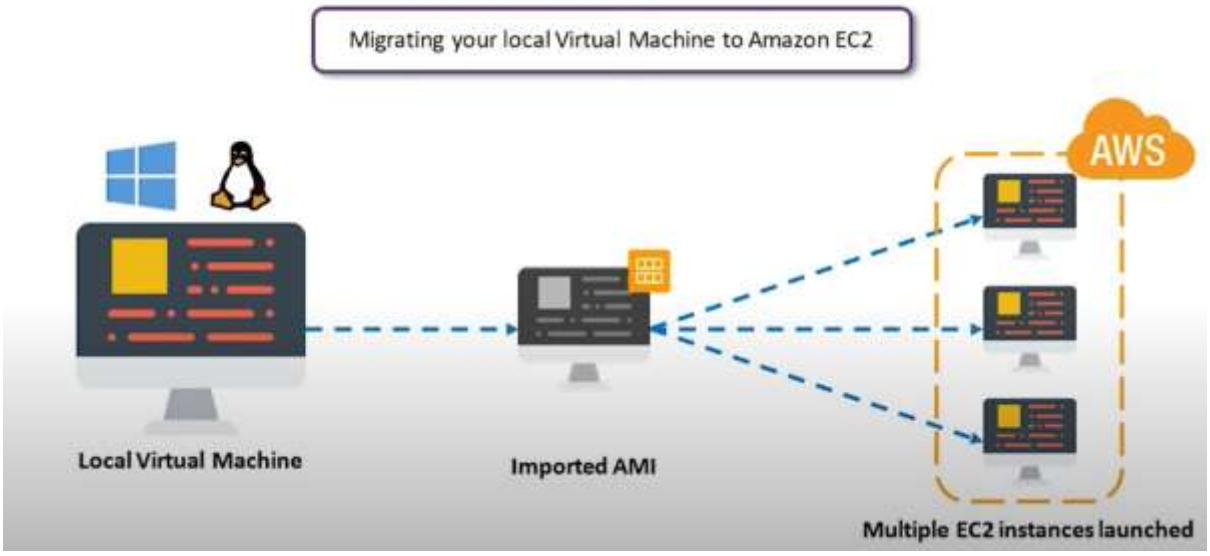
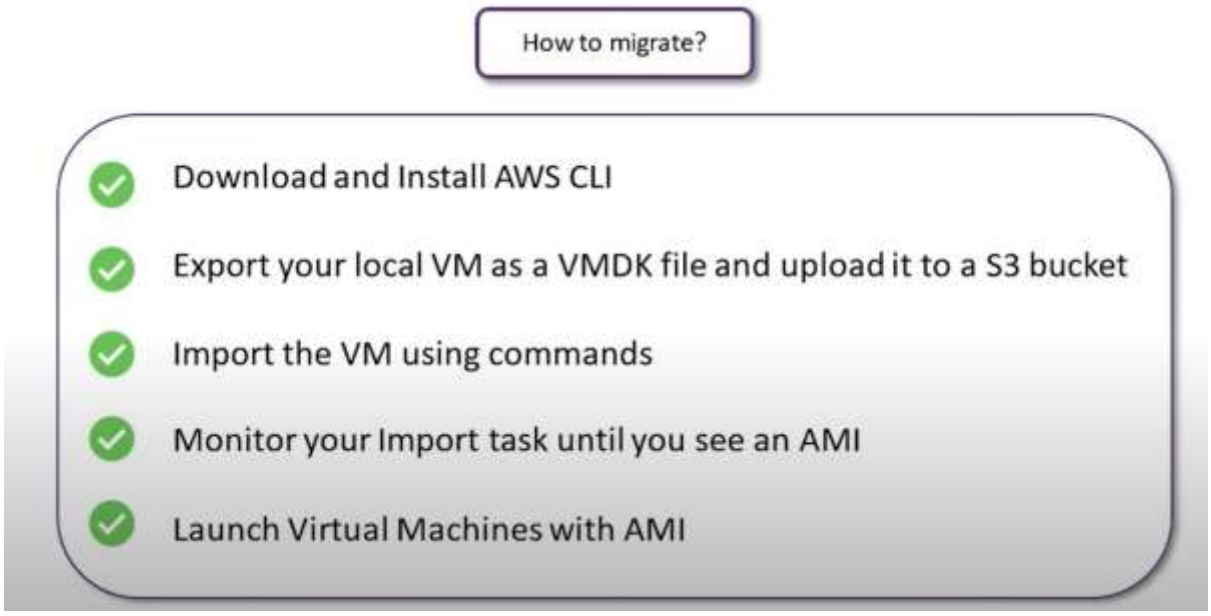
Virtual machine migration from on prem to cloud

Migrating your local Virtual machine to cloud (ec2)

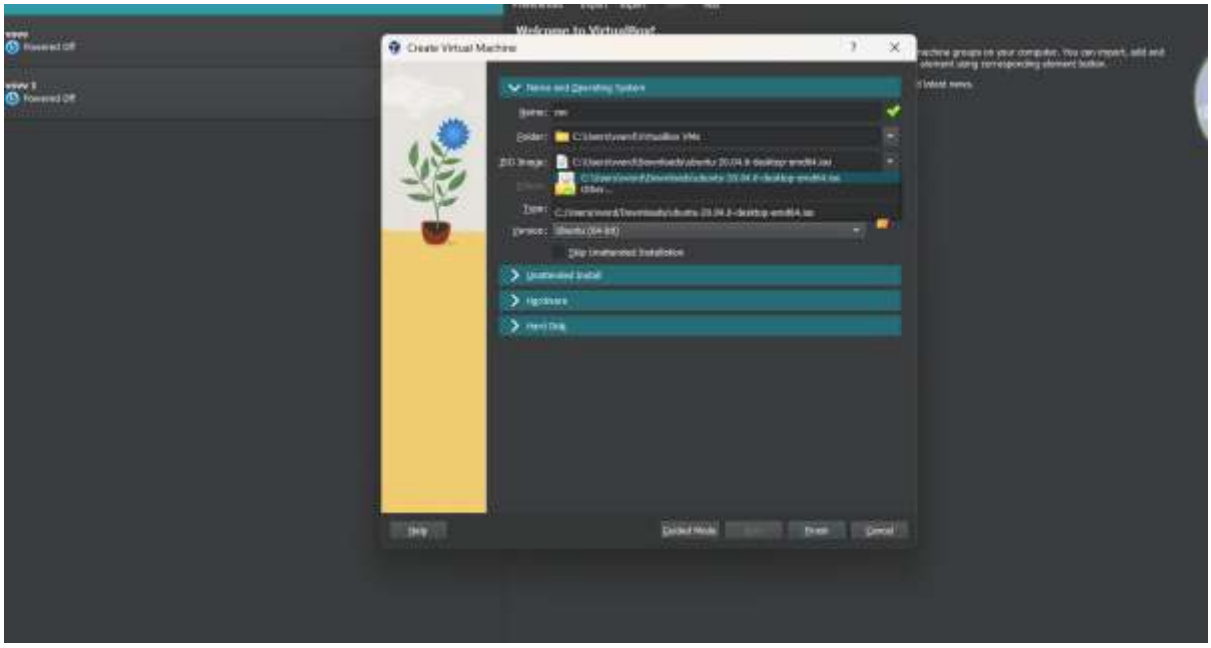




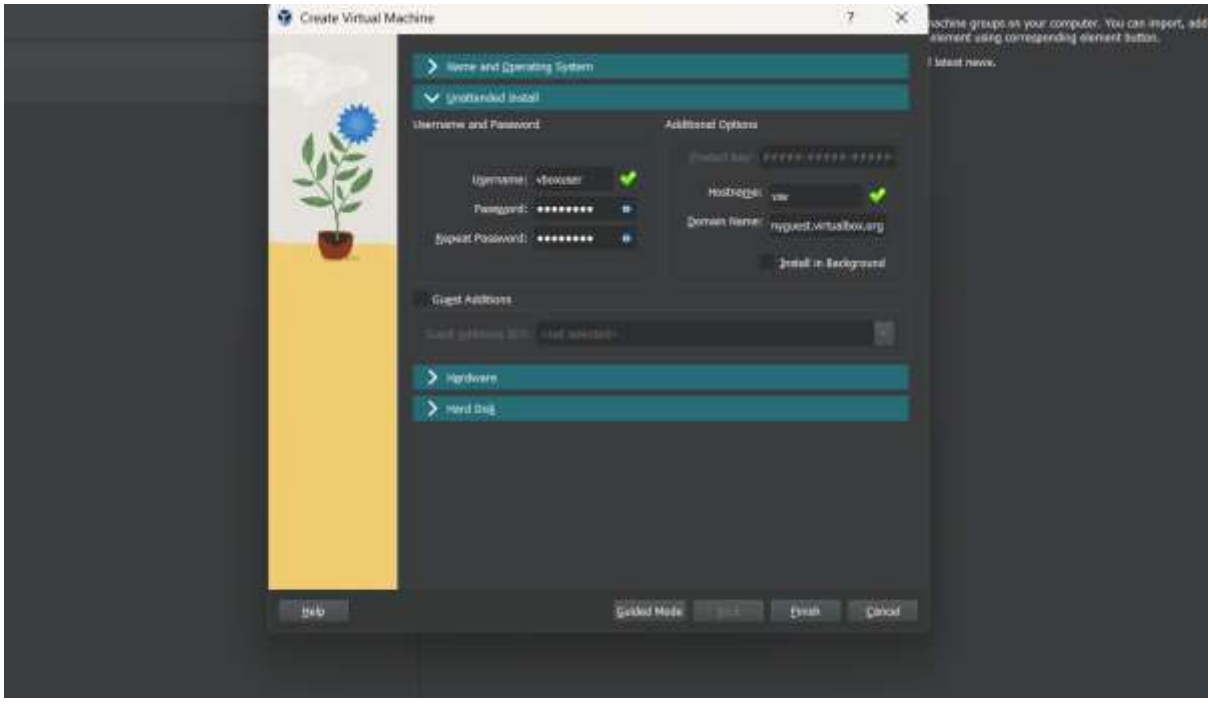
Click on new

https://www.releases.ubuntu.com/20.04/ubuntu-20.04.6-desktop-amd64.iso

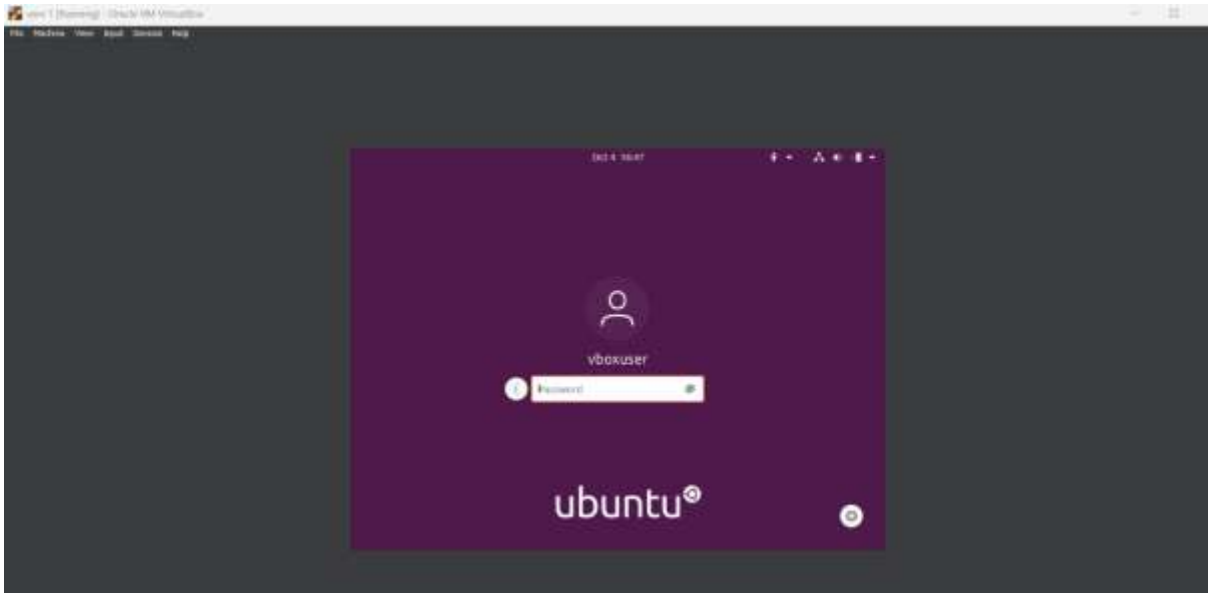
ubuntu desktop images download url



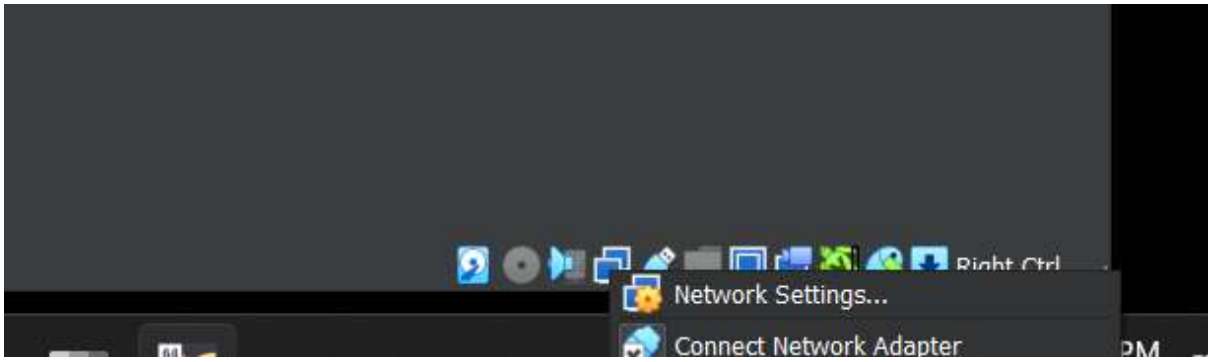
Select your ubuntu image file path



Here change your passowrd

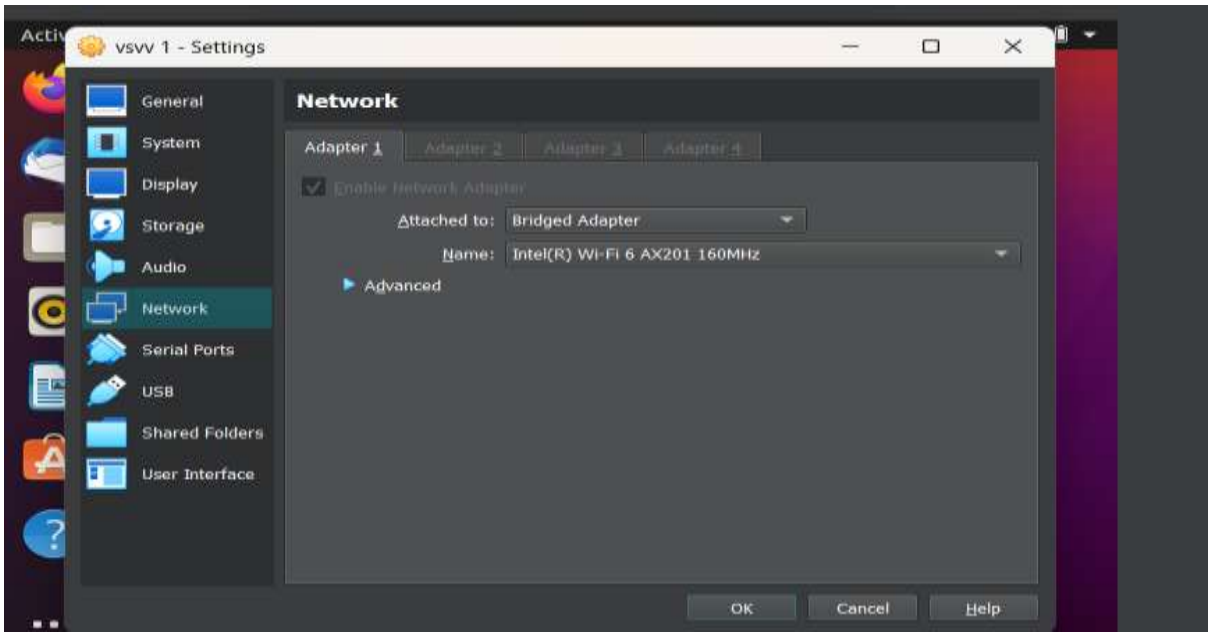


Login through your password

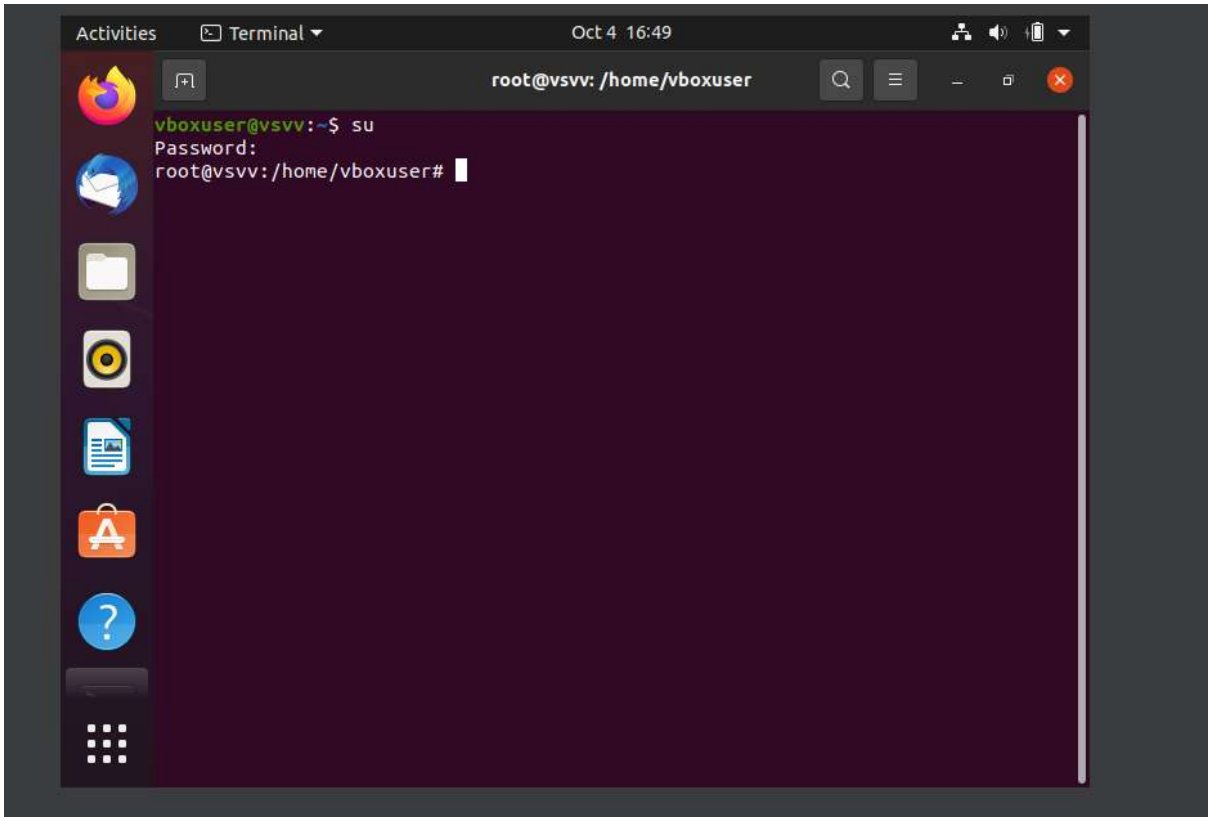


Change internet conectvity

Right click on network settings



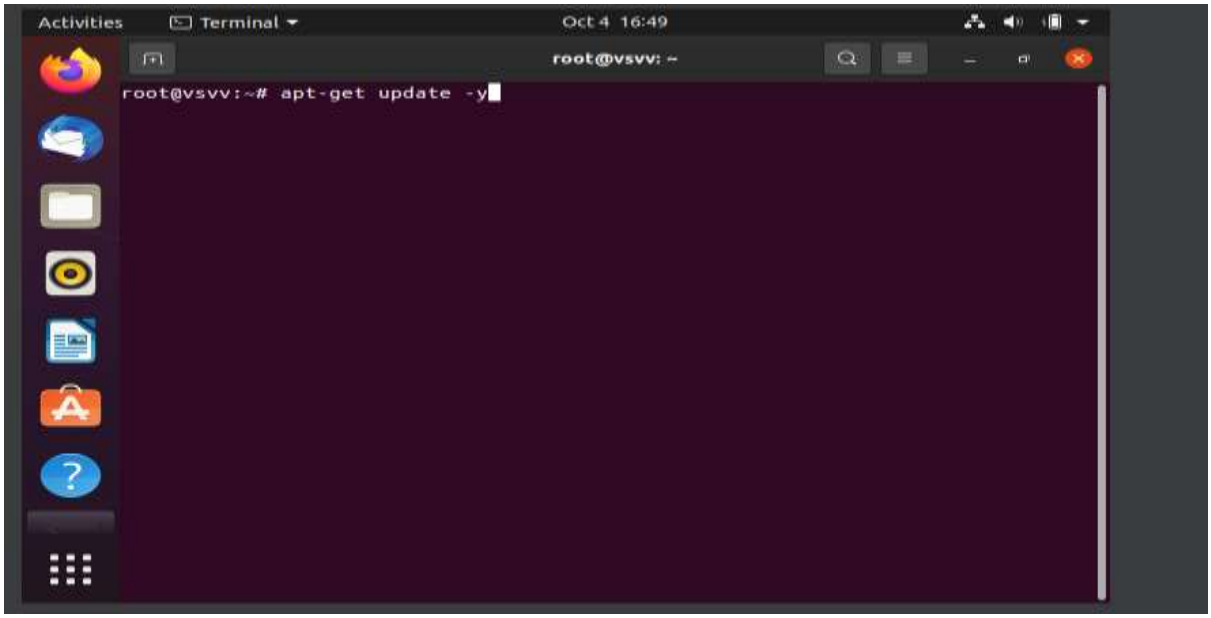
Select bridge adapter like above



Open the terminal in vm

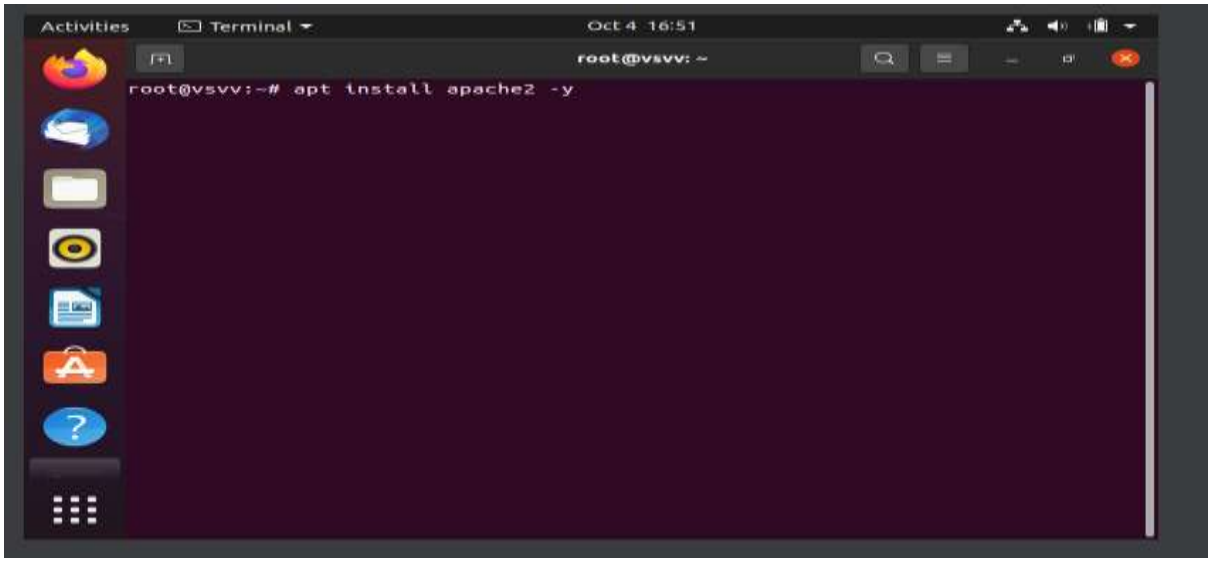
Switch to root using su

Password is your passord



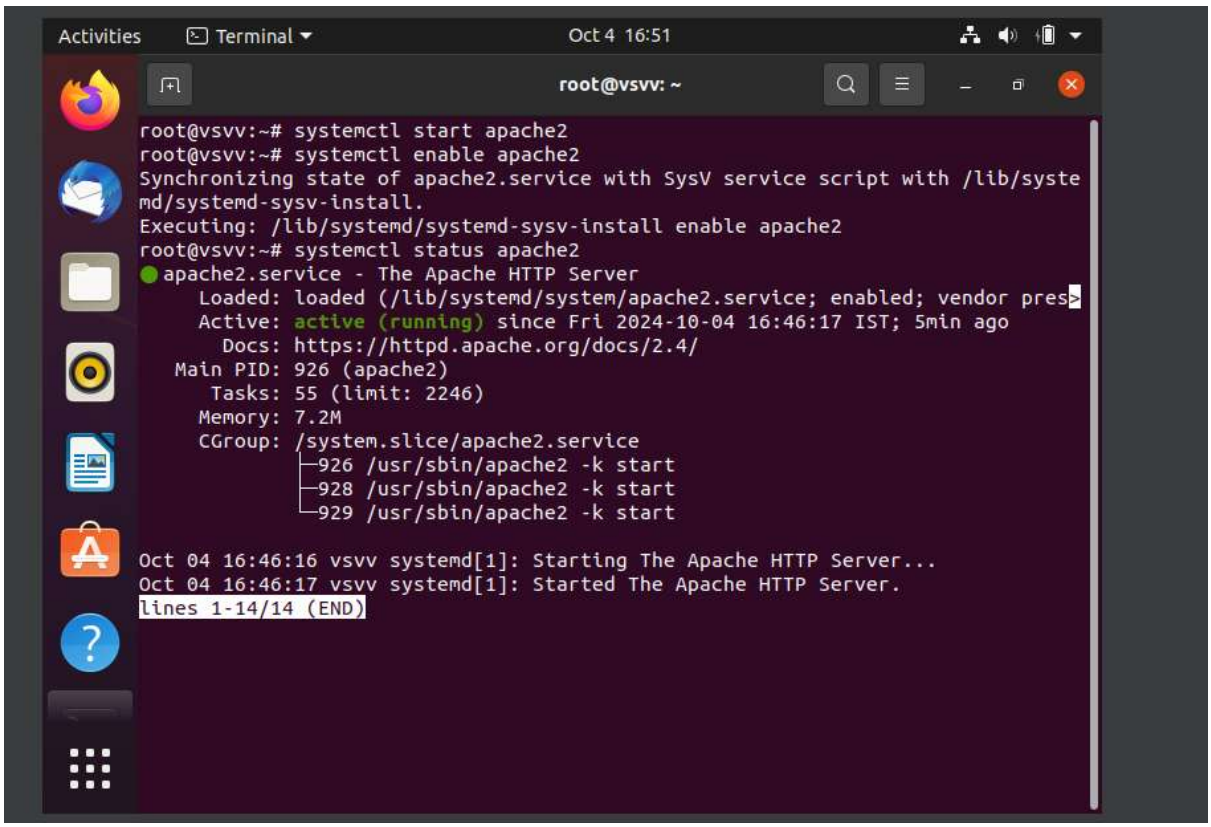
Update the system by using

apt-get update -y



Install apache2 in vm

apt install apache2 -y

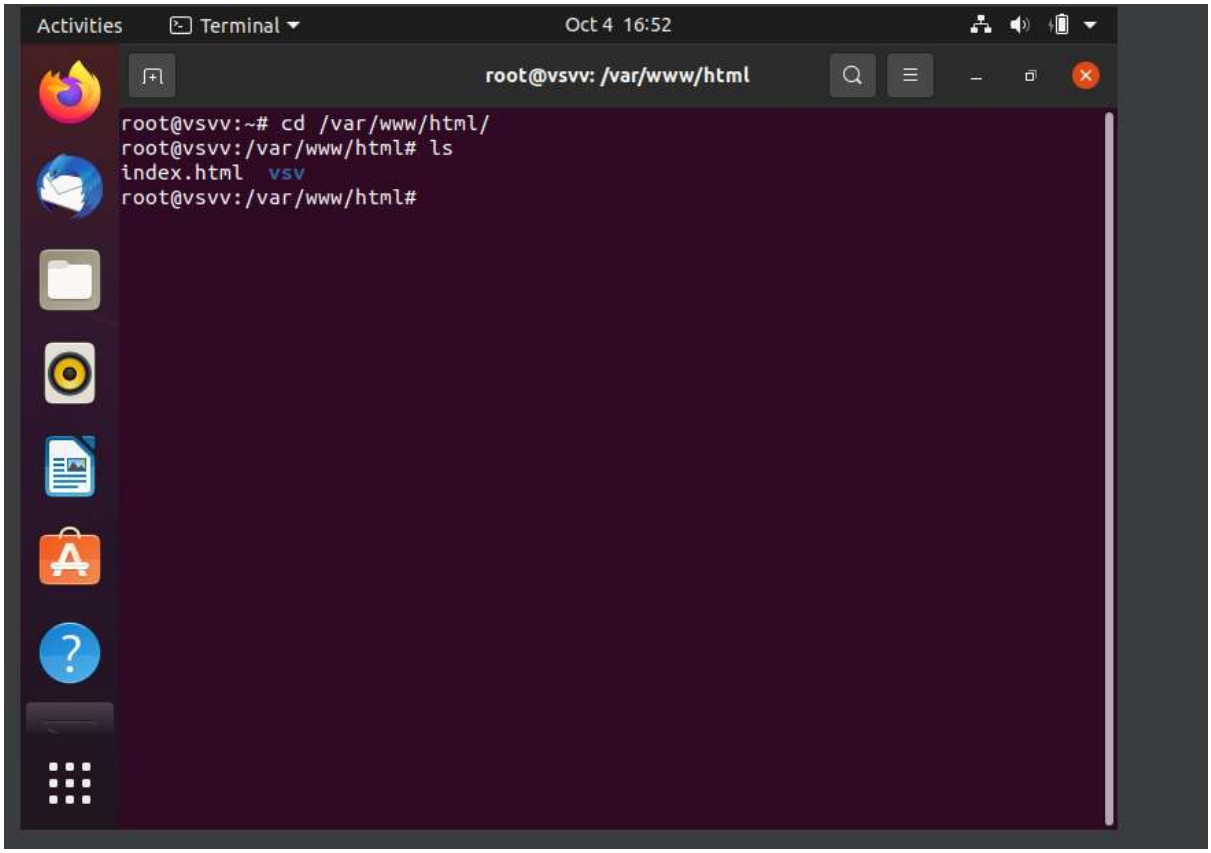


Start and enable the server

systemctl start apache2

systemctl enable apache2

systemctl status apache2



Go to apache default directory

cd /var/www/html

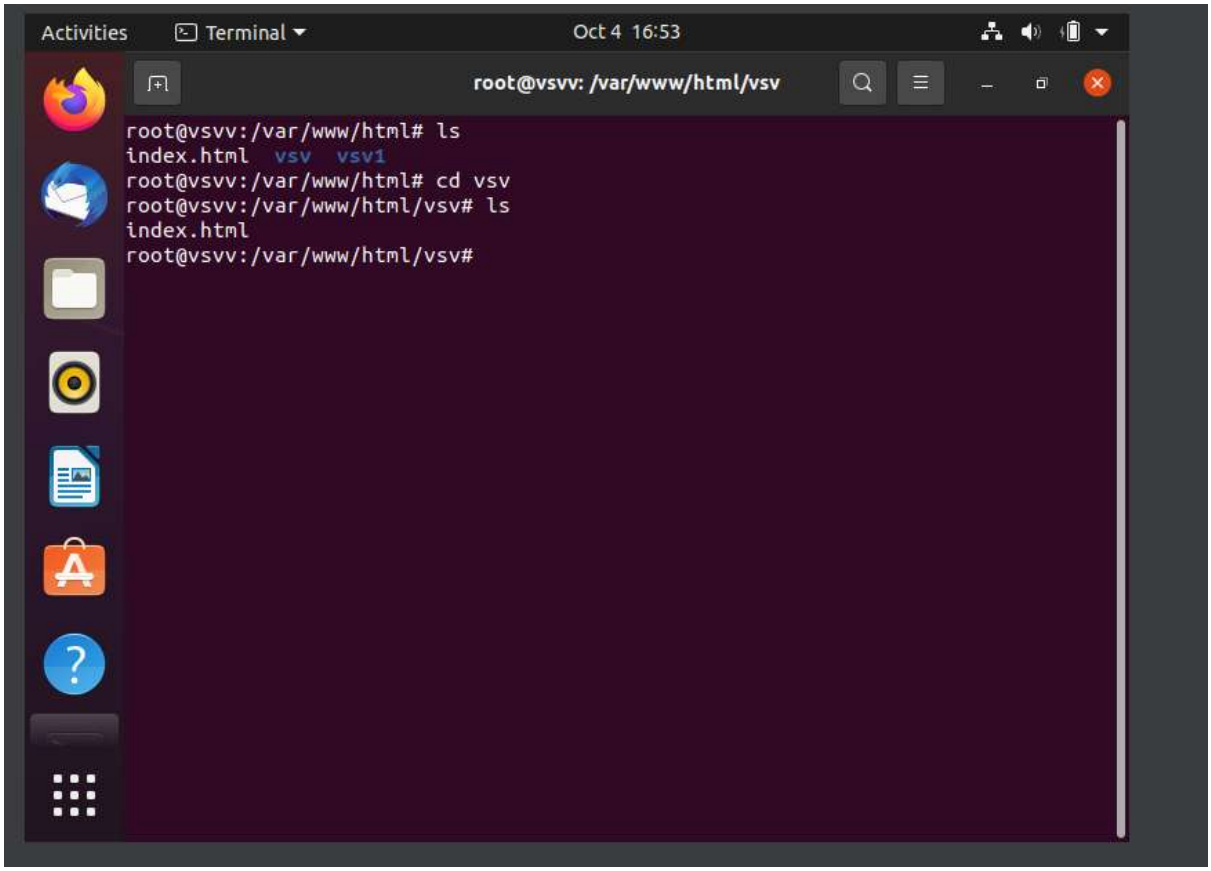
create new path

mkdir vsv

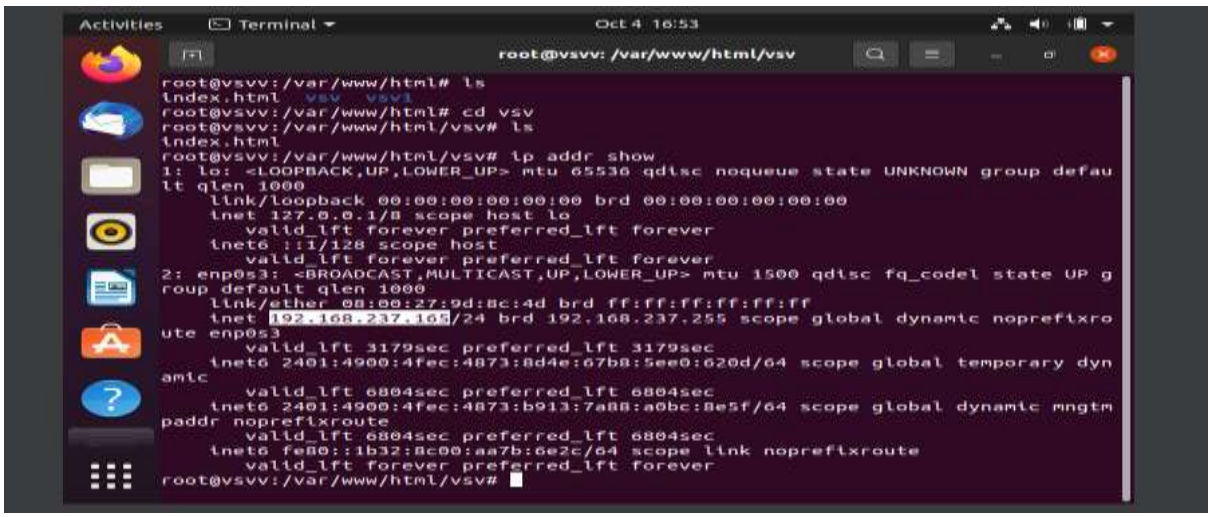
inside directory create new index file

vi index.html

---- ADD Sample Content --------



Here apache is completed



Ip addr show

This is your public ip of the vm

Check with ip you are getting response or not from web apache2

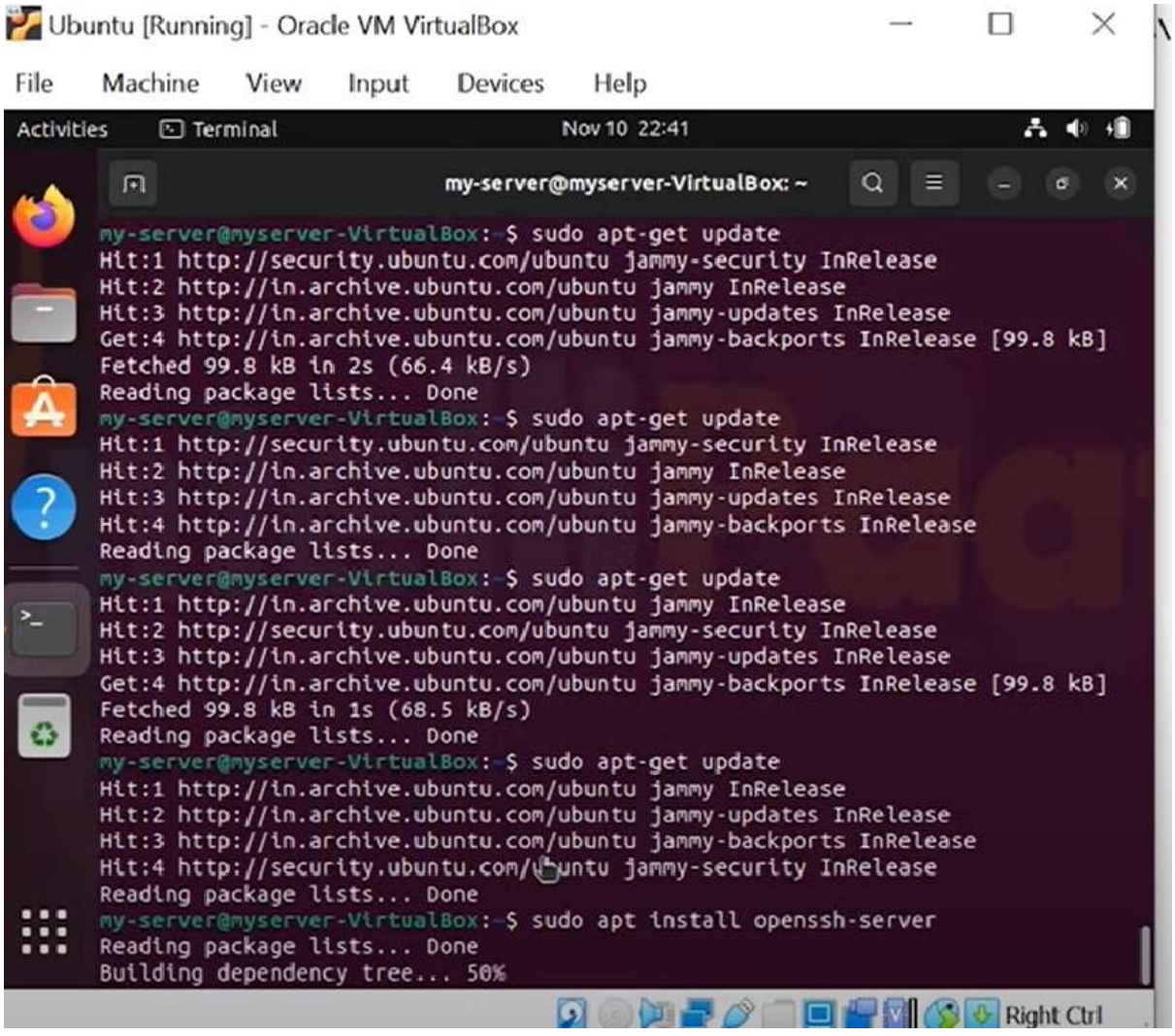
Very Important ***

Next install open ssh server for port enable to connect after migrate

$sudo apt update

$sudo apt –get update

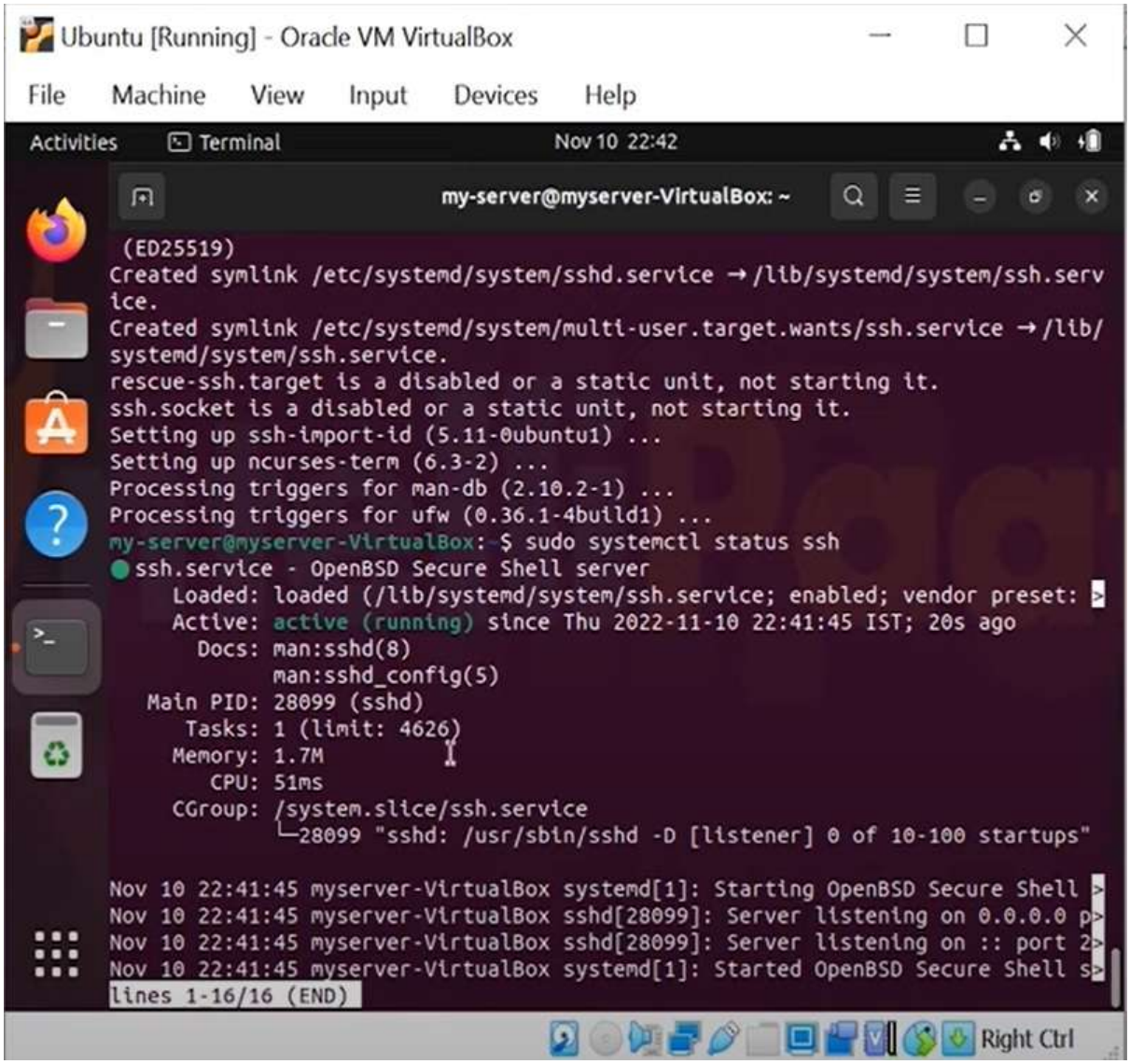
$sudo apt install openssh –server



sudo systemctl status ssh

sudo systemctl status ssh

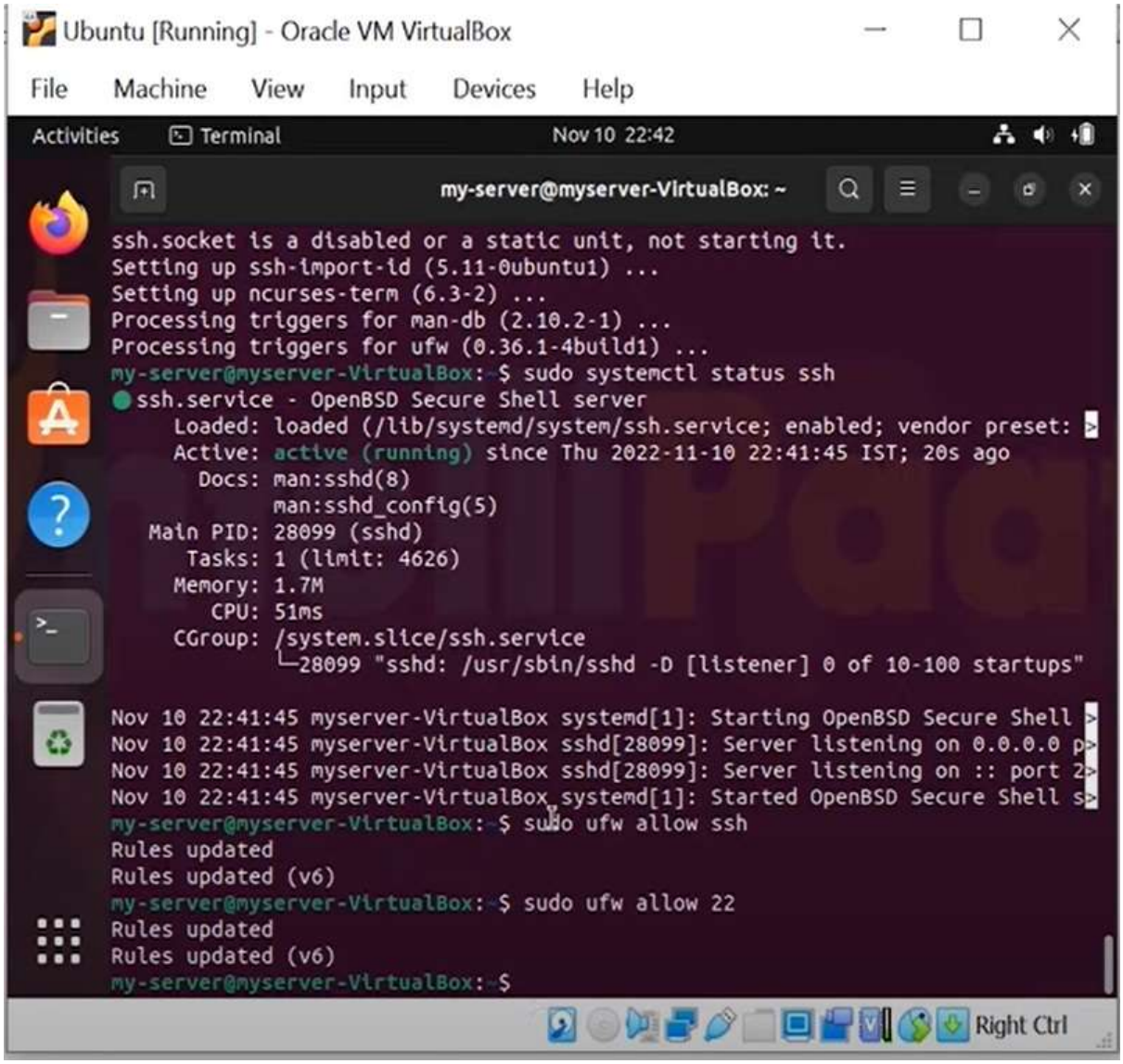
sudo systemctl start ssh

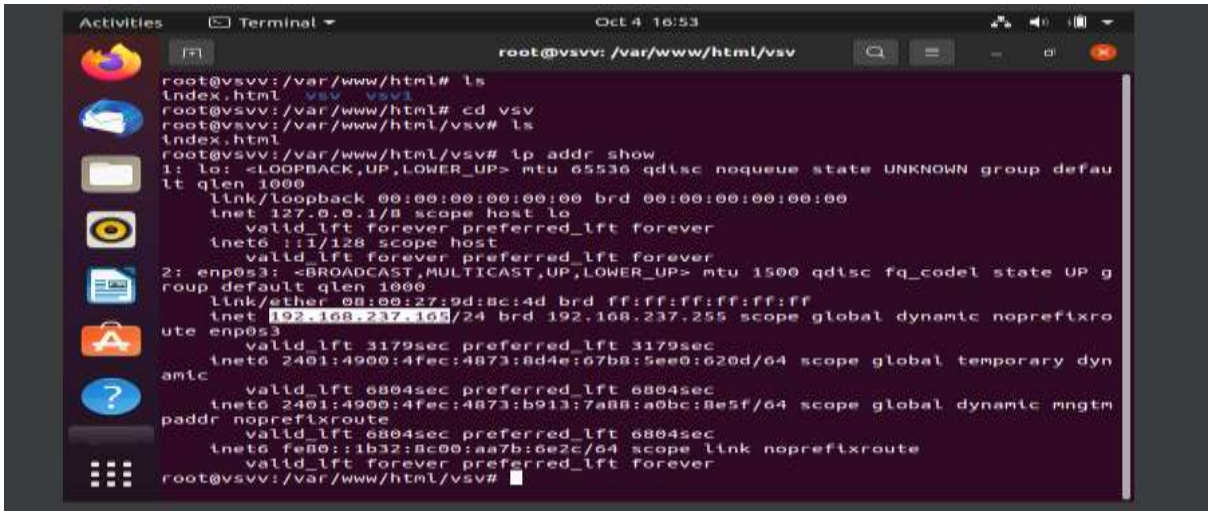


Now press ctrl +z

sudo ufw allow ssh

sudo ufw allow 22





After that try to connect your onprem vm into mobaxtreem

note

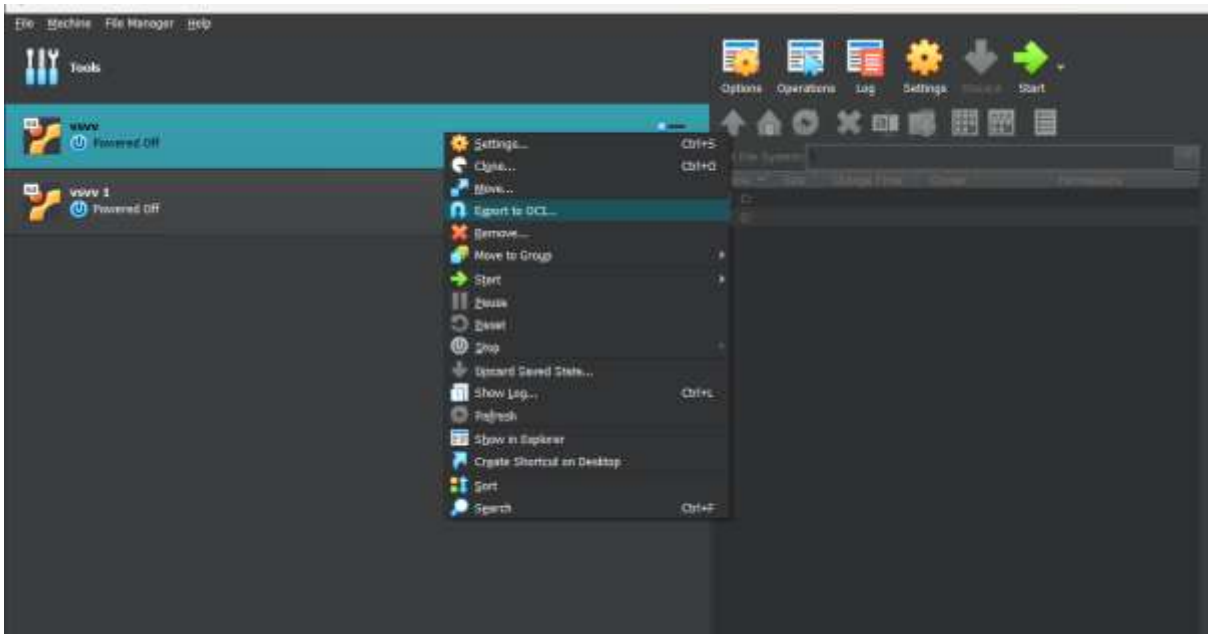
if u want to connect onprem vm we are able to connect from mobaxtreem or third party also

public ip

login as :vboxuser

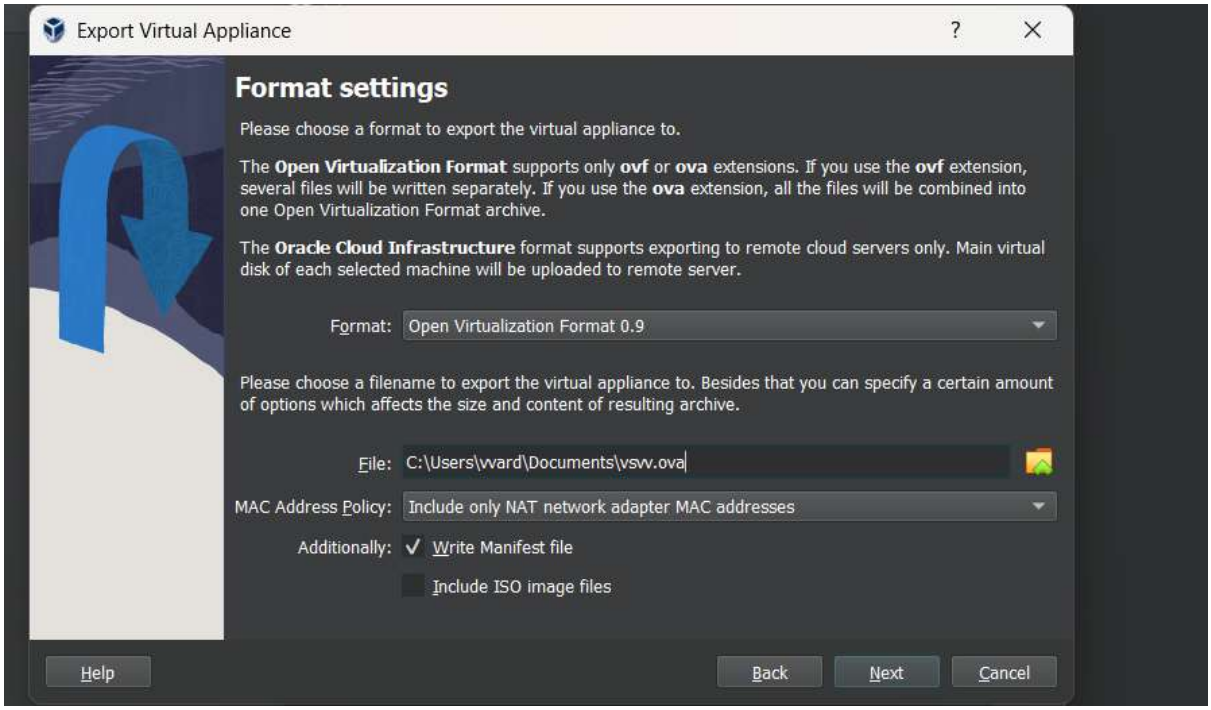
password:changeme or your custom onprem password

Next export vm to s3



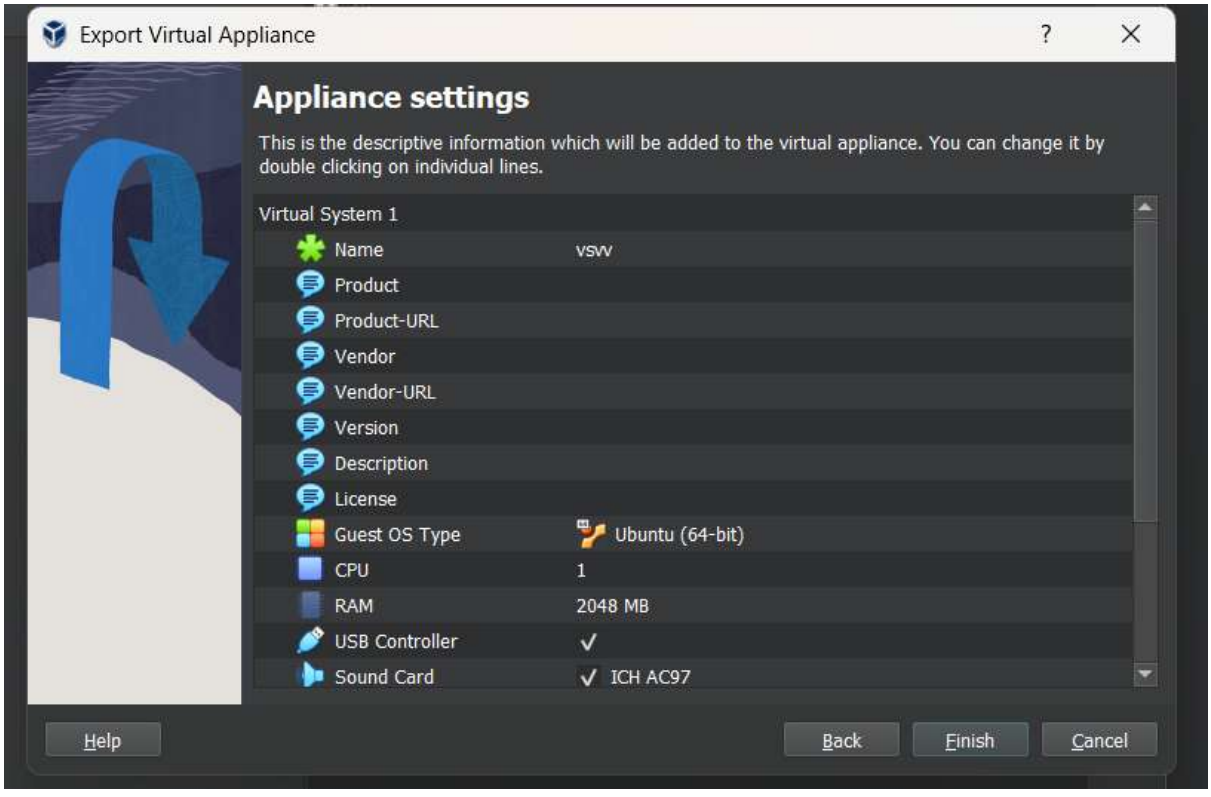
Right click on your vm

Click on export to oci



Select Window and Next

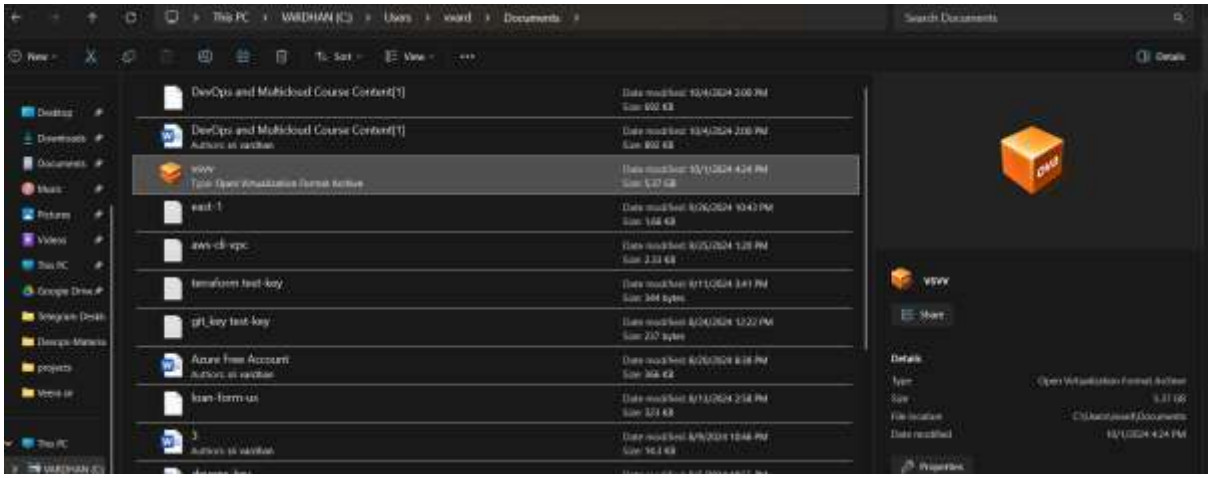
The above path this file is saved



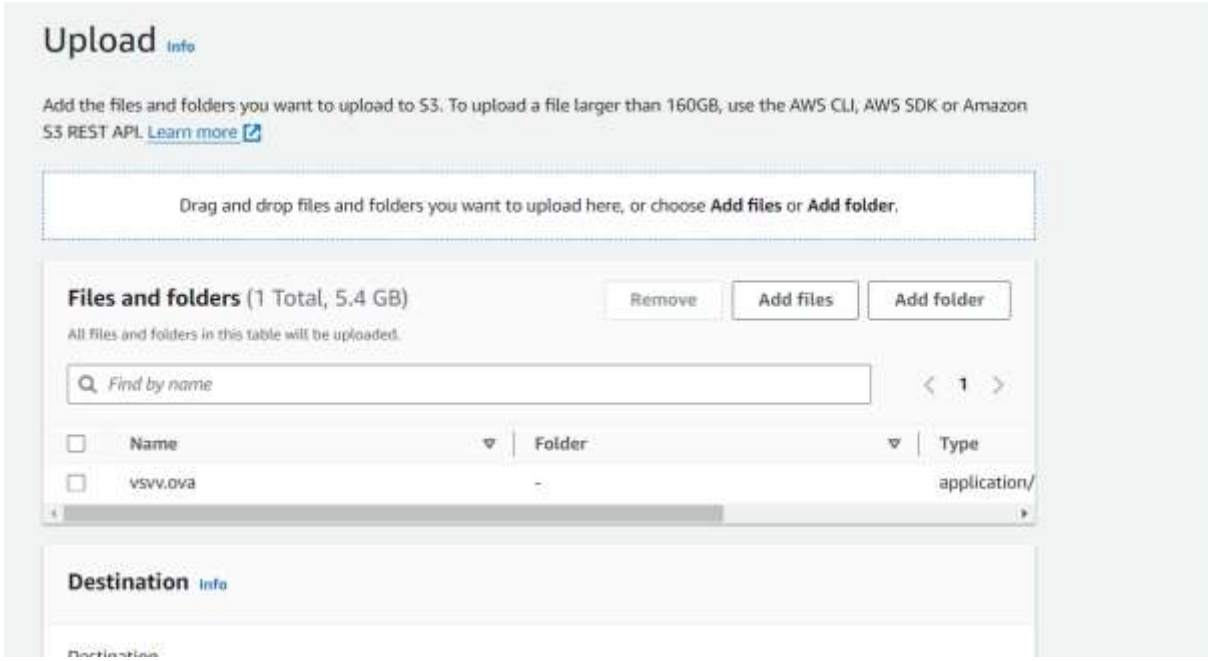
Click on finish

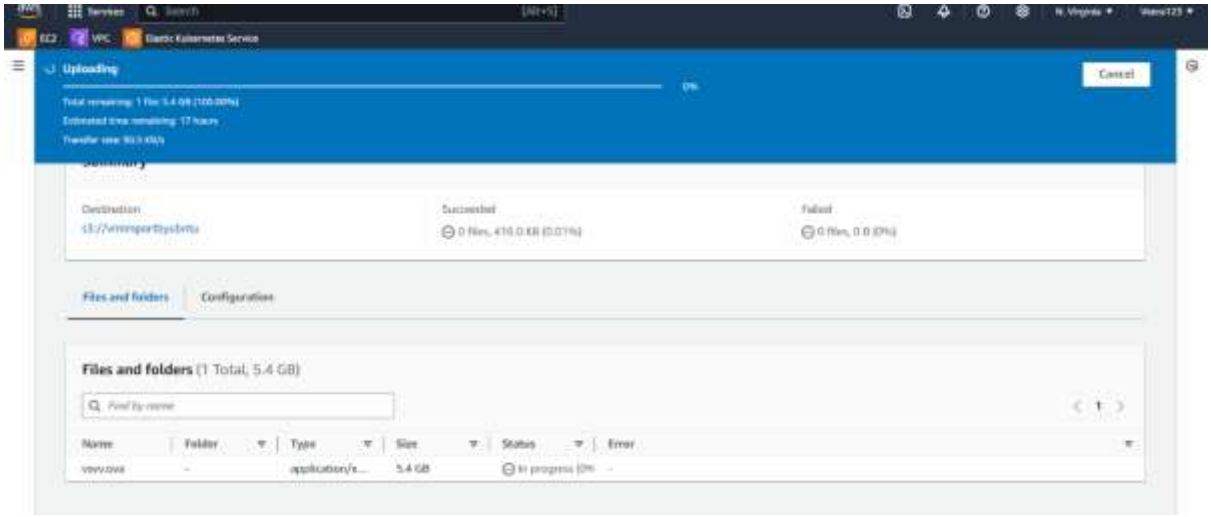
Go to the folder previously were created just check windows folder displayed over and save as type : .ova

After completing export applicance .Ova was created in our folder .we see image as below.



Open AWS console management and create s3 bucket and upload .ova file into your s3 bucket take nearly 40 mins .

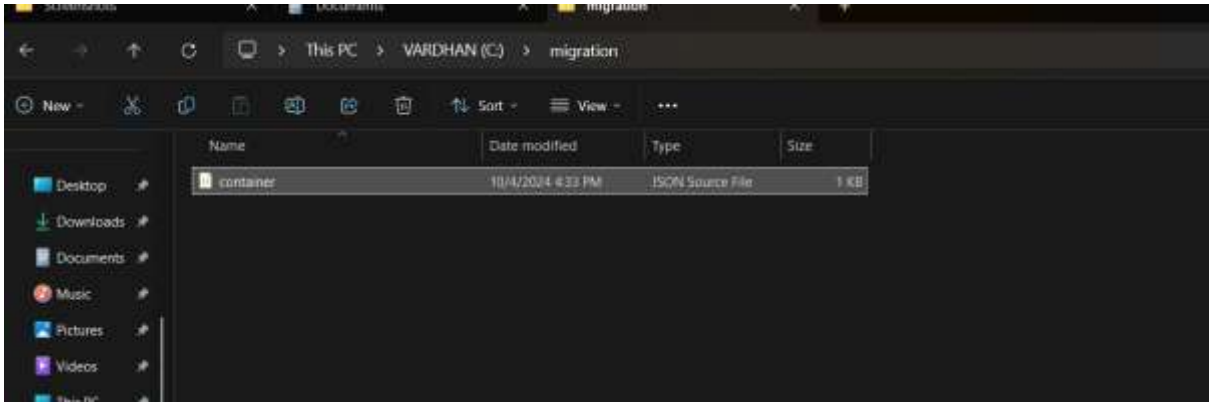




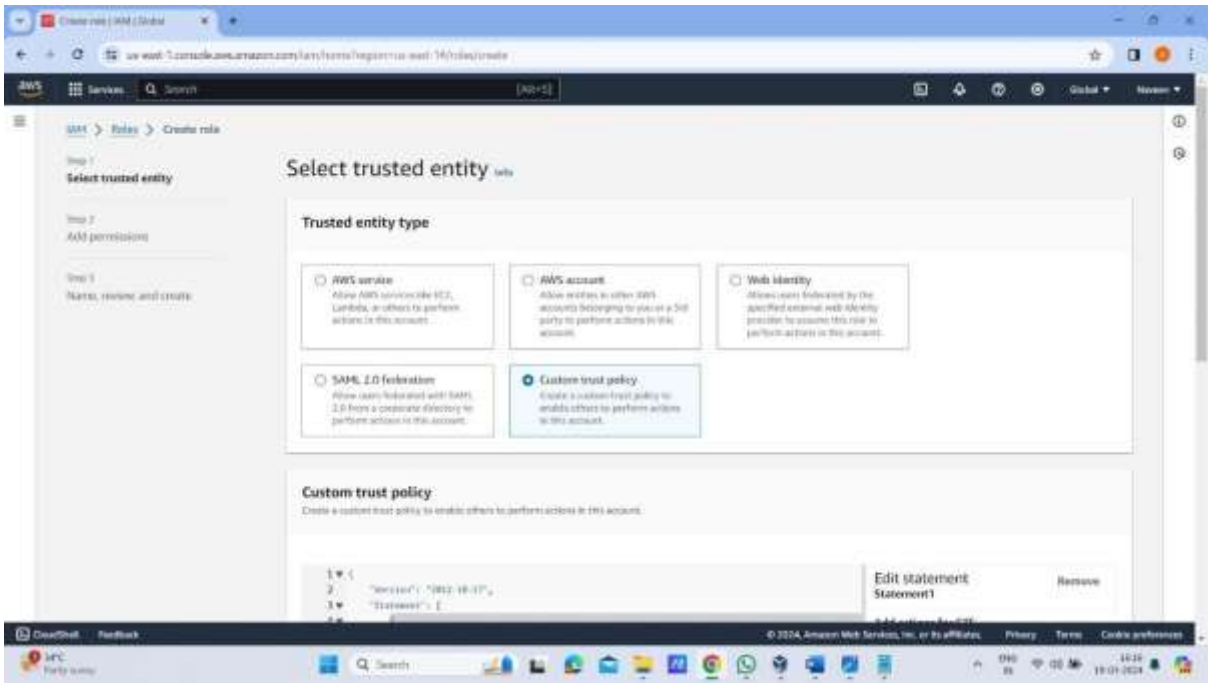
Meanwhile we can Create another folder (migration) in local laptop.

In that make template container.json file format by using vs code.

```
[
{
"Description":"ubuntu os",
"Format":"ova",
"UserBucket":
{
"S3Bucket": “vmimportbyubntu",
"S3Key": “server-1.ova”
}
}
]
```



After that go to console .Create IAM role select custom trust policy change principal as below mention:




```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Statement1",
      "Effect": "Allow",
      "Principal": {
        "Service": "vmie.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
```

```
    }
  ]
}
```

- After add permission EC2 , S3 full access to this role and save it .

Go Command prompt in local laptop.

- Configure aws using access and secret key.
- Run this command :

aws ec2 import-image --description "ubuntu" --disk-containers "file://C:\migration\container.json" --role ami-policy

aws ec2 describe-import-image-tasks –import-task-ids “(copy above image import-ami-id)”

aws ec2 describe-import-image-tasks --import-task-ids " import-ami-0a958649a4d59db16"

```
C:\Users\vvard>aws ec2 describe-import-image-tasks --import-task-ids "import-ami-019c07b289084f2af"
{
    "ImportImageTasks": [
        {
            "Description": "linux",
            "ImportTaskId": "import-ami-019c07b289084f2af",
            "Progress": "19",
            "SnapshotDetails": [
                {
                    "DiskImageSize": 6571246592.0,
                    "Format": "VMDK",
                    "Status": "active",
                    "UserBucket": {
                        "S3Bucket": "vmimportbyubntu",
                        "S3Key": "server-1.ova"
                    }
                }
            ],
            "Status": "active",
            "StatusMessage": "converting",
            "Tags": []
        }
    ]
}
```

aws ec2 describe-import-image-tasks --import-task-ids "import-ami-019c07b289084f2af"

- converting

```
C:\Users\vvard>aws ec2 describe-import-image-tasks --import-task-ids "import-ami-019c07b289084f2af"
{
    "ImportImageTasks": [
        {
            "Description": "linux",
            "ImportTaskId": "import-ami-019c07b289084f2af",
            "Progress": "20",
            "SnapshotDetails": [
                {
                    "DiskImageSize": 6571246592.0,
                    "Format": "VMDK",
                    "Status": "completed",
                    "UserBucket": {
                        "S3Bucket": "vmimportbyubntu",
                        "S3Key": "server-1.ova"
                    }
                }
            ],
            "Status": "active",
            "StatusMessage": "updating",
            "Tags": []
        }
    ]
}
```

aws ec2 describe-import-image-tasks --import-task-ids "import-ami-019c07b289084f2af"

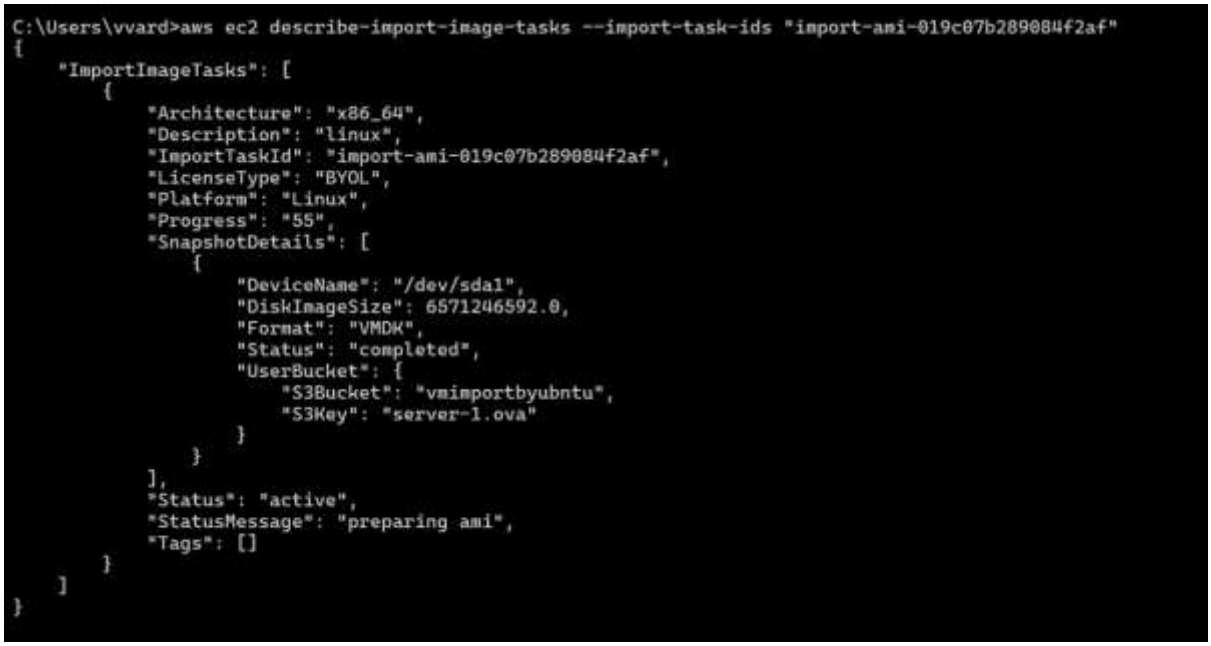
- updating

aws ec2 describe-import-image-tasks --import-task-ids "import-ami-019c07b289084f2af"

- booting

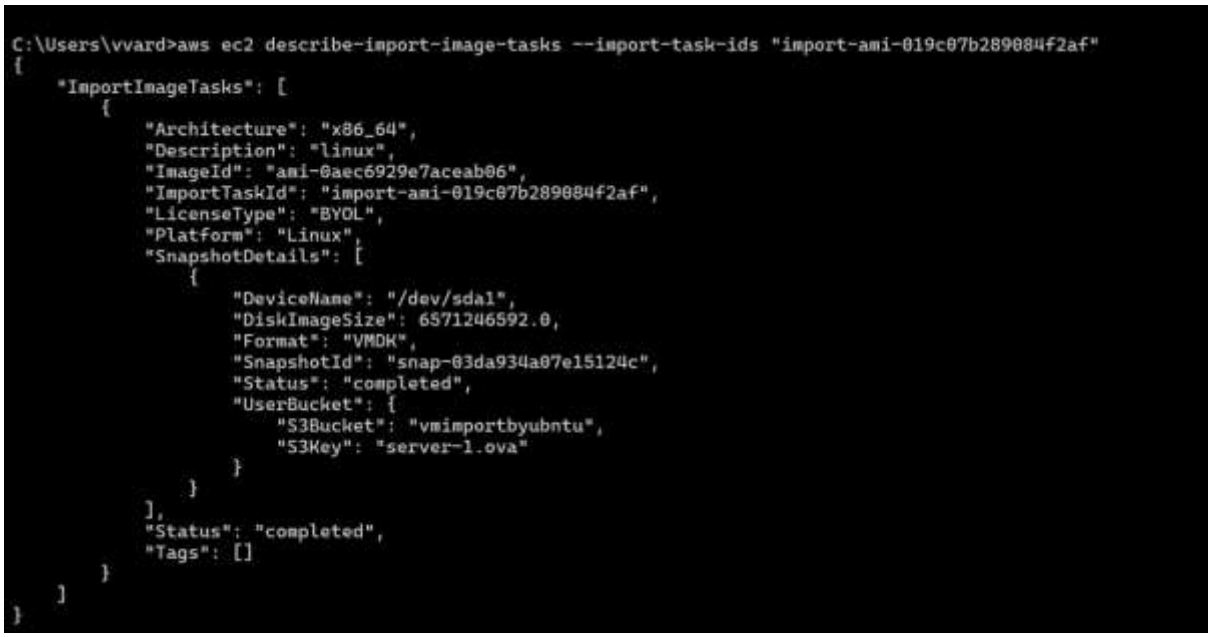
aws ec2 describe-import-image-tasks --import-task-ids "import-ami-019c07b289084f2af"

- booted



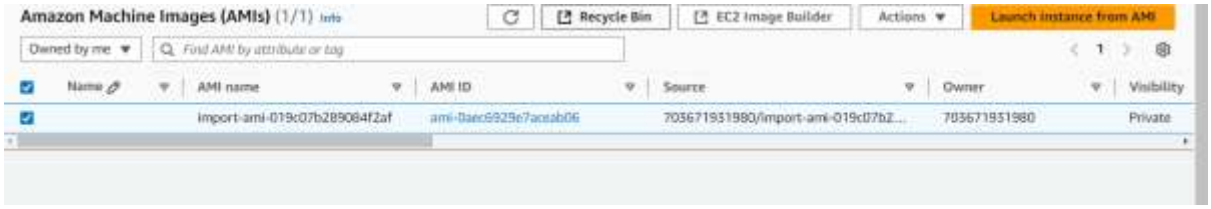
aws ec2 describe-import-image-tasks --import-task-ids "import-ami-019c07b289084f2af"

- preparing ami



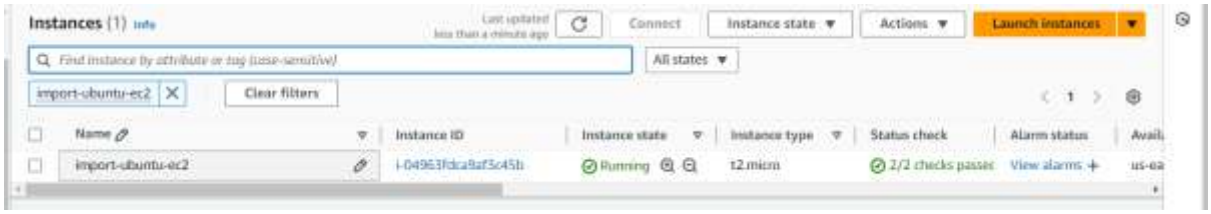
aws ec2 describe-import-image-tasks --import-task-ids "import-ami-019c07b289084f2af"

- completed
- After completed above process just go and check aws console over ec2 ami there you find imported image created by you.
- By using ami image launch ec2 instance.
- After launching instance take public ip

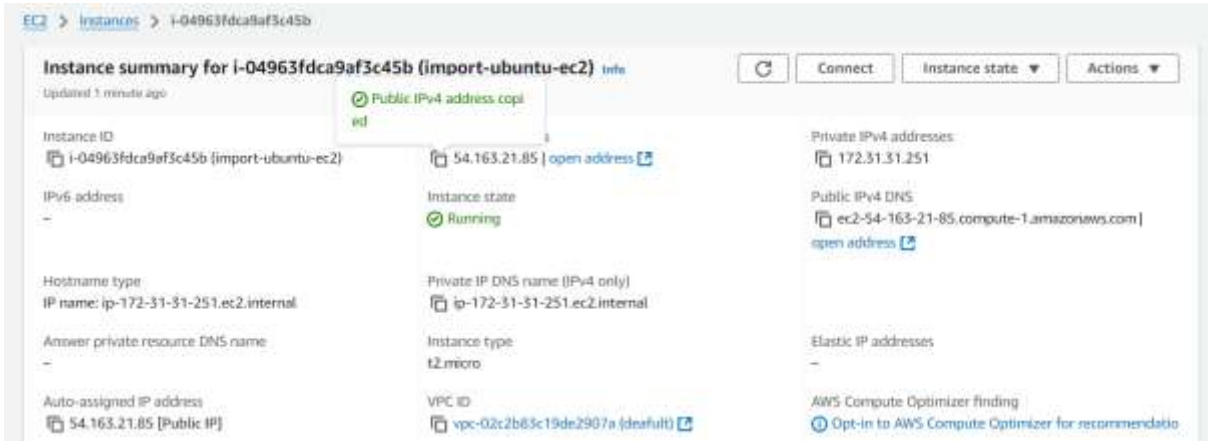


Above one is your ec2-

Select your ami and launch the e2



Open your ec2



The the public ip and search In web

Enetr your path after the public ip

iiiii aws migration project

Yeah we are successfully migrated on premises vm into aws cloud

###################### Before Migrating checks######################

Before migrating your **Ubuntu Linux VM** into AWS, it's essential to configure the VM to ensure that you can connect and access your applications after the migration. Here's what you need to install and configure to make the migration smooth and ensure post-migration connectivity and application access.

1. Enable SSH for Remote Access

Ensure **SSH** (Secure Shell) is installed, enabled, and configured to allow access to your Ubuntu VM once it is migrated to AWS.

- **Install and enable SSH**: If SSH is not installed, you can install it by running:

sudo apt update

sudo apt install openssh-server

- **Ensure SSH starts on boot**:

sudo systemctl enable ssh

sudo systemctl start ssh

- **Configure firewall (if applicable)**: If you have ufw (Uncomplicated Firewall) enabled, ensure port **22** (the default SSH port) is allowed:

sudo ufw allow ssh

sudo ufw enable

sudo ufw status

2. Install Cloud-Init

Cloud-init is a widely used package on Ubuntu for initializing cloud instances. AWS relies on it to perform instance setup tasks (such as network configuration, SSH key injection, etc.) after launching the instance.

- **Install cloud-init**:

sudo apt update

sudo apt install cloud-init

- Ensure cloud-init is enabled and correctly configured for your environment. This helps with EC2 instance setup upon launch (setting up SSH access, hostname, and networking).

3. Install Necessary Network Drivers and Dependencies

AWS instances use specific networking drivers (e.g., **ENA** for enhanced networking) and other storage drivers. However, modern Linux distributions like Ubuntu come with these drivers built-in.

sudo apt update && sudo apt upgrade -y

4. Ensure the Application and Services Start on Boot

After the migration, you'll want to ensure that your application starts automatically when the VM boots up on AWS EC2.

- **Configure your application to start on boot** using systemd. For example, if your application is a web server, you would ensure services like Apache or Nginx start automatically:

sudo systemctl enable apache2 # For Apache

add application into path   /var/www/html

sudo systemctl start apache2

note

if u want to connect ec2 we are able to connect from mobaxtreem or third party only

public ip

login as :vboxuser

password:changeme or your custom onprem password