

Vagrant Deployment

To install **Vagrant** on Ubuntu (including 22.04 and 24.04), follow these steps to get up and running quickly with VirtualBox:

✅ Step 1: Install VirtualBox

Vagrant needs a provider, and VirtualBox is the default.

bash

CopyEdit

```
sudo apt update
sudo apt install virtualbox -y
```

✅ Step 2: Install Vagrant

Download the `.deb` package from the official site and install it.

bash

CopyEdit

```
curl -O https://releases.hashicorp.com/vagrant/2.4.1/vagrant_2.4.1-1_amd64.deb
sudo apt install ./vagrant_2.4.1-1_amd64.deb
```

You can check for newer versions at: <https://www.vagrantup.com/downloads>

✅ Step 3: Verify Vagrant Installation

bash

CopyEdit

```
vagrant --version
```

Should return something like:

nginx

CopyEdit

```
Vagrant 2.4.1
```

✅ Step 4: Create Your Vagrant Environment

bash

CopyEdit

```
mkdir k8s-lab && cd k8s-lab
vagrant init ubuntu/jammy64
```

This creates a `Vagrantfile`. Now edit it to configure your Kubernetes control plane and worker nodes.



Example Vagrantfile for Kubernetes Cluster

Here's a simple example to spin up 1 control-plane and 2 workers:

ruby

CopyEdit

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/jammy64"

  (1..3).each do |i|
    name = i == 1 ? "control-plane" : "worker#{i - 1}"

    config.vm.define name do |node|
      node.vm.hostname = name
      node.vm.network "private_network", type: "dhcp"
      node.vm.provider "virtualbox" do |vb|
        vb.name = name
        vb.memory = 2048
        vb.cpus = 2
      end
    end
  end
end
```



Step 5: Bring Up the VMs

bash

CopyEdit

```
vagrant up
```

SSH into them:

bash

CopyEdit

```
vagrant ssh control-plane
vagrant ssh worker1
```

Let me know if you want:

- A pre-written Kubernetes setup script (`kubeadm-init.sh`, `join.sh`)
- Shared folders or synced configs
- A script to install Docker, kubeadm, kubelet automatically

You're really close to having a killer home lab. Want help setting up the kubeadm scripts next?

Vagrant-Home-K8S-Lab

Vagrantfile

```
Vagrant.configure("2") do |config|
```

```
  config.vm.define "control" do |control|
    control.vm.box = "ubuntu/jammy64"
    control.vm.hostname = "control"
    control.vm.network "public_network", bridge: "wlo1" # adjust for your interface
```

```

    control.vm.provider "virtualbox" do |vb|
      vb.name = "master-node-002"
      vb.memory = 2048
      vb.cpus = 2
    end
  end
```

```
  config.vm.define "cluster-node-003" do |worker|
    worker.vm.box = "ubuntu/jammy64"
    worker.vm.hostname = "cluster-node-003"
    worker.vm.network "public_network", bridge: "wlo1"
```

```

    worker.vm.provider "virtualbox" do |vb|
      vb.name = "cluster-node-003"
      vb.memory = 2048
      vb.cpus = 1
```

```
end
end

config.vm.define "cluster-node-004" do |worker|
  worker.vm.box = "ubuntu/jammy64"
  worker.vm.hostname = "cluster-node-004"
  worker.vm.network "public_network", bridge: "wlo1"

  worker.vm.provider "virtualbox" do |vb|
    vb.name = "cluster-node-004"
    vb.memory = 2048
    vb.cpus = 1
  end
end

end
```