Bachelor of Science in Information Technology

Linux System Administration

CBCS (APR-2019) Q.P Code: 57837

Q1 a) Why Red Hat Linux and Fedora has been so successful? Explain. (5)

- Red Hat went public in 1990, becoming the first Linux-based company on Wall Street. Because of the publicity stemming from its IPO, Red Hat and Linux received great exposure, and many companies started using it for their enterprise IT environments.
- It was initially used for applications, such as intranet web servers running Apache software. Soon Linux was also used for core financial applications. Today Linux in general and Red Hat Linux in particular is at the heart of the IT organization in many companies. Large parts of the Internet operate on Linux, using popular applications such as the Apache web server or the Squid proxy server. Stock exchanges use Linux in their real-time calculation systems, and large Linux servers are running essential business applications on top of Oracle and SAP. Linux has largely replaced UNIX, and Red Hat is a leading force in Linux.
- One reason why Red Hat has been so successful since the beginning is the level of support the company provides. Red Hat offers three types of support, and this gives companies the confidence they need to run vital business applications on Linux.

  The three types of Linux support provided by Red Hat are as follows:

  - **Hardware Support:** Red Hat has agreements with every major server hardware vendor to make sure that whatever server a customer buys, the hardware vendor will assist them in fixing hardware issues, when Red Hat is installed on it.

- **Software Support:** Red Hat has agreements with every major enterprise software vendor to make sure that their software runs properly on top of the Red Hat Linux operating system and that the enterprise software is also guaranteed to run on Red Hat Linux by the vendor of the operating system.
- **Hands-on Support:** This means that if a customer is experiencing problems accomplishing tasks with Red Hat software, the Red Hat Global Support organization is there to help them by fixing bugs and providing technical assistance.

- It is also important to realize that Red Hat is doing much more than just gathering the software pieces and putting them together on the installation media. Red Hat employs hundreds of developers who work on developing new solutions that will run on Red Hat Enterprise Linux in the near future.
- Even as Red Hat is actively developing software to be part of Red Hat Linux, it still is largely involved in the open source community. The most important approach to do this is by sponsoring the Fedora project. Fedora is a freely available Linux distribution that is completely comprised of open source software, and Red Hat is providing the funds and people to tackle this project. Both Red Hat and Fedora are free of charge; with Red Hat you pay only for updates and support.
- Fedora is used as a development platform for the latest and greatest version of Linux, which is provided free of charge for users who are interested. As such, Fedora can be used as a test platform for features that will eventually be included in Red Hat Enterprise Linux.
- Fedora makes an excellent choice to install on your personal computer, because it offers all the functions you would expect from a modern operating system—even some functions that are of interest only to home users.

Q1 b)Explain any five commands to view the contents of text files. (5)

- **Cat:** This command displays the contents of a fi le by dumping it to the screen. This can be useful if the contents of the fi le do not fi t on the screen. You will see some text scrolling by, and as the final result, you will see only the last lines of the fi le being displayed on the screen.
- **Tac:** This command does the same thing as cat but inverts the result; that is, not only is the name of tac the opposite of cat, but the result is the opposite as well. This command will dump the contents of a fi le to the screen, but with the last line first and the first line last.
- **Tail:** This command shows only the last lines of a text fi le. If no options are used, this command will show the last 10 lines of a text fi le. The command can also be modified to show any number of lines on the bottom of a fi le. For example, tail -n 2 /etc/passwd will show you the last two lines of the configuration file where usernames are stored. The option to keep tail open on a given log fi le is also very useful for monitoring what happens on your system. For example, if you use tail -f /var/log/messages, the most generic log file on your system is opened, and when a new line is written to the bottom of that fi le, you will see it immediately.
- **Head:** This command is the opposite of tail. It displays the first lines of a text file.
- **Less:** The last command used to monitor the contents of text fi les is less. This command will open a plain-text fi le viewer. In the viewer, you can browse the fi le using the Page Down key, Page Up key, or spacebar. It also offers a search capability. From within the less viewer, use /sometext to find sometext in the file. To quit less, use q.
- **More:** This command is similar to less but not as advanced.

Q1 c) Explain cut, copy, paste and deleting operation in Vi editor. (5)

- You do not need a graphical interface to use the cut, copy, and paste features. To cut and copy the contents of a fi le in a simple way, you can use the v command, which enters visual mode. In visual mode, you can select a block of text using the arrow keys. After selecting the block, you can cut, copy, and paste it.
  - Use d to cut the selection. This will remove the selection and place it in a buffer in memory.
  - Use y to copy the selection to the designated area reserved for that purpose in your server's memory.
  - Use p to paste the selection underneath the current line, or use P if you want to paste it above the current line. This will copy the selection you have just placed in the reserved area of your server's memory back into your document. For this purpose, it will always use your cursor's current position.
- Another action you will often do when working with vi is deleting text. There are many methods that can be used to delete text with vi. The easiest is from insert mode: just use the Delete and Backspace keys to get rid of any text you like. This works just like a word processor.

  Some options are available from vi command mode as well.
  - Use x to delete a single character. This has the same effect as using the Delete key while in insert mode.
  - Use dw to delete the rest of the word. That is, dw will delete anything from the current position of the cursor to the end of the word.
  - Use D to delete from the current cursor position up to the end of the line.
  - Use dd to delete a complete line.

Q1 d) What do you understand by mounting devices? Give example of mounting a USB flash drive. (5)

- Occasionally we need to make storage devices like USB flash drives, hard drives, or network shares available. To do this, you need to connect the device to a directory in the root file system. This process is known as *mounting* the device. If you're working from the graphical desktop, you'll notice that devices are mounted automatically.
- The graphical interface will create a subdirectory in the folder /media and make the contents of the USB drive accessible in that subdirectory. The problem, however, is that this works only from a graphical environment. If you're behind a server that was started in text mode, you'll need to mount your devices manually.

Mounting a USB Flash Drive

- Open a terminal, and make sure you have root privileges.
- Insert a USB flash drive in the USB port of your computer.
- Use dmesg to find the device name of the USB flash drive.
- Use fdisk -cul /dev/sdb to find current partitions on the USB flash driv
- Use mount /dev/sdb1 /mnt to mount the USB flash drive on the /mnt directory.
- Use cd /mnt to go into the /mnt directory.
- Type ls to verify that you see the contents of the USB flash drive.
- Now use umount /dev/sdb1 to try to dismount the USB flash drive. This won't work because you still are in the /mnt directory. You'll see the "device is busy" error message.
- Use cd without any arguments. This takes your current shell out of the /mnt directory and back to your home directory.
- At this point, you'll be able to dismount the USB flash drive successfully using umount /dev/sdb1.

Q1 e) How to set up your own repository? (5)

- If you have a Red Hat server installed that doesn't have access to the official RHN 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.
- First you'll copy all of the RPM fi les from the Red Hat installation DVD to a directory that you'll create on disk. Next you'll install and run the createrepo package and its dependencies. This package is used to create the metadata that yum uses while installing the software packages. While installing the createrepo package, you'll see that some dependency problems have to be handled as well.
    - Use mkdir /repo to create a directory that you can use as a repository in the root of your server's fi le system.
    - 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.
    - Use the cd /media/RHEL[Tab] command to go into the mounted DVD. Next use cd Packages, which brings you to the directory where all RPMs are by default. Now use cp * /repo to copy all of them to the /repo directory you just created. Once this is finished, you don't need the DVD anymore.
    - Now use cd /repo to go to the /repo directory. From this directory, type rpm –ivh createrepo[Tab]. This doesn't work, and it gives you a " Failed dependencies" error. To install createrepo, you fi rst need to install the deltarpm and python-deltarpm packages. Use rpm -ivh deltarpm[Tab] python-deltarpm[Tab] to install both of them. Next, use rpm -ivh createrepo[Tab] again to install the createrepo package.
    - Once the createrepo package has been installed, use createrepo /repo, which creates the metadata that allows you to use the /repo directory as a repository. This will take

a few minutes. When this procedure is finished, your repository is ready for use.

Q1 f) Explain installing and updating packages with yum. (5)

- Once you've found the package you were seeking, you can install it using yum install. For instance, if you want to install the network analysis tool nmap, after verifying that the name of the package is indeed nmap, you'd use yum install nmap to install the tool. Yum will then check the repositories to find out where it can find the most recent version of the program you're seeking, and after finding it, yum shows you what it wants to install. If there are no dependencies, it will show just one package. However, if there are dependencies, it displays a list of all the packages it needs to install in order to give you what you want. Next, type **Y** to confirm that you really want to install what yum has proposed, and the software will be installed.
- There are two useful options when working with yum install. The first option, -y, can be used to automate things a bit. If you don't use it, yum will first display a summary of what it wants to install. Next it will prompt you to confirm, after which it will start the installation. Use yum install -y to proceed immediately, without any additional prompts for confirmation.
- Another useful yum option is --nogpgcheck. If you occasionally don't want to perform a GPG check to install a package, just add --nogpgcheck to your yum install command. For instance, use yum install -y --nogpgcheck xinetd if you want to install the xinetd package without performing a GPG check and without having to confirm the installation.
- In some cases, you may need to install an individual software package that is not in a repository but that you've downloaded as an RPM package. To install such packages, you could use the command rpm -ivh packagename.rpm. However, this command doesn't update the yum database, and therefore it's not a good idea to install packages using the rpm command. Use yum localinstall instead. This will update the yum database and also check the repositories to try to fix all potential dependency

problems automatically, like you are used to when using yum install.

- If a package has already been installed, you can use yum update to update it. Use this command with the name of the specific package you want to update, or just use yum update to check all repositories and find out whether more recent versions of the packages you're updating are available.
- Normally, updating a package will remove the older version of a package, replacing it completely with the latest version. An exception occurs when you want to update the kernel. The command yum update kernel will install the newer version of the kernel, while keeping the older version on your server. It is useful because it allows you to boot the old kernel in case the new kernel is giving you problems.

Q2 a) What is use of fstab? Explain contents of fstab. (5)

- To make sure that the file system is mounted automatically across reboots, it must be placed in the /etc/fstab file. The /etc/fstab file is used to mount two different kinds of devices: file systems and system devices.
- When creating the /etc/fstab file, the device to be mounted must be referred. There are several ways of doing that. The easiest way is to use the device name, like /dev/sdb1, to indicate that the first partition on the second disk must be mounted.
- The disadvantage of this approach is that the names of these devices depend on the order in which they are detected while booting, and this order can change.
- Some server detects external USB drives before detecting internal devices that are connected to the SCSI bus.
- To avoid issues with the device names, RHEL partitions are normally mounted using the UUID(Universally Unique IDentifier) that is assigned to every partition. To find out the UUIDs od the device on the server, the blkid command is used.
- Contents of fstab
  - **/dev/device:** Indicates the partition to be mounted.

- **/dir/to/mount:** Designates the directory on which the partition is to be mounted.
- **Fstype:** Specifies the type of the file system.
- Parameters: Indicates the parameter to be supplied in the –o option of the mount command.
- **Fs_freq:** Tells the dump command how often the file system needs to be backed up.
- **Fs_passno:** Tells the fsck program at boot time to determine the order in which the file systems should be checked. File systems are always checked before they are mounted.

Q2 b) What is a swap space? Write procedure to add swap space. (5)

- Every server needs swap space, even if it's never going to use it. Swap space is allocated when your server is completely out of memory, and using swap space allows your server to continue to offer its services. Therefore, you should always have at least a minimal amount of swap space available. In many cases, it's enough to allocate just 1GB of swap space, just in case the server is out of memory. There are some scenarios in which you need more swap space.
  Here are some examples:
  - If you install on a laptop, you need RAM + 1GB to be able to close the lid of the laptop to suspend it. Typically, however, you don't use laptops for RHEL servers.
  - If you install an application that has specific demands in regard to the amount of swap space, make sure to honor these requirements. If you don't, you may no longer be supported. Oracle databases and SAP Netweaver are well-known examples of such applications.
- You would normally create swap space while installing the server, but you can also add it later. Adding swap space is a four-step procedure.
  - Make sure to create a device you're going to use as the swap device. Typically, this would be a partition or a

logical volume, but you can also use dd to create a large empty file. For the Linux kernel it doesn't matter—the kernel addresses swap space directly, no matter where it is.

- Use mkswap to format the swap device. This is similar to the creation of a file system on a storage device.
- Use swapon to activate the swap space. You can compare this to the mounting of the file system, which ensures you can actually put files on it.
- Create a line in /etc/fstab to activate the swap space automatically the next time you reboot your server.

Q2 c) What is NetworkManager configuration file? Explain any five network configuration file variables. (5)

- Whether you use the graphical NetworkManager or the text-based system-confi g-network, the changes you make are written to the same configuration fi les. In the directory /etc/sysconfig/network-scripts, you'll find a configuration fi le for each network interface on your server. The names of all of these files start with ifcfg- and are followed by the names of the specific network cards.
- In the network configuration scripts, variables are used to define different network settings.
  - **DEVICE :** Specifies the name of the device, as it is known on this **DEVICE :**

server.

- **NM_CONTROLLED:** Specifies whether the device is controlled by the NetworkManager service, which is the case by default.
- **ONBOOT:** Indicates that this device is started when the server boots.
- **TYPE:** Indicates the device type, which typically is Ethernet.
- **BOOTPROTO:** Set to dhcp if the device needs to get an IP address and additional configuration from a DHCP server. If set to anything else, a fixed IP address is used.

- **DEFROUTE:** If set to yes, the gateway that is set in this device is also used as the default route.
- **IPV4_FAILURE_FATAL:** Indicates whether the device should fail to come up if there is an error in the IPv4 configuration.
- **IPV6INIT:** Set to yes if you want to use IPv6.
- **NAME:** Use this to set a device name.
- **UUID:** As names of devices can change according to hardware configuration, it might make sense to set a universal unique ID (UUID). This UUID can then be used as a unique identifier for the device.
- **HWADDR:** Specifies the MAC address to be used. If you want to use a different MAC address than the one configured on your network card, this is where you should change it.
- **IPADDR:** Defines the IP address to be used on this inter face.
- **PREFIX:** This variable defines the subnet mask in CIDR format. The CIDR format defines the number of bits in the subnet mask and not the dot ted decimal number, so use 24 instead of 255.255.255.0.
- **GATEWAY:** Use this to set the gateway that is used for traffic on this network card. If the variable DEFROUTER is also set to yes, the router specified here is also used as the default router.
- **DNS1:** This parameter specifies the IP address of the first DNS server that should be used. To use additional DNS servers, use the variables DNS2 and, if you like, DNS3 as well.
- **USERCTL:** Set to yes if you want end users to be able to change the network configuration. Typically, this is not a very good idea on servers.

Q2 d) Write steps to enable and test the ssh server. (5)

- The SSH service is installed on your Red Hat Enterprise Linux server. It isn't enabled by default, however, so you should make sure to start it manually using the service sshd start command. After doing that, make sure that it is also started after a reboot of your server by using chkconfig sshd on. After performing these tasks, you can first do a basic connection test and connect to it using the ssh command.

Enabling and Testing the SSH Server

- From a terminal with root permissions, use the command service sshd start. In the unlikely event that this command shows an error, use yum install openssh-server to install the ssh server package.
- Use the chkconfig sshd on command to enable the SSH service, and add it to your server's run levels. This ensures that the SSH server also comes up after rebooting the server.
- Now it's time to test the SSH server. Open a new terminal window, and use ssh root@localhost to open an SSH session where you're logging in as root. Enter the password when prompted.
- You're now in an SSH session. In this example, you tested the connection from your own local machine. You can also test the connection from a remote machine. This will be discussed further in the sections that follow.
- Type exit to close the SSH sessions.

Q2 e) Explain different fields of /etc/passwd. (5)

Different fields are used in /etc/passwd.

- **Username:** The user's login name is stored in the first field in /etc/passwd. In older UNIX versions, there was a maximum-length limitation on login names, which was eight characters. In modern Linux distributions, such as Red Hat Enterprise Linux, this limitation no longer exists.
- **Password:** In the old days of UNIX, encrypted passwords were stored in this file. There is, however, one big problem with

passwords stored here—even if the password has been hashed, everyone is allowed to read /etc/passwd. Since this poses a security risk, passwords are stored in the confi guration file /etc/shadow nowadays, which is discussed in the next section.

- **UID:** As you have already learned, every user has a unique user ID. Red Hat Enterprise Linux starts numbering local user IDs at 500, and typically the highest number that is used is 60000 (the highest numbers are reserved for special-purpose accounts).
- **GID:** As discussed in the previous section, every user has a primary group. The group ID of this primary group is listed there. On Red Hat Enterprise Linux, every user is also a member of a private group that has the name of the user.
- **GECOS:** The General Electric Comprehensive Operating System (GECOS) field is used to include some additional information about the user. The field can contain anything you like, such as the department where the user works, the user's phone number, or anything else. This makes identifying a user easier for an administrator. The GECOS field is optional, and often you will see that it is not used at all.
- **Home Directory:** This field points to the directory of the user's home directory.
- **Shell:** The last field in /etc/passwd is used to refer to the program that is started automatically when a user logs in. Most often, this will be /bin/bash, but as discussed previously, every binary program can be referred to here as long as the complete path name is used.

Q2 f) Write short note on connecting to an LDAP Server. (5)

- An LDAP server is organized as a hierarchical structure, which looks a bit like the structure of domains and subdomains that is used in DNS. In many cases, the DNS structure is mapped one-to-one to the LDAP database—only the way it is written is different. The DNS domain referred to as example.com is referred to as dc=example, dc=com in LDAP.
- To connect to LDAP, you need to specify at a minimum what is used as the LDAP base DN. This is the branch of the

hierarchical structure in which you would expect to find user accounts. Also, you need to specify which LDAP server is used. This LDAP server provides access to the user database.

- To connect to the LDAP server, it is good practice to use TLS to encrypt connections. This is possible only if certificates are provided as well. Normally, every LDAP server is able to hand out its own certificates. As an administrator, you just have to specify where these certificates can be found.
- The second part of the configuration of LDAP authentication consists of selecting an authentication method. For best security, Kerberos passwords are used. As an alternative, you may also elect to use LDAP passwords. To log in using Kerberos, you need at least three parameters.
- **Realm** This is like a domain, but in Kerberos the domain name is referred to as a *realm*. The realm specifies where authentication should be handled.
- **KDCs** In Kerberos, a *key distribution center(KDC)* is used to hand out tickets that are needed while authenticating. This KDC is a specific server that has been configured as such.
- **Admin Servers** The admin server is the server that is used for administration tasks in a Kerberos environment. It is often the same as the KDC.
- Instead of entering these parameters manually, a Kerberos environment can also be set up to use DNS. DNS can be used to locate the KDC for a realm and also to resolve hosts to realms. This is typically the case when connecting to Kerberos in an Active Directory environment.

Q3 a) Define Tables, Chains, and Rules. Explain most common elements of rules. (5)

- Tables are the basic building blocks of a Netfilter firewall. Specific tables can be created to stipulate the specific functionality of the firewall. By default, the filter table is used. The NAT table is also frequently used.
- A table contains chains. A *chain* consists of a set of rules that is sequentially processed for each packet that enters the firewall

until it finds a match. The default strategy is "exit on match," which means that the firewall looks no further once the first rule that matches for a specific packet is found.

- Different elements can be used to specify properties of a packet in each rule. You don't have to specify each of these elements, but you are free to use them and make the rule more specific. Some elements are mandatory, though. Here is a list of some of the most common elements you'll find in rules:
    - **Modules** A *module* is an optional element that you can use in a rule. Modules offer enhancements to the Net filter firewall. They do that by loading a specific kernel module that adds functionality. A very common module in iptables rules is the *state* module, which looks at the state of a packet.
    - **Interface** On a server with multiple network cards, it makes sense to apply rules to specific interfaces only. However, if you're configuring your firewall on a typical server with one network card only, you can safely omit the interface specifi cation.
    - **IP Addresses** In a rule, you can allow or deny access to specific IP addresses or IP network addresses. For example, if you're in a school, you might want to differentiate between a safe internal network, which contains users who typically can be trusted, such as internal staff, and an unsafe internal network that is used by the students. IP addresses are also used in rules that configure port forwarding.
    - **Protocol** Most rules allow or deny access to specific ports. These ports are always connected to the UDP or TCP protocol. Therefore, if you want to state a specific port, you also need to indicate the protocol that is to be used. You can also include ICMP as the protocol to be used.
    - **Target** The target is also a mandatory component in a rule. A *target* specifies what needs to be done with a matching packet. Different targets can be used, of

which ACCEPT, DROP, REJECT, and LOG are the most important.

Q3 b) Write short note on IP Masquerading. (5)

- In *IP masquerading*, you can configure a server to connect your local network to the Internet. In this configuration, IP addresses from the private address ranges are used on the private network. These addresses cannot communicate on the Internet, but they will be translated to the public IP address on the interface that faces the Internet. This process is known as IP masquerading, also referred to as *Network Address Translation (NAT)*.
- The major benefit of using masquerading is that with just one public IP address, you can connect many devices on the private network to the Internet. IP masquerading is commonly used in home and corporate networks.
- To enable masquerading, you need to select the public interface. Once this interface is masqueraded, all packets are rewritten with the IP address of the public interface as the source address. To trace the packet back to its original sender, the NAT router maintains a NAT table. A port address is used to trace every connection in this NAT table. Once a reply to the packet comes back and has to be forwarded by the NAT router to the originating host, it will use the NAT table to find the address of the host from which the packet is originating, and it forwards the packet.
- You can also use *port forwarding* in combination with masquerading. This means you assign a port on the public interface of the NAT router and forward everything that comes in on that port to a specific host and port on the private network. You can use this approach if one of the computers on the private network is not directly reachable from the Internet, but it offers a specific service that you want to make available on the Internet.
- Users that want to use that service address the masquerading router and the specific port that is assigned on that router. Port forwarding will then forward the packet to the destination host.

Q3 c) Write steps to encrypt and decrypt files using GPG. (5)

We'll use the user accounts *Linda* and *Lisa* to practice encryption and decryption of files using GPG.

- Open a shell, and use su - Linda to become user Linda.
- As Linda, copy the fi le /etc/hosts to your home directory using cp /etc/hosts ~.
- Use gpg --listkeys to list the keys currently imported in Linda's environment, and note the exact name of the user Lisa.
- Encrypt the file using gpg -e hosts. When the user account is requested, enter the exact name of user Lisa as you found it in the previous step. Next press Enter on an empty line to complete the encryption procedure.
- Use cp ~/hosts.gpg /tmp to copy the gpg file to the tmp directory where Lisa can see and read it.
- Use exit to log out as Linda, and now use su - Lisa to become user Lisa. As Lisa, use gpg -d /tmp/hosts.gpg to decrypt the hosts file.

Q3 d) Explain subdirectories need to store the certificate. Write procedure to create your own Certificate Authority. (5)

- You need to store the certificates that you are going to create. You can do this in the home directory of user root if you want them to be well protected, or else you 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 have already been created on Red Hat Enterprise Linux.

  Here is an overview of how each of these directories is used:

  - **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 you temporarily store all new certificates until they have been signed. After you've signed them, you can remove them from this location.
- **Private**: This directory is used to store private keys. You must make sure that this directory is well protected because the private keys that are stored here are the proof of identity for your 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 700 and that user root is owner of this directory.
- **Crl:** A *certifti cate revocation list (CRL)* is a list of certificates that are invalid. If ever you need to revoke certificates, copy them to this directory.

Procedure to create your own Certificate Authority

- To create a certificate for your CA server, you can use the configuration file that you'll find in /etc/pki/openssl.cnf. This file contains default settings that are used to facilitate the creation of new certificates. Using this file makes creating certificates easier; all default values that are specified here don't have to be used on the command line. It's a good idea to read through this file so that you know which kinds of settings are available. Take special notice of the directory paths where you can find a reference to all the directories that are used while using the openssl command. If you're planning to put the certificates somewhere else on your server, you can change the HOME and dir variables from this file to reflect the directories you will be using

Partial contents of the /etc/pki/openssl.cnf file :

```
HOME = /root/rootCA
dir = /root/root-CA
...
certificate = $dir/root-CAcert.pem
...
private-key = $dir/root-CAkey.pem
```

- After checking the default values you want to use in openssl.cnf, you can start creating your own self-signed certificate. The following command allows you to create a certificate that uses a 1024-bit RSA key with a validity of 10 years:

```
openssl req -newkey rsa:1024 -x509 -days 3650
```

- In the previous code snippet, you can see that openssl is used as the master command. It is used with the req command. The openssl command uses subcommands. req is the command that is used to generate a certifi cate-signing request. With that request, a new key is created with an RSA length of 1024 bits in which x509 and a validity of 10 years is used.
- When creating a certificate that is to be used for a CA, it's a good idea to choose a long validity period. It should at least be longer than the period a normal certificate is typically valid, because when the CA certificate expires, all certificates that are signed by it will expire too.
- In this command, only a minimal number of parameters are used. The command could have been much longer, but that wasn't necessary, because the default options from openssl.cnf have been used. A more complex example follows, where a specific mention is made of where the private key and certifi cate must be written.

```
openssl req -newkey rsa:2048 -x509 -days 3650 -keyout
private/my-CAkey.pem -out
my-CAcert.pem
```

- When you are creating a key, the openssl command prompts for a passphrase. This will always occur, because a passphrase cannot be read from a configuration file. It is very important to select a secure passphrase. If the private key gets stolen, the passphrase is your last protection it has against being abused. If you don't want to be forced to enter a passphrase every time the service is restarted, you can just press Enter when prompted for the passphrase. This will set an empty passphrase, so the user won't be prompted.

Q3 e) Explain setting up NFSv4 with export options. (5)

- Setting up an NFS server is not difficult. It basically consists of two tasks:
  - Create the configuration file /etc/exports, which contains the shares you want to offer.
  - Start and enable the NFS service.
- You'll specify what you want to export in the file /etc/exports. The minimum setup also requires you to specify to whom you want to offer the share and what options you want to use on the share. The following line of code provides an example of a share that you can use in /etc/exports:

```
/data 192.168.1.0/24 (rw,no_root_squash)
```

- The line consists of three parts. First, there is the name of the directory you want to share, or /data in this example. Next, you specify to whom you want to offer access. In this example, access is offered to the network 192.168.1.0/24. As mentioned previously, the access restriction is always host-based, and many options are available to specify which hosts you want to give access. Here are some examples:
  - .example.com > This specifies all hosts in the DNS domain example.com. (This works only if reverse DNS is set up properly.)
  - * >This specifies all hosts with no restrictions.
  - server1 server2 > This specifies server1 and server2.
- The last part of the export line specifies which options to use on the export.

- **rw ->** Allows both read and w rite requests on the NFS volume.
- **ro ->** Allows only reads on the NFS share.
- **async ->** Improves performance by allowing the server to reply to requests before changes to the share are commit ted to storage. Using this option is faster but also increases the chance of losing data.
- **no_root_squash ->** Allows user root from an NFS client to work as user root on the NFS server as well. This should not be used because of security implications.

- **fsid ->** Gives each NFS share a unique ID, which makes it easier to track. This option is especially important in high-availability clusters.
- **all_squash ->** Maps the user ID of any user to the anonymous user, which ensures that the NFS user has minimum rights on the server.
- **anonuid ->** Specifies to which user to map anonymous user requests.

Q3 f) Write steps to set up a Samba server with example. (5)

- Setting up a Samba file server is fairly straightforward. To set one up, you need to do the following:
  - Create a directory on the Linux file system on the Samba server.
  - If needed, create Linux users and give the appropriate permissions to the directory you just created.
  - Install the Samba server.
  - Define the share in /etc/samba/smb.conf.
  - Create a Samba user account that has access to the share.
  - (Re)start the Samba service.
  - Tell SELinux to give access to the Samba share.

- There are different ways to offer user access on Samba. To set up a basic Samba server, you'll need both a Linux user and a Samba user. The Linux user is used for Linux permissions on the local Linux file system. A Windows user cannot authenticate with the credentials of a Linux user, however.
- This is why you'll also need a Samba user with the same name as the Linux user. Typically, the Windows user needs to work only with the Samba user, which also means that only the password of the Samba user is relevant. The Linux user is for local Linux access only. As an administrator, you only need to make sure that it has the appropriate rights.
- To define the Samba share, you'll put it in Samba's main configuration file, which is/etc/samba/smb.conf. The basic share definition is very simple: it gives a name to the share, and it tells

Samba what to share. A minimum share definition might appear as follows:

```
[myshare]
path=/mysharedfolder
```

- When defining a share, you can also use many options to define to whom and with which access permissions the share should be available. It could, for example, appear as follows:

```
[myshare]
path = /mydatafiles
comment = some shared files
allow hosts = 192.168.1.
writable = yes
public = yes
writelist = +mygroup
```

- In this example share, some options are added. First a comment is specified, which makes it easier for clients to identify the share when they are browsing the network for available services. Next the allow hosts parameter restricts access to hosts that have an IP address starting with 192.168.1 only. The option public = yes makes it a public share that is accessible by anyone who has a Samba account to authenticate to this server. It is also writable, which is indicated by the option writable = yes. To write to the share, though, a user must be a member of the local Linux group mygroup.

Q4 a) Explain the DNS lookup process. (5)

- To get information from a DNS server, a client computer is configured with a DNS resolver. This is the configuration that tells the client which DNS server to use. If the client computer is a Linux machine, the DNS resolver is in the configuration fi le /etc/resolv.conf.

- When a client needs to get information from DNS, it will always contact the name server that is configured in the DNS resolver to request that information. Because each DNS server is part of the worldwide DNS hierarchy, each DNS server should be able to handle client requests. In the DNS resolver, more than one name server is often configured to handle cases where the first DNS server in the list is not available.
- Let's assume that a client is in the example.com domain and wants to get the resource record for www.sander.fr. The following will occur:
  - When the request arrives at the name server of example.com, this name server will check its cache. If it has recently found the requested resource record, the name server will issue a recursive answer from cache, and nothing else needs to be done.
  - If the name server cannot answer the request from cache, it will first check whether a forwarder has been configured. A forwarder is a DNS name server to which requests are forwarded that cannot be answered by the local DNS server. For example, this can be the name server of a provider that serves many zones and that has a large DNS cache.
  - If no forwarder has been configured, the DNS server will resolve the name step-bystep. In the first step, it will contact the name servers of the DNS root domain to find out how to reach the name servers of the .fr domain.
  - After finding out which name servers are responsible for the .fr domain, the local DNS server, which still acts on behalf of the client that issued the original request, contacts a name server of the .fr domain to find out which name server to contact to obtain information about the sander domain.
  - After finding the name server that is authoritative for the sander.fr domain, the name server can then request the resource record it needs. It will cache this resource record and send the answer back to the client.

Q4 b) Write short note on DHCP (Dynamic Host Configuration Protocol). (5)

- The *Dynamic Host Configuration Protocol (DHCP)* is used to assign IP-related configuration to hosts in your network. Using a DHCP server makes managing a network a lot easier, because it gives the administrator the option to manage IP-related configuration on a single, central location on the network, instead of on multiple different hosts.
- Counter to common belief, DHCP offers much more than just the IP address to hosts that request its information. A DHCP server can be configured to assign more than 80 different parameters to its clients, of which the most commonly used are IP addresses, default gateways, and the IP address of the DNS name servers.
- When a client comes up, it will send a DHCP request on the network. This DHCP request is sent as a broadcast, and the DHCP server that receives the DHCP request will answer and assign an available IP address. Because the DHCP request is sent as a broadcast, you can have just one DHCP server per subnet. If multiple DHCP servers are available, there is no way to determine which DHCP server assigns the IP addresses. In such cases, it is common to set up failover DHCP, which means that two DHCP services together are servicing the same subnet, and one DHCP server completely takes over if something goes wrong.
- It is also good to know that each client, no matter which operating system is used on the client, remembers by default the last IP address it has used. When sending out a DHCP request, it will always request to use the last IP address again. If that IP address is no longer available, the DHCP server will give another IP address from the pool of available IP addresses.
- When configuring a DHCP server, it is a good idea to think about the default lease time. This is the amount of time that the client can use an IP address it has received without contacting the DHCP server again. In most cases, it's a good idea to set the default lease time to a rather short amount of time, which means

it doesn't take too long for an IP address to be given back to the DHCP server. This makes sense especially in an environment where users connect for a short period of time, because within the max-lease-time (two hours by default), the IP address is claimed and cannot be used by another client. In many cases, it makes sense to set the max-lease-time to a period much shorter than 7,200 seconds.

Q4 c) Explain different parameters used in the/etc/postfix/main.cf file. (5)

Different parameters used 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. This parameter is used as a variable in other parameters in the main.cf file, so it is useful to set it.
- **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 linda on server dfw.example.com sends a message, the recipient will see a message coming in from linda@dfw.example.com. This is often not what you want. To append the domain name only and not the entire FQDN, 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 your mail server cannot receive messages from the Internet. This is fi ne if the mail server only has to send messages and another server is used for email reception. However, you'll normally want to enable all inet_interfaces using inet_interfaces = all.
- **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. Also notice that in the default setting, message reception for $mydomain is off, so you'll need to change this.

- **mynetworks:** This parameter is optional. You can use it to specify the network address from which your MTA accepts messages for relaying without further authentication. It's a good idea to set this to your trusted network.
- **relayhost:** This parameter contains the name of a host that is used to relay all messages to. Use this if, for example, you want the mail server of your ISP to take care of all message delivery.

Q4 d) Explain components play role in the process of Internet mail. (5)

- Three components play a role in the process of Internet mail. First there is the *message transfer agent (MTA)*. The MTA uses the *Simple Mail Transfer Protocol (SMTP)* to exchange mail messages with other MTAs on the Internet.
- 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. Upon receiving 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. *Relaying* is a hot item in email delivery. Normally, an MTA doesn't relay messages for just anyone, but only for authenticated users or users who are known in some other way. If messages were relayed for everyone, this would likely mean that the MTA was being abused by spammers on the Internet.
- 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, but it can also be bounced if the destination user simply doesn't exist. Alternatively, it's nicer if an MTA is configured simply to generate an error if the message couldn't be delivered.

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

- Apache can be started in two different modes: the prefork mode and the worker mode. The prefork mode is the default mode. In this mode, a master httpd process is started, and this master process will start different httpd servers.
- As an alternative, the worker mode can be used. In this mode, one httpd process is active, and it uses different threads to serve client requests.
- Even if the worker mode is a bit more efficient with regard to resource usage, some modules cannot handle it, and therefore the prefork mode is used as default. However, if you need the best performance that httpd can offer and you don't use modules that are incompatible with worker mode, it's a good idea to use worker mode instead. Worker mode can be configured to serve more simultaneous processes.
- To change the default mode that Apache uses, you can modify the HTTPD parameter in /etc/sysconfig/httpd. To use the worker mode, you have to start the /usr/sbin/httpd. worker binary instead of /usr/sbin/httpd.
- To accomplish this, just remove the pound sign in front of the example line in /etc/sysconfig/httpd and restart the httpd process using service httpd restart.
- For both modes, you can set some performance parameters:

- **StartServers**: This is the number of server processes httpd should always start.
- **MinSpareServers:** This is the minimum amount of spare server processes that are kept. It is good to have a certain minimum because it allows httpd to serve client requests really fast. However, the minimum shouldn't be too high because each server uses system resources.
- **MinSpareThreads:** In worker mode, this is the minimum amount of spare threads that httpd should keep. You can see that it is set considerably higher than the MinSpareServers parameter in prefork mode.
- **MaxSpareServers and MaxSpareThreads**: This is the maximum amount of spare servers or threads that httpd should keep.
- **ServerLimit:** This is the total amount of server processes that can be started as a maximum. Note that the value of 256 is pretty high, and it should be sufficient for most servers.
- **MaxClients:** This is the maximum number of clients that can be connected. Note that in worker mode, one client can have several concurrent requests, which are opened simultaneously.
- **MaxRequestPerChild:** This is the number of requests that can be opened by a server process. In prefork mode, the maximum is capped at 4,000; in worker mode, there is no maximum setting.

Q4 f) How configure an Apache server to use LDAP? (5)

- Handling authentication based on a flat user file works, but it's hard to maintain if you have many users to whom you want to grant access.
- In such cases like this, it might be interesting to add user accounts to an LDAP server and configure your Apache server to get the credentials from LDAP. This type of authentication can be handled by using the mod_authnz_ldap module and associated parameters. The only requirement from the LDAP

side of the configuration is that the user account to which you'll refer has to exist.

- To configure an Apache server to use LDAP, the administrator needs to address two things. First the Apache server must be able to handle the LDAP certificate, and the basic authentication provider must be set to LDAP. This section of the configuration is in the generic part of httpd.conf so that it is available for all virtual servers' services of your Apache instance. Next the (virtual) server itself must be configured to use LDAP for authentication. To take care of this, you'll need to include the following in your httpd.conf file:
  LDAPTrustedGlobalCert CA_BASE64 /etc/httpd/your-ldap-server-certificate.crt
  AuthBasicProvider ldap
- The important part of the general segment of the confi guration is the LDAPTrustedGlobalCert parameter that tells Apache which LDAP certificate to use. This assumes that your LDAP server has its own certificate, which you have already copied to the appropriate location. After setting this up, you'll need to include the parameters shown below
  <Directory />
  AuthName Enter your credentials for access
  AuthType basic
  AuthBasicProvider ldap
  AuthLDAPUrl
  "ldap://yourserver.example.com/dc=example,dc=com" TLS
  Require valid-user
  </Directory>
- **AuthLDAPUrl** parameter specifi es the URL to use to get access to the LDAP user information. Note how this URL is comprised. The fi rst part is the fully qualified DNS name of the server, followed by the LDAP base context that is used to access user information. All other parameters are similar to the configuration of the basic user authentication that was discussed in the previous section.

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

There are three different ways to execute shell script.

- **Make it executable, and run it as a program.**
  - The most common way to run a shell script is by making it executable. 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. This means that if user linda created a script with the name hello in /home/linda, she has to run it using the command /home/ linda/hello. Alternatively, if she is already in /home/linda, she could use ./hello to run the script. In the latter example, the dot and slash tell the shell to run the command from the current directory.
- **Run it 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. It has to be in the current directory, or you would have to use a complete reference to the directory where it is located. This means that if the script is /home/linda/hello and your current directory is /tmp, you should run it using bash /home/linda/hello.
- **Source it.**

- 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. (This often happens in the scripts that are executed when you boot your computer.)
- If you source a script, you need to know what you're doing, or you may encounter unexpected problems. For example, if you use the exit command in a script that is sourced, it closes the current shell. Remember, the exit command exits the current script. To be more specific, it doesn't exit the script itself, but rather it tells the executing shell that the script is over and it has to return to its parent shell. Therefore, don't source scripts that contain the exit command.
- There are two ways to source a script. These two lines show you how to source a script that has the name settings:
  . settings
  source settings
- It doesn't really matter which one you use because both are completely equivalent.

Q5 b) What is boot loader? Explain content of grub.conf file. (5)

- When your server starts, it needs to know which operating system to start. A *boot loader* is loaded for this purpose. On Red Hat Enterprise Linux, the GRUB boot loader is used. On a standard installation, the first part of the boot loader program is installed in the master boot record, and from there the rest of the boot procedure is executed.
- The purpose of GRUB is to load the kernel and the initial RAM fi le system. The kernel is the heart of the operating system, which ensures that the hardware in the computer can be used. The initial RAM fi le system contains modules that are needed by the kernel to access the fi le system and load other modules.

To load the kernel and initial RAM file system, different sections can be included in grub.conf.

- The first thing that happens in each section is that the root device is referenced. This is *not* the device that contains the root file system; it is the device that GRUB will use to find all the files that are referenced in the grub.conf configuration file. The root device is referred to by its BIOS name, which in this case is hd0,0. This refers to the first partition on the first hard disk, which from the perspective of the kernel is known as /dev/sda1. GRUB has to use the BIOS name of the device because, at the initial boot stage when this file is read, the kernel name /dev/sda1 is unknown yet.
- Next the kernel is loaded and it is followed by a number of options that specify exactly how the kernel should be loaded.
- After specifying the name of the kernel, some kernel load options are specified. All options that appear in uppercase are variables that are passed to the init. If you're starting your server in troubleshooting mode, you can safely omit all of these parameters. The important boot parameters are listed next.

Q5 c) Explain different standard components (items) used in HA (High-Availability) cluster. (5)

When working with Red Hat high-availability add-ons, you'll use several different software components. The following sections introduce these components in order to prepare you to build the cluster.

- **Corosync**
  - *Corosync* takes care of the lower layers of the cluster. It uses the Totem protocol to ensure that all nodes in the cluster are still available.
  - If something goes wrong on the Corosync layer, it will notify the upper layers of the cluster, where the placement of services and resources is addressed. You'll also encounter the name OpenAIS, which is what was used before Corosync was adopted.

- The OpenAIS developers halted their work and joined forces with the Corosync developers.

➢ **Rgmanager**

- *Rgmanager* refers to the upper layers of the cluster. It consists of different processes, which ensure that services are running where they should be.
- If Corosync detects a failure on the cluster, the Rgmanager layer will move the services to a node where they can continue to do their work.

➢ **Pacemaker**

- Rgmanager is the traditional Red Hat HA cluster stack. There is another Linux-based HA solution, however, which is known as *Pacemaker*.
- It is a broadly adopted HA project, and Red Hat has also contributed to it. In RHEL 6, it is available as an unsupported technology preview solution. If testing goes well, it is likely that Pacemaker will be the default solution for the upper layers in the cluster in Red Hat Enterprise Linux 7.

➢ **Conga**

- *Conga* is the management interface that is used to create and manage clusters. It consists of two parts: ricci and luci. *Ricci* is the agent that should be installed on all nodes in the cluster, and *luci* is the management application that talks to ricci. As an administrator, you'll log in to the luci web interface to create and manage your cluster.

Q5 d) Write tips for Troubleshooting a Non-operational Cluster. **(5)**

Setting up a cluster involves connecting many components in the right way, and a small mistake may have huge consequences. If you don't succeed in getting the service operational, apply the following tips to try to get it working:

- Check the log files. The cluster writes many logs to /var/log/cluster, and one of them may contain a valuable hint as

to why the service isn't working. In particular, make sure to check /var/log/cluster/rgmanager.log.

- Don't perform your checks from the Conga interface only, because the information it gives may be faulty. Also, use clustat on both nodes to check the current service status, and verify that individual components have actually been started or not.
- From the Conga interface, disable the resource and try to activate everything manually. That is, use ip a a to add the IP address. Use mount to mount the file system, and use service httpd start to start the Apache service. This will probably allow you to narrow down the scope of the problem to one particular resource.
- If you have a problem with the file system resource, make sure to use /dev/disk naming, instead of device names like /dev/sdb2, which can change if something changes to the storage topology.
- If a service appears as disabled in both Conga and in clustat, use clusvcadm –e servicename to enable it. It may also help to relocate the service to another node. Use clusvcadm -r servicename -m nodename to relocate a service.
- Don't use the service command on local nodes to verify whether services are running. Use ps aux and grep for the process you are seeking..

Q5 e) Explain setting up the network installation server. (5)

Setting Up the Network Installation Server

- Insert the Red Hat Enterprise Linux installation DVD in the optical drive of your server.
- Use mkdir /www/docs/server1.example.com/install to create a subdirectory in the Apache document root for server1.example.com.
- Use cp -R * /www/docs/server1.example.com/install from the directory where the Red Hat Enterprise Linux installation DVD is mounted to copy all of the files on the DVD to the install directory in your web server document root.
- Modify the configuration fi le for the server1 virtual host in /etc/httpd/conf.d/server1.example.com, and make sure that it

includes the line Options Indexes. Without this line, the virtual host will show the contents of a directory only if it contains an index.html file.

- Use service httpd restart to restart the Apache web server.
- Star t a browser, and browse to http://server1.example.com/install. You should now see the contents of the installation DVD.
- Start Virtual Machine Manager, and create a new virtual machine. Give the virtual machine the name testnetinstall, and select Network Install when asked how to install the operating system.
- When asked for the installation URL, enter http://server1.example.com/install. The installation should now be started.
- You may now interrupt the installation procedure and remove the virtual machine. You have seen that the installation server is operational. It's time to move on to the next phase in the procedure.

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

- When you install a Red Hat system, a fi le 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 fi le 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
```

- You can also use a kickstart file while installing a virtual machine using Virtual Machine Manager

---