

**Department of Engineering Technology**

SET-222

Software Operations & Maintenance

Experiment # 11

**Experiment Title**

**High Availability & Scalability (Load balancing, horizontal vs. vertical scaling, auto-scaling)**

**Assessment of CLO(s): 03**

**Performed on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name:** |  | | |
| **Roll No.** |  | **Group** |  |
| **Semester** |  | **Session** |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S. No.** | **Perf. Level**  **Criteria** | **Excellent**  **(2.5)** | **Good**  **(2)** | **Satisfactory**  **(1.5)** | **Needs Improvement**  **(0 ~ 1)** | **Marks Obtained** |
| **1** | Project Execution & Implementation | Fully functional, optimized, and well-structured. | Minor errors, mostly functional. | Some errors, requires guidance. | Major errors, non-functional, or not Performed. |  |
| **2** | Results & Debugging  Or Troubleshooting | Accurate results with effective debugging  Or Troubleshooting. | Mostly correct, some debugging Or Troubleshooting needed. | Partial results, minimal debugging  Or Troubleshooting. | Incorrect results, no debugging Or Troubleshooting, or not attempted. |  |
| **3** | Problem-Solving & Adaptability  (VIVA) | Creative approach, efficiently solves challenges. | Adapts well, minor struggles. | Some adaptability, needs guidance. | Lacks innovation or no innovation, unable to solve problems. |  |
| **4** | Report Quality & Documentation | Clear, structured, with detailed visuals. | Mostly clear, minor gaps. | Some clarity issues, missing details. | Poorly structured, lacks clarity, or not submitted. |  |
| **Total Marks Obtained Out of 10** | | | | | |  |

**Experiment evaluated by**

|  |  |  |  |
| --- | --- | --- | --- |
| **Instructor’s Name** | **Ms. Shagufta Aftab** | | |
| **Date** |  | **Signature** |  |

## Copyright © Department of Engineering & Technology – UIT University Karachi

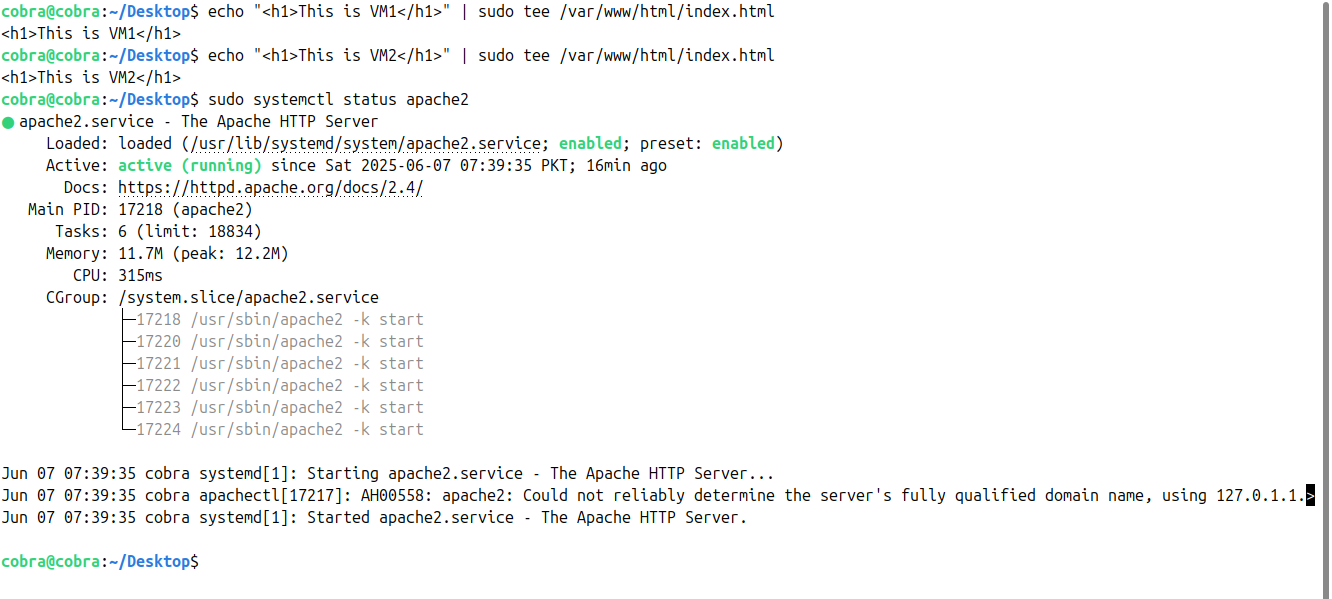
**Objective:**

To demonstrate the use of load balancing, horizontal and vertical scaling, and auto-scaling for improving the availability and scalability of IT services.

**Lab Tasks:**

**Task 1: Setup Two Web Servers**

* Deploy and configure two instances running a web server (e.g., Apache on VM1 and VM2).



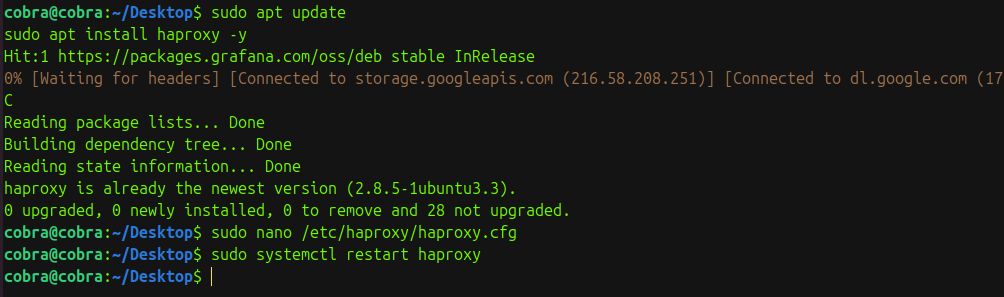
**Task 2: Configure Load Balancer**

Modify homepage to show the server name (for testing load distribution).

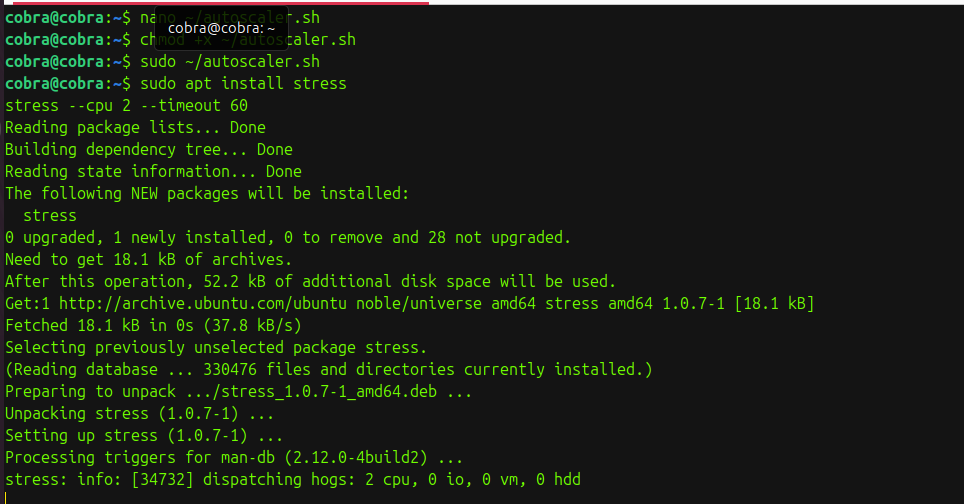
* **sudo nano /etc/haproxy/haproxy.cfg:**

At bottom put this code.

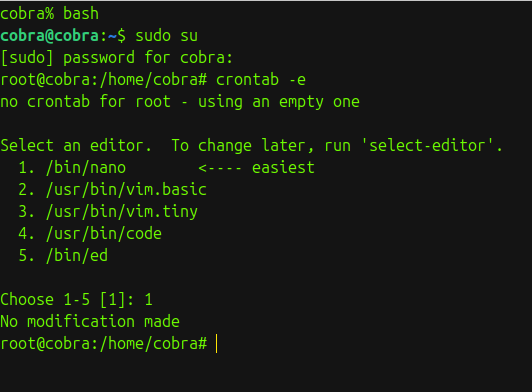
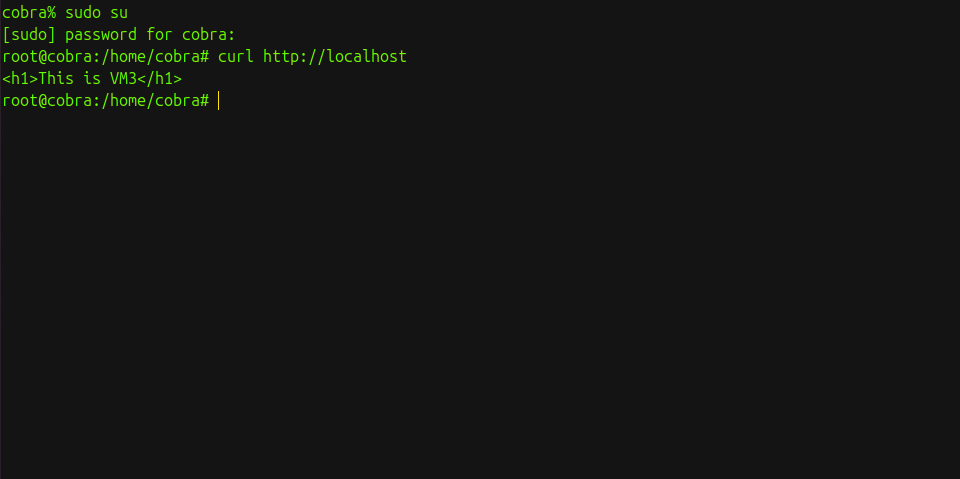
|  |
| --- |
| frontend http\_front  bind \*:80  default\_backend http\_back  backend http\_back  balance roundrobin  server vm1 <VM1-IP>:80 check  server vm2 <VM2-IP>:80 check |



**Task 3: Simulate Horizontal and Vertical Scaling**



**Task 4: Auto-Scaling (Optional - Cloud)**



autoscaler.sh:

|  |
| --- |
| #!/bin/bash  CPU\_USAGE=$(top -bn1 | grep "Cpu(s)" | awk '{print $2 + $4}')  UP\_THRESHOLD=60.0  DOWN\_THRESHOLD=20.0  NEW\_SERVER\_IP="192.168.100.5"  HAPROXY\_CFG="/etc/haproxy/haproxy.cfg"  if (( $(echo "$CPU\_USAGE > $UP\_THRESHOLD" | bc -l) )); then  echo "🔺 High CPU: $CPU\_USAGE%. Starting VM and adding to HAProxy..."  VBoxManage startvm "WebServer3" --type headless  sleep 10  if ! grep -q "$NEW\_SERVER\_IP" "$HAPROXY\_CFG"; then  sed -i "/backend http\_back/a \ server vm3 $NEW\_SERVER\_IP:80 check" $HAPROXY\_CFG  systemctl restart haproxy  fi  elif (( $(echo "$CPU\_USAGE < $DOWN\_THRESHOLD" | bc -l) )); then  echo "🔻 Low CPU: $CPU\_USAGE%. Shutting down extra server..."  VBoxManage controlvm "WebServer3" poweroff  sed -i "/$NEW\_SERVER\_IP/d" $HAPROXY\_CFG  systemctl restart haproxy  fi |

**Expected Outcomes:**

* Understand and implement a basic load balancer setup.
* Identify the difference between scaling methods.
* Observe benefits of auto-scaling in cloud environments.
* Assessment Questions:

1. Role of a Load Balancer in High Availability

* A load balancer distributes incoming traffic across multiple servers to ensure no single server fails or gets overloaded, improving uptime and reliability.

2. Horizontal vs. Vertical Scaling (with Examples)

* Horizontal Scaling: Add more machines (e.g., adding a 3rd web server).
* Vertical Scaling: Upgrade existing machine's hardware (e.g., increase RAM/CPU of one server).

3. Advantages of Auto-Scaling in Cloud Platforms

* Handles traffic spikes automatically
* Reduces costs by scaling down when demand is low
* Improves performance and reliability

4. How Scaling Improves System Performance

* Scaling increases system resources or capacity, reducing load on individual servers, lowering response time, and preventing downtime.

5. Tools for High Availability Solutions

* HAProxy (load balancing)
* Keepalived (failover)
* Docker Swarm / Kubernetes
* Pacemaker + Corosync
* AWS Auto Scaling / Azure VM Scale Sets