

IDS Network Monitoring PCAP Investigation Project

Intrusion Detected



Alexander Chait

Project Introduction: Snort IDS Installation and Malicious PCAP Analysis

Objective: This project aims to install and configure Snort IDS to evaluate its effectiveness in detecting security threats and injecting malicious PCAPs files into Snort for analysis. This controlled environment will provide insights into Snort's capabilities in identifying and responding to various network-based threats.

Scope:

1. Installation and Configuration:

- Deploy Snort IDS in a virtual environment.
- Configure Snort to process and analyze injected PCAP files.
- Optimize rules

2. PCAP Analysis:

- Inject pre-captured malicious PCAP files into Snort.
- Review and analyze the alerts generated by Snort to identify and understand potential threats.

Expected Outcomes:

- **Skill Development:** Practical experience with Snort IDS, including setup, configuration, and rule tuning.
- **Threat Detection:** Improved ability to identify and analyze threats within PCAP files.
- **Security Insight:** Enhanced understanding of network security threats and IDS functionality.

This project will provide valuable hands-on experience with Snort IDS and its role in network security, preparing me for more advanced cybersecurity tasks and incident response scenarios.

Step 1 - Installation and Configurations

I'll start with creating a Cloud based Ubuntu server & adding it the proper firewall rules to avoid outside intrusions

Name	IP Address	Created
 Snort 4 GB / 2 Intel vCPUs / 120 GB Disk / FRA1 ...	206.81.24.193	2 minutes ago

Only my public IP address is allowed as inbound traffic

Type	Protocol	Port Range	Sources	More
All TCP	TCP	All ports	[REDACTED]	More
All UDP	UDP	All ports	[REDACTED]	More
New rule				

Outbound Rules

Set the Firewall rules for outbound traffic. Outbound traffic will only be allowed to the specified ports. All other traffic will be blocked.

Type	Protocol	Port Range	Destinations	More
ICMP	ICMP		All IPv4 All IPv6	More
All TCP	TCP	All ports	All IPv4 All IPv6	More
All UDP	UDP	All ports	All IPv4 All IPv6	More

Now I'll be using PuTTY to login into my IaaS cloud

```
root@Snort: ~
login as: root
root@206.81.24.193's password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-122-generic x86_64)

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

System information as of Tue Aug 20 07:44:54 UTC 2024

System load: 0.0          Users logged in: 0
Usage of /: 1.4% of 116.12GB  IPv4 address for eth0: 206.81.24.193
Memory usage: 6%
Swap usage: 0%           IPv4 address for eth0: 10.19.0.5
Processes: 126            IPv4 address for eth1: 10.114.0.2

205 updates can be applied immediately.
166 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
```

And update the machine

```
root@Snort: ~
individual files in /usr/share/doc/*/copyright.

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

root@Snort:~#
root@Snort:~#
root@Snort:~# sudo apt-get update && sudo apt-get upgrade -y
Hit:1 http://mirrors.digitalocean.com/ubuntu focal InRelease
Hit:2 https://repos-droplet.digitalocean.com/apt/droplet-agent main InRelease
Hit:3 http://mirrors.digitalocean.com/ubuntu focal-updates InRelease
Hit:4 http://mirrors.digitalocean.com/ubuntu focal-backports InRelease
Hit:5 http://security.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages have been kept back:
  fwupd libfwupd2 libfwupdplugin5 linux-headers-generic linux-headers-virtual
    linux-image-virtual linux-virtual python3-update-manager
      ubuntu-adantage-tools update-manager-core
The following packages will be upgraded:
  accountsservice apparmor apport base-files bind9-dnsutils bind9-host
```

Now to snort installation, I'll start by making a directory called snort

```
root@Snort:~# mkdir snort
root@Snort:~# ls
snap  snort
root@Snort:~#
```

Now I'll install some prerequisites for Snort 3

```
root@Snort:~# sudo apt-get install -y build-essential autotools-dev libdumbnet-dev libluajit-5.1-dev libpcap-dev \ zlib-dev pkg-config libhwloc-dev cmake liblzma-dev openssl libssl-dev cpputest libsqlite3-dev \ libtool uuid-dev git autoconf bison flex libcmocka-dev libnetfilter-queue-dev libunwind-dev \ libmnl-dev et
```

Now I'll follow these next 12 steps to install all needed tools and dependencies for proper functionality

STEP 1 - Install pcre

```
cd ~/snort
wget
https://sourceforge.net/projects/pcre/files/pcre/8.45/pcre-8.45.tar.gz
tar -xzvf pcre-8.45.tar.gz
cd pcre-8.45
./configure
make
sudo make install
```

STEP 2 - Install gperftools

```
cd ~/snort  
  
wget  
https://github.com/gperftools/gperftools/releases/download/gperf  
tools-2.9.1/gperftools-2.9.1.tar.gz  
  
tar xzvf gperftools-2.9.1.tar.gz  
  
cd gperftools-2.9.1  
  
../configure  
  
make  
  
sudo make install
```

STEP 3 - Install Ragel

```
cd ~/snort  
  
wget http://www.colm.net/files/ragel/ragel-6.10.tar.gz  
  
tar -xzvf ragel-6.10.tar.gz  
  
cd ragel-6.10  
  
../configure  
  
make  
  
sudo make install
```

STEP 4 - download (but don't install) the Boost C++ Libraries:

```
cd ~/snort  
  
wget  
https://boostorg.jfrog.io/artifactory/main/release/1.77.0/source  
/boost_1_77_0.tar.gz  
  
tar -xvzf boost_1_77_0.tar.gz
```

STEP 5 - Install Hyperscan

```
cd ~/snort  
  
wget  
https://github.com/intel/hyperscan/archive/refs/tags/v5.4.2.tar.  
gz  
  
tar -xvzf v5.4.2.tar.gz  
  
mkdir ~/snort/hyperscan-5.4.2-build  
  
cd hyperscan-5.4.2-build/  
  
cmake -DCMAKE_INSTALL_PREFIX=/usr/local  
-DBOOST_ROOT=~/snort/boost_1_77_0/ ..../hyperscan-5.4.2  
  
make  
  
sudo make install
```

STEP 6 - Install flatbuffers

```
cd ~/snort  
  
wget  
https://github.com/google/flatbuffers/archive/refs/tags/v2.0.0.t  
ar.gz -O flatbuffers-v2.0.0.tar.gz  
  
tar -xzvf flatbuffers-v2.0.0.tar.gz  
  
mkdir flatbuffers-build  
  
cd flatbuffers-build  
  
cmake .. /flatbuffers-2.0.0  
  
make  
  
sudo make install
```

STEP 7 - Install Data Acquistion (DAQ) from Snort

```
cd ~/snort  
  
wget  
https://github.com/snort3/libdaq/archive/refs/tags/v3.0.13.tar.g  
z -O libdaq-3.0.13.tar.gz  
  
tar -xzvf libdaq-3.0.13.tar.gz  
  
cd libdaq-3.0.13  
  
. /bootstrap  
  
. /configure  
  
make  
  
sudo make install
```

STEP 8 - Update shared libraries

```
sudo ldconfig
```

STEP 9 - Download latest version of Snort 3

```
cd ~/snort
```

```
wget  
https://github.com/snort3/snort3/archive/refs/tags/3.1.74.0.tar.gz -O snort3-3.1.74.0.tar.gz
```

```
tar -xzvf snort3-3.1.74.0.tar.gz
```

```
cd snort3-3.1.74.0
```

```
./configure_cmake.sh --prefix=/usr/local --enable-jemalloc
```

```
cd build
```

```
make
```

```
sudo make install
```

```
Disable LRO & GRO using a service
```

STEP 10 -

[Unit]

Description=Ethtool Configuration for Network Interface

[Service]

Requires=network.target

Type=oneshot

ExecStart=/sbin/ethtool -K <network adapter> gro off

ExecStart=/sbin/ethtool -K <network adapter> lro off

[Install]

WantedBy=multi-user.target

STEP 11 -

```
git clone https://github.com/shirkdog/pulledpork3.git
```

```
cd ~/snort/pulledpork3
```

```
sudo mkdir /usr/local/bin/pulledpork3
```

```
sudo cp pulledpork.py /usr/local/bin/pulledpork3
```

```
sudo cp -r lib/ /usr/local/bin/pulledpork3
```

```
sudo chmod +x /usr/local/bin/pulledpork3/pulledpork.py
```

```
sudo mkdir /usr/local/etc/pulledpork3
```

```
sudo cp etc/pulledpork.conf /usr/local/etc/pulledpork3/
```

Great, SNORT is now installed

```
root@Snort:~/snort/snort3-3.1.74.0/build# /usr/local/bin/snort -V
      _-'--> Snort++ <*-_
o" )~ Version 3.1.74.0
'--- By Martin Roesch & The Snort Team
      http://snort.org/contact#team
      Copyright (C) 2014-2023 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using DAQ version 3.0.13
Using LuaJIT version 2.1.0-beta3
Using OpenSSL 1.1.1f 31 Mar 2020
Using libpcap version 1.9.1 (with TPACKET_V3)
Using PCRE version 8.45 2021-06-15
Using ZLIB version 1.2.11
Using Hyperscan version 5.4.2 2024-08-20
Using LZMA version 5.2.4

root@Snort:~/snort/snort3-3.1.74.0/build#
```

Now to validate the configs I'll use this command

```
root@Snort:~/snort/snort3-3.1.74.0/build# snort -c /usr/local/etc/snort/snort.lua
-----
o")~  Snort++ 3.1.74.0
```

Great, SNORT is now installed

```
pcap DAQ configured to passive.

Snort successfully validated the configuration (with 0 warnings).
o")~  Snort exiting
root@Snort:~/snort/snort3-3.1.74.0/build#
```

Next I'll set GRO & LRO to off on the network adapter

```
root@Snort:~/snort/snort3-3.1.74.0/build# sudo nano /lib/systemd/system/ethtool.service
```

The screenshot shows a terminal window titled "root@Snort: ~/snort/snort3-3.1.74.0/build". Inside the terminal, the command "sudo nano /lib/systemd/system/ethtool.service" is run. The file content is as follows:

```
GNU nano 4.8          /lib/systemd/system/ethtool.service      Modified
[Unit]
Description=Ethtool Configuration for Network Interface
[Service]
Requires=network.target
Type=oneshot
ExecStart=/sbin/ethtool -K eth0 gro off
ExecStart=/sbin/ethtool -K eth0 lro off
[Install]
WantedBy=multi-user.target
```

At the bottom of the terminal window, there is a menu bar with various keyboard shortcuts for navigating the nano editor.

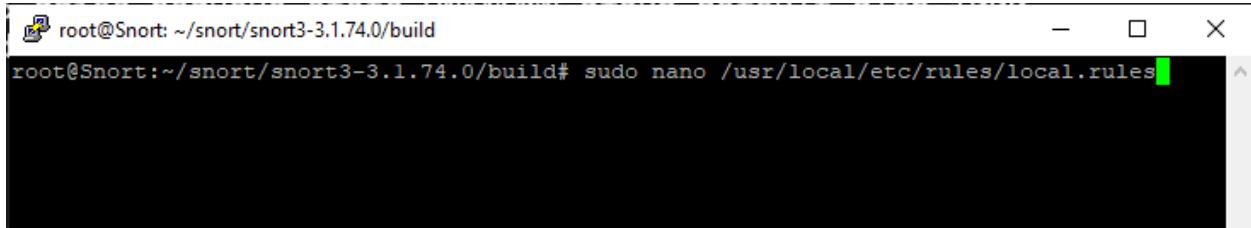
Now I'll enable and start the ethtool service

```
root@Snort:~/snort/snort3-3.1.74.0/build# sudo nano /lib/systemd/system/ethtool.service
root@Snort:~/snort/snort3-3.1.74.0/build# sudo systemctl enable ethtool
Created symlink /etc/systemd/system/multi-user.target.wants/ethtool.service → /lib/systemd/system/ethtool.service.
root@Snort:~/snort/snort3-3.1.74.0/build# sudo service ethtool start
root@Snort:~/snort/snort3-3.1.74.0/build#
```

Now I'll create a custom snort rule, I'll create a new directory first name rules

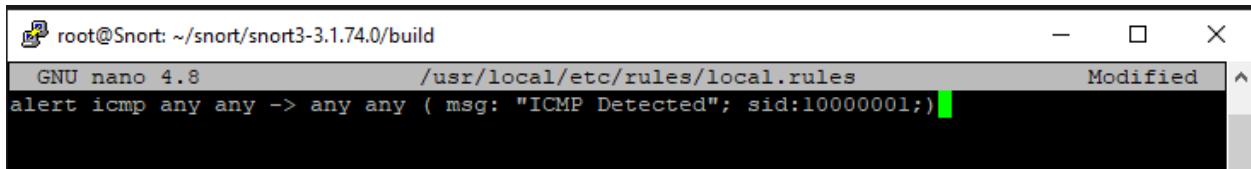
```
root@Snort:~/snort/snort3-3.1.74.0/build# sudo mkdir /usr/local/etc/rules
root@Snort:~/snort/snort3-3.1.74.0/build#
root@Snort:~/snort/snort3-3.1.74.0/build#
root@Snort:~/snort/snort3-3.1.74.0/build#
```

Next I'll create a rule files for snort



```
root@Snort:~/snort/snort3-3.1.74.0/build
root@Snort:~/snort/snort3-3.1.74.0/build# sudo nano /usr/local/etc/rules/local.rules
```

And I'll configure a new ICMP Detected rule



```
root@Snort:~/snort/snort3-3.1.74.0/build
GNU nano 4.8          /usr/local/etc/rules/local.rules      Modified
alert icmp any any -> any any ( msg: "ICMP Detected"; sid:10000001;)
```

The format is very flexible, any any is src address and src port + des address des port

Now to test the rule isn't generating any errors,
I'll type in

```
root@Snort:~/snort/snort3-3.1.74.0/build
root@Snort:~/snort/snort3-3.1.74.0/build# sudo nano /usr/local/etc/rules/local.rules
root@Snort:~/snort/snort3-3.1.74.0/build#
root@Snort:~/snort/snort3-3.1.74.0/build# snort -c /usr/local/etc/snort/snort.lua -R
ERROR: can't set -R
ERROR: usage: -R <rules> include this rules file in the default policy
FATAL: see prior 2 errors
Fatal Error, Quitting..
root@Snort:~/snort/snort3-3.1.74.0/build# snort -c /usr/local/etc/snort/snort.lua -R /usr
/local/etc/rules/local.rules
-----
o")~ Snort++ 3.1.74.0
-----
Loading /usr/local/etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:
    ssh
    http
    https
```

```
pcap DAQ configured to passive.

Snort successfully validated the configuration (with 0 warnings).
o")~ Snort exiting
root@Snort:~/snort/snort3-3.1.74.0/build#
```

And its working, Now I'll start Snort and listen on traffic, using my other machine to test

```
Snort successfully validated the configuration (with 0 warnings).
o")~ Snort exiting
root@Snort:~/snort/snort3-3.1.74.0/build# sudo snort -c /usr/local/etc/snort/snort.lua -R
/usr/local/etc/rules/local.rules -i eth0 -A alert_fast
```

(Make sure the right network adapter is config here)

I'll also set the alert to Fast one format

I'll ping the Snort machine with my other machine

```
C:\Users\alexc>ping 206.81.24.193

Pinging 206.81.24.193 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 206.81.24.193:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Not working, I forgot to activate ICMP in the firewall rules, I'll add it

Inbound Rules

Set the Firewall rules for incoming traffic. Only the specified ports will accept inbound connections. All other traffic will be blocked.

Type	Protocol	Port Range	Sources	
ICMP	ICMP		[REDACTED]	More ▾
All TCP	TCP	All ports	147.235.207.170	More ▾
All UDP	UDP	All ports	147.235.207.170	More ▾
New rule ▾				

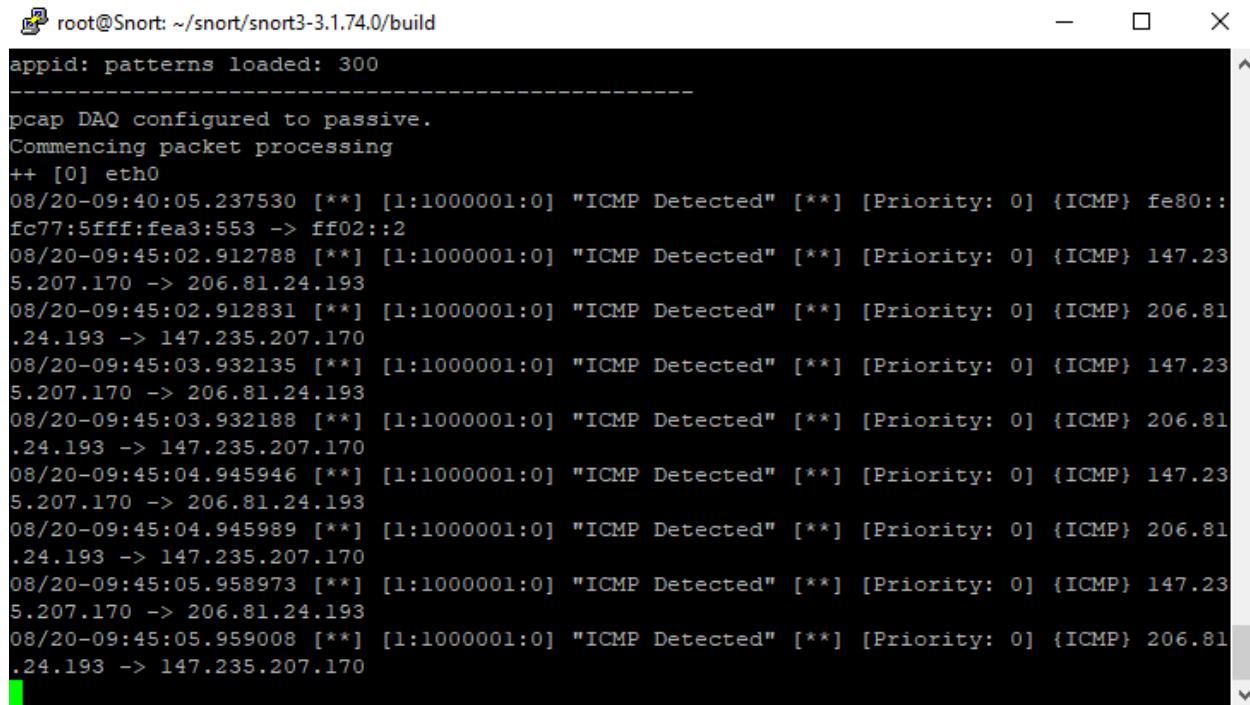
Now it's working

```
C:\Users\alexc>ping 206.81.24.193

Pinging 206.81.24.193 with 32 bytes of data:
Reply from 206.81.24.193: bytes=32 time=66ms TTL=50
Reply from 206.81.24.193: bytes=32 time=64ms TTL=50
Reply from 206.81.24.193: bytes=32 time=64ms TTL=50
Reply from 206.81.24.193: bytes=32 time=63ms TTL=50

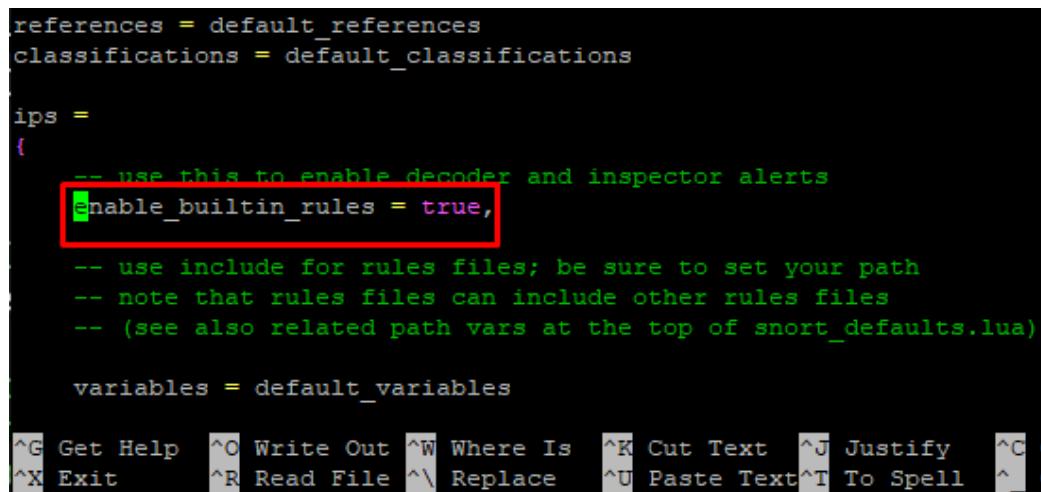
Ping statistics for 206.81.24.193:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 63ms, Maximum = 66ms, Average = 64ms
```

And Snort is getting the Alerts, GREAT!



```
root@Snort: ~/snort/snort3-3.1.74.0/build
appid: patterns loaded: 300
-----
pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth0
08/20-09:40:05.237530 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} fe80::fc77:5fff:fea3:553 -> ff02::2
08/20-09:45:02.912788 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} 147.23.5.207.170 -> 206.81.24.193
08/20-09:45:02.912831 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} 206.81.24.193 -> 147.235.207.170
08/20-09:45:03.932135 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} 147.23.5.207.170 -> 206.81.24.193
08/20-09:45:03.932188 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} 206.81.24.193 -> 147.235.207.170
08/20-09:45:04.945946 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} 147.23.5.207.170 -> 206.81.24.193
08/20-09:45:04.945989 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} 206.81.24.193 -> 147.235.207.170
08/20-09:45:05.958973 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} 147.23.5.207.170 -> 206.81.24.193
08/20-09:45:05.959008 [**] [1:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP} 206.81.24.193 -> 147.235.207.170
```

Next I'll config Snort, so I don't have to specify the rule location each time



```
references = default_references
classifications = default_classifications

ips =
{
    -- use this to enable decoder and inspector alerts
    enable_builtin_rules = true,
    -- use include for rules files; be sure to set your path
    -- note that rules files can include other rules files
    -- (see also related path vars at the top of snort_defaults.lua)

    variables = default_variables

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell ^_ Go To Line
```

We'll set snort to Alert fast but without specifying a specific rule path this time

```
root@Snort:~# sudo snort -c /usr/local/etc/snort/snort.lua -i ens34 -A alert_fast
```

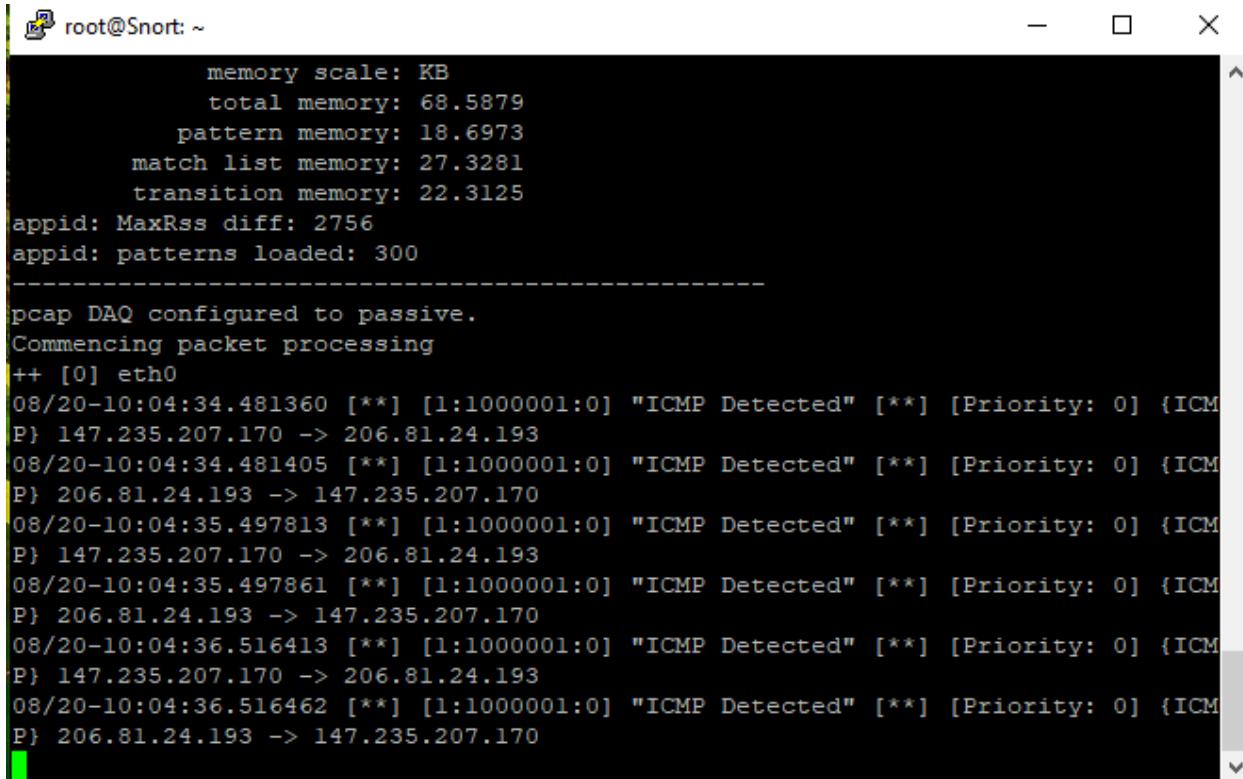
Error, after checking there was a typo in my configs of the .lua file

```
pcap DAO configured to passive.  
FATAL: see prior 1 errors (0 warnings)  
Fatal Error, Quitting..  
root@Snort:~# sudo nano /usr/local/etc/snort/snort.lua  
GNU nano 4.8          /usr/local/etc/snort/snort.lua      Modified
```

And it's working, now I'll ping again to check

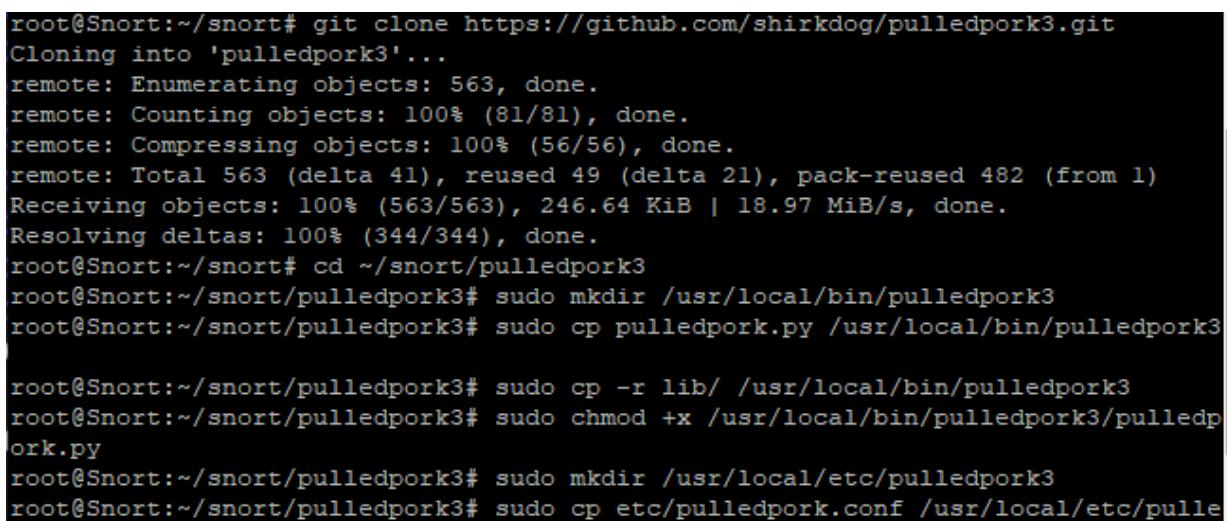
```
-----  
search engine (ac_bnfa)  
    instances: 2  
    patterns: 416  
    pattern chars: 2508  
        num states: 1778  
    num match states: 370  
        memory scale: KB  
        total memory: 68.5879  
        pattern memory: 18.6973  
    match list memory: 27.3281  
        transition memory: 22.3125  
appid: MaxRss diff: 2756  
appid: patterns loaded: 300  
-----  
pcap DAQ configured to passive.  
Commencing packet processing  
++ [0] eth0
```

And the rule is still working!



```
root@Snort: ~
memory scale: KB
    total memory: 68.5879
    pattern memory: 18.6973
    match list memory: 27.3281
    transition memory: 22.3125
appid: MaxRss diff: 2756
appid: patterns loaded: 300
-----
pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth0
08/20-10:04:34.481360 [**] [l:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP}
P} 147.235.207.170 -> 206.81.24.193
08/20-10:04:34.481405 [**] [l:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP}
P} 206.81.24.193 -> 147.235.207.170
08/20-10:04:35.497813 [**] [l:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP}
P} 147.235.207.170 -> 206.81.24.193
08/20-10:04:35.497861 [**] [l:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP}
P} 206.81.24.193 -> 147.235.207.170
08/20-10:04:36.516413 [**] [l:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP}
P} 147.235.207.170 -> 206.81.24.193
08/20-10:04:36.516462 [**] [l:1000001:0] "ICMP Detected" [**] [Priority: 0] {ICMP}
P} 206.81.24.193 -> 147.235.207.170
```

Next I want to expand my rule sets, I will download pulled pork (LOL)



```
root@Snort:~/snort# git clone https://github.com/shirkdog/pulledpork3.git
Cloning into 'pulledpork3'...
remote: Enumerating objects: 563, done.
remote: Counting objects: 100% (81/81), done.
remote: Compressing objects: 100% (56/56), done.
remote: Total 563 (delta 41), reused 49 (delta 21), pack-reused 482 (from 1)
Receiving objects: 100% (563/563), 246.64 KiB | 18.97 MiB/s, done.
Resolving deltas: 100% (344/344), done.
root@Snort:~/snort# cd ~/snort/pulledpork3
root@Snort:~/snort/pulledpork3# sudo mkdir /usr/local/bin/pulledpork3
root@Snort:~/snort/pulledpork3# sudo cp pulledpork.py /usr/local/bin/pulledpork3

root@Snort:~/snort/pulledpork3# sudo cp -r lib/ /usr/local/bin/pulledpork3
root@Snort:~/snort/pulledpork3# sudo chmod +x /usr/local/bin/pulledpork3/pulledpork.py
root@Snort:~/snort/pulledpork3# sudo mkdir /usr/local/etc/pulledpork3
root@Snort:~/snort/pulledpork3# sudo cp etc/pulledpork.conf /usr/local/etc/pulle
```

And its working!

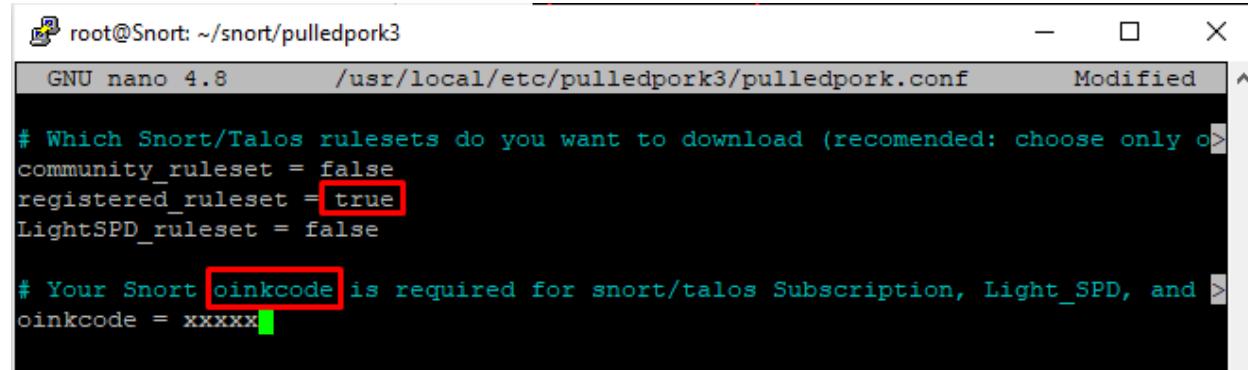
```
root@Snort:~/snort/pulledpork3# /usr/local/bin/pulledpork3/pulledpork.py -V
PulledPork v3.0.0.5

https://github.com/shirkdog/pulledpork3

      _----,\_____)   PulledPork v3.0.0.5
      \---=\\\_ /   Lowcountry yellow mustard bbq sauce is the best bbq sauce. F
ight me.
      \---=\\\\
      .-----Y|\\ \ Copyright (C) 2021 Noah Dietrich, Colin Grady, Michael Shirk
@/_          / 66\_ and the PulledPork Team!
      |        \ \_(")
      \_ \ /-| ||'--' Rules give me wings!
      \_ \ \_\\

root@Snort:~/snort/pulledpork3#
```

Now I'll config it



A screenshot of a terminal window titled "root@Snort: ~/snort/pulledpork3". The window shows the file "/usr/local/etc/pulledpork3/pulledpork.conf" being edited with the nano 4.8 text editor. The file contains configuration options for Snort/Talos rulesets. The "registered_ruleset" option is set to "true", and the "oinkcode" option is set to "xxxxxx". A red box highlights the "oinkcode" line.

```
root@Snort:~/snort/pulledpork3
GNU nano 4.8      /usr/local/etc/pulledpork3/pulledpork.conf      Modified ^

# Which Snort/Talos rulesets do you want to download (recomended: choose only one)
community_ruleset = false
registered_ruleset = true
LightSPD_ruleset = false

# Your Snort oinkcode is required for snort/talos Subscription, Light_SPD, and more
oinkcode = xxxxx
```

Using the Oinkcode



Time to run PulledPork3

```
ight me.  
     ``--=\`\\/  
.-----.Y|\\_ Copyright (C) 2021 Noah Dietrich, Colin Grady, Michael Shirk  
@/_      / 66\_\ and the PulledPork Team!  
  | \ \ \ _(")  
  \ /-| ||'--' Rules give me wings!  
  \\\ \\\\  
-----  
Loading configuration file: /usr/local/etc/pulledpork3/pulledpork.conf  
ERROR: `sorule_path` is configured but is not a directory: /usr/local/etc/so_rules/  
root@Snort:~/snort/pulledpork3#
```

Configuration error, I'll create a dir

```
root@Snort:~/snort/pulledpork3# cd /usr/local/etc  
root@Snort:/usr/local/etc#  
root@Snort:/usr/local/etc# ls  
pulledpork3  rules  snort  
root@Snort:/usr/local/etc# sudo mkdir so_rules  
root@Snort:/usr/local/etc# ls  
pulledpork3  rules  snort  so_rules  
root@Snort:/usr/local/etc# sudo /usr/local/bin/pulledpork3/pulledpork.py -c /usr  
/local/etc/pulledpork3/pulledpork.conf  
  
https://github.com/shirkdog/pulledpork3
```

Another error

```
-----  
Loading configuration file: /usr/local/etc/pulledpork3/pulledpork.conf  
WARNING: Unable to load rules archive: 422 Client Error: Unprocessable Content  
for url: https://snort.org/rules/snortrules-snapshot-31740.tar.gz?oinkcode=<hidden>  
root@Snort:/usr/local/etc#
```

Hardcoded a version in the python script

```
# URLs for supported rulesets (replace <version> and <oinkcode> when downloading)
RULESET_URL_SNORT_COMMUNITY = 'https://snort.org/downloads/community/snort3-community-rules.tar.gz'
RULESET_URL_SNORT_REGISTERED = 'https://snort.org/rules/snortrules-snapshot-31470.tar.gz'  
RULESET_URL_SNORT_LIGHTSPD = 'https://snort.org/rules/Talos_LightSPD.tar.gz'

# TODO: Support for the ET Rulesets has not yet been implemented
```

And its working now, great!

```
https://github.com/shirkdog/pulledpork3

      _\_)     PulledPork v3.0.0.5
     `--=\ \ /     Lowcountry yellow mustard bbq sauce is the best bbq sauce. Fight me.
       `--=\ \ \
      .-----Y|\\\_ Copyright (C) 2021 Noah Dietrich, Colin Grady, Michael Shirk
 @ /      / 66\_ and the PulledPork Team!
 |   \  \  _(")
 \  /-| ||'--' Rules give me wings!
 \_\ \_\ \\\

-----
Loading configuration file: /usr/local/etc/pulledpork3/pulledpork.conf
Processing Registered ruleset
loaded local rules file: Rules(loader:0, enabled:0, disabled:0) from /usr/local/etc/rules/local.rules
Preparing to modify rules by sid file
Completed processing all rulesets and local rules:
- Collected Rules: Rules(loader:50541, enabled:9826, disabled:40715)
- Collected Policies:
  - Policy(name:none, rules:0)
  - Policy(name:balanced, rules:9826)
  - Policy(name:max-detect, rules:40606)
  - Policy(name:security, rules:22534)
  - Policy(name:connectivity, rules:587)
Writing rules to: /usr/local/etc/rules/pulledpork.rules
Program execution complete.
root@snort:/usr/local/etc#
```

Next I'll modify snort conf file to point to the pulledpork3 rulesets

```
        (See also related path vars at the top of snort_.py)
include = "/usr/local/etc/rules/pulledpork.rules",
variables = default_variables
```

I'll run snort again with the specified plugin-path

```
root@Snort:/usr/local/etc
- Collected Rules: Rules(loader:50541, enabled:9826, disabled:40715)
- Collected Policies:
  - Policy(name:none, rules:0)
  - Policy(name:balanced, rules:9826)
  - Policy(name:max-detect, rules:40606)
  - Policy(name:security, rules:22534)
  - Policy(name:connectivity, rules:587)
Writing rules to: /usr/local/etc/rules/pulledpork.rules
Program execution complete.
root@Snort:/usr/local/etc#
root@Snort:/usr/local/etc# sudo nano /usr/local/etc/snort/snort.lua
root@Snort:/usr/local/etc# snort -c /usr/local/etc/snort/snort.lua --plugin-path
/usr/local/etc/snort/
o")~ Snort++ 3.1.74.0
-----
Loading /usr/local/etc/snort/snort.lua:
```

Working!

```
Snort successfully validated the configuration (with 0 warnings).
o")~ Snort exiting
root@Snort:/usr/local/etc#
```

Now it's time to generate so PCAP Signatures! I'll use PCAPs from malware-traffic-analysis

```
root@test:~/test# wget https://www.malware-traffic-analysis.net/2024/06/11/2024-06-11-CVE-2024-4577-probe.pcap.zip
--2024-08-20 10:51:19--  https://www.malware-traffic-analysis.net/2024/06/11/2024-06-11-CVE-2024-4577-probe.pcap.zip
Resolving www.malware-traffic-analysis.net (www.malware-traffic-analysis.net) ...
199.201.110.204
Connecting to www.malware-traffic-analysis.net (www.malware-traffic-analysis.net) |199.201.110.204|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1225 (1.2K) [application/zip]
Saving to: '2024-06-11-CVE-2024-4577-probe.pcap.zip'

2024-06-11-CVE-2024 100%[=====] 1.20K --.-KB/s in 0s

2024-08-20 10:51:19 (253 MB/s) - '2024-06-11-CVE-2024-4577-probe.pcap.zip' saved [1225/1225]
```

I'll feed this to snort and see what it can generate
Filtering to see signatures and alerts generated

```
runtime: 00:00:00
seconds: 0.043616
o")~ Snort exiting
root@test:~/test# snort -c /usr/local/etc/snort/snort.lua --plugin-path /usr/local/etc/snort_rules/ -r 2024-06-11-CVE-2024-4577-probe.pcap -A alert_fast -q
```

We can see that this PCAP has classified an attempted user privilege escalation attempt, cool!

```
o")~ Snort exiting
root@test:~/test# snort -c /usr/local/etc/snort/snort.lua --plugin-path /usr/local/etc/snort_rules/ -r 2024-06-11-CVE-2024-4577-probe.pcap -A alert_fast -q
06/11/22:47:26.511971 [*] [1:63598:1] "SERVER-WEBAAPP PHP PHP-CGI command execution attempt" [**] [Classification: Attempted User Privilege Gain] [Priority: 1]
{TCP} 221.122.67.75:60482 -> 76.223.105.231:80
root@test:~/test#
```

Let's try another one, this time a RAT

2024-03-14 (THURSDAY): ASYNCRAT AND XWORM INFECTION

NOTES:

- Mithrandir (@rerednawyer) ran across this activity and authored the references below.
- The Dropbox link was still active, so I recorded an infection run on a host in my lab.
- Mithrandir provided the copies of AsyncRAT and XWorm in the malware and artifacts zip archive below.
- Zip files are password-protected. Of note, this site has a new password scheme. For the password, see the

REFERENCES:

- https://www.linkedin.com/posts/unit42_asyncrat-xworm-xrat-activity-7174172958414802944-YI3c
- https://twitter.com/Unit42_Intel/status/1768408063621345565

ASSOCIATED FILES:

- 2024-03-14-AsyncRAT-and-XWorm-notes.txt.zip 2.1 kB (2,059 bytes)
- 2024-03-14-AsyncRAT-and-XWorm-infection-traffic.pcap.zip 41.9 MB (41,900,578 bytes)
- 2024-03-14-AsyncRAT-and-XWorm-malware-and-artifacts.zip 23.5 MB (23,464,505 bytes)

```
F
Archive: 2024-03-14-AsyncRAT-and-XWorm-infection-traffic.pcap.zip
[2024-03-14-AsyncRAT-and-XWorm-infection-traffic.pcap.zip] 2024-03-14-AsyncRAT-and-XWorm-infection-traffic.pcap password:
    inflating: 2024-03-14-AsyncRAT-and-XWorm-infection-traffic.pcap
root@Snort:~/test#
```

We can see an ARP_Spoofing alert (on path attack)

```
root@Snort:~/test# snort -c /usr/local/etc/snort.lua --plugin-path /usr/local/etc/snort_rules/ -r 2024-03-14-AsyncRAT-and-XWorm-infection-traffic.pcap -A alert_fast -q
03/14-20:21:41.024280 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->
03/14-20:22:23.035667 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->
03/14-20:23:05.025308 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->
03/14-20:23:19.070104 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->
03/14-20:23:47.025683 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->
03/14-20:24:19.562922 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->
03/14-20:24:29.525992 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->
03/14-20:24:52.567335 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->
```

2024-01-09 (TUESDAY): ASYNC RAT INFECTION

NOTES:

- Zip files are password-protected. Of note, this site has a new password scheme. For the password, see the "about" page.

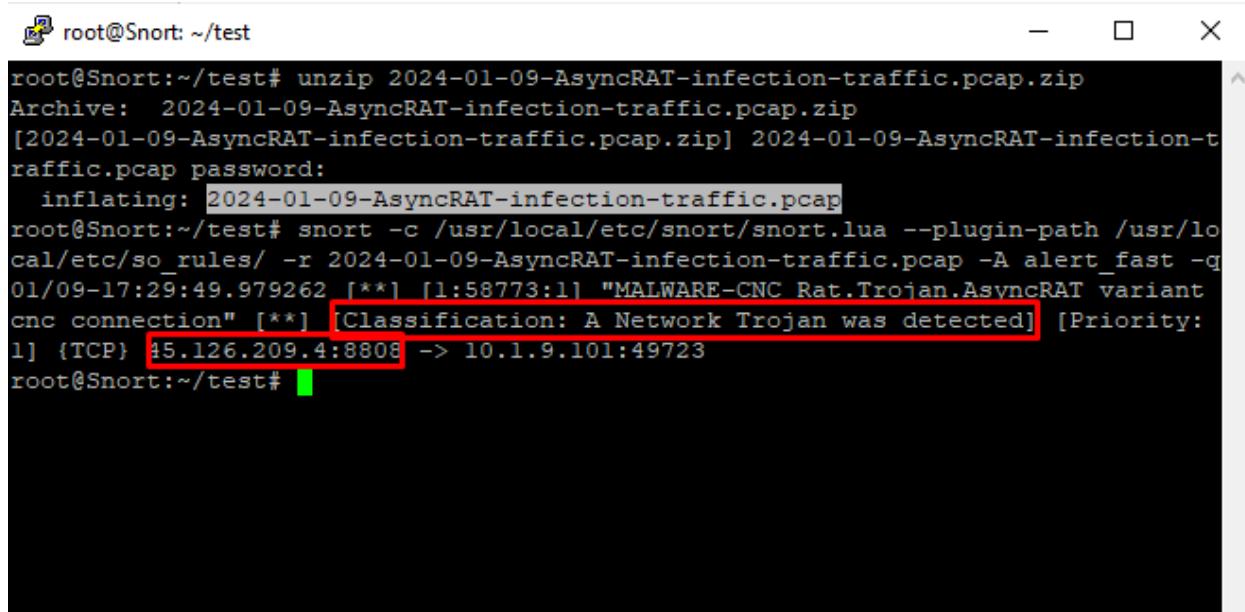
REFERENCE:

- <https://www.virustotal.com/gui/file/be78b500f71db3b870a6ab00f26fd1dcb54bc19a218c93698d6146a87b488ed5>

ASSOCIATED FILES:

- 2024-01-09-IOCS-from-AsyncRAT-infection.txt.zip 1.2 kB (1,185 bytes)
- 2024-01-09-AsyncRAT-infection-traffic.pcap.zip 194.0 kB (194,003 bytes)
- 2024-01-09-malware-and-artifacts-from-AsyncRAT-infection.zip 190.7 kB (190,719 bytes)

A Network Trojan was detected, Snort is a great tool
able to detect and classify attacks



The screenshot shows a terminal window titled 'root@Snort: ~/test'. The user runs 'unzip' on a file named '2024-01-09-AsyncRAT-infection-traffic.pcap.zip'. The archive is successfully extracted. Then, the user runs 'snort' with the command: '-c /usr/local/etc/snort/snort.lua --plugin-path /usr/local/etc/snort_rules/ -r 2024-01-09-AsyncRAT-infection-traffic.pcap -A alert_fast -q'. The output shows a detection message: '[**] [1:58773:1] "MALWARE-CNC Rat.Trojan.AsyncRAT variant cnc connection" [**] [Classification: A Network Trojan was detected] [Priority: 1] (TCP) 45.126.209.4:8808 -> 10.1.9.101:49723'. The line containing the classification message is highlighted with a red box.

```
root@Snort:~/test# unzip 2024-01-09-AsyncRAT-infection-traffic.pcap.zip
Archive: 2024-01-09-AsyncRAT-infection-traffic.pcap.zip
[2024-01-09-AsyncRAT-infection-traffic.pcap.zip] 2024-01-09-AsyncRAT-infection-traffic.pcap password:
  inflating: 2024-01-09-AsyncRAT-infection-traffic.pcap
root@Snort:~/test# snort -c /usr/local/etc/snort/snort.lua --plugin-path /usr/local/etc/snort_rules/ -r 2024-01-09-AsyncRAT-infection-traffic.pcap -A alert_fast -q
01/09/17:29:49.979262 [**] [1:58773:1] "MALWARE-CNC Rat.Trojan.AsyncRAT variant
cnc connection" [**] [Classification: A Network Trojan was detected] [Priority:
1] (TCP) 45.126.209.4:8808 -> 10.1.9.101:49723
root@Snort:~/test#
```

2024-05-09 (THURSDAY): GOOTLOADER ACTIVITY

NOTES:

- Zip files are password-protected. Of note, this site has a new password scheme. For the password, see the "about" page of this website.

REFERENCES:

- https://www.linkedin.com/posts/unit42_gootloader-unit42threatintel-timelythreatintel-activity-7194787295676313600-UyIW
- https://twitter.com/Unit42_Intel/status/1789021679634505978

ASSOCIATED FILES:

- 2024-05-09-IOCs-from-GootLoader-infection.txt.zip 1.5 kB (1,509 bytes)
- 2024-05-09-GootLoader-infection-traffic.pcap.zip 16.0 MB (15,960,806 bytes)
- 2024-05-09-GootLoader-malware-and-artifacts.zip 4.3 MB (4,273,356 bytes)

GootLoader-Infection is using the Arp_spoofing attack as well

```
on-traffic.pcap password:  
  inflating: 2024-05-09-GootLoader-infection-traffic.pcap  
root@Snort:~/test#  
root@Snort:~/test#  
root@Snort:~/test#  
root@Snort:~/test# ^C  
root@Snort:~/test# snort -c /usr/local/etc/snort/snort.lua --plugin-path /usr/local/etc/snort/_rules/ -r 2024-05-09-GootLoader-infection-traffic.pcap -A alert_fast -q  
05/09-21:33:05.401374 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->  
05/09-21:33:54.403569 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Priority: 3] {ARP} ->  
05/09-21:41:59.943193 [**] [116:444:1] "(ipv4) IPv4 option set" [**] [Priority: 3] {IP} 10.5.9.101 -> 224.0.0.22  
05/09-21:41:59.946719 [**] [116:444:1] "(ipv4) IPv4 option set" [**] [Priority: 3] {IP} 10.5.9.101 -> 224.0.0.22  
05/09-21:41:59.948113 [**] [116:444:1] "(ipv4) IPv4 option set" [**] [Priority: 3] {IP} 10.5.9.101 -> 224.0.0.22  
05/09-21:41:59.949010 [**] [116:444:1] "(ipv4) IPv4 option set" [**] [Priority: 3] {IP} 10.5.9.101 -> 224.0.0.22  
05/09-21:42:00.405842 [**] [116:444:1] "(ipv4) IPv4 option set" [**] [Priority: 3] {IP} 10.5.9.101 -> 224.0.0.22
```

2024-04-17 (WEDNESDAY): TA578 PUSHES SSLOAD MALWARE

NOTES:

- Zip files are password-protected. Of note, this site has a new password scheme. For the password, see the "about" page of this website.

ASSOCIATED FILES:

- 2024-04-17-IOCs-from-SSLoad-activity.txt.zip 1.7 kB (1,722 bytes)
- 2024-04-17-TA578-SSLoad-infection.pcap.zip 10.7 MB (10,701,579 bytes)
- 2024-04-17-SSLoad-malware-and-artifacts.zip 2.0 MB (2,047,958 bytes)

[Click here](#) to return to the main page.

The TA576-SSLoad-Infection is using a TCP Portsweep

```
root@test:~/test# ^C
root@test:~/test# unzip 2024-04-17-TA578-SSLoad-infection.pcap.zip
Archive: 2024-04-17-TA578-SSLoad-infection.pcap.zip
[2024-04-17-TA578-SSLoad-infection.pcap.zip] 2024-04-17-TA578-SSLoad-infection.pcap password:
    inflating: 2024-04-17-TA578-SSLoad-infection.pcap
root@test:~/test# ^C
root@test:~/test#
root@test:~/test#
root@test:~/test# snort -c /usr/local/etc/snort.lua --plugin-path /usr/local/etc/snort_rules/ -r 2024-04-17-TA578-SSLoad-infection.pcap -A alert_fast -q
04/17-19:39:40 151216 [**] [122:3:1] "(port_scan) TCP portsweep" [**] [Priority: 3] {TCP} 204.79.197.200:443 -> 10.4.17.101:49744
root@test:~/test#
```

Pretty cool abilities!

Project Summary: Snort IDS Installation and Malicious PCAP Investigation

Objective:

Installed and configured Snort IDS to process and analyze injected malicious PCAP files

Key Activities:

- **Snort Installation and Configuration:** Set up Snort IDS on a virtual machine, focusing on rule configuration and optimizing detection accuracy.
- **Malicious PCAP Injection and Analysis:** Injected pre-captured PCAP files into Snort, analyzed the generated alerts, and identified malicious activities based on predefined detection rules.

Skills Utilized:

- **Intrusion Detection:** Practical experience with Snort IDS, configuring detection rules, and tuning the system for analyzing PCAP data.
- **Security Analysis:** Interpreting Snort alerts, identifying network-based threats, and understanding malicious traffic patterns.
- **System Administration:** Installed and configured Snort IDS on a virtual environment, managing the injection of PCAP files for analysis.

Outcome:

The project provided hands-on experience in installing & configuring Snort IDS for analyzing malicious network traffic, a good skill for cybersecurity roles involving threat detection and incident response.

Project Introduction: Snort IDS Installation and Malicious PCAP Analysis

Objective: This project aims to install and configure Snort IDS to evaluate its effectiveness in detecting security threats and injecting malicious PCAPs files into Snort for analysis. This controlled environment will provide insights into Snort's capabilities in identifying and responding to various network-based threats.

Scope:

3. Installation and Configuration:

- Deploy Snort IDS in a virtual environment.
- Configure Snort to process and analyze injected PCAP files.
- Optimize rules

4. PCAP Analysis:

- Inject pre-captured malicious PCAP files into Snort.
- Review and analyze the alerts generated by Snort to identify and understand potential threats.

Expected Outcomes:

- **Skill Development:** Practical experience with Snort IDS, including setup, configuration, and rule tuning.
- **Threat Detection:** Improved ability to identify and analyze threats within PCAP files.
- **Security Insight:** Enhanced understanding of network security threats and IDS functionality.

This project will provide valuable hands-on experience with Snort IDS and its role in network security, preparing me for more advanced cybersecurity tasks and incident response scenarios.