Created by PERjham / perjham@gmail.com

Requirements

You will need five RHEL 8 virtual machines to be able to successfully complete all questions.

One VM will be configured as an Ansible control node. Other four VMs will be used to apply playbooks to solve the sample exam questions. The following FQDNs will be used throughout the sample exam.

- controller.example.com – Ansible control node
- node1.example.com – managed host
- node2.example.com – managed host
- node3.example.com – managed host
- node4.example.com – managed host

There are some requiremens that should be met before proceeding further:

- controller.example.com server has passwordless SSH access to all managed servers (using the root user).
- node2.example.com server and node3.example.com have a 2GB secondary /dev/vdb disk attached.
- node4.example.com server has a 512MB secondary /dev/vdb and a 1GB terciary /dev/vdc disk attached.
- node1.example.com has 1.3GB of RAM
- node2.example.com, node3.example.com and node4.example.com have 2GB ofRAM
- There are no regular users created on any of the servers.

TASK 1: Ansible Installation and Configuration

Install ansible package on the controller node (including any dependencies) and configure the following:

- Create a regular user ansible with the password of redhat. Use this user for all sample exam tasks and playbooks, unless you are working on the task that requires creating the ansible user on inventory hosts. You have root access to all five servers.
- All playbooks and other Ansible configuration that you create for this sample exam should be stored in /home/ansible/plays.

Create a configuration file /home/ansible/plays/ansible.cfg to meet the following requirements:

- The roles path should include /home/ansible/plays/roles, as well as any other path that may be required for the course of the sample exam.
- The inventory file path is /home/ansible/plays/inventory.
- Privilege escalation is disabled by default.
- Ansible should be able to manage 10 hosts at a single time.
- Ansible should connect to all managed nodes using the ansible user.

Create an inventory file /home/ansible/plays/inventory with the following:

- node1.example.com is a member of the proxy host group.
- node2.example.com is a member of the webserver host group.
- node3.example.com is a member of the webserver host group.
- node4.example.com is a member of the database host group
- haproxy group is a member of the frontend group.
- webserver and database group are member of the backend group.

TASK 2: Configure Repository I

Create a playbook with the name 02-ftp-repo.yml to set up the controller host as a repository host. Make sure this host meets the following requirements, which must be done by the play.

- The RHEL 8 installation ISO is loop-mounted on the directory /ftp_repo.
- The firewalld service is enabled.
- The vsftpd service is started as well as enabled and allows anonymous user access to the /ftp_repo directory.

TASK 3: Configure Repository II

Create a playbook with the name 03-http-repo.yml to set up the controller host as a repository host. Make sure this host meets the following requirements, which must be done by the play.

- /http_repo exists and has all the content of the ISO.
- The firewalld service is enabled.
- The httpd service is started as well as enabled, brinds /http_repo as Document Root.

TASK 4: Ad-Hoc Commands I

Create a script with the name 04-setuphosts.sh that uses ad hoc commands to complete configuration on the managed servers. This includes:

- Creating a user with the name ansible.
- Creating a sudo configuration that allows user ansible to run tasks with root privileges.
- Create ssh key par in the user ansible.
- Configure the ssh key that allows connect through ssh with no password for user ansible.
- Using an ad-hoc command to call the appropriate module to test connectivity to the remote hosts
- Installing Python.

After running the adhoc script, you should be able to SSH into all inventory hosts using the ansible user without password, as well as a run all privileged commands.

TASK 5: Ad-Hoc Commands II

Create a script with the name 05-configure-repos.sh that configures the managed servers as repository clients to the repository server that you have set up in the previous tasks (2 and 3). This script must use adhoc commands and perform the following tasks:

- Disable any currently existing repository.
- Enable access to the BaseOS repository on controller.example.com through ftp.
- Enable access to the AppStream repository on controller.example.com through http.

TASK 6: File Content

Create a playbook 06-motd.yml that runs on all inventory hosts and does the following:

- The playbook should replace any existing content of /etc/motd with text. Text depends on the host group.
- On hosts in the proxy host group the line should be "Welcome to HAProxy server".
- On hosts in the webserver host group the line should be "Welcome to Apache server".
- On hosts in the database host group the line should be "Welcome to MySQL server".

TASK 7: Configure SSH Server

Create a playbook 07-sshd.yml that runs on all inventory hosts and configures SSHD daemon as follows:

- banner is set to /etc/motd
- X11Forwarding is disabled
- MaxAuthTries is set to 3

TASK 8: LVM I

Create a playbook with the name 08-setupstorage.yml that accomplishes the following tasks:

- On all servers that have a second hard drive, create a partition with all the size of the disk.
- Use this partition to set up an LVM volume group with the name vgdata that uses physical
- extents with a size of 8 MiB.
- In the vgdata volume group, create a logical volume with the name lvdata and a size of 1
- GiB.
- Format this logical volume with the xfs file system.
- Ensure the volume is mounted persistently on the directory /data.

TASK 9: LVM II

Create a playbook with the name 09-setupstorage2.yml that accomplishes the following tasks:

- On all servers do this. If the volume group vgdata does not exist, the playbook must return the message "vgdata does not exist".
- If the volume group exists the playbook must return the message "vgdata exist".
- If the volume group exists but has less than 1 GiB storage available, the playbook must show the message "insufficient disk space available".

TASK 10: Users and Groups I

You have been provided with the list of users below.

Use /home/ansible/plays/vars/10-user_list.yml file to save this content.

```
---
users:
  - username: alice
    uid: 1201
  - username: vincent
    uid: 1202
  - username: sandy
    uid: 2201
  - username: patrick
    uid: 2202
```

Create Ansible vault file /home/ansible/plays/vars/10-secret.yml. Encryption/decryption password is devops.

Add the following variables to the vault:

```
user_password with value of devops
database_password with value of devops
```

Create a playbook 10-users.yml that uses the vault file secret.yml and var file 10-user_list.yml to achieve the following:

- Users whose user ID starts with 1 should be created on servers in the webserver host group. User password should be used from the user_password variable.
- Users whose user ID starts with 2 should be created on servers in the database host group. User password should be used from the user_password variable.
- All users should be members of a supplementary group wheel.
- Shell should be set to /bin/bash for all users.
- Account passwords should use the SHA512 hash format.

TASK 11: Users and groups II

Create a playbook with the name 11-users.yml. This playbook should create users based on the input file /home/ansible/plays/vars/11-user_list2.yml. Manually create this file, and ensure it has the following contents:

```
users:
  - name: linda
    password: password
    department: profs
  - name: lisa
    password: secret
    department: profs
  - name: anna
    password: geheim
    department: students
```

On servers that are in the webserver group, users who have the department set to profs should be created, and the department should be set as a secondary group to the user. Also make sure that the password that is specified in users_pass.yml is set as a SHA256-encrypted password while creating the users.

TASK 12: Scheduled Tasks

Create a playbook 12-regular_tasks.yml that runs on servers in the proxy host group and does the following:

- A root crontab record is created that runs every hour.
- The cron job appends the file /var/log/time.log with the output from the date command.

TASK 13: Software Repositories

Create a playbook 13-repository.yml that runs on servers in the database host group and does the following:

- A YUM repository file is created.
- The name of the repository is mysql8-community.
- The description of the repository is "MySQL 8 YUM Repo".
- Repository baseurl is https://repo.mysql.com/yum/mysql-8.0-community/el/8/x86_64/.
- Repository GPG key is at http://repo.mysql.com/RPM-GPG-KEY-mysql.
- Repository GPG check is enabled.
- Repository is enabled.

TASK 14: Create and Work with Roles I

Create a role called 14-sample-mysql and store it in /home/ansible/plays/roles. The role should satisfy the following requirements:

- A primary partition number 1 of size 700MB on device /dev/vdc is created.
- An LVM volume group called vg_database is created that uses the primary partition created above.
- An LVM logical volume called lv_mysql is created of size 512MB in the volume group vg_database.
- An XFS filesystem on the logical volume lv_mysql is created.
- Logical volume lv_mysql is permanently mounted on /mysql_backups.
- mysql-community-server package is installed, previosuly disable dnf module mysql 8.0 from RHEL AppStream repo.
- Firewall is configured to allow all incoming traffic on MySQL port TCP 3306.
- MySQL server should be started and enabled on boot.
- MySQL server configuration file is generated from the my.cnf.j2 Jinja2 template with the following content:

```
[mysqld]
bind_address = IPV4ADDRESS
skip_name_resolve
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock

symbolic-links=0
sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

Create a playbook /home/ansible/plays/14-mysql.yml that uses the role and runs on hosts in the database host group.

TASK15: Create and Work with Roles II

Create a role called 15-sample-apache and store it in /home/ansible/plays/roles. The role should satisfy the following requirements:

- The httpd, mod_ssl and php packages are installed. Apache service is running and enabled on boot.
- Firewall is configured to allow all incoming traffic on HTTP port TCP 80 and HTTPS port TCP 443.
- Apache service should be restarted every time the file /var/www/html/index.html is modified.
- A Jinja2 template file index.html.j2 is used to create the file /var/www/html/index.html with the following content:

```
The address of the server is: IPV4ADDRESS
```

- IPV4ADDRESS is the IP address of the managed node.

Create a playbook /home/ansible/plays/15-apache.yml that uses the role and runs on hosts in the webserver host group.

TASK 16: Download Roles From Ansible Galaxy and Use Them

Use Ansible Galaxy to download and install geerlingguy.haproxy role in /home/ansible/plays/roles.

Create a playbook 16-haproxy.yml that runs on servers in the proxy host group and does the following:

- Use geerlingguy.haproxy role to load balance request between hosts in the webserver host group.
- Use roundrobin load balancing method.
- HAProxy backend servers should be configured for HTTP only (port 80).
- Firewall is configured to allow all incoming traffic on port TCP 80.
- If your playbook works, then doing "curl http://node1.example.com/" should return output from the web server (see task #15). Running the command again should return output from the other web server.

TASK 17: Security

Create a playbook 17-selinux.yml that runs on hosts in the webserver host group and does the following:

- Uses the selinux RHEL system role.
- Enables httpd_can_network_connect SELinux boolean.
- The change must survive system reboot.

TASK 18: Use Conditionals to Control Play Execution

Create a playbook 18-sysctl.yml that runs on all inventory hosts and does the following:

- If a server has more than 1500MB of RAM, then parameter vm.swappiness is set to 10.
- If a server has less than 1500MB of RAM, then the following error message is displayed:

```
Server memory less than 1500 MB
```

TASK 19: Use Archiving

Create a playbook 19-archive.yml that runs on hosts in the database host group and does the following:

- A file /mysql_backups/database_list.txt is created that contains the following line: dev,test,qa,prod.
- A gzip archive of the file /mysql_backups/database_list.txt is created and stored in /mysql_backups/archive.gz.

TASK 20: Work with Ansible Facts

Create a playbook 20-facts.yml that runs on hosts in the database host group and does the following:

- A custom Ansible fact server_role=mysql is created that can be retrieved from ansible_local.custom.sample_exam when using Ansible setup module.

TASK 21: Create and Use Templates to Create Customized Configuration Files

Create a playbook /home/ansible/plays/21-server_list.yml that does the following:

- Playbook uses a Jinja2 template from /home/ansible/plays/templates called server_list.j2 to create a file /etc/server_list.txt on hosts in the database host group.
- The file /etc/server_list.txt is owned by the ansible user.
- File permissions are set to 0600.
- SELinux file label should be set to net_conf_t.
- The content of the file is a list of FQDNs of all inventory hosts.

After running the playbook, the content of the file /etc/server_list.txt should be the following:

```
node1.example.com
node2.example.com
node3.example.com
node4.example.com
```

Note: if the FQDN of any inventory host changes, re-running the playbook should update the file with the new values.

TASK 22: Facts

Create a playbook 22-packages.yml with the name packagefacts.yml that gathers facts about packages that are installed on your managed nodes. Have the playbook generate a report with the name /root/packages.txt. In this report, package versions should be printed for the packages listed next. Make sure the report is printed in the format packagename=version, such as zlib=1.2.11. Do this for the following packages:

- kernel
- bash
- glibc

TASK 23: VAULT

Create a vault-encrypted password file with the name 23-cloudpass. In this file, set the variable CLOUDID to the value myid, and set the variable CLOUDPASS to the password cloudpass. Encrypt the vault file with the password cloudsecret. Store this password in the file vaultpass.txt in such a way that it can be used while using this cloudpass.yml file in a playbook.

Next, create a playbook 23-cloudpass.yml with the name usevault.yml. This playbook should import the variables that are set in the cloudpass.yml file and use them to create a clear text readable file with the name /root/cloudcreds.txt. In this file, the variables and their values should be listed in the VARNAME=value format, like CLOUDPASS=cloudpass. Ensure this playbook can use the vault password file that you have created

TASK 24: Roles

Use the RHEL system role that manages time in a playbook with the name 24-settime.yml. Ensure that control.example.com is used as the time server, and set the appropriate parameter that allows changing time even if a large difference exists between time on the managed machine and time on the time server. At the end of the playbook, verify that time is synchronized. If this is not the case, the playbook should print the text "Unfortunately time could not be synchronized"

TASK 25: Manage of dnf group

Write a playbook anmed 25-groups.yml that installs software packages:

- Perl and php on servers in the groups database, webserver, and haproxy.
- All packages from the package group "Development tools" on the group database.
- Servers in the group webserver that are fully updated.