

04. AWS SecOnion Lab Setup - Traffic Mirroring for SOC

AWS SecOnion Lab Setup - Traffic Mirroring for SOC

Things to do:

Create Mirror Targets

Create Mirror Filters

Create a Mirror Session

Click on VPC and scroll down on the left pane to "Traffic mirroring"



Services



Search



VPC



EC2



IAM



Verified Access groups [New](#)

Verified Access endpoints

[New](#)

▼ Transit gateways

Transit gateways

Transit gateway
attachments

Transit gateway policy
tables

Transit gateway route
tables

Transit gateway multicast

▼ Traffic Mirroring

Mirror sessions



Mirror targets

Mirror filters

Creating a Mirror Target

Click on Mirror target > Create mirror target

The mirror target will be the Elastic Monitoring/Sniffing interface:

 eni-060070f0ecccea0af	1	←	0	-	-	10.0.6.30	ip-1
 eni-0306c7e73097a3d1d	0		0	-	44.219.46.44	10.0.1.106	ip-1

In the "Choose target" section ensure the following is edited:

Target type: Network Interface

Target: Your sniffing interface you created earlier:

Create traffic mirror target

Target settings
A description to help you identify the traffic mirror target

Name tag - *optional*

Description - *optional*

Choose target
Target type cannot be modified after creation.

Target type

Network Interface

Target

Click Save

Creating Mirror Filters

Next, in the left pane, click "Mirror filters"

Click Create traffic mirror filters

Traffic mirror filters Info
TrafficMirrorFilters

1

Create traffic mirror filter

Name	Filter ID	Description
------	-----------	-------------

Inbound Rule:

Inbound rule

ModifyTrafficMirrorFilterRule

10

Description - optional

Describe the rule.

Rule action

Select the action to take.

accept

Protocol

Select the IP traffic protocol.

All protocols

Source port range - optional

Enter the source address port range.

N/A

Destination port range - optional

Enter the destination address port range.

N/A

Source CIDR block

Enter the source CIDR block.

0.0.0.0/0

Destination CIDR block

Enter the destination CIDR block.

0.0.0.0/0

Outbound Rule:

Outbound rule

Rule number

100

Description - *optional*

Describe the rule.

Rule action

Select the action to take.

accept

Protocol

Select the IP traffic protocol.

All protocols

Source port range - *optional*

Enter the source address port range.

N/A

Destination port range - *optional*

Enter the destination address port range.

N/A

Source CIDR block

Enter the source CIDR block.

0.0.0.0/0

Destination CIDR block

Enter the destination CIDR block.

0.0.0.0/0

Save it and then go to next section.

Creating a Mirror Session


Next, in the left pane, click "Mirror sessions"

Click "Create a mirror session"



Edits to make:

Tag - Enter a session name if you'll have more than one session

Mirror Source: Linux interface

Filter network interfaces									
Interface ID	Device index	Card index	Description	Public IPv4 address	Private IPv4 address	Private IPv4 DNS	IPv6 addresses	Primary IPv6 address	IP
 eni-0e2814d8c95f5db8b	0	0	-	18.207.181.59	10.0.8.173	ip-10-0-8-173.ec2.int...	-	-	-

Mirror Target: Sniffing Interface

 eni-060070f0ecccea0af	1	← 0	—	—	10.0.6.30	ip-1
 eni-0306c7e73097a3d1d	0	0	—	44.219.46.44	10.0.1.106	ip-1

Session Settings

Create traffic mirror session

Session settings
Set description, source, and target.

Name tag - *optional*

Description - *optional*

Mirror source
The resource that you want to monitor.

Only network interfaces of type "interface" are allowed.

Mirror target
A network interface, or a network load balancer, or a gateway load balancer endpoint that is the destination for mirrored traffic.

Additional Settings

Additional settings

Set priority, packet length, etc.

Session number
The order sessions for the same resource are evaluated

Number between 1 and 32766

VNI - *optional*
The unique VXLAN network identifier that is included in the encapsulated mirrored packet that is sent to the target.

Number between 0 and 16777215

Packet length - *optional*
The number of bytes in each packet to mirror.

If not specified, the entire packet will be mirrored

Filter
Determines what traffic gets mirrored.

Mirror_Filter

Final Checks

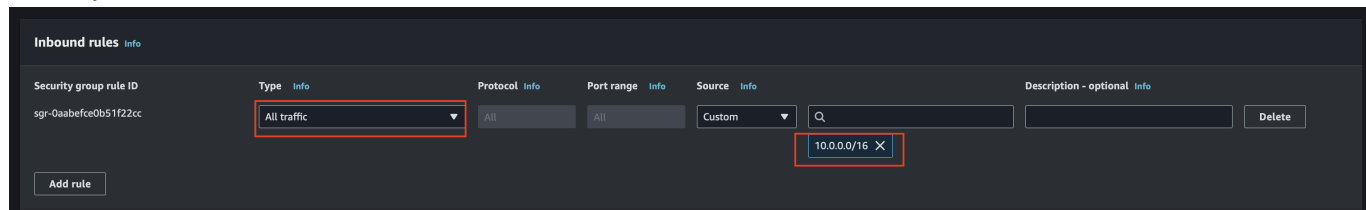
Ensuring that all security groups have an inbound rule that allows all internal traffic through the VPC:

Linux_SG

Manager_SG

Sniffing_SG

Example:

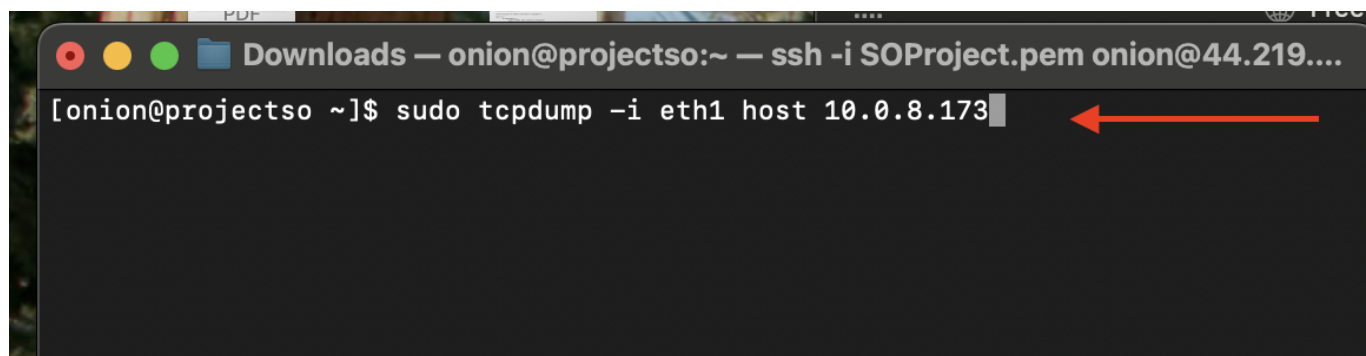


Test the connection by sending traffic to trigger alerts

In the SO cli run the tcpdump command pointing to the "private ip of the Unix/Linux host"

```
sudo tcpdump -i eth1 host <your-linux-host-ip>
```

```
sudo tcpdump -i eth1 host 10.0.8.173
```



Download or curl the traffic generator:

Navigate to tmNIDS on GitHub

<https://github.com/3CORESec/testmynids.org>

Scroll down on the README section and use the "Interactive Mode" cURL command:

testmynids.org

A website and framework for testing NIDS detection

What is it?

A simple project that aims to centralize testing for detection of malicious events by network intrusion detection systems (NIDS). The tests in this project are built against rulesets, not software. Therefore, if you're using ET Open, coverage for these tests will work.

There are two parts to it:

- **A website** - that is used to hold some tests/files. It will also hold sub-domains and DNS records for when testing via DNS is required.
- **A script** - That runs/simulates interaction with the website or with 3rd party websites, meant to be executed on the client for which you want to test coverage of your NIDS sensor.



Usage

Only requirements are `curl` and `nc`, which should be included in your distribution.

One-liner to download and execute in **interactive mode**:

```
curl -sSL https://raw.githubusercontent.com/3CORESec/testmynids.org/master/tmNIDS -o /tmp/tmNIDS
```

One-liner to download and execute in **script mode**:

```
curl -sSL https://raw.githubusercontent.com/3CORESec/testmynids.org/master/tmNIDS -o /tmp/tmNIDS
```

On the unix/linux host, run the tmNIDS traffic check to create alerts:
(Copy the command below and run it)

```
curl -sSL https://raw.githubusercontent.com/3CORESec/testmynids.org/master/tmNIDS -o /tmp/tmNIDS && chmod +x /tmp/tmNIDS && /tmp/tmNIDS
```



```
Downloads — ec2-user@ip-10-0-8-173:~ — ssh -i projects_linux.pem ec2-user@18...
[ec2-user@ip-10-0-8-173 ~]$ curl -sSL https://raw.githubusercontent.com/3CORESec/testmynids.org/master/tmNIDS -o /tmp/tmNIDS && chmod +x /tmp/tmNIDS && /tmp/tmNIDS
```

Hit Enter and type "11"

```
Downloads — ec2-user@ip-10-0-8-173:~ — ssh -i projects_linux.pem ec2-user@18.207.181.59 —
tmNIDS
tmNIDS - NIDS detection tester - @3CORESec
Project: https://github.com/3CORESec/testmynids.org

Choose which test you'd like to run:

1) Linux UID
2) HTTP Basic Authentication
3) HTTP Malware User-Agent
4) Bad Certificates & CAs
5) Tor .onion DNS response and known IPs connection
6) EXE or DLL download over HTTP
7) PDF download with Embedded File
8) Simulate SSH Outbound Scan
9) Miscellaneous domains (TLD's, Sinkhole, DDNS, etc)
10) Anonymous filesharing website
11) External IP Address Lookup
12) URL Shortener
13) Policy Violation - Gaming
14) Adware PUP
15) Malware - Command & Control - Beacon
16) CHAOS! RUN ALL!
17) Quit!
#? 11
```

Hit enter again and check the security onion GUI for traffic and then the cli for alerts:

```
Downloads — onion@projectso:~ — ssh -i SOProject.pem onion@44.219...
vxlan: VXLAN, flags [I] (0x08), vni 10368543
IP 233.211.203.35.bc.googleusercontent.com.54646 > ip-10-0-8-173.ec2.internal.46372: Flags [S], seq 787794939, win 65535, options [mss 1460], length 0
19:44:52.820500 IP ip-10-0-8-173.ec2.internal.65446 > ip-10-0-6-30.ec2.internal.46372: Flags [S], seq 787794939, win 65535, options [mss 1460], length 0
vxlan: VXLAN, flags [I] (0x08), vni 10368543
IP ip-10-0-8-173.ec2.internal.46372 > 233.211.203.35.bc.googleusercontent.com.54646: Flags [R.], seq 0, ack 787794940, win 0, length 0
19:44:56.849334 IP ip-10-0-8-173.ec2.internal.65531 > ip-10-0-6-30.ec2.internal.46372: Flags [S], seq 4141537818, win 65535, options [mss 1460], length 0
19:44:56.849513 IP ip-10-0-8-173.ec2.internal.65531 > ip-10-0-6-30.ec2.internal.46372: Flags [S], seq 4141537819, win 65535, options [mss 1460], length 0
vxlan: VXLAN, flags [I] (0x08), vni 10368543
IP ip-10-0-8-173.ec2.internal.13978 > 147.185.133.182.50360: Flags [R.], seq 0, ack 4141537819, win 0, length 0
19:45:03.176718 IP ip-10-0-8-173.ec2.internal.65446 > ip-10-0-6-30.ec2.internal.46372: Flags [S], seq 3497612851, win 65535, options [mss 1460], length 0
19:45:03.176893 IP ip-10-0-8-173.ec2.internal.65446 > ip-10-0-6-30.ec2.internal.46372: Flags [S], seq 3497612852, win 65535, options [mss 1460], length 0
vxlan: VXLAN, flags [I] (0x08), vni 10368543
IP ip-10-0-8-173.ec2.internal.complex-link > 198.235.24.149.54356: Flags [R.], seq 0, ack 3497612852, win 0, length 0
█
```

GUI

Overview

Alerts

Dashboards

Hunt

Cases

Detections

PCAP

Grid

Downloads

Administration

Tools

Kibana

Elastic Fleet

Osquery Manager

InfluxDB

Alerts

Options

Group By Name, Module

Fetch Limit 500

Filter Results

Count	rule.name	event.module	event.severity_label	rule.uuid
257	ET DROP Dshield Block Listed Source group 1	suricata	medium	2402000
28	ET CINS Active Threat Intelligence Poor Reputation IP group 29	suricata	medium	2403328
27	ET CINS Active Threat Intelligence Poor Reputation IP group 33	suricata	medium	2403332
26	ET CINS Active Threat Intelligence Poor Reputation IP group 36	suricata	medium	2403335
24	ET CINS Active Threat Intelligence Poor Reputation IP group 32	suricata	medium	2403331
23	ET CINS Active Threat Intelligence Poor Reputation IP group 30	suricata	medium	2403329
23	ET CINS Active Threat Intelligence Poor Reputation IP group 35	suricata	medium	2403334
22	ET CINS Active Threat Intelligence Poor Reputation IP group 28	suricata	medium	2403327
21	ET DROP Spamhaus DROP Listed Traffic Inbound group 8	suricata	medium	2400007
19	ET CINS Active Threat Intelligence Poor Reputation IP group 34	suricata	medium	2403333

Ensure the Unix/Linux client IP is visible in the logs

The screenshot shows the Elastic Search interface. At the top, the Elastic logo and a search bar are visible. Below the navigation bar, the 'logs-*' index pattern is selected. A filter for 'client.ip: 10.0.8.173' is applied, indicated by a red arrow. The left sidebar shows 'Popular fields' with 'client.ip' and 'Available fields' with 654 fields. The main area displays a time-series chart for October 11, 2024, and a 'Documents (7,427)' section. A tooltip suggests 'Get the best look at your search results' with a 'Take the tour' button. Below this, a document table is shown with columns for '@timestamp' and 'Document'. The first document entry is for October 12, 2024, at 14:47:16.003, with fields like '@version 1', 'client', '1-106.ec2.internal', 'cloud.machine.type t3a.2xlarge', 'cloud', 'logs_destination_ip 10.0.6.30', 'destination_port 4789', and 'ecs'.

elastic

Find apps, content, and more.

logs-*

Filter your data using KQL syntax

client.ip: 10.0.8.173

Search field names

Popular fields

client.ip

Available fields

@timestamp

@version

agent.ephemeral_id

agent.id

agent.name

agent.type

agent.version

audience

bacnet.bclv.function

bacnet.instance.number

bacnet.invoke.id

Auto interval

No breakdown

Documents (7,427)

Field statistics

Get the best look at your search results

Add relevant fields, reorder and sort columns, resize rows, and more in the document table.

Take the tour

Dismiss

@timestamp

Document

Oct 12, 2024 @ 14:47:16.003

@timestamp Oct 12, 2024 @ 14:47:16.003 @version 1 client 1-106.ec2.internal cloud.machine.type t3a.2xlarge cloud logs_destination_ip 10.0.6.30 destination_port 4789 ecs