MODULE - 1

Syllabus :
Introduction: .Unix Components/Architecture. Features of Unix. The UNIX Environment and UNIX Structure, Posix and Single Unix specification. General features of Unix commands/ command structure. Command arguments andoptions. Basic Unix commands such as echo, printf, ls, who, date, passwd, cal, Combining commands. Meaning of Internal and external commands. The type command: knowing the type of a command and locating it. The root login. Becoming the super user: su command.

Topics from chapter 2 , 3 and 15 of text book 1,chapter 1 from text book 2

Text Book 1: Sumitabha Das., Unix Concepts and Applications., 4th Edition., Tata McGraw Hill

Text Book 2: Behrouz A. Forouzan, Richard F. Gilberg : UNIX and Shell Programming- Cengage Learning – India Edition. 2009.

Introduction

An operating system is a control program for a computer that performs the following operations:

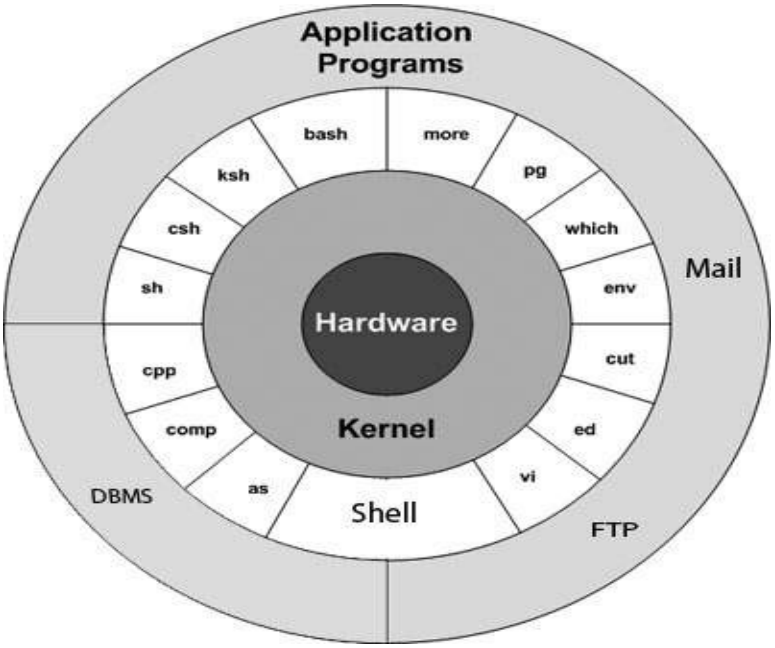
- allocates computer resources
- schedules routine tasks
- provides a platform to run application software for users to accomplish tasks
- provides an interface between the user & the computer

UNIX is an operating system which was first developed in the 1960s, and has been under constant development ever since. By operating system, we mean the suite of programs which make the computer work. It is a stable, multi-user, multi-tasking system for servers, desktops and laptops. UNIX systems also have a graphical user interface (GUI) similar to Microsoft Windows which provides an easy to use environment. However, knowledge of UNIX is required for operations which aren't covered by a graphical program, or for when there is no windows interface available, for example, in a telnet session. Unix was developed by Ken Thompson and Dennis Ritchie in AT & T Lab in 1969. Unix is basically Network Operating System generally used with Virtual Console (CLI). Everything is case sensitive in Unix including Usernames, commands, passwords and filenames.

Brief History

- The first versions of UNIX were written in "machine-dependent" program (such as PDP-7).
- Ken Thompson approach Dennis Ritchie developer of C program), and in 1973 they compiled UNIX in C programming language to make operating system "portable" to other computers systems.

Unix Components/UNIX Architecture



UNIX architecture comprises of two major components viz., the shell and the kernel. The kernel interacts with the machine's hardware and the shell with the user.

- The entire UNIX system is supported by a handful of essentially simple and abstract concepts.
- The success of UNIX, according to Thompson and Ritchie, "lies not so much in new inventions but rather in the full exploitation of a carefully selected fertile ideas, and especially in showing that they can be keys to the implementation of a small and yet powerful operating system".
- UNIX is no longer a small system, but it certainly is a powerful one.
- The UNIX architecture has three important agencies-
    - Division of labor: Kernel and shell
    - The file and process
    - The system calls

- ***Division of labor: Kernel and shell***
- The fertile ideas in the development of UNIX has two agencies – kernel and shell.
- The kernel interacts with the machine's hardware.
- The shell interacts with the user.

The Kernel

- The core of the operating system - a collection of routines mostly written in C.

- It is loaded into memory when the system is booted and communicates directly with the hardware.
- User programs (the applications) that need to access the hardware use the services of the kernel, which performs the job on the user's behalf.
- These programs access the kernel through a set of functions called system calls.
- Apart from providing support to user's program, kernel also does important housekeeping.
- It manages the system's memory, schedules processes, decides their priorities and so on.
- The kernel has to do a lot of this work even if no user program is running.
- The kernel is also called as the operating system - a programs gateway to the computer's resources.

The Shell

- Computers don't have any capability of translating commands into action.
- That requires a **command interpreter**, also called as the shell.
- Shell is actually interface between the user and the kernel.
- Most of the time, there's only one kernel running on the system, there could be several shells running – one for each user logged in.
- The shell accepts commands from user, if require rebuilds a user command, and finally communicates with the kernel to see that the command is executed.

Type of Shells

- Bourne shell (sh)
- C shell (csh)
- Korn shell (ksh)
- Bourne Again Shell (bash)

At one time only one shell runs.

Utilities

UNIX provides several hundred utility programs, often referred to as commands.

Accomplish universal functions –editing, file maintenance, printing, sorting, programming support, online info.

Features of UNIX OS

UNIX is an operating system, so it has all the features an operating system is expected to have.

- A Multiuser System
- A Multitasking System
- The building-block approach
- The UNIX toolkit
- Pattern Matching
- Programming Facility
- Documentation
- Portability
- Organized file system
- Device Independence
- Utilities

A Multiuser System

- UNIX is a multiprogramming system, it permits multiple programs to run and compete for the attention of the CPU.
- This can happen in two ways:
- Multiple users can run separate jobs
- A single user can also run multiple jobs

A Multitasking System

- A single user can also run multiple tasks concurrently.
- UNIX is a multitasking system.
- It is usual for a user to edit a file, print another one on the printer, send email to a friend and browse www - all without leaving any of applications.
- The kernel is designed to handle a user's multiple needs.
- In a multitasking environment, a user sees one job running in the foreground; the rest run in the background.
- User can switch jobs between background and foreground, suspend, or even terminate them.

The Building-block Approach

- The designer never attempted to pack too many features into a few tools.
- Instead, they felt "**small is beautiful**", and developed a few hundred commands each of which performed one simple job.
- UNIX offers the | (filters) to combine various simple tools to carry out complex jobs.
- Example:
    - **$ cat note** #cat displays the file contents WELCOME TO HIT

- **$ cat note | wc** #wc counts number of lines, words & characters in the file 1 3 15

The UNIX Toolkit

- Kernel itself doesn't do much useful task for users
- UNIX offers facility to add and remove many applications as and when remove many applications as and when required.
- Tools include general purpose tools, Text manipulation tools, Compilers, interpreters Networked applications and system administration tools.

Networking

While UNIX was developed to be an interactive, multiuser, multitasking system, networking is also incorporated into the heart of the operating system. Access to another system uses a standard communications protocol known as Transmission Control Protocol/Internet Protocol (TCP/IP).

Pattern Matching

- UNIX features very sophisticated pattern matching features.
- Example: The * (zero or more occurrences of characters) is a special character used by system to indicate that it can match a number of filenames.

Programming Facility

- The UNIX shell is also a programming language; it was designed for programmer, not for end user.
- It has all the necessary ingredients, like control structures, loops and variables, that establish powerful programming language.
- This features are used to design shell scripts – programs that can also invoke UNIX commands.
- Many of the system's functions can be controlled and automated by using these shell scripts.

Documentation

- The principal on-line help facility available is the man command, which remains the most important references for commands and their configuration files.
- Apart from the man documentation, there's a vast ocean of UNIX resources available on the Internet.

Portability

UNIX can be installed on many hardware platforms. Its widespread use can be traced to the decision to develop it using the C language.

Organized File System

UNIX has a very organized file and directory system that allows users to organize and maintain files.

Device Independence

UNIX treats input/output devices like ordinary files. The source or destination for file input and output is easily controlled through a UNIX design feature called redirection.

Utilities

UNIX provides a rich library of utilities that can be use to increase user productivity.

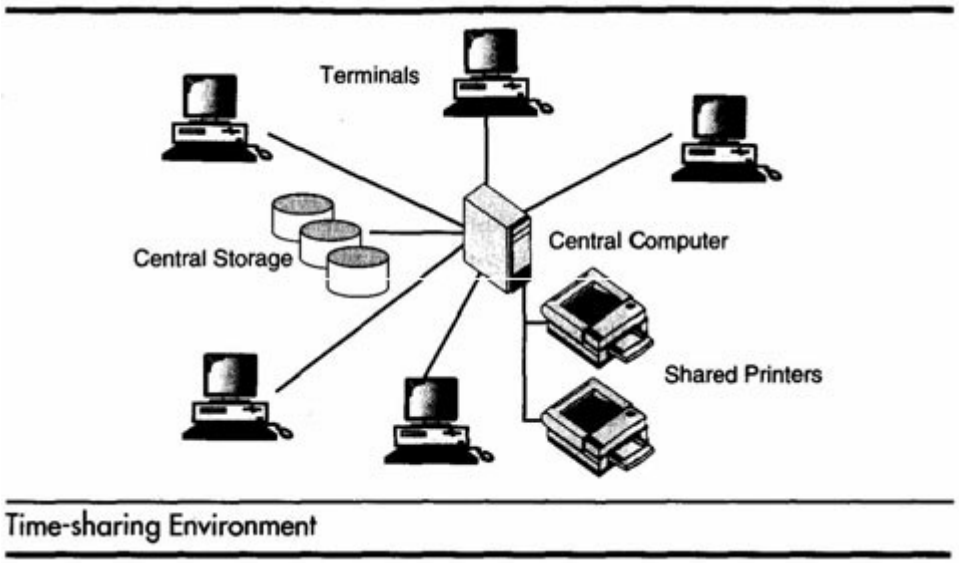
Unix Environment

Different Computing Environments
1. Stand alone Personal Environment
2. Time sharing Environment
3.Client/server system

Personal environment

With availability of Linux, a free Unix system personal unix system trend is accelerated

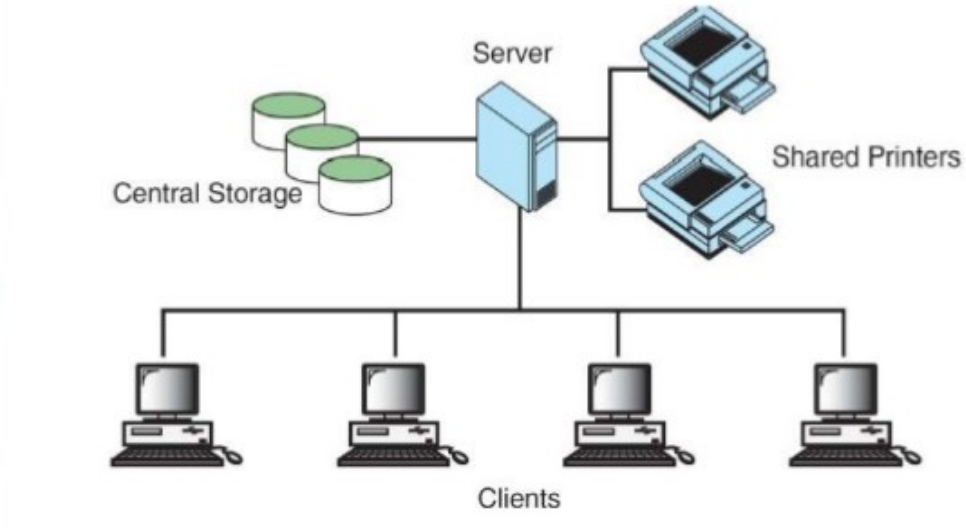
Time sharing environment



Time-sharing Environment

Many Users connected to one or more computers.
All computing must be done by the central computer which has many devices.

Central computer has to control shared resources, data and printers.
Client/Server Environment



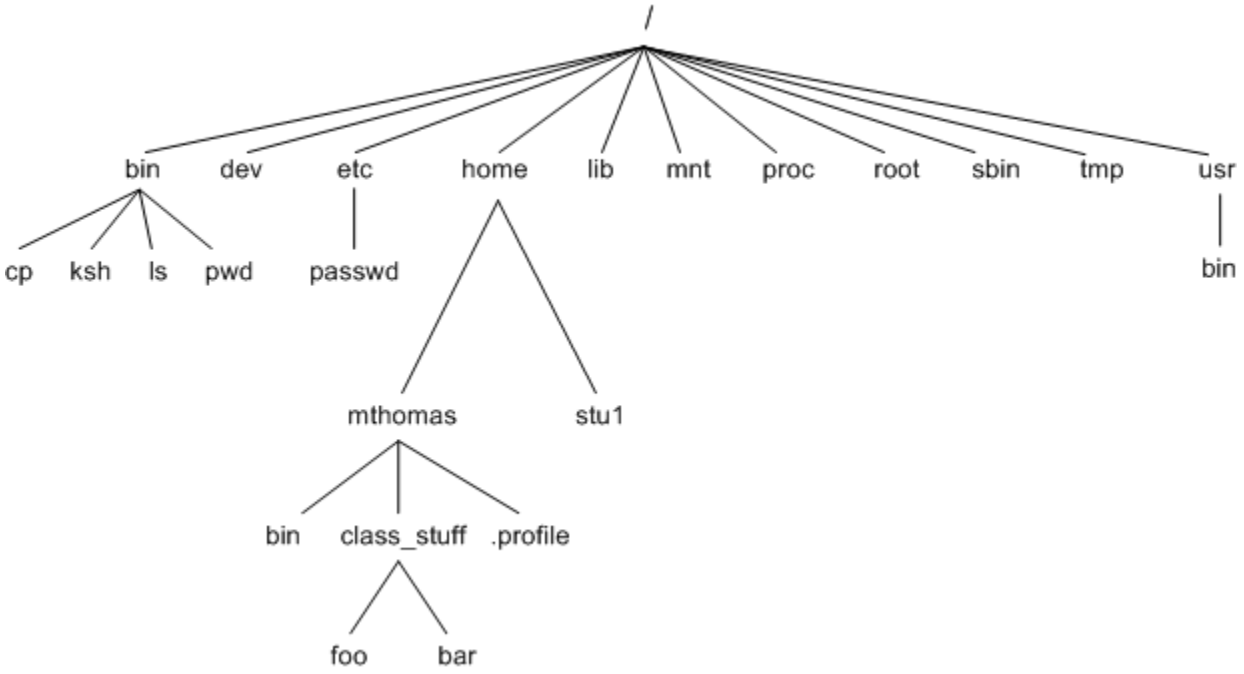
The computing function is split between a central computer(powerful mainframe) and users computers(Micro computers). Users use personal computers/workstations. The responsibility of central computer is moved from central computer and it is assigned to personal computers.

Central computer – server

Microcomputer - client

Advantage –Response time and monitor display are faster, users are more productive.

UNIX File Structure



The File and Process

"Files have places and processes have life".
All files are "flat": just a sequence of bytes File system is hierarchical.

A file is an array of bytes that stores information. It is also related to another file in the sense that both belong to a single hierarchical directory structure.

A process is the second abstraction UNIX provides. It can be treated as a time image of an executable file. Like files, processes also belong to a hierarchical structure. We will be discussing the processes in detain in a subsequent chapter.

Locating Files

All UNIX commands are single words like ls, cd, cat, etc. These names are in lowercase. These commands are essentially *files* containing programs, mainly written in C. Files are stored in directories, and so are the binaries associated with these commands. You can find the location of an executable program using type command:

```
$ type ls
ls is /bin/ls
```

This means that when you execute ls command, the shell locates this file in /bin directory and makes arrangements to execute it.

In bash shell ls is shellbuiltin

The Path

The sequence of directories that the shell searches to look for a command is specified in its own PATH variable. These directories are colon separated. When you issue a command, the shell searches this list in the sequence specified to locate and execute it.

POSIX and Single Unix Specification

POSIX:
Portable Operating System Interface for computing environments(POSIX) developed by IEEE. Two standards of POSIX family are
POSIX.1 : Deals with C application program interface – the system calls
POSIX.2 : Deals with shell and utilities

Single UNIX Specification:
Single Unix Specification version 3(SUSV3) is developed by X/Open and IEEE – Write once on any POSIX complaint UNIX system &adopt everywhere. Feature - Easy portability to any other POSIX complaint machine.

The Login Prompt
Login prompt indicates that the terminal is available to login (connect to machine) .Indiactes to user that he can enter with login name
login: kumar
Password:
System is now requesting for secrete code (Known only to you)
Login:kumar
Password:*******
The string entered by user as password is not displayed on screen. This is a security feature built into that hides password.
String entered in 1st part login prompt is – Login name or user-id or username
In 2nd part password prompt string entered by user is known as password.

Command Structure

UNIX commands take the following general form:

verb [options] [arguments]

where verb is the command name that can take a set of optional options and one or more optional arguments.
Commands, options and arguments have to be separated by spaces or tabs to enable the shell to interpret them as words. A contiguous string of spaces and tabs together is called a whitespace. The shell compresses multiple occurrences of whitespace into a single whitespace.
Unix arguments range from simple to complex. They consists of options, expressions, instructions, file names etc.

Options

An option is preceded by a minus sign (-) to distinguish it from filenames.
Example: $ ls –l

There must not be any whitespaces between – and l. Options are also arguments, but given a special name because they are predetermined. Options can be normally compined with only one – sign. i.e., instead of using

$ ls –l –a –t

we can as well use,

$ ls –lat

Because UNIX was developed by people who had their own ideas as to what options should look like, there will be variations in the options. Some commands use + as an option prefix instead of -.

Filename Arguments

Many UNIX commands use a filename as argument so that the command can take input from the file. If a command uses a filename as argument, it will usually be the last argument, after all options.

Example: cp file1 file2 file3 dest_dir
rm file1 file2 file3

The command with its options and argumens is known as the command line, which is considered as complete after *[Enter]* key is pressed, so that the entire line is fed to the shell as its input for interpretation and execution.

Exceptions

Some commands in UNIX like pwd do not take any options and arguments. Some commands like who may or may not be specified with arguments. The ls command can run without arguments (ls), with only options (ls –l), with only filenames (ls f1 f2), or using a combination of both (ls –l f1 f2). Some commands compulsorily take options (cut). Some commands like grep, sed can take an expression as an argument, or a set of instructions as argument.

UNIX provides flexibility in using the commands.

Entering a Command before previous command has finished

You need not have to wait for the previous command to finish before you can enter the next command. Subsequent commands entered at the keyboard are stored in a buffer (a temporary storage in memory) that is maintained by the kernel for all keyboard input. The next command will be passed on to the shell for interpretation after the previous command has completed its execution.

Basic Commands

echo, printf, ls, who, date, passwd, cal

echo command:(Displaying a message)

- It is an internal command – when you type echo shell won't look in its PATH to locate it. It will execute it from its own set of built in. It can be checked as follows

```
$type echo
echo is a shell builtin
```

- This message is often used in shell scripts to display diagnostic message on the terminal or to issue prompts for taking user inputs

Ex: 1.

```
$echo hello
hello
$_
```

Ex 2.

```
$echo "enter filename:  \c":
Enter filename: $_
```

- An escape sequence is generally a two character string beginning with a \(backslash)
- Bash interpret the escape sequence only when echo is used with **the –e** option

Escape sequences

- \a Bell
- \b backspace
- \c no new line
- \f formfeed
- \n newline
- \r carriage return
- \t tab
- \v vertical tab
- \\ backslash
-

printf command : An alternate to echo

It exists as an external command but only in bash shell it is built in

Usage

```
$printf  " Good Morning\n"
Good Morning
$_
```

It also accepts escape sequence and formatted strings

```
$printf" My Shell is %s\n" $SHELL
My Shell is /usr/bin/bash
```

- $_

Formats

%s string

%30s print in a space 30 characters

%d decimal integer

%6d print in a space 6 characters

%o octal integer

%x hexadecimal integer

%f floating point number

Ex.

$ printf “the value of 255 is %o in octal and %x in hexadecimal \n” 255 255

the value of 255 is 377 in octal and FF in hexadecimal

- Format strings can convert data from one form to another
- The %o and %x format strings are also used by awk and perl to convert a decimal integer to octal and hex, respectively

ls command : listing directory contents

- Use to obtain a list of all filename in the current directory

Syntax:

ls [options] [argument]

- Ex: ls

ls options

-a Lists all files, including those beginning with a dot (.).
-d Lists only names of directories, not the files in the directory
-F Indicates type of entry with a trailing symbol: executables with *, directories with / and symbolic links with @
-R Recursive list
- r sorts filesname in reverse order
-u Sorts filenames by last access time
-t Sorts filenames by last modification time
-i Displays inode number
-l Long listing: lists the mode, link information, owner, size, last modification (time).
-x output in multiple columns
-d dirname

who command : who are the users?

- Unix maintains an account of all users who are logged on to the system

Who command displays an informative listing of there users

- $ who

root console **aug 1 07:51** (:0)
kumarpts/10 **aug 1 07:56** (pc123.heavens.com)
Sharmapts/6 **aug 1 02:10** (pc123.heavens.com)

- User name or userid’s
- Device name of their there respectively terminals
- Logging time
- Machine name from where the user logged in
- Who

-H display with header information
-u prints coloumn header

- Who am i

kumarpts/10 **aug 1 07:56** (pc123.heavens.com)

date command: Displaying the system date

- The unix system maintains an internal clock. When the system is shut down, a battery backup keeps the clock ticking
- You can display the current date with the date command
- $date

Tue Aug 16 10:30:40 IST 2016

- $date +%m
  08
- $date +%h
  Aug
- $date +"%h%m"

Aug 08

Options

- d - day of the month
- y - last two digit of the year
- H, M, S - hour, min and sec
- D- date in mm/dd/yy format
- T- time in format hh:mm:ss

Note: When you use multiple format specifiers, you must enclose them within quotes and use a single + symbol before it

passwd command- changing your password

- $passwd

passwd: changing password for kumar
Enter login password:********
New password:*******
Re-enter New passowrd:*******
passwd(SYSTEM): Password successfully
changed for kumar

- Encryption is stored in a file named shadow in the /etc directory

cal command – The calendar

- cal can be used to see the calendar of any specific month or a complete year
- Syntax

cal [[month] year]
Every thing in rectangular bracket are optional
cal can be used without arguments
Ex 1.

$ cal
Aug 2016
montue…

Ex 2.

$ cal 03 2016
Ctrl-s to make it pause

Combining Commands

Instead of executing commands on separate lines, where each command is processed and executed before the next could be entered, UNIX allows you to specify more than one command in the single command line. Each command has to be separated from the other by a ; (semicolon).

wc sample.txt ; ls –l sample.txt

You can even group several commands together so that their combined output is redirected to a file.

(wc sample.txt ; ls –l sample.txt) >newfile

When a command line contains a semicolon, the shell understands that the command on each side of it needs to be processed separately. Here ; is known as a metacharacter.

Note: When a command overflows into the next line or needs to be split into multiple lines, just press enter, so that the secondary prompt (normally >) is displayed and you can enter the remaining part of the command on the next line.

Meaning of Internal and External Commands

Some commands are implemented as part of the shell itself rather than separate executable files. Such commands that are built-in are called internal commands. If a command exists both as an internal command of the shell as well as an external one (in /bin or /usr/bin), the shell will accord top priority to its own internal command with the same name. Some built-in commands are echo, pwd, etc.

type command:knowing the type of command and locating it

Ex1.
$type cat
cat is /bin/ls

Ex 2.
$type echo
echo is a shell builtin

Root login

The system administrator also known as superuser or root user.
The job of system administration involves the management of the entire system- ranging from maintaining user accounts, security and managing disk space to performing backups.
Root: The system administrator’s login

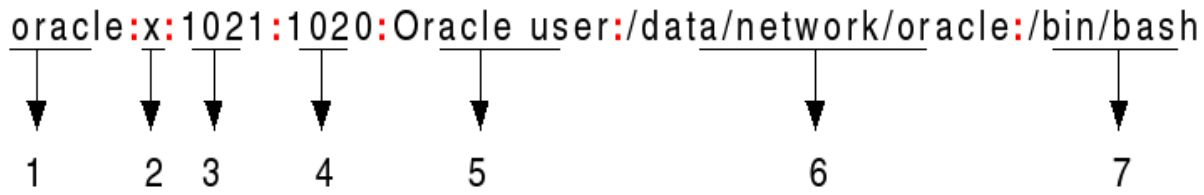
The unix system provides a special login name for the exclusive use of the administrator; it is called root.
This account doesn't need to be separately created but comes with every system.
Its password is generally set at the time of installation of the system and has to be used on logging in:
login: root
Password: ********* [Enter]
-
The prompt of root is #.
Once you login as a root you are placed in root's home directory.
Depending on the system, this directory could be / or /root.
Administrative commands are resident in /sbin and /usr/sbin in modern systems and in older system it resides in /etc.
Roots PATH list includes detailed path
, for example: /sbin:/bin:/usr/sbin:/usr/bin:/usr/dt/bin
Becoming the super user,
su command. :
su: Acquiring superuser status Any user can acquire superuser status with the su command if they know the root password.
For example, the user abc becomes a superuser in this way.
$ su
Password: **********
#pwd /home/abc
Though the current directory doesn't change,
the # prompt indicates that abc now has powers of a superuser. To be in root's home directory on superuser login, use su –l.

/etc/passwd
/etc/passwd file stores essential information, which is required during login
i.e. user account information.

/etc/passwd is a text file –
contains a list of the system's accounts, giving for Account has some information - user ID, group ID, home directory, shell, etc.

It should have general read permission as many utilities like ls use it to map user IDs to user names, but write access only for the superuser/root account.

Understanding fields in /etc/passwd
The /etc/passwd contains one entry per line for each user (or user account) of the system. All fields are separated by a colon (:) symbol.
Total seven fields as follows. Generally, passwd file entry looks as follows



- **Username**: It is used when user logs in. It should be between 1 and 32 characters in length.
- **Password**: An x character indicates that encrypted password is stored in /etc/shadow file. Please note that you need to use the passwd command to computes the hash of a password typed at the CLI or to store/update the hash of the password in /etc/shadow file.
- **User ID (UID)**: Each user must be assigned a user ID (UID). UID 0 (zero) is reserved for root and UIDs 1-99 are reserved for other predefined accounts. Further UID 100-999 are reserved by system for administrative and system accounts/groups.
- **Group ID (GID)**: The primary group ID (stored in /etc/group file)
- **User ID Info**: The comment field. It allow you to add extra information about the users such as user's full name, phone number etc. This field use by finger command.
- **Home directory**: The absolute path to the directory the user will be in when they log in. If this directory does not exists then users directory becomes /
- **Command/shell**: The absolute path of a command or shell (/bin/bash). Typically, this is a shell. Please note that it does not have to be a shell.

/etc/shadow

/etc/shadow file stores actual password in encrypted format (more like the hash of the password) for user's account with additional properties related to user password.

It stores secure user account information.
All fields are separated by a colon (:) symbol.
It contains one entry per line for each user listed in /etc/passwd file.
Generally, shadow file entry looks as follows

- **Username** : It is your login name.
- **Password** : It is your encrypted password. The password should be minimum 8-12 characters long including special characters, digits, lower case alphabetic and more. Usually password format is set to idsalt$hashed, The $id is the algorithm used On GNU/Linux as follows:
  - **1** is MD5

  - **$2a$** is Blowfish
  - **$2y$** is Blowfish
  - **5** is SHA-256
  - **6** is SHA-512
- **Last password change (lastchanged)** : Days since Jan 1, 1970 that password was last changed
- **Minimum** : The minimum number of days required between password changes i.e. the number of days left before the user is allowed to change his/her password
- **Maximum** : The maximum number of days the password is valid (after that user is forced to change his/her password)
- **Warn** : The number of days before password is to expire that user is warned that his/her password must be changed
- **Inactive** : The number of days after password expires that account is disabled
- **Expire** : days since Jan 1, 1970 that account is disabled i.e. an absolute date specifying when the login may no longer be used.

Commands to add, modify and delete users

User – Not only a person , it can represent a project or an application as well.
Unix provides 3 commands useradd, usermod and userdel
Groupadd: Adding group
Groupadd –g 241 dba 241 is the GID for dba
Command places this entry in /etc/group which you can also insert manually.
Dba:x:241:
Once the group has been made, an user can be added to group to the system.

useradd – Adding a user
create a new user or update default new user information

Tag	Description
useradd [*options*] *LOGIN* **useradd** -D **useradd** -D [*options*]	

When invoked without the **-D** option, the **useradd** command creates a new user account using the values specified on the command line and the default values from the system. Depending on command line options, the useradd command will update system files and may also create the new user's home directory and copy initial files

#useradd –u 210 –g dba -c "THE RDBMS" –d /home/oracle –s /bin/ksh –m oracle
Creates user oracle with UID of 210 and group name dba. The home directory is /home/oracle and the user will use the Korn shell

usermod and userdel: Modifying and removing users

usermod is used for modifying some of the parameters set with useradd. Users sometimes need to change their login shell

Ex
usermod –s /bin/bash oracle

Users are removed from the system with userdel. The following command removes the user oracle from the system

userdel oracle Does not delete user's files

This removes all entries pertaining to oracle from /etc/passwd,/etc/group and /etc/shadow. The users home directory does not get deleted in the process and has to be removed separately if needed.

- Source: Sumitabha Das, "UNIX – Concepts and Applications", 4th edition, Tata McGraw Hill, 2006