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 playbook should be under /home/student/ansible.

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

* 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

Cridentials 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
```

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

```
# sudo yum install ansible-navigator ansible tree vim -y (In Exam it will work)
```

```
podman login utility.lab.example.com
username: admin
password: redhat
```

```
# vim /home/student/.vimrc
 set ai ts=2 et cursorcolumn
```

```
# source /home/student/.vimrc
```

```
# mkdir /home/student/ansible
# cd /home/student/ansible
# vim /home/student/ansible/inventory
```

```
[dev]
servera.lab.example.com
[test]
serverb.lab.example.com
[prod]
serverc.lab.example.com
[balancers]
serverd.lab.example.com
[webservers:children]
prod
```

```
# vim /home/student/ansible/ansible.cfg
```

```
[defaults]
remote_user=devops
inventory=/home/student/ansible/inventory
roles_path=/home/student/ansible/roles
collections_paths=/home/student/ansible/collections

[privilege_escalation]
become=true
```

```
# ansible all -m command -a 'id'
```

(you should get the root user as output)

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

i) Name = baseos
Description = Baseos Description

Url = http://content/rhel9.0/x86_64/dvd/BaseOS

GPG is enabled.

Gpgkey = http://content.example.com/rhel9.0/x86_64/dvd/RPM-GPG-KEY-redhat-release

Repository is enabled.

ii) Name = appstream

Description = App Description

Url = 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.

```
# vim /home/student/ansible/adhoc.yml
---
- name: Creating yum repository
  hosts: all
  tasks:
   - name: Create BaseOS Repository
     ansible.builtin.yum_repository:
      name: "baseos"
      description: "Baseos 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: "App 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 -m command -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
```

```
# ls collections/ansible_collections (verify)
```

```
# ansible-navigator collections (verify)
```

4. installing the roles.

i) Create a directory 'roles' under /home/student/ansible

ii) Create a playbook called requirements.yml under the roles directory and download the given roles under the 'roles' directory using galaxy command under it.

iii) Role name should be balancer and download using this url http://content.example.com/Rhce/balancer.tgz.

iv) Role name phpinfo and download using this url http://content.example.com/Rhce/phpinfo.tgz.

```
# mkdir /home/student/ansible/roles
```

```
# vim /home/student/ansible/roles/requirements.yml
---
- src: http://content.example.com/Rhce/balancer.tgz
  name: balancer
- src: http://content.example.com/Rhce/phpinfo.tgz
  name: phpinfo
```

```
# ansible-galaxy install -r /home/student/ansible/roles/requirements.yml -p
/home/student/ansible/roles
```

5. Create offline role named apache under roles directory.

i) Install httpd package and the service should be start and enable the httpd service.

ii) Host the web page using the template.j2

iii) The template.j2 should contain

My host is HOSTNAME on IPADDRESS

Where HOSTNAME is fully qualified domain name.

iv) Create a playbook named apache_role.yml and run the role in dev group.

```
# ansible-galaxy init /home/student/ansible/roles/apache
```

```
# vim /home/student/ansible/roles/apache/templates/template.j2
My host is {{ ansible_fqdn }} on {{ ansible_default_ipv4.address }}
```

(or)

```
My host is {{ ansible_facts['fqdn'] }} on {{ ansible_facts['default_ipv4']['address'] }}
```

```
# vim /home/student/ansible/roles/apache/tasks/main.yml
```

```
- name: Install httpd package
  ansible.builtin.dnf:
    name:
      - httpd
      - firewalld
    state: present
- name: start service httpd
  ansible.builtin.service:
    name: httpd
    state: started
    enabled: yes
- name: start service firewalld
  ansible.builtin.service:
    name: firewalld
    state: started
    enabled: yes
- name: Add http service in firewall rule
  ansible.posix.firewalld:
    service: http
    state: enabled
    permanent: yes
    immediate: yes
- name: Copy the template.j2 file to web server directory
  ansible.builtin.template:
    src: template.j2
    dest: /var/www/html/index.html
```

```
# vim /home/student/ansible/apache_role.yml
---
- name: apache deploy
  hosts: dev
  roles:
    - apache
```

```
# ansible-navigator run apache_role.yml -m stdout
```

```
# curl http://servera.lab.example.com     #(Verify the output)
```

6. Create a playbook called roles.yml and it should run balancer and phpinfo roles.

i) Run the balancer role on balancers group.

ii) Run the phpinfo role on webservers group.

phpinfo output:

Access the url http://serverd.lab.example.com and you can see the content "Welcome to Advpro".

```
# vim roles.yml
---
- name: Run the phpinfo first
  hosts: webservers
  roles:
    - phpinfo
- name: Run the balancer
  hosts: balancers
  roles:
    - balancer
```

Note: (Do not change the above roles order)

```
# ansible-navigator run  roles.yml -m stdout
# curl http://serverd.lab.example.com
```

7.1 Create a playbook name timesync.yml and use system roles

i) Use ntp server 172.25.254.254 and enable iburst.

ii) Run this playbook on all the managed nodes.

```
# cp /home/student/ansible/collections/ansible_collections/redhat/rhel_system_roles/roles/* /home/student/roles/
```

```
# vim timesync.yml
---
- name: Using the timesync roles
  hosts: all
  vars:
    timesync_ntp_servers:
      - hostname: 172.25.254.254
        iburst: yes
  roles:
    - timesync
```

```
# ansible servera.lab.example.com -m command -a 'cat /etc/chrony.conf'    #(Pre verify the chrony file)
```

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

```
# ansible servera.lab.example.com -m command -a 'cat /etc/chrony.conf'    #(Verify the ouput)
```

7.2 Create a playbook name selinux.yml and use system roles

i) Set selinux mode as enforcing in all manage node

```
# cp /home/student/ansible/collections/ansible_collections/redhat/rhel_system_roles/roles/*
/home/student/roles/
```

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

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

```
# ansible all -m command -a "cat /etc/selinux/config"
```

8. Install packages in multiple group.

i) Install vsftpd and mariadb-server packages in dev and test group.
ii) Install "RPM Development Tools" group package in prod group.
iii) Update all packages in dev group.
iv) Use separate play for each task and playbook name should be packages.yml.

```
# vim packages.yml
---
- name: package installation
  hosts: dev,test
  tasks:
  - name: installing php and mariadb-server
    ansible.builtin.dnf:
      name:
        - vsftpd
        - mariadb-server
      state: present
- name: group package installation
  hosts: prod
  tasks:
  - name: installing group package 'Development tools'
    ansible.builtin.dnf:
      name: '@RPM Development Tools' #(in exam @RPM Development Tools)
      state: present
- name: update packages
  hosts: dev
  tasks:
  - name: updating all
    ansible.builtin.dnf:
      name: '*'
      state: latest
```

```
# ansible-navigator run packages.yml -m stdout
```

```
# ansible dev -m command -a 'yum list installed |grep vsftpd'      #(Verify the output)
```

```
# ansible prod -m command -a 'yum group list'      #(Verify the output)
```

9. Create a playbook webcontent.yml and it should run on dev group.

i) Create a directory /devweb and it should be owned by devops group.

ii) /devweb directory should have context type as "httpd"

iii) Assign the permission for user=rwx,group=rwx,others=rx and group special permission should be applied to /devweb.

iv) Create an index.html file under /devweb directory and the file should have the content "Developement".

v) Link the /devweb directory to /var/www/html/devweb.

```
# vim /home/student/ansible/webcontent.yml
---
- name: create a link
  hosts: dev
  tasks:
  - name: create a directory
    ansible.builtin.file:
      path: /devweb
      state: directory
      group: devops
      mode: '02775'
      setype: httpd_sys_content_t
  - name: create a file
    ansible.builtin.file:
      path: /devweb/index.html
      state: touch
  - name: copy the contents to index.html
    ansible.builtin.copy:
      content: "Development\n"
      dest: /devweb/index.html
  - name: link the directory
    ansible.builtin.file:
      src: /devweb
      dest: /var/www/html/devweb
      state: link
```

```
# ansible-navigator run  webcontent.yml -m stdout
```

```
# curl http://servera.lab.example.com/devweb/        #(Verify the output)
```

10. Collect hardware report using playbook in all nodes.

i) Download hwreport.txt from the url http://content.example.com/Rhce/hwreport.txt and save it under /root.

/root/hwreport.txt should have the content with node informations as key=value.

```
#hwreport
HOSTNAME=
MEMORY=
BIOS=
CPU=
DISK_SIZE_VDA=
DISK_SIZE_VDB=
```

ii) If there is no information it have to show "NONE".

iii) playbook name should be hwreport.yml.

Note: Copy the url "http://content.example.com/Rhce/hwreport.txt" and paste that on new tab and verify the content in it.

```
# ansible all -m command -a 'lsblk'          #(Verify the vdb disk exists)
```

```
# vim /home/student/ansible/hwreport.yml
---
- name: hwreport
```

```
hosts: all
ignore_errors: yes
tasks:
- name: Download the file
  ansible.builtin.get_url:
    url: "http://content.example.com/Rhce/hwreport.txt"
    dest: /root/hwreport.txt
- name: Collect report 1
  ansible.builtin.set_fact:
    HOSTNAME: "{{ ansible_hostname }}"
    MEMORY: "{{ ansible_memtotal_mb  }}"
    BIOS: "{{ ansible_bios_version }}"
    CPU: "{{ ansible_processor }}"
    DISK_SIZE_VDA: "{{ ansible_devices['vda']['size'] }}"
- name: Collect report 2
  ansible.builtin.set_fact:
    DISK_SIZE_VDB: "{{ ansible_devices['vdb']['size'] }}"
- name: Copy the content to the managed node
  ansible.builtin.copy:
    content:  |
      #hwreport
      HOSTNAME={{ HOSTNAME | default('NONE') }}
      MEMORY={{ MEMORY | default('NONE') }}
      BIOS={{ BIOS  | default('NONE') }}
      CPU={{ CPU | default('NONE') }}
      DISK_SIZE_VDA={{ DISK_SIZE_VDA | default('NONE') }}
      DISK_SIZE_VDB={{ DISK_SIZE_VDB | default('NONE') }}
    dest: /root/hwreport.txt
```

```
# ansible-navigator run  hwreport.yml -m stdout
# ansible all -m command -a 'cat /root/hwreport.txt'     #(Verify the output)
```

11. Replace the file /etc/issue on all managed nodes.

i) In dev group /etc/issue should have the content "Developement".

ii) In test group /etc/issue should have the content "Test".

iii) In prod group /etc/issue should have the content "Production".

iv) Playbook name issue.yml and run in all managed nodes.

```
# vim /home/student/ansible/issue.yml
---
- name: play for replace module
  hosts: all
  tasks:
  - name: replace the content in dev group
    ansible.builtin.copy:
      content: Development
      dest: /etc/issue
    when: inventory_hostname in groups['dev']
  - name: replace the content in test group
    ansible.builtin.copy:
      content: Test
      dest: /etc/issue
    when: inventory_hostname in groups['test']
  - name: replace the content in prod group
    ansible.builtin.copy:
      content: Production
      dest: /etc/issue
    when: inventory_hostname in groups['prod']
```

```
# ansible-navigator run issue.yml --diff -m stdout
# ansible all -m command -a 'cat /etc/issue'
```

12. Download the file http://content.example.com/Rhce/myhosts.j2.

i) myhosts.j2 is having the content.

```
127.0.0.1 localhost.localdomain localhost
192.168.0.1 localhost.localdomain localhost
```

ii) The file should collect all node information like ipaddress,fqdn,hostname
and it should be the same as in the /etc/hosts file,
if playbook run in all the managed node it must store in /etc/myhosts.

Finally /etc/myhosts file should contains like.

```
127.0.0.1 localhost.localdomain localhost
192.168.0.1 localhost.localdomain localhost

172.25.250.10 servera.lab.example.com servera
172.25.250.11 serverb.lab.example.com serverb
172.25.250.12 serverc.lab.example.com serverc
172.25.250.13 serverd.lab.example.com serverd
```

iii) playbook name hosts.yml and run in dev group.

```
# wget http://content.example.com/Rhce/myhosts.j2
```

```
# vim /home/student/ansible/myhosts.j2
```

last line:

```
{% for host in groups['all'] %}

{{hostvars[host] ['ansible_facts'] ['default_ipv4'] ['address']}} {{hostvars[host] ['ansible_facts'] ['fqdn']}} {{hostvars[host] ['ansible_facts'] ['hostname']}}

{% endfor %}
```

```
# vim hosts.yml

---

- name: Collect the all node information

  hosts: all

  tasks:

  - name: copy the template to the managed node

    ansible.builtin.template:

      src: myhosts.j2

      dest: /etc/myhosts

    when: inventory_hostname in groups['dev']
```

```
# ansible-navigator run hosts.yml -m stdout
```

```
# ansible dev -m command -a 'cat /etc/myhosts'     #(Verify the output)
```

13. Create a variable file vault.yml and that file should contains the variable and its value.

pw_developer is value lamdev

pw_manager is value lammgr

i) vault.yml file should be encrpted using the password "P@sswOrd".

ii) Store the password in secret.txt file and which is used for encrypt the variable file.

```
# vim secret.txt

P@sswOrd
```

```
# ansible-vault create vault.yml --vault-password-file=secret.txt
```

```
pw_developer: lamdev
pw_manager: lammgr
```

```
# ansible-vault view vault.yml --vault-password-file=secret.txt     #(verify the output)
```

14. Download the variable file http://content.example.com/Rhce/user_list.yml and
Playbook name users.yml and run in all nodes using two variable files user_list.yml and vault.yml

i) * Create a group opsdev
* Create user from users variable who's job is equal to developer and need to be in opsdev group
* Assign a password using SHA512 format and run the playbook on dev and test.
* User password is {{ pw_developer }}

ii) * Create a group opsmgr
* Create user from users variable who's job is equal to manager and need to be in opsmgr group
* Assign a password using SHA512 format and run the playbook on test.
* User password is {{ pw_manager }}

iii)* Use when condition for each play.

```
# wget http://content.example.com/Rhce/user_list.yml
```

```
# vim users.yml
---
- name: Create an users and groups
  hosts: all
  vars_files:
    - user_list.yml
    - vault.yml
  tasks:
  - name: Create group 1
    ansible.builtin.group:
      name: opsdev
      state: present
    when: inventory_hostname in groups['dev'] or inventory_hostname in groups['test']
  - name: Create group 2
    ansible.builtin.group:
      name: opsmgr
      state: present
    when: inventory_hostname in groups['test']
  - name: User create 1
    ansible.builtin.user:
      name: "{{ item.name }}"
      uid: "{{ item.uid }}"
      password: "{{ pw_developer | password_hash('sha512') }}"
      password_expire_max: "{{ item.password_expire_days }}"
      groups: opsdev
      state: present
    loop:
      "{{ users }}"
    when: item.job == "developer" and  (inventory_hostname in groups['dev'] or inventory_hostname in groups['test'])
  - name: user create 2
    ansible.builtin.user:
      name: "{{ item.name }}"
      uid: "{{ item.uid }}"
      password: "{{ pw_manager | password_hash('sha512') }}"
      password_expire_max: "{{ item.password_expire_days }}"
      groups: opsmgr
      state: present
    loop:
      "{{ users }}"
    when: item.job == "manager" and inventory_hostname in groups['test']
```

```
# ansible-navigator run users.yml --vault-password-file=secret.txt -m stdout
```

```
# ansible dev  -m command -a 'tail /etc/group'       #(verify the output)
```

```
# ansible test -m command -a 'tail /etc/group'       #(verify the output)
```

15. Rekey the variable file from http://content.example.com/Rhce/solaris.yml.

  i) Old password: cisco

  ii) New password: redhat

```
# wget http://content/Rhce/solaris.yml
```

```
# ansible-vault rekey solaris.yml
Old password:
New password:
Confirm new password:
```

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 -m command -a "ls /var/spool/cron/"
```

```
# ansible all -m command -a "crontab -lu student"
```

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 "ext3" .

iv) Do not perform any mounting for this LV.

iv) The playbook name lvm.yml and run the playbook in all nodes.

___

#Not for Exam# To create volume group "research"

```
# wget http://content/Rhce/initialscripts.sh
```

```
# chmod +x  initialscripts.sh
```

```
# sh initialscripts.sh
```

___

```
- name: Creating LVM storage
  hosts: all
  ignore_errors: yes
  tasks:
   - name: Check Volume group is present
     ansible.builtin.command: vgdisplay research
     register: vginfo
   - ansible.builtin.debug:
       msg: "vg not found"
     when: vginfo is failed
   - name: Creating LVM in 1500 MB
     ansible.builtin.command: 'lvcreate -L 1500M -n data research'
     when: vginfo is success
     register: lv1
   - ansible.builtin.debug:
       msg: "Insufficient size of vg"
     when: lv1 is failed
   - name: Create LVM in 800 MB
     ansible.builtin.command: 'lvcreate -L 800M -n data research'
     when: lv1 is failed
   - name: Assign filesystem as ext3
     ansible.builtin.command: 'mkfs.ext3 /dev/research/data'
```

```
# ansible-navigator run lvm.yml -m stdout
```

```
# ansible all -m command -a 'lsblk'
```

```
# ansible all -m command -a 'lsblk -fp'
```