EX-294

Read the instruction carefully.

1. There will be one controller machine and 5 nodes.

2. You will be provided the sudo user and it’s password, and you have to create playbooks via user. [ Login on controller node with sudo user ]

3. Create all your playbooks in /home/admin/ansible [ mkdir /home/admin/ansible ]

4. Create config and inventory file in /home/admin/ansible/

5. Create a roles directory in /home/admin/ansible/ [ mkdir /home/admin/ansible/roles ]

Note: vim command will not be working, so you have to install vim in your controller machine (yum install vim -y) but vi command will be accessible.

First step -> read instructions carefully

Second step -> ssh <given user>@< master/controller node name >

Third key step - > yum install vim -y

(Optional Part just to configure vim in controller node)

In admin home directory to create the .vimrc file

sudo vim /home/admin/.vimrc

set ai

set ts=2

set et

set cursorcolumn

Q1. Install and configure Ansible on the control node control.example.com as follows:

\* Install the required packages

\* Create a static inventory file called /home/admin/ansible/inventory as follows:

-- node1.example.com is a member of the dev host group

-- node2.example.com is a member of the test host group

-- node3.example.com and node4.example.com are members of the prod host group

-- node5.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/admin/ansible/inventory is defined

-- The location of roles used in playbooks is defined as /home/admin/ansible/roles

Solution:

> sudo mkdir /home/admin/ansible

> sudo yum install ansible -y

> vim /home/admin/ansible/inventory # make sure to copy paste the details to the inventory

# instead of typing.

# Contents of inventory file starts from new line after the Line below this line

# Note: don’t write controller fqdn/hostname/nodename anywhere in inventory file

[dev]

node1.example.com

[test]

node2.example.com

[prod]

node[3:4].example.com # range method to define the multiple fqdns in single string range

[balancer]

node5.example.com

[webservers:children] #super group logic given in book

prod

# file ends here

> vim /home/admin/ansible/ansible.cfg

[defaults]

inventory = /etc/ansible/hosts

roles\_path = /home/admin/ansible/roles:/usr/share/ansible/roles/

host\_key\_checking = False

remote\_user = admin

vault\_password\_file = /home/admin/ansible/secret

ask\_pass = False

[privilege\_escalation]

become=True

become\_method=sudo

become\_user=root

become\_ask\_pass=False

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 adhoc.sh that runs an Ansible ad-hoc command to create a yum repository on each of the managed nodes as follows:

NOTE: you need to create 2 repos ( Base and APPStream ) in managed node side

i) BaseOS

a. name: Base OS

b. baseurl: http://classroom.example.com/content/rhel8.0/x86\_64/dvd/BaseOS/

c. description: Base OS Repo

d. gpgcheck: yes

e. enabled: yes

f. gpgkey: http://classroom.example.com/content/rhel8.0/x86\_64/dvd/RPM-GPG-KEY-redhat-release

ii) AppStream

a. name: AppStream

b. baseurl: http://classroom.example.com/content/rhel8.0/x86\_64/dvd/AppStream/

c. description: AppStream Repo

d. gpgcheck: yes

e. enabled: yes

f. gpgkey: http://classroom.example.com/content/rhel8.0/x86\_64/dvd/RPM-GPG-KEY-redhat-release

Solution:

vim adhoc

#!/usr/bin/bash

ansible all -m yum\_repository -a "file=external.repo name=’App Stream’ description='AppStream Repo' baseurl=<http://classroom.example.com/content/rhel8.0/x86_64/dvd/AppStream/> gpgcheck=yes gpgkey=http://classroom.example.com/content/rhel8.0/x86\_64/dvd/RPM-GPG-KEY-redhat-release enabled=yes"-b

ansible all -m rpm\_key -a “key=http://classroom.example.com/content/rhel8.0/x86\_64/dvd/RPM-GPG-KEY-redhat-release state=present” -b

ansible all -m yum\_repository -a “file=external.repo name=’Base OS’ description=’BaseOS Repo’ baseurl=http://classroom.example.com/content/rhel8.0/x86\_64/dvd/BaseOS/ gpgcheck=yes gpgkey=http://classroom.example.com/content/rhel8.0/x86\_64/dvd/RPM-GPG-KEY-redhat-release enabled=yes state=present” -b

ansible all -m rpm\_key -a 'key=http://classroom.example.com/content/rhel8.0/x86\_64/dvd/RPM-GPG-KEY-redhat-release state=present' -b

chmod +x adhoc.sh

./adhoc.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 RPM 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

Solution:

> vim packages.yml

---

- name: Playbook for packages.yml

hosts: all

vars:

pkgs:

- php

- mariadb

tasks:

- name: install the packages

yum:

name: "{{ item }}"

state: present

loop: "{{ pkgs }}"

when: inventory\_hostname in groups['dev'] or inventory\_hostname in groups['test'] or inventory\_hostname in groups['prod']

- name: install the RPM development tool package group

yum:

name: "@RPM Development tools"

state: present

when: inventory\_hostname in groups['dev']

- name: update all packages

yum:

name: '\*'

state: latest

when: inventory\_hostname in groups['dev']

**or**  
  
- hosts: dev

tasks:

- yum:

name:

- 'mariadb'

- 'php'

state: present

- hosts: test

tasks:

- yum:

name: "@RPM Development Tools"

state: persent

- yum:

name: '\*'

state: latest

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

-- Runs on all managed hosts

-- Uses the timesync role

-- Configures the role to use the time server 172.25.254.250

-- Configures the role to set the iburst parameter as enabled

Solution:

sudo yum install rhel-system-roles

vim ntp.yml

---

- hosts: localhost

task:

- yum:

name: “rhel-system-roles”

state: present

- hosts: dev

vars:

timesync\_ntp\_servers:

- hostname: <server name given in question>

iburst: yes

roles:

- rhel-system-roles.timesync

...

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

- 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

Solution:

> ansible-galaxy init --init-path=roles apache

> vim roles/apache/tasks/main.yml

- package:

name: "httpd"

state: present

- service:

name: "firewalld"

state: started

enabled: yes

- template:

src: hosts.j2

dest: /var/www/html/index.html

- firewalld:

port: 80/tcp

immediate: yes

permanent: yes

state: enabled

- service:

name: "httpd"

state: started

enabled: yes

> vim roles/apache/templates/hosts.j2 # common for both way

Welcome to {{ ansible\_facts['fqdn'] }} on {{ ansible\_facts['default\_ipv4']['address'] }}

Or

> vim roles/apache/tasks/main.yml

- name: install all packages

yum:

name: "{{ item }}"

state: present

loop: "{{ pkgs }}"

- name: start and enable the services

service:

name: "{{ item }}"

state: started

enabled: true

loop: "{{ pkgs }}"

- name:

firewalld:

service: "{{ item }}"

permanent: true

immediate: true

state: enabled

loop: "{{ rule }}"

- name: create info file from template

template:

src: index.html.j2

dest: /var/www/html/index.html

> vim roles/apache/vars/main.yml # for vars for above role task

pkgs:

- httpd

- firewalld

rule:

- http

- https

> vim roles/apache/templates/index.html.j2 # common for both way

Welcome to {{ ansible\_facts['fqdn'] }} on {{ ansible\_facts['default\_ipv4']['address'] }}

> vim apacherole.yml # common for both ways

---

- name: playbook for apache role

hosts: prod

roles:

- role: apache

...

Q6. Use Ansible Galaxy with a requirements file called /home/admin/ansible/roles/requirement.yml to download and install roles to

/home/admin/ansible/roles from the following URLs:

-- http://classroom.example.com/content/examfun.tar.gz

The name of this role should be balancer

-- http://classroom.example.com/content/examfun.tar.gz

The name of this role should be phpinfo

Solution:

vim roles/requirements.yml

---

- src: http://classroom.example.com/content/examfun.tar.gz

name: balancer

- src: http://classroom.example.com/content/examfun.tar.gz

name: phpinfo

...

ansible-galaxy install -r roles/requirement.yml -p roles/

# pwd before running the command must be /home/admin/ansible/

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 phpinfo 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.

Solution:

vim balance.yml

---  
  
- hosts: all

task: []

- name: play for balancer group

hosts: balancer

roles:

- haproxy

- name: play for webserver group

hosts: prod

roles:

- phpinfo

...

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

Solution:

vim web.yml

---

- hosts: prod

tasks:

- group:

name: webdev

- file:

state: directory

path: /webdev

mode: "2775"

group: webdev

owner: apache

setype: "httpd\_sys\_content\_t"

- file:

src: /webdev

path: /var/www/html/myweb

state: link

mode: 2775

owner: apache

setype: "httpd\_sys\_content\_t"

- copy:

dest: /webdev/index.html

mode: 0640

content: "Depolyment"

owner: apache

setype: "httpd\_sys\_content\_t"

...

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

\* The name of the vault is vault.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

Solution:

vim password.txt

atenorth

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

dev\_pass: wakennym

mgr\_pass: rocky

chmod 0600 password.txt

Q10. Generate a hosts file:

\* Download an initial template file called hosts.j2 from http://classroom.example.com/content/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/hosts

\* 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/myhosts 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

Solution:

wget <http://classroom.example.com/content/hosts.j2> # pwd should be /home/admin/ansible

vim hosts.j2

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

{% for x in groups['all'] %}

{{ hostvars[x]['ansible\_facts']['default\_ipv4']['address'] }} {{ hostvars[x]['ansible\_facts']['fqdn'] }} {{ hostvars[x]['ansible\_facts']['hostname'] }}

{% endfor %}

vim gen\_hosts.yml

---

- name: playbook for gen\_hosts.yml

hosts: all

tasks:

- name: use template file to create /etc/myhosts file

template:

src: hosts.j2

dest: /etc/myhosts

when: inventory\_hostname in groups['dev']

...

Or

- hosts: all

tasks: []

- hosts: dev

tasks:

- template:

src: "/home/user/anisble/hostdetails.j2"

dest: "/etc/myhosts"

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://classroom.example.com/content/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 NULL.

Solution:

(Note: remove vdb from serverd - for making the scenerio)

wget -O /root/hwreport.txt http://classroom.example.com/content/hwreport.empty

cat /root/hwreport.txt

vim hwreport.j2

-- Inventory host name : {{ ansible\_facts['fqdn'] }}

-- Total memory in MB: {{ ansible\_facts['memtotal\_mb'] }}

-- BIOS version: {{ ansible\_facts['bios\_version'] }}

-- Size of disk device vda: {{ ansible\_facts['devices']['vda']['size'] }}

-- Size of disk device vdb: {{ ansible\_facts['devices']['vdb']['size'] }}

vim hwreport2.j2

-- Inventory host name : {{ ansible\_facts['fqdn'] }}

-- Total memory in MB: {{ ansible\_facts['memtotal\_mb'] }}

-- BIOS version: {{ ansible\_facts['bios\_version'] }}

-- Size of disk device vda: {{ ansible\_facts['devices']['vda']['size'] }}

{% if 'AnsibleUndefinedVariable' in hwreport.msg %}

-- Size of disk device vdb: NULL

{% endif %}

vim hwreport.yml

---

- hosts: all

tasks:

- get\_url:

url: "<put url of hwreport file>"

dest: "/root/hwreport.txt"

- replace:

path: "/root/hwreport.txt"

regexp: "hostname"

replace: "{{ ansible\_fqdn }}"

- replace:

path: "/root/hwreport.txt"

regexp: "bios\_version"

replace: "{{ ansible\_bios\_version }}"

- replace:

path: "/root/hwreport.txt"

regexp: "memory"

replace: "{{ ansible\_memtotal\_mb }}"

- replace:

path: "/root/hwreport.txt"

regexp: "sda\_size"

replace: "{{ ansible\_facts['devices']['sda']['size'] }}"

ignore\_errors: yes

register: x

- replace:

path: "/root/hwreport.txt"

regexp: "sda\_size"

replace: "NONE"

when: x.failed == true

- replace:

path: "/root/hwreport.txt"

regexp: "sdb\_size"

replace: "{{ ansible\_facts['devices']['sdb']['size'] }}"

ignore\_errors: yes

register: y

- replace:

path: "/root/hwreport.txt"

regexp: "sdb\_size"

replace: "NONE"

when: y.failed == true

...

Q12. Modify file content in all hosts. 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

Solution:

vim modify.yml

---

- name: playbook for modify.yml

hosts: all

tasks:

- name: replace the content in dev group

copy:

content: "Development"

dest: /etc/issue

when: inventory\_hostname in groups['dev']

- name: replace the content in test group

copy:

content: "Test"

dest: /etc/issue

when: inventory\_hostname in groups['test']

- name: replace the content in prod group

copy:

content: "Production"

dest: /etc/issue

when: inventory\_hostname in groups['prod']

...

Q13. Rekey an existing Ansible vault as follows:

\* Download the Ansible vault from "http://classroom.example.com/content/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

Solution:

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

ansible-vault rekey --ask-vault-pass secret.yml

Vault password: curabete

New Vault password: newvare

Confirm New Vault password: newvare

Rekey successful

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/content/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 test host groups assigned the password from the dev\_pass variable and is 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 and is 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.

Solution:

wget http://classroom.example.com/content/user\_list.yml

cat user\_list.yml

users:

- name: adam

job: developer

- name: gabriel

job: manager

- name: lucifer

job: developer

vim create\_user.yml

---

- name: playbook for create\_user.yml

hosts: all

vars\_files:

- user\_list.yml

- vault.yml

tasks:

- name: create the groups

group:

name: "{{ item }}"

state: present

loop:

- devops

- opsmgr

- name: create the users with developer job profile

user:

name: "{{ item.name }}"

groups: devops

append: true

password: "{{ dev\_pass | password\_hash('sha512') }}"

state: present

loop: "{{ users }}"

when:

- item.job == 'developer'

- inventory\_hostname in groups['dev'] or inventory\_hostname in groups['test']

- name: create the users with manager job profile

user:

name: "{{ item.name }}"

groups: opsmgr

append: true

password: "{{ mgr\_pass | password\_hash('sha512') }}"

state: present

loop: "{{ users }}"

when:

- item.job == 'manager'

- inventory\_hostname in groups['prod']

...

ansible-playbook create\_user.yml --vault-password-file=password.txt

Q15. Create a playbook storage.yml for creating Logical volumes 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 filesystem.

-- If Volume Group does not exist then it should print the message "VG Not found"

-- If the VG can not acccomodate 1500M size then it should print "LV Can not be created with following size"

-- then the LV should be created with 800M of size.

-- Do not perform any mounting for this LV.

Solution:

[Only to do here for creating the scenerio]

vim lvm.yml

---

- name: playbook for lvm creation

hosts: all

tasks:

- name: create new partion

parted:

device: /dev/vdb

number: 1

state: present

part\_end: 2GB

when: ansible\_hostname == 'servera' or ansible\_hostname == 'serverb' or ansible\_hostname == 'serverc'

- name: create a volume group

lvg:

vg: research

pvs: /dev/vdb1

when: ansible\_hostname == 'servera' or ansible\_hostname == 'serverb'

- name: create new partion

parted:

device: /dev/vdb

number: 1

state: present

part\_end: 1GB

when: ansible\_hostname == 'serverd'

- name: create a volume group

lvg:

vg: research

pvs: /dev/vdb1

when: ansible\_hostname == 'serverd'

...

vim storage.yml

---

- name: playbook for storage.yml

hosts: all

tasks:

- block:

- name: Create a logical volume of 1500M

lvol:

vg: research

lv: data

size: 1500m

register: lv\_info

rescue:

- debug:

var: lv\_info

- debug:

msg: " VG Not found"

when: '"does not exist" in lv\_info.msg'

- debug:

msg: "LV Can not be created with following size"

when: '"insufficient free space" in lv\_info.err'

- name: Create a logical volume of 800M

lvol:

vg: research

lv: data

size: 800m

register: lv\_info\_new

when: '"insufficient free space" in lv\_info.err'

- debug:

var: lv\_info\_new

always:

- name: create filesystem

filesystem:

fstype: ext4

dev: "/dev/research/data"

...

check :-

ansible all -a 'lvdisplay'

ansible all -a 'blkid'