Performance Testing Plan

Testing Methodology Testing will be conducted using a "Black Box" approach, treating the Server VM via remote interfaces.

Workload Type	Justification & Expected Profile	Monitoring Tools & Metrics
CPU-Intensive	Goal: Force 100% CPU load. Profile: High User %, High Load Average, increased temperature.	top, htop Metrics: %CPU, LoadAvg
Memory-Intensive	Goal: Allocate/free large RAM chunks. Profile: High-Used RAM, potential Swap usage.	free -h, vmstat Metrics: Used/Free, Swap In/Out
I/O-Intensive	Goal: Saturate disks write buffers. Profile: High %iowait, low CPU usage.	iostat -x, iotop Metrics: %iowait, tps (transfers/sec)
Network/Server	Goal: Handle concurrent HTTP requests. Profile: Moderate CPU/RAM, high Network I/O.	iftop, netstat, Apache Bench Metrics: Req/sec, Bandwidth

Scenarios

Prepare: *sudo apt update && sudo apt install -y stress sysstat apache2*

Scenario 1: Establishing the Baseline (Idle System)

Before running any load, we must know how the system behaves at rest.

1. Execute Command (Workstation): *vmstat 1 5*
2. Action: Observe the output for 5 seconds while the server is doing nothing.
3. Analyze Result:
   - Look at the id (idle) column under CPU. Expect: Value close to 100.
   - Look at wa (wait). Expect: 0.
   - Look at si/so (swap in/out). Expect: 0.

Scenario 2: CPU Stress Testing

Goal: Saturation of processor cores to test scheduler and cooling.

1. Execute Command (window 1): *stress --cpu 2 --timeout 60s*

2. Execute Monitoring (window 2): *top -bn1 | head -15*

3. Analyze Result:

◦ Locate the stress processes in the list.

◦ Check %CPU column. Expect: Near 100% for stress processes.

◦ Check Load Average (top right). Expect: Value rising above 1.0 or 2.0.

Scenario C: Memory Saturation Testing

Goal: Force memory allocation to test RAM limits and Swap usage.

1. Execute Command (window 1): *stress --vm 2 --vm-bytes 256M --timeout 60s*

2. Execute Monitoring (window 2): *free -h*

3. Analyze Result:

◦ Compare used vs available memory. Expect: available drops significantly.

◦ Check Swap row. Expect: If RAM is full, used swap should increase (value > 0).

Scenario D: Disk I/O Bottleneck Testing

Goal: Saturate the disk write buffer to observe system latency.

1. Execute Command (window 1): *stress --io 4 --timeout 60s*

2. Execute Monitoring (window 2): *iostat -x 1 5*

3. Analyze Result:

◦ Look at %iowait (CPU waiting for disk). Expect: Value significantly higher than 0.

◦ Look at await (latency in ms). Expect: High values (e.g., >10ms), indicating the disk cannot keep up.

Scenario E: Network Throughput Testing

Goal: Simulate web traffic to test bandwidth limits.

1. Execute Command: *ab -n 10000 -c 100 192.168.56.10*

3. Analyze Result:

◦ Look at Requests per second in ab output. Expect: A stable number (e.g., 2000+ req/sec).

◦ Look at RX bytes (receive) on server interface. Expect: Rapid increase during the test.

Security Configuration Checklist

Category	Security Control	Rationale / Mitigation Strategy
Network Security	Firewall (UFW)	Configure a "Deny All Incoming" default policy. Only allow SSH (Port 22) from the Workstation IP to prevent unauthorized network access.
Authentication	SSH Hardening	Disable PasswordAuthentication and enforce PubkeyAuthentication (SSH Keys). Disable PermitRootLogin to prevent direct brute-force attacks on the root account.
Privilege management	User Management	Adhere to the Principle of Least Privilege. Create a non-root administrative user with sudo access for daily tasks instead of using the root account.
Access Control	AppArmor / SELinux	Enforce Mandatory Access Control (MAC) profiles to restrict application capabilities even if a service is compromised.
Intrusion Detection	Fail2Ban	Monitor log files for repeated failed login attempts and automatically ban offending IPs to mitigate brute-force attacks.
Maintenance	Automatic Updates	Configure unattended upgrades to automatically patch critical security vulnerabilities (CVEs) and maintain system integrity.

Threat Model

Threat 1: SSH Brute-Force Attacks

- Risk: Attackers use scripts to guess passwords, potentially gaining root access via the default SSH port.
- Mitigation: Disable password authentication (enforce SSH keys), disable PermitRootLogin, and install Fail2Ban to block repeated failed attempts.

Threat 2: Privilege Escalation

- Risk: A compromised standard user exploits weak file permissions (e.g., SUID binaries) to gain root privileges.
- Mitigation: Apply the Principle of Least Privilege, audit SUID/SGID files, and restrict sudo access to essential commands only.

Threat 3: Unpatched Software Vulnerabilities (CVEs)

- Risk: Exploitation of known bugs in outdated services (e.g., Log4Shell) leads to Remote Code Execution (RCE).
- Mitigation: Enable Automatic Security Updates (unattended upgrades) to ensure critical patches are applied immediately