sort <file>
  Sort in character order
- Sort -r : in reverse order
- Sort -ru : output identical lines once.

Sed cmd

Sed is Stream EDitor.

- Sed- e's/abc/def/' test file
  abc replaces with def in test file
- sed 's/^[\t]*//' test file
  will remove any vaid space or tab will be remove

Regular expression [Lec-16]

. match with any character
[] with any character listed inside the brackets
[^] any character not listed inside the brackets
^ match the begenning of the line
$ match with end of line
* match previous element 0 or more times
+ matches previous element 1 or more time
? matches previous element 0 or 1 time

Symbolic link

its a special file which is just a refrence to the name of another one (file or directory)

- use ful to reduce disk usage and complete when two files have same info

eg.
anaken_skywalker → deathvader

Creating Symbolic link

- ln -s file_name link_name
- create a link to a file with another directory with same name

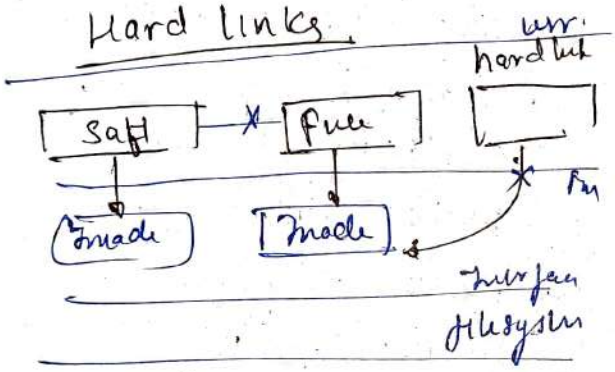
  ln -s../Readme.txt
- create multiple links at once.

  ln -s file1 file2 file3
- To remove a link

  rm link_rm.

  (don't remove linked file)

Hard links



- help
  - help to query command available in unix.

manual pages

man <keyword>

display one or several manual

Info pages

info <command>

gives additional info

File access write

- Read access (r)
- Write access (w)
- Execute rights (x)

3 access levels

→ user (u) owner of file

→ group (g) - each file also have group

→ others (o) for all users.

Access rights constraints

- x is sufficient to execute binary
- both x and r required for shell script
- r to list contents
  x to access contents
- you can't rename, remove copy files if you don't have w access.

Chmode changing permission

chmod <permission> <files>

2 formats for permission

- octact format (base)

  a, b, c = r*4 + w*2 + x

  (w r x booleans)
- or symbolic formats

  chmod go+r - read to group

  chmod u-w: remove write permissions from user

  chmod a-x remove execute permission from all

Chmode -

- chmod -R a+rX
  make everything available to everyone.

File ownership

- chown -R sco /home/linux/src
  (R: recursive) makes user sco new user for all files /home/linux/src
- chgrp -R empm /home/pty
  make empm the new group of /home/pty
- chown -R borg:alien USSS_enterprise/chown
  can be used to change ownership & group at same pom
  using root account.
  if you have root password
  su (switch user
  In modern distribution sudo command gives you access to some root with own user password
  sudo mount /dev/hda4 /home

LEC-18.

LINUX I/O

Redirection pipes

All command output by default sent on your terminal do it by switching writing to standard outputs

- Standard output can be redirected as to file using (>) symbol
- Standard output appending to extended file using the >> symbol

Standard input

Lots of commands which not given input arguments can take input

- sort
  window
  linux
  (ctrl)(D)
  Unix
  windows

Sort < part.txt
Standard input

PIPES (|)

Unix pipes are very useful to redirect Standard output of a command to Standard input of another one.
eg.
cat *.log | grep 'error' | sort
grep 'error' | grep -v 'ignored'
sort -u | > error.log
cat /home/*/homework.txt | grep mark | more

- Most powerful feature in Unix

Standard error

Errormessages are usually output (if programme is well written) to standard error instead of standard output

- it can refer through 2> or 2>>
- Eg

Cat f1 f2 nofile > nufile 2> errfile

Task Control [Lec 19]

Full task controls on tasks

- unix supports true pseudo multitasking
- ability to run tasks in parallel & abort them if they corrupt
- ability to choose which task to run
- ability to chose which input your programme takes

Process

- Every thing in unix is a file
- several instances of same programme can run at same time
- instance of a running programe
- data associated to process open file, allocated memory, stack, priority, pointer etc

Background Job

Same use through all shells

- useful
- for command which output can crammed later

Saving a task ada & at the end of line

Run_my_backup_programe &

[formerly background Job = %]

its control

- Jobs

return the list of background job from same shell

[1] running - bin/find —

[2]+ running make mistakes

> fg

makes mistake

> [ctrl] Z

[2] + makes mistakes &

> kill %. 1

[1]+ Terminated - bin/find —

- ps -ux list all process belonging to current user
- ps -Aux lists all process
- TOP display most cpu process sorted by cpu %
- kill <pids>

send abord signale to process lets process save data & let it exit itself

- kill -9 terminated by current user

: kill -9 -1 kills all the process of current user

: killall / xkill - all jobs will be terminated

Sequential commands.

- can type 1 command while and allowes you to type next even when current is not done
- can separate them using (;) symbol

echo "I am learning" ; sleep 10; echo "Hot"

- conditionals : use || (or) && (and)

mordhod || echo "sorry, files dont exist"

runs echo only if 1st commands fails

- in and condition 2nd one only runs if 1st is successful.

Chor 20

LINUX SHELL Environment

Quoting

double (") quotes can be used to prevent the shell from interpreting spaces as arguments separators as well as to prevent file name pattern expansion

> echo "hello world"
hello world

> echo "F.M$"
F.M b

> echo *.log
gives all files which have .log extention

> echo "*.log"
*.log

Single quoting (' ')

- Does's same functionality but what b/w quotes is never substituted

> echo '$USER'
$USER

- Back quotes used to call a command within another

> cd /lib/modules/`uname -r`
"uname -r" ; pwd
/lib/modules/2.6.9.1.6.fc2

* environment variables

- shell let user defined variables
- they can be accessed in shell commands (convention lower case names)
- you can also define environment variables are also within strips or executables called from the shell

env : list all defined variables and their value

- in your location can within current dir not can

Standard library variables

LD_LIBRARY_PATH
- shared library path

Screen id to display
- DISPLAY

EDITOR default editor

HOSTNAME name of local host

MANPATH manual page search path

PATH command search path

PRINTER default printer name

SHELL current shell type

TERM - current terminal type

USER - current user type

- Path Specify shell search order for commands

/home/ecov/bin:/usr/local/bin:/usr/kerberos/bin

Alias

Shell let you defined command with default arguments

command mapping

the which command.

before you run a command which tells you where it is found.

command history (!)

history

display latest cmd that you run and then number you can copy paste cmd by

LEC-21

Text editors

Graphical text editors
- nedit, Emacs, Xemacs, Kate, gedit

vi, Vim, nano

Nedit

- easy to select text
- highlight syntax
- easy to customize

Vi

- Difficult for beginners
- Very productive for power users
- often can't be replaced to edit files in System
- System embedded or when you just have text consoles

Vim

- found in most Gnu/Linux boots
- comes with GTK
- etc

Linux compression & archive

Lec-22

measuring Disk usage

du -h <file> (disk usage)

-h: return size on disk of given file, on human readable format:

K (kilobyte)
M (megabyte)
G (Gigabyte)

without h, du returns the raw number of disk blocks used by file

- du -s hedir>

-s: return the sum of disk usage

- df -h <dir>

return disk usage and free space

the h option only exists in GNU df

- df -h return disk file for all file systems

Compressing & decompressing

g[n]zip <file>

- it will create .gz file
ordinary performance

- b[un]zip2 <file>

more recent & efficient than .bzip2.

[un]lzma <file>

much better ratio than previous .bzip2)

compatible command line option.

Archiving

useful to backup or relocate set of files within 1 file

- tar: tap archive

Creating archive →

tar -cvf <archive> <files>

c: create
v: verbose useful
f: file

eg.

tar -cvf /backup/home.tar /home/bzip2/backup/home.tar.bzip2

- Viewing content in archive

tar -tvf <archive>

t: test

- extracting all files

tar -xvf <archive>

files or directories are given given with path relative to archive root directory

Extra for GNU Tar

J : uncompress on fly bzip2

Z : uncompress on fly with bzip.

Checking integrity

md5sum gets an hash value is used to check the integrity.

LEC-23

* Print & Sysc command

Unix printing.

- multijob, multipersion multiprinter in linux
- Printer independent system
- Robust System.

environment variable : PRINTER

- Lpr [-Pqueue] <files>
- A2ps [-P queue] <files> (conv to postscript format)
- Lpq [-Pqueue] list all print job
- Cancel <job no.> [<queue>]

* Using postscript & pdf files

- postscript was bad outputs but quality is very poor

better convert to pdf

Printing a pdf file

- you dont need to to open a pdf reader
- better convert to post script with dvdf2ps

* Smart directory copy with rsync

[rsync] remote sync has been to designed to keep in sync directories in same size are checked by checksum

- only transfer the block that differ with a file!
- can compress with transfer block
- can work through ssh

LEC-24

File comparison

diff file1 file2 report differ b/w two files

diff -r dir1/dir2

eg. tkdiff, Kompare

Find command

Find . -name "*.pdf"

list all .pdf in current directory

* find docs -name "*.pdf" -exec xpdf {} \;
find and display one after another.

Locate command

much faster regular expression search alternate to find

- locate keys: list all with key in name
- locate "*.pdf": list all with .pdf file
- locate "/home/gnd/beer*" lists all beer in absolute path
- locate is much faster because it indexed all files in database
- find is better to search through recent files

Info about users

- who - list all logged users
- whoami - tell what user am i logged as
- groups - tells which group i belongs to
- groups <user> - tell which group <user> belongs
- finger <user> - full user details

Change user

- su hydi - change to hydi account
- su - jekyll - log in jekyll account

log in with password

su - no argument given

The wget command

wget main features
- http and ftp support
- ~~can interrupt~~
- can resume interrupt downloads
- can download entire script
- proxy support (http-proxy env

Commands -

wget -c \ (continue download)
wget -m (mirror a site)
wget -r -np
(recursive downloads)

misc

sleep 60 - (60 sec sleep)
wc report.txt -> (word counting of file)
bc - (basic calculator)
date (current date)

Lec - 25

Basic Network Administration

- Ipconfig -a
print details about all network interface available on system

- Ip config eth0
  lists about eth0 connection
- Ip config eth0 192.168.0.100
  assigns 192.168.0.100 Ip to eth0.
  (1 Ip per interface)
- Ip config eth0 down
  shutdown eth0 interface
- Route add default gw 192.168.0.1
  Sets new route for Out - Side the local network
- Route -n lists existing routes
- Route del default or route del <IP> - delete given route

- your programme need to know what Ip address is given host name
- Domain name srver (DNS) can take care of this
- you have to specify Ip for 1 or more dns server in /etc/resolv.conf file
  nameserver 217.19.192.132
- Change take effect Immediately

Network testing

- First try to to ping Ip addrss of your gateway this will confirm network adaptar.
- make sure you can ping server Ip
- finaly you can ping any host

Lec-26

File system & Devices

Examples
- mkfs.ext2 /dev/sda1 - formats your usb key
- mkfs.ext2 -f disk.img formeet a disk image in ext-2
- mkfs.vfat -v -f 32 /dev/sda1 (-v, verbos) format back to fat 32.
- mkfs.vfat -v -f 32 disk.img formats disk image file in Fat32
- Blank disks images can be created using dd if=/dev/zero of=disk.img bs=1M

mounting devices
to make file system any many devices CD rom real or external storage
- first time create a mountpoint on Sys.

mkdir /mnt/usbdisk

- now mount it

mount -t vfat /dev/sda1 /mnt/usbdisk

-t specifies file system format time.

(.ext2, .ext3, .vfat .etc)

- you can also mount a file system in regular file (loop device).
  - useful to develope filesys for another machine
  - to access the contents of ISO cdrom image
  - use full to have linux file system inside a file in windows partition.

cp /dev/sda1 usbkey.img

mount -o loop -t vfat usbkey.img /mnt/usbdisk

- just use [mount] command with no argument it show all mounted systems

* umocnd / unmount

- unmount /mnt/usbdisk
  - to unmount disk
- close application in data it open.
- cmd used can be lsof <point>

* Saftware pakages

in unix we cans

sudo - apt get updat

- to install gimp pakage

sudo apt-get install <pakage>

- to remove gimp pkg

sudo apt-get remove <pakage>

- to show info

— " — show — " —

* to update pakage

— " — dist — " —

- graphical interface
  - GNOME

* Shut down

halt - hurting system

reboot - restart

[ctrl][alt][del] - shut down

LCC-27

* Shell Introduction

Agenda

- History
- Input / Output
- Variables
- sequence control
- arithmatic operations.
- Login operations
- file control & manipulate
- other feature such as traps.

History

- Shell is a command line interpreter and provides
  - interface b/w user & OS/kernel
  - interpret the command
  - executes action
  - captures the result
  - makes life easier
- provides flexibility & powerful.

Special feature of shell
- scripts can be invoked by commands
- may be used interactively or non-interactively
- allow both sync & Async commands
- support I/O redirect pipes etc.

Save with (.sh) extentions

*Executing the programme

- normal command line typing
- executable file
  - chmod -x <file name>
  - ./file
- sh <filename>
- #! /bin/bash
- debugging shell
  - -x option

Shell comments & variables

- # in the line
- comment must
  - mention why rather than what
  - more to next is more meaningful than to add 1.

variable

- No need to declare a variables beforehand
- how to read variable from reader
- how to print variable
- See variable.sh

LEC-29

Shell variables

- Shell variable can be used to store temporary value
  - $files="notes.txt suport.txt"
  - double quotes are needed because value contains space
  - See files.sh
- print out value of a shell variable with echo command.
  - $ echo $files
  - dollar tells the shell to insert variable's value into command line
- Environment variables.
  - shell variable are private to shell
  - special type of

Shell variable called environment variable
- programs environment is the set of environment variables are passed programs to run from shell

- in bash use to export a shell variable into environment

$_file = " notes.txt
$_export files or $export files
= notes.txt

Few Sys Variables

Variable	Discription
HOME	home directory
PATH	directory to be searched
PS1	prompt for interactive shell
PS2	prompt for multiline command
SHELL	Login shell environment
$$	process no
$!	process no of bg process
$?	exit value of last cmd.
$#	no of cmd line arguments
$0	cmd or program name
$n	positional parameters n=0,1,2,3,...
$*	All command line arguments

SHELL Array

LEC - 30

- groups data logically
- usually of same data type
- Bash
  - arrayname=(var1 var2)
  - access through ${arrayname[index]}
  - all element ${arrayname[*]} or ${arrayname[@]}

Arithmetic operations

- use external programs such as expr or awk
- four basic operators
  - → Add +
  - → Subtract -
  - → multiply * usually *
  - → division /
  - → Modulo %
- what is floating point no?
  - we will use [bc] tool or basic calculator

LEC - 31

Shell condition & Relation

IF-THEN-ELSE-FI

- condition == $!= operation

Relation operators

- find relation b/w two quantities
  - less than, equal to, greater than etc
- numerals & strings are handled differently in Shell
- For us with numerals use
  - -ne, -eq, -gt, -lt, -ge, -le

Boolean operators
- combine relation
- take multiple decisions before moving either or else.

! = not && and || or

(Exactly logic gates logic)

LEC-32

Shell examples

- touch: current date time

#!/bin/bash
greeting
echo "Hello, Dear $(logname)"
echo "Current date time is 'date'"
echo "User is whoami"
echo "present working directory 'pwd'"
echo "no of time in file: wc -l /example.sh"

Shell functions LEC-33

how to add functions in the shell

LEC-34

previous continues.

Loop control LEC 35

the for: for: in; do construct is used to repeat group of commands once for each item provided list.

Construct how following:

For VARIABLE in LIST
do
    Command List
done

For loop

$ for file in *.txt
> do
> mv -v $file $file.old;
> done

LEC-36

* shell script variations

For statement
exiting loop early (etc)
Break command is used.

The [continue] statement breaks current loop and continues the loop with exception of that specific line

LEC 37

Shell pattern matching

- echo ${path#/*/}
  delete longest path from starting that matches
- echo ${path#/*/}
  delete longest path from beginning that matches

- echo ${path%/*}
deletes shortest path from the end
- echo ${path%%/*}
delete longest path from the end that matches

/home/mysramoo/long/lazy.long
/filename.. this is example

① /home/mysramoo/long/lazy

Shell co-routine LEC 38

- when two procs are explicitly programmed to run simultaneously and possibly communicate with each other we call them co routine

output of proc 1 = input of proc 2

- how to get proc running some command of proc don't need to communicate

Ans - run prog process in bg

Adv - slight improvement in proc

Adv - if its simple proc other may be conflicts

Shell signals & traps

Signal
- software interrupt sent to a programme
- Indicate that an important event has occured
- eg: an interrupt signal indicate that the user has asked to do something that is not usual flow of control
- other signal include user defined signals

Basic Signal :-

SIGHUP - 1 - death of controlling proc
SIGHINT - 2 - interrupt from keyboard
SIGQUIT - 3 - Quit from keyboard
SIGFPE - 8 - float point exception
SIGKILL - 9 - Kill signals
SIGSEGV - 11 - Invalid memory refrence
SIGPIPE - 13 - broken pipe
SIGALRM - 14 - Term timer signal from alarm
SIGTERM - 15 Term Termination signal
SIGUSR1 - 30, 10, 16 - Term user def - sig1
SIGUSR2 - 13, 12, 17 - Term user def - sig2

Trap

Trap command enable or disable the actions taken when a signal receives

```
Hello(){
    echo "Hello world"
}
trap Hello 1 2 3 15
```

(use signal 15 to kill)

Lec-41

Shell Subshell

Ignoring signals
Trap '' Signals
trap : Signals

Ignoring signals using critical operations

```
trap '' 1 2 3 15
DoImportantStuff
trap 1 2 3 15
```

Sub Shell

have parent child subshell
- parent-child relation
- inheritance relation
- current directory
- environment variables
- standard input output
- signals that are ignored

Items not ignored

- shell variable except (.bashrc

Use of Subshell

- running program in parallel
- interprocess communication can be handled using traps kill commands

nested Subshell

- you don't need separate ~~scripts~~ examples
- (echo "Hello world")

code inside parentheses will run ~~separate~~ seperate

Eval

used to execute cmd twice
- see the escape ' ' it let you skip the process of the variable
- let us demonstrate with an example

```
lsfiles=ls | more
echo this 'lsfiles'
eval this 'lsfiles'
```

- you can modify arrays cmd according to your need

Shell declarations

Typed variables

declare integer variable
- declare -i <variable>
- -x exports variable (allows)
- -r make var read only
- -a variable treated as array

Module 03 shell examples

Module 04

Recap

- History
- various shell
- features of Bourne shell
* 3 modes of executing shells
- comments
- variables
  - shell, environment, sys var
- typed variable
- sequence & repetation
  - for loop - two ways
  - while do
  - until do
- Break and continues
- file test operations
- If then else
- If then elif
- case statement
- String, integer comparison
- expr evaluation
- arrays
- operations - Arithmetic, boolean etc
- function procedure recursion
- pattern matching
- coroutines
- signals
- traps
- subshell
- eval &
- examples

TLEC-45

Basics of Networking

- millions of connected computers : host and systems
- running network apps
- communication links
  * fiber, copper, radio
  * transmission rate = bandwidth
- Router: forward packets chunk of data
- Protocols - control sending receiving of msgs
  - TCP, IP, HTTP, FTP
- Internet
  - loosely hierarchical
  - public internet vs private intranet
- Internet standards
  - RFC Request for comments
  - IETF Internet Engineering task force
- Communication infra -
  - Streams
    - web, email, games
- communication service to provide to apps

whats a protocol

- "what is true"
- "how a person"
- interface

Structure of network [Lec-06]

- network edge
  - application and hosts
- network core
  - routers
  - network of network
- access network physical media
  - communication links

Network edge

- end systems (hosts)
  - run programs (app)
  - at edge of network
- client / server model
  - client host requests receives service from always on server
  - web browser/server; email client/server
- peer-to-peer model.
  minimal (or no) use of the dedicated servers
  - skype etc

☆ connection oriented service

goal: data transfer b/w end systems

- hand shake setup: (prepare for) data transfer ahead of time

★ hello hello back human protocol

- sets up "state" in two communicating hosts
- TCP transmission control protocol
  - Internet connection oriented service

TCP service [RFC 793]

☆

- reliable in order byte stream data transfer
  - loss: acknowledg and retransmission

flow control.

- Sender won't overwhelm receiver
- congestion control

" sender slow down sending rate " when network congested

Connection service

- UDP (User datagram protocol) [RFC 786]
  - connectionless
  - unreliable data transfer
  - no flow control
  - no congestion control

Apps using TCP

HTTP (web), FTP (file)
Telnet (remote login
SMTP (email

Apps using UDP

- streaming media
- teleconferencing
- DNS
- Internet telephony

Lec 47

* Network Core

- mesh of interconnected routers
- Data transfer through mesh
  - circuit switching (telephone)
  - packet switching (in chunks)

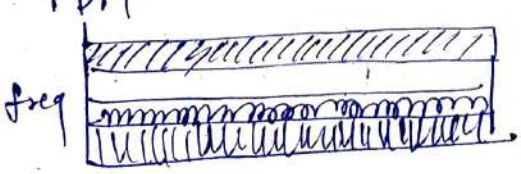
Circuit Switching
- link bandwidth switch capacity
- dedicated resource no sharing
- circuit like (guaranteed) performance
- call setup required

- bandwidth will be divided into pieces
  - pieces allocated to call
  - resource piece idle not used by owning call
- dividing link bandwidth into pieces
  - FDM
  - TDM

Frequency division multiplex

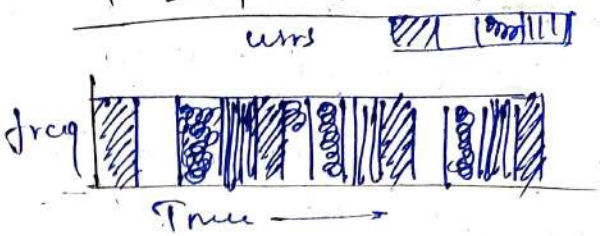
users

FDM



TDM

users



* Packet switching

each end to end data stream divided into packets
- user AB packet share network resources
- each packet uses full ~~length~~ link bandwidth

resource contention
- aggregate resource demand exceeded amount available
- packet queue wait for link use: congestion
- store & forward packets move one at a time

Lec-48

* Network access & Physical media

how connect end system to edge router
- residential access nets
- institutional access networks
- mobile access networks

Keep in mind
- Bandwidth
- Shared or dedicated

Residental access:

point to point access

- dial up via modem

up to 56 kbps direct access to router

can't surf on phone at same time

can't be always on

- ADSL

- up to 1 Mbps upstream
- up to 8 Mbps downstream

FDM 50kHz - 1MHz ↑

4kHz - 50kHz ↓

0kHz - 4kHz for phone

- CAN

Connects end system to edge routers

- ethernets
  - shared or dedicated link
  - 10 mbs 100 mbps gigabit ethernet

Wireless Access Network

Shared wireless access network connects end system to router

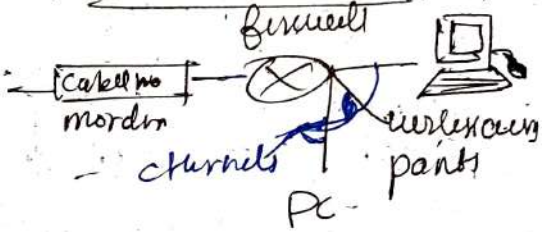
via base station "aka" access points

- wireless LANs
  - 802.11 b/g (wifi) 11 or 54 mbps

Wide area wireless access

- provided by telecom operator
- 3G - 384 kbps
- GPRS in europe/US

* home network



Physical media

- Bit: propogates b/w transmitter rcvr pairs
- physical link: what lies b/w transmitter and reciever
- guided media
  signal propogate in solid media
  copper, fiber, coax
- unguided media -
  ~~cable~~, wireless radio

Twisted pair

two insulated ~~media~~ copper wires

- Category 3 traditional phone wires 10 Mbps ethernet
- Category 5
  100 Mbps ethernet

coaxial cable

- two concentric coppor wires
- bidirectional
- base band
  - sngle cabamal an cable
  - leagacy ethernet
- broad band
  - multiplayar channil an cable
  - HFC

Fiber optic cable

- glass fiber carrying light pulse a bet
- high speed aprarion
  - high speed point-to-point transmitter
- low error rate: repealer placed for apart; immune to electromagratic noise

Radio

- carried an em spectrum
- no physical wire
- bi directional
- propogation environmut effects:
  - reflection abstruction interjace

Radio link typs

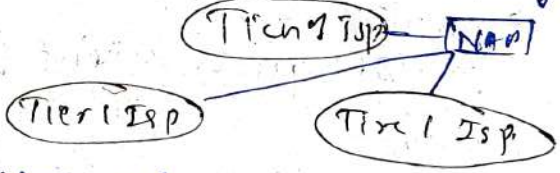
- terrestial microwave eg up to 45 mbps channels
- LAN (wifi) 11 Mbps, 54 mbps
- Wide area (cellular)
  - eg 3g ar 4G
- Satterlite
  - Kbps to 45 mbps per channel
  - 270 msec delay
  - geosychronous vs low altitude

Tee 49

Structure for ISP

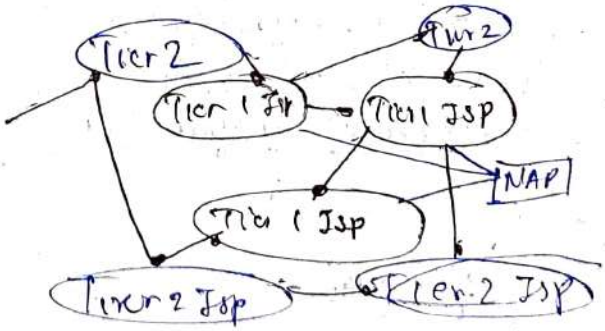
Internet structure network of network

- Roughly hierachical
- at center "tier 1" ISP's MCI sprint AT&T
- Treat each other as equa
- national internation coverage



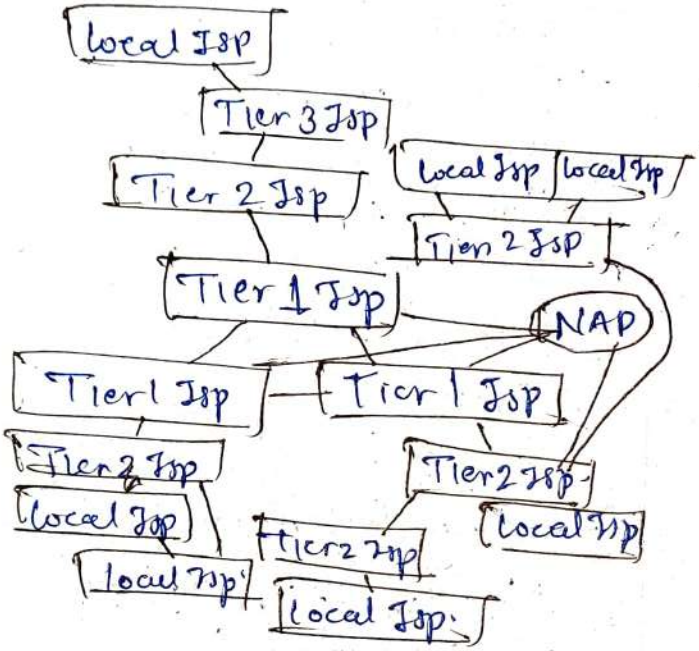
:. Nap - network acess points

→ search Tier 1 Isp
— Tier 2 Isp
smaller often regional Isp.

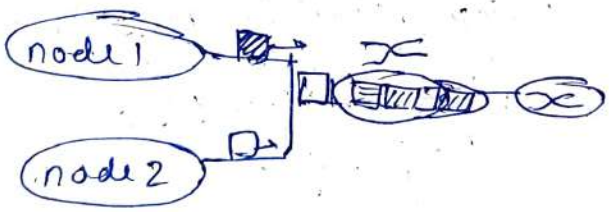


- also peer-to peer with each other
- Tier 2 depends on Tier 1 for internet

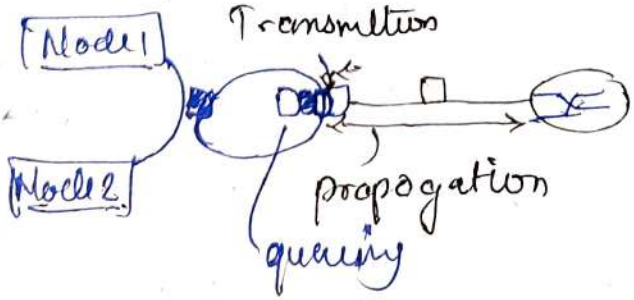
- Tier 3 ISP and local ISP (last hop (access) network (closest to end systems)



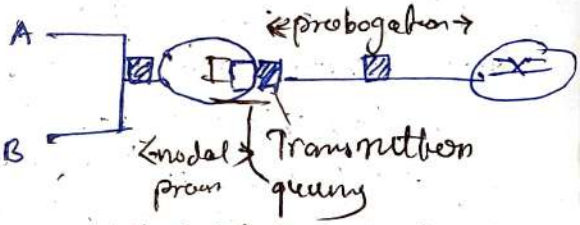
How do loss delay occure?
- packet arrival rate link exceeds output link capacity
- packet queue wait for turn



Sources of packet delay
- nodal processing:→
  - check bit errors
  - determine output link



- queuing
  - waiting at link for transmition
  - depends on congestion level of router
- Transmition delay
  R = link bandwidth (bps)
  L = packet length (bits)
  time to send bit $\frac{L}{R}$
- propogation delay
  - d = length of physical link
  - S = propogation speed in medium (~ 2×10^8 m/sec)
  propogation delay d/s



nodal delay

$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$

d_{proc} - process delay
few microseconds or less

d_{queue} - queuing delay
depend on congestion

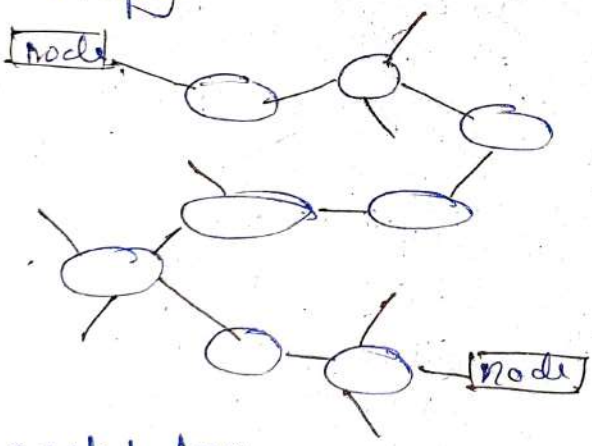
d_{trans} = Transmition delay
L/R

d_{prob} propogation delay
few microseconds

Real Internet delay & routes

what do 'real' internet delay & loss looks like

- Traceroute program provides delay measurement from source to router along end-end internet path towards
- Sends 3 packets that will reach router i on path towards dest
- router i will send packets to sender
- sender times interval b/w transmission & reply.



packet loss

- queue aka buffer preceding link in buffer has finite capacity
- when packet arrives to full queue packet dropped (aka lost)
- lost packet may be retransmitted by prev node, by source end system or not retransmitted at all

Network protocol layer

- hosts, Routers, links of various media
- application, Protocol hard ware software

→ each layer implements a service.
- via its own internal layer a services.
- relying on service provided by a layer below

IP stack

- application: supporting network application
  → FTP, SMTP, HTTP
- Transport: proces-proces data transfer.
  → TCP, UDP
- network - routers of data grams. from source to destination
  → IP routing
- link - Data transfer b/w neighbouring nodes
  → PPP, Ethernet.
- Physical - Bits on wire

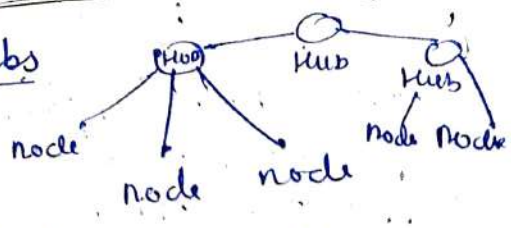
Encapsulation

Transport layer - create a segement and creates heads and in form wrapped form then it will pass through the layer.

T-5'1

Network devices

- **Hubs**



- interconnect lan segements
- extend max distance b/w nodes

- **Switch**
  - link layer
  - stores & forward ethernet frames
  - when frame is forwarded to a segment CSMA/CD used
  - host are unaware presence of switches
  - plug & play self learning

when switch receive frame

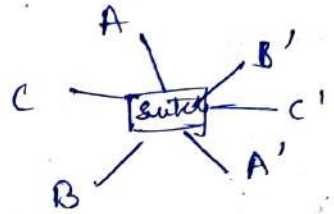
index switch using mac dest
if entry found
  then {dest from frame is arrived
    if (drop frame)
    else (forward frame on outgoing)
  }
else (flood)

Traffic isolation

- switch breaks subnet into lan segements
- switch filters packets
- collision can occur within single hub with same node (common)

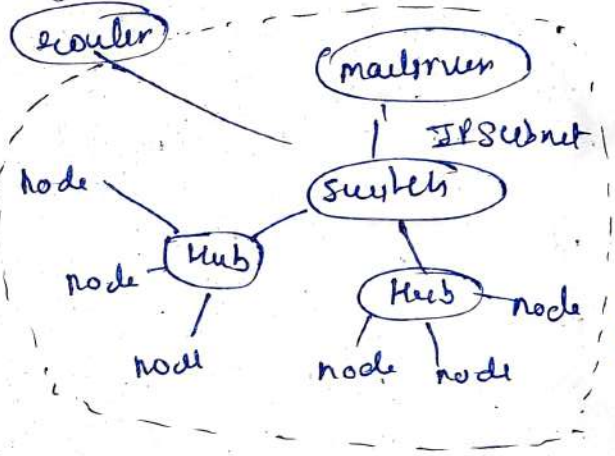
dedicated access

- it has many interface
- host have direct connection with switch
- No collision full duplex



A → A' & B → B' simultaneously no collision occured.

Small network



Switch and routers
- both store & forward
routers - network layer
switches - link layer
- Router maintain routing tables implementing routing algos
- Switch maintain switch tables implementing filtering & learning algo

Sum up	hub	router	switch
isolation	X	✓	✓
plug & play	✓	X	✓
optimal routing	X	✓	X
cut through	✓	X	✓

Network Security

- what is it ?
  - confidentiality
  - Authentication
  - messag intigrity
  - Accss & Availability

Cryptography

- Symmetric key - sndr & recvr have identical key
- Public key - encryption public key decryption privab ky.

Symmetric key cryptography

Subtitution cipher: replacing one thing for another

- monoalphabetic cipher

plaintext : abcd
ciphertext : mnbv

eg :
a b
↓ ↓
n v

how to break simple cipher
- brute force
- others

DES

data encryption standard
- us encryption standard [NIST 1993]
- 56 bit symmetric key 64 bit patin text input

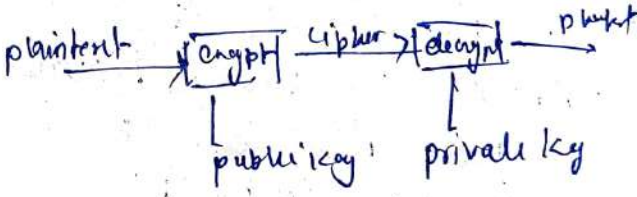
DES challange -
56 bit key encryped phase decrypt in 4 months
- No known backdoor
→ uses security keys
3DES on each datum
- use cipher block chaing

AES

- new symmetric key NIST replacing DES
- data processing 128 bit block
- 128 192 or 256 bit keys
- Brute fore take 1sec on DES & takes 149 trillion years for AES

Public key cryptography

- Radically different approach (diffe hellman, RSA etc)
- reciver, sender dont share similar key
- public encryption key known to all
- private decryption key known to reiver only



RSA (useful property)

$K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$

use public key first followed by private key

private key first followed by public key

Authentication

goal: B wants A to prove its identity

protocol ap1.0 = A says "I'm A".

Let can easily be pretended or authentication failed by C

protocol ap 2.0 : A says "I'm A" in an IP packet containing her source address.

Protocol ap 3.0 A she says "I am A" and sends A secret password to "prove" it

- in this C records B's packet and later plays it back to B

Protocol ap 3.1 A says I'm A and sends encrypted secret password.

but C's record & playback still works

Protocol ap 4.0 avoid playback attack

Nonce: Number (R) used only once in a lifetime

AP. 04 to prove A. (um) B send A on nonce (R) A must return (R) encrypted with shared secret key

Auth protocol AP 5.0

using Nonce. public key cryptography

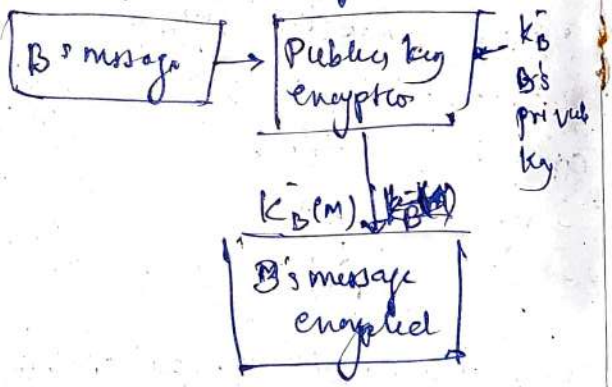
Digital Signatures

Cryptographic techniques analogous to hand-written signatures

- sender (B) digitally signed document.
- verifiable & nonforgeable recipient A can prove some one that B has no one else including A must have signed document

Digital signature for message

B signs m by encrypting with his private key K_B^- creating a signed message $K_B^-(m)$



- Suppose A receive message m. digital Signature $K_B^-(m)$
- A verifies m signed by B by applying B's public key K_B^+ to $K_B^-(m)$ then check $K_B^+(K_B^-(m)) = m$
- If $K_B^+(K_B^-(m)) = m$ whenever Signed used B's private key

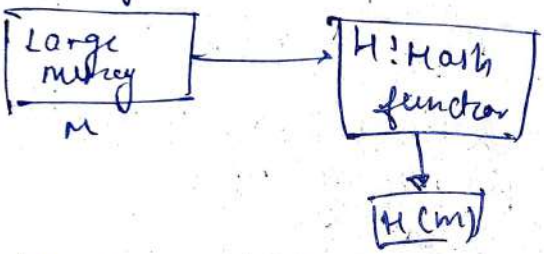
A thus verifies that

- B signed m
- B signed m not m'

Acem take m & signature $k_B(m)$ to prove B signed m

Message Digests

goal: fixed length easy to compute digital fingerprint
- apply Hash function to m get fixed size message digest (H(m))



Hash properties

- Produces fixed-size message digest (fingerprint)
- given message digest x computationally infeasible to find m such that $x = H(m)$

Hash function Algorithms

- MD5 (RFC 1321)
  - computes 128 bit message digest in 4 step process
  - arbitrary 128 bit string x appears difficult to construct msg m whose MD5 hash is equal to x
- SHA-1
  - US standard (NIST, FIPS, PUB 180-1)
  - 160 bit message digest.

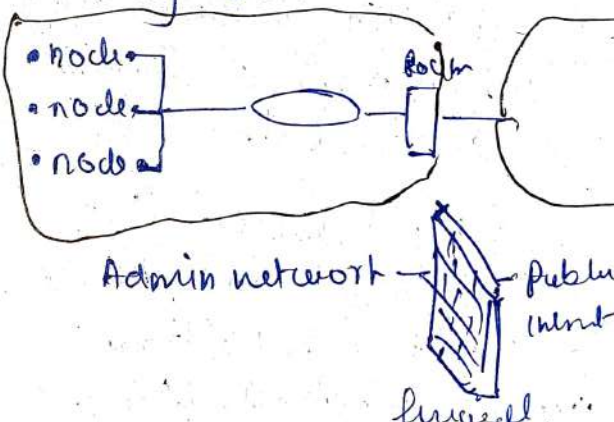
A certificate contains:

- Serial number
- Info about certificate owner, including algo and key value
- Info about certificate issuer
- valid dates
- digital signature issuer

In practice (Live) [ST

Firewall

isolates organization internal net from larger internet, allowing some packets to pass, blocking others.



- Prevent denial of service attacks
- Prevent illegal modification/access of internal data
- allows only authorized access to inside network

Types of firewall
- application level
- packet level filtering

Packet filtering

- internal network connected to internet via router firewall
- router filters packet by packet decision to forward/drop packet on
  - source IP / destination IP
  - TCP, UDP source & destination port numbers
  - ICMP message type

eg Block incoming & outgoing datagrams with IP protocol field 17 with either source or dest port 23

- all UDP incoming and outgoing either combination are blocked.

eg - block inbound TCP segment with ACK=0

- prevent external from making connections with internal clients to connect to outside.

Application Gateway

filters packets on application data as well as on IP/TCP/UDP fields

- allows specific users in to ~~telnet~~ from internal net to through gateway

Limitations of firewall & gateway

- IP spoofing: router don't know if data comes from claimed source
- client software know to contact gateway
  - must set IP of proxy on web
- filters use all or nothing policy of UDP
- Trade off: degree of communication with outside world
- many high protected sites still suffer from attacks

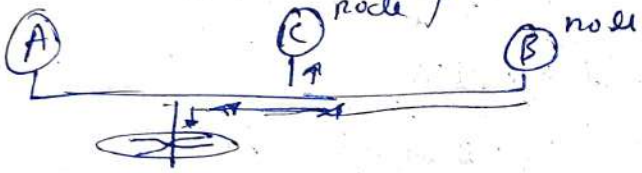
Internet sec. threats

Mapping →

- Before attacking - "case the joint" find out what service is implemented on network
- use [ping] to determine which host has address on network
- Port Scanning: Try to establish TCP connection to each port in sequence
- nmap.

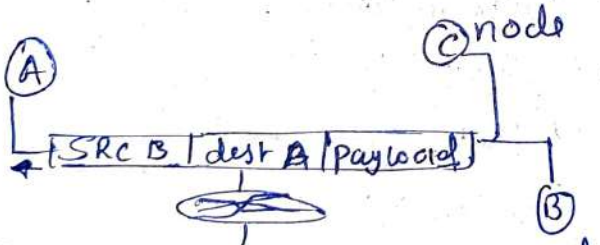
countermeasures —
- record traffic entering the network
- look for suspicious network activity.

Packet sniffing
- Broad cast media
- Promiscious NIC reads all packets passing by
- can read all unencrypted messages passing by

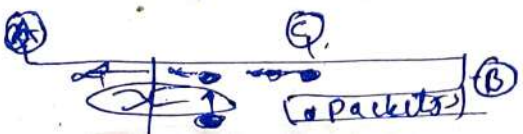


A can sniff data of B.

Ingress filtering
router should not forward outgoing filters with in valid source address (eg datagram, source address etc)



Denial of Service (DOS)
- flood of maliciously generated packets "swamp" receiver
- Distributed DOS (DDOS)
- multiple sources of attack
- botnets C and remote hosts attack A



countermeasures —
- filtering out flooded packets (eg SYN) before reaching hosts
- Traceback source of floods (mostly on compromised machine) (hacked device)

Secure Socket layer (SSL)
- Transport layer security to any TCP-based app using SSL services
- used b/w web browsers. services for ecommerce (shttp)
- Security service
  - Server authentication
  - data encryption
  - client authentication
- Server authentication
  - SSL enabled browser includes public keys
  - Browser requests server certificates issued by trusted CA's
  - Browser uses CA's public key
- encrypted SSL Session
  - Browser generates symmetric session key
  - server private key server decrypts session key
- SSL : Basis of IETF transport layer security

IPsec - network layer security
- Network layer security
  - Sending hosts encrypts data in IP datagram

TCP & UDP segements ! ICMP and SNMP messages
- Network layar Acethentcation
- destination hosts can authenticate sourc Ip address
- Two principle protocls
- authentication header protocol (AH)
- encapsulation security payload (ESP)
- for both AH & ESP, sourc destination handshake logical channel called security association (SA)
- each SA unidirectional
- uniquely delimited by
- security protocol (AH) or (ESP)
- sourc IP
- 32 bit connection IP

WIRESHARK L57

Network protocol analyzer
- computer s/w or H/w undersepts or logs traffic pasing over the network
- captures packets decodes and analyze contents
- network analyser used for
- trouble shooting problm on network
- Analyzing performen of network to detrom bottlenet
- network intrusion detection
- analyse the aprator of application
- packet sniffing applicaton
- very similer in funally to Tcpdump.
- GUI front end

Features

- understand structen of different network protocols
- Display encapsulaton signal fields and intputs their meaning
- only supported by Pcap (only capturn)
- cross platorm running on various OS

WinPcap

- Industries level tool for link layor network access environment
- allows application to capture and transmitts network by passing protocols stack
- Consists of driver extends OS

- consist of library for easy access to low level network layers
- also windows version of LibPcap UnixAPI.
- don't generate network data passive tool
- detail info protocol it understands
- it can only capture data as well as the OS / trojan / drivers
- an example of capturing data over wireless network

SNORT

- Intrusion detection defined problem Identifying who are using computer systems without authorization
- intrusion detection is not intrusion prevention
- Policy

Successful intrusion detection depends on policy and management as much as technology
- Security policy
- notification
- Response coordination

INTRO - multimode packet analysis tool
. sniffer, packet logger forensic data analysis tool.
- network intrusion detection system

metrics -
- small - 800 k source download
- portable (Linux, windows, etc
- fast (@ 100 mps, durability)
- configurable.
- free

Design -
- light weight on network
- Libpcap-based sniffing system
- Rules based detection engine
- plugin system allows endless flexibility

Plug-in's -
- Preprocessors
packets are examined manipulated before being handed to the detection engine
- Detection
perform single simple tests on single field/aspect of the packet
- Output
report results from other plugins

uses -
- Standard packet sniffing

NIDS
- policy enhancement
- Scan, detection / traps.

Using Snort —

— 3 main operational modes
- Sniffer
- packet logger
- NIDS

- they are configured by command line switches

Snort automatically tries to go into NIDS mode if no command line switches are given look for snort.conf configuration file

NIDS modes

- Uses all phases of Snort + plug ins to analyse traffic to both misuse detection and anomalous activity
- Can perform port scan detection, IP Defragmentation TCP stream, reassembly application layer analysis and normalization etc

modes —

- output options available
  - Database (mySQL)(orac...
  - XML (snml DTD from CMU/CERT)
  - Tcp dump binary format
- Unified format
- ASCII, sys log, win popup (SMB)
- etc