Bachelor of Science in Information Technology
Linux System Administration
CBCS (NOV-2019) Q.P Code: 75433

Q1 a) Explain the duty of the Linux system administrator in configuring a secure system. (5)

- There are major attacks done on machines connected to Internet, the majority of these attacks have not targeted Linux systems, but Linux systems are not entirely immune, either to direct attack or to the effects of attacks on machines running other operating systems.
- For example: Distributed Denial of Service (**DDoS**) attacks, attacks of the summer of 2001, Linux machines themselves were invulnerable, but the huge amount of traffic generated by this "worm" infection prevented many Linux machines from getting much Web-based work done for several weeks.
- The system administrator's task is to make sure that no data on the machine or network are likely to become corrupted, whether by hardware or power failure, by misconfiguration or user error or by malicious or unintentional intrusion from elsewhere.
- Depending on how and to what a Linux machine is connected, the sensitivity of the data it contains and the uses to which it is put, security can be as simple as turning off unneeded services, monitoring the Red Hat Linux security mailing list to make sure that all security advisories are followed, and otherwise engaging in good computing practices to make sure the system runs robustly.
- It is almost full-time job involving levels of security permissions within the system and systems to which it is connected, elaborate firewalling to protect not just Linux machines but machines that, through their use of non-Linux software, are far more vulnerable, and physical security.
- For any machine that is connected to any other machine, security means hardening against attack and making certain that no one is using your machine as a platform for

launching attacks against others. If you are running Web, ftp, or mail servers, it means giving access to those who are entitled to it while locking out everyone else.

- It means making sure that passwords are not easily guessed and not made available to unauthorized persons like former employees should not have access to the system, and no unauthorized person should copy files from your machine or machines.
- As a system administrator keep right balance between maximum utility and maximum safety and knows that confidence in a secure machine today says nothing about the machine's security tomorrow.

Q1 b) Write short note on Piping and Redirection. (5)

- The piping and redirection options are among the most powerful features of the Linux command line
- *Piping* is used to send the result of a command to another command, and *redirection* sends the output of a command to a file.

Piping

- The goal of piping is to execute a command and send the output of that command to the next command so that it can do something with it.
- **Example 1 :**

# ps aux
This command provides a list of all the processes that are currently running on your computer.
a = show processes for all users
u = display the process's user/owner
x = also show processes not attached to a terminal Pipe ps with less command
# ps aux | less
We can see the complete result page by page

- **Example 2:**

If we want to check whether a user with the name neha exists in the user database /etc/passwd then use cat with grep command.

```
# cat /etc/password | grep neha
```

Redirection

- Redirection sends the result of a command to a file. This file can be a text file or a device file.
- There are various types of Redirections.
  - Redirecting Output (STDOUT) to a file.
  - Using Redirect ion of STDIN
  - Separating STDERR from STDOUT
  - Redirecting Output to device files
  - Cloning Devices Using Output Redirection

Q1 c) Explain Bash shell (5)

BASH SHELL:

- Bash stands for Bourne Again Shell and it is the default shell on many Linux distributions.
- The Bash shell is a combination of features from the Bourne Shell and the C Shell.
- It has a command-line editor that allows the use of the cursor keys in a more "user friendly" manner than the Korn shell.
- It also has a useful help facility allowing you to get a list of commands by typing the first few letters followed by the "TAB" key known as **automatic completion**.
- Most used bash feature is **history mechanism**.
- The default prompt is $
- This shell is located under /bin/bash.
- Command to open this shell is $bash

Useful Bash Key Sequences:

- **Ctrl+C** Use this key sequence to quit a command that is not responding or taking too long to complete.
- **Ctrl+D** This key sequence is used to send the end-of-file

(EOF) signal to a command. Use this when the command is waiting for more input. It will indicate this by displaying the secondary prompt >.

- **Ctrl+R** This is the reverse search feature. It opens the reverse-i-search prompt. This feature helps you locate commands you have used previously.
- **Ctrl+Z** Some people use Ctrl+Z to stop a command. It does stop the command, but it does not terminate it. A command that is interrupted with Ctrl+Z is just halted until it is started again with the **fg** command as a foreground job or with the **bg** command as a background job.
- **Ctrl+A** the Ctrl+A keystroke brings the cursor to the beginning of the current command line.
- **Ctrl+B**the Ctrl+B keystroke moves the cursor to the end of the current command line.

Q1 d) Explain cron in detail for job scheduling. (5)

- For some tasks, it makes sense to have them started automatically. Think, for example, of a backup job that you want to execute automatically every night. To start jobs automatically, you can use cron.
- Cron consists of two parts. First there is the *cron daemon*, a process that starts automatically when your server boots. The second part is the cron configuration. This is a set of different configuration fi les that tell cron what to do. The cron daemon checks its configuration every minute to see whether there are any new tasks that should be executed.
- Some cron jobs are started from the directories /etc/cron.hourly, /etc/cron.daily, /etc/cron.weekly, and /etc/cron.monthly. Typically, as an administrator, you're not involved in managing these jobs. Programs and services that need some tasks to be executed on a regular basis just put a script in the directory where they need it, which makes sure that the task is automatically executed.
- There are two ways you can start a cron job as a specific user:

you can log in as that specific user or use su - to start a subshell as that particular user. After doing that, you'll use the command crontab -e, which starts the crontab editor, which by default is a vi interface. That means you work from crontab –e in a similar way that you are used to working in vi. As root, you can also use crontab -u user -e to create a cron job for a specifi c user.

- In a crontab file created with crontab -e, you'll specify which command is to be executed and when on separate lines. Here is an example of a crontab line:

```
0 2 * * * /root/bin/runscript.sh
```

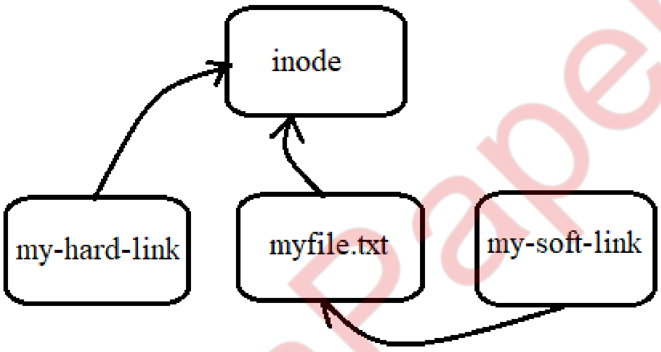
- In the definition of cron jobs, it is very important that you specify to have it start at the right moment. To do that, five different positions are used to specify date and time. You can use the following time and date indicators:

Field	Allowed value
Minute	0–59
Hour	0–23
Day of month	1–31
Month	1–12
Day of week	0–7 (0 and 7 are Sunday)

- This means that, in a crontab specification, the time indicator 0 2 3 4 * indicates that a cron job will start on minute 0 of hour 2 (which is 2 a.m.) on the third day of the fourth month. Day of week in this example is not specified, which means the job would run on any day of the week.

Q1 e) Give differences between symbolic and Hard Links. (5)

- In a Linux file system, we can access a single file from different locations using links.
- Because of this we need not copy files to different locations and make versions.
- A *link* looks like a regular file, but it's more like a pointer that exists in one location to show we how to get to another location.
- In Linux, there are two different types of links. A ***symbolic link*** and ***Hard link.***



Sr.no.	Symbolic Link/ soft link	Hard Link
1.	A file can be accessed using different references pointing is known as a soft link.	A file can be accessed using many different names known as hard links.
2.	Soft Links have different inodes numbers than original files.	Hard Links have same inodes number as of original files.
3.	Soft Link contains the path of original file and not the contents.	These links have actual file contents.
4.	When the original file is removed, the link becomes a 'dangling' link that points to nonexistent file.	Even if the original file is removed, the link will still show you the contents of the file.
5.	You can make soft links to files and directories.	You cannot create a Hard Link for a directory.
6.	Symbolic links can point to any file system as they are simply the name of another file.	Hard links are only valid within the same File System.
7.	The command used for a soft link is "ln –s"	the command used for the creation of hard link is "ln"
8.	Memory consumption is less	Memory consumption is more.
9.	Relative path and absolute path both are allowed in soft links	Relative path is not allowed in a hard link.

Q1 f) Write steps to Create and Manage Repositories. (5)

- If you have a Red Hat server installed that doesn't have access to the official Red Hat Network (RHN) server repositories, you'll need to set up your own repositories.
- This procedure is also useful if you want to copy all of your RPMs to a directory and use that directory as a repository.
  - ➢ Create a directory that you can use as a repository in the root of your server's file system.
    # mkdir /repo
  - ➢ Insert the Red Hat installation DVD in the optical drive of your server. Assuming that you run the server in graphical mode, the DVD will be mounted automatically.
  - ➢ **Go into the mounted dvd**
    #cd/ media/ RHEL[tab]
    Goto the directory where all RPMs are by default. Now use
    # cd Packages
    Copy all packages to the /repo directory you just created.
    # cp * /repo
  - ➢ Go to the /repo directory.
    # cd /repo
    To install **createrepo**, you first need to install the **deltarpm** and **python-deltarpm** packages.
    # rpm -ivh deltarpm[Tab] python-deltarpm[Tab]
    Install the createrepo package.
    # rpm -ivh createrepo[Tab]
  - ➢ Create the metadata that allows you to use the /repo directory as a repository.
    # createrepo /repo
  - ➢ Create a file with the name **/etc/yum.repos.d/myrepo.repo** using vi editor.

```
# vi
/etc/yum.repos.d/myrepo.repo
[myrepo]
name=myrepo
baseurl=file:///repo
```

```
gpgcheck=0
```

- Check outputs.

```
# yum repolist
# yum search nmap
```

- In myrepo.repoconfiguration file, The **file://** part is the URI, which tells yum that it has to look at a file, and after that,you need a complete path to the file or directory i.e**/repo.**
- To check whether packages have been tampered usegpgcheck=1 else gpgcheck=0. If it is set to 1 then we need to configure GPG check.

Q2 a) List and explain different File Systems supported by Linux. (5)

- Several file systems are available on Red Hat Enterprise Linux, currently Ext4 is used as the default file system.

Ext2/3

- The predecessors of the Ext4 file system. Ext2 doesn't use a file system journal, and therefore it is a good choice for very small partitions (less than 100MB).
- ext3 provides all the features of ext2, and also features journaling and backward compatibility with ext2.
- The backward compatibility enables you to still run kernels that are only ext2 aware with ext3 partitions. You can upgrade an ext2 file system to an ext3 file system without losing any of your data.
- ext3's journaling feature speeds up the amount of time it takes to bring the file system back to a normal state if it's not been cleanly unmounted (that is, in the event of a power outage or a system crash).
- Under ext2, when a file system is uncleanly mounted, the whole file system must be checked. This takes a long time on large file systems. ext3 keeps a record of uncommitted file transactions and applies only those transactions when the system is brought back up.
- ext3's journaling feature involves a small performance hit to

maintain the file system transaction journal.

- Therefore, it’s recommended that you use ext3 mostly for your larger file systems, where the ext3 journaling performance hit is made up for in time saved by not having to run fsck on a huge ext2 file system.

Ext4

- The default file system on RHEL. It is a general-purpose file system. Ext4 uses file system journaling feature. The file system journal works as a transaction log in which the file system keeps records of files that are open for modification at any given time.
- The benefit of using a file system journal is that, if the server crashes, it can check to see what files were open at the time of the crash and find the damaged files.
- The drawback of using a journal is that it takes up disk space. For example, an average of 50MB normally on Ext4. That means it’s not a good idea to create a journal on very small file systems because it might leave insufficient space to hold your files.

XFS

- Provides good performance for very large file systems and very large files.

Btrfs

- Btrfs is the next generation of Linux file system. It is based on B-tree database, which makes the file system faster. It also has features like Copy on Write, which makes it very easy to revert to a previous version of a file. This file system is easy to grow and shrink. Btrfs is currently available as a tech preview version only, which means that it is not supported and not yet ready for production.

VFAT and MS-DOS

- Sometimes it’s useful to put files on a USB drive to exchange them among Windows users. This is the purpose of the VFAT and MS-DOS file systems.

GFS

- GFS is Red Hat 's Global File System. It is designed for use in high availability clusters where multiple nodes need to be able to write to the same file system simultaneously.

Q2 b) What are Snapshots? Give steps to Manage Snapshot. (5)

- **LVMsnapshot** allows you to freeze the current state of an LVM volume.
- Snapshot keeps the current state of a volume so that we can revert back to this state if required later.
- They are used to create backups safely. Instead of making a backup of the normal LVM volume where files may be opened, you can create a backup from the snapshot volume, where no file will be open at any time.
- A volume consists of two essential parts:
    - ➢ File system metadata
    - ➢ Actual blocks containing data in a file.
- The file system uses the metadata pointers to find the file's data blocks. When we create snapshot it copies the file system metadata to the newly created snapshot volume, the file blocks stay on the original volume.
- But, when a file changes on the original volume, the original blocks are copied to the snapshot volume before the change is committed to the file system.
- Hence we can say that the size of snapshot increases over the period. This also means that we have to estimate the number of changes that are going to take place on the original volume to create the right size snapshot.
- If only a few changes are expected in original volume then we can create 5 percent backup.
- Every snapshot has a life cycle. That means snapshot does not exist forever. If you no longer need the snapshot, you can delete it using the lvremove command.

1. **Use vgs to get an overview of current available disk space in your volume groups. # vgs**
2. Mount the logical volume on the /mnt directory and Copy some files to the /mnt directory i.e logical volume.

```
# mount
/dev/volg
/lv /mnt #
cp /etc/*
/mnt
```

3. **Create a snapshot with name snap**
   # lvcreate -s -L 50M -n snap /dev/volg/lv
4. Verify the creation of the snapshot volume.
   #lvs
5. create a temporary mounting point for the snapshot and check its contents
   # mkdir /snapmount
   # mount /dev/volg/snap /snapmount
   #cd /snapmount
   # ls
6. remove all files from /mnt directory (i.e logical volume contents are removed)
   # rm -rf /mnt/ *
7. Merge of the snapshot back into the original volume at the next volume activation.
   # lvconvert --merge /dev/volg/snap
8. Unmount the original volume and deactivate then activate the original volume, which is a required step in merging the snapshot back into the original volume. We don't need to remove snapshot. By converting it to original volume, it is automatically removed. Unmount the snapshot using, as it will no longer be available after deactivation and activation of original volume.
   # umount /snapmount
   # umount /mnt.
   # lvchange -a n /dev/volg/lv; lvchange -a y /dev/volg/lv
9. Check /mnt directory to verify contents
   # mount /dev/volg/lv /mnt # ls /mnt

Q2 c) How can rc scripts be managed using chkconfig? (5)

- **chkconfig** manages services that are started at boot time.
- The **chkconfig** utility is a command-line tool that allows you to specify in which runlevel to start a selected service, as well as to list all available services along with their current setting.

Listing the Services

- To display a list of system services either type **chkconfig --list**, or use **chkconfig** with no additional arguments
- Each line consists of the name of the service followed by its status (*on* or *off*) for each of the seven numbered runlevels.
- To display the current settings for a selected service only, use **chkconfig --list** followed by the name of the service:n **chkconfig** --list *service_name.*
- For example, to display the current settings for the **bluetooth** service, type: # **chkconfig --list Bluetooth**
- bluetooth 0:off 1:off 2:on 3:on 4:on 5:on 6:off

Enabling a Service

- To enable a service in runlevels 2, 3, 4, and 5, type the following at a shell prompt as **root user**: **chkconfig** *service_name* on.
- For example, to enable the **bluetooth** service in these four runlevels, type: # chkconfig bluetooth on
- To enable a service in certain runlevels only, add the --level option followed by numbers from 0 to 6 representing each runlevel in which you want the service to run: **chkconfig** *service_name* on --level *runlevels*
- For instance, to enable the **bluetooth** service in runlevels 3 and 5, type: # **chkconfig bluetooth on --level 35**

Disabling a Service

- To disable a service in runlevels 2, 3, 4, and 5, type the following at a shell prompt as root: # **chkconfig** *service_name* off
- For instance, to disable the **bluetooth** service in these four runlevels, type: # chkconfig bluetooth off
- To disable a service in certain runlevels only, add the --level option followed by numbers from 0 to 6 representing each

runlevel in which you do *not* want the service to run:
#chkconfig *service_name* off --level *runlevels*

- For instance, to disable the **bluetooth** in runlevels 2 and 4, type: # **chkconfig bluetooth off --level 24**
- With chkconfig command, the service will be started or stopped the next time you enter one of these runlevels. If you need to start or stop the service immediately, Red Hat/Fedora also have a useful script called **service** which can be used to start or stop any service for the current session. For example, to start Apache web server using the service script, the command is as follows –
  # service httpd start and to stop the service...
  # service httpd stop

Type **# service httpd status**, this should tell you that the httpd service is currently stopped.

Q2 d) What are different SSH security settings? Explain. (5)

SSH security settings:

- **Port:** By default, SSH listens on port 22. Every hacker knows this. This means that if you offer SSH services on port 22 of your server and it is connected directly to the Internet, you will see the first brute-force attack, launching a dictionary attack against your server within minutes. So if you're directly connected to the Internet, change the SSH port to something less obvious. I like putting it on port 443 (the average hacker expects HTTPS to be offered on that port and therefore will launch an HTTPS attack that will not work). The disadvantage of using port 443, however, is that you can't use HTTPS anymore. So, use any port you like, as long as it's not port 22.
- **ListenAddress:** By default, your SSH server offers its services on all IP addresses. In some cases, you might want to restrict this to only the IP addresses that are visible from the internal network and not from the Internet. If this is the case, change 0.0.0.0 to the specific IP address on which your SSH server should offer its services.
- **PermitRootLogin:** By default, this parameter allows the user

root to log in to your SSH server. This is not a good idea. If root is permitted to log in, the potential hacker only has to guess the root password. It's better to switch off root login by giving this parameter the value no. This means you'll have to connect as an ordinary user, and once connected, you'll have to use su - to escalate your privileges to the root level.

- **PasswordAuthentication:** By default, this parameter allows users to log in using passwords. If you have created public/private key pairs, you might consider switching off password authentication completely. Be careful, though: switching off password authentication also makes it difficult for you to log in from an unknown machine where your private key is not available.
- **AllowUsers:** This is a very nice parameter that is not in sshd_config by default. Everyone should use it and add a list of only those users you want to allow to log in to your SSH server. This makes it really hard for hackers, because they will have to guess the name of that user before starting this evil work!

Q2 e) Explain group configuration file. (5)

There are two group configuration files.

- **/etc/group**: Group account information.
- **/etc/gshadow**: Secure group account information.

- **/etc/group**
  - To work with groups we must understand group file /etc/group. It has one entry per line, and each line has the format: ***groupname*:*password*:*gid*:*userlist***
  - ***groupname***is the name of the group
  - ***password***is an optional field containing the encrypted group password
  - ***gid***is the numeric group ID number
  - ***userlist***is a comma-separated list of the user account names that comprise the group
  - If x appears in the password field, nonmembers of the group cannot join it using the newgrp command.

admins:x:507:Maya,sahil,meena

- *groupname* is admins; *password* is empty, meaning no group password has been set; *gid* is 503; and *userlist* is Maya,sahil,meena.

➢ **/etc/gshadow**

- 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:
- **Group name** — The name of the group.
- ***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.
- **Group administrators** — Group members listed here (in a comma delimited list) can add or remove group members using the gpasswd command**.**
- **Group members** — Group members listed here (in a comma delimited list) are regular, non- administrative members of the group.

Q2 f) What are file and directory permissions? How to change permissions. **(5)**

- The three basic permissions allow you to read, write, and execute. The effect of these permissions is different for files and directories.

Files

- The **read** permission gives you the right to open the fi le for reading.
- The **write** permission allows you to write/ modify that file.
- **Execute** permission is required to execute a file. It is never set

by default, which makes Linux almost completely immune to viruses.

- **Directory**
- **Read** allows you to list the contents of that directory. This permission does not allow us to read files in the directory.
- **Write** permission allows you to create and remove new subdirectories and files, but we need execute as well to go down into the directory.
- **Execute** permission indicates that the user can use the cd command to go to that directory.This permission is required to change to a particular directory or create files in that directory.
- **Use of Read, Write, Execute permissions.**

Permission	**Applied to files**	**Applied to Directories**
Read	**Open a file**	**List contents of the directory**
Write	**Change contents of a file**	**Create and delete files**
Execute	**Run a program file**	**Change to the directory**

Applying Read, Write, and Execute Permissions:

- Use the **chmod** command to apply permissions.
- We can use this command in two modes: relative and absolute. In the absolute mode, three digits are used to set the basic permissions.
- **Numerical Representaion of Permission.**

Permission	**Numerical Representation**
Read	**4**
Write	**2**
Execute	**1**

- For example, if we want to set read, write, execute for the user, read and execute for the group, and read and execute for others on the file /somefile, we would use the chmod command
  # chmod 755 /somefile.

- When using chmod this way, all current permissions are replaced by the permissions you set.
- If we want to modify permissions relative to the current permissions, we can use chmod in relative mode.
- When using chmod in relative mode, you work with three indicators to specify what you want to do. First you'll specify for whom you want to change permissions. To do this, you can choose between user (u), group (g), and others (o).
- Next you use an operator to add or remove permissions from the current mode or set them in an absolute way.
- At the end, you use r, w, and x to specify the permissions you want to set.
- When changing permissions in relative mode, you may omit the "to whom" part to add or remove a permission for all entities.
- For example, # **chmod +x somefile** would add the execute permission for all users
- When working in relative mode, you may use more complex commands as well.
- For example: **#chmod g+w,o-rsomefile**. This command add the write permission to the group and remove read for others.

Q3 a) What is Firewall? How to allow basic services through firewall? (5)

- Red Hat Enterprise Linux provides services on the Internet. That means that unauthorized users will try to attack our server and try to get access. To prevent this, we need to install a firewall.
- A firewall works through *packet inspection*.
- This means that the firewall screens incoming and outgoing packets to check whether the address, protocol, and port of the packet is either allowed or denied.
- Firewall works on 3rd, 4th and 5th layers of OSI Model.
- A firewall cannot check the user that has sent the packet and the actual data portion of the packet.
- In large company networks, firewalls are placed on router that

connects the network to the Internet. Everything behind the router is considered to be secure and doesn't need a firewall of its own. But, if a server is directly connected to the Internet, the server does require a firewall.

- ***Netfilter***is the default firewall in Linux. To configure Netfilter on Red Hat Enterprise Linux, you can use the system-config-firewall graphical tool or iptables in command line Interface.
- There are two main configuration files for netfilter firewall that are stored in the directory /etc/sysconfig.
- In the file **iptables**, you'll find all the rules that you've added to the firewall.
- In the file **iptables-config**, the configuration of the firewall is stored

Steps to allow basic services through firewall:

I. From the GNOME graphical interface, select System→Administration→Firewall or system- config-firewall from shell.

II. From the list of trusted services, select DNS, FTP, SSH, and WWW, and click Apply to save the configuration.

III. Close system-config-firewall, and open a shell prompt.

IV. Display the current status of the iptables service in the runlevels on your server.

# chkconfig | grep iptables iptables 0:off 1:off **2:on 3:on 4:on 5:on** 6:off

V. If the iptables service is not enabled then enable it.

# chkconfig iptables on

VI. Display current status of iptables.

# service iptables status

VII. If service is not enabled for the current session then start it.

# service iptables start

VIII. Display firewall rules.

# iptables -L –v

Q3 b) What is NAT? Give steps to Configure NAT. (5)

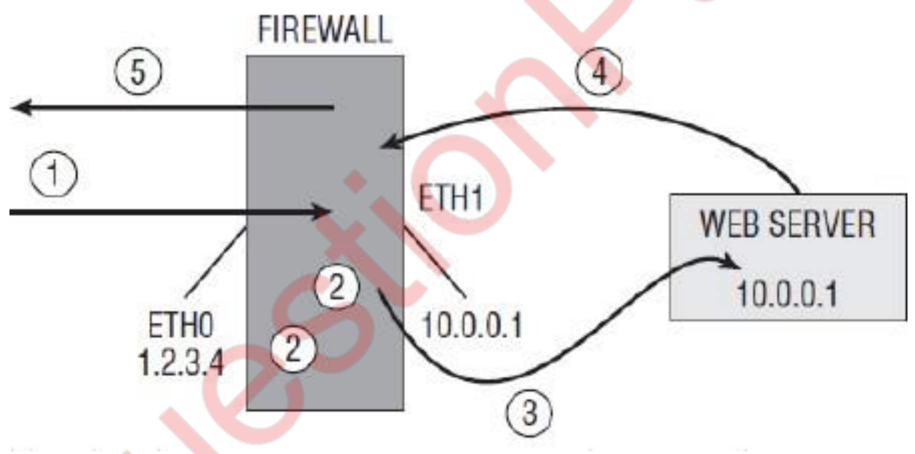
- Network Address Translation (NAT) is a common technique that can be used on routers to allow computers on the private network go out with one registered IP address on the public network.
- We can use NAT for three purposes:

➢ To change the source IP address to the IP address of the firewall before it is sent to the Internet, you need the **MASQUERADE** target.

➢ To change the source IP address of a specific host to the IP address of the firewall before it is sent to the Internet, you need the **SNAT** target.

➢ To redirect traffic that is sent to a specific IP address and port on the public IP address to an IP address and port on the private network, you need the **DNAT** target.

FIGURE : Packet processing by a NAT router



➢ A packet targeted at IP address 1.2.3.4:80 comes on the NAT router. This packet has to be sent to the web server that listens at IP address 10.0.0.10 on the internal network.

➢ The packet is processed by the firewall. To make sure DNAT is used to send the packet to the web server,use this command**:** **iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE**. You also need to enable routing (echo 1 > /proc/sys/net/ipv4/ip_forward) and set the policy for the routing chain to ALLOW: **iptables -P FORWARD ALLOW**.

➢ Because of the rule in the PREROUTING chain of the NAT table, here the packet can be delivered to the web server.

- The web server sends back the answer. However, the source address of the web server is used in the answer, and this address is unknown on the Internet.
- Thus, a masquerading rule must be defined.

<u>Steps to Configure NAT:</u>

- One router machine Use **iptables -F, iptables -P INPUT ALLOW, iptables –P OUTPUT ALLOW,** and **iptables -P FORWARD ALLOW**. Next use **service iptables save** to save this configuration.
- Repeat step 1 on the host computer, and use **service httpd stop** then **chkconfig httpd off** to make sure the web server is stopped on the host computer.
- On the host computer, enable routing by opening **/etc/sysctl.conf** with an editor.
- Make sure it includes the line **net.ipv.ip_forward = 1**, and use **sysctl -p** to thesysctl service to make the change effective.On the host, use **iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to-destination 192.168.1.1**.
- On the host, use **iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE** to enable IP masquerading.
- On another computer connected to the same network as your RHEL host, test thatyou can reach the HTTP port on the host. After verifying that this works, restart both the host computer and the virtualmachine to clear the NAT configuration.

Q3 c) Write steps for Creating and Managing Self-Signed Certificate. **(5)**

- To create and manage certificates, you can use the openssl command-line utility.
- We need to store the certificates that we are going to create, in the home directory of user root if we want them to be well protected, or else we can put the certificates in the directory /etc/pki/tls, which exists for this purpose by default.
- Within this directory, you need four subdirectories to store the certificates: **certs, newcerts, private, and crl**. These subdirectories are already created on Red Hat Enterprise Linux.

- **certs**: This is the location used to store all signed PKI certificates. The directory can be open for public access because it contains only public keys and no private ones.
- **newcerts:** This is where we temporarily store all new certificates until they are signed. After we've signed them, we can remove them from this location.
- **private**: This directory is used to store private keys. We must make sure that this directory is well protected because the private keys that are stored here are the proof of identity for our server. If the private keys are compromised, anyone can pretend to be your server.To protect the private keys, make sure that the private directory has permission mode 700and that user root is owner of this directory.
- **crl**: A *certificate revocation list (CRL)* is a list of certificates that are invalid. If you need to revoke certificates, copy them to this directory.

• Use the configuration file in /etc/pki/openssl.cnf. This file contains default settings that are used to create new certificates. This file contains default values that are not required to input from command line.
  # openssl req -newkey rsa:1024 -x509 -days 3650.

• The openssl command uses subcommands. req is the command that is used to generate a certificate-signing request. With that request, a new key iscreated with an RSA length of 1024 bits in which x509 and a validity of 10 years is used.

• Another way of creating a self-signed certificate is by using the **genkey** command. Thiscommand provides a text user interface that guides the user through the process of creatinga certificate.

Steps:

1.Install RPM Package that contains genkey command.
  # yum install -y crypto-utils mod_ssl

2.Execute genkey command with servername. This command automatically places the resulting PKI certificate and private key in the correct directories, which are /etc/pki/tls/private and /etc/pki/tls/certs.

# genkey --days 365 yourserver.example.com Select the key size you want to use. The default key size is set to 1024. If you need a higher security level, you canselect a bigger key size, but by doing that you will slow your system.

3.Move the mouse and type some command to help system generate some random bits that are necessary to produce the key.

4.Once the key is generated, genkey asks whether you want to create a certificate signing request. For self-signed certificate click No.

5.If you click yes then after creating the certificate signing request, send this request to the CA. Once you receive the signed certificate back, you can copy it to the /etc/pki/tls/certs directory and tell your application to use the signed certificate.

6.Set a passphrase for the private key. It is a good idea to protect your private keys with a passphrase, so select Yes and set a passphrase that is difficult to guess. The longer the passphrase, the harder it gets to guess its value after a private key has become compromised.

7.Now enter the appropriate information to identify your server. After you enter the appropriate identification, the public and private keys are written to the appropriate directories and are ready for use.

Q3 d) How to Encrypt Files with GPG? (5)

- GPG is commonly used to encrypt files.
- The base command to do this is **gpg –e yourfile**.
- The gpg command will next ask for a user ID. This is the ID of the user to which you want to send the encrypted file. This must be a user who is already in your GPGkeyring. Enter the name of each user for whom you want to encrypt the file on a separate line. When you're done, just press Enter on an empty line

Steps to encrypt file with GPG:

Step 1: Open a shell, and become user linda.

```
# su - linda
```

Step 2: As linda, copy the file /etc/hosts to your home directory.

```
cp /etc/hosts ~
```

Step 3: List the keys currently imported in Linda's environment, and note the exact name of the user lisa.

```
gpg –listkeys
```

Step 4: Encrypt the file **hosts**, when the user account is requested, enter the exact name of user lisa as you found it in the previous step of this exercise. Nextpress Enter on an empty line to complete the encryption procedure.

```
gpg -e hosts
```

Step 5: Copy the gpg file to the tmp directory where lisa can see and read it.

```
cp ~/hosts.gpg /tmp
```

Step 6: Use exit to log out as linda, and become user lisa.

```
exit
s – lisa
```

Step 7: As lisa, decrypt the hosts file.

```
gpg -d /tmp/hosts.gpg
```

Q3 e) What is NFS? What are advantages and disadvantages of NFS? (5)

- NFS, the Network File System, is the most common method for providing file sharing services on Linux and Unix networks. It is a distributed file system that enables local access to remote disks and file systems.
- The server component of NFS consists of the physical disks that contain the file systems you want to share and several daemons that make these shared file systems visible to and available for use by client systems on the network. When an NFS server is sharing a file system in this manner, it is said to be *exporting a file system*. Similarly, the shared file system is referred to as an *NFS export*.

NFS Advantages

1. The biggest advantage NFS provides is **centralized administration**, because It is much easier, to back up a file system stored on a server than it is to back up /home directories scattered throughout the network, on systems that are geographically dispersed, and that might or might not be

accessible when the backup is made.

2. NFS, especially when used with NIS, makes it **trivially simple to update key configuration files**, provide access to shared disk space, or limit access to sensitive data.
3. NFS can also **conserve disk space and prevent duplication of resources** because file systems that change infrequently or those are usually read-only, such as /usr, can be exported as read-only NFS mounts.
4. It is **easy to upgrade applications employed by users throughout a network** because it simply becomes a matter of installing the new application and changing the exported file system to point at the new application.
5. End users benefit from NFS when NFS is combined with NIS because users can **log in from any system**, even remotely, and still have access to their home directories and see a uniform view of shared data.
6. Users can **protect important or sensitive data** that would be impossible or time consuming to recreate by storing it on an NFS mounted file system that is regularly backed up.

NFS Disadvantages

1. As a distributed, network-based file system, NFS is sensitive to network congestion that is
   heavy network traffic slows down NFS performance.
2. Similarly, **heavy disk activity on the NFS server adversely affects NFS's performance**, In such cases, NFS clients seem to be running slowly because disk reads and writes take longer.
3. If the disk or system exporting vital data or application becomes unavailable for any reason (say, due to a **disk crash or server failure**), no one can access that resource.

NFS suffers **potential security problems** because its design assumes a trusted network, not a hostile environment in which systems are constantly being probed and attacked.

Q3 f) Give Steps to Configure Samba Server. **(5)**

- NFS is fast and convenient, but it has a major disadvantage: it works only between Linux machines.
- To get access to other types of clients, you need a file sharing protocol that is available on those clients as well.
- The ***Common Internet File System (CIFS)***is a protocol available on all windows machines, and it is offered by the Linux Samba server.
- *Samba* service can be used for file sharing, printers sharing and windows domain services.

Setting Up a Samba Server:

Step 1: Create a directory on the Linux file system and grant the appropriate permissions to this directory.

```
# mkdir /sambafiles
# chmod 777 /sambafiles
```

Step 2: Install the Samba packages

```
# yum -y install samba*
```

Step 3: Create the share in Samba.

- Open the file /etc/samba/smb.conf with an editor.

```
# vi /etc/samba/smb.conf
```

- Locate the workgroup parameter, and change it to **workgroup = MYSAMBA**.
- Go to the bottom of the configuration file, and add the following share configuration:

```
[sambafiles]
comment =
samba files
path =
/sambafiles
writable = yes
valid user = lucky
```

Step 4: check the syntax of the samba configuration file.

```
#testparm
```

Step 5: Create a Samba account, which makes the Samba server accessible for the Samba users.

# useradd lucky

- Don't set the password, because Samba users don't need a password on the Linux system.

# smbpasswd -a lucky

- The above command create Samba user lucky.

Step 6: (re)start the Samba service and make sure that it starts on every server boots.

# service smb restart # chkconfig smb on

Q4 a) What is DNS server? List and explain DNS server Types. (5)

- Computer only understand numbers (IP address) but it is verydifficult for humans to remember IP address.Domain Name System (DNS) provides solution by associating hostnames with IP addresses. It converts the people friendly names into computer friendly numbers called IP address.
- Each time we type a Web site's address into browser, the **Domain Name System (DNS)** converts it into IP address.

DNS Server Types

• The three types of local domain name servers are **Primary/master, Secondary/slave, and caching servers.**

Primary/Master name server:

- Every zone has at least a primary name server, also referred to as the master name server.
- This server is responsible for a zone and modifications can be made to this server.
- To increase redundancy in case the master name server goes down, zones are also often configured with a **secondary or slave name server**.
- To keep the primary and secondary name servers synchronized, a process known as zone transfer is used. In a zone transfer, a primary server can push its database to the secondary name server, or the secondary name server can request updates from

the primary name server.
The master contains all the information about the domain and gives this information when requested.

- A master server is listed as an **authoritative server** when it contains the information you are searching and it can provide that information.
- Apart from authoritative name servers, there are also **recursive name servers.** These are name servers that are capable of giving an answer, but they don't get the answer from their own database and use cache. This is possible because, by default, every DNS name server caches its most recent request.

Secondary/Slave name server:

- The slave DNS server is used as backup.
- In case the master server goes down or is not available the slave DNS server takes its place.
- This server contains the same information as the master DNS server and provides it when requested if the master server cannot be contacted.
- In DNS traffic, both primary and secondary name servers are considered to be authoritative name servers. This means that if a client gets an answer from the secondary name server about a resource record within the zone of that name server, it is considered to be an authoritative reply. This is because the answer comes from a name server that has direct knowledge of the resource records in that zone.

Cache-only name server:

- A caching server does not provide information to outside sources; it is used to provide domain information to other servers and workstations on the local network.
- The caching server remembers the domains that are accessed previously. Caching server speeds up searches since the domain information is already stored in memory.

Q4 b) Write steps for Setting Up a Cache-Only Name Server. (5)

- Domain Name System (DNS) is used for name address resolution.

- Name address resolution is the **conversion of people friendly names into computer friendly numbers called IP address**.
- The Berkeley Internet Name Domain (BIND) service is used to offer DNS services on Red Hat Enterprise Linux.

Cache-Only Name Server

- Cache-Only name server is useful when optimizing DNS requests in your network.
- To configure a cache-only name server install the BIND service and make sure that it allows incoming traffic.
- Once the resource record is found, BIND stores it in cache.This means that the next time a client needs the same information, it can be provided much faster. It will do the recursion on behalf of all clients.
- Normally, A **Forwarder** is also configured with cache-only name servers.

Steps:

1.Open a terminal, log in as root, and run **#yum install bind*** on the host computer to install the bind package.

2.With an editor, open the confi guration file /etc/named.conf.
# vi /etc/named.conf

3.Change the file to include the following parameters:

```
listen-on port
53 { any; };
allow-query {
any; };
```

This opens your DNS server to accept queries on any network inter face from any client.

4.In /etc/named.conf, change the parameter
dnssec-validation; to **dnsserver-validation no;**

5.Finally, insert the line
forwarders x.x.x.x

in the same configuration file and give it the value of the IP address of the DNS server we normally use for our Internet connection. This ensures that the DNS server of our Internet provider is used for DNS recursion and that requests are not sent directly to the name servers of the root domain.

6.Use the service named restart command to restart the DNS server.

7.From the RHEL host, use

```
dig redhat.com
```

You should get an answer, which is sent by your DNS server. We can see this in the SERVER line in the dig response.

Q4 c) Explain the components of email delivery process (5)

Message Transfer Agent:

- If a user sends a mail message to a user on another domain on the Internet, it's the responsibility of the MTA to contact the MTA of the other domain and deliver the message there.
- To find out which MTA serves the other domain, the DNS MX record is used.
- The MTA uses the Simple Mail Transfer Protocol (SMTP) to exchange mail messages with other MTAs on the Internet.
- When MTA receive a message, the MTA checks whether it is the final destination. If it is, it will deliver the message to the **local message delivery agent (MDA)**, which takes care of delivering the message to the mailbox of the user. If the MTA itself is not the final destination, the MTA relays the message to the MTA of the final destination.
- An MTA relay messages only for authenticated users or users who are known in some other way.
- If, for some reason, the MTA cannot deliver the message to the other MTA, it will queue it. Queuing means that the MTA stores the message in a local directory and will try to deliver it again later. As an administrator, you can flush the queues, which means that you can tell the MTA to send all queued messages now.
- Upon delivery, it sometimes happens that the MTA, which contacted an exterior MTA and delivered the message there, receives it back. This process is referred to as **bouncing**.
- In general, a message is bounced if it doesn't comply with the rules of the receiving MTA or if the destination user doesn't exist.

- For example: sendmail, postfix, Qmail.

Mail Delivery Agent/ Local delivery Agent:

- When MTA receives the mail, it delivers it to the mail delivery agent.
- MDA is the software component that takes care of delivering the mail message to the destination user's mailbox.
- The MDA delivers mail to the recipient's local message store, which by default on Red Hat Enterprise Linux is the directory **/var/spool/mail/<username>**.
- We can use the POP or IMAP server, which is an addition to a mail solution that makes it easier for users to get their messages, if they're not on the same machine where the MDA is running. POP server allows users to download messages on one machine whereas an IMAP server allows users to connect to the mail server and read the messages while they are online.

For example: **procmail**

Mail User Agent

- Finally, the mail message arrives in the mail user agent (MUA). This is the **mail client** that end users use to read their messages or to compose new messages.
- Users install MUA which allows them to work with email on their computer, tablet, or smartphone. Popular MUAa are Outlook, Evolution, and the Linux command-line Mutt tool, Pine.

Q4 d) Explain different Secure Internet Configurations for Postfix. **(5)**

- To make a secure Internet configuration, set parameters in the **/etc/postfix/main.cf** file.

myhostname

- This parameter specifies the name of this host. If not specified, it is set to the full DNS domain name (FQDN) of this host.

mydomain

- This parameter specifies the domain of this host. If not set, the domain name part of the FQDN is used.

myorigin

- This parameter determines the domain seen by the email recipient when receiving messages. The default is to use the FQDN of this host. This means that if user rahul on server host1.example.com sends a message, the recipient will see a message coming in from rahul@host1.example.com. use myorigin = $mydomain.

inet_interfaces

- This parameter specifies the IP addresses of the mail server to which it binds. By default, it is set to localhost only, which means that our mail server cannot receive messages from the Internet. Use inet_interfaces = all for sending mail to external users.

mydestination

- This parameter contains a list of all domains for which this server will receive messages. Messages that are addressed to users in other domains will be rejected.
- Make sure that this parameter contains a list of all domains serviced by this server. Change $mydomain to on.

mynetworks

- This parameter is optional. We can use it to specify the network address from which our MTA accepts messages for relaying without further authentication.

relayhost

- This parameter contains the name of a host that is used to relay all messages to. Use this if, for example, we want the mail server of your ISP to take care of all message delivery. To change any of these parameters make changes in /etc/postfix/main.cf, we can change the configuration file by hand and restart Postfix.

Q4 e) What are modes of Apache? Explain some performance parameters for these modes. (5)

- To make a secure Internet configuration, set parameters in the **/etc/postfix/main.cf** file. **Myhostname**
- This parameter specifies the name of this host. If not specified, it is set to the full DNS domain name (FQDN) of this host.

mydomain

- This parameter specifies the domain of this host. If not set, the domain name part of the FQDN is used.

myorigin

- This parameter determines the domain seen by the email recipient when receiving messages. The default is to use the FQDN of this host. This means that if user rahul on server host1.example.com sends a message, the recipient will see a message coming in from rahul@host1.example.com. use myorigin = $mydomain.

inet_interfaces

- This parameter specifies the IP addresses of the mail server to which it binds. By default, it is set to localhost only, which means that our mail server cannot receive messages from the Internet. Use inet_interfaces = all for sending mail to external users.

mydestination

- This parameter contains a list of all domains for which this server will receive messages. Messages that are addressed to users in other domains will be rejected.
- Make sure that this parameter contains a list of all domains serviced by this server. Change $mydomain to on.

mynetworks

- This parameter is optional. We can use it to specify the network address from which our MTA accepts messages for relaying without further authentication.

relayhost

- This parameter contains the name of a host that is used to relay all messages to. Use this if, for example, we want the mail server of your ISP to take care of all message delivery.

To change any of these parameters make changes in /etc/postfix/main.cf, we can change the configuration file by hand and restart Postfix.

Q4 f) What is Apache Module? How to add modules in Apache web server? (5)

- To extend the functionality of httpd process we can use **Dynamic Shared Objects (DSOs),** more commonly known as ***modules***.
- **Modules extend Apache's capabilities and add new features without requiring recompilation** because they can be loaded and unloaded at runtime, just as shared libraries.
- To include Apache modules, they first need to be installed. By default, some of the most common modules are installed to the **/etc/httpd/modules** directory.
- To tell Apache that it should load a specific module, we need to use the **LoadModule directive**.
- If a module is loaded, it can also have a specific configuration. There are three ways to load additional configurations for modules:

1. Use the IfModule directive in httpd.conf.
2. Put it in an include file.
3. If a module is common, its parameters can be entered in httpd.conf without further specification.

1. Using **IfModule** parameter in the httpd.conf file. This approach is more practical for modules that have a limited number of specific directives.

For example:

```
<IfModule
prefork.c>
StartServer
s 8
MinSpareServers 5
MaxSpareServers 20
ServerLimit 256
MaxClients 256
```

```
    MaxRequestsPerChild 4000
  </IfModule>
```

2. By default, some modules put their configuration in a separate configuration file and store that file in the directory **/etc/httpd/conf.d**.
The directive **include conf.d/*.conf** ensures that all configuration files where the name ends in
.conf are included by defaultwhen Apache starts.
3. If a module is very common and almost always used, its **parameters can simply be entered** in the httpd.conf file.

Q5 a) What are different ways to execute shell script? Explain in detail. (5)

There are three different ways to execute scripts:

- Make it executable, and run it as a program.
- Run it as an argument of the bash command.
- Source it.

Making the Script Executable:

The most common way to run a shell script is by making it executable. To do this with the hello script, use the following command:

```
chmod +x hello
```

After making the script executable, you can run it just like any other command. The only limitation is the exact location in the directory structure of your script. If it is in the search path, you can run it by typing any command. If it is not in the search path, you have to run it from the exact directory where it is located.

Running the Script as an Argument of the Bash Command:

The second option for running a script is to specify its name as the argument of the bash command. For example, the script hello would run using the command bash hello. The advantage of running the script this way is that there is no need to make it executable first. There’s one additional benefit too: if you run it this way, you can specify an argument to the bash command while running it. Make sure you are using a complete path to the location of the script when running it this way.

Sourcing the Script

The third way of running a script is completely different. You can source the script. By sourcing a script, you don't run it as a subshell. Rather, you include it in the current shell. This can be useful if the script contains variables that you want to be active in the current shell.

There are two ways to source a script. These two lines show you how to source a script that has the name settings:

.
settin
gs
source
settin
gs

It doesn't really matter which one you use because both are completely equivalent.

Q5 b) Write a script to which accept the number from user and print multiplication table. **(5)**

```
echo "Enter a Number"
read n
i=0
while [ $i -le 10 ]
do
    echo " $n x $i = `expr $n \* $i`"
    i=`expr $i + 1`
done
```

Q5 c) How will you setup bonding? (5)

- To set up a bonded network interface, you'll have to follow these steps:
    1. Identify the physical network cards that you want to configure in the bonding interface.
    2. Change the configuration for the physical network cards to make them slaves for the bonding interface.
    3. Create a configuration file for the bonding interface. Make sure the bonding kernel module is loaded.

1. Identify the physical network cards that you want to configure in the bonding interface.

- Assume, identified network card is em1 and em2. These are going to be used in a bonded configuration.

2. Change the configuration for the physical network cards to make them slaves for the bonding interface.

- Make sure that configuration file of em1 and em2 has following:

```
DEVICE=em1 (or em2 forsecond network card)
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

3. Create a configuration file for the bonding interface.

- The bonding interface devices will use names bond0, bond1, and so on, and the configuration file will be /etc/sysconfig/network-scripts/ifcfg-bond0 for bond device bond0 and so on.
- Configuration file for bond0

```
DEVICE=bond0
IPADDR=192.168.1.100
PREFIX=24
ONBOOT=yes
BOOTPROTO=none
```

```
USERCTL=no
BONDING_OPTS="mode=1 miimon=100"
```

4. Make sure the bonding kernel module is loaded.
   - To do this, create a file with the name **/etc/modprobe.d/bonding.conf**, and put in the line **alias bond0 bonding.**

After performing all of these steps, your bond device is ready for use. Restart the network, and use the **ip** a command to verify the working.

Q5 d) Explain Configuration of GFS2 File Systems. (5)

- If we have used Ext4 file system for Cluster service, then it is possible that it has failed when multiple nodes try to write in simultaneously.
- If multiple nodes in the cluster need access to the same fi le system at the same time, you'll need a cluster file system.
- Red Hat offers the **Global File System 2 (GFS2)** as the default cluster file system.
- Using GFS2 lets you to write to the same file system from multiple nodes at the same time.

Steps to configure GFS2 file system:

- ➢ One of the nodes, use fdisk to create a partition on the SAN device, and make sure to mark it as partition type 0x8e. Reboot both nodes to make sure the partitions are visible on both nodes, and verify this is the case before continuing.
- ➢ On both nodes, use **yum install -y lvm2-cluster gfs2-utils** to install **cLVM** and the **GFS** software.
- ➢ On both nodes, use **service clvmd start** to start the cLVM service and **chkconfig clvmd on** to enable it.
- ➢ On one node, use **pvcreate /dev/sdb3** to mark the LVM partition on the SAN device as a physical volume.
- ➢ Use **vgcreate -c y clusgroup /dev/sdb3** to create a cluster-enabled volume group.

- Use **lvcreate -l 100%FREE -n clusvolclusgroup** to create a cluster-enabled volume with the name clusvol.
- On both nodes, use **lvs** to verify that the cluster-enabled LVM volume has been created.
- Use **mkfs.gfs2 -p lock_dlm -t name_of_your_cluster:gfs -j 2 /dev/clusgroup/clusvol**. This will format the clustered LVM volume as a GFS2 file system.
  - Option -p tells mkfs to use the lock_dlm lock table.
  - Option -t specifies the name of your cluster.
  - Option -j 2 tells mkfs to create two GFS journals; one for each node that accesses the GFS volume.
- On both nodes, mount the GFS2 file system temporarily on /mnt, using **mount /dev/clusgroup/clusvol /mnt**. On both nodes, create some files on the file system, these files appear on other node also.
- Use **mkdir /gfsvol** to create a directory on which you can mount the GFS volume.
- Make the mount persistent by adding the following line to /etc/fstab: **/dev/clusgroup/clusvol /gfsvol gfs2 _netdev 0 0**
- Use **chkconfig gfs2 on** to enable the GFS2 service, which is needed to mount GFS2 volumes from /etc/fstab.

Reboot both nodes to verify that the GFS file system is mounted automatically.

Q5 e) Write steps for Configuring the DHCP Server for PXE Boot. **(5)**

PXE boot allows you to boot a server from the network card of the server.

- The PXE server then hands out a boot image, which the server you want to install uses to start the initial phase of the boot.
- Two steps are involved:
  1. **We need to install a TFTP server and have it provide a boot image to PXE clients.**
  2. **We need to configure DHCP to talk to the TFTP server to provide the boot image to PXE clients.**

Installing the TFTP Server

- First you need to install the TFTP server package using **yum -y install tftp-server**.
- TFTP is managed by the xinetd service, and to tell xinetd that it should allow access to TFTP, open the **/etc/xinetd.d/tftp** file and change the disabled parameter from Yes to No and restart the xinetdservice using **service xinetd restart**and include xinetd in your start **chkconfig tftp on**.

Steps:

- Use **yum install -y tftpserver** to install the TFTP server. Because TFTP is managedby xinetd, use **chkconfig xinetd on** to add xinetd to your runlevels.
- Open the configuration file **/etc/xinetd.d/tftp** with an editor, and change the line **disabled = yes to disabled = no.**
- If not yet installed, **install a DHCP server**. Open the configuration file /etc/dhcp/dhcpd.conf and write

```
option space pxelinux;
option pxelinux.magic code 208 = string;
option pxelinux.configfile code 209 = text;
option pxelinux.pathprefix code 210 = text;
option pxelinux.reboottime code 211 = unsigned integer 32 ;
subnet 192.168.1.0 netmask 255.255.255.0 {
option routers 192.168.1.1 ;
range 192.168.1.200 192.168.1.250 ;
class "pxeclients" {
match if substring (option vendor-class-identifier, 0, 9) =
"PXEClient";
next-server 192.168.1.1;
filename "pxelinux/pxelinux.0";
}
}
```

- Copy **syslinux<version>.rpm** from the Packages directory on the RHEL installation disc to **/tmp**. Extract the file **pxelinux.0** from it. This is an essential file for setting up the PXE boot environment. To extract the RPM file, use **cd /tmp** to goto the

/tmp directory, and from there, use **rpm2cpio syslinux<version>.rpm | cpio-idmv** to extract the file.

- Copy the **/usr/share/syslinx/pxelinux.0**file to **/var/lib/tftpboot/pxelinux.**
- Use **mkdir /var/lib/tftpboot/pxelinux/pxelinux.cfg** to create the directory in which you'll store the pxelinux configuration file.
- In **/var/lib/tftpboot/pxelinux/pxelinux.cfg**, create a file with the name **default** that contains the following lines:

```
default Linux
prompt 1
timeout 10
display boot.msg
label Linux
menu label ^Install RHEL
menu default
kernel vmlinuz
append initrd=initrd.img
```

- If you want to use a splash image file while doing the PXE boot, copy the **/boot/grub/splash.xpm.gz** file to **/var/lib/tftptboot/pxelinux/**.
- We can find the files **vmlinuz** and **initrd.img** in the directory images/pxeboot onthe Red Hat installation disc. Copy these to the directory **/var/lib/tftpboot/pxelinux/**.
- Use **service dhcpd restart** and **service xinetd restart**to restart the required services.
- Use **tail -f /var/log/message** to trace what is happening on the server. Connect a computer directly to the server, and from that computer, choose PXE boot in the boot menu. The computer starts the PXE boot and loads the installation image that you have prepared for it.
- If you want to continue the installation, when the installation program asks "What media contains the packages to be installed?" select URL. Next, enter the URL to the web

server installation image you created **http://server1.example.com/install**.

Q5 f)Explain kickstart file to perform an automated installation.(5)

- When you install a Red Hat system, a file with the name anaconda-ks.cfg is created in the home directory of the root user. This file contains most settings that were used while installing your computer.
- It is a good starting point if you want to try an automated kickstart installation.
- To specify that you want to use a kickstart file to install a server, you need to tell the installer where to find the file. If you want to perform an installation from a local Red Hat installation disc, add the linux ks= boot parameter while installing. (Make sure you include the exact location of the kickstart file after the = sign.) As an argument to this parameter, add a complete link to the file. For example, if you copied the kickstart file to the server1.example.com web server document root, add the following line as a boot option while installing from a DVD:

```
linux ks=http://server1.example.com/anaconda-ks.cfg
```

- To use a kickstart file in an automated installation from a TFTP server, you need to add the kickstart file to the section in the TFTP default file that starts the installation. In this case, the section that you need to install the server would appear as follows:

```
label Linux
    menu
    label
    ^Install
    RHEL
    menu
    default
    kernel vmlinuz
```

```
append initrd=initrd.img
ks=http://server1.example.com/anaconda-ks.cfg
```