

Vulnerability Report

Object: Group2

Date: May 23, 2025

CONFIDENTIAL INFORMATION

Conducted by:

Bernardo Walker Leichtweis

Enzzo Machado Silvino

Cassio Viecei Filho

Contents

1	Executive Summary	2
2	Introduction	2
2.1	Objective	2
2.2	Methodology	2
3	Test Overview	3
4	Project Scope	3
5	Enumeration	3
6	Vulnerabilities	4
7	Identified Vulnerabilities	5
7.1	Weak Password Policy	5
7.2	Broken Access Control	7
7.3	Lack of Brute Force Protection	9
7.4	Software Version Exposure	11
7.4.1	SSH (Port 22)	11
7.4.2	HTTP (Port 80 - nginx)	11
7.4.3	HTTP (Port 8080 - WordPress)	13
7.4.4	HTTP (Port 8180 - phpMyAdmin)	15
7.4.5	MySQL (Port 3306)	16
7.5	Information Exposed in robots.txt	17
8	Conclusion	18
9	Appendices	18
9.1	General Definitions	18
9.2	Severity Levels	19
9.3	Vulnerability Mapping by Asset	19
9.4	Tools Used	19

1 Executive Summary

Over a period of 2 days, a penetration test (*Gray Box*) was conducted on the internal infrastructure of **Group2**, focusing on the assets `192.168.1.50` and `192.168.1.34`. 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 **nine vulnerabilities** were found, classified as:

- **1 Critical:** Weak password policy in *phpMyAdmin*, allowing unauthorized access to the administrative panel.
- **1 High:** Broken access control, exposing credentials in configuration files.
- **1 Medium:** Lack of protection against brute force attacks on *SSH*.
- **6 Low/Informational:** Exposure of service versions and information in `robots.txt`.

Immediate remediation of critical and high-severity vulnerabilities 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 `192.168.1.50` and `192.168.1.34` of Group2. 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 Group2'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	9
Critical	1
High	1
Medium	1
Low	5
Informational	1
Zero-Day	0
Easily Exploitable	2

4 Project Scope

1. 192.168.1.50
2. 192.168.1.34

5 Enumeration

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

Host	Port/Service	Details
192.168.1.50	22/tcp: ssh	OpenSSH 9.6p1 Ubuntu 3ubuntu13.11
	25/tcp: smtp	Postfix smtpd
	80/tcp: http	nginx 1.27.5
	3306/tcp: mysql	MySQL 5.7.44
	8080/tcp: http	Apache httpd 2.4.62
	8180/tcp: http	Apache httpd 2.4.62
192.168.1.34	22/tcp: ssh	OpenSSH 9.6p1 Ubuntu 3ubuntu13.11
	139/tcp: netbios-ssn	Samba smbd 4
	445/tcp: netbios-ssn	Samba smbd 4

6 Vulnerabilities

Critical

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

High

- **[NOT REMEDIATED] Broken Access Control**
Total affected assets: 1 – Remediated: 0 – Retested: 0 – Not remediated: 1

Medium

- **[NOT REMEDIATED] Lack of Brute Force Protection**
Total affected assets: 2 – Remediated: 0 – Retested: 0 – Not remediated: 2

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: 1 – Remediated: 0 – Retested: 0 – Not remediated: 1
- **[NOT REMEDIATED] Server Discloses Software Version (HTTP - 8080)**
Total affected assets: 1 – Remediated: 0 – Retested: 0 – Not remediated: 1
- **[NOT REMEDIATED] Server Discloses Software Version (HTTP - 8180)**
Total affected assets: 1 – Remediated: 0 – Retested: 0 – Not remediated: 1
- **[NOT REMEDIATED] Server Discloses Software Version (MySQL)**
Total affected assets: 1 – Remediated: 0 – Retested: 0 – Not remediated: 1

Informational

- **[NOT REMEDIATED] Information Exposed in Robots.txt**
Total affected assets: 1 – Remediated: 0 – Retested: 0 – Not remediated: 1

7 Identified Vulnerabilities

7.1 Weak Password Policy

Severity: Critical

Affected Asset: 192.168.1.50

CVSS v3.1: 9.8 (AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H)

Reference: CWE-521: Weak Password Requirements

Description: The phpMyAdmin service, publicly accessible on port 8180, uses extremely weak credentials (`root:password`). This configuration allows any attacker, without prior knowledge of the system, to access the database administrative panel, compromising the confidentiality, integrity, and availability of stored data. The public exposure of the service amplifies the risk, making it a critical attack vector.

Attack Scenario: A remote attacker, using scanning tools like `nmap`, identifies the phpMyAdmin service on port 8180. By attempting default or widely known credentials, such as `root:password`, the attacker gains immediate access to the administrative panel. With this, they can:

- Exfiltrate sensitive data, such as customer information or stored credentials.
- Modify or delete database tables, causing service disruptions.
- Inject malicious code (e.g., stored XSS) to compromise other connected systems.

This access can serve as an entry point for lateral movements within the network, potentially compromising the entire infrastructure.

Recommendations:

- **Immediate credential change:** Replace `root:password` with a strong password (minimum 12 characters, including uppercase, lowercase, numbers, and symbols).
- **Access restriction:** Configure firewall rules (`iptables`) or a VPN to limit access to phpMyAdmin to authorized IP addresses.
- **Multi-factor authentication (MFA):** Implement MFA using solutions like Google Authenticator or Duo Security for administrative systems.
- **Disable in production:** Disable phpMyAdmin on internet-exposed servers, if possible.
- **Regular auditing:** Establish routines to review and update credentials for critical systems.

Proof of Concept (PoC):

1. Access `http://192.168.1.50:8180` via a browser or `curl`.
2. Enter the credentials `root:password` in the phpMyAdmin login form.
3. Upon successful authentication, gain full access to the panel, allowing viewing, editing, and deleting databases.

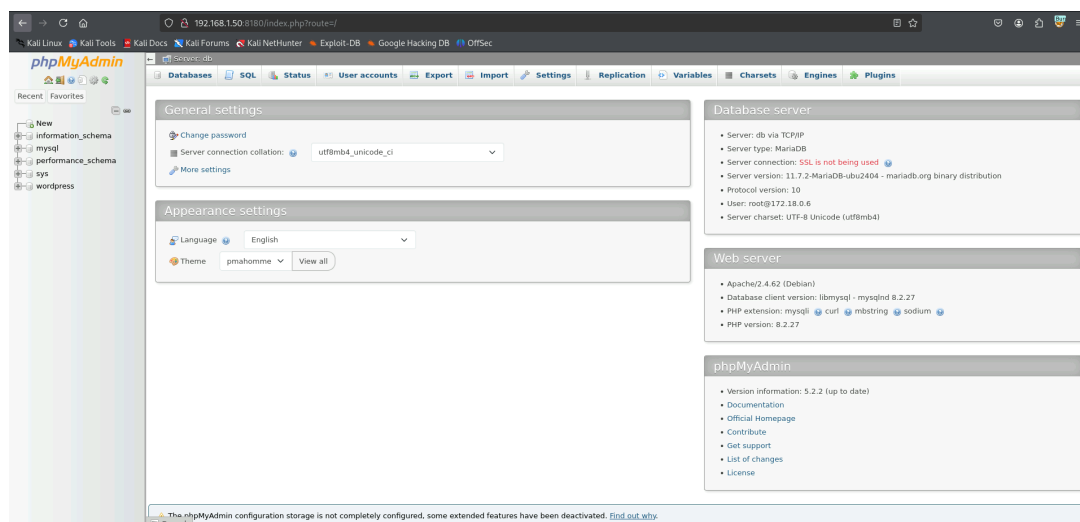


Figure 1: Unauthorized access to phpMyAdmin using default credentials.

7.2 Broken Access Control

Severity: High

Affected Asset: 192.168.1.50

CVSS v3.1: 7.8 (AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N)

References: OWASP A05:2021 - Broken Access Control, CWE-269: Improper Privilege Management

Description: The test user `cebolinha`, with minimal privileges, has read access to sensitive configuration files (`docker-compose.yml`). These files contain plaintext credentials, including `root:password`, which allow access to `phpMyAdmin` on port 8180. This access control failure violates the principle of least privilege, exposing critical information to unauthorized users.

Attack Scenario: An attacker with access to the `cebolinha:c3b011nh4` credentials, obtained through social engineering, account compromise, or password reuse, can:

- List directories in `/home/ExpGrupo2` using `ls -la`.
- Read the `docker-compose.yml` file in the `wordpress` directory, extracting the `root:password` credentials.
- Use these credentials to access `phpMyAdmin` on port 8180, gaining full control over the database.
- Escalate privileges within the network by exploiting other services or connected systems.

This scenario is particularly dangerous in multi-user environments, where a compromised account can lead to a large-scale breach.

Recommendations:

- **Permission review:** Restrict access to the `/home/ExpGrupo2/*` directories to administrative users only, using `chmod 600 docker-compose.yml` and `chown root:root`.
- **Secret management:** Store credentials in secure tools like HashiCorp Vault or AWS Secrets Manager, avoiding plaintext.
- **Least privilege:** Audit and limit permissions for all users, ensuring minimal-privilege accounts cannot access sensitive data.
- **File encryption:** Protect sensitive files with encryption at rest, requiring additional authentication for access.
- **Monitoring:** Configure logs to detect unauthorized access to critical files.

Proof of Concept (PoC):

1. Authenticate on the host 192.168.1.50 via SSH with `cebolinha:c3b011nh4`.
2. Run `ls -la /home/ExpGrupo2/wordpress` to list files.
3. Read the file with `cat /home/ExpGrupo2/wordpress/docker-compose.yml`, identifying `root:password`.
4. Access `http://192.168.1.50:8180` and authenticate in `phpMyAdmin` with the extracted credentials.


```
cebolinha@ene22-ep-pucpr:/home/ExpGrupo2/wordpress$ cat docker-compose.yml
version: "3.6"
services:
  wordpress:
    image: wordpress:latest
    container_name: wordpress
    volumes:
      - ./wp-content:/var/www/html/wp-content
    environment:
      - WORDPRESS_DB_NAME=wordpress
      - WORDPRESS_TABLE_PREFIX=wp_
      - WORDPRESS_DB_HOST=db
      - WORDPRESS_DB_USER=root
      - WORDPRESS_DB_PASSWORD=password
    depends_on:
      - db
      - phpmyadmin
    restart: always
    ports:
      - 8080:80
    networks:
      - expgrupo2_net

  db:
    image: mariadb:latest
    container_name: db
    volumes:
      - db_data:/var/lib/mysql
      # This is optional!!!
      - ./dump.sql:/docker-entrypoint-initdb.d/dump.sql
      # # #
    environment:
      - MYSQL_ROOT_PASSWORD=password
      - MYSQL_USER=root
      - MYSQL_PASSWORD=password
      - MYSQL_DATABASE=wordpress
    restart: always
    networks:
      - expgrupo2_net

  phpmyadmin:
    depends_on:
      - db
    image: phpmyadmin/phpmyadmin:latest
    container_name: phpmyadmin
    restart: always
    ports:
      - 8180:80
    environment:
      PMA_HOST: db
      MYSQL_ROOT_PASSWORD: password
    networks:
      - expgrupo2_net
```

Figure 2: Contents of the docker-compose.yml file with exposed credentials.

7.3 Lack of Brute Force Protection

Severity: Medium

Affected Assets: 192.168.1.50, 192.168.1.34

CVSS v3.1: 6.5 (AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N)

References: CWE-307: Improper Restriction of Excessive Authentication Attempts

Description: The OpenSSH services on hosts 192.168.1.50 and 192.168.1.34 lack mechanisms to protect against brute force attacks, allowing unlimited authentication attempts. This absence of controls, such as IP blocking or progressive delays, facilitates automated attacks that test username and password combinations until valid credentials are found.

Attack Scenario: A remote attacker identifies the SSH service on port 22 using `nmap`. Using tools like `Hydra` or `Medusa`, they perform a brute force attack, testing common password lists against known accounts (e.g., `root`, `admin`, `cebolinha`). Without attempt limitations, the attacker can:

- Compromise accounts with weak passwords, gaining system access.
- Escalate privileges if the compromised account has elevated permissions.
- Establish persistence in the network by installing backdoors or exfiltrating data.

This risk is heightened in internet-exposed networks, where automated bots frequently scan for SSH services.

Recommendations:

- **Implement fail2ban:** Configure `fail2ban` to block IPs after 5 failed login attempts within 10 minutes.
- **Progressive delays:** Adjust `sshd_config` to introduce delays after authentication failures (e.g., `LoginGraceTime 30`).
- **Multi-factor authentication:** Enable MFA for SSH using `Google Authenticator` or certificate-based SSH keys.
- **Strong passwords:** Enforce complex password policies (minimum 12 characters, with diverse characters).
- **Access restriction:** Limit SSH connections to trusted IPs via `iptables` or `tcpwrappers`.

Proof of Concept (PoC):

1. Perform a brute force attack with:
`hydra -l cebolinha -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.50.`
2. Observe that the system allows hundreds of attempts without blocking or delay.
3. Demonstration: Successful authentication with `cebolinha:c3b011nh4` after multiple attempts.

```

$ sudo hydra -l cebolinha -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.50 -V
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal pu
is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-05-23 17:58:38
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwri
.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344400 login tries (l:1/p:14344400), ~896525 tries per task
[DATA] attacking ssh://192.168.1.50:22/
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "123456" - 1 of 14344400 [child 0] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "12345" - 2 of 14344400 [child 1] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "123456789" - 3 of 14344400 [child 2] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "password" - 4 of 14344400 [child 3] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "iloveyou" - 5 of 14344400 [child 4] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "princess" - 6 of 14344400 [child 5] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "1234567" - 7 of 14344400 [child 6] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "rockyou" - 8 of 14344400 [child 7] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "12345678" - 9 of 14344400 [child 8] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "abc123" - 10 of 14344400 [child 9] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "nicole" - 11 of 14344400 [child 10] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "daniel" - 12 of 14344400 [child 11] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "babygirl" - 13 of 14344400 [child 12] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "monkey" - 14 of 14344400 [child 13] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "lovely" - 15 of 14344400 [child 14] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "jessica" - 16 of 14344400 [child 15] (0/0)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "654321" - 17 of 14344400 [child 6] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "michael" - 18 of 14344400 [child 10] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "ashley" - 19 of 14344400 [child 1] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "qwerty" - 20 of 14344400 [child 2] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "111111" - 21 of 14344400 [child 9] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "iloveu" - 22 of 14344400 [child 0] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "000000" - 23 of 14344400 [child 4] (0/2)

```

Figure 3: Brute force attack with Hydra on the SSH service.

```

[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "purple" - 33 of 14344400 [child 1] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "angel" - 34 of 14344400 [child 2] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "jordan" - 35 of 14344400 [child 9] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "liverpool" - 36 of 14344400 [child 0] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "justin" - 37 of 14344400 [child 4] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "loveme" - 38 of 14344400 [child 5] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "fuckyou" - 39 of 14344400 [child 12] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "123123" - 40 of 14344400 [child 13] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "football" - 41 of 14344400 [child 3] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "secret" - 42 of 14344400 [child 7] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "andrea" - 43 of 14344400 [child 8] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "carlos" - 44 of 14344400 [child 15] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "jennifer" - 45 of 14344400 [child 6] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "joshua" - 46 of 14344400 [child 10] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "bubbles" - 47 of 14344400 [child 1] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "1234567890" - 48 of 14344400 [child 2] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "superman" - 49 of 14344400 [child 9] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "hannah" - 50 of 14344400 [child 0] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "amanda" - 51 of 14344400 [child 4] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "loveyou" - 52 of 14344400 [child 5] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "pretty" - 53 of 14344400 [child 12] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "basketball" - 54 of 14344400 [child 13] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "andrew" - 55 of 14344400 [child 3] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "angels" - 56 of 14344400 [child 7] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "tweety" - 57 of 14344400 [child 8] (0/2)
[ATTEMPT] target 192.168.1.50 - login "cebolinha" - pass "c3b01nh4" - 58 of 14344400 [child 15] (0/2)
[22][ssh] host: 192.168.1.50 login: cebolinha password: c3b01nh4
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 2 final worker threads did not complete until end.
[ERROR] 2 targets did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-05-23 17:59:01

```

Figure 4: Successful brute force attack.

7.4 Software Version Exposure

Severity: [Low](#)

Affected Assets: 192.168.1.50, 192.168.1.34

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

7.4.1 SSH (Port 22)

Description: The OpenSSH services on hosts 192.168.1.50 and 192.168.1.34 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.

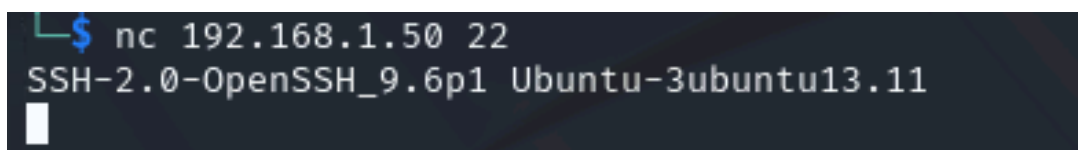
This risk is higher if the OpenSSH version is outdated.

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 192.168.1.50 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 192.168.1.50 22
SSH-2.0-OpenSSH_9.6p1 Ubuntu-3ubuntu13.11
```

Figure 5: SSH banner revealing the OpenSSH version.

7.4.2 HTTP (Port 80 - nginx)

Description: The nginx server on port 80 of host 192.168.1.50 exposes its version (nginx/1.27.5) 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://192.168.1.50` to capture the `Server: nginx/1.27.5` 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:

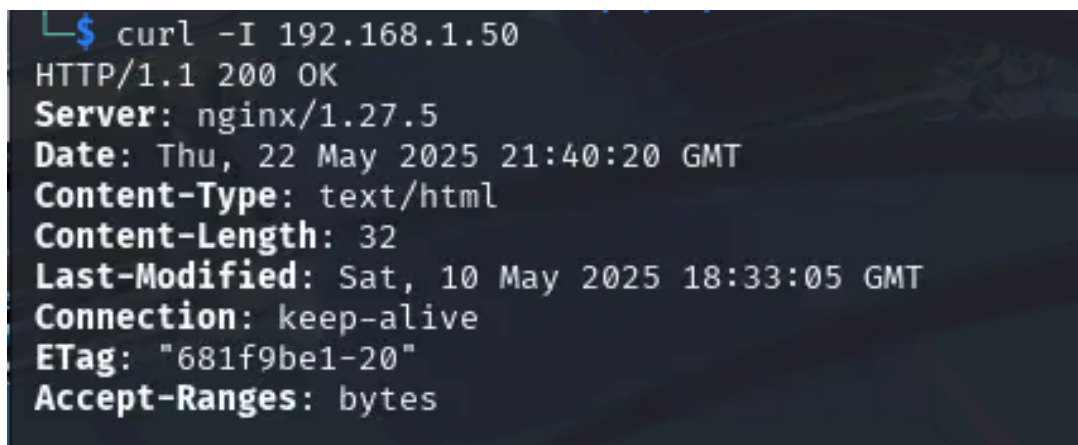
- Exploit a known flaw to compromise the web server.
- Perform additional reconnaissance based on the identified version.

Recommendations:

- **Hide version:** Add `server_tokens off;` to the `/etc/nginx/nginx.conf` file and restart the service (`systemctl restart nginx`).
- **Updates:** Keep `nginx` on the latest version.
- **Secure headers:** Configure additional security headers, such as `X-Content-Type-Options: nosniff`.

Proof of Concept (PoC):

1. Run `curl -I http://192.168.1.50`.
2. Observe the `Server: nginx/1.27.5` header.



```
⌚$ curl -I 192.168.1.50
HTTP/1.1 200 OK
Server: nginx/1.27.5
Date: Thu, 22 May 2025 21:40:20 GMT
Content-Type: text/html
Content-Length: 32
Last-Modified: Sat, 10 May 2025 18:33:05 GMT
Connection: keep-alive
ETag: "681f9be1-20"
Accept-Ranges: bytes
```

Figure 6: HTTP header exposing the nginx version.

7.4.3 HTTP (Port 8080 - WordPress)

Description: The WordPress application on port 8080 of host 192.168.1.50 exposes its version (WordPress 5.8.3) via the generator meta tag and files like `readme.html`. Additionally, HTTP headers reveal the versions of Apache (2.4.62) and PHP (8.2.28), enabling detailed fingerprinting.

Attack Scenario: An attacker accesses `http://192.168.1.50:8080`, inspects the source code, and identifies `<meta name="generator" content="WordPress 5.8.3" />`. They also capture headers with `curl -I`, obtaining `Server: Apache/2.4.62` and `X-Powered-By: PHP/8.2.28`. With this information, the attacker can:

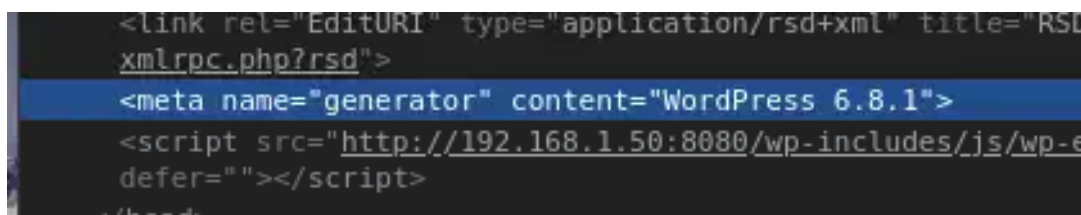
- Search for exploits for known vulnerabilities in WordPress 5.8.3, such as outdated plugins.
- Exploit flaws in Apache or PHP to execute remote code or access sensitive files.
- Combine this information with other vulnerabilities (e.g., brute force) to gain full access.

Recommendations:

- **Hide versions:** Remove the generator meta tag in the WordPress theme (edit `functions.php`) and delete files like `readme.html`.
- **Apache configuration:** Set `ServerSignature Off` and `ServerTokens Prod` in `httpd.conf`.
- **PHP configuration:** Adjust `expose_php = Off` in `php.ini`.
- **Updates:** Keep WordPress, Apache, and PHP updated.

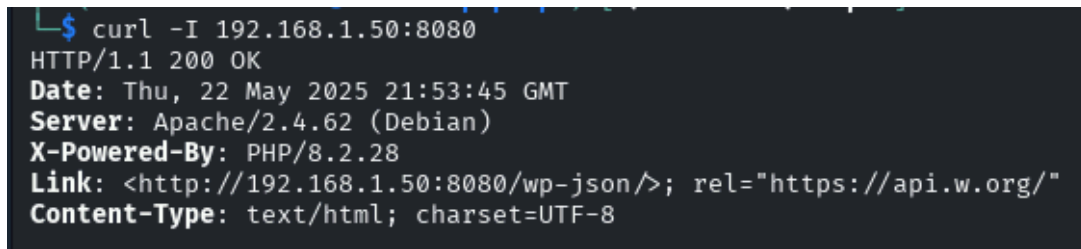
Proof of Concept (PoC):

1. Access `http://192.168.1.50:8080` and inspect the source code to identify `<meta name="generator" content="WordPress 5.8.3" />`.
2. Run `curl -I http://192.168.1.50:8080` to capture `Server: Apache/2.4.62` and `X-Powered-By: PHP/8.2.28`.



```
<link rel="EditURI" type="application/rsd+xml" title="RSD
xmlrpc.php?rsd">
<meta name="generator" content="WordPress 6.8.1">
<script src="http://192.168.1.50:8080/wp-includes/js/wp-e
defer=""></script>
</head>
```

Figure 7: Meta tag exposing the WordPress version.

A terminal window with a dark background and light blue text. The command 'curl -I 192.168.1.50:8080' is entered. The output shows an HTTP 200 OK status and several headers: Date, Server (Apache/2.4.62 (Debian)), X-Powered-By (PHP/8.2.28), Link, and Content-Type (text/html; charset=UTF-8).

```
$ curl -I 192.168.1.50:8080
HTTP/1.1 200 OK
Date: Thu, 22 May 2025 21:53:45 GMT
Server: Apache/2.4.62 (Debian)
X-Powered-By: PHP/8.2.28
Link: <http://192.168.1.50:8080/wp-json/>; rel="https://api.w.org/"
Content-Type: text/html; charset=UTF-8
```

Figure 8: HTTP headers exposing Apache and PHP versions.

7.4.4 HTTP (Port 8180 - phpMyAdmin)

Description: The phpMyAdmin service on port 8180 of host 192.168.1.50 does not directly expose its own version but reveals the versions of Apache (2.4.29) and PHP (7.2.24) in HTTP headers, enabling fingerprinting and identification of known vulnerabilities.

Attack Scenario: An attacker runs `curl -I http://192.168.1.50:8180/phpmyadmin` and captures `Server: Apache/2.4.29` and `X-Powered-By: PHP/7.2.24`. With this information, they search for exploits in NVD or Exploit-DB for specific flaws in these versions, such as remote code execution vulnerabilities in PHP or misconfigurations in Apache. The attacker can:

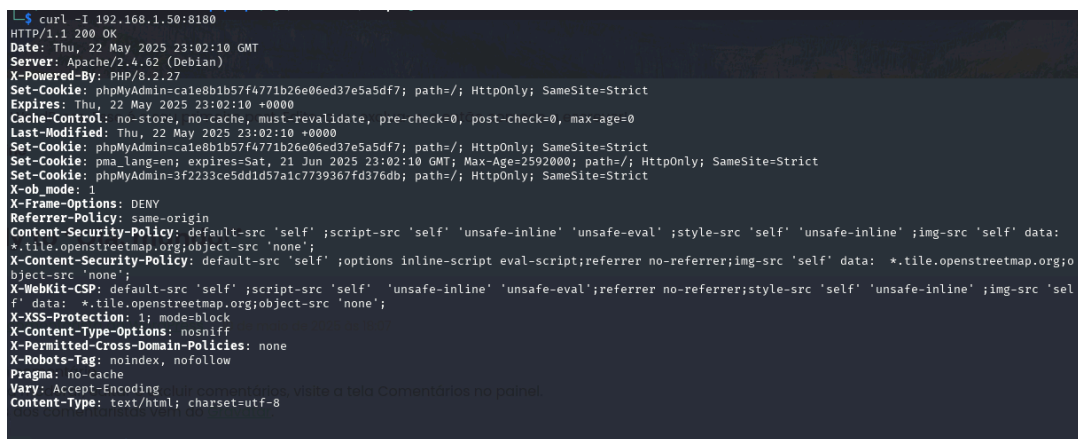
- Exploit a vulnerability to compromise the web server.
- Combine with the weak password policy to access phpMyAdmin.

Recommendations:

- **Hide versions:** Configure `ServerTokens Prod` and `ServerSignature Off` in `httpd.conf` and `expose_php = Off` in `php.ini`.
- **Access restriction:** Limit phpMyAdmin to trusted IPs via `.htaccess` or firewall.
- **Updates:** Keep Apache and PHP updated.

Proof of Concept (PoC):

1. Run `curl -I http://192.168.1.50:8180/phpmyadmin`.
2. Observe headers `Server: Apache/2.4.29` and `X-Powered-By: PHP/7.2.24`.



```
$ curl -I http://192.168.1.50:8180
HTTP/1.1 200 OK
Date: Thu, 22 May 2025 23:02:10 GMT
Server: Apache/2.4.29 (Debian)
X-Powered-By: PHP/7.2.24
Set-Cookie: phpMyAdmin=ca1e8b1b57f4771b26e06ed37e5a5df7; path=/; HttpOnly; SameSite=Strict
Expires: Thu, 22 May 2025 23:02:10 +0000
Cache-Control: no-store, no-cache, must-revalidate, pre-check=0, post-check=0, max-age=0
Last-Modified: Thu, 22 May 2025 23:02:10 +0000
Set-Cookie: phpMyAdmin=ca1e8b1b57f4771b26e06ed37e5a5df7; path=/; HttpOnly; SameSite=Strict
Set-Cookie: pma_lang=en; expires=Sat, 21 Jun 2025 23:02:10 GMT; Max-Age=2592000; path=/; HttpOnly; SameSite=Strict
Set-Cookie: phpMyAdmin=3f2233ce5dd1d57a1c7739367fd376db; path=/; HttpOnly; SameSite=Strict
X-ob_mode: 1
X-Frame-Options: DENY
Referer-Policy: same-origin
Content-Security-Policy: default-src 'self'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; style-src 'self' 'unsafe-inline'; img-src 'self' data: *.tile.openstreetmap.org; object-src 'none';
X-Content-Security-Policy: default-src 'self'; options inline-script eval-script;referrer no-referrer;img-src 'self' data: *.tile.openstreetmap.org;object-src 'none';
X-WebKit-CSP: default-src 'self'; script-src 'self' 'unsafe-inline' 'unsafe-eval';referrer no-referrer;style-src 'self' 'unsafe-inline';img-src 'self' data: *.tile.openstreetmap.org;object-src 'none';
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Robots-Tag: noindex, nofollow
Pragma: no-cache
Vary: Accept-Encoding
Content-Type: text/html; charset=utf-8
```

Figure 9: HTTP headers exposing Apache and PHP versions.

7.4.5 MySQL (Port 3306)

Description: The MySQL service on port 3306 of host 192.168.1.50 reveals its version (MySQL 5.7.44) during the initial TCP handshake. This exposure allows attackers to identify specific vulnerabilities associated with the database version.

Attack Scenario: An attacker uses `nmap` with the `mysql-info` script to capture the MySQL 5.7.44 version during the handshake. With this information, they search NVD for related CVEs, such as authentication or privilege escalation flaws. The attacker can:

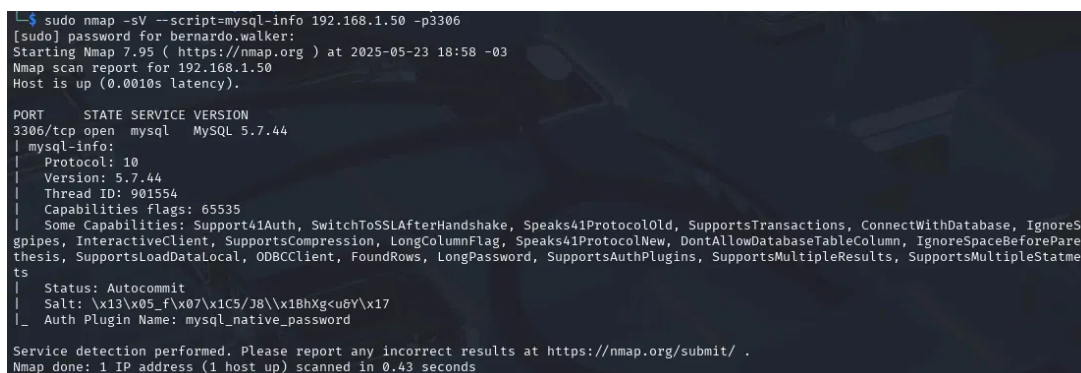
- Exploit a vulnerability to access the database without credentials.
- Combine with the weak password policy to gain full control.

Recommendations:

- **Hide version:** Configure MySQL to not reveal the version in the handshake (requires advanced adjustments or proxies).
- **Access restriction:** Limit port 3306 to trusted IPs via `iptables`.
- **Updates:** Keep MySQL on the latest version.
- **Monitoring:** Log MySQL connection attempts in logs.

Proof of Concept (PoC):

1. Run `nmap -sV --script=mysql-info -p 3306 192.168.1.50`.
2. Observe the response containing MySQL 5.7.44.



```
└─$ sudo nmap -sV --script=mysql-info 192.168.1.50 -p3306
[sudo] password for bernardo.walker:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-23 18:58 -03
Nmap scan report for 192.168.1.50
Host is up (0.0010s latency).

PORT      STATE SERVICE VERSION
3306/tcp  open  mysql   MySQL 5.7.44
| mysql-info:
|   Protocol: 10
|   Version: 5.7.44
|   Thread ID: 901554
|   Capabilities flags: 65535
|   Some Capabilities: Support41Auth, SwitchToSSLAfterHandshake, Speaks41ProtocolOld, SupportsTransactions, ConnectWithDatabase, IgnoreSi
gpipes, InteractiveClient, SupportsCompression, LongColumnFlag, Speaks41ProtocolNew, DontAllowDatabaseTableColumn, IgnoreSpaceBeforeParen
thesis, SupportsLoadDataLocal, ODBCClient, FoundRows, LongPassword, SupportsAuthPlugins, SupportsMultipleResults, SupportsMultipleStatmen
ts
|   Status: Autocommit
|   Salt: \x13\x05_f\x07\x1C5/J8\\x1BhXg<u@Y\x17
|_  Auth Plugin Name: mysql_native_password

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.43 seconds
```

Figure 10: MySQL handshake exposing the service version.

7.5 Information Exposed in robots.txt

Severity: Informational

Affected Asset: 192.168.1.50:8080

CVSS v3.1: 0 (AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N)

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

Description: The `robots.txt` file, publicly accessible on port 8080, lists directories and files that the administrator wishes to hide from search engine crawlers. While not a direct vulnerability, the exposure of these paths can reveal sensitive areas of the site, such as administrative panels or configuration files, which attackers can exploit in conjunction with other flaws.

Attack Scenario: An attacker accesses `http://192.168.1.50:8080/robots.txt` and identifies directories like `/admin` or `/config`. Although these paths may be protected, the exposure facilitates mapping of the site's structure. The attacker can:

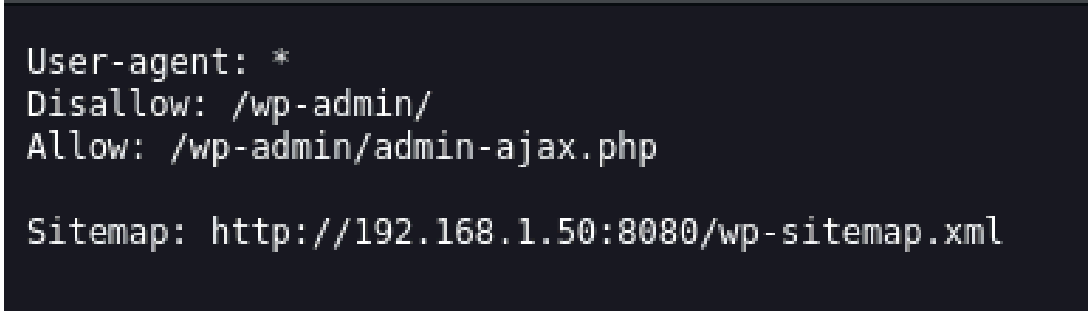
- Test the listed directories for unprotected entry points.
- Combine with other vulnerabilities, such as brute force or broken access control, to access restricted areas.
- Perform social engineering attacks based on the obtained information.

Recommendations:

- **Review robots.txt:** Remove references to sensitive directories or files, keeping only generic entries.
- **Access control:** Protect listed paths with robust authentication (e.g., `.htaccess` or application-level authentication).
- **Monitoring:** Configure logs to detect frequent access to `robots.txt`, indicating malicious scans.
- **Obfuscation:** Consider removing `robots.txt` if not essential or use security tools to block malicious crawlers.

Proof of Concept (PoC):

1. Access `http://192.168.1.50:8080/robots.txt` via a browser or `curl`.
2. Observe the returned content.



```
User-agent: *  
Disallow: /wp-admin/  
Allow: /wp-admin/admin-ajax.php  
  
Sitemap: http://192.168.1.50:8080/wp-sitemap.xml
```

Figure 11: Contents of the robots.txt file revealing directories.

8 Conclusion

The penetration test conducted on Group2's infrastructure revealed a set of vulnerabilities that significantly compromise the security of hosts 192.168.1.50 and 192.168.1.34. The critical vulnerability identified, related to the weak password policy in `phpMyAdmin`, poses an immediate threat, allowing unauthorized access to the database administrative panel. This flaw can result in the exfiltration of sensitive data, manipulation of critical information, or complete service disruption, directly impacting the confidentiality, integrity, and availability of the systems.

Additionally, high and medium-severity vulnerabilities, such as broken access control and the lack of brute force protection on `SSH`, expose the infrastructure to significant risks, including account compromise and privilege escalation. Low-severity issues, such as software version exposure, while less urgent, facilitate attacker enumeration and can be combined with other vulnerabilities to amplify an attack's impact. The exposure of information in `robots.txt`, classified as informational, underscores the need to review configurations to prevent the disclosure of sensitive data.

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: 192.168.1.50

- **Critical**: Weak password policy (phpMyAdmin).
- **High**: Broken access control (/home/ExpGrupo2/).
- **Medium**: Lack of brute force protection (SSH).
- **Low**: Version exposure (OpenSSH, nginx, MySQL, Apache, PHP).
- **Informational**: Exposure in robots.txt.

Asset: 192.168.1.34

- **Medium**: Lack of brute force protection (SSH).
- **Low**: Version exposure (OpenSSH).

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
WordPress vulnerabilities	WPScan
Web vulnerability analysis	Nikto, Gobuster, Burp Suite, XSSStrike, SQLMap
Exploitation and post-exploitation	Metasploit
Privilege escalation	LinPEAS