Sudan University of Science and Technology

College of Engineering

Electronics Engineering

EVALUATION PERFORMANCE OF SNORT NETWORK INTRUSION DETECTION SYSTEM

A Research Submitted In Partial fulfillment for the Requirements of the Degree of B.Sc. (Honors) in Electronics Engineering

Prepared By:

1. Eman Ahmed Alnour Ibrahim
2. Hadeel Siragaldeen Albdri Mohmed
3. Joudy Omer Abdallah Abusham
4. Marwa Ezuldeen Mohmed Ali

Supervised By:

Dr. Ahmed Abdalla

October 2017

بسم الله الرحمن الرحيم

قال تعالى:

(قُلِ الرُّوحُ مِنْ أَمْرِ رَبِّي وَمَا أُوتِيتُمْ مِنَ الْعِلْمِ إِلَّا قَلِيلًا)

﴿ الإسراء٨٥﴾

صدق الله العظيم

Dedication

We dedicate our work to our families, whom without their continuous “PRAYERS” nothing could be achieved. And to the spirit of colleague Anas Fathi.(رحمه الله عليه)

ACKNOWLEDGEMENT

In the name of ALLAH, the most merciful and compassionate, who his mercy followed us during the long path of this research.

We would like to express our appreciation to our supervisor Dr.Ahmed–Abdalla ,who was cheerfully answered our queries provided us with materials, checked our examples , assisted us in myriad ways with the writing and helpfully commented on earlier drafts of this research.

We would also be very grateful to our parents, family and friends for their good humor and support throughout the production of this research.

ABSTRACT

With the thriving technology and the great increase in the usage of computer networks, the risk of having these network to be under attacks have been increased. Number of techniques have been created and designed to help in detecting such attacks. One common technique is the use of Network Intrusion Detection System NIDS. Today, number of open sources and commercial Intrusion Detection Systems are available to match enterprises requirements but the performance of these Intrusion Detection Systems is still the main concern. In this research ,an open source snort was implemented on Linux platform used for testing, analyzing packets attacks in Defense advanced Research Project Agency 1999 and comparing the result of it with ground truth table to evaluate the accuracy and performance of snort according to different metrics (true positive ,false positive, false negative, true negative, speed of snort to capture packet and analyze).The precision of the snort became high because so many rules defined (true positive) ,and still group of undefined rules false positive and false negative that effect the precision .The rustle of the obtained performance was medium ratio. Therefore , snort can deals better under that performance rate in offline traffic, if the rate becomes higher the performance will be reduced .

المستخلص

مع ازدهار التكنولوجيا و الزيادة الكبيرة في استخدام شبكات الحاسوب، ازداد خطر تعرض هذه الشبكات للهجمات. وقد تم إنشاء عدد من التقنيات مصممة للمساعدة في الكشف عن مثل هذه الهجمات. إحدى هذه التقنيات الشائعة هي استخدام انظمة كشف الاختراقات في الشبكات. يوجد اليوم عدد من المصادر المفتوحة وأنظمة كشف التسلل التجارية المتاحه لتتناسب مع متطلبات الشركات ولكن أداء هذه ألانظمة لكشف التسلل لا يزال مصدر قلق رئيسي. في هذا البحث، تم تطبيق المصدر المفتوح سنورت في بيئة التشغيل لينكس ومقارنة نتيجة ذلك مع جداول داربا لتقييم دقة و اداء سنورت المستخدمه للاختبار و تحليل حزم الهجمات وفقا لمقاييس مختلفه (صحيح ايجابي ,ايجابيه كاذبه,سلبيه كاذبه ,صحيح سلبي ,سرعة سنورت لالتقاط الحزم و تحليلها).اصبحت دقة سنورت عاليه نسبة لوجود الكثير من القواعد المعرفه (صحيح ايجابي), و لكن ما زال هنالك مجموعه من القواعد الغير معرفه (سلبيه كاذبه , ايجابيه كاذبه) التي تؤثر على الدقه. كانت نسبة الأداء الذي تم الحصول عليه نسبه متوسطه , لذلك يمكن ان يتعامل سنورت بشكل أفضل إذا كان معدل الاداء اقل من هذه النسبه , وإذا إرتفع من هذه النسبه ذلك سيؤدي الى نقصان الاداء بشكل ملحوظ.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

BPF	Berkeley Packet Filter
BPS	Bit Per Second
BSD	Berkeley Software Distribution
CGI	Common Gateway Interface
CSV	Comma Separated Values
DARPA	Defense Advanced Research Agency
DDOS	Distributed Denial of Service
DOS	Denial of Service
DPI	Deep Packets Inspection
ESXI	Elastic Sky X Is the server
FN	False Negative
FP	False Positive
FTP	File Transfer Protocol
HIDSs	Host-based Intrusion Detection Systems
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection Systems

IFS	IOS File System
IOS	Iphone Operating System
IP	Internet Protocol
NIC	Network Interface Card
NIDSs	Network-based Intrusion Detection Systems
NMAP	Network Mapping
NT	New Technology
PHF	Perturbative Hartree Fock
R2L	Remote to Local
TCP	Transmission Control Protocol
TN	True Negative
TP	True Positive
U2R	User To Root
UDP	User Datagram Protocol
URL	Uniform Resourse Locator
VM	Virtual Machine

CHAPTER ONE

INTRODUCTION

CHAPTER ONE

1. INTRODUCTION

1.1 Introduction

1.2 Problem Statement

1.3 Research Objectives

1.4 Proposed Solutions

1.5 Research Outlines

1.1 Introduction

The global growth of the Internet and networking has made securing networks and information one of the most challenging tasks in the field of network communication. Today, intrusion attacks are generating significant worldwide epidemic to network security environment and bad impact involving financial loss. Intruders have the capability to infect thousands of hosts and networks within few minutes before human action takes place.

Intrusion detection can be defined as the process of monitoring and identifying the computer and network events, to determine the emergence of any abnormal incident, as consequence, this unusual event is considered to be an intrusion. It can be defined as the process of identifying and responding to malicious activity targeted at computing and networking resources. It detects unwanted exploitation to computer system, both through the Internet and Intranet.

In general, Intrusion Detection Systems (IDSs) can divide into two basic classes based on their position in the network or audit source location: host-based IDSs (HIDSs) and network-based (NIDSs). HIDS monitors a single machine and audit data, such as resource usage and system logs, traced by the hosting operating system. On the other hand, NIDS, such as snort, monitors a network and analysis the traffic which flows through the segment. NIDSs have the following advantages: In contrast to HIDSs, the deployment of new host in network does not need more effort to monitor the network activity of that new host. Generally, it is easier to update one component of NIDSs than many components of HIDSs on hosts [1].

NIDSs also can be classified based on its detection model into two categories: signature-based and anomaly-based. The signature-based NIDSs, also named misused-based, works similar to anti-virus software. It employs a signature (pattern that correspond to a known threat) database of known attacks, and if a successful match with current input, an alert is raised. A well-known example of this type is Snort which is an open source IDS that monitors network by matching each packet it observes against a set of rules. Anomaly-based or behavior-based NIDS works by building a model of normal traffic data pattern during a training phase, and then it compares new inputs to the model [1].

There are two methods basis on the source of data to be analyzed in NIDSs: packet-based NIDSs and flow-based. Packet- based traditional NIDS also named Deep Packet Inspection (DPI) has to analyze the whole payload content beside headers. In flow-based NIDS, rather than looking at all packets going through a network link, it looks at aggregated information of related packets of network traffic in the form of flow, so the amount of data to be analyzed is reduced [1].

Packet-based mostly provides signature-based NIDSs valuable information to detect attacks while flow-based support anomaly-based NIDSs to have ability to detect anomalies.

IDS differs from firewall in that a firewall looks outwardly for intrusion in order to stop them from happening ,firewall limit access between network to prevent intrusion and do not signal an attack inside the network, but ids can detect attack inside and outside the network.

1.2 Problem Statement

At the present time security is an important issue for all networks. There are many NIDSs that work against various intrusions. However, measuring performance and detection accuracy of NIDS is always a challenge because there are many factors that influence its performance and accuracy.

1.3 Aim and Objectives

To evaluate Snort Network Intrusion Detection System in terms of:

1. Offline performance.

2. Detection Accuracy.

1.4 Proposed Solution

Snort has been installed and configured in Linux. Benchmark dataset DARPA1999 has been downloaded and analyzed, the result of snort alerts have been compared against ground truth .

1.5 Thesis Outlines

The rest of this thesis is organized as follows:

Chapter two of this thesis contains general background about network security and the techniques that have been developed for that concept. Network Intrusion Detection Systems (NIDSs) is one of those techniques.

The methodology of the work has been presented in chapter three. It contains the process flow to reach the proposed solution for the project problem. Besides, a brief description of key snort commands- that would be used to analyze DARPA 1999- is presented. In chapter four the analyzed results of snort evaluation are presented and discussed.

Chapter five gives a conclusion of what have been done during the research and makes suggestions for future work.

CHAPTER TWO

LITERATURE REVIEW

CHAPTER TWO

2. Literature Review

2.1 Introduction

2.2 Background

2.3 Snort program

2.4 Accuracy Metrics in NIDS

2.1 Introduction

This chapter contains a background about the network security and the techniques that have been developed to secure the network.

In the first section of the background a general idea of NIDS (network intrusion detection system) was introduced, its definition, why we need it, architecture of its components, taxonomy that categorizes NIDS according to specific accepts . Second section is about snort information, snort traffics (online / offline) , snort mechanism and its modes .The third section goes through about evaluation performance of snort .

2.2 Background

Security is a big issue for all networks in today's enterprise environment. Hackers and intruders have made many successful attempts to bring down high-profile company networks and web services. Many methods have been developed to secure the network infrastructure and communication over the Internet, among them the use of firewalls, encryption, and virtual private networks. Intrusion detection is a relatively new addition to such techniques. Intrusion detection methods started appearing in the last few years. Using intrusion detection methods, you can collect and use information from known types of attacks and find out if someone is trying to attack your network or particular hosts. The information collected this way can be used to harden your network security, as well as for legal purposes. Both commercial and open source products are now available for this purpose. Many vulnerability assessment tools are also available in the market that can be used to assess different types of security holes present in your network [6].

Network Intrusion Detection Systems (NIDSs) proved to be an efficient technique that can process large volume of networks traffic and detect intrusions in their early stages in order to limit its catastrophic damages.

Today, intrusion attacks are generating significant worldwide epidemic to network security environment and bad impact involving financial loss.

The reasons that we need IDS (Intrusion Detection System) can be concluded into following three points :

1- The inherent Vulnerabilities in the traditional network security hierarchy demonstrate that it is impossible to ensure network security without any external protection .
2- Firewalls cannot guarantee 100 percent security.
3- The prevalent flaws in Web application also declare that the introduction of intrusion detection system is desirable [3].

NIDS monitors network traffic for particular network segments or devices and analyzes the network and application of protocol activity to identify suspicious activity [4]. It can be installed on active network elements, for example on routers. NIDS utilizes the source and destination IP addresses to deduce security-related parameters, like the number of total connection arrivals in a certain period of time, the number of packets to/from a ceiling machine, or the arrival time between packets. These parameters can be used to detect port scans or DoS (Denial of Service) attempts [3].

2.2.1 NIDS Architecture

In general any NIDS can be viewed as architecture of the components shown in [2]. These components are :

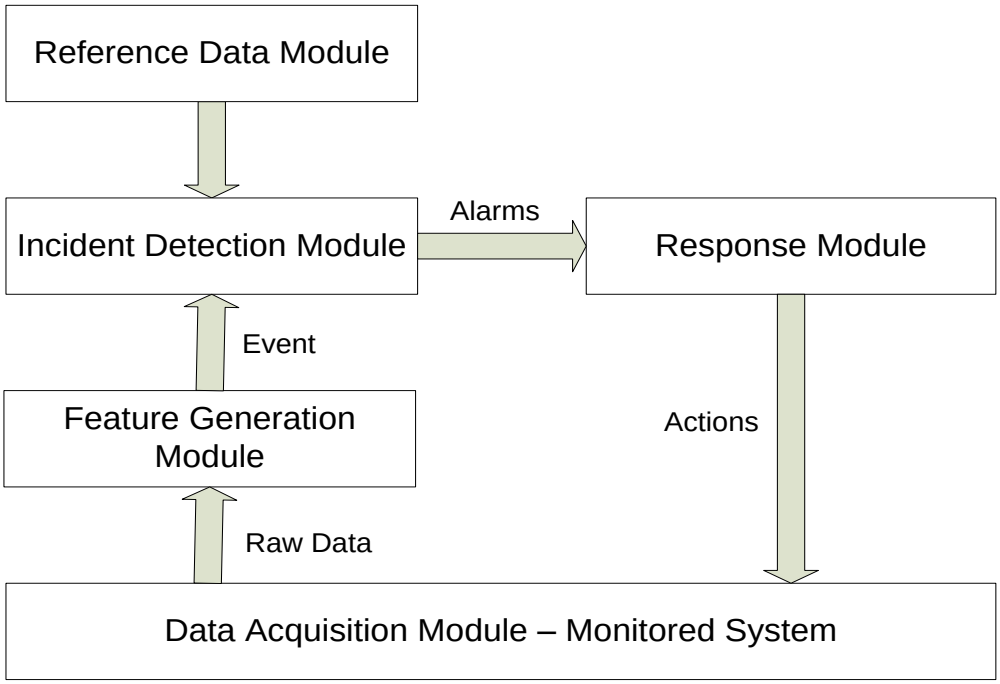


Figure: 2-1 Architecture of NIDS [3]

1. Data Acquisition Module: responsible for collecting network traffic data.
2. Feature Generator Module: responsible for extracting a set of selected traffic features (packet header features, payload features, flow features)
3. Incident detection Module: responsible of identifying intrusion and generating alarms by comparing the data generated from the feature generation module with that of the reference module. Generally there are two methods of this detection; misuse-based detection and anomaly based detection
4. Response Module: Once an alert is received, this module is responsible for initiating actions in response of a possible intrusion
5. Reference Data Module: This module contains the reference data that is to be used by the incident detector to compare with. If the detection method is misuse-based then this data would be based on the description of all known intrusion (intrusion database), and if

the detection method is anomaly-based then it would be based on description of normal attack free network operation (network profile).

2.2.2 NIDS Taxonomy

Several taxonomies had been proposed for categorizing NIDS, but the most common and acceptable one is that one used by [7] and [5] DARPA1999 has been used a taxonomy based on these works. This taxonomy categorizes NIDSs according to the following aspects :

1. Analysis strategy (Detection Method): This categorization distinguishes NIDS according to the nature of reference data used for identifying intrusions which is either misuse-based or anomaly based.
2. Information source: NIDS can be packet-level based or flow-level. Packet-level NIDS can be further divided to packet header based or payload-based.
3. Architecture (Locus of Detection): NIDS can be centralized when it is placed in single position or distributed when there are several points for monitoring .
4. Response to intrusion: NIDS can be either passive when it is just to raise an alarm on detecting intrusion or active when a further action is to be done in response to that intrusion.
5. Usage frequency: NIDS can work in real time and detect intrusions while they are taking place or by batch (none real time).

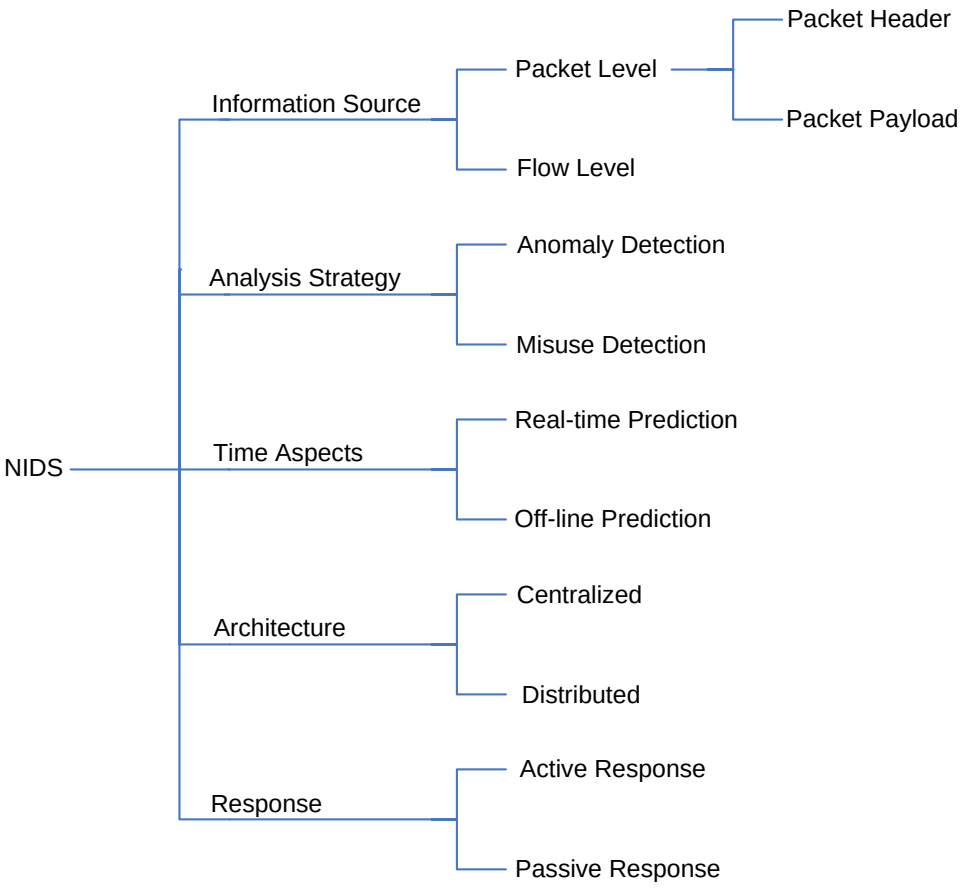


Figure 2-2: NIDS Taxonomy [5]

2.2.3 Network Attacks

The words intrusions and attacks are usually used interchangeably. However, there are a lot of definitions of the word "attack" that make a different definition if it compared to intrusion. For example, if a system is attacked (either from the outside or the inside) a defensive firewall around the system may prevent that intrusion. Therefore, the author of [4] stated that an attack is an intrusion attempt, and an intrusion results from an attack that has been (at least partially) successful.

2.2.4 Type of Attacks

1. Denial of Service (DoS)

A Denial of Service (DoS) attack attempts to crash a computer or a process in a computer, a network or do any action to prevent authorized users to use available resources or services [5]Two subclasses of DoS attacks can be encountered:

1. Operating System DoS where these attacks target vulnerabilities in certain operating systems.
2. Networking Attacks DoS where these attacks exploit inherent limitations of infrastructures and protocols of networking and data communication.

DoS attacks may not seek to crash the attacked host rather than to keep it busy with numerous connections between fake machines thus preventing access to a service, preventing authorized users from accessing a service, disrupting service. In the advanced variation of DoS, which is the distributed DoS (DDoS), numerous machines are compromised and deployed to achieve this goal.

2. Probing

Probing attacks are also called surveillance and scanning attacks. These attacks probe the networks to find out available hosts or services and collect information about them. From this information an attacker makes a list of vulnerabilities that may be used later to launch an attack against selected hosts or entities. There are several types of probing attacks. Such types include network scanning or IP sweep (scanning the network hosts for a target service on a specific port), host scan or port sweep (scanning many ports to find out available services on a single host), NMAP (a common network mapping tool).

3. Compromises

Compromises are these attacks which use known vulnerabilities and weak security points such as buffer overflow [7] to login illegally into systems and gain privileged ac Types of Attacks matching with time and IP address. According to attack source (outside attack versus inside attack), the compromises can be further classified into these two categories R2L (Remote to Local) and U2R (User to Root).

R2L attacks are exploits where attackers start out accessing an account of a normal user on the system (e.g.: by sniffing passwords, a dictionary attack, or social engineering) and then seek a way to exploit some vulnerability in order to gain an administrator privileges to the system.

In U2R attack, attackers who have the authority to send packets to a host remotely over a network but do not have an account on that host exploit some vulnerability to gain access privileges of a local user of that host.

1. Replicators

These types of attacks are malicious codes that replicate on victim host and deploy network to propagate to other hosts. They can be classified into three main types: 1. Viruses: Are malicious codes that replicate themselves by being attached to other benign programs and infecting them. They may cause significant damages (e.g. delete files on the hard drive) or they may only do some harmless, yet annoying actions (e.g. display funny messages on the screen). Viruses usually need human interaction (e.g. copying files from a portable disk or opening email attachments) to replicate and spread to other computer systems. 2. Worms: Are self-replicating malicious codes that spread through a network aggressively. They take advantage of automatic packet send and receive features which are found on many computers. 3. Trojan horses (or simply Trojans): Are malicious, security-breaking codes that appeared as something innocent [2] . For example, a user may download a program

that appears as a free game, but when it is executed, it may delete all files on the computer.

2.3 Snort Program

snort was created by Martin Roesch in 1998, and is an open source network intrusion detection and prevention system Snort can be used in three different ways; as a packet sniffer like tcp dump,
a packet logger or as a network intrusion detection and prevention system [11].

When used as a packet sniffer, Snort will read network packets and display them on the console, and when used as packet logger Snort will log packets to disk. In intrusion detection mode it will monitor the network traffic and analyze the traffic against a rule set defined by the user.

In intrusion detection mode, Snort uses a number of rules that define anomalous traffic. Most of these rules are made by Source fire, and other rules are made by the community, and it is possible to make own rules as well. In addition to rules, Snort has several preprocessors which enable modules to view and a alter packets before they get inspected by the intrusion detection system [11].

When running Snort, it works by detecting and reporting malicious traffic or so called events. The process of reporting events can be configured through event handling. By configuring thresholds one can reduce the number of logged alerts for noisy rules. This helps Snorts to handle more traffic. [6]

Snort has the capability to configured to send output to various locations, when certain Snort rules is triggered. The most common output module is the alert syslog. [10] Snort is capable of performing real time traffic analysis, which means that it can detect ongoing intrusions. It can perform logging on IP networks, perform protocol analysis, content searching and content matching, and it can be used to detect a variety of attacks . [10]

Snort is logically divided into multiple components. These components work together to detect particular attacks and to generate output in a required format from the detection system.

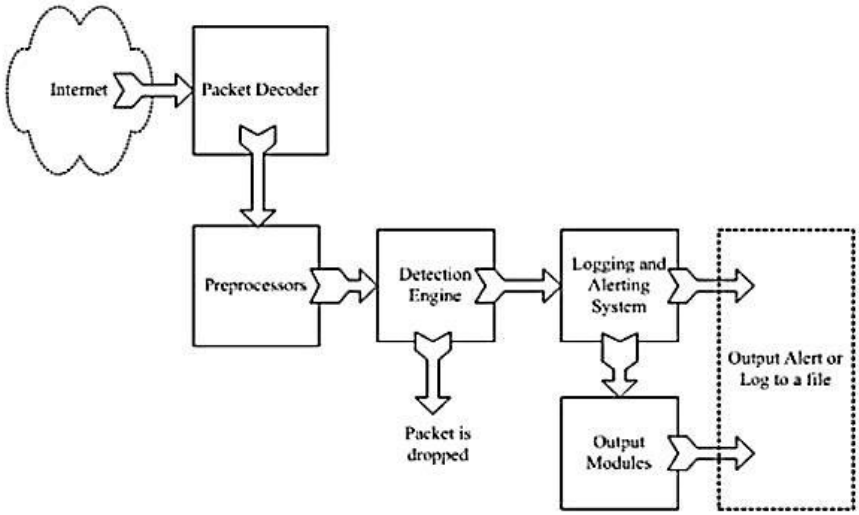


Figure 2-3: The Basic Elements of Snort Architecture [11]

2.3.1 Three main operational modes of snort

- Sniffer Mode :

Berkeley Packet Filter (BPF) filtering interface available to shape displayed network traffic

- Packet Logger Mode :

Log network packet to log file ,log directory (var / log / snort)

- Network intrusion detection system :

Start snort with your set of rule ,setup packet analysis tool and let it ,do its work .

Snort will analysis packet in real time and non-real time , generate alert and log offending packet .

2.4 Accuracy Metrics in NIDS

1. Accuracy

I. Detection Rate: ratio between the number of correctly detected attacks and the total number of attacks.
II. False alarm (false positive) rate: ratio between the number of normal connections that are incorrectly misclassified as attacks and the total number of normal connections.

2. Performance : Ability of snort to process incoming packets , measuring by (rate/bps)
3. Completeness
4. Timely response
5. True positive (TP): intrusions that are successfully detected by the IDS.
6. False positive (FP): normal/non-intrusive behavior that is wrongly classified as intrusive by the IDS.
7. True negative (TN): normal/non-intrusive behavior that is successfully labeled as normal/non-intrusive by the IDS.
8. False negative (FN): intrusions that are missed by the IDS, and classified as normal/non-intrusive [11].

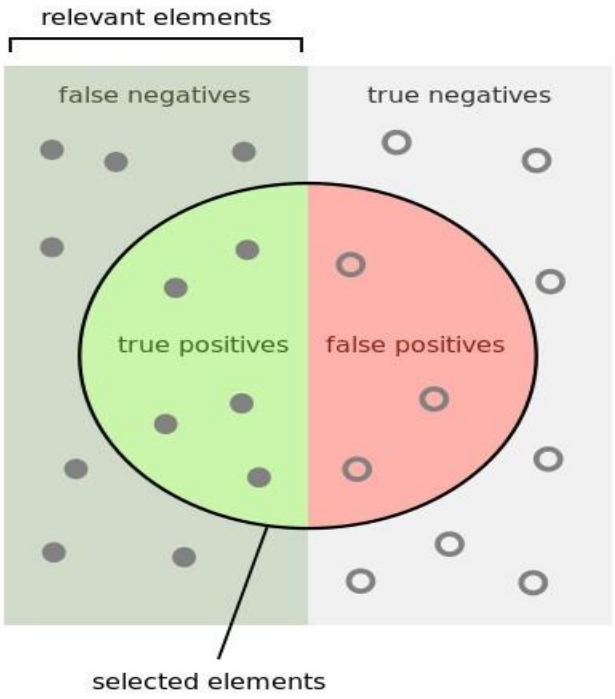


Figure 2 - 4: Detection accuracy [13]

5.2 Related Works

Recently there have been some studies that tried to contribute within the scope of the study, for example:[11]a review of various Intrusion detection Systems and its tools by focusing on SNORT IDS-an open source tool. Also, I have presented an extension of SNORT IDS by adding a new pre-processor in snort detection engine to find the detection anomalies. This engine filters all the files and loads the attacked or infected files into its loader by. conf file command. The study of [6] testing and analyzing of the performance of the well know IDS system Snort and the new coming IDS system Suricata. Both Snort and Suricata were implemented on three different platforms (ESXi virtual server, Linux 2.6 and FreeBSD) to simulate a real environment. The study of [8] Snort will log any intrusion it detected in a logfile called Alert. The mobile agent platform which is known as Tahiti Server will initialize two mobile agents during startup; abs.ReconAglet and abs.RespondAglet. abs.ReconAglet is to read the Alert log file and if anintrusion occurred, it will inform abs.RespondAglet

which will clone itself and migrate to drop the attacker'spacket at the victim's host through firewall The intrusion or test selected for this project is of DOS attacks types. Three operating systems (OpenBSD, FreeBSD and Linux)

CHAPTER THREE

RESEARC H METHODOLOGY

CHAPTER THREE

3. RESEARCH METHODOLOGY

3.1 Introduction

3.2 Activity Steps

3.3 Software Tools

3.4 Configuration of Snort

3.5 Activate Snort Rules

3.6 Test Snort with a Benchmark Dataset (DARPA1999)

3.1 Introduction

In this chapter the methodology of the work has been presented. The first section contains a general overview of what has been done during the work.

The second section contains the proposed solution for the research problem and a general block diagram of the system which is desired to be reached.

In the third section the process of snort configuration is explained. after that testing dataset which is DARPA1999. The description of what has been done is illustrated at this chapter .

3.2 Activity Steps

The steps that have been followed as the process flow of the project are illustrated in the figure below:

1. Literature Review: by reading couple of papers about IDs the general idea of the project was been understood, IDS concept, why we need it After understanding what is meant by IDS in general the efforts were been focused in performance and accuracy of snort NIDS. Finally snort manual was been read to learn how to deal with it. Snort manual downloaded automatically when installing snort in "C:\Snort\doc" directory .
2. Requirements Analysis: to solve the problem, other efforts was been searched for; that have the same situation to help us finding a proper way to solve our problem. The solution that has been suggested to solve the problem by using snort program and ability of it to detect intrusion.
3. Snort Installation and Configuration: snort was configured to work in NIDS mode .

4. Rules: download and activating all snort rules activate.
5. Obtaining Dataset: the traffic that has been used is DARP1999 dataset (second week).
6. Test Snort with a Benchmark Dataset: the tcpdumps for the five days were been stored as alerts.
7. Analyzing alerts: the results that have been produced from the captured data were compared with the standard results in DARPA1999 dataset to find number of attacks detected their types and to calculate some metrics percentages.

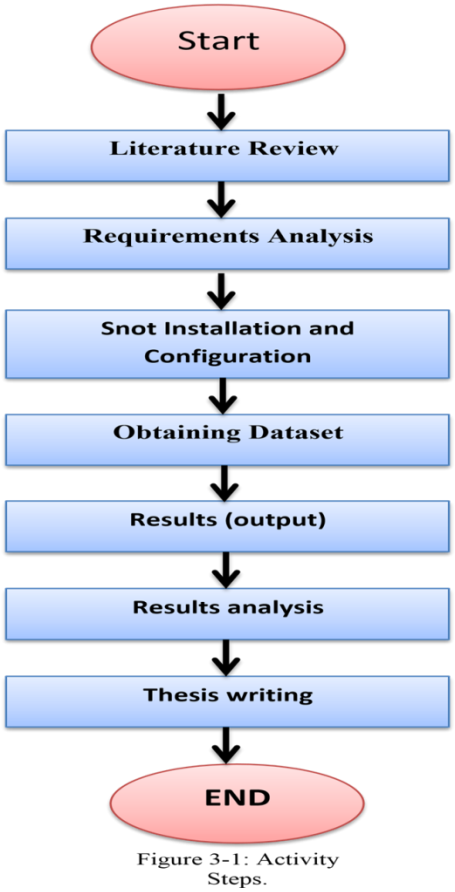


Figure 3-1: Activity Steps.

Figure 3-1: Activity Steps

3.3 Software Tools

VMware machines, to installed (Ubuntu in it version (14.4)) Version (2.9.9.0) of snort was been installed. For Snort to work properly, network interface card (NIC) has to been putted into promiscuous has been

installed .Snort rules-snapshot-(2.9.9.0).tar.gz file is the snort rules version that has been used. Snort has been used because it is the most famous and successful to detect most intrusion .

3.4 Configuration of Snort

As the snort2.conf that is contained inside the etc/ directory of the snort Tarbell is a snapshot in time (at the time of the Tarbell release), it's

necessary to occasionally update the snort2.conf in order to take advantages of different settings for the preprocessors and include new rule files [9].

The snort2.conf file controls everything about what Snort watches, how

it defends itself from attack, what rules it uses to find malicious traffic, and even how it watches for potentially dangerous traffic that isn't defined by a signature. A thorough understanding of what is in this file and how to configure it is essential to a successful deployment of Snort as an IDS in your environment [9] .

The file is organized into several sections:

1. Variable definitions, where you define different variables. These variables are used in Snort rules as well as for other purposes, likes pacifying the location of rule files.
2. Configure parameters, these parameters specify different Snort configuration options. Some of them can also be used on the command line.
3. . Preprocessor configuration, preprocessors are used to perform certain actions before a packet is operated by the main Snort detection engine.
4. Output module configuration, output modules control how Snort data will be logged.

5. Defining new action types, if the predefined action types are not sufficient for your environment, you can define custom action types in the Snort configuration file.
6. Rules configuration and include files, although you can add any rules in the main snort. conf file, the convention is to use separate files for rules. These files are then included inside the main configuration file using the include keyword [8].

3.5 Activate Snort Rules

All rules at the "C:\Snort\rules" directory have been activated. The activation is done by removing the # sign from the entire rule lines (uncomment them).

3.6 Test Snort with a Benchmark Dataset (DARPA1999)

There were two parts to the 1999 DARPA Intrusion Detection Evaluation: an off-line evaluation and a real-time evaluation. Intrusion detection systems were tested in the off-line evaluation using network traffic and audit logs collected on a simulation network. The systems processed this data in batch mode and attempted to identify attack sessions in the midst of normal activities. Intrusion detection systems were tested as part of the off-line evaluation .

3.6.1 Training Data

Three weeks of training data were provided for the 1999 DARPA Intrusion Detection off-line evaluation. The first and third weeks of the training data do not contain any attacks. This data was provided to facilitate the training of anomaly detection systems. The second week of the training data contains a select subset of attacks from the 1998 evaluation in addition to several new attacks .The primary purpose in

presenting these attacks was to provide examples of how to report attacks that are detected .

The following files are provided for each day in the training set:

1. Outside sniffing data (Tcp dump format)

2. Inside sniffing data (Tcp dump format) .[9]

3.6.2 Testing Snort

Snort was being tested using outside and inside Tcp dump data for each day of the second week by downloading those data. The results of the testing DARPA1999 were saved as comma separated values (CSV) file.

Then the file was analyzed by the signature IDs and the IP addresses in the source and destination(victim) according to the time of the attack defined in the ground truth table Figure (3.3) from the beginning of the attack occurrence until the appearance of new attack. Then study of the extent of correlation to which a particular IP address Figure (3.2) with several signature ID in the specified time for the attack .The amount of the total attacks were found. The final results of each inside and outside traffic were calculated for each day. The total number of attacks to measure the accuracy of Snort to detect intrusions based on the increase of the true positive ratio to maximum rate and reduce the percentage of false positive and false negative to the minimum rate , then calculate the performance of Snort and that by knowing the size of any Inside and Outside packet ,determine the time taken in the analysis to find the ratio between the size of the packet to the total time (Kbit/second) .

IP Address	Hostname	Operating System	Notes
135.13.216.191	alpha.apple.edu	Redhat 5.0	kernel 2.0.32
135.8.60.182	beta.banana.edu	Solaris 2.5.1	
194.27.251.21	gamma.grape.mil	SunOS 4.1.4	
194.7.248.153	delta.peach.mil	Redhat 5.0	kernel 2.0.32
195.115.218.108	epsilon.pear.com	Solaris 2.5.1	
195.73.151.50	lambda.orange.com	SunOS 4.1.4	
196.37.75.158	jupiter.cherry.org	Redhat 5.0	kernel 2.0.32
196.227.33.189	saturn.kiwi.org	Solaris 2.5.1	
197.182.91.233	mars.avocado.net	SunOS 4.1.4	
197.218.177.69	pluto.plum.net	Redhat 5.0	kernel 2.0.32
192.168.1.30	monitor	MacOS	AF SNMP monitor
192.168.1.10	calvin.world.net		Outside gateway
192.168.1.20	aesop.world.net		Outside Web Server
192.168.1.90	solomon.world.net		Not Part of Simulation

Figure (3.2) : Simulation Network Hosts –DARPA 1999

Detections List

ID	Date	Start_Time	Destination	Score	Name
1	03/08/1999	08:01:01	hume.eyrie.af.mil	1	NTinfoscan
2	03/08/1999	08:50:15	zeno.eyrie.af.mil	1	Pod
3	03/08/1999	09:39:16	marx.eyrie.af.mil	1	Back
4	03/08/1999	12:09:18	pascal.eyrie.af.mil	1	Httptunnel
5	03/08/1999	15:57:15	pascal.eyrie.af.mil	1	Land
6	03/08/1999	17:27:13	marx.eyrie.af.mil	1	secret
7	03/08/1999	19:09:17	pascal.eyrie.af.mil	1	ps attack
8	03/09/1999	08:44:17	marx.eyrie.af.mil	1	portsweep
9	03/09/1999	09:43:51	pascal.eyrie.af.mil	1	eject
10	03/09/1999	10:06:43	marx.eyrie.af.mil	1	back
11	03/09/1999	10:54:19	zeno.eyrie.af.mil	1	loadmodule
12	03/09/1999	11:49:13	pascal.eyrie.af.mil	1	secret
13	03/09/1999	14:25:16	pascal.eyrie.af.mil	1	mailbomb
14	03/09/1999	13:05:10	172.016.112. 001-114.254	1	ipsweep
15	03/09/1999	16:11:15	marx.eyrie.af.mil	1	phf
16	03/09/1999	18:06:17	pascal.eyrie.af.mil	1	httptunnel
17	03/10/1999	12:02:13	marx.eyrie.af.mil	1	satan

Figure (3.3) Detections List File in DARPA1999

CHAPTER FOUR

RESULTS

CHAPTER FOUR

RESULTS

4.1 Snort Testing

4.2 Attacks Packet Results

4.3 Performance Rate

4.4 Accuracy Rate

4.1 Snort Testing

After analysis inside and outside traffic snort save csv file as alert to be like the following .

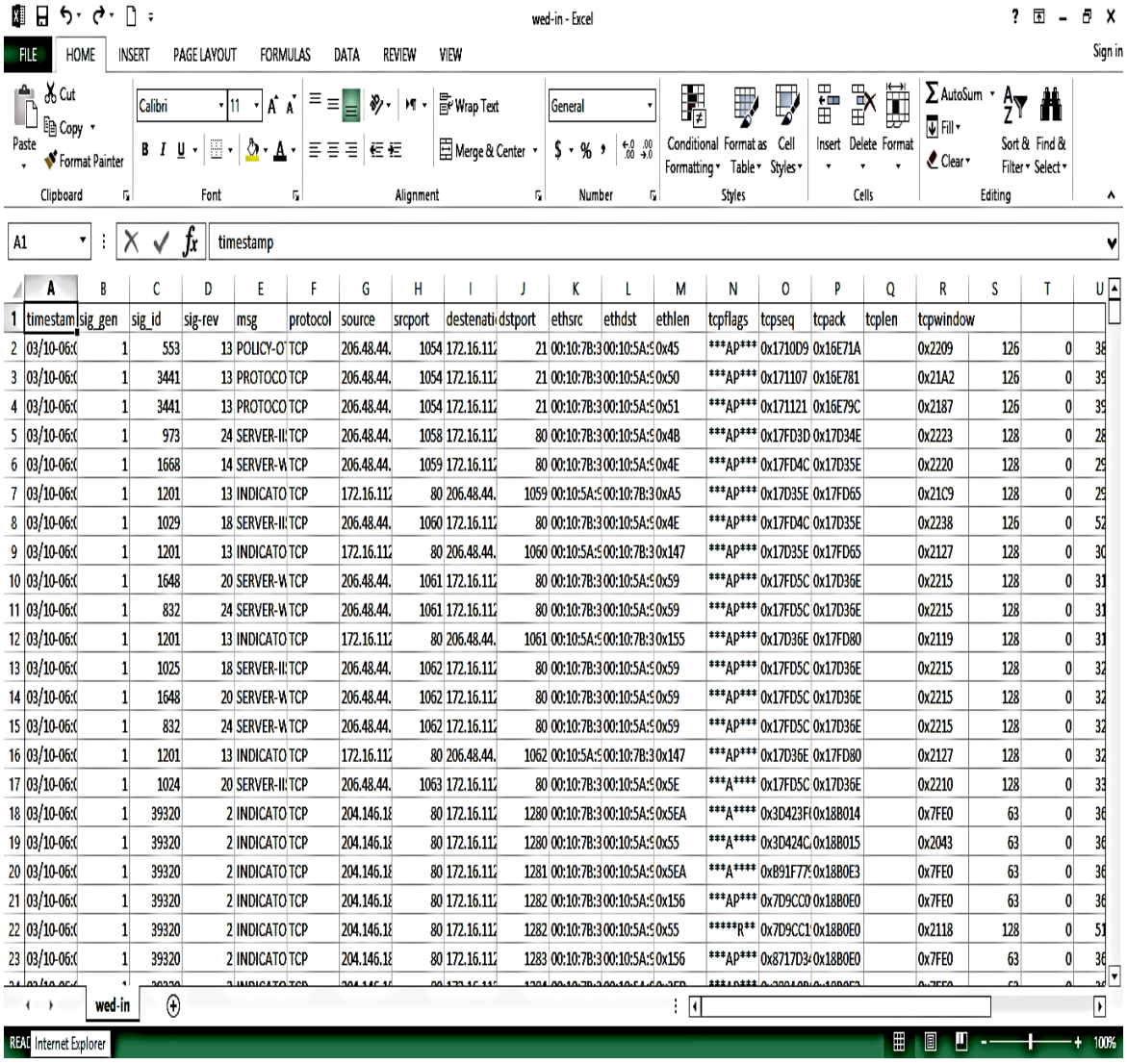


Figure 4-1: Sample Traffic output

4.2 Attacks Packet Types

Table 4.1: Types of Attacks

Name of Attacks	Description
NTinfoscan	A process by which the attacker scans an NT machine for information concerning its configuration, including ftp services, telnet services, web services, system account information, file systems and permissions.
Pod	Denial of service ping of death.
Back	Denial of service attack against apache webserver where a client requests a URL containing many backslashes.
Http tunnel	There are two phases to this attack: Setup — a web "client" is setup on the machine being attacked, which is configured, perhaps via crontab, to periodically make requests of a "server" running on a non-privileged port on the attacking machine.
Secret	Is an attack where the attacker maliciously or mistakenly transfers data which they have access to a place where it doesn't belong
Portsweep	Surveillance sweep through many ports to determine which services are supported on a single host.
Eject	Buffer overflow using eject program on Solaris. Leads to a user->root transition if successful

Mailbomb	A Denial of Service attack where we send the mail server many large messages for delivery in order to slow it down, perhaps effectively halting normal operation .
Phf	Exploitable CGI (Common Gateway Interface) script which allows a client to execute arbitrary commands on a machine with a misconfigured web server.
Satan	Network probing Tool which looks for well-known weaknesses. Operates at three different levels. Level 0 is light.
Crashiis	A single, malformed http request causes the webserver to crash.
Ipsweep	Surveillance sweep performing either a port sweep or ping on multiple host addresses.
Prel	Perl attack which sets the user id to root in a Perl script and creates a root shell.
Neptune	Syn flood denial of service on one or more ports.
ftp-write	Remote FTP (The File Transfer Protocol) user creates .r host file in world writable anonymous FTP directory and obtains local login.
Ps	Ps takes advantage of a race condition in the Ps command in Sol. 2.5, allowing a user to gain root access.

Load module	Non-stealthy load module attack which resets IFS (IOS File System) for a normal user and creates a root shell.
Land	Denial of service where a remote host is sent a UDP packet with the same source and destination.

Table 4.2: Types of Attacks matching with time and IPaddress

Day	Start Time	End Time	IP address	Attack type	Inside Attacks	Outside Attacks
Mon-day	6:01:01	6:50:15	172.16.112.100	Ntinfoscan	65	63
	6:50:15	7:39:16	172.16.113.50	Pod	-	-
	7:39:16	10:09:18	172.16.114.50	Back	36	85
	10:09:18	13:57:15	172.16.112.50	Http tunnel	34	2
	13:57:15	15:27:13	172.16.112.50	Land	7	-
	15:27:13	17:09:17	172.16.114.50	Secret	7	7
	17:09:17	-	172.16.112.50	Ps	-	-
Tues-day	6:44:17	7:43:50	172.16.114.50	Portsweep	998	1000
	7:43:51	8:06:43	172.16.112.50	Eject	3	3

	8:06:43	8:54:19	172.16.114.50	Back	87	6
	8:54:19	9:49:13	172.16.113.50	Loadmodule	-	-
	9:49:13	11:05:10	172.16.112.50	Secret	5	4
	11:05:10	12:25:16	172.16.112.1	Ipsweep	-	2
	12:25:16	14:11:15	172.16.112.50	Mailbomb	6	5
	14:11:15	16:06:17	172.16.114.50	PHF	13	9
	16:6:17	-	172.16.112.50	Httptunnel	-	-
Wednesday	10:02:13	11:44:18	172.16.114.50	Satan	4	9
	11:44:18	13:25:18	172.16.112.50	Mail bomb	9	3
	13:25:18	18:17:10	172.16.114.50	Perl	19	11
	18:17:10	21:23:00	172.16.112.1	Ipsweep	-	2
	21:23:00	21:56:14	172.16.112.50	Eject	-	-
	21:56:14	-	172.16.112.100	Crashiis	-	-
Thursday	6:04:17	7:33:17	172.16.112.100	Crashiis	162	162

	7:33:17	8:50:11	172.16.114.50	Satan	34	11
	8:50:11	9:04:16	172.16.114.50	Portsweep	808	808
	9:04:16	10:57:13	172.16.114.207	Neptune	35	28
	10:57:13	12:25:17	172.16.114.50	Secret	14	10
	12:25:17	13:47:15	172.16.114.50	Perl	2	2
	13:47:15	14:36:10	172.16.112.50	Land	6	3
	14:36:10	17:16:18	172.16.112.1	Ipsweep	-	2
	17:16:18	-	172.16.112.50	Ftp write	2	2
Friday	6:07:17	6:10:40	172.16.114.50	Phf	-	4
	6:10:40	6:16:46	172.16.114.50	Perl	1	1
	6:16:46	7:18:15	172.16.112.50	PS	1	4
	7:18:15	9:20:15	172.16.113:84	Pod	27	21
	9:20:15	10:40:12	172.16.114.50	Neptune	12	6
	10:40:12	11:12:17	172.16.112.100	Crashiis	18	18

	11:12:17	12:06:17	172.16.113.50	Loadmodule	5	2
	12:06:17	12:24:18	172.16.114.50	Perl	2	-
	12:24:18	13:24:16	172.16.112.50	Ps	7	2
	13:24;16	15:13:10	172.16.112.50	Eject	65	9
	15:13:10	15:43:18	172.16.112.50	Portsweep	4955	5001
	15:43:18	-	172.16.112.50	Ftp write	-	-

Table 4-3: Captured Results Packet

Type	**Name of attack**	**NO . of Alert**	
		In side	**Out side**
Probes	**NTinfoscan**	**65**	**63**
	IPsweep	**-**	**6**
	Satan	**38**	**20**
DOS	**Pod**	**27**	**21**
	Portsweep	**6761**	**6809**
	Mailbomb	**15**	**3**
	Back	**123**	**91**
	Land	**6**	**3**

	Crashiis	**180**	**180**
	Neptune	**47**	**34**
U-to-L	**Httptunnel**	**34**	**2**
	Phf	**13**	**13**
	Ftp-write	**2**	**2**
U-to-R	**Eject**	**68**	**12**
	Ps	**8**	**6**
	Perl	**23**	**13**
	Loadmodule	**5**	**2**
Data	**Secret**	**14**	**14**

4.2 Performance Rate

Performance is depending on speed of snort to analysis traffic attached throughit.

The Performance of snort calculated as follow:

Performance = total size of packets (kb/s)

Total time

Total size of packet is total value of size for all those are analysis by snort .

Total time is total time those are need to analysis packet .

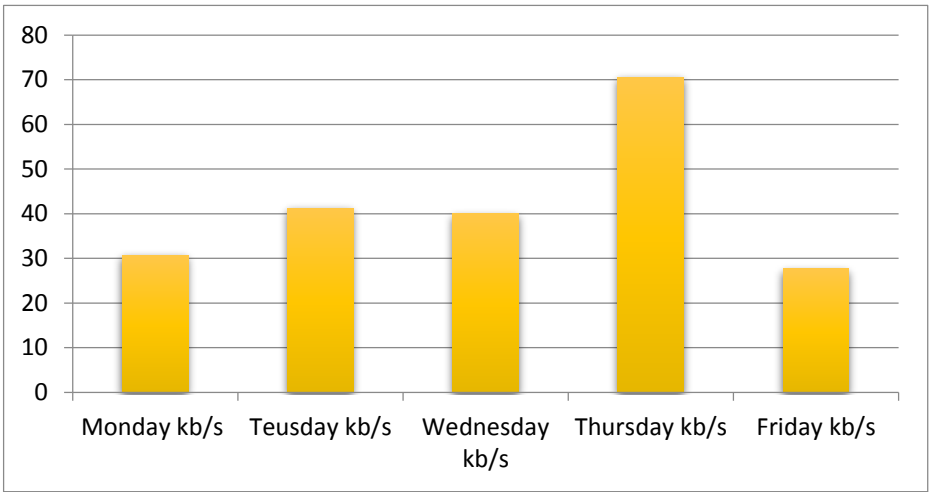


Figure (4.2) Performance in inside packet

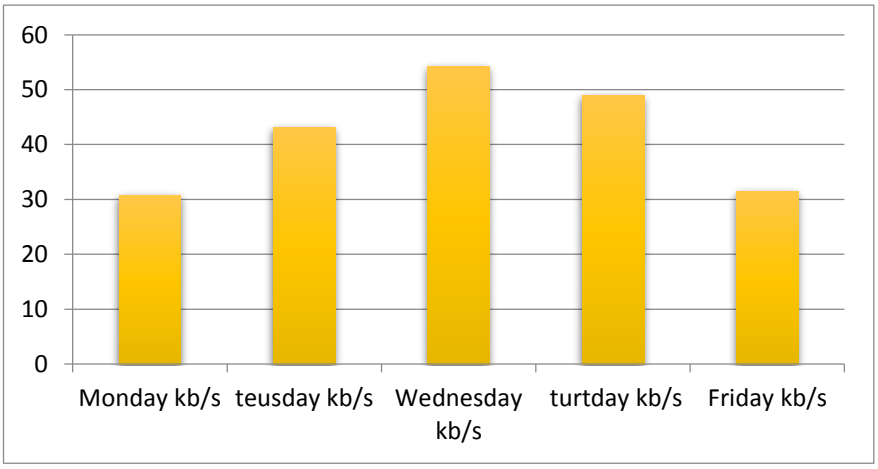


Figure (4.3) Performance in outside packet

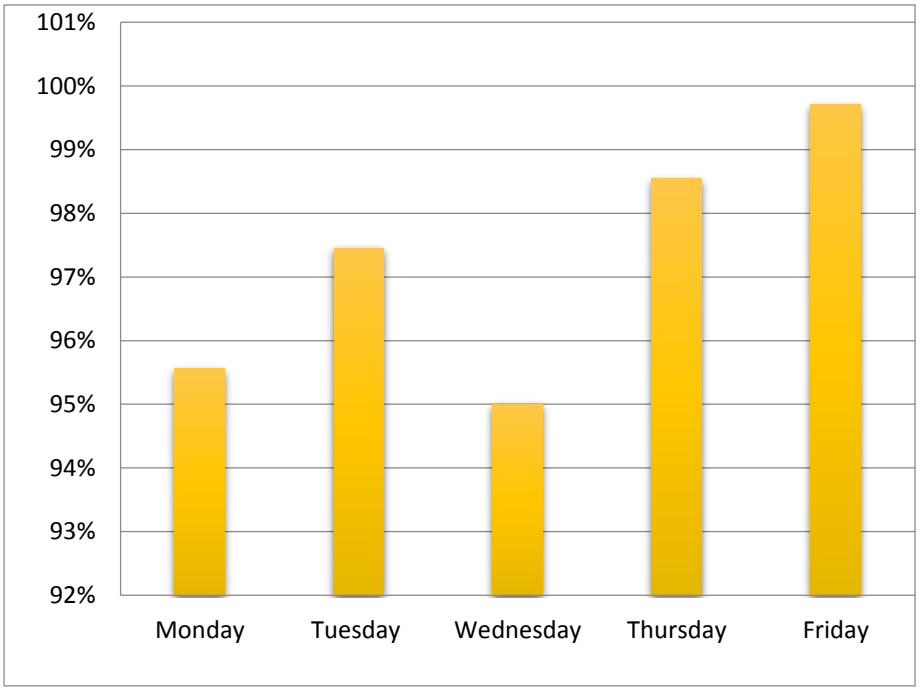
4.3 Accuracy Rate

TP(instance) is number of alerts in snort defined as attack correlation with attack in ground truth table .

FP(intance) is number of alert in snort define as attack and not correlation with any attack in groung truth table .

$$\text{Precision} = \frac{FP_{INSTANCE}}{FP_{INSTANCE} + TP_{INSTANCE}}$$

Figure 4-4: precision in days



CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

CHAPTER FIVE

5. CONCLUSION AND RECOMANDITONS

5.1 Conclusion

5.2 Suggestions for Future Work

5.1 Conclusion

Snort was designed to fulfill the requirements of network intrusion detection system. It has become a small, flexible, and highly capable system that is in use around the world on both large and small networks. This thesis has focused on determining the accuracy , performance of snort and detecting new attacks .The work has been done based on the objectives of the research as follows:

Offline Performance is done by setting and configuring snort Capturing a benchmark (DARPA1999) and analyzing , then the performance has been calculated by using size of a benchmark (DARPA1999) and total time .Accuracy has been calculated by comparing (DARPA1999) with ground truth and percent of efficient accuracy is 99.99%. After calculating he accuracy of the snort and knowledge of its ability to detect all intrusion in the system or not ,it was noted that the accuracy was very high because the snort generates alert only in the event of an attack whether this attack occurred actually or a false alarm . but when calculating the accuracy of it , has been ignored some of the measurements such as (true negative , false negative) because it not have the ability to detect such intrusion and ignore them and the inability to analyze which led to the increase of special precision with it.

The rustles that have been obtained for performance rate is 40.23 kb/second. Therefore , snort can deals better under that performance rate in offline traffic , if the rate becomes higher the performance will be reduced .

5.2 Suggestions for Future Work

For future researchers to make more enhancements in this research:

- that has been faced is the use of an old dataset (DARPA1999) , so our suggestion is to use the last update of dataset ,because more attacks defined each single year.
- Evaluate snort in real time system using an online traffic .
- New rules for detection can be created and added to snort.
- Comparing between other open sources in NIDS like evaluate snort and bro programs .

References

1. Alaidaros, H.M., M. Mahmuddin, and A. Al Mazari. From Packet-based Towards Hybrid Packet-based and Flow-based Monitoring for Efficient Intrusion Detection: An overview. in Second International Conference on Communication and Information Technology. 2012.
2. Binde, B., R. McRee, and T.J. O'Connor, Assessing outbound traffic to uncover advanced persistent threat. SANS Institute. Whitepaper, 2011.
3. Gómez, J., et al. Design of a snort-based hybrid intrusion detection system. in International Work-Conference on Artificial Neural Networks. 2009. Springer.
4. Khan, M.A., A survey of security issues for cloud computing. Journal of Network and Computer Applications, 2016. **71**: p. 11-29.
5. Kumar, G., Evaluation metrics for intrusion detection systems-a study. International Journal of Computer Science and Mobile Applications, II, 2014. **11**.
6. Rehman, R.U., Intrusion detection systems with Snort: advanced IDS techniques using Snort, Apache, MySQL, PHP, and ACID. 2003: Prentice Hall Professional.
7. Rødfoss, J.T., Comparison of open source network intrusion detection systems. 2011.
8. Schaffrath, G. and B. Stiller, Conceptual integration of flow-based and packet-based network intrusion detection. Resilient Networks and Services, 2008: p. 190-194.
9. Soleimani, M., et al. RAAS: a reliable analyzer and archiver for snort intrusion detection system. in Proceedings of the 2007 ACM symposium on Applied computing. 2007. ACM.
10. Yuan, W., An intrusion detection system on network security for web application. 2006.
11. Alaidaros, H., M. Mahmuddin, and A. Al-Mazari, An overview of flow-based and packet-based intrusion detection performance in high speed networks. 2011.