Q. Explain “useradd” command.

Ans :-

Adding a New User

To add a new user to the system, typing the following at a shell prompt as root:

#useradd [*options*] *username*

…where *options* are command line options as described in . “useradd command line options”
By default, the useradd command creates a locked user account. To unlock the account, run the following command as root to assign a password:

passwd *username*

- **useradd command line options**

Option	Description
-c <'comment'>	comment can be replaced with any string. This option is generally used to specify the full name of a user.
-d <home_directory path>	Home directory to be used instead of default /home/username/.
-e <date>	Date for the account to be disabled in the format YYYY-MM-DD.
-f <days >	Number of days after the password expires until the account is disabled. If 0 is specified, the account is disabled immediately after the password expires. If -1 is specified, the account is not be disabled after the password expires.
-g <group_name>	Group name or group number for the user's default group. The group must exist prior to being specified here.
-G <group_list >	List of additional (other than default) group names or group numbers, separated

	by commas, of which the user is a member. The groups must exist prior to being specified here.
-m	Create the home directory if it does not exist.
-M	Do not create the home directory.
-N	Do not create a user private group for the user.
-p <password >	The password encrypted with crypt.
-r	Create a system account with a UID less than 500 and without a home directory.
-s	User's login shell, which defaults to /bin/bash.
-u <uid >	User ID for the user, which must be unique and greater than 500.

Explaining the Process

The following steps illustrate what happens if the command useradd prakash is issued on a system that has shadow passwords enabled:

1. A new line for prakash is created in /etc/passwd:

   prakash:x:501:501::/home/prakash:/bin/bash

   The line has the following characteristics:

   - It begins with the username prakash.
   - There is an x for the password field indicating that the system is using shadow passwords.
   - A UID greater than 500 is created. Under Red Hat Enterprise Linux, UIDs below 500 are reserved for system use and should not be assigned to users.
   - A GID greater than 500 is created. Under Red Hat Enterprise Linux, GIDs below 500 are reserved for system use and should not be assigned to users.
   - The optional *GECOS* information is left blank. The GECOS field can be

used to provide additional information about the user, such as their full name or phone number.

- The home directory for prakash is set to /home/prakash/.
- The default shell is set to /bin/bash.

2. A new line for prakash is created in /etc/shadow:

   prakash:!!:14798:0:99999:7:::

   The line has the following characteristics:

   - It begins with the username prakash.
   - Two exclamation marks (!!) appear in the password field of the /etc/shadow file, which locks the account.

     Note

     If an encrypted password is passed using the -p flag, it is placed in the /etc/shadow file on the new line for the user.
   - The password is set to never expire.

3. A new line for a group named prakash is created in /etc/group:

   prakash:x:501:

   A group with the same name as a user is called a *user private group*. .
   The line created in /etc/group has the following characteristics:

   - It begins with the group name prakash.
   - An x appears in the password field indicating that the system is using shadow group passwords.
   - The GID matches the one listed for user prakash in /etc/passwd.

4. A new line for a group named prakash is created in /etc/gshadow:

   prakash:!::

   The line has the following characteristics:

- It begins with the group name prakash.
- An exclamation mark (!) appears in the password field of the /etc/gshadow file, which locks the group.
- All other fields are blank.

5. A directory for user prakash is created in the /home/ directory:

```
# ls -l /home
total 4
drwx------. 4 prakash prakash 4096 Mar  3 18:23 prakash
```

   This directory is owned by user prakash and group prakash. It has *read*, *write*, and *execute* privileges *only* for the user prakash. All other permissions are denied.

6. The files within the /etc/skel/ directory (which contain default user settings) are copied into the new /home/prakash/ directory:

```
# ls -la /home/prakash

total 28
drwx------.  4 prakash prakash 4096  Mar   3 18:23  .
drwxr-xr-x. 5 root      root       4096  Mar   3 18:23  ..
-rw-r--r--.   1 prakash prakash   18    Jun  22  2010  .bash_logout
-rw-r--r--.   1 prakash prakash  176   Jun  22  2010  .bash_profile
-rw-r—r--.  1 prakash prakash  124   Jun  22  2010  .bashrc
drwxr-xr-x. 2 prakash prakash 4096  Jul  14  2010   .gnome2
drwxr-xr-x. 4 prakash prakash 4096  Nov 23 15:09  .mozilla
```

At this point, account called prakash exists on the system. the administrator must next assign a password to the account using the passwd command and, optionally, set password aging guidelines.

Explain The /etc/skel Directory

The */etc/skel* directory contains files and directories that are automatically copied over to a new user's *home directory* when such user is created by the *useradd* program.

A home directory, also called a *login directory*, is the directory on Unix-like operating systems that serves as the repository for a user's personal files, directories and programs, including personal configuration files. It is also the directory that a user is first in after logging into the system. The */etc* directory and its subdirectories contain the many important configuration files for the system.

The *useradd* program is located in the */usr/sbin/* directory, and on most systems it is accessible only by the *root* (i.e., administrative) user.

/etc/skel allows a system administrator to create a default home directory for all new users on a computer or network and thus to make certain that all users begin with the same settings or *environment*.

Several user configuration files are placed in /etc/skel by default when the operating system is installed. Typically they might include

1. .
2. ..
3. .bash_logout
4. .bashrc
5. .gnome2
6. .mkshrc
7. .zshrc
8. .bash_profile

9. .emacs
10. .kshrc
11. .mozilla

The dots preceding the names of these files indicate that they are *hidden files*, i.e., files that are not normally visible in order to avoid visual clutter and help reduce the chances of accidental damage.

The contents of /etc/skel can be viewed by using the *ls* (i.e., *list*) command with its *-a* option (which shows all files and directories, including hidden ones), i.e.,

```
ls -a /etc/skel
```

The location of /etc/skel can be changed by editing the line that begins with SKEL= in the configuration file */etc/default/useradd*. By default this line says SKEL=/etc/skel.

It is usually better to keep /etc/skel as small as possible and put system-wide configuration items into global configuration files such as */etc/profile*. This is because the latter makes it much easier to update existing users' files because its settings take effect as soon as the system is turned on and apply to new users and old uses alike.

When a user is removed from the system by an administrator with the *userdel* command, that user's home directory, including the files and directories that have been copied into it from /etc/skel, remains intact.

The name of the directory *skel* is derived from the word *skeleton,* because the files it contains form the basic structure for users' home directories.

Q. Explain password policies :-

A *password policy* sets certain standards for passwords, such as the password complexity and the rules for changing passwords. A password policy minimizes the inherent risk of using passwords by ensuring that they meet adequate complexity standards to thwart brute force attacks and they are changed frequently enough to mitigate the risk of someone revealing or discovering a password.
There are three main configuration areas that are defined within the password policy:

- Strength or complexity requirements
- History
- Account lockout

- **<u>Explain chage command :-</u>**

Option	Description
-m <days>	Specifies the minimum number of days between which the user must change passwords. If the value is 0, the password does not expire.
-M <days>	Specifies the maximum number of days for which the password isesakal valid. When the number of days specified by this option plus the number of days specified with the -d option is less than the current day, the user must change passwords before using the account.
-d <days>	Specifies the number of days since January 1, 1970 the password was changed
-I <days>	Specifies the number of inactive days after the password expiration before locking the account. If the value is 0, the account is not locked after the password expires.
-E <date>	Specifies the date on which the account is locked, in the format YYYY-MM-DD. Instead of the date, the number of days since January 1, 1970 can also be used.
-W <days>	Specifies the number of days before the password expiration date to warn the user.

1. List the password and its related details for an user

As shown below, any user can execute the chage command for himself to identify when his password is about to expire.

Syntax:	#chage --list username (or) chage -l username

```
$ chage --list prakash

Last password change                                    : Dec 03, 2013
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
```

If user prakash tries to execute the same command for user uday, he'll get the following permission denied message.

```
$ chage --list uday
chage: permission denied
```

Note: However, a root user can execute chage command for any user account.

2. Set Password Expiry Date for an user using chage option -M

Root user (system administrators) can set the password expiry date for any user. In the following example, user prakash password is set to expire 10 days from the last password change.

Please note that option -M will update both "Password expires" and "Maximum number of days between password change" entries as shown below.

Syntax:	# chage -M number-of-days username

```
# chage -M 10 prakash
```

```
# chage --list prakash

Last password change                                    : Dec 03, 2013
Password expires                                        : Dec 13, 2013
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 10
Number of days of warning before password expires       : 7
```

3. Password Expiry Warning message during login

By default the number of days of warning before password expires is set to 7. So, in the above example, when the user prakash tries to login on Apr 30, 2009 — he’ll get the following message.

```
$ ssh prakash@example.com
prakash@example.com password:
Warning: your password will expire in 3 days
```

4. User Forced to Change Password after Expiry Date

If the password expiry date reaches and user doesn’t change their password, the system will force the user to change the password before the login as shown below.

```
$ ssh prakash@example.com
prakash@example.com password:

You are required to change your password immediately (password aged)
WARNING: Your password has expired.
You must change your password now and login again!
Changing password for prakash
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
```

5. Set the Account Expiry Date for an User

You can also use chage command to set the account expiry date as shown below using option -E. The date given below is in “YYYY-MM-DD” format. This will update the “Account expires” value as shown below.

```
# chage -E "2013-01-31" prakash
```

```
# chage -l prakash

Last password change                                    : Dec 03, 2013
Password expires                                        : Jan 03, 2014
Password inactive                                       : never
Account expires                                         : Jan 31, 2014
Minimum number of days between password change          : 0
Maximum number of days between password change          : 10
Number of days of warning before password expires       : 7
```

6. Force the user account to be locked after X number of inactivity days

Typically if the password is expired, users are forced to change it during their next login. You can also set an additional condition, where after the password is expired, if the user never tried to login for 10 days, you can automatically lock their account using option -I as shown below. In this example, the “Password inactive” date is set to 10 days from the “Password expires” value.

Once an account is locked, only system administrators will be able to unlock it.

```
# chage -I 10 prakash
```

```
# chage -I 10 prakash

# chage -l prakash
Last password change                                    : Dec 03, 2013
Password expires                                        : Jan 03, 2014
Password inactive                                       : Jan 13, 2014
Account expires                                         : Jan 31, 2014
Minimum number of days between password change          : 0
Maximum number of days between password change          : 10
Number of days of warning before password expires       : 7
```

7. How to disable password aging for an user account

To turn off the password expiration for an user account, set the following:

-m 0 will set the minimum number of days between password change to 0

-M 99999 will set the maximum number of days between password change to 99999

-I -1 (number minus one) will set the “Password inactive” to never

-E -1 (number minus one) will set “Account expires” to never.

```
# chage -m 0 -M 99999 -I -1 -E -1 prakash
# chage --list dhinesh
Last password change                                    : Dec 03, 2013
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
```

How to remove user :-

A user account may also be deleted from command-line using the *userdel* utility:

#userdel prakash

It is also possible to remove the user's home directory and mail spool as part of the deletion process:

#userdel --remove prakash

or

#userdel -rf prakash

Group

To add a new group to the system,

type the following at a shell prompt as root:

groupadd [*options*] *group_name*

…where *options* are command line options as described in . “groupadd command line options”

groupadd command line options

Option	Description
-f, --force	When used with -g gid and gid already exists, groupadd will choose another unique gid for the group.
-g gid	Group ID for the group, which must be unique and greater than 499.
-o, --non-unique	Allow to create groups with duplicate.
-p, --password password	Use this encrypted password for the new group.
-r	Create a system group with a GID less than 500.

The **groupmod** command modifies the definition of the specified GROUP by modifying the appropriate entry in the group database.

Options

-g, --gid *GID*	The group ID of the given *GROUP* will be changed to GID. The value of GID must be a non-negative decimal integer. This value must be unique, unless the **-o** option is used. Users who use the group as primary group will be updated to keep the group as their primary group. Any files that have the old group ID and must continue to belong to *GROUP*, must have their group ID changed manually.
-h, --help	Display help message and exit.
-n, --new-name *NEW_GROUP*	The name of the group will be changed from *GROUP* to *NEW_GROUP*.
-o, --non-unique	When used with the **-g** option, allow to change the group GID to a non-unique value.
-p, --password *PASSWORD*	The encrypted password, as returned by crypt. **Note:** This option is not recommended because the password (or encrypted password) will be visible by users listing the processes.

	You should make sure the password respects the system's password policy.
-R, --root *CHROOT_DIR*	Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. See also chroot.
-G	new list of secondary group.
-a	append the user to the supplemental GROUPS mentioned by the -G option without removing him/her from other groups.

To add an existing user to an existing group from the command-line:

```
#su -
#usermod -G accounts prakash
```

To add an existing user to a number of existing groups:

```
#su -
#usermod -G accounts,sales,support prakash
```

Note that the above commands remove the user from any supplementary groups which are not listed after the -G, but to which the user is currently a member. To retain any current group memberships, use the -a flag to append the new group memberships:

```
#su -
#usermod -a -G accounts,sales,support prakash
```

to change a primary group of user

```
#useradd -g sales prakash
```

Note- u can't change user's primary group once the user is added to server and u must have group created in your server before assigning it as primary group to user

To delete a group use groupdel command. For example delete a group called ftpusers:
Code:

```
#groupdel ftpusers
```

Q: - Can you explain **/etc/passwd** file format for Linux operating systems

Ans: -

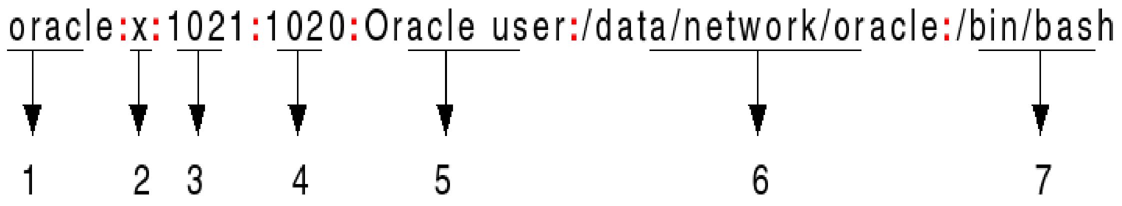
/etc/passwd file stores essential information, which is required during login i.e. user account information. /etc/passwd is a text file, that contains a list of the system's accounts, giving for each account some useful information like user ID, group ID, home directory, shell, etc. It should have general read permission as many utilities, like ls use it to map user IDs to user names, but write access only for the root user.

Notes: -

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 :-



```
oracle:x:1021:1020:Oracle user:/data/network/oracle:/bin/bash
```

1. **Username**: It is used when user logs in. It should be between 1 and 32 characters in length.

2. **Password**: An x character indicates that encrypted password is stored in /etc/shadow file.

3. **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 100-500 are reserved by system for administrative and system accounts/groups.

4. **Group ID (GID)**: The primary group ID (stored in /etc/group file)

5. **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.

6. **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 /

7. **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.

Some FAQ on /etc/passwd :-

Q. Can two users in Linux have same UID?

Ans:- Yes, just edit UID field in /etc/password to the required UID you want.

Q. How to see what are the shells available in a Linux ?

Ans:- #chsh -l will show all the shell available in the machine.

- **/bin/sh**
- **/bin/bash**
- **/sbin/nologin**
- **/bin/tcsh**
- **/bin/csh**
- **/bin/mksh**
- **/bin/ksh**
- **/bin/zsh**
-

Q. What are the major files modified when you create a user?

Ans:- The following files are modified when a user is created

- **/etc/passwd**

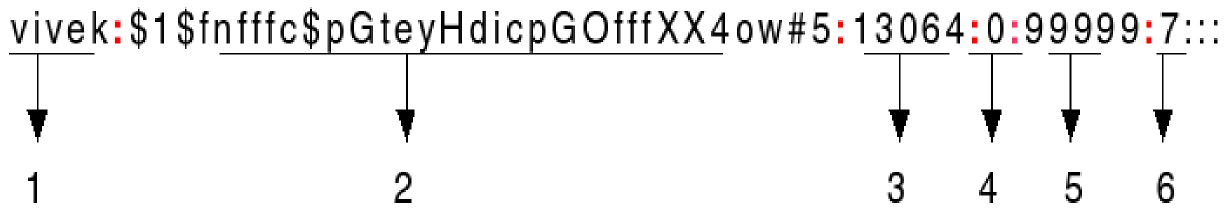
- **/etc/shadow**
- **/etc/group**
- **/etc/gshadow**

Q. Can you explain **/etc/shadow** file used under Linux or UNIX?

<u>Ans: -</u>

/etc/shadow file stores actual password in encrypted format for user's account with additional properties related to user password i.e. 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 :-



<u>Notes: -</u>

<u>Understanding fields in /etc/shadow</u>

1. User name : It is your login name

2. Password: It your encrypted password. The password should be minimum 6-8 characters long including special characters/digits

3. Last password change (lastchanged): Days since Jan 1, 1970 that password was last changed

4. 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

5. Maximum: The maximum number of days the password is valid (after that user is forced to change his/her password

6. Warn : The number of days before password is to expire that user is warned that his/her password must be changed

7. Inactive : The number of days after password expires that account is disabled

8. Expire : days since Jan 1, 1970 that account is disabled i.e. an absolute date specifying when the login may no longer be used

The last 6 fields provides password aging and account lockout features (you need to use #chage command to setup password aging). The password field must be filled. The encrypted password consists of 13 to 24 characters from the 64 character alphabet 'a' through 'z', 'A' through 'Z', '0' through '9', '\.' and '/.' Optionally it can start with a "$" character. This means the encrypted password was generated using another (not DES) algorithm. For example if it starts with "1" it means the MD5-based algorithm was used

Q. Can you explain me the format of **/etc/group** user group file under Linux operating systems?

Ans: -

/etc/group is a text file which defines the groups to which users belong under Linux and UNIX operating system. Under Unix / Linux multiple users can be categorized into groups. Unix file system permissions are organized into three classes, user, group, and others. The use of groups allows additional abilities to be delegated in an organized fashion, such as access to disks, printers, and other peripherals. This method, amongst others, also enables the Superuser to delegate some administrative tasks to normal users.

group file entry looks as follows :-

It stores group information or defines the user groups i.e. it defines the groups to which users belong. There is one entry per line, and each line has the following format (all fields are separated by a colon (:)

```
cdrom:x:24: vivek,student13,raj
_____  _  _     _____
 |     | |      |
 |     | |      |
 1     2 3      4
```

Notes: -

Understanding fields in /etc/group

1. **group_name**: It is the name of group. If you run ls -l command, you will see this name printed in the group field.

2. **Password**: Generally password is not used, hence it is empty/blank. It can store

encrypted password. This is useful to implement privileged groups.

3. **Group ID (GID)**: Each user must be assigned a group ID. You can see this number in your /etc/passwd file.

4. **Group List**: It is a list of user names of users who are members of the group. The user names, must be separated by commas.

<u>More About User Groups :-</u>

Users on Linux and UNIX systems are assigned to one or more groups for the following reasons:

- To share files or other resource with a small number of users
- Ease of user management
- Ease of user monitoring
- Group membership is perfect solution for large Linux (UNIX) installation.
- Group membership gives you or your user special access to files and directories or devices which are permitted to that group

<u>Some FAQ on /etc/group :-</u>

Q. View Current Groups Settings

Type any one of the following command:

```
# less /etc/group
```

OR

```
# more /etc/group
```

Q.Task: Find Out the Groups a User Is In

Type the following command:

```
# groups {username}
# groups
# groups vivek
```

Sample outputs:

```
vivek : vivek adm dialout cdrom plugdev lpadmin netdev admin sambashare libvirtd
```

Q. Print user / group Identity

Use the id command to display information about the given user.

Display only the group ID, enter:

```
$ id -g
$ id -g user
$ id -g vivek
OR
$ id -gn vivek
```

Q. Display only the group ID and the supplementary groups, enter:

```
$ id -G
$ id -G user prakash
$ id -G vivek
```

OR

```
$ id -Gn vivek
```

Q. Can you explain me the format of **/etc/gshadow** user group file under Linux operating systems?

Ans: -

The /etc/gshadow file is readable only by the root user and contains an encrypted password for each group, as well as group membership and administrator information. Just as in the /etc/group file, each group's information is on a separate line. Each of these lines is a colon delimited list including the following information:

```
general:!!:shelley:juan,bob
   |     |     |        |
   |     |     |        |
   1     2     3        4
```

Notes: -

Understanding fields in /etc/gshadow

1. *Group name* — The name of the group. Used by various utility programs as a human-readable identifier for the group.

2. *Encrypted password* — The encrypted password for the group. If set, non-members of the group can join the group by typing the password for that group using the newgrp command. If the value of this field is !, then no user is allowed to access the group using the newgrp command. A value of !! is treated the same as a value of ! — however, it also indicates that a password has never been set before. If the value is null, only group members can log into the group.

3. *Group administrators* — Group members listed here (in a comma delimited list) can add or remove group members using the gpasswd command.

4. *Group members* — Group members listed here (in a comma delimited list) are regular, non-administrative members of the group.