Introduction to UNIX Operating System

What is UNIX?

- Unix is an Operating System.

Types of UNIX?

- Unix
- Red Hat Linux
- Mac OSX
- Sun Solaris
- GNU/Linux
- Ubantu

Features of UNIX

- Multiuser Capability
- Multitasking Capability
- Communication
- Security
- Portability
- Time-Sharing
- File-System

Features of UNIX …

- Shell
- Pipes and Filters
- Programming Facility
- Background Processing

Features of UNIX: Multiuser Capabilities

- In a multiuser system, the same computer resource are accessible to multiple users.
- A number of terminals are connected to the host computer (server).
- A terminal can be of any of the following types
  - Dumb Terminal
  - Terminal Emulation
  - Dial In Terminal

Features of UNIX: Multitasking Capabilities

- Capable of carrying out more than one job at some time.

Features of UNIX: Communication

- The communication within the network of a single main computer (Server) or between two or more such computer networks.

Features of UNIX: Security

- UNIX allows sharing of data but not indiscriminately.
- UNIX has an inherent provisions for protecting data by
  - Assigning password and login names to individual user.
  - Reserving read and write permission for yourself.
  - File encryption utility encodes your file into an unreadable format.

Features of UNIX: Portability

- UNIX is ported in almost any computer system.

Features of UNIX: Time Sharing

- In UNIX more than one user can use the computer at the same time and more than task can be given to the system by a single user.

Features of UNIX: File System

- Everything in a UNIX system is a file.
- UNIX has a hierarchical file structure that allows the files to grow dynamically.

Features of UNIX: Shell

- The shell is the command interpreter which interacts with the kernel.

Features of UNIX: Pipes and Filters

- In UNIX the output of one command can be input of another command.
- Pipes are used to join the two commands.
- Filters are programs that perform simple transformation on data as it flows through.

Features of UNIX: Programming Facility

- UNIX was designed for programming.

Features of UNIX: Background Processing

- The background processing features allow UNIX to carry on a process in the foreground while keeping another waiting in the background.

Type of Terminals in UNIX

- Dumb Terminal
- Terminal Emulation
- Dial In Terminal

Comparison between UNIX and Windows

Factor	Linux	Windows
Open Source	Open Source and is Free to Use	Not Open Source and is Not Free to Use.
Case Sensitivity	File System is Case Sensitive	File System is Case Insensitive
Kernel Type	Monolithic Kernel	Micro Kernel

Comparison between UNIX and Windows

Factor	Linux	Windows
Efficiency	More Efficient	Less Efficient
Path Separator	Forward Slash as Path Separator between Directories.	Backward Slash as Path Separator between Directories
Security	Highly Secure	Less Secure

Type of Terminals in UNIX: Dumb Terminal

- The terminal is having the capability for only I/O and having no capability for processing.
- Terminals like keyboard, monitor etc.

Type of Terminals in UNIX: Terminal Emulation

- PC has own microprocessor, memory and dis drives.
- When this PC is connected with host computer, we can emulate the host.

Type of Terminals in UNIX: Dial In Terminal

- The terminals that use telephone lines to connect with the host.

UNIX Commands

- The UNIX system is command-based i.e. things happen because of the commands that you key in.

Types of UNIX Commands

- Internal Commands
- External Commands

Types of UNIX Commands: Internal Commands

- Commands which are built into the shell.
- For all the shell built-in commands, execution of the same is fast in the sense that the shell doesn't have to search the given path for them in the PATH variable, and also no process needs to be spawned for executing it.
- Examples are cd, fg etc.

Types of UNIX Commands: Internal Commands

- Commands which aren't built into the shell.
- When an external command has to be executed, the shell looks for its path given in the PATH variable, and also a new process has to be spawned and the command gets executed.
- These are usually located in /bin or /usr/bin.
- For example, when you execute the "cat" command, which usually is at /usr/bin, the executable /usr/bin/cat gets executed.
- The examples are ls, cat etc.

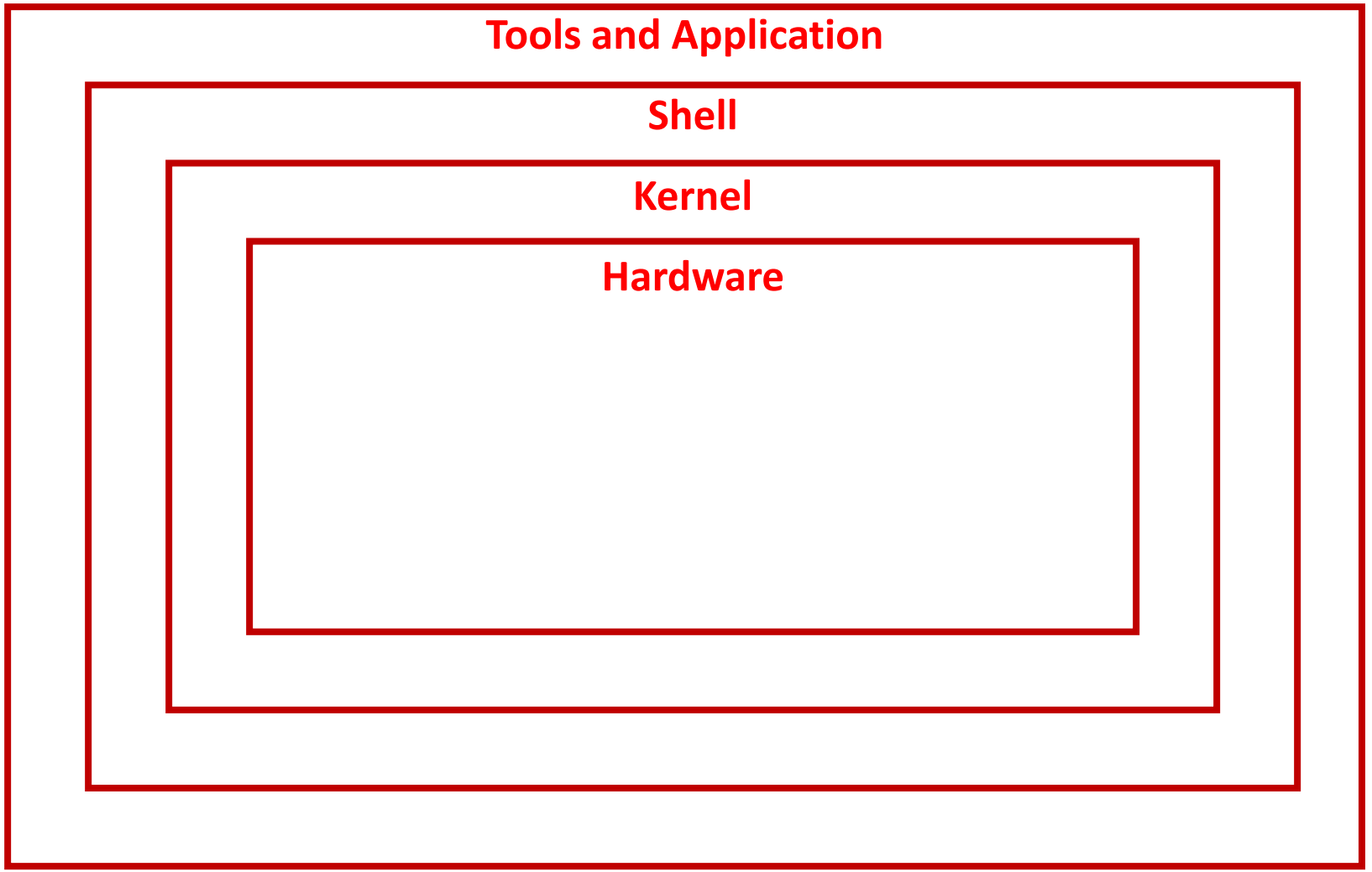
Types of UNIX Commands: Internal Commands

- UNIX Command “ls”
- Since ls is a program or file having an independent existence in the /bin directory(or /usr/bin).
- It is an external command i.e. “ls” command is not built into the shell and these are executables present in a separate file.
- In other words, when you will key in the ls command, to be executed it will be found in /bin.

UNIX System Structure

- The components of a UNIX OS are: -
  - Kernel
  - Shell
  - Application Software

UNIX System Organization



UNIX System Structure: Kernel

- The kernel is the lowest layer of the UNIX system.
- It is the part of the operating system that interacts directly with the hardware of the computer.
- It acts as an interface between the shell and hardware.

UNIX System Structure: Shell

- The shell is the middle layer of the UNIX operating system.
- The shell is a command line interpreter that enables the user to interact with the kernel.
- The shell acts as an interface between the user and the kernel.

UNIX System Structure: Application Programs

- Application software is the outer-most layer of the UNIX OS.
- It is a program that performs specific type of task.
- It includes utilities and application programs.
- The user application program interacts with the kernel through a set of standard system calls.

UNIX Shells

- The Bourne Shell
- The C Shell
- The Korn Shell
- The Bash

UNIX Shells: Bourne Shell

- Primary shell
- All UNIX systems have Bourne Shell.
- The shell is small and fast.
- Features of The Bourne Shell
  - The scripts can be invoked as commands by using their file names.
  - May be used interactively or non interactively.
  - Supports I/O redirection and pipelines.

UNIX Shells: Bourne Shell …

- Features of The Bourne Shell …
  - Provides a set of built in commands.
  - Provides local and global variable scope.

UNIX Shells: C Shell

- Created by Bill Joy.
- Allows aliasing of commands i.e. we can decide what name we want to call a command by. Instead of typing the entire command we can simply use the short alias at the command line.
- C shell has a command history feature that helps in calling the previously typed command.

UNIX Shells: Korn Shell

- Developed by David Korn.
- Also termed as Ksh shell.
- Ksh is backwards compatible with the Bourne Shell and includes many features of C shell as well as a command history.
- May be used as a programming language.

Files and Processes in UNIX

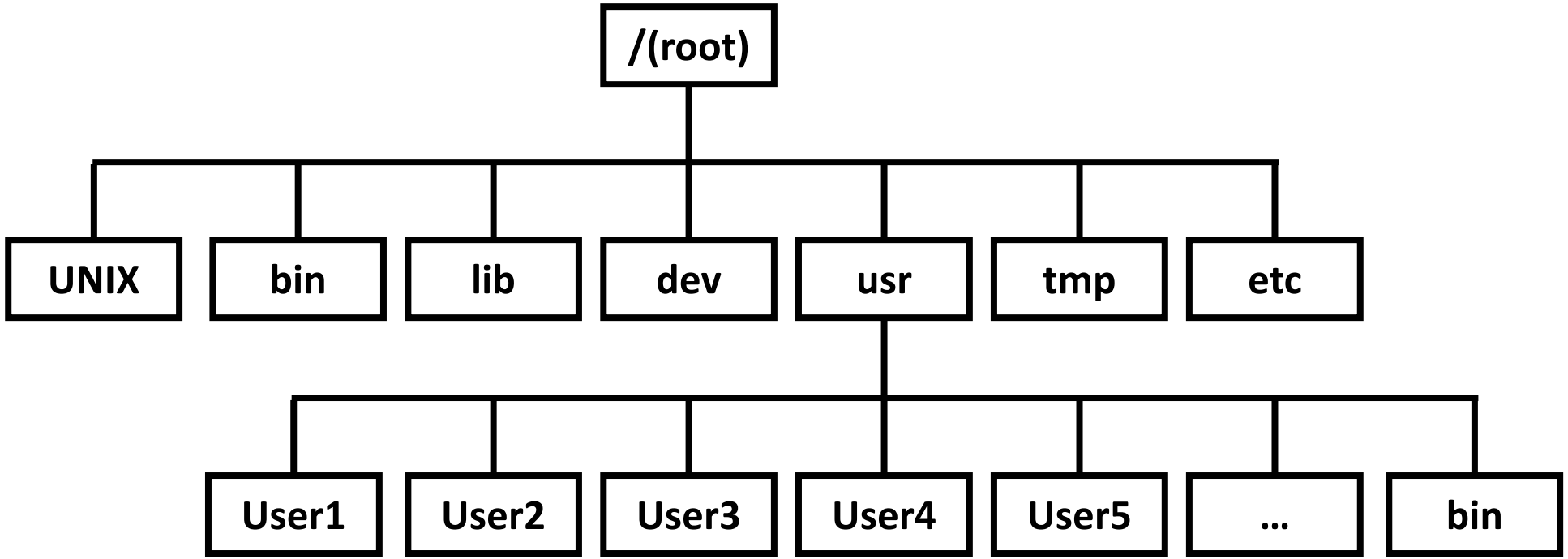
- Everything in UNIX is either a file or a process.
- A process is an executing program identified by a unique process ID.
- A file is a collection of data.
- The file is created by user using text editors, running compilers etc.

Directory Structure in UNIX

- All files are grouped in the directory structure.
- The file system is arranged in a hierarchical structure like an inverted tree.
- The top of the hierarchy is termed as root and is denoted as /(slash).
- The “root” directory contains a file called as UNIX which is nothing but the UNIX kernel.

Directory Structure in UNIX …

- The “bin” directory contains the executable files for most of the UNIX commands.



Directory Structure in UNIX …

- The “lib” directory contains all the library functions provided by the UNIX for the programmers.
- The “dev” directory contains all the files that control the various input / output devices like the terminals, printers, disk drivers etc.
- In “usr” directory, there are several directories, each associated with a particular user.
- The “tmp” directory contains the temporary files created by UNIX or by the users.

Process Management in UNIX

- Process is a program in execution i.e. a unit of work for Operating System.
- Apart of code of the program, a process has state snapshot of process execution which consists of procedure stack, program's static and dynamic data etc.
- To keep track of process states, the operating system maintains a structure called process control block.

Process Management in UNIX …

- Time sharing operating system interleaves execution of process giving users an illusion of simultaneous process execution.
- Operating System may cycle through all processes giving each a change to use CPU.
- Five state model introduces blocked state where process is waiting on some event to occur.

Process Management in UNIX …

- Process Creation
- Process Scheduling
- Scheduling Queue
- Types of Schedulers
- Context Switch

Process Management in UNIX: Process Creation

- A process can create another process.
- The original process is termed as the parent process.
- The new process is termed as child process.
- The child process is an (almost) identical copy of parent process (i.e. same code, same data etc.)
- The parent process can either wait for the child process to complete or continue executing in parallel (concurrently) with the child process.

Process Management in UNIX: Process Creation …

- In UNIX, a process creates a child process using the system call fork().
- In child process, the system call fork() returns 0.
- In parent process, the system call fork() returns process ID of the new child.
- Child often uses the system call exec() to start another completely different program.

Process States in UNIX

- CPU cycle can still be wasted in 5 state model.
- All processes in main memory can be blocked on input / output.
- Use virtual memory to admit more processes hoping that they will keep CPU busy.
- Blocked and ready state has to be split depending on whether a process is a swapped out on disk or in memory.

Process States in UNIX …

- Running state is also split depending on the mode kernel or user.
- Unix Process States are
  - Created: just created not ready to run.
  - Ready (memory): ready as soon as kernel schedules it.
  - Ready(disk): ready but needs to be swapped to memory.
  - Asleep – blocked (memory): waiting on event in memory.
  - Asleep – blocked(disk): waiting on event in disk.

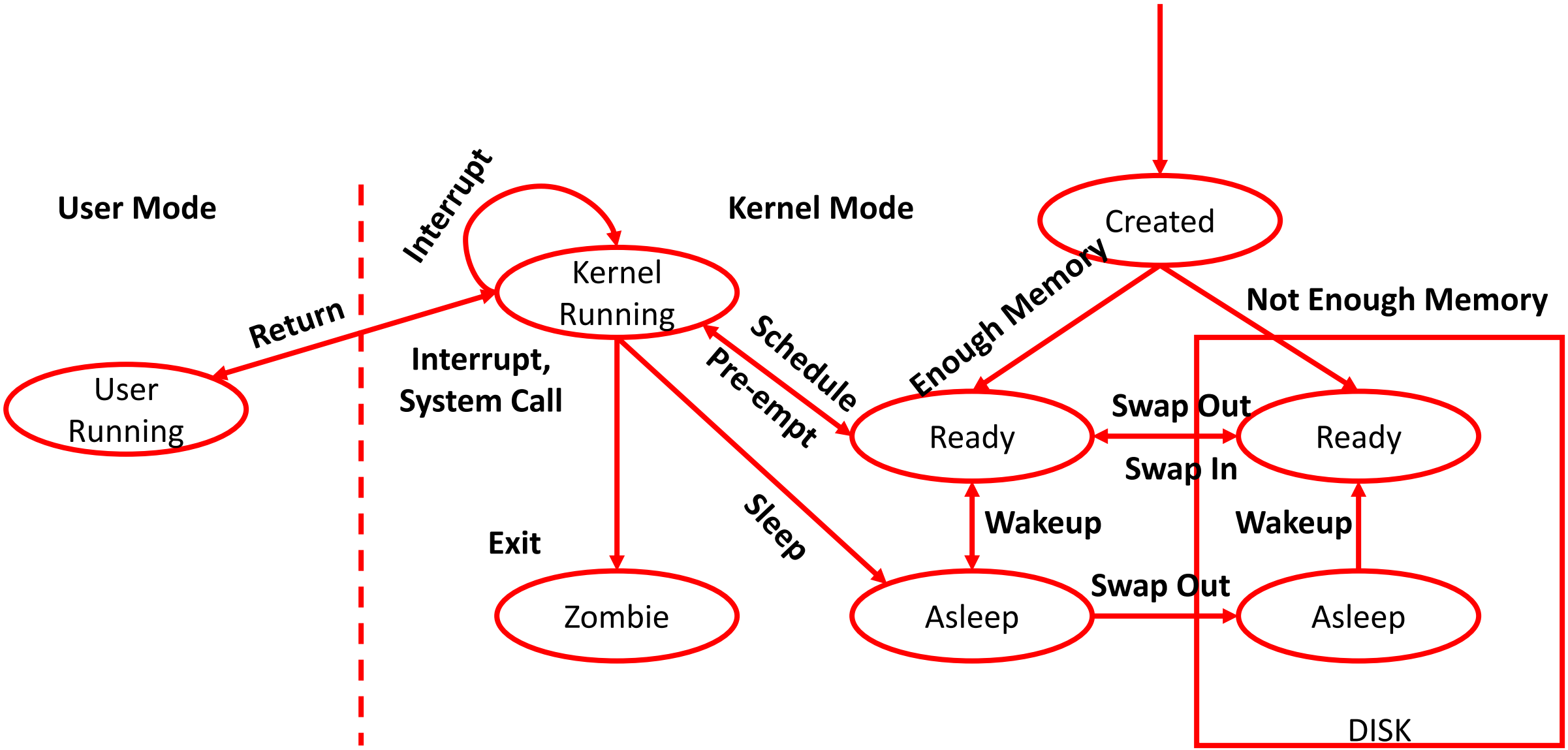
Process States in UNIX …

- Unix Process States are …
  - Running (Kernel): Executing in Kernel mode.
  - Running(User): Executing in user mode.
  - Zombie – process: Exited but left a record for parent to collect.

Process Scheduling in UNIX

- Process is running in user mode until an interrupt occurs or it executes a system call.
- If time slice expires, the process is pre – empted and another is scheduled.
- A process goes to sleep, if it needs to wait for some event to occur and is woken up when this event occurs.
- When process is created, decision is made whether to put it in memory or disk.

Process Scheduling in UNIX …



Process Scheduling Queue in UNIX

- Operating System organizes all waiting processes (and their PCBs) into a number of queues.
  - Queue for ready processes.
  - Queue for processes waiting on each device or type of event

Types of Schedulers in UNIX

- Long Term Schedulers
- Medium term Schedulers
- Short Term Schedulers

Types of Schedulers in UNIX: Long Term

- Job Scheduler.
- Selects job from spooled jobs and loads it into memory.
- Executes infrequently, may be only when process leaves system.
- Controls the degree of multiprogramming.
- Goal is the mixture of CPU bound and I/O bound processes.

Types of Schedulers in UNIX: Medium Term

- Does not really exist on modern time sharing systems.
- On time sharing systems, does some of what long term scheduler used to do.
- May swap processes out of memory temporarily.
- May suspend and resume processes.
- Balances load for better throughput.

Types of Schedulers in UNIX: Short Term

- CPU scheduler.
- Executes frequently, about one hundred times per second.
- Runs whenever Process is created or terminated.
- Process switches from running to blocked.
- Interrupt occurs.
- Selects process from those that are ready to execute and allocates CPU to that process.

Types of Schedulers in UNIX: Short Term …

- Goals
  - Minimize response time
  - Minimize variance of average response time predictability may be important.
  - Maximize throughput.
  - Minimize overhead i.e. operating system overhead, context switching etc.
  - Efficient use of resources.
  - Sharing CPU in equitable manner.

Types of Users in UNIX

- The four types of user in UNIX operating system are as follows:-
  - Root-User
  - File-Owner
  - Group-Owner
  - Other-User

Types of Users in UNIX: Root User

- The System Administrator is known as the root-user.

Types of Users in UNIX: File Owner

- The user who creates a file is known as the File – Owner.

Types of Users in UNIX: Group Owner

- A set of users that belong to a particular group is called Group – Owner.

Types of Users in UNIX: Other Users

- Users who are not members of any group are called other – user.

File Structure in UNIX

- The file-system in UNIX is organized in a tree-based structure known as hierarchical structure.
- The main features of UNIX File System are as follows:-
  - Hierarchical Structure
  - File Expansion
  - Security
  - File Sharing
  - File and Device Independence

File Structure in UNIX: Hierarchical Structure

- The UNIX file system is organized into a tree-based structure in which a root directory is present at the top of the directory tree.
- The root directory always contains sub-directories that have its own files and sub-directories.

File Structure in UNIX: File Extension

- File can grow dynamically, so users do not have to specify in advance the size of the file.

File Structure in UNIX: Security

- Files can be protected from unauthorized users by providing security at directory-level and file-level.

File Structure in UNIX: File Sharing

- A file can be accessed by several programs simultaneously.

File Structure in UNIX: File and Device Independence

- The key characteristics of UNIX file system is that everything is defined as a file.
- UNIX treats files and I/O-devices identically.

Type of File in UNIX

- There are three types of files in UNIX: -
  - Ordinary Files
  - Directory Files
  - Device Files

Type of File in UNIX: Ordinary Files

- Ordinary files are also known as regular files.
- These files contain source program, executable program and all UNIX programs as well as files created by the users.

Type of File in UNIX: Directory Files

- A directory is a specialized form of file that maintains the file system based on the specific use of that directory.

Type of File in UNIX: Device Files

- A device file is a special file that is associated with input/output devices.
- These files are stored in standard UNIX directories such as /dev and /etc.

Pipe in UNIX

- The symbol "|" is the UNIX pipe symbol that is used on the command line.
- The pipe symbol is used to place two or more unix commands at command line.
- The example is "mkdir a|cd a", where "mkdir" and "cd" are two unix commands and 'a' is the directory to be created and changed.

Pipe in UNIX …

- What it means is that the standard output of the command to the left of the pipe gets sent as standard input of the command to the right of the pipe.