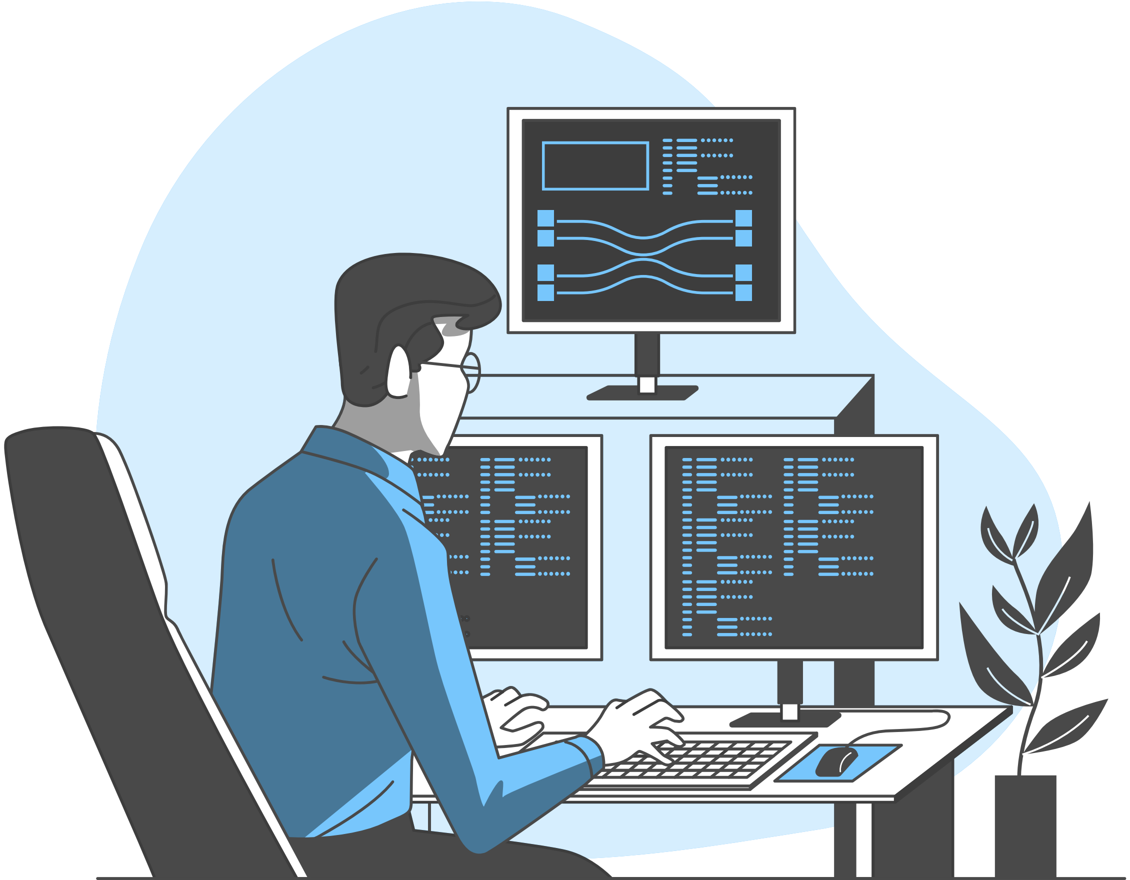


IDS Alert System

Ryan Becker

01
The Question

How can we build a wireless IDS to send real time alerts with suricata and python?

Limitations with Question

01 Managed Switch

Key to Port Mirroring router traffic for Datalink IDS

02 Virtual Switch

Got Close but lacked visibility of any attacks

03 TCPDump

Challenges with TCPDump implementation with python, acts too passive from what I found

04 Mobile Texts

Some platforms are free, but seemed off with format of site. So resorted away from text alerts

Slightly new Question: Can I create an IDS behind an AP, where a user can connect their devices and get real time alerts via Discord of network activity, while they are away?

Potential Limitations

01

DDoS

DDoS Delays alerts due to flooding via SYN or other packets. Therefore, does not correctly alert in real time.

02

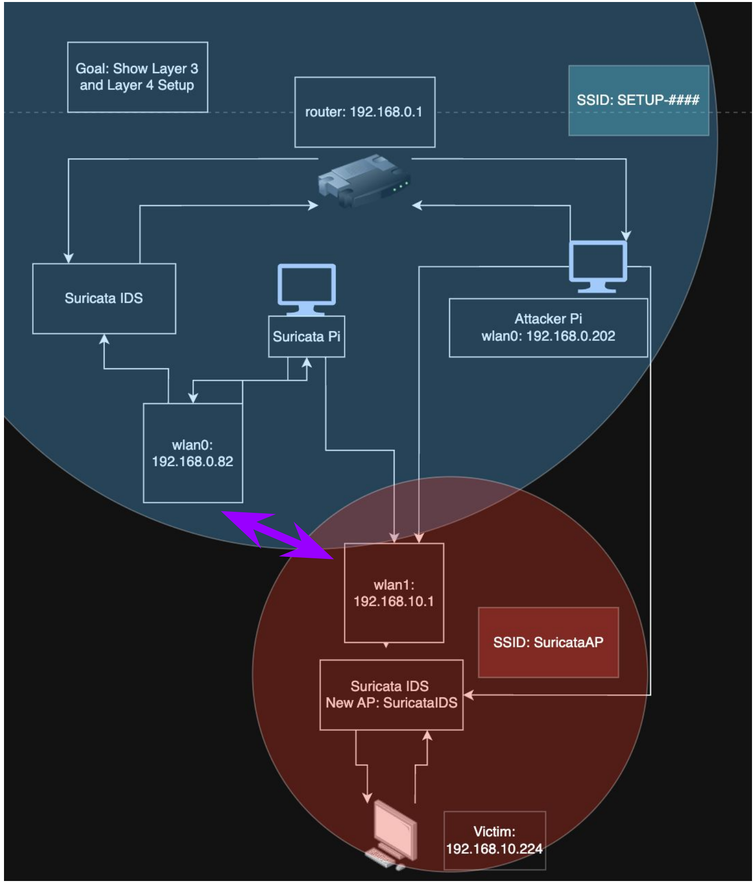
ARP Attack

Will not alert if there is an ARP MItM attack on the SETUP-#### SSID. This is due to suricata being specific to SuricataAP.

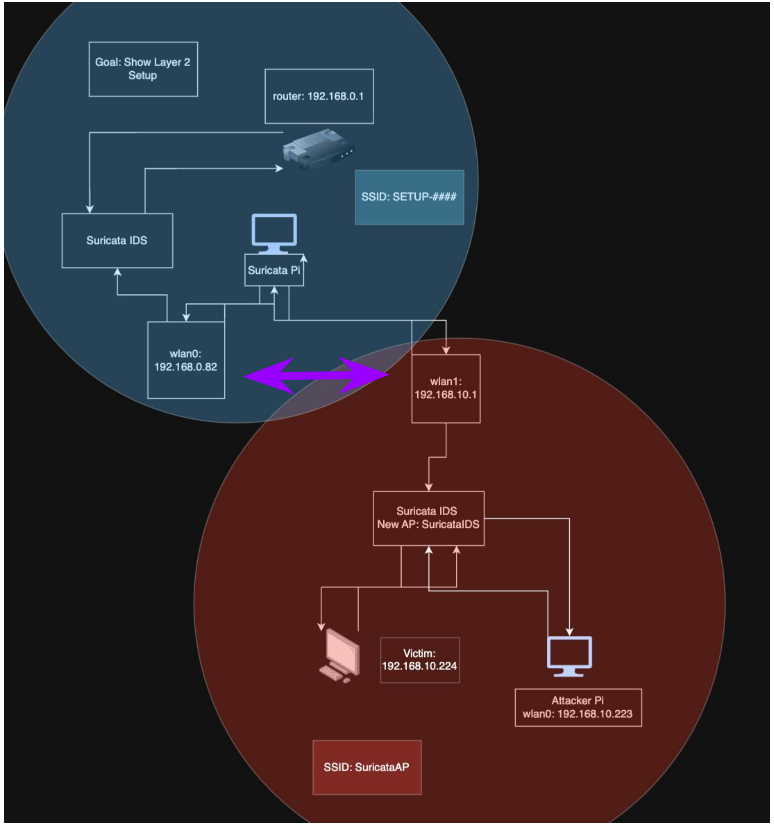
02

Topology

Layer 3 and 4 IDS Setup



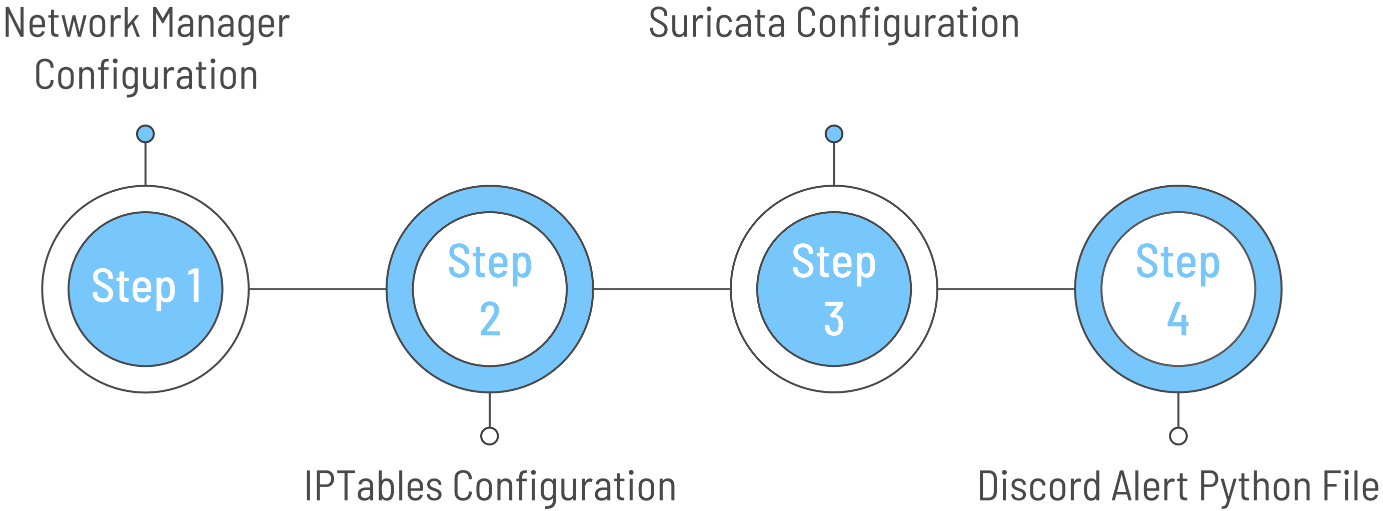
Layer 2 IDS Setup



03

Suricata PI Setup

Suricata Process



Step 1

Network Manager Configuration

Step 2

IPTables Configuration

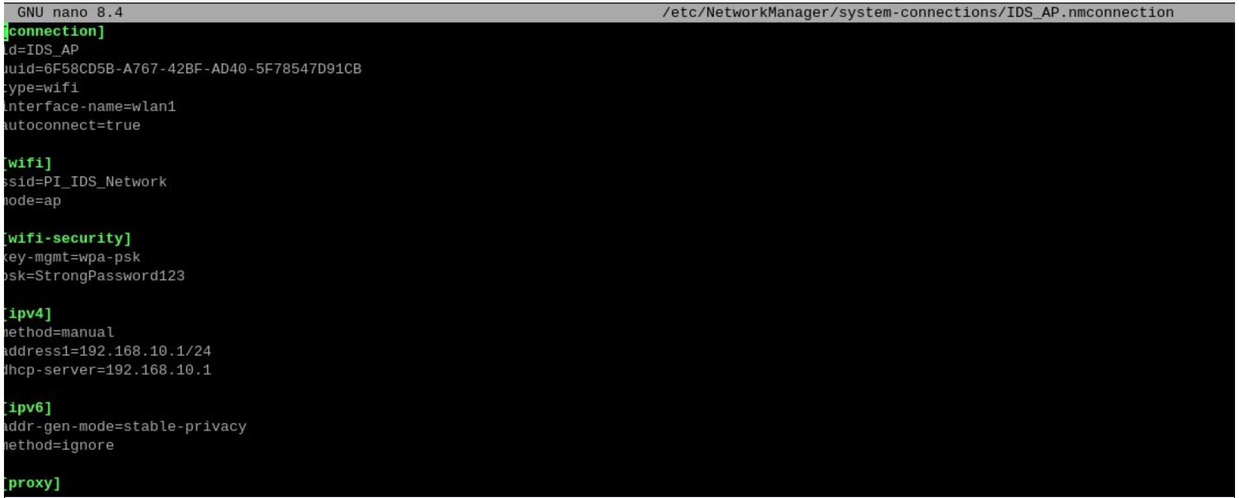
Step 3

Suricata Configuration

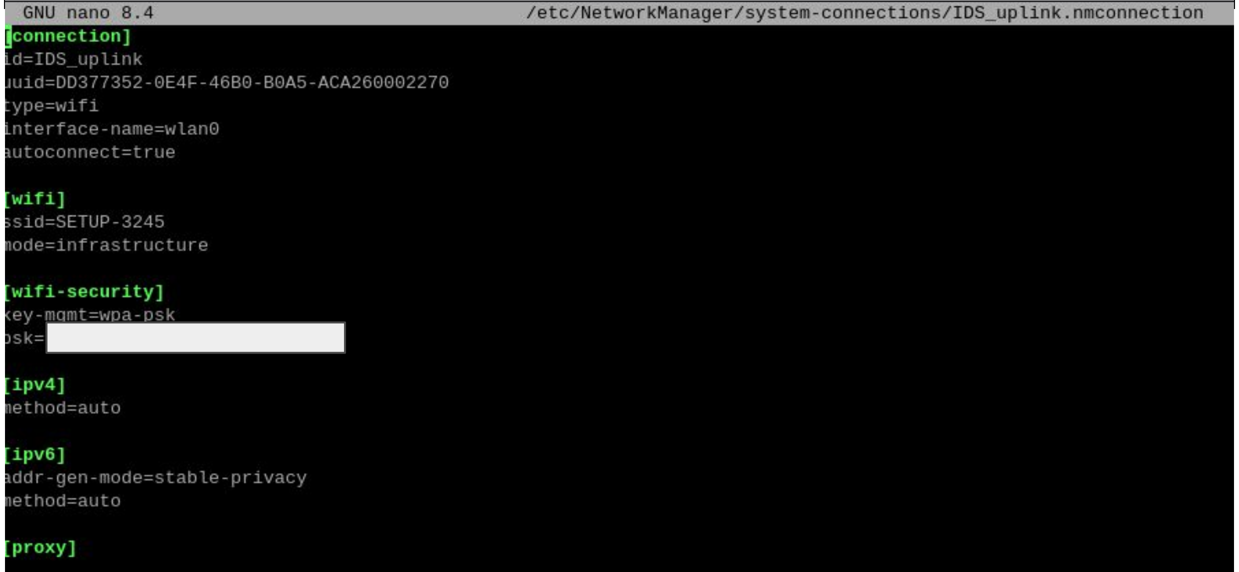
Step 4

Discord Alert Python File

Step 1

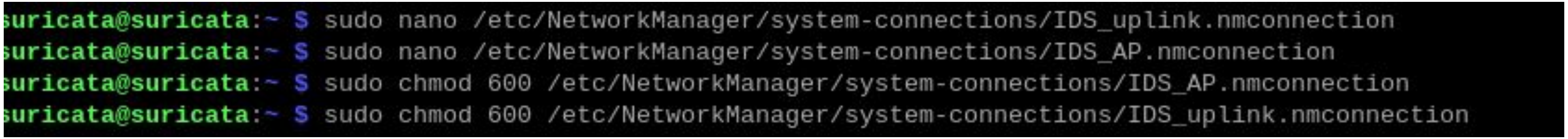


```
GNU nano 8.4                    /etc/NetworkManager/system-connections/IDS_AP.nmconnection
[connection]
id=IDS_AP
uuid=6F58CD5B-A767-42BF-AD40-5F78547D91CB
type=wifi
interface-name=wlan1
autoconnect=true

[wifi]
ssid=PI_IDS_Network
mode=ap

[wifi-security]
key-mgmt=wpa-psk
psk=StrongPassword123

[ipv4]
method=manual
address1=192.168.10.1/24
dhcp-server=192.168.10.1

[ipv6]
addr-gen-mode=stable-privacy
method=ignore

[proxy]
```



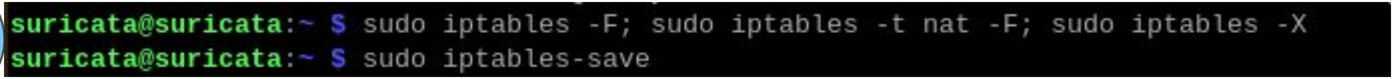
```
GNU nano 8.4                /etc/NetworkManager/system-connections/IDS_uplink.nmconnection
[connection]
id=IDS_uplink
uuid=DD377352-0E4F-46B0-B0A5-ACA260002270
type=wifi
interface-name=wlan0
autoconnect=true

[wifi]
ssid=SETUP-3245
mode=infrastructure

[wifi-security]
key-mgmt=wpa-psk
psk=[redacted]

[ipv4]
method=auto

[ipv6]
addr-gen-mode=stable-privacy
method=auto

[proxy]
```

Step 1

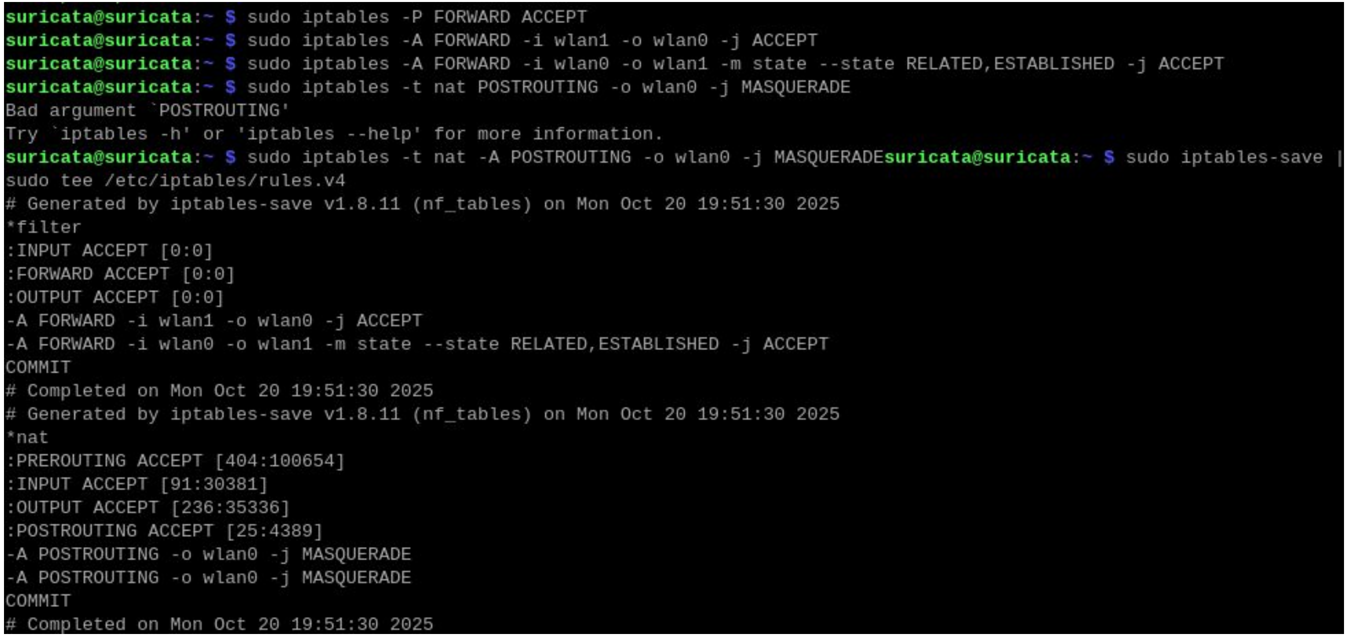


```
suricata@suricata:~ $ sudo nano /etc/NetworkManager/system-connections/IDS_uplink.nmconnection
suricata@suricata:~ $ sudo nano /etc/NetworkManager/system-connections/IDS_AP.nmconnection
suricata@suricata:~ $ sudo chmod 600 /etc/NetworkManager/system-connections/IDS_AP.nmconnection
suricata@suricata:~ $ sudo chmod 600 /etc/NetworkManager/system-connections/IDS_uplink.nmconnection
```

Step 2

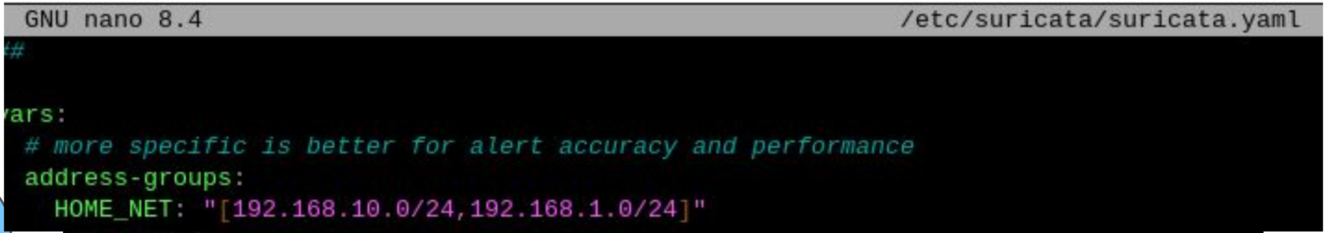


```
suricata@suricata:~ $ sudo iptables -F; sudo iptables -t nat -F; sudo iptables -X
suricata@suricata:~ $ sudo iptables-save
```

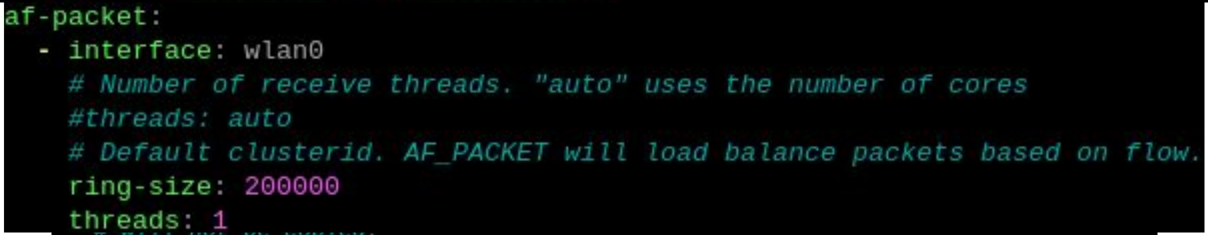


```
suricata@suricata:~ $ sudo iptables -P FORWARD ACCEPT
suricata@suricata:~ $ sudo iptables -A FORWARD -i wlan1 -o wlan0 -j ACCEPT
suricata@suricata:~ $ sudo iptables -A FORWARD -i wlan0 -o wlan1 -m state --state RELATED,ESTABLISHED -j ACCEPT
suricata@suricata:~ $ sudo iptables -t nat POSTROUTING -o wlan0 -j MASQUERADE
Bad argument `POSTROUTING'
Try `iptables -h' or 'iptables --help' for more information.
suricata@suricata:~ $ sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADEsuricata@suricata:~ $ sudo iptables-save |
sudo tee /etc/iptables/rules.v4
# Generated by iptables-save v1.8.11 (nf_tables) on Mon Oct 20 19:51:30 2025
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -i wlan1 -o wlan0 -j ACCEPT
-A FORWARD -i wlan0 -o wlan1 -m state --state RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Mon Oct 20 19:51:30 2025
# Generated by iptables-save v1.8.11 (nf_tables) on Mon Oct 20 19:51:30 2025
*nat
:PREROUTING ACCEPT [404:100654]
:INPUT ACCEPT [91:30381]
:OUTPUT ACCEPT [236:35336]
:POSTROUTING ACCEPT [25:4389]
-A POSTROUTING -o wlan0 -j MASQUERADE
-A POSTROUTING -o wlan0 -j MASQUERADE
COMMIT
# Completed on Mon Oct 20 19:51:30 2025
```

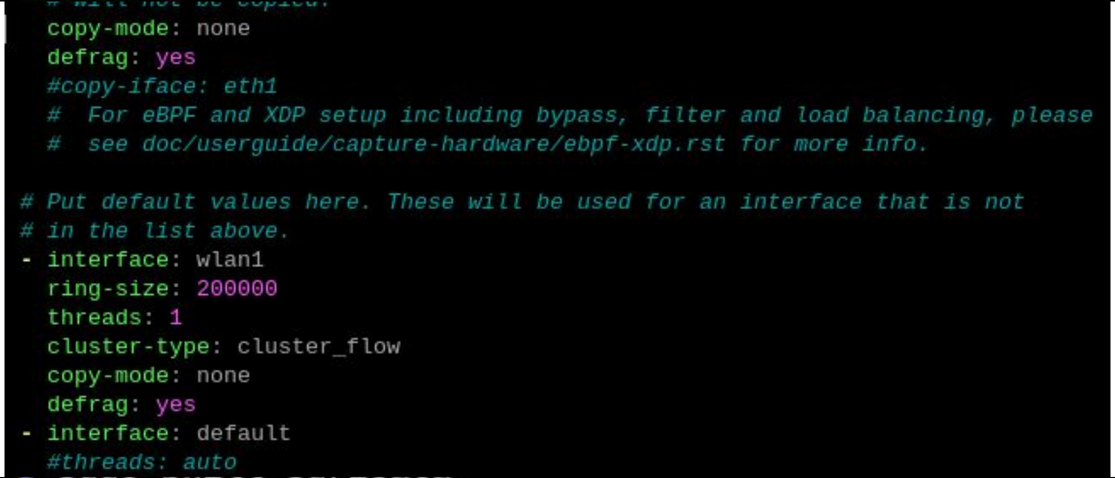
Step 3



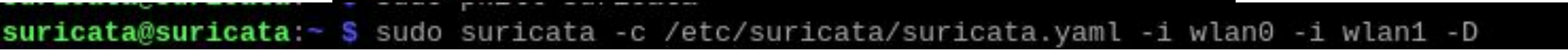
```
GNU nano 8.4                                        /etc/suricata/suricata.yaml
##

vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[192.168.10.0/24,192.168.1.0/24]"
```



```
af-packet:
  - interface: wlan0
    # Number of receive threads. "auto" uses the number of cores
    #threads: auto
    # Default clusterid. AF_PACKET will load balance packets based on flow.
    ring-size: 200000
    threads: 1
```

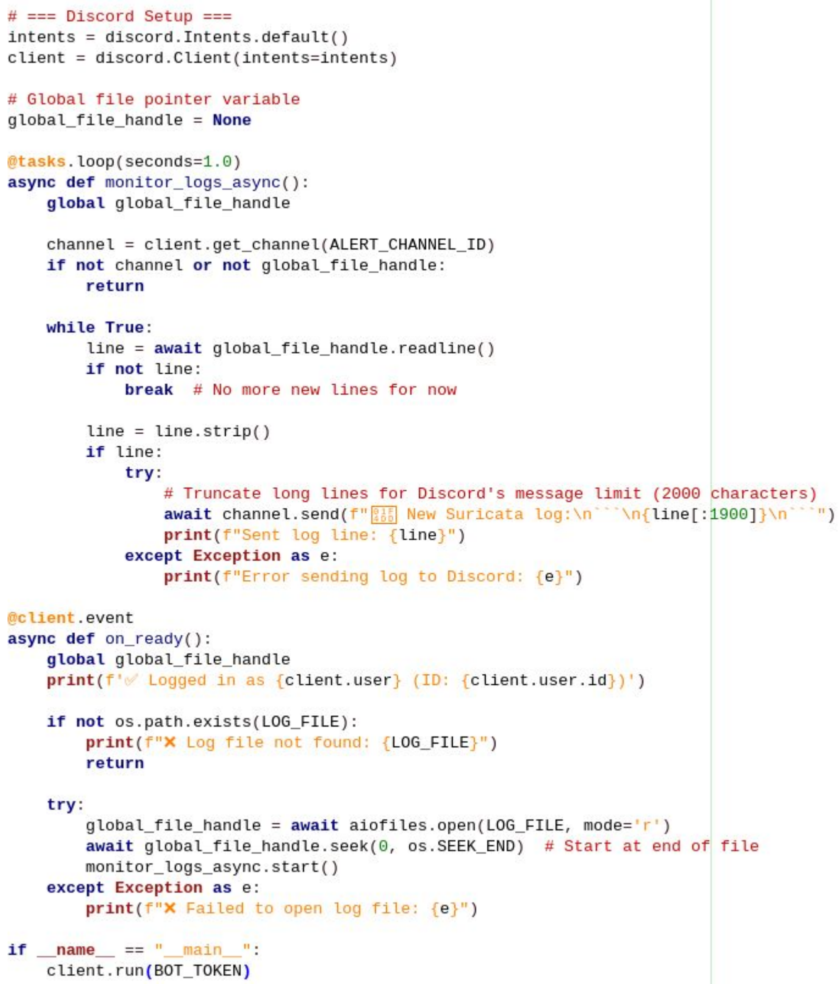


```
    # will not be copied.
    copy-mode: none
    defrag: yes
    #copy-iface: eth1
    #  For eBPF and XDP setup including bypass, filter and load balancing, please
    #  see doc/userguide/capture-hardware/ebpf-xdp.rst for more info.

  # Put default values here. These will be used for an interface that is not
  # in the list above.
  - interface: wlan1
    ring-size: 200000
    threads: 1
    cluster-type: cluster_flow
    copy-mode: none
    defrag: yes
  - interface: default
    #threads: auto
```



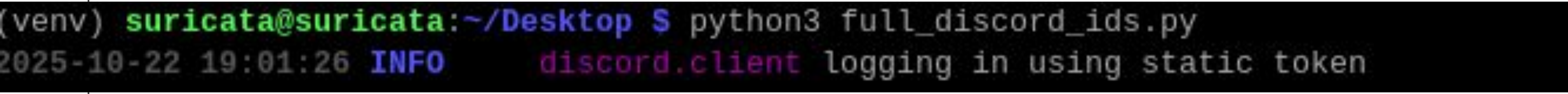
```
suricata@suricata:~ $ sudo suricata -c /etc/suricata/suricata.yaml -i wlan0 -i wlan1 -D
```

Step 4

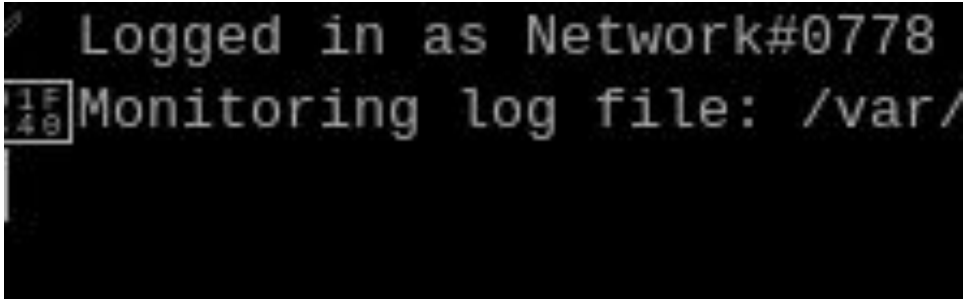


```python
# === Discord Setup ===
intents = discord.Intents.default()
client = discord.Client(intents=intents)

# Global file pointer variable
global_file_handle = None

@tasks.loop(seconds=1.0)
async def monitor_logs_async():
    global global_file_handle

    channel = client.get_channel(ALERT_CHANNEL_ID)
    if not channel or not global_file_handle:
        return

    while True:
        line = await global_file_handle.readline()
        if not line:
            break  # No more new lines for now

        line = line.strip()
        if line:
            try:
                # Truncate long lines for Discord's message limit (2000 characters)
                await channel.send(f"📝 New Suricata log:\n```\n{line[:1900]}\n```")
                print(f"Sent log line: {line}")
            except Exception as e:
                print(f"Error sending log to Discord: {e}")

@client.event
async def on_ready():
    global global_file_handle
    print(f'✅ Logged in as {client.user} (ID: {client.user.id})')

    if not os.path.exists(LOG_FILE):
        print(f"❌ Log file not found: {LOG_FILE}")
        return

    try:
        global_file_handle = await aiofiles.open(LOG_FILE, mode='r')
        await global_file_handle.seek(0, os.SEEK_END)  # Start at end of file
        monitor_logs_async.start()
    except Exception as e:
        print(f"❌ Failed to open log file: {e}")

if __name__ == "__main__":
    client.run(BOT_TOKEN)
```

Step 4



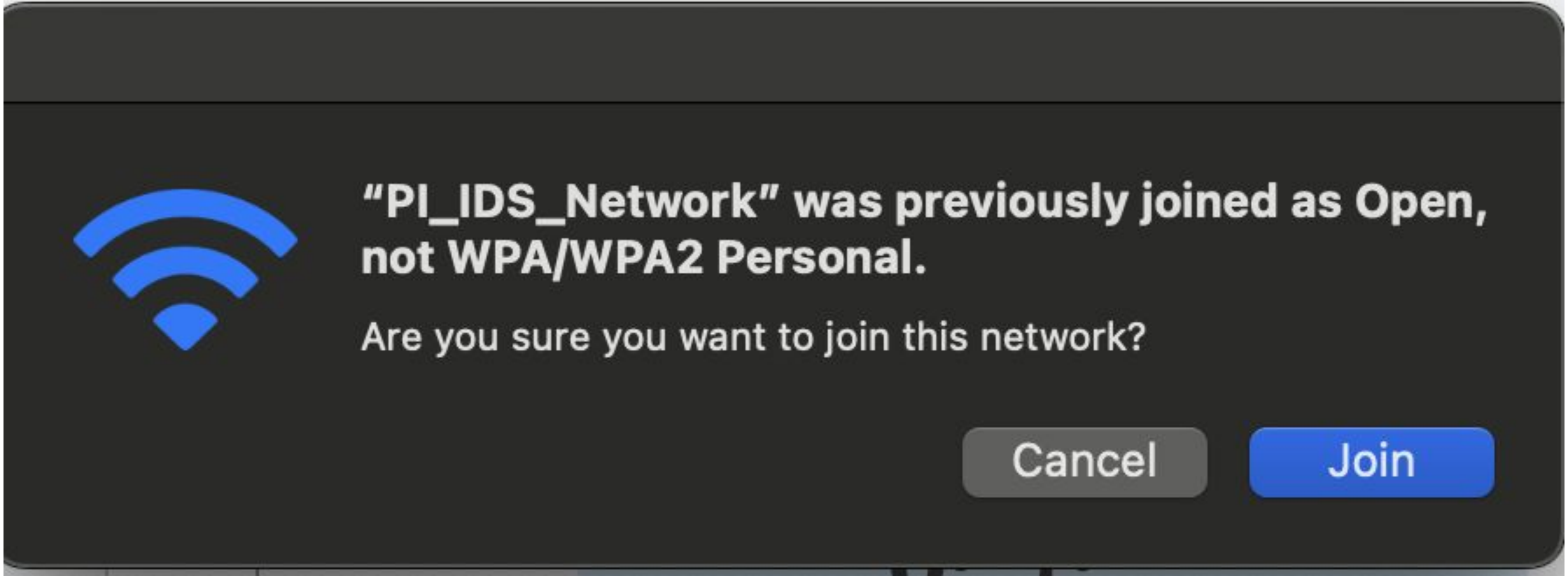
```
(venv) suricata@suricata:~/Desktop $ python3 full_discord_ids.py
2025-10-22 19:01:26 INFO     discord.client logging in using static token
```



```
  Logged in as Network#0778
  Monitoring log file: /var/
```

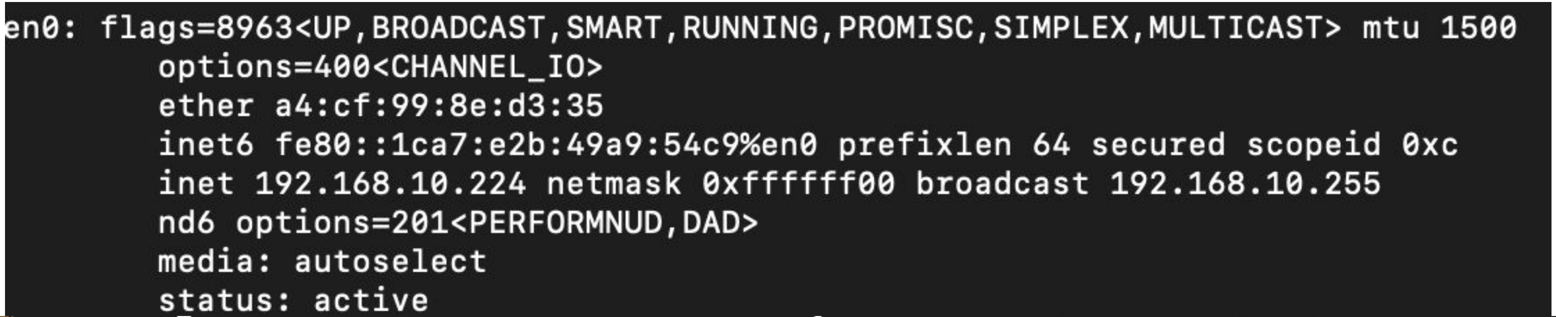
04

Victim Device Setup

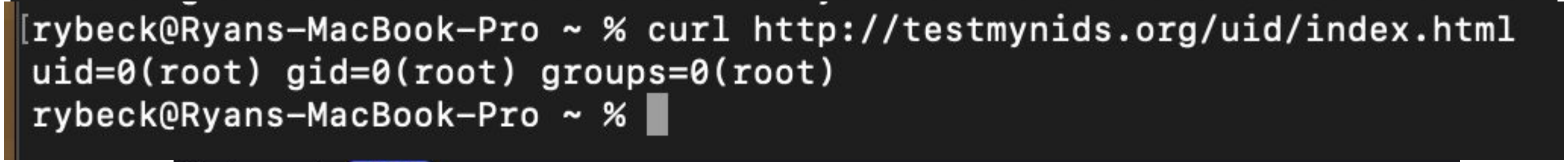


“PI_IDS_Network” was previously joined as Open, not WPA/WPA2 Personal.

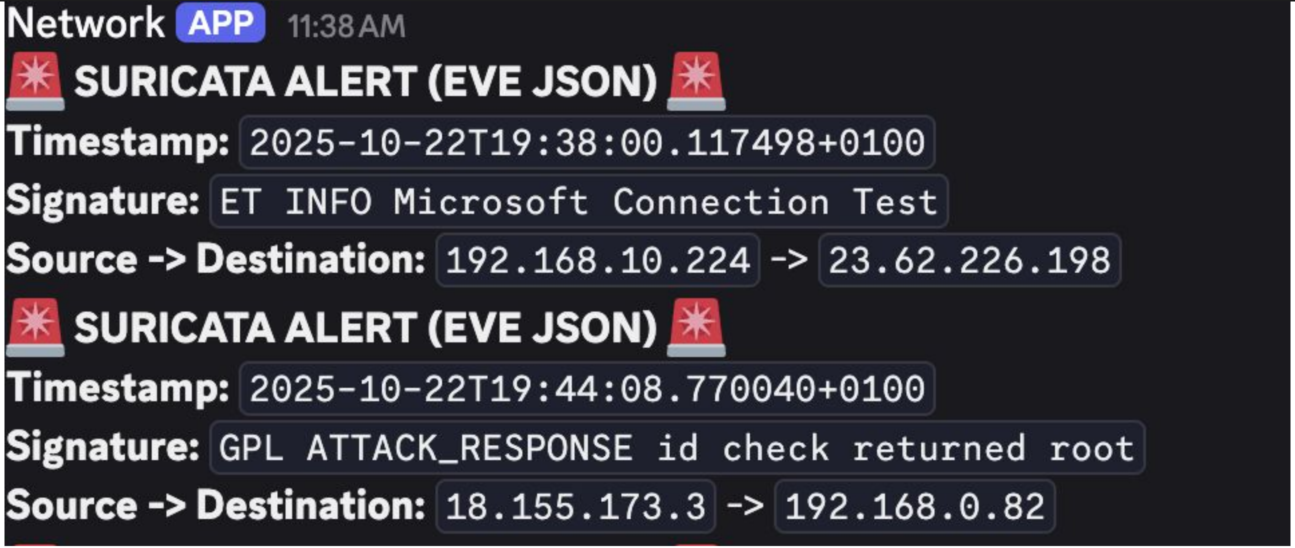
Are you sure you want to join this network?



```
en0: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether a4:cf:99:8e:d3:35
        inet6 fe80::1ca7:e2b:49a9:54c9%en0 prefixlen 64 secured scopeid 0xc
        inet 192.168.10.224 netmask 0xffffff00 broadcast 192.168.10.255
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
```



```
rybeck@Ryans-MacBook-Pro ~ % curl http://testmynids.org/uid/index.html
uid=0(root) gid=0(root) groups=0(root)
rybeck@Ryans-MacBook-Pro ~ %
```



Network APP 11:38 AM

🚨 **SURICATA ALERT (EVE JSON)** 🚨

Timestamp: `2025-10-22T19:38:00.117498+0100`

Signature: `ET INFO Microsoft Connection Test`

Source -> Destination: `192.168.10.224` -> `23.62.226.198`

🚨 **SURICATA ALERT (EVE JSON)** 🚨

Timestamp: `2025-10-22T19:44:08.770040+0100`

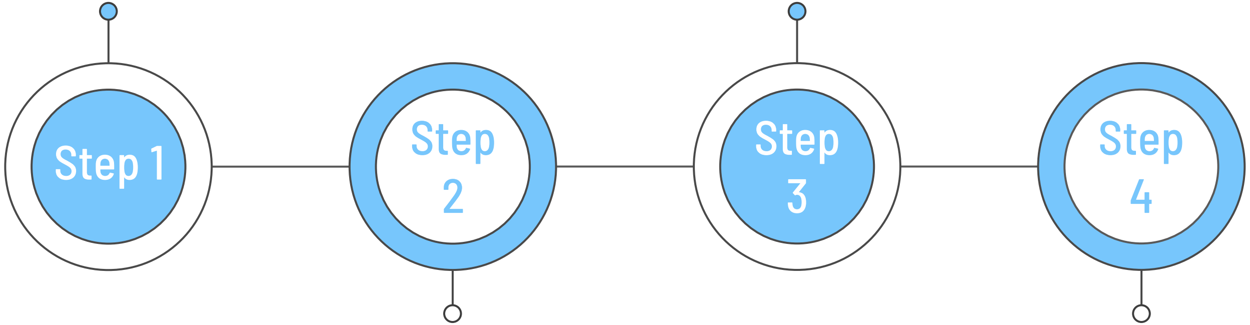
Signature: `GPL ATTACK_RESPONSE id check returned root`

Source -> Destination: `18.155.173.3` -> `192.168.0.82`

05

Attacker PI Setup

Attacker Pi Process



Step 1

Use Nmap to map the network (wlan1)

Step 2

Use Nmap to map the network (wlan0)

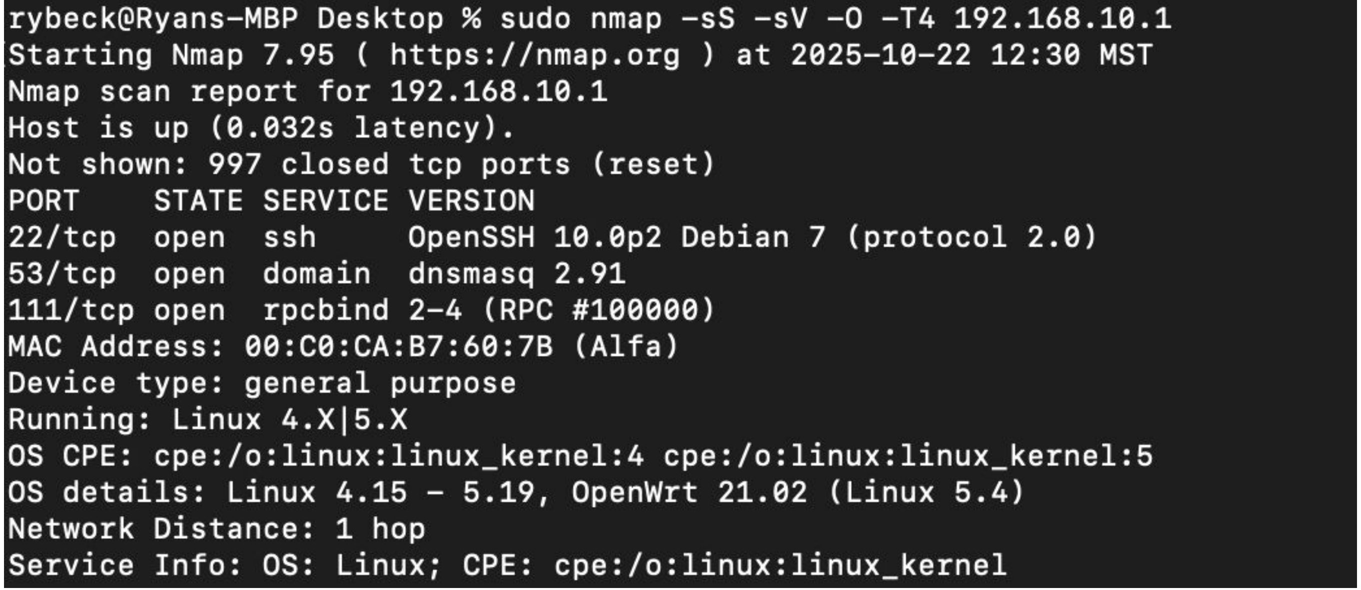
Step 3

DDoS attack against wlan0

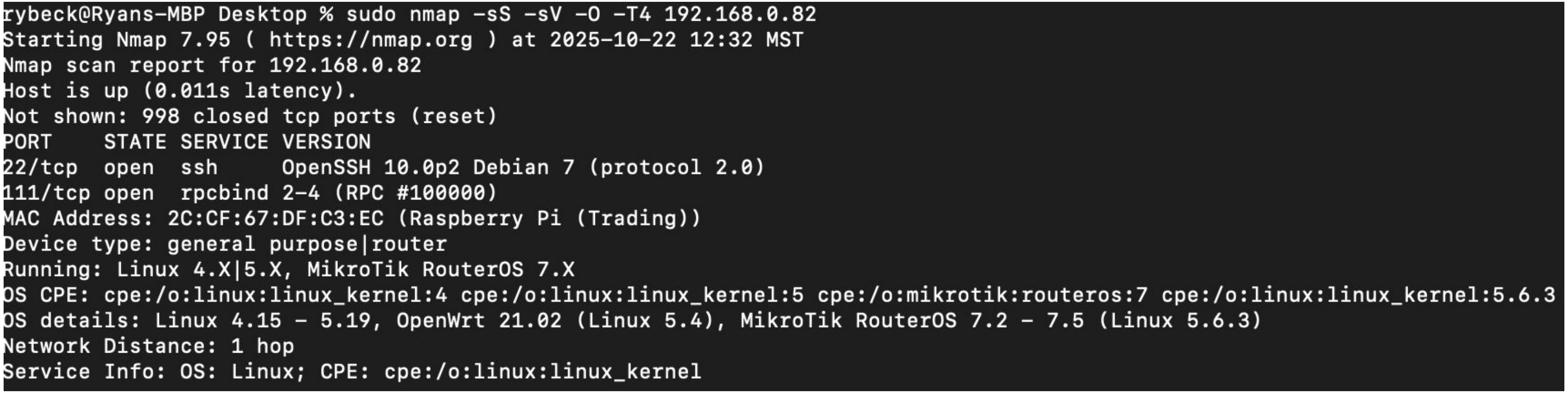
Step 4

Inspect Logs

Step 1

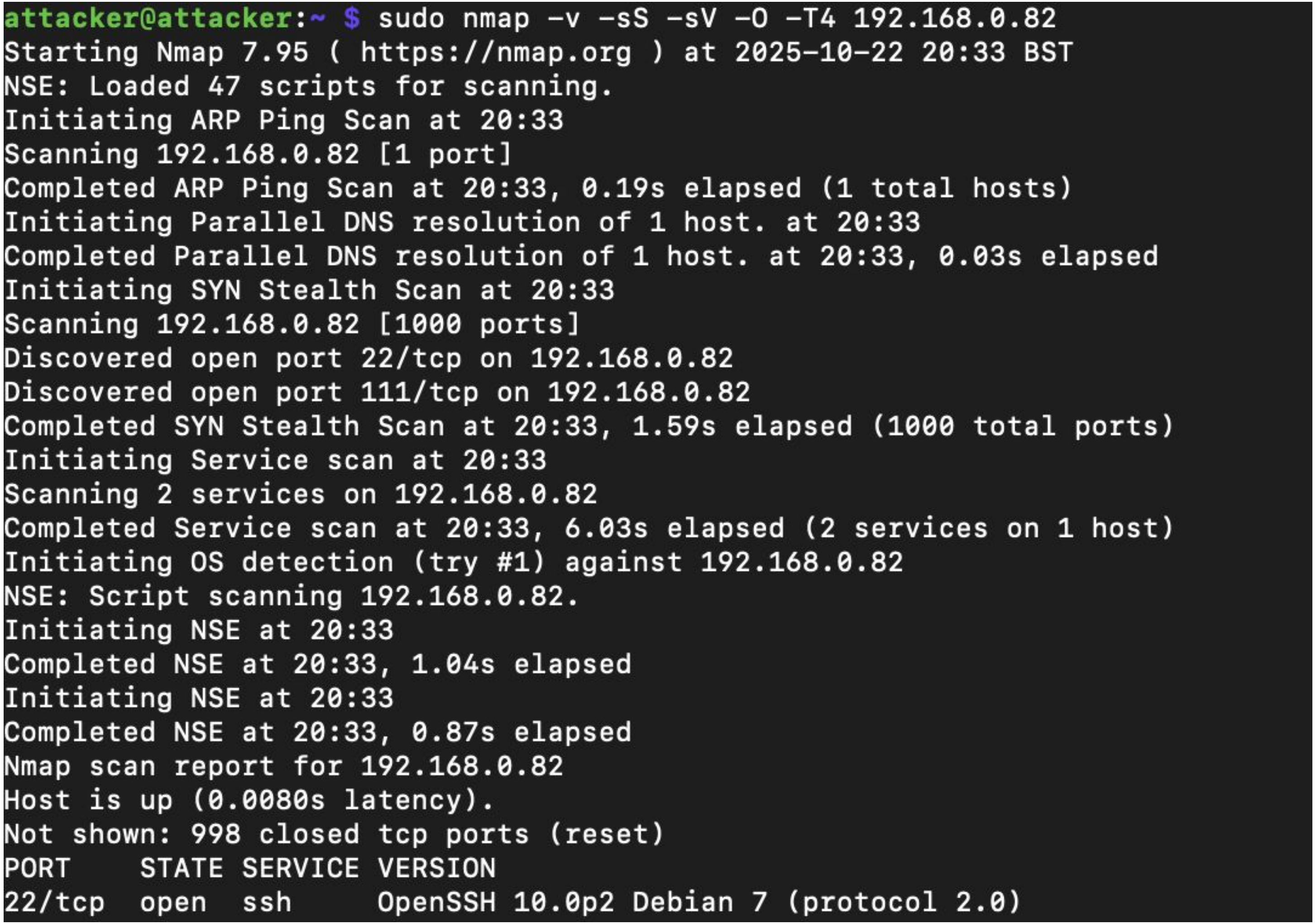


```
rybeck@Ryans-MBP Desktop % sudo nmap -sS -sV -O -T4 192.168.10.1
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-22 12:30 MST
Nmap scan report for 192.168.10.1
Host is up (0.032s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 10.0p2 Debian 7 (protocol 2.0)
53/tcp  open  domain  dnsmasq 2.91
111/tcp open  rpcbind 2-4 (RPC #100000)
MAC Address: 00:C0:CA:B7:60:7B (Alfa)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4)
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```



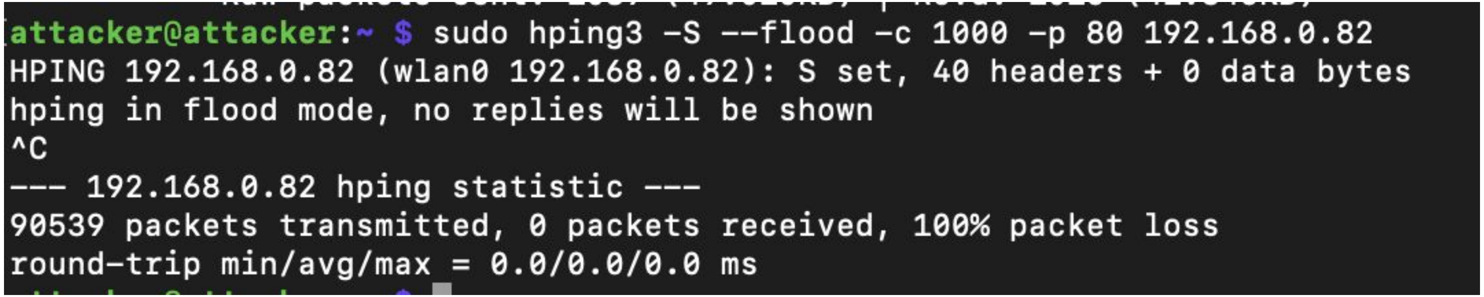
```
rybeck@Ryans-MBP Desktop % sudo nmap -sS -sV -O -T4 192.168.0.82
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-22 12:32 MST
Nmap scan report for 192.168.0.82
Host is up (0.011s latency).
Not shown: 998 closed tcp ports (reset)
PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 10.0p2 Debian 7 (protocol 2.0)
111/tcp open  rpcbind 2-4 (RPC #100000)
MAC Address: 2C:CF:67:DF:C3:EC (Raspberry Pi (Trading))
Device type: general purpose|router
Running: Linux 4.X|5.X, MikroTik RouterOS 7.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5 cpe:/o:mikrotik:routeros:7 cpe:/o:linux:linux_kernel:5.6.3
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4), MikroTik RouterOS 7.2 - 7.5 (Linux 5.6.3)
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Step 2

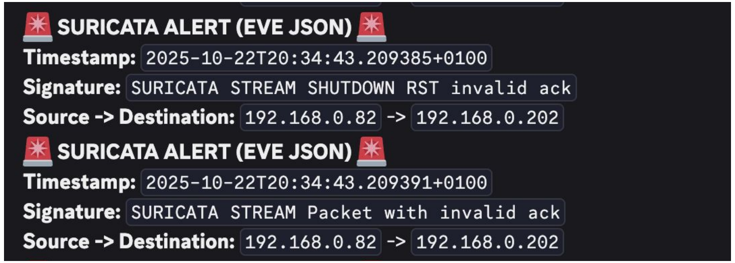


```
attacker@attacker:~ $ sudo nmap -v -sS -sV -O -T4 192.168.0.82
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-22 20:33 BST
NSE: Loaded 47 scripts for scanning.
Initiating ARP Ping Scan at 20:33
Scanning 192.168.0.82 [1 port]
Completed ARP Ping Scan at 20:33, 0.19s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 20:33
Completed Parallel DNS resolution of 1 host. at 20:33, 0.03s elapsed
Initiating SYN Stealth Scan at 20:33
Scanning 192.168.0.82 [1000 ports]
Discovered open port 22/tcp on 192.168.0.82
Discovered open port 111/tcp on 192.168.0.82
Completed SYN Stealth Scan at 20:33, 1.59s elapsed (1000 total ports)
Initiating Service scan at 20:33
Scanning 2 services on 192.168.0.82
Completed Service scan at 20:33, 6.03s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against 192.168.0.82
NSE: Script scanning 192.168.0.82.
Initiating NSE at 20:33
Completed NSE at 20:33, 1.04s elapsed
Initiating NSE at 20:33
Completed NSE at 20:33, 0.87s elapsed
Nmap scan report for 192.168.0.82
Host is up (0.0080s latency).
Not shown: 998 closed tcp ports (reset)
PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 10.0p2 Debian 7 (protocol 2.0)
```

Step 3



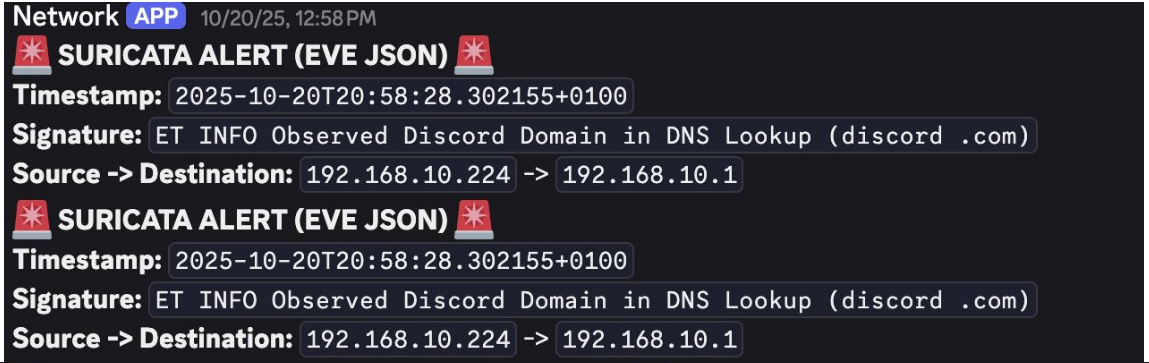
```
attacker@attacker:~ $ sudo hping3 -S --flood -c 1000 -p 80 192.168.0.82
HPING 192.168.0.82 (wlan0 192.168.0.82): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.0.82 hping statistic ---
90539 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```



🚨 **SURICATA ALERT (EVE JSON)** 🚨
Timestamp: `2025-10-22T20:34:43.209385+0100`
Signature: `SURICATA STREAM SHUTDOWN RST invalid ack`
Source -> Destination: `192.168.0.82` -> `192.168.0.202`

🚨 **SURICATA ALERT (EVE JSON)** 🚨
Timestamp: `2025-10-22T20:34:43.209391+0100`
Signature: `SURICATA STREAM Packet with invalid ack`
Source -> Destination: `192.168.0.82` -> `192.168.0.202`

Step 4



Network APP 10/20/25, 12:58 PM

🚨 **SURICATA ALERT (EVE JSON)** 🚨

Timestamp: `2025-10-20T20:58:28.302155+0100`

Signature: `ET INFO Observed Discord Domain in DNS Lookup (discord .com)`

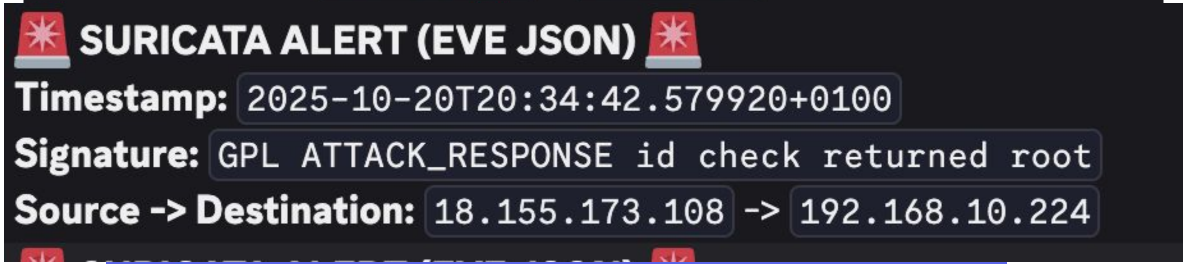
Source -> Destination: `192.168.10.224` -> `192.168.10.1`

🚨 **SURICATA ALERT (EVE JSON)** 🚨

Timestamp: `2025-10-20T20:58:28.302155+0100`

Signature: `ET INFO Observed Discord Domain in DNS Lookup (discord .com)`

Source -> Destination: `192.168.10.224` -> `192.168.10.1`

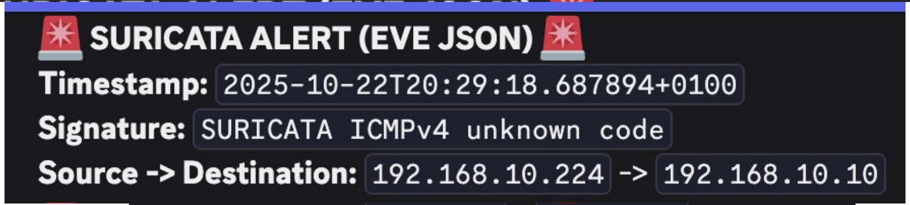


🚨 **SURICATA ALERT (EVE JSON)** 🚨

Timestamp: `2025-10-20T20:34:42.579920+0100`

Signature: `GPL ATTACK_RESPONSE id check returned root`

Source -> Destination: `18.155.173.108` -> `192.168.10.224`

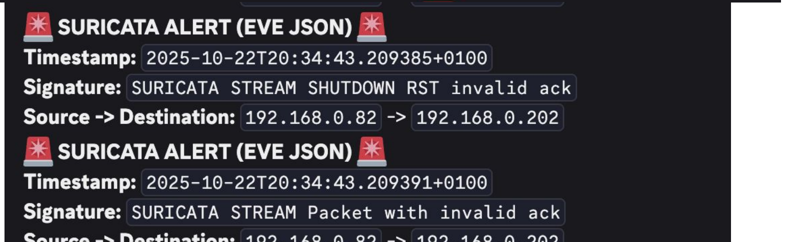


🚨 **SURICATA ALERT (EVE JSON)** 🚨

Timestamp: `2025-10-22T20:29:18.687894+0100`

Signature: `SURICATA ICMPv4 unknown code`

Source -> Destination: `192.168.10.224` -> `192.168.10.10`



🚨 **SURICATA ALERT (EVE JSON)** 🚨

Timestamp: `2025-10-22T20:34:43.209385+0100`

Signature: `SURICATA STREAM SHUTDOWN RST invalid ack`

Source -> Destination: `192.168.0.82` -> `192.168.0.202`

🚨 **SURICATA ALERT (EVE JSON)** 🚨

Timestamp: `2025-10-22T20:34:43.209391+0100`

Signature: `SURICATA STREAM Packet with invalid ack`

Source -> Destination: `192.168.0.82` -> `192.168.0.202`

Thank You!

Questions?