VIRTUALIZATION & CLOUD COMPUTING ASSIGNMENT-1 REPORT

by

DEEPRAJ MAJUMDAR (M23CSE012)

Under the Supervision of

Dr. Sumit Kalra

Title : Deploying a Microservice-Based Application Using Virtual Machines

Objective: The objective of this project is to create and configure multiple **Virtual Machines (VMs)** using **VirtualBox**, establish a network between them, and deploy a microservice-based application across the connected VMs. The project will involve setting up **Ubuntu** as the main **API server**, **Lubuntu** as the request sender, and **Xubuntu** as a load balancer.

Introduction:

Virtualization and Its Importance

Virtualization is the process of creating virtual instances of computer resources, such as operating systems, storage, and networks, on a single physical machine. This allows multiple **virtual machines (VMs)** to run on the same hardware independently, optimizing resource usage and improving scalability.

Types of Virtualization

1. **Full Virtualization** - The entire OS runs on a **virtual environment**, independent of the host system.
2. **Para-Virtualization** - The guest OS is aware that it is virtualized and interacts with the **hypervisor** efficiently.
3. **Container-Based Virtualization** - Applications run in isolated containers within the same **OS kernel**.

In this project, we use **Full Virtualization** with **VirtualBox** to deploy multiple VMs.

Technologies Used

VirtualBox

Oracle VirtualBox is a free and open-source hypervisor that allows users to run multiple operating systems on a single machine. It supports various OS environments, networking configurations, and snapshots for system recovery.

Operating Systems Used

Introduction to Operating Systems

An Operating System (OS) is system software that manages hardware and software resources, providing a user interface and facilitating the execution of applications.

Overview of Ubuntu, Lubuntu, and Xubuntu

- **Ubuntu**: A popular Linux distribution based on Debian, known for its ease of use and extensive community support.
- **Lubuntu**: A lightweight version of Ubuntu designed for low-resource environments, using the LXQt desktop environment.
- **Xubuntu**: Another lightweight Ubuntu variant, optimized for performance with the Xfce desktop environment.

Advantages and Disadvantages of Each OS

OS	Advantages	Disadvantages
Ubuntu	User-friendly,widely supported, strong security	Higher resource consumption
Lubuntu	Lightweight, fast, minimal hardware requirements	Fewer pre-installed applications
Xubuntu	Balanced performance, stability, and low resource usage	Limited customization compared to Ubuntu

Type of Operating Systems Used

All three operating systems are **Linux-based**, making them **stable, secure,** and well-suited for networking and server deployment.

Role of Each OS in the Project

- **Ubuntu (Main API Server)**: Hosts the **Node.js microservice** and handles **API requests.**
- **Lubuntu (Request Sender)**: Acts as the client, sending requests to the API.

- **Xubuntu (Load Balancer)**: Uses Nginx to distribute incoming requests to multiple backend servers.

Architecture Design

Below is a diagram illustrating the network architecture of our microservice deployment

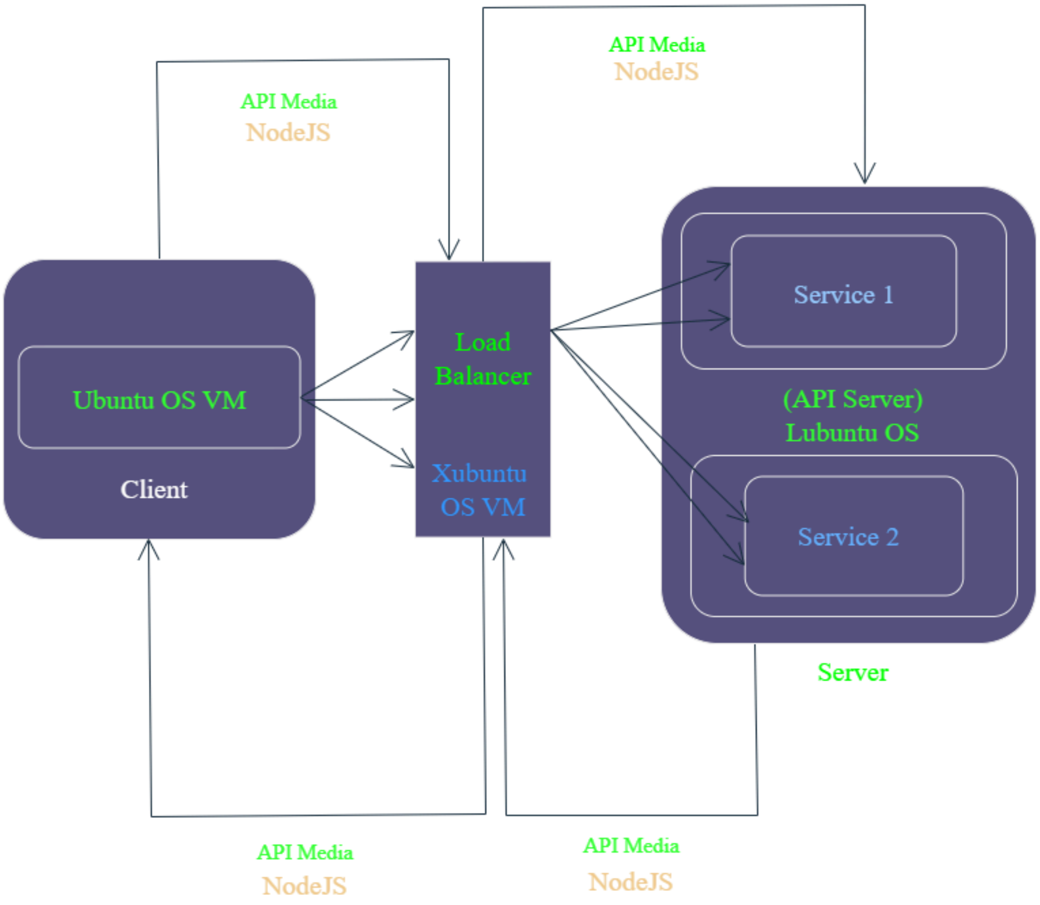


Fig 1.3 The Architecture of Microservices implemented across Virtual Machines

- The **Lubuntu VM** sends requests to the **Xubuntu VM**.
- The **Xubuntu VM (Nginx Load Balancer)** forwards requests to the **Ubuntu VM (API Server)**.
- The **Ubuntu VM** processes requests and returns responses.

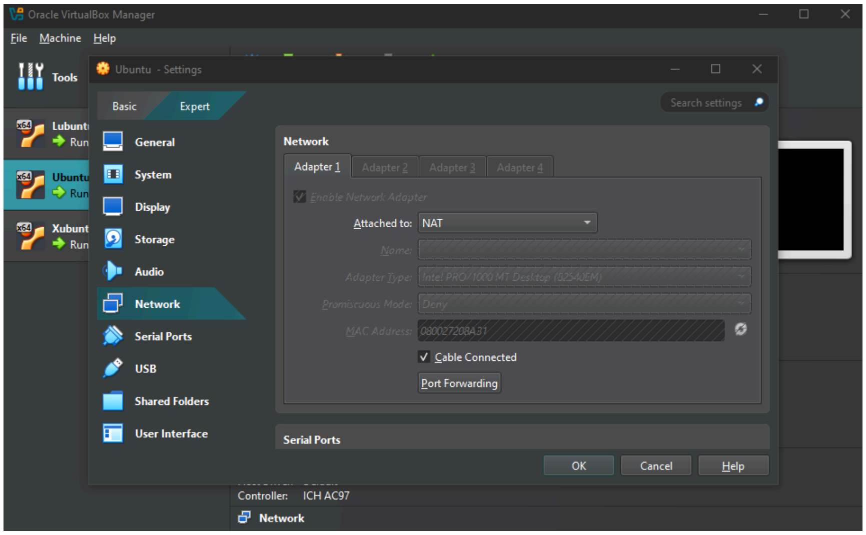
Step-by-Step Implementation Workflow

Step 1: Installing VirtualBox and Creating VMs

1. Download and install **VirtualBox** from [VirtualBox Website](#).
2. Create three virtual machines and install **Ubuntu, Lubuntu, and Xubuntu.**

Step 2: Configuring Network Settings

1. Set all VMs to **Host-Only Network** mode in VirtualBox.
2. Use ip a to check **IP addresses** and ensure communication between **VMs.**



Network

☑ Enable Network Adapter

Attached to: NAT

Name:

Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Deny

MAC Address: 080027208A31

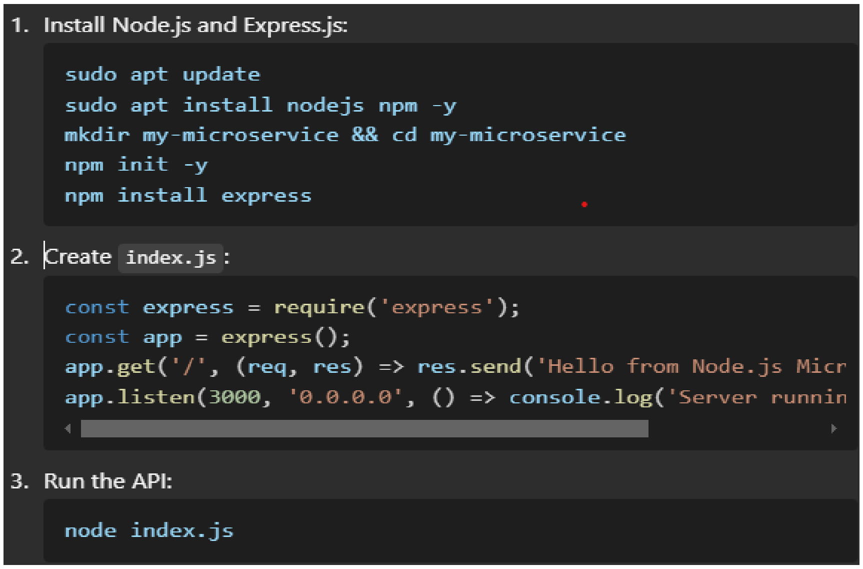
☑ Cable Connected

Serial Ports

Controller: ICH AC97

Network

Step 3: Deploying the API on Ubuntu VM



1. Install Node.js and Express.js:

```
sudo apt update
sudo apt install nodejs npm -y
mkdir my-microservice && cd my-microservice
npm init -y
npm install express
```

2. Create `index.js`:

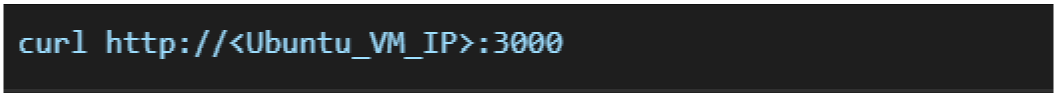
```
const express = require('express');
const app = express();
app.get('/', (req, res) => res.send('Hello from Node.js Micr
app.listen(3000, '0.0.0.0', () => console.log('Server runnin
```

3. Run the API:

```
node index.js
```

Step 4 : Testing API from Lubuntu VM

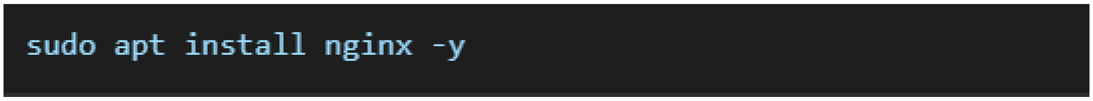
1. **Use curl to check connectivity:**



```
curl http://<Ubuntu_VM_IP>:3000
```

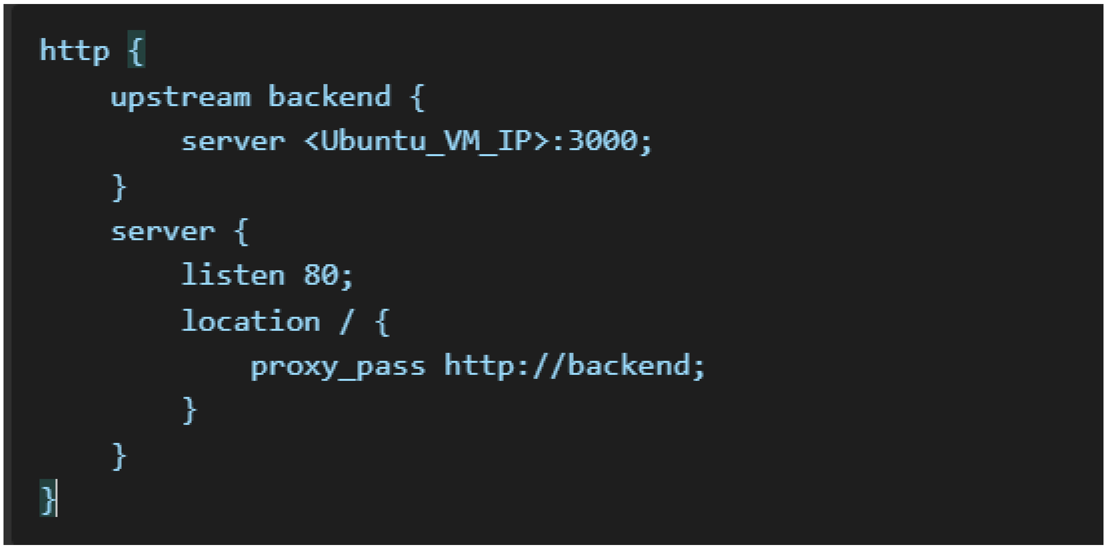
Step 5 : Configuring Xubuntu as a Load Balancer

1. **Install Nginx:**



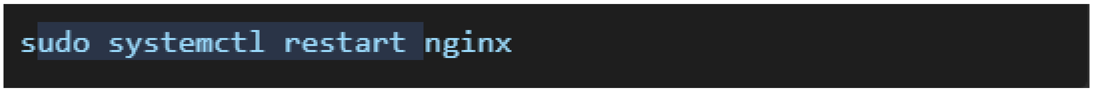
```
sudo apt install nginx -y
```

2. **Modify Nginx configuration (/etc/nginx/nginx.conf):**



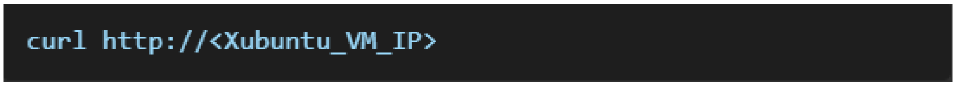
```
http {
    upstream backend {
        server <Ubuntu_VM_IP>:3000;
    }
    server {
        listen 80;
        location / {
            proxy_pass http://backend;
        }
    }
}
```

3. **Restart Nginx**



```
sudo systemctl restart nginx
```

4. **Test Load Balancing**

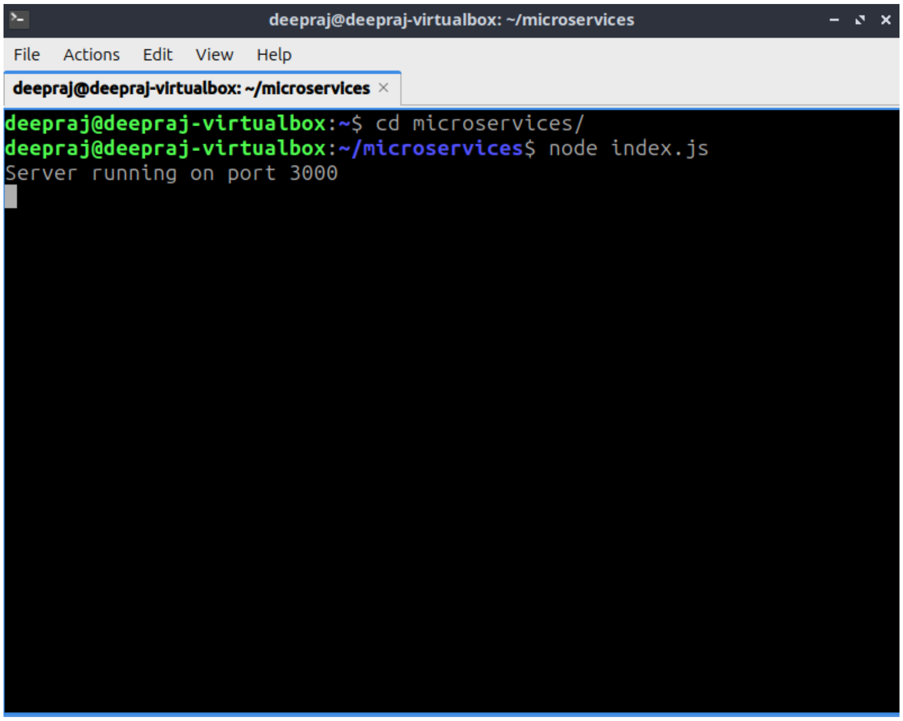


```
curl http://<Xubuntu_VM_IP>
```

Results

Here I have shown the three VMs with the outputs

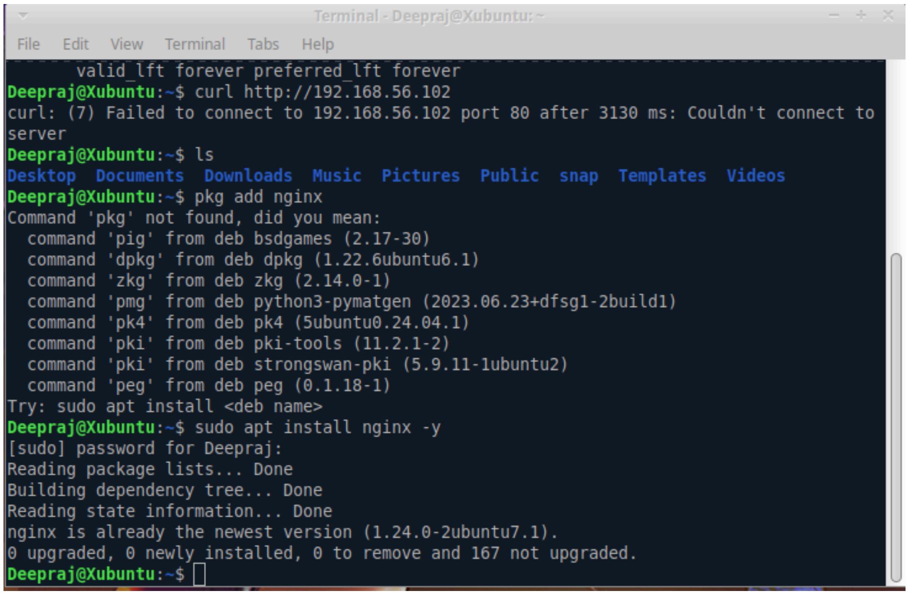
1. The main server (Lubuntu VM). It has the Node Js installed in it. Once setup it runs the index.js file



deepraj@deepraj-virtualbox: ~/microservices
File Actions Edit View Help
deepraj@deepraj-virtualbox: ~/microservices

```
deepraj@deepraj-virtualbox:~$ cd microservices/
deepraj@deepraj-virtualbox:~/microservices$ node index.js
Server running on port 3000
```

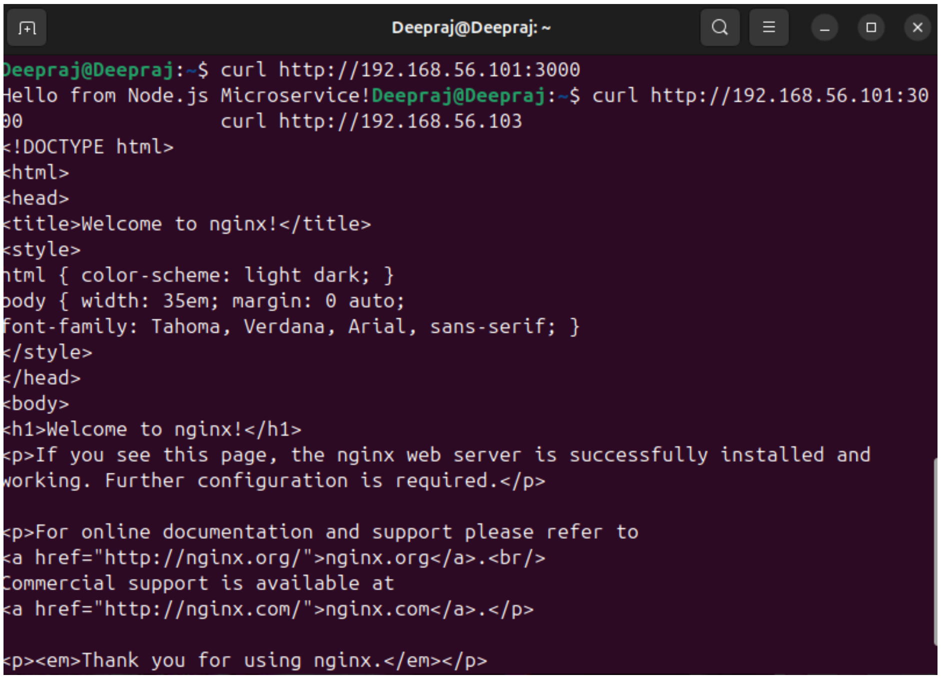
2. The Xubuntu VM acts as the Load balancer with NGINX installed in it and configured



Terminal - Deepraj@Xubuntu: ~
File Edit View Terminal Tabs Help

```
       valid_lft forever preferred_lft forever
Deepraj@Xubuntu:~$ curl http://192.168.56.102
curl: (7) Failed to connect to 192.168.56.102 port 80 after 3130 ms: Couldn't connect to
server
Deepraj@Xubuntu:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos
Deepraj@Xubuntu:~$ pkg add nginx
Command 'pkg' not found, did you mean:
  command 'pig' from deb bsdgames (2.17-30)
  command 'dpkg' from deb dpkg (1.22.6ubuntu6.1)
  command 'zkg' from deb zkg (2.14.0-1)
  command 'pmg' from deb python3-pymatgen (2023.06.23+dfsg1-2build1)
  command 'pk4' from deb pk4 (5ubuntu0.24.04.1)
  command 'pki' from deb pki-tools (11.2.1-2)
  command 'pki' from deb strongswan-pki (5.9.11-1ubuntu2)
  command 'peg' from deb peg (0.1.18-1)
Try: sudo apt install <deb name>
Deepraj@Xubuntu:~$ sudo apt install nginx -y
[sudo] password for Deepraj:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nginx is already the newest version (1.24.0-2ubuntu7.1).
0 upgraded, 0 newly installed, 0 to remove and 167 not upgraded.
Deepraj@Xubuntu:~$
```

3. The Ubuntu VM acts as the client requesting service from the Lubuntu VM (main server) as well as communicating with the Xubuntu Machine (Load Balancer).



```
Deepraj@Deepraj:~$ curl http://192.168.56.101:3000
Hello from Node.js Microservice!Deepraj@Deepraj:~$ curl http://192.168.56.101:30
00                   curl http://192.168.56.103
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
```

Conclusion

This project successfully deployed a microservice-based application using multiple VMs. **Lubuntu** acted as the **API server**, **Ubuntu sent requests**, and **Xubuntu** balanced traffic using **Nginx**. VirtualBox facilitated seamless networking between the VMs.

Future Work

- Implement Docker for containerized deployments.
- Use Kubernetes for dynamic load balancing.
- Automate VM provisioning with Vagrant.
-

Links

Github :
Recorded Video :
https://drive.google.com/file/d/1ikkEA_use75uaCJV5z9aNd3uGoW0EVTl/view?usp=sharing