This is a sample Ansible exam that I've created to prepare for the new RHCE exam EX294.

As with the real exam, no answers to the sample exam questions will be provided.

Requirements

There are 18 questions in total.

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.

- **ansible-control.hl.local – Ansible control node**
- **ansible2.hl.local – managed host**
- **ansible3.hl.local – managed host**
- **ansible4.hl.local – managed host**
- **ansible5.hl.local – managed host**

There are a couple of requirements that should be met before proceeding further:

- **ansible-control.hl.local server has passwordless SSH access to all managed servers (using the root user).**
- **ansible5.hl.local server has a 1GB secondary `/dev/sdb` disk attached.**
- **There are no regular users created on any of the servers.**

Tips and Suggestions

I tried to cover as many exam objectives as possible, however, note that there will be no questions related to dynamic inventory.

Some questions may depend on the outcome of others. Please read all questions before proceeding.

Note that the purpose of the sample exam is to test your skills. Please don't post your playbooks in the comments section.

Sample Exam Questions

Note: you have root access to all five servers.

Task 1: Ansible Installation and Configuration

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

- **Create a regular user automation with the password of devops. Use this user for all sample exam tasks and playbooks, unless you are working on the task #2 that requires**

creating the automation 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/automation/plays`.

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

- The roles path should include `/home/automation/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/automation/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 automation user.

Create an inventory file `/home/automation/plays/inventory` with the following:

- ansible2.hl.local is a member of the proxy host group.
- ansible3.hl.local is a member of the webservers host group.
- ansible4.hl.local is a member of the webservers host group.
- ansible5.hl.local is a member of the database host group.

Task 2: Ad-Hoc Commands

Generate an SSH keypair on the control node. You can perform this step manually.

Write a script `/home/automation/plays/adhoc` that uses Ansible ad-hoc commands to achieve the following:

- User automation is created on all inventory hosts (not the control node).
- SSH key (that you generated) is copied to all inventory hosts for the automation user and stored in `/home/automation/.ssh/authorized_keys`.
- The automation user is allowed to elevate privileges on all inventory hosts without having to provide a password.

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

Task 3: File Content

Create a playbook `/home/automation/plays/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 webservers 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 4: Configure SSH Server

Create a playbook `/home/automation/plays/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 5: Ansible Vault

Create Ansible vault file `/home/automation/plays/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**

Store Ansible vault password in the file `/home/automation/plays/vault_key`.

Task 6: Users and Groups

You have been provided with the list of users below.

Use `/home/automation/plays/vars/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 a playbook `/home/automation/plays/users.yml` that uses the vault file `/home/automation/plays/secret.yml` to achieve the following:

- **Users whose user ID starts with 1 should be created on servers in the webservers 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.**

- Each user should have an SSH key uploaded (use the SSH key that you created previously, see task #2).

After running the playbook, users should be able to SSH into their respective servers without passwords.

Task 7: Scheduled Tasks

Create a playbook `/home/automation/plays/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 8: Software Repositories

Create a playbook `/home/automation/plays/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 mysql80-community.
- The description of the repository is “MySQL 8.0 YUM Repo”.
- Repository baseurl is http://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 9: Create and Work with Roles

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

- A primary partition number 1 of size 800MB on device `/dev/sdb` 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 `/mnt/mysql_backups`.
- mysql-community-server package is installed.
- Firewall is configured to allow all incoming traffic on MySQL port TCP 3306.
- MySQL root user password should be set from the variable database_password (see task #5).
- 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 = {{ ansible_default_ipv4.address }}
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/automation/plays/mysql.yml` that uses the role and runs on hosts in the database host group.

Task 10: Create and Work with Roles (Some More)

Create a role called sample-apache and store it in `/home/automation/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/automation/plays/apache.yml` that uses the role and runs on hosts in the web servers host group.

Task 11: Download Roles From Ansible Galaxy and Use Them

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

Create a playbook `/home/automation/plays/haproxy.yml` that runs on servers in the proxy host group and does the following:

- Use geerlingguy.haproxy role to load balance requests between hosts in the web servers 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://ansible2.hl.local/” should return output from the web server (see task #10). Running the command again should return output from the other web server.

Task 12: Security

Create a playbook `/home/automation/plays/selinux.yml` that runs on hosts in the webservers 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 13: Use Conditionals to Control Play Execution

Create a playbook `/home/automation/plays/sysctl.yml` that runs on all inventory hosts and does the following:

- If a server has more than 2048MB of RAM, then parameter vm.swappiness is set to 10.
- If a server has less than 2048MB of RAM, then the following error message is displayed:

Server memory less than 2048MB

Task 14: Use Archiving

Create a playbook `/home/automation/plays/archive.yml` that runs on hosts in the database host group and does the following:

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

Task 15: Work with Ansible Facts

Create a playbook `/home/automation/plays/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 16: Software Packages

Create a playbook `/home/automation/plays/packages.yml` that runs on all inventory hosts and does the following:

- Installs tcpdump and mailx packages on hosts in the proxy host groups.

- Installs lsof and mailx packages on hosts in the database host groups.

Task 17: Services

Create a playbook `/home/automation/plays/target.yml` that runs on hosts in the webservers host group and does the following:

- Sets the default boot target to multi-user.

Task 18. Create and Use Templates to Create Customized Configuration Files

Create a playbook `/home/automation/plays/server_list.yml` that does the following:

- Playbook uses a Jinja2 template `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 automation 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:

```
ansible2.hl.local
ansible3.hl.local
ansible4.hl.local
ansible5.hl.local
```

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