**A picture containing clipart

Description generated with very high confidence**

ANSIBLE HANS-ON

DevOps Certification Training

[**support@intellipaat.com**](mailto:support@intellipaat.com)

**+91-7022374614**

**US: 1-800-216-8930(Toll Free)**

**ANSIBLE HANDS-ON**

Ansible hands-on documentation has been divided into 3 segments.

**A.** Creating Ansible Playbook

**B.** Creating Ansible Roles

**C.** Using Ansible Roles in Playbook

**Prerequisites:**

1. Ansible needs to be installed in master.
2. Connection between Master and Host needs to be set through ssh. For more information refer to the Ansible Installation Documentation.

**A-** **Creating Ansible Playbook**

This playbook consists of two plays with following tasks:

* Play 1: Execute a command in host1, Execute a script in host1
* Play 2: Execute a script in host2, Install nginx in host2

**Step 1:** Create the .yml file.

sudo nano <playbookname>



**Step 2:** Add the following content in the .yml file.

---

- hosts: host1

sudo: yes

name: Play 1

tasks:

- name: Execute command ‘Date’

command: date

- name: Execute script on server

script: test\_script.sh

- hosts: host2

name: Play 2

sudo: yes

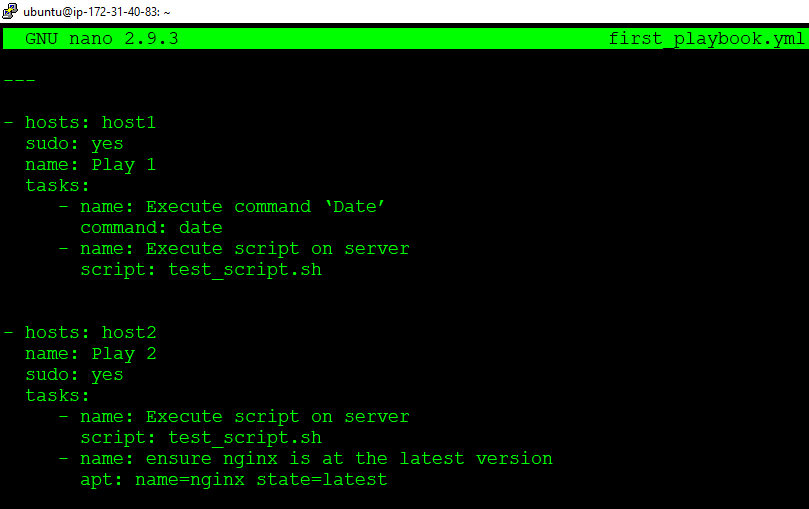
tasks:

- name: Execute script on server

script: test\_script.sh

- name: Install nginx

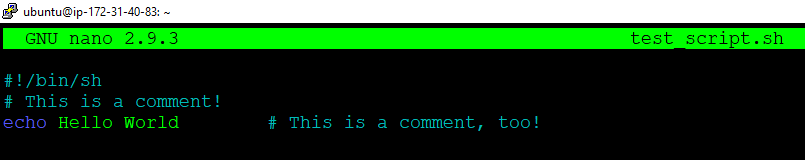
apt: name=nginx state=latest



**Step 3:** Now to be able to perform “Execute script on server” task we need to have the **.sh** file (unix (linux) shell executables files) in master machine. Create test.sh file as shown.

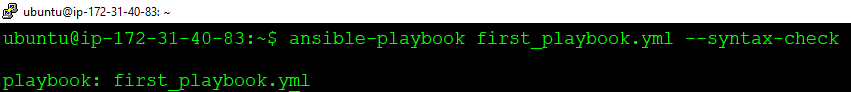
sudo nano <file\_name>





**Step 4:** Before executing the playbook that we just created we need to have to check for syntax errors.

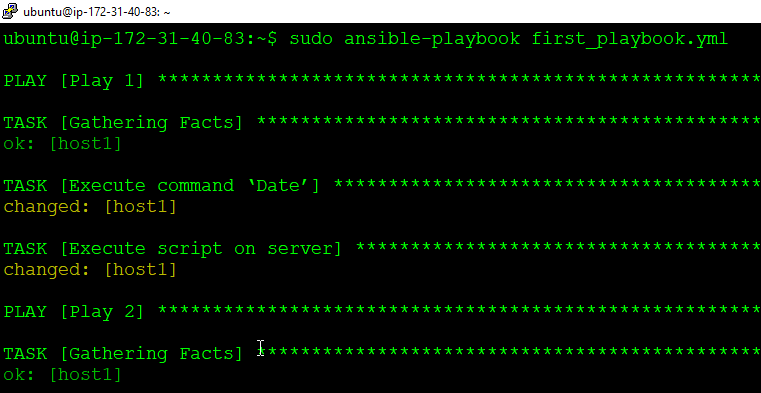
ansible-playbook <playbook> --syntax-check



This means our playbook is syntax error free. Let us move ahead and execute the playbook.

**Step 5:** To execute the playbook use the following command.

ansible-playbook <playbook>



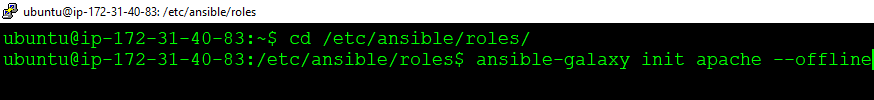
Great! We have successfully created our very first Ansible playbook.

Remember that using playbook we can run the same command repeatedly, but if everything was configured on the first run, then all subsequent runs make no changes.

**B- Creating Ansible Roles**

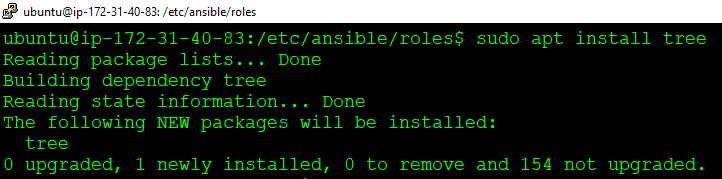
**Step 1:** Ansible roles should be written inside “*/etc/ansible/roles/* ”. Use the following command to create one Ansible role.

ansible-galaxy init <role name> --offline



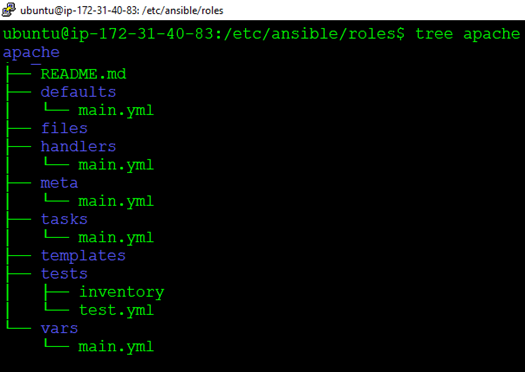
**Step 2:** Install tree package using sudo apt install tree. Use tree command to view structure of the role

sudo apt install tree



Now let us see the structure of the role that we just created using the following command.

tree <role name>

****

Now we are ready to create the tasks that our roles are supposed to perform.

**Step 3:** Go inside *task* folder inside *apache* directory. Edit *main.yml* using the following command. Make changes as shown. Save and then exit.

sudo nano main.yml



Now we will divide the tasks to be performed into three categories. Install, configure and services. We will create three different .yml files to reduce the complexity. Include those separate task files in the main.yml file as shown.

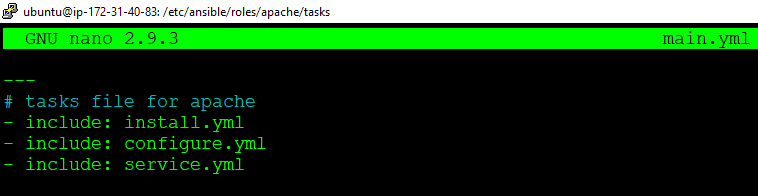
---

# tasks file for apache

- include: install.yml

- include: configure.yml

- include: service.yml

****

Remember that order of the list in yml file matters. So here install.yml gets executed first, then configure.yml and then service.yml.

**Step 4:** Now inside *task* folder, create *install.yml* and add the installation tasks to be performed as shown below.

sudo nano install.yml

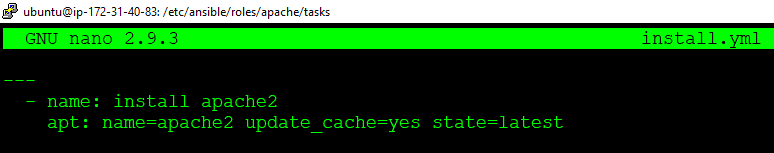
****

We will install the latest version of apache2 in the remote machine with the help of apt module as shown below.

---

- name: install apache2

apt: name=apache2 update\_cache=yes state=latest

****

**Step 5:** Then create *configure.yml* and add the required configurations that need to be performed on remote machine as shown below.

sudo nano configure.yml

****

We will configure apache2.conf file in the remote machine and also, we will restart the apache2 service. Then we will send one file from */etc/ansible/roles/apache/files* folder to the remote machine. The destination path has been set to /home/ubuntu/ as shown.

--

#configure apache2.conf and send copy.html file

- name: apache2.conf file

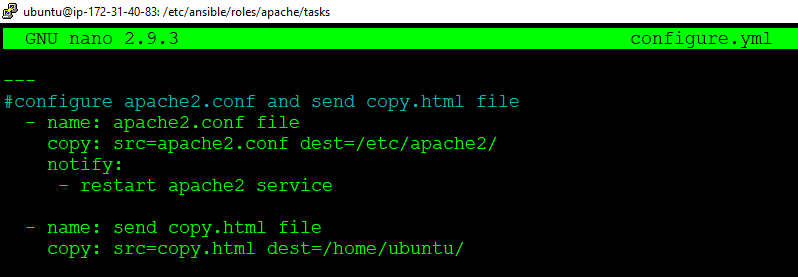
copy: src=apache2.conf dest=/etc/apache2/

notify:

- restart apache2 service

- name: send copy.html file

copy: src=copy.html dest=/home/ubuntu/

****

**Step 6:** Again, inside *task* folder, create *service.yml* and add the required configurations that need to be performed on remote machine as shown below.

sudo nano service.yml

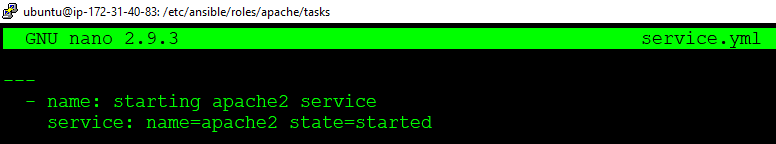
****

We will configure apache2.conf file in the remote machine

---

- name: starting apache2 service

service: name=apache2 state=started

****

**Step 7:** Now, Now go inside files. Store the files that needs to be pushed to the remote machine. Copy the apache2.conf file from */etc/apache2* directory to */etc/ansible/roles/apache/files* and create the html file.

cp /etc/apache2/apache2.conf /etc/ansible/roles/apache/files



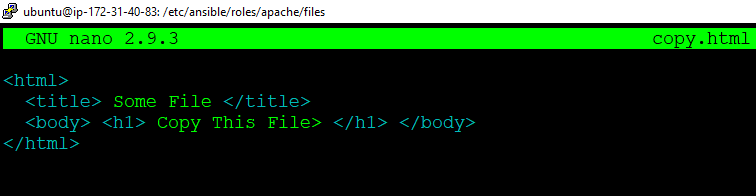
Create one html file as well. My dummy html file looks like this.

<html>

<title> Some File </title>

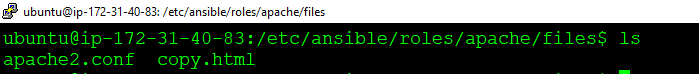
<body> <h1> Copy This File> </h1> </body>

</html>



Check whether our files are ready or not by using the following command.

ls



**Step 8:** Go inside handlers and add the action that needs to be performed after *notify* from *configure.yml* is executed. Use the following two commands.

cd /etc/ansible/roles/apache/handlers/

sudo nano main.yml

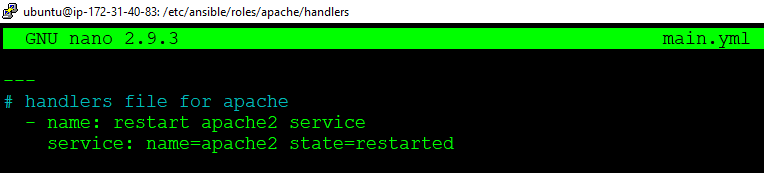
Add the following content inside handlers file.

---

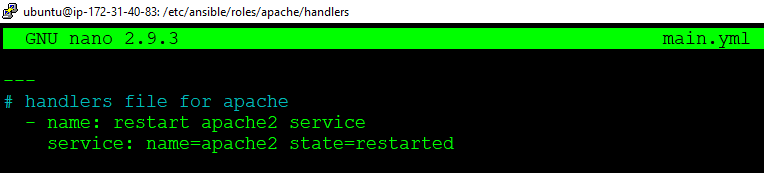
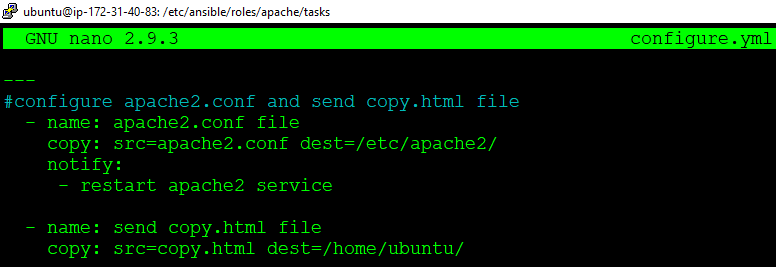
# handlers file for apache

- name: restart apache2 service

service: name=apache2 state=restarted



Remember that notify name and handler name should match.



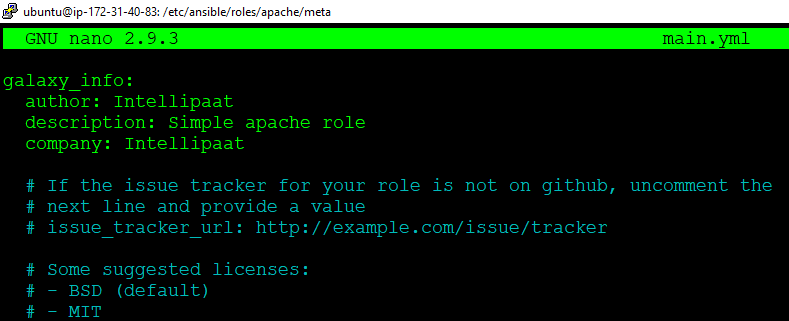
**Step 9:** Go inside meta and add information related to the role.

cd /etc/ansible/roles/apache/handlers/

sudo nano main.yml



Add author information, role descriptions, company information etc. as shown below.



**Step 10:** Go to the */etc/ansible/* and create one top level *.yml* file where we can add hosts and roles to be executed. Execute apache role on the hosts that is under the group name servers, added in the inventory file */etc/ansible/hosts*

cd /etc/ansible/

sudo nano site.yml



For more than one hosts following commands can be used.

---

- hosts: servers

sudo: yes

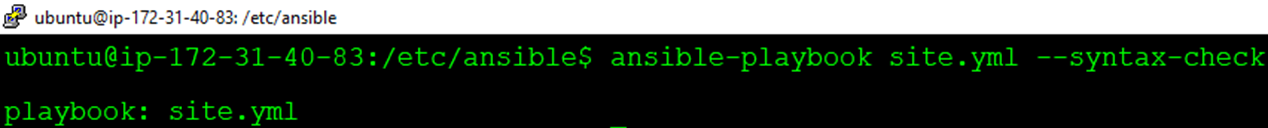
roles:

- apache



**Step 11:** Before we execute our top level yml file we will check for syntax errors.

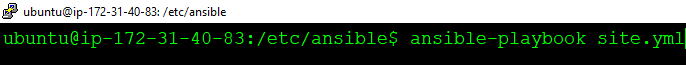
ansible-playbook <filename.yml> --syntax-check



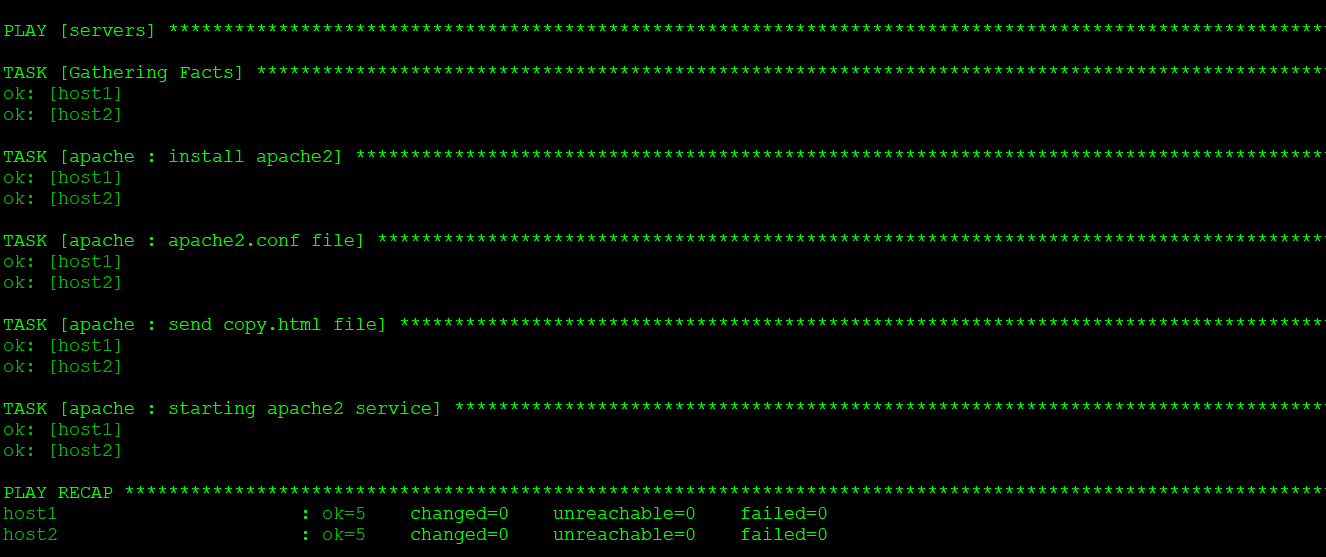
put our configuration in there as shown below.

**Step 12:** Execute the top level *.yml* file

ansible-playbook <filename.yml>



The output looks like this.



Congratulations! You have successfully created Ansible Role.

Now let us see how to use this Ansible role that we’ve just created along with other tasks in a Ansible Playbook.

**C-** **Using Ansible Roles in Playbook**

**Step 1:** To use ansible roles along with other tasks in playbook use *import\_role* and *include\_role.* Create one playbook called to execute on the remote machines along with two *debug* tasks before and after *apache role*.

sudo nano <playbook name>



Add the following .yml file as shown.

---

- hosts: servers

sudo: yes

tasks:

- debug:

msg: "before we run our role"

- import\_role:

name: apache

- include\_role:

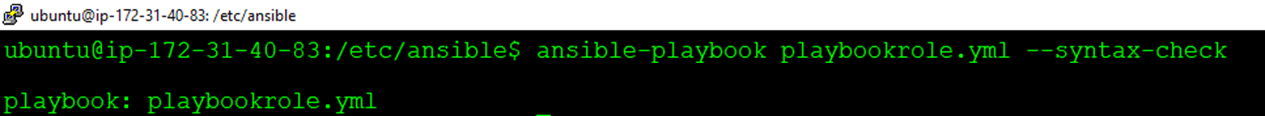
name: apache

- debug:

msg: "after we ran our role"

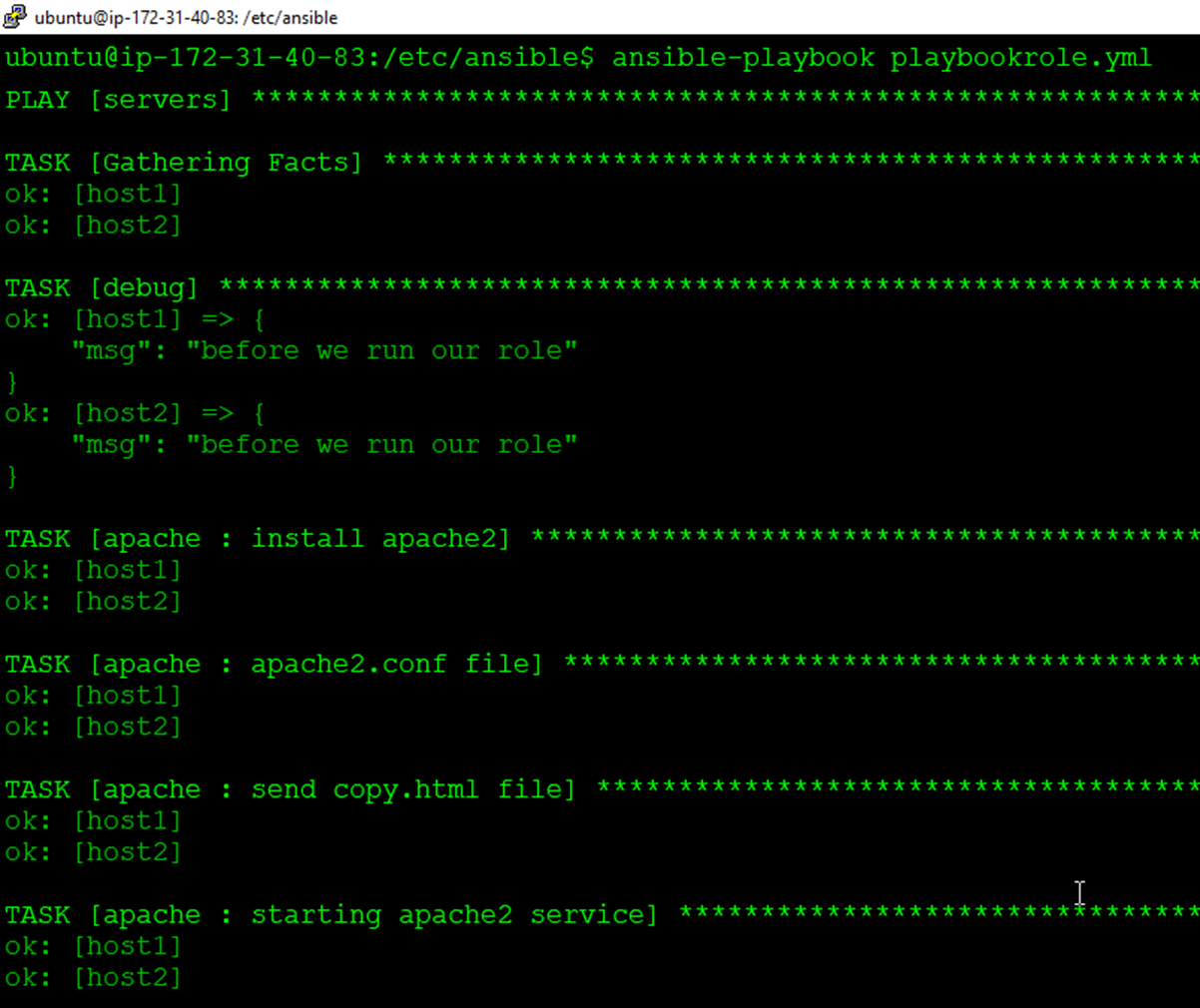
**Step 2:** Check for syntax error and execute the playbook with roles.

ansible-playbook <playbookname> --syntax-check



**Step 3:** Check for syntax error and execute the playbook with roles.

ansible-playbook <playbookname>



**Congratulations!** You have successfully integrated Ansible roles with Ansible playbook.