Red Hat Enterprise Linux Automation with Ansible

Lab: Implementing Task Control

Install the Apache web server and secure it using `mod_ssl`. You use conditions, handlers, and task failure handling in your playbook to deploy the environment.

Outcomes

- Define conditionals in Ansible Playbooks

- Set up loops that iterate over elements
- Define handlers in playbooks
- Handle task errors.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command prepares your environment and ensures that all required resources are available.

```
[student@workstation ~]$ lab start control-review
```

Instructions

1. On the `workstation` machine, change to the `/home/student/control-review` directory.

   ```
   [student@workstation ~]$ cd ~/control-review
   [student@workstation control-review]$
   ```

   Hide Solution

2. The project directory contains a partially completed play in the `playbook.yml` playbook. Under the `#Fail fast message` comment, add a task that uses the `ansible.builtin.fail` module. Provide an appropriate name for the task.

   This task should only be executed when the remote system does not meet the following minimum requirements:

   - Has at least the amount of RAM specified by the `min_ram_mb` variable. The `min_ram_mb` variable is defined in the `vars.yml` file and has a value of `256`.
   - Is running Red Hat Enterprise Linux.

   The completed task should consist of the following content:

   ```
   tasks:
     #Fail fast message
     - name: Show failed system requirements message
       ansible.builtin.fail:
         msg: "The {{ inventory_hostname }} did not meet minimum reqs."
       when: >
         ansible_facts['memtotal_mb'] < min_ram_mb or
         ansible_facts['distribution'] != "RedHat"
   ```

   Hide Solution

3. Under the `#Install all packages` comment, add a task named `Ensure required packages are present` to install the latest version of any missing packages. Required packages are specified by the `packages` variable, which is defined in the `vars.yml` file.

   The completed task should consist of the following content:

   ```
     #Install all packages
     - name: Ensure required packages are present
       ansible.builtin.dnf:
         name: "{{ packages }}"
         state: latest
   ```

   Hide Solution

4. Under the `#Enable and start services` comment, add a task to start services. All services specified by the `services` variable, which is defined in the `vars.yml` file, should be started and enabled. Provide an appropriate name for the task.

   The completed task should consist of the following content:

```
#Enable and start services
- name: Ensure services are started and enabled
  ansible.builtin.service:
    name: "{{ item }}"
    state: started
    enabled: true
  loop: "{{ services }}"
```

Hide Solution

5. Under the `#Block of config tasks` comment, add a task block to the play. This block contains two tasks:

   - A task to ensure that the directory specified by the `ssl_cert_dir` variable exists on the remote host. This directory stores the web server's certificates.
   - A task to copy all files specified by the `web_config_files` variable to the remote host. Examine the structure of the `web_config_files` variable in the `vars.yml` file. Configure the task to copy each file to the correct destination on the remote host.

     This task should trigger the `Restart web service` handler if any of these files are changed on the remote server.

   Additionally, a debug task is executed if either of the two tasks above fail. In this case, the task prints the following message: `One or more of the configuration changes failed, but the web service is still active.`

   Provide an appropriate name for all tasks.

   The completed task block should consist of the following content:

```
#Block of config tasks
- name: Setting up the SSL cert directory and config files
  block:
    - name: Create SSL cert directory
      ansible.builtin.file:
        path: "{{ ssl_cert_dir }}"
        state: directory

    - name: Copy config files
      ansible.builtin.copy:
        src: "{{ item['src'] }}"
        dest: "{{ item['dest'] }}"
      loop: "{{ web_config_files }}"
      notify: Restart web service

  rescue:
    - name: Configuration error message
      ansible.builtin.debug:
        msg: >
          One or more of the configuration
          changes failed, but the web service
          is still active.
```

Hide Solution

6. The play configures the remote host to listen for standard HTTPS requests. Under the `#Configure the firewall` comment, add a task to configure `firewalld`.

   Ensure that the task configures the remote host to accept standard HTTP and HTTPS connections. The configuration changes must be effective immediately and persist after a reboot. Provide an appropriate name for the task.

   The completed task should consist of the following content:

```
  #Configure the firewall
  - name: Ensure web server ports are open
    ansible.posix.firewalld:
      service: "{{ item }}"
      immediate: true
      permanent: true
      state: enabled
    loop:
      - http
      - https
```

Hide Solution

7. Define the `Restart web service` handler.

   When triggered, this task should restart the web service defined by the `web_service` variable, defined in the `vars.yml` file.

   Add a `handlers` section to the end of the play:

```
handlers:
  - name: Restart web service
    ansible.builtin.service:
      name: "{{ web_service }}"
      state: restarted
```

   The completed playbook should consist of the following content:

```
---
- name: Playbook Control Lab
  hosts: webservers
  vars_files: vars.yml
  tasks:
    #Fail fast message
    - name: Show failed system requirements message
      ansible.builtin.fail:
        msg: "The {{ inventory_hostname }} did not meet minimum reqs."
      when: >
        ansible_facts['memtotal_mb'] < min_ram_mb or
        ansible_facts['distribution'] != "RedHat"

    #Install all packages
    - name: Ensure required packages are present
      ansible.builtin.dnf:
        name: "{{ packages }}"
        state: latest

    #Enable and start services
    - name: Ensure services are started and enabled
      ansible.builtin.service:
        name: "{{ item }}"
        state: started
        enabled: true
      loop: "{{ services }}"

    #Block of config tasks
    - name: Setting up the SSL cert directory and config files
      block:
        - name: Create SSL cert directory
          ansible.builtin.file:
            path: "{{ ssl_cert_dir }}"
            state: directory

        - name: Copy config files
          ansible.builtin.copy:
            src: "{{ item['src'] }}"
            dest: "{{ item['dest'] }}"
          loop: "{{ web_config_files }}"
          notify: Restart web service

      rescue:
        - name: Configuration error message
          ansible.builtin.debug:
            msg: >
              One or more of the configuration
              changes failed, but the web service
              is still active.

    #Configure the firewall
    - name: Ensure web server ports are open
      ansible.posix.firewalld:
        service: "{{ item }}"
        immediate: true
        permanent: true
        state: enabled
      loop:
        - http
        - https

  #Add handlers
  handlers:
    - name: Restart web service
      ansible.builtin.service:
        name: "{{ web_service }}"
        state: restarted
```

Hide Solution

8. From the `~/control-review` directory, run the `playbook.yml` playbook. The playbook should execute without errors, and trigger the execution of the handler task.

```
[student@workstation control-review]$ ansible-navigator run \
> -m stdout playbook.yml

PLAY [Playbook Control Lab] ****************************************************

TASK [Gathering Facts] *********************************************************
ok: [serverb.lab.example.com]

TASK [Show failed system requirements message] *********************************
skipping: [serverb.lab.example.com]

TASK [Ensure required packages are present] ************************************
changed: [serverb.lab.example.com]

TASK [Ensure services are started and enabled] *********************************
changed: [serverb.lab.example.com] => (item=httpd)
ok: [serverb.lab.example.com] => (item=firewalld)

TASK [Create SSL cert directory] ***********************************************
changed: [serverb.lab.example.com]

TASK [Copy config files] *******************************************************
changed: [serverb.lab.example.com] => (item={'src': 'server.key', 'dest': '/etc/httpd/conf.d/ssl'})
changed: [serverb.lab.example.com] => (item={'src': 'server.crt', 'dest': '/etc/httpd/conf.d/ssl'})
changed: [serverb.lab.example.com] => (item={'src': 'ssl.conf', 'dest': '/etc/httpd/conf.d'})
changed: [serverb.lab.example.com] => (item={'src': 'index.html', 'dest': '/var/www/html'})

TASK [Ensure web server ports are open] ****************************************
changed: [serverb.lab.example.com] => (item=http)
changed: [serverb.lab.example.com] => (item=https)

RUNNING HANDLER [Restart web service] ******************************************
changed: [serverb.lab.example.com]

PLAY RECAP *********************************************************************
serverb.lab.example.com    : ok=7    changed=6    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

Hide Solution

9. Verify that the web server now responds to HTTPS requests, using the self-signed custom certificate to encrypt the connection. The web server should return `Configured for both HTTP and HTTPS`.

```
[student@workstation control-review]$ curl -k -vvv https://serverb.lab.example.com
*   Trying 172.25.250.11:443...
* Connected to serverb.lab.example.com (172.25.250.11) port 443 (#0)
...output omitted...
< HTTP/1.1 200 OK
< Date: Tue, 05 Jul 2022 19:36:48 GMT
< Server: Apache/2.4.51 (Red Hat Enterprise Linux) OpenSSL/3.0.1
< Last-Modified: Tue, 05 Jul 2022 19:35:30 GMT
< ETag: "24-5e313f48fbb2c"
< Accept-Ranges: bytes
< Content-Length: 36
< Content-Type: text/html; charset=UTF-8
<
Configured for both HTTP and HTTPS.
* Connection #0 to host serverb.lab.example.com left intact
```

Hide Solution

Evaluation

As the `student` user on the `workstation` machine, use the `lab` command to grade your work. Correct any reported failures and rerun the command until successful.

```
[student@workstation ~]$ lab grade control-review
```

Finish

On the `workstation` machine, change to the `student` user home directory and use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish control-review
```

(/rol/app/courses/rh294-9.0/pages/ch04s06) Previous

(/rol/app/cou Next

Red Hat
Learning Community

Discuss Red Hat Enterprise Linux Automation with Ansible (https://learn.redhat.com/t5/RH294-Red-Hat-Linux-Automation/gh-p/RH294RedHatLinuxAutomationwithAnsible)

Red Hat Linux Automation with Ansible (RH294)
Haley_Ruccio Jul 24, 2023
Learn how to automate Linux system administration tasks with Red Hat Ansible...
3 1 980

curl finds no route to host?
TPeters Jul 23, 2023
I am doing the self-paced course RH294 . With two exercises (Ch.3 final Lab and Ch....
1 6 1702

Welcome to the Red Hat Linux Automation with Ansible (RH294)...
cschunke Jul 18, 2023
We are excited to launch a space dedicated to the Red Hat Training course Red Hat...
8 1 351

Revision: rh294-9.0-4e8f1e1