Lab - Use Ansible to Automate Installing a Web Server

A diagram of software development

Description automatically generated

# 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

* Virtual Machine on the University’s VDrive
* Run the following command to enable clipboard copy/paste: **sudo VBoxClient --clipboard**

# Instructions

**Part 1: Configure Ansible**

The 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 LAB VM.

### Install SSH

lab@lab-virtualbox:~$ **sudo apt-get install openssh-server**

### Install sshpass

lab@lab-virtualbox:~$ **sudo apt-get install sshpass**

## Step 2: Enable the SSH server.

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

lab@lab-virtualbox:~$ **sudo systemctl start ssh**

lab@lab-virtualbox:~$

## Step 3: Open the ansible directory in VS Code.

## Create a directory ansible in the devopsLabs directory. In ansible directory make a subdirectory ansible-apache. In this subdirectory create following files: hosts, servers, ansible.cfg

1. Open **VS Code**.
2. Click **File > Open Folder...** and navigate to the **/devopsLabs/ansible** folder.
3. Click **OK**.
4. The subdirectory for the Ansible lab is 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

1. The first thing you need to load the module “dummy”, you can also add the option “numdummies = 2” to immediately create two dummyX interface:

**sudo modprobe -v dummy numdummies=2**

Let’s see if the module is loaded: **lsmod | grep dummy**

Let’s see if the interfaces were created: **ifconfig -a | grep dummy**

You can add or remove an IP address from the dummy0 interface, for example:

**sudo ip addr add 192.168.1.150/24 dev dummy0**

Change the MAC address as follows:

**sudo ip link set dummy0 address 00:00:00:11:11:11**

1. Open the **hosts** file in the **ansible-apache** directory.
2. Add the following lines to the **hosts** file and save.

[webservers]

192.168.1.150 ansible\_ssh\_user=lab ansible\_ssh\_pass=changeme

1. The credentials **lab** and **changeme** are admin credentials for the lab 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.

lab@lab-virtualbox:~/devopsLabs/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 pfifo\_fast state UP group default qlen 1000**

**link/ether 08:00:27:fd:fb:20 brd ff:ff:ff:ff:ff:ff**

**inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3**

**valid\_lft 83646sec preferred\_lft 83646sec**

**inet6 fe80::fc77:45e:bf44:273f/64 scope link noprefixroute**

**valid\_lft forever preferred\_lft forever**

**3: br-2ee8d365e62c: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default**

**link/ether 02:42:87:0e:30:df brd ff:ff:ff:ff:ff:ff**

**inet 172.18.0.1/16 brd 172.18.255.255 scope global br-2ee8d365e62c**

**valid\_lft forever preferred\_lft forever**

**4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default**

**link/ether 02:42:95:90:e5:56 brd ff:ff:ff:ff:ff:ff**

**inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0**

**valid\_lft forever preferred\_lft forever**

**5: dummy0: <BROADCAST,NOARP> mtu 1500 qdisc noop state DOWN group default qlen 1000**

**link/ether 46:73:09:03:2e:73 brd ff:ff:ff:ff:ff:ff**

**inet 192.168.1.150/24 scope global dummy0**

**valid\_lft forever preferred\_lft forever**

**6: dummy1: <BROADCAST,NOARP> mtu 1500 qdisc noop state DOWN group default qlen 1000**

**link/ether 06:af:03:b4:ee:66 brd ff:ff:ff:ff:ff:ff**

lab@lab-virtualbox:~/devopsLabs/ansible$

## Step 5: Edit the ansible.cfg file.

1. In the **ansible-apache** subdirectory, Open the **ansible.cfg**.
2. 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 # Hide deprecation warnings

Deprecation\_warnings = 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.

Using either the terminal within VS Code or a separate terminal changed to the path **~/devopsLabs/ansible/ansible-apache**. Use the Ansible **ping** module to verify communications with the devices listed within the **webservers** group of your **hosts** inventory file.

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$ **ansible webservers -m ping**

192.168.1.150 | SUCCESS => {

"ansible\_facts": {

"discovered\_interpreter\_python": "/usr/bin/python3"

},

"changed": false, "ping": "pong"

}

lab@lab-virtualbox:~/devopsLabs/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”.

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$ **ansible webservers -m command -a "/bin/echo hello world"**

192.168.1.150| CHANGED | rc=0 >> hello world

lab@lab-virtualbox:~/devopsLabs/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.

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

### test\_apache\_playbook.yaml

1. Add the following information to the file. Make sure you use the proper YAML indentation. Every space and dash are 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 Ansible 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.

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$ **ansible-playbook -v test\_apache\_playbook.yaml**

Using /home/lab/devopsLabs/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 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

192.168.1.150

skipped=0

: ok=2 changed=1 ignored=0

unreachable=0

failed=0

rescued=0

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$

## Step 3: Create your Ansible playbook to install Apache.

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

### install\_apache\_playbook.yaml

1. Add the following information to the file. Make sure you use the proper YAML indentation. Every space and dash are 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 is installed, the task **INSTALL APACHE2** will take anywhere from 30 seconds to a few minutes depending on your internet speed. Enter the password when prompts.

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$ **ansible-playbook -v install\_apache\_playbook.yaml -bK**

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 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

192.168.1.150 : ok=3 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$

The PLAY RECAP should display **ok=4** and **failed=0** indicating a successful playbook execution.

## Step 6: Verify Apache has been installed.

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

lab@lab-virtualbox:~/devopsLabs/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

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$

1. Open the Chromium web browser and enter the IPv4 address for your new server, **192.168.1.150**, 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.

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

### install\_apache\_options\_playbook.yaml

1. 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.

lab@lab-virtualbox:~/devopsLabs/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 ---

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$

lab@lab-virtualbox:~/devopsLabs/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 ---

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$

## Step 3: Run the Ansible Playbook.

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

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$ **ansible-playbook install\_apache\_options\_playbook.yaml -bK**

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

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ok: [192.168.1.150]

TASK [INSTALL APACHE2] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ok: [192.168.1.150]

TASK [ENABLED MOD\_REWRITE] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ok: [192.168.1.150]

TASK [APACHE2 LISTEN ON PORT 8081] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ok: [192.168.1.150]

TASK [APACHE2 VIRTUALHOST ON PORT 8081] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ok: [192.168.1.150]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

192.168.1.150

skipped=0 rescued=0

: ok=6 changed=0 ignored=0

unreachable=0

failed=0

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$

## Step 4: Verify that Apache has been installed.

1. 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.

lab@lab-virtualbox:~/devopsLabs/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 lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$

lab@lab-virtualbox:~/devopsLabs/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 OMMITED ---

lab@lab-virtualbox:~/devopsLabs/ansible/ansible-apache$

1. Open the Chromium web browser and enter the IPv4 address for your new server. But this time specify 8081 as the port number, **192.168.1.150:8081**, 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.