NETWORK SECURITY	Student information	Date	Project
	Matricola N. 0001/24312 Name: Carlos Sampedro Mdz.	25-1-25	Intrusion Detection System

Intrusion Detecting Project Summary

The project was designed to simulate the stages of a penetration test, using a victim, attacker, and detection system in a controlled environment, in this case of Virtual Machines. The goal was to simulate each step of an intrusion: Footprinting, Scanning, Enumeration, and Exploitation, while implementing and testing different intrusion detection techniques to monitor the activity.

Set-up

The setup involved configuring a network of 3 Virtual Machines using Oracle VirtualBox. The machines included the following roles:

- Victim: A Metasploitable VM, which is vulnerable to exploitation. This VM was set up with default configurations for this intrusive pursopes.
- Attacker: A Kali Linux VM, used as the attacker machine to perform various penetration testing activities. This included scanning, enumeration, and exploitation attempts.
- Detection: Initially, an Ubuntu Server was configured to run Suricata, Wireshark (tshark in this case as it was a CLI), tcpdump… as an Intrusion Detection System (IDS). However, during the setup, multiple issues took place:
  - The Ubuntu server had a network interface issue where the system used enp0s3 instead of the expected eth0. As a result, Suricata failed to detect the traffic properly, as the IDS was unable to listen on the correct network interface.
  - Due to this issue, the planned detection system had to be changed for another Kali Linux VM configured to run this detection. This allowed Suricata to monitor traffic correctly.

To guarantee that all machines are in the same network the following command has to be executed:

```
sudo ifconfig eth0 192.168.1.xx netmask 255.255.255.0 up
```

In my case victim is 10, attacker 20 and detection 30.

NETWORK SECURITY	Student information	Date	Project
	Matricola N. 0001/24312 Name: Carlos Sampedro Mdz.	25-1-25	Intrusion Detection System

1. Footprinting

Objective: Gather preliminary information about the target without directly interacting with it.

Tools and Techniques:

1.1 WHOIS Lookup (for external IPs or domains):

```
whois <target-domain>
```

1.2 DNS Enumeration (for identifying domain records):

- nslookup:

  ```
  nslookup 192.168.1.10
  ```

- dig:

  ```
  dig @192.168.1.10 example.com
  ```

1.3 Ping Sweep

```
ping 192.168.1.10
```

2. Scanning

Objective: Identify live hosts, open ports, and running services in the network.

Tools and Techniques:

2.1. Network Scanning with Nmap

1. nmap -sn 192.168.1.0/24
   - Scans the network to identify which hosts are live.
2. Port Scanning (all ports):

   ```
   nmap -p- 192.168.1.10
   ```
3. Service Version Detection:

   ```
   nmap -sV -p 22,80,443 192.168.1.10
   ```

   Detects the version of services on specific ports (in this case, SSH, HTTP, and HTTPS).
4. Aggressive Scanning (Detailed Enumeration):

```
nmap -A 192.168.1.10
```

- Includes OS detection, service version detection, script scanning, and traceroute.

2.2. Masscan for Fast Port Scanning

Masscan is an extremely fast port scanner. It can be used to scan large networks quickly.

```
masscan 192.168.1.10 -p0-65535
```

- Scans all ports (0-65535) on the target.

2.3. ARP (Address Resolution Protocol) Scanning

ARP scanning is useful to discover devices in a local network.
For detecting ARP requests or responses on your local network, we can use tcpdump instead of Suricata.

```
arp-a
```

2.5. Curl for HTTP Requests

Curl can be used to test HTTP endpoints and send custom requests.

```
curl http://192.168.1.10
```

- Sends a GET request to the web server running on port 80 of the target.

3. Enumeration

Objective: Gather detailed information about services, users, and configurations on the target.

3.1. FTP Enumeration

FTP is a common service for file transfers, and it can be tested for anonymous login or weak credentials.

1. Check for Anonymous Login (not working, just with credentials):

```
ftp 192.168.1.10
```

- Attempt to log in without a username and password to see if anonymous access is allowed. If successful, you may be able to interact with the file system.

2. **Brute-Force FTP with Hydra:**

   hydra -l msfadmin -P hydraIncursion/random-passwords.txt ftp://192.168.1.10

   - Use Hydra to perform a dictionary-based brute-force attack on FTP, testing the msfadmin username with a list of potential passwords.

3.2. SMB Enumeration

SMB (Server Message Block) is used for sharing files and printers.

enum4linux -a 192.168.1.10

- This tool is used to enumerate SMB information such as user accounts, group memberships, and share details. The -a flag runs all enumeration options.

3.3. SSH Enumeration

SSH (Secure Shell) is often used for remote login.

1. Test for Open SSH Port:
2. nmap -p 22 192.168.1.10
   - This checks if port 22 (SSH) is open.
3. Brute-Force SSH with Hydra:
4. hydra -L /path/to/usernames.txt -P /path/to/passwordlist.txt ssh://192.168.1.10
   - Use Hydra to perform a brute-force attack on SSH by trying a list of usernames (usernames.txt) and passwords (passwordlist.txt).
5. Manual SSH Login:

   ssh msfadmin@192.168.1.10

   - Try manually logging into the target system via SSH using the credentials found through brute-forcing.

3.4. Username Harvesting with Cewl (not working)

Cewl is a tool for gathering potential usernames from a website by scraping content.

Harvest Usernames from a Website:

cewl http://192.168.1.10 -d 3 -w usernames.txt

4. Exploitation

Objective: Exploit vulnerabilities to gain access or execute actions on the target system.

4.1. FTP Exploitation

FTP can be exploited by uploading files to the target system or by gaining access with weak credentials. Obtained with hydra.

1. Upload Files to FTP:
   - ftp 192.168.1.10
   - put file.txt
2. Download Files from FTP:
   - get file.txt

4.2. SSH Exploitation

SSH is another method to gain remote access to a system. If you obtained valid SSH credentials using Hydra, you can attempt to log in remotely.

1. SSH Login:
   - Using the credentials obtained from Hydra, log into the victim machine via SSH:
   - ssh msfadmin@192.168.1.10
   - Once logged in, you will have access to the remote system's shell and can execute commands.
2. Exploitation with Privilege Escalation:

- You can use sudo to escalate privileges if the user has the right permissions: sudo su

4.3. DDoS Exploitation with hping3

You can simulate a Distributed Denial of Service (DDoS) attack using hping3 to flood the target system with traffic and potentially cause a service disruption.

1. Flood Port 80 with SYN Packets:
2. hping3 -S --flood -p 80 192.168.1.10
   - This command sends a flood of SYN packets to port 80 (HTTP), attempting to flood the target
   -

4.4. HTTP Exploitation with Slowloris

Slowloris is a tool used to launch HTTP denial-of-service (DoS) attacks by keeping connections open and consuming server resources.

Launch a Slowloris Attack:

- slowloris 192.168.1.10 -p 80
-

Summary of Exploitation Techniques:

- FTP: Upload and download files, test weak credentials.
- SSH: Log in with credentials obtained from Hydra.
- DDoS: Use hping3 to perform a DoS attack on HTTP.
- HTTP: Use Slowloris to exhaust server resources.

Detection

1. Footprinting

1.1 WHOIS Lookup

- **Attacker Command**:

whois 192.168.1.10

- **Alert**:

  This action cannot be detected by Suricata.

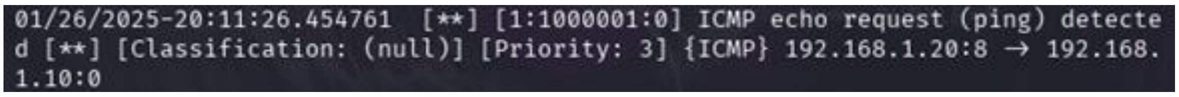
1.2 Ping Sweep

- **Attacker Command**:

  ping 192.168.1.10

- **Alert**:

  ICMP Echo Request (Ping) detected by Suricata.
  Signature: alert icmp any any -> any any (msg:"ICMP echo request (ping) detected"; itype:8; sid:1000001;)

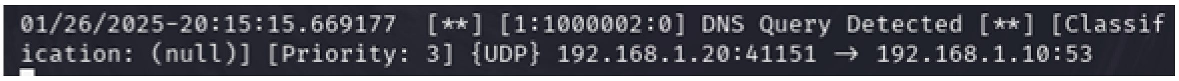


```
01/26/2025-20:11:26.454761  [**] [1:1000001:0] ICMP echo request (ping) detecte
d [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.1.20:8 → 192.168.
1.10:0
```

1.3 DNS Query (with dig)

- **Attacker Command**:
- dig @192.168.1.10 example.com
- **Alert**:

  DNS query detected.
  Signature: alert udp any any -> any 53 (msg:"DNS Query Detected"; content:"|00 00 01|"; sid:1000002;)



```
01/26/2025-20:15:15.669177  [**] [1:1000002:0] DNS Query Detected [**] [Classif
ication: (null)] [Priority: 3] {UDP} 192.168.1.20:41151 → 192.168.1.10:53
```

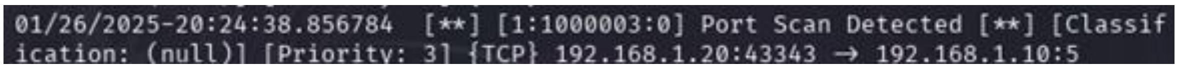
2. Scanning

2.1 Single Port Scan with Nmap

- **Attacker Command**:
- sudo nmap -p 80 192.168.1.10
- **Alert**:

  Port Scan Detected.

Signature: alert ip any any -> any any (msg:"Port Scan Detected"; flags:S; threshold:type threshold, track by_dst, count 10, seconds 5; sid:1000003;)



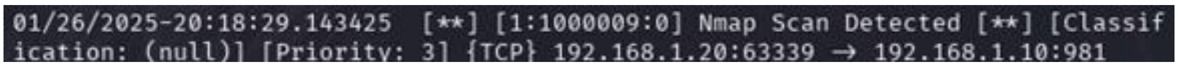
```
01/26/2025-20:24:38.856784  [**] [1:1000003:0] Port Scan Detected [**] [Classif
ication: (null)] [Priority: 3] {TCP} 192.168.1.20:43343 → 192.168.1.10:5
```

2.2 Full Port Scan with Nmap

- **Attacker Command**:
- sudo nmap 192.168.1.10
- **Alert**:

  Port Scan Detected.

  Signature: alert ip any any -> any any (msg:"Port Scan Detected"; flags:S; threshold:type threshold, track by_dst, count 10, seconds 5; sid:1000003;)



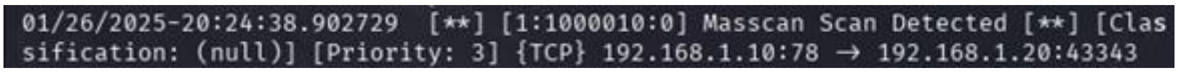
```
01/26/2025-20:18:29.143425  [**] [1:1000009:0] Nmap Scan Detected [**] [Classif
ication: (null)] [Priority: 3] {TCP} 192.168.1.20:63339 → 192.168.1.10:981
```

2.3 Masscan Full Port Scan

- **Attacker Command**:
- masscan 192.168.1.10 -p0-65535
- **Alert**:

  Masscan detected; abnormal scanning patterns.

  Signature: alert ip any any -> any any (msg:"Masscan detected"; sid:1000004;)



```
01/26/2025-20:24:38.902729  [**] [1:1000010:0] Masscan Scan Detected [**] [Clas
sification: (null)] [Priority: 3] {TCP} 192.168.1.10:78 → 192.168.1.20:43343
```

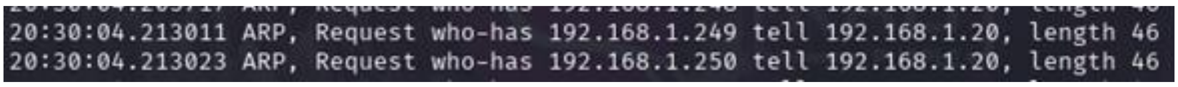
2.4 ARP Scan

- **Attacker Command**:
- arp -a
- **Alert**:

  Detected ARP traffic via tcpdump.

  Command:

- sudo tcpdump -i eth0 arp



```
20:30:04.213011 ARP, Request who-has 192.168.1.249 tell 192.168.1.20, length 46
20:30:04.213023 ARP, Request who-has 192.168.1.250 tell 192.168.1.20, length 46
```

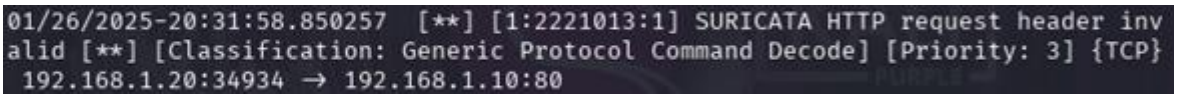
NETWORK SECURITY	Student information	Date	Project
	Matricola N. 0001/24312 Name: Carlos Sampedro Mdz.	25-1-25	Intrusion Detection System

2.5 HTTP Request with Curl

- **Attacker Command**:
- curl http://192.168.1.10
- **Alert**:

  HTTP Request Detected.

  Signature: alert http any any -> any any (msg:"HTTP request detected"; content:"GET"; sid:1000005;)



```
01/26/2025-20:31:58.850257  [**] [1:2221013:1] SURICATA HTTP request header inv
alid [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP}
 192.168.1.20:34934 → 192.168.1.10:80
```

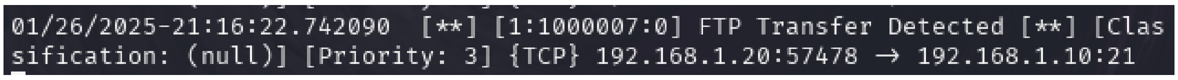
3. Enumeration

3.1 FTP Enumeration

- **Attacker Command**:
- ftp 192.168.1.10
- **Alert**:

  FTP Transfer Detected.

  Signature: alert tcp any any -> any 21 (msg:"FTP Transfer Detected"; sid:1000007;)



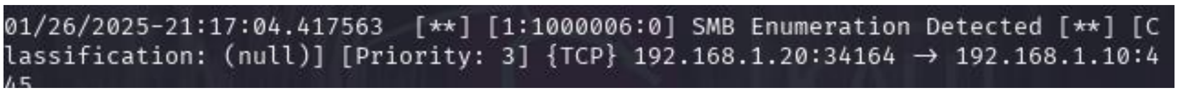
```
01/26/2025-21:16:22.742090  [**] [1:1000007:0] FTP Transfer Detected [**] [Clas
sification: (null)] [Priority: 3] {TCP} 192.168.1.20:57478 → 192.168.1.10:21
```

3.2 SMB Enumeration

- **Attacker Command**:
- enum4linux -a 192.168.1.10
- **Alert**:

  SMB Enumeration Detected.

  Signature: alert tcp any any -> any 445 (msg:"SMB Enumeration Detected"; sid:1000006;)



```
01/26/2025-21:17:04.417563  [**] [1:1000006:0] SMB Enumeration Detected [**] [C
lassification: (null)] [Priority: 3] {TCP} 192.168.1.20:34164 → 192.168.1.10:4
45
```

3.3 Cewl

- **Attacker Command**:

  Cewl http://192.168.1.10 -d 3 -w usernames.txt

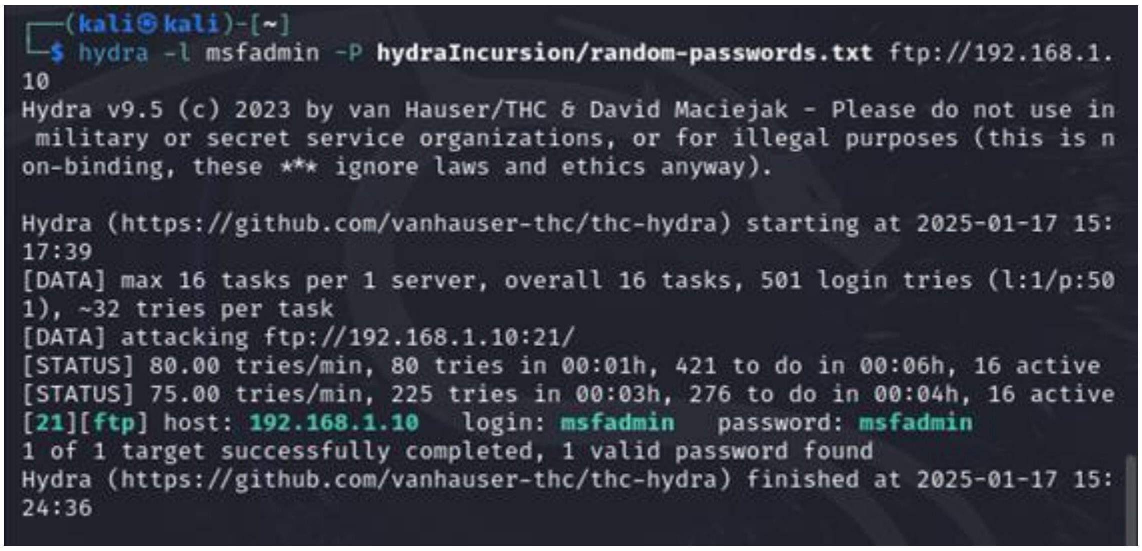
- **Alert**:

  Detects http requests and port scans

4. Exploitation

Firstly, we need to get the credentials using hydra

Command: hydra -l msfadmin -P hydraIncursion/random-passwords.txt ftp://192.168.1.10



```
┌──(kali㉿kali)-[~]
└─$ hydra -l msfadmin -P hydraIncursion/random-passwords.txt ftp://192.168.1.
10
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 n
on-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-01-17 15:
17:39
[DATA] max 16 tasks per 1 server, overall 16 tasks, 501 login tries (l:1/p:50
1), ~32 tries per task
[DATA] attacking ftp://192.168.1.10:21/
[STATUS] 80.00 tries/min, 80 tries in 00:01h, 421 to do in 00:06h, 16 active
[STATUS] 75.00 tries/min, 225 tries in 00:03h, 276 to do in 00:04h, 16 active
[21][ftp] host: 192.168.1.10   login: msfadmin   password: msfadmin
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-01-17 15:
24:36
```

With this we got the password and we can access freely FTP, SSH…

4.1 FTP Exploitation

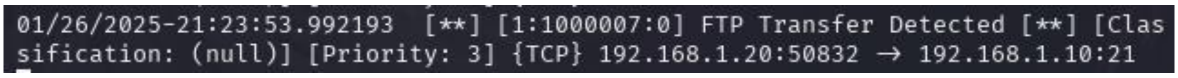
- **Attacker Command**:

  ftp 192.168.1.10

- **Alert**:

  FTP Transfer Detected.

  Signature: alert tcp any any -> any 21 (msg:"FTP Transfer Detected"; sid:1000007;)



```
01/26/2025-21:23:53.992193  [**] [1:1000007:0] FTP Transfer Detected [**] [Clas
sification: (null)] [Priority: 3] {TCP} 192.168.1.20:50832 → 192.168.1.10:21
```

4.2 SSH Exploitation

- **Attacker Command**:

  ssh -o HostKeyAlgorithms=+ssh-rsa msfadmin@192.168.1.10

- **Alert**:

  SSH Login Detected.

  Signature: alert tcp any any -> any 22 (msg:"SSH Login Detected"; sid:1000009;)

4.3 DDoS with hping3

- **Attacker Command**:

  hping3 -S --flood -p 80 192.168.1.10

- **Alert**:

  SYN Flood Detected.

  Signature: alert tcp any any -> any 80 (msg:"SYN Flood Detected"; flags:S; sid:1000010;

4.4 HTTP Exploitation with Slowloris

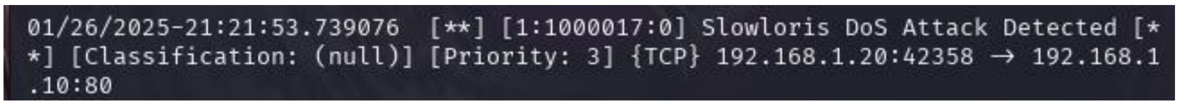
- **Attacker Command**:

  slowloris 192.168.1.10 -p 80 -s 1000 --sleeptime 200

- **Alert**:

  Slowloris Activity Detected.

  Signature: alert tcp any any -> any 80 (msg:"Slowloris Activity Detected"; sid:1000011;)



```
01/26/2025-21:21:53.739076  [**] [1:1000017:0] Slowloris DoS Attack Detected [*
*] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.20:42358 → 192.168.1
.10:80
```

Make Suricata work correctly

1. Configure Network Interface

Ensure all Virtual Machines (VMs) are configured to be in the same network, with the following IP addresses:

- **Victim:** 192.168.1.10
- **Attacker:** 192.168.1.20
- **Detection:** 192.168.1.30

For this, use the following command to ensure the network interfaces are up:

```
sudo ifconfig eth0 192.168.1.xx netmask 255.255.255.0 up
```

Replace xx with the respective digits (10, 20, or 30) for each VM.

2. Configure Suricata for Detection

- Open the Suricata configuration file:

  ```
  sudo nano /etc/suricata/suricata.yaml
  ```

- **Add the custom rules file** to the configuration under rule-files:

  ```
  rule-files:
    - /etc/suricata/rules/custom.rules
   - /etc/suricata/rules/suricata.rules
  ```

- **Configure the packet capture settings:** Make sure Suricata listens to the correct interface (in this case, eth0):

  ```
  af-packet:
    - interface: eth0
      cluster-id: 0
      cluster-type: cluster_flow
      fanout: no
  ```

3. Add Custom Rules

Now, add custom rules to detect intrusions:

```
sudo nano /etc/suricata/rules/custom.rules
```

Add some basic detection rules, for example:

```
# ICMP Echo Request (Ping) Detection
```

```
alert icmp any any -> any any (msg:"ICMP Echo Request (Ping) Detected"; itype:8; sid:1000001;)
```

4. Restart Suricata

After modifying the rules, restart Suricata:

```
sudo suricata -T -c /etc/suricata/suricata.yaml
sudo systemctl restart suricata
```

5. View Suricata Alerts

You can view the detection alerts generated by Suricata using the following commands:

```
sudo tail -f /var/log/suricata/eve.json
sudo tail -f /var/log/suricata/fast.log
```