Name: Harsh Mishra
Roll no: 64
Batch: T21

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 behavior. 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

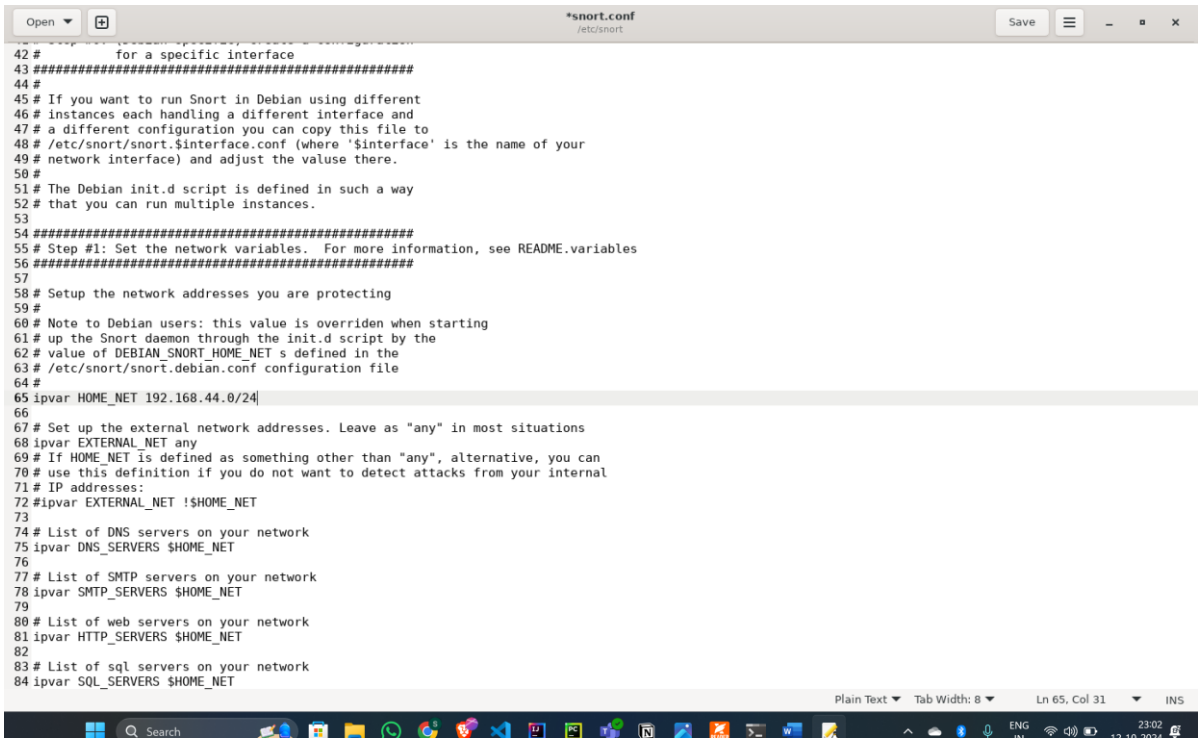
- **Log Review**: Regular log monitoring is crucial for intrusion detection. Administrators can analyze logs manually or use web-based interfaces to visualize and manage alerts more effectively.
- **Integration with Tools**: For more efficient monitoring, Snort can be integrated with visualization and 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

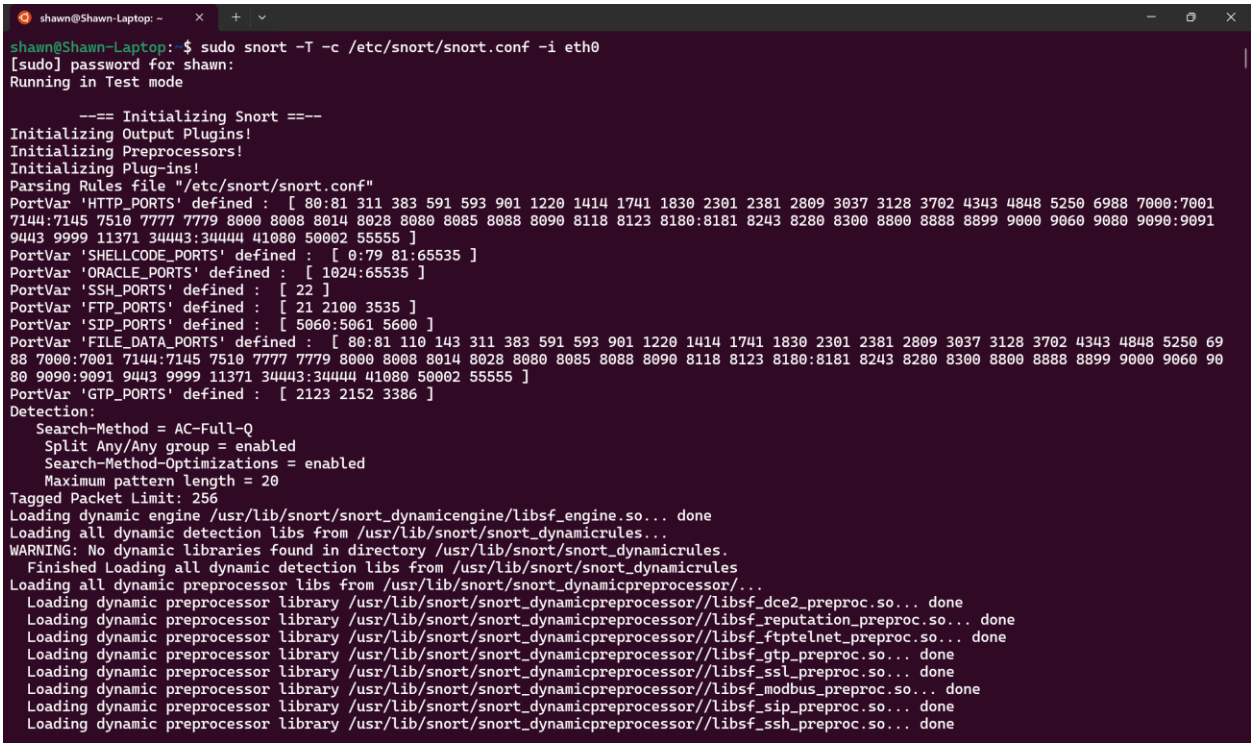
Name: Harsh Mishra
Roll no: 64
Batch: T21

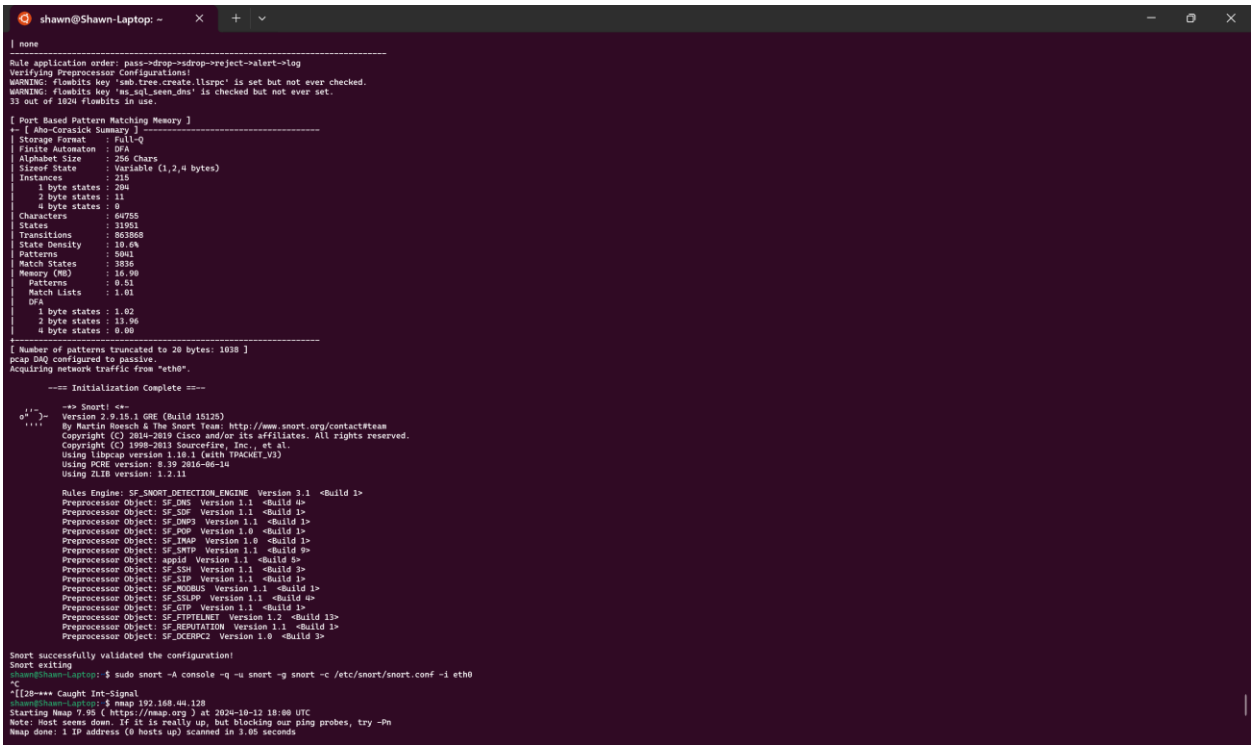


Name: Harsh Mishra
Roll no: 64
Batch: T21

```
harsh@DESKTOP-8O5FLRG:~$ 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/harsh/.gnupg' created
gpg: keybox '/home/harsh/.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: harsh
Email address: harshmftw19@gmail.com
You selected this USER-ID:
    "harsh <harshmftw19@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/harsh/.gnupg/trustdb.gpg: trustdb created
gpg: key 155CC747DD09D833 marked as ultimately trusted
gpg: directory '/home/harsh/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/harsh/.gnupg/openpgp-revocs.d/3C2898C0
95348F8A7D310D5F155CC747DD09D833.rev'
public and secret key created and signed.

pub   rsa3072 2024-10-21 [SC] [expires: 2026-10-21]
      3C2898C095348F8A7D310D5F155CC747DD09D833
uid                      harsh <harshmftw19@gmail.com>
sub   rsa3072 2024-10-21 [E] [expires: 2026-10-21]
```

Name: Harsh Mishra
Roll no: 64
Batch: T21





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