RHEL-9 RHCE EXAM MODEL PAPER
EX294

Duration: 4Hrs
Total Marks: 300

Instructions:

control node: workstaion.lab.example.com

managed node: servera.lab.example.com,
serverb.lab.example.com,
serverc.lab.example.com,
serverd.lab.example.com

* All node root password is 'redhat' and Ansible control node user name is student.

* Create a directory 'ansible' under the path /home/student and all the playbooks should be under /home/student/ansible.

* All playbooks should be owned/executed by student and Ansible managed node user name is root.

* Ansible control node user password is student

* Unless advised password should be 'redhat' for all users

Ansible Automation Platform 2.2 is utility.lab.example.com
Credentials are admin, redhat

Note: In Exam, If they not given the Managed node user use the control node user as remote user

---

```
# ssh student@workstation
$ podman login utility.lab.example.com
$ sudo yum install ansible ansible-navigator
```

1. Install and Configure Ansible on the control node as follows:

  * Install the required packages.
  * Create a static inventory file called /home/student/ansible/inventory as follows:
     -- servera.lab.example.com is a member of the dev host group
     -- serverb.lab.example.com is a member of the test host group

-- serverc.lab.example.com is a member of the prod host group
-- serverd.lab.example.com is a member of the balancers host group
-- The prod group is a member of the webservers host group

* Create a configuration file called ansible.cfg as follows:
-- The host inventory file /home/student/ansible/inventory is defined
-- The location of roles used in playbooks is defined as /home/student/ansible/roles
-- The location of collections used in playbooks is defined as /home/student/ansible/collections

ANS:-

```
$ mkdir /home/student/ansible
$ cd /home/student/ansible

$ vim /home/student/ansible/inventory

[dev]
servera
[test]
serverb
[prod]
serverc
[balancers]
serverd
[webservers:children]
prod

$ vim /home/student/ansible/ansible.cfg

[defaults]
remote_user=student
inventory=/home/student/ansible/inventory
roles_path=/home/student/ansible/roles
collections_path=/home/student/ansible/mycollections
ask_pass=false

[privilege_escalation]
become=true
become_method=sudo
become_user=root
become_ask_pass=false

$ ansible all -m ping
```

2. Create a playbook adhoc.yml for configuring repository in all nodes.

i) Name=baseos
Description=’RH294 Description’
baseUrl=http://content/rhel9.0/x86_64/dvd/BaseOS
gpgcheck=true
gpgkey=http://content.example.com/rhel9.0/x86_64/dvd/RPM-GPG-KEY-redhat-release
Repository is enabled.

ii) Name = appstream
Description = RH294 Description
baseUrl= http://content/rhel9.0/x86_64/dvd/AppStream
GPG is enabled.
Gpgkey = http://content.example.com/rhel9.0/x86_64/dvd/RPM-GPG-KEY-redhat-release
Repository is enabled.

ANS:-

```
$ vim /home/student/ansible/yum_repo.yml
---
- name: Creating yum repository
  hosts: all
  tasks:
   - name: Create BaseOS Repository
     ansible.builtin.yum_repository:
      name: "baseos"
      description: ‘Rh294 Description’
      baseurl: http://content/rhel9.0/x86_64/dvd/BaseOS
      gpgcheck: yes
      gpgkey: http://content.example.com/rhel9.0/x86_64/dvd/RPM-GPG-KEY-redhat-release
      enabled: yes
   - name: Create Appstream Repository
     ansible.builtin.yum_repository:
      name: "appstream"
      description: ‘RH294 Description’
      baseurl: http://content/rhel9.0/x86_64/dvd/AppStream
      gpgcheck: yes
      gpgkey: http://content.example.com/rhel9.0/x86_64/dvd/RPM-GPG-KEY-redhat-release
      enabled: yes
```

```
$ ansible-navigator run adhoc.yml -m stdout
$ ansible all -a ‘yum repolist all’              (# verify the output)
```

3. Installing the Collection.
i) Create a directory "collections" under the /home/student/ansible.

ii) Using the url 'http://content/Rhce/ansible-posix-1.4.0.tar.gz' to install the ansible.posix collection under collection directory.
iii) Using the url 'http://content/Rhce/redhat-rhel_system_roles-1.0.0.tar.gz' to install the system roles collection under collection directory.

Note: In Exam, you need to install ansible collections also,

```
$ mkdir /home/student/ansible/collections
$ ansible-galaxy collection install  http://content/Rhce/ansible-posix-1.4.0.tar.gz -p collections
$ ansible-galaxy collection install  http://content/Rhce/redhat-rhel_system_roles-1.0.0.tar.gz -p collections
$ ansible-galaxy collection list (verify)
$ ls collections/ansible_collections (verify)
$ ansible-navigator collections (verify)
```

4. Create a playbook called packages.yml that:

- Installs the php and mariadb packages on hosts in the dev, balancers, and prod host groups
- Installs the Development Tools package group on hosts in the dev host group
- Updates all packages to the latest version on hosts in the dev host group

Ans:

```
# vim  packages.yml

---
- hosts: dev,balancers,prod
  remote_user: student
  become: true
  tasks:
    - ansible.built in.yum:
       name: "{{ item }}"
       state: latest
      loop:
        - php
        - mariadb
    - ansible.built in.yum:
       name: '@Development Tools'
       state: latest
      when: inventory_hostname in groups['dev']
    - ansible.built in.yum:
       name: '*'
       state: latest
      when: inventory_hostname in groups['dev']
```

```
$ ansible-navigator run paqckages.yml -m stdout
```

5. Install the RHEL system roles package and create a playbook called timesync.yml that:

-- Runs on prod and balancers hosts groups
-- Uses the timesync role
-- Configures the role to use the time server 172.25.254.254
-- Configures the role to set the iburst parameter as enabled

Ans:

```
$  yum  install rhel-system-roles -y

$ cp -r /usr/share/ansible/roles/rhel-system-roles.timesync /home/student/ansible/roles

$ ansible-galaxy list

$  vim  timesync.yml

---
- hosts: balancers,prod
  remote_user: admin
  become: true
  vars:
    timesync_ntp_servers:
      - hostname: 172.25.254.254
        iburst: true
  roles:
   - rhel-system-roles.timesync

$ ansible-navigator run timesync.yml -m stdout
```

6. 7.2 Create a playbook name selinux.yml and use system roles
  i) Set selinux mode as enforcing in all manage node

```
$ sudo yum install rhel-system-roles -y
$ cp -r /usr/share/ansible/roles/rhel-system-roles.selinux.yml /home/student/roles/
```

```
$ vim selinux.yml
---
- name: Configure selinux as enforcing mode
  hosts: all
  vars:
    - selinux_state: enforcing
  roles:
    - rhel-system-roles.selinux
```

```
$ ansible-playbook selinux.yml --syntax-check
```

```
$ ansible-navigator run selinux.yml -m stdout
```

```
$ ansible all -a "cat /etc/selinux/config"
```

Q6. Create a role called apache in /home/admin/ansible/roles with the following requirements

- The httpd package is installed, enabled on boot, and started
- The firewall is enabled and running with a rule to allow access to the web server
- A template file index.html.j2 exists (you have to create this file) and is used to create the file /var/www/html/index.html with the following output:

Welcome to HOSTNAME on IPADDRESS

- where HOSTNAME is the fully qualified domain name of the managed node and IPADDRESS is the IP address of the managed node.
- Create a playbook called httpd.yml that uses this role as follows:

  * The playbook runs on hosts in the webservers host group

Ans:

```
$  cd  roles
```

```
$  ansible-galaxy  init  apache
```

```
$  vim  apache/tasks/main.yml

---
# tasks file for roles/apache
- ansible.built in.yum:
    name: "{{ item }}"
    state: latest
  loop:
     - httpd
     - firewalld

- ansible.built in.service:
    name: "{{ item }}"
    state: started
    enabled: true
  loop:
    - httpd
    - firewalld

- ansible.posix.firewalld:
    service: http
    state: enabled
    immediate: true
    permanent: true

- ansible.built in.template:
    src: index.html.j2
    dest: /var/www/html/index.html
```

```
$  vim  apache/tasks/templates/index.html.j2
```

```
Welcome to {{ ansible_fqdn }} on {{ ansible_default_ipv4.address }}
```

```
$  cd ..

$  vim  httpd.yml
```

```
---
- hosts: webservers
  remote_user: admin
  become: true
  roles:
    - apache
```

```
$  ansible-navigator run httpd.yml  -m stdout
```

Q6. Use Ansible Galaxy with a requirements file called /home/admin/ansible/roles/install.yml to download and install roles to
/home/admin/ansible/roles from the following URLs:

-- http://content.example.com/rh294/role1.tar.gz
The name of this role should be balancer

-- http://content.example.com/rh294/role2.tar.gz
The name of this role should be phphello

Ans:

```
$  vim  /home/admin/ansible/roles/install.yml
```

```
---
- src: http://content.example.com/rh294/role1.tar.gz
  name: balancer
- src: http://content.example.com/rh294/role2.tar.gz
  name: phphello
```

```
$  ansible-galaxy  install   -r  roles/install.yml  -p  roles

$  ansible-galaxy  list
```

Q7. Create a playbook called balance.yml as follows:
The playbook contains a play that runs on hosts in the balancers host group and uses the balancer role.

- This role configures a service to load balance web server requests between hosts in the webservers host group.
- When implemented, browsing to hosts in the balancers host group (for example http://node5.example.com) should produce the following output:
  Welcome to node3.example.com on 192.168.10.z

- Reloading the browser should return output from the alternate web server:

  Welcome to node4.example.com on 192.168.10.a

* The playbook contains a play that runs on hosts in the webservers host group and uses the phphello role.

When implemented, browsing to hosts in the webservers host group with the URL /hello.php should produce the following output:

Hello PHP World from FQDN

where FQDN is the fully qualified domain name of the host.

For example, browsing to http://node3.example.com/hello.php, should produce the following output:

Hello PHP World from node3.example.com

along with various details of the PHP configuration including the version of PHP that is installed.

* Similarly, browsing to http://node4.example.com/hello.php, should produce the following output:

Hello PHP World from node4.example.com

along with various details of the PHP configuration including the version of PHP that is installed.

Ans:

```
$ vim  balance.yml
```

```
---
- hosts: webservers
  remote_user: admin
  become: true
  roles:
    - phphello
- hosts: balancers
  remote_user: admin
  become: true
  roles:
    - balancer
```

```
$  ansible-navigator run  balance.yml  -m stdout
```

Q8. Create a playbook called web.yml as follows:

* The playbook runs on managed nodes in the dev host group
* Create the directory /webdev with the following requirements:
- membership in the apache group
* regular permissions: owner=read+write+execute, group=read+write+execute, other=read+execute
  special permissions: set group ID

* Symbolically link /var/www/html/webdev to /webdev
* Create the file /webdev/index.html with a single line of text that reads: Development

Ans:

```
$ vim   web.yml

---
- hosts: dev
  remote_user: admin
  become: true
  tasks:
   - ansible.builtin.file:
      path: /webdev
      state: directory
      mode: '2775'
      group: apache
      setype: httpd_sys_content_t
   - ansible.builtin.file:
      src: /webdev
      dest: /var/www/html/webdev
      state: link
      setype: httpd_sys_content_t
   - ansible.builtin.copy:
      content: "Development"
      dest: /webdev/index.html
      setype: httpd_sys_content_t
  - ansible.posix.firewalld:
       service:http
       state: enabled
       immediate: true
       permanent: true
```

```
$ ansible-navigator run httpd.yml -m stdout
```

Q9. Create an Ansible vault to store user passwords as follows:

- * The name of the vault is valut.yml
- * The vault contains two variables as follows:
  - - dev_pass with value wakennym
  - - mgr_pass with value rocky

- * The password to encrypt and decrypt the vault is atenorth
- * The password is stored in the file /home/admin/ansible/password.txt

Ans:

```
$  vim   vault.yml

---
dev_pass: wakennym
mgr_pass: rocky


$ echo "atenorth" >> password.txt


$  ansible-vault  encrypt  vault.yml  --vault-password-file  password.txt
```

Q10. Generate a hosts file:

* Download an initial template file called hosts.j2 from http://192.168.10.254/ex407/ to /home/admin/ansible/ Complete the template so that it can be used to generate a file with a line for each inventory host in the same format as /etc/hosts

* Create a playbook called gen_hosts.yml that uses this template to generate the file /etc/hosts on hosts in the dev host group.

* When completed, the file /etc/hosts on hosts in the dev host group should have a line for each managed host:

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
```

```
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.10.x node1.example.com node1
192.168.10.y node2.example.com node2
192.168.10.z node3.example.com node3
192.168.10.a node4.example.com node4
192.168.10.b node5.example.com node5
```

Ans:

```
$  wget  <url>
```

```
$ vim  gen_hosts.yml
```

```
---
- hosts: all
  remote_user: admin
  become: true
  tasks:
   - ansible.builtin.template:
      src: hosts.j2
      dest: /etc/hosts
     when: inventory_hostname in groups['dev']
```

```
$ vim  hosts.j2  ( add a for loop )
```

```
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

{% for host in groups['all'] %}
{{ hostvars[host]['ansible_default_ipv4']['address'] }} {{ hostvars[host]['ansible_hostname'] }} {{ hostvars[host]['ansible_fqdn'] }}
{% endfor %}
```

```
$ ansible-navigator run  gen_hosts.yml -m stdout
```

Q11. Create a playbook called hwreport.yml that produces an output file called /root/hwreport.txt on all managed nodes with the following information:

-- Inventory host name
-- Total memory in MB
-- BIOS version
-- Size of disk device vda
-- Size of disk device vdb

Each line of the output file contains a single keyvalue pair.

* Your playbook should:

-- Download the file hwreport.empty from the URL http://192.168.10.254/ex407/hwreport.empty and save it as /root/hwreport.txt

-- Modify with the correct values.

NOTE: If a hardware item does not exist, the associated value should be set to NONE

Ans:

```
# vim  hwreport.yml

---
- hosts: all
  tasks:
    - ansible.builtin.get_url:
        url: http://content.example.com/rh294/hwreport.empty
        dest: /root/hwreport.txt
    - ansible.builtin.replace:
        regexp: "{{ item.src }}"
        replace: "{{ item.dest }}"
        dest: /root/hwreport.txt
      loop:
        - src: "hostname"
          dest: "{{ ansible_hostname }}"
        - src: "biosversion"
          dest: "{{ ansible_bios_version }}"
        - src: "memory"
          dest: "{{ ansible_memtotal_mb }}"
        - src: "vdasize"
          dest: "{{ ansible_devices.vda.size }}"
        - src: "vdbsize"
          dest: "{{ ansible_devices.vdb.size }}"

     - ansible.builtin.replace:
        regexp: "{{ item.src }}"
        replace: "{{ item.dest }}"
        dest: /root/hwreport.txt
      loop:
        - src: "hostname"
          dest: "{{ ansible_hostname }}"
        - src: "biosversion"
          dest: "{{ ansible_bios_version }}"
        - src: "memory"
          dest: "{{ ansible_memtotal_mb }}"
        - src: "vdasize"
          dest: "{{ ansible_devices.vda.size }}"
        - src: "vdbsize"
          dest: NONE
     When: ansible_devices.vdb.size is not defined
```

```
$  ansible-navigator  hwreport.yml -m stdout
```

Q12. Modify file content. Create a playbook called /home/admin/ansible/modify.yml as follows:

* The playbook runs on all inventory hosts
* The playbook replaces the contents of /etc/issue with a single line of text as follows:

  - On hosts in the dev host group, the line reads: Development
  - On hosts in the balancers host group, the line reads: Test
  - On hosts in the prod host group, the line reads: Production

Ans:

```
$ vim   modify.yml
```

```
---
- hosts: all
  tasks:
   - ansible.builtin.copy:
      content: "Development"
      dest: /etc/issue
    when: inventory_hostname in groups['dev']
  - ansible.builtin.copy:
      content: "Test"
      dest: /etc/issue
    when: inventory_hostname in groups['balancers']
  - ansible.builtin.copy:
      content: "production"
      dest: /etc/issue
    when: inventory_hostname in groups['prod']
$ ansible-navigator run modify.yml -m stdout
```

Q13. Rekey an existing Ansible vault as follows:

* Download the Ansible vault from "http://content.example.com/rh294/secret.yml"
* The current vault password is curabete
* The new vault password is newvare
* The vault remains in an encrypted state with the new password

Ans:

```
$  wget  http://content.example.com/rh294/secret.yml

$  ansible-vault  rekey  secret.yml
```

Q14. Create user accounts. A list of users to be created can be found in the file called user_list.yml which you should download from "http://content.example.com/rh294/user_list.yml" and save to /home/admin/ansible/.

* Using the password vault created elsewhere in this exam, create a playbook called create_user.yml that creates user accounts as follows:
* Users with a job description of developer should be:
    * created on managed nodes in the dev and balancers host groups assigned the password from the dev_pass variable a member of supplementary group devops

* Users with a job description of manager should be:
    * created on managed nodes in the prod host group assigned the password from the mgr_pass variable a member of supplementary group opsmgr

* Passwords should use the SHA512 hash format. Your playbook should work using the vault password file created elsewhere in this exam.

Ans:

```
$  wget  http://content.example.com/rh294/user_list.yml

$ vim  create_user.yml

---
- hosts: all
  vars_files:
    - user_list.yml
    - valut.yml

  tasks:
   - group:
      name: "{{ item }}"
      state: present
     loop:
      - devops
      - opsmgr
   - ansible.builtin.user:
      name: "{{ item.name }}"
      state: present
      groups: devops
      password: "{{ dev_pass | password_hash ('sha512') }}"
     loop: "{{ user }}"
     when: (inventory_hostname in groups['dev'] or inventory_hostname in groups['balancers'])
and item.job == "developer"

   - ansible.builtin.user:
      name: "{{ item.name }}"
      state: present
      groups: opsmgr
      password: "{{ mgr_pass | password_hash ('sha512') }}"
     loop: "{{ user }}"
     when: inventory_hostname in groups['prod'] and item.job == "manager"


$  ansible-navigator  create_user.yml -m stdout   --vault-password-file=password.txt
```

17. Create a logical volume named data of 1500M size from the volume group research and if 1500M size is not created, then atleast it should create 800M size.

i) Verify if vg not exist, then it should debug msg "vg not found" .
ii) 1500M lv size is not created, then it should debug msg "Insufficient size of vg" .
iii) If Logical volume is created, then assign file system as "ext4" .
iv) Do not perform any mounting for this LV.
iv) The playbook name lvm.yml and run the playbook in all nodes.

Playbook: --

```
$ vim lvm.yml
---
- name: Creating LVM storage
  hosts: all
  ignore_errors: yes
  tasks:
   - name: create a logical volume
     community.general.lvol:
       lv: data
       vg: research
       size: 1500
   - name: display message
     ansible.builtin.debug:
       msg: "vg not found"
     when: ansible_lvm.vgs.research is not defined
   - name: display message lv
     ansible.builtin.debug:
       msg: "Insufficient size of vg"
     when: ansible_lvm.vgs.research.size_g < 1.5
   - name: create lv with 800M
     community.general.lvol:
       lv: data
       vg: research
       size: 800
     when: ansible_lvm.vgs.research.size_g < 1.5
   - name: formate with file system
     community.general.filesystem:
       fstype: ext4
       dev: /dev/research/data
     when: ansible_lvm.vgs.research.size_g < 1.5
```

```
$ ansible-navigator run lvm.yml -m stdout
$ ansible all -m command -a 'lsblk'
```

16. Create a cronjob for user student in all nodes, the playbook name crontab.yml and the job details are below
  i) Every 2 minutes the job will execute logger "EX294 in progress"

```
# vim crontab.yml
---
- name : Create a cronjob
  hosts: all
  tasks:
  - name: Cronjob for logger
    ansible.builtin.cron:
      name: Create logger
      user: student
      minute: "*/2"
      job: logger "EX294 in progress"
      state: present
```

```
$ ansible-navigator run crontab.yml -m stdout
```

```
$ ansible all -a "ls /var/spool/cron/"
```

```
$ ansible all -a "crontab -lu student"
```