Lab 02: Intelligent Load Balancing with Bash & Python on Linux

Objective

In this lab, you will simulate a load balancer using Bash and Python scripts on a Linux environment. You will learn to redirect traffic based on server load conditions and introduce basic AI-style decision logic without using machine learning libraries.

Requirements

A Linux system or laptop with:

- At least **4 GB RAM**
- A working Linux distribution (Ubuntu recommended)
- **3 terminals or VMs/containers** to simulate Load Balancer, and two servers

Software:

- Linux (Ubuntu 20.04+)
- Python 3
- Bash
- `iperf3` – for traffic generation
- `ping`, `netstat`, `ss` – for monitoring
- `tc` (optional) – for traffic simulation

Network Configuration

- One machine acts as a **Load Balancer (LB)**
- Two machines act as **Servers (S1 and S2)**
- All three must be on the same local network (LAN, Docker bridge, or loopback ports)

Tasks

Task 1: Set Up Server Simulations

Install `iperf3` on all three systems:

```
sudo apt update
sudo apt install iperf3 -y
```

On **S1** and **S2**, start iperf3 in server mode:

```
iperf3 -s
```

Note down the IP addresses of both servers for use in the LB script.

Task 2: Create Load Balancer Script (Bash)

On the Load Balancer machine:

- Create a Bash script (`load_balancer.sh`)
- Simulate load values using random numbers or `netstat` output
- Compare load values and redirect client traffic to the less loaded server using `iperf3 -c`

Use chmod to make the script executable:

```
chmod +x load_balancer.sh
```

Run the script and observe which server it selects.

Task 3: Enhance Logic Using Python

- Create a Python script (`ai_load_balancer.py`)
- Simulate current server loads
- Add basic history tracking to avoid overusing one server
- Use simple logic to balance load over time

Optional: Add `os.system()` calls to send traffic as the decision is made.

Task 4: Analyze Performance and Decisions

- Run both scripts multiple times
- Compare how Bash and Python scripts make different decisions
- Optionally use `ping` or `iperf3` statistics to analyze throughput and latency
- Observe how history tracking changes the choice logic in Python

Submission

Submit a **zip file** named `E19XXX_Lab02.zip`, where `XXX` is your enrollment number. Your submission must include:

- **`Load_balancer.sh`**
- **`Ai_load_balancer.py`**
- `A Report named` **`E19XXX_Lab02.pdf,`** `including all your observations, commands, and supporting screenshots. Make sure to include the steps used to simulate servers and traffic.`

Notes

- Run all traffic tests multiple times for consistent results
- Use `Ctrl+C` to stop `iperf3` servers
- Ensure all systems have open ports and allow traffic if running in containers or VMs