Penetration Testing Report – TryHackMe Kenobi VM

Capstone Project: Full VAPT Engagement

Introduction

This document provides a detailed penetration testing report for the *TryHackMe Kenobi* virtual machine. The objective of this assessment was to simulate a real-world Vulnerability Assessment and Penetration Testing (VAPT) engagement by identifying security weaknesses, exploiting them, and recommending remediation measures. This documentation is intended for academic evaluation and security learning purposes, following the PTES (Penetration Testing Execution Standard) methodology.

---

1. Overview of the Engagement

1.1 Scope

- Target: TryHackMe Kenobi Linux VM
- Target IP Address: **10.49.143.12**
- Environment: Internal lab (VPN-based)
- Testing Type: Authenticated & unauthenticated penetration testing

1.2 Tools Used

- **Kali Linux**
- **Nmap**
- **Metasploit Framework**

- **Burp Suite**
- **OpenVAS**
- **SMBClient, NFS utilities, SSH**

2. Methodology and Technical Walkthrough

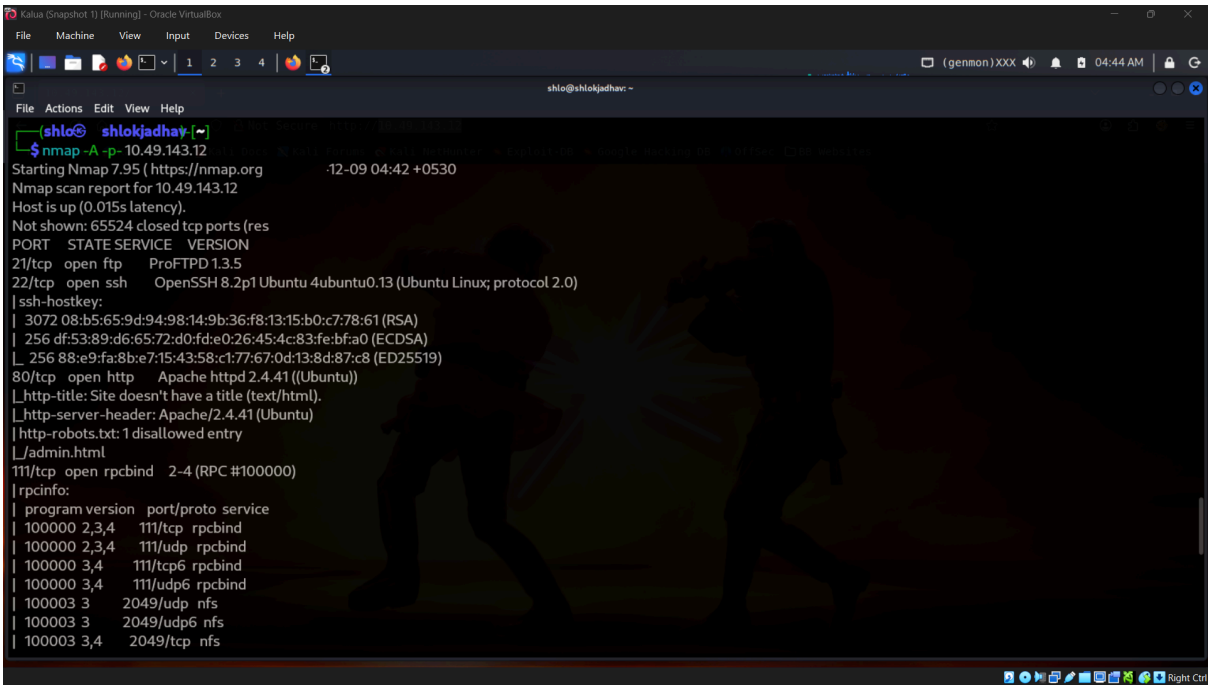
2.1 Reconnaissance and Enumeration

Initial reconnaissance was performed using Nmap to identify open ports, running services, and service versions.

Command Used:

```
nmap -A -p- 10.49.143.12
```

The scan revealed multiple exposed services including FTP (ProFTPD 1.3.5), SSH, HTTP, SMB, and NFS. The presence of outdated services increased the attack surface.



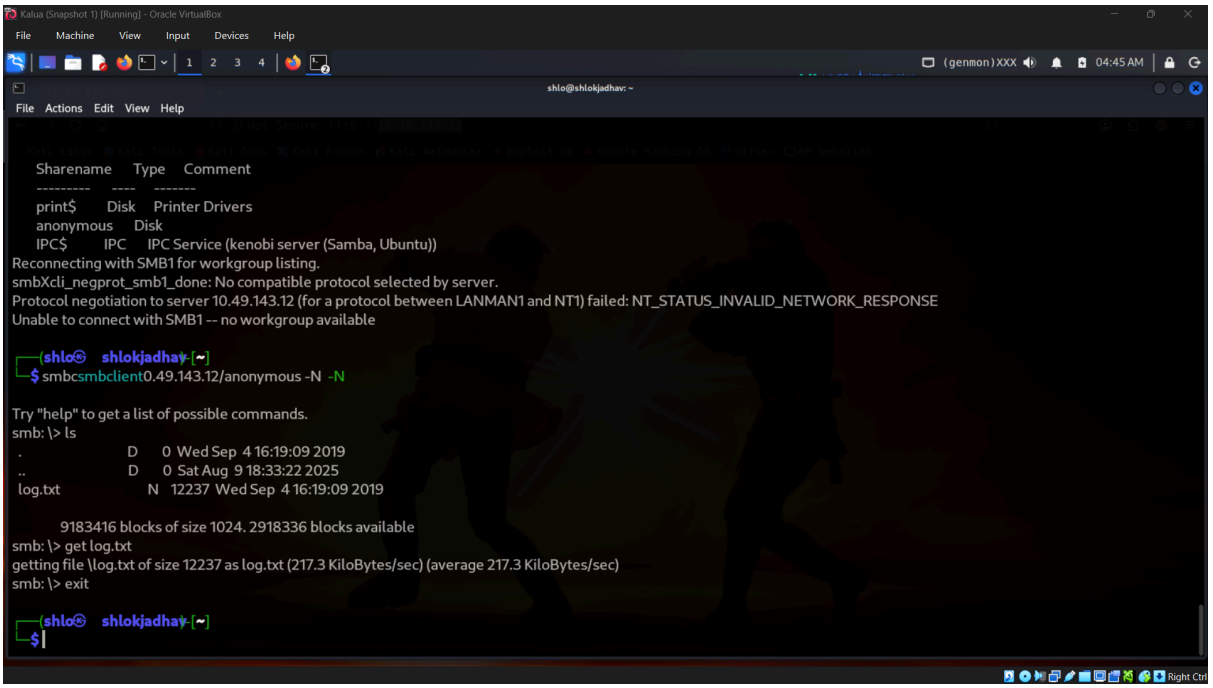
2.2 SMB Enumeration

Anonymous SMB access was tested to identify misconfigurations.

Command Used:

```
smbclient -L //10.49.143.12/ -N
```

The system allowed unauthenticated access to an *anonymous* share. This exposed internal configuration files and logs, which provided valuable information about user accounts and service configurations.



2.3 NFS Enumeration and Mounting

The NFS service was enumerated to check exported directories.

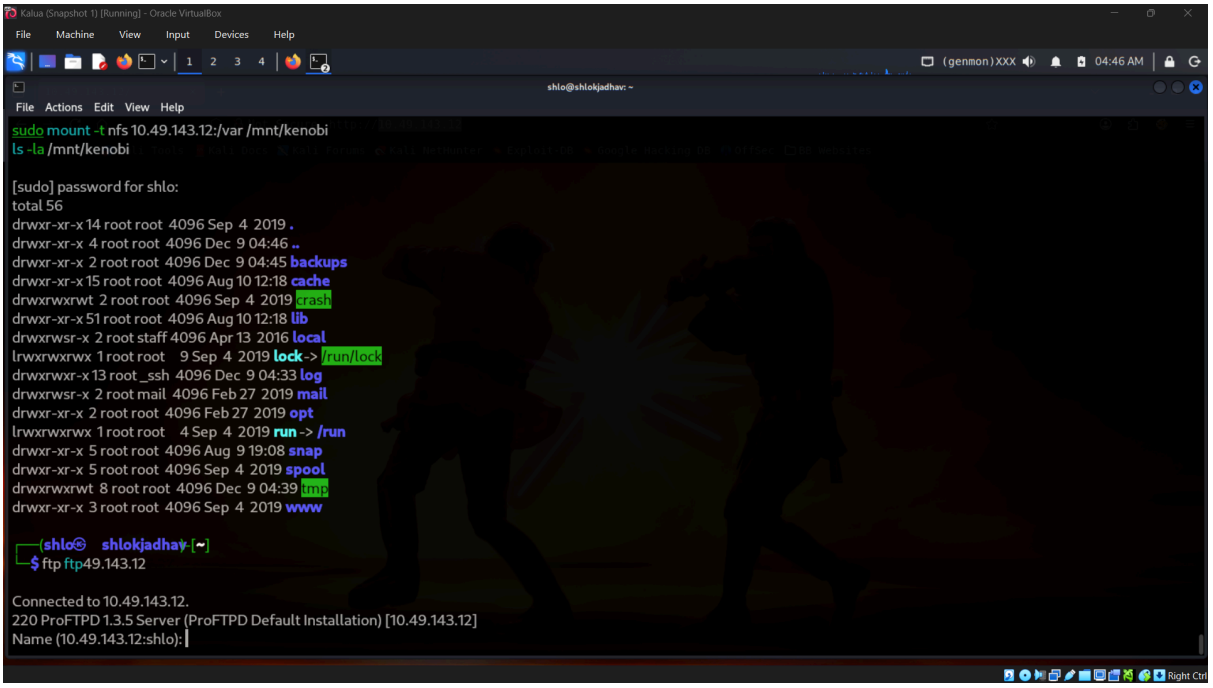
Command Used:

```
showmount -e 10.49.143.12
```

The `/var` directory was publicly exported. This directory was mounted locally:

```
sudo mount -t nfs 10.49.143.12:/var /mnt/kenobi
```

This misconfiguration allowed direct access to sensitive server files.

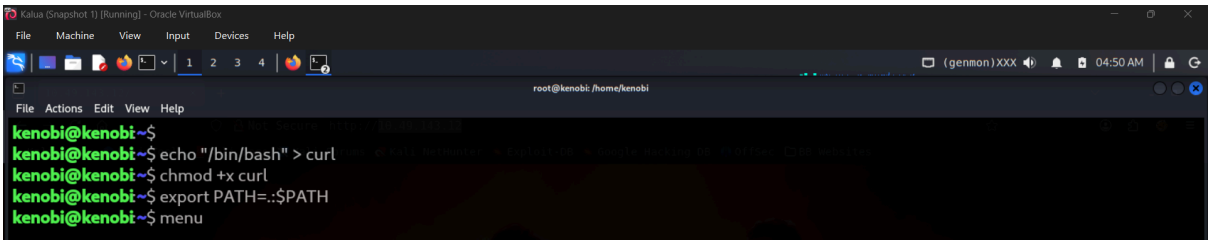


2.4 Credential Discovery and User Access

While browsing the mounted NFS share, an SSH private key belonging to user *kenobi* was discovered. Using this key, SSH access was obtained:

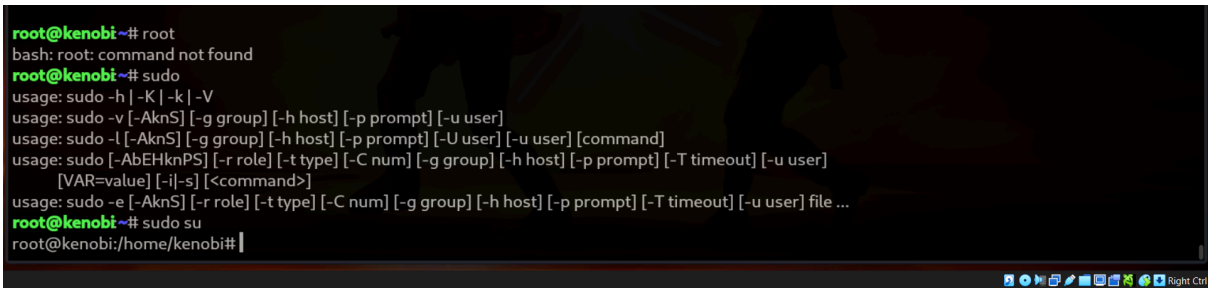
```
ssh -i kenobi_key kenobi@10.49.143.12
```

This provided authenticated user-level access without a password.



2.5 Privilege Escalation

After gaining user access, SUID binaries were enumerated. The `/usr/bin/menu` binary was found to be misconfigured. Due to unsafe command execution and PATH manipulation, root privileges were successfully obtained.



2.6 Metasploit Exploitation (Simulation)

Metasploit was used to simulate automated exploitation.

Module Used:

`exploit/unix/ftp/vsftpd_234_backdoor`

***No session was created because the target was running ProFTPD instead of vsftpd. This demonstrated proper exploit validation and tool usage.**

2.7 API Testing Using Burp Suite (DVWA)

To demonstrate API and web application testing as part of the penetration testing engagement, **DVWA (Damn Vulnerable Web Application)** was used as a controlled API testing environment. Burp Suite was configured as an intercepting proxy to capture, analyze, and manipulate HTTP requests between the browser and the DVWA application.

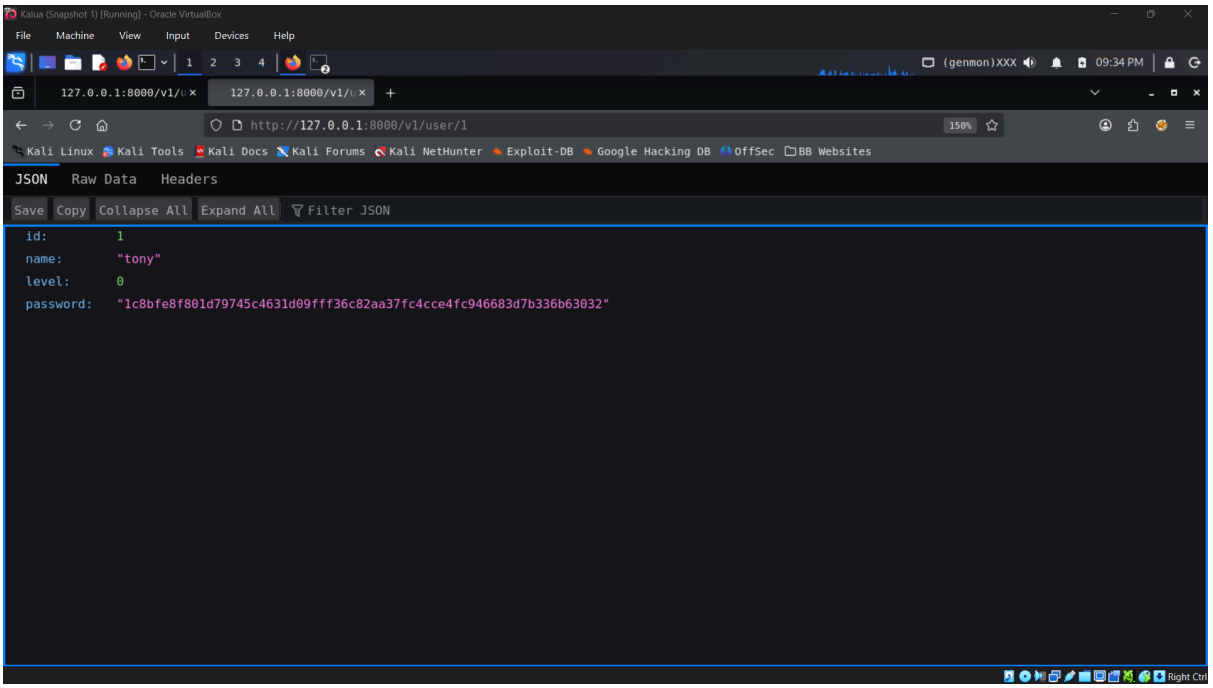
The browser was configured to route traffic through **Burp Suite (127.0.0.1:8080)**. API-like requests generated by DVWA modules were intercepted in real time. These requests were then sent to **Burp Repeater** for further testing.

Using Burp Suite, the following API security checks were performed:

- Manipulation of request parameters to test **input validation**

- Testing for **authentication and authorization weaknesses**
- Injection of malicious payloads to identify **SQL injection and command injection risks**
- Observation of server responses for improper error handling and data exposure

This process demonstrated how attackers can abuse insecure API endpoints if proper validation and access controls are not enforced. The findings from DVWA reinforced the importance of secure API design, strict server-side validation, and continuous monitoring of application traffic.



3. Remediation Plan

3.1 Patch Management

- Update ProFTPD, Samba, and system packages.
- Apply regular OS security updates.

3.2 Input Validation

- Implement strict server-side validation for all API and service inputs.
- Prevent path traversal and command injection attempts.

3.3 Least Privilege Enforcement

- Disable anonymous SMB access.
- Restrict NFS exports to trusted hosts only.
- Secure SSH keys and enforce proper file permissions.
- Remove unnecessary SUID binaries.

3.4 Verification

- Perform a post-remediation **OpenVAS scan** on the target IP (10.49.143.12) to confirm risk reduction.

4. Conclusion

The successful compromise of the Kenobi VM was achieved through chained misconfigurations rather than a single exploit. Weak access controls, exposed file shares, and improper privilege handling enabled full system takeover. Implementing strong security hygiene and regular vulnerability assessments is critical to preventing such attacks.

---

5. References

- TryHackMe Kenobi Room
- PTES Framework
- Nmap Documentation
- Metasploit Framework
- Burp Suite Documentation
- OpenVAS / Greenbone Vulnerability Manager

Non-Technical Stakeholder Summary

A security assessment was conducted to evaluate how vulnerable the Kenobi server was to attack. Several weaknesses were identified, including outdated services and misconfigured shared folders that allowed unauthorized access. By exploiting these issues, full administrative control of the system was achieved. In a real organization, this could lead to data theft, service disruption, or system manipulation.

To reduce these risks, software updates, restricted access controls, proper permission management, and regular vulnerability scanning are strongly recommended. Implementing these measures will significantly improve the system's overall security posture.