Group 1 Final Project

Presented by Andrew, Mahmood, Sasha, Arc, Gerald, and Daniel

Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Offensive (Red) Team

Table of Contents

This document contains the following resources:

01

Network Topology & Critical Vulnerabilities

02

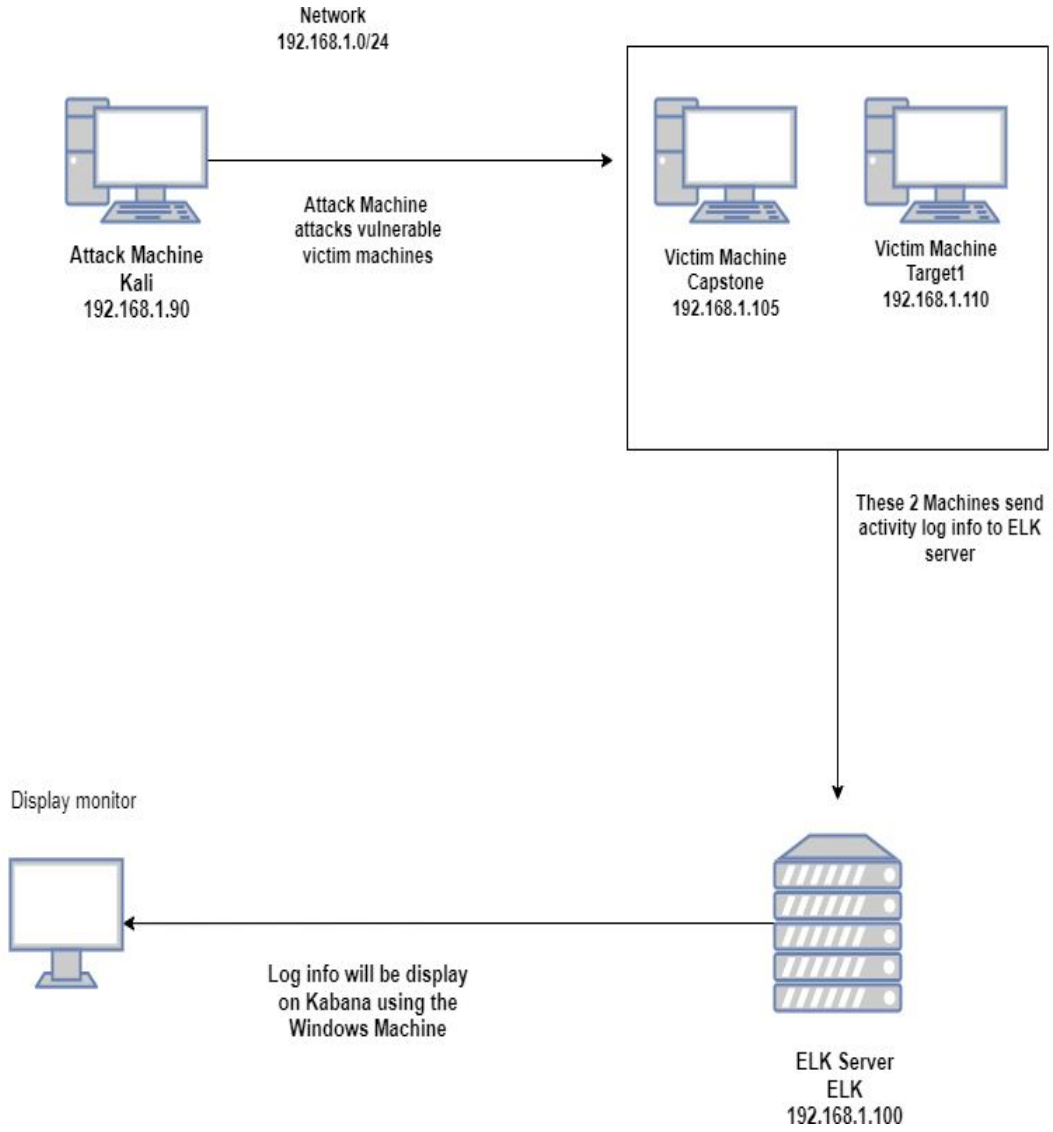
Exploits Used

03

Methods Used to Avoiding Detect

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: 192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux
Hostname: Target1

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Weak password	User ‘michael’ password was ‘michael’	This guessed password allowed us user access to the system via SSH
MySQL Database Access	Wordpress configuration php file has database credentials available to the user	Able to gain access to the database, and extract confidential data

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Wordpress XMLRPC GHOST Vulnerability scanner CVE-2015-0235	The GHOST vulnerability is an exploit that can be used to gain control of a system with out prior knowledge of system logins and passwords	This exploit affects the gethostbyname functions in the GNU C Library (glibc)
Wordpress XMLRPC Dos CVE-2014-5266	The Incutio XML-RPC (IXR) Library does not limit the number of elements in an XML document	allows remote attackers to cause a denial of service (CPU consumption) via a large document.
Wordpress XMLRPC login CVE-1999-0502	This module attempts to authenticate against a Wordpress-site (via XMLRPC) using username and password combinations	Login access

Critical Vulnerabilities: Target 1

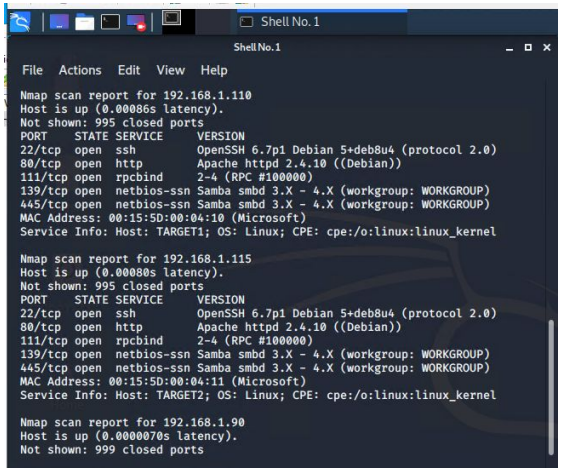
Our assessment uncovered the following critical vulnerabilities in **Target 1**.

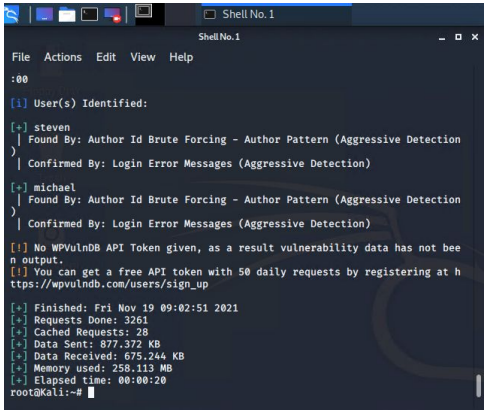
Vulnerability	Description	Impact
Wordpress pingback locator CVE-2013-0235	This module will scan for wordpress sites with the Pingback API enabled	By interfacing with the API an attacker can cause the wordpress site to port scan an external target and return result
Wordpress version 4.8.7	Insecure version	WordPress is prone to multiple vulnerabilities, unpatched version can be exploit

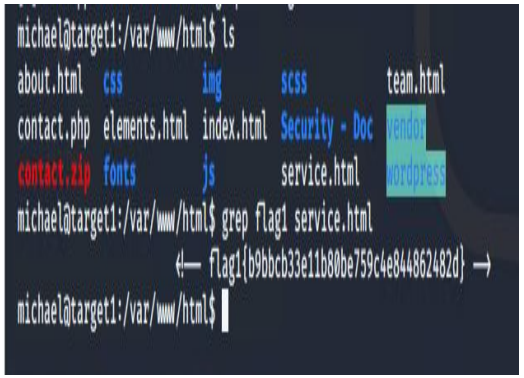
Exploits Used

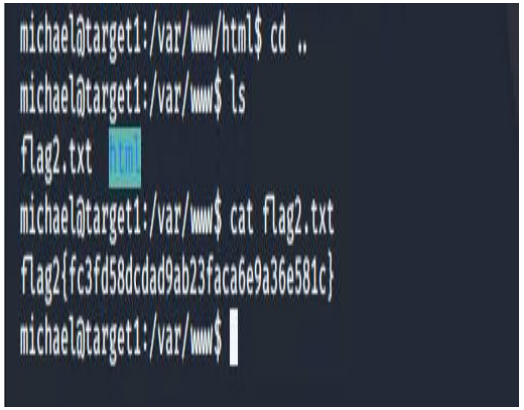
Exploitation: Weak Password

- We used wpscan to find users and guessed the weak password that gave us SSH to the system.
- The exploit granted us access to Michael account, by navigate to htm dir we was able to find flag2 and by using the grep command in the same dir we was able to find flag.



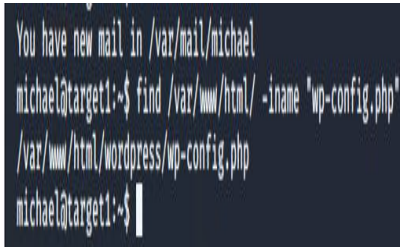


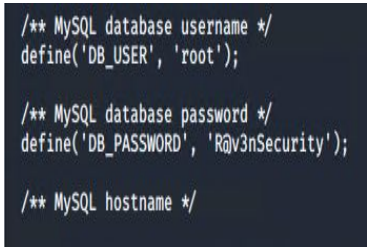


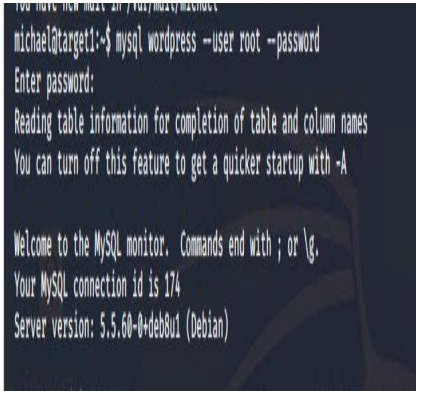


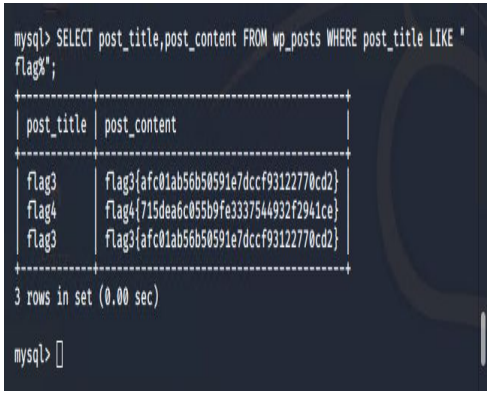
Exploitation: [SQL Database]

- We were able to find the username and password for SQL database in the wp-confg.php file in plaintext.
- The exploit granted us mysql access and we use SELECT post_tittle to find flag3,4.



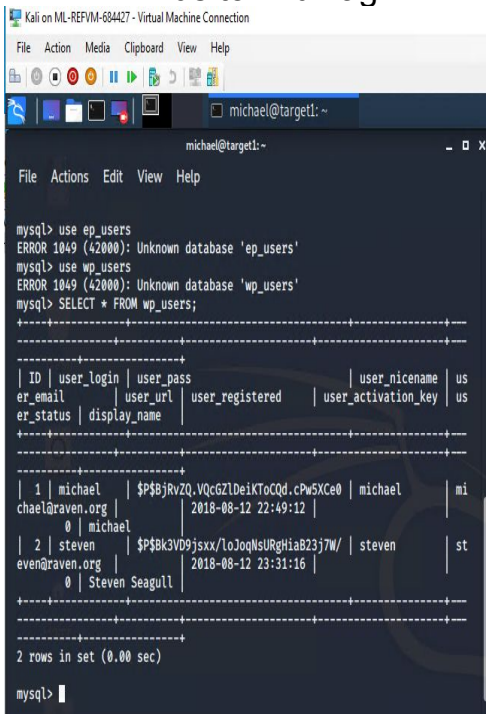


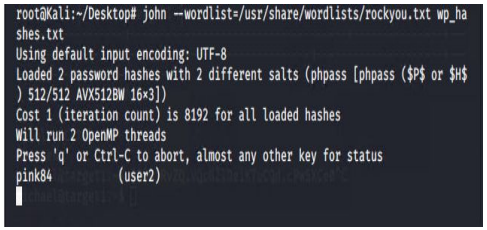


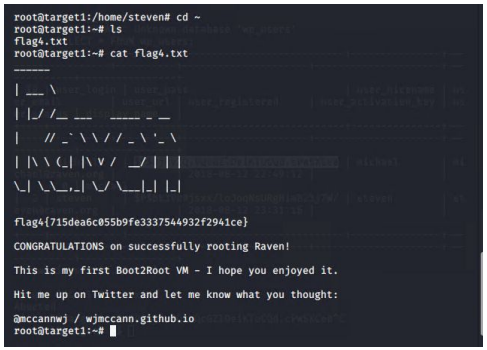


Exploitation: [Privilege Escalation]

- The password hash of Steven was obtained from the SQL database and by crack the password using john the ripper we can access stevet account.
- Exploiting Steven Python's sudo privileges through a spawn shell gave us root access and allowed us to find flag 4.







Avoiding Detection

Stealth Exploitation of [open port 22 and weak password]

Monitoring Overview

- An email alert, when someone logs in to the server via ssh, can be pretty useful to track who is actually using the server.
- Monitor SSH port for unauthorized access.
- Triggers when three attempts to access system over port 22.

Mitigating Detection

- SSH through different open port.
- We can use the reverse shell as alternative exploits .

Stealth Exploitation of [SQL Database]

Monitoring Overview

- Set alert for failed logins
- unauthorized attempts to access SQL database.
- Triggers when three attempts to access SQL database.

Mitigating Detection

- SQL Injection Attack.
- Using brute force on a SQL database with a password cracking tool.

Stealth Exploitation of [privilege Escalation]

Monitoring Overview

- Privilege Escalation Alert
- Monitor unauthorized root access
- Triggers when unauthorized sudo commands are executed

Mitigating Detection

- Kernel Exploit, vulnerabilities are discovered in the Linux kernel. Attackers can exploit these vulnerabilities to gain root access to a Linux system, and once the system is infected with the exploit, there is no way to defend against it

Attackers go through the following steps:

1. Learn about the vulnerabilities
2. Develop or acquire exploit code
3. Transfer the exploit onto the target
4. Execute the exploit on the target

Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Defensive (Blue) Team

Table of Contents

This document contains the following resources:

Alerts Implemented

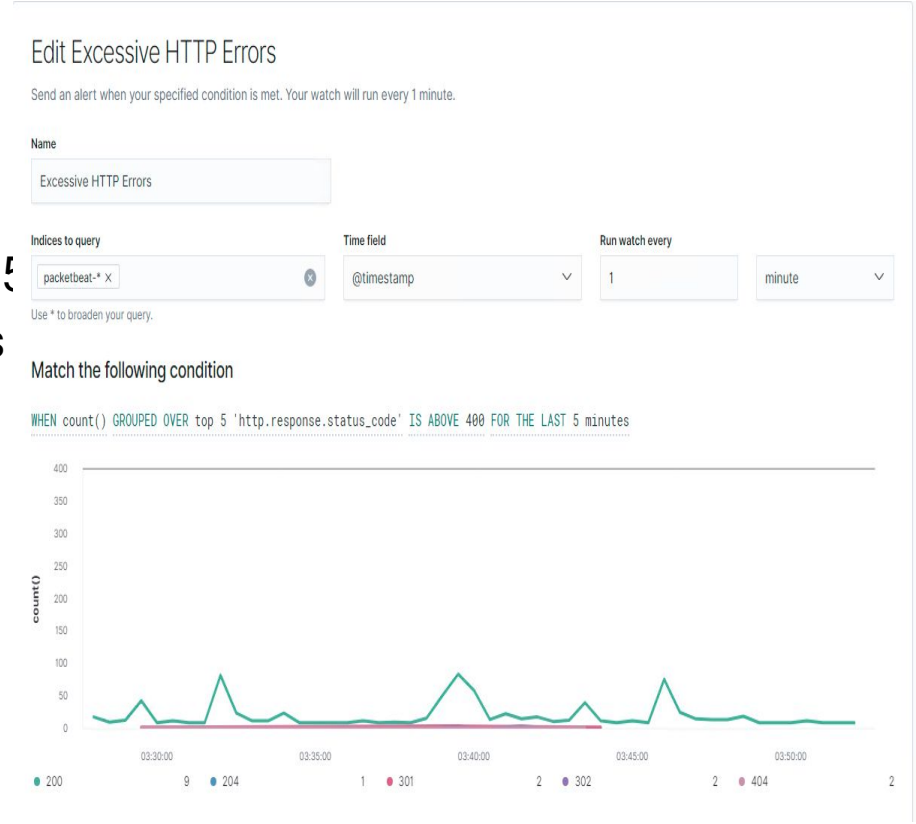
Alert 1: Excessive HTTP Errors

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

- Which **metric** does this alert monitor?
  Number of HTTP errors

- What is the **threshold** it fires at?

  When number grouped over top 5
  above 400 for the last 5 minutes

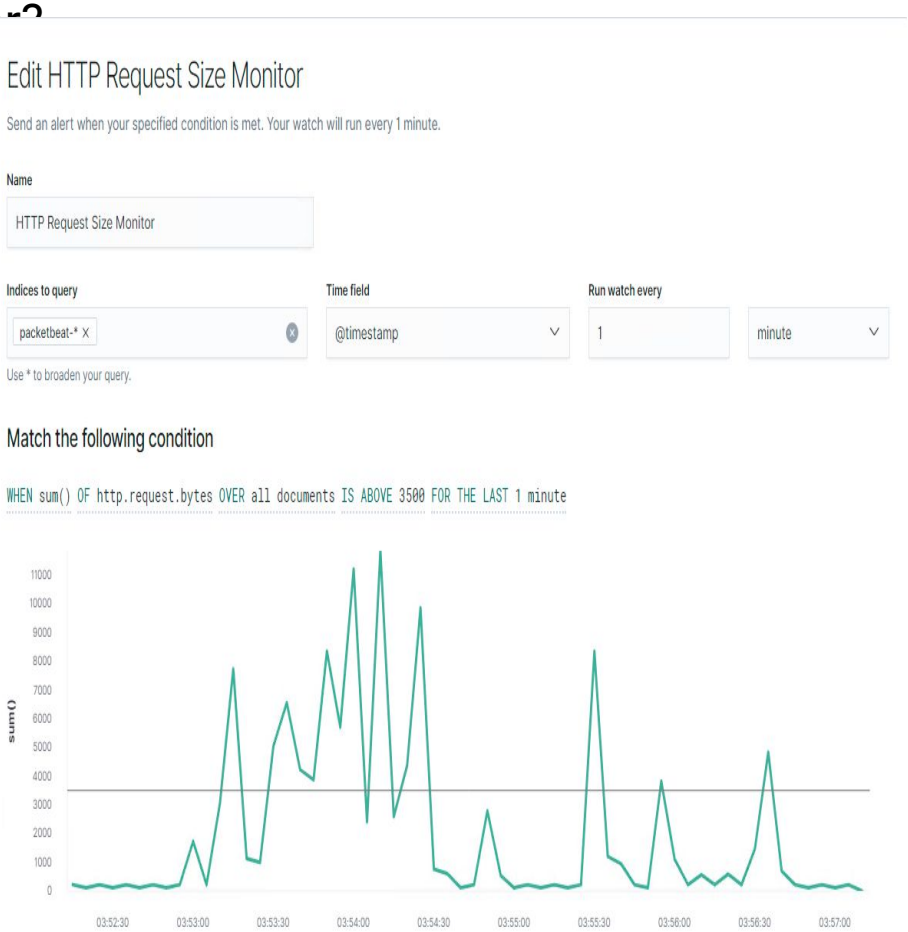


Alert 2: HTTP Request Size Monitor

WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500
FOR THE LAST 1 minute

- Which **metric** does this alert monitor?
  HTTP request size over all docume

- What is the **threshold** it fires at?
  When total is above 3500 bytes for the last 1 minute

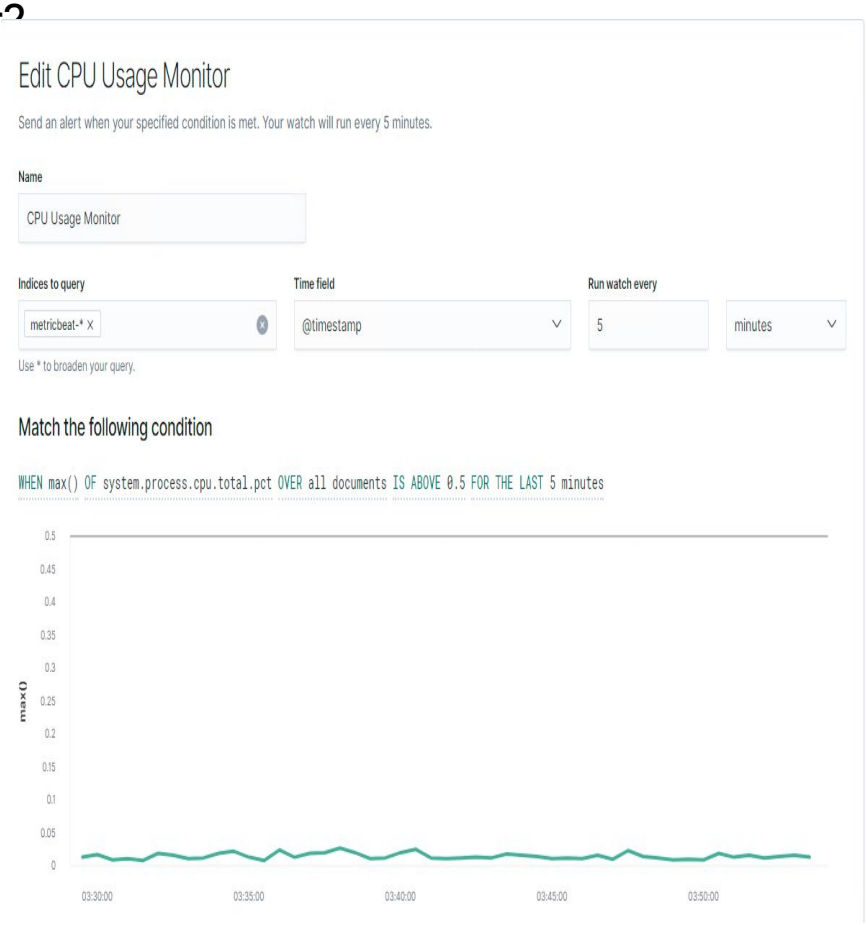


Alert 3: CPU Usage Monitor

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

- Which **metric** does this alert monitor?
  Total system CPU usage over all documents

- What is the **threshold** it fires at?
  When max is above 0.5% for the last 5 minutes



Hardening

Hardening Against Weak Password on Target 1

Target 1 exhibited 3 vulnerabilities as follows:

1. **An exposed WordPress configuration file**

   Change permission of the wp-config.php file so that only the owner can read it

   ```
   Command: $ chmod 440 /var/www/html/wordpress/wp-config.php
   ```

2. **Weak WordPress User passwords**

   Install a WordPress plugin to enforce strong passwords by users of WordPress

3. **Weak SSH password**

   Remove ssh password authentication on the server and require public key login instead

   ```
   Command:  $nano /etc/ssh/sshd_config
   ```

   Then change ‘PasswordAuthentication’ to ‘no’

   Copy a user’s public key from the workstation to the server

   ```
   Command:   $ ssh-copy-id michael@192.168.1.110
   ```

Hardening Against SQL Database Access on Target 1

1. Remove all anonymous accounts
2. Change default port mappings
3. Limit which hosts have access to MySQL
4. Do not run MySQL with root level privileges
5. Disable remote logins
6. Limit or Disable SHOW DATABASES command
7. Obfuscate the **root** account , change it to something else
8. Set the proper file permissions

Hardening Against Outdated Software on Target 1

Target 1 has Wordpress version 4.8.17 that should have been updated to latest version 5.7.2

1. With latest versions, vulnerable plugins and themes are fixed
2. Do not use nulled or free plugins and themes
3. Use Wordpress security plugins

Implementing Patches

Implementing Patches with Ansible

Playbook Overview

1. Make sure that **ansible** and **sshpass** are installed on the host where you are running the playbook.
2. Then edit the `/etc/ansible/hosts` file to add the IP address of the target machine. Edit `/etc/ansible/ansible.cfg` to add the remote user for the target machine (‘vagrant’)
3. Copy over the SSH public key for the user you are running the playbook as with the `‘ssh-copy-id user@192.168.1.110’` command. Do this before running the playbook, otherwise you won’t be able to do this later.

Then to address the exposed WordPress configuration file and weak SSH password we can run this playbook.

Implementing Patches with Ansible

```
---
  - name: Harden SSH and WordPress config
    hosts: all
    become: true
    tasks:

    - name: Change permission of wp-config.php
      file:
        path: /var/www/html/wordpress/wp-config.php
        mode: 440

    - name: Copy SSH key to target host
      authorized_key:
        user: michael
        state: present
        key: "{{ lookup('file', lookup('env','HOME') + '/.ssh/id_rsa.pub') }}"
      authorized_key:
        user: steven
        state: present
        key: "{{ lookup('file', lookup('env','HOME') + '/.ssh/id_rsa.pub') }}"
```

Implementing Patches with Ansible

```
- name: Disable SSH Password Authentication
  lineinfile:
        dest=/etc/ssh/sshd_config
        regexp='^PasswordAuthentication'
        line="PasswordAuthentication no"
        state=present
        backup=yes

- name: restart ssh
  service:
    name: sshd
    state: restarted
```

Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Network Analysis

Table of Contents

This document contains the following resources:

Traffic Profile

Traffic Profile

Our analysis identified the following characteristics of the traffic on the

Feature	Value	Description
Top Talkers (IP Addresses)	172.16.4.205: 51,364 (49%) 185.243.115.84: 30,344 (29%) 10.0.0.201: 19,503 (19%)	Machines that sent the most traffic.
Most Common Protocols	UDP: 11,697 (11.2%) TCP: 92,280 (88.6%) ARP: 212 (0.2%)	Three most common protocols on the network.
# of Unique IP Addresses	808	Count of observed IP addresses.
Subnets	10.6.12.0/24 172.16.4.0/24	Observed subnet ranges.

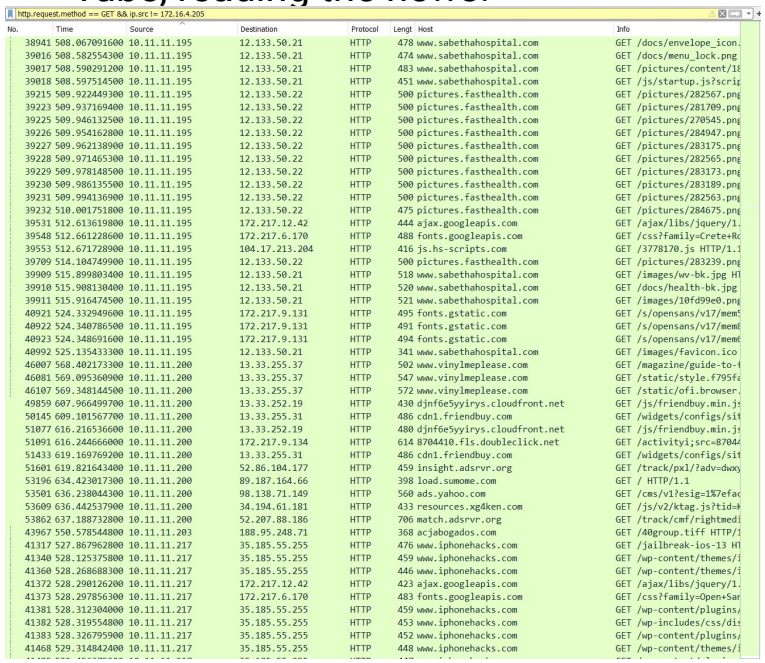
Behavioral Analysis

Purpose of Traffic on the Network - Users were observed engaging in the following kinds of activity.

1. **“Normal” Activity**

You

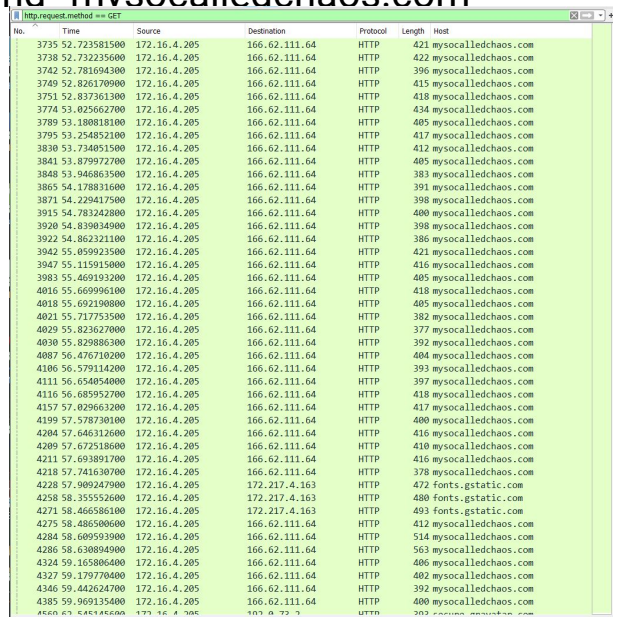
Tube, reading the news.



2. **Suspicious Activity**

For example: Sending malware, phishing.

For example: accessing “mysocalledchaos.com”



Normal Activity

Normal Behavior

Summarize the following:

- Type of traffic observed:

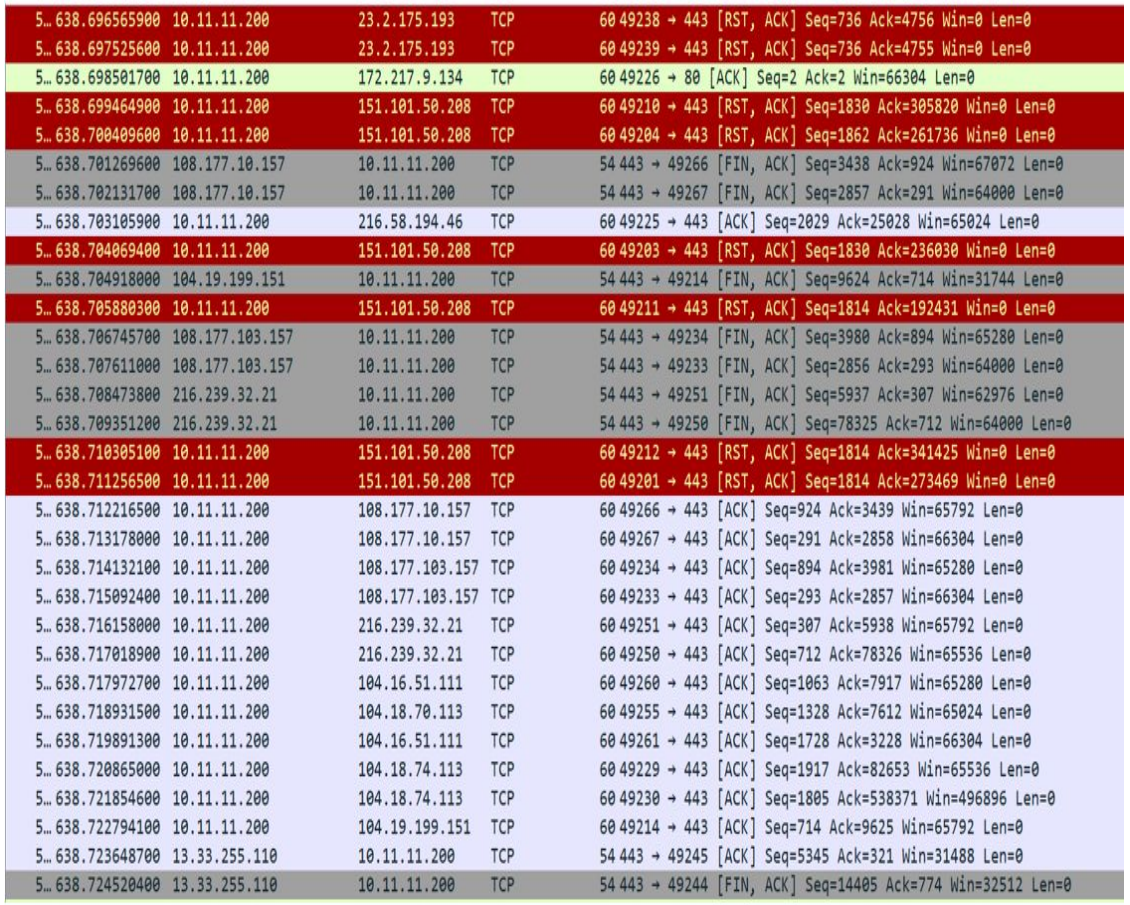
  Protocol(s) used:TCP

  typical traffic, http traffic

- User activity: Three way handshake - RST ACK FIN

  web surfing,

- Description of any interesting files:



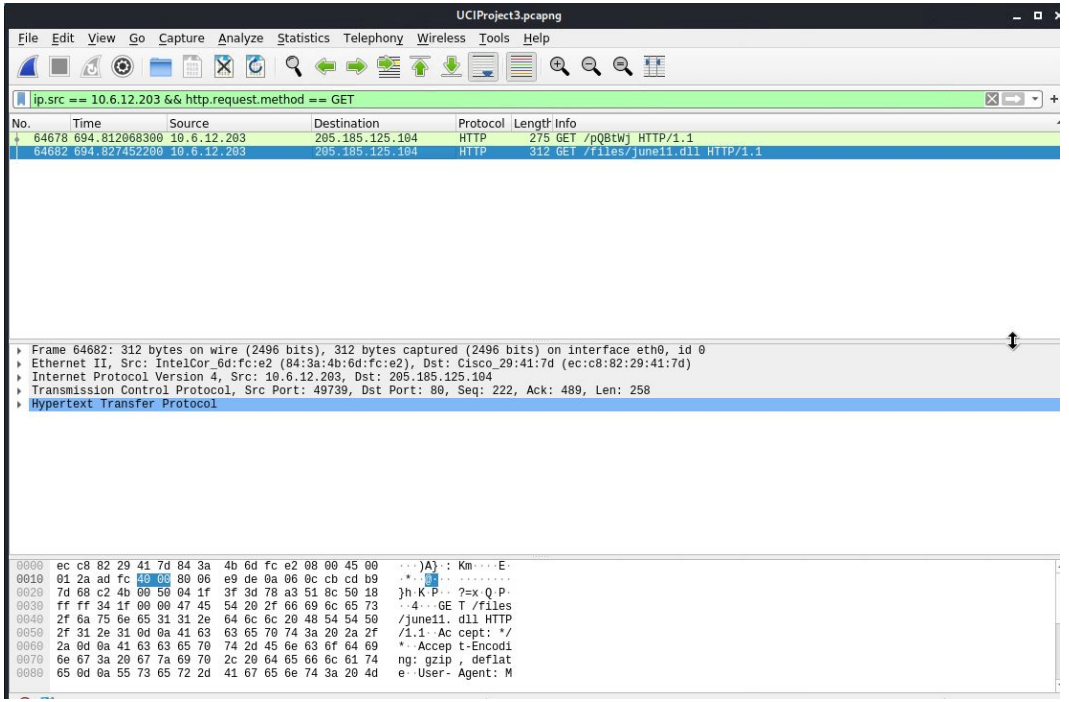
5… 638.696565900	10.11.11.200	23.2.175.193	TCP	60 49238 → 443 [RST, ACK] Seq=736 Ack=4756 Win=0 Len=0
5… 638.697525600	10.11.11.200	23.2.175.193	TCP	60 49239 → 443 [RST, ACK] Seq=736 Ack=4755 Win=0 Len=0
5… 638.698501700	10.11.11.200	172.217.9.134	TCP	60 49226 → 80 [ACK] Seq=2 Ack=2 Win=66304 Len=0
5… 638.699464900	10.11.11.200	151.101.50.208	TCP	60 49210 → 443 [RST, ACK] Seq=1830 Ack=305820 Win=0 Len=0
5… 638.700409600	10.11.11.200	151.101.50.208	TCP	60 49204 → 443 [RST, ACK] Seq=1862 Ack=261736 Win=0 Len=0
5… 638.701269600	108.177.10.157	10.11.11.200	TCP	54 443 → 49266 [FIN, ACK] Seq=3438 Ack=924 Win=67072 Len=0
5… 638.702131700	108.177.10.157	10.11.11.200	TCP	54 443 → 49267 [FIN, ACK] Seq=2857 Ack=291 Win=64000 Len=0
5… 638.703105900	10.11.11.200	216.58.194.46	TCP	60 49225 → 443 [ACK] Seq=2029 Ack=25028 Win=65024 Len=0
5… 638.704069400	10.11.11.200	151.101.50.208	TCP	60 49203 → 443 [RST, ACK] Seq=1830 Ack=236030 Win=0 Len=0
5… 638.704918000	104.19.199.151	10.11.11.200	TCP	54 443 → 49214 [FIN, ACK] Seq=9624 Ack=714 Win=31744 Len=0
5… 638.705880300	10.11.11.200	151.101.50.208	TCP	60 49211 → 443 [RST, ACK] Seq=1814 Ack=192431 Win=0 Len=0
5… 638.706745700	108.177.103.157	10.11.11.200	TCP	54 443 → 49234 [FIN, ACK] Seq=3980 Ack=894 Win=65280 Len=0
5… 638.707611000	108.177.103.157	10.11.11.200	TCP	54 443 → 49233 [FIN, ACK] Seq=2856 Ack=293 Win=64000 Len=0
5… 638.708473800	216.239.32.21	10.11.11.200	TCP	54 443 → 49251 [FIN, ACK] Seq=5937 Ack=307 Win=62976 Len=0
5… 638.709351200	216.239.32.21	10.11.11.200	TCP	54 443 → 49250 [FIN, ACK] Seq=78325 Ack=712 Win=64000 Len=0
5… 638.710305100	10.11.11.200	151.101.50.208	TCP	60 49212 → 443 [RST, ACK] Seq=1814 Ack=341425 Win=0 Len=0
5… 638.711256500	10.11.11.200	151.101.50.208	TCP	60 49201 → 443 [RST, ACK] Seq=1814 Ack=273469 Win=0 Len=0
5… 638.712216500	10.11.11.200	108.177.10.157	TCP	60 49266 → 443 [ACK] Seq=924 Ack=3439 Win=65792 Len=0
5… 638.713178000	10.11.11.200	108.177.10.157	TCP	60 49267 → 443 [ACK] Seq=291 Ack=2858 Win=66304 Len=0
5… 638.714132100	10.11.11.200	108.177.103.157	TCP	60 49234 → 443 [ACK] Seq=894 Ack=3981 Win=65280 Len=0
5… 638.715092400	10.11.11.200	108.177.103.157	TCP	60 49233 → 443 [ACK] Seq=293 Ack=2857 Win=66304 Len=0
5… 638.716158000	10.11.11.200	216.239.32.21	TCP	60 49251 → 443 [ACK] Seq=307 Ack=5938 Win=65792 Len=0
5… 638.717018900	10.11.11.200	216.239.32.21	TCP	60 49250 → 443 [ACK] Seq=712 Ack=78326 Win=65536 Len=0
5… 638.717972700	10.11.11.200	104.16.51.111	TCP	60 49260 → 443 [ACK] Seq=1063 Ack=7917 Win=65280 Len=0
5… 638.718931500	10.11.11.200	104.18.70.113	TCP	60 49255 → 443 [ACK] Seq=1328 Ack=7612 Win=65024 Len=0
5… 638.719891300	10.11.11.200	104.16.51.111	TCP	60 49261 → 443 [ACK] Seq=1728 Ack=3228 Win=66304 Len=0
5… 638.720865000	10.11.11.200	104.18.74.113	TCP	60 49229 → 443 [ACK] Seq=1917 Ack=82653 Win=65536 Len=0
5… 638.721854600	10.11.11.200	104.18.74.113	TCP	60 49230 → 443 [ACK] Seq=1805 Ack=538371 Win=496896 Len=0
5… 638.722794100	10.11.11.200	104.19.199.151	TCP	60 49214 → 443 [ACK] Seq=714 Ack=9625 Win=65792 Len=0
5… 638.723648700	13.33.255.110	10.11.11.200	TCP	54 443 → 49245 [ACK] Seq=5345 Ack=321 Win=31488 Len=0
5… 638.724520400	13.33.255.110	10.11.11.200	TCP	54 443 → 49244 [FIN, ACK] Seq=14405 Ack=774 Win=32512 Len=0

Malicious Activity

[Malicious Behavior: June11.dll malware download

Wireshark search string : Ip.src == 10.6.12.203 && http.request.method == GET

- Domain name of the users' custom site: Wpad.Frank'n'Ted.com (windows proxy auto discovery)
- IP address of the Domain Controller (DC) of the AD network : 10.6.12.12
- Type of traffic observed : GET request was made by IP - 10.6.12.203 for a known malware file called "June11.dll"
- File was exported by us and posted to Virustotal.com
- Specific user activity (browsing, POST GET Etc): No other mentions of file June11.dll were found.
- Description of any interesting files: June11.dll is listed on virus total as a Trojan type malware with a HIGH threat level by 49 security vendors



June11.dll Threat level: High 10 out of 10

49 security vendors have identified this file as a Trojan type malware

https://maltiverse.com › sample

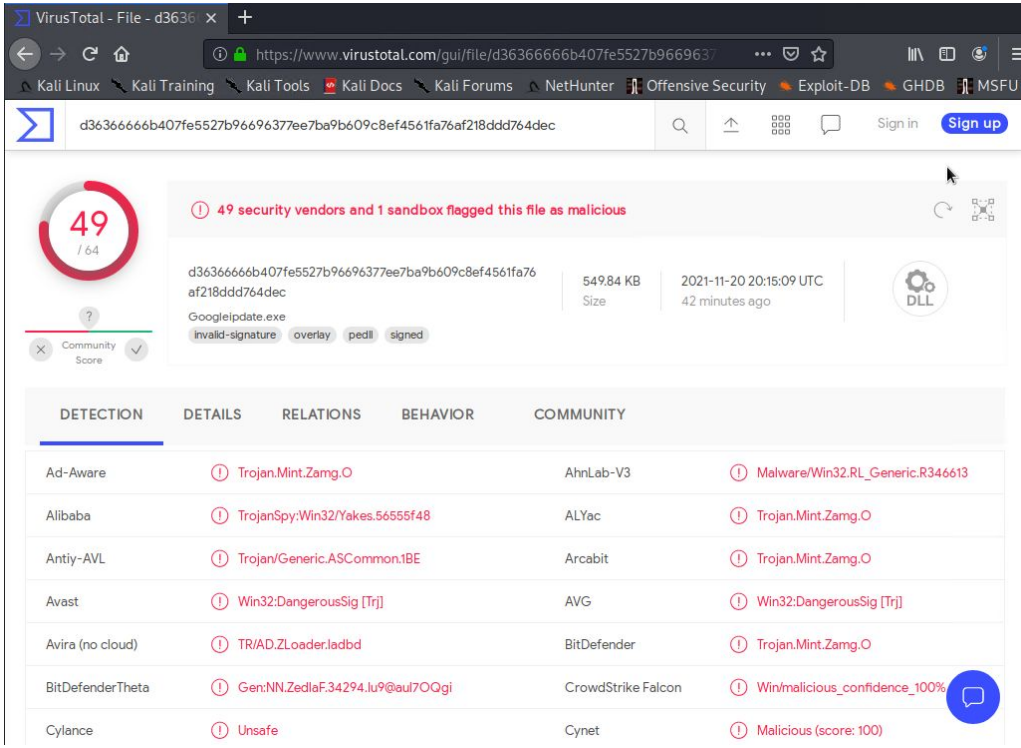
june11.dll - Malicious Sample - Maltiverse

Jul 13, 2020 — **june11.dll**. Classification: malicious. Tags. Blacklist sightings. Description, Source, First Seen, Last Seen, Labels. Trojan.

https://app.any.run › tasks

june11.dll (MD5: 2545B15483165D00D1B6D63D9FD0821D)

Jun 11, 2021 — Interactive malware hunting service. Live testing of most type of threats in any environments. No installation and no waiting necessary.



Vulnerable Windows Machines

Infected machine

Information about the infected Windows machine:

- Host name: Rotterdam-PC
- IP address: 172.16.4.205
- MAC address: (00:59:07:b0:63:a4)

- Username of the Windows user whose computer is infected:

  mattijs.devries

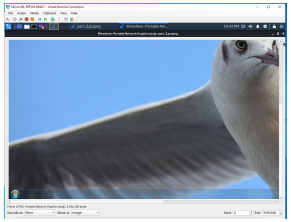
- IPs used in the infected traffic 172.16.4.205 and 172.16.4.4
- The Screenshot of this desktop

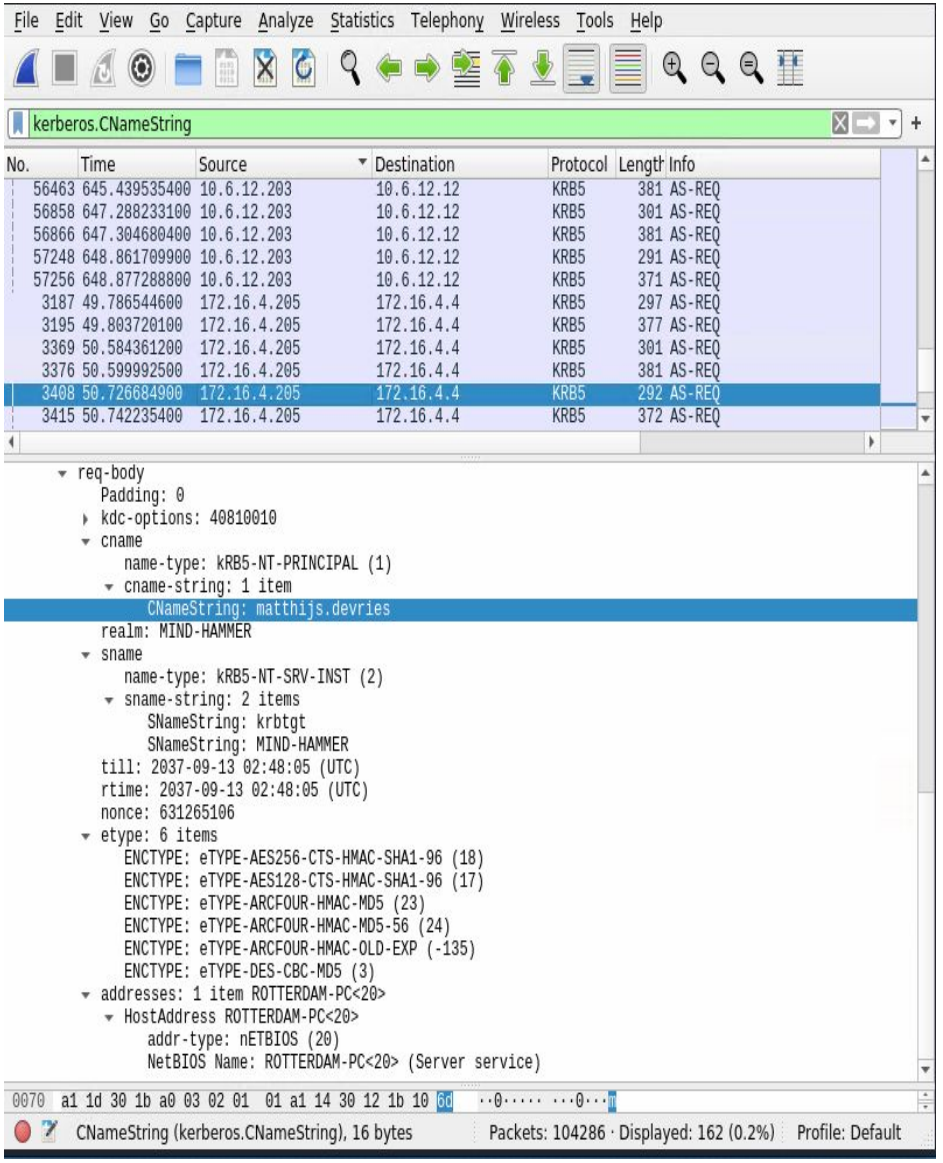
  was captured by isolating

  http traffic in PNG format.

  “Ctrl--Shift-O” allows wireshark

  view images.

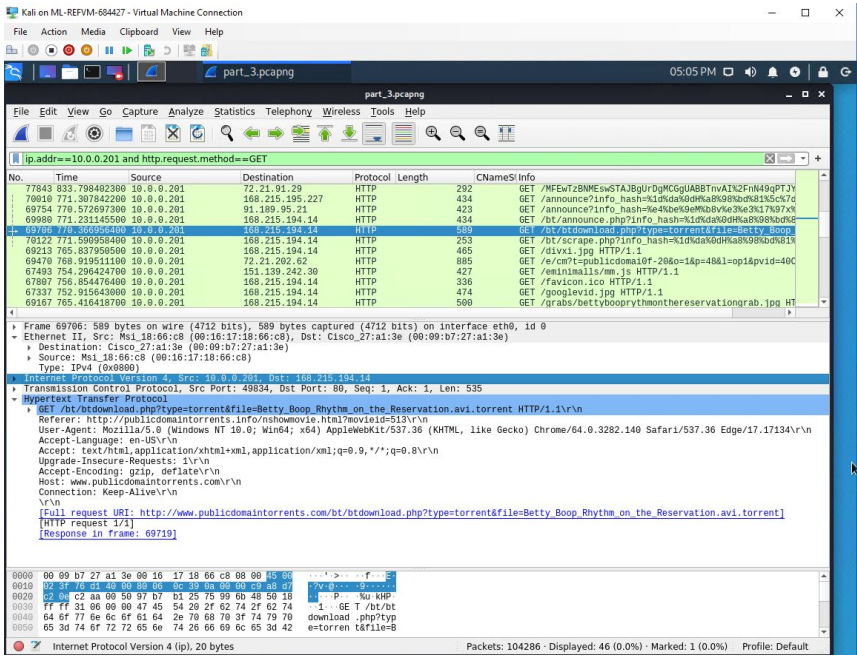




[Torrent downloading]

Summary - The machines using torrents live in the range `10.0.0.0/24` and are clients of an AD domain.

- IP address 10.0.0.201 Belongs to Elmer Blanco
- MAC address : (00:16:17:18:66:c8)
- Windows username: elmer.blanco
- OS version: Windows NT 10.0
- The DC is associated with the domain dogoftheyear.net



- Specific user activity browsing: GET request of copyrighted Torrent file: Betty_Boop_Rhythm_on_the_Reservation.avi.torrent
- Description of any interesting files: most of the **torrent files are known to contain copyrighted material**. ... If they find any trace of illegal torrent file downloads, or any copyrighted torrent file, the torrent users are liable to face legal consequences for their illegal actions over the internet.

The End