Ansible Tutorial

1. Create a Vagrant File to create two machines with different private keys in the same network to connect both of them. One machine is called master and the other one slave.

```
Vagrant.configure(2) do |config|
  config.vm.define "master" do |master|
    master.vm.box ="ubuntu/focal64"
    master.vm.network "private_network", ip: "10.0.0.200"
  end
  config.vm.define "slave" do |slave|
    slave.vm.box ="ubuntu/focal64"
    slave.vm.network "private_network", ip: "10.0.0.201"
  end
end
```

2. Run command vagrant up to run the instructions of the Vagrantfile
3. Run command vagrant ssh [MACHINE NAME] to connect to both virtual machines.

vmMaster: vagrant ssh master

vmSlave: vagrant ssh slave

4. Create a SSH key. vmMaster: ssh-keygen -t rsa
5. Copy public key and add it to the aouthorized_keys file on the slave machine. vmSlave: nano ~/.ssh/authorized_keys
6. Get Ip address of both machines.

vmMaster: ip --brief addr show

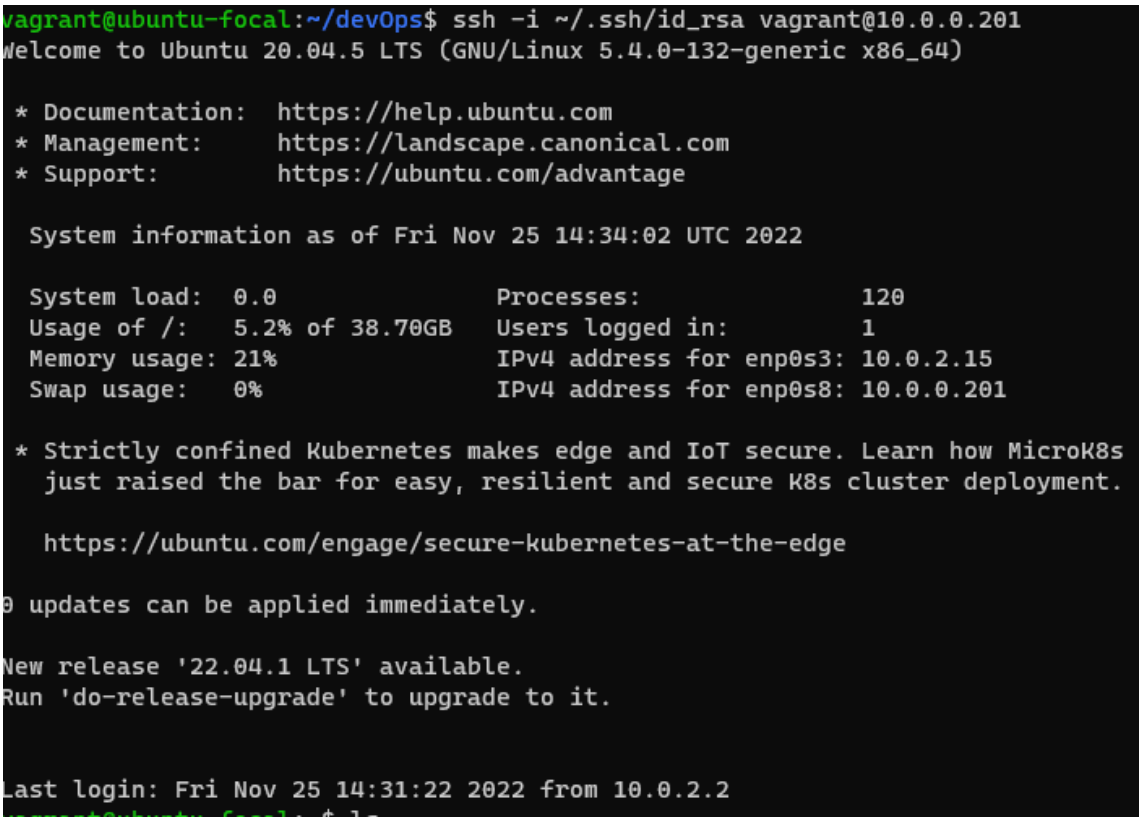
```
vagrant@ubuntu-focal:~/devOps$  ip --brief addr show
lo               UNKNOWN        127.0.0.1/8 ::1/128
enp0s3           UP             10.0.2.15/24 fe80::9a:94ff:fec9:30e2/64
enp0s8           UP             10.0.0.200/24 fe80::a00:27ff:fe5e:fe99/64
```

vmSlave: ip --brief addr show

```
vagrant@ubuntu-focal:~$ ip --brief addr show
lo               UNKNOWN        127.0.0.1/8 ::1/128
enp0s3           UP             10.0.2.15/24 fe80::9a:94ff:fec9:30e2/64
enp0s8           UP             10.0.0.201/24 fe80::a00:27ff:fe8b:dade/64
```

7. Test SSH connection between both machines.

vmMaster: ssh -i ~/.ssh/id_rsa vagrant@10.0.0.201



8. Add an official Ansible repository to our Ubuntu package manager to install ansible.

vmMaster: sudo apt-add-repository ppa:ansible/ansible

9. Now update your package lists and install Ansible:

vmMaster: sudo apt update

vmMaster: sudo apt install -y ansible

10. Create an inventory file to define the hosts and groups of hosts upon which commands, modules, and tasks of a playbook file.

```
[linuxslave]
10.0.0.201
```

11. To make shure our inventory list sintax is correct, and our managed nodes are added successfully we use ansible-inventory command with the -y flag to format our output in YAML.

vmMaster: ansible-inventory --list -i inventory

```
vagrant@ubuntu-focal:~/devOps$ ansible-inventory --list -i inventory
{
    "_meta": {
        "hostvars": {}
    },
    "all": {
        "children": [
            "linuxslave",
            "ungrouped"
        ]
    },
    "linuxslave": {
        "hosts": [
            "10.0.0.201"
        ]
    }
}
```

12. Test connection to our managed nodes.

vmMaster: ansible all -m ping -i inventory

```
vagrant@ubuntu-focal:~/devOps$ ansible all -m ping -i inventory
10.0.0.201 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

13. Create a playbook.yml to lists of tasks that automatically execute against hosts.

```
- name: linux host
  hosts: linuxslave
  become: true
  tasks:
    - name: list
      command: ls
```

14. Run Ansible playbook

vmMaster: ansible-playbook -i inventory playbook.yaml

```
vagrant@ubuntu-focal:~/devOps$ ansible-playbook -i inventory playbook.yaml

PLAY [linux host] ***************************************************************************************

TASK [Gathering Facts] **********************************************************************************
ok: [10.0.0.201]

TASK [list] *********************************************************************************************
changed: [10.0.0.201]

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