`Vagrant is a tool that that is used to automate the creation of virtual machine.`

`Almost all interaction with Vagrant is done through the command-line interface.`

To create the virtual machine automatically the vagrant used Vargentfile as configuration file for virtual machine.

we can use vagrant with multiple providers

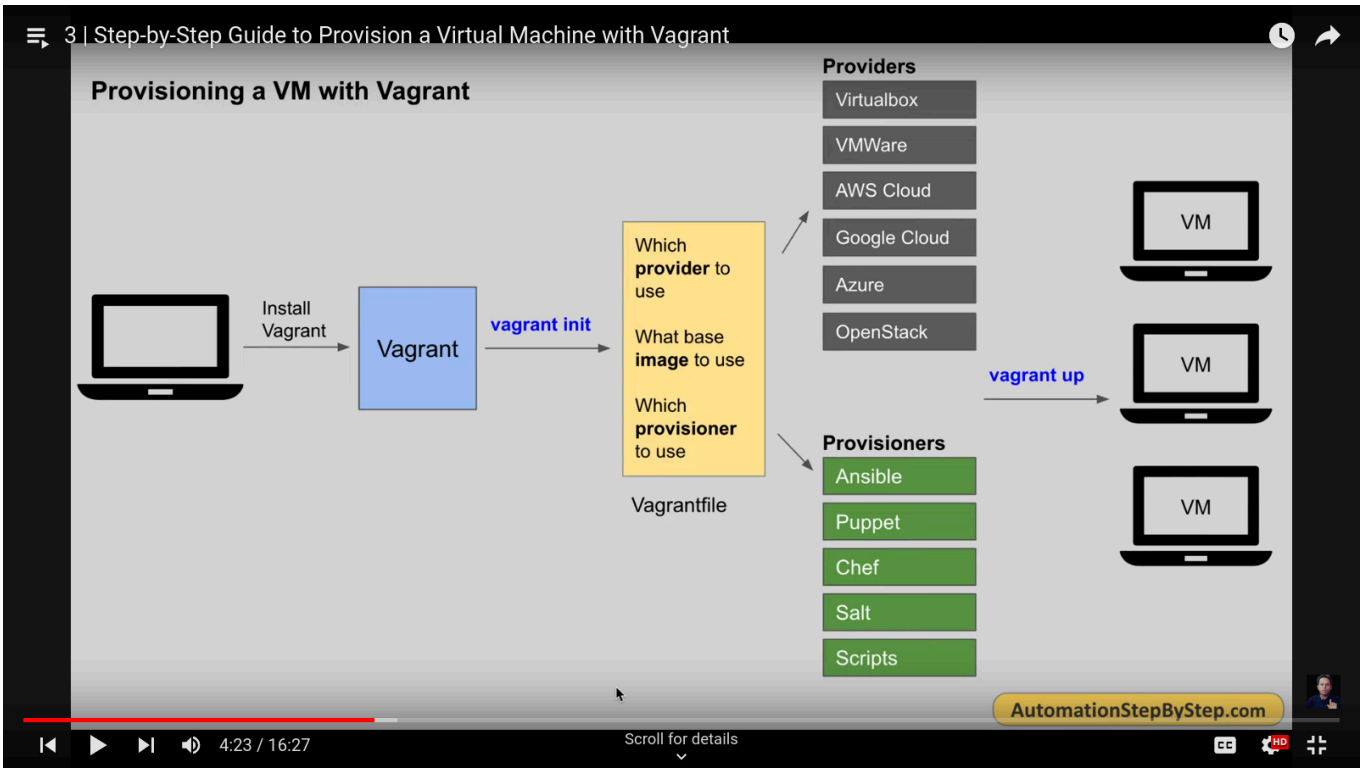
1. virtual-box
2. KVM/livbirt
3. Hyper-V

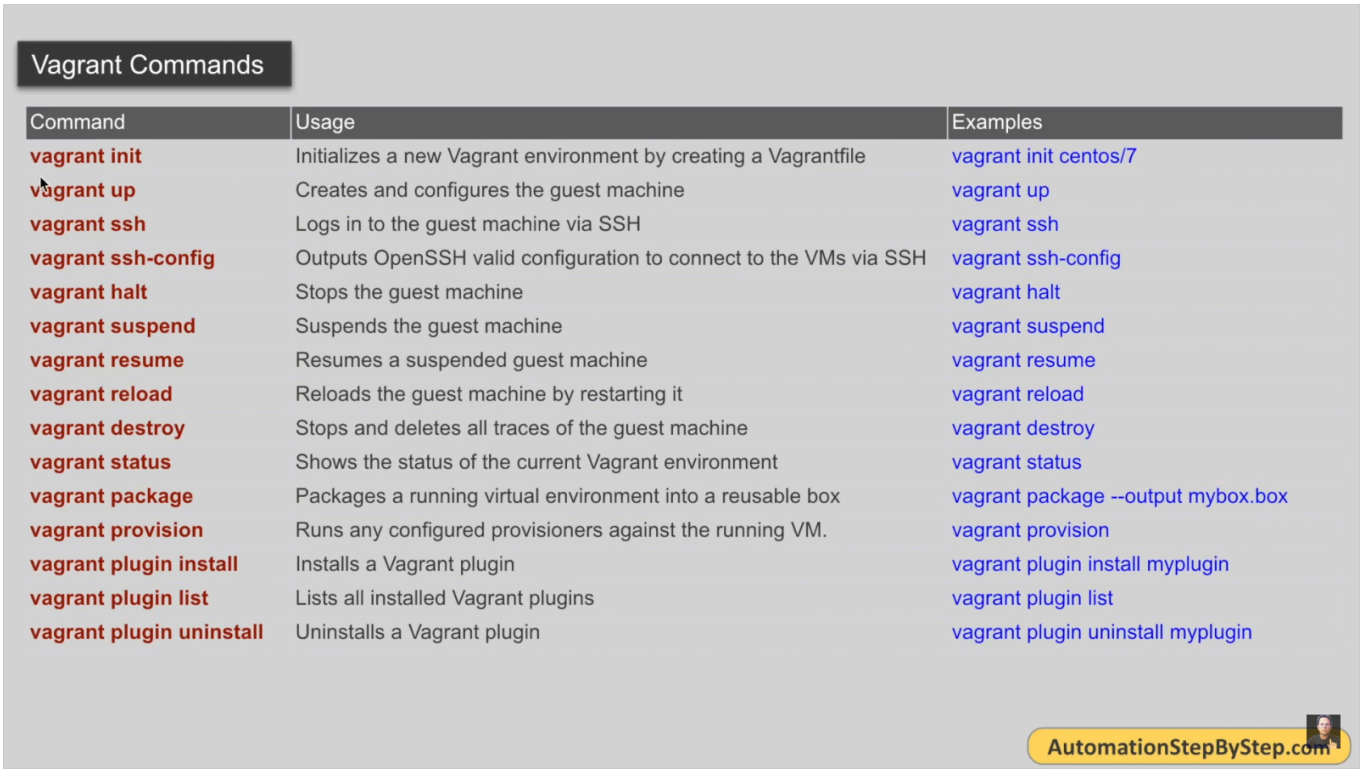
vagrant installation
https://developer.hashicorp.com/vagrant/downloads

Quick steps

```
#quick steps for vagrant
apt install vagrant # install vagrant after configuirng repos by default it provides lower version
vagrant --version
mkdir vagarnt_projects
cd vagrant_projects
#create woking envirment (suppose we are provisioning ubuntu vm)
mkdir ubuntu_vm && cd ubuntu_vm
vagrant init # it will create vagant file
vim Vagrantfile # edit the baseimage box entry with box name for Eg:-(generic/ubuntu2204)
# save and exit the file
vagrant up --provider virtualbox # Select the provider that you are using
vagrant ssh # Take ssh of the provisioned machine
```

To stop the vm gracefully `vagrant halt`
To restart the vm `vagrant reload`
`vagrant destroy` stops and deletes all traces of the vagrant machine





Vagrant Commands

Command	Usage	Examples
vagrant init	Initializes a new Vagrant environment by creating a Vagrantfile	vagrant init centos/7
vagrant up	Creates and configures the guest machine	vagrant up
vagrant ssh	Logs in to the guest machine via SSH	vagrant ssh
vagrant ssh-config	Outputs OpenSSH valid configuration to connect to the VMs via SSH	vagrant ssh-config
vagrant halt	Stops the guest machine	vagrant halt
vagrant suspend	Suspends the guest machine	vagrant suspend
vagrant resume	Resumes a suspended guest machine	vagrant resume
vagrant reload	Reloads the guest machine by restarting it	vagrant reload
vagrant destroy	Stops and deletes all traces of the guest machine	vagrant destroy
vagrant status	Shows the status of the current Vagrant environment	vagrant status
vagrant package	Packages a running virtual environment into a reusable box	vagrant package --output mybox.box
vagrant provision	Runs any configured provisioners against the running VM.	vagrant provision
vagrant plugin install	Installs a Vagrant plugin	vagrant plugin install myplugin
vagrant plugin list	Lists all installed Vagrant plugins	vagrant plugin list
vagrant plugin uninstall	Uninstalls a Vagrant plugin	vagrant plugin uninstall myplugin

vagrant box

it is similar as docker image functionality . you can download any image by visiting vagrant boxes . many pre-configured image are available already

```
#To add new image in system
vagrant box add image_name
#To remove vagrant box
vagrant box remove
#To list existing boxes
vagrant box list
#check if any boxes is outdated
```

What is 'Provision'

`It means process of setting up and configuring a virtual machine with the necessary software and resources needed for particular task or application`

Steps to create new vagrant vm

1. create a new directory
2. `vagrant init` it will initalize vagrant file
3. edit the vagrant file according to your need (add image name)
4. `vagrant up` this command will create and provision the vm

`vagrant ssh` To get ssh of newly created vm

Vagrant file

```
# Automatically Generated by Vagrant Config Generator, see
https://github.com/jianan1104/vagrantfile-generator
Vagrant.configure('2') do |config|
  (1..1).each do |i|
  config.vm.define "sandy" do |machine|
    machine.vm.box = 'generic/ubuntu2204'
    machine.vm.network "private_network", ip: 'd'
    machine.vm.hostname = "sandy"
    machine.vm.provider "virtualbox" do |vb|
      vb.name = "mywebserver-#{i}"
      vb.cpus = '2'
      vb.memory = '1024'
    end
  end
  end
end
```

increase memory

```
vb.memory=2048
```

Increase disk size

`vagrant plugin install vagrant-disksize`

```
config.disksize.size = '50GB'
```

setting host name

```
config.vm.hostname = "myhost.local"
```

Add user via shell script

```
# To add user via shell script
Add user
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/bionic64"  # Replace with your desired Ubuntu box

  # Provisioning script to create user 'sandeep' with password 'dfjksdjf'
  config.vm.provision "shell", inline: <<-SHELL
    # Add user 'sandeep' with password 'dfjksdjf'
    sudo useradd -m sandeep
    echo 'sandeep:dfjksdjf' | sudo chpasswd
  SHELL
end
```

Shard folder

`it will consume space only on host but it will reflect at both ( guest os and host os )` ```

```
config.vm.synced_folder ".", "/vagrant", disabled: true
```

Package existing running environment of vm

change the directory to vagrant environment that you wants to package
`vagrnat package` it will create a vagrant package.box
Now add this box to vagrant box storage
`vagrant box add package.box --name name_you_want`

vagrant network configuration

https://developer.hashicorp.com/vagrant/docs/networking/

port forwarding

If we want to access the services from the host OS then we have to forward the port

we can forward a specific port of our choice using vagrant file

```
config.vm.network "forwarded_port", guest: 80, host: 8080
```

set bridge network by dhcp

```
Vagrant.configure("2") do |config|
  config.vm.box = "your_box_name"

  # Configure a bridge network with DHCP
  config.vm.network "public_network", bridge: "your_network_interface",
type: "dhcp"

  # Other configurations...
end
```

bridge static ip

```
config.vm.network "public_network", bridge: "en0p1:", ip: "192.168.1.20"
```

private network by dhcp

```
Vagrant.configure("2") do |config|
  config.vm.network "private_network", type: "dhcp"
end
```

vagrant snapshot

Full copy of the vm in its current sate

use cases:-

1. full backup of the vm
2. Restore to previous version in need

if you want safe and consistent state of vm then it is recommended to shut down vm first then take snapshot.

```
vagrant snapshot save fresh.snapshot  # To take snapshot
vagrant snapshot list # List all available snapshots of vm
vagrant snapshot restore fresh.snapshot # Restore to previous snapshot
vagrant snapshot delete fresh.snapshot
```

vagrant plug-ins

`plug-ins for additional functionalties`

`vagrant plugin install vagrant-libvirt`

`vagrant up --provider=libvirt`

Create a separate file having Provision Scripts and provide the location in Vagrant file

`config.vm.provision :shell, path: "provision.sh`

kvm bridge setup

https://www.dzombak.com/blog/2024/02/Setting-up-KVM-virtual-machines-using-a-bridged-network.html