Final Engagement

Attack, Defense, and
Analysis of a Vulnerable Network

Table of Contents

This document contains the following resources:

Network Topology
& Critical Vulnerabilities

Network Topology

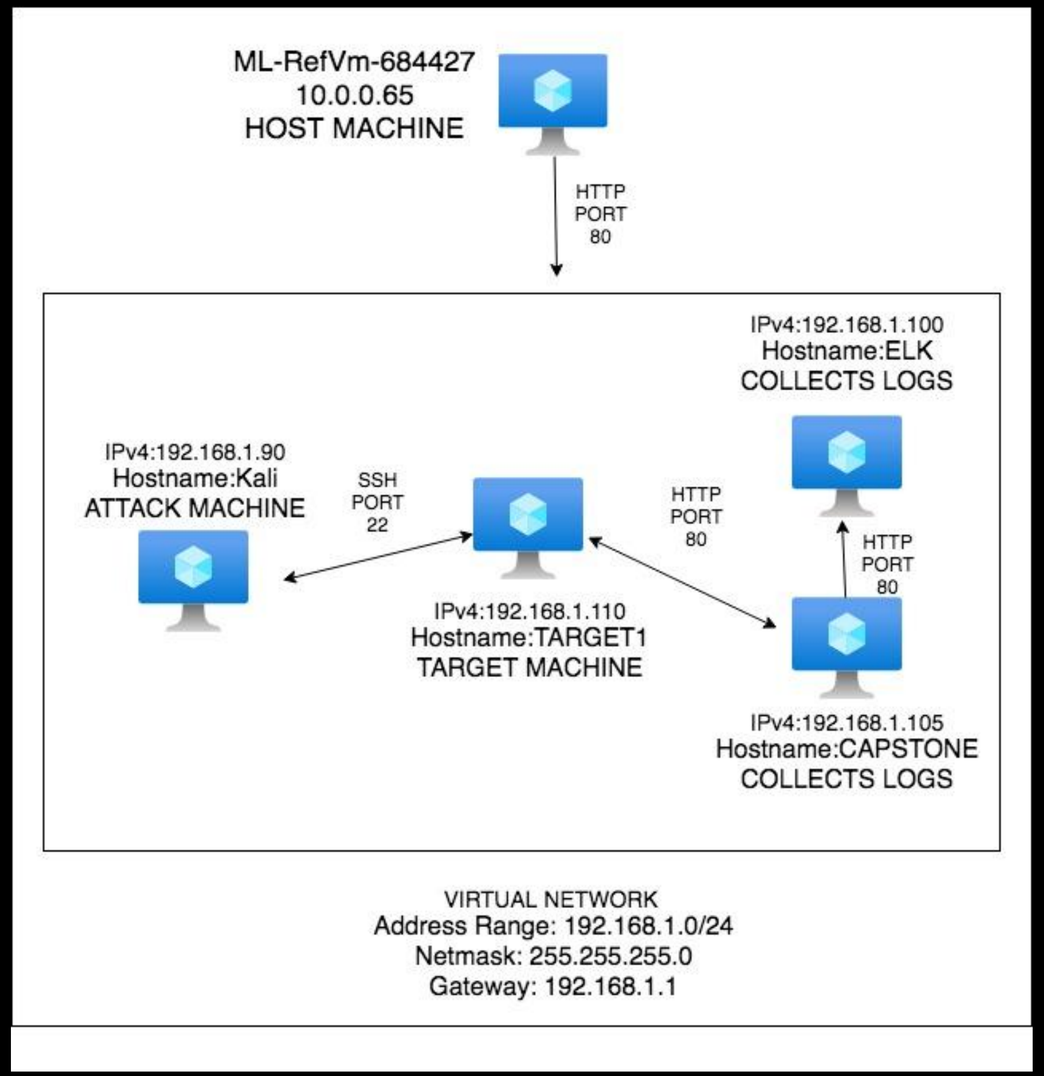
Network
Address Range: 192.168.1.0/24
Netmask: 255.255.255.0
Gateway: 192.168.1.1

Machines
IPv4:192.168.1.90
OS:Kali GNU/Linux Rolling
Hostname:Kali

IPv4: 192.168.1.100
OS:Ubuntu 18.04.4 LTS
Hostname:ELK

IPv4:192.168.1.110
OS:Debian GNU/Linux 8 (jessie)
Hostname: target1

IPv4:192.168.1.105
OS:Ubuntu 18.04.1 LTS
Hostname: server1 (CAPSTONE)



Critical Vulnerabilities: Target 1

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

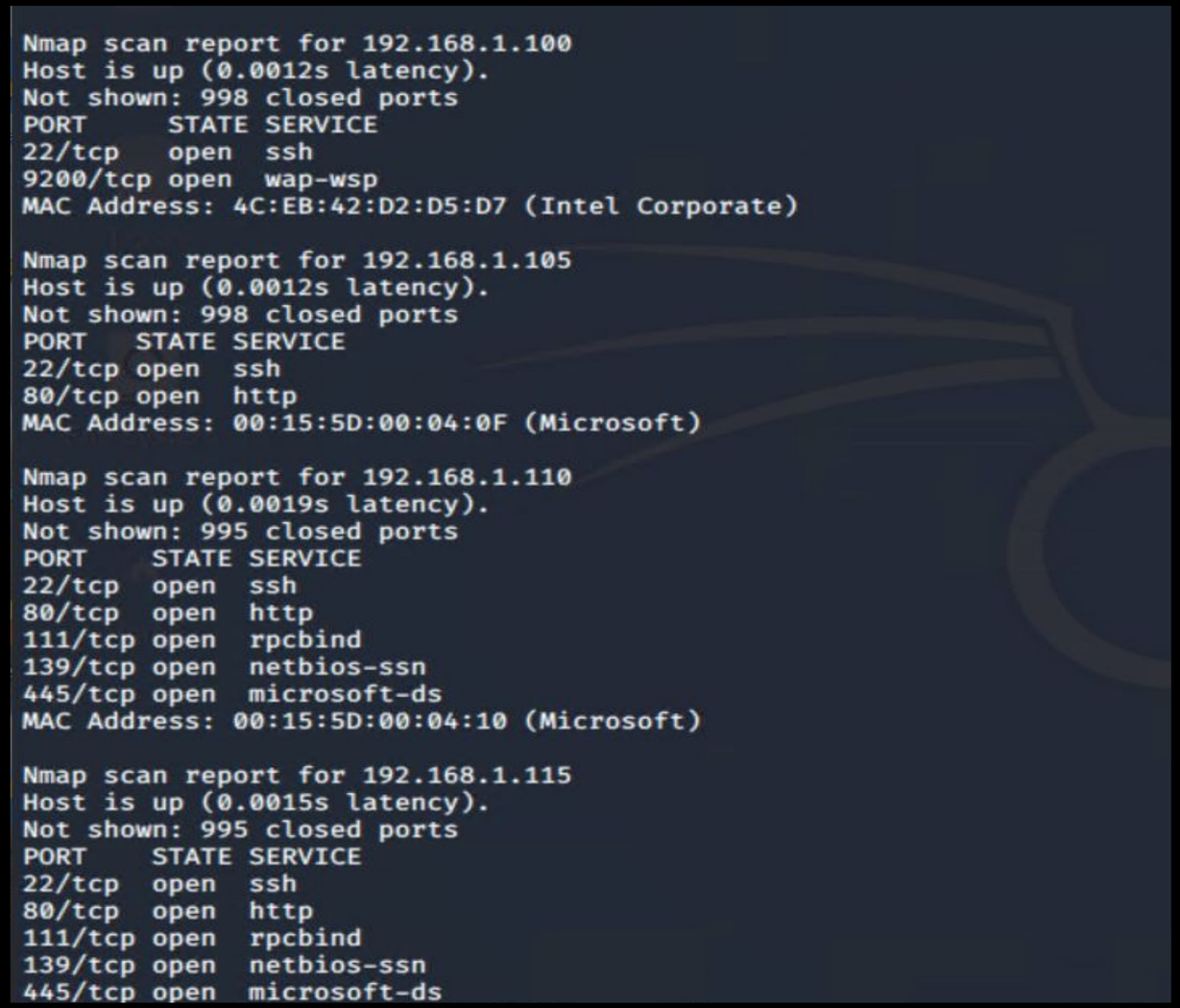
Vulnerability	Description	Impact
Broken Authentication	Attackers have access to millions of valid username and password combinations for automated brute force attacks. These can be exploited to gain access to systems.	Attacker was able to gain access to the database and found usernames and passwords that were easily exploited with a brute force attack.
Security Misconfiguration	Attackers will attempt to exploit unpatched flaws or access unprotected files and directories to gain unauthorized access to the system.	Attacker was able to view the wp-config.php file which contained the admin login and password.
Sensitive Data Exposure	Attackers will steal password hashes off of the server. This information will be used to infiltrate the system.	Attacker was able to brute force data found unencrypted in a secret file and used it to infiltrate the system.
Broken Access Control	Attackers can use tools to attempt to remotely execute attacks. This exploitation, once executed properly, can have severe consequences.	Attacker was able to gain root access with a python bash shell which led to a successful root escalation attack.

Exploits Used

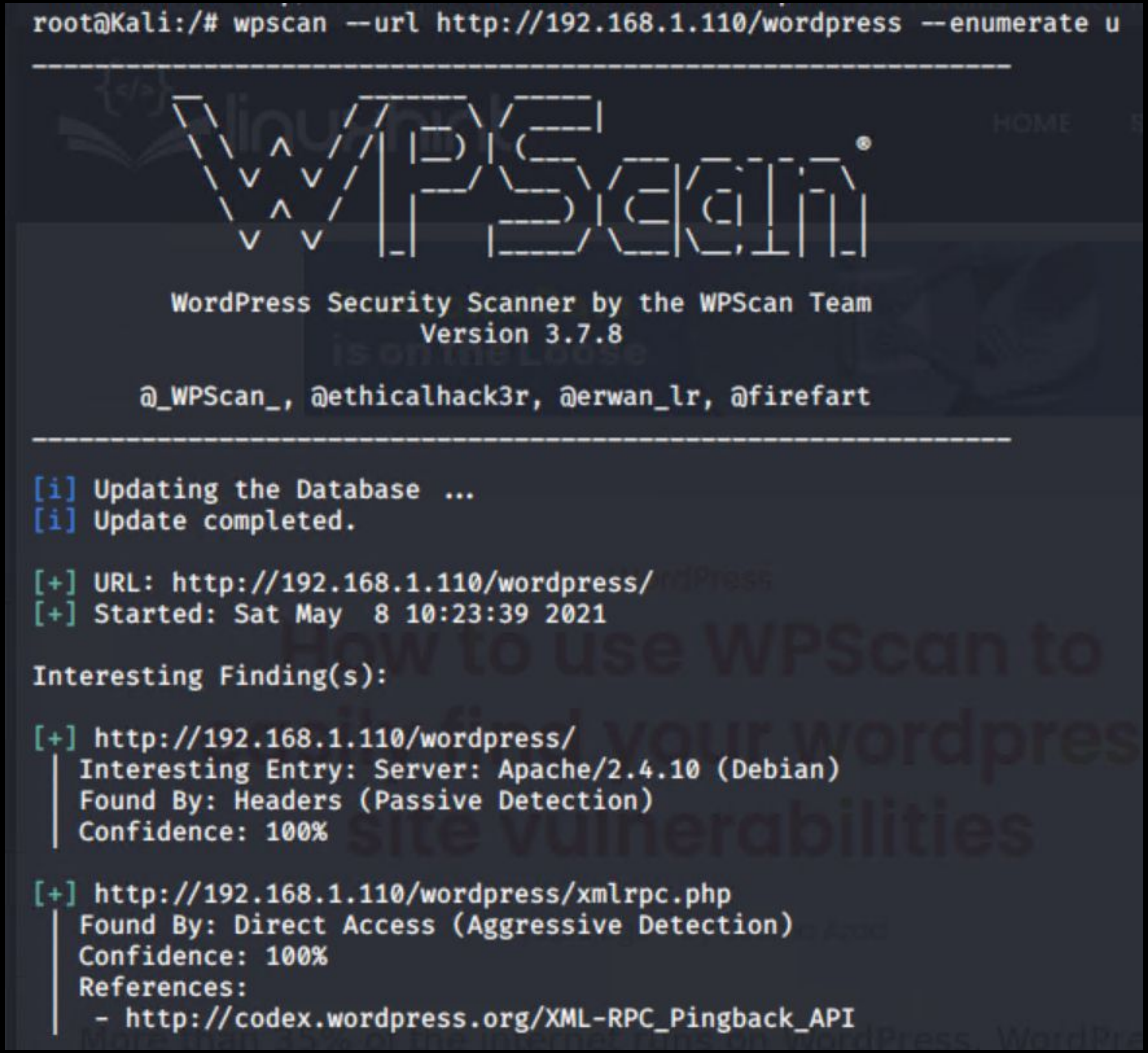
Exploitation: Broken Authentication

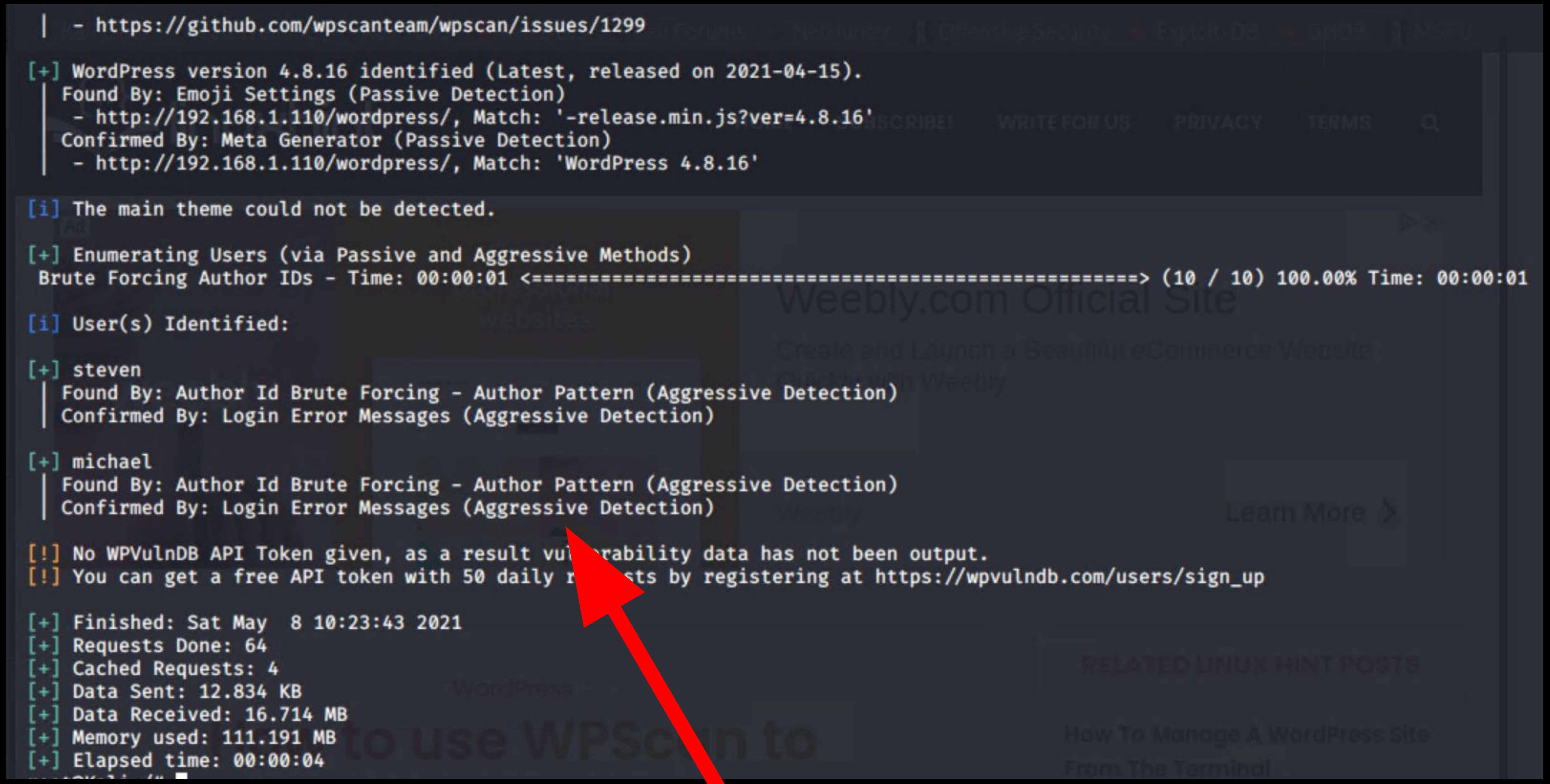
- Nmap was used to scan and identify the IP address of Target 1 and listed the open ports and services. After gaining access to the site and discovering the users through wpscan, the attacker was able to use hydra to brute force Michael's password.

- This allowed the attacker to ssh into a user shell and gain access to the system, which led to the further exploit opportunities.



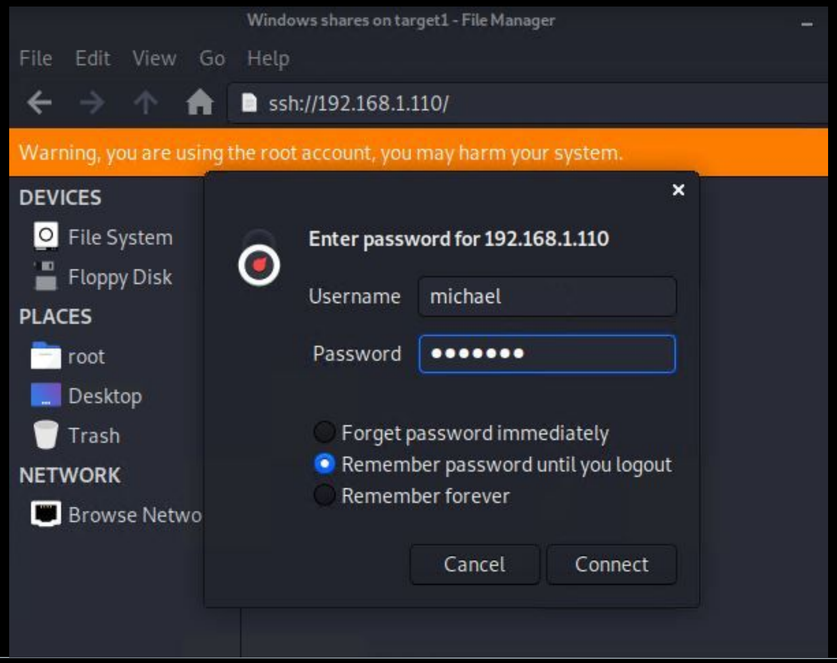
Broken Authentication Screenshots Cont.





Exploitation: Security Misconfiguration

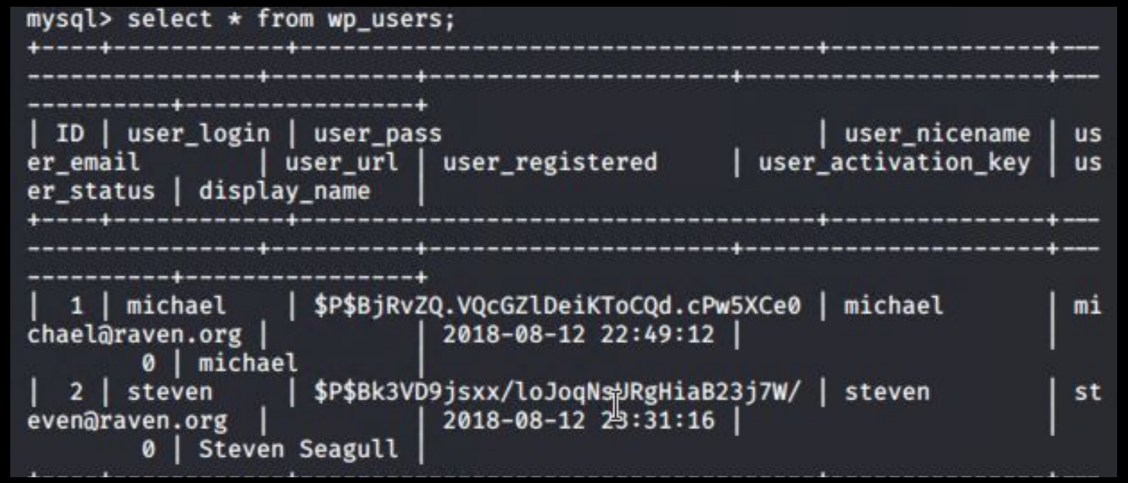
- The attacker was able to exploit the unprotected files within the system. SSH was used to gain access through the file manager, where the wp-config.php file was located, which contained the username and password for the mySQL database.
- This exploit allowed the attacker to gain access to the mySQL database, which contained password hashes.

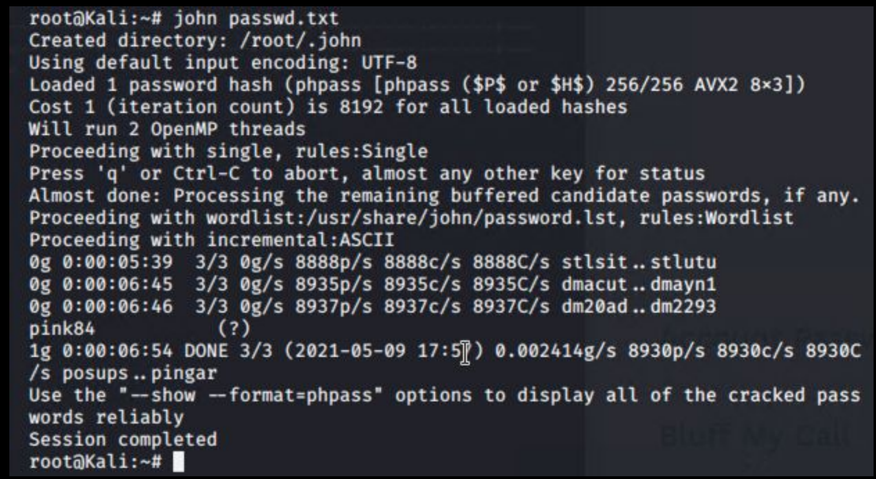



```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

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

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

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');
```

Exploitation: Sensitive Data Exposure

- The file containing mySQL credentials was unencrypted and therefore exposing sensitive information. Once in mySQL, the attacker was able to locate the password hash for Steven.
- This exploit allowed the attacker to run John the Ripper, which cracked Steven's password and led to the ability to SSH into Steven's account.





Exploitation: Broken Access Control

The attacker was able to SSH into the system using Steven's information which led to the discovery of a broken access control vulnerability. After discovering this possible exploit, the attacker was able to run the following command: sudo python -c 'import pty;pty.spawn("/bin/bash")', which led to a successful root escalation attack.

This exploit allowed the attacker to escalate from Steven to root, which led to the access of sensitive files - such as flag4!

```
root@target1:~# cat flag4.txt
______
| ___ \
| |_/ /__ ___   ____ _ __
|    // _` \ \ / / _ \ '_ \
| |\ \ (_| |\ V /  __/ | | |
\_| \_\__,_| \_/ \___|_| |_|

flag4{715dea6c055b9fe3337544932f2941ce}

CONGRATULATIONS on successfully rooting Raven!

This is my first Boot2Root VM - I hope you enjoyed it.
```

```
$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

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

```
$ sudo python -c 'import pty;pty.spawn("/bin/bash")'
root@target1:/var# pwd
/var
root@target1:/var# whoami
root
```

Avoiding Detection

Stealth Exploitation of Broken Authentication

Monitoring Overview

- Which alerts detect this exploit?
  - The broken authentication vulnerability is detected by our excessive HTTP alert.
- Which metrics do they measure?

```
WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes
```

- Which thresholds do they fire at?
  - Above 400 for the last 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - By spacing out the brute force attempts over more time, you can make the attack appear less noticeable; such as running a brute force slow drip attack.
- Are there alternative exploits that may perform better?
  - Top 3 alternatives to WPScan are WPSeku, Wordpress Exploit Framework, and Vane.

Stealth Exploitation of Security Misconfiguration

Monitoring Overview

- Which alerts detect this exploit?
  - The security misconfiguration vulnerability is detected by our CPU usage monitor alert.
- Which metrics do they measure?

```
WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes
```

- Which thresholds do they fire at?
  - Above 0.5, or 50% CPU usage, for the last 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - nmap and hydra would have triggered this alert as opposed to the SSH. A separate alert could monitor external SSH connections.
- Are there alternative exploits that may perform better?
  - Alternative exploits may be DIRB, Gobuster, Shodan, and DirBuster, but these may also trigger alerts.
  - Google dorking, also called “Google hacking”, is when you attempt to access information that is not intended to be public and is another exploit that could be used.

Stealth Exploitation of Sensitive Data Exposure

Monitoring Overview

- Which alerts detect this exploit?
  - The sensitive data exposure vulnerability is detected by our CPU usage monitor alert.
- Which metrics do they measure?

```
WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes
```

- Which thresholds do they fire at?
  - Above 0.5, 50% CPU usage, for the last 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Keep login attempts at a minimum to remain below the threshold.
- Are there alternative exploits that may perform better?
  - You can access MySQL database through metasploit, which would allow you to quickly dump and print the data preventing high cpu alerts from triggering.

Stealth Exploitation of Broken Access Control

Monitoring Overview

- Which alerts detect this exploit?
  - The broken access control vulnerability is detected by our sudo command monitoring alert.
- Which metrics do they measure?

```
WHEN count() GROUPED OVER top 5 'system.auth.sudo.command' IS ABOVE 2 FOR THE LAST 5 minutes
```

- Which thresholds do they fire at?
  - Above 2 for the last 5 minutes.

Mitigating Detection

- How can you execute the same exploit without triggering the alert?

  By attempting to minimize the amount of times sudo is used, you can minimize the alert triggers.
- Are there alternative exploits that may perform better?
  - Since CVE 2019-0217 is listed as a known vulnerability, we would be able to run a metasploit exploit to gain root.
  - By creating a backdoor into the system, we would be able to edit the sudoers file, eliminating the need to use sudo and the python script. You can escalate the privileges of the user to root with a UID of 0.