Name: Pankaj Parihar

Roll No.: 74

Batch: T21

Assignment – 11

Aim : To install snort, configuring it in Intrusion Detection mode and writing rules for detecting pinging activity.

Theory :

1. Installing Snort

- **Installation**: Snort is available for both Linux and Windows. The installation involves downloading the Snort package from its official source and following the setup process. During installation, you specify the network interface that Snort will monitor.

2. Adding Rules

- **Rules**: Snort uses predefined rules to detect specific types of network activity that could indicate malicious behaviour. These rules define patterns, actions to take (such as logging or alerting), and the traffic to inspect. Users can create custom rules or use community-contributed rule sets.
- **Structure**: A Snort rule consists of an action (alert, log, etc.), protocol, source/destination IP addresses, ports, and specific options that define the detection logic.

3. Configuring Snort

- **Configuration File**: The main Snort configuration file specifies the network variables, rule paths, and preprocessors (used for advanced traffic detection). It also defines how Snort handles and logs alerts and what traffic patterns to monitor (such as internal vs. external networks).
- **Preprocessors**: These are modular add-ons that extend Snort's capabilities, enabling it to detect various network anomalies, such as port scanning or fragmented packets.

4. Validating Configuration

- **Validation**: Before running Snort, it is important to validate the configuration to ensure that there are no syntax errors or misconfigurations. This process checks the integrity of the configuration file and ensures all rules and preprocessors are correctly set up.

5. Monitoring for Intrusions

- **Running Snort in IDS Mode**: Once Snort is configured, it can be run in intrusion detection mode. In this mode, Snort monitors network traffic in real-time and checks for matches against the active rule sets. When malicious traffic is detected, Snort generates alerts.
- **Alerting and Logging**: Snort can be configured to log alerts in various formats, such as text files or centralized logging systems. Alerts can be displayed on the console or sent to external logging services for further analysis.

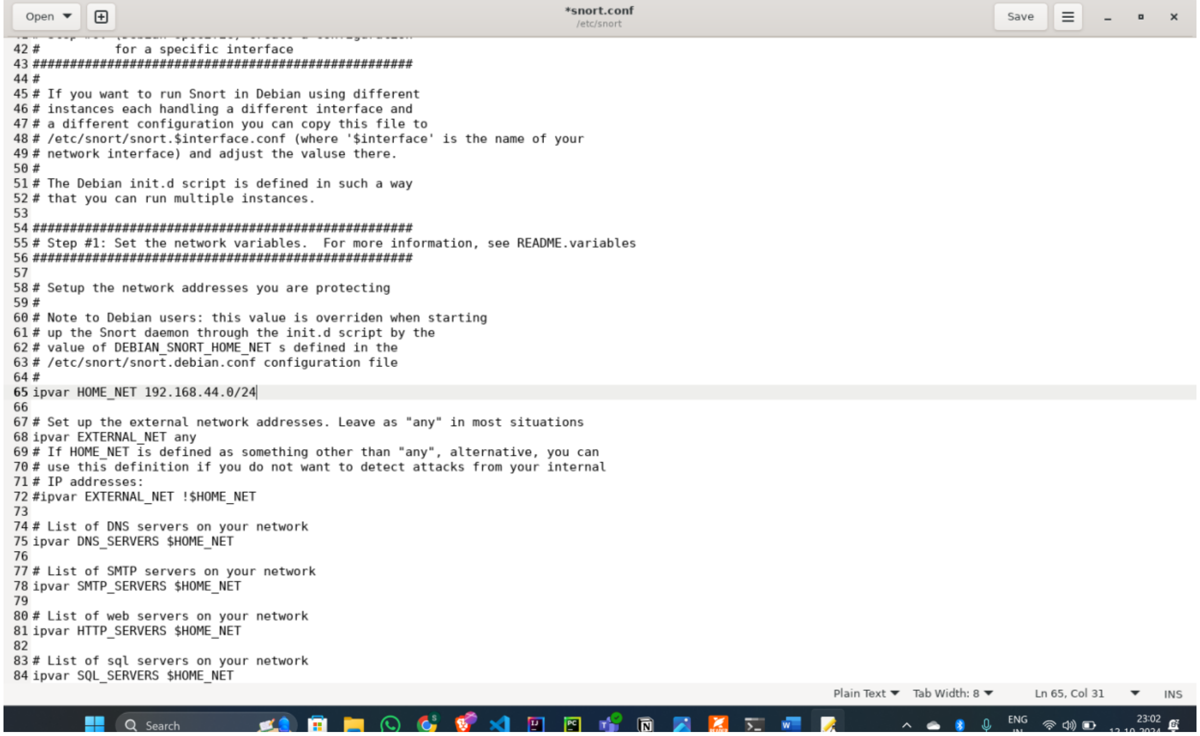
6. Monitoring and Analyzing Logs

- **Log Review**: Regular log monitoring is crucial for intrusion detection. Administrators can analyze logs manually or use web-based interfaces to visualize and manage alerts more effectively.
- **Integration with Tools**: For more efficient monitoring, Snort can be integrated with visualization and reporting tools like Snorby or BASE, which provide a graphical interface for analyzing intrusion alerts and trends over time.

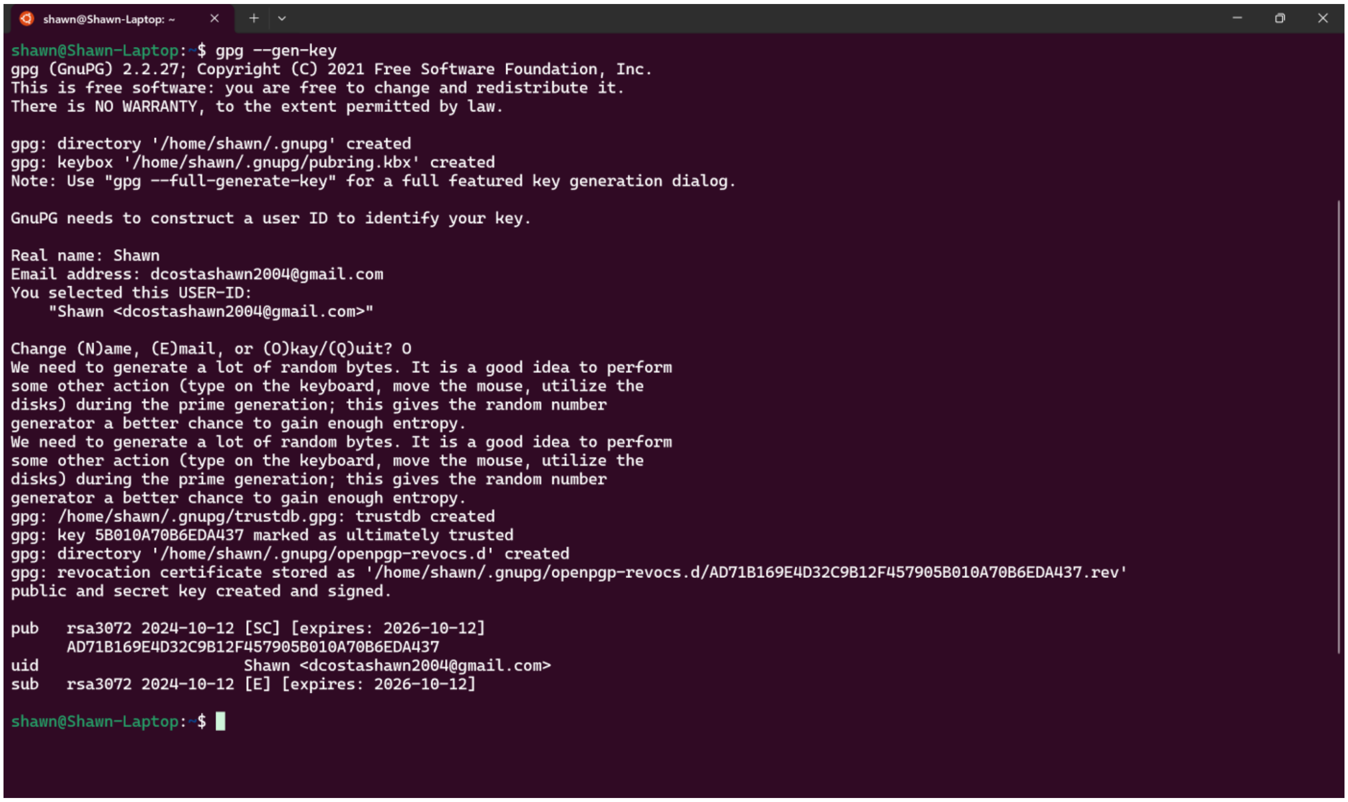
This process provides a robust way to detect and respond to network-based attacks using Snort IDS.

Output:

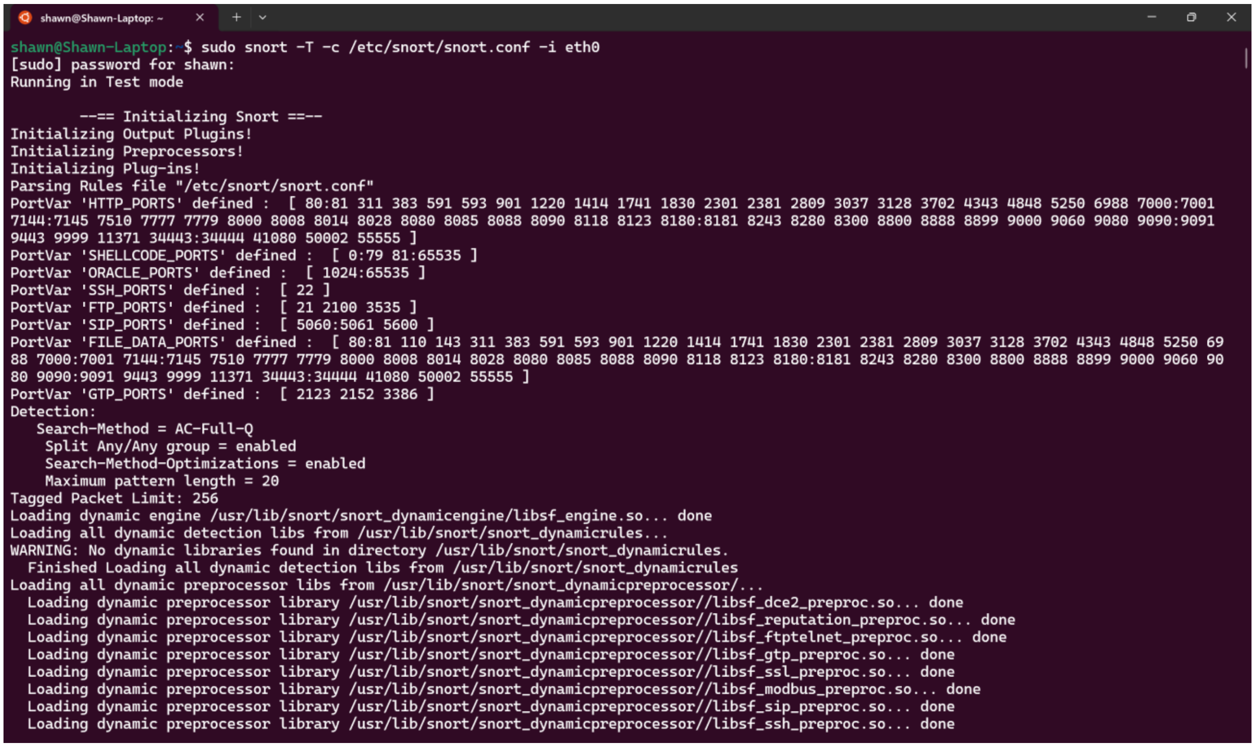
sudo gedit/etc/snort/snort.conf



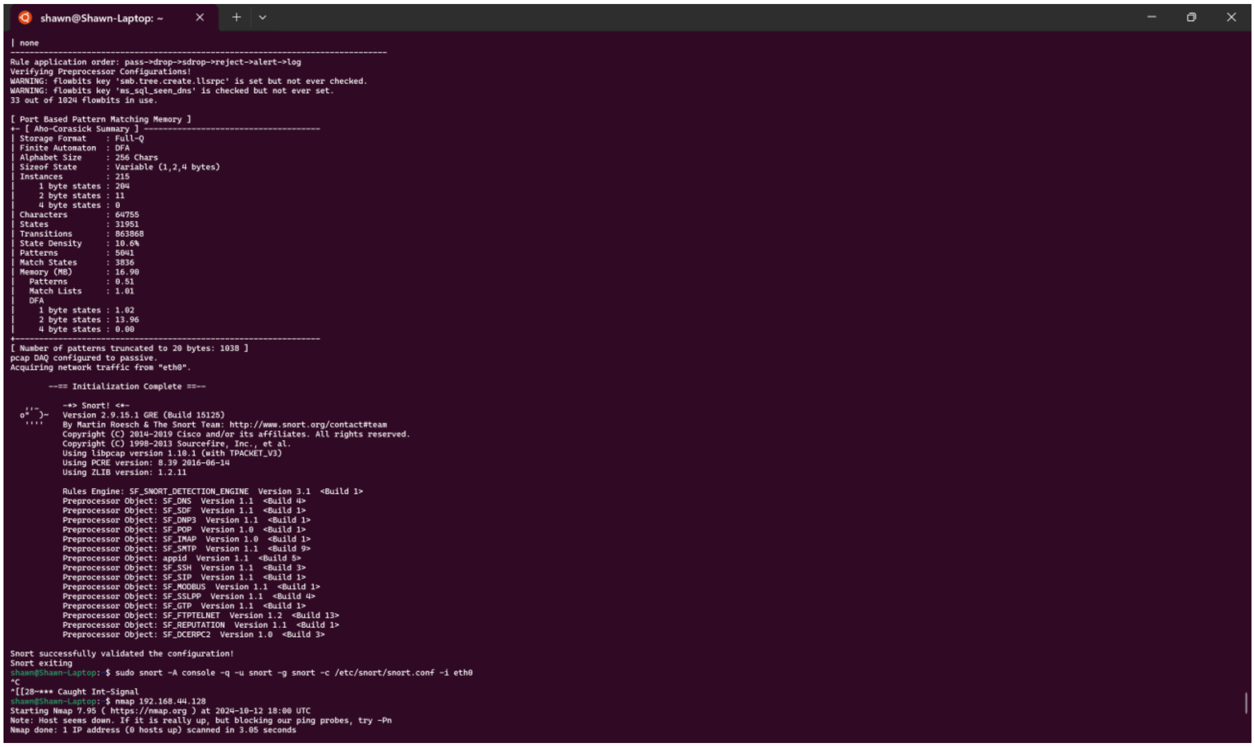
```
#           for a specific interface
####################################################
#
# If you want to run Snort in Debian using different
# instances each handling a different interface and
# a different configuration you can copy this file to
# /etc/snort/snort.$interface.conf (where '$interface' is the name of your
# network interface) and adjust the valuse there.
#
# The Debian init.d script is defined in such a way
# that you can run multiple instances.

####################################################
# Step #1: Set the network variables.  For more information, see README.variables
####################################################

# Setup the network addresses you are protecting
#
# Note to Debian users: this value is overriden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET 192.168.44.0/24

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET
```



```
shawn@Shawn-Laptop:~$ gpg --gen-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/home/shawn/.gnupg' created
gpg: keybox '/home/shawn/.gnupg/pubring.kbx' created
Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name: Shawn
Email address: dcostashawn2004@gmail.com
You selected this USER-ID:
    "Shawn <dcostashawn2004@gmail.com>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit? O
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/shawn/.gnupg/trustdb.gpg: trustdb created
gpg: key 5B010A70B6EDA437 marked as ultimately trusted
gpg: directory '/home/shawn/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/shawn/.gnupg/openpgp-revocs.d/AD71B169E4D32C9B12F457905B010A70B6EDA437.rev'
public and secret key created and signed.

pub   rsa3072 2024-10-12 [SC] [expires: 2026-10-12]
      AD71B169E4D32C9B12F457905B010A70B6EDA437
uid                      Shawn <dcostashawn2004@gmail.com>
sub   rsa3072 2024-10-12 [E] [expires: 2026-10-12]

shawn@Shawn-Laptop:~$ 
```



```
shawn@Shawn-Laptop:~$ sudo snort -T -c /etc/snort/snort.conf -i eth0
[sudo] password for shawn:
Running in Test 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 1414 1741 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001
7144:7145 7510 7777 7779 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091
9443 9999 11371 34443:34444 41080 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 ]
PortVar 'FILE_DATA_PORTS' defined :  [ 80:81 110 143 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 69
88 7000:7001 7144:7145 7510 7777 7779 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 90
80 9090:9091 9443 9999 11371 34443:34444 41080 50002 55555 ]
PortVar 'GTP_PORTS' defined :  [ 2123 2152 3386 ]
Detection:
   Search-Method = AC-Full-Q
    Split Any/Any group = enabled
    Search-Method-Optimizations = enabled
    Maximum pattern length = 20
Tagged Packet Limit: 256
Loading dynamic engine /usr/lib/snort/snort_dynamicengine/libsf_engine.so... done
Loading all dynamic detection libs from /usr/lib/snort/snort_dynamicrules...
WARNING: No dynamic libraries found in directory /usr/lib/snort/snort_dynamicrules.
  Finished Loading all dynamic detection libs from /usr/lib/snort/snort_dynamicrules
Loading all dynamic preprocessor libs from /usr/lib/snort/snort_dynamicpreprocessor/...
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_dce2_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_reputation_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_ftptelnet_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_gtp_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_ssl_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_modbus_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_sip_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_ssh_preproc.so... done
```



```
| none
-------------------------------------------------------------------------------
Rule application order: pass->drop->sdrop->reject->alert->log
Verifying Preprocessor Configurations!
WARNING: flowbits key 'smb.tree.create.llsrpc' is set but not ever checked.
WARNING: flowbits key 'ms_sql_seen_dns' is checked but not ever set.
33 out of 1024 flowbits in use.

[ Port Based Pattern Matching Memory ]
+- [ Aho-Corasick Summary ] -------------------------------------
| Storage Format    : Full-Q
| Finite Automaton  : DFA
| Alphabet Size     : 256 Chars
| Sizeof State      : Variable (1,2,4 bytes)
| Instances         : 215
|     1 byte states : 204
|     2 byte states : 11
|     4 byte states : 0
| Characters        : 64755
| States            : 31951
| Transitions       : 863868
| State Density     : 10.6%
| Patterns          : 5041
| Match States      : 3836
| Memory (MB)       : 16.90
|   Patterns        : 0.51
|   Match Lists     : 1.01
|   DFA
|     1 byte states : 1.02
|     2 byte states : 13.96
|     4 byte states : 0.00
+----------------------------------------------------------------
[ Number of patterns truncated to 20 bytes: 1038 ]
pcap DAQ configured to passive.
Acquiring network traffic from "eth0".

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   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

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.1  <Build 1>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: appid  Version 1.1  <Build 5>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>

Snort successfully validated the configuration!
Snort exiting
shawn@Shawn-Laptop:~$ sudo snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i eth0
^C
^[[28~*** Caught Int-Signal
shawn@Shawn-Laptop:~$ nmap 192.168.44.128
Starting Nmap 7.95 ( https://nmap.org ) at 2024-10-12 18:00 UTC
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.05 seconds
```

Conclusion: Demonstrated the network security system using open source tools (LO6 is achieved).