Week 1 — System Architecture & Environment Planning

Navigation: Home | **Week 1** | Week 2 →

Overview

Week 1 focuses on planning and validating the system architecture and deployment environment for a Linux server prior to production configuration. The activities in this phase establish a clear, reproducible, and secure-by-default foundation that supports later security hardening, performance monitoring, and analytical evaluation. The server is intentionally deployed **headless** (without a graphical interface) to reduce resource overhead and to enforce professional, command-line–only administration practices aligned with industry standards.

Objectives

- Design a dual-system architecture (administration workstation + headless server)
- Select and justify the host operating system and virtualisation platform
- Define network topology and IP addressing
- Plan directory structure and GitHub repository organisation
- Prepare the environment for secure remote management via SSH

Deliverables

- System architecture diagram
- Virtualisation and network design documentation
- Planned directory and GitHub repository structure
- Evidence of host and virtual machine environment setup

---

1. System Architecture Design

1.1 High-Level Architecture

Architecture Description: - The host machine runs **macOS** and acts as the administration workstation. - Virtualisation is provided using **Oracle VirtualBox**. - **Ubuntu Server LTS** runs as a guest virtual machine in headless mode. - All system management is performed remotely via **SSH (port 22)**. - An **isolated host-only**

network is used to minimise attack surface and ensure ethical security testing.

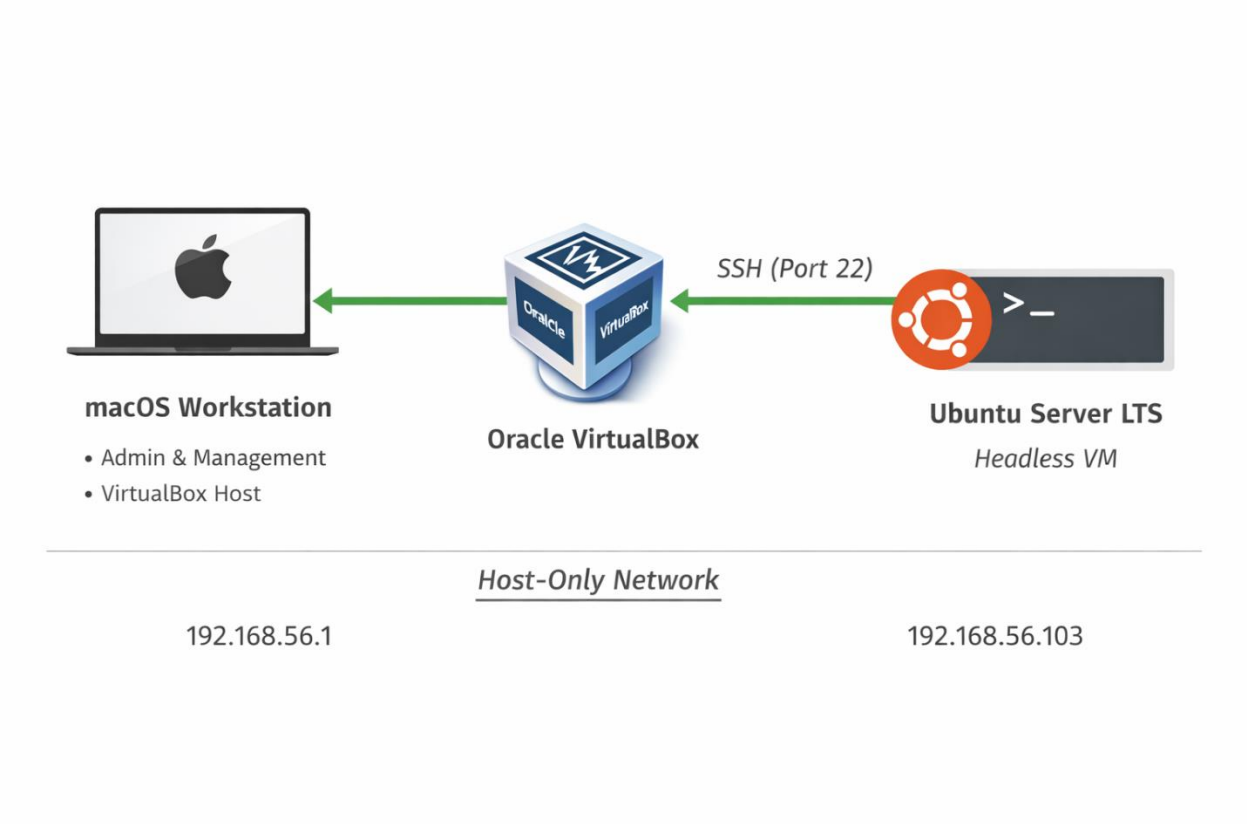

Figure W1-1: High-level system architecture showing host workstation, virtual machine, and secure management access.

1.2 Host Environment

Host System Details: - **Operating System:** macOS - **Role:** Development workstation and management console

Responsibilities: - Initiating SSH connections to the server - Executing remote monitoring scripts - Capturing command-line evidence - Managing the GitHub repository and coursework documentation

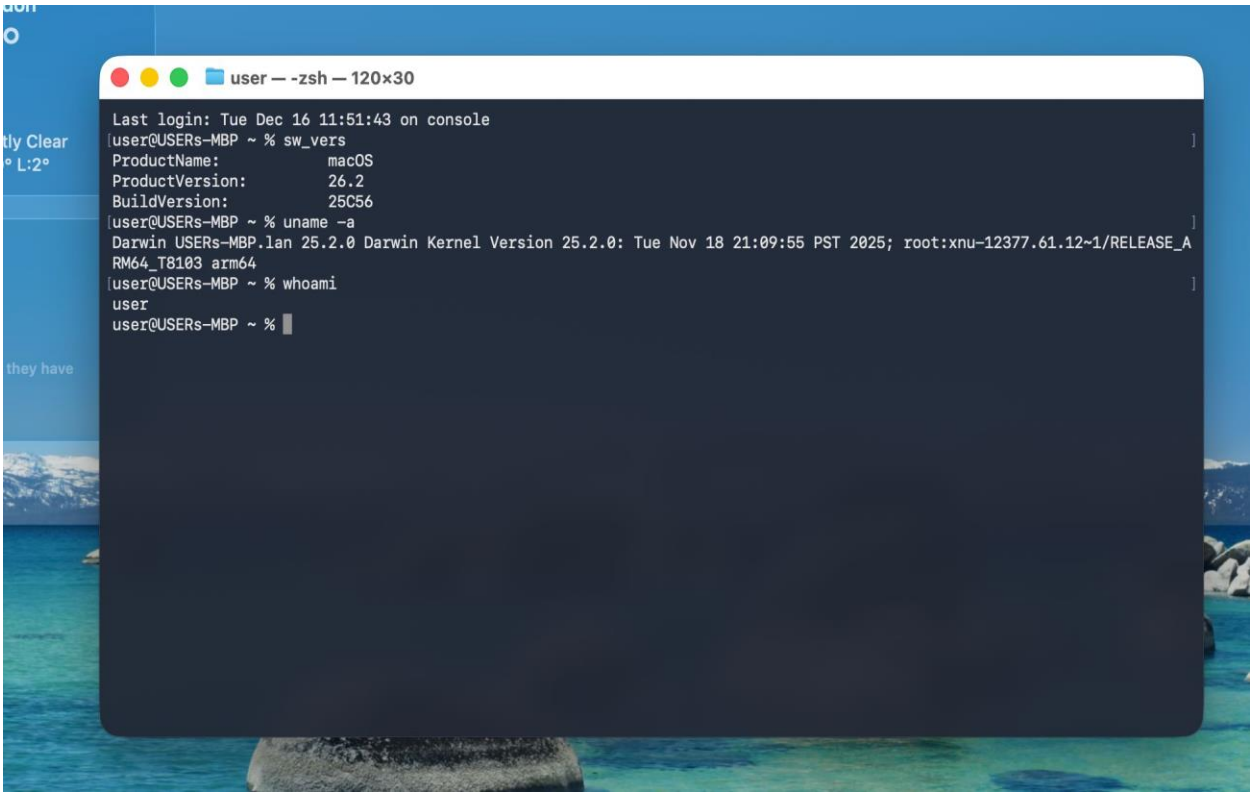

Figure W1-2: Host system information confirming the development and management environment.

---

2. Virtualisation Platform

2.1 VirtualBox Configuration

Virtualisation Tool: Oracle VirtualBox
Guest Operating System: Ubuntu Server LTS

Planned Virtual Machine Resources: - **CPU:** 2 vCPUs - **Memory:** 2–4 GB RAM - **Storage:** 20–40 GB (VDI)

These values were selected to balance performance testing requirements with host system resource constraints, enabling meaningful performance analysis without oversubscribing the host system.

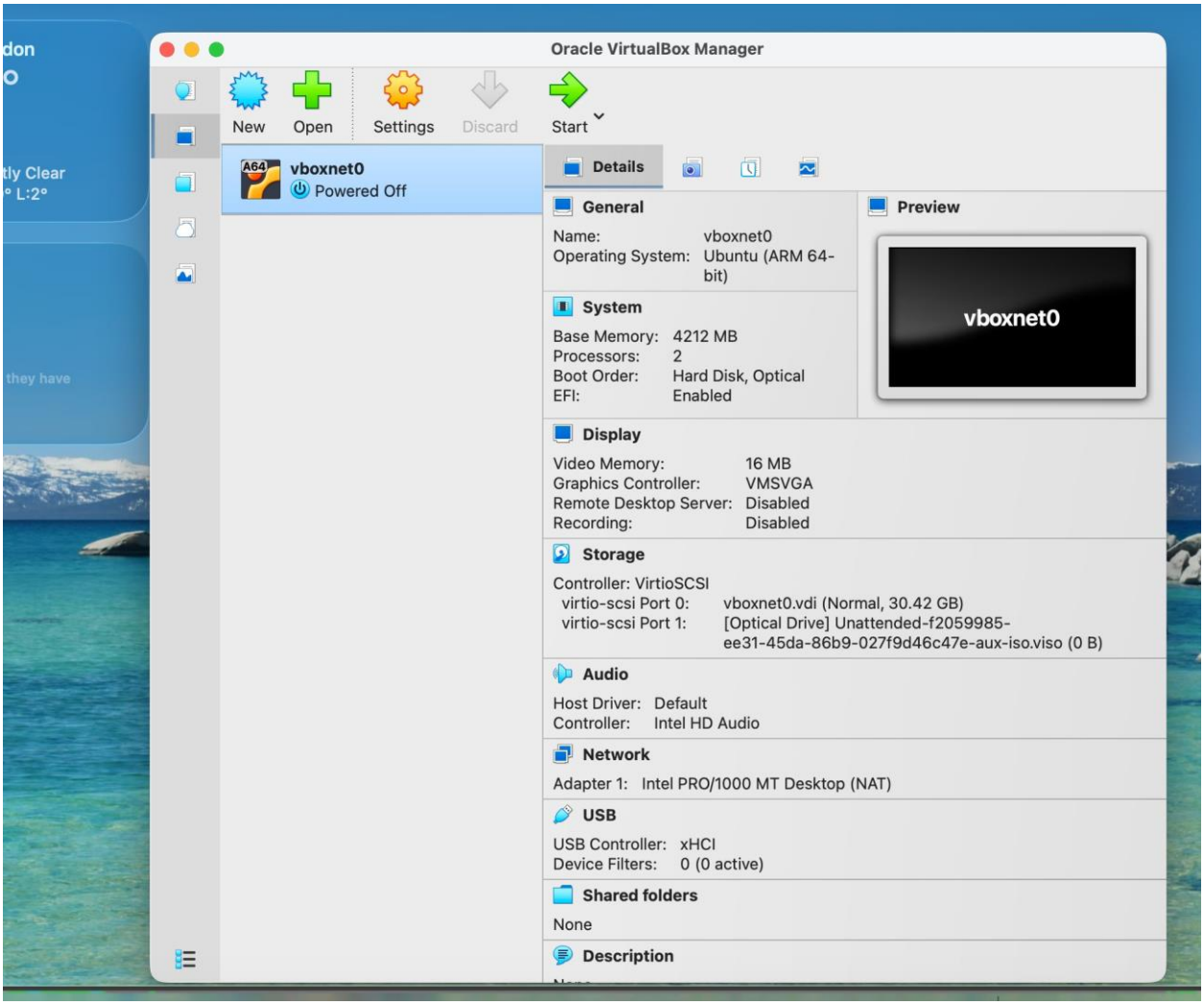

Figure W1-3: VirtualBox VM configuration showing allocated CPU, memory, and storage.

2.2 Guest Operating System Selection

OS Selection Rationale: Ubuntu Server LTS - Long-term security updates and stability - Extensive official documentation and community support - Native integration of **AppArmor** for mandatory access control - Widely used in enterprise and cloud environments - Lightweight and well-suited to headless deployments

This choice aligns with professional infrastructure practices and supports later implementation of security hardening and performance optimisation.

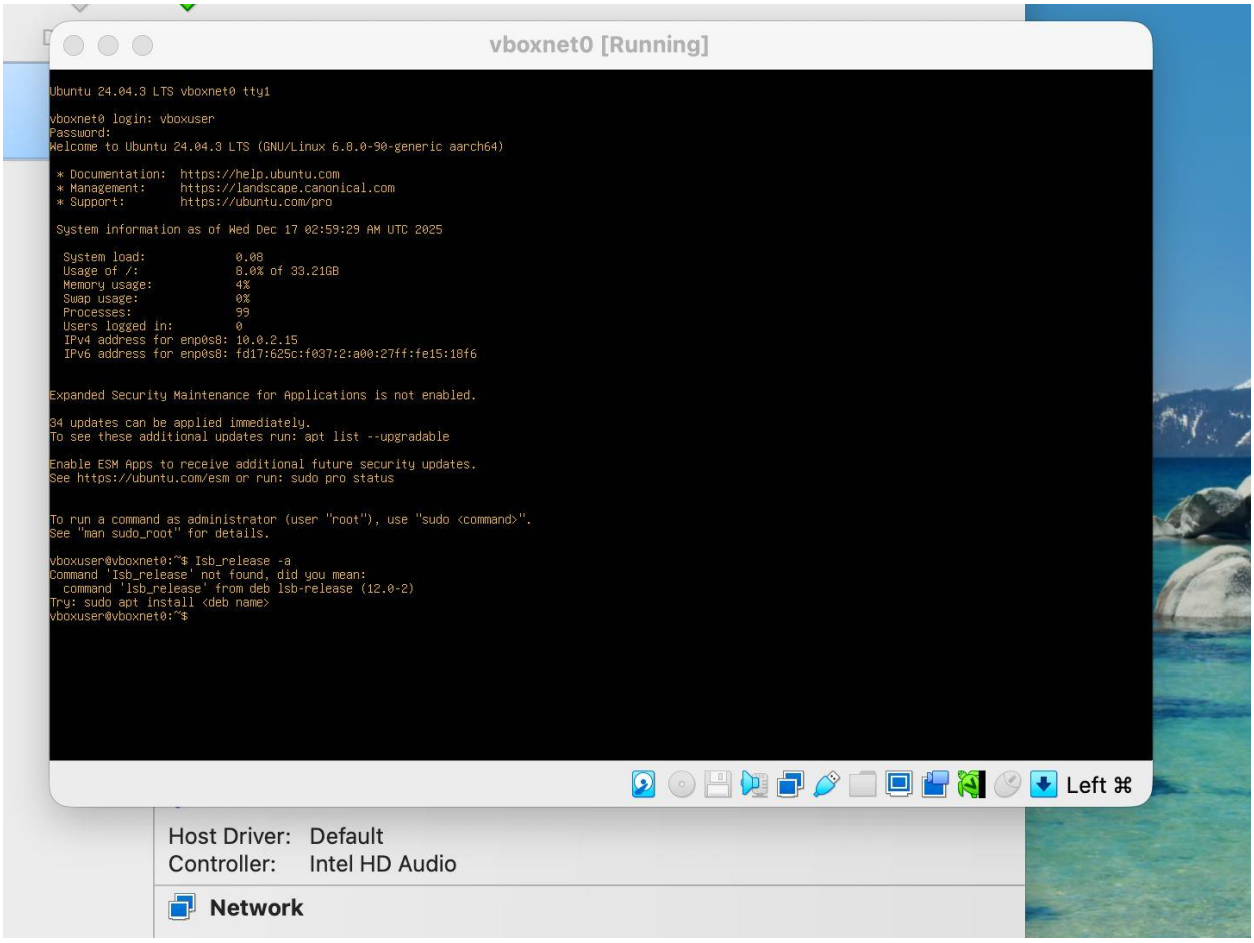

Figure W1-4: Ubuntu Server successfully installed and booted in headless mode.

---

3. Network Design

3.1 Network Topology

Network Mode: Host-only Adapter

Design Rationale: - Fully isolated from public and university networks - Minimises external attack surface - Enables secure server management via SSH - Provides predictable and stable IP addressing - Suitable for controlled security testing and demonstrations

Predictable IP addressing simplifies firewall rule definition and SSH access control implemented in later phases.

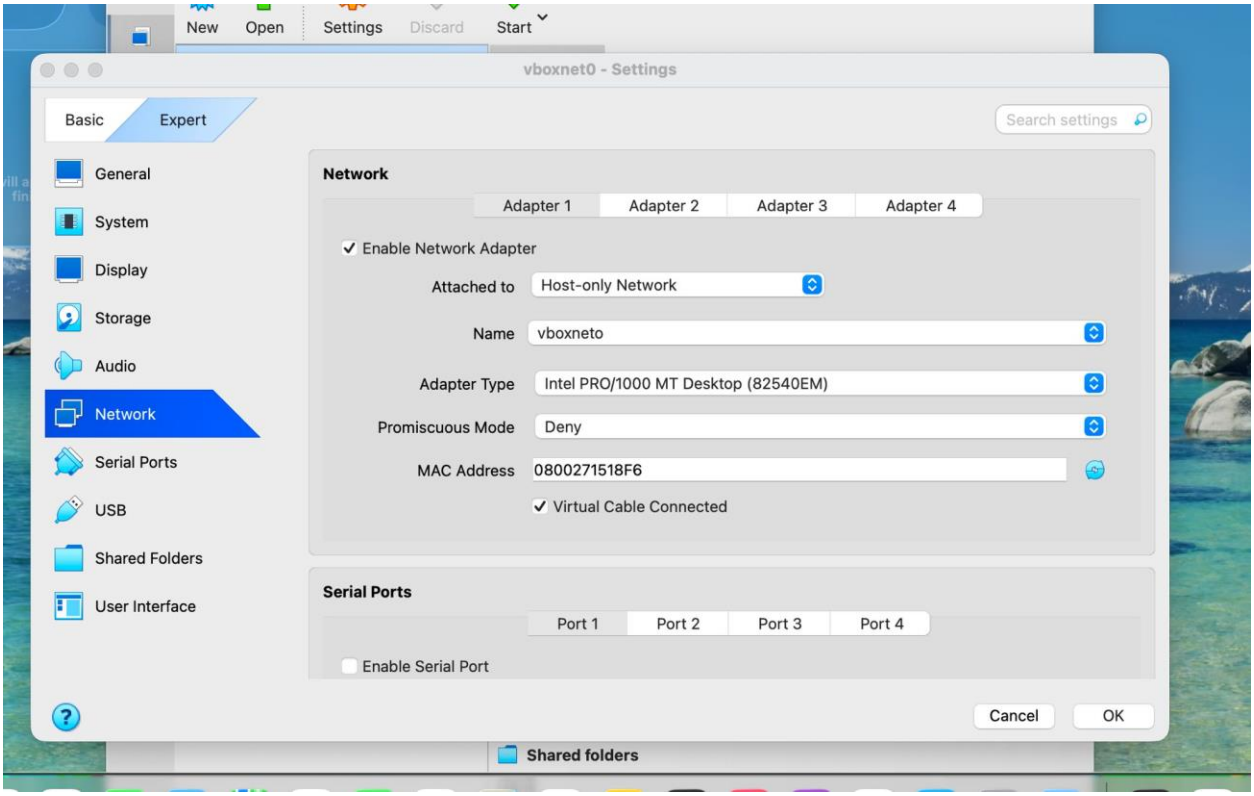

Figure W1-5: Host-only network configuration in VirtualBox.

3.2 IP Addressing Plan

Component	IP Address
Host (macOS)	192.168.56.1
Ubuntu Server	192.168.56.103

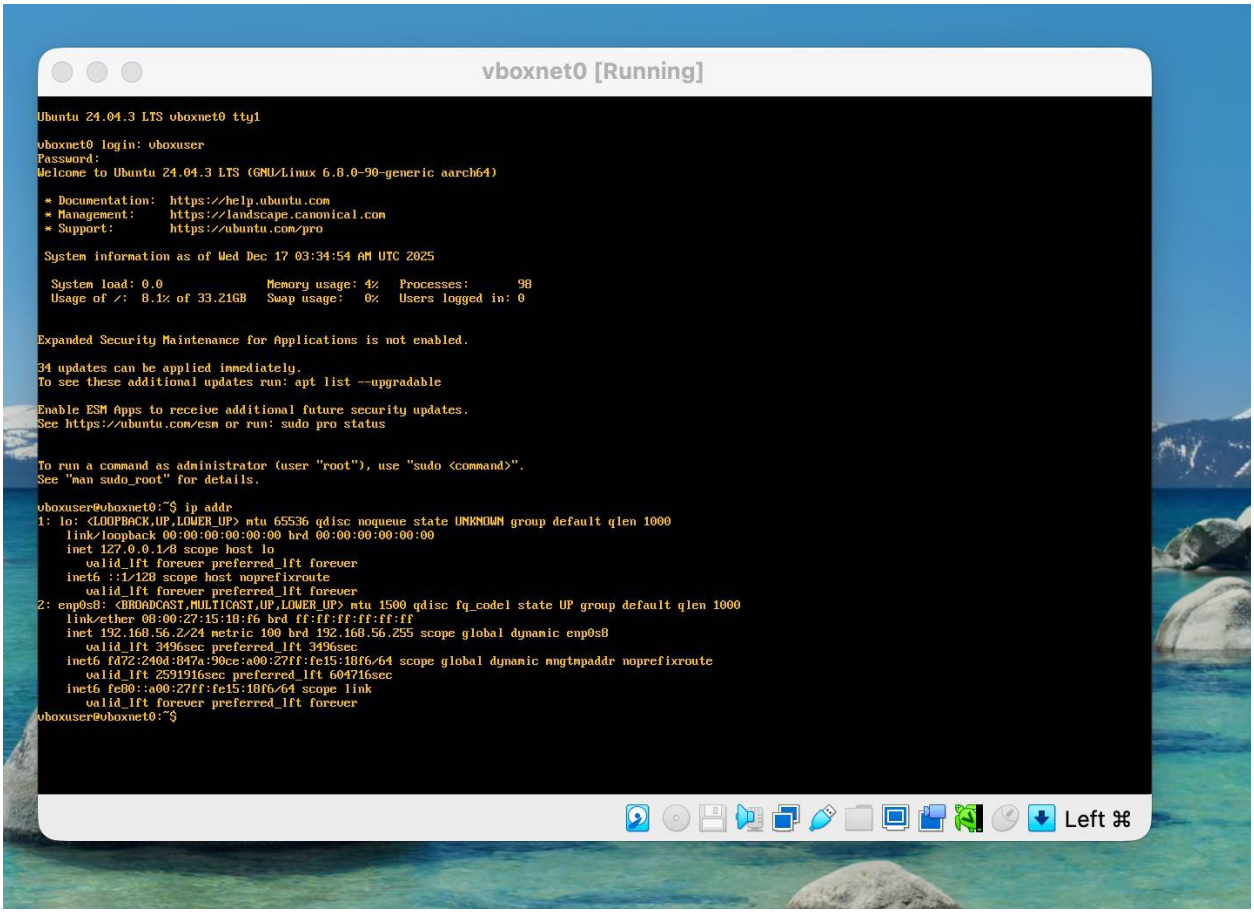

Figure W1-6: IP configuration verification on the Ubuntu Server.

---

4. Directory & Repository Structure

4.1 Planned Server Directory Structure

```
/opt/project/
├── scripts/
├── data/
├── logs/
└── backups/
```

Purpose: - `scripts/` – Monitoring, automation, and verification scripts - `data/` – CSV outputs and performance metrics - `logs/` – System and application log files - `backups/` – Configuration backups and snapshots

This structure promotes clarity, maintainability, and scalability, mirroring professional server organisation practices.

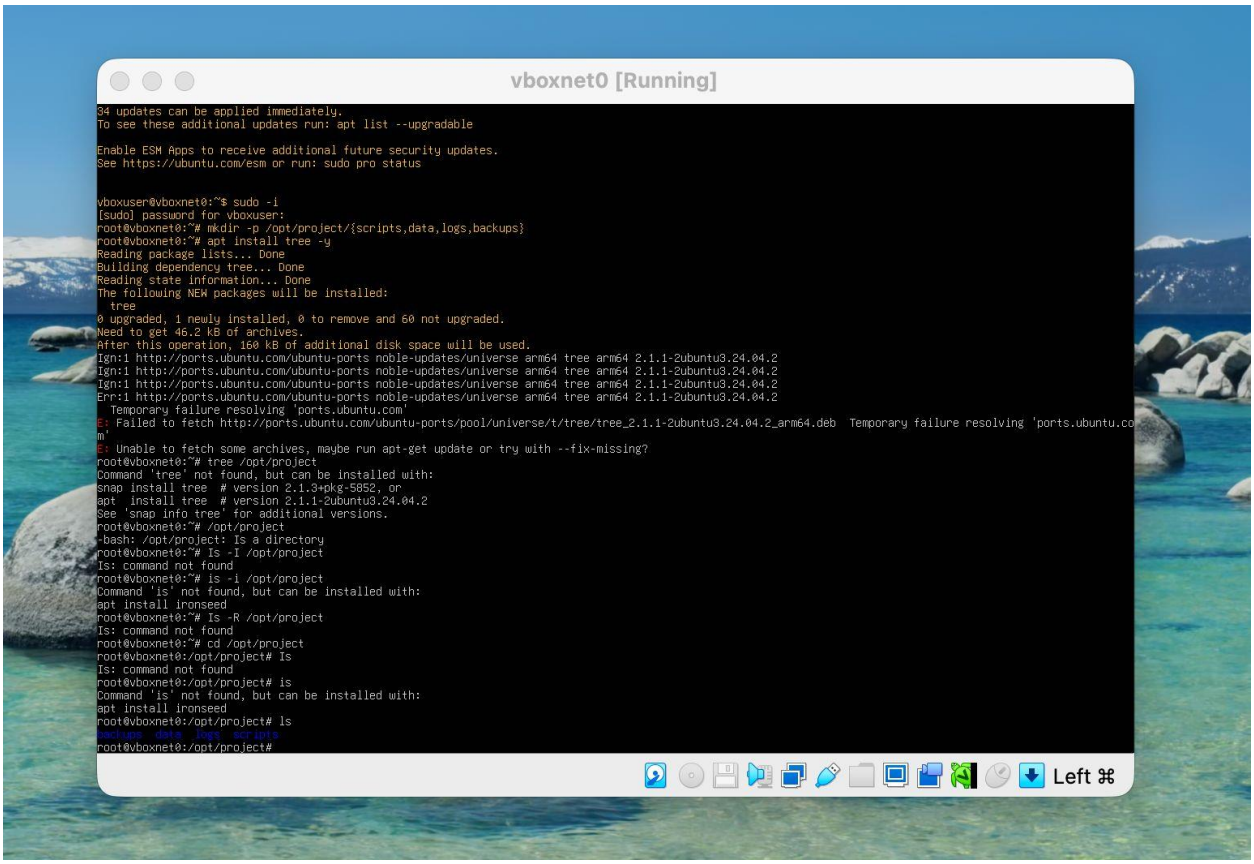

Figure W1-7: Planned directory structure created on the server.

4.2 GitHub Repository Structure

```
repo-root/
├── README.md
├── docs/
│   ├── week1-system-planning.md
│   ├── week2-security-planning.md
│   └── week3-application-selection.md
├── screenshots/
│   ├── week1/
│   └── week2/
└── scripts/
```

This structure ensures clear separation of documentation, evidence, and scripts, supporting professional version control workflows and progressive coursework development.

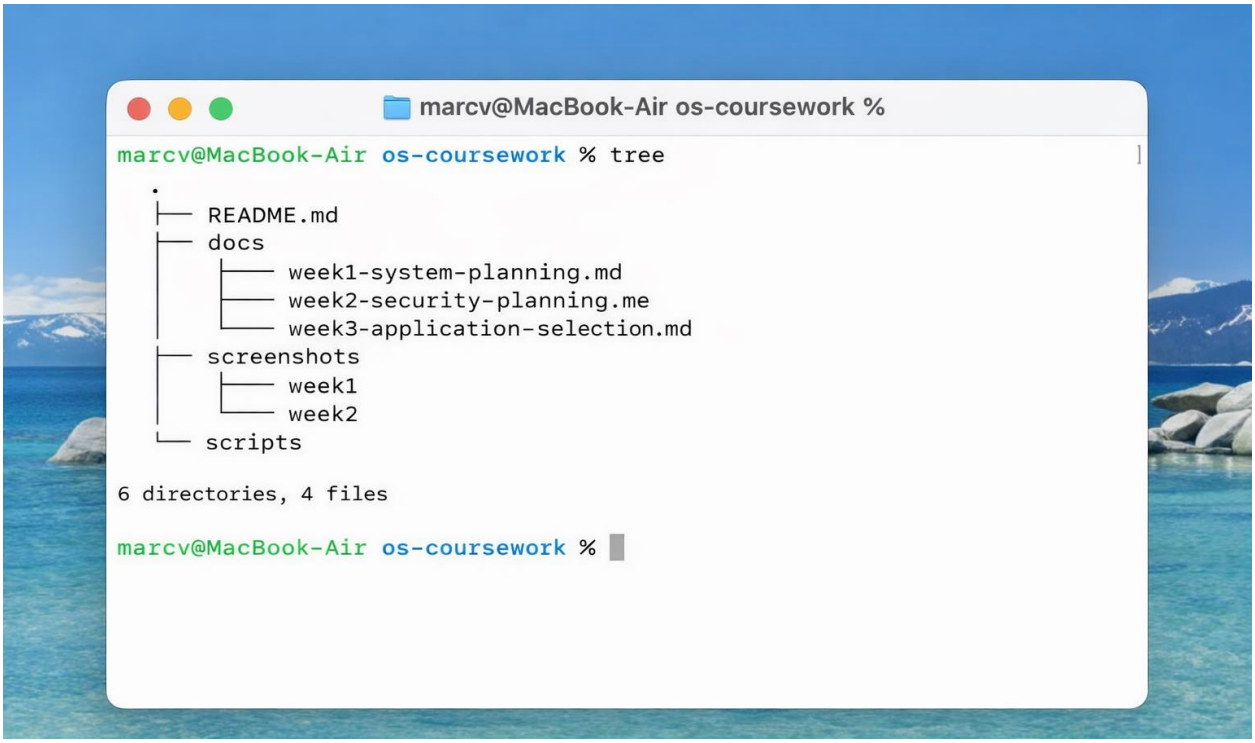

Figure W1-8: GitHub repository structure prepared for documentation and evidence.

---

5. Remote Management Plan

SSH Access Strategy: - SSH enabled on Ubuntu Server during installation - Key-based authentication planned for implementation in Week 2 - Access to be restricted to the host IP address via firewall rules

SSH will be used for: - System configuration - Performance monitoring - Security auditing - Evidence collection

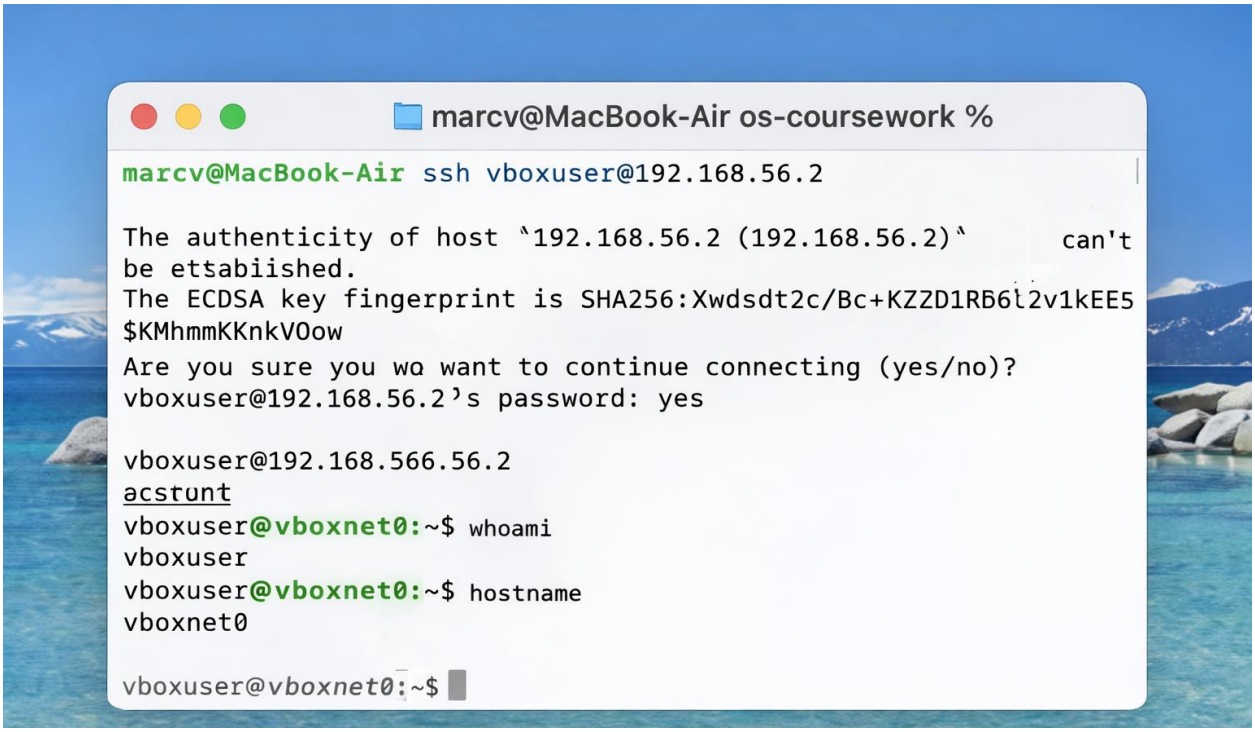

Figure W1-9: Initial SSH login from host to Ubuntu Server.

---

6. System Specification Verification (CLI Evidence)

The following commands were executed on the Ubuntu Server **via SSH** to verify system specifications and confirm correct deployment:

```
uname -a
free -h
df -h
ip addr
lsb_release -a
```

Command Purpose: - `uname -a` confirms the running Linux kernel version and system architecture - `free -h` verifies available and used memory in human-readable format - `df -h` confirms disk capacity and usage - `ip addr` validates network interface configuration and assigned IP address - `lsb_release -a` confirms the Ubuntu Server distribution and release version

These commands collectively demonstrate successful installation, correct resource allocation, and functional networking.

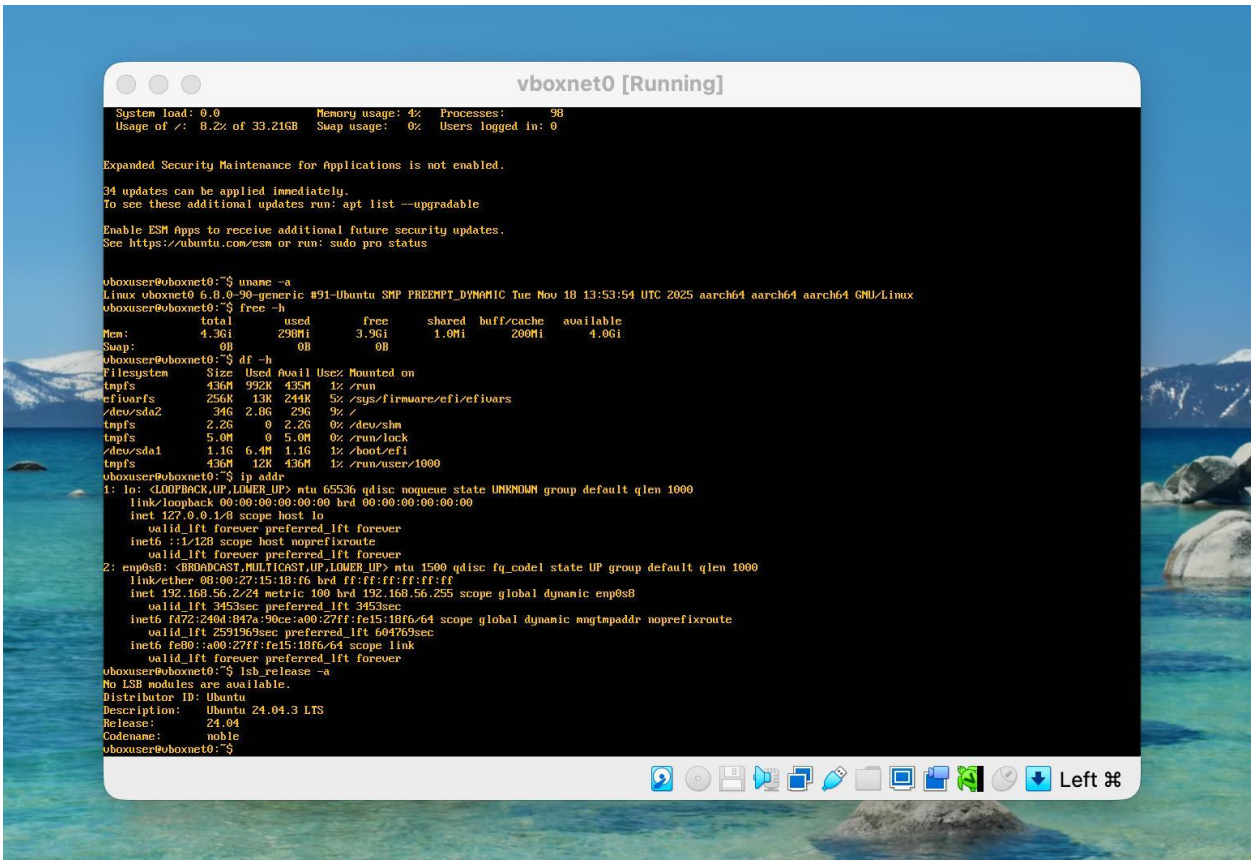

Figure W1-10: Command-line output showing OS, memory, disk, and network configuration.

Evidence Summary

Evidence	Purpose
System architecture diagram	Demonstrates overall design
VirtualBox VM settings	Validates resource planning
Network configuration	Confirms isolation and security
Directory structure	Shows organisation and foresight
SSH connectivity	Confirms remote manageability
CLI verification	Demonstrates command-line proficiency

Reflection

Key Design Decisions

- **Virtualisation:** VirtualBox selected for stability and compatibility with macOS
- **Networking:** Host-only networking chosen to reduce attack surface and enable ethical testing

- **Operating System:** Ubuntu Server LTS selected for security longevity and enterprise relevance
- **Headless Deployment:** Improves efficiency and enforces CLI-based administration

Anticipated Challenges

- Managing host resource constraints during performance testing
- Maintaining consistent IP addressing
- Ensuring comprehensive, well-organised evidence across weeks

Learning Outcomes Achieved

- ✓Infrastructure planning prior to deployment
- ✓Understanding virtualisation and networking concepts
- ✓Designing secure-by-default environments
- ✓Structuring professional technical documentation

This week contributes directly to **Learning Outcome 4** by developing foundational command-line and system administration skills, and **Learning Outcome 5** by analysing operating system design decisions and trade-offs.

References

[1] Oracle, “Oracle VM VirtualBox User Manual,” 2025. [Online]. Available: https://www.virtualbox.org/manual/. [Accessed: Jan. 2025].
[2] Canonical, “Ubuntu Server Documentation,” 2025. [Online]. Available: https://documentation.ubuntu.com/server/. [Accessed: Jan. 2025].