Vagrant - Provisioning

In the context of **Vagrant**, **provisioning** refers to the automated process of setting up the software, configuration, and environment inside a newly created virtual machine (VM). It's what makes your box useful beyond just "booting up" — it turns an empty OS into a ready-to-use development environment.

What is Provisioning in Vagrant?
Provisioning is the **post-boot process** that configures the VM after it's created from a base box. You use provisioning to:

- Install packages (e.g., Apache, MySQL, Python)
- Configure settings (e.g., timezone, user accounts)
- Run shell scripts or automation tools (e.g., Ansible, Chef, Puppet)

Vagrant Provisioning Methods
Vagrant supports several provisioning options:

Shell Scripts
Most common method
You provide a Bash (or PowerShell) script to install and configure software

Example in Vagrantfile:

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

Inline Shell
You can embed small shell commands directly in the Vagrantfile

```
config.vm.provision "shell", inline: "apt update && apt install -y nginx"
```

Puppet
Declarative language for system configuration

Vagrant can apply a Puppet manifest:

```
config.vm.provision "puppet" do |puppet|
  puppet.manifests_path = "manifests"
  puppet.manifest_file  = "site.pp"
end
```

Chef (Solo or Client)
Uses Chef cookbooks to automate setup

Example:

```
config.vm.provision "chef_solo" do |chef|
  chef.add_recipe "apache"
end
```

Ansible
Modern automation tool, often used with multiple servers

Vagrant supports Ansible directly:

```
config.vm.provision "ansible" do |ansible|
  ansible.playbook = "playbook.yml"
end
```

🚀 When Does Provisioning Happen?

Provisioning typically occurs during:
vagrant up (only on first VM creation)
vagrant provision (manual re-run)
vagrant reload --provision (reboot + reprovision)

📝 Example Workflow

```
# Start and provision
vagrant up

# Re-run provisioning manually
vagrant provision

# Restart and reprovision
vagrant reload --provision
```

1. Navigate to the VM Folder and Create the VM

```
@Ubuntu:~/Testdir/vagrant/myvirtualmachines$ ls
================================================================================
config  machine1  machine2  readme  scripts
================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines$ mkdir machine3

@Ubuntu:~/Testdir/vagrant/myvirtualmachines$ ls
================================================================================
config  machine1  machine2  machine3  readme  scripts
================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines$ cd machine3

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$
vagrant global-status
================================================================================
id       name    provider   state   directory
--------------------------------------------------------------------------------
--------------------------
8b0c7cc  default virtualbox running /home/tiago-paquete/Testdir/vagrant/
myvirtualmachines/machine1

The above shows information about all known Vagrant environments
on this machine. This data is cached and may not be completely
up-to-date (use "vagrant global-status --prune" to prune invalid
entries). To interact with any of the machines, you can go to that
directory and run Vagrant, or you can use the ID directly with
Vagrant commands from any directory. For example:
"vagrant destroy 1a2b3c4d"
================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant status
================================================================================
A Vagrant environment or target machine is required to run this
command. Run `vagrant init` to create a new Vagrant environment. Or,
get an ID of a target machine from `vagrant global-status` to run
this command on. A final option is to change to a directory with a
Vagrantfile and to try again.
================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant init
ubuntu/jammy64
================================================================================
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
================================================================================



@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant status
```

```
==============================================================================
Current machine states:

default                   not created (virtualbox)

The environment has not yet been created. Run `vagrant up` to
create the environment. If a machine is not created, only the
default provider will be shown. So if a provider is not listed,
then the machine is not created for that environment.
==============================================================================
```

2. Add Initial Provisioning Block to the Vagrantfile

```
@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ cat Vagrantfile
================================================================================
Vagrant.configure("2") do |config|
  # Set the base box to Ubuntu 22.04 (Jammy Jellyfish) 64-bit
  config.vm.box = "ubuntu/jammy64"

  # Configure a private network with a static IP address
  config.vm.network "private_network", ip: "192.168.56.10"

  # Configure a public (bridged) network using the specified network
interface
  config.vm.network "public_network", bridge: "wlp0s20f3"

  # Set VirtualBox provider-specific options
  config.vm.provider "virtualbox" do |vb|
    # Allocate 1024 MB of RAM to the VM
    vb.memory = 1024
    # Allocate 2 CPU cores to the VM
    vb.cpus = 2
  end

  # Sync the local ./scripts folder to /opt/scripts in the VM
  config.vm.synced_folder "./scripts", "/opt/scripts"
end
================================================================================
```

Edit your Vagrantfile:
nano Vagrantfile

Insert the following **inside your existing configuration block**, just before the final end:

```
  # Initial provisioning: update system and install Apache2
  config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
    systemctl enable apache2
    systemctl start apache2
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/html/
index.html
  SHELL
```

Save and exit.

```
@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ cat Vagrantfile
================================================================================
Vagrant.configure("2") do |config|
  # Set the base box to Ubuntu 22.04 (Jammy Jellyfish) 64-bit
  config.vm.box = "ubuntu/jammy64"

  # Configure a private network with a static IP address
  config.vm.network "private_network", ip: "192.168.56.10"

  # Configure a public (bridged) network using the specified network
interface
  config.vm.network "public_network", bridge: "wlp0s20f3"

  # Set VirtualBox provider-specific options
  config.vm.provider "virtualbox" do |vb|
    # Allocate 1024 MB of RAM to the VM
    vb.memory = 1024
    # Allocate 2 CPU cores to the VM
    vb.cpus = 2
  end

  # Sync the local ./scripts folder to /opt/scripts in the VM
  config.vm.synced_folder "./scripts", "/opt/scripts"

 config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
    systemctl enable apache2
    systemctl start apache2
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/
html/index.html
  SHELL

end
================================================================================
```

3. Start and Provision the VM

Run:
vagrant up

This will:
- Create and boot the VM
- Apply your base config
- Run the provisioning script to install and start Apache

```
@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant up
==============================================================================
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'ubuntu/jammy64'...

...

==============================================================================
```

4. Test Apache

Open a browser and go to:
http://192.168.56.10

You should see the message:
"Hello from Provisioned Apache on Ubuntu Jammy!”

5. Add More Provisioning Steps (Logging System Info)

Update your provisioning block to append system resource details:

```
  config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
    systemctl enable apache2
    systemctl start apache2
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/
html/index.html

    # Create a logs directory and capture system info
    mkdir -p /opt/provision-logs
    free -m > /opt/provision-logs/memory.txt
    df -h > /opt/provision-logs/disk.txt
    uname -a > /opt/provision-logs/uname.txt
  SHELL
```

Save, then run:
```
vagrant provision
```

Then:
```
vagrant ssh
```

```
cat /opt/provision-logs/memory.txt
cat /opt/provision-logs/disk.txt
cat /opt/provision-logs/uname.txt
exit
```

```
@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ cat Vagrantfile
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|
  # Set the base box to Ubuntu 22.04 (Jammy Jellyfish) 64-bit
  config.vm.box = "ubuntu/jammy64"

  # Configure a private network with a static IP address
  config.vm.network "private_network", ip: "192.168.56.10"

  # Configure a public (bridged) network using the specified network
interface
  config.vm.network "public_network", bridge: "wlp0s20f3"

  # Set VirtualBox provider-specific options
  config.vm.provider "virtualbox" do |vb|
    # Allocate 1024 MB of RAM to the VM
    vb.memory = 1024
```

```
    # Allocate 2 CPU cores to the VM
    vb.cpus = 2
  end

  # Sync the local ./scripts folder to /opt/scripts in the VM
  config.vm.synced_folder "./scripts", "/opt/scripts"

 config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
    systemctl enable apache2
    systemctl start apache2
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/
html/index.html
  SHELL

  config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
    systemctl enable apache2
    systemctl start apache2
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/
html/index.html

    # Create a logs directory and capture system info
    mkdir -p /opt/provision-logs
    free -m > /opt/provision-logs/memory.txt
    df -h > /opt/provision-logs/disk.txt
    uname -a > /opt/provision-logs/uname.txt
  SHELL



end
================================================================================


@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant provision
================================================================================
==> default: Running provisioner: shell...
    default: Running: inline script

…

================================================================================


@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant ssh
================================================================================
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-140-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Thu Jun  5 10:07:29 UTC 2025
```

```
  System load:             0.05
  Usage of /:              4.4% of 38.70GB
  Memory usage:            24%
  Swap usage:              0%
  Processes:               110
  Users logged in:         0
  IPv4 address for enp0s3: 10.0.2.15
  IPv4 address for enp0s9: 172.20.10.3
  IPv6 address for enp0s9: 2a00:20:8:4862:a00:27ff:fe84:b0a9


Expanded Security Maintenance for Applications is not enabled.

31 updates can be applied immediately.
25 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

New release '24.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
================================================================================


vagrant@ubuntu-jammy:~$ ls -la /opt/provision-logs
================================================================================
total 20
drwxr-xr-x 2 root root 4096 Jun  5 10:06 .
drwxr-xr-x 4 root root 4096 Jun  5 10:06 ..
-rw-r--r-- 1 root root  378 Jun  5 10:06 disk.txt
-rw-r--r-- 1 root root  207 Jun  5 10:06 memory.txt
-rw-r--r-- 1 root root  114 Jun  5 10:06 uname.txt
================================================================================

vagrant@ubuntu-jammy:~$ cat /opt/provision-logs/memory.txt
================================================================================
               total        used        free      shared  buff/cache
available
Mem:             957         180         178           1         598
620
Swap:              0           0           0
================================================================================
```

Explanation:

- Mem: line shows approx **957 MB of RAM total**, consistent with your vb.memory = 1024 setting (some reserved for the system).
- used: 180 MB currently in use.
- free: 178 MB completely unused.
- buff/cache: 598 MB used by kernel buffers and cache (reclaimable).
- available: 620 MB realistically usable without swapping.
- Swap: is 0, because **no swap file or partition** is configured in the VM.

```
vagrant@ubuntu-jammy:~$ cat /opt/provision-logs/disk.txt
========================================================================
Filesystem      Size  Used Avail Use% Mounted on
tmpfs            96M  1.0M   95M   2% /run
/dev/sda1        39G  1.7G   37G   5% /
tmpfs           479M     0  479M   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
vagrant          53G   36G   18G  68% /vagrant
opt_scripts      53G   36G   18G  68% /opt/scripts
tmpfs            96M  4.0K   96M   1% /run/user/1000
========================================================================
```

Explanation:

```
/dev/sda1       39G  1.7G   37G   5% /              # Main disk
...
vagrant         53G   36G   18G  68% /vagrant       # Synced folder from host
opt_scripts     53G   36G   18G  68% /opt/scripts   # Also synced folder
```

Key points:

- /dev/sda1 is your VM's root disk: 39 GB total, 1.7 GB used.
- /vagrant and /opt/scripts are shared folders from your host system (your actual hard drive). They show as 53 GB with 18 GB free, which matches your host's disk.
- Several tmpfs entries are temporary in-memory filesystems (RAM-backed), used for performance.

```
vagrant@ubuntu-jammy:~$ cat /opt/provision-logs/uname.txt
========================================================================
Linux ubuntu-jammy 5.15.0-140-generic #150-Ubuntu SMP Sat Apr 12
06:00:09 UTC 2025 x86_64 x86_64 x86_64 GNU/Linux
========================================================================
```

Explanation:

- **OS:** Linux (Ubuntu 22.04 "Jammy Jellyfish")
- **Kernel version:** 5.15.0-140-generic
- **Architecture:** x86_64 (64-bit)
- **Date of kernel build:** April 12, 2025

```
vagrant@ubuntu-jammy:~$ exit
========================================================================
logout
========================================================================
```

6. Reprovision with Additional Commands

Add a few more lines to your provisioning block, e.g., to disable the firewall:

systemctl stop ufw
systemctl disable ufw

Then run:
vagrant provision

You can check UFW status again inside the VM to confirm it's off:
vagrant ssh
systemctl status ufw
exit

```
@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ cat Vagrantfile
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|
  # Set the base box to Ubuntu 22.04 (Jammy Jellyfish) 64-bit
  config.vm.box = "ubuntu/jammy64"

  # Configure a private network with a static IP address
  config.vm.network "private_network", ip: "192.168.56.10"

  # Configure a public (bridged) network using the specified network
interface
  config.vm.network "public_network", bridge: "wlp0s20f3"

  # Set VirtualBox provider-specific options
  config.vm.provider "virtualbox" do |vb|
    # Allocate 1024 MB of RAM to the VM
    vb.memory = 1024
    # Allocate 2 CPU cores to the VM
    vb.cpus = 2
  end

  # Sync the local ./scripts folder to /opt/scripts in the VM
  config.vm.synced_folder "./scripts", "/opt/scripts"

 config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
    systemctl enable apache2
    systemctl start apache2
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/
html/index.html
  SHELL

  config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
```

```
    systemctl enable apache2
    systemctl start apache2
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/
html/index.html

    # Create a logs directory and capture system info
    mkdir -p /opt/provision-logs
    free -m > /opt/provision-logs/memory.txt
    df -h > /opt/provision-logs/disk.txt
    uname -a > /opt/provision-logs/uname.txt
    systemctl stop ufw
    systemctl disable ufw
  SHELL


end
================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant provision
================================================================================
==> default: Running provisioner: shell...
    default: Running: inline script

…

================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant ssh
================================================================================
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-140-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Thu Jun  5 10:22:34 UTC 2025

  System load:             0.2
  Usage of /:              4.4% of 38.70GB
  Memory usage:            24%
  Swap usage:              0%
  Processes:               113
  Users logged in:         0
  IPv4 address for enp0s3: 10.0.2.15
  IPv4 address for enp0s9: 172.20.10.3
  IPv6 address for enp0s9: 2a00:20:8:4862:a00:27ff:fe84:b0a9


Expanded Security Maintenance for Applications is not enabled.

31 updates can be applied immediately.
25 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
```

```
New release '24.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Thu Jun  5 10:07:29 2025 from 10.0.2.2
================================================================================

vagrant@ubuntu-jammy:~$ systemctl status ufw
================================================================================
○ ufw.service - Uncomplicated firewall
     Loaded: loaded (/lib/systemd/system/ufw.service; disabled; vendor
preset: enabled)
     Active: inactive (dead)
       Docs: man:ufw(8)
================================================================================

vagrant@ubuntu-jammy:~$ exit
================================================================================
logout
================================================================================
```

7. Optional: Use the Synced Folder

Add a script to ./scripts locally, e.g., install.sh:

#!/bin/bash
echo "This is from the synced script!" > /opt/provision-logs/from-script.txt

Make it executable:
chmod +x ./scripts/install.sh

Then modify your Vagrantfile to add another provisioner that runs this script:
config.vm.provision "shell", path: "/opt/scripts/install.sh"

Then run:
vagrant provision

And check:s
vagrant ssh
cat /opt/provision-logs/from-script.txt
exit

```
@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ touch install.sh

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ nano install.sh

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ cat install.sh
==========================================================================
#!/bin/bash
echo "This is from the synced script!" > /opt/provision-logs/from-
script.txt
==========================================================================


@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ touch install.sh

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ nano install.sh

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ cat install.sh
==========================================================================
#!/bin/bash
echo "This is from the synced script!" > /opt/provision-logs/from-
script.txt
==========================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ chmod +x ./
scripts/install.sh
==========================================================================
chmod: cannot access './scripts/install.sh': No such file or directory
==========================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ ls
```

```
================================================================================
Vagrantfile  install.sh  scripts
================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ mv install.sh
scripts

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ ls
================================================================================
Vagrantfile  scripts
================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ cd scripts

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3/scripts$ ls
================================================================================
install.sh
================================================================================

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3/scripts$ cd ..

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ chmod +x ./
scripts/install.sh


@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ nano Vagrantfile

@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ cat Vagrantfile
================================================================================
Vagrant.configure("2") do |config|
  # Set the base box to Ubuntu 22.04 (Jammy Jellyfish) 64-bit
  config.vm.box = "ubuntu/jammy64"

  # Configure a private network with a static IP address
  config.vm.network "private_network", ip: "192.168.56.10"

  # Configure a public (bridged) network using the specified network
interface
  config.vm.network "public_network", bridge: "wlp0s20f3"

  # Set VirtualBox provider-specific options
  config.vm.provider "virtualbox" do |vb|
    # Allocate 1024 MB of RAM to the VM
    vb.memory = 1024
    # Allocate 2 CPU cores to the VM
    vb.cpus = 2
  end

  # Sync the local ./scripts folder to /opt/scripts in the VM
  config.vm.synced_folder "./scripts", "/opt/scripts"

 config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
    systemctl enable apache2
    systemctl start apache2
```

```
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/
html/index.html
  SHELL

  config.vm.provision "shell", inline: <<-SHELL
    apt update -y
    apt install -y apache2
    systemctl enable apache2
    systemctl start apache2
    echo "Hello from Provisioned Apache on Ubuntu Jammy!" > /var/www/
html/index.html

    # Create a logs directory and capture system info
    mkdir -p /opt/provision-logs
    free -m > /opt/provision-logs/memory.txt
    df -h > /opt/provision-logs/disk.txt
    uname -a > /opt/provision-logs/uname.txt
    systemctl stop ufw
    systemctl disable ufw
  SHELL

    config.vm.provision "shell", path: "scripts/install.sh"


end
================================================================================


@Ubuntu:~/Testdir/vagrant/myvirtualmachines/machine3$ vagrant provision
================================================================================
==> default: Running provisioner: shell...
    default: Running: inline script

…

================================================================================


tiago-paquete@Ubuntu1:~/Testdir/vagrant/myvirtualmachines/machine3$
vagrant ssh
================================================================================
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-140-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Thu Jun  5 10:42:34 UTC 2025

  System load:             0.1
  Usage of /:              4.4% of 38.70GB
  Memory usage:            23%
  Swap usage:              0%
  Processes:               113
  Users logged in:         0
  IPv4 address for enp0s3: 10.0.2.15
```

```
  IPv4 address for enp0s9: 172.20.10.3
  IPv6 address for enp0s9: 2a00:20:8:4862:a00:27ff:fe84:b0a9


Expanded Security Maintenance for Applications is not enabled.

31 updates can be applied immediately.
25 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

New release '24.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Thu Jun  5 10:27:57 2025 from 10.0.2.2
==============================================================================

vagrant@ubuntu-jammy:~$ cat /opt/provision-logs/from-script.txt
==============================================================================
This is from the synced script!
==============================================================================

vagrant@ubuntu-jammy:~$ exit
==============================================================================
logout
==============================================================================
```

Summary of Provisioning Commands

Command	Purpose
vagrant up	Creates and boots VM + initial provisioning
vagrant provision	Re-runs provisioning scripts
vagrant reload	Reboots VM (does *not* reprovision)
vagrant reload --provision	Reboots and reprovisions