TDX ARENA

Certification Report

TDX ARENA

Anthony Frederick
Final Assessment Report Submission

Case: Pigs Rule

8/5/2024

Executive Summary

The Flying Piglet post office gained intel about hacktivists who planned to launch a hacking campaign. To try and stay ahead of the attackers, we needed to sniff incoming traffic, identify the malicious traffic, and then configure a Snort IDS rule to capture said malicious traffic without capturing legitimate traffic.

Findings and Analysis

Finding	Finding Details	Description
Packet Attributes	Small TCP window sizes	Attributes of network packets relevant to the investigation

Using small TCP window sizes for malicious purposes can involve various strategies, particularly in the context of network attacks and evading detection including (but not limited to) network reconnaissance, denial of service attacks, and TCP/IP stack overloading. I believed the small window sizes I found to be indicative of a reconnaissance called fingerprinting.

Finding	Finding Details	Description
Attack	Fingerprinting	A process of gathering detailed information about a system or network to create a unique profile of it.

A possible technique the hacktivist group could be using to compromise our systems by actively sending crafted packets and analyzing the responses to determine our OS to find any vulnerabilities we may have.

Finding	Finding Details	Description
Attack	Denial of Service (DoS)	A type of cyber attack aimed at disrupting the normal functioning of a targeted service, network, or system.

By sending numerous small window size packets, an attacker can cause the receiver's buffer to become overwhelmed. This can lead to resource exhaustion on the receiver's side, potentially causing a DoS condition where legitimate traffic cannot be processed.

Finding	Finding Details	Description
IP Address	172.29.0.1	The IP address which was sending multiple packets at our server.

This IP that was logged in Snorby after setting up the rule to alert of small tcp window sizes. Possibly the source of an attempted DoS attack.

Methodology

Tools and Technologies Used

List and describe the tools and technologies employed in the investigation, including a brief explanation of why they were used.

Snort: An open-source network intrusion detection system (NIDS) that monitors and analyzes network traffic for malicious activity based on predefined rules.

Snorby: A web-based front-end interface for Snort that provides a graphical user interface for managing, viewing, and analyzing Snort alerts and logs.

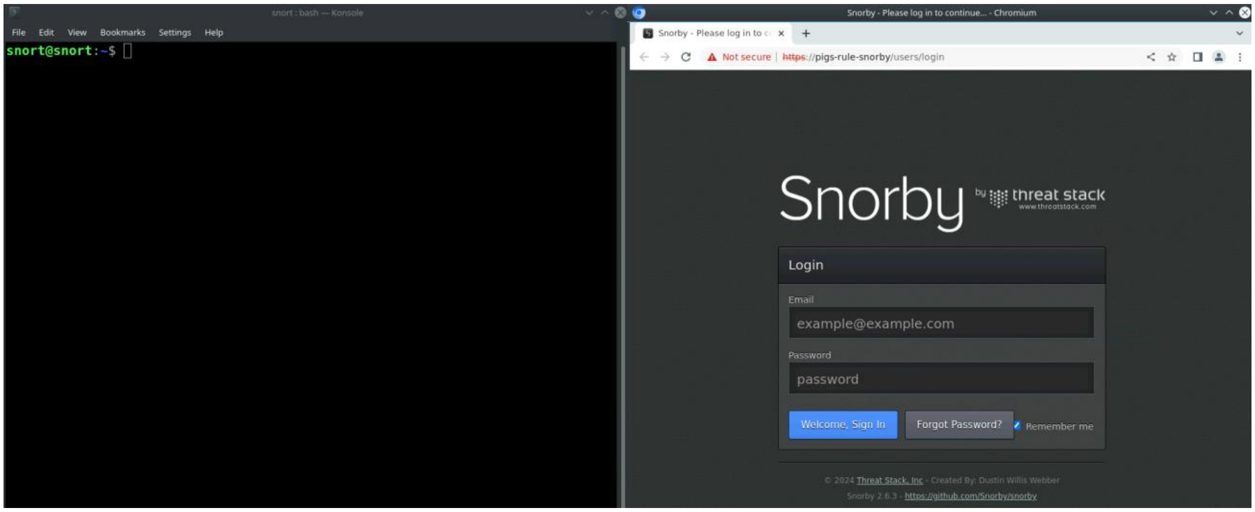
TCPDump: A command-line packet analyzer tool used to capture and analyze network traffic, often used for debugging and network troubleshooting.

TCP Window: A TCP header field that specifies the amount of data the sender is willing to receive before acknowledging. It helps manage flow control in TCP connections.

Investigation Process

1) Upon entering the machine, we could see an open terminal and a web browser opened to Snorby (IMAGE 1). On the desktop there was a README file in the home directory, which provided some basic snort commands (IMAGE 2).

IMAGE 1



snort : bash — Konsole
File Edit View Bookmarks Settings Help
Snorby - Please log in to continue... - Chromium
Snorby - Please log in to c
Not secure | https://pigs-rule-snorby/users/login
Snorby by threat stack www.threatstack.com
Login
Welcome, Sign In
Forgot Password?
© 2024 Threat Stack, Inc - Created By: Dustin Willis Webber
Snorby 2.6.3 - https://github.com/Snorby/snorby

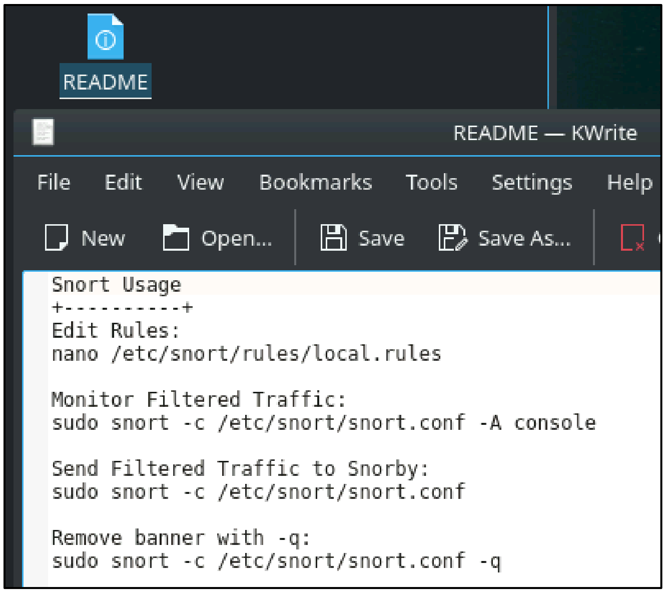
```
snort@snort:~$ 
```

Email: example@example.com

Password: password

☑ Remember me

IMAGE 2

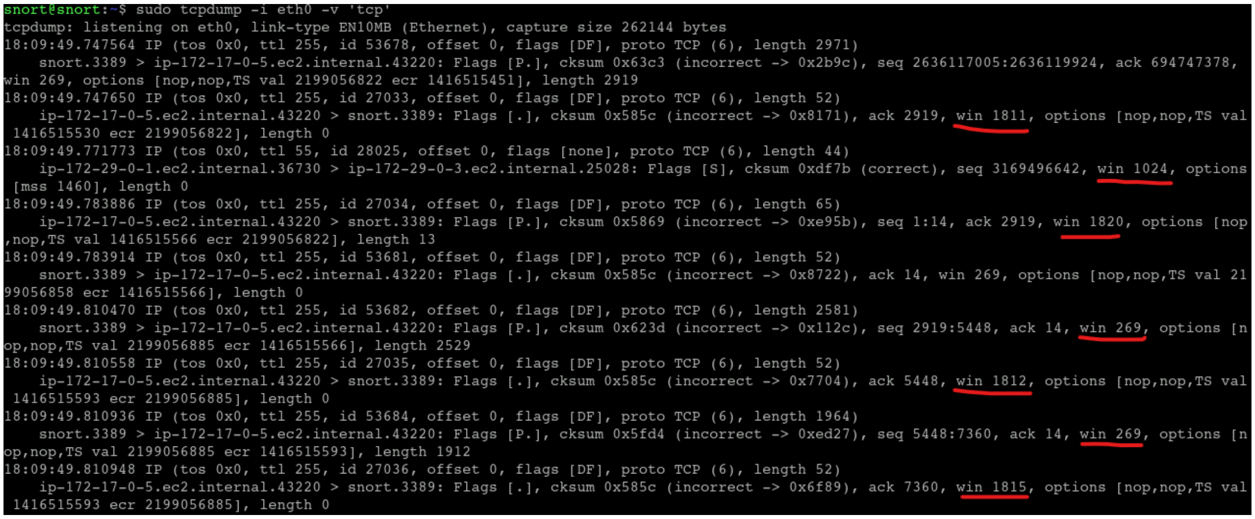


README
README — KWrite
File Edit View Bookmarks Tools Settings Help
New Open... Save Save As...

```
Snort Usage
+----------+
Edit Rules:
nano /etc/snort/rules/local.rules

Monitor Filtered Traffic:
sudo snort -c /etc/snort/snort.conf -A console

Send Filtered Traffic to Snorby:
sudo snort -c /etc/snort/snort.conf

Remove banner with -q:
sudo snort -c /etc/snort/snort.conf -q
```

2) Knowing that I needed to review incoming traffic I ran tcpdump with my interface (-i) as eth0 and I used verbose (-v) just to provide a little more detail on the information coming in. I ran a few of the lines returned through chatgpt to see if they saw anything suspicious, and what was returned educated the small TCP Window sizes with multiple different source IPs could indicate a certain type of scanning or fingerprinting activity. (IMAGE3)

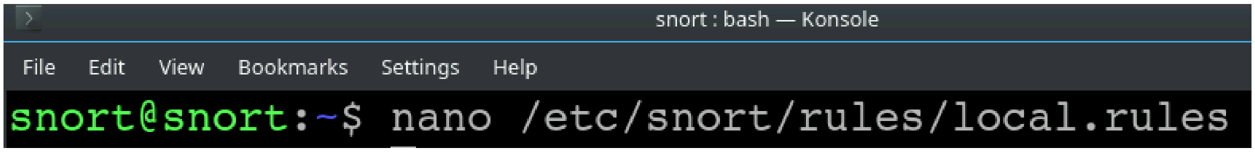
IMAGE 3



```
snort@snort:~$ sudo tcpdump -i eth0 -v 'tcp'
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
18:09:49.747564 IP (tos 0x0, ttl 255, id 53678, offset 0, flags [DF], proto TCP (6), length 2971)
    snort.3389 > ip-172-17-0-5.ec2.internal.43220: Flags [P.], cksum 0x63c3 (incorrect -> 0x2b9c), seq 2636117005:2636119924, ack 694747378,
win 269, options [nop,nop,TS val 2199056822 ecr 1416515451], length 2919
18:09:49.747650 IP (tos 0x0, ttl 255, id 27033, offset 0, flags [DF], proto TCP (6), length 52)
    ip-172-17-0-5.ec2.internal.43220 > snort.3389: Flags [.], cksum 0x585c (incorrect -> 0x8171), ack 2919, win 1811, options [nop,nop,TS val
 1416515530 ecr 2199056822], length 0
18:09:49.771773 IP (tos 0x0, ttl 55, id 28025, offset 0, flags [none], proto TCP (6), length 44)
    ip-172-29-0-1.ec2.internal.36730 > ip-172-29-0-3.ec2.internal.25028: Flags [S], cksum 0xdf7b (correct), seq 3169496642, win 1024, options
 [mss 1460], length 0
18:09:49.783886 IP (tos 0x0, ttl 255, id 27034, offset 0, flags [DF], proto TCP (6), length 65)
    ip-172-17-0-5.ec2.internal.43220 > snort.3389: Flags [P.], cksum 0x5869 (incorrect -> 0xe95b), seq 1:14, ack 2919, win 1820, options [nop
,nop,TS val 1416515566 ecr 2199056822], length 13
18:09:49.783914 IP (tos 0x0, ttl 255, id 53681, offset 0, flags [DF], proto TCP (6), length 52)
    snort.3389 > ip-172-17-0-5.ec2.internal.43220: Flags [.], cksum 0x585c (incorrect -> 0x8722), ack 14, win 269, options [nop,nop,TS val 21
99056858 ecr 1416515566], length 0
18:09:49.810470 IP (tos 0x0, ttl 255, id 53682, offset 0, flags [DF], proto TCP (6), length 2581)
    snort.3389 > ip-172-17-0-5.ec2.internal.43220: Flags [P.], cksum 0x623d (incorrect -> 0x112c), seq 2919:5448, ack 14, win 269, options [n
op,nop,TS val 2199056885 ecr 1416515566], length 2529
18:09:49.810558 IP (tos 0x0, ttl 255, id 27035, offset 0, flags [DF], proto TCP (6), length 52)
    ip-172-17-0-5.ec2.internal.43220 > snort.3389: Flags [.], cksum 0x585c (incorrect -> 0x7704), ack 5448, win 1812, options [nop,nop,TS val
 1416515593 ecr 2199056885], length 0
18:09:49.810936 IP (tos 0x0, ttl 255, id 53684, offset 0, flags [DF], proto TCP (6), length 1964)
    snort.3389 > ip-172-17-0-5.ec2.internal.43220: Flags [P.], cksum 0x5fd4 (incorrect -> 0xed27), seq 5448:7360, ack 14, win 269, options [n
op,nop,TS val 2199056885 ecr 1416515593], length 1912
18:09:49.810948 IP (tos 0x0, ttl 255, id 27036, offset 0, flags [DF], proto TCP (6), length 52)
    ip-172-17-0-5.ec2.internal.43220 > snort.3389: Flags [.], cksum 0x585c (incorrect -> 0x6f89), ack 7360, win 1815, options [nop,nop,TS val
 1416515593 ecr 2199056885], length 0
```

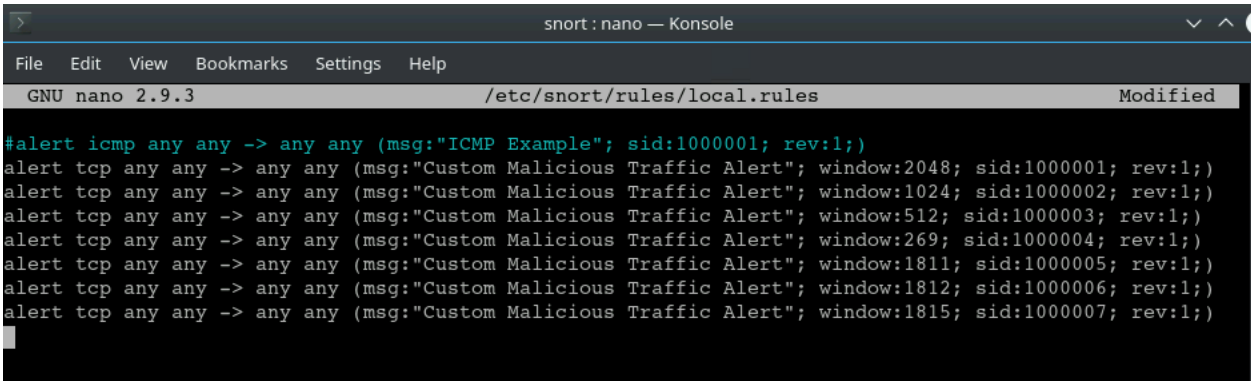
3) At this point, my hypothesis was the hacktivist group was trying to use fingerprinting or scanning to find vulnerabilities in our systems or software. Next, I needed to create a specific rule in snort that would sniff out low tcp window sizes and then run snort through Snorby. Unfortunately, I was not able to create a rule that allowed for me to look for these window sizes like I would have wanted to using a condition like "less than (<)." Instead, I needed to create a few different alerts with specific numerical values within Snort's local rules based on window sizes I saw in my initial scan. (IMAGE4 and IMAGE 5).

IMAGE 4



```
snort@snort:~$ nano /etc/snort/rules/local.rules
```

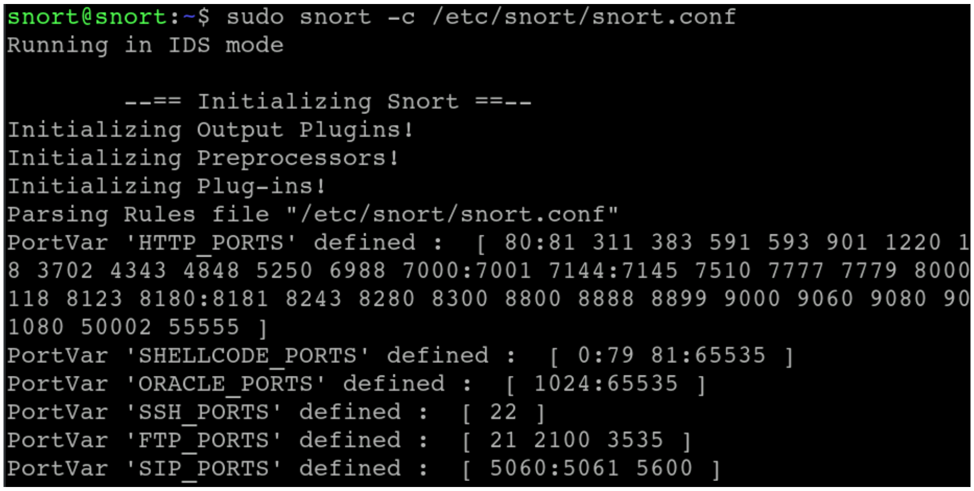
IMAGE 5



```
  GNU nano 2.9.3                 /etc/snort/rules/local.rules                 Modified

#alert icmp any any -> any any (msg:"ICMP Example"; sid:1000001; rev:1;)
alert tcp any any -> any any (msg:"Custom Malicious Traffic Alert"; window:2048; sid:1000001; rev:1;)
alert tcp any any -> any any (msg:"Custom Malicious Traffic Alert"; window:1024; sid:1000002; rev:1;)
alert tcp any any -> any any (msg:"Custom Malicious Traffic Alert"; window:512; sid:1000003; rev:1;)
alert tcp any any -> any any (msg:"Custom Malicious Traffic Alert"; window:269; sid:1000004; rev:1;)
alert tcp any any -> any any (msg:"Custom Malicious Traffic Alert"; window:1811; sid:1000005; rev:1;)
alert tcp any any -> any any (msg:"Custom Malicious Traffic Alert"; window:1812; sid:1000006; rev:1;)
alert tcp any any -> any any (msg:"Custom Malicious Traffic Alert"; window:1815; sid:1000007; rev:1;)
```

4) After setting up the new alert rule, I then ran the command to run snort which would notify of any alerts through Snorby (IMAGE 6).

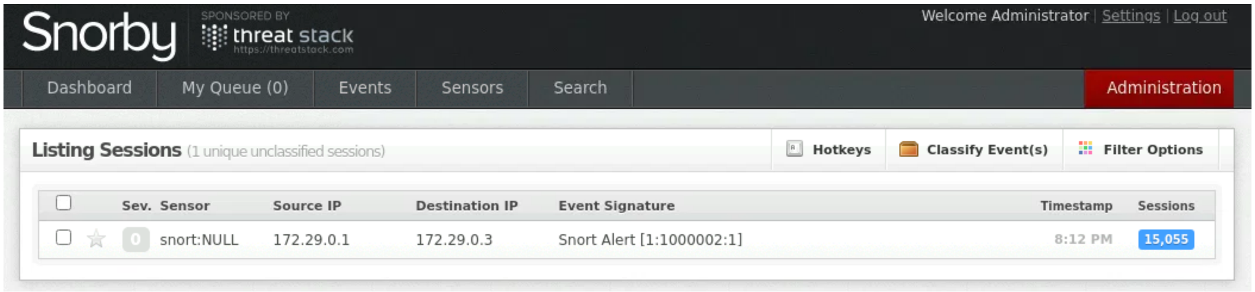
IMAGE 6



```
snort@snort:~$ sudo snort -c /etc/snort/snort.conf
Running in IDS mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
PortVar 'HTTP_PORTS' defined :  [ 80:81 311 383 591 593 901 1220 1
8 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 7779 8000
118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 90
1080 50002 55555 ]
PortVar 'SHELLCODE_PORTS' defined :  [ 0:79 81:65535 ]
PortVar 'ORACLE_PORTS' defined :  [ 1024:65535 ]
PortVar 'SSH_PORTS' defined :  [ 22 ]
PortVar 'FTP_PORTS' defined :  [ 21 2100 3535 ]
PortVar 'SIP_PORTS' defined :  [ 5060:5061 5600 ]
```

5) I Signed into Snorby and went to check if I was collecting any events. Snorby showed multiple sessions coming in with my alert, but only for my 2nd rule for size 1024. In this I could also see one source IP and one destination IP (IMAGE 7). When I tried to run a whois on this IP address I returned zero results.

IMAGE 7

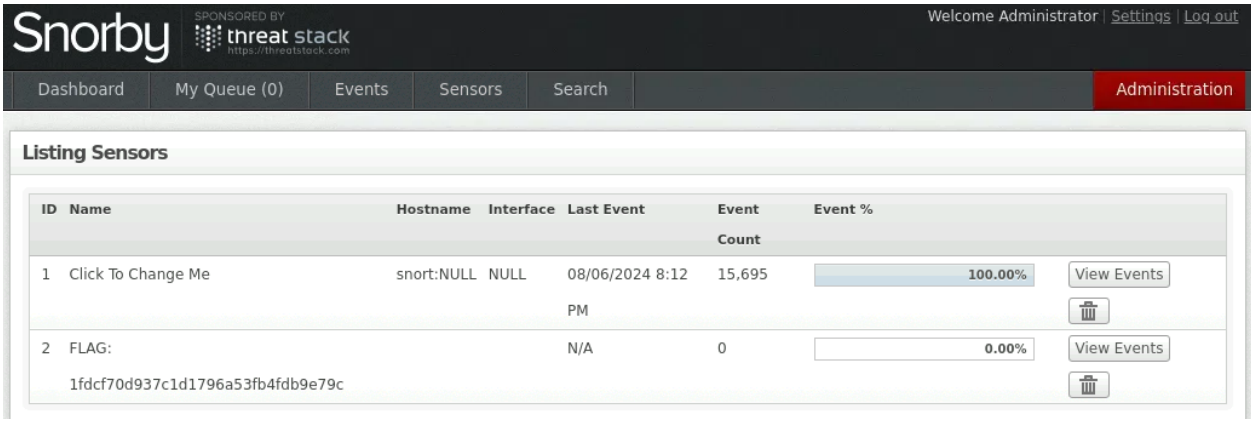


Listing Sessions (1 unique unclassified sessions)

□		Sev.	Sensor	Source IP	Destination IP	Event Signature	Timestamp	Sessions
□	☆	0	snort:NULL	172.29.0.1	172.29.0.3	Snort Alert [1:1000002:1]	8:12 PM	15,055

6) When checking the sensors tab, I was able to see the flag for the challenge (IMAGE 8). I am also considering my original hypothesis of fingerprinting to be incorrect as the event count shows this could be an attempt at a DoS attack.

IMAGE 8



Listing Sensors

ID	Name	Hostname	Interface	Last Event	Event Count	Event %	
1	Click To Change Me	snort:NULL	NULL	08/06/2024 8:12 PM	15,695	100.00%	View Events
2	FLAG: 1fdcf70d937c1d1796a53fb4fdb9e79c			N/A	0	0.00%	View Events

Recommendations

Based on the findings, provide clear and actionable recommendations to mitigate the identified risks, secure the systems, and prevent future incidents.)

Network Segmentation: Isolate critical systems and control traffic flow.

Firewall and Intrusion Detection/Prevention: Detect and block suspicious activities in real-time. Set up firewall rules to block or rate limit suspicious IP addresses.

Regular Updates/Patching: Protect against known vulnerabilities with up-to-date systems.

Access Controls: Implement MFA and least privilege on any devices that may need it.