CMP325
Operating Systems
Lecture 02

Introduction to Linux Environment

Fall 2021

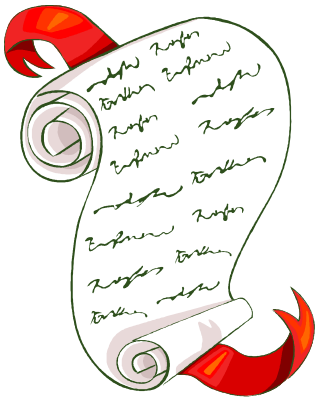
Arif Butt (PUCIT)

Note:
Some slides and/or pictures are adapted from course text book and Lecture slides of

- Dr Syed Mansoor Sarwar
- Dr Kubiatowicz
- Dr P. Bhat
- Dr Hank Levy
- Dr Indranil Gupta

For practical implementation of operating system concepts discussed in these slides, students are advised to watch and practice video lectures on the subject of **OS with Linux** by Arif Butt available on the following link:
http://www.arifbutt.me/category/os-with-linux/

Today's Agenda

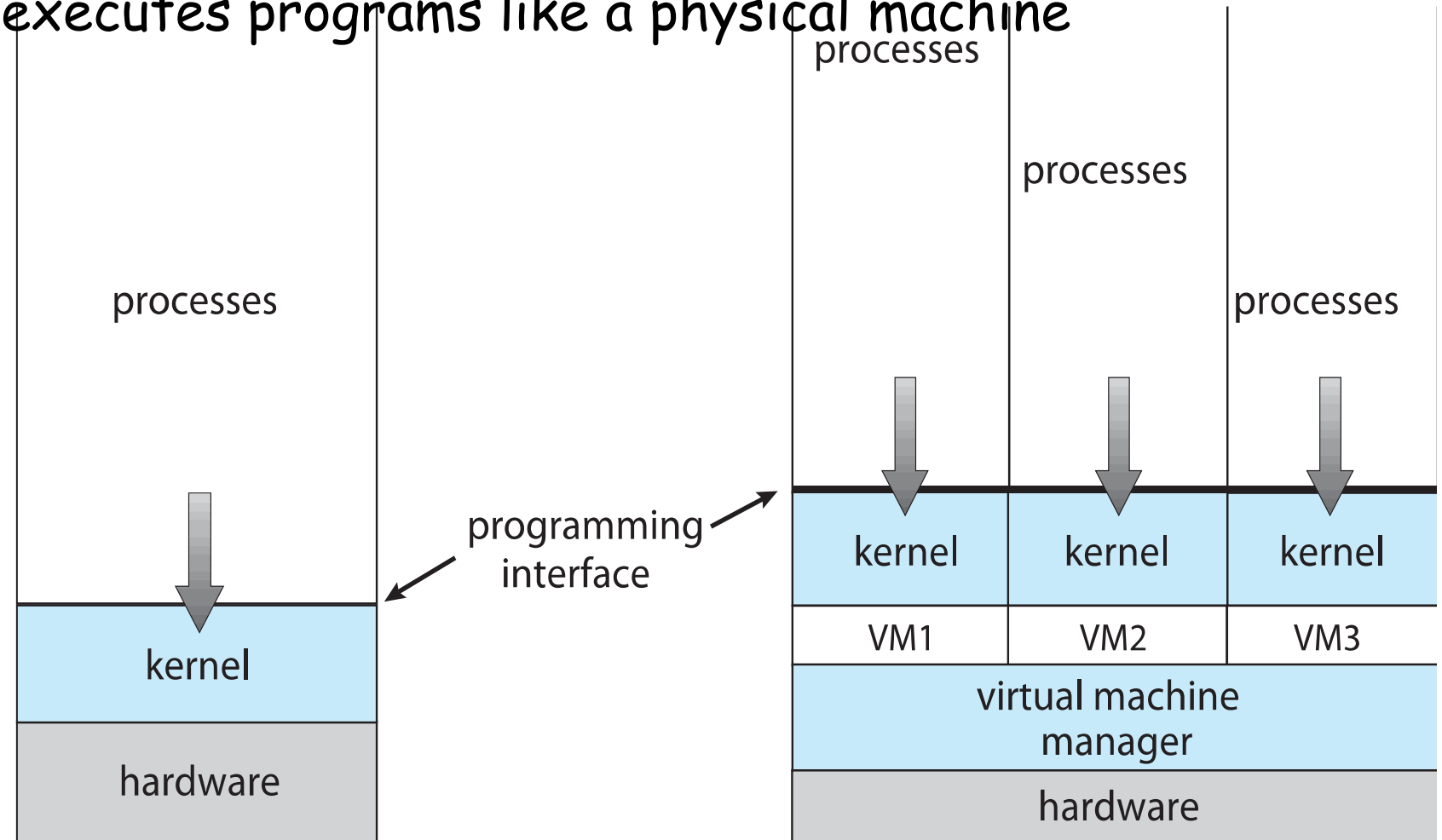


- Review of previous Lecture
- Virtualization and Hypervisors
- Introduction to Linux Distributions
- Installing Linux on Sun Virtual Box
- Interacting with Linux OS
- Linux Shell Commands
- Linux File Hierarchy Standard
- Linux System Calls Interface
- Compiling a C program in Linux

Concept of Virtualization

Virtualization

- Virtualization is a framework or methodology of dividing the resources of a computer system into multiple execution environments
- A virtual machine is a s/w implementation of a machine that executes programs like a physical machine



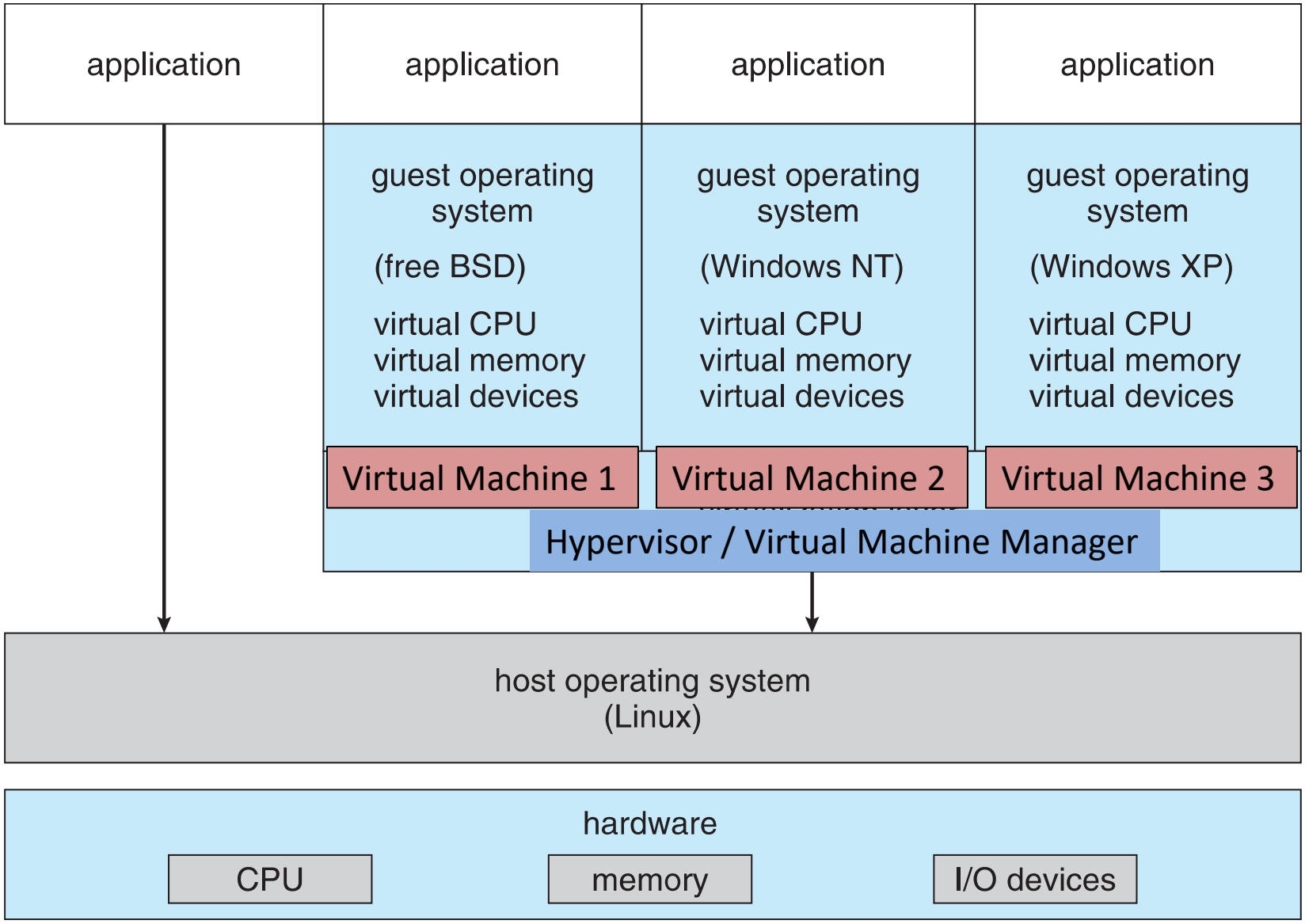
Implementation of VMMs/Hypervisors

- **Type 2 hypervisors:** Applications that run on standard operating system (host OS), and provide VMM features to guest operating systems. Examples: Oracle VirtualBox, VMware Workstation and Fusion
- **Type 1 hypervisors:** Sits right on top of h/w, so there is no concept of host OS. Guest OSs runs on top of hypervisor. Examples: Oracle VMServer for SPARC and x86, Vmware ESX, Critix XenServer, MS Windows Server with HyperV, RedHat Linux with KVM
- **Type 0 hypervisors:** Hardware-based solutions that provide support for virtual machine creation and management via firmware. Examples: IBM LPARs and Oracle LDOMs

Other Variants of VMMs/Hypervisors

- **Paravirtualization:** Technique in which the guest OS is modified to work in cooperation with the VMM to optimize performance
- **Programming-environment virtualization:** VMMs do not virtualize real hardware but instead create an optimized virtual system. Example: Oracle Java Virtual Machine and Microsoft.Net
- **Emulators:** Allow applications written for one hardware environment to run on a very different hardware environment, such as a different type of CPU. Example is Qemu
- **Application containment:** Not virtualization at all but rather provides virtualization-like features by segregating applications from the operating system, making them more secure, manageable. Including Oracle Solaris Zones, BSD Jails

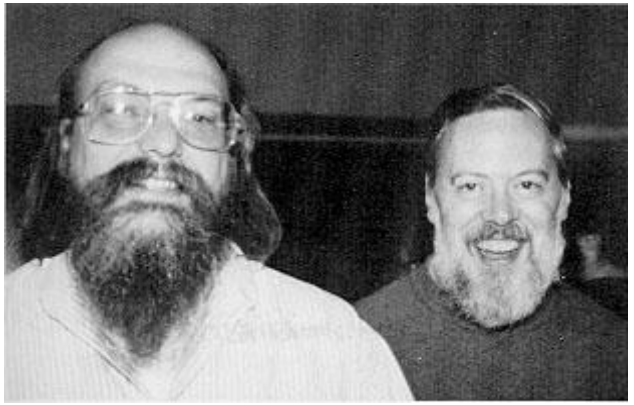
Architecture of Type2 Hypervisor



Intro to Linux OS

History of UNIX

- All modern operating systems have their roots in 1969, when Dennis Ritchie and Ken Thompson developed the C language and the UNIX operating system at AT&T Bell labs



Ken Thompson Dennis Ritchie

- Since the source code of UNIX was widely available, various organizations developed their own versions, which led to chaos as far as UNIX history is concerned.
- Two major versions developed:
  - System V, from AT&T
  - BSD (Berkley Software Distribution from UC Berkeley) Minor variation includes FreeBSD, OpenBSD and NetBSD.
- To make it possible to write programs that could run on any UNIX system, IEEE developed a standard for UNIX, called POSIX and later SUSv3, that most versions of UNIX now support

History of Linux Kernel

- In 1991, Linus Torvald, a student of university of Helsinki Finland, bought a 386 computer and tried to write a brand new POSIX compliant kernel, which became what we call Linux today
- Todays Linux run on:
  - 97% of all world's super computers (including top 10)
  - 80% of all smart phones
  - Millions of desktop computers
  - 70% of all web servers run on Linux
  - Embedded Systems (routers, Rpi boards, self deriving cars, washing machines etc)
- Source code of latest stable kernel (4.18.5) can be downloaded from https://www.kernel.org

Linux Distributions

A Linux distribution is a compilation of Linux Kernel bundled with:

- System management tools
- Server softwares
- Desktop applications
- Documentations

Some popular Linux distributions are:

- Kali Linux (https://www.kali.org)
- Red Hat (https://www.redhat.com/en)
- Ubuntu (https://www.ubuntu.com)
- CentOS (https://www.centos.org)
- Debian (https://www.debian.org)
- Linux Mint (https://www.linuxmint.com)
- OpenSuSe (https://www.opensuse.org)

UNIX is basically a **Simple** Operating System

But YOU have to be a GENIUS to understand the Simplicity

Dennis Ritchie

Linux Installation on Sun Virtual Box

New Settings Discard Start

Details Machine Tools Global Tools

- **Kali-Linux** Saved
- **Ubuntu-Server** Saved
- **Windows10** Saved
- **WinXP** Saved
- **Minix-320** Saved
- **vyatta** Saved
- **CentOS** Saved
- **BackTrack** Saved
- **GnackTrack** Saved
- **RHEL6** Saved
- **Ununtu-17** Saved
- **Win 8.1** Saved
- **prostar** Saved

General

Name: Kali-Linux
Operating System: Ubuntu (64-bit)

System

Base Memory: 3096 MB
Processors: 3
Boot Order: Floppy, Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, KVM Paravirtualization

Preview

Display

Video Memory: 16 MB
Remote Desktop Server: Disabled
Video Capture: Disabled

Storage

Controller: IDE
IDE Secondary Master: [Optical Drive] Empty
Controller: SATA
SATA Port 0: Kali-Linux.vdi (Normal, 100.00 GB)

Audio

Host Driver: CoreAudio
Controller: ICH AC97

Network

Adapter 1: Intel PRO/1000 MT Desktop (Bridged Adapter, en0: Wi-Fi (AirPort))

USB

USB Controller: OHCI, EHCI
Device Filters: 0 (0 active)

Shared folders

Shared Folders: 1

Description

None

Contains a list of Virtual Machine details

Interacting with Linux OS

Interacting with Linux OS

Option-I: Use a desktop/laptop computer of your own running either a

- real Linux distribution, (may be dual boot)
- MS Windows operating system executing cigwin dll
- guest Linux operating system using some virtualization software

Option-II: You may like to remotely login using ssh, telnet, putty, or some other remote login facility on PUCIT LAN

```
ssh username@172.16.0.21
ssh username@172.16.0.103
```

Option-III: You can also login using WAN on following machines as well (if permitted)

```
ssh  username@202.147.169.197  (Solaris 11.0)
ssh  username@202.147.169.196  (PC BSD)
```

Interacting with Linux OS

- For a user of an operating system there are two types of interfaces, using which a user can give commands to perform various operations:
- **Graphical User Interface:** GNOME, KDE, Unity, Xfce, Enlightenment, Sugar
- **Command Line Interface:** Also called a shell. A Linux shell is an interactive program that accepts commands from user via key board, parse them from left to right and execute them. Most of the shells available in todays Linux provides the features of executing user commands and programs, I/O handling, programminng ability (scripts and binaries). Example shells are Bourne shell, Bourne Again Shell, C Shell, Korn Shell

Linux Shell Commands

- A shell command can be internal/built-in or External
- The code to execute an internal command is part of the shell process, e.g., **`cd, dot, echo, pwd`**.
- The code to process an external command resides in a file in the form of a binary executable program file or a shell script, e.g., **`cat, ls, mkdir, more`**.
- The general syntax of a shell command is

  command [option(s)] [argument(s)]

- After reading the command the shell determines whether the command is internal or external
- It processes all internal commands by using the corresponding code segments that are within its own code
- To execute an external command, it searches the command in the **search path**. Directories names stored in the PATH variable. [echo $PATH]

Linux Shell Commands

Basic Commands	Description
`who, whoami, finger, users`	User information look up programs
`logout, exit, ^D`	Terminate the current shell session
`alias, unalias`	Used to create/remove pseudonyms for commands
`passwd, chfn`	Used to change user password, user info
`date`	Prints or sets the system date and time
`cal`	Displays calender for specific month or year
`clear`	Clear the terminal screen
`hostname`	Display/set the system hostname
`uname -a`	Prints system information
`man [-k]`	Displays online documentation (/usr/share/man/)
`apropos, mandb`	Searches the whatis database for strings
`whatis, updatedb`	Searches the whatis database for complete words
`info`	Reads information documents
`whereis filelist`	Locate binary(-b), source(-s), man pages(-m)
`which, type`	Locate cmd and display its pathname/alias
`watch`	Used to execute a program every 2 seconds

Linux Shell Commands

Commands for Dirs only	Description
`cd`	Change directory
`mkdir -[p], rmdir -[p]`	Create and remove a directory.
`pwd`	Display present working directory

Commands for Files only	Description
`cat, less, more, head, tail`	View contents of a file
`file`	Determines file type
`wc`	Displays line, word, character count of file(s)
`uniq`	Report or omit repeated lines
`sort`	Sort lines of files
`cut`	Remove col(s) from tabular files (tab,collon,space)
`paste`	Horizontally concatenate contents of two or more file
`grep`	Prints lines of files where a pattern is matched
`gzip, gunzip, bzip2, bunzip2`	Compression and un-compression softwares

Linux Shell Commands

Commands for Files/Dirs	Description
`cp -[rpif]`	Copy files and directories
`mv`	Move/rename files
`rm -[rfi]`	Removes files/directories
`stat`	Displays file/directory statistics
`touch`	Update timestamp of a file/dir (coreutils)
`find / -name mv`	Search a file based on attribute in a dir hierarchy
`locate, updatedb`	Searches for the string in database(s)
`ls [-aldihFvStr]`	Displays calender for specific month or year
`ln`	Create soft/hard links
`tar`	Archiving utility
`chmod`	Change file mode bits
`chown`	Change file ownership and group
`umask`	Display/Set file mode creation mask

Linux Shell Commands

Advance Commands	Description
`pipe, tee, mkfifo, mknod`	Used for IPC (pipes and fifos)
`bg, fg, kill`	Send a signal to a process
`adduser`	To create or delete a user
`deluser -[remove-home]`	Delete a user as well as his home directory
`addgroup, delgroup`	For creating/deleting a group
`usermod, groupmod`	Modify a user/group information
`ps, top, uptime,`	To retrieve process related information
`vmstat`	Display virtual memory status
`nice, renice`	To run/alter the nice value of a process (-20 to +19)
`shutdown`	Bring the system down
`reboot, halt, poweroff`	Used to reboot or stop the system
`telinit`	Change system runlevel
`runlevel`	Outputs previous and current runlevel
`sysv-rc-conf`	Used for startup service(s)
`cron, anacron`	Used to scheduler commands

Linux Shell Commands

Advance Commands	Description
`fdisk`	Manipulate disk partition table
`df`	Disk full, report file system diskspace usage
`du`	Estimate file space usage
`free`	Display amount of free and used memory in system
`mount [-t fstype] [dev] [mp]`	Mount a file system
`cpio`	Copy files to and from archives
`script`	Make typescript of terminal session
`lpr`	Print files
`stty`	Change and print terminal line settings
`ar, ranlib`	For static libraries
`source`	Execute a script by the current interpreter
`export`	To export a variable into the environment

Linux Shell Commands

Network Commands	Description
`ping`	NW diagnostic tool
`mesg`	Allows or disallows writing messages to screen
`write <user> [tty]`	Allows realtime messaging between users on NW
`telnet`	Remote login program
`ssh`	Remote login program -SSH client
`netstat`	Network statistics utility
`scp`	Remote file copy program
`service`	Command used to start/stop OS services
`initctl`	Init daemon control tool

Linux Shell Commands

Commands Related to C Progam	Description
`gcc, g++`	C and C++ Compilst
`gdb`	GNU Debugger
`indent`	Changes the appearance of a C program
`make`	Utility for managing large programs
`ar, ranlib`	Used for static libraries
`nm`	List symbols from object files
`strace`	Trace system calls and signals
`od`	Dump files in octal and other formats
`strip`	Discard symbols from object files
`objdump`	Display information from object files
`objcopy`	Copy and translate object files
`addr2line`	Convert addresses into file names and line numbers

UNIX Manuals

- Don't expect to remember everything… I don't!
- Use **man** program to display help pages from /usr/local/share/man/ directory having further sub-directories each for following:
  - 1 – Shell commands; *e.g., mv, ls, cat, …*
  - 2 – System calls; *e.g., read(), write(), open(), …*
  - 3 – Library calls; *e.g., printf(), scanf(), …*
  - 4 – Device & NW specific information
  - 5 – File Formats; *e.g.,* /etc/passwd, /etc/shadow,
  - 6 – Games & demos; *e.g., fortune, worms, …*
  - 7 – Miscellaneous; *e.g., ascii character map, …*
  - 8 – Admin functions; *e.g.,* fsck, network daemons

File Hierarchy Standard

Linux File Hierarchy Standard

- All UNIX based OSs normally follow the FHS. To get info of your file system hierarchy you can give the command **`$man hier`** or can visit the following link

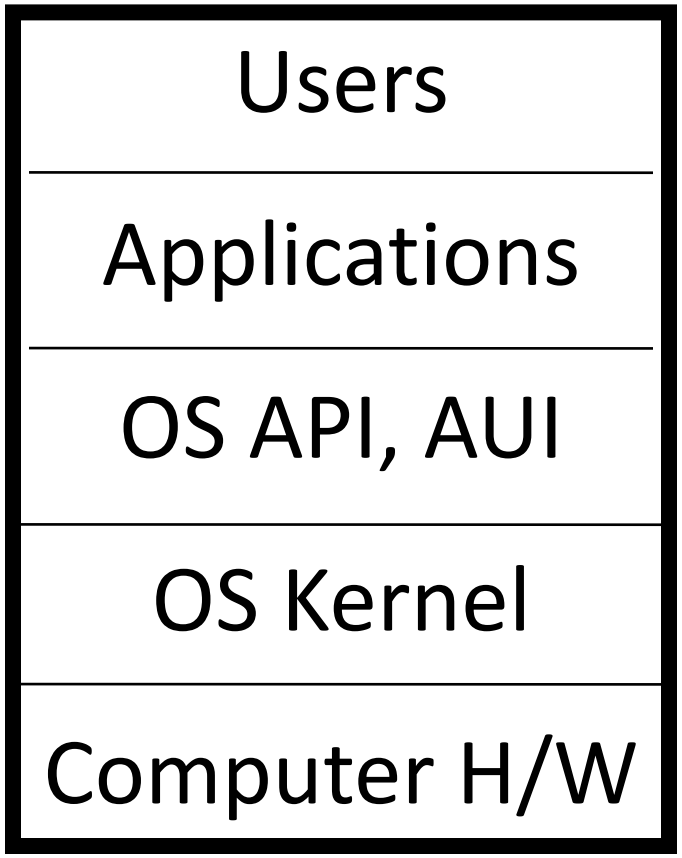
  http://www.pathname.com/fhs/pub/fhs-2.3.pdf

- Every thing that exist on your Linux system can be found below the root (/) directory. Some important directories are:
- Binary Directories: `bin, sbin, lib, opt`
- Configuration Directories: `boot, etc`
- Data Directories: `home, root, media, mnt, tmp`
- In-memory Directories: `dev, proc, sys`
- System resources: `usr`
- Variable data: `var`

Linux
System Call Interface

OS Kernel

- **Kernel** consists of everything below the **System Call** interface and above the physical h/w.
- Kernel is the place where real work is done, it provides the process mgmt, memory mgmt, I/O mgmt, file mgmt, CPU scheduling, and other OS functions.
- Kernel is also called message exchange, because no component can communicate without it.
- Kernel is never paged out of memory and its execution is never preempted.



Types of Entry Points to Kernel

- **Kernel code will be executed in one of the following four occasions:**
  - When a program makes a **System Call**.
  - When an I/O device has generated an **Interrupt**; e.g. a disk controller has generated an interrupt to CPU that my reading is complete the data is now sitting in my buffer, You can go and get it.
  - When a **trap** occurs; e.g. If a program has made a division by zero, a trap will be generated which will execute a different piece of code in kernel (TSR).
  - A **signal** comes to a process. For that as well some piece of kernel code will be executed.

System Call

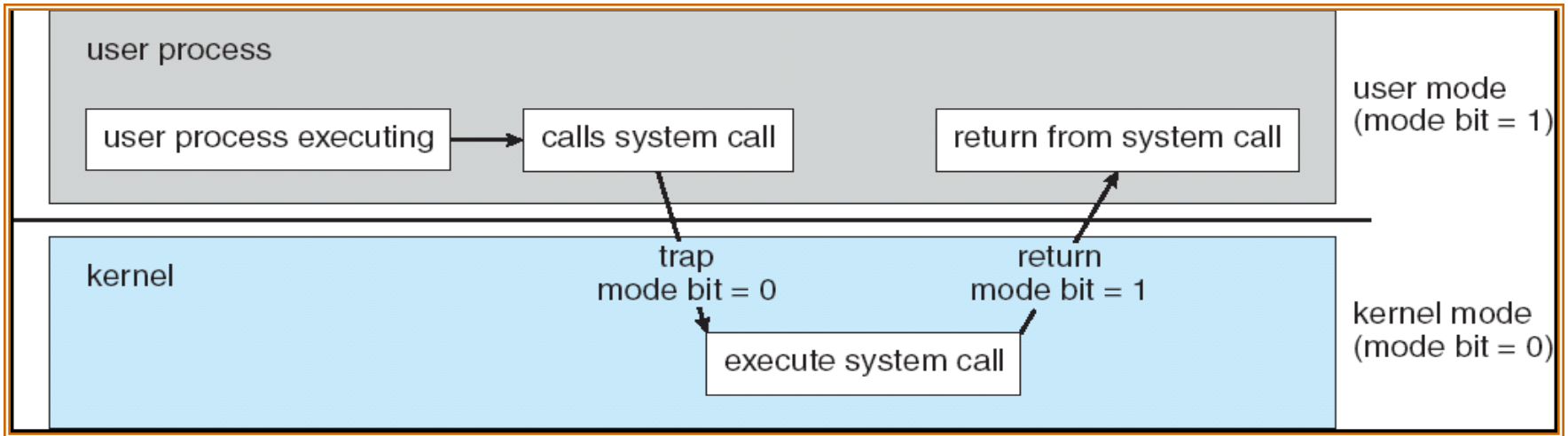
A system call is the *controlled* entry point into the kernel code, allowing a process to request the kernel to perform a privileged operation. Before going into the details of how a system call works, following points need to be understood:

- A system call changes the processor state from user mode to kernel mode, so that the CPU can access protected kernel memory
- The set of system calls is fixed. Each system call is identified by a unique number
- Each system call may have a set of arguments that specify information to be transferred from user space to kernel space and vice versa
- One must go through the man pages for better understanding:

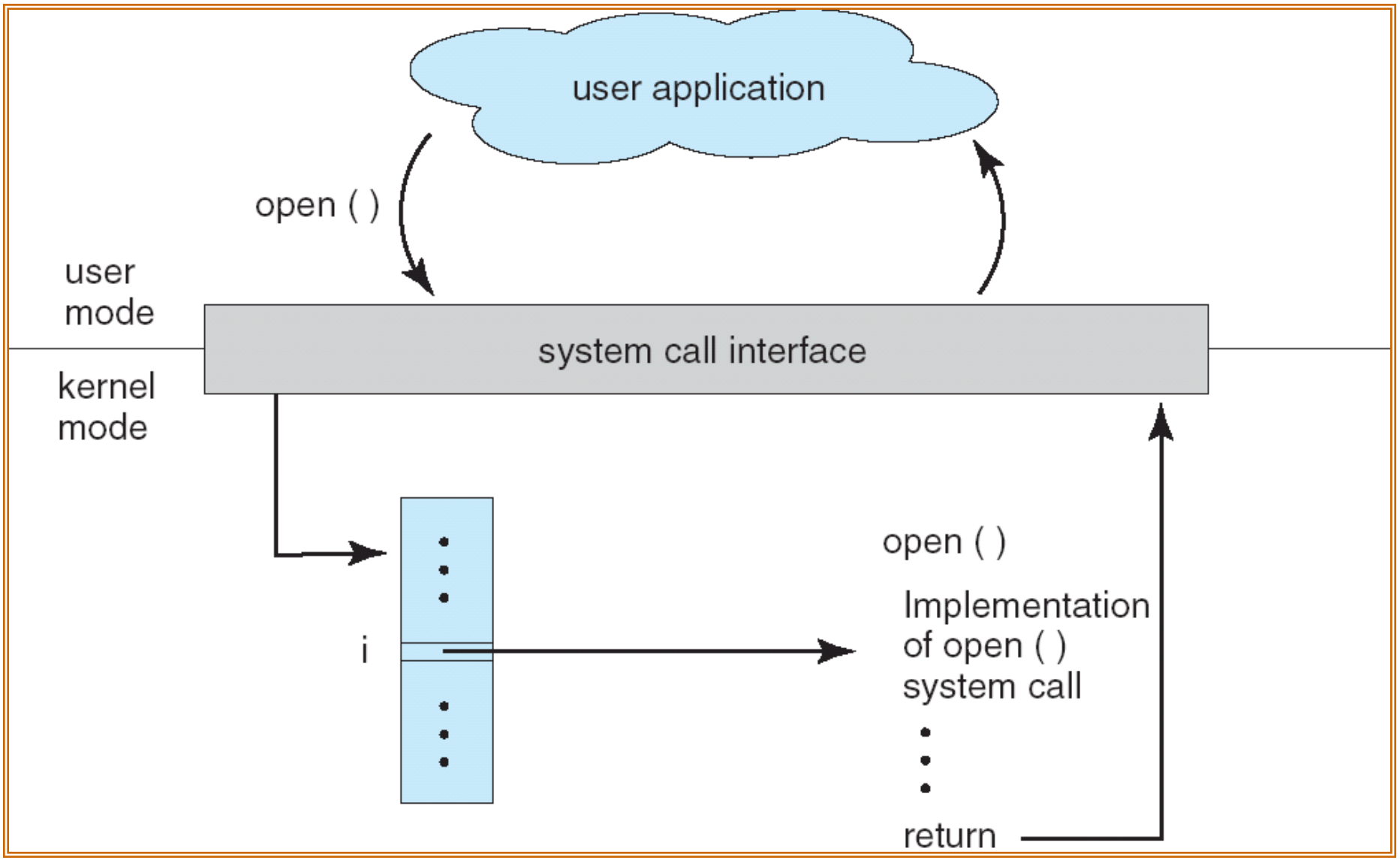
`man 2 intro` Introduction to Section 2 of man pages
`man syscalls` List of system calls (wrappers)
`man syscall` Used to invoke a syscall having no wrapper with its ID
`man _syscall` Macro used to make a system call (deprecated)

System Call (cont…)

- If a process is running a user program in user mode and needs a system service, such as reading data from a file, it has to execute a trap or system call instruction to transfer control to the OS. The OS then figures out what the calling process wants by inspecting the parameters. Then it carries out the system call and returns control to the instruction following the system call.
- Making a system call is similar to making a procedure call, difference is that system call enter the kernel code and procedure call do not.



System Call (cont…)



System Call (…)

Types of System Calls

- Process Control
  - End, abort
  - Load, execute
  - Create, terminate
  - Get/set process attributes
  - Allocate/de-allocate memory to processes
- File Management
  - Create, delete
  - Open, close
  - Read, write
  - Get/Set file attributes
- Information Maintenance
  - Get/Set date, time, or system data
  - Get/set process, file or device attributes
- Communication
  - Create/Delete communication connection
  - Send/Receive message
  - Attach/detach remote devices

All OS's offer their own System Calls

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

System Call vs Library Call

System Call

- Executed by OS kernel.
- Perform simple single operation.
- System calls usually return an integer:

int res = sys_call(some_args)

If return value >= 0 (OK)

If return value < 0 (Error)

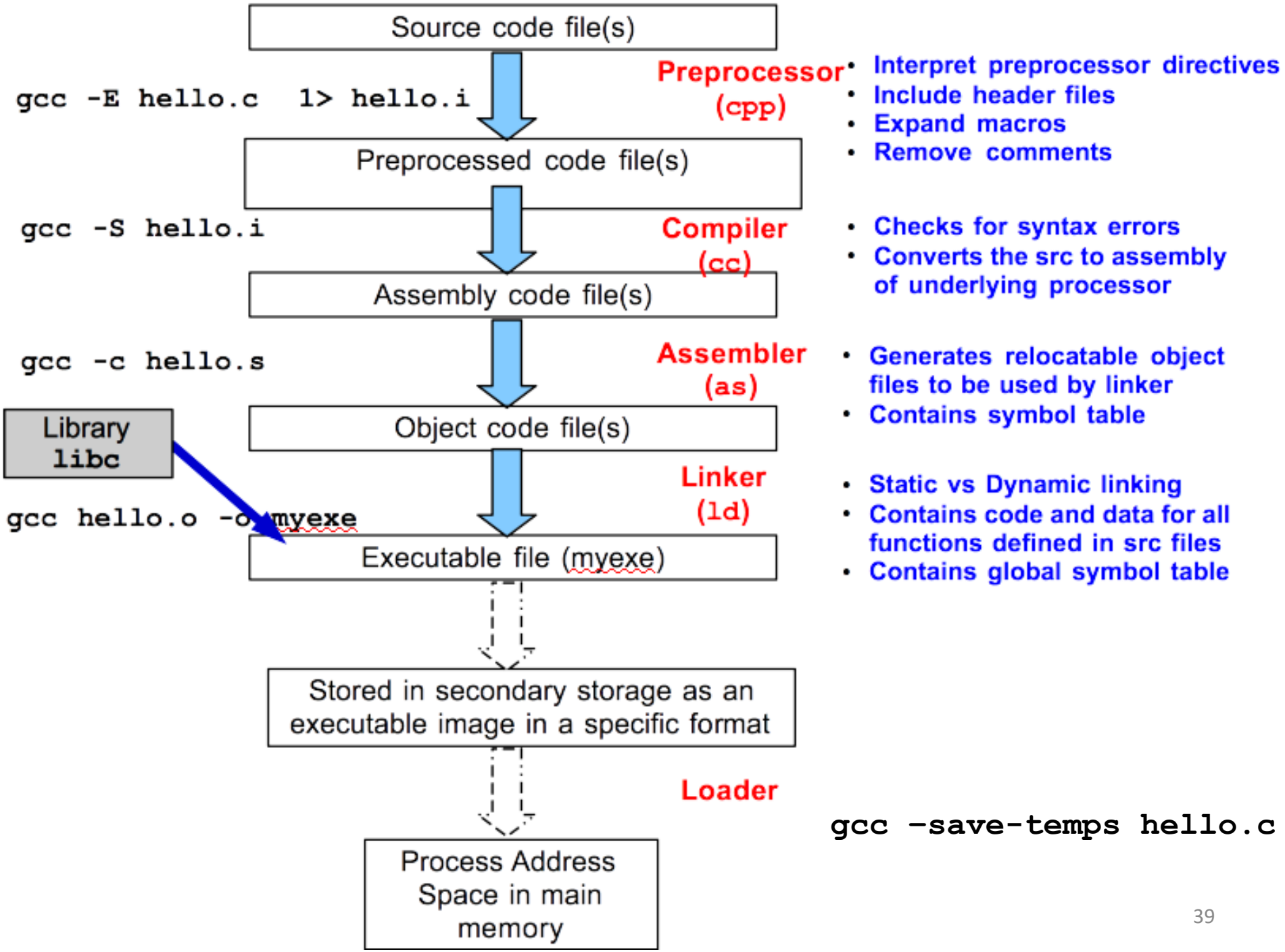
Library Call

- Executed in the User Program.
- May perform several tasks and may call system calls.
- Library functions often return pointers:

FILE * fp = fopen("file1", "r")

Return NULL for failure


LIVE FREE OR DIE
UNIX*
TRADEMARK OF BELL LABS*


Source code file(s)
gcc -E hello.c 1> hello.i
Preprocessor (cpp)
Preprocessed code file(s)
gcc -S hello.i
Compiler (cc)
Assembly code file(s)
gcc -c hello.s
Assembler (as)
Library libc
Object code file(s)
Linker (ld)
gcc hello.o -o myexe
Executable file (myexe)
Stored in secondary storage as an executable image in a specific format
Loader
Process Address Space in main memory
Interpret preprocessor directives
Include header files
Expand macros
Remove comments
Checks for syntax errors
Converts the src to assembly of underlying processor
Generates relocatable object files to be used by linker
Contains symbol table
Static vs Dynamic linking
Contains code and data for all functions defined in src files
Contains global symbol table
gcc -save-temps hello.c

Compiling and Running C Programs

```
#include <stdio.h>
int main(){
  printf("Hello World \n");
}
```

Use any editor to type your program and then to compile use gcc compiler:

$ gcc prog1.c

This will create an executable file **a.out** in the pwd. Now to execute the file

$ ./a.out

If you just type **$ a.out**, it will say a.out not found. So either use **./** before the **exe** name or add the current directory in the search path using following command

$ PATH=$PATH:.

Once you compile another program in the same directory, the executable name is again **a.out** which will overwrite the previous executable file. To over come this use **-o** flag when compiling your source file.

$ gcc prog1.c -o prog1

Now this will create an executable file with the name as the second argument i.e., prog1 in this case. To execute the file give following command.

$ ./prog1

SUMMARY

We're done for now, but Todo's for you after this lecture…

- **Go through the slides and Book Sections: 2.3, 2.8**
- **Go through Unix The Text Book Chapters: 3,4,5, 7**
- **Make your hands dirty by writing some basic C programs in UNIX using gcc compiler.**
  - A program that receives two command line arguments, a text file name and a string via command line parameters, opens that file, search the string and display the line(s) containing that string only. (See grep command)
  - A program that is passed a file name as command line parameter, it opens the file, encrypts its contents and saves the encrypted file in the same directory with another name. (cipher).
  - Make a decipher program also which do vice versa of above.

If you have problems (in finding drawbacks) visit me in counseling hours. . . .