

Student:

Isaiah Mosley

Email:

isaiahmosley80@gmail.com

Time on Task:

104 hours, 46 minutes

Progress:

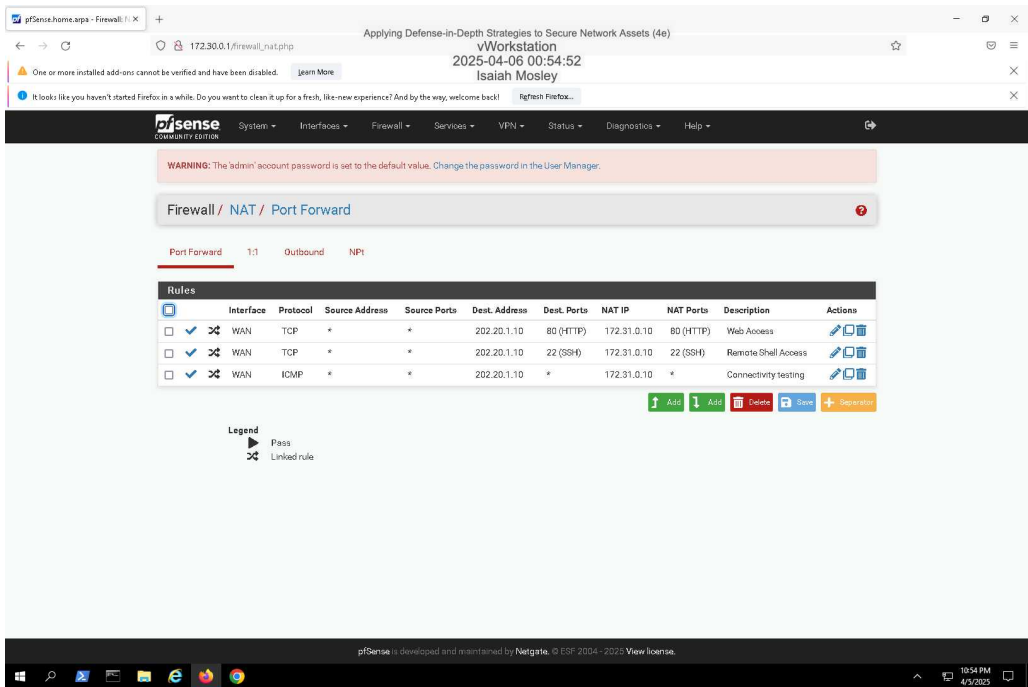
100%

Report Generated: Tuesday, December 2, 2025 at 1:25 PM

Hands-On Demonstration

Part 1: Update the Firewall Rules

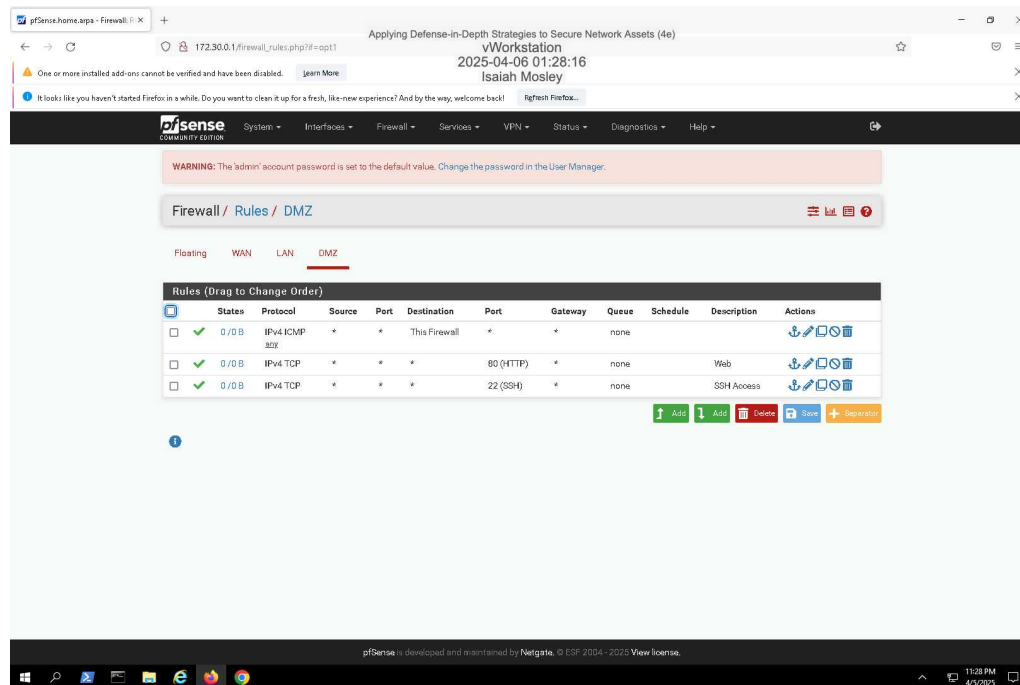
10. Make a screen capture showing the Port Forward rules for WebServer01.



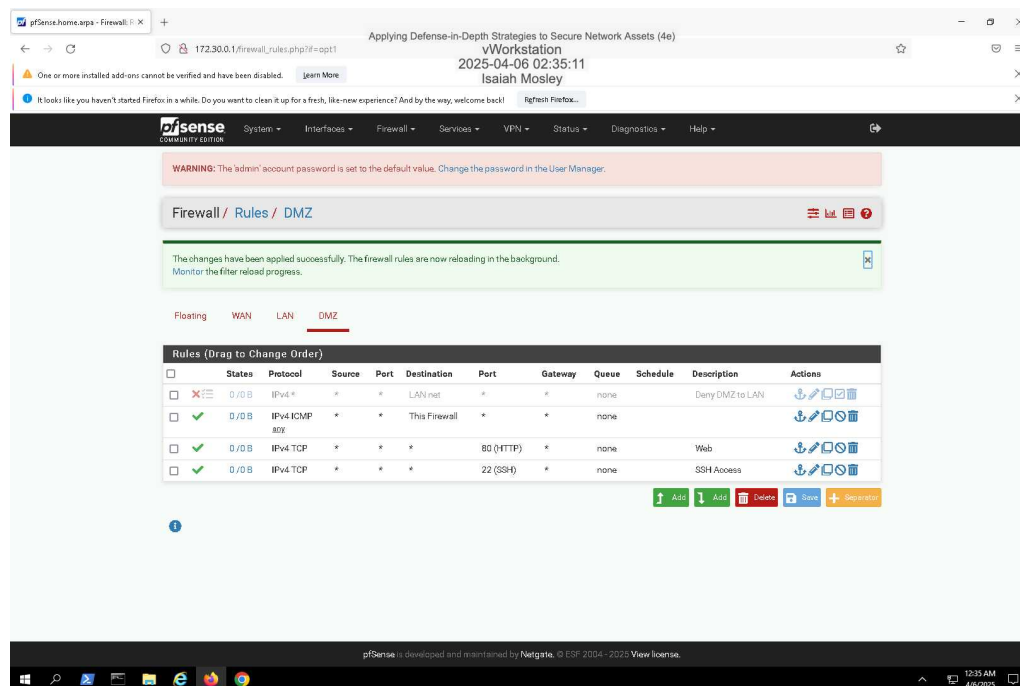
Applying Defense-in-Depth Strategies to Secure Network Assets (4e)

Ethical Hacking, Fourth Edition - Lab 10

14. Make a screen capture showing the permissive DMZ ruleset.

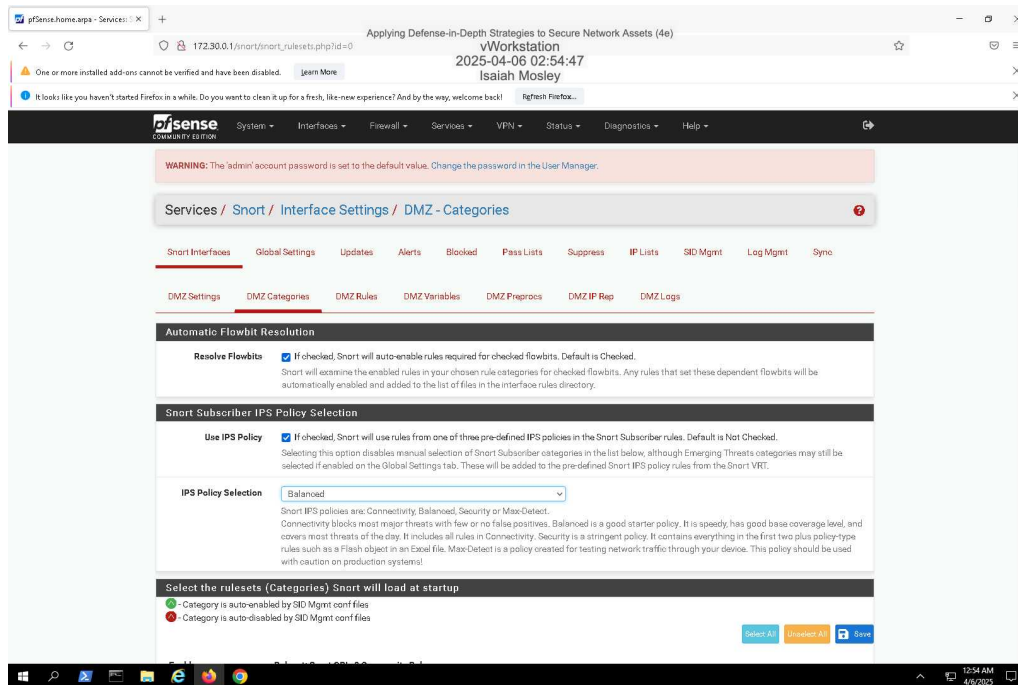


25. Make a screen capture showing the updated firewall ruleset for the DMZ.

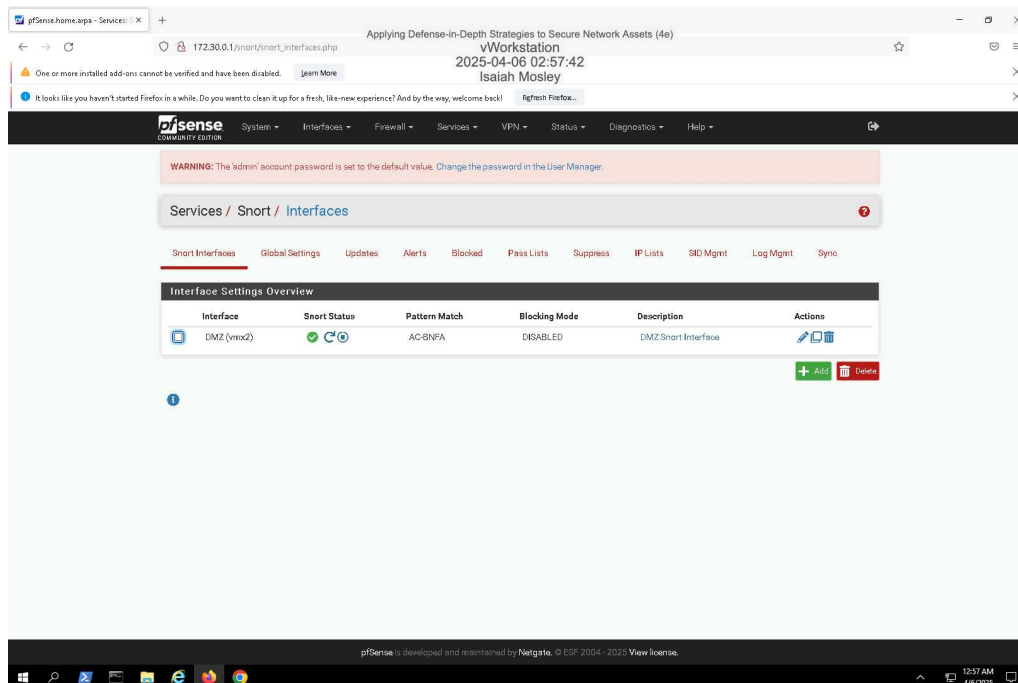


Part 2: Configure a Network-based IDS on pfSense

23. Make a screen capture showing a Balanced IPS policy.

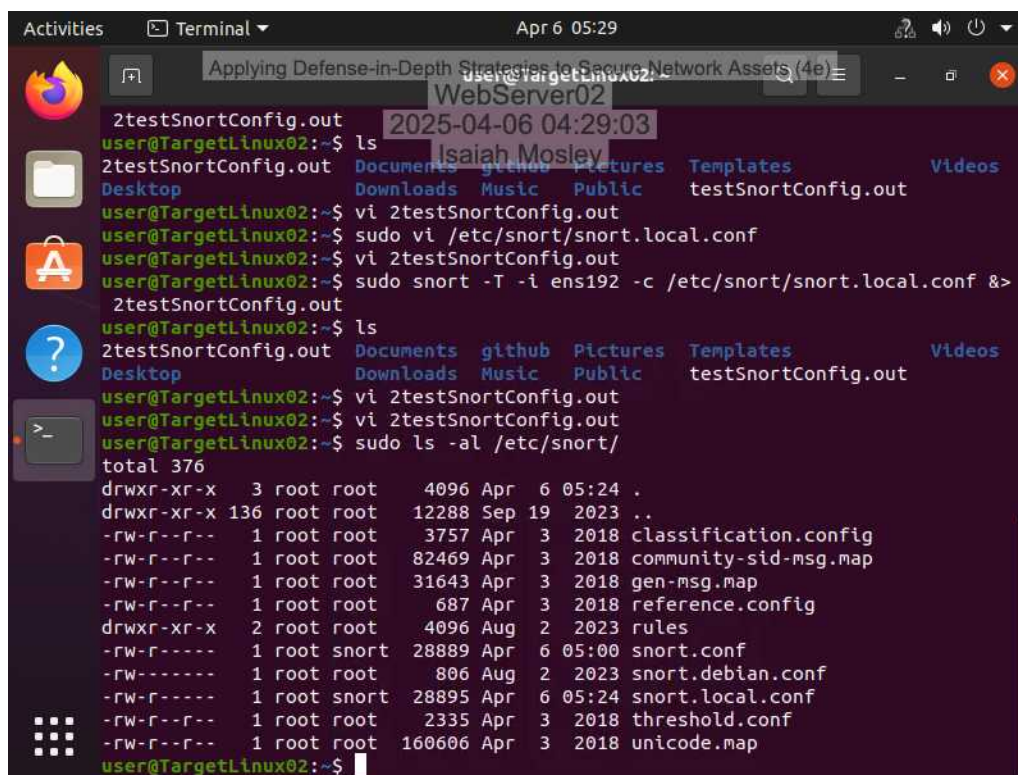


30. Make a screen capture showing that the Snort (DMZ) Interface is running in the Interface Settings Overview.



Part 3: Configure a Host-based IDS (HIDS) on Linux

38. Make a screen capture showing the contents of the /etc/snort directory with your local config file.



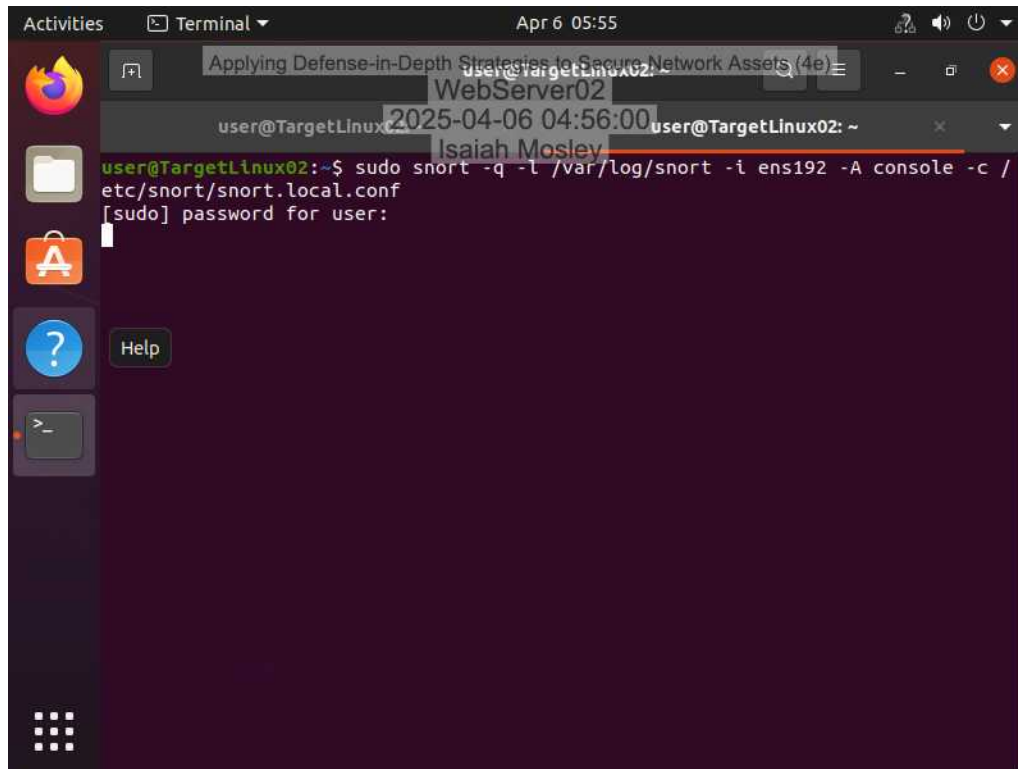
A terminal window titled 'Terminal' with a date and time of 'Apr 6 05:29'. The window shows a series of commands and their outputs. The user is at a prompt 'user@TargetLinux02:~\$'. They run 'ls' and see a list of files including '2testSnortConfig.out', 'Documents', 'github', 'Pictures', 'Templates', 'Videos', 'Desktop', 'Downloads', 'Music', 'Public', and 'testSnortConfig.out'. They then run 'vi 2testSnortConfig.out', 'sudo vi /etc/snort/snort.local.conf', and 'sudo snort -T -i ens192 -c /etc/snort/snort.local.conf &'. Finally, they run 'sudo ls -al /etc/snort/' and see a detailed directory listing of the /etc/snort directory.

```
user@TargetLinux02:~$ ls
2testSnortConfig.out  Documents  github  Pictures  Templates  Videos
Desktop               Downloads  Music   Public   testSnortConfig.out

user@TargetLinux02:~$ vi 2testSnortConfig.out
user@TargetLinux02:~$ sudo vi /etc/snort/snort.local.conf
user@TargetLinux02:~$ vi 2testSnortConfig.out
user@TargetLinux02:~$ sudo snort -T -i ens192 -c /etc/snort/snort.local.conf &
2testSnortConfig.out
user@TargetLinux02:~$ ls
2testSnortConfig.out  Documents  github  Pictures  Templates  Videos
Desktop               Downloads  Music   Public   testSnortConfig.out

user@TargetLinux02:~$ vi 2testSnortConfig.out
user@TargetLinux02:~$ vi 2testSnortConfig.out
user@TargetLinux02:~$ sudo ls -al /etc/snort/
total 376
drwxr-xr-x  3 root root    4096 Apr  6 05:24 .
drwxr-xr-x 136 root root   12288 Sep 19  2023 ..
-rw-r--r--  1 root root    3757 Apr  3  2018 classification.config
-rw-r--r--  1 root root   82469 Apr  3  2018 community-sid-msg.map
-rw-r--r--  1 root root   31643 Apr  3  2018 gen-msg.map
-rw-r--r--  1 root root    687 Apr  3  2018 reference.config
drwxr-xr-x  2 root root    4096 Aug  2  2023 rules
-rw-r----- 1 root snort  28889 Apr  6 05:00 snort.conf
-rw-r----- 1 root root    806 Aug  2  2023 snort.debian.conf
-rw-r----- 1 root snort  28895 Apr  6 05:24 snort.local.conf
-rw-r--r--  1 root root    2335 Apr  3  2018 threshold.conf
-rw-r--r--  1 root root   160606 Apr  3  2018 unicode.map
user@TargetLinux02:~$
```

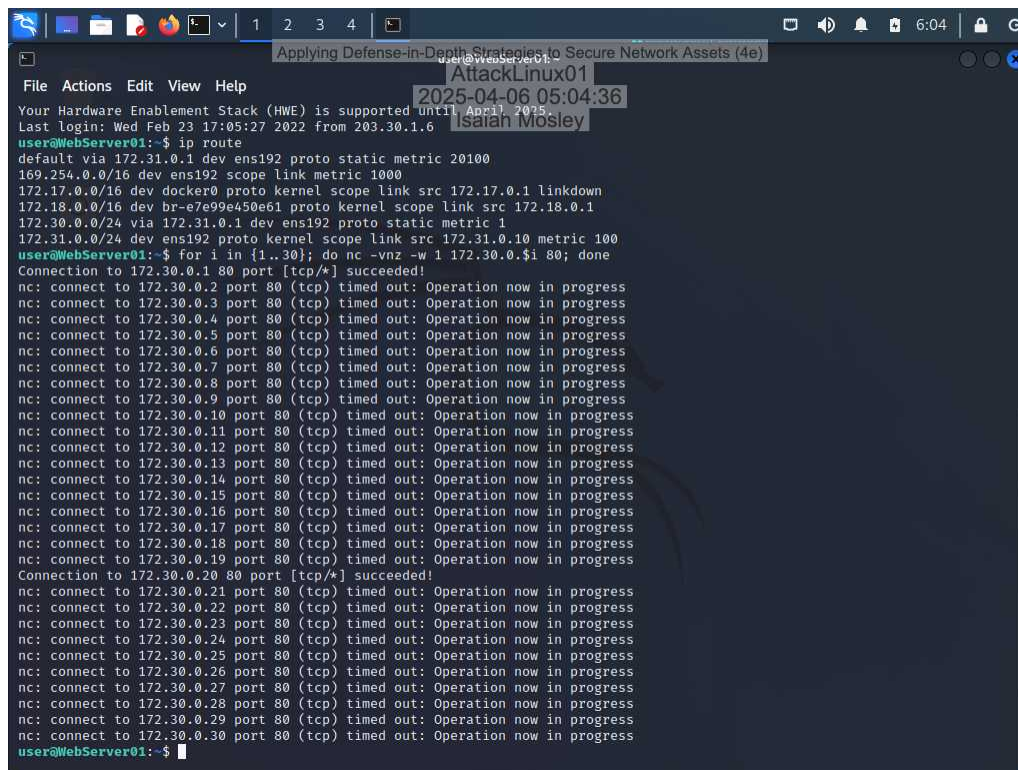
51. Make a screen capture showing the successful start of Snort as HIDS on WebServer02.



Applied Learning

Part 1: Validate Your IDS Security Controls

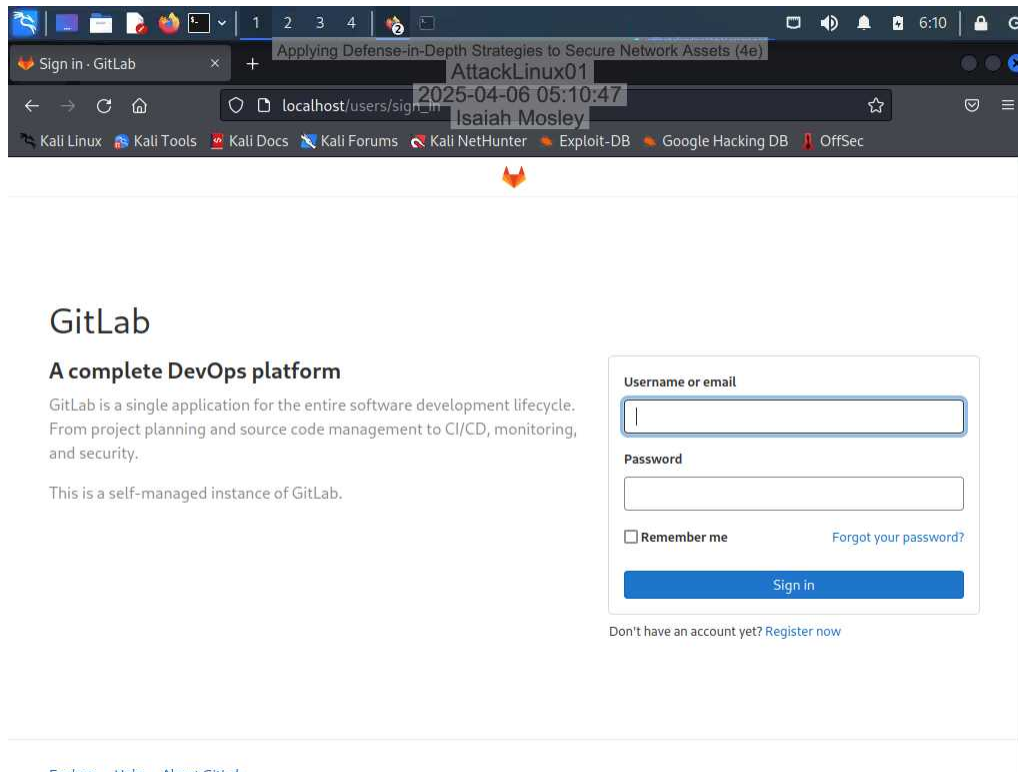
9. Make a screen capture showing the successful Netcat connection to WebServer02 at 172.30.0.20.



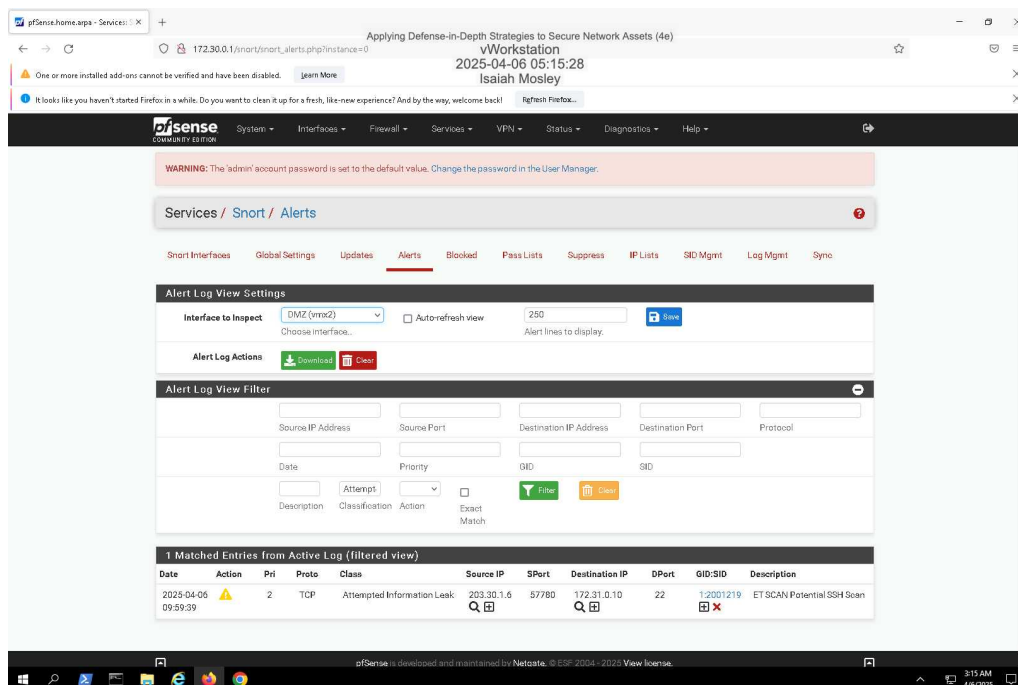
The screenshot shows a terminal window titled "Applying Defense-in-Depth Strategies to Secure Network Assets (4e)". The terminal output shows the user running the command `ip route` and then a loop of Netcat connections to various IP addresses in the 172.30.0.0/24 range. The connection to 172.30.0.20 is successful, while all others time out. A watermark for "AttackLinux01" and "Isaiah Mosley" is visible over the terminal output.

```
File Actions Edit View Help
Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Wed Feb 23 17:05:27 2022 from 203.30.1.6
user@WebServer01:~$ ip route
default via 172.31.0.1 dev ens192 proto static metric 20100
169.254.0.0/16 dev ens192 scope link metric 1000
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
172.18.0.0/16 dev br-e7e99e450e61 proto kernel scope link src 172.18.0.1
172.30.0.0/24 via 172.31.0.1 dev ens192 proto static metric 1
172.31.0.0/24 dev ens192 proto kernel scope link src 172.31.0.1 metric 100
user@WebServer01:~$ for i in {1..30}; do nc -vzn -w 1 172.30.0.$i 80; done
Connection to 172.30.0.1 80 port [tcp/*] succeeded!
nc: connect to 172.30.0.2 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.3 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.4 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.5 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.6 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.7 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.8 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.9 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.10 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.11 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.12 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.13 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.14 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.15 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.16 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.17 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.18 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.19 port 80 (tcp) timed out: Operation now in progress
Connection to 172.30.0.20 80 port [tcp/*] succeeded!
nc: connect to 172.30.0.21 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.22 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.23 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.24 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.25 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.26 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.27 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.28 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.29 port 80 (tcp) timed out: Operation now in progress
nc: connect to 172.30.0.30 port 80 (tcp) timed out: Operation now in progress
user@WebServer01:~$
```


17. Make a screen capture showing the login page of the internal GitLab server at localhost/users/sign_in.



25. Make a screen capture showing the Attempted Information Leak alert.



Activities Terminal Apr 6 06:16

Applying Defense-in-Depth Strategies to Secure Network Assets (4e)

user@TargetLinux02: ~

2025-04-06 05:16:25

Isaiah Mosley

```

rriority: 0] {TCP} 172.31.0.10:45138 -> 172.30.0.20:80
04/06-06:16:08.527843 [**] [1:382:7] ICMP PING Windows [**] [Classification: M
isc activity] [Priority: 3] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:08.527843 [**] [1:1000001:1] DETECTED: ICMP Ping [**] [Priority: 0
] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:08.527843 [**] [1:384:5] ICMP PING [**] [Classification: Misc acti
vity] [Priority: 3] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:09.535226 [**] [1:382:7] ICMP PING Windows [**] [Classification: M
isc activity] [Priority: 3] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:09.535226 [**] [1:1000001:1] DETECTED: ICMP Ping [**] [Priority: 0
] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:09.535226 [**] [1:384:5] ICMP PING [**] [Classification: Misc acti
vity] [Priority: 3] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:10.551644 [**] [1:382:7] ICMP PING Windows [**] [Classification: M
isc activity] [Priority: 3] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:10.551644 [**] [1:1000001:1] DETECTED: ICMP Ping [**] [Priority: 0
] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:10.551644 [**] [1:384:5] ICMP PING [**] [Classification: Misc acti
vity] [Priority: 3] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:11.567317 [**] [1:382:7] ICMP PING Windows [**] [Classification: M
isc activity] [Priority: 3] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:11.567317 [**] [1:1000001:1] DETECTED: ICMP Ping [**] [Priority: 0
] {ICMP} 172.30.0.2 -> 172.30.0.20
04/06-06:16:11.567317 [**] [1:384:5] ICMP PING [**] [Classification: Misc acti
vity] [Priority: 3] {ICMP} 172.30.0.2 -> 172.30.0.20

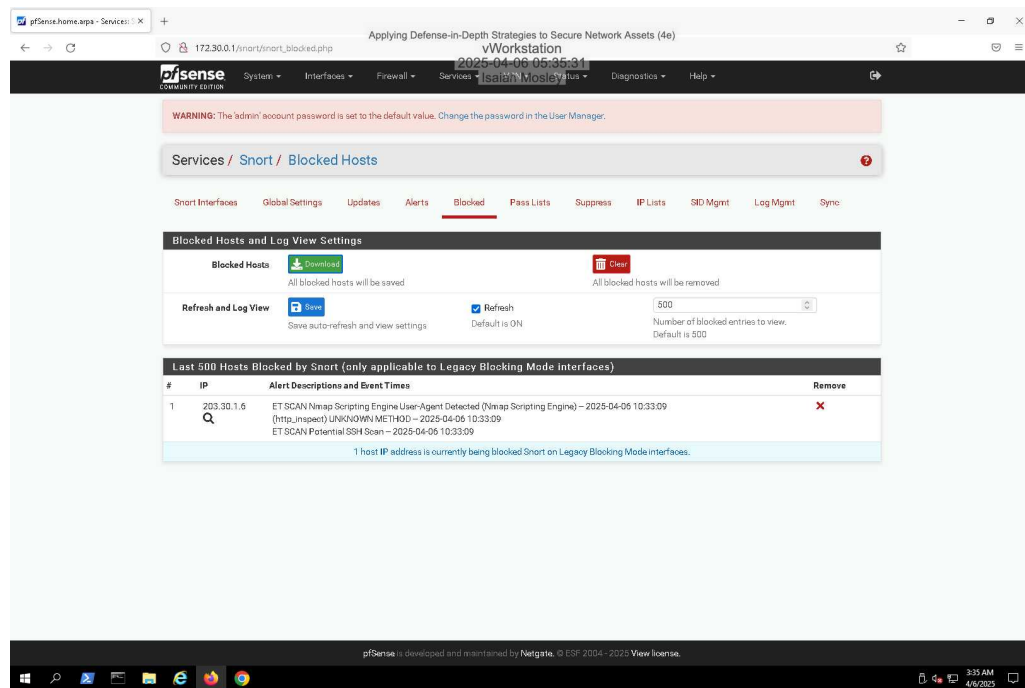
```

Page 8 of 12

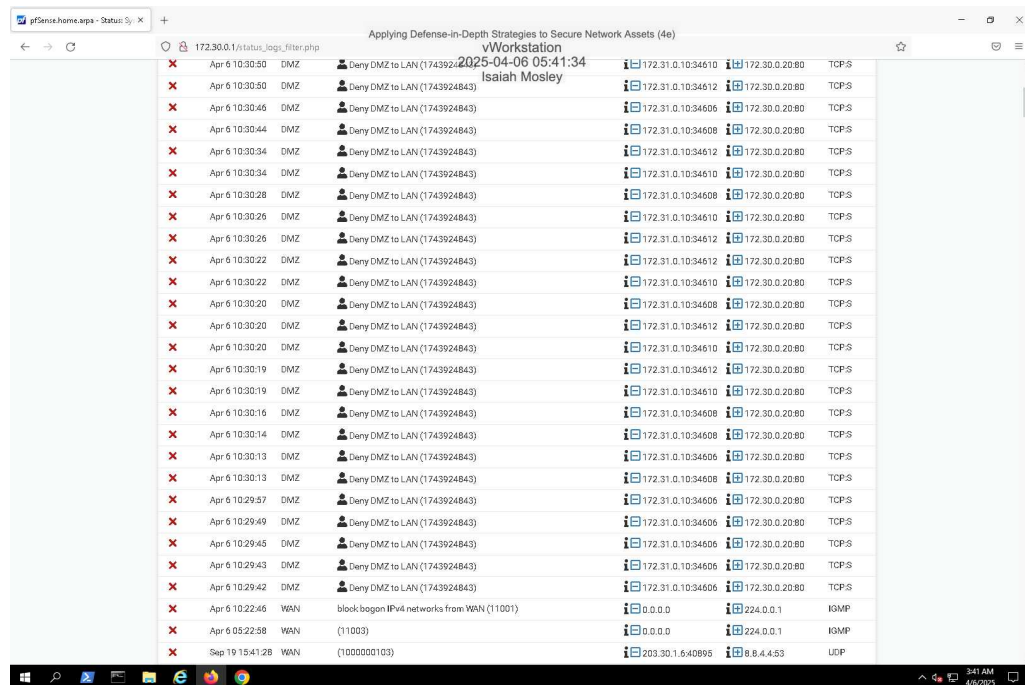
Applying Defense-in-Depth Strategies to Secure Network Assets (4e)

Ethical Hacking, Fourth Edition - Lab 10

27. Make a screen capture showing the blocked IP in the *Last 500 Hosts Blocked by Snort* widget.

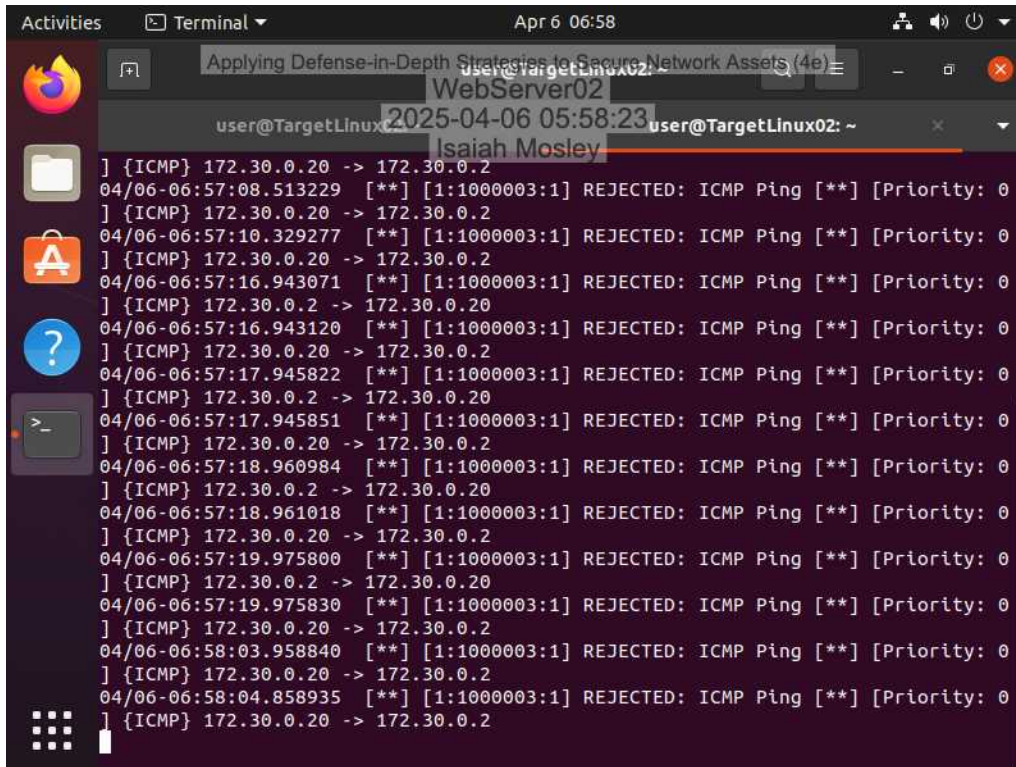


30. Make a screen capture showing the blocked connection from the DMZ to the LAN.



Part 3: Explore Protection Mode Options (HIPS) on Host-based IDS

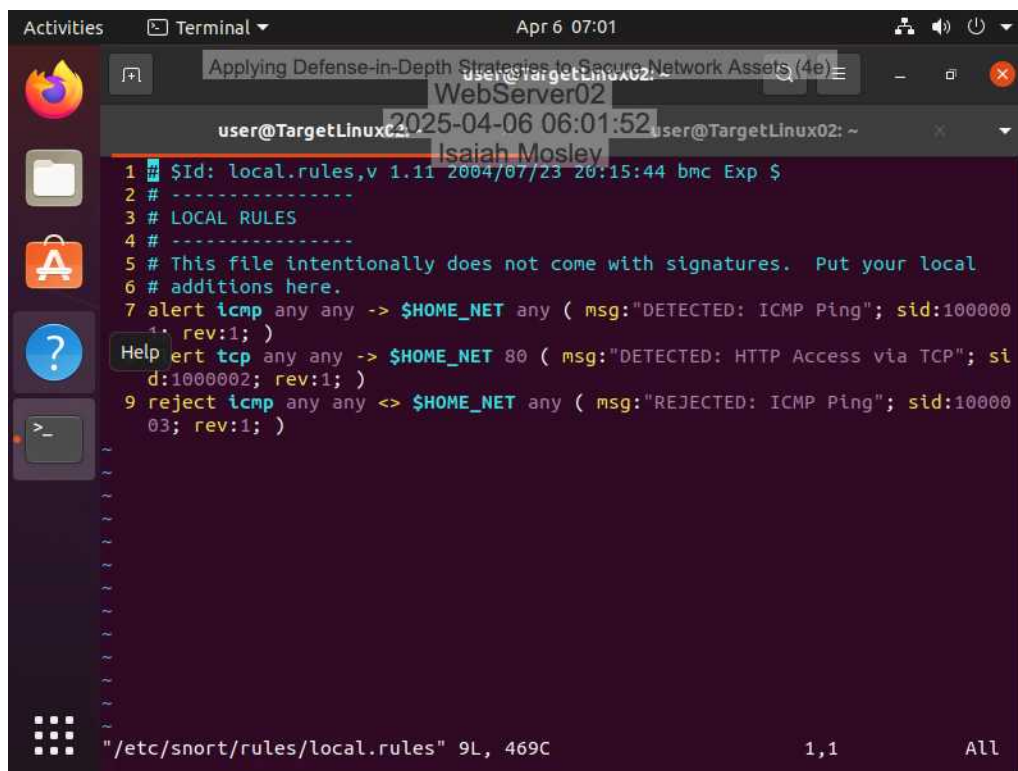
24. Make a screen capture showing the ping responses from WebServer02.



The screenshot shows a Linux terminal window titled "Terminal" with a timestamp of "Apr 6 06:58". The terminal output displays a series of failed ICMP ping attempts to the IP address 172.30.0.2. Each line shows the time, a sequence number, and the result "REJECTED: ICMP Ping". The terminal also shows the user "user@TargetLinux02" and the hostname "WebServer02".

```
user@TargetLinux02: ~  
] {ICMP} 172.30.0.20 -> 172.30.0.2  
04/06-06:57:08.513229  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.2  
04/06-06:57:10.329277  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.2  
04/06-06:57:16.943071  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.20  
04/06-06:57:16.943120  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.2  
04/06-06:57:17.945822  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.20  
04/06-06:57:17.945851  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.2  
04/06-06:57:18.960984  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.20  
04/06-06:57:18.961018  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.2  
04/06-06:57:19.975800  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.20  
04/06-06:57:19.975830  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.2  
04/06-06:58:03.958840  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.2  
04/06-06:58:04.858935  [**] [1:1000003:1] REJECTED: ICMP Ping [**] [Priority: 0  
] {ICMP} 172.30.0.20 -> 172.30.0.2
```

26. Make a screen capture showing output that indicates that the rejection rules fired.



```
1 $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
2 # -----
3 # LOCAL RULES
4 # -----
5 # This file intentionally does not come with signatures. Put your local
6 # additions here.
7 alert icmp any any -> $HOME_NET any ( msg:"DETECTED: ICMP Ping"; sid:100000
8   rev:1; )
9 alert tcp any any -> $HOME_NET 80 ( msg:"DETECTED: HTTP Access via TCP"; si
10   d:1000002; rev:1; )
11 reject icmp any any <=> $HOME_NET any ( msg:"REJECTED: ICMP Ping"; sid:10000
12   03; rev:1; )
```

27. Document this protection deficiency and suggest an alternative for protecting the vulnerable WebServer02 from insider attacks for Secure Labs on Demand.

Snort needs to operate in inline mode for the "drops" features to take effect. Additionally, alerting needs to be initiated for the specified alert/drop option pair, which enables the "drops" options function successfully.

"enable_decode_drops"-This is the option if in inline mode, which alerts drop packets are on.

Challenge and Analysis

Part 1: Snort: Identifying and Suppressing False Positive Alerts

Record your reasoning that leads to the conclusion that these alerts are false positives.

After viewing the http_inspect alerts, i concluded that they were set off by a authorized HTTP traffic that doesn't present as a malicious threat. The alerts are constant with normal behavior within the environment, which is classified as false positives. By suppressing them, it will help minimize the alert fatigue and focus the attention on actual threats.

Make a screen capture showing the **suppression list** in the Snort interface.

