Red Hat RHELv8 EX294 Exam
Linux Automation by Ansible **update Dec20**

PreExam:

- They will provide 6vm's:

control.realmX.example.com → workstation.lab.example.com

node1.realmX.example.com → servera.lab.example.com

node2.realmX.example.com → serverb.lab.example.com

node3.realmX.example.com → serverc.lab.example.com

node4.realmX.example.com → serverd.lab.example.com

node5.realmX.example.com → serverd.lab.example.com (in classroom practice)

NOTE: To practice on classroom, after full reset the hosts, download rhcev8.sh first and run it, it will prepare hosts for exam mock test.

```
-username: root
 password: redhat
-username: admin
 password: redhat
```

NOTE: don't change **root** or **admin** password.

NOTE: no need to create ssh-keygen for access, its pre-defined (check it first in classroom and exam hosts too)

NOTE: SELinux is in enforcing mode and firewalld is disabled/stop on whole managed hosts.

Q1. Install and configure Ansible on the control-node **control.realmX.example.com** as follows:
-install the required packages
-create a static inventory file called **/home/admin/ansible/inventory** as follows:

node1.realmX.example.com is a member of the **dev** host group
node2.realmX.example.com is a member of the **test** host group
node3.realmX.example.com & node4.realmX.example.com are members of the **prod** host group
node5.realmX.example.com is a member of the **balancers** host group.
prod group members be member of the **webservers** host group too

-create a configuration file called **ansible.cfg** as follows:
-the host inventory file **/home/admin/ansible/inventory** is defined
-the location of roles used in playbooks is defined as **/home/admin/ansible/roles**

NOTE: through physical host, login to **workstation.lab.example.com** with user **root** user

Ans:-

```
# yum install ansible*  vim  platform-python* -y

# su - admin

$ vim ~/.vimrc
autocmd FileType yaml setlocal ai ts=2 sw=2 et cursorcolumn
:wq!
$ mkdir ansible

$ cd ansible

$ mkdir roles

$ vim inventory
servera.lab.example.com
serverb.lab.example.com
serverc.lab.example.com
serverd.lab.example.com

[test]
servera.example.com

[prod]
serverb.example.com
serverc.example.com

[webservers]
serverb.example.com
serverc.example.com

[balancer]
serverd.lab.example.com
!wq
$ vim ansible.cfg

[defaults]
inventory=./inventory
remote_user=admin
ask_pass=false
roles_path=./roles

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

$ ansible all --list-hosts
$ ansible -m ping all
```

Q2. Create and run an Ansible **ad-hoc** command

As a system administrator, you will need to install software on the managed nodes. Create a shell script called **yum-pack.sh** that runs an Ansible ad-hoc command to create yum-repository on each of the managed nodes as follows:

BaseOs repo:

-The name of the repository is **EX407**
-The description is **Ex407 Description**
-The base URL is **http://content.example.com/rhel8.0/x86_64/dvd/BaseOS/**
-GPG signature checking is enabled
-The GPG key URL is **http://content.example.com/rhel8.0/x86_64/dvd/RPM-GPG-KEY-redhat-release**
-The repository is enabled

AppStream repo:

-The name of the repository is **EXX407**
-The description is **Exx407 Description**
-The base URL is **http://content.example.com/rhel8.0/x86_64/dvd/AppStream/**
-GPG signature checking is enabled
-The GPG key URL is **http://content.example.com/rhel8.0/x86_64/dvd/RPM-GPG-KEY-redhat-release**
-The repository is enabled

Ans:-

```
$ pwd
/home/admin/ansible/
$ vim yum-pack.sh
#!/bin/bash
ansible all -m yum_repository -a 'file=BaseOs name=EX407 description=Ex407 baseurl=http://content.example.com/rhel8.0/x86_64/dvd/BaseOS/
gpgcheck=yes gpgkey=http://content.example.com/rhel8.0/x86_64/dvd/RPM-GPG-KEY-redhat-release enabled=yes'
ansible all -m yum_repository -a 'file=AppStream name=EXX407 description=Exx407 baseurl=http://content.example.com/rhel8.0/x86_64/dvd/AppStream/
gpgcheck=yes gpgkey=http://content.example.com/rhel8.0/x86_64/dvd/RPM-GPG- KEY-redhat-release enabled=yes'
!wq
$ chmod +x yum-pack.sh
$ bash yum-pack.sh
```

Q3. Create a playbook called **packages.yml** that:

- installs the **php** and **mariadb** packages on hosts in the **dev**, **test**, 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:-

```
$ pwd
home/admin/ansible/
$ vim packages.yml
---
- name:
  hosts: dev,test,prod
  tasks:
    - name:
      yum:
        name: "{{item}}"
        state: latest
      loop:
        - php
        - mariadb
- name:
  hosts: dev
  tasks:
    - name:
      yum:
        name: "@Development Tools"
        state: latest
- name:
  hosts: dev
  tasks:
    - name:
      yum:
        name: "*"
        state: latest
        exclude: kernel*
...
$ ansible-playbook package.yml --syntax-check
$ ansible-playbook package.yml
```

Q4. Install the RHEL system roles package and create a playbook called **timesync.yml** that:

-runs over all **managed hosts**.

-uses the **timesync role**.

-configures the role to use the time server **classroom.example.com** in exam **192.168.10.254**

-configures the role to set the **iburst** parameter as enabled.

Ans:-

```
$ pwd

home/admin/ansible/

$ sudo yum install rhel-system-roles.noarch -y

$ cd roles/

$ ansible-galaxy list

$ cp -r /usr/share/ansible/roles/rhelsystem-roles.timesync  .

$ vim timesync.yml

---
- name:
  hosts: all
  vars:
    timesync_ntp_provider: chrony
    timesync_ntp_servers:
      - hostname: classroom.example.com
        iburst: yes
    timezone: Asia/Kolkata
  roles:
    - rhel-system-roles.timesync
  tasks:
    - name:
      timezone:
        name: "{{timezone}}"
...
:wq!
# ansible-playbook timesync.yml --syntax-check
# ansible-playbook timesync.yml
```

Q5. 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.

-template file **index.j2** is used to create the file **'/var/www/html/index.html'** with the output:

Welcome to HOSTNAME on IPADDRESS

Where **HOSTNAME** is the **fqdn** of the managed node and **IPADDRESS** is the **IP-Address** of the managed node.
Create a playbook called **httpd.yml** that uses this role and the playbook runs on hosts in the **webservers** host group.

Ans:-

```
$ pwd

/home/admin/ansible/roles/

$ ansible-galaxy init apache

$ vim apache/vars/main.yml
---
$ vars file for apache
pkg: httpd
srv: httpd
fw: http
webpage: /var/www/html/index.html
template: index.j2
:wq!

$ vim apache/tasks/install.yml
---
- name: install {{pkg}}
  yum:
    name: "{{pkg}}"
    state: latest
...
:wq!

$ vim apache/tasks/service.yml
---
- name: enable/start {{srv}}
  service:
    name: "{{srv}}"
    enabled: true
    state: started
...
:wq!

$ vim apache/tasks/firewall.yml
---
- name: open {{fw}} on firewalld
  firewalld:
    service: "{{fw}}"
    state: enabled
    immediate: yes
    permanent: true
...
:wq!

$ vim apache/tasks/webpage.yml
---
- name: create {{webpage}}
  template:
    src: "{{template}}"
    dest: "{{webpage}}"
  notify: restart_httpd
...
!wq

$ vim apache/handlers/main.yml
---
# handlers file for roles/apache
  - name: restart_httpd
    service:
      name: "{{http_srv}}"
      state: restarted
:wq!
$ vim apache/templates/index.j2
Welcome to {{ansible_fqdn}} on {{ansible_default_ipv4.address}}
$ vim apache/tasks/main.yml
---
# tasks file for apache
 -include_tasks: install.yml
 -include_tasks: service.yml
 -include_tasks: firewall.yml
 -include_tasks: webpage.yml
:wq!
```

```
$ cd ..
$ pwd
/home/admin/ansible/
$ vim httpd.yml
---
- name:
  hosts: webservers
  roles:
    - apache
...
:wq!
$ ansible-playbook httpd.yml --syntax-check
$ ansible-playbook httpd.yml
```

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://classroom.example.com/role1.tar.gz	the name of this role should be **balancer**
http://classroom.example.com/role2.tar.gz	the name of this role should be **phphello**

Ans:-

```
$ pwd
/home/admin/ansible/roles
$ vim install.yml
---
  - src: http://content.example.com/exam/rhel8/rhce/role1.tar.gz
    name: balancer
  - src: http://content.example.com/exam/rhel8/rhce/role2.tar.gz
    name: phphello
...
:wq!
$ pwd
/home/admin/ansible
$ ansible-galaxy install -r roles/install.yml -p roles
```

Q7. Create a playbook called balance.yml as follows:

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

-this role configures a service to loadbalance webserver requests between hosts in the webservers host group.

-when implemented, browsing to hosts in the balancers host group (http://node5.example.com) should produce the following output:

Welcome to node3.example.com on 192.168.10.z

-reloading the browser (press F5) 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 **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

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

Hello PHP World from node4.example.com

Ans:-

```
$ pwd
/home/admin/ansible/
$ vim balancer.yml
---
- name:
  hosts: webservers
  roles:
    -./roles/phphello
- name:
  hosts: balancer
  roles:
    -./roles/balancer
:wq!
$ ansible-playbook balancer.yml --syntax-check
$ ansible-playbook balancer.yml
```

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=r+w+execute, group=r+w+execute, other=r+execute s.p=set group-id
-symbolically link **/webdev** to **/var/www/html/webdev**
-create the file **/webdev/index.html** with a single line of text that reads: **Development**
-it should be available on http://servera.lab.example.com/webdev/index.html

Ans:-

```
$ pwd
/home/admin/ansible/
$ vim web.yml
---
- name:
  hosts: dev
  tasks:
    - name:
      service:
        name: httpd
        enabled: true
        state: started
    - name:
      file:
        path: /webdev
        state: directory
        mode: 775
        group: apache
        setype: httpd_sys_content_t
    - name:
      raw: 'chmod 2775 /webdev'
    - name:
      file:
        path: /webdev/index.html
        state: touch
        setype: httpd_sys_content_t
    - name:
      lineinfile:
        path: /webdev/index.html
        line: "Development"
        state: present
    - name: firewall
      firewalld:
        service: http
        state: enabled
        immediate: yes
        permanent: true
    - name:
      file:
        src: /webdev/
        dest: /var/www/html/webdev
        state: link
        force: yes
    - name:
      service:
        name: httpd
        state: restarted
...
:wq!

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

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:-
```
$ pwd
/home/admin/ansible
$ echo "atenorth" >password.txt
$ chmod 600 password.txt
$ ansible-vault create vault.yml --vault-password-file=password.txt
---
 - dev_pass: wakennym
 - mgr_pass: rocky
:wq
$ ansible-vault view vault.yml
password:******
```

Q10. Generate a hosts file:
Download an initial template file **hosts.j2** from **http://classroom.example.com/hosts.j2** 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/myhosts**
172.25.250.9 workstation.lab.example.com workstation
Create a playbook called **gen_hosts.yml** that uses this template to generate the file **/etc/myhosts** 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
172.25.250.10 serevra.lab.example.com servera

Ans:-
```
$ pwd
/home/admin/ansible

$ wget http://classroom.example.com/hosts.j2

$ vim hosts.j2
127.0.0.1        localhost localhost.localdomain localhost4 localhost4.localdomain4
::1              localhost localhost.localdomain localhost6 localhost6.localdomain6
{{ansible_default_ipv4.address}} {{ansible_fqdn}} {{ansible_hostname}}
:wq!

$ vim gen_hosts.yml
---
- name:
  hosts: dev
  tasks:
    - name:
      template:
        src: ./hosts.j2
        dest: /etc/myhosts
...
:wq
$ ansible-playbook gen_hosts.yml --syntax-check
$ ansible-playbook gen_hosts.yml
```

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 key-value pair.

Your playbook should:

- Download the file 'hwreport.empty' from the URL 'http://classroom.example.com/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:-

```
$ pwd
/home/admin/ansible
$ vim hwreport.yml
---
- name:
  hosts: all
  tasks:
    - name:
      get_url:
        url: http://content.example.com/rhexam/rh294/materials/hwreport.empty
        dest: /root/hwreport.txt
    - name:
      replace:
        regexp: "{{item.src}}"
        replace: "{{item.dest}}"
        dest: /root/hwreport.txt
      loop:
        - src: INVENTORY_HOSTNAME
          dest: "{{ansible_hostname}}"
        - src: BIOS_VERSION
          dest: "{{ansible_bios_version}}"
        - src:  TOTAL_MEMORY
          dest: "{{ansible_memtotal_mb}}"
        - src: VDA_SIZE
          dest: "{{ansible_devices.vda.size}}"
        - src: VDB_SIZE
          dest: "{{ansible_devices.vdb.size}}"
...
$ ansible-playbook hwreport.yml --syntax-check
$ ansible-playbook hwreport.yml
```

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 test host group, the line reads: "Test"
-On hosts in the prod host group, the line reads: "Production"

Ans:-

```
$ pwd
/home/admin/ansible
$ vim modify.yml
---
- name:
  hosts: all
  tasks:
    - name:
      copy:
        dest: /etc/issue
        content: "Development"
      when: inventory_hostname in groups['dev']
    - name:
      copy:
        dest: /etc/issue
        content: "Test"
      when: inventory_hostname in groups['test']
    - name:
      copy:
        dest: /etc/issue
        content: "Production"
      when: inventory_hostname in groups['prod']
...
:wq
$ ansible-playbook modify.yml --syntax-check
$ ansible-playbook modify.yml
```

Q13. Rekey an existing Ansible vault as follows:

-Download Ansible vault from **http://classroom.example.com/secret.yml** to **/home/admin/ansible/**
-The current vault password is **curabete**
-The new vault password is **newvare**
-The vault remains in an encrypted state with the new password

Ans:-

```
$ pwd

/home/admin/ansible/

$ wget http://classroom.example.com/secret.yml

$ ansible-vault view secret.yml

vault password: *****

$ ansible-vault rekey secret.yml

vault password: *****

new vault password:  *****

confirm new vault password: ****

$ ansible-vault view 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://classroom.example.com/user_list.yml and save to **/home/admin/ansible/** and 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 test 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:-

```
$ pwd
/home/admin/ansible
$ wget http://classroom.example.com/user_list.yml
$ cat user_list.yml
$ vim create_user.yml
---
- name:
  hosts: all
  vars_files:
    - ./user_list.yml
    - ./vault.yml
  tasks:
    - name:
      group:
        name: "{{item}}"
        state: present
      loop:
        - devops
        - opsmgr
    - name:
      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['test']) and item.job == "developer"
    - name:
      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"
:wq!
$ ansible-playbook create_user.yml --vault-password-file=password.txt --syntax-check
$ ansible-playbook create_user.yml --vault-password-file=password.txt
```

Q15. Create Logical volumes with **lvm.yml** in all nodes according to following requirements.
Create a new Logical volume named as **data**

- -LV should be the member of **research** Volume Group
- -LV size should be **1500M**
- -It should be formatted with **ext4** file system.
- -If Volume Group does not exist then it should print the message **"VG Not found"**
- -If the VG cannot accommodate 1500M size then it should print **"LV cannot be created with following size"**, then the LV should be created with **800M** of size.
- -Do not perform any mounting for this LV.

Ans:-

```
$ pwd
/home/admin/ansible
$ vim lvm.yml
---
- name:
  hosts: all
  ignore_errors: yes
  tasks:
    - name:
      lvol:
        lv: data
        size: "1500"
        vg: research
    - debug:
        msg: "VG Not found"
      when: ansible_lvm.vgs.research is not defined
    - debug:
        msg: "LV Cannot be created with following size"
      when: ansible_lvm.vgs.research.size_g < "1.5"
    - name:
      lvol:
        lv: data
        size: "500"
        vg: research
      when: ansible_lvm.vgs.research.size_g < "1.5"
    - name:
      filesystem:
        fstype: ext4
        dev: /dev/research/data
...
:wq!
$ ansible-playbook lvm.yml --syntax-check
$ ansible-playbook lvm.yml
```