IDS Network Monitoring PCAP Investigation Project

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.

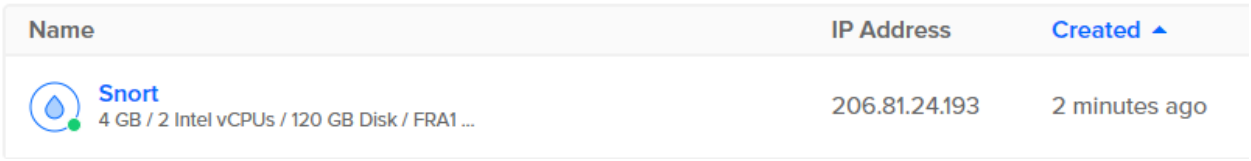
Expected Outcomes:

- **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.

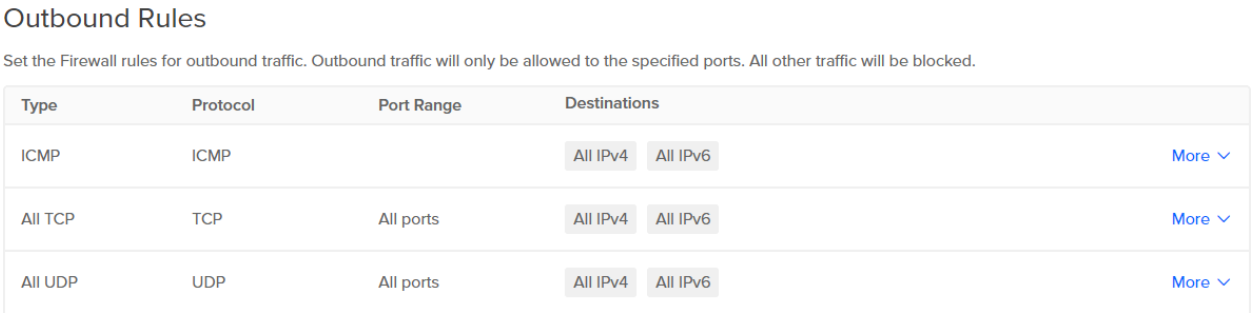
This project will provide valuable hands-on experience with Snort IDS and its role in network security, preparing me for more advanced cybersecurity tasks and incident response scenarios.

Step 1 - Installation and Configurations

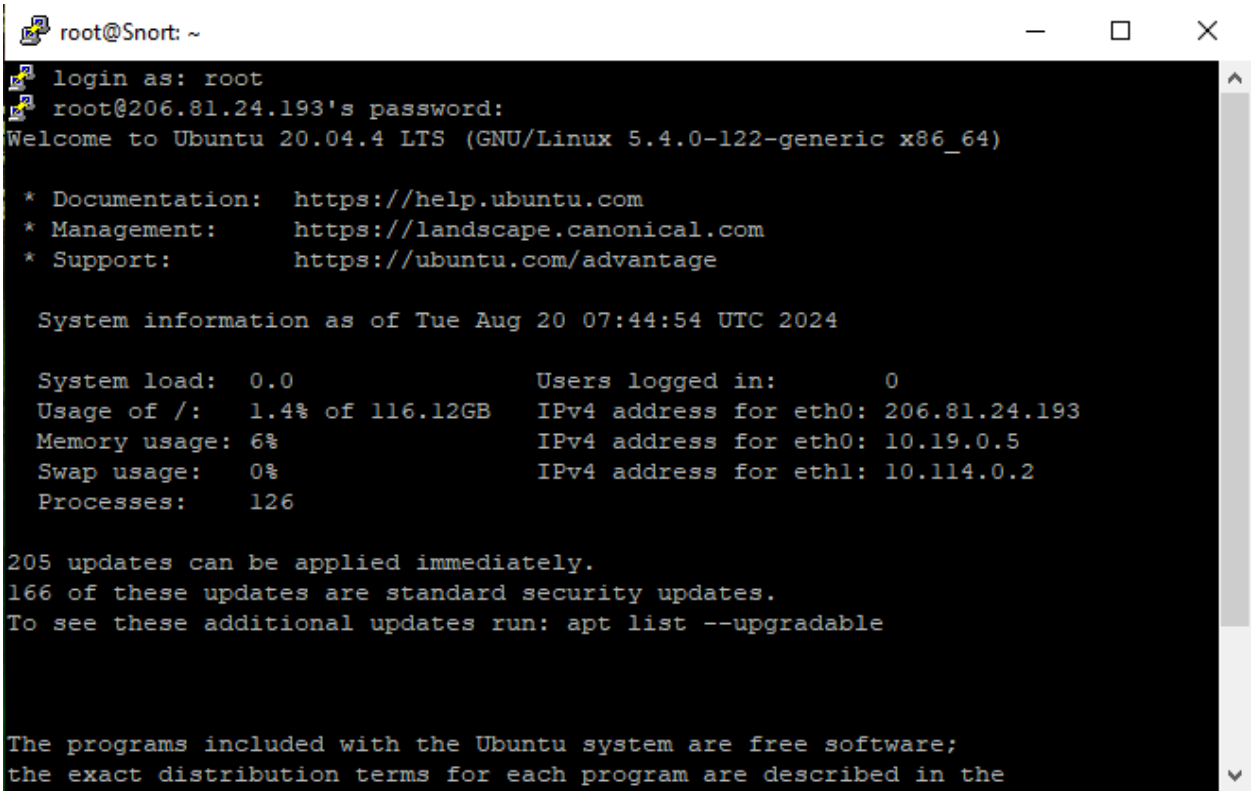
I'll start with creating a Cloud based Ubuntu server & adding it the proper firewall rules to avoid outside intrusions



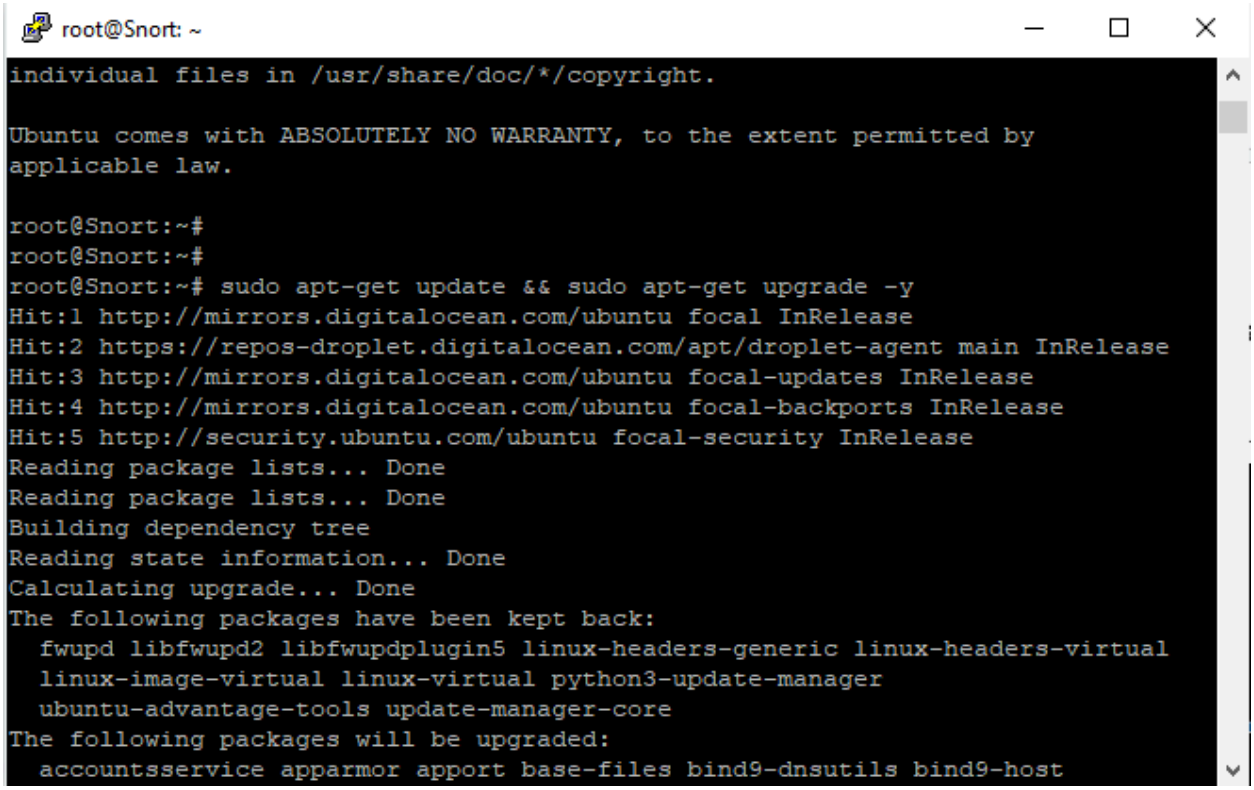
Only my public IP address is allowed as inbound traffic



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



And update the machine



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-d
ev libluajit-5.1-dev libpcap-dev \ zliblg-dev pkg-config libhwloc-dev cmake libl
zma-dev openssl libssl-dev cpputest libsqlite3-dev \ libtool uuid-dev git autoco
nf 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.tar.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.gz -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 Configration 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 reser
ved.
           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

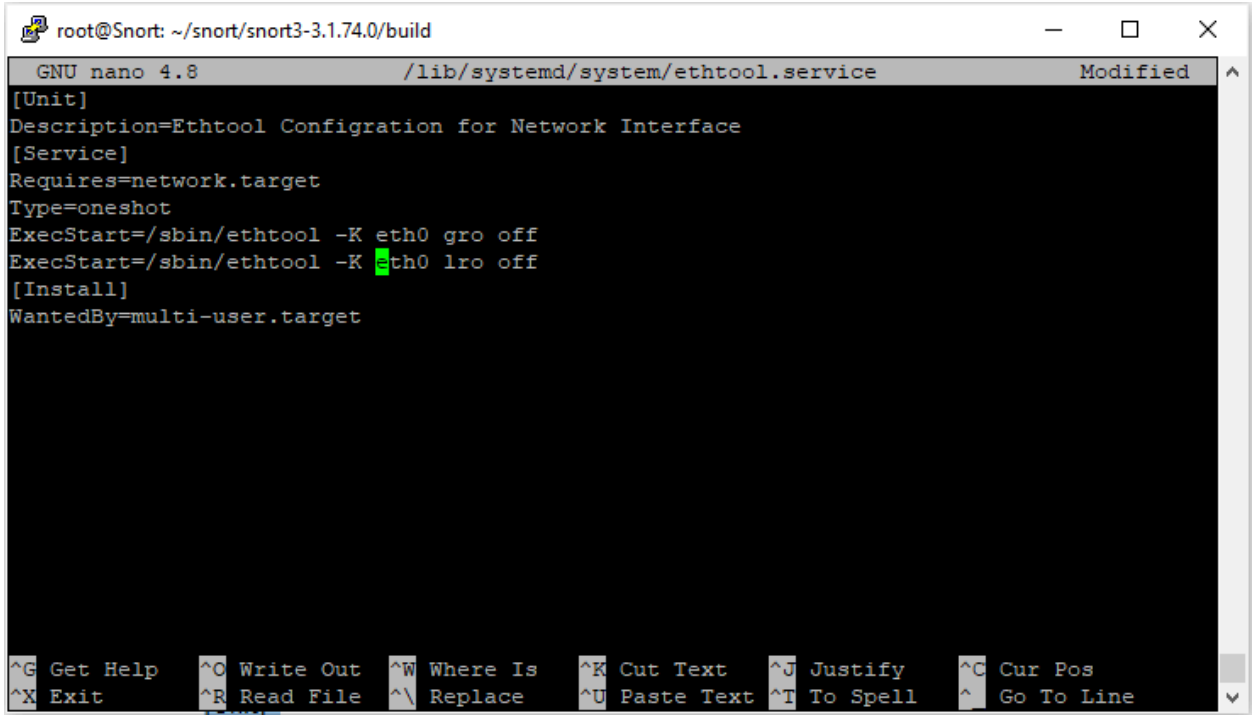
```
root@Snort:~/snort/snort3-3.1.74.0/build# snort -c /usr/local/etc/snort/snort.lu
a
--------------------------------------------------
o")~   Snort++ 3.1.74.0
```

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

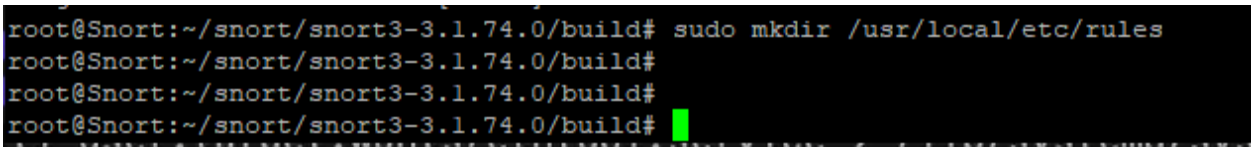
```
root@Snort:~/snort/snort3-3.1.74.0/build# sudo nano /lib/systemd/system/ethtool.service
```



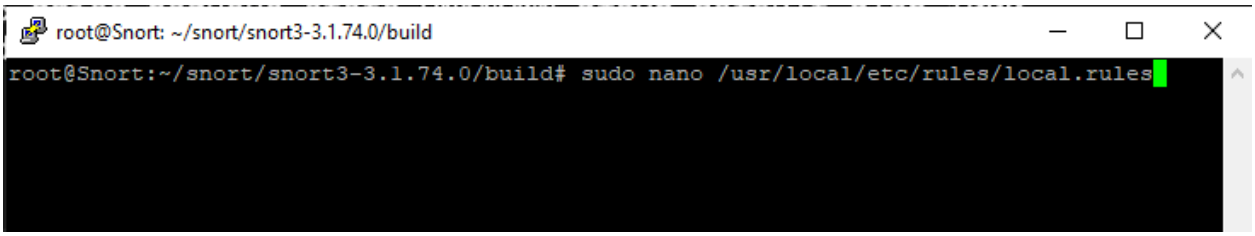
Now I'll enable and start the ethtool service

```
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/system
d/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#
```

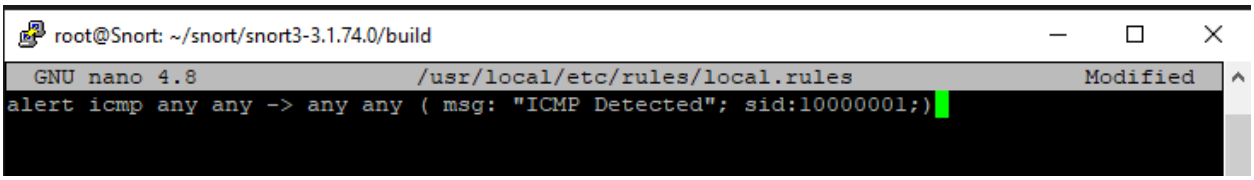
Now I'll create a custom snort rule, I'll create a new directory first name rules



Next I'll create a rule files for snort

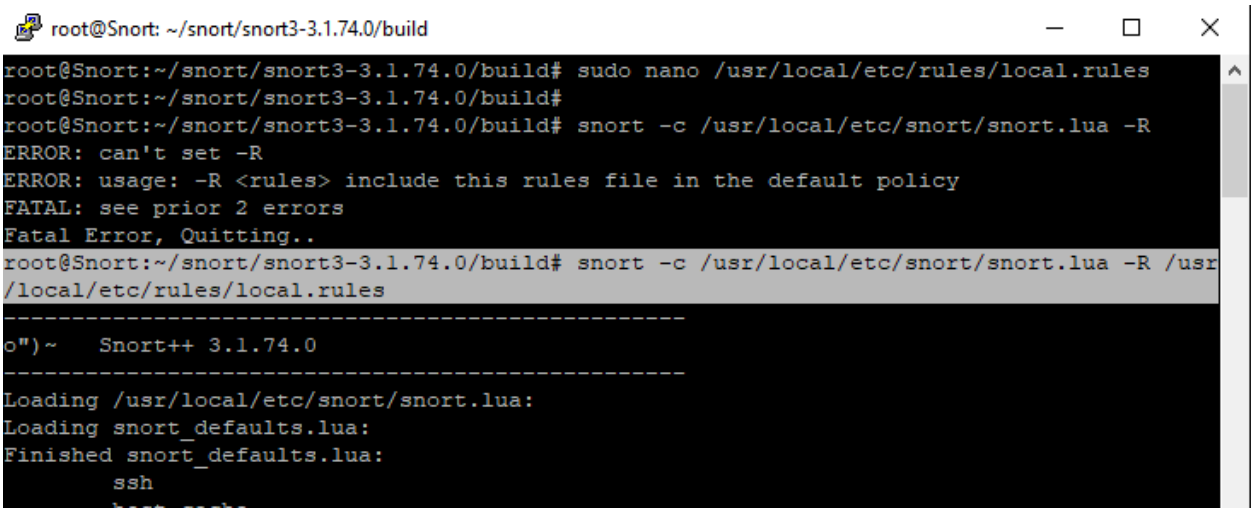


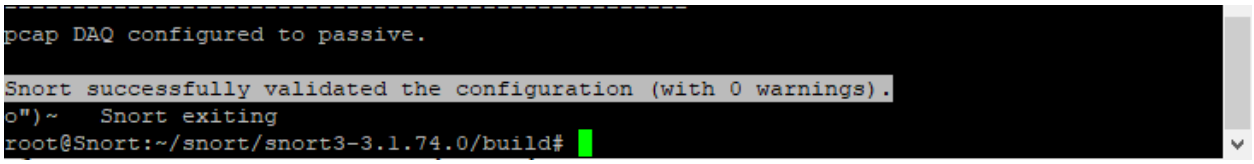
And I'll configure a new ICMP Detected rule



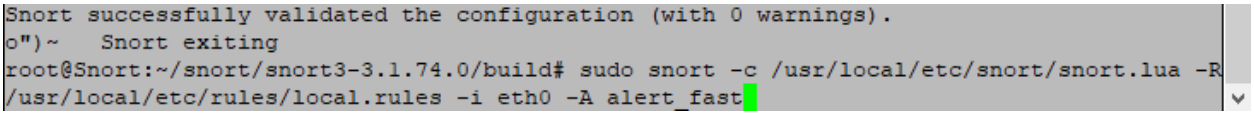
The format is very flexible, any any is src address and src port + des address des port

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





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



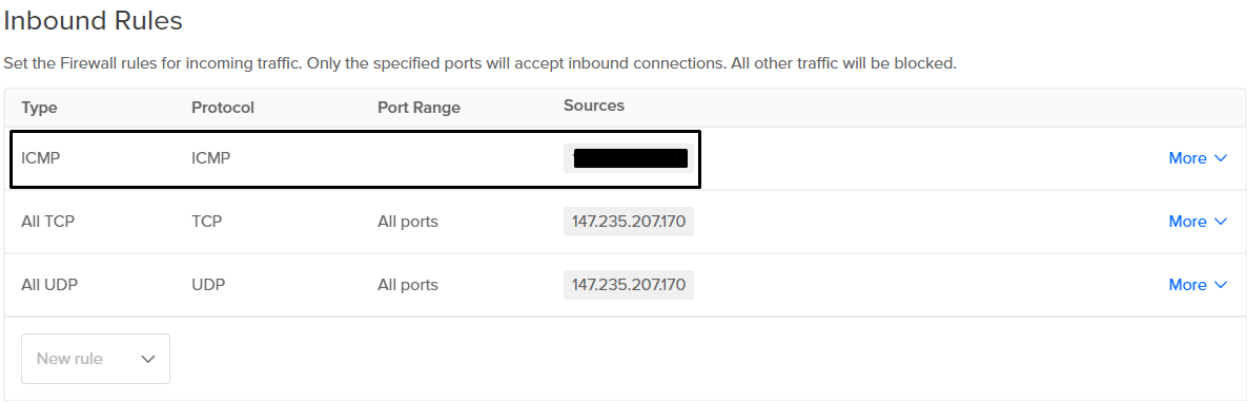
(Make sure the right network adapter is confg here)

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),
```

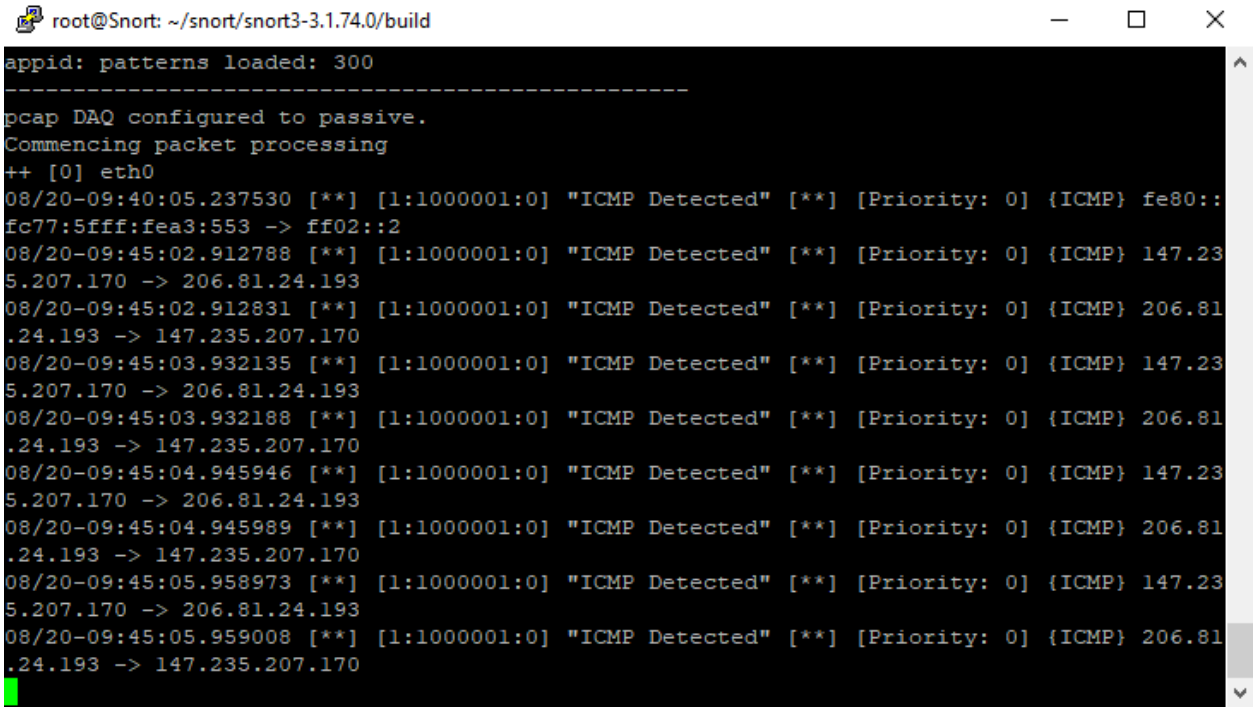
Not working, I forgot to activate ICMP in the firewall rules, I’ll add it



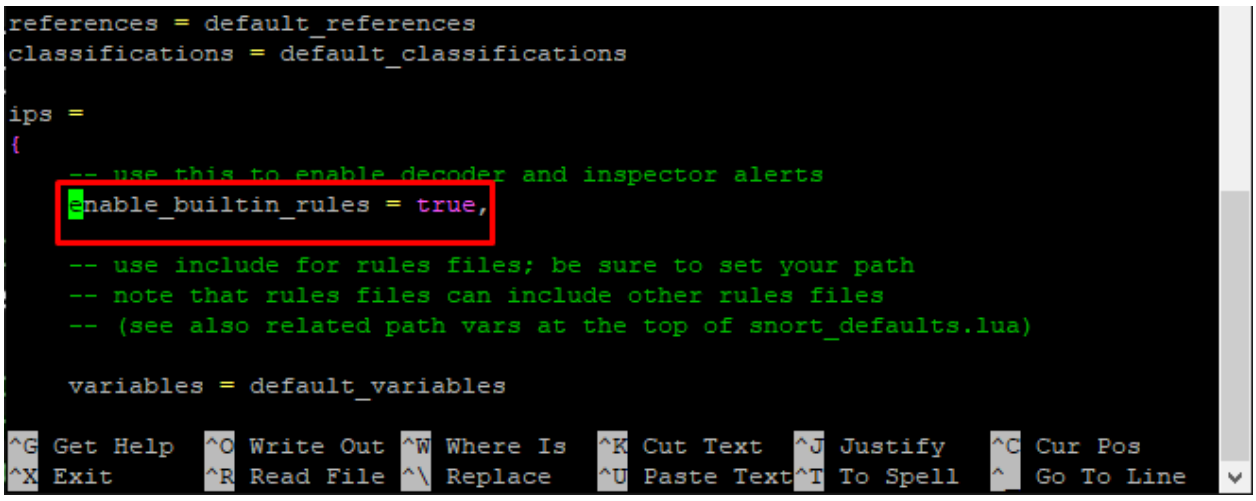
Now it’s working

```
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!



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



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_fas
t
```

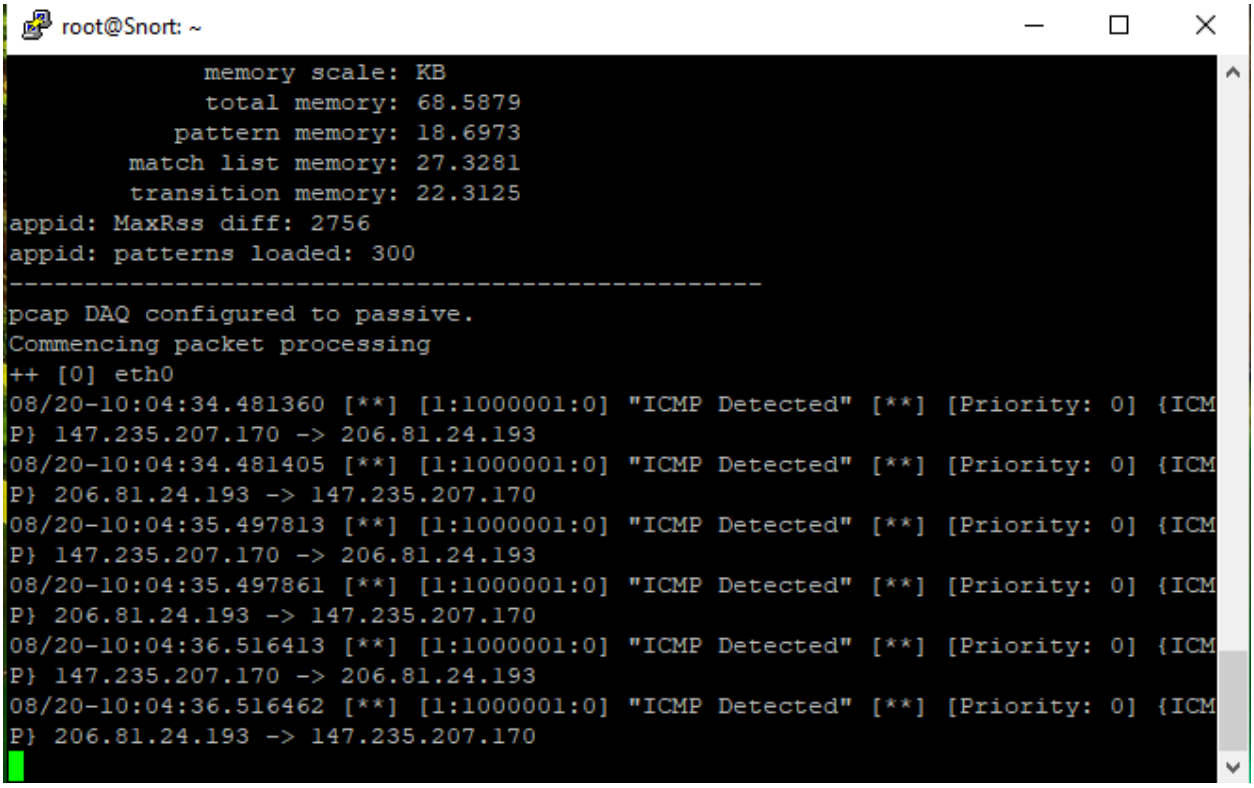
Error, after checking there was a typo in my configs of the .lua file

```
pcap DAQ 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
```

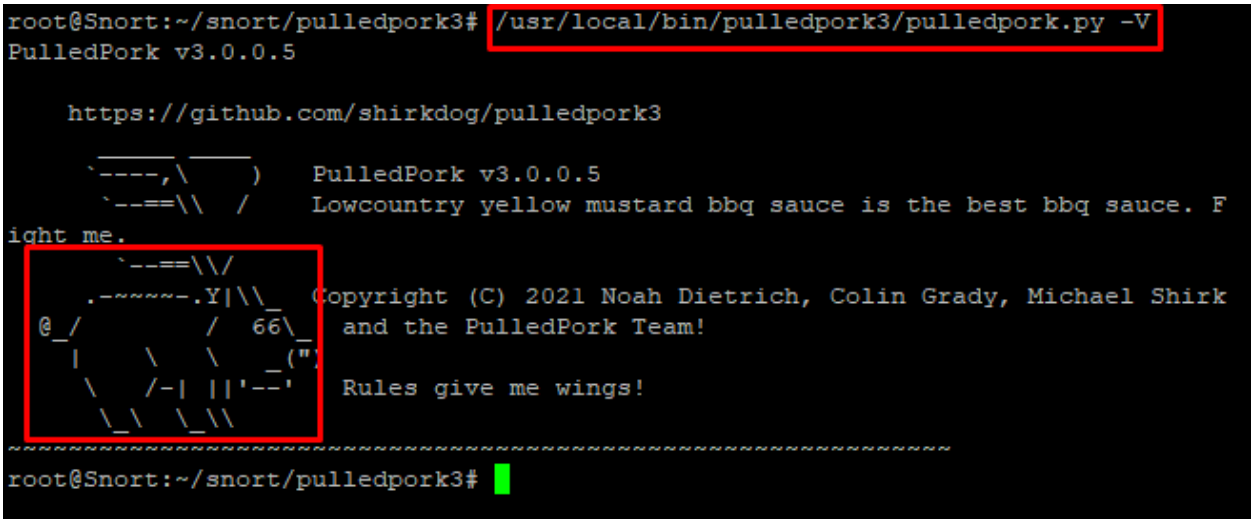
And the rule is still working!



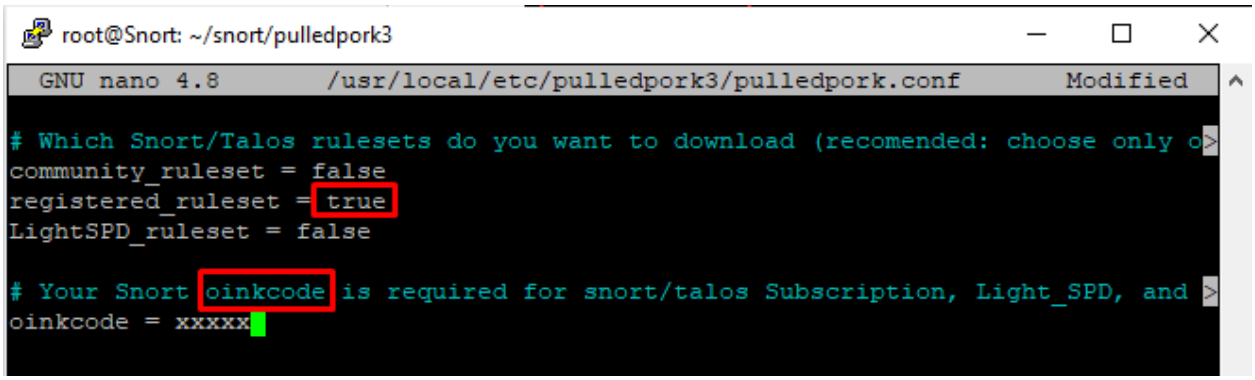
Next I want to expand my rule sets, I will download pulled pork (LOL)

```
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/pulledp
ork.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!

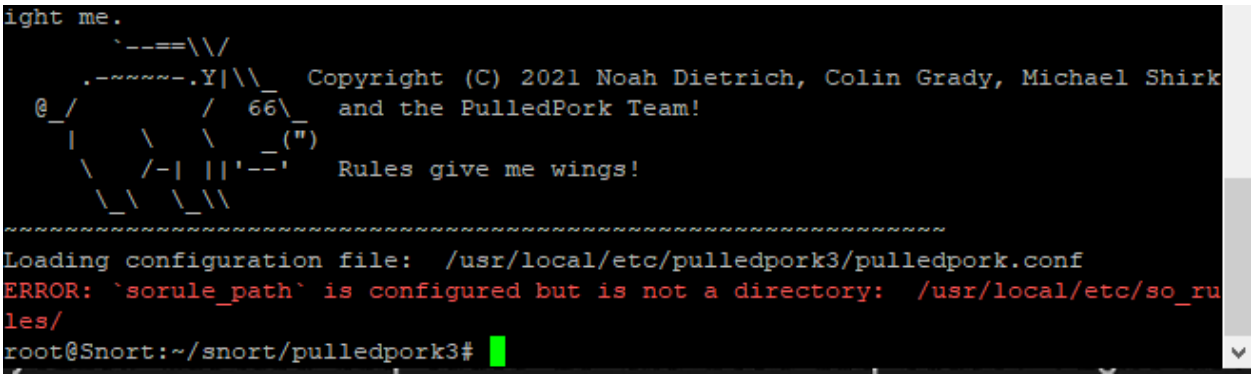


Now I'll config it

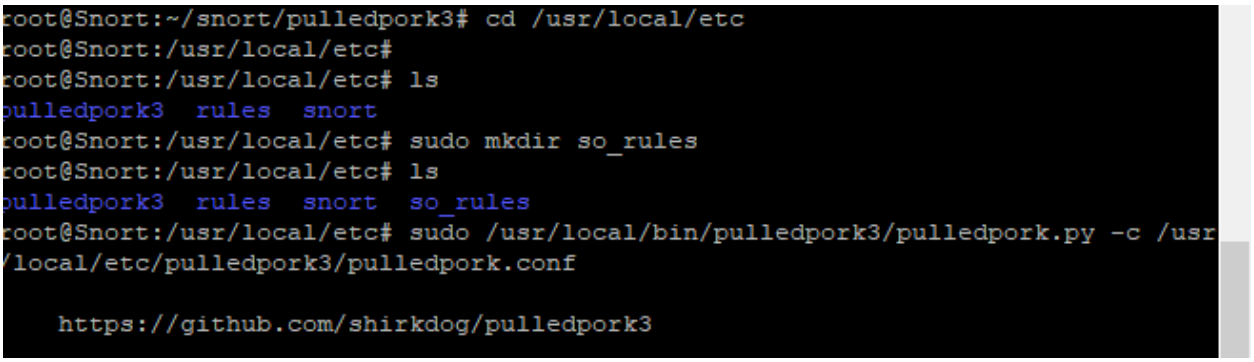


Using the Oinkcode

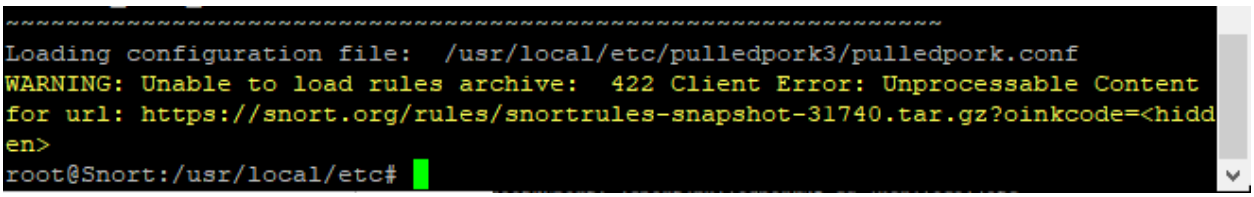
Time to run PulledPork3



Configuration error, I'll create a dir



Another error



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
```

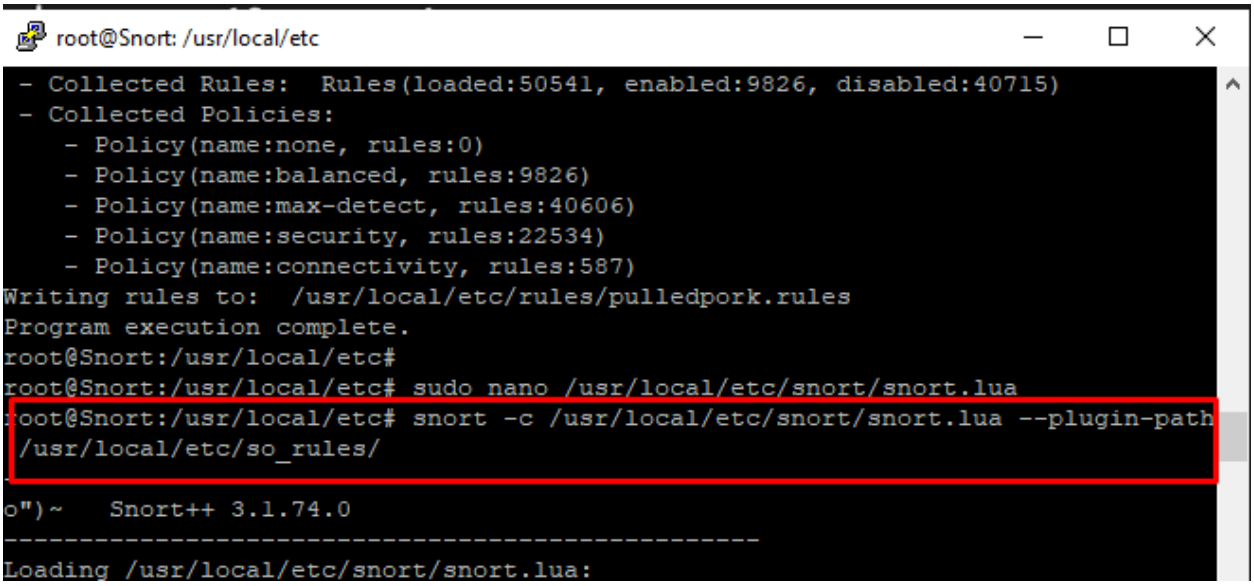
And its working now, great!

```
    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(loaded:0, enabled:0, disabled:0) from /usr/local/etc/rules/local.rul
Preparing to modify rules by sid file
Completed processing all rulesets and local rules:
 - Collected Rules:  Rules(loaded: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#
```

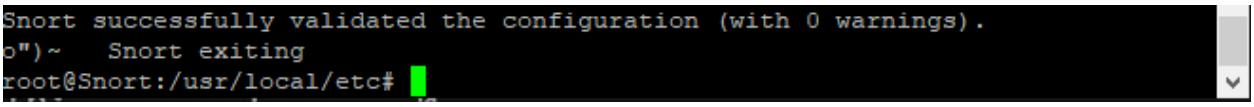
Next I’ll modify snort conf file to point to the pulledpork3 rulesets

```
include = "/usr/local/etc/rules/pulledpork.rules",
variables = default_variables
```

I’ll run snort again with the specified pluging-path



Working!



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

```
root@Snort:~/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/2024/06/11/202
4-06-11-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

```
                seconds: 0.043616
o")~   Snort exiting
root@Snort:~/test# snort -c /usr/local/etc/snort/snort.lua --plugin-path /usr/lo
cal/etc/so_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@Snort:~/test# snort -c /usr/local/etc/snort/snort.lua --plugin-path /usr/lo
cal/etc/so_rules/ -r 2024-06-11-CVE-2024-4577-probe.pcap -A alert_fast -q
06/11-22:47:26.511971 [**] [1:63598:1] "SERVER-WEBAPP PHP PHP-CGI command execut
ion attempt" [**] [Classification: Attempted User Privilege Gain] [Priority: 1]
{TCP} 221.122.67.75:60482 -> 76.223.105.231:80
root@Snort:~/test#
```

Let's try another one, this time a RAT

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

NOTES:

- Mithrandir (**@rerednawyerg**) 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-Yl3c**
- **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)

```
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-a
nd-XWorm-infection-traffic.pcap password:
  inflating: 2024-03-14-AsyncRAT-and-XWorm-infection-traffic.pcap
root@Snort:~/test#
```

We can see and ARP_Spoofing alert (on path attack)

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

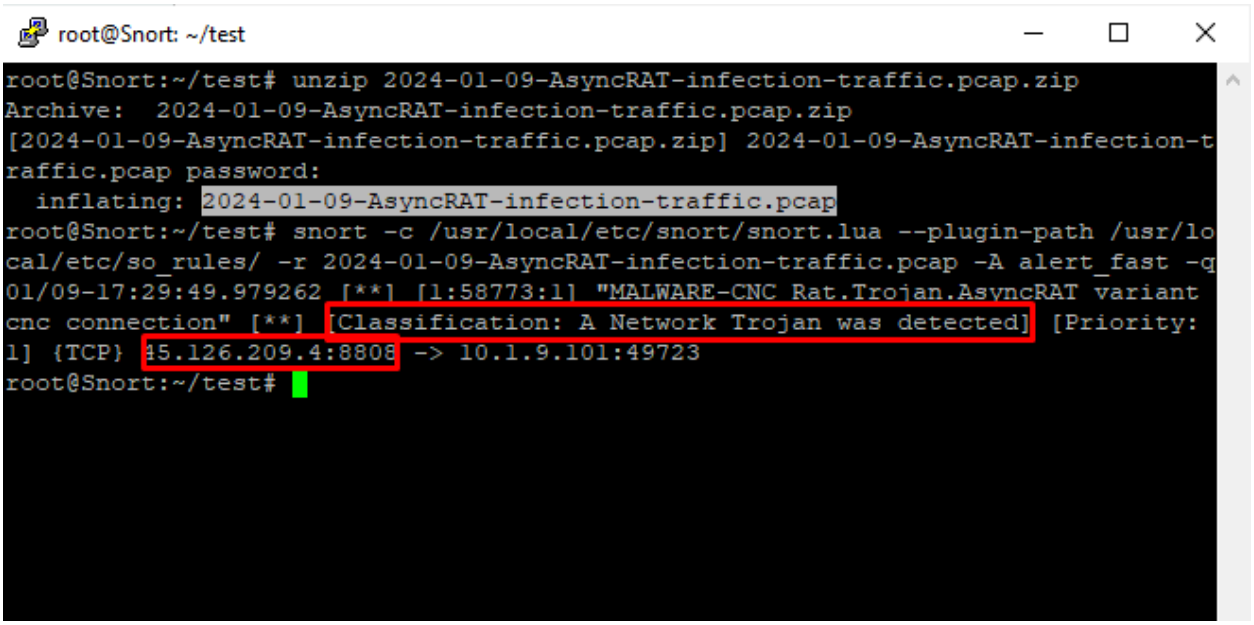
REFERENCE:

- **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



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/lo
cal/etc/so_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" [**] [Pri
ority: 3] {ARP}  ->
05/09-21:33:54.403569 [**] [112:1:1] "(arp_spoof) unicast ARP request" [**] [Pri
ority: 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:
```

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@Snort:~/test# ^C
root@Snort:~/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.p
cap password:
  inflating: 2024-04-17-TA578-SSLoad-infection.pcap
root@Snort:~/test# ^C
root@Snort:~/test#
root@Snort:~/test#
root@Snort:~/test# snort -c /usr/local/etc/snort/snort.lua --plugin-path /usr/lo
cal/etc/so_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@Snort:~/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.