

# Vulnerability Report

Object: Group4

Date: May 23, 2025

**CONFIDENTIAL INFORMATION**

Conducted by:

Bernardo Walker Leichtweis

Enzzo Machado Silvino

Cassio Viecei Filho

# Contents

<b>1</b>	<b>Executive Summary</b>	<b>2</b>
<b>2</b>	<b>Introduction</b>	<b>2</b>
2.1	Objective	2
2.2	Methodology	2
<b>3</b>	<b>Test Overview</b>	<b>3</b>
<b>4</b>	<b>Project Scope</b>	<b>3</b>
<b>5</b>	<b>Enumeration</b>	<b>3</b>
<b>6</b>	<b>Vulnerabilities</b>	<b>4</b>
<b>7</b>	<b>Identified Vulnerabilities</b>	<b>5</b>
7.1	Hard-Coded Weak Password	5
7.2	Software Version Exposure	7
7.2.1	SSH (Port 22)	7
7.2.2	HTTP (Port 80 - Apache)	7
<b>8</b>	<b>Conclusion</b>	<b>9</b>
<b>9</b>	<b>Appendices</b>	<b>9</b>
9.1	General Definitions	9
9.2	Severity Levels	10
9.3	Vulnerability Mapping by Asset	10
9.4	Tools Used	10

## 1 Executive Summary

Over a period of 2 days, a penetration test (*Gray Box*) was conducted on the internal infrastructure of **Group4**, focusing on the assets 192.168.2.50 and 192.168.2.34. The test, carried out by Bernardo W. Leichtweis, Enzzo M. Silvino, and Cassio V. Filho, used credentials of a user with minimal privileges (`cebolinha:c3b011nh4`) and followed the **PTES** methodology.

The primary objective was to identify exploitable vulnerabilities that pose a risk to network security. A total of **three vulnerabilities** were found, classified as:

- **1 Critical:** Hard-coded credentials and weak administrator password (`admin:admin`) expose the web application to unauthorized access.
- **2 Low:** Exposure of service versions.

Immediate remediation of the critical vulnerability is recommended, with a focus on strengthening authentication policies and permission management.

## 2 Introduction

This report presents the results of the security assessment on the hosts 192.168.2.50 and 192.168.2.34 of Group4. The test, conducted in *Gray Box* mode with test credentials (`cebolinha:c3b011nh4`), aimed to identify vulnerabilities that compromise the confidentiality, integrity, or availability of the systems.

### 2.1 Objective

To identify vulnerabilities in the specified systems, provide practical recommendations to mitigate risks, and improve Group4's security posture.

### 2.2 Methodology

The assessment followed the **PTES** standard, complemented by the **OWASP Top 10**, and was structured in seven phases:

- **Pre-Engagement Interactions:** Definition of scope and rules.
- **Intelligence Gathering:** Scanning with `nmap`.
- **Threat Modeling:** Analysis of services (*SSH*, *nginx*, *MySQL*).
- **Vulnerability Analysis:** Manual and automated checks.
- **Exploitability:** Validation of vulnerabilities.
- **Post-Exploitation:** Impact analysis.
- **Reporting:** Detailed documentation.

Tools such as `nmap`, Burp Suite, Hydra, and Nessus were used, complemented by manual verifications.

### 3 Test Overview

Category	Quantity
Total Unique Vulnerabilities	3
Critical	1
High	0
Medium	0
Low	2
Informational	0
Zero-Day	0
Easily Exploitable	1

### 4 Project Scope

1. 192.168.2.50
2. 192.168.2.34

### 5 Enumeration

Active services were identified using `nmap` (`nmap -sV -A -Pn -p- -T4 <host>`):

Host	Port/Service	Details
192.168.2.50	22/tcp: ssh	OpenSSH 9.6p1 Ubuntu 3ubuntu13.11
	25/tcp: smtp	Postfix smtpd
	80/tcp: http	Apache httpd 2.4.58
	143/tcp: imap	Dovecot imapd
	993/tcp: ssl/imap	Dovecot imapd
	10050/tcp: tcpwrapped	(Unidentified service)
192.168.2.34	22/tcp: ssh	OpenSSH 9.6p1 Ubuntu 3ubuntu13.11
	25/tcp: smtp	Postfix smtpd
	80/tcp: http	Apache httpd 2.4.58 ((Ubuntu))
	139/tcp: netbios-ssn	Samba smbd 4
	445/tcp: netbios-ssn	Samba smbd 4
	3306/tcp: mysql	MySQL (unauthorized)
	3389/tcp: ms-wbt-server	Microsoft Terminal Service
	5900/tcp: vnc	VNC (protocol 3.8)
	8200/tcp: http	Golang net/http server
	9200/tcp: ssl/http	Amazon OpenSearch REST API (Basic auth)

## 6 Vulnerabilities

### Critical

- **[NOT REMEDIATED] Hard-Coded Weak Password**  
Total affected assets: 1 – Remediated: 0 – Retested: 0 – Not remediated: 1

### Low

- **[NOT REMEDIATED] Server Discloses Software Version (SSH)**  
Total affected assets: 2 – Remediated: 0 – Retested: 0 – Not remediated: 2
- **[NOT REMEDIATED] Server Discloses Software Version (HTTP - 80)**  
Total affected assets: 2 – Remediated: 0 – Retested: 0 – Not remediated: 2

## 7 Identified Vulnerabilities

### 7.1 Hard-Coded Weak Password

**Severity:** Critical

**Affected Asset:** 192.168.2.50

**CVSS v3.1:** 9.1 (AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N)

**Reference:** CWE-798 – Use of Hard-coded Credentials, CWE-521: Weak Password Requirements

**Description:** It was identified that the application has hard-coded administrator credentials directly in the source code, with an extremely weak password (`admin:admin`). This practice represents a serious vulnerability, as it exposes sensitive information and facilitates unauthorized access to the system, especially in environments where the source code is accessible to users or attackers.

**Attack Scenario:** An attacker with access to the server (through prior exploitation or misconfigured permissions) examines the application directory files and finds administrator credentials embedded in the code. With this, they can:

- Gain direct access to the application’s administrative interface (if operational).
- Use the credentials in other interfaces or services reusing the same username/password pair.
- Modify or exfiltrate sensitive data by exploiting administrative privileges.

This type of flaw represents a breach of basic security best practices and a violation of the principles of confidentiality and credential segregation.

**Recommendations:**

- **Remove hard-coded credentials:** Never include usernames or passwords directly in the source code.
- **Use environment variables or vaults:** Store credentials securely outside the code.
- **Enforce strong password policy:** Replace weak passwords with robust combinations of at least 12 characters, including letters, numbers, and symbols.
- **Audit and control access to code:** Ensure only authorized users have access to the application directory.
- **Periodic password rotation:** Implement policies for password expiration and rotation.

**Proof of Concept (PoC):**

1. Access the server or open a session with read permissions in the application directory: `/var/www/html/php`.
2. Locate the file with embedded credentials (`login.php`).
3. The weak and hard-coded password can be used for administrative authentication (if the system becomes operational again).

```
if ($_SERVER["REQUEST_METHOD"] === "POST") {  
    $email = filter_input(INPUT_POST, "email", FILTER_SANITIZE_EMAIL);  
    $senha = $_POST["senha"];  
  
    // Exceção para o usuário admin  
    if ($email === "admin" && $senha === "admin") {  
        $_SESSION["usuario"] = "Administrador"; // Armazena o nome do administra  
dor na sessão  
        echo json_encode(["success" => true, "message" => "Login como administra  
dor bem-sucedido!"]);  
        exit;  
    }  
}
```

Figure 1: Source code snippet exposing hard-coded credentials with a weak password.

## 7.2 Software Version Exposure

**Severity:** [Low](#)

**Affected Assets:** 192.168.2.50, 192.168.2.34

**CVSS v3.1:** 3.7 (AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N)

**References:** CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

### 7.2.1 SSH (Port 22)

**Description:** The OpenSSH services on hosts 192.168.2.50 and 192.168.2.34 reveal the exact version (OpenSSH 9.6p1 Ubuntu 3ubuntu13.11) in the connection banner. This information allows attackers to identify public vulnerabilities associated with the specific version, facilitating targeted attacks.

**Attack Scenario:** An attacker uses `nmap` or `netcat` to capture the SSH banner (SSH-2.0-OpenSSH\_9.6p1 Ubuntu-3ubuntu13.11). With the version identified, they search vulnerability databases (e.g., CVE, Exploit-DB) for known issues, such as buffer overflows or authentication flaws. If an exploitable vulnerability exists, the attacker can:

- Compromise the SSH service, gaining system access.
- Execute remote code or escalate privileges.

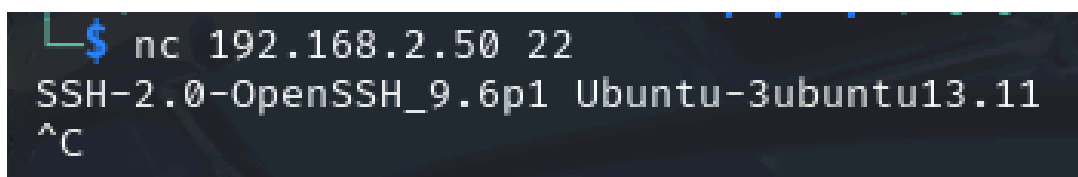
This risk is higher if the OpenSSH version is outdated.

#### Recommendations:

- **Hide banner:** Configure `DebianBanner no` in the `/etc/ssh/sshd_config` file and restart the service (`systemctl restart sshd`).
- **Regular updates:** Keep OpenSSH updated with the latest security patches.
- **Monitoring:** Log SSH connection attempts to detect malicious scans.
- **Firewall:** Restrict access to port 22 to trusted IPs.

#### Proof of Concept (PoC):

1. Run `nc 192.168.2.50 22` to capture the SSH banner.
2. Observe the response: `SSH-2.0-OpenSSH_9.6p1 Ubuntu-3ubuntu13.11`.
3. Search for vulnerabilities associated with the version in Exploit-DB.



```
$ nc 192.168.2.50 22
SSH-2.0-OpenSSH_9.6p1 Ubuntu-3ubuntu13.11
^C
```

Figure 2: SSH banner revealing the OpenSSH version.

### 7.2.2 HTTP (Port 80 - Apache)

**Description:** The Apache server on port 80 of hosts 192.168.2.50 and 192.168.2.34 exposes its version (Apache/2.4.58) in the HTTP `Server` header. This information can be used by attackers to identify known vulnerabilities associated with the specific version.



**Attack Scenario:** An attacker uses `curl -I http://192.168.2.50` to capture the `Server: Apache/2.4.58` header. With the version identified, they query vulnerability databases (e.g., CVE) for public exploits, such as configuration flaws or denial-of-service vulnerabilities. The attacker can:

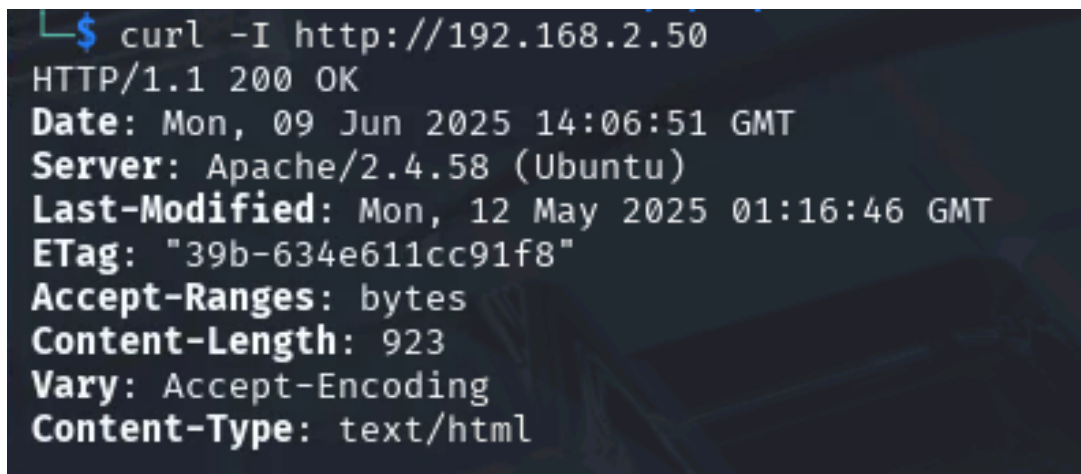
- Exploit a known flaw to compromise the web server.
- Perform additional reconnaissance based on the identified version.

#### Recommendations:

- **Hide version:** Configure the `ServerTokens Prod` and `ServerSignature Off` directives in the `/etc/apache2/apache2.conf` file or equivalent, and restart the service with `systemctl restart apache2`.
- **Updates:** Keep Apache `httpd` updated to the latest stable version available in the distribution's repository.
- **Secure headers:** Add security headers such as `X-Content-Type-Options "nosniff"`, `X-Frame-Options "DENY"`, and `Content-Security-Policy` using the `mod_headers` module.

#### Proof of Concept (PoC):

1. Run `curl -I http://192.168.2.50`.
2. Observe the `Server: Apache/2.4.58` header.



```
$ curl -I http://192.168.2.50
HTTP/1.1 200 OK
Date: Mon, 09 Jun 2025 14:06:51 GMT
Server: Apache/2.4.58 (Ubuntu)
Last-Modified: Mon, 12 May 2025 01:16:46 GMT
ETag: "39b-634e611cc91f8"
Accept-Ranges: bytes
Content-Length: 923
Vary: Accept-Encoding
Content-Type: text/html
```

Figure 3: HTTP header exposing the Apache version.

## 8 Conclusion

The penetration test conducted on the assets `192.168.2.50` and `192.168.2.34` revealed critical vulnerabilities that compromise the security of the infrastructure. Notably, the use of hard-coded credentials with a weak administrative password represents a severe protection failure. Additionally, the exposure of SSH and Apache service versions was identified, facilitating environmental reconnaissance by potential attackers and increasing the attack surface.

These flaws highlight the urgent need to implement strict controls for credential management, concealment of sensitive information, and continuous service updates to mitigate risks and strengthen the network's security posture.

## 9 Appendices

### 9.1 General Definitions

Term	Description
Total Unique Vulnerabilities	Distinct vulnerabilities identified within the scope.
Zero-Day Vulnerability	Flaw unknown to the vendor, exploitable before patches are available.
Easily Exploitable Vulnerability	Flaws detectable by automated tools or with public exploits.
Critical Vulnerability	High impact on confidentiality, integrity, or availability.
High Vulnerability	Requires immediate attention due to potential impact.
Medium Vulnerability	Less urgent but can cause serious issues.
Low Vulnerability	Not imminent but should be mitigated in the long term.

Table 1: Definitions of terms used in the report.

## 9.2 Severity Levels

Level	Description
Critical	CVSS 9.0–10.0: High probability and impact.
High	CVSS 7.0–8.9: Medium to high probability and impact.
Medium	CVSS 4.0–6.9: Low to medium probability or impact.
Low	CVSS 0.1–3.9: Low probability and impact.
Informational	No direct impact but provides useful information.

Table 2: Severity levels of vulnerabilities.

## 9.3 Vulnerability Mapping by Asset

Asset: 192.168.2.50

- **Critical**: Hard-Coded Weak Password.
- **Low**: Service version exposure (OpenSSH, Apache).

Asset: 192.168.2.34

- **Low**: Service version exposure (OpenSSH, Apache).

## 9.4 Tools Used

Purpose	Tool
Network mapping and scanning	Nmap
SMTP server testing	Swaks
Communication and port testing	Netcat
Brute force attacks	Hydra
Vulnerability analysis	Nessus
Web vulnerability analysis	SkipFish, Nikto, Gobuster Burp Suite, XSSStrike, SQLMap
Exploitation and post-exploitation	Metasploit
Privilege escalation	LinPEAS