RHEL-9 RHCE EXAM MODEL PAPER EX294

Duration: 4Hrs   Marks: 300

Read The Instructions Carefully to understand Exam Envirinment

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'**

Remote user name is Student and Password: student . This user exists in control node and managed nodes

*** 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 student**

Container Registry Credentials:

Registry name: utility.lab.example.com

Username: admin & Password: redhat

# 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
podman login utility.lab.example.com
username: admin
password: redhat
$ vim /home/student/.vimrc
 set ai ts=2 cuc
:wq!
$ mkdir /home/student/ansible
$ cd /home/student/ansible
$ vim /home/student/ansible/inventory
[dev]
servera
[test]
serverb
[prod]
serverc
[balancers]
serverd
[webservers:children]
Prod
:wq
$ 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/collections
```

```
ask_pass=false
[privilege_escalation]
become=true
become_method=sudo
become_user=root
become_ask_pass=false
:wq
$ ansible all -m ping
```

==

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

```
i) Name=baseos
   Description="Baseos 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 = 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/yum_repo.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

:wq!
```

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

$ ansible-navigator run -m stdout  yum_repo.yml

$ 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

$ ansible-galaxy collection list [To verify installed collections]
```

==

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.

ANS:

```
$ 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

:wq

$ ansible-galaxy install  /home/admin/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 index.html.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 httpd.yml and run the role in dev group.

ANS:

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

$ 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: index.html.j2
    dest: /var/www/html/index.html
:wq
$ vim /home/student/ansible/roles/apache/templates/index.html.j2
My host is {{ ansible_fqdn }} on {{ ansible_default_ipv4.address }}
:wq
$ vim /home/student/ansible/httpd.yml
---
- name: apache deploy
  hosts: prod
  roles:
    - apache
$ ansible-navigator run -m stdout  httpd.yml
================================================================================
```

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".

ANS:

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

Note: (Do not change the above roles order)

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

Note: Verify with links which they gave you in question

```
================================================================================
```

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.

ANS:

```
$ sudo  yum install rhel-system-roles -y
$ cp -r /home/student/ansible/roles/rhel-system-roles.timesync.yml /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:
    - rhel-system-roles.timesync.yml
:wq
$ ansible-playbook timesync.yml --syntax-check
$ ansible-navigator run timesync.yml -m stdout
```

==

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

i) Set selinux mode as enforcing in all manage node

ANS:

```
$ sudo yum install rhel-system-roles -y
$ cp -r /home/student/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:
    - selinux
:wq
$ ansible-playbook selinux.yml --syntax-check
$ ansible-navigator run selinux.yml -m stdout
$ ansible all -a "cat /etc/selinux/config"
```

==

8. Install packages in multiple group.

i) Install php and mariadb packages in dev,test and prod group.

ii) Install "RPM Development Tools" group package in dev group.

iii) Update all packages in dev group.

ANS:

vim packages.yml

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

```
$ ansible-playbook packages.yml --syntax-check
$ ansible-navigator run packages.yml -m stdout
```

==

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

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

ANS:

```
$ ansible dev -a "systemctl status httpd"

$ ansible dev -a "systemctl status firewalld" ( if firewall service not available users need to install )

# vim /home/student/ansible/webcontent.yml
```

```
---

- name: create a directory /devweb

  hosts: dev

  tasks:

  - name: create a directory

    ansible.builtin.file:

      path: /devweb

      state: directory

      group: apache

      mode: '02775'

      setype: httpd_sys_content_t

  - name: create a file

    ansible.builtin.file:

      path: /devweb/index.html

      state: touch

      setype: httpd_sys_content_t

  - name: copy the contents to index.html

    ansible.builtin.copy:

      content: "Development"
      dest: /devweb/index.html
  - name: link the directory
    ansible.builtin.file:
      src: /devweb
      dest: /var/www/html/devweb
      state: link
      force: yes
  - name: allow http from from firewall
    ansible.posix.firewalld:
      service: http
      state: enabled
      permanent: yes
      immediate: yes
:wq
$ ansible-playbook webcontent.yml --syntax-check
$ ansible-navigator run  webcontent.yml -m stdout
```

Note: Verify out with the link in question

==

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/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.

ANS:

```
$ 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 from url

    ansible.builtin.get_url:

      url: "http://content.example.com/Rhce/hwreport.txt"

      dest: /root/hwreport.txt

  - name: collect hardware report from all managed nodes

    ansible.builtin.replace:

      regexp: "{{item.src}}"

      replace: "{{item.dest}}"

      dest: /root/hwreport.txt

    loop:

      - src: hostname

        dest: "{{ansible_hostname}}"

      - src: totalmemory

        dest: "{{ansible_memtotal_mb}}"

      - src: bios version

        dest: "{{ansible_bios_version}}"

      - src: vda size

        dest: "{{ansible_devices.vda.size}}"

      - src: vdb size

        dest: "{{ansible_devices.vdb.size}}"

:wq

$ ansible-playbook hwreport.yml --syntax-check

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

==

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.

ANS:

```
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']

:wq
```

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

$ ansible-navigator run issue.yml -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.

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

ANS:

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


$ vim /home/student/ansible/myhosts.j2


{{ansible_defaults_ipv4.address}} {{ansible_fqdn}} {{ansible_hostname}}


$ 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.

```
dev_pass: wakennym

mgr_pass: rocky
```

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.

ANS:

```
$ vim secret.txt

P@sswOrd

$ ansible-vault create vault.yml --vault-password-file=secret.txt

dev_pass: wakennym

mgr_pass: rocky

$ 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 create_users.yml and run in all nodes using two variable files user_list.yml and vault.yml

I)* Create user from users variable who's job is equal to developer and need to be supplementary group of devops

* Assign a password from dev_pass variable using SHA512 format and run the playbook on dev and test.

II) * Create user from users variable who's job is equal to manager and need to be supplementary group of opsmgr

* Assign a password from mgr_pass variable using SHA512 format and run the playbook on test.

iii)* Use when condition for each play.

Use password vault file , which is created else where in exam

ANS:

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

vim create_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: "{{item}}"
      state: present
    loop:
      - devops
      - opsmgr
  - name: create a user as a developer
    ansible.builtin.user:
      name: "{{ item.name }}"
      password: "{{ dev_pass | password_hash('sha512') }}"
      password_expire_max: "{{ item.password_expire_days }}"
      groups: devops
      state: present
    loop:
      "{{ users }}"
    when: item.job == "developer" and  (inventory_hostname in groups['dev'] or inventory_hostname in groups['test'])
  - name: create a user as manager
    ansible.builtin.user:
      name: "{{ item.name }}"
      password: "{{ mgr_pass | 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['prod']
```

```
$ ansible-playbook users.yml --syntax-check
$ ansible-navigator run users.yml --vault-password-file=secret.txt -m stdout

$ ansible dev,test -a 'tail /etc/group'    #(verify the output)
```

==

Q15. Rekey an existing Ansible vault as follows:

* Download the Ansible vault from http://192.168.10.254/ex407/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://192.168.10.254/ex407/secret.yml

$ ansible-vault rekey secret.yml

$ ansible-vault view secret.yml ----- > verify with 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"

ANS:

```
$ vim /home/student/ansible/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 "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 "ext4" .

iv) Do not perform any mounting for this LV.

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

ANS:

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

GOOD LUCK FOR Practice