 **A step-by-step guide to creating a load-balanced architecture using VirtualBox, Ubuntu 24.04 LTS, HAProxy, and Nginx. Perfect for students, developers, or home lab enthusiasts.**

---

## Part 1: Install VirtualBox on Ubuntu

### ◆ Step 1: Update Your System

```
sudo apt update && sudo apt upgrade -y
```

### ◆ Step 2: Install Required Dependencies

```
sudo apt install -y dkms build-essential linux-headers-$(uname -r)
```

### ◆ Step 3: Add Oracle VirtualBox Repository

#### Add GPG Key

```
wget -q https://www.virtualbox.org/download/oracle_vbox_2016.asc -O- | \
sudo gpg --dearmor -o /usr/share/keyrings/oracle-virtualbox.gpg
```

#### Add Repository


```
echo "deb [signed-by=/usr/share/keyrings/oracle-virtualbox.gpg]
https://download.virtualbox.org/virtualbox/debian $(lsb_release -cs) contrib" | \
sudo tee /etc/apt/sources.list.d/virtualbox.list
```

#### Update Package Index

```
sudo apt update
```

### ◆ Step 4: Install VirtualBox

```
sudo apt install virtualbox-7.0 -y
```

 Replace 7.0 with another version (like 6.1) if needed.

#### ◆ Step 5 (Optional): Install Extension Pack

Download from 🖱️ <https://www.virtualbox.org/wiki/Downloads>

Then install:

```
sudo VBoxManage extpack install Oracle_VM_VirtualBox_Extension_Pack-*.vbox-extpack
```

#### ◆ Step 6: Launch VirtualBox

```
virtualbox
```

Or via GUI: **Activities Menu → Oracle VM VirtualBox**

#### 📌 Check Installed Version

```
VBoxManage --version
```

Expected output:

```
7.0.18r162988
```

---

---

## Part 2: Ubuntu 24.04 LTS VM Setup (VirtualBox)

### VM Settings

- **Format:** VDI
- **Storage:** Dynamically allocated
- **Disk Size:** 20 GB
- **RAM:**  $\geq 2$  GB
- **Adapter 1:** Bridged Adapter (internet)
- **Adapter 2:** Host-only Adapter (vboxnet0)

### Install Ubuntu

1. Download ISO from [ubuntu.com](https://ubuntu.com).
2. Mount ISO via VirtualBox Storage → Optical Drive.
3. Start VM and install Ubuntu:
  - Set hostname (web1, web2, loadbalancer)
  - Create a user and strong password

---

## Part 3: Post-Installation Setup

```
sudo apt update && sudo apt upgrade -y  
timedatectl set-ntp true
```

---

## Part 4: Configure Network (Netplan)


### Get Your Gateway

On host OS :

```
ip route | grep default
```

Example output:

```
default via 10.X.X.XX dev wlp3s0
```

 **Configure /etc/netplan/01-netcfg.yaml (e.g. for web1, replace XX with the last part of the IP: web1 → 11, web2 → 12, loadbalancer → 10, etc.)**

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: no
      addresses: [10.X.X.50/24]
      routes:
        - to: 0.0.0.0/0
          via: 10.X.X.XX
      nameservers:
        addresses: [8.8.8.8, 1.1.1.1]

    enp0s8:
      dhcp4: no
      addresses: [192.168.56.XX/24]
```

⚠ Don't add gateway4 to enp0s8.

**\* Please use 2 SPACES per level where needed, no TABs**

### Apply Changes

```
sudo chmod 644 /etc/netplan/01-netcfg.yaml
sudo netplan apply
```

### Verify

```
ip a
ping 8.8.8.8
curl https://google.com
```

---

## **Part 5: Edit /etc/hosts on All VMs**

```
sudo nano /etc/hosts
```

Add:

```
192.168.56.10 loadbalancer
```

```
192.168.56.11 web1
```

```
192.168.56.12 web2
```

```
192.168.56.13 web3
```

CopyEdit

---

## **Part 6: HAProxy Load Balancer (on loadbalancer VM)**

### **Install HAProxy**

```
sudo apt update
```

```
sudo apt install haproxy -y
```

```
sudo systemctl enable haproxy
```

```
sudo cp /etc/haproxy/haproxy.cfg /etc/haproxy/haproxy.cfg.backup
```

```
sudo nano /etc/haproxy/haproxy.cfg
```

## **HAProxy Config:**

(in the Next page 😊)

global

log /dev/log local0

log /dev/log local1 notice

daemon

defaults

log global

mode http

option httplog

option dontlognull

timeout connect 5000

timeout client 50000

timeout server 50000

frontend http\_front

bind \*:80

default\_backend http\_back

backend http\_back

balance roundrobin

server web1 192.168.56.11:80 check

server web2 192.168.56.12:80 check

server web3 192.168.56.13:80 check

listen stats

bind \*:8080

stats enable

stats uri /stats

stats refresh 10s

stats admin if TRUE

## Test & Start HAProxy

```
sudo haproxy -c -f /etc/haproxy/haproxy.cfg  
sudo systemctl restart haproxy  
sudo systemctl status haproxy
```

---

## Part 7: Web Servers (web1, web2, web3)

### Install Nginx

```
sudo apt update  
sudo apt install nginx -y  
sudo systemctl enable nginx  
sudo systemctl start nginx
```

### Create Web Pages

#### #On Web 1

```
echo "<h1>Welcome from Web Server 1</h1><p>Server IP: 192.168.56.11</p>" | sudo tee  
/var/www/html/index.html
```

#### #On Web 2

```
echo "<h1>Welcome from Web Server 2</h1><p>Server IP: 192.168.56.12</p>" | sudo tee  
/var/www/html/index.html
```

#### #On Web 3

```
echo "<h1>Welcome from Web Server 3</h1><p>Server IP: 192.168.56.13</p>" | sudo tee  
/var/www/html/index.html
```

---

---

## Part 8: Configure UFW (Firewall)

```
sudo ufw allow 'Nginx Full'
sudo ufw enable
```

---

## Part 9: Testing & Verification

### Test Load Balancing

```
for i in {1..10}; do curl http://192.168.56.10; echo "---"; done
```

### Browser Testing (on Host)

Then visit:

- <http://loadbalancer.local>
  - <http://192.168.56.10:8080/stats>
- 

## Part 10: Logs & Monitoring

### HAProxy Logs

```
sudo tail -f /var/log/haproxy.log
```

### Nginx Logs

```
sudo tail -f /var/log/nginx/access.log
sudo tail -f /var/log/nginx/error.log
```

---

---



---

## ✅ Final Notes

- This setup is a great foundation to learn load balancing and server management.
  - You can expand it by adding Node.js apps, databases, or Docker.
  - Feel free to test this setup on real hardware (e.g., Raspberry Pi or old PCs).
  - Customize and build on top of this for your own projects or portfolio!
- 

## 🙌 Thanks for Reading

If this helped you, consider sharing it with others.

Made with  by **[Rishabh Aka Raipius]**

Bhind, Madhya Pradesh, India |

|rishabhsinghrajawat.dev@gmail.com|[\[https://github.com/Raipius\]](https://github.com/Raipius) ||

|[\[https://www.linkedin.com/in/rishabh-singh-rajawat-5a1b782bb/\]](https://www.linkedin.com/in/rishabh-singh-rajawat-5a1b782bb/)|

---