UNIT 12: SYSTEM ADMINISTRATION TASKS

UNIT STRUCTURE

12.1 Learning Objectives

12.2 Introduction

12.3 Getting Started With Your System

12.4 Users Account and Group

12.5 System Monitoring

12.6 Getting System Information With Uname

12.7 Backup and Restore Files

12.8 kudzu

12.9 X-windows

12.10 Gnome graphical interfaces

12.11 KDE Installation and Configuration

12.12 Let Us Sum Up

12.13 Further Readings

12.14 Answer to Check Your Progress

12.15 Model Questions

12.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- Understand the login process of Linux
- create, modify and remove users and groups
- learn system monitoring tools
- learn to navigate X windows, GNOME and KDE

12.2 INTRODUCTION

In the previous unit, we have learnt how to install and configure a printer in Linux. In this unit, we will discuss some common system administration task. The job of a Linux system administrator is to manage the operations of a computer system like to maintain, enhance, create user account/report, taking backups using Linux tools and command-line

interface tools. So, in this unit we will mainly focus on system administrative tasks like managing user accounts, granting access, checking system performance, and taking corrective measures. At first, we will go through the login process.

12.3 GETTING STARTED WITH YOUR SYSTEM

In Linux, the login page is handled by the GNOME Display Manager (GDM) or the Light. When we press the power button, an electrical signal passes through the system's motherboard and awakens the entire hardware system. As we all know, the Linux kernel works in a very different way than the traditional booting system. In Linux, the boot process includes a few steps. The booting process activates the kernel user mode that subsequently activates the BIOS, MBR, boot menu, GRUB, and the login page. When the power button gets ON, it powers up the BIOS, and then the BIOS starts looking for a boot device to run the operating system. BIOS then invokes GRUB (Grand unified bootloader) from MBR (Master boot records). MBR partition holds not only GRUB, but it also stores the boot records, the booting related file and data about all the other storage devices and how they will be acting on the Linux system. The Initial RAM Disk (intrd) is designed to be used as a temporary filesystem for a Linux system to find the mounted EFI boot files. In most Linux distributions, the bootloader files are stored inside the /boot/efi directory. When the bootloader is executed, a login stage appears to enter into the Linux system. When the kernel is first loaded into memory, it initializes itself and the hardware that may be attached to the computer. The run level state of a Linux system is defined as when our Linux system has finished the booting process and is ready to be used. Simply, that immediate state of a computer where the power options, user-mode option, and the entire environment can be operated is known as the run level state.

Once the kernel is established enough to be able to run programs, it does. The first program is called "init"; its job is to function as the grandparent of all processes.

When init starts, it reads a file called inittab, usually located in /etc. This file tells init which programs should be run under which conditions. Not only does init run the startup scripts that bring the rest of the system up, but init also takes care of shutting the system down. After that init starts getty process which in turn displays login prompt. The entire login process can be summarized as:

Login process steps

Init starts getty process

getty process initiates login prompt on terminal

login command check user credentials from /etc/passwd

getty starts user shell process

shell reads the system wide files /etc/profile, /etc/bashrc

Shell reads user specific files .profile, .login

Now it reads shell specific configuration file .bashrc

Shell displays the default prompt

Init starts getty

When Linux system boots, it goes through various booting stages. In last stage it starts init, which reads a file called inittab which is located in /etc, where it finds out in which run level it has to execute. Once init process completes run-level execution and executing commands in /etc/rc.local, it will start a process called getty. Getty is the process which will take care of complete login process.

Getty shows login prompt

getty is short for "get terminal". A getty program initiates login command, it opens the terminal device, initializes it, prints login: and waits for a user name to be entered.

getty starts /etc/login

Once user enters his login name, getty starts /etc/login, this in-turn will prompt for user password. The password will be hidden and will not be shown on screen.

getty verifies the credential and starts users shell

In next stage getty checks the user credentials by verifying with / etc/passwd and /etc/shadow file; if password matches then it initiates user

properties gathering and starts user's shell. If password doesn't match then getty terminates login process and re-initiates again with new login: prompt. In next stage, the getty process reads the user properties (username, UID, GID, home directory, user shell etc.) from /etc/passwd file. After that, it reads /etc/motd file for displaying content banner message.

Shell reads system wide default files and specific default files

For getting shell related properties, user aliases and variables, getty process reads appropriate system wide default files /etc/profile or /etc/bashrc. After the system wide default files are read, the shell reads user specific login files.profile or.login.

Shell specific file read

At last stage, it reads shell specific configuration files (.bashrc,.bash_profile etc. for bash shell) of that user which it gets on the users home directory.

Shell Prompt

When all startup files are read, the shell displays the default prompt, normally PS1 prompt for user to execute their commands.

12.4 USERS ACCOUNT AND GROUP

To add a new user account, we can run either of the following two commands as root.

- adduser
- useradd

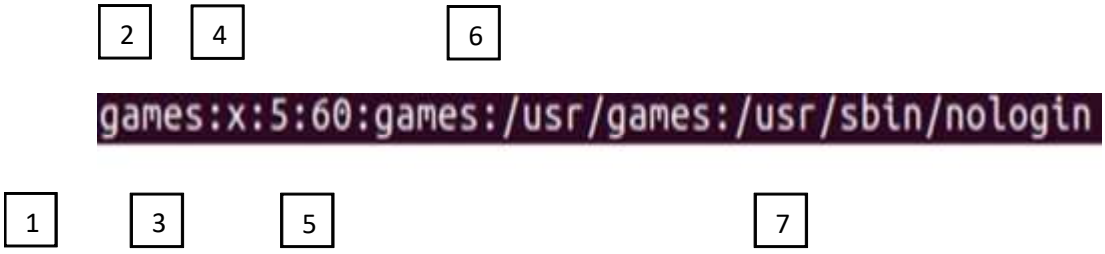
When a new user account is added to the system, the following operations are performed.

- His/her home directory is created (/home/username by default).
- The following hidden files are copied into the user's home directory, and will be used to provide environment variables for his/her user session.
- .bash_logout
- .bash_profile
- .bashrc

- A mail spool is created for the user at /var/spool/mail/username.
- A group is created and given the same name as the new user account.

Understanding /etc/passwd file format:

When user enters login ID and password, entries are verified against the information stored in /etc/passwd file. The entries in the /etc/passwd file have some information like username, userid, home directory and the first program that should be started after login succeeds.



- **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 we need to use the passwd command to compute 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 (GECOS):** As the comment field, it allows to add extra information about the users such as user's full name, phone number etc. This field is used 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 exist, then user's 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. For example, sysadmin can use the nologin shell, which acts as a

replacement shell for the user accounts. If shell sets to /sbin/nologin and the user tries to log in to the Linux system directly, the /sbin/nologin shell closes the connection.

So general format is like username: password: UID:GID: comment: home: shell

Understanding /etc/group

Group information is stored in the /etc/group file. Each record has the following format.

[Group name]:[Group password]:[GID]:[Group members]

- [Group name] is the name of group.
- An x in [Group password] indicates group passwords are not being used.
- [GID]: same as in /etc/passwd.
- [Group members]: a comma separated list of users who are members of [Group name].

Creating users

For this, we will be making use of the useradd command. This command is pretty flexible and allows us to create users that can login or even users that cannot login (in the case of creating a user for a software installation only).

The basic syntax of the command is:

useradd [options] username

Suppose we want to create the user "test" such that it has a home directory and can log in. The command will be

$sudo useradd test

where sudo (super user do) is the command which allows us to be the super user. The user would be added, without a home directory and be locked out of logging in. If we want to create the user and also create the user's home directory to match the username then following command will do the magic.

$sudo useradd -m test

So, if we look in the /home directory, we will now see test. Now to add password to the account, either one of two ways can be used. If we

have already created the user, we can issue the command:

$sudo passwd test

We will be prompted to enter and verify the new password. At this point, the user account will be unlocked, and they can login. If we want to do this all in a single step, that command would look like this:

$sudo useradd -m test -p PASSWORD

Where PASSWORD is the password, we want to use for the user test. Once the user logs in, they can change their password by using the passwd command, entering their current password, and then entering/verifying their new password.

Deleting User Account

For deleting user account, we will use userdel command.

The basic syntax of the command is: $userdel [option] username

When invoked, the command reads the content of the /etc/login.defs file. Properties defined in this file override the default behaviour of userdel. If USERGROUPS_ENAB is set to yes in this file, userdel deletes the group with the same name as the user, only if no other user is a member of this group. We will discuss about user group in next section. The userdel command removes the user entries from the /etc/passwd and /etc/shadow files. In most Linux distributions, when removing a user account with userdel, the user home and mail spool directories are not removed. We will use the -r (--remove) option to force userdel to remove the user's home directory and mail spool. The command will look like $userdel -r username. Suppose we want to delete recently created test account, then command will be

$sudo userdel -r test

The user account named "test" creation with password "PASSWORD" and thereafter deleting it are shown here.



Groups make it easy to associate users with similar access-control requirements, so managing users and groups is a key responsibility for sysadmins.

Groups in Linux

While administering a Linux machine that houses multiple users, there might be times when he needs to take more control over those users than the basic user tools offer. This idea comes to the fore especially when the administrator needs to manage permissions for certain users. For example, we have a directory that needs to be accessed with read/write permissions by one group of users and only read permissions for another group. So, users can be listed in different groups. Group allows us to set permission on the group level instead of setting the permission on individual level. Every Linux distribution has a graphical tool to manage groups. Groups can be managed by graphical tools, command line tools and by vi editor depending upon the user's experience. Here we will discuss basic group commands

The three commands used to manage group accounts on most of the Linux distribution are:

- Add a group: groupadd
- Modify a group: groupmod
- Delete a group: groupdel

Creating Group

Sysadmins are likely to add groups that represent specific and varied needs to Linux servers. Most of these need focus on controlling access to files, directories, or other resources. To create a group, we can use following command:

$ sudo groupadd [option] group_name

This command without option adds group_name group to the /etc/ group file. The entry of this file is already discussed. The group membership is empty by default. There are a few options for groupadd, but a common one is --gid (-g for short), to specify a group ID (GID) number.

Modification of Group

Creating a group is only the first step. We might also need to modify the group by using the groupmod command. The most common group modifications consists of name changes.

Group modification options include:

--gid (-g): Change the GID

--new-name (-n): Change the group name

Therefore, to update a group name from group_name to gropu_name1, use the following command:

$ sudo groupmod --new-name group_name1group_name

Adding User to Group

By default, groups are empty. The group members can be edited with usermod or useradd command. If a group is not listed then by default, usermod command will remove the user from every group of which he is a member. Here, -a (append) option is used to prevent this from happening Suppose we want to add a user named test to group group_name1, then the command will be as follows.

$sudo usermod -a -G group_name1 test

We can remove users from group using gpasswd command itself. If we remove test from group_name1 , then command will looks like

$sudo gpasswd -d test group_name1

We can also remove a user from a group by manually editing the file '/etc/group'. The effects of this method will apply to the user upon reboot.

Deleting Group

We may need to delete a group. Deleting a group does not delete the user accounts that are members of the group. To delete the group "group_name1", command is :

$ sudo groupdel group_name1

Run the command groups to see what user-groups the user belongs to.

Permission and Ownership in Linux

Linux is a multi-user operating system. There may be multiple users

accessing the same system. But if any user could access and modify all files belonging to other users or system files, this would certainly be a security risk. This is why UNIX and thus Linux (Linux is a Unix-like system) have built-in security measures in place. This ensures that a file or directory can be accessed, modified, or executed by only desired users. Which file would be accessed by which user is decided by two factors in Linux:

- File ownership
- File permission

File ownership in Linux

Every file and directory in Linux have three kinds of owners:

User

User is the owner of the file. When we create a file, we become the owner of the file. The ownership can be changed as well.

Group

Every user is part of a certain group(s). A group consists of several users, and this is one way to manage users in a multi-user environment.

Other

'Other' can be considered as a super group with all the users on the system. Basically, anyone with access to the system belongs to this group.

File permissions in Linux

Every file and directory in Linux have the following three permissions for all the three kinds of owners:

- Permissions for files

  Read - Can view or copy file contents

  Write - Can modify file content

  Execute - Can run the file (if its executable)

- Permissions for directories

  Read - Can list all files and copy the files from directory

  Write - Can add or delete files into directory (needs execute permission as well)

Execute - Can enter the directory

We have already discussed the output of ls-l command in Unit-4, for example we have taken this output as reference

-rw-rw-r--1 ubuntu ubuntu 19 May 20 06:09 test0.txt

Here we have found that the out has following fields

- File type: Denotes the type of file. d means directory, - means regular file, l means a symbolic link.
- Permissions: This field shows the permission set on a file.
- Hard link count: Shows if the file has hard links. Default count is one.
- User: The user who owns the files.
- Group: The group that has access to this file. Only one group can be the owner of a file at a time.
- File size: Size of the file in bytes.
- Modification time: The date and time the file was last modified.
- Filename: Obviously, the name of the file or directory.

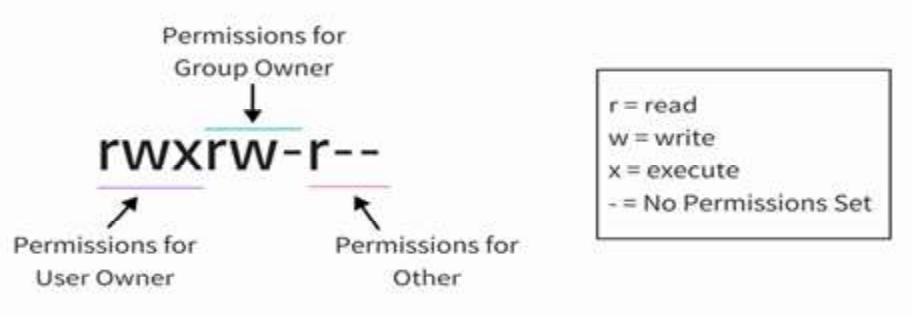
In the above line, we can see that Permission field has 9 characters:

-rw-rw-r-

Each letter denotes a particular permission:

- r: Read permission
- w: Write permission
- x: Execute permission
- -: No permission set

Permissions are always in the order of read, write and execute, i.e., rwx. And then these permissions are set for all three kinds of owners in the order of User, Group and Other.



Change file permissions in Linux

We can use chmod command for changing the permissions on a

file in Linux. There are two ways to use the chmod command: Absolute mode and Symbolic mode. In the absolute mode, permissions are represented in numeric form usually in octal form. In this system, each file permission is represented by a number.

- r (read) = 4
- w (write) = 2
- x (execute) = 1
- - (no permission) = 0

So, rw- permission is represented as 6 i.e. 4 for read permission and 2 for write permission. Similarly, rwx permission will be 7 (4+2+1) in absolute mode. So, if we want to assign read-write-execute (rwx) permission to user, read-write (rw-) permission to group owner and only read(r--) permission to other owner for test0.txt file then command will be like:

$sudo chmod 761 test0.txt

Symbolic mode

The problem with the absolute mode is that we should always provide three numbers for all the three owners even if we want to change the permission set for just one owner. This is where we can use the symbolic mode with chmod command. In symbolic mode, owners are denoted with the following symbols:

- u = user owner
- g = group owner
- o = other
- a = all (user + group + other)

The symbolic mode uses mathematical operators to perform the permission changes:

- + for adding permissions
- - for removing permissions
- = for overriding existing permissions with new value

In our previous example, if we want to perform it with symbolic mode, command will be look like:

$chmod u+rwx, g+rw, o+r test0.txt

To only add execute permission for other owner, we can use chmod command like this: $chmod o+x test0.txt

Change file ownership in Linux

To change the ownership of a file, we can use the command chown i.e. change owner. We can change the user owner of a file using chown command like:

$chown <new_user_name> <filename>

If we want to change the user as well as group, we can use chown command like this:

$chown <new_user_name>:<new_user_group> <filename>

If we just want to change the group, we can either use chown command in this manner:

$chown :<new_user_group> <filename>

or we can use chgrp (change group) command specifically used for changing group owner of a file or directory.

chgrp <new_user_group> <filename>

In our example so far, if we want to change the user owner and group to root, the chown command will be like this:

$sudo chown root:root test0.txt

The passwd Command

The passwd command in Linux is used to change the user account passwords. The root user reserves the privilege to change the password for any user on the system, while a normal user can only change the account password for his or her own account.

Syntax:

passwd [options] [username]

```
ubuntu@ubuntu:~$ passwd ubuntu
New password:
Retype new password:
passwd: password updated successfully
```

Processing in passwd command:

- **Verify current user password:** Once the user enters passwd command, it prompts for current user password, which is verified against

the password stored in /etc/shadow file user. The root user can bypass this step and can directly change the password.

- **Verify password aging information:** In Linux, a user password can be set to expire after a given period of time. Also, a user can be prohibited to change his/her password for a period. This password aging information (and the password itself) is stored in a file /etc/shadow.

Change the password: After authentication, the user is prompted to enter the new password and verify it by retyping the password.

passwd options:

- **-d, -delete:** This option deletes the user password and makes the account password-less.
- **-e, -expire:** This option immediately expires the account password and forces the user to change password on their next login.
- **-h, -help:** Display help related information to the passwd command.
- **-i, -inactive INACTIVE_DAYS:** This option is followed by an integer, INACTIVE_DAYS, which is the number of days after the password expires that the account will be deactivated.
- **-k, -keep-tokens:** This option is used when we want to change the password if it is expired. It keeps the authentication tokens for the authentication if the password is not yet expired, even if it is requested to change it. Note that if the expiry period for a user is set to 99999, then this option will not keep tokens and the password will be changed.
- **-l, -lock:** Lock the password of user. This appends the encrypted password of the user with a character '!', and thus making it unable to match with any of input password combinations. This does not disable the account but prevents the user from logging in using a password like:

$sudo passwd -l ubuntu Locking password for user ubuntu

- **-u , -unlock :** Unlock the password of the user, e.g.$sudo passwd -u ubuntu
- **-n, -mindays MIN_DAYS:** Change the minimum number of days between password changes to MIN_DAYS so that the user can't change the password for MIN_DAYS.

- **-q, -quiet:** This option is used for quiet mode. While using this option to change a password, the message "Changing password for $user.", usually gets printed before changing a password, does not get echoed.
- **-x, -maxdays MAX_DAYS:** Set the maximum number of days for which the password remains valid. After MAX_DAYS, the password will expire, and the user will be forced to change password.

  Example: Use the following flags to set minimum password lifetime (-n), maximum password lifetime (-x), warning before expiration (-w), and inactive to disabled (-i) in days for each. The order of the options doesn't matter.

  $ sudo passwd -n 1 -x 90 -w 3 -i 10 ubuntu

  The user ubuntu can not change the password for 1 day (-n 1), after 90 days (-x 90) password will expire and if the user remains inactive for 10 days (-i 10) then also password will expire.

  Root vs Sudo

  In a Linux system a root user can do anything, but normal user has no permissions. To run any command, they need to ask for permissions from the superuser. The easy and common way to grant administrative privileges to non-root users is that a user can use su command and temporarily become the root, but users must know the root's password. Sharing root password in multiuser system is dangerous. To overcome this risk, sudo command comes in trend. It allows a user to run a command as a root or as any other user after providing the user's own password for authentication. This information are defined in the /etc/sudoers file.

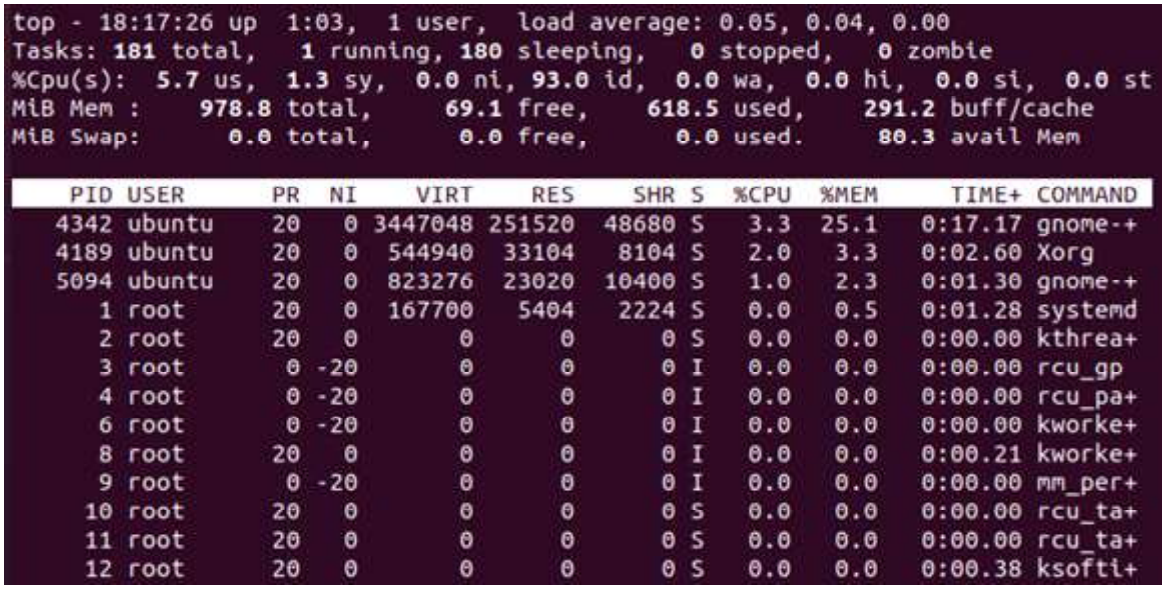
12.5 SYSTEM MONITORING

Some system monitoring tools available with Linux distribution which are used to check memory usage, disk space usage and I/O activity are described in the following.

TOP

TOP is part of every Linux distribution. Following screenshot shows

the result of top command. As we can see in the following screenshot running tasks, their CPU, memory and swap usage. We can do different specific task along with this output according to our need. It has a large number of options/switches which system administration can use easily for different purposes.



Hot keys /Switches associated with top command

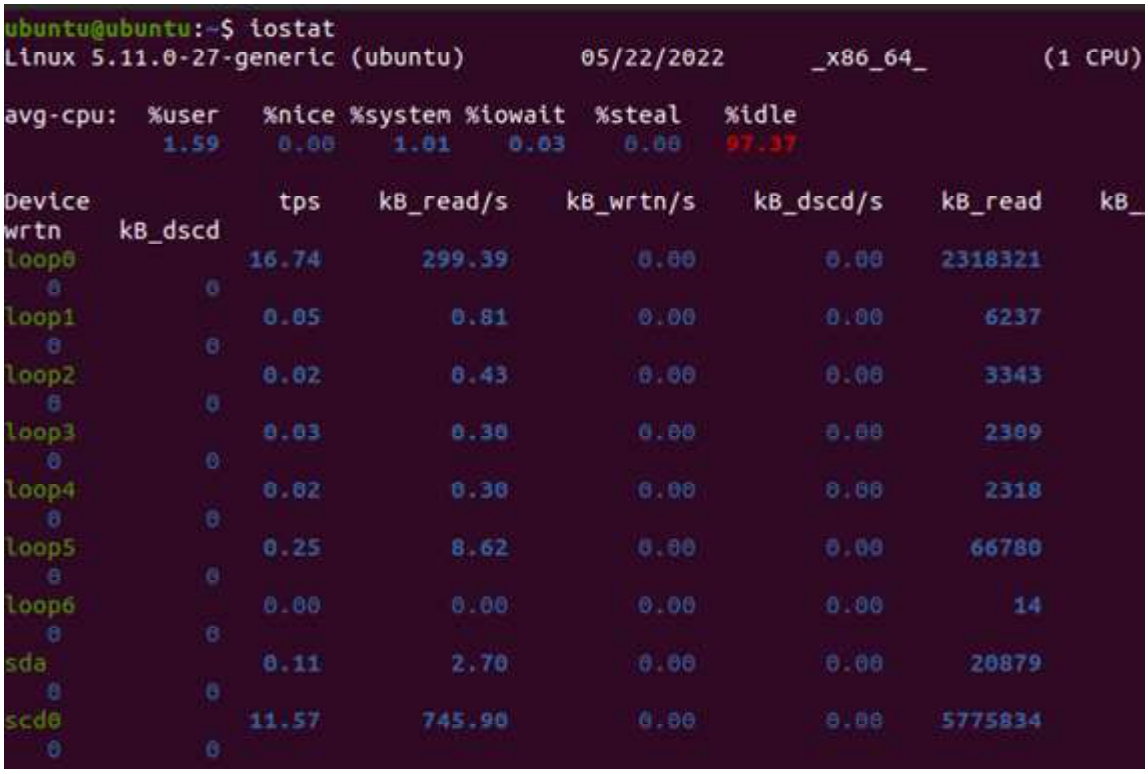
Hot Key	Usage
t	Displays summary information off and on.
m	Displays memory information off and on.
A	Sorts the display by top consumers of various system resources. Useful for quick identification of performance-hungry tasks on a system.
f	Enters an interactive configuration screen for top. Helpful for setting up top for a specific task.
o	Enables you to interactively select the ordering within top.
r	Issues renice command.
k	Issues kill command.
z	Turn on or off color/mono

Free

Free also comes pre-installed with Linux distributions to check the memory usage. It shows us the buffers and cached memory as well. There are several formats of memory size like KB, MB and GB. We need to use 'm' and 'g' parameters with the free command.

Iostat

Iostat command reports us the CPU and disk I/O stats. Reads and writes are shown as block read and block writes. We can get the idle percentage of the CPU to check how much time it has not done any heavy task. Following is the output of iostat command. As we can see in the following screenshot that system is 97.7 percent idle while sda represents the hard disk.



lsof: The job of lsof command is to "list open files" in the system. An open file does not mean pdf or a text file, it includes disk files or pipes used by processes in the background.

ps - Displays the Linux processes, ps command will report a snapshot of the current processes. To select all processes, use the -A or -e option.

12.6 GETTING SYSTEM INFORMATION WITH UNAME

The uname is a command line tool used to determine the processor architecture, the system hostname and the version of the kernel running

on the system. Syntax of this command is uname -options

Different options available with uname include:

-a, --all: print all information, in the following order, except omit -p and -i if 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

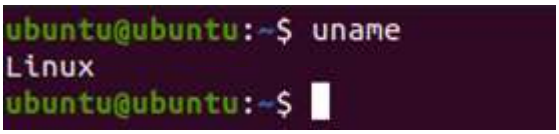
-p, --processor: print the processor type (non-portable)

-i, --hardware-platform: print the hardware platform (non-portable)

-o, --operating-system: print the operating system

--version: output version information and exit

When we run uname command without any option, it just prints the core operating system name. For example, here the core operating system name is Linux



If we use -a option to display all the information in this order.

- Kernel name
- Nodename/hostname
- Kernel release
- Kernel version
- Machine
- Processor
- Hardware platform
- Operating System

For example, after using uname -a command the following output has been shown:



Here

- Kernel name: Linux
- Nodename/hostname: ubuntu
- Kernel release: 5.11.0-27-generic
- Kernel version: #29~20.04.1-Ubuntu SMP Wed Aug 11 15:58:17 UTC 2021
- Machine: x86_64
- Processor: x86_64
- Hardware platform: x86_64
- Operating System: GNU/Linux

12.7 BACKUP AND RESTORE FILES

Unix and Linux backup and restore can be done using backup commands tar, cpio ufsdump, dump and restore. Though these commands may be sufficient for small setups in order to take an enterprise backup, we need to go for custom backup solutions provided by different software solutions.

Backup Restore using tar command

The tar (tape archive) command is used for single or multiple files backup and restore on/from a tape or file. The tar cannot backup special character & block device files, showing as 0 byte files with first letter of permissions as b or c for block or character and it works only on mounted file system. It cannot access the files on unmounted file system.

Backing up all files in a directory including subdirectories to a tape device (/dev/rmt/0) or a file.

$tar cvf /dev/rmt/0 *

Where Options cvf indicates create, verbose and file or archive device respectively; * refers to all files and directories. Together the command means creating a tar file on /dev/rmt/0 from all file and directories in the current directory.

$tar cvf /home/backup *

Create a tar called backup in home directory, from all file and

directories in the current directory.

Viewing a tar backup on a tape or file:

t option is used to see the table of content in a tar file. So

$tar tvf /dev/rmt/0 ## view files backed up on a tape device.

$tar tvf /home/backup ## view files backed up inside the backup

Extracting tar backup from the tape:

x option is used to extract the files from tar file.

$tar xvf /dev/rmt/0 ##extract/restore files into current directory.

$tar xvf /home/backup ## extract/restore files into current directory

Some useful tar options are-

-p, -preserve-permissions : extract & keep the file permissions (default for superuser)

-z, -gzip : Filter the archive through gzip

-j, -bzip2 : Filter the archive through bzip2

-u, -update : Only append files newer than copy in archive

-r, -append : Append files to the end of an archive

-delete : Delete from the archive (not on mag tapes!)

-d, -diff, -compare : Find differences between archive and file system

-test-label : Test the archive volume label and exit

-C, -directory=DIR Change to directory DIR

Backup/restore using cpio command

The cpio command is used for single or multiple files backup. It can backup special character & block device files which tar cannot. Like tar, it also works only on mounted file system. The cpio command needs a list of files to be backed up. It preserves hard links and time stamps of the files.

Using cpio command to backup all the files in current directory to tape, we can use:

#find . -depth -print | cpio -ovcB > /dev/rmt/0

Where cpio expects a list of files and find command provides the list, cpio has to put these files on some destination and '>' sign redirects these files to tape. This can be a file as well.

Viewing cpio files on a tape:

cpio -ivtB < /dev/rmt/0 ## Options i -> input ; v->verbose; t-table of content; B-> set I/O block size to 5120 bytes

Restoring a cpio backup:

cpio -ivcB < /dev/rmt/0 ## Options i -> input ; v->verbose; t-table of content; B-> set I/O block size to 5120 bytes

Dump and Restore:

The dump command in Linux is used for backup the filesystem to some storage device. It backs up the complete file system and not the individual files. But it is not come with a preinstall. We have to install it first to use its utilities. It works only with ext2/ext3 filesystem and not with others like FAT and ReiserFS. It allows user to take incremental backup like weekly basis.

Syntax for dump command is: **dump -[options]**

-level # : The dump level which is an integer ranging from 0-9. If the user has asked to perform a full backup or a backup of only those new files added after the last dump of a lower level.

- f file : This specifies the file where the backup will be written to. The file can be a tape drive, a floppy disk, ordinary file or standard output.

-u : This records and updates the backup in /etc/dumpdates file.

sudo dump 0uf /dev/qft0 /dev/sda6

-B records : It displays the number of dump records per volumes. In other words, it shows the amount of data that can fit in the tape. This is not always required as DUMP can detect the end of media or storage. This takes a numerical value and is used in combination with -b as mentioned below.

-b blocksize : The block size specifies the number of kilobytes per dump record. The default value of block size is 10. Notice here how in the 4th line of DUMP the block size is 20 instead of 10.

-W : Lists the file systems that need to be backed up as given below:

dump -W

Restore Command

restore command in Linux system is used for restoring files from a backup created using dump. The restore command performs the exact inverse function of dump. A full backup of a file system is being restored and subsequent incremental backups layered is being kept on top of it. Single files and directory subtrees can easily be restored from full or partial backups. Restore simply works across a network. Other arguments which need to be passed to the command are file or directory names specifying the files that need to be restored.

Options:

-a: In -i o -x mode, restore will ask the user for the volume number on which the files need to be extracted. The -a option simply disables this behaviour and reads all the volumes starting with 1.

-c: Normally, restore will try to determine dynamically whether the dump was made out of an old or from a new format file system. The -c flag simply disables this check, and it only allows reading a dump with the method of the old format.

-d: The -d (debug) flag causes the restore to print all the debug information.

-f: This option reads the backup from the file that can be a special device file like /devfffrrr5/st0, /dev/sda1 (a disk drive) etc.

-h: Extract the actual directory, rather than the files which is being referencing. This prevents the hierarchical restoration of complete subtrees from the dump.

-C: This flag allows comparison of files from a dump. Restore command reads the backup and compares it with the contents of files present on the disk.

-i: This flag allows the interactive restoration of files from a dump. After reading the directory information from the dump, restore command provides a shell-like interface which simply allows the user to move around the directory tree selecting files to be extracted.

-P file: Restore creates a new Quickly File Access file from an existing dump file without any kind of restoring its contents.

-R: Restore requests a particular tape of the multi-volume set to restart a full restore This is proven useful if the restore has been interrupted.

-r: Restore a file system. The target file system needs to be made pristine with mke2fs(8), mounted, and the user cd'd into a pristine file system before starting the restoration of the initial level 0 backup.

-t: The names of the specified files are being listed if they occur on the backup. If no file argument is already given, the root directory is listed by default.

-x: The names of the files are read from the given media. If a named file matches a particular directory whose contents are on the backup and the -h flag is not specified, the directory is then recursively extracted.

-k: Uses the Kerberos authentication when contacting the remote tape server (Only available if this option was already enabled when restore was compiled).

-l: When we are doing remote restores, then assume the remote file is a regular file (instead of a tape device). If we are supposed to be restoring a remote compressed file, then we should need to specify this option or restore will fail to access it correctly.

-m: Extracted by the inode numbers rather than by file name. This is proven useful if only a few files are being extracted, and one wants to avoid the regenerating to be completed pathname to the file.

-N: The -N flag causes the restore to perform a full execution as requested by one of the -i, -R, -r, t or x command without actually writing any file on the disk.

-o: The -o flag causes the restore to automatically restore the current directory permissions without asking for the operator whether he needs to do so in one of -i or -x modes.

-u: Whenever certain types of files are created, restore may generate a warning diagnostic if they already exist in the target directory. To prevent this, the -u (unlink) flag can cause restore to remove the old entries before attempting to create new ones.

-v: Normally restore does this work silently under the hood. The -v (verbose) flag causes it to type the name each of file it simply treats preceded

by its file type.

-V: Enables reading a multi-volume non-tape medium like CDROMs.

-y: Do not ask for the user whether to abort the restore in the event of an error. It always tries to skip over the bad block and then continues.

Dumping and Restoring

Dump can do full back up or incremental backup. Numerical argument as 0 means full backup and subsequent number represents the incremental backup corresponding to full back.

Syntax : dump - < level number > -f < source filesystem device >

sudo dump -0u -f /dev/st0 /dev/sda9 //this command will take full backup(as shown by -0)

of /dev/sda9

to restore the same file, command will be:

sudo restore -f /dev/st0

Important Files related to backup

/dev/st0: default tape unit to dump to

/etc/dumpdates: dump date records

/etc/fstab: dump table: file systems

/etc/mtab: dump table: mounted file systems

/etc/group: to find group operator

12.8 KUDZU

When we add or remove hardware from our Red Hat linux systems and reboot it, a window appears (New system may not show it) during the reboot process suggesting that hardware has either been added or removed and asking whether we want to reconfigure it. The code that detects and reconfigures hardware is called kudzu.

The kudzu script is a hardware autodetection and cut tool that runs automatically at boot time. We can start kudzu while Linux is running without rebooting also. In either case, here is what kudzu does:

- It checks the hardware connected to the computer.
- It compares the hardware it finds to the database of hardware

information stored in the /etc/sysconfig/hwconf file.

- It prompts to modify the system configuration, based on new or removed hardware that has been detected.
- If no previous database exists, kudzu attempts to determine what devices have already been configured, by looking at /etc/modprobe.conf, /etc/sysconfig/network-scripts/, and /etc/X11/xorg.conf.

Files Related to kudzu

/etc/sysconfig/hwconf : Lists the current installed hardware.

/etc/sysconfig/kudzu: Configuration for the boot-time hardware probe. Set 'SAFE' to something other than 'no' to force only safe probes.

/etc/modprobe.conf: Module configuration file.

/etc/sysconfig/network-scripts/ifcfg-*: Network interface configuration files.

The following is a list of hardware that kudzu can detect (according to the kudzu README file), followed by a version of what kudzu does to configure the device. Other devices may be detected as well (such as USB devices).

Network devices - Adds an Ethernet interface alias (eth0, eth1, etc.) if necessary and either migrates the old device configuration or creates a new one.

SCSI - Adds an alias for scsi_hostadapter.

Video card - Runs the Xconfigurator command to configure the video card.

Sound card - Runs the sndconfig command to configure and test the sound card.

Mouse - Links the new mouse device to /dev/mouse and runs the mouseconfig command to

configure and test the mouse.

Modem - Links the new modem device to /dev/modem.

Scanner - Links the new scanner device to /dev/scanner.

Keyboard - Runs the kbdconfig command to reconfigure the keyboard. Also, if you are using a serial console, it makes sure /etc/inittab and /etc/securetty are configured to be used by a serial console.

The following is a list of actions kudzu takes when a device is removed:

Network - Removes the alias for the Ethernet interface (eth0, ethl, etc.).

SCSI - Removes the alias for the SCSI host adapter (scsi_hostadapter).

Mouse - Removes the link to /dev/mouse.

Modem - Removes the link to /dev/modem.

CD-ROM - Removes the link to /dev/cdrom.

Scanner - Removes the link to /dev/scanner.

The only required problems with kudzu is its serial probe which disturbs any currently in-use devices and returns odd results if used on machines acting as serial consoles. On some older graphics cards, the DDC probe can do strange things.

In many cases, kudzu will find the right modules needed by the hardware and configure the system to load them automatically observed and identified separately. If we want to ownership kudzu without rebooting, we should first shut down the graphical desktop interface. We can do that by changing to run level 3 (init 3). Then, as root user, simply type kudzu.

12.9 X-WINDOWS

The X Window System, aka "X" is a GUI which sits on top of Linux. Unlike Microsoft Windows, the X Window System can look and operate in a large variety of different ways. It can operate very primitively or very advanced. Getting X working properly can range from simple to very complicated. Fortunately, such configuration is becoming easier and more automated in the newer distributions of Linux. Although in a majority of cases, X can be configured automatically.

There are two main methods of getting X for Linux. The first and easiest method, is to use ``Xconfigurator'' utility. The utility tries to detect the system's hardware and installs the applicable X software with the appropriate configuration settings.

If we are still unsuccessful after trying out various settings with Xconfigurator, we may use the "xf86config" utility. Although certainly not as user-friendly or attractive as Xconfigurator is, but it gives us finer control over the configuration process.

Finally, if it is still not working, we may go for editing the "/etc/X11/XF86Config" file by hand and tweaking various settings. If this is the case, we need help from experts which may be found in Linux communities.

After getting X working properly, it may be the lack of rich colours. This is because X uses a default 8-bit per pixel (``bpp'') colour depth. We can use higher colour depths, if video hardware will support them.

The various colour depths are listed in "/etc/X11/XF86Config" file, and look like this:

```
Subsection "Display"
    Depth       24
    Modes       "800x600" "1024x768"
    ViewPort    0 0
    Virtual     1024 768
EndSubsection
```

The above section shows the possible resolutions which are available when using the 24-bit colour depth (800x600 and 1024x768, as listed in the "Modes" line); these resolutions can be switched between "on-the-fly" using the <Alt><+> and <Alt><-> keys.

We can test each colour depth manually by typing, "startx -- -bpp 24" (for the 24-bit depth) and make sure X is working properly for the colour depth we wish to use.

If we are able to successfully use a higher colour depth and wish to use it as the default, we will need to create a "/etc/X11/xinit/xserverrc" file as follows:

```
exec X :0 -bpp 24
```

The above change will allow X to use 24 bits per pixel (if we have problems with this, we can try 16 or 32 instead of 24).

Assuming we have configured X properly, starting it is a simple matter of typing "startx" as any user. The X GUI will start, and after we

have finished our session and quit X, we will be returned to the regular Linux console.

We may also use the X Desktop Manager ("xdm") to start up the X Window System automatically at system boot time. This allows Linux system to always run under X (although we can switch from the GUI to the regular consoles with <Ctrl>-<Alt>-<F1>, and then back again to the GUI with <Alt>-<F7> as needed). This is a nice way of providing an attractive and friendly environment for users, and avoid having to type "startx" all the time.

To enable xdm, simply edit the "/etc/inittab" file and change the line that reads "id:3:initdefault:" to the following:

id:5:initdefault:

The above change will switch Linux to run level 5 upon system boot up; this run level, by default, will start xdm. We may also wish to check our "/etc/inittab" file, probably near the bottom, to ensure whether the following line is present:

x:5:respawn:/usr/bin/X11/xdm -nodaemon

If we have enabled xdm and wish to use a higher "bpp" value, then the default of 8 (and video card and monitor will support it), we will need to modify the "/etc/X11/xdm/Xservers" file as follows:

:0 local /usr/X11R6/bin/X -bpp 24

The above change will start the xdm at 24 bits per pixel.

We may also wish to edit the "/etc/X11/xdm/Xsetup_0" file and with a "#" character, comment out the line that starts "xbanner" as shown:

#/usr/X11R6/bin/xbanner

This will prevent the default xdm banner screen from displaying for a fragmented second between KDE sessions.

Choosing a Window Manager for X:

Now, we should decide on a window manager. The X Window System is simply the environment which allows graphics to be displayed on our system's hardware; the window manager is responsible for how X looks and how it interacts with us and our applications. The Red Hat distribution of Linux contains several window managers, including fvwm,

olvm, twm, AfterStep, and others. The default one we see when starting up X for the first time is fvwm95, a Win95-like environment.

12.10 GNOME GRAPHICAL INTERFACES

The GNU Network Object Model Environment (GNOME) is a windowing environment that enhances X window environment. It is fully featured containing a large selection of applications.

Generally, GNOME is included with the distribution. Otherwise, we will need to download the latest RPM distribution of the package. After we have all the necessary files, the GNOME package can be installed with a simple command, typed as "root":

#rpm -Uvh gtk*.rpm *.rpm

(The above command ensures the GTK libraries are installed first, to avoid dependency errors from occurring).

GNOME is actually not a Window manager, but provides added functionality to it. Therefore, once we have installed GNOME, we should decide which window manager to use, and create a ".xinitrc'' file in our directory which loads the appropriate window manager and starts GNOME. The file should look something like this:

afterstep &

exec gnome-session

The above file will load AfterStep for the window manager, and then run GNOME on top of it.

More information on the GNU Network Object Model Environment can be found on the GNOME web page at http://www.gnome.org/. Don't forget to check out the screen shots, located at http://www.gnome.org/screenshots/.

12.11 KDE INSTALLATION AND CONFIGURATION

The K Desktop Package (KDE) is another popular window manager that is somewhat more mature. However, it seems to require a bit more memory resources than GNOME.

The first step for installing KDE is to download the latest RPM distribution of the package. To do so, locate an FTP mirror at http:// www.kde.org/mirrors.html. Try to choose a mirror that is close to our geographic location, but make sure whichever one we choose is updated often.

When we have found a suitable mirror, download all the RPM files which are applicable to Linux version.

Once we have all the necessary files, the KDE package can be installed with the following simple commands, typed as "root" (make sure you are in the directory where all your KDE rpm files are):

#rpm -Uvh qt*.rpm

#install-kde-1.1-base

The above commands will install the Qt libraries first, and then install the KDE base package. Once this is done, we should log off and log back in so that our path environment is set appropriately. Then type:

#install-kde-1.1-apps

The above command will install the applications programs.

This installation procedure is discussed in more detail in the file "readme-redhat-rpms.txt" that should have been included with the KDE files which we downloaded.

If all goes well, and KDE has been installed without any error messages, we may have to configure KDE to be the default window manager for any of our users (the one they will see immediately after typing "startx"), by typing the following, again as "root":

/opt/kde/bin/usekde userid

More information on the K Desktop Environment can be found on the KDE web page at http://www.kde.org/. Don't forget to check out the screen shots, located at http://www.kde.org/kde2shots.html.



CHECK YOUR PROGRESS

Q 1: Which command is used to add a new user?

Q 2: Which command is used to delete an existing user account?

Q 3: Which command is used to create a group?

Q 4: Which command is used to modify a group?

Q 5: Which command is used to delete an existing group?

Q 6: What is the use of top command?

Q 7: What is the use of free command?

Q8: What is the use of tar command?



12.12 LET US SUM UP

- In system administration, we have to perform tasks like managing user accounts, granting access, checking system performance, and taking corrective measures. Absolute path names are full path name to the file whereas relative pathnames are relative to the working directory.
- Linux login process is not a single process, it has different sub processes along with parent process together.
- Linux is a multiuser system, so in same system we are able generate different users and different groups and accordingly we can give authorization for file access.
- We have lots of system monitoring tools to check system health in Linux.
- There is no inbuilt backup system for Linux, but we can use dump and restore package for it.
- The X Window System, aka "X" (commonly and incorrectly known by many as "X-Windows") is a GUI which sits on top of Linux. GNOME and KDE sits top of X windows and make it more easier to use.



12.13 FURTHER READINGS

1) Linux Administration Cookbook By Adam K. Dean
2) Linux Administration By Jason Cannon



12.14 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: adduser

Ans to Q No 2: userdel

Ans to Q No 3: groupadd

Ans to Q No 4: groupmod

Ans to Q No 5: groupdel

Ans to Q No 6: Top command is used to view the running tasks, their CPU, memory and swap usage.

Ans to Q No 7: Free command is used to check the memory usage along with buffers and cached memory.

Ans to Q No 8: tar command is used for single or multiple files backup and restore on/from a tape or file.



12.15 MODEL QUESTIONS

Q 1: Explain login process in Linux in details.

Q 2: Create a user account with password in your system add it to a specific group and modify it to only active for 3days from creation.

Q 3: Explain different system monitoring tools and their uses.

Q 4: Explain backup and restore process of Linux system

Q 5: What is kudzu? Why it is useful?

Q 6: Write short note on X-Windows, GNOME, KDE.

*** ***** ***