Module 4

System Administration

Administration & Administrator

- The key person in an organization is the system administrator.
- System administrators are engaged in planning activities as per the organizational technical environment.
- They are also in charge of supporting and maintaining servers and desktops and monitoring the network.
- They should have good knowledge of programming also as sometimes scripting or the programming of systems is also required.
- The prime role is user administrator.
- To maintain smooth running of the system.
- To check that the peripherals are working.
- To manage hardware and arrange timely repair of hardware in case of any failure.

- To monitor system performance to the utmost.
- To create file systems as per the organizational requirements.
- To install software as per the organizational requirements.
- To formulate and administer the backup and recover policy to counter any crisis or data loss.
- To manage network communication for better coordination among users.
- To keep updating systems as soon as a new version of OS or application software is released.
- To implemented security policies for use of the computer system and network.
- To formulate security policies for users. A system administrator should a strong grasp of computer security (Eg., firewalls and intrusion detection systems).

Qualities of a good administrator

- Should be able to update themselves about the latest developments in the field, such that new technologies that will save time and money are implemented.
- Should take conscious efforts for making security of the network and data as a primary objective.
- Should aware of the various tools to identify and monitor potential problems, which even include purchasing of expensive equipment with remote monitoring ability for efficient network operations.
- Should be capable of dealing with requests from users and explaining problems to them in non-technical manner.

- Should be capable enough to adopt a sensible network policy to control network use, keeping in mind that it should not impinge users in operation.
- Should be competent enough with multi-environment.
- Should have adequate understanding of new technologies, regarding how to implement them into the existing network and to design a capable solution for the betterment of the organization.

Administrative Commands

- /bin – Contains commands for modifying our disk partitions (such as fdisk), changing boot procedure (grub), and changing system states (init).
- /usr/sbin – contains commands for managing user accounts (such as useradd) and configuring our mouse (mouseconfig) or keyboard (kbdconfig). Many commands that are run as daemon processes are also contained in this directory.
- Many administrative commands are contained in regular directories (such as /bin and /usr/bin).

Configuration Files

- Configuration files are another mainstay of Linux administration.
- Almost everything we set up for our particular computer – user accounts, network address, or GUI preferences is stored in plain text files.
- Configuration files are stored in these locations:
- $HOME – All users store information in their home directories that directs how their login accounts behave. Most configuration files starts with dot(.), so they don’t appear as a users directory when u use ls command. There are dot files that define how each users shell behaves, the look and feel of the desktop, and what options are used with our text editor.

- /etc – Contains most of the basic Linux system configuration files. Following are the major:
  - adjtime – Holds to data to adjust the hardware clock.
  - aliases – contain distribution lists used by the Linux mail service.
  - Bashrc – Sets system wide defaults for bash shell users.
  - Cdrecord.conf – contains defaults used for recording CDs.
  - Crontab – sets cron environment and times for running automated tasks.
  - Csh.cshrc, exports, fdprm, fstab, ftp*, group, gshadow, host.conf, hosts, hosts.allow, hosts.deny, info-dir etc

- /etc/x11 – contains subdirectories that each contain system wide configuration files used by X and different X window managers available with Linux.
- /etc/alternative
- /etc/Amanda
- /etc/cipe
- /etc/cron*
- /etc/cups
- /etc/default
- /etc/httpd
- /etc/init.d
- /etc/mail
- /etc/pcmcia
- /etc/ppp
- /etc/sysconfig etc

Log Files

- Linux keeps track of it self using log files.
- General system logging is done by syslogd
- Logging is done according to the information in the/etc/syslog.conf.
- Messages are directed to log files in /var/log directory.
- To View Log files
- Go to /var/logs directory: for that use cd
- # cd /var/logs

- /var/log/message: General message and system related stuff
- /var/log/auth.log: Authenication logs
- /var/log/kern.log: Kernel logs
- /var/log/cron.log: Crond logs (cron job)
- /var/log/maillog: Mail server logs
- /var/log/qmail/ : Qmail log directory (more files inside this directory)
- /var/log/httpd/: Apache access and error logs directory
- /var/log/lighttpd: Lighttpd access and error logs directory
- /var/log/boot.log : System boot log
- /var/log/mysqld.log: MySQL database server log file
- /var/log/secure: Authentication log
- /var/log/utmp or /var/log/wtmp : Login records file
- /var/log/yum.log: Yum log files

Log Files

- One integral part of any UNIX system are the logging facilities.
- The majority of logging in Linux is provided by two main programs, **syslogd** and **klogd**, the first providing logging services to programs and applications, the second providing logging capability to the Linux kernel.
- **Klogd** actually sends most messages to the syslogd facility but will on occasion pop up messages at the console (i.e. kernel panics).
- **Syslogd** actually handles the task of processing most messages and sending them to the appropriate file or device, this is configured from within **/etc/syslog.conf**.
- By default most logging to files takes place in **/var/log/.**

Managing User Accounts

Adding users

- Straight method for creating a new user from the shell is with the useradd command.
- After opening terminal with root permission, we simply invoke the useradd command at the command prompt, passing the details of the new account as parameters.
- The only parameter required to useradd is the login name of the user.

The passwd File

- Every account on the system has an entry in the file */etc/passwd*. This file contains entries, one line per user, which specifies several attributes for each account, such as the username, real name, and other informations. This file contains 7 semicolon-delimited fields.
- Each entry in this file is of the format:

  Username;password;uid;gid;gecos;homedir;shell
- *Username* - A unique character string, identifying the account. For personal accounts, this is the name the user logs in with. On most systems it is limited to eight alphanumeric characters--for example, larry or kirsten.

- Password - An encrypted representation of the user's password. This field is set using the passwd program to set the account's password; it uses a one-way encryption scheme that is difficult (but not impossible) to break.
- Uid - The user ID, a unique integer the system uses to identify the account. The system uses the uid field internally when dealing with process and file permissions; it's easier and more compact to deal with integers than byte strings. Therefore, both the uid and the username identify a particular account: the uid is more important to the system, while username is more convenient for humans.

- Gid - The group ID, an integer referring to the user's default group, found in the file /etc/group.
- Gecos - Miscellaneous information about the user, such as the user's real name, and optional "location information" such as the user's office address or phone number. Such programs as mail and finger use this information to identify users on the system.
- Homedir - The user's home directory, for his personal use. When the user first logs in, her shell finds its current working directory in the named homedir.
- *Shell* - The name of the program to run when the user logs in; in most cases, this is the full pathname of a shell, such as*/bin/bash* or */bin/tcsh*.

- Many of these fields are optional; the only required fields are *username*, *uid*, *gid*, and *homedir*. Most user accounts have all fields filled in, but "imaginary" or administrative accounts may use only a few.
- Note that as the system administrator, it's not usually necessary to modify the */etc/passwd* file directly. There are several programs available that can help you create and maintain user accounts;

Shadow Passwords

- Any user on the system may read the password file. To some extent, it is a security risk to let everybody with access to the system view the encrypted passwords in/etc/passwd.
- To overcome this potential security risk, shadow passwords have been invented.
- When shadow passwords are used, the password field in /etc/passwd contains only an x or a *, which can never occur in the encrypted version of a password.
- Instead, a second file called /etc/shadow is used. This file contains entries that look very similar to those in /etc/passwd, but contain the real encrypted password in the password field.
- /etc/shadow is readable only by root, so that normal users do not have access to the encrypted passwords.

The Group File

- Each file on the system has both a user and a group owner associated with it.
- Every user is assigned to at least one group, which you specify in the gid field of the /etc/passwd file.
- However, a user can be a member of multiple groups.
- The file /etc/group contains a one-line entry for each group on the system, very similar in nature to /etc/passwd. The format of this file is:

```
groupname:password:gid:members
```

- Here, groupname is a character string identifying the group; it is the group name printed when using commands such as ls -l.

- password is an optional password associated with the group, which allows users not in this group to access the group with the newgrp command.
- gid is the group ID used by the system to refer to the group; it is the number used in the gid field of /etc/passwd to specify a user's default group.
- members is a comma-separated list of usernames (with no whitespace in between), identifying those users who are members of this group

Creating Accounts

- Creating a user account manually requires the following steps:
  - Edit /etc/passwd with ***vipw*** and add a new line for the new account
  - Edit /etc/group with ***vigr***
  - Create the home directory of the user with ***mkdir*** command.
  - Copy the files from /etc/skel to the new home directory
  - Fix ownerships and permissions with chown and chmod command
  - Set the password with ***passwd*** command
- The password setting is done as the last step because the user may log in simply while we are still copying the files. After this is done, the new account is ready log in

Deleting and Disabling Accounts

- To delete an account, you must remove the user's entry in /etc/passwd, remove any references to the user in /etc/group, and delete the user's home directory, as well as any additional files created or owned by the user.
- For example, if the user has an incoming mailbox in /var/spool/mail, it must be deleted as well.
- The command userdel deletes an account and the account's home directory. For example:

```
userdel -r Norbert
```

- Will remove the account for norbert. The -r option forces the home directory to be removed as well. Other files associated with the user--for example, the incoming mailbox, crontab files, and so forth--must be removed by hand.

Temporarily disabling a user account

- Temporarily (or not-so-temporarily) disabling a user account, is even simpler. You can either remove the user's entry in */etc/passwd* (leaving the home directory and other files intact), or add an asterisk to the first character of the *password* field of the */etc/passwd* entry, as so:

aclark:*BjDf5hBysDsii:104:50:Anna Clark: /home/aclark: /bin/bash

- Chsh –s /usr/local/lib/no-login/security anuj
- chsh changes a user's login shell.
- Su – tester
- This account has been closed due to a security breach.

Modifying User Accounts

- Modifying attributes of user accounts and groups is usually a simple matter of editing /etc/passwd and /etc/group. Many systems provide commands such as usermod and groupmod to do this; it's often easier to edit the files by hand.
- There are a few commands for changing various properties of an account. They are:
  - chfn – change the full name field
  - chsh – change the login shell
  - passwd – change the password
- To change a user's password, use the passwd command, which will prompt for a password, encrypt it, and store the encrypted password in the /etc/passwd file.
- If you need to change the user ID of an existing account, you can do this by editing the uid field of /etc/passwd directly. However, you should also chown the files owned by the user to that of the new uid.

Changing Permissions and Ownerships

- -bash: cd: /root/: Permission denied
- That was one demonstration of Linux's security features. Linux, like UNIX, is a multi-user system and file permissions are one way the system protects against malicious tampering.
- All files and directories are "owned" by the person who created them. You created the file foo.txt in your login directory, so foo.txt belongs to you.
- Reading, writing, and executing are the three main settings in permissions.
- Chmod command

Creating and Mounting File system

- A file system is a combination of methods and data structures, which is also useful for any operating system to keep track of files on a disk or partition and states the way the files are organized on the disk.
- The file system is a process which starts much before a partition or disk can be used as a file system, prior to this the book keeping data structures need to be written to the disk.
- Central concept of file system.

Creating File System

- mkfs command is used to create file system.
- mkfs [-c] [-t fstype] device [size]
- mkfs is used to build a Linux filesystem on a device, usually a hard disk partition. The device argument is either the device name (e.g. /dev/hda1, /dev/sdb2), or a regular file that will contain the filesystem. The size argument is the number of blocks to be used for the file system.
- -c forces a check for bad blocks.
- -t fstype specifies the file system types.
- Ex:

  mkfs –t ext2 /dev/fd0 – make a ext2 file system

  on a floppy.

Mounting a File System

- The file system built on the floppy disk can be linked to the existing file system on the hard disk using the mount command.
- Once mounted we can create files and directories in the new file system and treat it as a normal directory existing in a file system.
- Like all file system, each mounted file system too has a root directory and all its directories fan out from the root.
- Mounting a file system , we are simply attaching its root directory to a particular point in the existing system.
- This point of attachment is called ‘mount point’ for that file system.

- Linux provides a default mount point called /mnt.
- #/etc/mount /dev/fd096ds15 /mnt
- /mnt is an empty directory in the (/) root directory, with root (superuser) as its owner.
- All users have the permission to access this directory.
- The **mount** command mounts a storage device or file system, making it accessible and attaching it to an existing directory structure.
- All files accessible in Linux, are arranged in one big tree: the file hierarchy, rooted at /. These files can be spread out over several devices. The mount command attaches a files system, located on some device or other, to the file tree.

- mount -t type device dir
- This tells the kernel to attach the file system found on **device** (which is of type **type**) at the directory **dir**.
- If only **directory** or **device** is given, for example:

  mount /dir
- then **mount** looks for a corresponding mountpoint entry in the **/etc/fstab** file, and attempts to mount it.

Unmounting a File System

- Unmount command can be used to unmounts any existing file system.
- On execution unmounts delinks the new file system from the root directory of the existing file system.
- The **umount** command "unmounts" a mounted filesystem, informing the system to complete any pending read or write operations, and safely detaching it.
- umount [-t vfstype] [-O options]
- /etc/unmount /mntsss

Checking and Monitoring System Performance

Checking System Space

- Running out of disk space on our computer is not a good situation.
- Using tools that comes with Linux we can keep track of how much disk space has been used on our computer.

Displaying system space with df

- We can display the space available in our file system using df command.
- $df
- $df –h
- -t , -x, -a, -l

Checking disk usage with du

- To find how much space is being consumed by a particular directory, we can use du command.

Finding Disk consumption with find

- The find command is used to find file consumption of our hard disk using a variety of criteria.
- We need to be root user to run this command.
- Eg:

```
find / -user jake –print  | ls –lds > /tmp/jake
```

- Find command searches the root file system(/) for any files owned by the user named jake and prints the filenames. The output of the find command is then listed with a long listing in size order. Finally that output is sent to the file /tmp/jake.
- `#find / -size 100k –print | ls –lds > /tmp/size`
- Instead of looking for a users files this command like looks for files that are larger than 100 kilobytes.

Monitoring CPU power

- CPU utilization statistics are:
- User versus System – the percentage of time spent performing user-level processing versus system level processing can point out whether a systems load is primarily due to running applications or due to operating system overheads.
- Context Switches – A context switch happens when the CPU stops running one process and starts running another. Since each context switch requires the os to take control of the cpu.
- Interrupts – interrupts are situations where the processing being performed by cpu abruptly changed. Interrupts are due to hardware or software. High interrupts rates lead to higher system level cpu consumption.

- Runnable processes – processes may be in different states:
  - Waiting for an I/O operation to complete
  - Waiting for the memory management subsystem to handle a page fault.

Number of Runnable processes.

Problems with I/o subsystem and memory

utilization statistics.

Monitoring CPU usage with top

- The **top** program provides a dynamic real-time view of a running system. It can display system summary information, as well as a list of processes or threads currently being managed by the kernel.
- Top command displays information about the system such as:
  - Uptime : The top command displays the time at which the system was started.
  - Processes: displays the number of users connected to the server. It also provides information on the load on the server.

– Load average: Three load average numbers are the average number of processes, which were ready to run and waiting their turn on the processor in the past 1, 5 and 15 minutes. The load average is a measure of the load on the system. Higher load averages means that the processor is unable to handle the demands of the tasks running on the system.
– CPU States: It displays the time allocated to users and system. It displays tasks whose nice levels have been changed. Also displays the idle time of the system.
– Mem: Displays statistics on memory usage, including total available memory, free memory, used memory, shared memory, and memory used for buffers.
– Swap: Displays statistics on swap space, including total swap space, available swap space, and used swap space.

- PID: Displays unique Process ID.
- PPID: Displays the parent process ID.
- UID: user Id of the tasks owner.
- USER: user name of the tasks owner
- PRI: Priority of the task.
- NI: Nice value of the task.
- STAT: Information about the status of the process. It can be following values
  - R: Runnable
  - S: Sleeping
  - D: Uninterruptible sleep
  - T: Stopped or traced
  - Z: Zombie process

- Commonly used commands with the top command
- Space bar – Updates value on the screen
- h or ? – displays help about various commands
- K – Kills a process
- I – ignores idle and zombie processes
- N – sets the number of the processes that should be displayed on the screen.
- Q – Quits the top command
- S – Modifies the interval in which the values of the top command are updated.

Monitoring memory utilization

- Page Ins/Page Outs – flow of pages from system memory to attached mass storage devices.
- Active/Inactive pages
- Free, shared, buffered, and catched pages
- Swap Ins/Swap Outs

File Security

- On a Linux system, every file is owned by a user and a group user.
- For each category of users, read, write and execute permissions can be granted or denied.

```
 marise:~> ls -l To_Do
-rw-rw-r--    1 marise  users      5 Jan 15 12:39 To_Do
marise:~> ls -l /bin/ls
-rwxr-xr-x    1 root    root   45948 Aug  9 15:01 /bin/ls*
```

- the first three characters in this series of nine display access rights for the actual user that owns the file. The next three are for the group owner of the file, the last three for other users. The permissions are always in the same order: read, write, execute for the user, the group and the others.

Code	Meaning
0 or -	The access right that is supposed to be on this place is not granted.
4 or r	read access is granted to the user category defined in this place
2 or w	write permission is granted to the user category defined in this place
1 or x	execute permission is granted to the user category defined in this place

Access mode codes

User group codes

Code	Meaning
u	user permissions
g	group permissions
o	permissions for others

- Chmod command

Su command

- The **su** command, which is short for *substitute user* or *switch user*, is used to become another user during a login session.
- If no username is specified, **su** defaults to becoming the superuser (root).
- To become superuser type su command then Enter key; when prompted for password supply root user password.

- $su
- Password:
- #
- To exit superuser status, type exit or press CTRL+D
- #exit
- >
- $

Getting system information using uname

- Print information about the current system.
- Print certain system information. If no *OPTION* is specified, **uname** assumes the **-s**

-a, **--all**	Prints all information, omitting **-p** and **-i** if the information is unknown.

-s, **--kernel-name**	Print the kernel name.
-n, **--nodename**	Print the network node hostname.
-r, **--kernel-release**	Print the kernel release.
-v, **--kernel-version**	Print the kernel version.
-m, **--machine**	Print the machine hardware name.
-p, **--processor**	Print the processor type
-i, **--hardware-platform**	Print the hardware platform, or.
-o, **--operating-system**	Print the operating system.
--help	Display a help message, and exit.
--version	Display version information, and exit.

- chown – change file owner and group
- Usage: chown [OPTION]... OWNER[:[GROUP]] FILE...
- eg. chown remo myfile.txt

Hostname command

- The **hostname** command shows or sets the system hostname.
- **hostname** is used to display the system's DNS name, and to display or set its hostname or NIS (Network Information Services) domain name.
- When called without any arguments, **hostname** will display the name of the system.
- When called with one argument or with the **--file** option, **hostname** will set the system's host name.
- he host name is usually set once at system startup in the script **/etc/init.d/hostname.sh**
- hostname

  Displays the hostname of the system.

fdisk command

- **fdisk** is a partition table manipulator for Linux.
- **fdisk** is a menu-driven program for creation and manipulation of partition tables.
- Hard disks can be divided into one or more logical disks called partitions. This division is recorded in the partition table, found in sector 0 of the disk.
- fdisk [-uc] [-b sectorsize] [-C cyls] [-H heads] [-S sects] device
- fdisk -l [-u] [device...]

-b*sectorsize*	Specify the sector size of the disk. Valid values are **512**, **1024**, **2048** or **4096**. Recent kernels know the sector size.
-c[**=***mode*]	Specify the compatiblity mode, '**dos**' or '**nondos**'. The default is non-DOS mode.
-C *cyls*	Specify the number of cylinders of the disk. This would be a very strange thing to want to do, but if so desired, this option will get it done.
-H *heads*	Specify the number of heads of the disk. Not the physical number, but the number used for partition tables. Reasonable values are **255** and **16**.
-S *sects*	Specify the number of sectors per track of the disk. Not the physical number, but the number used for partition tables. A reasonable value is **63**.
-h	Print help and then exit.
-l	List the partition tables for the specified devices and then exit. If no devices are given, those mentioned in **/proc/partitions** (if that exists) are used.
-u[**=***unit*]	When listing partition tables, show sizes in '**sectors**' or in '**cylinders**'. The default is to show sizes in sectors.
-v	Print version information, and exit.

Red Hat Package Manager(RPM)

- RPM, the Red Hat Package Manager, is a tool that automates the installation of software binaries and remembers what files are needed so that you can be assured the software will run properly.
- An RPM package consists of an archive of files and package information, including name, version, and the description.

Working with RPM

- rpm [options]
- Depending on the options that we select, we might have to specify the package name, the source RPM name, or a particular package file.
- In the basic level we will perform following activities with RPM:
  - Installing
  - Upgrading
  - Uninstalling
  - Querying
  - Verifying

Installing Packages

- RPM packages have file names typically as some_package-2.0-1.i386.rpm.
- Some_package is the name of package, 2.0 is the version, 1 is the release, and i386 is the architecture of the package.
- General form:
- Rpm –i[install options] <package_file>
- -i option used to install new packages.
- -v option used to verify files that are being installed or uninstalled.
- -h option used to print hash marks as the package is being installed.

- Example:
- #rpm –ivh taper-6.9b-3.i386.rpm
- Preparing…   ######################## [100%]
- l:taper   ######################## [100%]
- The taper package version 6.9b, release 3 for Intel platforms, is being installed.

- Before installing the packages, we can test if the package can be installed without any problem using –test option.
- # rpm –ivh taper-6.9b-3.i386.rpm –test
- Above command would display the # prompt with no error message if the package can be successfully installed.
- If package cannot be installed, an error message will be reported.

- If our machine contains insufficient space to install a particular package the rpm command will display an error message stating the amount of space that it expects in the file system on which it installs the package.
- # rpm –ivh taper-6.9b-3.i386.rpm
- Preparing…   ##################### [100 %]
- Installing package taper-6.9b-3 needs 1024Kb on the filesystem.

- # rpm –ivh taper-6.9b-3.i386.rpm
- Preparing…   ################## [100 %]
- Package taper-6.9b-3 is already installed.
- We can install package even it is already installed use –replacepkgs option
- # rpm –ivh taper-6.9b-3.i386.rpm -replacepkgs

- Some packages depends on other packages, which means that before install the required package, we must install all the packages depends.
- If we try to install
- #rpm –ivh mysql-server-3.23.36-1.i386.rpm
- Error: failed

  mysql=3.23.36 is needed by mysql-server-3.23.36-1

  libmysqlclient.so.10 is needed by mysql-server-3.23.36-1
- If we want to force the installation of the packages even though it depends on other packages, use the –nodeps option.
- #rpm –ivh mysql-server-3.23.36-1.i386.rpm -nodeps

Upgrading packages

- -U option used to upgrade.
- rpm -U[install-options]<package file name>
- Rpm automatically uninstalls the previous version of the software and installs the new versions.
- #rpm –Uvh telnet-0.7-18.i386.rpm
- We can use –U option to install new package.
- When upgrading packages, we may receive the following message:
- Saving /etc/some_package.conf as /etc/some_package.conf.rpmsave
- Configuration file formats are different.
- If we try to upgrade a package and RPM finds a newer version of the software installed on the system, error
- #rpm –Uvh-0.17-10.i386.rpm
- Package telnet-0.17-18 is already installed

Uninstalling packages

- The –e(erase) option is used with the rpm command to do uninstalling.
- #rpm –e <package name>
- Don’t need to specify the full package name.
- #rpm –e netscape-communicator

- When we uninstall packages, an error can occur, if some other package depends on it.
- Ex:
- #rpm –e tar
- Error: removing these packages would break dependencies:

  tar is needed by mkinittrd-3.0.10-1

  tar is needed by Amanda-2.4.2p2-1
- To ignore this error and uninstall the package use the –nodeps options