MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

ZTS Assignment

on

Intrusion Detection System using Snort

SUBMITTED

BY

Vaishnavi – 251091010011

Disha Bhat – 251091010027

Under the Guidance of:

Dr.Manoj T
Assistant Professor
School of Computer Engineering Manipal Institute of Technology

Zero Trust Security (CSS5112)
July-November 2025

1. INTRODUCTION TO SNORT

1.1 What is Snort?

Snort is a powerful, open-source network-based intrusion detection system (NIDS) and intrusion prevention system (IPS) created in 1998 by Martin Roesch, founder and former CTO of Sourcefire. Snort is currently developed and maintained by Cisco, which acquired Sourcefire in 2013. In 2009, Snort entered InfoWorld's Open Source Hall of Fame as one of the "greatest pieces of open source software of all time."

Snort performs real-time traffic analysis and packet logging on IP networks, enabling it to detect various attacks as they occur. With millions of downloads and approximately 400,000 registered users, Snort has become the industry standard for intrusion detection and prevention systems.

1.2 Why Use Snort?

Snort is widely used in cybersecurity for several compelling reasons:

- **Real-Time Traffic Analysis**: Monitors network traffic and detects threats as they happen
- **Protocol Analysis**: Understands and analyzes various network protocols (TCP, UDP, ICMP, etc.)
- **Content Searching and Matching**: Identifies malicious patterns in network traffic
- **Attack Detection**: Detects buffer overflows, port scans, DoS attacks, SQL injection, and more
- **Flexible Rule-Based Engine**: Customizable detection through user-defined rules
- **Multiple Operating Modes**: Can function as sniffer, packet logger, or full NIDS/IPS
- **Open Source and Free**: No licensing costs, with active community support
- **Cross-Platform**: Works on Linux, Windows, Unix, and BSD systems

1.3 How Snort Works

Snort operates through a modular architecture with four primary components:

1. **Packet Capture and Decoder Engine**: Acquires network traffic using libpcap and decodes packet headers
2. **Preprocessors**: Normalize, reassemble, and prepare packets for analysis
3. **Detection Engine**: Applies rule-based signature detection to identify threats
4. **Output Plugins**: Generate alerts and log events in various formats (text, database, syslog, etc.)

Detection Process:

1. Network packets are captured from the wire
2. Packets are decoded and preprocessed
3. The detection engine compares packets against rule set
4. When a match is found, configured action is taken (alert, log, drop, etc.)
5. Output plugins record the event for analysis.

1.4 Key Capabilities

- **Live Traffic Analysis**: Real-time monitoring and analysis
- **Attack and Probe Detection**: Identifies reconnaissance and attack attempts
- **Packet Logging**: Records network packets for forensic analysis
- **Protocol Analysis**: Deep inspection of network protocols
- **Real-Time Alerting**: Immediate notification of security events
- **Modular Architecture**: Extensible through plugins and preprocessors
- **Cross-Platform Support**: Runs on Linux, Windows, Unix, BSD

1.5 Types of Detection

Signature-Based Detection: - Compares network traffic against known attack patterns - Excellent for detecting known threats - Cannot detect zero-day or unknown attacks

Anomaly-Based Detection: - Identifies deviations from normal behavior - Can detect new/unknown threats - Higher false positive rate

Snort primarily uses signature-based detection but can be configured with anomaly detection capabilities through custom rules and preprocessors.

2. INSTALLATION ON DIFFERENT OPERATING SYSTEMS

2.1 Linux Installation (Ubuntu/Debian)

Step 1: Update System Packages

```
sudo apt update
sudo apt upgrade -y
```

Explanation: Updates package lists and upgrades existing packages to latest versions.

Step 2: Install Snort

```
sudo apt install snort -y
```

Explanation: Snort installation from Ubuntu repositories. During installation, you'll be prompted to configure: - Network interface to monitor (e.g., eth0, ens33) - HOME_NET address range (e.g., 192.168.1.0/24)

Step 3: Configure Network Interface

During installation, when prompted: - **Interface**: Enter your network interface name (check with *ip a* command) - **HOME_NET**: Enter your local network CIDR (e.g., 192.168.1.0/24)

Step 4: Verify Installation

```
snort -V
```

Expected Output:

```
  ,,_     -*> Snort! <*-
 o"  )~   Version 2.9.15.1 GRE (Build 82)
  ""    By Martin Roesch & The Snort Team
        Copyright (C) 1998-2019 Sourcefire, Inc., et al.
```

Step 5: Test Configuration

```
sudo snort -T -c /etc/snort/snort.conf
```

Explanation: Tests (*-T*) the configuration file (*-c*) without actually running Snort. Should show "Snort successfully validated the configuration!"

2.2 Installation on Windows

Step 1: Download Snort

1. Visit: https://www.snort.org/downloads
2. Download the latest Windows installer (e.g., *Snort_2_9_15_1_Installer.exe*)
3. Register for a free community account if prompted

Step 2: Install WinPcap or Npcap

Recommended: Npcap

1. Download from: *https://npcap.com/#download*
2. Run the installer with administrator privileges
3. Select "Install Npcap in WinPcap API-compatible Mode"
4. Complete installation

Step 3: Install Visual C++ Redistributable

1. Download from Microsoft's website
2. Install Visual C++ Redistributable 2015-2019 (x64)
3. Required for Snort to run properly

Step 4: Install Snort

1. Run Snort installer as Administrator
2. Choose installation directory (default: *C:\Snort*)
3. Complete installation wizard
4. Installation creates directories:
    - *C:\Snort\bin* - Snort executable
    - *C:\Snort\etc* - Configuration files
    - *C:\Snort\rules* - Rule files
    - *C:\Snort\log* - Log directory

Step 5: Configure Snort

1. Navigate to *C:\Snort\etc*
2. Edit *snort.conf* in text editor
3. Set *HOME_NET* variable to your network (e.g., *192.168.1.0/24*)
4. Set paths to rule files and output plugins

Step 6: Verify Installation

Open Command Prompt as Administrator:

```
cd C:\Snort\bin
snort -V
```

Expected Output: Version information similar to Linux

Step 7: Test Configuration

```
snort -T -c C:\Snort\etc\snort.conf
```

3. SNORT OPERATIONAL MODES

Snort can operate in three primary modes, each serving different purposes:

3.1 Sniffer Mode

Purpose: Capture and display network packets in real-time on console (like tcpdump or Wireshark).

Basic Sniffer Command:

```
snort -v
```

Explanation: - *-v*: Verbose mode, displays packet headers and data

Display IP Headers Only:

```
snort -vd
```

Explanation: - *-v*: Verbose mode - *-d*: Display application layer data

Display Full Packet Details:

```
snort -vde
```

Explanation: - *-v*: Verbose - *-d*: Display data - *-e*: Display data link layer headers

Sniff on Specific Interface:

```
snort -v -i eth0
```

Explanation: - *-i eth0*: Monitor specific interface (replace eth0 with your interface)

3.2 Packet Logger Mode

Purpose: Record packets to disk for later analysis and forensics.

Log Packets to Directory:

```
snort -dev -l C:\Snort\log
```

Explanation: *-dev*: Display full packet information - *-l /var/log/snort*: Log directory path

Log in Binary Format (pcap):

```
snort -b -l C:\Snort\log
```

Explanation: *-b*: Log in binary tcpdump format - Faster logging, readable by Wireshark

Log Packets from Specific Network:

snort -dev -l C:\Snort\log -h 192.168.1.0/24

Explanation: *-h 192.168.1.0/24*: Define HOME_NET (your network)

3.3 Network Intrusion Detection System (NIDS) Mode

Purpose: Monitor network traffic, apply rules, and generate alerts for suspicious activity.

Run Snort in NIDS Mode:

snort -A console -q -c C:\Snort\etc\snort.conf -i 1

Explanation: - *-A console*: Alert mode - display alerts on console - *-q*: Quiet mode (suppress banner and status info) - *-c C:\Snort\etc\snort.conf*: Configuration file path - *-i eth0*: Interface to monitor

NIDS Mode with Logging:

snort -c C:\Snort\etc\snort.conf -l C:\Snort\log -i 1

Explanation: - Logs alerts to */var/log/snort* directory - Uses rules from configuration file

Run as Background Daemon:

snort -D -c C:\Snort\etc\snort.conf -i 1

Explanation: - *-D*: Run as daemon (background process) - Suitable for production deployment

3.4 Inline/IPS Mode

Purpose: Actively block malicious traffic (requires iptables integration).

Basic IPS Mode:

snort -Q -c C:\Snort\etc\snort.conf -i 1

Explanation: - *-Q*: Enable inline mode - Can drop/reject packets matching rules with “drop” or “reject” actions - Requires proper iptables configuration

4. CONFIGURATION AND RULES

4.1 Snort Configuration File (snort.conf)

The main configuration file is typically located at: - **Linux**: */etc/snort/snort.conf* - **Windows**: *C:\Snort\etc\snort.conf*

Key Sections in snort.conf:

4.1.1 Variable Definitions

```
# Define your network
var HOME_NET 192.168.1.0/24

# Define external network
var EXTERNAL_NET !$HOME_NET

# Define servers
var DNS_SERVERS $HOME_NET
var SMTP_SERVERS $HOME_NET
var HTTP_SERVERS $HOME_NET
var SQL_SERVERS $HOME_NET

# Define ports
var HTTP_PORTS [80,443,8080,8443]
var SHELLCODE_PORTS !80
var SSH_PORTS 22
```

Explanation: - *HOME_NET*: Your protected internal network - *EXTERNAL_NET*: Everything outside your network (uses negation *!*) - Server variables: Specific servers to protect - Port variables: Common service ports

4.1.2 Preprocessor Configuration

```
# HTTP Inspect preprocessor
preprocessor http_inspect: global iis_unicode_map unicode.map 1252

preprocessor http_inspect_server: server default \
    profile all \
    ports { 80 8080 8180 } \
    oversize_dir_length 500

# Stream5 preprocessor (for TCP reassembly)
preprocessor stream5_global: track_tcp yes, \
   track_udp yes, track_icmp no

preprocessor stream5_tcp: policy windows, \
   detect_anomalies, require_3whs 180, \
   overlap_limit 10, small_segments 3 bytes 150
```

Explanation: - Preprocessors analyze and normalize traffic before detection - *http_inspect*: Decodes HTTP traffic - *stream5*: Tracks TCP sessions and reassembles streams - Essential for detecting fragmented attacks

4.1.3 Output Configuration

```
# Alert to console
output alert_fast: stdout

# Log to file
output alert_full: /var/log/snort/alert
```

```
# Log to unified2 format (for Barnyard2)
output unified2: filename snort.u2, limit 128
# Log to syslog
output alert_syslog: LOG_AUTH LOG_ALERT
```

Explanation: - Multiple output plugins can be active simultaneously - *alert_fast*: Quick summary format - *alert_full*: Detailed alert information - *unified2*: Binary format for external processors - *alert_syslog*: System log integration

4.2 Rule File Locations

Linux: - */etc/snort/rules/* - Main rule directory - */etc/snort/rules/local.rules* - Custom local rules

Windows: - *C:\Snort\rules* - Main rule directory - *C:\Snort\rules\local.rules* - Custom local rules

5. ESSENTIAL SNORT COMMANDS AND USAGE

5.1 Basic Commands

Command	Purpose
snort -V	Display version information
snort -?	Display help and options
snort -v	Run in sniffer mode (verbose)
snort -vd	Sniffer with data payload
snort -vde	Sniffer with data link layer

5.2 Configuration Testing

Command	Purpose
snort -T -c /etc/snort/snort.conf	Test configuration
snort -T -c /etc/snort/snort.conf -i eth0	Test on specific interface

5.3 Packet Logging Commands

Command	Purpose
snort -dev -l ./log	Log packets to ./log directory
snort -b -l ./log	Log in binary (pcap) format
snort -dev -l ./log -h 192.168.1.0/24	Log from specific network
snort -r packet.log	Read from captured packet file

5.4 NIDS Mode Commands

```
# Console alerts
snort -A console -q -c C:\Snort\etc\snort.conf -i 1

# Fast alert mode
snort -A fast -c C:\Snort\etc\snort.conf -i 1
```

```
# Full alert mode
snort -A full -c C:\Snort\etc\snort.conf -i 1

# Run as daemon
snort -D -c C:\Snort\etc\snort.conf -i 1
```

5.5 Alert Output Options

Option	Description	Output Format
-A console	Display alerts on console	Real-time console output
-A fast	Fast alert format	Timestamp, alert message, IP info
-A full	Full alert details	Complete packet headers and data
-A none	No alerts	Silent mode
-A cmg	CMG format	Computer Misuse Group format

5.6 Useful Command Combinations

Test Rules Before Deployment:

```
snort -T -c C:\Snort\etc\snort.conf
```

Run with Custom Rule File:

```
snort -c C:\Snort\etc\snort.conf -R C:\Snort\rules\local.rules -i 1
```

Read Captured Traffic:

```
snort -r capture.pcap -c C:\Snort\etc\snort.conf
```

Process Multiple PCAP Files:

```
snort --pcap-dir=/var/log/pcaps -c C:\Snort\etc\snort.conf
```

6. WRITING SNORT RULES

6.1 Snort Rule Structure

A Snort rule consists of two logical sections: **Rule Header** and **Rule Options**

Basic Syntax:

```
action protocol source_ip source_port direction destination_ip destination_port (rule options)
```

Example:

```
alert tcp any any -> 192.168.1.0/24 80 (msg:"HTTP Traffic Detected"; sid:1000001; rev:1;)
```

6.2 Rule Header Components

6.2.1 Rule Actions

Action	Behavior
alert	Generate an alert and log a packet

Action	Behavior
log	Log packet only (no alert)
pass	Ignore packet (whitelist)
drop	Drop packet and log (IPS mode)
reject	Drop packet and send reset (IPS mode)
sdrop	Drop packet silently (no log)

6.2.2 Protocols

Protocol	Description
tcp	Transmission Control Protocol
udp	User Datagram Protocol
icmp	Internet Control Message Protocol
ip	Any IP protocol

6.2.3 IP Addresses

Notation	Meaning
any	Any IP address
192.168.1.1	Specific IP
192.168.1.0/24	Network range (CIDR)
[192.168.1.1,192.168.1.5]	IP list
!192.168.1.1	NOT this IP (negation)
$HOME_NET	Variable from snort.conf

6.2.4 Port Numbers

Notation	Meaning
any	Any port
80	Specific port
1:1024	Port range
[80,443,8080]	Port list
!22	NOT this port
$HTTP_PORTS	Variable from Snort.conf

6.3 Rule Options

Rule options define specific detection criteria and output behavior.

Basic Syntax:

(option1:"value1"; option2:"value2"; option3:"value3";)

6.3.1 Essential Rule Options

Option	Purpose	Example
msg	Alert message	*msg:"SQL Injection Attempt";*

Option	Purpose	Example
sid	Snort ID (unique identifier)	*sid:1000001;*
rev	Revision number	*rev:1;*
classtype	Attack classification	*classtype:web-application-attack;*
priority	Alert priority (1-4)	*priority:1;*
reference	External reference	*reference:cve,2021-44228;*

Note: SID numbers < 1,000,000 are reserved for official rules. Use SID >= 1,000,000 for custom rules.

6.3.2 Detection Options

Option	Purpose	Example
content	Match a specific string	*content:"admin";*
nocase	Case-insensitive match	*content:"admin"; nocase;*
offset	Start search at byte position	*content:"GET"; offset:0;*
depth	Search within the first N bytes	*content:"GET"; depth:4;*
distance	Bytes after the previous match	*distance:10;*
within	Search within N bytes of the previous	*within:20;*
pcre	Perl Compatible Regex	*pcre:"/admin\.php/i";*
byte_test	Test byte values	*byte_test:4,>,1000,0;*
byte_jump	Skip bytes	*byte_jump:4,12,relative;*

6.3.3 Non-Payload Detection

Option	Purpose	Example
flags	TCP flags	*flags:S;* (SYN flag)
flow	Connection state	*flow:established,to_server;*
ttl	Time To Live	*ttl:>50;*
itype	ICMP type	*itype:8;* (Echo Request)
icode	ICMP code	*icode:0;*
tos	Type of Service	*tos:10;*
id	IP identification	*id:0;*
fragbits	IP fragmentation	*fragbits:D;* (Don’t Fragment)

6.4 Rule Examples

Example 1: Detect ICMP Ping

alert icmp any any -> $HOME_NET any (msg:"ICMP Ping Detected"; itype:8; sid:1000001; rev:1;)

Explanation: - *alert*: Generate alert - *icmp*: Protocol - *any any*: From any source IP and port -> *$HOME_NET any*: To home network, any port - *itype:8*: ICMP Echo Request (ping)

Example 2: Detect SQL Injection Attempt

alert tcp any any -> $HTTP_SERVERS $HTTP_PORTS (msg:"Possible SQL Injection"; con tent:"union"; nocase; content:"select"; nocase; sid:1000002; rev:1; classtype:web-application-attack; priority:1;)

Explanation: - Detects TCP traffic to web servers - Looks for “union” and “select” (case-insensitive) - Classifies as web application attack - High priority (1)

Example 3: Detect SSH Brute Force

alert tcp any any -> $HOME_NET 22 (msg:"SSH Brute Force Attempt"; flags:S; threshold:ty pe threshold, track by_src, count 5, seconds 60; sid:1000003; rev:1; classtype:attempted-reco n;)

Explanation: - Monitors SSH connections (port 22) - Tracks SYN flags - Alerts if 5 or more connections from the same source in 60 seconds - Indicates reconnaissance activity

6.5 Testing Custom Rules

Step 1: Create local.rules file

notepad C:\Snort\rules\local.rules

Step 2: Add your custom rule

alert icmp any any -> $HOME_NET any (msg:"ICMP Test Alert"; sid:1000001; rev:1;)

Step 3: Test configuration

snort -T -c C:\Snort\etc\snort.conf

Step 4: Run Snort with the rule

snort -A console -q -c /etc/snort/snort.conf -i eth0

Step 5: Test the rule

From another machine, ping the Snort machine:

ping <snort_machine_ip>

You should see alerts in Snort console.

7. PRACTICAL USE CASES

Use Case 1: Network Traffic Monitoring and Analysis

Scenario: Network administrator needs to monitor all network traffic for suspicious activity.

Objective: Set up Snort to capture and analyze network traffic in real-time.

Commands:

 # Step 1: Run Snort in sniffer mode to view traffic
snort -v -i 1

 # Step 2: Log all traffic for later analysis
snort -dev -l C:\Snort\log -i 5

 # Step 3: Analyze logged traffic
snort -r C:\Snort\log\snort.log.xxxxxxxxx

Business Value: - Real-time visibility into network traffic - Forensic evidence for security incidents - Network troubleshooting and performance analysis.

Use Case 2: Detecting Port Scanning Activities

Scenario: The Security team needs to detect reconnaissance activities like port scans.

Objective: Configure Snort rules to identify and alert on port scanning attempts.

Rule:

alert tcp any any -> $HOME_NET any (msg:"Potential Port Scan Detected"; flags:S; detectio n_filter:track by_src, count 20, seconds 60; sid:1000010; rev:1; classtype:attempted-recon; pr iority:2;)

Explanation: - Monitors TCP SYN packets - Alerts if 20+ different ports accessed from same source in 60 seconds - Indicates nmap or similar scanning tool activity

Testing:

 # From attacker machine, run port scan
nmap -sS 192.168.1.100

 # Snort will generate alerts

Business Value: - Early warning of potential attacks - Identifies reconnaissance phase of cyber attacks - Enables proactive response before actual exploit.

Use Case 3: Detecting SQL Injection Attacks

Scenario: Web application server needs protection against SQL injection attempts.

Objective: Create Snort rules to detect SQL injection patterns in HTTP traffic.

Rules:

alert tcp any any -> $HTTP_SERVERS $HTTP_PORTS (msg:"SQL Injection - UNION SEL ECT"; content:"union"; nocase; content:"select"; nocase; distance:0; sid:1000020; rev:1; class type:web-application-attack; priority:1;)

alert tcp any any -> $HTTP_SERVERS $HTTP_PORTS (msg:"SQL Injection - OR 1=1"; co ntent:"or"; nocase; content:"1=1"; nocase; distance:0; sid:1000021; rev:1; classtype:web-appl ication-attack; priority:1;)

alert tcp any any -> $HTTP_SERVERS $HTTP_PORTS (msg:"SQL Injection - DROP TAB LE"; content:"drop"; nocase; content:"table"; nocase; distance:0; sid:1000022; rev:1; classtyp e:web-application-attack; priority:1;)

Configuration:

 # Add rules to local.rules
notepad C:\Snort\rules\local.rules

 # Test configuration
snort -T -c C:\Snort\etc\snort.conf

 # Run Snort in NIDS mode
snort -A console -q -c C:\Snort\etc\snort.conf -i 1

Business Value: - Protects web applications from database compromise - Detects automated SQL injection scanning tools - Prevents data breaches and exfiltration.

8. COMMON ISSUES AND TROUBLESHOOTING

Issue 1: Snort Configuration Test Fails

Error Message:

ERROR: /etc/snort/snort.conf(104) => Unable to open rules file: /etc/snort/rules/community.rules
Fatal Error, Quitting..

Root Cause: - Rule file doesn't exist or incorrect path - Missing rule files after installation - Permission issues

Solution 1: Check Rule File Existence

dir C:\Snort\rules

If file doesn't exist, comment it out in snort.conf:

notepad C:\Snort\etc\snort.conf
 # Comment the line
 # include $RULE_PATH/community.rules

Solution 2: Download Community Rules

 # Download community rules
wget https://www.snort.org/downloads/community/community-rules.tar.gz

 # Extract to rules directory
community-rules.tar.gz to C:\Snort\rules

Solution 3: Fix Permissions

sudo chmod 644 /etc/snort/rules/*.rules
sudo chown -R snort:snort /etc/snort/rules/

Issue 2: No Interface Specified Error

Error Message:

ERROR: No interface specified. Use -i <interface>
Fatal Error, Quitting..

Root Cause: - Interface not specified in the command line - Incorrect interface name

Solution 1: Specify Interface

 # Check available interfaces
ip a

 # Run Snort with the correct interface
snort -c C:\Snort\etc\snort.conf -i 1

Solution 2: Set the Default Interface in snort.conf

notepad C:\Snort\etc\snort.conf

 # Add near top of file
config interface: 1

Issue 3: Permission Denied When Running Snort

Error Message:

ERROR: Can't open pcap library: socket: Operation not permitted
Fatal Error, Quitting..

Root Cause: - Running Snort without root/sudo privileges - Insufficient permissions for packet capture

Solution:

 # Always run Snort with sudo
snort -c C:\Snort\etc\snort.conf -i 1

 # Or add user to snort group in linux
sudo usermod -aG snort $USER

Troubleshooting Quick Reference

Problem	Quick Fix	Verification
Config test fails	Check rule file paths	*sudo snort -T -c /etc/snort/snort.conf*
No alerts	Test with simple ICMP rule	*ping <snort_ip>*
Permission denied	Run with sudo	*sudo snort ...*
Interface error	Specify interface with *-i*	*ip a* to list interfaces
High CPU usage	Disable unused rule categories	Monitor with *top*
Rule not triggering	Verify traffic matches rule	Use *tcpdump* to inspect traffic

9. Scenarios

9.1 Use case 1: Live Network Monitoring

Step 1: Verbose Packet Capture on Specific Interface

Input : snort -v -i 5

Output :

```
C:\Snort\bin>snort -v -i 5
Running in packet dump mode

        --== Initializing Snort ==--
Initializing Output Plugins!
pcap DAQ configured to passive.
The DAQ version does not support reload.
Acquiring network traffic from "\Device\NPF_{BFE24C28-A54D-4961-A648-4EF148DBF2E7}".
Decoding Ethernet

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.20-WIN64 GRE (Build 82)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using PCRE version: 8.10 2010-06-25
           Using ZLIB version: 1.2.11

Commencing packet processing (pid=2588)
WARNING: No preprocessors configured for policy 0.
11/03-18:54:15.031177 192.168.31.1 -> 224.0.0.1
PROTO:002 TTL:1 TOS:0xC0 ID:62770 IpLen:24 DgmLen:36 DF
IP Options (1) => RTRALT
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
11/03-18:54:15.031177 fe80:0000:0000:0000:3e55:dbff:fea9:8b45 -> ff02:0000:0000:0000:0000:0000:0000:0001
IPV6-ICMP TTL:1 TOS:0x0 ID:0 IpLen:40 DgmLen:76
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
11/03-18:54:15.031530 fe80:0000:0000:0000:3e55:dbff:fea9:8b45 -> ff02:0000:0000:0000:0000:0000:0000:0001
IPV6-ICMP TTL:1 TOS:0x0 ID:0 IpLen:40 DgmLen:76
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

```
===============================================================================
Run time for packet processing was 7.92000 seconds
Snort processed 21 packets.
Snort ran for 0 days 0 hours 0 minutes 7 seconds
   Pkts/sec:            3
===============================================================================
Packet I/O Totals:
   Received:           21
   Analyzed:           21 (100.000%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:            0 (  0.000%)
   Injected:            0
===============================================================================
Breakdown by protocol (includes rebuilt packets):
        Eth:           21 (100.000%)
       VLAN:            0 (  0.000%)
        IP4:            3 ( 14.286%)
       Frag:            0 (  0.000%)
       ICMP:            0 (  0.000%)
        UDP:            0 (  0.000%)
        TCP:            0 (  0.000%)
        IP6:           18 ( 85.714%)
    IP6 Ext:           25 (119.048%)
   IP6 Opts:            7 ( 33.333%)
      Frag6:            0 (  0.000%)
      ICMP6:            8 ( 38.095%)
       UDP6:            4 ( 19.048%)
       TCP6:            6 ( 28.571%)
     Teredo:            0 (  0.000%)
    ICMP-IP:            0 (  0.000%)
      EAPOL:            0 (  0.000%)
```

```
===============================================================================

Memory Statistics for File at:Mon Nov  3 18:54:20 2025

Total buffers allocated:          0
Total buffers freed:              0
Total buffers released:           0
Total file mempool:               0
Total allocated file mempool:     0
Total freed file mempool:         0
Total released file mempool:      0

Heap Statistics of file:
          Total Statistics:
               Memory in use:              0 bytes
                No of allocs:              0
                 No of frees:              0
===============================================================================
```

Snort starts capturing all network traffic on interface 5 and displays packet details in real-time to the console. Each packet shows source/destination IP addresses, ports, protocols (TCP/UDP/ICMP), and flags. This is useful for monitoring live traffic and understanding network activity without applying detection rules.

Step 2: Capture and Log Packets with Header and Data

Input: snort -dev -l C:\Snort\log -i 5

Output:

```
C:\Snort\bin>snort -dev -l C:\Snort\log -i 5
Running in packet logging mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Log directory = C:\Snort\log
pcap DAQ configured to passive.
The DAQ version does not support reload.
Acquiring network traffic from "\Device\NPF_{BFE24C28-A54D-4961-A648-4EF148DBF2E7}".
Decoding Ethernet

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.20-WIN64 GRE (Build 82)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using PCRE version: 8.10 2010-06-25
           Using ZLIB version: 1.2.11

Commencing packet processing (pid=13612)
```

```
WARNING: No preprocessors configured for policy 0.
===============================================================================
Run time for packet processing was 4.75000 seconds
Snort processed 41 packets.
Snort ran for 0 days 0 hours 0 minutes 4 seconds
   Pkts/sec:           10
===============================================================================
Packet I/O Totals:
   Received:           87
   Analyzed:           41 ( 47.126%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:           46 ( 52.874%)
   Injected:            0
===============================================================================
Breakdown by protocol (includes rebuilt packets):
        Eth:           41 (100.000%)
       VLAN:            0 (  0.000%)
        IP4:           34 ( 82.927%)
       Frag:            0 (  0.000%)
       ICMP:            0 (  0.000%)
        UDP:            0 (  0.000%)
        TCP:           29 ( 70.732%)
        IP6:            7 ( 17.073%)
    IP6 Ext:           10 ( 24.390%)
   IP6 Opts:            3 (  7.317%)
      Frag6:            0 (  0.000%)
      ICMP6:            4 (  9.756%)
       UDP6:            0 (  0.000%)
       TCP6:            3 (  7.317%)
     Teredo:            0 (  0.000%)
    ICMP-IP:            0 (  0.000%)
      EAPOL:            0 (  0.000%)
```

```
Memory Statistics for File at:Mon Nov  3 19:10:22 2025

Total buffers allocated:           0
Total buffers freed:               0
Total buffers released:            0
Total file mempool:                0
Total allocated file mempool:      0
Total freed file mempool:          0
Total released file mempool:       0

Heap Statistics of file:
          Total Statistics:
               Memory in use:              0 bytes
                No of allocs:              0
                 No of frees:              0
===============================================================================
Snort exiting
```

Snort captures all packets on interface 5 and saves them to binary log files in the C:\Snort\log directory. Each packet is logged with full headers (Ethernet, IP, transport layer) and payload data. The log files are saved as snort.log.TIMESTAMP format and can be analyzed later or read with Snort's offline analysis mode.

Step 3: Read and Analyse Captured Logs Offline

Input: snort -r C:\Snort\log\snort.log.1762177218

Output:

```
C:\Snort\bin>snort -r C:\Snort\log\snort.log.1762177218
Running in packet dump mode

        --== Initializing Snort ==--
Initializing Output Plugins!
pcap DAQ configured to read-file.
The DAQ version does not support reload.
Acquiring network traffic from "C:\Snort\log\snort.log.1762177218".

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.20-WIN64 GRE (Build 82)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using PCRE version: 8.10 2010-06-25
           Using ZLIB version: 1.2.11

Commencing packet processing (pid=18796)
WARNING: No preprocessors configured for policy 0.
11/03-19:10:19.552835 192.168.31.194:54887 -> 40.126.18.32:443
TCP TTL:128 TOS:0x0 ID:40554 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA1E9D411  Ack: 0x0  Win: 0xFFFF  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP WS: 8 NOP NOP SackOK
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
11/03-19:10:19.657334 40.126.18.32:443 -> 192.168.31.194:54887
TCP TTL:111 TOS:0x0 ID:31840 IpLen:20 DgmLen:52 DF
***A**S* Seq: 0x10DE2204  Ack: 0xA1E9D412  Win: 0xFFFF  TcpLen: 32
TCP Options (6) => MSS: 1282 NOP WS: 8 NOP NOP SackOK
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
11/03-19:10:19.657574 192.168.31.194:54887 -> 40.126.18.32:443
TCP TTL:128 TOS:0x0 ID:40555 IpLen:20 DgmLen:40 DF
***A**** Seq: 0xA1E9D412  Ack: 0x10DE2205  Win: 0xFF  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
11/03-19:10:19.665202 192.168.31.194:54887 -> 40.126.18.32:443
TCP TTL:128 TOS:0x0 ID:40556 IpLen:20 DgmLen:504 DF
***AP*** Seq: 0xA1E9D412  Ack: 0x10DE2205  Win: 0xFF  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

```
===============================================================================
Run time for packet processing was 0.22000 seconds
Snort processed 41 packets.
Snort ran for 0 days 0 hours 0 minutes 0 seconds
   Pkts/sec:           41
===============================================================================
Packet I/O Totals:
   Received:           41
   Analyzed:           41 (100.000%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:            0 (  0.000%)
   Injected:            0
===============================================================================
Breakdown by protocol (includes rebuilt packets):
        Eth:           41 (100.000%)
       VLAN:            0 (  0.000%)
        IP4:           34 ( 82.927%)
       Frag:            0 (  0.000%)
       ICMP:            0 (  0.000%)
        UDP:            0 (  0.000%)
        TCP:           29 ( 70.732%)
        IP6:            7 ( 17.073%)
    IP6 Ext:           10 ( 24.390%)
   IP6 Opts:            3 (  7.317%)
      Frag6:            0 (  0.000%)
      ICMP6:            4 (  9.756%)
       UDP6:            0 (  0.000%)
       TCP6:            3 (  7.317%)
     Teredo:            0 (  0.000%)
    ICMP-IP:            0 (  0.000%)
```

```
Memory Statistics for File at:Mon Nov  3 21:12:57 2025

Total buffers allocated:          0
Total buffers freed:              0
Total buffers released:           0
Total file mempool:               0
Total allocated file mempool:     0
Total freed file mempool:         0
Total released file mempool:      0

Heap Statistics of file:
         Total Statistics:
              Memory in use:              0 bytes
               No of allocs:              0
                No of frees:              0
===============================================================================
Snort exiting
```

Snort reads the previously captured binary log file and displays the contents in human-readable format. This allows you to review captured traffic after the capture session ends. Snort reconstructs and displays all packets from the log file, showing the same detailed information that was captured during the live session (source/destination IPs, ports, protocols, packet data).

10. BEST PRACTICES

10.1 Deployment Best Practices

1. Network Placement - Deploy Snort at network perimeter (monitor all inbound/outbound traffic) - Use network TAP or SPAN port for IDS deployment - For IPS mode, deploy inline between firewall and protected network

2. Hardware Sizing - Minimum: 2 CPU cores, 4GB RAM for small networks (<100 Mbps) - Recommended: 4+ CPU cores, 8GB+ RAM for enterprise networks - Consider network bandwidth when sizing hardware

3. Rule Management - Start with community rules for baseline coverage - Subscribe to Snort Subscriber rules or VRT rules for better protection - Regular rule updates (daily or weekly) - Disable unnecessary rules to reduce false positives - Create custom rules for organization-specific threats

4. Logging Strategy - Use unified2 output for performance - Rotate logs regularly to prevent disk space issues - Implement centralized logging (syslog, SIEM integration) - Keep 30-90 days of logs for forensic analysis.

10.2 Performance Optimization

1. Preprocessor Tuning - Tune stream5 memory limits based on available RAM - Adjust HTTP inspect settings for your web traffic patterns - Enable only required preprocessors.

2. Rule Optimization - Use rule profiling to identify slow rules - Optimize content matching (use offset, depth, distance, within) - Avoid complex PCRE when possible - Group related content matches.

3. System Tuning - Increase receive buffer size: *sysctl -w net.core.rmem_max=8388608* - Disable unnecessary services on Snort sensor - Use dedicated network interface for monitoring.

10.3 Security Best Practices

1. Rule Updates - Automate rule updates with cron jobs - Test rule updates in non-production environment first - Subscribe to Snort mailing list for security advisories.

2. Alert Management - Triage alerts by priority (1=high, 4=low) - Investigate all priority 1 alerts - Tune rules to reduce false positives - Document false positive rules and reasons.

3. Response Procedures - Define incident response procedures for different alert types - Integrate Snort with firewall for automated blocking (careful!) - Maintain runbooks for common attack scenarios.

10.4 Monitoring and Maintenance

1. Health Monitoring - Monitor Snort process (ensure it's running) - Check disk space regularly - Monitor packet drop statistics - Set up alerts for Snort failures.

2. Regular Maintenance - Review and tune rules monthly - Analyze alert trends to identify new threats - Update Snort version quarterly - Test configuration changes before production deployment.

3. Documentation - Document custom rules and their purpose - Maintain network diagrams showing Snort placement - Keep a change log of configuration modifications.

10.5 Integration with Security Stack

1. SIEM Integration - Forward Snort alerts to SIEM (Splunk, ELK, etc.) - Correlate Snort alerts with other security events - Create dashboards for real-time monitoring.

2. Firewall Integration - Automated blocking of malicious IPs - Use Guardian or fail2ban for integration - Careful with false positives causing legitimate blocks.

3. Threat Intelligence - Integrate with threat intelligence feeds - Create rules based on IOCs (Indicators of Compromise) - Share threat intelligence with the community.

11. CONCLUSION

Key Takeaways

Snort is a powerful, flexible, and widely adopted intrusion detection and prevention system that plays a crucial role in network security. This comprehensive guide has covered:

1. Understanding Snort - Open-source network-based IDS/IPS created by Martin Roesch - Three operational modes: Sniffer, Packet Logger, and NIDS/IPS - Signature-based detection using flexible rule engine - Modular architecture with preprocessors and output plugins.

2. Installation and Configuration - Cross-platform support (Linux, Windows, Unix, BSD) - Straightforward installation on major operating systems - Configuration through snort.conf file - Multiple output options for alerts and logging.

3. Operational Capabilities - Real-time network traffic analysis - Protocol decoding and inspection - Attack detection across various threat categories - Flexible rule-based detection engine - Integration with security ecosystem.

4. Practical Applications - Port scan detection - SQL injection prevention - Malware C2 communication detection - DDoS attack identification - SSH brute force monitoring - Custom threat detection.

5. Best Practices - Strategic network placement - Regular rule updates and tuning - Performance optimization - Integration with SIEM and firewalls - Proper alert management and response.