

# SNORT INTRUSION DETECTION SYSTEM

Syed Muneeb  
CODEALPHA

## Introduction

With an increasing number of IoT devices connecting to both private and public networks, securing sensitive data has become critical. Snort and other intrusion detection systems (IDS) are important in network security because they monitor traffic, detect suspicious activity, and generate alerts based on predetermined patterns or aberrant behavior. This lab shows how to install, configure, and test Snort's capability, allowing us to build a tailored network defense layer to protect sensitive data delivered and other endpoints.

## Objective

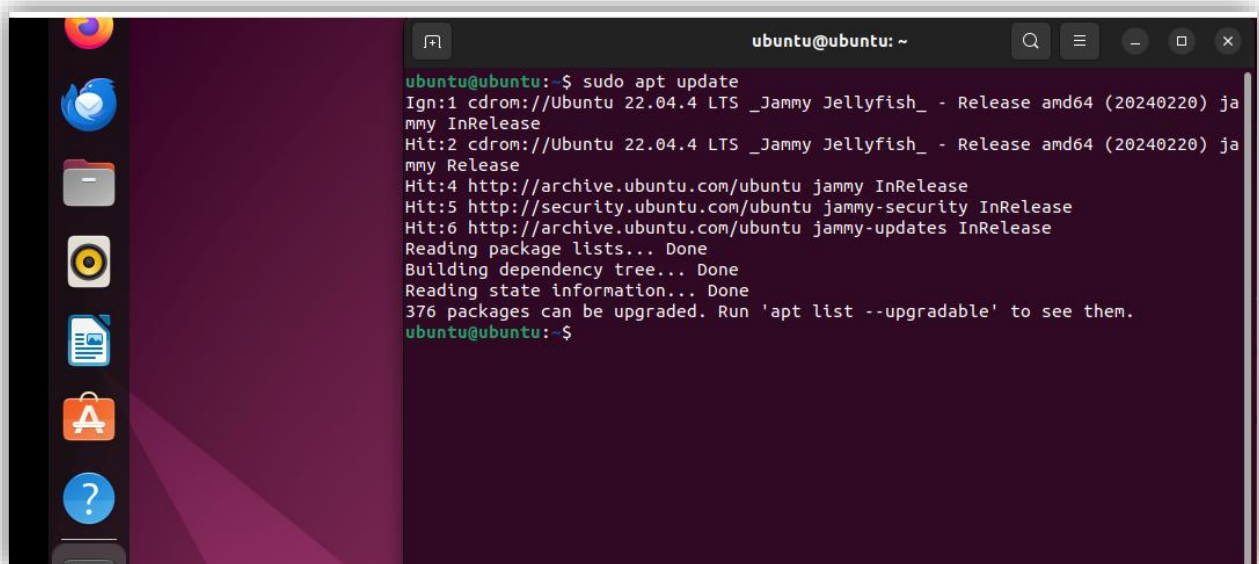
This lab is intended to provide an in-depth overview of Snort's configuration and rule-writing capabilities, covering.

## Snort installation on Ubuntu Linux.

Basic configuration and tweaking of Snort's main configuration file. Custom Snort rules can be created to detect specific network traffic, such as ICMP (ping) queries and HTTPS requests to specific URLs. Testing these configurations to watch and analyze Snort's alert creation. By the end, users will have hands-on experience using Snort to actively monitor and safeguard network traffic.

## Installing Snort on Linux (Ubuntu)

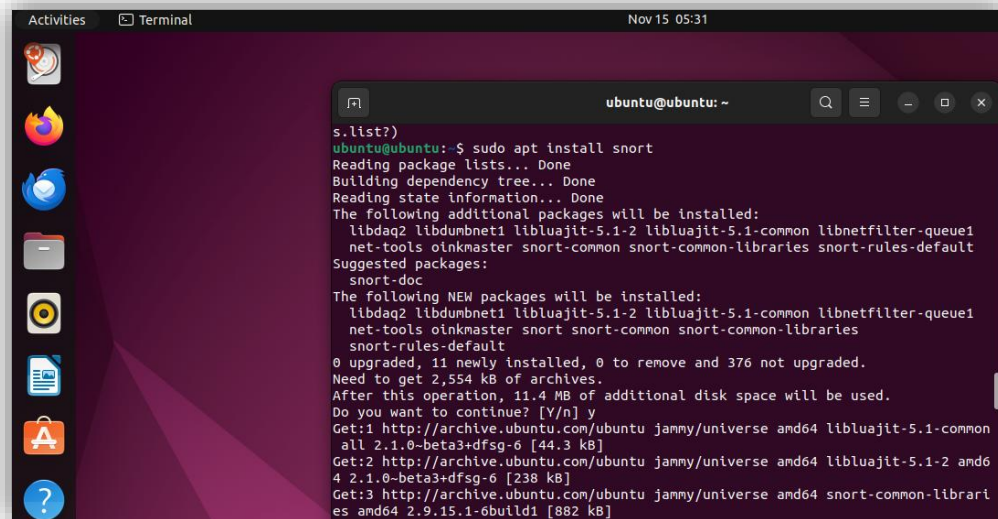
To ensure a smooth installation, first update the system package list:



```
ubuntu@ubuntu: ~  
ubuntu@ubuntu:~$ sudo apt update  
Ign:1 cdrom://Ubuntu 22.04.4 LTS _Jammy Jellyfish_ - Release amd64 (20240220) ja  
mmy InRelease  
Hit:2 cdrom://Ubuntu 22.04.4 LTS _Jammy Jellyfish_ - Release amd64 (20240220) ja  
mmy Release  
Hit:4 http://archive.ubuntu.com/ubuntu jammy InRelease  
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease  
Hit:6 http://archive.ubuntu.com/ubuntu jammy-updates InRelease  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
376 packages can be upgraded. Run 'apt list --upgradable' to see them.  
ubuntu@ubuntu:~$
```

## Explanation:

Regularly upgrading packages guarantees that all installed software, including Snort's dependencies, is up to date and free of known security issues. The -y option automatically validates the upgrade, which speeds up the process.

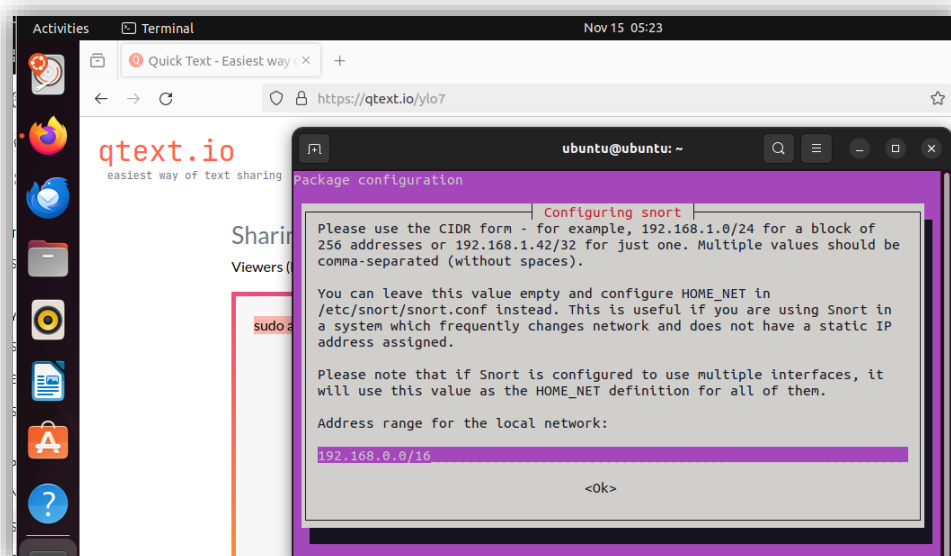


```
Activities  Terminal  Nov 15 05:31

ubuntu@ubuntu: ~
s.list?)
ubuntu@ubuntu:~$ sudo apt install snort
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libdaq2 libdumbnet1 liblua5.1-2 liblua5.1-common libnetfilter-queue1
  net-tools oinkmaster snort-common snort-common-libraries snort-rules-default
Suggested packages:
  snort-doc
The following NEW packages will be installed:
  libdaq2 libdumbnet1 liblua5.1-2 liblua5.1-common libnetfilter-queue1
  net-tools oinkmaster snort snort-common snort-common-libraries
  snort-rules-default
0 upgraded, 11 newly installed, 0 to remove and 376 not upgraded.
Need to get 2,554 kB of archives.
After this operation, 11.4 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 liblua5.1-common
all 2.1.0-beta3+dfsg-6 [44.3 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/universe amd64 liblua5.1-2 amd6
4 2.1.0-beta3+dfsg-6 [238 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/universe amd64 snort-common-librari
es amd64 2.9.15.1-6build1 [882 kB]
```

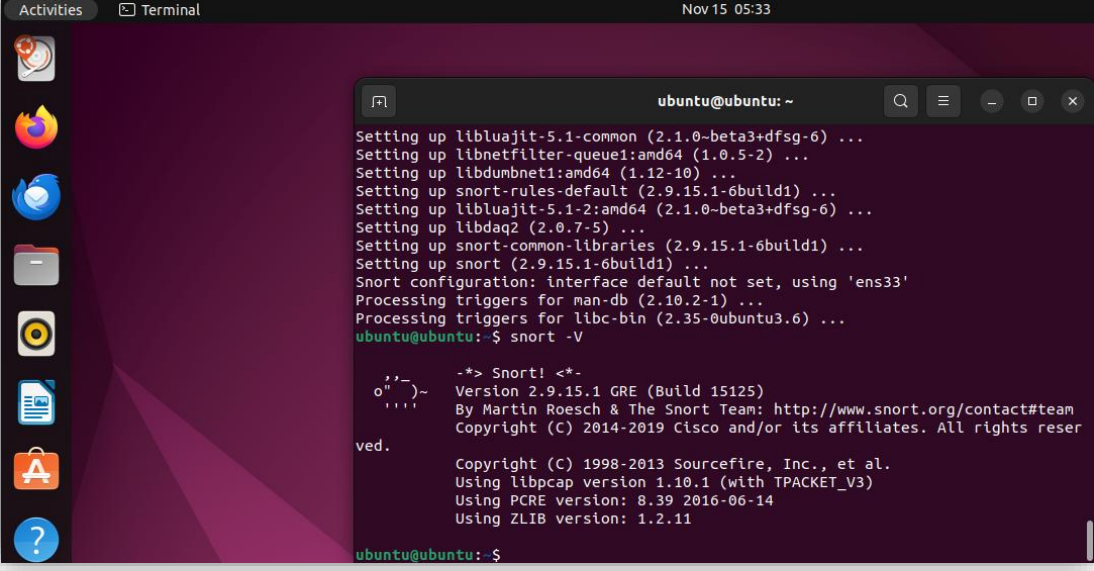
### Explanation:

This script leverages Ubuntu's Advanced Package Tool (APT) to locate and install Snort from official repositories, resulting in a clean installation with few compatibility difficulties. The -y parameter automatically accepts prompts, making the installation non-interactive and beneficial when configuring many devices.



## Verify the Installation:

Make sure Snort installed successfully by looking at the version.

A terminal window on an Ubuntu system showing the installation of Snort and its verification. The window title is "Terminal" and the date/time is "Nov 15 05:33". The terminal output shows the installation of various dependencies and Snort itself. After the installation, the user runs the command "snort -V" to check the version. The output displays the Snort version as 2.9.15.1 GRE (Build 15125) and lists the versions of the underlying libraries: libpcap 1.10.1, PCRE 8.39, and ZLIB 1.2.11.

```
Setting up liblua5.1-common (2.1.0-beta3+dfsg-6) ...
Setting up libnetfilter-queue1:amd64 (1.0.5-2) ...
Setting up libdumbnet1:amd64 (1.12-10) ...
Setting up snort-rules-default (2.9.15.1-6build1) ...
Setting up liblua5.1-2:amd64 (2.1.0-beta3+dfsg-6) ...
Setting up libdaq2 (2.0.7-5) ...
Setting up snort-common-libraries (2.9.15.1-6build1) ...
Setting up snort (2.9.15.1-6build1) ...
Snort configuration: interface default not set, using 'ens33'
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.6) ...
ubuntu@ubuntu:~$ snort -V

-*> Snort! <*-
o"')~
'''
ved.

Version 2.9.15.1 GRE (Build 15125)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.

Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.10.1 (with TPACKET_V3)
Using PCRE version: 8.39 2016-06-14
Using ZLIB version: 1.2.11

ubuntu@ubuntu:~$
```

## Justification:

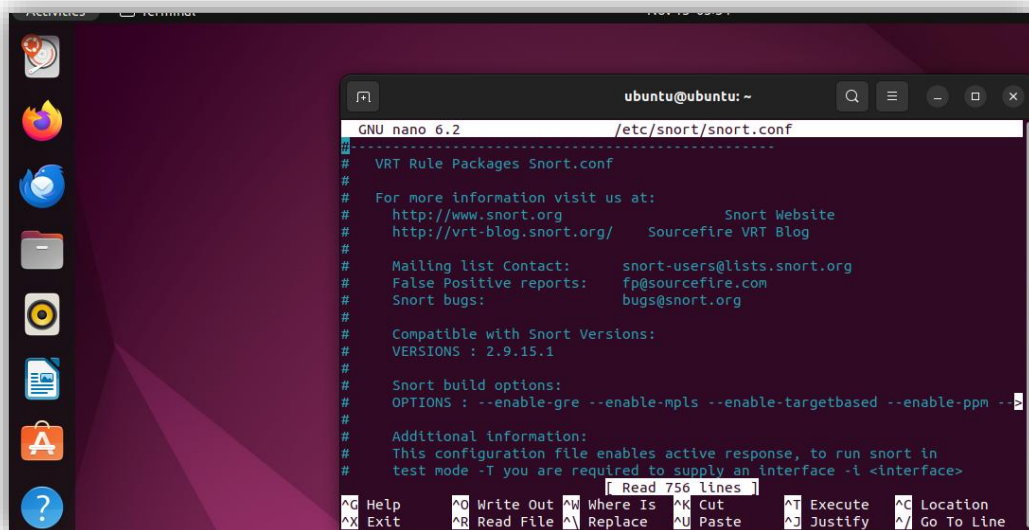
It's important to make sure the right Snort version is installed because different versions may have different features or rule syntax. Verifying the version enables us to correctly refer to Snort's documentation and steer clear of compatibility issues when setting up and creating rules.

## Configuring and Starting Snort

### Examining the Configuration File for Snort

Snort's primary configuration file, `snort.conf`, which is found at `/etc/snort/snort.conf`, is the foundation of its functionality. From creating IP variables to configuring paths for rules, log files, and plug-ins, this file manages Snort's operations.

To open `snort.conf`, use this command.



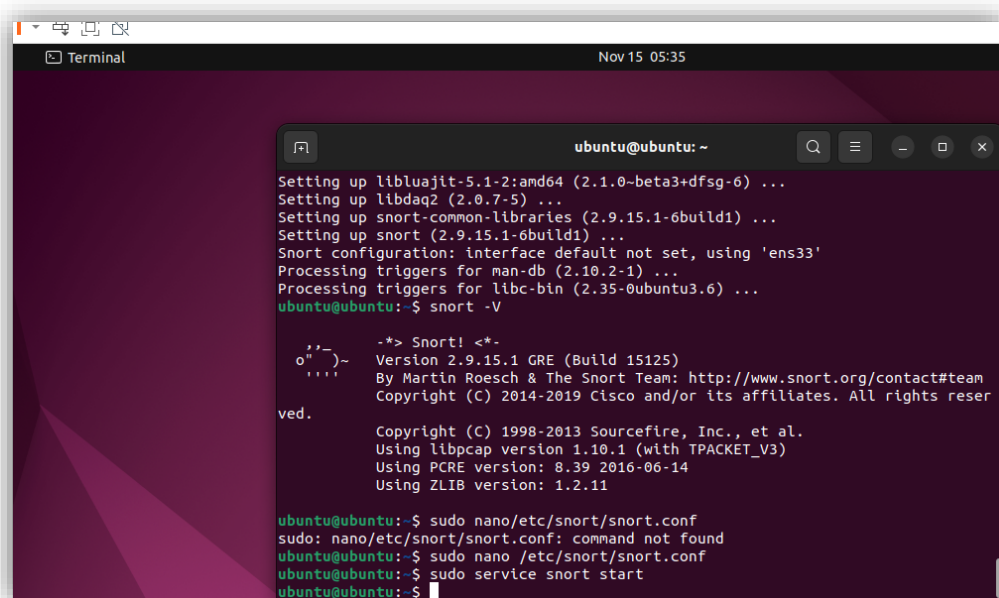
```
ubuntu@ubuntu: ~  
GNU nano 6.2 /etc/snort/snort.conf  
-----  
# VRT Rule Packages Snort.conf  
#  
# For more information visit us at:  
#   http://www.snort.org           Snort Website  
#   http://vrt-blog.snort.org/     Sourcefire VRT Blog  
#  
# Mailing list Contact:   snort-users@lists.snort.org  
# False Positive reports: fp@sourcefire.com  
# Snort bugs:            bugs@snort.org  
#  
# Compatible with Snort Versions:  
# VERSIONS : 2.9.15.1  
#  
# Snort build options:  
# OPTIONS : --enable-gre --enable-mpls --enable-targetbased --enable-ppm --enable-  
#  
# Additional information:  
# This configuration file enables active response, to run snort in  
# test mode -T you are required to supply an interface -i <interface>  
#  
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location  
^X Exit      ^R Read File ^M Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

### Explanation:

Editing the configuration file directly in the terminal is simple when using nano as a text editor. Sections specifying IP addresses, ports, and the locations of extra rule files can be found in snort.conf. By altering these sections, Snort's focus can be tailored to exclusively monitor particular network segments or types of traffic.

### Launching the Snort Program

After configuration, launch the service to activate Snort:



```
Nov 15 05:35  
ubuntu@ubuntu: ~  
Setting up libluajit-5.1-2:amd64 (2.1.0-beta3+dfsg-6) ...  
Setting up libdaq2 (2.0.7-5) ...  
Setting up snort-common-libraries (2.9.15.1-6build1) ...  
Setting up snort (2.9.15.1-6build1) ...  
Snort configuration: interface default not set, using 'ens33'  
Processing triggers for man-db (2.10.2-1) ...  
Processing triggers for libc-bin (2.35-0ubuntu3.6) ...  
ubuntu@ubuntu:~$ snort -V  
  
-*> Snort! <*-  
o''-)- Version 2.9.15.1 GRE (Build 15125)  
'''- By Martin Roesch & The Snort Team: http://www.snort.org/contact#team  
ved. Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.  
Copyright (C) 1998-2013 Sourcefire, Inc., et al.  
Using libpcap version 1.10.1 (with TPACKET_V3)  
Using PCRE version: 8.39 2016-06-14  
Using ZLIB version: 1.2.11  
  
ubuntu@ubuntu:~$ sudo nano/etc/snort/snort.conf  
sudo: nano/etc/snort/snort.conf: command not found  
ubuntu@ubuntu:~$ sudo nano /etc/snort/snort.conf  
ubuntu@ubuntu:~$ sudo service snort start  
ubuntu@ubuntu:~$
```

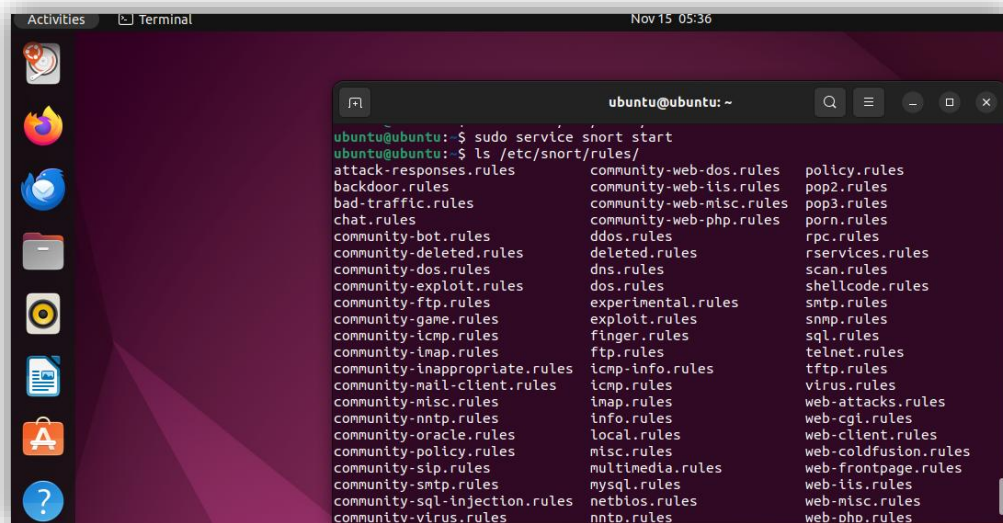
### Explanation:

Snort may continually monitor traffic in accordance with the parameters in snort.conf when it is run as a background service. Snort is perfect for continuous intrusion detection in this mode, which is necessary to detect threats in real time across all connected devices.

## Finding Snort Rules

### Examining the Predefined Rules of Snort

/etc/snort/rules/ contains Snort's rules. Enumerate every rule that is available using:



```
ubuntu@ubuntu: ~  
$ sudo service snort start  
$ ls /etc/snort/rules/  
attack-responses.rules      community-web-dos.rules    policy.rules  
backdoor.rules             community-web-iis.rules   pop2.rules  
bad-traffic.rules          community-web-misc.rules  pop3.rules  
chat.rules                 community-web-php.rules   porn.rules  
community-bot.rules        ddos.rules                rpc.rules  
community-deleted.rules    deleted.rules             rservices.rules  
community-dos.rules        dns.rules                 scan.rules  
community-exploit.rules    dos.rules                 shellcode.rules  
community-ftp.rules        experimental.rules        smtp.rules  
community-game.rules       exploit.rules             snmp.rules  
community-icmp.rules       finger.rules              sql.rules  
community-imap.rules       ftp.rules                 telnet.rules  
community-inappropriate.rules  icmp-info.rules          tftp.rules  
community-mail-client.rules  icmp.rules                virus.rules  
community-misc.rules        imap.rules                web-attacks.rules  
community-nntp.rules        info.rules                web-cgi.rules  
community-oracle.rules      local.rules               web-client.rules  
community-policy.rules      misc.rules                web-coldfusion.rules  
community-sip.rules         multimedia.rules          web-frontpage.rules  
community-smtp.rules        mysql.rules               web-iis.rules  
community-sql-injection.rules  netbios.rules            web-misc.rules  
community-virus.rules       nntp.rules                web-php.rules
```

### Justification:

Snort's pre-established rule sets target different kinds of network attacks, such as trojans, viruses, and denial-of-service attacks. These rules serve as Snort's signature-based detection system, which sends out alerts in response to recognized threat indicators or particular packet patterns. Before adding new rules, we can understand Snort's baseline capabilities by reviewing these rules, which provide insight into the types of threats Snort can detect.

### Types of Snort Rules Available

Several rule categories are included with Snort, including:

**Malware Rules:** Look for patterns in known malware communication.  
**Web Attack Rules:** Determine which HTTP requests are malicious and directed at web servers.  
**DoS Attack Guidelines:** Identify possible signs of a denial-of-service attack.  
We can choose extra rules that meet our unique network security requirements by being aware of the default rule kinds.

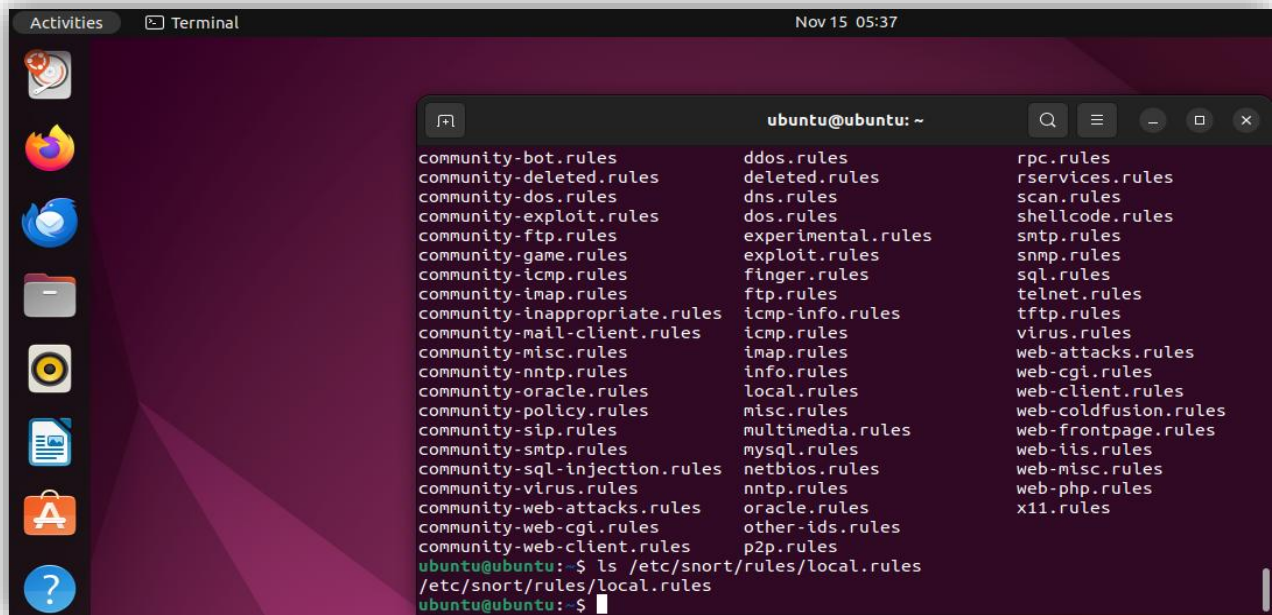


## Writing and Adding a Snort Rule

### Opening the file local.rules

User-defined rules are stored in the local.rules file. To access it, use.

Justification: Snort's modular rule structure enables the inclusion of new rules in local.rules that are distinct from the default rules. Isolating user-specific rules with local.rules facilitates maintenance and upgrades, particularly when importing new default rules from Snort updates.



The screenshot shows a terminal window titled 'Terminal' with the date 'Nov 15 05:37'. The terminal displays the contents of the file `/etc/snort/rules/local.rules`. The file lists various rule categories, including community rules, ddos, deleted, dns, dos, experimental, exploit, finger, ftp, icmp, info, local, misc, multimedia, mysql, netbios, nntp, oracle, other-ids, p2p, rpc, rservices, scan, shellcode, smtp, snmp, sql, telnet, tftp, virus, web-attacks, web-cgi, web-client, web-coldfusion, web-frontpage, web-iis, web-misc, web-php, and x11.

```
ubuntu@ubuntu: ~  
community-bot.rules      ddos.rules               rpc.rules  
community-deleted.rules  deleted.rules            rservices.rules  
community-dos.rules      dns.rules                scan.rules  
community-exploit.rules  dos.rules                shellcode.rules  
community-ftp.rules      experimental.rules       smtp.rules  
community-game.rules     exploit.rules            snmp.rules  
community-icmp.rules     finger.rules             sql.rules  
community-imap.rules     ftp.rules                telnet.rules  
community-inappropriate.rules  icmp-info.rules        tftp.rules  
community-mail-client.rules  icmp.rules              virus.rules  
community-misc.rules       imap.rules               web-attacks.rules  
community-nntp.rules      info.rules                web-cgi.rules  
community-oracle.rules    local.rules               web-client.rules  
community-policy.rules    misc.rules                web-coldfusion.rules  
community-sip.rules       multimedia.rules          web-frontpage.rules  
community-smtp.rules      mysql.rules               web-iis.rules  
community-sql-injection.rules  netbios.rules           web-misc.rules  
community-virus.rules     nntp.rules                web-php.rules  
community-web-attacks.rules  oracle.rules             x11.rules  
community-web-cgi.rules    other-ids.rules  
community-web-client.rules  p2p.rules  
ubuntu@ubuntu:~$ ls /etc/snort/rules/local.rules  
/etc/snort/rules/local.rules  
ubuntu@ubuntu:~$
```

### Developing an ICMP Detection Rule

ICMP traffic, which is frequently used for ping testing, is detected by the following rule.

```
GNU nano 6.2 /etc/snort/rules/local.rules *
#alert icmp any any->any any (msg: "ICMP Packets Founded";sid:1000001;rev:1;)
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures. Put your local
# additions here.

Cancelled

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^D Justify   ^_ Go To Line
```

## Justification:

**ICMP Alerts:** Although ICMP packets are helpful for confirming connectivity, they can also be used for reconnaissance or network mapping. When an ICMP packet (like a ping) travels across the network, this rule notifies us.

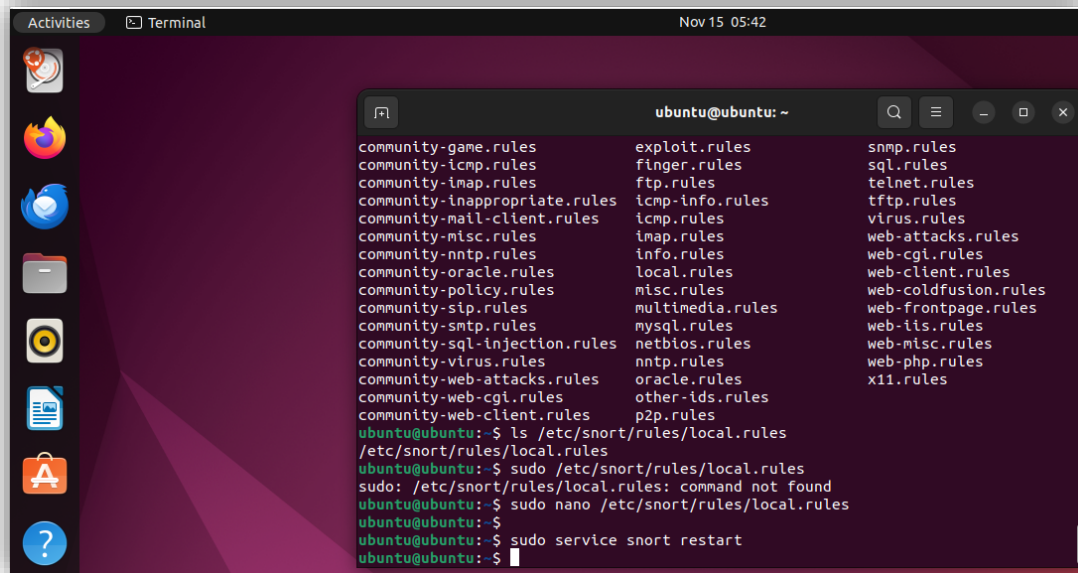
**SID and REV:** These IDs are crucial for versioning and tracking rules. Each rule has its own SID (Snort ID), and REV keeps note of changes to allow for future modifications without repeating rules.

```
GNU nano 6.2 /etc/snort/rules/local.rules *
#alert icmp any any->any any (msg: "ICMP Packets Founded";sid:1000001;rev:1;)
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures. Put your local
# additions here.

Executing...

^G Help      ^M-F New Buffer ^S Spell Check ^J Full Justify ^V Cut Till End
^C Cancel    ^M-\ Pipe Text ^Y Linter      ^O Formatter   ^Z Suspend
```





```
community-game.rules      exploit.rules              snmp.rules
community-icmp.rules      finger.rules              sql.rules
community-imap.rules      ftp.rules                 telnet.rules
community-inappropriate.rules  icmp-info.rules          tftp.rules
community-mail-client.rules  icmp.rules                virus.rules
community-misc.rules        imap.rules                web-attacks.rules
community-nntp.rules        info.rules                web-cgi.rules
community-oracle.rules      local.rules               web-client.rules
community-policy.rules      misc.rules                web-coldfusion.rules
community-sip.rules         multimedia.rules           web-frontpage.rules
community-smtp.rules        mysql.rules                web-iis.rules
community-sql-injection.rules  netbios.rules             web-misc.rules
community-virus.rules        nntp.rules                web-php.rules
community-web-attacks.rules  oracle.rules              x11.rules
community-web-cgi.rules     other-ids.rules
community-web-client.rules  p2p.rules
ubuntu@ubuntu:~$ ls /etc/snort/rules/local.rules
/etc/snort/rules/local.rules
ubuntu@ubuntu:~$ sudo /etc/snort/rules/local.rules
sudo: /etc/snort/rules/local.rules: command not found
ubuntu@ubuntu:~$ sudo nano /etc/snort/rules/local.rules
ubuntu@ubuntu:~$
ubuntu@ubuntu:~$ sudo service snort restart
ubuntu@ubuntu:~$
```

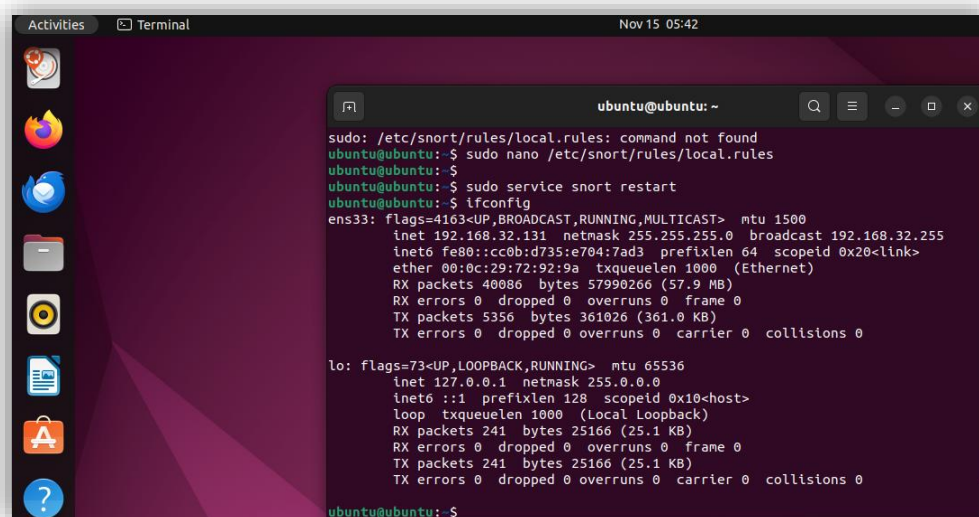
### Reason:

By restarting, you may make that Snort reads and uses the most recent version of the local.rules file. Snort will continue to utilize the prior rule set without restarting, so any modifications made to local.rules will be lost.

## Testing the ICMP Detection Rule

### Discovering the Snort System's IP Address

The Snort system's IP address can be found using the ifconfig command.



```
ubuntu@ubuntu:~$ sudo nano /etc/snort/rules/local.rules
ubuntu@ubuntu:~$ sudo service snort restart
ubuntu@ubuntu:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.32.131 netmask 255.255.255.0 broadcast 192.168.32.255
    inet6 fe80::cc0b:d735:e704:7ad3 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:72:92:9a txqueuelen 1000 (Ethernet)
    RX packets 40086 bytes 57990266 (57.9 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5356 bytes 361026 (361.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 241 bytes 25166 (25.1 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 241 bytes 25166 (25.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

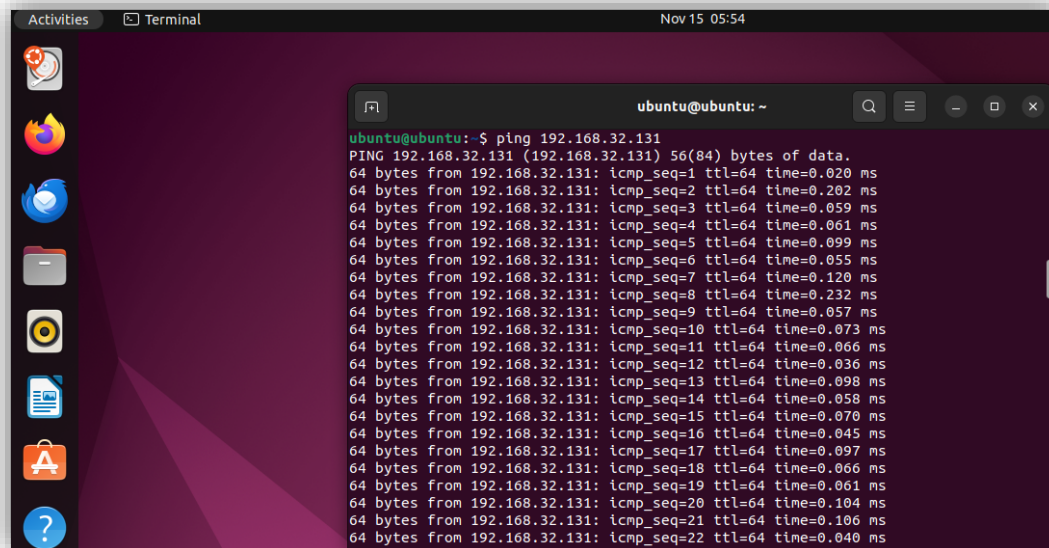
ubuntu@ubuntu:~$
```

## Explanation:

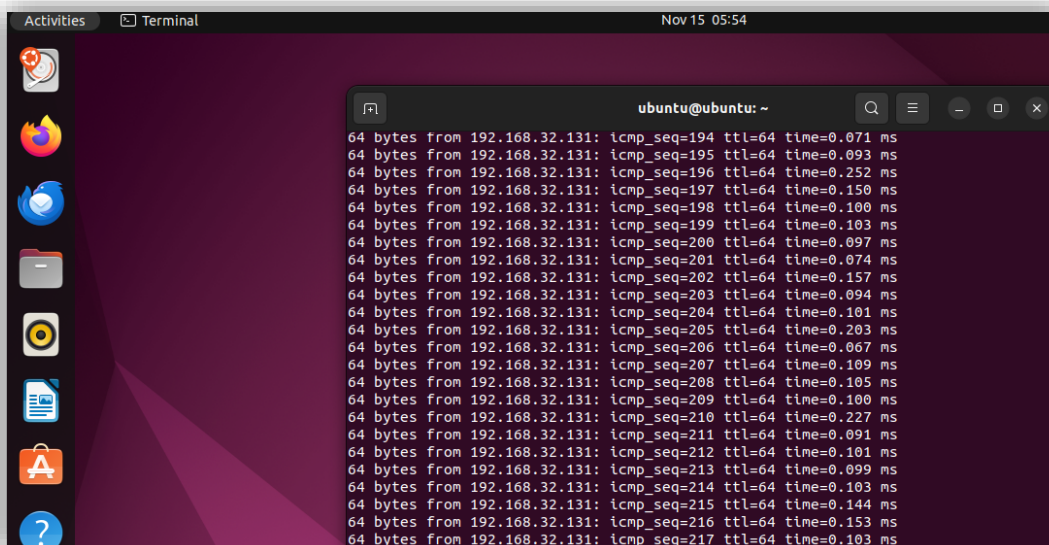
By providing traffic to a certain interface, we may test rules by knowing the Snort system's IP address. By restricting analysis to important traffic flows, this phase helps network administrators to choose precise segments for monitoring in production, which lowers resource consumption.

## Transmitting ICMP Data

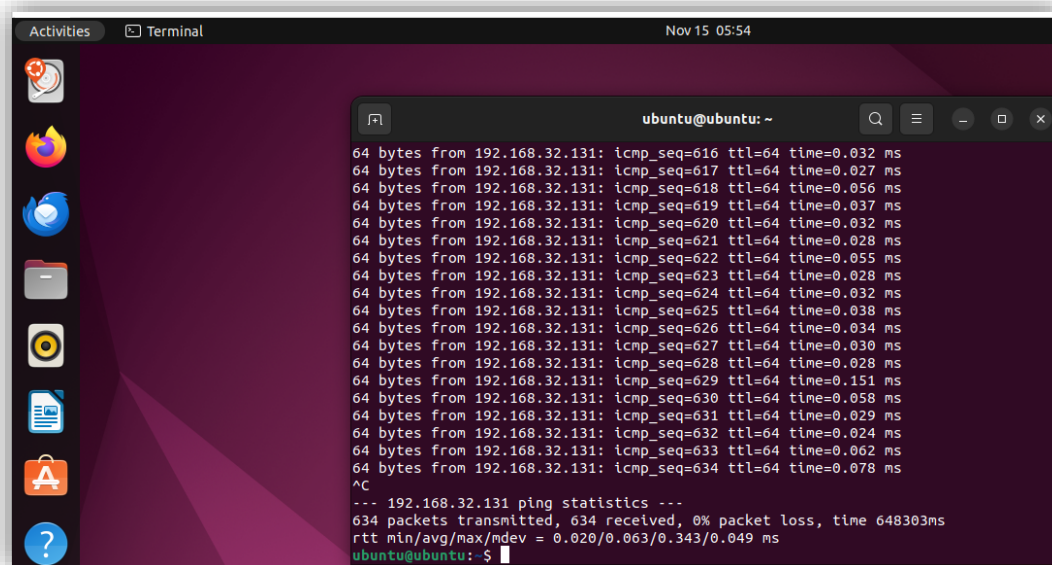
Ping Snort from a different computer to test the ICMP rule

A terminal window titled 'Terminal' with a date and time of 'Nov 15 05:54'. The prompt is 'ubuntu@ubuntu: ~'. The command 'ping 192.168.32.131' has been executed. The output shows 22 successful ping responses, each with 64 bytes of data, a TTL of 64, and a time ranging from 0.020 ms to 0.232 ms.

```
ubuntu@ubuntu: ~  
$ ping 192.168.32.131  
PING 192.168.32.131 (192.168.32.131) 56(84) bytes of data:  
64 bytes from 192.168.32.131: icmp_seq=1 ttl=64 time=0.020 ms  
64 bytes from 192.168.32.131: icmp_seq=2 ttl=64 time=0.202 ms  
64 bytes from 192.168.32.131: icmp_seq=3 ttl=64 time=0.059 ms  
64 bytes from 192.168.32.131: icmp_seq=4 ttl=64 time=0.061 ms  
64 bytes from 192.168.32.131: icmp_seq=5 ttl=64 time=0.099 ms  
64 bytes from 192.168.32.131: icmp_seq=6 ttl=64 time=0.055 ms  
64 bytes from 192.168.32.131: icmp_seq=7 ttl=64 time=0.120 ms  
64 bytes from 192.168.32.131: icmp_seq=8 ttl=64 time=0.232 ms  
64 bytes from 192.168.32.131: icmp_seq=9 ttl=64 time=0.057 ms  
64 bytes from 192.168.32.131: icmp_seq=10 ttl=64 time=0.073 ms  
64 bytes from 192.168.32.131: icmp_seq=11 ttl=64 time=0.066 ms  
64 bytes from 192.168.32.131: icmp_seq=12 ttl=64 time=0.036 ms  
64 bytes from 192.168.32.131: icmp_seq=13 ttl=64 time=0.098 ms  
64 bytes from 192.168.32.131: icmp_seq=14 ttl=64 time=0.058 ms  
64 bytes from 192.168.32.131: icmp_seq=15 ttl=64 time=0.070 ms  
64 bytes from 192.168.32.131: icmp_seq=16 ttl=64 time=0.045 ms  
64 bytes from 192.168.32.131: icmp_seq=17 ttl=64 time=0.097 ms  
64 bytes from 192.168.32.131: icmp_seq=18 ttl=64 time=0.066 ms  
64 bytes from 192.168.32.131: icmp_seq=19 ttl=64 time=0.061 ms  
64 bytes from 192.168.32.131: icmp_seq=20 ttl=64 time=0.104 ms  
64 bytes from 192.168.32.131: icmp_seq=21 ttl=64 time=0.106 ms  
64 bytes from 192.168.32.131: icmp_seq=22 ttl=64 time=0.040 ms
```

A terminal window titled 'Terminal' with a date and time of 'Nov 15 05:54'. The prompt is 'ubuntu@ubuntu: ~'. The command 'ping 192.168.32.131' has been executed. The output shows 24 successful ping responses, each with 64 bytes of data, a TTL of 64, and a time ranging from 0.071 ms to 0.252 ms.

```
ubuntu@ubuntu: ~  
$ ping 192.168.32.131  
64 bytes from 192.168.32.131: icmp_seq=194 ttl=64 time=0.071 ms  
64 bytes from 192.168.32.131: icmp_seq=195 ttl=64 time=0.093 ms  
64 bytes from 192.168.32.131: icmp_seq=196 ttl=64 time=0.252 ms  
64 bytes from 192.168.32.131: icmp_seq=197 ttl=64 time=0.150 ms  
64 bytes from 192.168.32.131: icmp_seq=198 ttl=64 time=0.100 ms  
64 bytes from 192.168.32.131: icmp_seq=199 ttl=64 time=0.103 ms  
64 bytes from 192.168.32.131: icmp_seq=200 ttl=64 time=0.097 ms  
64 bytes from 192.168.32.131: icmp_seq=201 ttl=64 time=0.074 ms  
64 bytes from 192.168.32.131: icmp_seq=202 ttl=64 time=0.157 ms  
64 bytes from 192.168.32.131: icmp_seq=203 ttl=64 time=0.094 ms  
64 bytes from 192.168.32.131: icmp_seq=204 ttl=64 time=0.101 ms  
64 bytes from 192.168.32.131: icmp_seq=205 ttl=64 time=0.203 ms  
64 bytes from 192.168.32.131: icmp_seq=206 ttl=64 time=0.067 ms  
64 bytes from 192.168.32.131: icmp_seq=207 ttl=64 time=0.109 ms  
64 bytes from 192.168.32.131: icmp_seq=208 ttl=64 time=0.105 ms  
64 bytes from 192.168.32.131: icmp_seq=209 ttl=64 time=0.100 ms  
64 bytes from 192.168.32.131: icmp_seq=210 ttl=64 time=0.227 ms  
64 bytes from 192.168.32.131: icmp_seq=211 ttl=64 time=0.091 ms  
64 bytes from 192.168.32.131: icmp_seq=212 ttl=64 time=0.101 ms  
64 bytes from 192.168.32.131: icmp_seq=213 ttl=64 time=0.099 ms  
64 bytes from 192.168.32.131: icmp_seq=214 ttl=64 time=0.103 ms  
64 bytes from 192.168.32.131: icmp_seq=215 ttl=64 time=0.144 ms  
64 bytes from 192.168.32.131: icmp_seq=216 ttl=64 time=0.153 ms  
64 bytes from 192.168.32.131: icmp_seq=217 ttl=64 time=0.103 ms
```



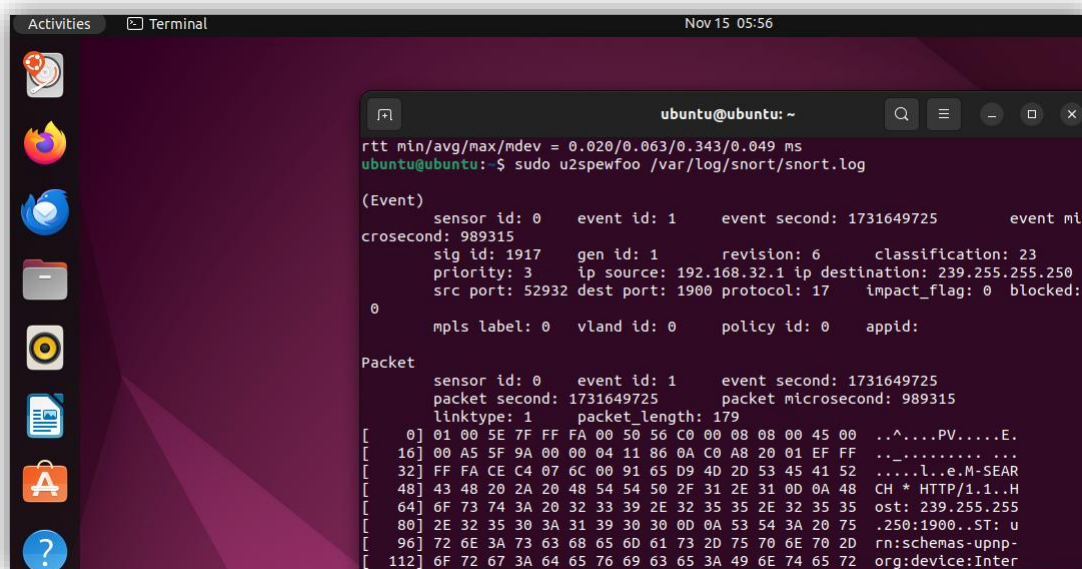
```
64 bytes from 192.168.32.131: icmp_seq=616 ttl=64 time=0.032 ms
64 bytes from 192.168.32.131: icmp_seq=617 ttl=64 time=0.027 ms
64 bytes from 192.168.32.131: icmp_seq=618 ttl=64 time=0.056 ms
64 bytes from 192.168.32.131: icmp_seq=619 ttl=64 time=0.037 ms
64 bytes from 192.168.32.131: icmp_seq=620 ttl=64 time=0.032 ms
64 bytes from 192.168.32.131: icmp_seq=621 ttl=64 time=0.028 ms
64 bytes from 192.168.32.131: icmp_seq=622 ttl=64 time=0.055 ms
64 bytes from 192.168.32.131: icmp_seq=623 ttl=64 time=0.028 ms
64 bytes from 192.168.32.131: icmp_seq=624 ttl=64 time=0.032 ms
64 bytes from 192.168.32.131: icmp_seq=625 ttl=64 time=0.038 ms
64 bytes from 192.168.32.131: icmp_seq=626 ttl=64 time=0.034 ms
64 bytes from 192.168.32.131: icmp_seq=627 ttl=64 time=0.030 ms
64 bytes from 192.168.32.131: icmp_seq=628 ttl=64 time=0.028 ms
64 bytes from 192.168.32.131: icmp_seq=629 ttl=64 time=0.151 ms
64 bytes from 192.168.32.131: icmp_seq=630 ttl=64 time=0.058 ms
64 bytes from 192.168.32.131: icmp_seq=631 ttl=64 time=0.029 ms
64 bytes from 192.168.32.131: icmp_seq=632 ttl=64 time=0.024 ms
64 bytes from 192.168.32.131: icmp_seq=633 ttl=64 time=0.062 ms
64 bytes from 192.168.32.131: icmp_seq=634 ttl=64 time=0.078 ms
^C
--- 192.168.32.131 ping statistics ---
634 packets transmitted, 634 received, 0% packet loss, time 648303ms
rtt min/avg/max/mdev = 0.020/0.063/0.343/0.049 ms
ubuntu@ubuntu:~$
```

## Justification:

Snort should record an alert if this ICMP traffic matches our detection rule. Although ICMP requests are frequently innocuous, they can occasionally be used by attackers as reconnaissance tools.

## Examining Alert Records

Use u2spewfoo to view Snort's alarm logs.



```
rtt min/avg/max/mdev = 0.020/0.063/0.343/0.049 ms
ubuntu@ubuntu:~$ sudo u2spewfoo /var/log/snort/snort.log

(Event)
  sensor id: 0      event id: 1      event second: 1731649725      event mi
crosecond: 989315
  sig id: 1917      gen id: 1      revision: 6      classification: 23
  priority: 3      ip source: 192.168.32.1 ip destination: 239.255.255.250
  src port: 52932   dest port: 1900 protocol: 17      impact_flag: 0 blocked:
0
  mpls label: 0     vland id: 0     policy id: 0     appid:

Packet
  sensor id: 0      event id: 1      event second: 1731649725
  packet second: 1731649725      packet microsecond: 989315
  linktype: 1      packet_length: 179
[ 0] 01 00 5E 7F FF FA 00 50 56 C0 00 08 08 00 45 00 ..^....PV.....E.
[ 16] 00 A5 5F 9A 00 00 04 11 86 0A C0 A8 20 01 EF FF .....
[ 32] FF FA CE C4 07 6C 00 91 65 D9 4D 2D 53 45 41 52 .....l..e.M-SEAR
[ 48] 43 48 20 2A 20 48 54 54 50 2F 31 2E 31 0D 0A 48 CH * HTTP/1.1..H
[ 64] 6F 73 74 3A 20 32 33 39 2E 32 35 35 2E 32 35 35 ost: 239.255.255
[ 80] 2E 32 35 30 3A 31 39 30 30 0D 0A 53 54 3A 20 75 .250:1900..ST: u
[ 96] 72 6E 3A 73 63 68 65 6D 61 73 2D 75 70 6E 70 2D rn:schemas-upnp-
[ 112] 6F 72 67 3A 64 65 76 69 63 65 3A 49 6E 74 65 72 org:device:Inter
```

```
Activities Terminal Nov 15 05:56

[ 128] 6E 65 74 47 61 74 65 77 61 79 44 65 76 69 63 65 netGatewayDevice
[ 144] 3A 31 0D 0A 4D 61 6E 3A 20 22 73 73 64 70 3A 64 :1..Man: "ssdp:d
[ 160] 69 73 63 6F 76 65 72 22 0D 0A 4D 58 3A 20 33 0D iscover"..MX: 3.
[ 176] 0A 0D 0A ...

(Event)
sensor id: 0 event id: 2 event second: 1731649728 event mi
crosecond: 996685
sig id: 1917 gen id: 1 revision: 6 classification: 23
priority: 3 ip source: 192.168.32.1 ip destination: 239.255.255.250
src port: 52932 dest port: 1900 protocol: 17 impact_flag: 0 blocked:
0
mpls label: 0 vland id: 0 policy id: 0 appid:

Packet
sensor id: 0 event id: 2 event second: 1731649728
packet second: 1731649728 packet microsecond: 996685
linktype: 1 packet_length: 179
[ 0] 01 00 5E 7F FF FA 00 50 56 C0 00 08 08 00 45 00 ..^....PV....E.
[ 16] 00 A5 5F 9B 00 00 04 11 86 09 C0 A8 20 01 EF FF .....
[ 32] FF FA CE C4 07 6C 00 91 65 D9 4D 2D 53 45 41 52 .....l..e.M-SEAR
[ 48] 43 48 20 2A 20 48 54 54 50 2F 31 2E 31 0D 0A 48 CH * HTTP/1.1..H
[ 64] 6F 73 74 3A 20 32 33 39 2E 32 35 35 2E 32 35 35 ost: 239.255.255
[ 80] 2E 32 35 30 3A 31 39 30 30 0D 0A 53 54 3A 20 75 .250:1900..ST: u
```

```
Activities Terminal Nov 15 05:56

crosecond: 11715
sig id: 1917 gen id: 1 revision: 6 classification: 23
priority: 3 ip source: 192.168.32.1 ip destination: 239.255.255.250
src port: 52932 dest port: 1900 protocol: 17 impact_flag: 0 blocked:
0
mpls label: 0 vland id: 0 policy id: 0 appid:

Packet
sensor id: 0 event id: 3 event second: 1731649732
packet second: 1731649732 packet microsecond: 11715
linktype: 1 packet_length: 179
[ 0] 01 00 5E 7F FF FA 00 50 56 C0 00 08 08 00 45 00 ..^....PV....E.
[ 16] 00 A5 5F 9C 00 00 04 11 86 08 C0 A8 20 01 EF FF .....
[ 32] FF FA CE C4 07 6C 00 91 65 D9 4D 2D 53 45 41 52 .....l..e.M-SEAR
[ 48] 43 48 20 2A 20 48 54 54 50 2F 31 2E 31 0D 0A 48 CH * HTTP/1.1..H
[ 64] 6F 73 74 3A 20 32 33 39 2E 32 35 35 2E 32 35 35 ost: 239.255.255
[ 80] 2E 32 35 30 3A 31 39 30 30 0D 0A 53 54 3A 20 75 .250:1900..ST: u
[ 96] 72 6E 3A 73 63 68 65 6D 61 73 2D 75 70 6E 70 2D rn:schemas-upnp-
[ 112] 6F 72 67 3A 64 65 76 69 63 65 3A 49 6E 74 65 72 org:device:Inter
[ 128] 6E 65 74 47 61 74 65 77 61 79 44 65 76 69 63 65 netGatewayDevice
[ 144] 3A 31 0D 0A 4D 61 6E 3A 20 22 73 73 64 70 3A 64 :1..Man: "ssdp:d
[ 160] 69 73 63 6F 76 65 72 22 0D 0A 4D 58 3A 20 33 0D iscover"..MX: 3.
[ 176] 0A 0D 0A ...

ubuntu@ubuntu:~$
```

### Explanation:

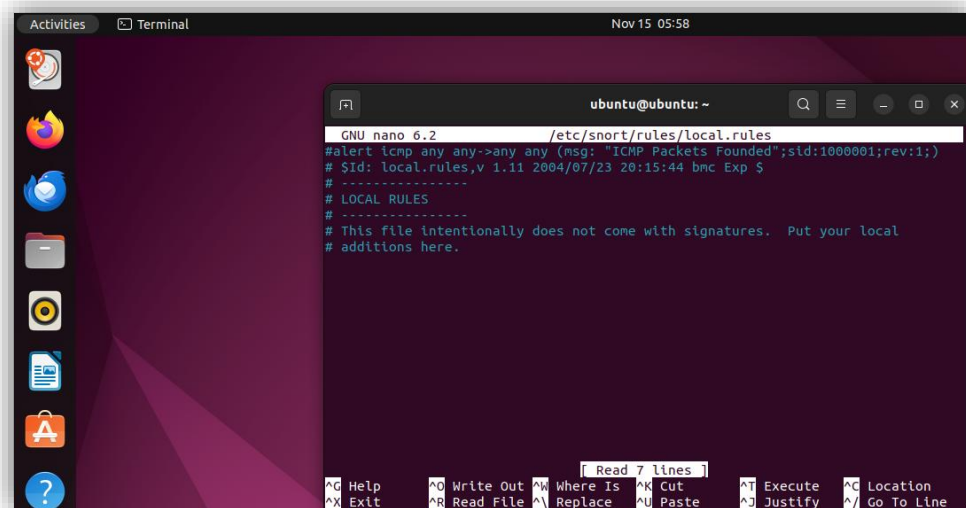
The tool u2spewfoo is used to decode the binary log format of Snort. By checking the alert log, you can verify whether our custom ICMP rule was activated and that Snort is actively keeping an eye out for unwanted traffic in accordance with our configurations.

## Writing a Rule to Trigger an Alert for HTTPS Traffic to a Specific URL

### Including a Rule for HTTPS

Add the following rule to local.rules to track HTTPS requests to a particular website.

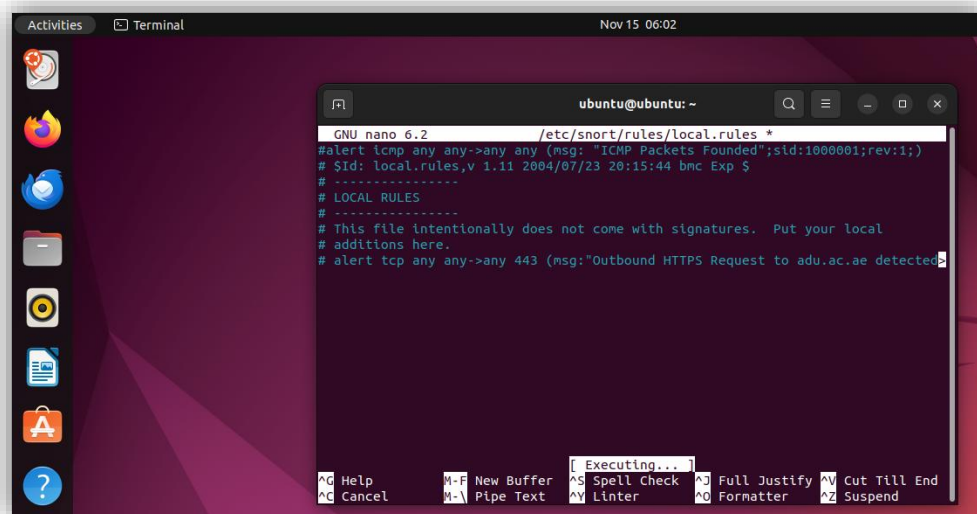




```
GNU nano 6.2 /etc/snort/rules/local.rules
#alert icmp any any->any any (msg: "ICMP Packets Founded";sid:1000001;rev:1;)
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures.  Put your local
# additions here.
```

### Justification:

Content Correspondence: Within HTTPS traffic, content: "GET / HTTP/1.1"; content: "Host: www.adu.ac.ae"; targets HTTP headers. Snort can examine packet payloads thanks to content-based rules, which is helpful for identifying certain application-layer requests. Details of the Port (443): We concentrate on encrypted HTTPS communication by restricting detection to port 443. Security teams can spot possible exfiltration or unusual connections concealed by encryption by using HTTPS monitoring. Restart Snort to Activate the New Rule:

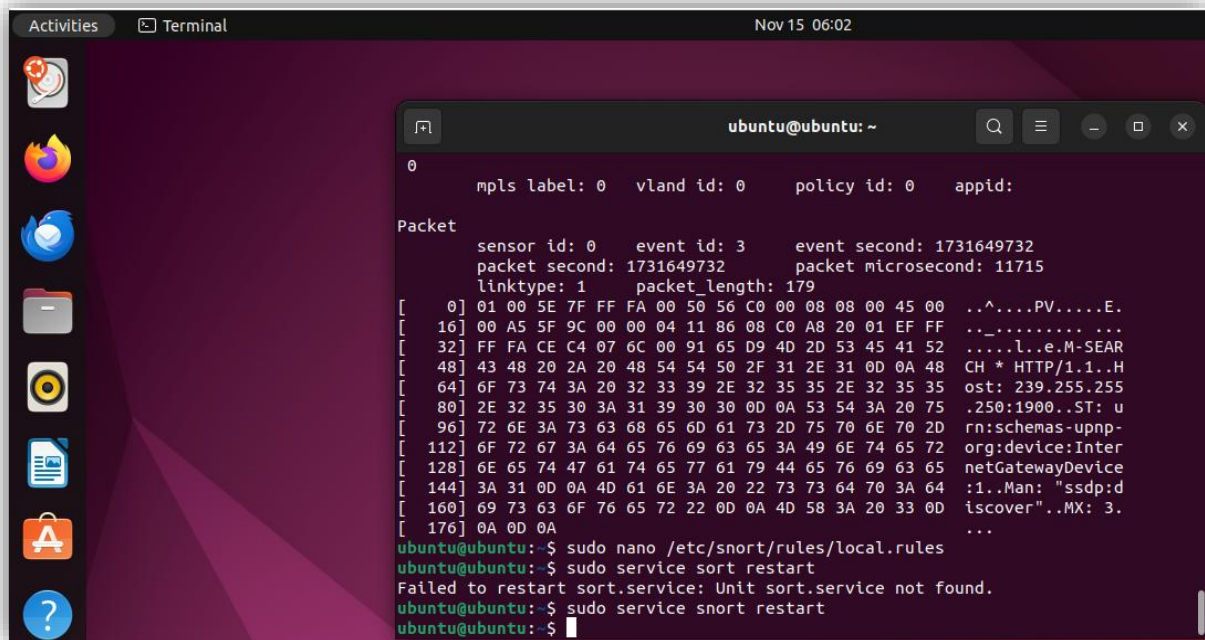


```
GNU nano 6.2 /etc/snort/rules/local.rules *
#alert icmp any any->any any (msg: "ICMP Packets Founded";sid:1000001;rev:1;)
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures.  Put your local
# additions here.
# alert tcp any any->any 443 (msg:"Outbound HTTPS Request to adu.ac.ae detected";
```

### Turn on the HTTPS Rule and restart Snort

After restarting Snort, go to <https://www.adu.ac.ae> to activate the rule. Explanation: Network traffic with the appropriate headers is produced when the given URL is browsed. This rule can

serve as an early alert for hacked computers trying to visit dubious domains by detecting anomalous HTTPS traffic.

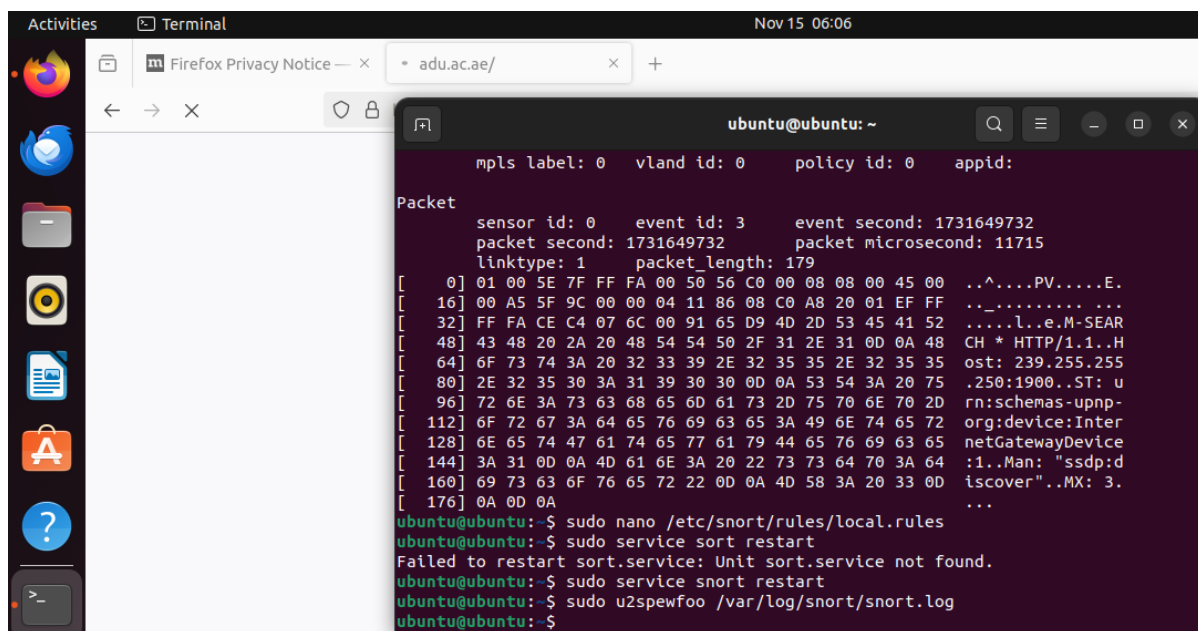


```
Activities Terminal Nov 15 06:02

0      mpls label: 0      vland id: 0      policy id: 0      appid:

Packet
  sensor id: 0      event id: 3      event second: 1731649732
  packet second: 1731649732      packet microsecond: 11715
  linktype: 1      packet_length: 179
[ 0] 01 00 5E 7F FF FA 00 50 56 C0 00 08 08 00 45 00 ..^....PV....E.
[ 16] 00 A5 5F 9C 00 00 04 11 86 08 C0 A8 20 01 EF FF .....
[ 32] FF FA CE C4 07 6C 00 91 65 D9 4D 2D 53 45 41 52 .....l..e.M-SEAR
[ 48] 43 48 20 2A 20 48 54 54 50 2F 31 2E 31 0D 0A 48 CH * HTTP/1.1..H
[ 64] 6F 73 74 3A 20 32 33 39 2E 32 35 35 2E 32 35 35 ost: 239.255.255
[ 80] 2E 32 35 30 3A 31 39 30 30 0D 0A 53 54 3A 20 75 .250:1900..ST: u
[ 96] 72 6E 3A 73 63 68 65 6D 61 73 2D 75 70 6E 70 2D rn:schemas-upnp-
[ 112] 6F 72 67 3A 64 65 76 69 63 65 3A 49 6E 74 65 72 org:device:Inter
[ 128] 6E 65 74 47 61 74 65 77 61 79 44 65 76 69 63 65 netGatewayDevice
[ 144] 3A 31 0D 0A 4D 61 6E 3A 20 22 73 73 64 70 3A 64 :1..Man: "ssdp:d
[ 160] 69 73 63 6F 76 65 72 22 0D 0A 4D 58 3A 20 33 0D iscover"..MX: 3.
[ 176] 0A 0D 0A ...

ubuntu@ubuntu:~$ sudo nano /etc/snort/rules/local.rules
ubuntu@ubuntu:~$ sudo service sort restart
Failed to restart sort.service: Unit sort.service not found.
ubuntu@ubuntu:~$ sudo service snort restart
ubuntu@ubuntu:~$
```



```
Activities Terminal Nov 15 06:06
Firefox Privacy Notice — x adu.ac.ae/ x +

← → X

0      mpls label: 0      vland id: 0      policy id: 0      appid:

Packet
  sensor id: 0      event id: 3      event second: 1731649732
  packet second: 1731649732      packet microsecond: 11715
  linktype: 1      packet_length: 179
[ 0] 01 00 5E 7F FF FA 00 50 56 C0 00 08 08 00 45 00 ..^....PV....E.
[ 16] 00 A5 5F 9C 00 00 04 11 86 08 C0 A8 20 01 EF FF .....
[ 32] FF FA CE C4 07 6C 00 91 65 D9 4D 2D 53 45 41 52 .....l..e.M-SEAR
[ 48] 43 48 20 2A 20 48 54 54 50 2F 31 2E 31 0D 0A 48 CH * HTTP/1.1..H
[ 64] 6F 73 74 3A 20 32 33 39 2E 32 35 35 2E 32 35 35 ost: 239.255.255
[ 80] 2E 32 35 30 3A 31 39 30 30 0D 0A 53 54 3A 20 75 .250:1900..ST: u
[ 96] 72 6E 3A 73 63 68 65 6D 61 73 2D 75 70 6E 70 2D rn:schemas-upnp-
[ 112] 6F 72 67 3A 64 65 76 69 63 65 3A 49 6E 74 65 72 org:device:Inter
[ 128] 6E 65 74 47 61 74 65 77 61 79 44 65 76 69 63 65 netGatewayDevice
[ 144] 3A 31 0D 0A 4D 61 6E 3A 20 22 73 73 64 70 3A 64 :1..Man: "ssdp:d
[ 160] 69 73 63 6F 76 65 72 22 0D 0A 4D 58 3A 20 33 0D iscover"..MX: 3.
[ 176] 0A 0D 0A ...

ubuntu@ubuntu:~$ sudo nano /etc/snort/rules/local.rules
ubuntu@ubuntu:~$ sudo service sort restart
Failed to restart sort.service: Unit sort.service not found.
ubuntu@ubuntu:~$ sudo service snort restart
ubuntu@ubuntu:~$ sudo u2spewfoo /var/log/snort/snort.log
ubuntu@ubuntu:~$
```

## conclusion

In the quickly changing world of technology, the rise in the use of mobile devices has created both enormous security challenges and previously unheard-of convenience. Strong security measures like Intrusion Detection Systems (IDS) must be put in place because of the increased risk of bad



actors taking advantage of these devices as they link to both public and private networks. Strong open-source intrusion detection system (IDS) Snort has established itself as an essential tool for protecting networks against a wide range of attacks. This lab report demonstrated Snort's usefulness in improving network security by thoroughly examining its installation, configuration, and rule modification.

Snort, which provides real-time traffic analysis and threat detection capabilities, is an essential tool for contemporary network security. We investigated its installation, configuration, and customization in this lab, showcasing its capacity to identify and notify users of a range of network activity. We demonstrated Snort's adaptability in monitoring particular traffic patterns, such as ICMP packets and HTTPS requests, by customizing its rules and parameters. These practical experiments demonstrated how Snort evolves from a general-purpose intrusion detection system to a specialized defense mechanism that can handle particular security threats.

The useful knowledge acquired from this lab highlights Snort's function in protecting a variety of settings, such as enterprise and mobile networks. We were able to identify particular dangers, like illegal HTTPS connections or reconnaissance efforts, thanks to custom rules, and recording methods gave us useful information for threat research. Snort continues to be an essential part of layered security solutions, guaranteeing strong defense against changing cyberthreats, despite issues like false positives and resource limitations. This exercise gives us useful abilities for practical security applications and emphasizes the significance of proactive network monitoring.