***Pre Task (OK)

Go to important configuration and take note of important details (control node FQDN, the specified user you have to work on this exam on)

ssh to admin node as specified user (e.g. curtis, which is previously created)

Note that all work has to be owned/created by specified user not root

Work has to be created in /home/<*specified-user*>/ansible

Tip #1: Since we are running this as regular user, use become: true in every playbook for privilege escalation except for ad-hoc command, which you have to use sudo …

*example; set to 'true'/'yes' to activate privilege escalation.

```
- name: Ensure the httpd service is running
  service:
    name: httpd
    state: started
  become: true
```

**Do not use --become in command line, write in the playbook instead

Tip #2: Run every playbook after completion to ensure that playbook can run and because changes need to happen to be **awarded marks**

Tip #3: If need to download anything use wget

Tip #4: Use Debug module to capture output for diagnostic purposes

```
- name: Installs a package and prints the result
  hosts: all
  tasks:
    - name: Install the package
      yum:
        name: httpd
        state: installed
      register: install_result
```

```
  - debug:
      var: install_result
```

Task 1: Ansible Installation and Configuration (OK)

-> Install ansible package on the control node (including any dependencies)

sudo yum install ansible

Use ansible version to check

ansible --version

-> Create an inventory file /home/automation/plays/inventory with the following:

1. ansible2.hl.local is a member of the **proxy** host group.
2. ansible3.hl.local is a member of the **webservers** host group.
3. ansible4.hl.local is a member of the **webservers** host group.
4. ansible5.hl.local is a member of the **database** host group.

[proxy]

ansible2.hl.local

[webservers]

ansible3.hl.local

ansible4.hl.local

-> Configure the following:

1. The roles path should include /home/<*specified-user*>/ansible/roles, as well as any other path that may be required for the course of the sample exam.
2. The inventory file path is /home/<*specified-user*>/ansible/inventory.

[defaults]

inventory = /home/<specified-user>/ansible/inventory

roles_path = /home/<specified-user>/ansible/roles

Task 2: Ad-Hoc Commands (OK)

Install yum repository on nodes using ad-hoc commands.

Write a script /home/<*location*> that uses Ansible ad-hoc commands to achieve the following:

-> The name of the repository is ***<exam-repository>***.

-> The description of the repository is “exam repository description”.

-> install specified packages

-> uses base URL

-> GPG signature checking enabled and using specified GPG URL

-> repository is enabled

Run using “sudo ./home/<*location*>”

Practice:

```
[vagrant@ansible-engine1 ansible]$ ansible node1 -m yum_repository -a "name=exam
-repository description='example repository description' includepkgs=pkg baseurl
=url gpgcheck=no enabled=yes" -b
node1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "repo": "exam-repository",
    "state": "present"
}
```

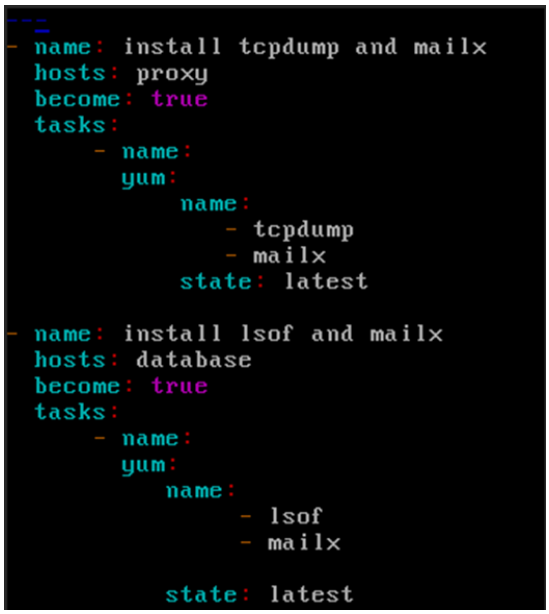
* ./setup.sh -> modify the setup.sh

Task 3: Software Packages (OK)

Create a playbook /home/<*specified-user*>/ansible/packages.yml that runs on all inventory hosts and does the following:

1. Installs **tcpdump** and **mailx** packages on hosts in the **proxy** host groups.
2. Installs **lsof** and **mailx** packages on hosts in the **database** host groups.

-> Make sure lsof and mailx are updated to the latest version



```
---
- name: install tcpdump and mailx
  hosts: proxy
  become: true
  tasks:
      - name:
        yum:
            name:
               - tcpdump
               - mailx
            state: latest

- name: install lsof and mailx
  hosts: database
  become: true
  tasks:
      - name:
        yum:
           name:
                - lsof
                - mailx

           state: latest
```

Task 4: RHEL System Role (OK)

Create a playbook /home/<*specified-user*>/ansible/rhelrole.yml that runs on hosts in the **webservers** host group and does the following:

1. Use timesync role
2. Configure NTP as provider
3. Uses specific time server
4. Uses iburst parameter

Practice:

sudo yum install rhel-system-roles

```
---
- name:
  hosts: webservers
  become: true
  vars:
     timesync_ntp_provider: ntp
     timesync_ntp_servers:
      - hostname: something.example.com
        iburst: yes

  roles:
    - rhel-system-roles.timesync
```

Task 5: Install roles

Install roles from a specific .tar file on the system

Practice:

wget to download

- wget https://galaxy.ansible.com/download/ansible-posix-1.4.0.tar.gz

use ansible-galaxy to install the role

- ansible-galaxy collection install ansible-posix-1.4.0.tar.gz
- ansible-galaxy collection install ansible-posix-1.4.0.tar.gz -p ~/ansible/roles

Task 6: Create and Work with Roles

Create a role called **apache** and store it in /home/<*specified-user*>/ansible/roles. The role should satisfy the following requirements:

1. The **httpd**, **mod_ssl** and **php** packages are installed. Apache service is running and enabled on boot.
2. Firewall is configured to allow all incoming traffic on HTTP port TCP 80 and HTTPS port TCP 443. Firewall is enabled and started
3. Apache service should be restarted every time the file /var/www/html/index.html is modified.
4. A Jinja2 template file index.html.j2 is used to create the file /var/www/html/index.html with the following content:

The address of the server is: <IPV4ADDRESS>

*IPV4ADDRESS is the IP address of the managed node and you have to use something like ansible magic variable

E.g. {{ hostvars[inventory_hostname]['ansible_default_ipv4']['address'] }}

Create a playbook /home/<*specified-user*>/ansible/apache.yml that uses the role and runs on hosts in the **webservers** host group.

Practice:

Cd ~/roles

Ansible-galaxy init apache

Vim ~/ansible/roles/apache/tasks/main.yml

```
# tasks file for apache
- name:
  yum:
     name:
          - httpd
          - mod_ssl
          - php
     state: latest

- name:
  service:
       name: httpd
       state: started
       enabled: yes

- name:
  service:
       name: firewalld
       state: started
       enabled: yes
```

Sameplaybook as previous screenshot

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

- name: permit traffic in default zone for https service
  ansible.posix.firewalld:
    service: https
    immediate: true
    permanent: true
    state: enabled


- name: template
  template:
        src: index.html.j2
        dest: /var/www/html/index.html
        mode: '0644'
  notify: restart httpd
```

Vim ~/ansible/roles/apache/templates/index.html

```
the address of the server is {{ansible_facts['all_ipv4_addresses']}}
```

Vim ~/ansible/roles/apache/handlers/main.yml

```
---
# handlers file for apache
- name: restart httpd
  service:
          name: httpd
          state: restarted
```

```
[vagrant@ansible-engine1 ansible]$ curl node2
the address of the server is ['192.168.50.22', '10.0.2.15']


[vagrant@ansible-engine1 ansible]$ curl node3
the address of the server is ['192.168.50.23', '10.0.2.15']
```

Task 7: File Content (OK with suggestions)

Create a playbook /home/<*specified-user*>/ansible/motd.yml that runs on all inventory hosts and does the following:

1. The playbook should replace any existing content of /etc/motd with text. Text depends on the host group.
2. On hosts in the **proxy** host group the line should be “Welcome to HAProxy server”.
3. On hosts in the **webservers** host group the line should be “Welcome to Apache server”.
4. On hosts in the **database** host group the line should be “Welcome to MySQL server”.

Vim ~/ansible/motd.yml

```
- name:
  hosts: all
  remote_user: root
  become: true
  tasks:
          - name:
            file:
               path: /etc/motd
               state: touch
          - name:
            blockinfile:
               path: /etc/motd
               block: "{{block}}"
```

```
[vagrant@ansible-engine1 ansible]$ ssh node1
# BEGIN ANSIBLE MANAGED BLOCK
"Welcome to HAPro  xy server"
# END ANSIBLE MANAGED BLOCK
Last login: Wed Jul 13 06:43:19 2022 from 192.168.50.11
[vagrant@ansible-node-1 ~]$
```

*may want to put **when: inventory_hostname in groups['proxy']**

```
---
- name: file content
  hosts: all
  become: true
  tasks:
    - name: copy to file
      copy:
        content: "On hosts in the proxy host group the line should be Welcome tO
HAProy server."
        dest: /etc/motd
      when: inventory_hostname in groups['proxy']

    - name: copy to file
      copy:
        content: "On hosts in the proxy host group the line should be Welcome tO
mysql server."
        dest: /etc/motd
      when: inventory_hostname in groups['database']
```

Task 8: Create logical volume (suggest changes)

Create a logical volume using playbook. The role should satisfy the following requirements:.

1. An LVM called exam_lvm is created of size 512MB in the volume group vg_database.
2. ext4 filesystem is used as default and volume is not mounted.
3. If insufficient space, should stop creating volume and print “Not enough space”

4. If volume group does not exist, should stop creating volume and print “Does not exist”

Try out Guided Exercise: Managing Storage

```
---
- name: creating LVM
  hosts: node4
  become: true
  tasks:
      - name: install lvm
        yum:
            name: "lvm*"
            state: present

      - name: Create a new primary partition for LVM
        parted:
          device: /dev/sdb
          number: 1
          flags: [ lvm ]
          state: present
          part_start: 5MiB
          part_end: 700MiB



      - name: make volume group
        lvg:
              vg: vg_database
              pvs: /dev/sdb1


      - name: make lvm
        lvol:
            vg: vg_database
            lv: exam_lvm
            size: 512M
        register: result
```

```
      - name: unsure what the output will be if volume group does not exists
        debug:
            msg: "Does not exist"
        when: '"Volume group does not exists" in result'

      - name: unsure what the output will be if insufficient space
        debug:
            msg: "Not enough space"
        when: '"Insufficient space"  in result'


      - name: Create a ext4 filesystem
        filesystem:
          fstype: ext4
          dev: /dev/vg_database/exam_lvm
```

*this task probably involve block and handlers; but a way to work around it will be to use ansible facts to check whether the vg exists and print the debug messages accordingly (i.e. manually check using ansible facts on each host first, then run the automation task depending on what is in the particular host)

Task 9: Create Hardware report (OK with suggestions)

Create a hardware report on individual nodes using playbook.

Download template which is provided, consist of key=value pairs (e.g. version=X.X)

If value does not exist, then it should be reflected as key=NONE

Need to get a value out of the nested json file -> try out Guided Exercise: Managing Facts in the course material to see how to extract all the hardware info

Practice:

Vim Template.j2

```
the Chasis number is_{% if "NA" in ansible_facts['chassis_serial'] %}
 key=NONE
{% else %}
 {{ansible_facts['chassis_serial']}}
{% endif %}

architecture: {{hostvars['node1']['ansible_facts']['architecture']}}

interfaces: {{ansible_facts['interfaces']}}

total memory:{{ansible_facts['memtotal_mb']}}m


~
```

```
[vagrant@ansible-engine1 ansible]$ ansible node1 -m command -a "cat /root/test1.
j2" -b
node1 | CHANGED | rc=0 >>
the Chasis number is  key=NONE

architecture: x86_64

interfaces: ['lo', 'eth0', 'eth1']

total memory:969m
```

- Used some random facts

*Good to use the setup module to inspect the specs of the machines first to double check if the printed output is correct

Task 10: Populate /etc/host using templates (OK)

Use a template to update /etc/host on individual nodes using playbook.

Download template which is provided, which has certain text on it and you have to append your code to it according to the format

<default text>

{{ ip address }} {{ FQDN }} {{ DNS Name }}

*Will have more than one set of ipaddr/fqdn/dns for this as there may be more than one host in each group

https://www.lisenet.com/2019/ansible-use-hostvars-to-generate-hosts-file/

E.g. for webservers group the expected answer will look like

<default text>

{{ ip address 1 }} {{ FQDN 1 }} {{ DNS Name 1 }}

{{ ip address 2 }} {{ FQDN 2 }} {{ DNS Name 2 }}

Vim ~/hosts.yml

```
---
- name:
  hosts: all
  remote_user: root
  tasks:
      - name:
        template:
              src: host_vars_testing.j2
              dest: /home/vagrant/testing.j2
```

Vim ./host_vars_testing.j2

```
_ {{hostvars['node1']['ansible_facts']['all_ipv4_addresses'][0]}} {{hostvars['no
de1']['ansible_facts']['fqdn'] }} {{hostvars['node1']['ansible_facts']['dns']['n
ameservers']}} \n
  {{hostvars['node2']['ansible_facts']['all_ipv4_addresses'][0]}} {{hostvars['no
de2']['ansible_facts']['fqdn'] }} {{hostvars['node2']['ansible_facts']['dns']['n
ameservers']}} \n
  {{hostvars['node3']['ansible_facts']['all_ipv4_addresses'][0]}} {{hostvars['no
de3']['ansible_facts']['fqdn'] }} {{hostvars['node3']['ansible_facts']['dns']['n
ameservers']}} \n
  {{hostvars['node4']['ansible_facts']['all_ipv4_addresses'][0]}} {{hostvars['no
de4']['ansible_facts']['fqdn'] }} {{hostvars['node4']['ansible_facts']['dns']['n
ameservers']}} \n
```

output

```
192.168.50.21 ansible-node-1 ['10.0.2.3'] \n
10.0.2.15 ansible-node-2 ['10.0.2.3'] \n
192.168.50.23 ansible-node-3 ['10.0.2.3'] \n
192.168.50.24 ansible-node-4 ['10.0.2.3'] \n
```

Task 11: Create and Work with Roles (Existing roles)

Use roles from earlier downloaded tar, create a playbook to activate the roles on host groups.

Role called **balancers** should satisfy the following requirements:

1. Upon activation, when browsing to <given domain name> an output would be expected. Refreshing it will produce a new output as the workload is load balanced to the other host

Role called **phpinfo** should satisfy the following requirements:

1. Edit some template file to include {{ FQDN }}
2. Upon activation, when browsing to <given domain name>/hello.php the message: welcome to {{FQDN}}, alongside side other relevant php configuration info (like version etc) will be displayed

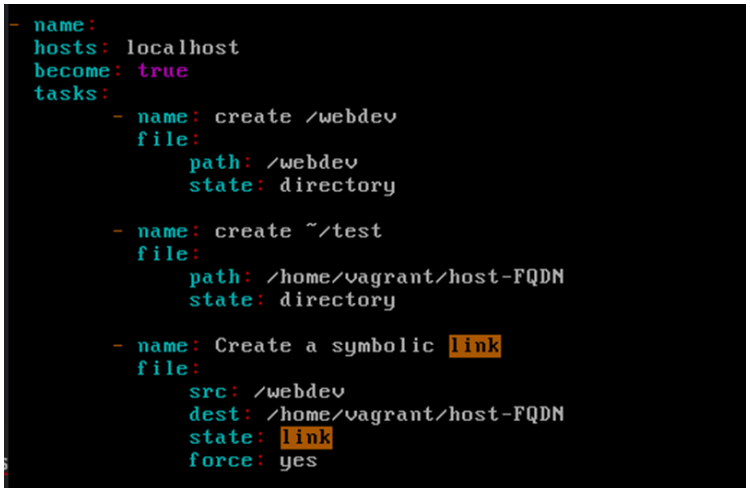
Will

Task 12: Configure Web dev directory (OK with suggestion)

Playbook will operate on /webdev directory and should satisfy the following requirements:

1. Owned by specific user, set regular permissions and special permissions (based on group ID)
2. Create symbolic link to location
3. Update /webdev/index.html with word “Development”
4. Browsing to host-FQDN/webdev will show the word “Development”

Practice:



```
- name:
  hosts: localhost
  become: true
  tasks:
        - name: create /webdev
          file:
              path: /webdev
              state: directory

        - name: create ~/test
          file:
              path: /home/vagrant/host-FQDN
              state: directory

        - name: Create a symbolic link
          file:
              src: /webdev
              dest: /home/vagrant/host-FQDN
              state: link
              force: yes
```

*Might ask you to do this for a certain group so remember to specify the when: condition where needed or specify the group at the top if only need to perform playbook on a particular group

Task 13: Ansible Vault (OK)

Create Ansible vault file /home/<*specified-user*>/ansible/secret.yml. Encryption/decryption password is **whenyouwishuponastar**.

Add the following variables to the vault:

1. **user_password** with value of **devops**
2. **database_password** with value of **devops**

Store Ansible vault password in the file /home/<*specified-user*>/ansible/vault_key.txt.

Practice:

Ansible-vault create secret.yml

Enter the password: whenyouwishuponastar

- User_password: devops
- Database_password: devops

vim /ansible/vault_key.txt

- whenyouwishuponastar

Task 14: Users and Groups

You have been provided with the list of users below.

https://www.lisenet.com/2019/ansible-generate-crypted-passwords-for-the-user-module/

Reference

```
---
users:
  - username: alice
    uid: 1201
  - username: vincent
    uid: 1202
  - username: sandy
    uid: 2201
  - username: patrick
    uid: 2202
--
```

Create a playbook /home/<*specified-user*>/ansible/users.yml to achieve the following:

1. Users whose user ID starts with 1 should be created on servers in the **webservers** host group. User password should be used from the **user_password** variable.
2. Users whose user ID starts with 2 should be created on servers in the **database** host group. User password should be used from the **user_password** variable.
3. All users whose user ID starts with 1 should be members of a supplementary group **wheel**.
4. All users whose user ID starts with 1 should be members of a supplementary group **axel**.
5. Account passwords should use the SHA512 hash format.

*Can try this password: {{ upassword | password_hash('sha512') }}

Should use the earlier ansible vault created in task 13.

Practice:

Vim ~/ansible/users.yml

```yaml
- name:
  hosts: webservers
  become: true
  vars:
      dict1:
         - username: alice
           uid: 1201
         - username: vincent
           uid: 1202
  vars_files: secret.yml
  tasks:

      - name: make group
        group:
            name: axel
            state: present

      - name: make group2
        group:
              name: wheel
              state: present
```

```yaml
      - name:
        user:
              name: "{{item.username}}"
              uid: "{{item.uid}}"
              password: "{{user_password| password_hash('sha512') }}"
              group: wheel
        loop: "{{dict1}}"
```

```yaml
- name: xxxxxxxxxxxxxxxxxxxxxxxx
  hosts: database
  become: true
  vars_files:
         - secret.yml
  tasks:
      - name: uuuuuuuuuuuuuuu
        user:
            name: "sandy"
            uid: "2201"
            password: "{{user_password|password_hash('sha512')}}"


      - name:
        user:
            name: "patrick"
            uid:  "2202"
            password:  "{{user_password|password_hash('sha512')}}"
```

Ansible-playbook users.yml –vault-pass-file vault_key.txt

Task 15: Rekey Ansible Vault (OK)

Download Ansible vault file from system. Change existing password from “abcd” to “efgh”.

Make sure the password is still encrypted after.

Practice:

Ansible-vault rekey vault_file

Efgh

*use wget where needed