Week 2 — Security Planning & Testing Methodology

Overview

Week 2 focuses on **security planning and performance testing methodology** prior to implementing any security controls on the server. The objective of this phase is to design a clear security baseline, define a structured testing approach, and identify realistic threats that could affect a Linux server operating in a networked environment. Planning security controls before deployment reflects professional best practice and reduces the likelihood of misconfiguration and vulnerabilities.

This week emphasises **analysis and design rather than configuration**, ensuring that all future implementations are justified, measurable, and aligned with operating system security principles.

Objectives

- Design a performance testing and monitoring methodology
- Define a comprehensive security configuration checklist
- Identify realistic operating system security threats
- Map threats to mitigation strategies
- Prepare for secure system hardening in later phases

Deliverables

- Performance testing plan
- Security configuration checklist
- Threat model with mitigation strategies

---

1. Performance Testing Plan

1.1 Testing Philosophy

Performance testing will be conducted remotely from the workstation to reflect real-world system administration practices. All measurements will be collected **via SSH** using command-line tools to ensure accuracy and repeatability. Baseline metrics will be recorded before workload execution and compared against metrics collected under load and after optimisation.

This approach enables quantitative comparison and supports later analysis of operating system behaviour under different workloads.

1.2 Metrics to be Collected

The following performance metrics were selected as they directly reflect operating system resource management:

Metric	Description	Tools (Planned)
CPU Usage	Processor utilisation under load	`top`, `htop`, `mpstat`
Memory Usage	RAM allocation and availability	`free`, `vmstat`
Disk I/O	Read/write throughput	`iostat`, `dd`
Network Performance	Latency and throughput	`ping`, `iperf3`
System Load	Overall system pressure	`uptime`, `top`

1.3 Testing Methodology

For each selected application or service, the following testing stages will be applied:

1. **Baseline Testing** – Metrics recorded with no additional load
2. **Load Testing** – Metrics recorded while the application is running under stress
3. **Bottleneck Identification** – Analysis of constrained resources
4. **Optimisation Testing** – Re-testing after configuration improvements

All results will be logged and later visualised using tables and graphs to support critical evaluation.

2. Security Configuration Checklist

The following checklist defines the **minimum security baseline** required for a production-ready Linux server. Each control will be implemented, verified, and documented in later weeks.

2.1 Authentication & Access Control

- Disable SSH password authentication
- Enforce SSH key-based authentication
- Disable direct root login via SSH
- Create a non-root administrative user
- Apply least-privilege principles using `sudo`

2.2 Network Security

- Enable and configure a host-based firewall (UFW)
- Restrict SSH access to a single trusted IP address
- Close all unused ports
- Use isolated host-only networking for testing

2.3 System Hardening

- Enable mandatory access control (AppArmor)
- Configure automatic security updates
- Remove or disable unnecessary services
- Apply secure file permissions

2.4 Intrusion Detection & Monitoring

- Install and configure Fail2Ban
- Monitor authentication logs for suspicious activity
- Track firewall activity and access attempts

2.5 Verification & Automation

- Create a security baseline verification script
- Automate checks for firewall status, SSH configuration, and running services
- Document all configurations with before/after evidence

---

3. Threat Model

3.1 Threat Modelling Approach

The threat model identifies realistic risks affecting a Linux server connected to a network. Each threat is assessed based on **likelihood**, **impact**, and **mitigation strategy**, ensuring that security controls are proportionate and justified.

3.2 Identified Threats and Mitigations

Threat	Description	Potential Impact	Mitigation Strategy
Brute-force SSH attacks	Automated login attempts targeting SSH	Unauthorized access	SSH key-based authentication, Fail2Ban, firewall rules
Unauth	Open ports or	Data exposure, service	UFW firewall, port restrictions

Threat	Description	Potential Impact	Mitigation Strategy
orised network access	weak firewall rules	compromise	
Privilege escalation	Exploiting misconfigured permissions	Full system compromise	Least privilege, non-root users, AppArmor
Unpatched vulnerabilities	Outdated software packages	Remote exploitation	Automatic security updates
Denial of Service (DoS)	Resource exhaustion attacks	Service unavailability	Monitoring, resource limits, firewall rules

---

4. Ethical and Security Considerations

All security testing and threat analysis will be conducted **strictly within the isolated VirtualBox host-only network**. No scanning, intrusion detection, or penetration testing will be performed on external or university networks. This ensures compliance with ethical guidelines and responsible security practices.

---

5. Reflection

Key Planning Outcomes

- Designed a structured and measurable performance testing methodology
- Defined a comprehensive and realistic security baseline
- Identified common Linux server threats and appropriate mitigations
- Reinforced the importance of planning security controls before implementation

Anticipated Challenges

- Balancing security controls with system performance
- Ensuring accurate and repeatable performance measurements
- Avoiding misconfiguration during security hardening

Learning Outcomes Achieved

- ✓Understanding operating system security principles

- ✓Planning performance measurement strategies
- ✓Applying threat modelling techniques
- ✓Preparing for secure system hardening

This week strongly supports **Learning Outcome 3** by analysing security threats and mitigations, and **Learning Outcome 5** by considering security, performance, and system design trade-offs.

References

- Ubuntu Security Documentation. Available: https://ubuntu.com/security (Accessed: 2025)
- OpenSSH Manual Pages. Available: https://man.openbsd.org/ssh (Accessed: 2025)
- NIST SP 800-53 Security Controls. Available: https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final (Accessed: 2025)