11. Automation using Ansible: Spin up a new Linux VM using Ansible playbook.

Ansible is an open-source automation tool for provisioning, application deployment (WordPress deployment in this case), and configuration management. Gone are the days of SSH'ing into your server to run a command or hacking together bash scripts to semi-automate laborious tasks. Whether you're managing a single server or an entire fleet, Ansible can not only simplify the process but save you time. So, what makes Ansible so great?

Ansible is completely agent-less, meaning you don't have to install any software on your managed hosts. All commands are run through Ansible via SSH and if Ansible needs updating you only need to update your single control machine and not any remote machines. The only prerequisite to running Ansible commands is to have Python installed on your control machine.

Procedure:

Installation

Step 1: First, ensure that pip is installed.

```
sudo easy_install pip
```

Step 2: Then install Ansible.

```
sudo pip install ansible
```

Step 3: Once the installation has completed you can verify that everything installed correctly by issuing:

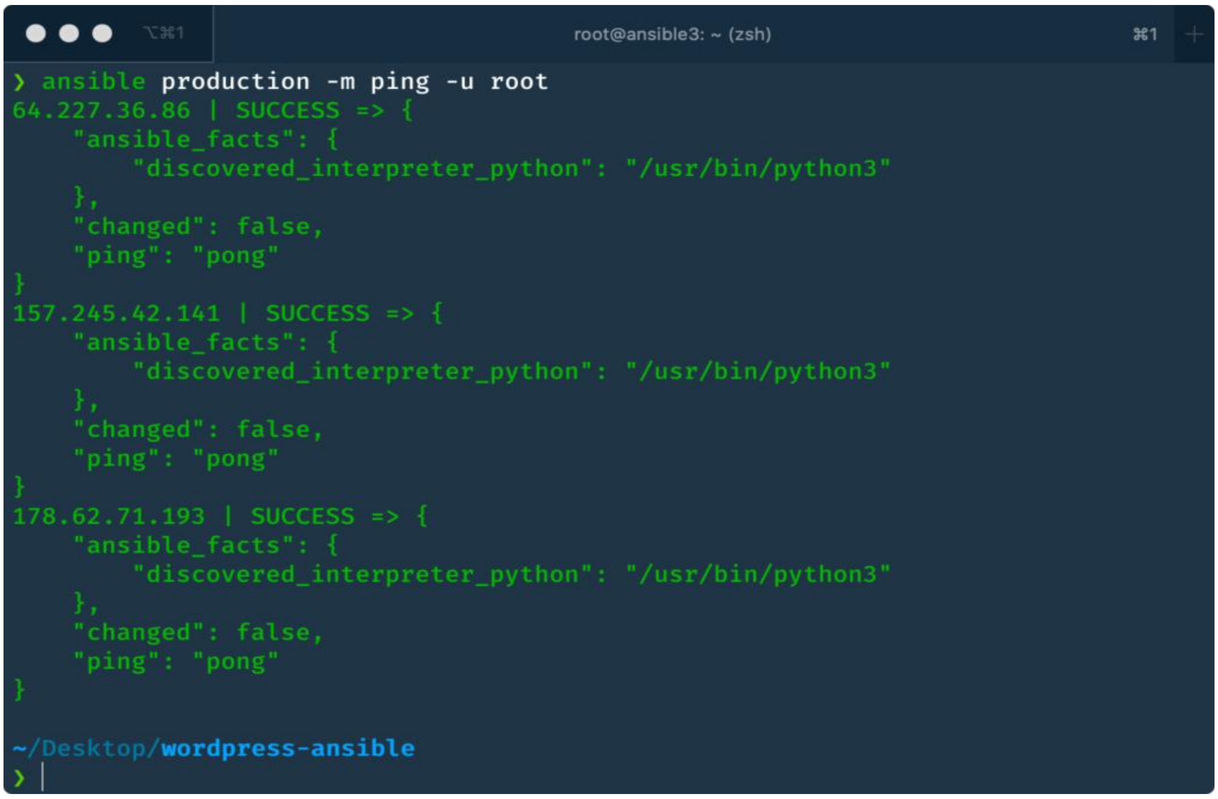
```
ansible --version
```

Step 4: If you were installing Ansible on Ubuntu the commands would be:

```
sudo apt update

sudo apt install software-properties-common

sudo apt-add-repository --yes --update ppa:ansible/ansible

sudo apt install ansible
```

Running Commands

ansible production -m ping -u root



```
❯ ansible production -m ping -u root
64.227.36.86 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
157.245.42.141 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
178.62.71.193 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}

~/Desktop/wordpress-ansible
❯ |
```

Playbooks

Playbooks allow you to chain commands together, essentially creating a blueprint or set of procedural instructions. Ansible will execute the playbook in sequence and ensure the state of each command is as desired before moving onto the next. This is what makes Ansible idempotent. If you cancel the playbook execution partway through and restart it later, only the commands that haven’t completed previously will execute. The rest will be skipped.

Playbooks allow you to create truly complex instructions, but if you’re not careful they can quickly become unwieldy (think of god classes in OOP), which brings us onto roles.

Roles add organization to playbooks. They allow you to split your complex build instructions into smaller reusable chunks, very much like a function in programming terms. This makes it possible to share your roles across different playbooks, without duplicating code. For example, you may have a role for installing Nginx and configuring sensible defaults, which can be used across multiple hosting environments.

Organization of Playbook

ansible.cfg

hosts

provision.yml

roles

nginx

handlers

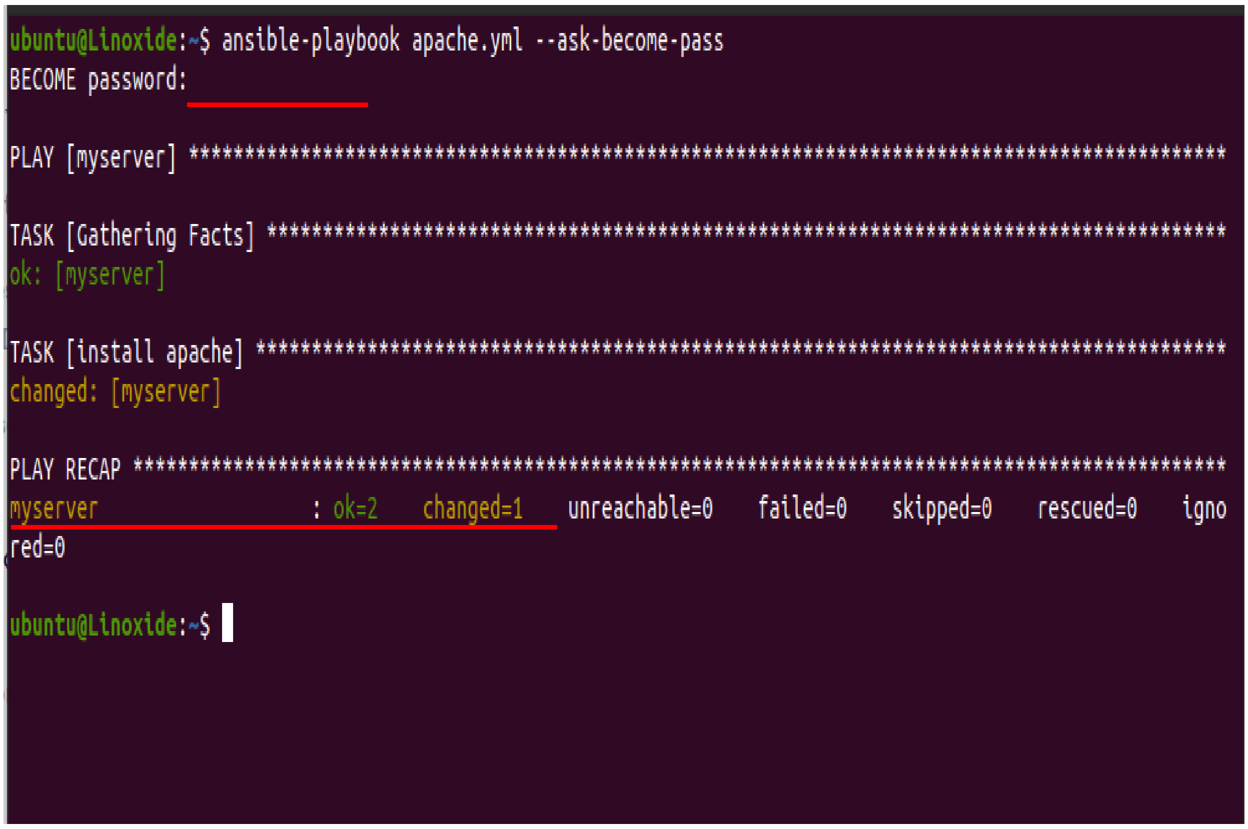
main.yml

tasks

main.yml

```
--- - hosts: production

user: root

 vars:
username: ashley
password:
$6$rlLdG6wd1CT8v7i$7psP8l26lmaPhT3cigoYYXhjG28CtD1ifILq9KzvA0W0
TH2Hj4.iO43RkPWgJGIi60Mz0CsxWbRVBSQkAY95W0
public_key: ~/.ssh/id_rsa.pub
 roles:
common
- ufw
- user
- nginx
- php
- mariadb
- wp
-cli
- ssh
```

OUTPUT



```
ubuntu@Linoxide:~$ ansible-playbook apache.yml --ask-become-pass
BECOME password:

PLAY [myserver] *****************************************************************************

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

TASK [install apache] ***********************************************************************
changed: [myserver]

PLAY RECAP **********************************************************************************
myserver                   : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    igno
red=0

ubuntu@Linoxide:~$ 
```