COMPARATIVE ANALYSIS OF INTRUSION PREVENTION SYSTEM

BY

MD. NAZRUL ISLAM

ID: 151-15-5503

This Thesis Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Dr. Sheak Rashed Haider Noori

Associate Professor and Associate Head

Department of CSE

Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

DECEMBER 2018

APPROVAL

This Thesis titled **"Comparative Analysis of Intrusion Prevention System**", submitted by Md. Nazrul Islam, ID No: 151-15-5503 to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 11th December 2018

BOARD OF EXAMINERS

____

Dr. Syed Akhter Hossain **Chairman**
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

____

Narayan Ranjan Chakraborty **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

____

Md. Tarek Habib **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

____

Dr. Mohammad Shorif Uddin **External Examiner**
Professor
Department of Computer Science and Engineering
Jahangirnagar University

DECLARATION

I hereby declare that, this thesis has been done by me under the supervision of **Dr. Sheak Rashed Haider Noori, Associate Professor and Associate Head, Department of CSE** Daffodil International University. I also declare that neither this thesis nor any part of this thesis has been submitted elsewhere for award of any degree or diploma.

Supervised by:

____

Dr. Sheak Rashed Haider Noori
Associate Professor and Associate Head
Department of CSE
Daffodil International University

Submitted by:

____

Md. Nazrul Islam
ID: 151-15-5503
Department of CSE
Daffodil International University

ACKNOWLEDGEMENT

First I express my heartiest thanks and gratefulness to almighty Allah for His divine blessing makes me possible to complete this thesis successfully.

I fell grateful to and wish my profound my indebtedness to **Dr. Sheak Rashed Haider Noori, Associate Professor and Associate Head**, Department of CSE Daffodil International University, Dhaka. Deep Knowledge & keen interest of my supervisor in the field of network security influenced me to carry out this. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior draft and correcting them at all stage have made it possible to complete this.

I would like to express my heartiest gratitude to **Dr. Sheak Rashed Haider Noori, Associate Professor and Associate Head**, for his kind help to finish my thesis and also to other faculty member and the staff of CSE department of Daffodil International University.

I would like to thank my entire course mate in Daffodil International University, who took part in this discuss while completing the course work.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

ABSTRACT

This thesis is on "**Comparative Analysis of Intrusion Prevention System.**". The main domain of this thesis is network security. Intrusion Prevention System is a well-known and important part of network security. Intrusion Prevention System provides critical infrastructures security by preventing intrusions in the network and computer systems. The aim of this thesis is to learn more about Intrusion Prevention System, know their implementation procedures, knowledge gathering on deep level packet inspection and find out the performance of most common open Intrusion Prevention Systems. In this study two most common Intrusion Prevention System (Snort and Suricata) is used to learn and experiment the performance on latest intrusion dataset named CICIDS2017. Performance are measured based on the CPU Utilization, Packet Processing speed, and on detection and prevention accuracy rate. Detection and prevention accuracy is measured using data mining techniques where different Machine Learning algorithms has been used.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

The worldwide system named due to the fact the Internet has become a part and parcel of our existence. Consistently peoples have interaction with the Internet and plenty of them link their life with it. The Internet carried out numerous parts of life for example banking, shopping, learning, installments, business, payments and transactions. In this term due to the rapid growth of computer networks during the past two decades security has turn into a critical issue for the Internet. This quick growth has exposed computer networks to an increasing number of security threats. There are a variety number of security threats such as worms, viruses, adware, malware and approach to hack something on Internet developing every day. The threats don't seem to be solely to computers and hardware that we tend to connect with the Internet, however to the information and knowledge that resides among that infrastructure.

There are a lot of diverse ways and technique to increase the security of network and computer systems. However, in this study, I focus on Intrusion Prevention Systems (IPS). IPS are a hardware device or software system of network and computer security which detect and prevent intrusive activity both from insider network and outsider network. They cowl the large part of network security which allow us to manage major aspects. The aim of the Intrusion Prevention System is to prevent different kinds of intrusions and activities that are very dangerous for network and computer systems. Intrusions can be an attack against privilege escalation, unauthorized access to various sensitive files, network attacks against different critical vulnerable services, actions of harmful malware can be Trojans, viruses and worms. In general, IPS are placed either after or before the placement of firewall device in an organized network. In Figure 1.1, indicates general placement of Intrusion Prevention System in a pictorial format.

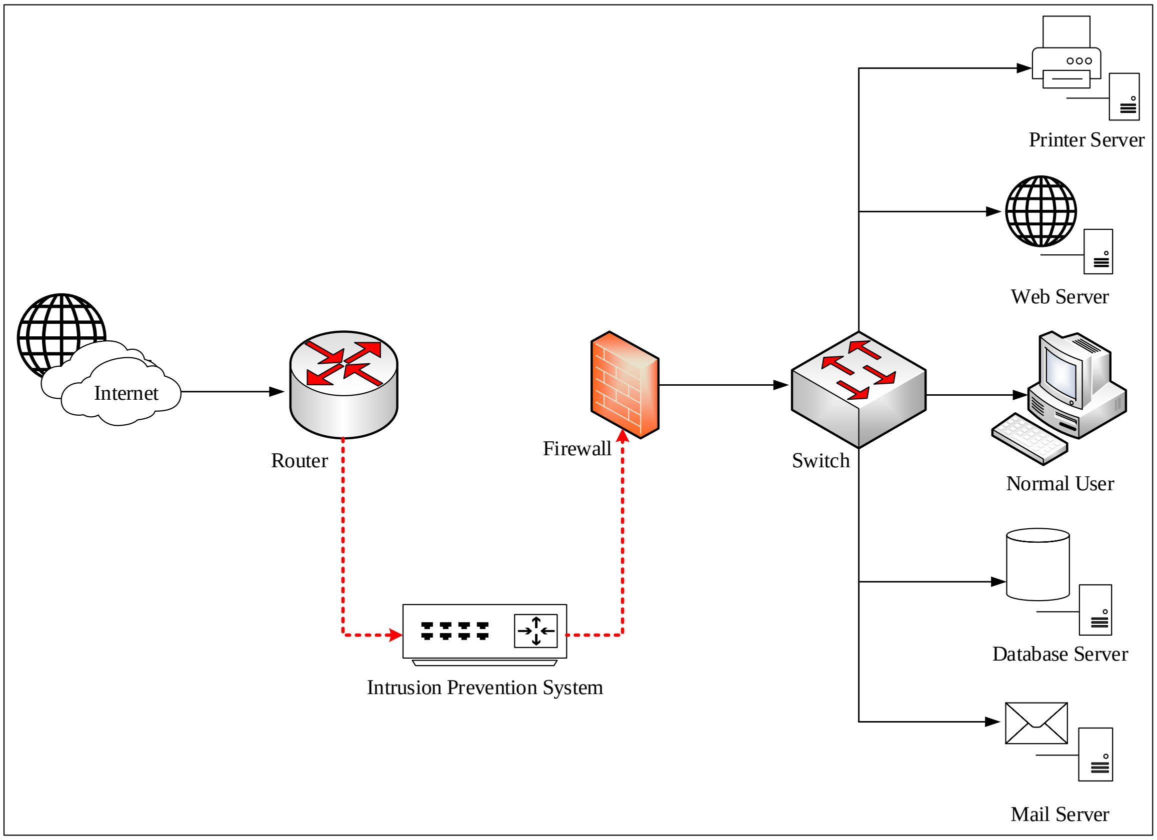

Figure 1.1: Intrusion Prevention System Placement

There are many valid ways to classify the Intrusion Prevention Systems. Scarfone et. al., [1] have used three types of IPS classification in a research. These are i) Host-based Intrusion Prevention Systems (HIPS), ii) Network-based Intrusion Prevention System (NIPS), and iii) Wireless Intrusion Prevention System (WIPS). Purpose of these IPS are given below.

- **Host-based Intrusion Prevention System (HIPS):** Host-based IPS detect and prevent intrusions that are generally affect end user. These type of IPS analyze traffics those are communicate with between the insider program and the internet or external network of a host. Host-based IPS must be installed to a host to make it workable.
- **Network-based Intrusion Prevention System (NIPS):** Network-based IPS monitors the network traffic and prevent suspicious data stream or packet. NIPS are work as a router also. All the traffics are passed over NIPS in network.

- **Wireless Intrusion Prevention Systems (WIPS):** Wireless IPS monitor actions in the wireless networks. Generally, it prevent the network from man-in-the-middle attacks, MAC address spoofing, wrong configured wireless access points and so on.

This study is conducted based on the performance, and prevention accuracy comparison of two most famous free and open source Network-based Intrusion Prevention called Snort and Suricata. These NIPS are helping the network security community way better.

1.1.1 Snort

The Snort IDS and IPS system became a worldwide famous feature to protect network. Snort is built based on five import unique module. There are i) Packet capture, ii) Packet Decoder, iii) Preprocessor, iv) Detection Engine and v) Output module.

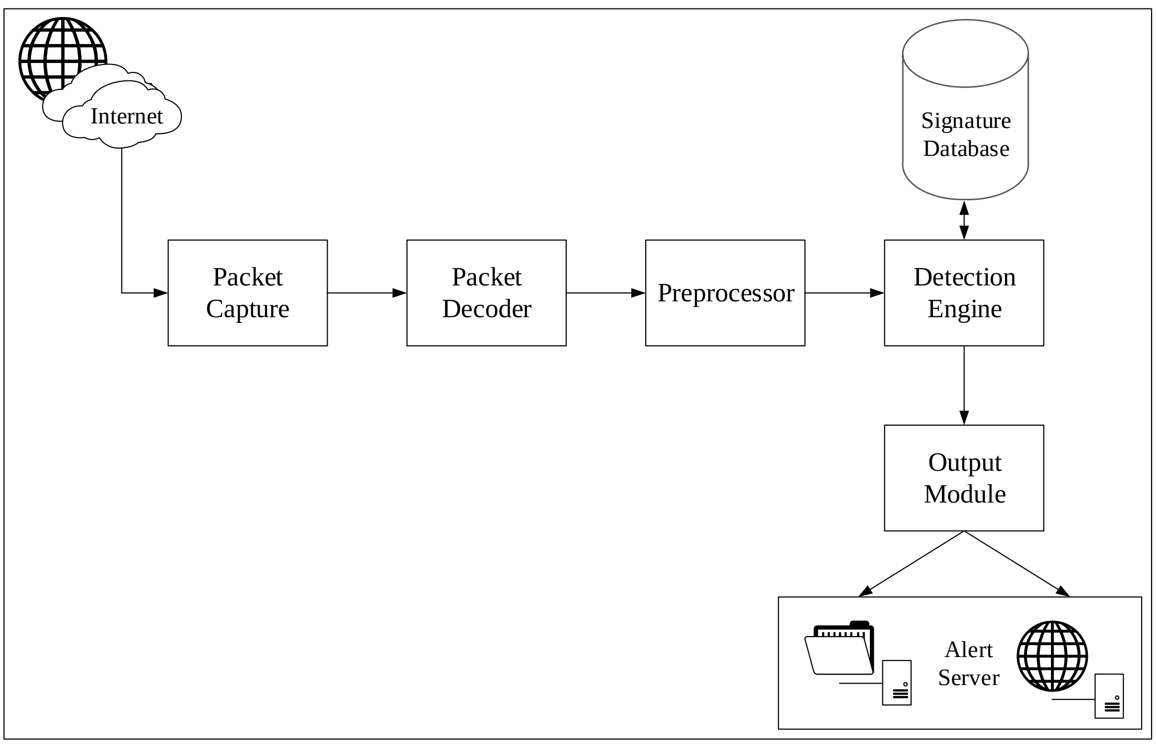

Figure 1.2: Architecture of Snort

Packet capture: In this module packets are captured using sniffer in the backend part of Snort. This module is responsible for capture the data transmitted over the network. For subsequent transmission to decoder with the help of a library named Data Acquisition (DAQ), it has done it job.

Packet decoder: Packet decoder deals with parsing the headers of captured packets. Decoding human readable information from raw packet by parsing them, the analysis of TCP flags, except for certain protocols of further analysis, finding anomalies and deviations from the RFC, and other similar work packet decoder done its job.

Preprocessor: The preprocessors of Snort are intended to do in-depth analysis and normalization protocols at each layer of TCP/IP model. Amongst most used preprocessor in Snort frag3, stream5, http_inspect, RPC2, sfPortscan are very popular. To work with fragmented traffic frag3 preprocessor is used. Similarly, for the reconstruction of TCP flows stream5, for normalizing HTTP traffic http_inspect preprocessor are used. To detect port scans in network sfPortscan preprocessor is used in Snort. And decoders for different types of protocol such as SSH, IMAP, SMTP, FTP, SIP, Telnet are also used in this module.

Detection engine: Detection engine of Snort consist of two parts. Of them one part is used to collect various signature from its database, and another is responsible for deep-level inspection where it match the signatures with the real-time network traffic.

Output module: Output module is responsible for alert to the administrator based on the detection of attacks, for logging the attacks, capture the network traffic for further analysis as pcap format and writing them in binary format on the base machine using Unified2.

1.1.2 Suricata

Suricata is a referred to as a free and open source, advanced, robust and fast network intrusion detection and prevention engine. It is capable of real-time intrusion detection and inline prevention (IDPS), monitoring network security and offline processing of captured pcap files. Suricata analyze network traffic with its powerful and sizable rules and signature language, and has effective Lua scripting support for the detection of complicated modern intrusion.

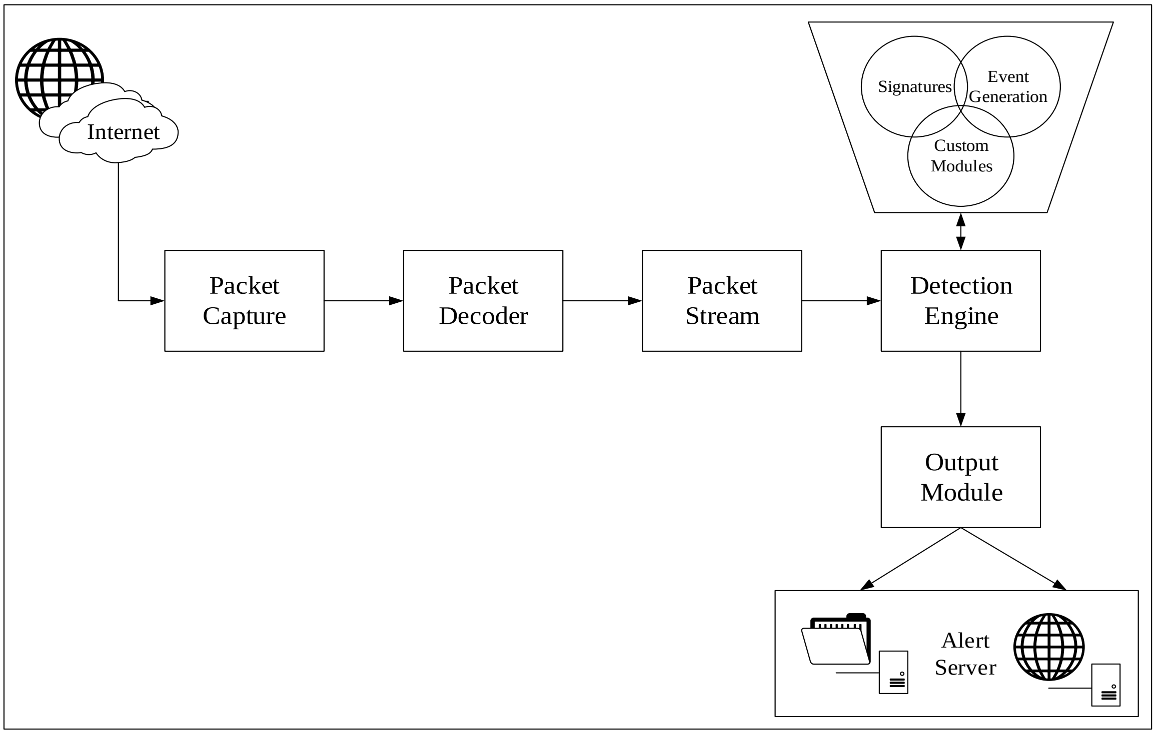

Figure 1.3: Architecture of Suricata

Architecture of Suricata is almost similar to Snort but has difference in some parts of its module. Suricata use PF_RING a high-speed packet processing framework which a new type of network socket that dramatically improves the packet capture speed [2] for capturing packet from the internet or other source. Packet stream is like preprocessor which is basically deals with network streams. Detection engine of Suricata support multi-threading techniques and that’s why its processing speed is way better.

1.2 Motivation

Security threats are an alarming issue for the modern world. Attacks which are success in their motive called intrusion. In recent years from various study it is said that, cyber threats are increasing rapidly with modern techniques and tactics. Due to the increasing threat Cyber Crime is a big issue that hampers regular activity of our society and our systems.

Intellectual Property Theft and Cybercrime become commonplace during the 2000s. So, to protect our critical infrastructures, network and computer system necessary steps should be taken. Intrusion Prevention Systems are a solution to protect network and computer system from different threats and attack.

1.3 Research Questions

Research question of this study are as follows:

- ► Are Network-based IPSs are capable to protect network and computer systems, critical infrastructures from modern intrusion?
- ► Does IPS are enough to secure todays networks?
- ► Does Snort with single-thread processing capability better than Suricata?
- ► Does Suricata's CPU Utilization better than Snort?

1.4 Expected Output

From this study is expected to learn deep level packet analysis, know how to analyze real time network traffic in a structured way with well-known solutions. It is a great way to learn about intrusion, cyber threat, detection and prevention techniques and so on. Expected outcome would be identify the best solution to prevent modern threat in real world. Another would be to know about the way to secure critical infrastructures.

1.5 Thesis Layout

This study contains of six chapter in which have described the whole of the thesis. Thesis layout consists of the preview of all the chapters.

1. Chapter one covers introduction, motivation, research questions, expected output and thesis layout of the study.
2. Related research work have discussed in Chapter two.
3. Chapter three research methodology includes introduction, research design, lab architecture, and dataset collection procedure.

4. Chapter four of this study discussed on requirement analysis, requirement installation, and requirement configuration for the success of the experiment.
5. Experimental results on CPU Utilization, experimental results on CICIDS2017 dataset, descriptive analysis and result comparison are discussed in Chapter 5.
6. Finally, in chapter six have discussed about conclusion and future study.

CHAPTER 2

BACKGROUND

2.1 Related Works

Intrusion prevention has become significantly more important due to, with the increase in difficulty and regularity of Internet threats in recent years. Various tech companies and organizations working to develop the equipment and produces different product including open source and proprietary. One amongst the most well-known and widespread open-source intrusion detection and prevention system is Snort which works on signature-based detection and prevention. Snort was maintained by SourceFire Company, now acquired by Cisco Systems Limited. Martin Roesch developed Snort in 1998. It was mainly developed to monitor the network packet of layer 7 which is application layer of OSI model. But nowadays it is used in the backend part of most of the next-generation firewall and intrusion prevention systems. In 2009, after a decade another open source community named Open Information Security Foundation (OISF) announced another signature-based intrusion detection and prevention system called Suricata. The signification difference between Snort and Suricata is in their internal architecture. The advancement in Suricata is it's able to execute native multithreaded processes. Many research has been done in terms of testing and comparing different type intrusion prevention system in recent years. Researcher Sergey identified pros and cons of Snort and Security Onion in his thesis [3]. Ahmad Iftikhar, et al. recognized intrusion detection approached in their research with comparison [4]. Study on intrusion detection and prevention system are huge. Researcher B.Santos Kumar et al., identified type and prevention of intrusion detection system in their research [5]. A great thesis on analysis and comparison of Snort and Suricata was published in 2011 by Eugene [6]. Also many article has been published focused on intrusion detection and prevention system. Due to the rapid growth of Internet, need to be ensure its security first. And Intrusion prevention system can be a great technology in terms of its solution.

2.2 Research Challenges

Challenges of the study relies on the experiment part.

- Resources are limited and most of them not rich.
- A strong background on networking and OSI layer is must.
- In-depth knowledge on networking packet architecture is necessary to deploy the experiment.
- Previous basic knowledge on Intrusion Detection and Prevention Systems is also necessary.
- Hand-on working knowledge and experience on Linux is a must to fulfil the goal of this study.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

The research methodology discussed how the research has been done to complete the thesis. In-depth study on Intrusion Prevention System has been done prior to the experiment. This chapter includes the research design, lab architecture and dataset collection procedure and reason behind choosing the dataset.

3.2 Research Design

Research design shown in Figure 3.1 indicates how the whole research has been conducted.

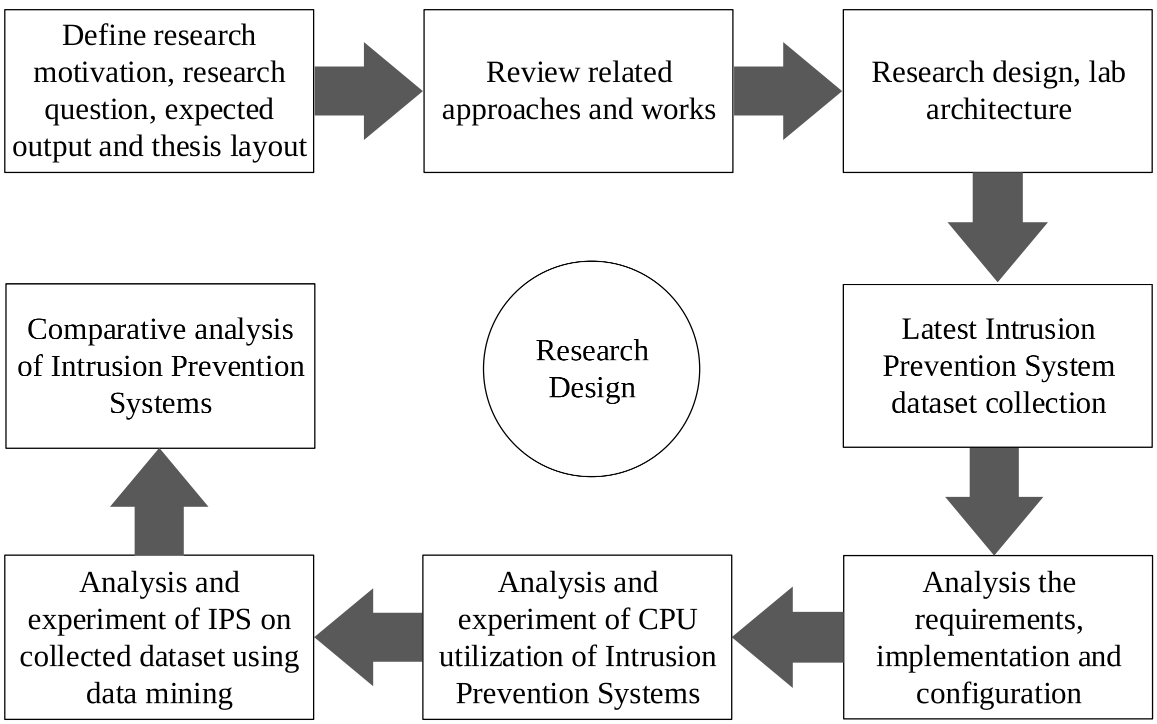

Figure 3.1: Research design

3.3 Lab Architecture

Lab architecture includes 4 PC (Attacker, Normal User, Victim PC1 and Victim PC2), 1 network switch, 1 network router PC (Intrusion Prevention System) and 2 logical Class B private networks which include 172.16.10.0/24 and 172.16.20.0/24 where each network hosts are connected with the switch and IPS router. Figure 3.2 shows the lab architecture of this research.

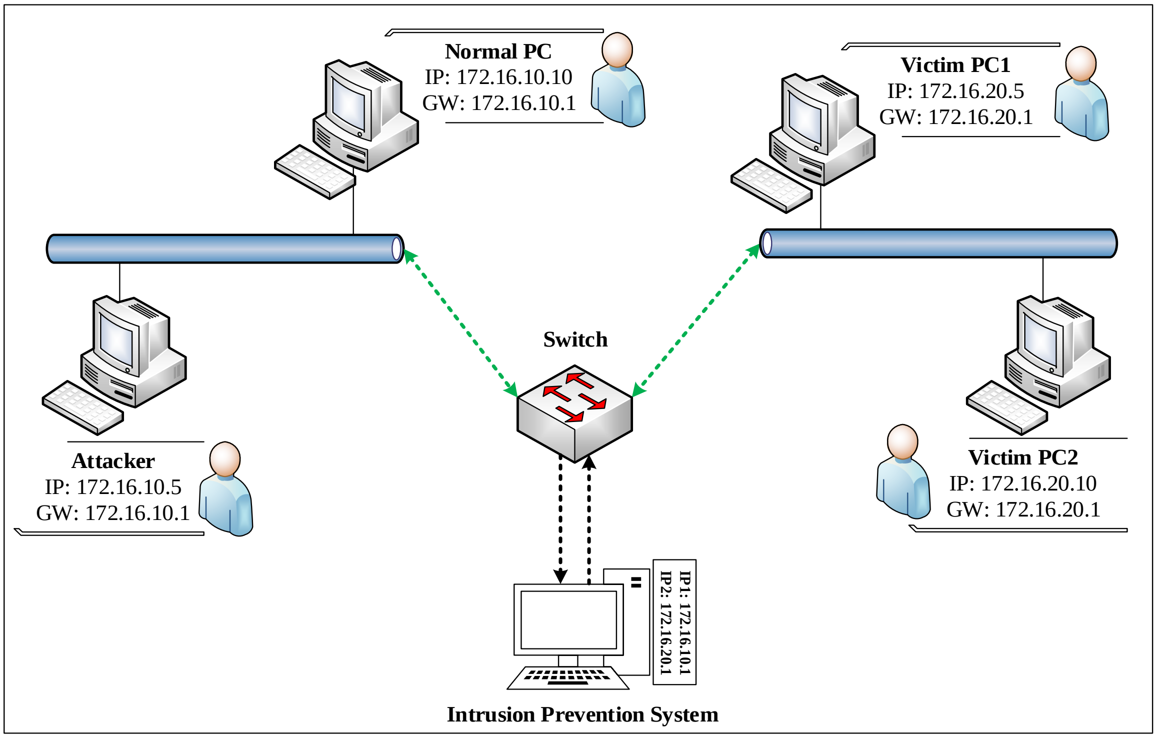

Figure 3.2: Lab Architecture

3.4 Dataset Collection

The experiment of this study has been conducted on one of the latest IPS dataset named CICIDS2017 collected from Canadian Institute for Cybersecurity (CIC) which is not publicly available on the Internet. Many researchers has been used this dataset for inventive research purpose. Among many researchers, Gobinath Loganath used this dataset for Real-time Intrusion Detection purpose [7]. Darya Lavrova et. al., also used CICIDS2017 dataset for "Wavelet-analysis of network traffic time-series for detection of attacks on digital production infrastructures" [8] purpose. The CICIDS2017 dataset contains benign and the most up-to-date common attacks, which resembles the true real-world data (PCAPs). Attack diversity and count of flows can be found on Table 4.1. This dataset also includes the results of the network traffic analysis using CICFlowMeter with labeled flows based on the time stamp, source and destination IPs, source and destination ports, protocols and attack (CSV files) [9]. That's why CICIDS2017 dataset has been used in this study for experiment.

Table 3.1: Attack Types and flows in CICIDS2017

#	Attack Type	Total flow
1	Heartbleed	11
2	Web Attack: SQL Injection	21
3	Infiltration	36
4	Web Attack: XSS	652
5	Web Attack: Brute Force	1507
6	Botnet	1966
7	DoS Slowhttptest	5499
8	DoS Slowloris	5796
9	SSH Patator	5897
10	FTP Patator	7938
11	DoS GoldenEye	10293
12	DDoS	41835
13	Port Scan	158930
14	DoS Hulk	231073
15	BENIGN	2358036

Next chapter will discuss the requirement analysis, installation and configuration.

CHAPTER 4

REQUIREMENT ANALYSIS, INSTALLATION AND CONFIGURATION

4.1 Introduction

This chapter discussed on requirement analysis, installation and environment configuration for the experiment. Requirement analysis includes hardware and software requirements which are the most important part for the experiment of this research.

4.2 Requirement analysis

Both hardware and software requirements are necessary to study the experiment. Table 4.1 shows the overview of both hardware and software requirements. Requirements are needed to be ready before the experiment. PCs of victim network and Attacker network both have 4GB of RAM. Intrusion Prevention System has 4 GB of RAM. After successful implementation of hardware requirements, software requirements was implemented where different software were installed and configured for the experiment.

Table 4.1: Overview of hardware and software requirements

Hardware Requirements				**Software Requirements**
	Machine	**Operating System**	**IP Address**	
Victim Network	Victim PC1 Victim PC2	Windows 10 x64 Ubuntu 16.04.5	172.16.20.5 172.16.20.10	XAMPP, Mysql, Apache2, DVWA
	Intrusion Prevention System (IPS)	Ubuntu 16.04.5	172.16.10.1 172.16.20.1	Snort, DAQ, Barnyard, Pulledpork, Mysql,

				Suricata, WebSnort, Wireshark, Atop
Attacker Network	Attacker Normal User	Kali Linux Windows 10 x64	172.16.10.5 172.16.10.10	Tcpreplay, Wireshark, Nmap, Atop

4.3 Requirements Installation

To make the environment ready for the experiment, firstly hardware requirements were setup properly. According to the lab architecture can be found on Chapter 3 Switch, Router and PCs were connected with necessary network cables. And two logical private network 172.16.10.0/24 and 172.16.20.0/24 has been configured and tested on Router and PCs prior to the installation of software requirements. It is mentioned that Internet connection was ensured to download necessary software for the experiment. Table 4.2 shows the specific version of software which were used in this research.

Table 4.2: Specific version of used software

Software	Version
XAMPP	7.2.10
MySQL	5.7.16
Apache2	2.4.34
DVWA	1.9
Snort	2.9.11.1
DAQ	2.0.6
Barnyard2	2-1.14
PulledPork	0.7.4
Suricata	4.0.5

Software	Version
Websnort	0.8
Wireshark	2.6.4
Atop	2.3.0
Tcpreplay	4.2.5
Nmap	7.70

4.3.1 Snort, DAQ, Barnyard2, PulledPork and WebSnort installation

Here for installing Snort, DAQ, Barnyard2, PulledPork and WebSnort have used an interactive automated script named Snorter_IPS.sh developed by Joan Bono along with one of the contributor named Md. Nazrul Islam [10]. This script was taken from open source platform GitHub and then modified. Function of the script includes-

```
function main(),
function update_upgrade(),
function nghttp2_install(),
function snort_install(),
function snort_edit(),
function snort_test(),
function barnyard2_ask(),
function pulledpork_ask() ,
function service_create(),
function websnort_ask(),
function last_steps(),
function system_reboot()
```

The script start from the function main() and step by step and install and configured NGHTTP2, Snort, DAQ, Barnyard, PulledPork and WebSnort along with their dependencies. Figure 4.1 shows nghttp2_install a function of Snorter_IPS.sh script which install and configure NGHTTP2. NGHTTP2 is necessary for Snort to run as IPS mode.

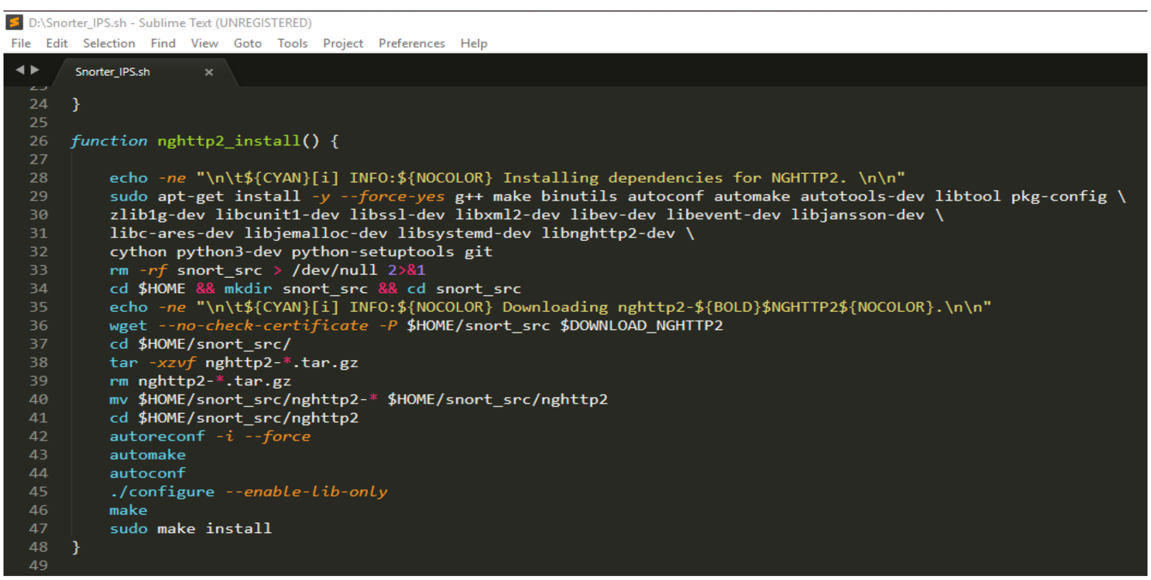

Figure 4.1: nghttp2_install function

Dependencies for Snorter_IPS.sh script:

```
jq, curl
```

Dependencies for NGHTTP2:

```
cython libxml2-dev python3-dev binutils libevent-dev git libev-dev libssl-dev libjansson-dev zlib1g-dev python-setuptools automake libjemalloc-dev pkg-config libnghttp2-dev libc-ares-dev autotools-dev g++ make autoconf libtool libcunit1-dev libsystemd-dev
```

Dependencies for Snort:

```
gcc libpcre3-dev libnghttp2-dev openssl libdnet bison zlib1g-dev libpcap-dev libssl-dev libdumbnet-dev flex
```

Dependencies for Barnyard2:

```
mysql-server libmysqlclient-dev mysql-client autoconf libtool libdnet checkinstall yagiuda libdnet-dev locate
```

Dependencies for PulledPork:

```
libcrypt-ssleay-perl liblwp-useragent-determined-perl
```

Figure 4.2 shows the running script where options –i indicates the interface of the machine and –o indicates the oinkcode (A unique code for snort individual user).

Figure 4.2: Running Snorter_IPS.sh script

Figure 4.3 shows that daq-2.0.6 and snort-2.9.11.1 is downloading automatically. It is mentioned that the script always find the latest version of required softwares. At the time of the experiment daq-2.0.6 and snort-2.9.11.1 was the latest version.

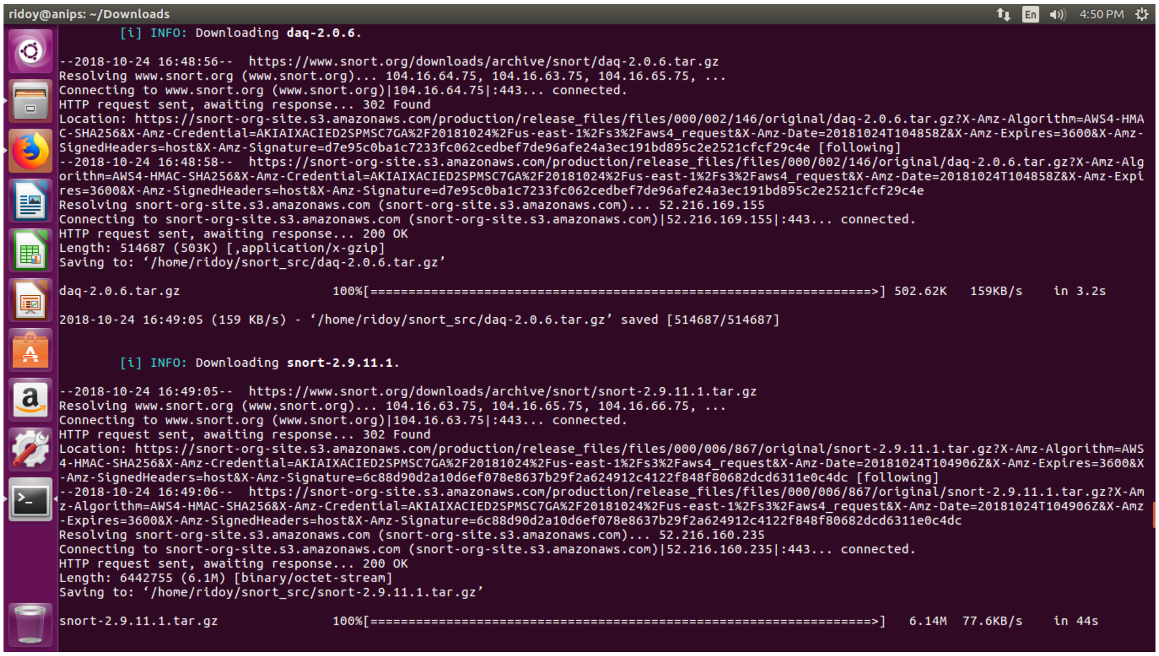

Figure 4.3: Downloading DAQ and Snort with automated script

To run Snort software as intrusion prevention mode nfqueue is necessary. So it must be needed to ensure that nfqueue is enable in DAQ module before compiling.

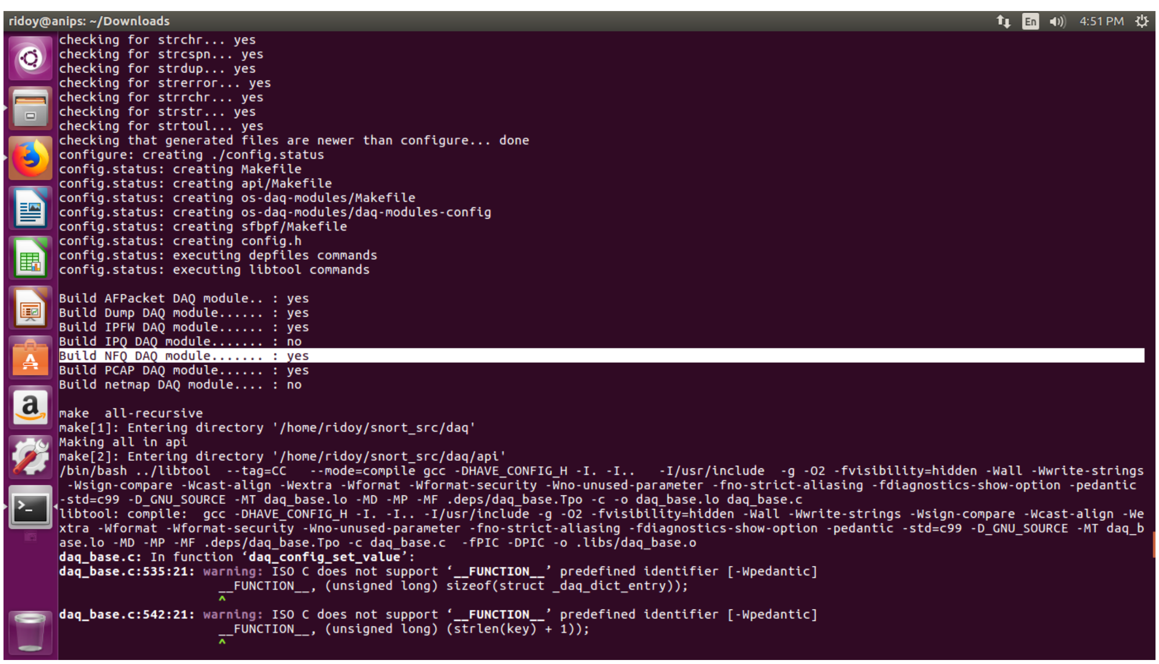

Figure 4.4: NFQ DAQ modules functions

Figure 4.4 shows DAQ modules where NFQ DAQ module is successfully enabled with yes notation. After downloading and DAQ and Snort, DAQ module was compiled before Snort installation. Because DAQ module is a must pre-requirement module of Snort software. Figure 4.5 indicates Snort successfully installed and configured. Snort installation can also be verified using the command-

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

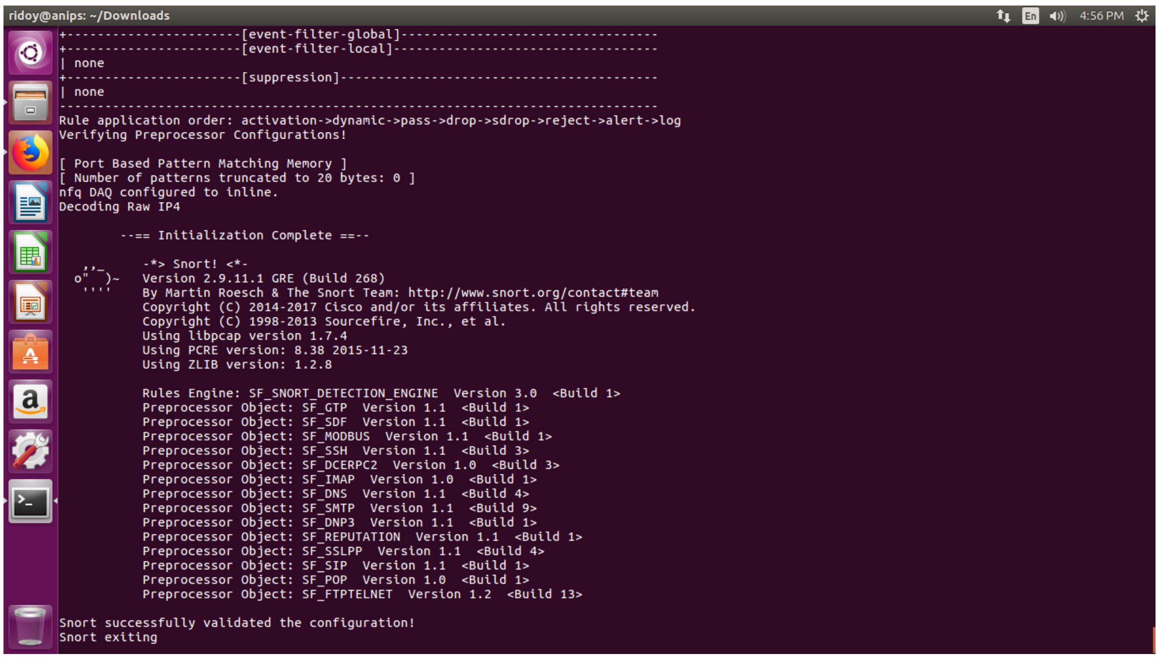

Figure 4.5: Snort

If the command returned successful indication without any error means that snort installation and its configuration is ok. Snort works based on detection and prevention rules. Everyday new intrusion are discovered and new rules are generated against them to prevent propagation in the world. So it is necessary to have the latest detection and prevention rules. Using pulledpork an open source software automatically download the latest rules every day at a scheduled time. Figure 4.6 shows pulledpork is downloading latest community, opensource, emerging-rules and snort-snapshot rules. Snort-snapshot rules are especially for snort user. These rules are identified and download based on the unique oinkcode.

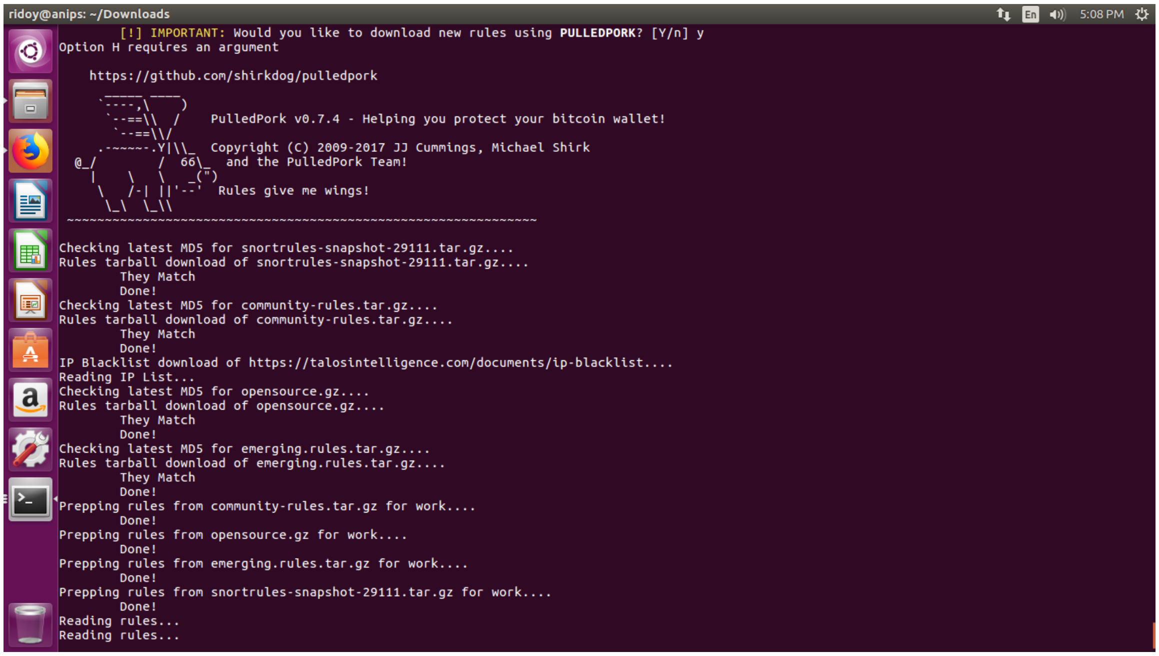

Figure 4.6: PulledPork download IPS rules automatically

Snort ruled are located at `/etc/snort/rules/` .When rules download was complete, websnort was installed and configured successfully.

4.3.2 Suricata installation

Suracata also known as open-source network based IPS developed by Open Information Security Foundation (OISF). Suricata also capable to capture real-time network packet and able to identify network intrusion and protect them using inline prevention mode. Suricata use NetfilterQueue a.k.a NFQ for performing inline functionality [11].

Suricata dependencies:

```
autoconf libjansson-dev libcap-ng-dev libjansson4 libnet1-dev libpcre3-dbg
libmagic-dev libtool libpcre3-dev automake libpcap-dev libyaml-dev zlib1g-dev
```

Suricata dependencies for IPS:

```
libnetfilter-queue-dev libnetfilter-queue1 libnfnetlink-dev
```

Figure 4.7 shows the installation process of Suricata dependencies. After that Suricata was downloaded, installed and configured.

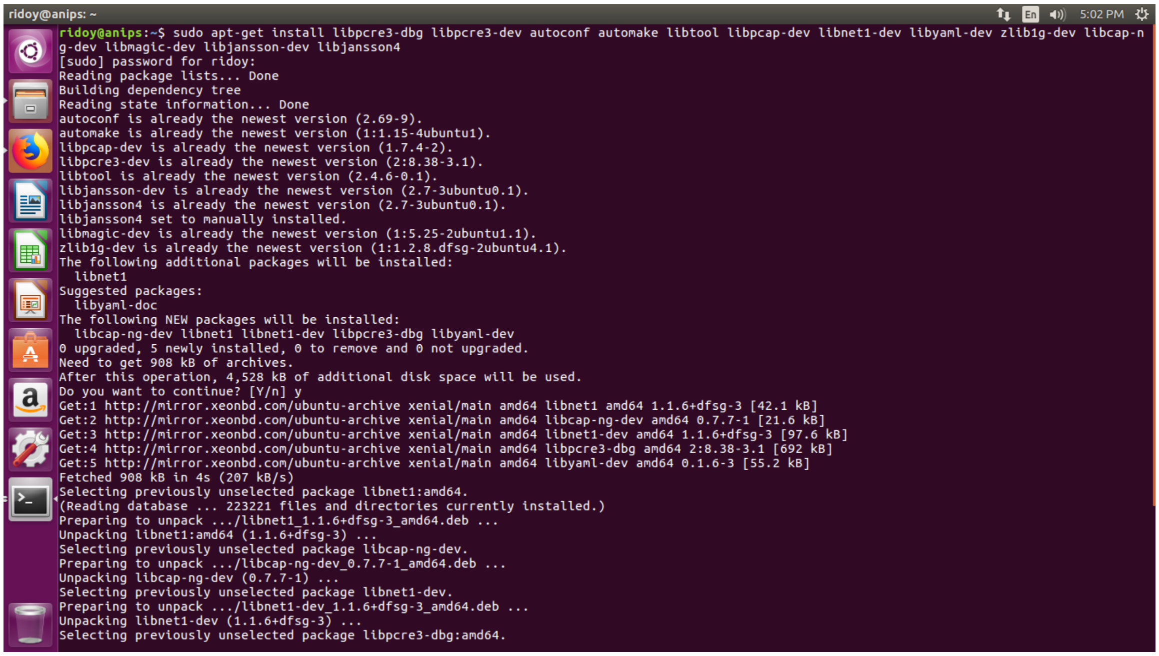

Figure 4.7: Installing Suricata dependencies

Suricata latest version (suricata-4.0.5) is downloaded using command-

```
wget https://www.openinfosecfoundation.org/download/suricata-current.tar.gz
```

After download tar file was extracted and Figure 4.8 show the insider installation files.

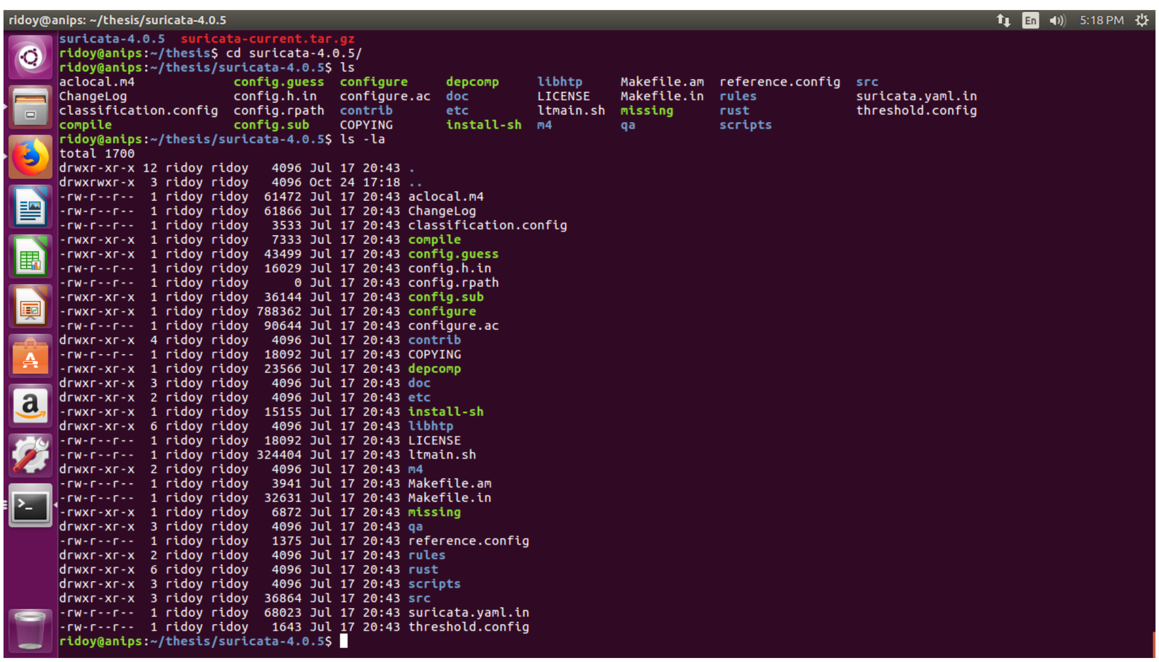

Figure 4.8: Suricata installation files

Before installation of Suricata software configure file is needed to be compiled first with enabling necessary module such as nfqueue module. The command –

```
sudo ./configure --enable-nfqueue --prefix=/usr --sysconfdir=/etc --localstatedir=/var
```

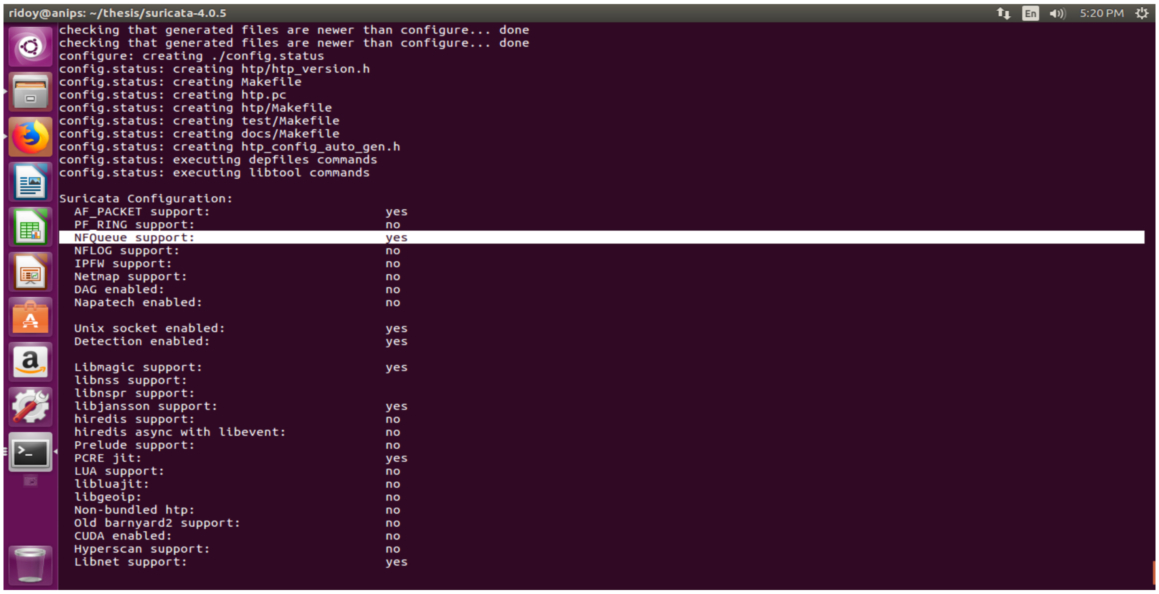

Figure 4.9: NFQueue support of Suricata

Figure 4.9 shows NFQueue module is enabled and supported notation as yes. So Suricata can be run as Intrusion Prevention mode. After that Suricata is installed and configured.

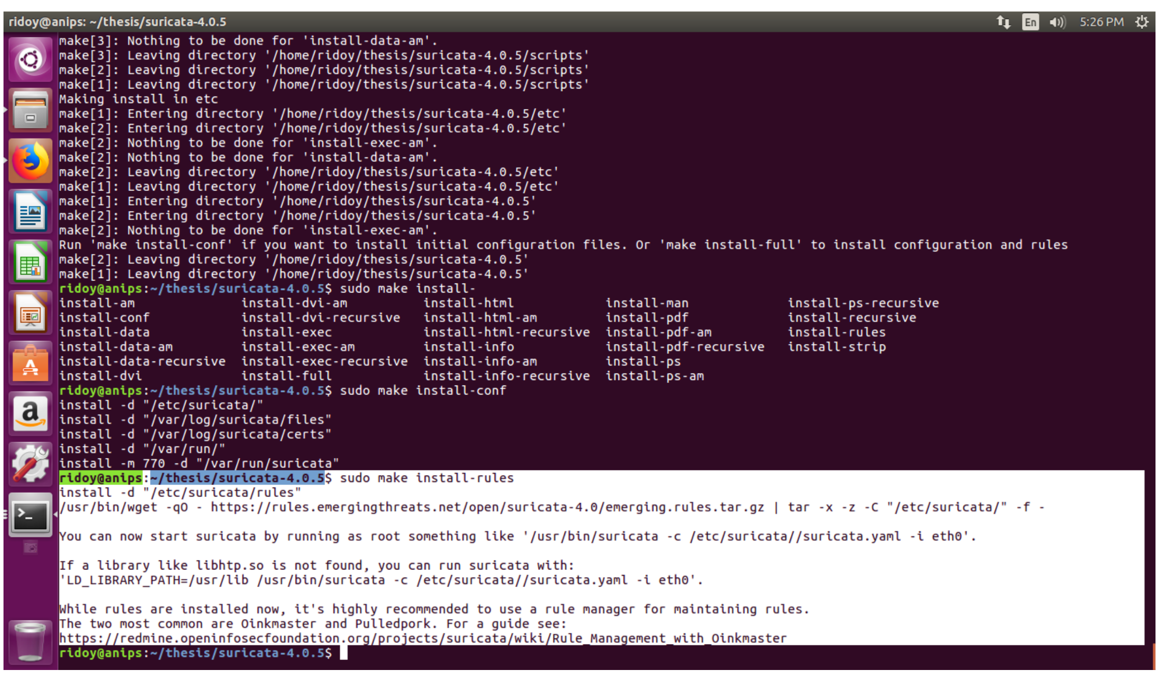

Figure 4.10 Suricata installation

Figure 4.10 shows the installation of Suricata where firstly installed configuration and then Suricata rules. Surcata detection and prevention rules are located at `/etc/suricata/rules` directory. Figure 4.11 shows Suricata detection and prevention rules.

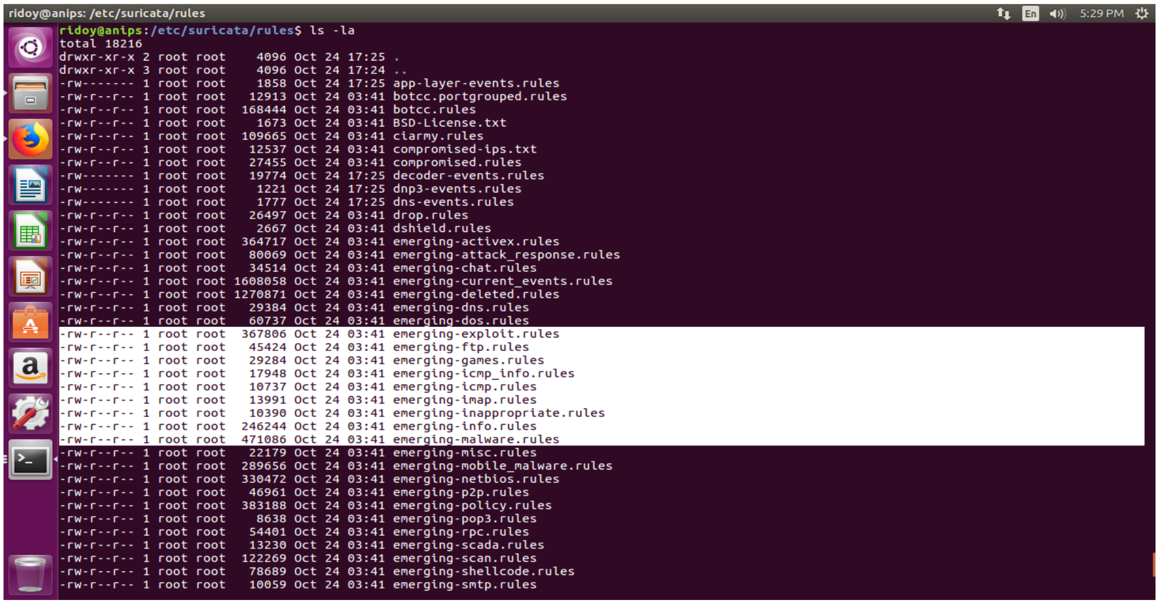

Figure 4.11: Suricata detection and prevention rules

Rules extension is .rule and can be open through any text editor software such as vi, vim, gedit, nano and so on. Figure 4.12 shows the rules for ICMP packet.

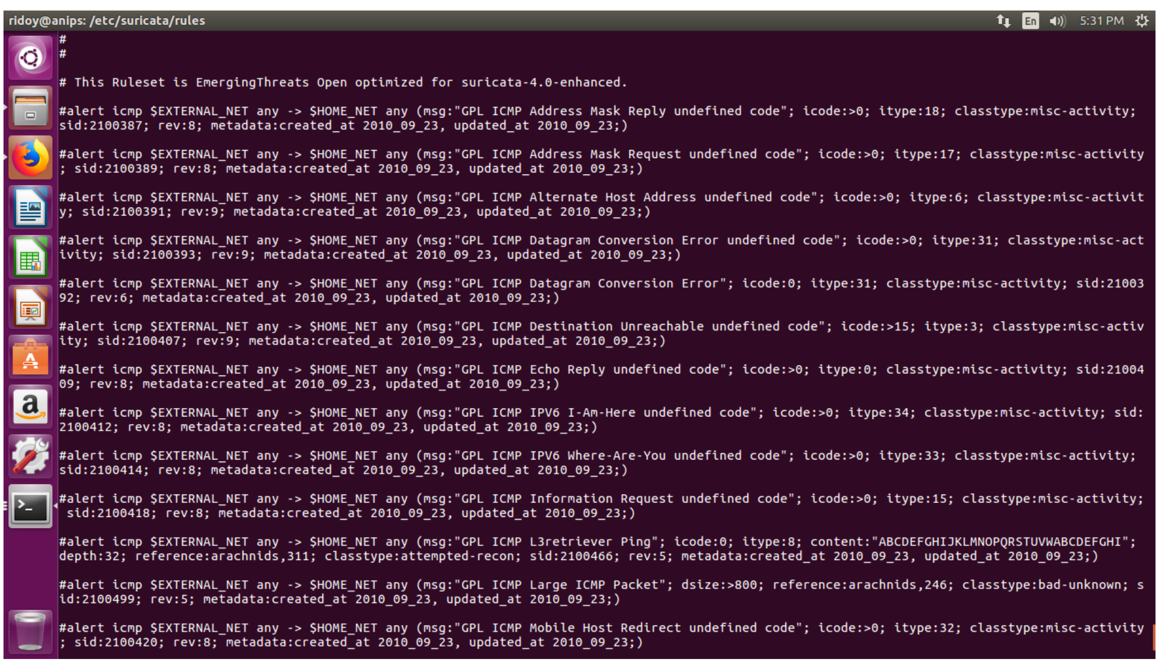

Figure 4.12: Suricata ICMP detection and prevention rules

While rules and other necessary configuration was complete, Suricata main configuration file was configured to make ready for run.

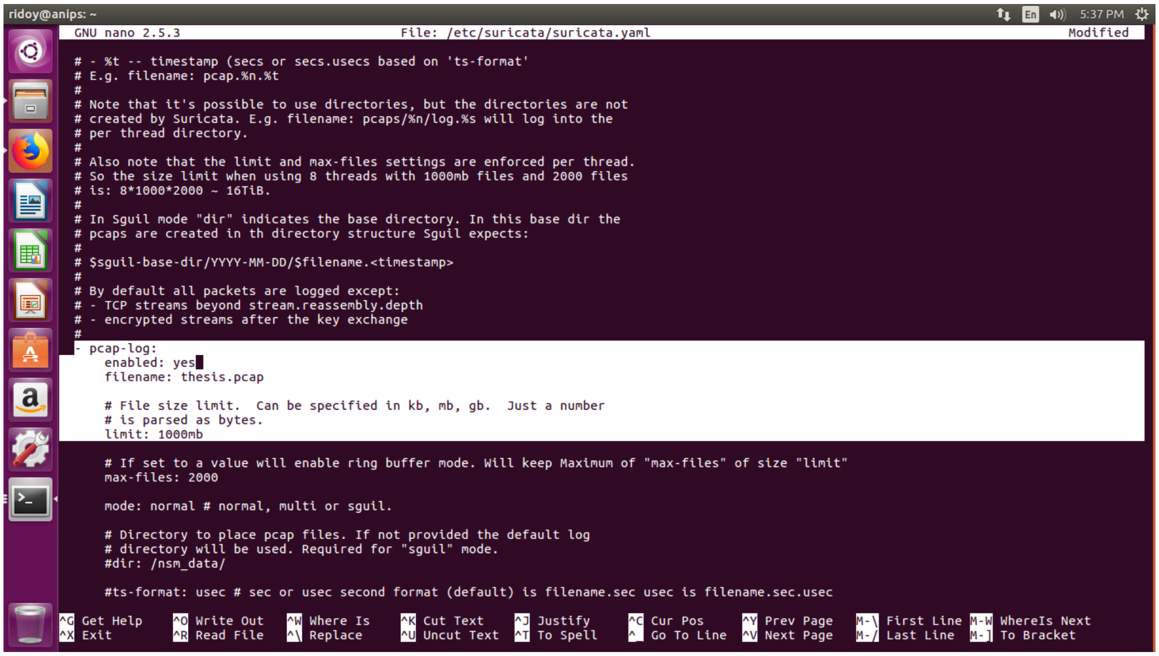

Figure 4.13: Suricata configuration

Suricata configuration file is located `/etc/suricata/suricata.yaml`. Configuration has several parts like network setup, output setup, and log setup and so on. Figure 4.15 indicates the Suricata configuration file where pcap-log is enabled with filename as thesis.pcap. In the next chapter experimental results will be performed and discussed.

CHAPTER 5

EXPERIMENTAL RESULTS AND DISCUSSION

5.1 Introduction

The experiment tested and compared with Snort and Suricata Intrusion Prevention system in performance and accuracy of detection and prevention in a real setup environment. Performance evaluated by measuring the percentage of memory usage, network usages and CPU Utilization in this experiment. Accuracy was measured and compared based on the generated alert of detection and prevention of each prevention system using machine learning and data mining technique on CICIDS2017 dataset.

5.2 Experimental Results on CPU Utilization

The experiment was conducted in two stage. Where in first stage Packet Processing and CPU Utilization of both Intrusion Prevention System (Snort and Suricata) was measured and calculated. And in another stage detection and prevention of Suricata and Snort was analyzed and measured. Packet processing was logged and calculated using Wireshark. From Wireshark packets I/O value was taken as a csv file. Figure 5.1 and Figure 5.2 shows the packets processing graphs of Suricata and Snort respectably. Suricata Intrusion Prevention System processed 351.70 packet/s on an average with a high value 4295 packet

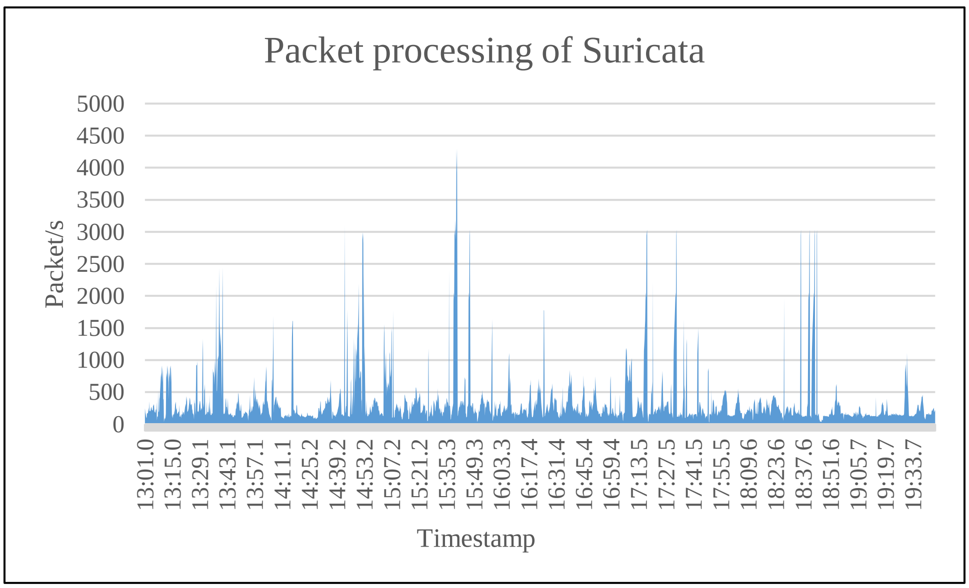

Figure 5.1: Suricata I/O of packet

in a second. Suricata was running for 6 minutes during the experiment. On the other hand Snort was running for 10 minutes shows in Figure 5.2 processed 327.22 packets/s on an average with a high value 5578 packet in a second during the experiment.

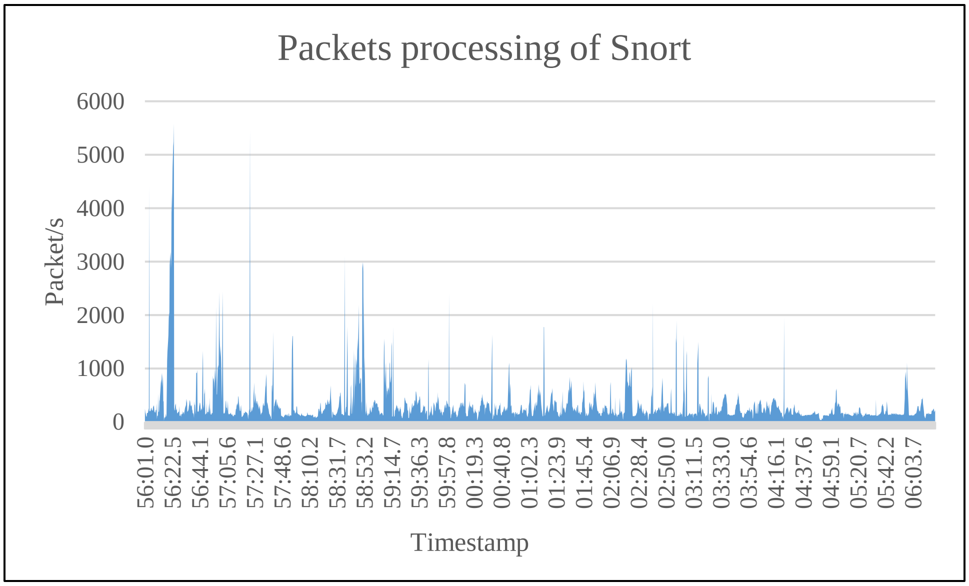

Figure 5.2: Snort I/O of packet

CPU Utilization of Suricata and Snort was measured using atop a tool that is capable of reporting the activity of all processes like CPU utilization, memory growth, disk utilization, priority, username, state, exit code and so on. Figure 5.3 shows the interface of atop tool.

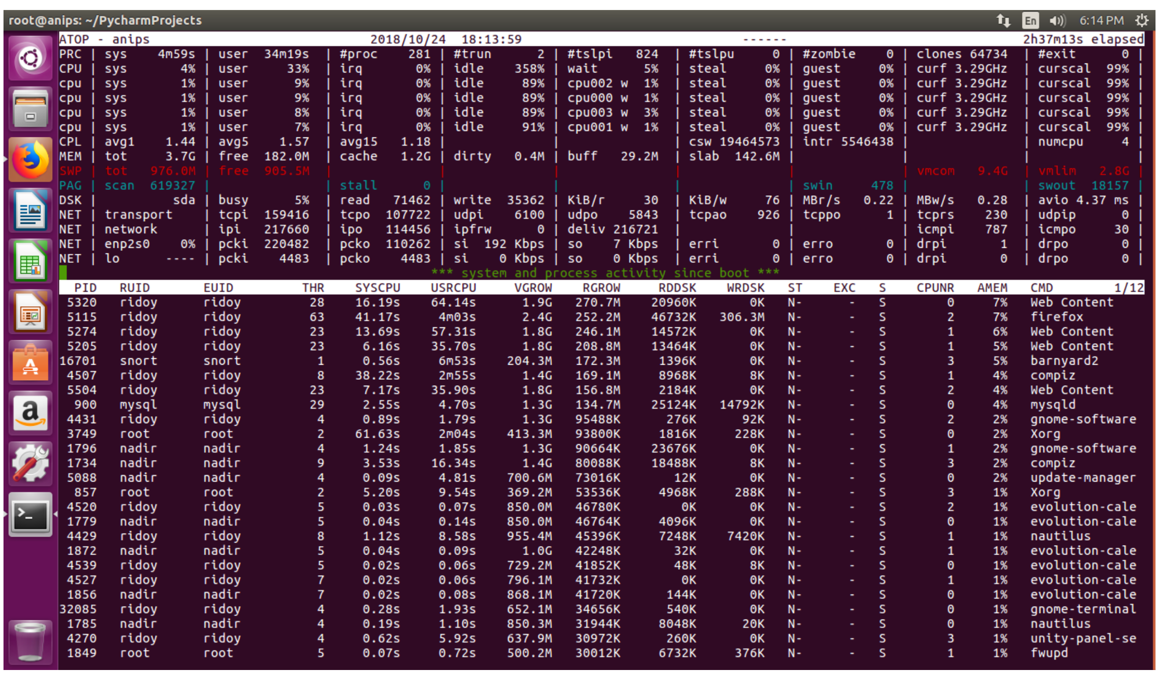

Figure 5.3: Interface of atop tool.

During the experiment CPU Utilization of Suricata was 44% on overage while Suricata ran for 6 minutes. Figure 5.4 indicates the utilization of CPU by Suricata Intrusion Prevention System.

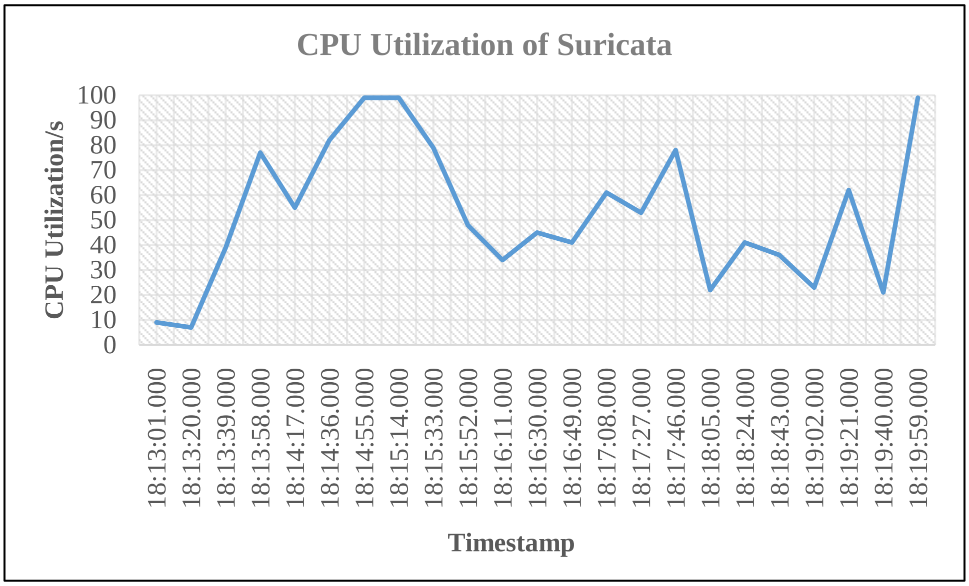

Figure 5.4: CPU Utilization of Suricata

On the other hand in Figure 5.5 shows the CPU Utilization of Snort where average CPU Utilization was 59% during Snort ran for 10 minutes.

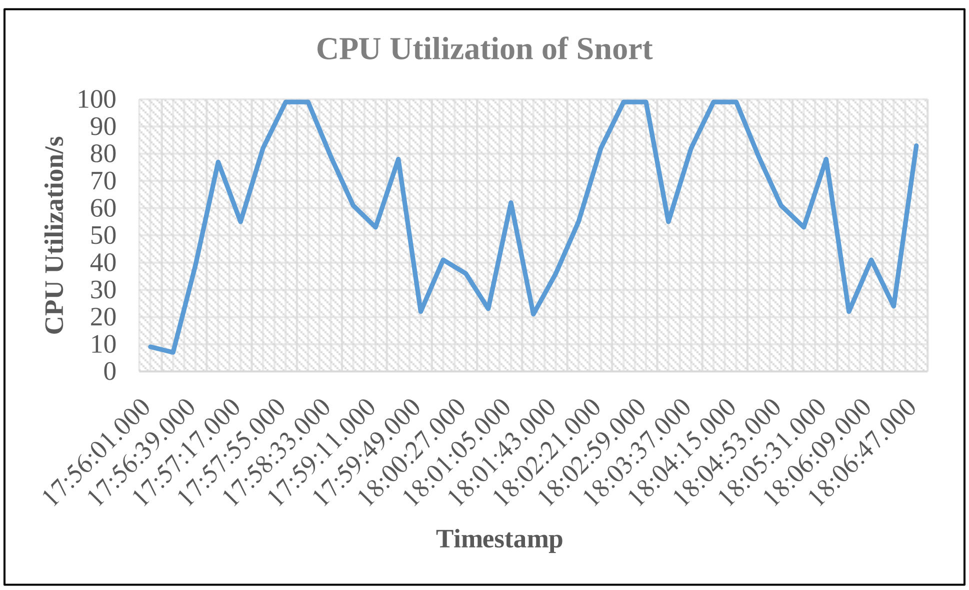

Figure 5.5: CPU Utilization of Snort

Deep level network packet analysis was conducted with most well-known and powerful packet analyzer tool named Wireshark [12]. With this powerful tool packet pattern has been identified and inspected the malicious and suspicious pattern on a packet. During the experiment most of the network packet was under TCP protocol and less was ICMP and UDP, DNS, HTTP and other protocols network packet. It was identified that same packet was sent from one network to another network during the attacks in several times. And most of them were fragmented and aimed to make denial-of-service of the victim server. Figure 5.6 indicates the UDP packet analysis using Wireshark network packet analyzer tool.

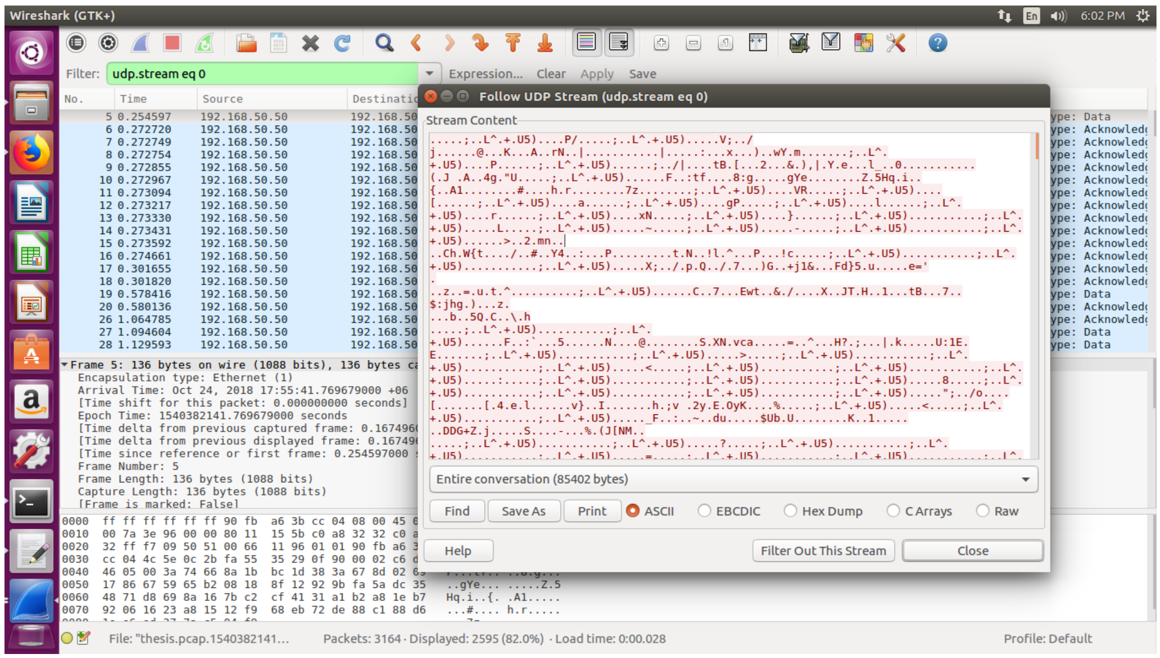

Figure 5.6: Analysis of UDP packet using Wireshark

At the same time during the experiment while attacks were launched from attacker network to victim network both Intrusion Prevention System Snort and Suricata generated prevention notification based on their rules against malicious and suspicious packet called intrusion. Suricata generated alert against 31427 enabled rules and they were downloaded and configured with PulledPork during Suricata installation. Then again Snort were generated alerts against 29471 enabled rules and also downloaded via PulledPork during the installation of Snort. After that generated logs of Suricata and Snort were collected for calculating their accuracy and performance using machine learning algorithms and data mining techniques.

5.3 Experimental Results on CICIDS2017 dataset

Data mining techniques was applied to calculate the prevention accuracy using five machine learning algorithm J48, IBk, MLP (Multilayer Perceptron), BayesNet and Naïve Bayes on 810 data for Suricata and 673 data for Snort of CICIDS2017 Intrusion Prevention System dataset. Table 5.1 indicates the results for Suricata and it is seen that overall classification accuracy of five machine learning algorithm J48, IBk, MLP, BayesNet and Naïve Bayes, J48 is performs better with 97.65% overall classification accuracy.

Table 5.1: Experimental results of Suricata

Algorithms	TPR (%)	FPR (%)	FNR (%)	Pr. (%)	F-1 (%)	OA (%)
J48	97.70	0.50	2.30	97.80	97.60	97.65
IBk	92.70	1.30	7.30	93.00	92.70	92.71
MLP	91.00	3.60	9.00	90.80	90.60	90.99
BayesNet	82.20	1.70	17.80	85.60	82.70	82.22
Naïve Bayes	68.10	3.60	31.90	84.50	72.10	68.15

**** TPR = True Positive Rate, FPR = False Positive Rate, FNR = False Negative Rate, Pr. = Precision, F-1 = F-measure, OA = Overall Accuracy [13]*

From the Table 5.2 show results for Snort and among five machine learning algorithm J48 classification accuracy is better with 97.33% accuracy.

Table 5.2: Experimental results of Snort

Algorithms	TPR (%)	FPR (%)	FNR (%)	Pr. (%)	F-1 (%)	OA (%)
J48	97.30	0.40	2.30	97.30	97.20	97.33
IBk	93.20	1.10	7.30	93.30	93.10	93.16
MLP	90.30	1.30	9.00	90.70	89.90	90.34
BayesNet	85.90	1.60	17.80	86.10	85.10	85.88
Naïve Bayes	81.90	2.10	31.90	86.80	82.80	81.87

**** TPR = True Positive Rate, FPR = False Positive Rate, FNR = False Negative Rate, Pr. = Precision, F-1 = F-measure, OA = Overall Accuracy*

5.4 Descriptive Analysis and Results Comparison

From experimental result it is found that, Suricata processed 351.70 packet/s on an average in 6 minutes where Snort processed 327.22 packets/s on an average in 10 minutes. So, at this point Suricata performs better than Snort. In terms of CPU Utilization, Suricata used 44% CPU on an average where Snort CPU Utilization was 59% on an average. It is also identified that, in intrusion prevention part Overall accuracy of Suricata is slightly better than Snort. So, after the experiment on results it is proved that, in all cases Suricata performs better than Snort. It is also identified that Suricata perform well due to its multi-thread architectural design and multi-CPU affinity capability where Snort can deal with single-thread process. Figure 5.7 shows the multi-thread architectural design and multi-CPU affinity of Suricata in pictorial format.

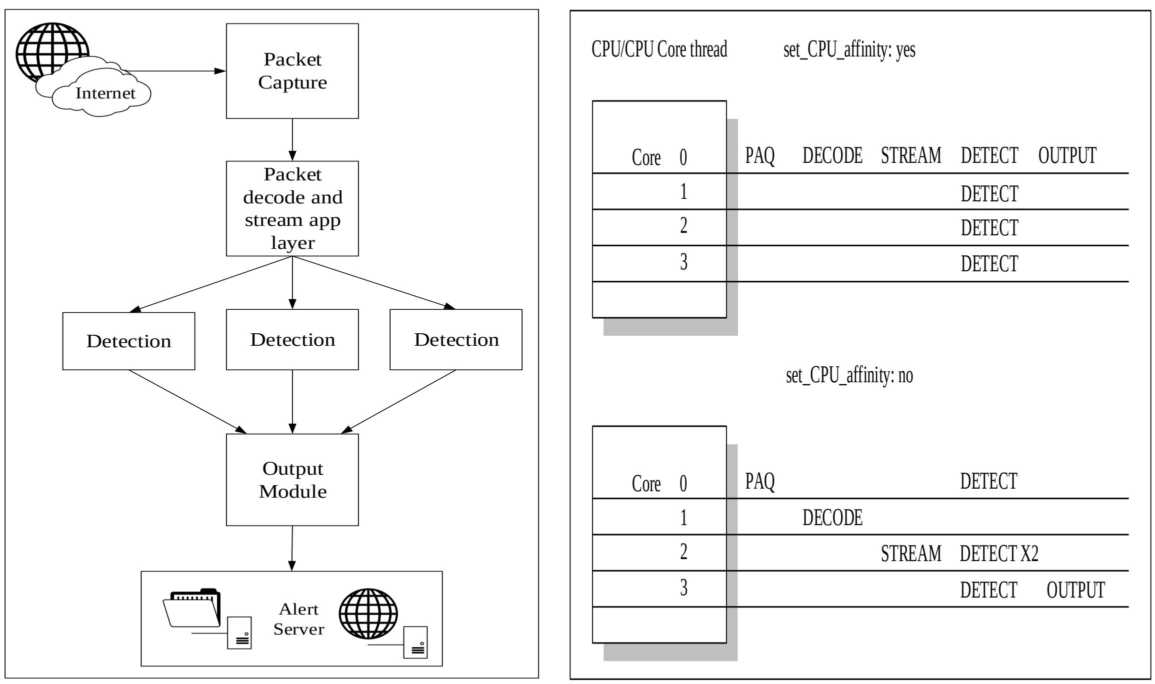

Figure 5.7: Multi-thread, Multi-thread CPU affinity of Suricata [14]

Key difference of both Intrusion Prevention System (Suricata and Snort) also identified and Table 5.3 indicates the key comparison of Suricata and Snort.

Table 5.3: Key difference between Snort and Suricata

Parameter	Suricata	Snort
Intrusion Prevention Feature	Yes	Yes
VRT rule support	Yes	Yes
Emerging threat rules support	Yes	Yes
SO rule support	No	Yes
Multi-thread support	Yes	No
IPv6 support	Yes	Yes
Capture accelerator support	Yes	No
Ease of installation	No	Yes
Configuration filename	suricata.yaml	snort.conf

5.5 Summary

The experimental results shows that Suricata performed well than Snort in terms of latest threats or intrusion on CICIDS2017 dataset. Day by day zero day exploits, malware, ransomware are made to interrupt the network and computer systems. It is necessary to improve existing Intrusion Prevention Systems like Suricata and Snort, make them more efficient to protect from modern intrusions.

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 Conclusion

The study conducted with two most common and well-known open-source network-based intrusion prevention system. CPU utilization and performance accuracy was evaluated and compared of both systems. In same structured lab environment experiment was deployed to fulfil the goal of the study. Performance was evaluated based on the latest intrusion prevention system dataset to test their ability for preventing modern intrusions. Both system performed very well during the experiment. But in some cases Suricata's performance was really noteworthy. Due to the difference in their internal structure like multi-thread detecting engine and multi-affinity CPU capability, performance was varied. It was also identified that Suricata used more RAM than Snort for multi-processing functionality. After the experiment it is stated that existing Intrusion Prevention Systems are capable to work against modern known threats. And it is also recommended to use Intrusion Prevention System in Internet-based companies and organization to protect critical infrastructures and to improve data security.

6.2 Future work

Future work of the study could be develop an enhance Intrusion Prevention System that will be capable to identify and protect unknown intrusion in both network and computer systems. As existing IPS are very much dependent on their rules. So they are only capable to protect known threats. Zero-day attack is on the rise. So, it is necessary to improve existing Intrusion Prevention Systems or develop an enhance system.

APPENDIX

APPENDIX A: LIST OF ABBREVIATION

IDS	Intrusion Detection System
IPS	Intrusion Prevention System
HIPS	Host-based Intrusion Prevention System
NIPS	Network-based Intrusion Prevention System
WIPS	Wireless Intrusion Prevention System
IP	Internet Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
PC	Personal Computer
CPU	Central Processing Unit
RAM	Random Access Memory
MAC	Media Access Control
OISF	Open Information Security Foundation
DAQ	Data Acquisition
DVWA	Damn Vulnerable Web Application
NFQ	Netfilter Queue

APPENDIX B: RELATED ISSUES

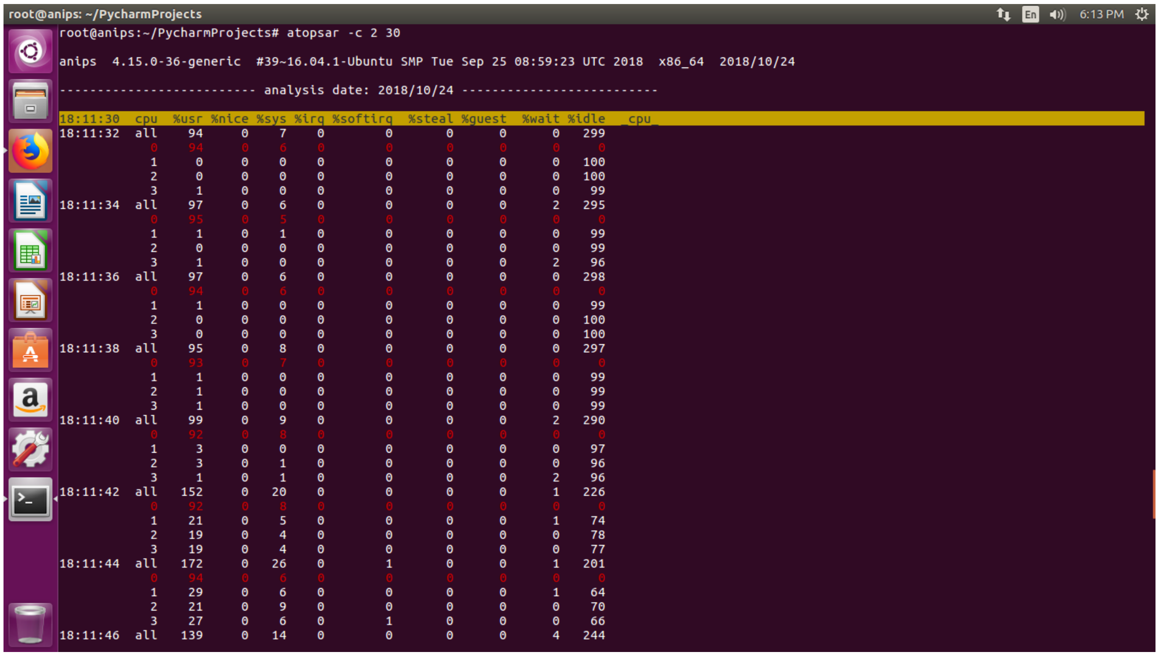

Figure A1: Atop output for experimenting CPU utilization

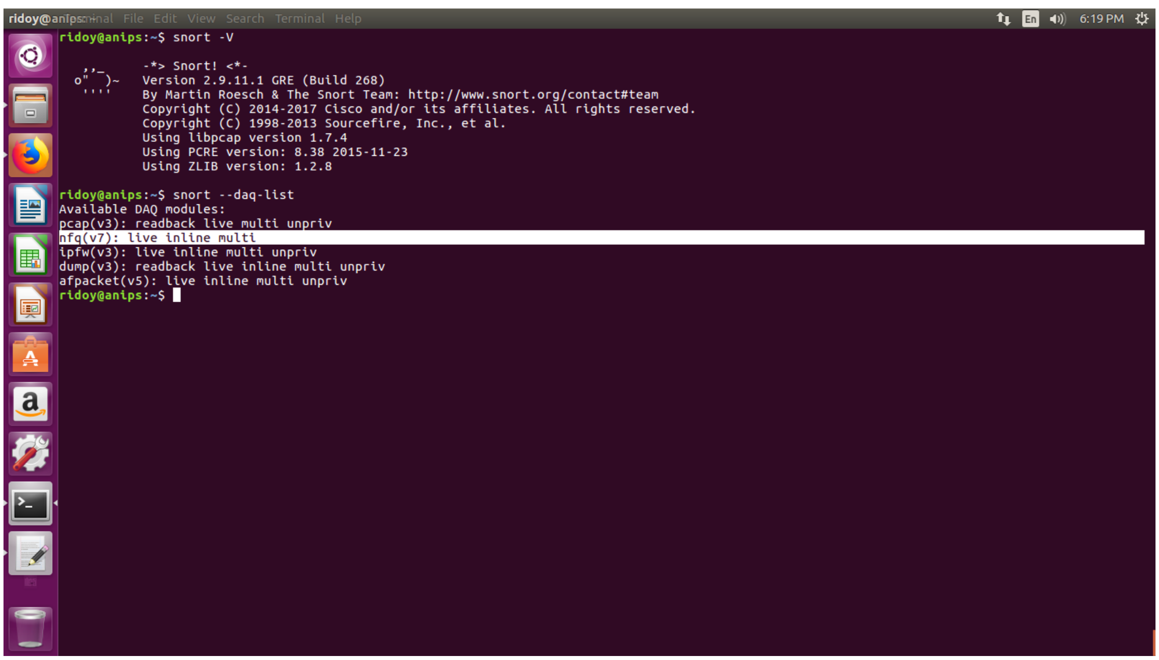

Figure A2: Checking Snort NFQ mode

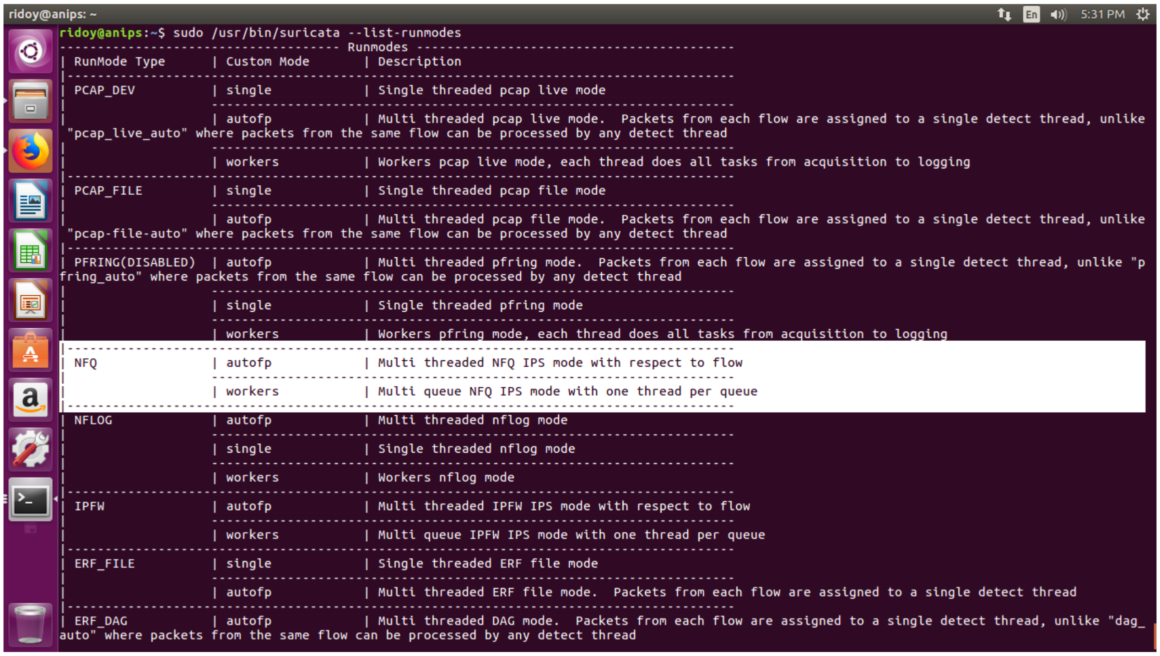

Figure A3: Prove of Suricata's multi-threading capability

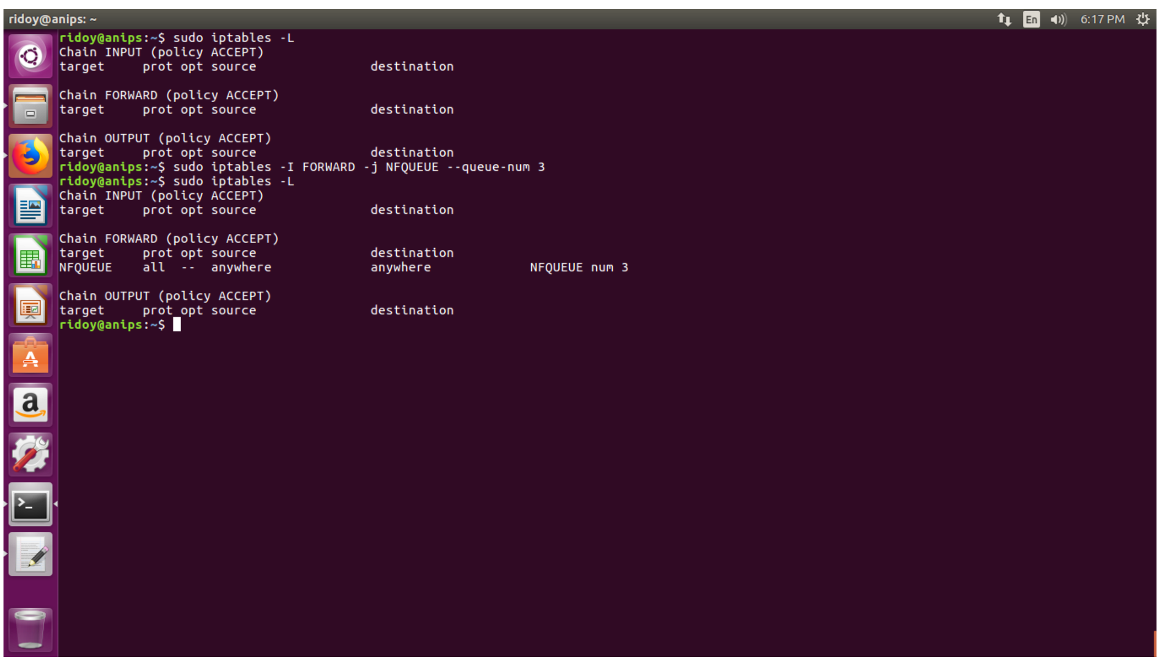

Figure A4: Netfilter Queue (NFQ)

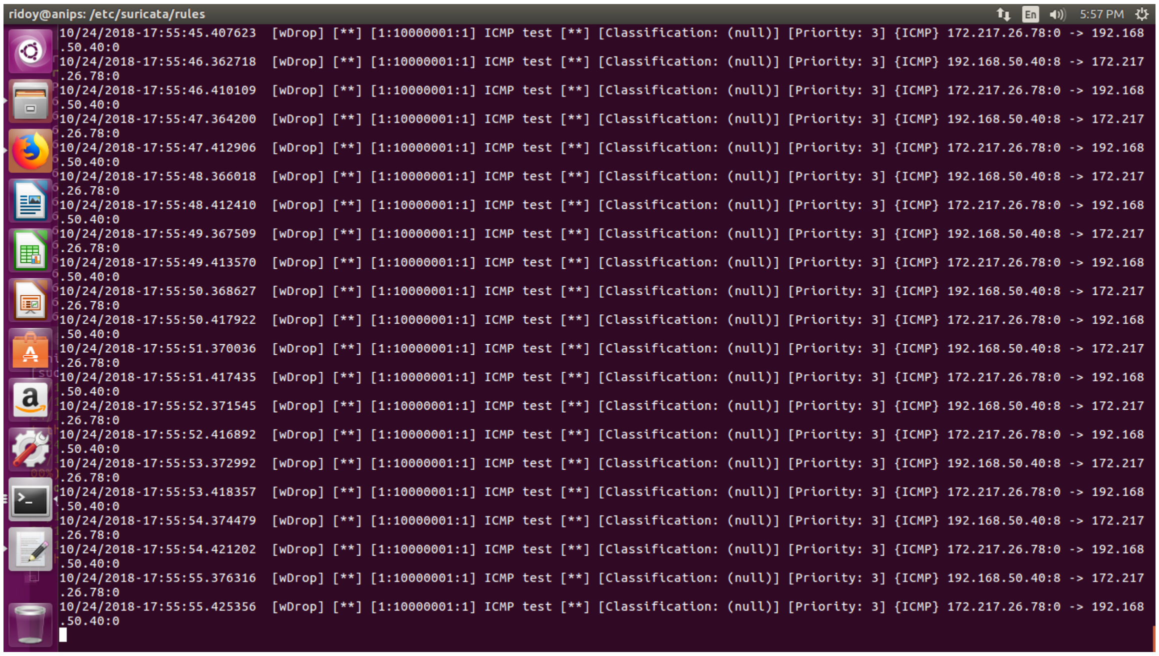

Figure A5: Packet dropping in Suricata

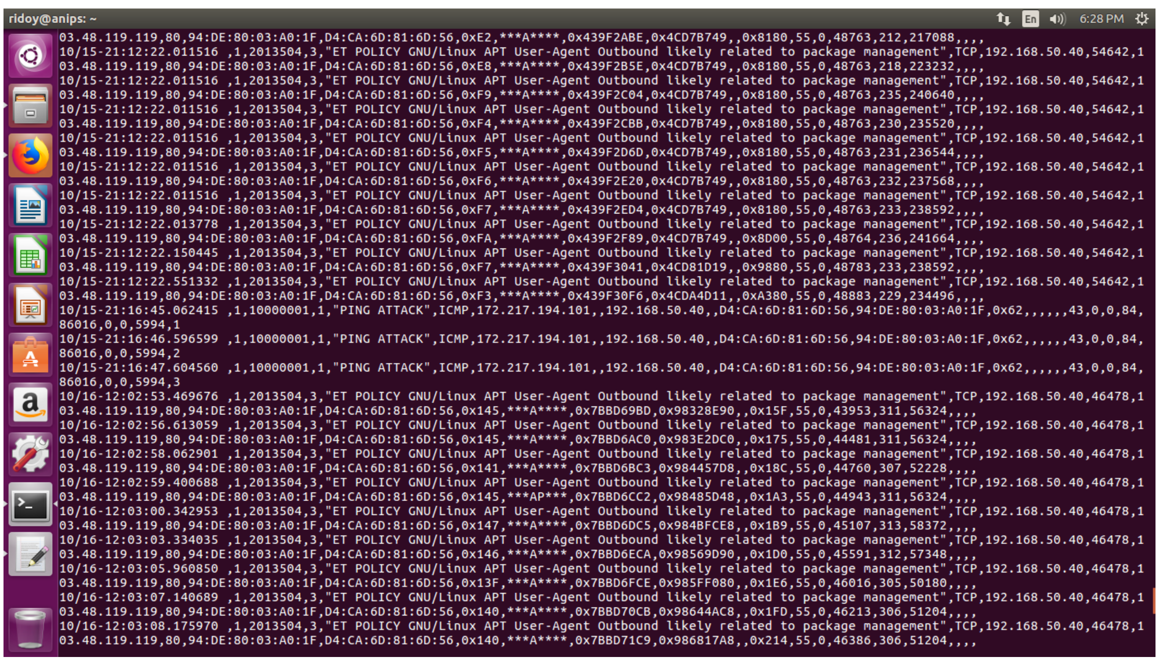

Figure A6: Packet dropping in Snort

REFERENCES

[1] Specarfone Karen, and Peter Mell. "Guide to Intrusion Detection and Prevention Systems (IDPS)." *Guide to Intrusion Detection and Prevention Systems (IDPS)*, Feb. 2007, csrc.nist.gov/publications/detail/sp/800-94/final.

[2] "PF_RING™ High-Speed Packet Capture, Filtering and Analysis." *Ntop*, www.ntop.org/products/packet-capture/pf_ring/.

[3] Sergey Bezborodov. "Intrusion Detection System and Intrusion Prevention System with Snort Provided by Security Onion." *Mikkeli University of Applied Sciences*, 2016.

[4] Ahmad, Iftikhar, et al. "Comparative Analysis of Intrusion Detection Approaches." *International Conference on Computer Modelling and Simulation*, 2010, doi:10.1109/UKSIM.2010.112.

[5] Kumar, B.Santos, et al. "Intrusion Detection System- Types and Prevention." *International Journal of Computer Science and Information Technologies*, vol. 4, no. 1, 2013, pp. 77–82.

[6] Albin, Eugene. "A Comparative Analysis of the Snort and Suricata Intrusion-Detection Systems." *Naval Postgraduate School, September*, 2011.

[7] Loganathan, Gobinath. "Real-Time Intrusion Detection Using Multidimensional Sequence-to-Sequence Machine Learning and Adaptive Stream Processing." *The University of Western Ontario*, 2018.

[8] Lavrova, Darya, et al. "Wavelet-Analysis of Network Traffic Time-Series for Detection of Attacks on Digital Production Infrastructure." *SHS Web of Conferences*, 2018, doi:https://doi.org/10.1051/shsconf/20184400052.

[9] *Intrusion Detection Evaluation Dataset (CICIDS2017)*. Canadian Institute for Cybersecurity (CIC), 5 Feb. 2018, www.unb.ca/cic/datasets/ids-2017.html.

[10] bono, Joan, and Md. Nazrul Islam. "Joanbono/Snorter." *GitHub*, 22 Aug. 2018, github.com/joanbono/Snorter.

[11] *Suricata User Guide*. Open Information Security Foundation (OISF), 2018, media.readthedocs.org/pdf/suricata/latest/suricata.pdf.

[12] Wikepedia. “Wireshark.” *Wireshark-Wikipedia*, https://en.wikipedia.org/wiki/Wireshark.

[13] Wikipedia. *Receiver Operating Characteristic*. *Confusion Matrix* https://en.wikipedia.org/wiki/Receiver_operating_characteristic.

[14] “Threading.” *Suricata*, suricata.readthedocs.io/en/suricata-4.0.5/configuration/suricata-yaml.html.