Lab - Use Ansible to Automate Installing a Web Server

***** MODIFIED FOR NETLAB+ *****

Objectives

Part 1: Configure Ansible

Part 2 Verify Communications with the Local Webserver

Part 3: Create Ansible Playbooks to Automate Webserver Installation

Part 4: Add Options to Your Ansible Playbook for Apache Web Servers

Background / Scenario

In this lab, you will first configure Ansible so that it can communicate with a webserver application. You will then create a playbook that will automate the process of installing Apache on the webserver. You will also create a customized playbook that installs Apache with specific instructions.

Required Resources

- DEVASC Virtual Machine

Instructions

Part 1: Configure Ansible

The DEVASC VM comes preinstalled with a number of dummy IPv4 addresses you can use for various scenarios and simulations. In this Part, you will configure Ansible to use one of the dummy IPv4 address for a local webserver.

Step 1: Open a terminal in the DEVASC-LABVM.

Step 2: Enable the SSH server.

The SSH server is disabled in the DEVASC-LABVM, along with other services that are typically not required. Start it with the following command.

```
devasc@labvm:~$ sudo systemctl start ssh
devasc@labvm:~$
```

Note: The SSH server and **sshpass** utility have already been installed in your VM. For your reference, these are installed using the following commands:

Install SSH

```
devasc@labvm:~$ sudo apt-get install openssh-server
```

Install sshpass

```
devasc@labvm:~$ sudo apt-get install sshpass
```

Step 3: Open the ansible directory in VS Code.

a. Open **VS Code**.

b. Click **File > Open Folder...** and navigate to the **/labs/devnet-src/ansible** folder.

c. Click **OK**.

d. The two subdirectories for the Ansible labs are now loaded in the VS Code **EXPLORER** pane for your convenience. In this lab, you will work with the **ansible-apache** directory.

Step 4: Edit the Ansible inventory file

a. Open the **hosts** file in the **ansible-apache** directory.

b. Add the following lines to the **hosts** file and save.

```
[webservers]
192.0.2.3 ansible_ssh_user=devasc ansible_ssh_pass=Cisco123!
```

c. The credentials **devasc** and **Cisco123!** are admin credentials for the DEVASC VM. The IPv4 address you will use for this lab is 192.0.2.3. This is a static IPv4 address on the VM under the dummy0 interface, as shown in the output for the **ip addr** command.

```
devasc@labvm:~/labs/devnet-src/ansible$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen
1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
default qlen 1000
    link/ether 08:00:27:97:ae:11 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 45882sec preferred_lft 45882sec
    inet6 fe80::a00:27ff:fe97:ae11/64 scope link
       valid_lft forever preferred_lft forever
3: dummy0: <BROADCAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/ether a6:44:a7:e8:6a:9e brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.1/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.2/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.3/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.4/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.5/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet6 fe80::a444:a7ff:fee8:6a9e/64 scope link
       valid_lft forever preferred_lft forever
devasc@labvm:~/labs/devnet-src/ansible$
```

Step 5: Edit the ansible.cfg file.

a. In the **ansible-apache** subdirectory, Open the **ansible.cfg**.

b. You can remove the comment. Add the following lines to the file and save it. The **ansible.cfg** file tells Ansible where to find the inventory file and sets certain default parameters.

```
[defaults]
# Use local hosts file in this folder
inventory=./hosts
# Don't worry about RSA Fingerprints
host_key_checking = False
# Do not create retry files
retry_files_enabled = False
```

Part 2: Verify Communications with the Local Webserver

In this Part, you will verify that Ansible can send commands to the local webserver.

Step 1: Use the ping module to verify that Ansible can ping the webserver.

Use the Ansible **ping** module to verify communications with the devices listed within the **webservers** group of your **hosts** inventory file.

```
devasc@labvm:~/labs/ansible/ansible-apache$ ansible webservers -m ping
192.0.2.3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
devasc@labvm:~/labs/ansible/ansible-apache$
```

If multiple devices were listed under the **webservers** group in your **hosts** inventory file, the output would indicate similar information for each device.

Step 2: Use the command module to verify Ansible can communicate with the webserver.

Use the Ansible **command** module to verify communications with the devices listed within the **webservers** group of your **hosts** inventory file. In this example you send the argument -**a "/bin/echo hello world"** to ask the local webserver to respond with “hello world”.

```
devasc@labvm:~/labs/ansible/ansible-apache$ ansible webservers -m command -a
"/bin/echo hello world"
192.0.2.3 | CHANGED | rc=0 >>
hello world
devasc@labvm:~/labs/ansible/ansible-apache$
```

Part 3: Create Ansible Playbooks to Automate Webserver Installation

In this Part, you will create two Ansible playbooks. The first playbook will automate the echo test you did in the previous Part. Imagine you are bringing a hundred webservers online. The [webserver] group in the **hosts** file would list all the necessary information for each webserver. You can then use a simple playbook to verify communications with all of them with one command. The second playbook you will create will automate the installation of Apache webserver software.

Step 1: Create your Ansible playbook to test your webserver group.

In this step you will create an Ansible playbook to perform the same **echo** command.

a. In VS Code, create a new file in the **ansible-apache** directory with the following name:

test_apache_playbook.yaml

b. Add the following information to the file. Make sure you use the proper YAML indentation. Every space and dash is significant. You may lose some formatting if you copy and paste.

```
---
- hosts: webservers
  tasks:
    - name: run echo command
      command: /bin/echo hello world
```

Step 2: Run the Ansibl playbook to test your webserver group.

Run the Ansible playbook using the **ansible-playbook** command using the **-v** verbose option. You should see output similar to the following.

```
devasc@labvm:~/labs/ansible/ansible-apache$ ansible-playbook -v
test_apache_playbook.yaml
Using /home/devasc/labs/ansible/ansible-apache/ansible.cfg as config file

PLAY [webservers] ************************************************************

TASK [Gathering Facts] *******************************************************
ok: [192.0.2.3]

TASK [run echo command] ******************************************************
changed: [192.0.2.3] => {"changed": true, "cmd": ["/bin/echo", "hello", "world"],
"delta": "0:00:00.002062", "end": "2020-05-20 21:35:32.346595", "rc": 0, "start":
"2020-05-20 21:35:32.344533", "stderr": "", "stderr_lines": [], "stdout": "hello
world", "stdout_lines": ["hello world"]}

PLAY RECAP *******************************************************************
192.0.2.3                  : ok=2    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0

devasc@labvm:~/labs/ansible/ansible-apache$
```

Step 3: Create your Ansible playbook to install Apache.

a. In VS Code, create a new file in the **ansible-apache** directory with the following name:

install_apache_playbook.yaml

b. Add the following information to the file. Make sure you use the proper YAML indentation. Every space and dash is significant. You may lose some formatting if you copy and paste. The highlighted text is explained in the next step.

```
---
- hosts: webservers
  become: yes
  tasks:
    - name: INSTALL APACHE2
      apt: name=apache2 update_cache=yes state=latest

    - name: ENABLED MOD_REWRITE
      apache2_module: name=rewrite state=present
      notify:
        - RESTART APACHE2

  handlers:
    - name: RESTART APACHE2
      service: name=apache2 state=restarted
```

Step 4: Examine your Ansible playbook.

The following is an explanation of some of the significant lines in your playbook:

- **`hosts: webservers`** - This references the **webservers** group of devices in your **hosts** inventory file. This playbook will be run for all the devices with this group.
- **`become: yes`** - The **become** keyword activates **sudo** command execution, which will allow tasks such as installing applications.
- **`apt:`** - The **apt** module is used to manage packages and application installations on Linux.
- **`handlers:`** - Handlers are similar to a task but are not run automatically. They are called by a task. Notice that the task **ENABLED MOD_REWRITE** calls the handler **RESTART APACHE2**.

Step 5: Run the Ansible backup to install Apache.

Run the Ansible playbook using the **ansible-playbook** command using the **-v** verbose option. The first time Apache installed on your VM, the task **INSTALL APACHE2** will take anywhere from 30 seconds to a few minutes depending on your internet speed.

```
devasc@labvm:~/labs/ansible/ansible-apache$ ansible-playbook -v
install_apache_playbook.yaml
Using /home/devasc/labs/ansible/ansible-apache/ansible.cfg as config file

PLAY [webservers] ************************************************************

TASK [Gathering Facts] *******************************************************
ok: [192.0.2.3]

TASK [INSTALL APACHE2] *******************************************************
ok: [192.0.2.3] => {"cache_update_time": 1590010855, "cache_updated": true, "changed":
false}

TASK [ENABLED MOD_REWRITE] ***************************************************
ok: [192.0.2.3] => {"changed": false, "result": "Module rewrite enabled"}

PLAY RECAP *******************************************************************
192.0.2.3  : ok=3  changed=0  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0

devasc@labvm:~/labs/ansible/ansible-apache$
```

The PLAY RECAP should display **ok** and **failed=0** indicating a successful playbook execution.

Step 6: Verify Apache has been installed.

a. Use the following command to verify that Apache is now installed. Press "q" to quit.

```
devasc@labvm:~/labs/ansible/ansible-apache$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese>
     Active: active (running) since Wed 2020-05-20 03:48:49 UTC; 10min ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 8201 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SU>
   Main PID: 8225 (apache2)
      Tasks: 55 (limit: 4654)
     Memory: 5.3M
     CGroup: /system.slice/apache2.service
             ├─8225 /usr/sbin/apache2 -k start
             ├─8229 /usr/sbin/apache2 -k start
             └─8230 /usr/sbin/apache2 -k start
devasc@labvm:~/labs/ansible/ansible-apache$
```

b. Open the Chromium web browser and enter the IPv4 address for your new server, **192.0.2.3**, to see the default Apache2 web page.

Part 4: Add Options to Your Ansible Playbook for Apache Web Servers

In a production environment, the Apache2 default installation is typically customized for the specific features needed by the organization. An Ansible playbook can help automate these configuration tasks, as well. In this part, you will customize your playbook by specifying that the Apache server use a different port number.

Step 1: Create your Ansible playbook for installing Apache.

a. In VS Code, create a new file in the **ansible-apache** directory with the following name:

install_apache_options_playbook.yaml

b. Add the following information to the file. Make sure you use the proper YAML indentation. Every space and dash is significant. You may lose some formatting if you copy and paste.

```
---
- hosts: webservers
  become: yes
  tasks:
   - name: INSTALL APACHE2
     apt: name=apache2 update_cache=yes state=latest

   - name: ENABLED MOD_REWRITE
     apache2_module: name=rewrite state=present
     notify:
       - RESTART APACHE2

   - name: APACHE2 LISTEN ON PORT 8081
     lineinfile: dest=/etc/apache2/ports.conf regexp="^Listen 80"
line="Listen 8081" state=present
```

```
    notify:
      - RESTART APACHE2

  - name: APACHE2 VIRTUALHOST ON PORT 8081
    lineinfile: dest=/etc/apache2/sites-available/000-default.conf
regexp="^<VirtualHost \*:80>" line="<VirtualHost *:8081>" state=present
    notify:
      - RESTART APACHE2

 handlers:
  - name: RESTART APACHE2
    service: name=apache2 state=restarted
```

This playlist is very similar to the previous one with the addition of two tasks that have the webservers listen on port 8081 instead of port 80.

The **lineinfile** module is used to replace existing lines in the /etc/apache2/ports.conf and /etc/apache2/sites-available/000-default.conf files. You can search the Ansible documentation for more information on the **lineinfile** module.

Step 2: Examine the two files that will be modified by the playbook.

Display the files **/etc/apache2/ports.conf** and **/etc/apache2/sites-available/000-default.conf**. Notice the webserver is currently listening on port 80.

```
devasc@labvm:~/labs/ansible/ansible-apache$ cat /etc/apache2/ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
        Listen 443
                                --- OUTPUT OMMITED ---

devasc@labvm:~/labs/ansible/ansible-apache$ cat /etc/apache2/sites-
available/000-default.conf
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
                                --- OUTPUT OMMITED ---
devasc@labvm:~/labs/ansible/ansible-apache$
```

Step 3: Run the Ansible Playbook.

a. Run the Ansible playbook using the **ansible-playbook** command.

```
devasc@labvm:~/labs/ansible/ansible-apache$ ansible-playbook
install_apache_options_playbook.yaml

PLAY [webservers] ************************************************************

TASK [Gathering Facts] *******************************************************
ok: [192.0.2.3]

TASK [INSTALL APACHE2] *******************************************************
ok: [192.0.2.3]

TASK [ENABLED MOD_REWRITE] ***************************************************
ok: [192.0.2.3]

TASK [APACHE2 LISTEN ON PORT 8081] *******************************************
ok: [192.0.2.3]

TASK [APACHE2 VIRTUALHOST ON PORT 8081] **************************************
ok: [192.0.2.3]

PLAY RECAP *******************************************************************
192.0.2.3                  : ok=5    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0

devasc@labvm:~/labs/ansible/ansible-apache$
```

Step 4: Verify that Apache has been installed.

a. View the files **/etc/apache2/ports.conf** and **/etc/apache2/sites-available/000-default.conf** again. Notice that the playbook modified these files to listen on port 8081.

```
devasc@labvm:~/labs/ansible/ansible-apache$ cat /etc/apache2/ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8081

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
devasc@labvm:~/labs/ansible/ansible-apache$
```

```
devasc@labvm:~/labs/ansible/ansible-apache$ cat /etc/apache2/sites-
available/000-default.conf
<VirtualHost *:8081>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
<output omitted>
devasc@labvm:~/labs/ansible/ansible-apache$
```

b. Open the Chromium web browser and enter the IPv4 address for your new server. But this time specify 8081 as the port number, **192.0.2.3:8081**, to see the default Apache2 web page.

Note: Although you can see in the **ports.conf** file that Apache2 is also listening on port 443, this is for secure HTTP. You have not yet configured Apache2 for secure access. This, of course, would be added to your Ansible playbook, but is beyond the scope of this course.