

Network Intrusion Detection Systems - NIDS



- Sarthak Sharma
- Manuvjeet Singh Kondal
- Aditya Aggarwal
- Jagrit Singh

About Institute



CENTER FOR DEVELOPMENT OF
ADVANCED COMPUTING

C-DAC represents a unique facet working in close junction with MeitY to realize nation's policy and pragmatic interventions and initiatives in Information Technology. As an institution

The centre is engaged in design and deployment of world class IT and electronics solutions in the following domains:

- Health Informatics
- Multilingual Technologies
- Professional Electronics
- Software Technologies
- Cyber Forensics and Security
- Multimedia Technologies

Acknowledgment

Many thanks to the Faculty of C-DAC Institute, to our project supervisor—Mr.Sanjay Khandelwal, and all who in one way or another contributed to the successful completion of this project.

Contents

Declaration	i
Approval.....	i
Dedication	ii
Acknowledgment	iii
List of Acronyms.....	ix
Abstract	x
1 Introduction – Aim & Goal	1
1.1 Background of the study	1
1.2 Statement of the problem	4
1.3 Objectives	5
1.3.1 General Objective.....	5
1.3.2 Specific Objectives.....	5
1.4 Scope.....	5
1.5 Significance of the Study	6
2 Literature Survey	7
2.1 Introduction.....	7
2.2 Overview of Computer Attacks.....	9
2.2.1 Why Intrusion Detection Systems	11

2.3	Types of Intrusion Detection Systems	15
2.3.1	Misuse detection (Knowledge-based intrusion detection)	15
2.3.2	Anomaly detection (Behavior Based Intrusion Detection)	16
2.3.3	Hybrid Intrusion Detection Systems (Anomaly-based plus Signature-based intrusion detection systems)	19
2.4	IDS Architecture	20
2.4.1	Host-based Intrusion detection System (HIDS).....	20
2.4.2	Network-based Intrusion Detection System (NIDS)	23
2.4.3	Distributed Intrusion Detection Systems.	24
2.5	Network Security Tools today	26
2.5.1	OSSEC HIDS	26
2.5.2	Basic Analysis Security Engine (BASE)	26
2.5.3	Sguil	26
2.5.4	Snort	27
2.5.5	Other Network Security tools:	27
2.6	Implementation Strategy	27
3	Methodology	29
3.1	Introduction	29
3.2	Model Description and Design.....	29
3.3	Data Collection and Analysis	29
3.4	The Network Setup	31
3.5	Configuration and Validation of the IDS	32

3.5.1	Snort for Linux	32
3.5.2	Snort for Windows/Linux	35
	
3.6	Service Configurations	37
3.6.1	Firewall Configuration	37
3.6.2	DHCP Configuration	38
3.6.3	Configuring the Mail Server	38
3.6.4	HIDS and NIDS Configuration	38
3.6.5	Intrusion and attack Simulation	38
3.6.6	Validation of IDS	39
4	<i>PRESENTATION AND DISCUSSION OF TOOLS & TECHNIQUES</i>	41
4.1	Current Technology.....	41
4.2	Conceptual framework for an effective intrusion detection system	43
4.2.1	Implementation results	46
5	Summary, Conclusion and Future Scope	54
5.1	Introduction	54
5.2	Recommendation.....	54
5.2.1	Implementation	54
5.2.2	Good Practices.....	55
	APPENDIX A - SNORT CONFIGURATION FILE.....	62
	APPENDIX B - SNORT RULES FILE	65
	APPENDIX C - SNORT MYSQL DATABASE.....	66
6	References	69-72

List of Figures

Figures.....	Page
2.1.1 Possible locations of an IDS.....	8
2.4.2 HIDS with agents that send information to the agent console.....	21
3.4.1 A model setup of a network with an (IDS).....	31
3.5.2 IDS-Snort program(1)	36
3.5.3 IDS-Snort program(2)	37
3.6.4 Verification of Network traffic using Packet Statistics	4
4.1.1 Stand alone architecture	41
4.1.2 Networked computers connecting to the Internet through a firewall.....	42
4.2.3 Conceptual framework for an effective intrusion detection system	43
4.2.4 Network query tool	47
4.2.5 Base alert monitor.....	48
4.2.6 Base graph for TCP port source against alerts	49
4.2.7 Base graph for TCP port source against alerts	50
4.2.8 NTOPNG graph for traffic monitoring	51

4.2.9 NTOPNG graphical impression of network load statistics.....	52
4.2.10 NTOPNG summarized traffic distribution.....	53
4.2.11 Other NTOPNG features.....	54

List of Acronyms

AI	Artificial Intelligence
BASE	Basic Analysis and Security Engine
CERT	Computer Emergency Response Team
DHCP	Dynamic Host Control Protocol
DNS	Domain Name System
DoS	Denial of Service
FDDI	Fiber Distributed Data Interface
HIDS	Host-based Intrusion Detection Systems
ICMP	Internet Control Message Protocol
ISP	Internet Service Providers
IDS	Intrusion Detection Systems
IPS	Intrusion Prevention Systems
NIC	Network Interface Card
NIDS	Network-based Intrusion Detection Systems
NSM	Network Security Model
RADIUS	Remote Authentication Dial In User Service
RDBMS	Relational Database Management System
SSH	Secure Shell
TACACS	Terminal Access Controller Access System
TCP	Transmission Control Protocol
VLANS	Virtual Local Area Networks
VPN	Virtual Private Networks

Abstract

As organizations go online, a number of advantages are realized and as a result academic institutions have not been left behind in this move. However most of the techniques organizations have used to guard against unauthorized system access have been found lacking and as a result many disastrous unauthorized access have been reported. This is worsened by the fact that applications used to break into systems are now easily accessed in some shops and the Internet. The interest of this work is to adopt an Intrusion Detection System (IDS) for academic institutions to provide early detection and prevent network intrusion. The idea is to provide an integrated system that will minimize the weaknesses of the different intrusion prevention techniques while putting to the most the strength of each of them. We define IDS, discuss the different IDS types, architecture, and compare different IDS's and look at an effective implementation strategy.

Chapter 1

Introduction

1.1 Background of the study

As a result of the various advantages offered by the Internet, businesses have become more open to supporting Internet-powered initiatives such as customer care, e-commerce, and extranet collaboration. However this presents a challenge. Many enterprise networks have been broken into by hackers. A case in point was the Citibank security breach in which by the time the heist was reported in 1994, 10million dollars was already lost. Only 400,000 dollars was eventually recovered.

It was observed that there was a high number of unauthorized security events, and in the year 2000, 70 percent of organizations at least reported a security incident. This represented a 42 percent increase from the 1996 report. continueto write that the Computer Emergency Response Team (CERT) reported 3734 incidents in 1998,9859 in 1999 and within only the first six months of 2000, 8836 incidences where already reported. These are but just a few of the identified and published forms of computer system attacks.

Academic institutions have not been left behind in this move, many of their resources are online. As a result, there has been an increasing need to protect these financial data, examinations, lecture notes and other information that may be considered critical such as student transcripts.

Academic institutions are characterized by limited resources and therefore many times will need to have these resources shared between the lecturers, students, permanent and part time staff. In addition students are at different levels of education such as Diplomas, undergraduate level and postgraduate students. Similarly part time and full time lecturers will also require different network access levels. This will thus call for sections of the network to have relatively higher security. Some Internet sites such as pornographic and those that may provide immediate solutions to online examinations may need to be controlled or restricted. This will not only improve on network security but also bandwidth utilization. Students many times turn out to be so inquisitive as to exploring every area of the network and hence the need to protect networks with critical information and part time staff need not be entrusted with all the network resources. This highlights the need for an appropriate system to detect unauthorized access to these resources on specified sections before incurring serious damages.

Organizations are striving to maintain confidentiality, integrity and availability of their networked resources and a number of techniques have been employed to guard against network intrusion. However, even though these measures provide a level of security, they have been found to be lacking in a number of ways.

1. The use of firewall. A firewall is a hardware or software solutions used to enforce security policy on a private network. It is mostly used to control traffic to or from a private network. However, these are but just a list of permit and deny rules, therefore they may not always have the ability to detect intrusions.

Firewall, user authentication, data encryption and Virtual Private Networks (VPN) provide a level of security but they are limited by the fact that they cannot give protection against malicious codes, inside attacks or unsecured modems. They therefore

would only be effective as one of the available lines of defense.

2. For institutions that already have intrusion prevention systems, perfectly secure system are hard to come by. There are always a number of system flaws in addition to possible administrator configuration errors. Intrusion detection systems can thus be used to supplement the already existing systems.
3. Many times intrusion prevention systems are implemented on expensive routers. These are sometimes not flexible enough to allow changes such as in network design and topology. Intrusion detection systems provide the flexibility needed to provide adequate security without necessarily acquiring specialized hardware.
4. Cryptography hides information from unauthorized users, however this method makes it hard to know whether any attack has taken place.

Generally key management is not an easy task. Crypto systems may require special key management systems such as the use of a Terminal Access Controller Access System (TACACS) or Remote Authentication Dial in User Service (RADIUS) server. This could mean specialized hardware or configuration. Otherwise hackers could gain access to these keys and break into the system.

5. The provision of physical security to the network site or to servers. However these are limited by the fact that physical security may not provide a practical solution to attackers who employ telnet sessions to gain access to a network.
6. Authentication. A technique used to verify users of a network resource. The effectiveness of this is weakened by the fact that many still “use easy to crack passwords” while some user

are either un-trustworthy or are just careless with their passwords such that many times can easily be got by unauthorized users.

7. Many organizations have also employed anti viruses, however this may not provide information as to whether there has been an intrusion or not. Anti-viruses also require frequent updates.

Denning also presented four basic motivations for IDS's. He observed that most systems have security flows susceptible to intrusion, most of the systems are not easily replaceable, difficulty to develop perfectly secure systems and available systems are vulnerable to abuse such as by inside users. This shows a greater need for intrusion detection systems.

1.2 Statement of the problem

As computer systems go online, institutional resources get vulnerable to unauthorized access. On the one hand, sophistication of hacker tools has been faster than the technical knowledge required to counter the hacker techniques. This calls for a means to counter these threats. Academic institutions are also going online because of the advantages offered and yet many have not given network security a priority in their budget. They thus remain prone to these threats. In addition their networks are shared among students, lecturers and at the same time having critical information such student academic transcripts and financial reports. This calls for even more vigilant monitoring. These networks left unplanned will lead to heavy financial losses resulting from intrusion and inefficient resource utilization such as bandwidth waste through unwarranted server request from network nodes. Deploying only a single intrusion prevention technique such as a firewall may not always be effective (See Section 2.2). All this point to the need for a system that will provide intrusion detection in a manageable way.

1.3 Objectives

1.3.1 General Objective

The project aims at adapting an intrusion detection system appropriate to academic institutions, emphasizing the importance of these to network security. It points to the need for integration of various intrusion prevention mechanisms so as to harden the intrusion detection system. This work takes into account the limitation of resources available to academic institutions in providing network security.

1.3.2 Specific Objectives

The specific objectives of the research has been to:

1. Investigate Network Intrusion Detection Systems (IDS).
2. Design an effective intrusion detection system.
3. Implementation of an effective scalable intrusion detection system.
4. Validation of an intrusion detection system.

1.4 Scope

The project has looked at an internal intrusion detection system and has concentrated mainly on intrusion detection systems in medium sized education institutions. This work does not look at the internal components of an Intrusion Detection System (IDS) but looks at the intrusion detection system as a black box.

1.5 Significance of the Study

An institution would always try to have a fast network, a conducive working environment free from viruses and without disruptive messages this will help reduce on bandwidth utilization. Well planned intrusion detection will also simplify network management. As the network expands, combining different techniques gives a better coverage and more effective intrusion detection and hence prevention. In the long run, deploying IDSs greatly cut down on costs as a result of unauthorized access, allowing network managers or system administrators engage on other productive endeavors. Online examination malpractice will greatly be reduced.

Chapter 2

Literature Review

2.1 Introduction

The chapter presents an overview of computer attacks and some of the techniques employed against intrusion. IDS architectures, models and implementations are also discussed.

An intrusion is a successful violation of a network's security policy.

An Intrusion Detection System (IDS) is an ad hoc security solution to protect flawed computer systems. The system sends a notification or takes an action should an intruder attempt to go past the security mechanism such as authentication, or firewall.

Intrusion Detection Systems (IDS) maybe defined as systems that have the ability to detect both internal and external attacks on a computer system and undertake some measures to eliminate them.

Intrusion Detection Systems not only detect intrusion but also identify failed intrusion attempts, providing necessary information for precautionary measures and sometimes counter intrusions.

Intrusion detection systems will detect an intrusion or intrusion attempt by examining features such as:

1. Network traffic,
2. CPU and Input/Output(I/O) utilization,

3. File activity or even user location for signs of attacks (Herringshaw, 1999) [10].

Apart from sending a notification in the form of an electronic mail or by sounding an alarm, the system could also be configured to terminate a TCP (Transport Control Protocol) session and replace the connection with the administrators connection in cases of breaches to the security policy. On the other hand the systems could log suspicious activity for subsequent review.

Normally, for the security of networks, IDS's will have monitoring agents or sensors on local networks, between subnets or on links to remote networks such as the Internet. The agents analyze traffic for any signs of attack and send this information to a management console.

The IDS could be placed just behind a network firewall or behind a perimeter router as shown next on Figure 2.1.1 to monitor network traffic.

INTRUSION DETECTION SYSTEM

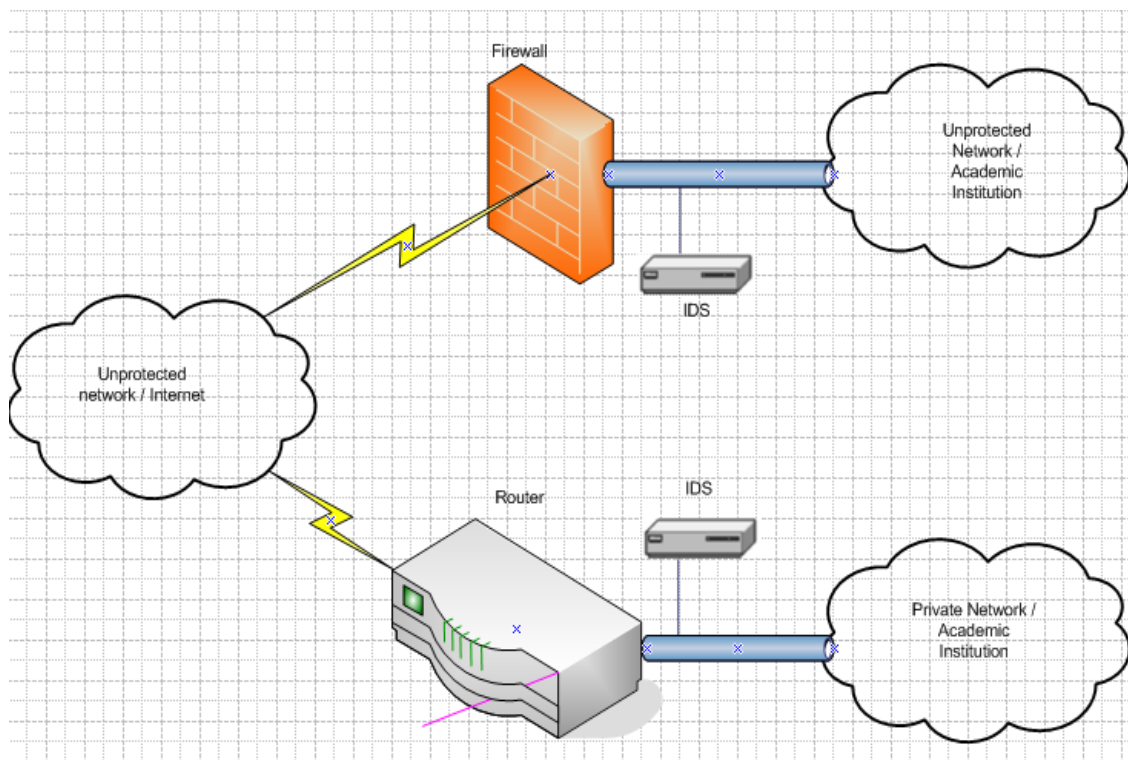


Figure 2.1.1: Possible locations of an IDS

Data sources used to detect intrusion could also be from audit trails produced by the operating system, network traffic flowing between systems, application logs or data collected from system probes for example the file system alteration monitors. Which data can be analyzed in real time or stored for batch mode analysis.

2.2 Overview of Computer Attacks

A computer attack is any malicious activity directed at a computer system or the services provided by the system. The attacks could come in the form of viruses, Denial of Service (DoS), exploitation of bugs, use of services by those unauthorized or even a physical attack against the computer hardware.

An understanding of some techniques that have been used by crackers or even script kiddies will go a long way in helping provide a system that will protect institutions against the intrusions.

Hackers have employed a varied number of techniques to break into systems and these include and are not limited to:

1. Social Engineering: This is the type of attack where the attacker fools an authorized network user into divulging information that leads to access to network resources. The details may include: passwords, security policies, network or host addresses. The attacker could also impersonate genuine users, service providers or the support staff and sometimes uses this identity to deliver harmful software such as Trojan.
2. Abuse of Feature: In abuse of feature, the authorized users perform action that are considered illegitimate. He may open illegitimate sites, send request or open up telnet sessions with into prohibited servers or hosts that could for example fill up the user's assigned quarters or storage space.

3. Configuration errors - The attacker may gain access to a network with configuration errors.

These weaknesses could be allowing access without prompting for any form of authentication such as assigning user's guest accounts which are left without password requirements. Users may erroneously be assigned administrative privileges. A user may also obtain privileges of other users by obtaining their passwords or bypassing control that restrict access.

4. Masquerading - Having monitored the flow of traffic, the attacker can modify a TCP packet or originate a packet but send it with a forged source address to give the impression that it is from a trusted source. The attacker can there in send Trojan, perform a denial of service or make request for sensitive data.

5. Worms - These are destructive self-replicating programs that spread across the network.

They can be sent as e-mail attachments to hamper network performance.

6. Viruses - Are programs that replicate when a user performs some actions, such as running a program. The viruses may have the ability to destroy sensitive documents.

7. Implementation Bug - Most trusted applications and programs have bugs. These bugs are exploited by hackers to gain access to the computer systems or networks. Examples of these include: buffer over flows, race conditions or even mishandled files. This is also referred to as client attack, where the attacker may exploit bugs on either the client, server, network software or protocol.

Generally these attacks can be classified into four major classes, interception, interruptions, modification and fabrications.

1. **Interception:** This means that some unauthorized party gains access to an asset. This could be a program, person or a computing system. An example of this could be wiretapping, or illicit copying of program or data files. The damage could be worsened when the attacker leaves no traces on the network.
2. **Interruption:** In this situation an asset of the system is made unavailable or unusable. An example is a malicious destruction of a hardware device, erasure of a program or data file or malfunctioning of an operating system so that it does not get say a particular disk.
3. **Modification:** The attacker both accesses and tampers with the asset. This may include altering the program so that it can perform an additional computation or modify data being transmitted. They may range from simple changes to more subtle changes that may not even be detected.
4. **Fabrication:** This involves creating counterfeit objects on the computing system. When skillfully done may also go undetected and thus a very serious threat to the network security.

2.2.1 Why Intrusion Detection Systems

While most of the above mentioned are prudent steps towards network security, an IDS provides an in depth by detecting and logging hostile activities. It can watch over the network whenever other protective measures are bypassed.

We also need IDS despite having a firewall because firewalls simply shut off all communication and turn on only a few well-chosen traffic. It does not have the capability to detect internal attacks. Firewalls are mostly employed at the boundary of the network, and thus may only control traffic entering or leaving the network. And yet a greater percentage of potential hackers may be from within the network. Protecting the network from internal attacks for a boundary firewall will thus

be almost impossible.

Some of the reasons therefore for using IDS include:

1. To double check erroneously configured firewalls. The goal of security professionals is to provide products that will both mitigate intrusion while at the same time allowing access to essential network services. Therefore other than providing a prudent approach toward network security like other intrusion prevention mechanisms, IDS's will help network professionals with added features that will not only provide for intrusion detection but also provide more network management functions.
2. To catch attacks that firewalls illegitimately allows through, for example the attacks on web servers. It is critical that when systems are subverted, the administrator is alerted. Intrusion detection systems will do this by sending alarms or mails to a console.
3. IDSs also through their logs allow catching of failed attempts.
4. Catches inside attacks. Some of the biggest dangers to any corporate network are its own internal users situated within the network perimeter since firewalls that are mostly being used, are deployed on the network perimeter. And yet attacks originating from within the network can result into serious legal difficulties or even Internet connectivity. IDS's can monitor both internal and external attacks.
5. Costs. For a long time network tools have been expensive and free tools were almost exclusive to UNIX or LINUX systems however this is no longer the case with the advent of WinCap, WinDump and NMap for windows. The addition of the GUI interfaces as a feature on most open source applications such as snort has further changed this view. As a result it is no longer as costly to acquire them.

ABILITIES OF INTRUSION PREVENTION MECHANISMS

Solutions	Firewalls	Anti-Viruses	Intrusion Detection Systems (IDSs)
Ability to control internal attacks	No	Limited way	Yes
Ability to log intrusion attempts	No	Limited way	Yes
Ability to provide a history for attacks	No	Yes	Yes
Ability to send alerts	No	No	Yes
Ability to detect attacks	No	Limited way	Yes
Ability to reacting to attacks	No	Limited way	Yes

Table 2.1: Comparing intrusion prevention mechanisms with IDS

6. IDSs provide information that can be used as evidence for criminal investigations. This scares off would be attackers in the fear of being traced and yet this could also be used to analyze the vulnerability of a network.

Table 2.1 next gives a basic comparison between Intrusion Prevention systems (IPSs) commonly used today and intrusion detection systems.

However although IDSs are valuable additions to an organizations security system, it is good to note that they also have shortcomings to be guarded against. IDS's have limits as to what they will do. This should therefore be put into consideration whenever any IDS is implemented.

1. Intrusion detection systems will not compensate for any missing security mechanism in the prevention infrastructure such as firewalls, authentication or access control mechanism, and cannot therefore be exclusively be relied upon to provide network security.
2. They may not function effectively on heavy-network traffic.
3. Intrusion detection a straight forward process given the fact that every other day hackers are finding more complex ways of breaking into networks hence the need to continually update intrusion signatures.
4. IDSs will not automatically investigate attacks without human intervention.

5. Most intrusion detection systems do not have enough correlation to provide for a perfect intrusion detection. This is especially true when the network administrators do not specifically relate the signatures to the attack.

Apart from these and other limitations, IDSs on a network greatly enhance the security of the network. Their effectiveness can be enhanced by reducing on the limitations mentioned above which can be done by deploying them as part of an integrated system.

Unfortunately, many institutions have not used intrusion detection systems. And this has been basically because:

1. There has been a general lack of awareness and increasing concerns of manageability and cost aspects of the general organizational setup. Managers are more inclined towards getting quick results.
2. Most institutions have not appreciated the fact that connection to the Internet can expose critical data to the public. The intranets are placed under the mercy of the service providers in most cases when they are given all the installation responsibilities. It becomes possible for them to remotely log onto the system at the customer premises should acquire the necessary details.
3. Many organizations and institutions just have not seen the relevance of IDS in providing secure networks and hence protecting their critical resources. Many are ignorant about the dangers associated with going online. As such even the relatively large institutions and organizations have not taken heed of the dangers on the Internet and therefore not deployed any serious measures to enhance network security.

2.3 Types of Intrusion Detection Systems.

The success of an intruder is highly dependent on the type and architecture of the intrusion detection systems that are in place. There are a number of possible Intrusion detection system types and architectures that could fit an organizations security goals. We now look at the different types of IDSs, the different IDS architectures showing their strengths and weaknesses.

There are different Intrusion detection types, which types handle intrusion differently. Classified intrusion detection systems in two groups:

1. Misuse detection, and
2. Anomaly detection

2.3.1 Misuse detection (Knowledge-based intrusion detection)

In misuse detection, attacks follow well-defined patterns that exploit system weaknesses and application software. Since these attacks follow well-defined patterns and signatures, they are usually encoded in advance and thereafter used to match against the user behavior. It implies that misuse detection requires specific knowledge of given intrusive behavior.

IDSs operate on the assumption that any traffic will be accepted unless it has been marked as prohibited. The advantage here is that there are low false positives as long as attacks are clearly defined in advance.

However this type of intrusion detection systems have a number weaknesses.

1. It can be seen that misuse detection requires specific knowledge of intrusive behavior. Collected data before the intrusion could be out of date and yet many times it is hard to detect newer or unknown attacks.

2. Misuse detection has a well-known problem of raising alerts regardless of the outcome. For example a window worm trying to attack a Linux system, the misuse IDS will send so many alerts for unsuccessful attacks which may be hard to manage.
3. Maintenance of the knowledge base of the intrusion detection systems requires careful analysis of each of the vulnerabilities which will demand for lots of time. It is also rendered inefficient by the fact that vulnerabilities are being created and identified by unauthorized users “every other day”.
4. This model may not always be so practical for inside attacks involving abuse of privileges.
5. The knowledge about attacks is very dependent on the operating system, version and application hence tied to specific environments.

2.3.2 Anomaly detection (Behavior Based Intrusion Detection)

Sometimes referred to as statistical intrusion detection, analyzes audit-log data for abnormal user and system behavior. A breach or deviation from normal system use is taken for an intrusion. Anomaly detection is also referred to as behavior based detection or generic intrusion detection model.

However this can result into a number of false positives as not all deviations from normal use can be attributed to intrusion.

Dorothy Denning had observed from a pioneer research work on intrusion detection system and had reported an intrusion detection model called the Intrusion Detection Expert System (IDES). This system employed a model independent of any particular system, application, system vulnerability or type of intrusion. It maintains a set of profiles for subjects and monitors system services such as file access, executable programs and logins.

Every record generated by the system is matched with an appropriate profile and then the system makes decisions on updating the profiles. Six main components were described in this work and they are:

1. Subjects — These are the typical users of the network resources
2. Objects — These are system resources such as files
3. Audit records — Output generated by the system when subjects perform or attempt to perform actions on the objects
4. Profiles — Data that shall characterize normal behavior of the subjects on particular objects.

The model maintains a set of profiles usually for users though sometimes it may include profiles for more than just a user's.

5. Anomaly records — Are generated when audit records are considered abnormal with respect to a given profile.
6. Activity rules — Actions taken when a condition is satisfied.

In behavior based intrusion detection systems, logs are processed to see whether new profiles can be detected that deviate from accepted profiles. When an audit record is generated, this model compares the profile with the profiles considered normal and makes reports of any anomalies detected.

However the challenge encountered here is that of developing profiles of normal behavior that will take into account all the users, the system itself and network resources. This has to be done to avail statistical information that will define normal behavior.

1. High rate of false alarms — It is most unlikely that the entire scope of the behavior of an information system is covered. This will mean that there is likelihood that an intrusion is considered as normal traffic especially if its profile is not well defined in advance.
2. Behavior changes over time. Hence the need for periodic retraining of the profiles. This calls for a lot of time and resources.
3. It is also possible that the systems may undergo attacks when the system is still learning the behavior.

Abnormal behavior could be things that are there and are not supposed to be there or things that are supposed to be there but are not there.

An implication that there should be a good trend analysis to develop meaningful profiles that can be used to analyze audit records for intrusion detection. There are five classes of events that are significant in developing trends to identify possible security deviations:

1. Activities of the system as a whole
2. Activities of users
3. Activities of particular terminals
4. Transactions involving particular sensitive files or programs
5. Transactions involving particular sensitive system files or programs Other examples include:
6. Variation in normal login time or resource usage of resources and types of commands that are being used.

7. Variation from a normal pattern of usage — this is authorized user penetration.
8. Variation in CPU or I/O resources — this may be an indicator of a Trojan attack.
9. Increase in storage requirement of executable programs or unusual number of executable programs being rewritten. The possible cause being a virus that may be replicating itself onto different files.
10. Abnormally high resource usage by a user relative to other users. This could point to a Denial of Service (DoS) attack.
11. High password failure rate — unauthorized users may be attempting to access a protected resource. This is referred to as dictionary attacks.

However, the above-mentioned symptoms may not necessarily point to an intrusions.

1. The fact that hackers are all out to beat any form of system protection means that they have always invested a lot of time in this, therefore new attack methods are devised within short periods of time. Available profiles may not always provide all the security. Sometimes attack symptoms only appear after damages have already been incurred.
2. High rates of password failure may be as a result of other causes other than intrusion attempts. It may be as a result of periodic password changes as stipulated in the organizations security policy and users may take time to master their own passwords.

2.3.3 Hybrid Intrusion Detection Systems (Anomaly-based plus Signature-based intrusion detection systems)

To try to reap from the advantages of both the signature-based and Anomaly-based intrusion detection systems while eliminating some of their disadvantages, vendors have developed hybrid

systems that are rule based, checking for known attacks and at the same time use Anomaly based feature to protect against new types of attacks.

However the future of intrusion detection to ensure better efficiency is heading toward Artificial Intelligence (AI). This could be beneficial as they could let Anomaly detection track changes in profiles and thus reduce the number of false alarms. Systems will extend intrusion detection to the software level to cover Java, Shockwave, and other advanced Internet technologies.

2.4 IDS Architecture

Not only will an effective IDS architecture enhance the IDS but it will also maximize on the bandwidth utilization. This is especially important when it comes to deciding which and how many machines are to be protected. There are three basic IDS architectures that have been proposed for intrusion detection systems and they are:

1. Host-based IDS
2. Network-based IDS
3. Distributed IDS

2.4.1 Host-based Intrusion detection System (HIDS)

A piece of software is loaded onto a system to detect intrusion. The software uses log files or system auditing agents, which look at communication traffic. The software then checks the integrity of system files. Agents are installed on publicly accessible servers such as corporate mail servers or application servers. The agents then report events to a central console that is protected by an agent software. What may be monitored for intrusion detection include:

1. Log file analyzers - analyzes log files for patterns that indicate intrusion
2. File system monitor - monitors the system to check for integrity of files and directories
3. Connection analyzers - to monitor connection attempts
4. Kernel based analyzer- to detect malicious activity on a kernel

With HIDS as shown in Figure 2.4.2 next, the hosts within the private network will have an intrusion detection system that will send alerts to the agent console from where they are analyzed.

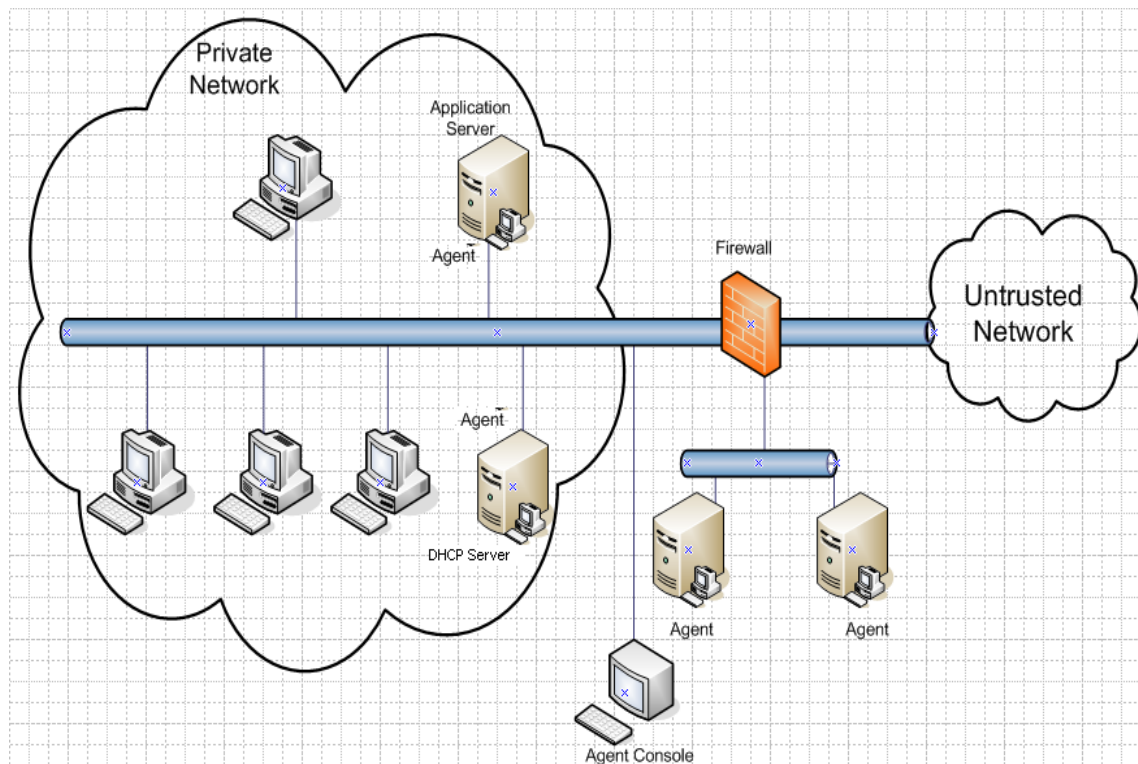


Figure 2.4.2: HIDS with agents that send information to the agent console

Advantages offered by HIDS include:

1. HIDS perform explicit parallel processing. All hosts monitor for intrusion within the machines themselves and a minimum data is broadcasted. In this way each host performs

services only for itself.

2. Secondly, since they are deployed on every host, they eliminate single points of failure that could have been the case with network intrusion detection systems which are mostly applied at the network perimeter to detect intrusion for a number of machines. It is also true that in the presence of high traffic, IDS performance may be compromised.
3. And additionally, HIDS approach can be tailored to various types of hosts.
4. They minimize the weaknesses of NIDS. For example NIDS may not understand all protocols or be able to analyze encrypted data, they also reduce the possibility of having a single points of failure as would be the case with NIDS. It is also possible to detect misuse that could never have passed over the network router such as internal attacks. Network loads may also exceed the capability of the network based detection system. And hence the need for a host based intrusion detection system in addition.

However it adds that HIDS also have a number of shortcomings:

1. HIDS are invasive — Software has to be installed and maintained on each of the protected hosts.
2. High maintenance costs — especially when there are a number of hosts to be managed.
3. Lack of low-level network events — HIDS reside on individual computers and primarily use data from operating system audit trails as their input. Some even make interactive queries of the host system to compensate for data that does not exist.

From the above it can be seen that HIDS's can be practical for a relatively small number of hosts, while not that cost effective for networks with a large number of workstations.

2.4.2 Network-based Intrusion Detection System (NIDS)

Network Intrusion Detection systems (NIDS) monitor the network by capturing network packets. They parse the packets, analyze them and extract useful information from them. The network segment is used as their data sources. All the analysis of the different packets are done without changing or inserting any data on the network.

A sensor is used to monitor packets traveling on that particular segment. The IDS determine if the traffic matches any known signatures. Examples of these known signatures may include:

1. String Signatures— this represents a text string that may indicate a possible network intrusion.
2. Port Signatures— Watches out for connection attempts to well-known ports.
3. Header Signatures represent dangerous header combinations that could characterize intrusion.

NIDS offer a number of advantages and some observed were of the following:

1. They are practical were we have a high number of workstations since it monitors the network as a whole from a particular point of deployment.
2. Monitors a number of hosts simultaneously — The NIDS uses a promiscuous mode, reviewing all network traffic that is received. A process that can take place simultaneously without affecting network performance since no packets are added on the network in the process.

The disadvantage with NIDS is that:

1. NIDS use raw data — Data used by NIDS don't usually provide information to detect low-level vulnerabilities, such as network topology and operating systems on different hosts. This all limits the network administrator's ability to prevent further attacks. Yet trying to overcome this limitation, such as by reassembly of the packet fragments to analyze the network will render the IDS vulnerable to Denial of Service (DOS) attacks.
2. NIDS sometimes miss some traffic when on heavily loaded networks such as on high-speed networks like 100 Megabit Ethernet and Fiber Distributed Data Interface (FDDI).
3. Network aspects — Hackers can also take advantage of network aspects.

Low-level encryption of traffic at the network layer may hide malicious traffic. This in effect denies NIDS from detecting intrusions. It would thus just detect the packet, but have no information about the payload.

Data encapsulated in ICMP packets which are normally not checked, may allow hackers to send malicious scripts into the network.

Multiple routes to the network a hacker may send a malicious sequence of fragments through different routes. If the NIDS are not deployed on all these ingress routes, then the NIDS will be rendered ineffective.

NIDS will be more effective in settings with relatively big number of hosts and deploying HIDS would not be cost effective. And whenever they are deployed, this should be done in at least all network entry points.

2.4.3 Distributed Intrusion Detection Systems.

Despite their advantages, HIDS and NIDS still have a number of limitation arising from the fact they all have to collect data either from audit trails or by monitoring packets in the network to a

Centralized location where they are analyzed. And the problems associated with these are:

1. Because of having a centralized analyzer, a single point of failure is introduced. Should the attacker fail the central point, then intrusion detection will be ineffective.
2. There is limited scalability, since after a given limit the central analyzer will not be able to keep up with the flow of traffic.
3. It is difficult to reconfigure or add capabilities to the system since this would require editing a configuration file, adding an entry to a table or installing a new module.
4. The attacker is given the possibility of performing insertion or invasion attacks when analysis of network traffic is done at a single point.

This will thus call for a distributed intrusion detection system (DIDS). They consist of independently running entities that can be removed or added without unnecessarily having to restart the IDS. This reduces on the problem of HIDS and NIDS.

Distributed intrusion detection systems are also described as a system that uses varying techniques to combine elements of both NIDS and HIDS. They are designed to maximize on the strengths of NIDS and HIDS, while minimizing on their weaknesses.

This could possibly happen in high security networks such as server farms and they may also be implemented in settings where some hosts on the same network will require relatively more security and therefore in addition to NIDS, HIDS may be installed in them. Networks have different security needs. Some networks may require just a HIDS, others will require NIDS while some may need a distributed system. In a nutshell, the type of intrusion detection system architecture will depend on a number of factors that could range from the size of the network in terms of number of hosts, security policy or level of security required.

2.5 Network Security Tools today

Though most institutions are not using any network monitoring tools or intrusion detection systems, there are a number of them currently available. In this section we mention some of those.

2.5.1 OSSEC HIDS

OSSEC is one of the open source host-based intrusion detection systems. Among other things, it performs integrity checking, windows registry monitoring, time based alerting, rootkit detection and active response. OSSEC thus provides both host agent and file integrity checks.

It does log analysis for a number of services such as Bind, Apache and squid when deployed either as a stand-alone or part of a distributed network of agents.

OSSEC will monitor multiple systems and runs on most operating systems such as windows, Linux, Manichitosh and solaris.

However, the server will be configured for the console only, without any X window, KDE or Gnome.

OSSEC is not documented enough to meet user demands.

2.5.2 Basic Analysis Security Engine (BASE)

BASE is a software PHP-based analysis engine used together with other network monitoring tools such as firewall and IDS programs to search and to process a database of security events. It is normally applicable together with other Intrusion detection systems, which means that it will rarely work independently.

2.5.3 Sguil

Sguil is yet another implementation of a Network Security Monitoring (NSM) system. It provides an analyst console for network security monitoring. An NSM is a collection, analysis, and escala-

tion of indications and warnings to detect and respond to intrusions. Sguil consists of components to collect NSM data, and the components includes alert data, session data, full content data and statistical data.

Some advantages with sguil is that it is an open source collection of software components for NSM. It will therefore not cost the institution dearly to implement it. It is also prepared to run over most of the operating systems.

However the sguil client is written in Tk programming language. This is cross-platform toolkit that gives a library of elements for building a graphical user interface (GUI). It therefore requires knowledge of this language to effectively use it.

Sguil is more of an interface used by NSM. By itself it cannot be considered a fully-fledged intrusion detection system.

2.5.4 Snort

Snort is an open source network intrusion detection system capable of performing a real time traffic analysis and packet logging for IP networks. alight weight intrusion detection and prevention system that has excelled in logging and analysis of IP networks. Snort is open source and has the ability to search and detect thousands of worms, port scans and a number of intrusion attempts.

2.5.5 Other Network Security tools:

Other network security tools that have been used are: Netfilter, Cain and Abel, Tcpdump and Knoppix.

2.6 Implementation Strategy

We notice that though intrusion detection systems will not always prevent intrusion from occurring. However a good IDS implementation should:

1. Run continually with minimal human intervention.
2. An IDS should have the ability to recover from crashes and re initializations. It should thus be fault tolerant.
3. It must impose minimal overhead on the system over which it is running.
4. It should allow for configuration as agreed upon by the security policy.
5. The IDS should easily adopt to changes. These changes could be user or system changes.
6. The IDS should also be able to monitor itself for possible subversion or changes by un ethical hackers.

In its implementation, to ensure that the IDS itself is protected from attackers, the following should be ensured:

1. Avoid running any services on the IDS sensor since network servers are among most tar-geted. IDS should run on dedicated machines where possible.
2. Install the latest security patches.
3. Configure the IDS not to respond to ping packet, else they may be a target for a denial ofservice attack.
4. When using Linux configure Net filter or iptable to block any unwanted data.

Intrusion detection systems mainly detect intrusions and provide only a basic intrusion prevention. And therefore given that a network could be as secure as its weakest link, IDSs will need to be supported by tools like Network vulnerability scanners to double check for possible unidentified

holes in the system.

Chapter 3

Methodology

3.1 Introduction

This section demonstrates how the specific objectives of the project will be achieved.

3.2 Model Description and Design

We start by designing a conceptual framework of an integrated intrusion detection system. The framework explains the flow of packets in and out of a protected network. It takes into consideration unauthorized resource access from both within the network perimeter and those that may be within. We then go on to show possible intrusion prevention techniques that will aid the efficiency of the IDS.

3.3 Data Collection and Analysis

The work has used an open source intrusion detection system - Snort is configured to log traffic flowing into a private network 192.168.0.0. The collected data is then used to investigate the relevance of an IDS system on the protected network. Snort has been taken as the intrusion detection system because:

1. It is an open source intrusion detection system. It is therefore useful where it is not cost efficient to apply NIDS sensors.

2. It is a lightweight application. It is economical when it comes to resource utilization.
3. Snort can be configured to more than just detect intrusion but also to prevent intrusion.
4. Snort is also based on open source flexible rules. Snort has over 2400 attack signatures in its database. Users can easily customize the rules to meet their varied network needs.
5. Snort is also available for both Linux and windows environments. It is also one of the most widely used IDS and has wide documentation.

The basic requirements for snort configuration can be classified as functional and non-functional requirements. The functional requirements are:

1. The set-up is integrated with some of the intrusion prevention systems to add to the effectiveness of the IDS. These could be firewalls, Virtual Private Networks (VPN) or even subnetted networks.
2. IDSs are shipped with up to 256MB of Random Access Memory (RAM), however these low RAM sometimes leads to dropping of packets by the IDS. Sensors are now shipped from manufacturing with + 512MB of RAM.
3. A layer two switch is used to segment the network. The switch makes use of MAC addresses in switching packets. However for an increase in granularity, a layer three switch could be preferred.

A layer three switch would also allow for the integration of Virtual Local Area Networks (VLANs). VLANs allow for an increase in the level of security by logically subdividing the networks that will allow different groups (Students, administrators and support staff) share the same infrastructure.

While the non-functional requirements include:

1. Providing physical security to network devices,
2. Define good practices on the use of networked resources. For example on the choice and use of passwords.
3. Training students and staff on network use.

3.4 The Network Setup

Intrusion Detection Systems can be deployed to protect different sections on the network. Hosts or even the server farms. Figure 3.4.1 below represents the possible setup of the network on which the IDS is configured.

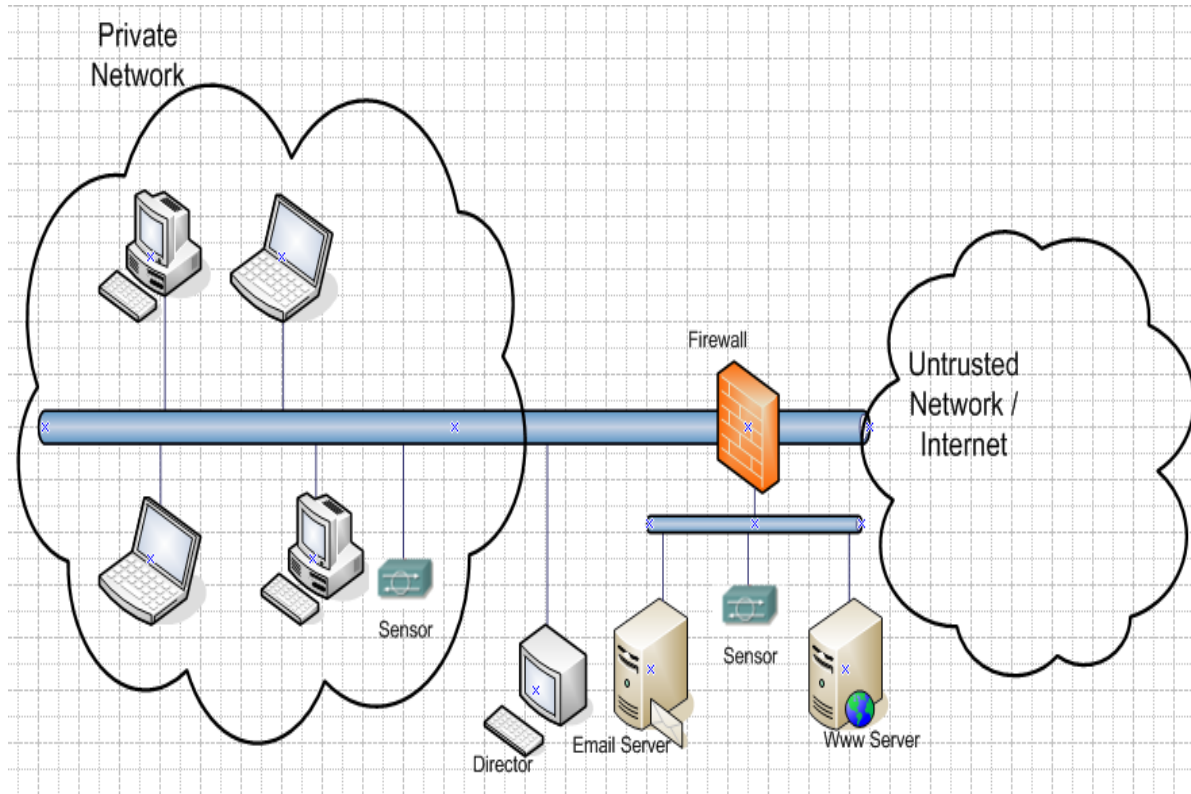


Figure 3.4.1: A model setup of a network with an (IDS)

3.5 Configuration and Validation of the IDS

Snort is configured on a Linux VM box running Ubuntu 20.04 operating system to detect unwanted traffic into a private network. Users for any specific services can then be monitored with alerts being logged or sent to the administrators.

Presence of alerts is evidence that the network will have been well configured and traffic monitoring is taking place.

3.5.1 Snort for Linux

On a Linux VM box running Ubuntu 20.04, configurations are made for the firewall, Basic Analysis and Security Engine(BASE) to provide a user friendly web front end to simplify querying and analysis of alerts, MySQL database that is an open source Relational Database Management System (RDBMS), Apache a widely available http server that supports PHP languages, Secure Shell(SSH) to enable secure remote login into the network, and PHP a hypertext preprocessor enables creation of dynamic content and interaction with databases.

Yum which is an update tool for rpm packages, is used for the installation because this can be configured to automatically keep the system up-to-date. In doing this, an open source for digital signatures and encrypted messages is downloaded to allow for authentication of downloaded rpms. Like snort for windows, Linux snort is downloaded from <http://www.snort.org>. It can be downloaded either in a compiled or precompiled state. The version used for this project was version 3.1.1.

After installing, we edit the `/etc/snort/snort.conf` file to customize this file to the needs of the protected network.

In the `snort.conf` file, we start by defining the variables HOME-NET to specify the network to be

monitored. The network configured to be monitored was network 192.168.0.0. EXTERNAL-NET described the source of traffic to be monitored and the work monitors all networks trying to access the home network.

The configurations specify the servers defined on the network. Among these servers are the Simple Mail Transfer Protocol (SMTP) for mail transfer and the Dynamic Host Configuration Protocol (DHCP) for dynamic assignment of IP addresses to client machines.

The following can also be set:

1. Preprocessors.

For stream reassembly and stateful analysis capabilities and the ability to track more than 256 simultaneous TCP streams, the stream4 module is enabled. The module consists of two modules stream4 and stream4 reassembly which both have to be used.

Port scan and port scan-ignore hosts are preprocessors that will help port scan detection and hosts for which port scan detection will be ignored respectively. For port scan the value assigned can be a network address for which traffic is to be monitored. This work monitors the network 192.168.0.0. Whereas port scan-ignore host, specified the DNS server, reason being that a DNS "talks" too much in the process of name resolution. The port scan-ignore host value used is 192.168.1.1 this being the address of the DNS.

Other preprocessors are the uni code used to normalize http and uni code attacks, rpc to normalize rpc decode on ports and telnet-decode for telnet negotiation.

2. Output Modules.

The alert-syslog module is configured to send alerts to the system log and database to log into a MySQL database. If snortsnarf is being used additional option LOG-PID shall be

required.

3. Snort rules.

This provides the basis of snort. Versions 1.8 and onwards will classify the rules into types and also provide for prioritization of traffic which configuration is done in the `/etc/snort/classification.config` file.

In the `snort.conf` file, we also specify the path that to the snort rules that can again be customized to reflect the needs of the protected networks.

4. `/etc/rc.d/init.d/snortd`.

This file allows for specifying the interface that will be used. And the interface we use in this project is `enp0s3`.

5. `/etc/snort/snort-check`

The file is used to send popups via the `smbclient` or sending mails to specific persons. And under this section we specify the hosts that will receive snort messages using the files `/etc/snort/hosts`, `/etc/snort/recipients` specifies mails of persons to receive snort alarms.

6. Internal Statistics.

The snort statistics can then be printed out using the file `/bin/kill-SIGUSR1 [pid of snort]` or `/bin/killall -U SR1 snort` for more than one process running on the same machine.

7. MySQL.

Snort has to send alerts to the MySQL database. This has therefore to be configured. Most Linux distributions have this already available. If they are not available, they can be compiled

and downloaded from <http://www.mysql.org/>.

3.5.2 Snort for Windows (Optional)

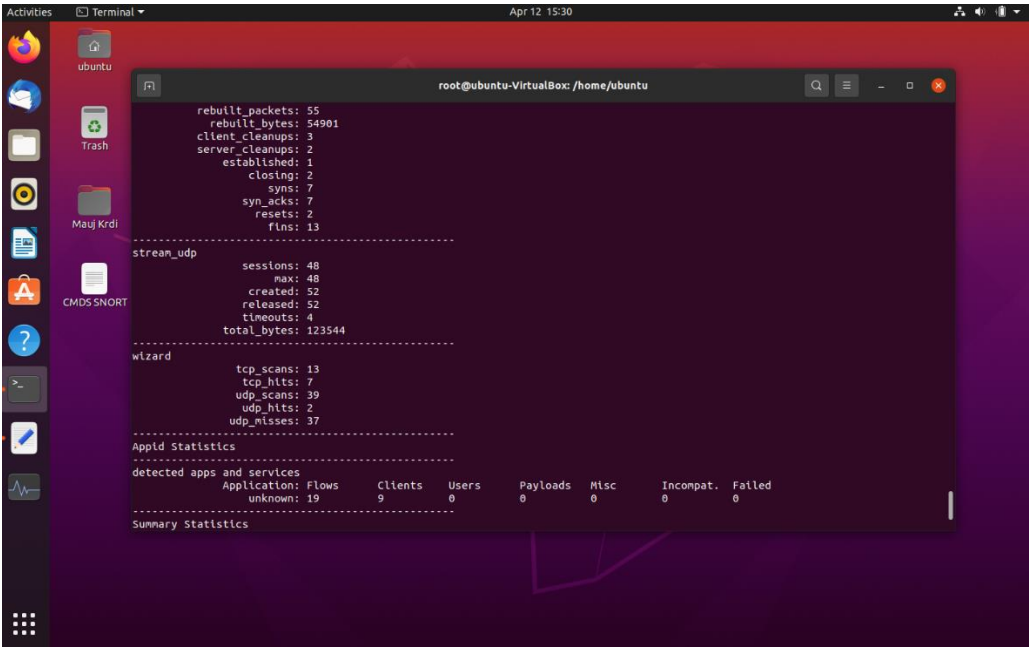
For windows intrusion detection, there is need for a packet capture driver for a windows environment and an example of this is WinPcap. Snort is downloaded in zipped format from <http://www.snort.org>.

It is unzipped and run on a local disk to carry out the installation. The simple steps that follow are the customization of snort for the network. It includes creation of snort configuration files also known as the snort.conf files and verification of the Home-Net and DNS-Server variables.

On command line configuration, the command below is used to allow for provision of full alerts, use configuration file and then to run as a daemon.

```
snort-A full-c snort.conf-D
```

However for a user friendly interface and easier configuration procedure, the IDS-Snort becomes helpful providing a graphical interface for controlling, managing, and auditing SNORT IDS for WIN32. This provision can be downloaded in zip format from <http://www.whitehats.com> or <http://www.snort.org>.



The screenshot shows a terminal window titled 'root@ubuntu-VirtualBox: /home/ubuntu' with the following output:

```
rebuild_packets: 55
rebuild_bytes: 54901
client_cleanups: 3
server_cleanups: 2
established: 1
closing: 2
syns: 7
syn_acks: 7
resets: 2
fins: 13
-----
stream_udp
sessions: 48
max: 48
created: 52
released: 52
timeouts: 4
total_bytes: 123544
-----
wizard
tcp_scans: 13
tcp_hits: 7
udp_scans: 39
udp_hits: 2
udp_misses: 37
-----
Appld Statistics
-----
detected apps and services
Application: Flows    Clients    Users    Payloads    Misc    Incompat.    Failed
unknown: 19          9         0         0           0       0            0
-----
Summary Statistics
```

Figure 3.5.1: Snort Script - snort-A full-c snort.conf-D

After configuration, the following program in Figure 3.5.2 appears: From the program shown in figure 3.5.2 below, a version is chosen and version 1.7 is chosen because of its enhanced features. We then specify the location of the rules file. We then select the log files/Alerts for the following configuration.

CONFIGURING SNORT FOR LINUX ENVIRONMENT (1)

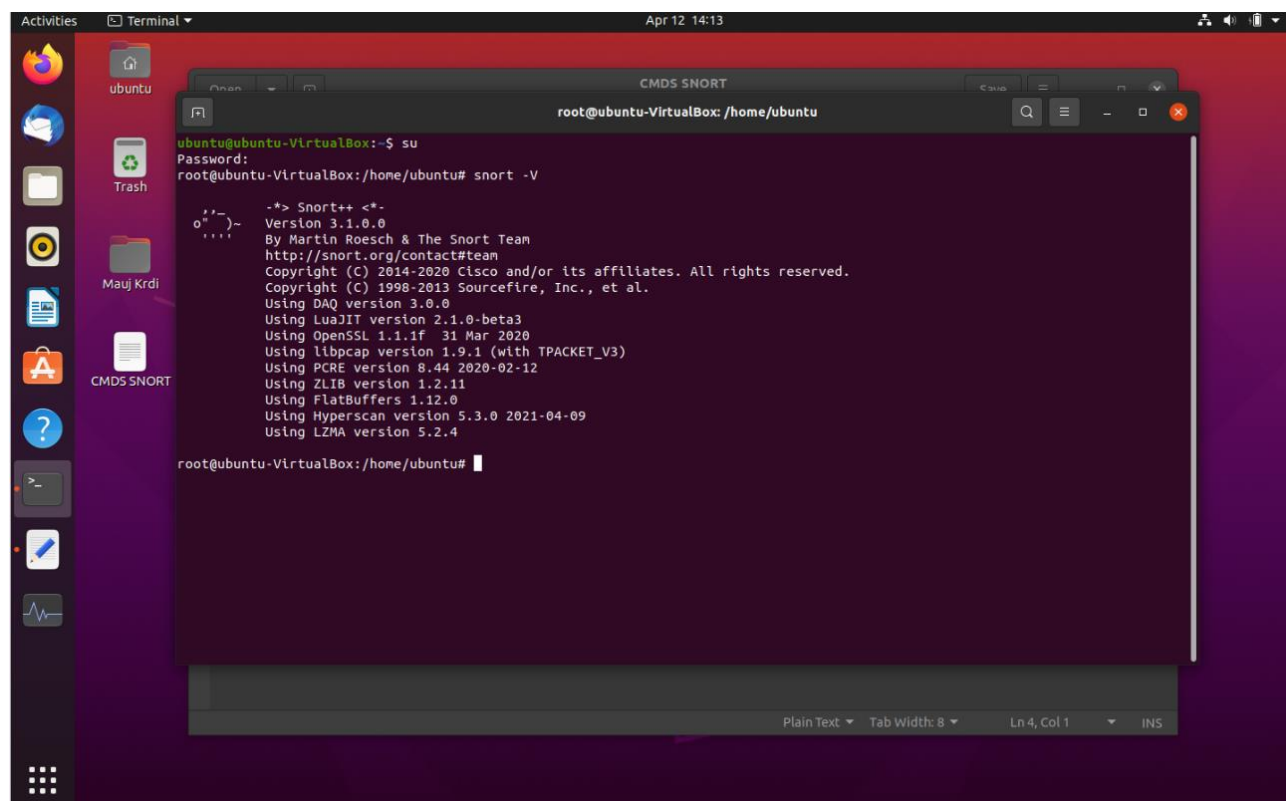
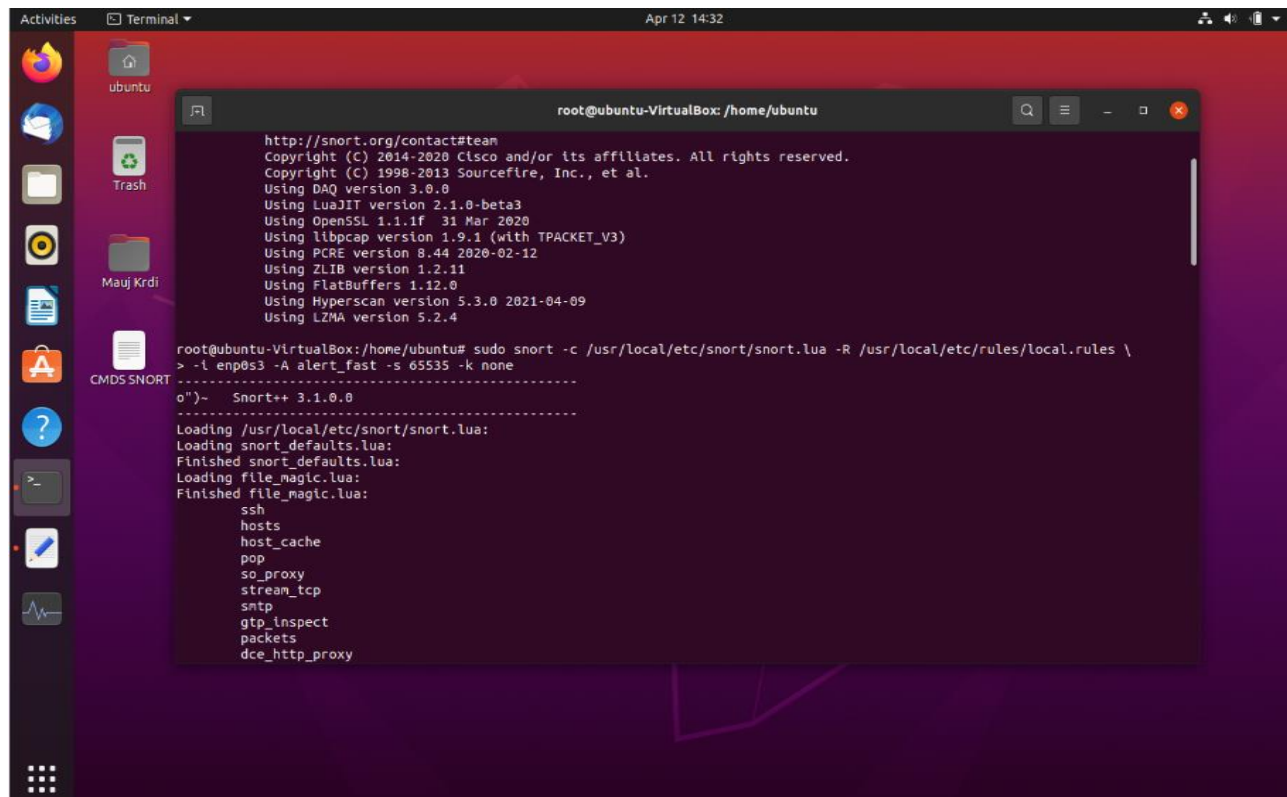
A screenshot of a Linux desktop environment with a terminal window open. The terminal window title is 'CMD5 SNORT' and the prompt is 'root@ubuntu-VirtualBox: /home/ubuntu'. The user has entered 'su' to become root, then 'snort -V' to display the version. The output shows Snort++ version 3.1.0.0, compiled by Martin Roesch & The Snort Team, with various dependencies listed: DAQ 3.0.0, LuaJIT 2.1.0-beta3, OpenSSL 1.1.1f, libpcap 1.9.1, PCRE 8.44, ZLIB 1.2.11, FlatBuffers 1.12.0, Hyperscan 5.3.0, and LZMA 5.2.4. The desktop background is purple, and the left sidebar shows icons for Activities, Terminal, ubuntu, Trash, and Mauj Krdi. The terminal window has a dark theme and a status bar at the bottom showing 'Plain Text', 'Tab Width: 8', 'Ln 4, Col 1', and 'INS'.

Figure 3.5.2: IDS-Snort Program

We click the apply tab as shown in figure 3.5.3 below to create the snort script and start or stop the snort intrusion detection.

CONFIGURING SNORT FOR LINUX ENVIRONMENT (2)



```
http://snort.org/contact#team
Copyright (C) 2014-2020 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using DAQ version 3.0.0
Using LuaJIT version 2.1.0-beta3
Using OpenSSL 1.1.1f 31 Mar 2020
Using libpcap version 1.9.1 (with TPACKET_V3)
Using PCRE version 8.44 2020-02-12
Using ZLIB version 1.2.11
Using FlatBuffers 1.12.0
Using Hyperscan version 5.3.0 2021-04-09
Using LZMA version 5.2.4

root@ubuntu-VirtualBox:/home/ubuntu# sudo snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/rules/local.rules \
> -i enp0s3 -A alert_fast -s 65535 -k none
-----
o")~  Snort++ 3.1.0.0
-----
Loading /usr/local/etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:
Loading file_magic.lua:
Finished file_magic.lua:
ssh
hosts
host_cache
pop
so_proxy
stream_tcp
snmp
gtp_inspect
packets
dce_http_proxy
```

Figure 3.5.3: Snort Packet Capturing Script

3.6 Service Configurations

3.6.1 Firewall Configuration

In the Linux environment a firewall has been configured at the perimeter to restrict entry to and exit from the private network. Snort provides a feature to support firewall configuration.

3.6.2 DHCP Configuration

For dynamic assignment of IP addresses, a Dynamic Host Control Protocol (DHCP) is identified on the network 192.168.0.0/24. The DHCP is always configured with a pool of addresses to assign to client machines on the network. The IP address of the DNS server being reserved and not assigned to other connecting hosts as a good practice.

3.6.3 Configuring the Mail Server

The mail server is configured to provide one of the services that will require monitoring. It provides mail services that can be monitored in settings such as during online examinations. The IDS detects this by monitoring the protocol specifically used for this service - The IMAP. The IDS in this case is configured to monitor a section of the network or subnetwork.

3.6.4 HIDS and NIDS Configuration

Since snort is being deployed as the intrusion detection system, When the HOME-NET is defined as a single host address, it becomes Host Intrusion Detection System (HIDS) and a Network Intrusion Detection System (NIDS) when the HOME-NET is a network address. Their configuration therefore follows the same steps as in section 3.5.1.

3.6.5 Intrusion and attack Simulation

Most of the network attacks or unauthorized resource access is dependent on the flow of packets to and out of the network. This being the case, control of unauthorized access of networked resources will be highly dependent on control of this flow of packets. Requests that are sent to an internal host from unauthorized users will therefore constitute an attack.

A model network is setup and an intrusion simulated. The flow of packets were analyzed with and

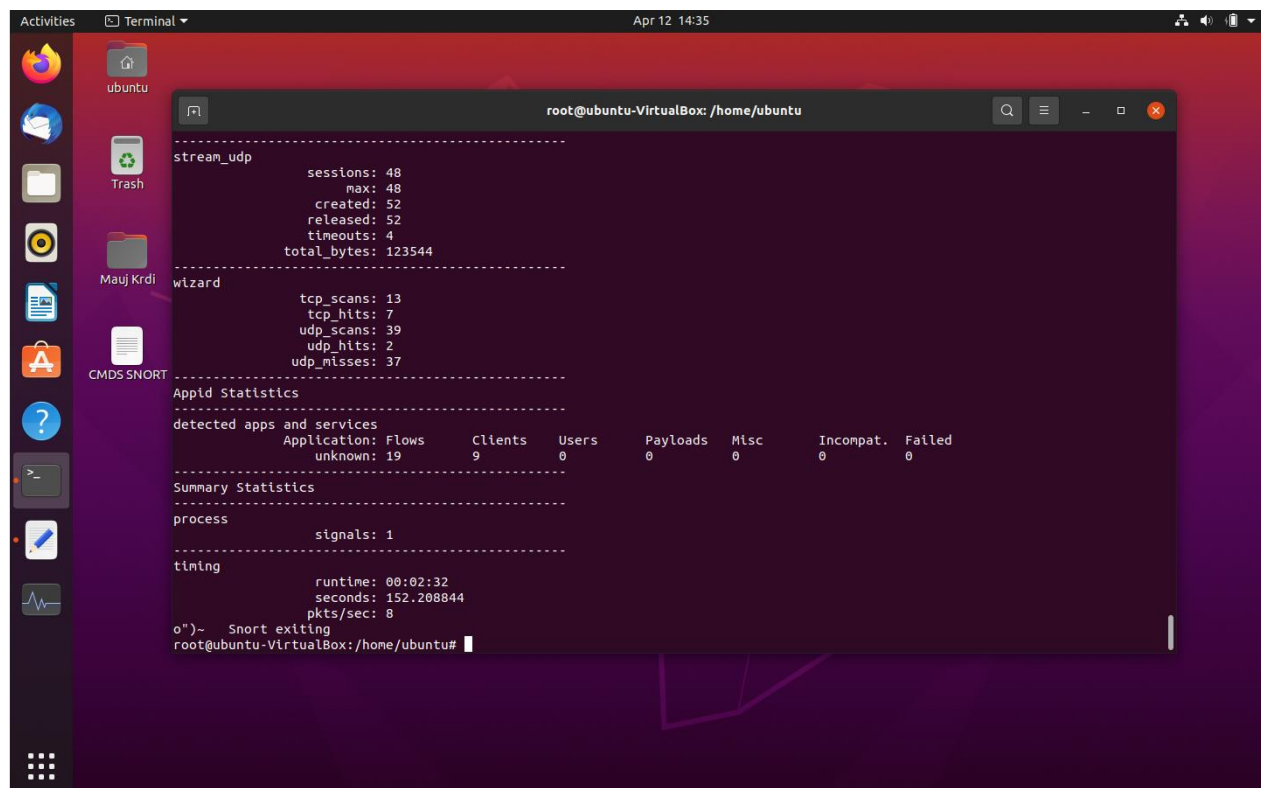
without the open source IDS configured on the network (Fig3).

3.6.6 Validation of IDS

The approach that is used to verify the IDS is one that lets the IDS be aware of the environment in which it is operating and the configuration of the server under attack.

To validate the intrusion detection system implementation, the IDS observes the reaction of the server to a given request. And this verification method has also been used because:

1. The approach is passive and therefore will eliminate mapping of the system being monitored and host based verification.
2. Analysis of the header field of a response from the server is just enough to determine attack results.

A screenshot of a terminal window on an Ubuntu system. The terminal displays the output of the 'snort -V' command, showing various statistics and configuration details. The output is organized into sections: 'stream_udp', 'wizard', 'Appld Statistics', 'detected apps and services', 'Summary Statistics', 'process', and 'timing'. The 'stream_udp' section shows session counts and byte totals. The 'wizard' section shows scan and hit statistics. The 'Appld Statistics' section shows detected applications and services. The 'detected apps and services' section shows a table of detected applications. The 'Summary Statistics' section shows process and signal counts. The 'process' section shows the number of signals. The 'timing' section shows runtime and packet processing statistics. The terminal prompt is 'root@ubuntu-VirtualBox: /home/ubuntu#'.

```
root@ubuntu-VirtualBox: /home/ubuntu# snort -V
stream_udp
  sessions: 48
    max: 48
  created: 52
  released: 52
  timeouts: 4
  total_bytes: 123544

wizard
  tcp_scans: 13
  tcp_hits: 7
  udp_scans: 39
  udp_hits: 2
  udp_misses: 37

Appld Statistics
-----
detected apps and services
Application: Flows    Clients  Users  Payloads  Misc  Incompat.  Failed
unknown: 19         9       0        0       0        0         0

Summary Statistics
-----
process
  signals: 1

timing
  runtime: 00:02:32
  seconds: 152.208844
  pkts/sec: 8

o")~ Snort exiting
root@ubuntu-VirtualBox: /home/ubuntu#
```

Figure 3.5.4: Successful Validation of IDS System using Software Based Application Snort

Figure 3.6.4 below gives a program that will simplify the verification.

The detailed and informative logs are viewed on the IDS center interface in text format. However for simplicity, windows environments support applications for graphical user interface. An example is the Packet Statistics that presents the snort logs in an easy to read textual form. This application has added advantage because it also provides information about the attack such as source address and port accessed.

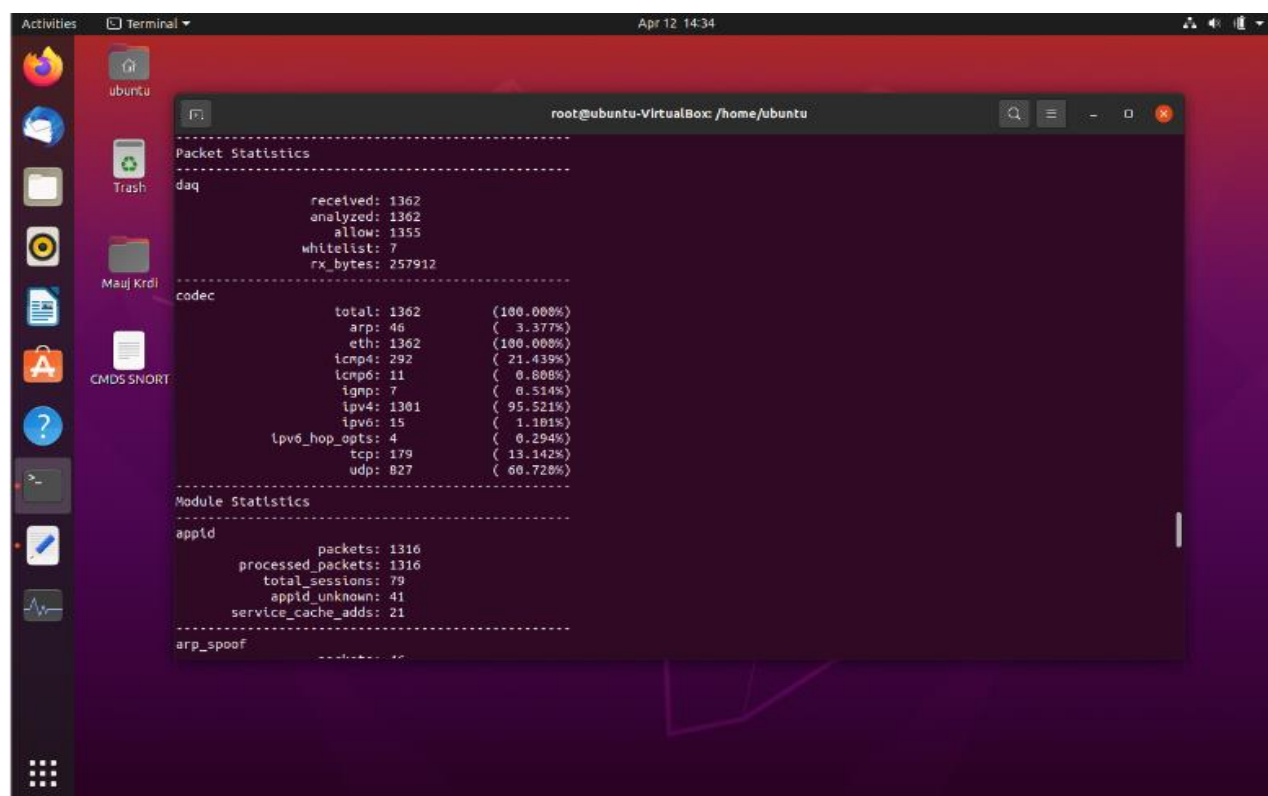


Figure 3.6.4: Verification of Network traffic using Packet Statistics

The application is downloaded into a temporary file from where the setup is run. In Fig 3.6.4 the Packet Statistics is used to automatically generate an output from the current alert files while the arp_spoof will update the files at a specified time interval.

Chapter 4

Presentation and Discussion of Results

4.1 Current Technology

Seven institutions visited and non was found to have deployed any intrusion detection systems. Of these most had computers ranging from 5 to 20, the computers where stand alone (Not Networked) and the few that where networked solely put all their trust on the service providers.

The non-networked computers with standalone hosts as illustrated in the Figure 4.1.1 below, services such as printing, fax and scanning only connected to only one or a few hosts. Users have to move to and from this host to access these services.

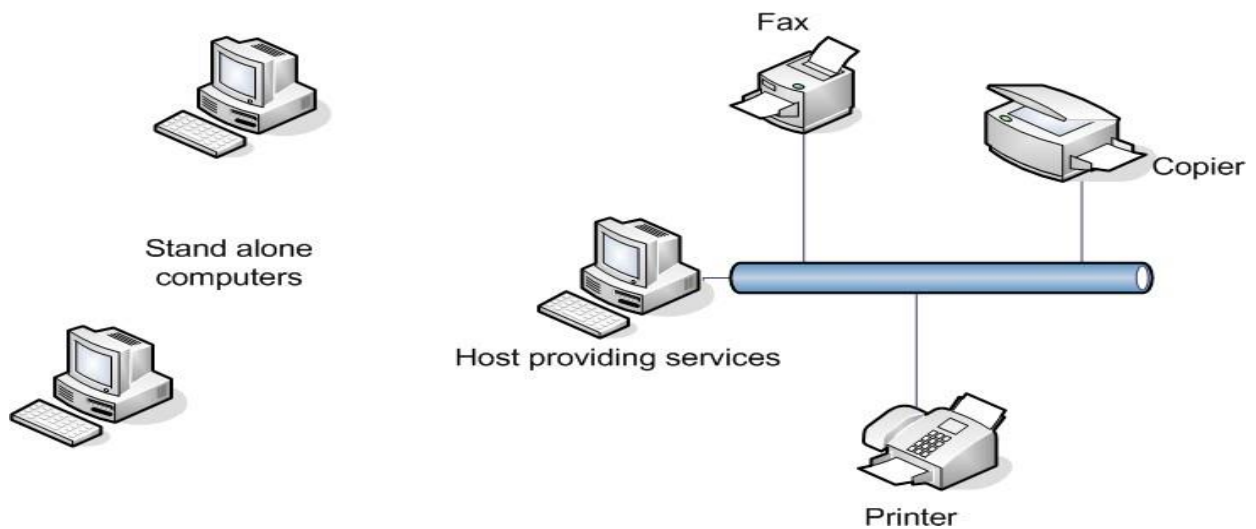


Figure 4.1.1: Stand alone architecture

Though this architecture limits remote access, its administration is not centralized and therefore burdensome especially when there are large numbers of devices.

However a few institutions had up to hundreds of networked devices. These institutions had Internet connections with limited security to the network. They mostly put a their trust on service providers. Nothing was in place to monitor traffic into the intranet. While some connected to the Internet through perimeter firewalls.

Figure 4.1.2 below provides a general setup that has been deployed by these institutions. Hosts on the intranet such as A, B and C may access the internal servers such as the mail server found within the internal network. While traffic from the external network will have to pass the rules defined on the firewall to have access into the intranet.

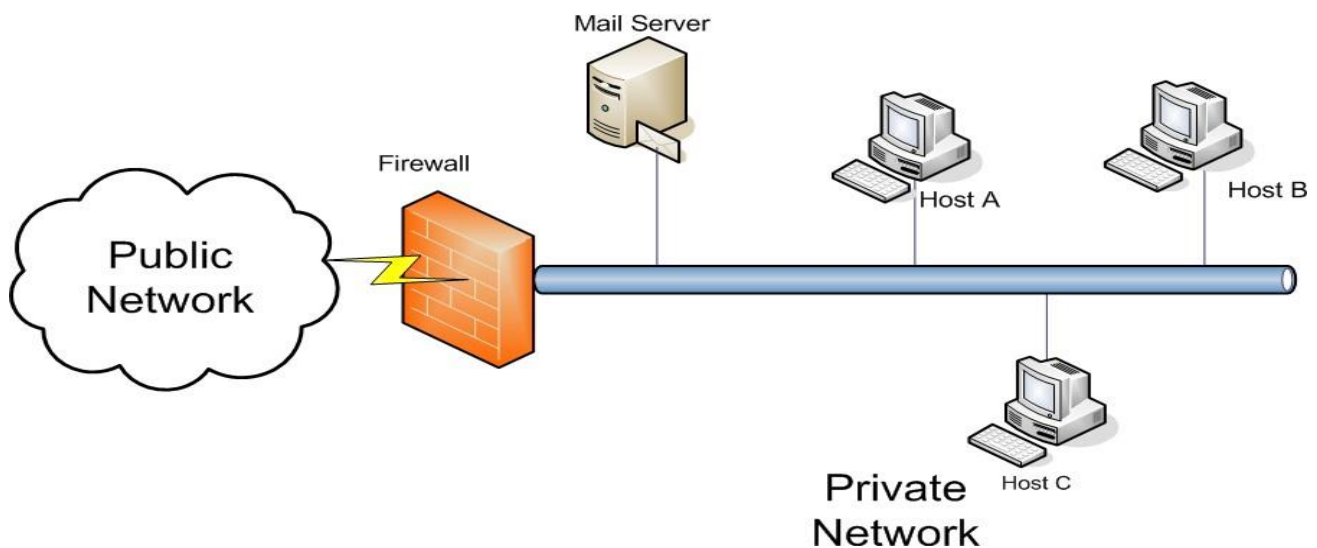
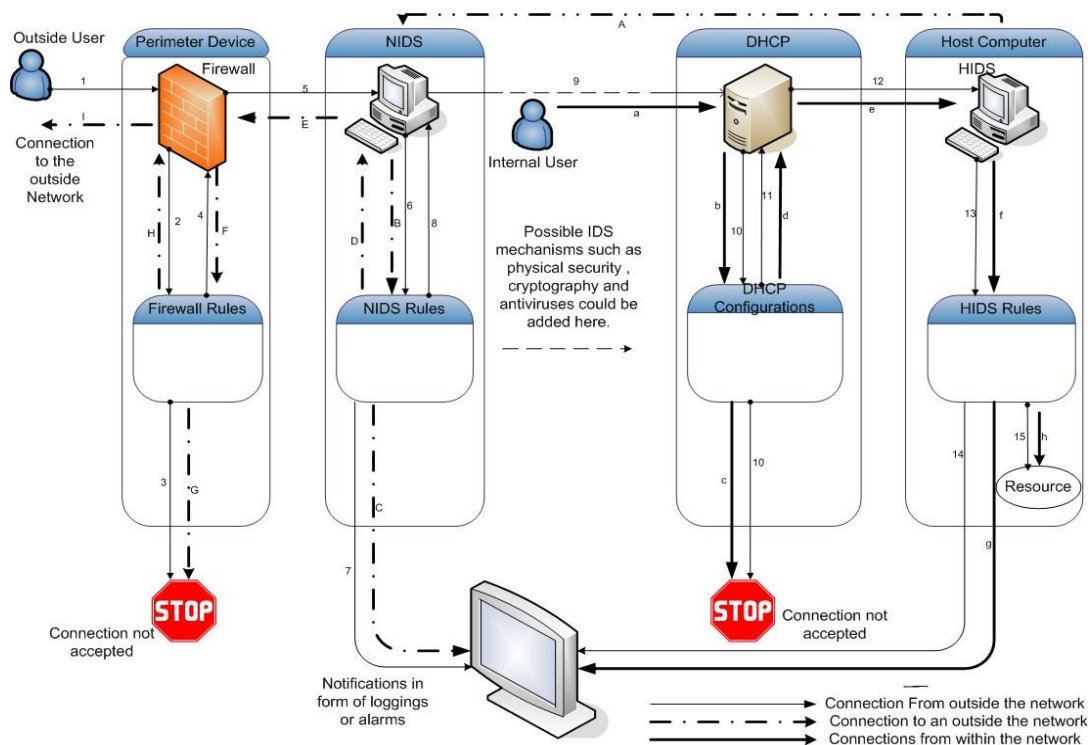


Figure 4.1.2: Networked computers connecting to the Internet through a firewall

4.2 Conceptual framework for an effective intrusion detection system

This section demonstrates an implementation of an integrated intrusion detection system that will support scalable networks in protecting against intrusion. It takes into consideration intrusion from within the network, those from the outside of the network and access of restricted public sites.



The design takes into account the knowledge of how dynamic the Information Technology (I.T) industry has been. It puts scalability and appropriate security in mind.

The framework is made up of a Network Intrusion Detection System (NIDS) for detection of traffic to and from a given network or subnetwork, a Host based Intrusion Detection System (HIDS) and a line for possible Intrusion Prevention Systems (IPS). The NIDS detects against network level

Intrusion, and since traffic such as encrypted data may not be detected by the NIDS, host level intrusion detection systems to further enhance intrusion detection. The framework provides a flexible approach that will enable the network designer put in place prevention mechanism that will enhance intrusion detection and meet the required level of security while at the same time trying to strike a balance between security and resource utilization. Each of these have rules to refer to in determining which traffic is unwanted.

We have the intrusion detection systems behind a firewall. A firewall is a software or hardware or a combination of the two use to control traffic into or outside a private network. This architecture helps detect any of the traffic that may have bypassed the firewall. It also decreases on the workload for the IDS and hence IDS efficiency.

It also has a Dynamic Host Configuration Protocol (DHCP) machine. A DHCP is a protocol used to dynamically assign subnet masks, IP address of a network gateway that could be on the network and the Dynamic Name Server (DNS) to a disk less computer or a machine that starts up. The DHCP if not well protected by a good line defense or intrusion prevention techniques can be compromised by the attacker through denial of service attacks. The DHCP also by itself may not force clients to release their leases at shutdown. Malicious hosts for example may therefore continue allocating new addresses without releasing them at all, leading to exhaustion of addresses. Secondly, when external clients are intentionally or accidentally configured to use addresses of internal clients, regulation of addresses will become hard and therefore leads to a security disaster. Hence the need for more security enhancing features.

For high security inside the network, port security is configured. Access to switch ports could be given only to specific Mac addresses. However the project uses IP addresses.

Either intrusion or attempts to access a restricted resource are logged to a central location from where the network administrator is notified and he takes relevant actions to protect the network. The packet flow is as follows: A packet from an outside network passes through a firewall. The purpose of the firewall is to match the traffic against a set of pre-defined rules such as permit or deny for the packet to be allowed or denied access respectively into the protected network. When the packet does not pass the defined rules connection is configured to terminate. When the traffic passes the rules, it is allowed into the network. However, there is a likelihood of the firewall being compromised. The traffic will need to pass the test of the NIDS. Packets from the firewall head to the NIDS (5,6). The packet is once more matched against the NIDS rules, when it fails, an alarm or with additional software an email can be sent to the administrator. Packets that pass this test go through a line of intrusion prevention mechanisms (8, 9) to a DHCP that assigns IP addresses to eligible client machines on the network. If the DHCP is not configured to accept the connecting machine, the connection will not proceed, it will be stopped or dropped (10) else the connection is allowed to the destination host (11, 12). The host has an HIDS. This also matches the packet with its rules (13). A failure will result into a signal being sent to the administrator and hence a failure to connect to the resource (14). The resource could be a service such as mail services from the machine. The packet will only accept a connection to the resource when the packet passes all the tests (15).

However because not all traffic originates from within the network, the DHCP and the HIDS could be used to provide a relatively secure internal network as indicated by the steps a, b, c, d, e, f, g and h. This represents the steps as described above but without the firewall that functions at the perimeter and the Network Intrusion Detection System (IDS).

While connections to the outside network could similarly be restricted and detected through A, B, C, D, E, F, G, H and I. A is an internal host making a request for a service that is outside the perimeter. It thus has to cross the perimeter through the Network Intrusion Detection System (NIDS) that by consulting its rules (B) sends alerts (C). If the request passes this test, it is allowed to the perimeter firewall (D,E) that will match this request with its predefined rules (F) to either drop (G)the request or allow it (H,I).

The traffic being monitored in this work is IMAP protocol that is used by client machines to access mail services.

The network is configured not to allow direct communication between hosts without the involvement of a server. This is very important because it reduces the chances of viruses spreading from one machine to another in a client server environment where the spread of viruses is controlled at the server.

4.2.1 Implementation results

The results obtained here reflect some of the features that this implementation has over ordinary mechanisms put in place by most institutions.

1. PHP Configuration

The PHP Hypertext Preprocessor is a programming language that will allow creation of dynamic content that interacts with databases.

Accessing network query tool can then be done using the address <https://192.168.1.17> as shown in Figure 4.2.4 next. Where 192.168.1.17 is the address of the machine hosting the intrusion detection system.

With PHP configuration enabled, the network query tool simplifies troubleshooting and

hence management of the networked client machines. The IP address or host under consideration is specified in the "Enter host or IP" section and the administrator will either get DNS records, resolve or even ping the specified host.

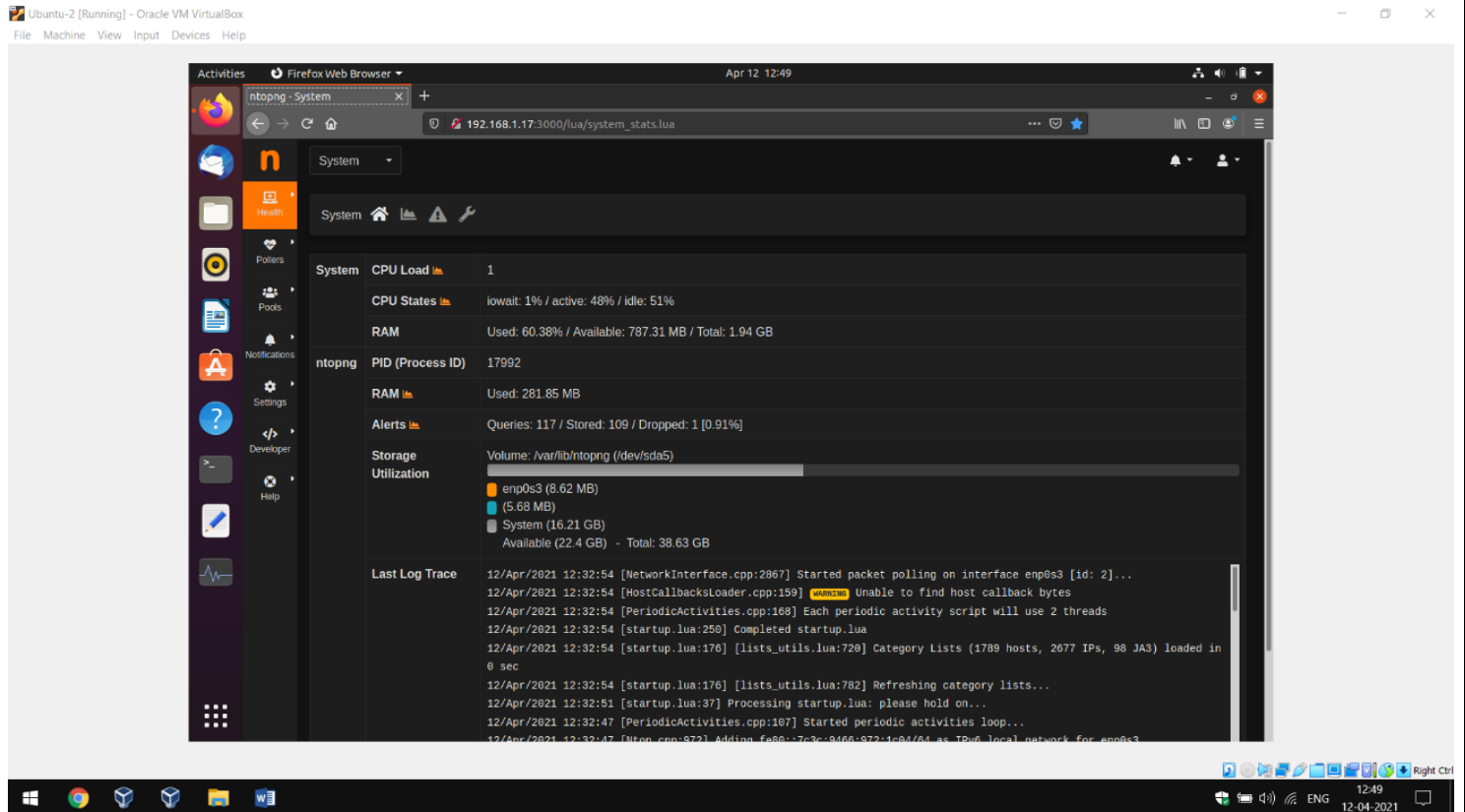


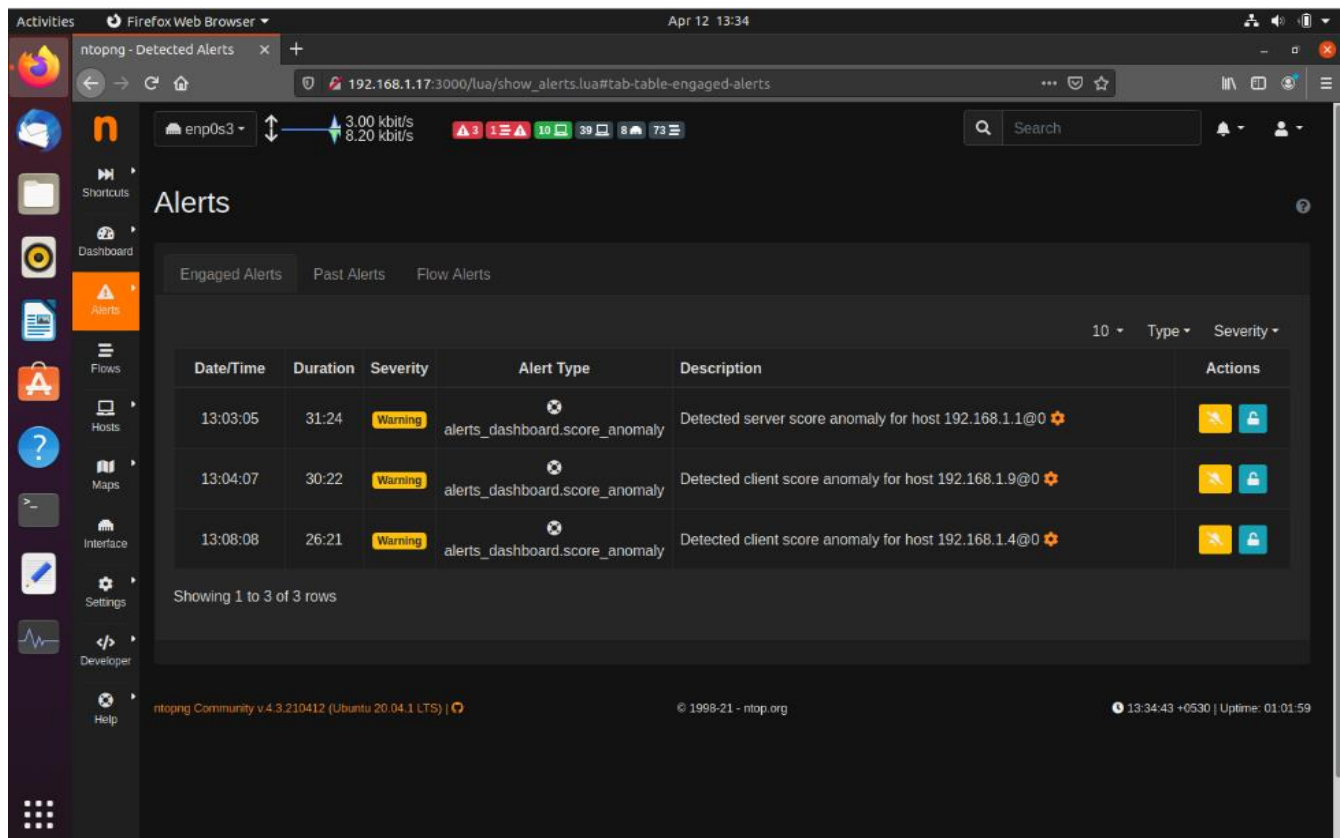
Figure 4.2.4: Network query tool

2. Basic Analysis Security Engine(BASE) configuration

By using the address <https://192.168.1.17:3000> or <https://localhost:3000> from the url, The administrator can immediately get notifications of alerts received. Eight alerts are seen to have been sent as reflected on Figure 4.2.5 next.

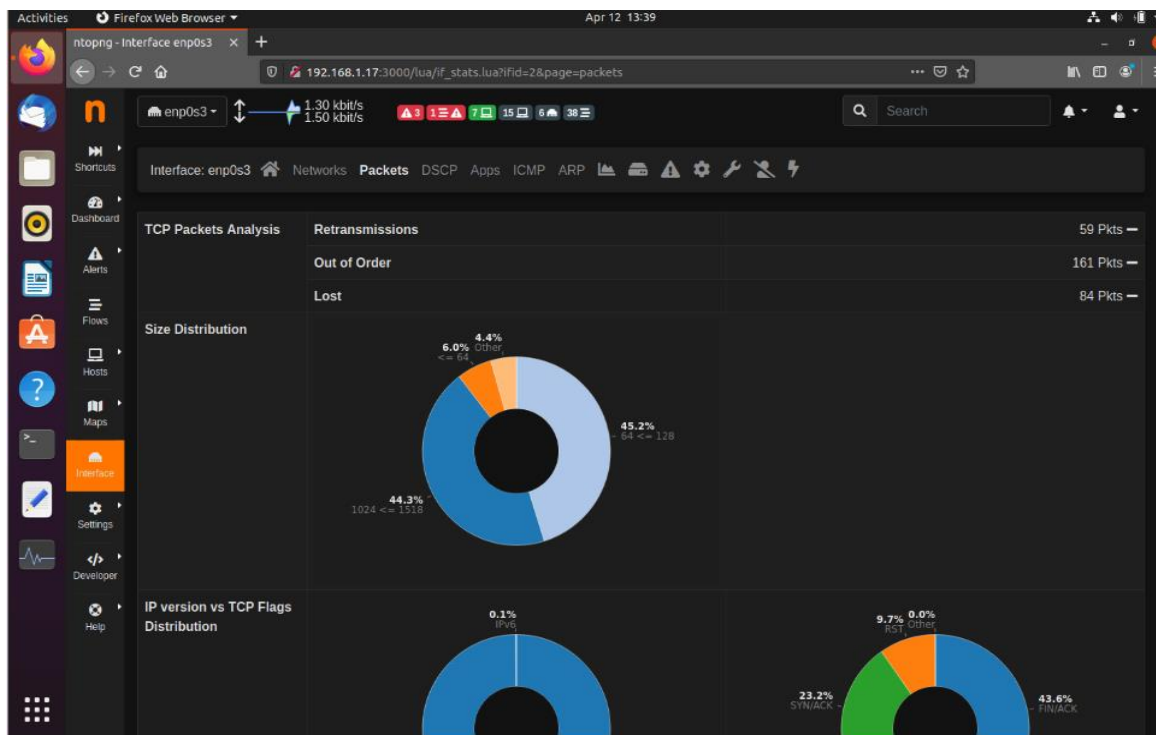
The network administrator can specify the duration over which the network needs to be monitored and preferred graphical outputs such as bar chart, pie chart or line graphs obtained. This is done by entering values for chart begin and chart end. The figure specifies monitoring traffic beginning 13:03 hours on the 12th of April the year 2021 to 13:08, 12th the same month and year.

Figure 4.2.5: Base alert monitor



Base provides options to generate any of the graphs, pie charts, bar graph or line graph of the captured results. And a bar graph such as the one below was generated.

Figure 4.2.6: Base graph for TCP port source against alerts



3. NTOPNG

NTOPNG can enable monitoring of other network traffic and protocols. And from the output below the protocols results traffic show a summary for each of the defined hosts as indicated on the first column. Bandwidth utilization could also be monitored as a basis for detecting intrusion.

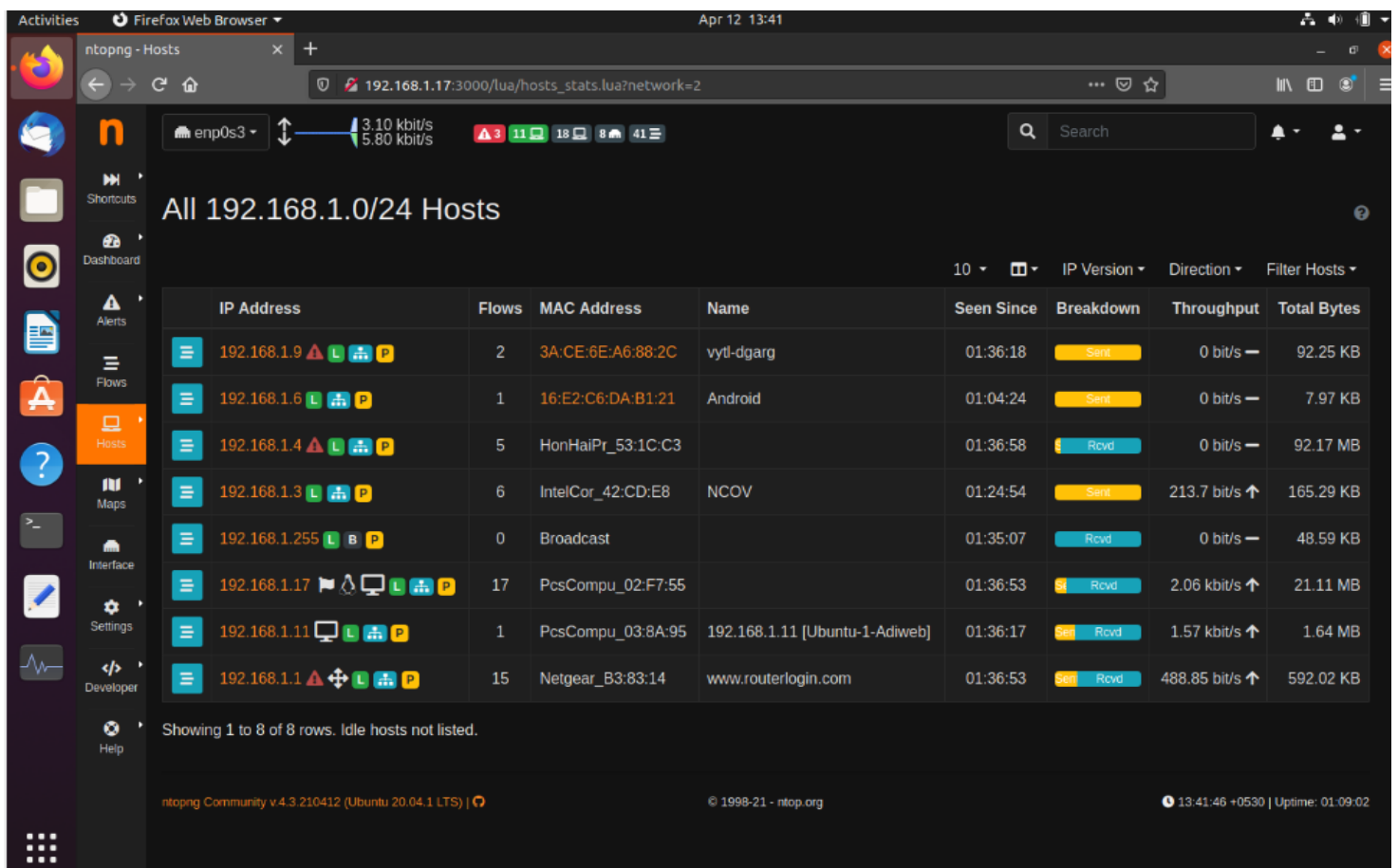
Figure 4.2.7 below shows that snort not only aids in detecting intrusions but it also simplifies network management. The result shows that the system could monitor traffic that are

either local, remote only, sent, received or all traffic. And for this case only local traffic is monitored.

The first column shows the list of hosts being monitored and as an example it will even specifies that the second machine is a gateway router by the added symbol.

Apart from these, it also indicates how dangerous the traffic would be to the network by adding a flag that could be green or red. Red showing that the traffic requires urgent attention.

Figure 4.2.7: Base graph for TCP port source against alerts



For a visual protocol analysis, results can also be put into graphical representation. Figure 4.2.8 below indicates the bandwidth consumption for the protocols UDP that was utilizing up to 6.7MB to IGMP and the other IP protocols that consume only 568.5KB of bandwidth.

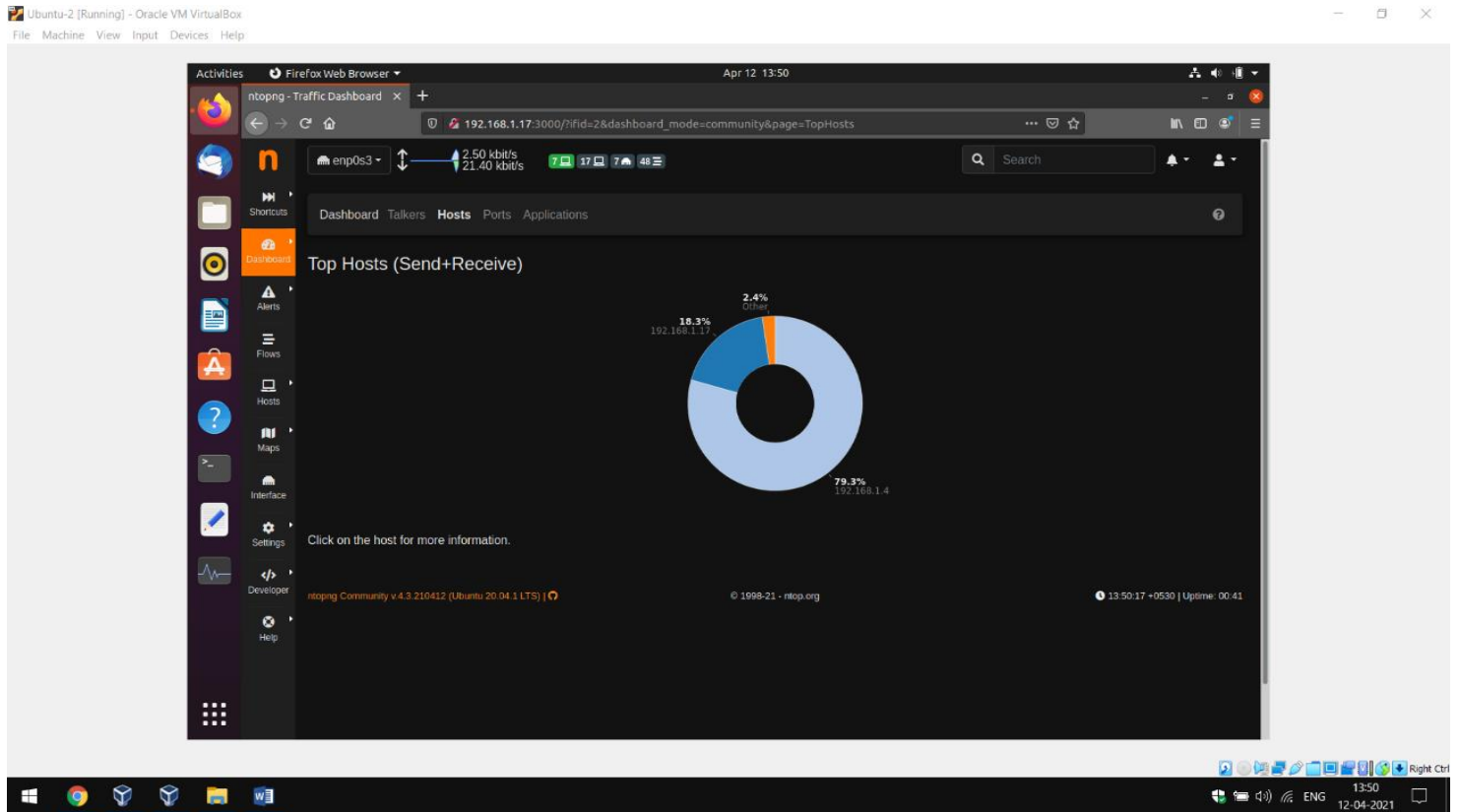


Figure 4.2.8: NTOP graph for traffic monitoring

Network monitoring and hence management is further enhanced by a graphical representation of network load statistics is illustrated next. A deviation from normal trend in the network load or normal network load at awkward times could be an indicator of intrusion. The graph can be generated for different time intervals. The graphs indicate graphs for the last ten minutes and one hour respectively.

This feature helps provide statistical/historical on network traffic in a given period of time.

Figure 4.2.9 below gives a graphical presentation of the behavior of network traffic within the previous ten minutes and for the previous one hour.

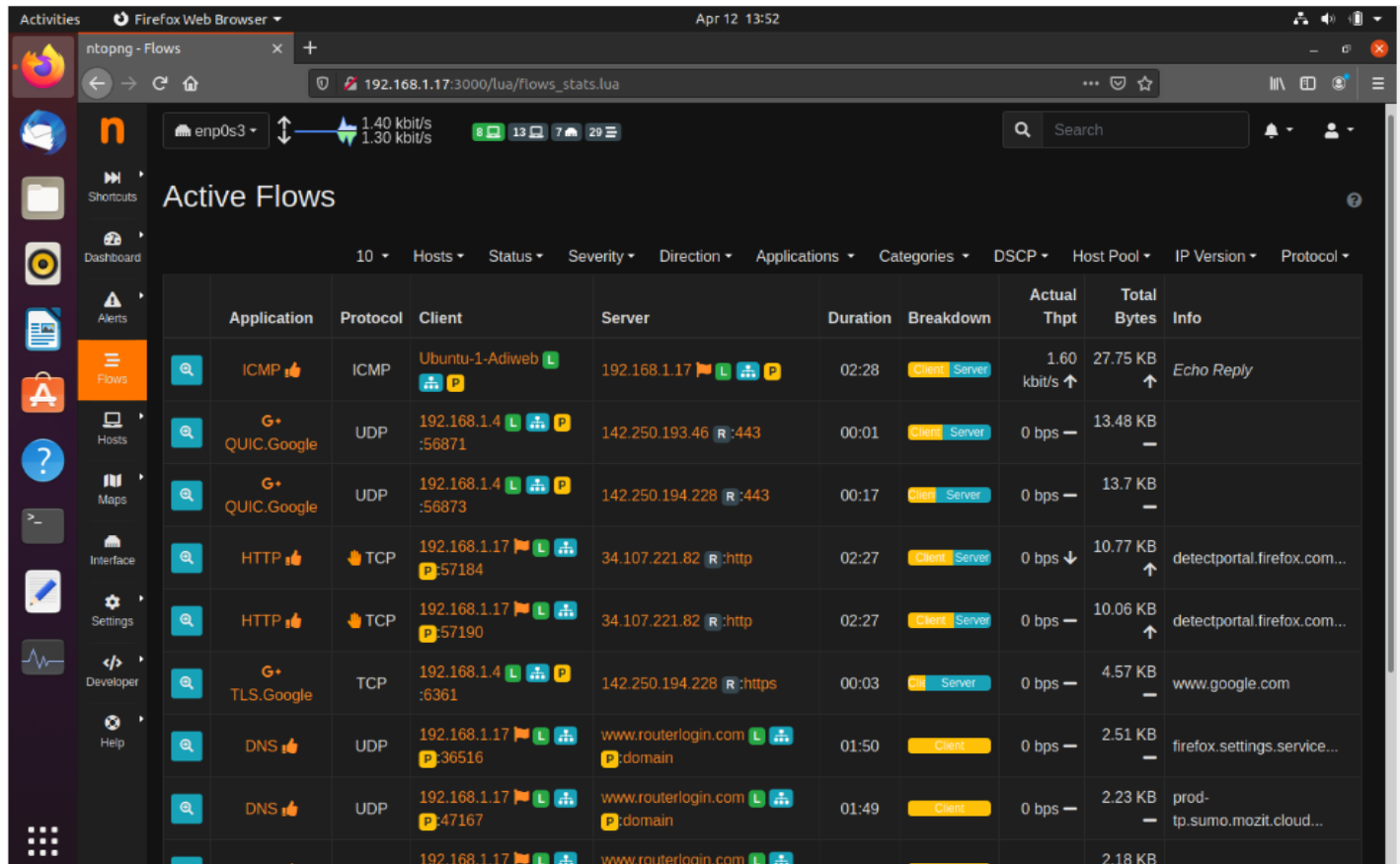


Figure 4.2.9: NTOP graphical impression of network load statistics

The incorporation of NTOP in snort supports other features as indicated in the figures next.

Figure 4.2.10 illustrated next gives further detailed information such as protocols that use IP and proportion of the bandwidth they consume. While Figure 4.2.11 gives other features provided by NTOP such as ability to protect specified urls.

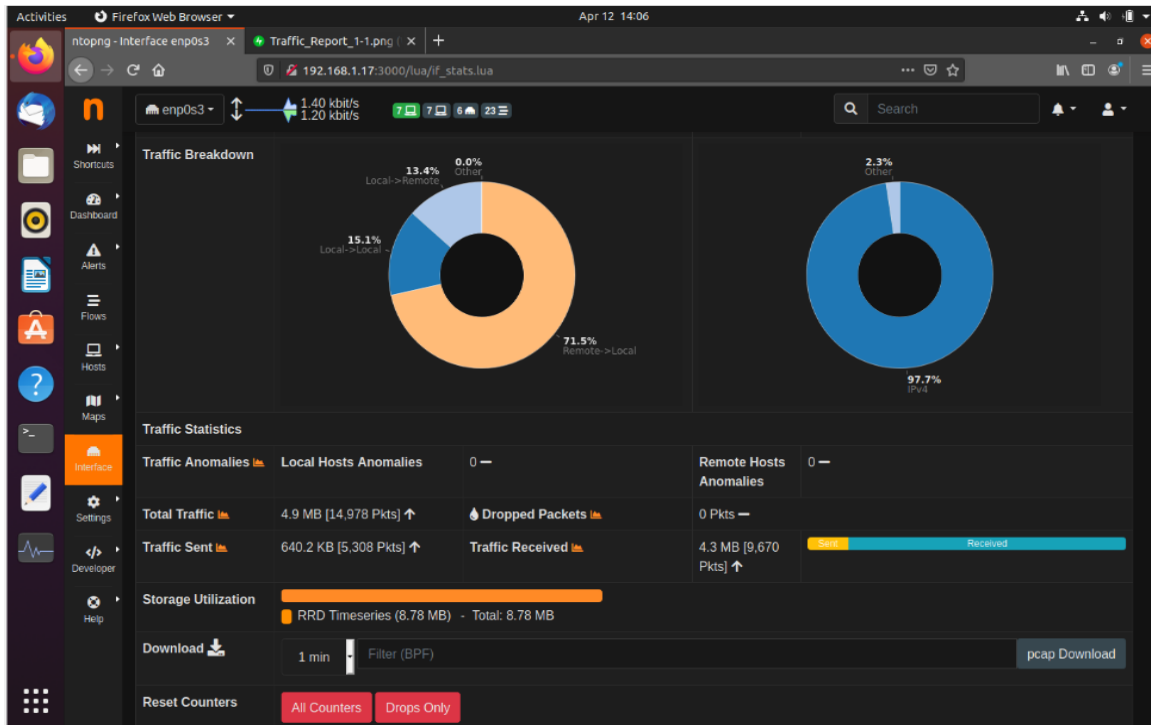


Figure 4.2.10: NTOP summarized traffic distribution

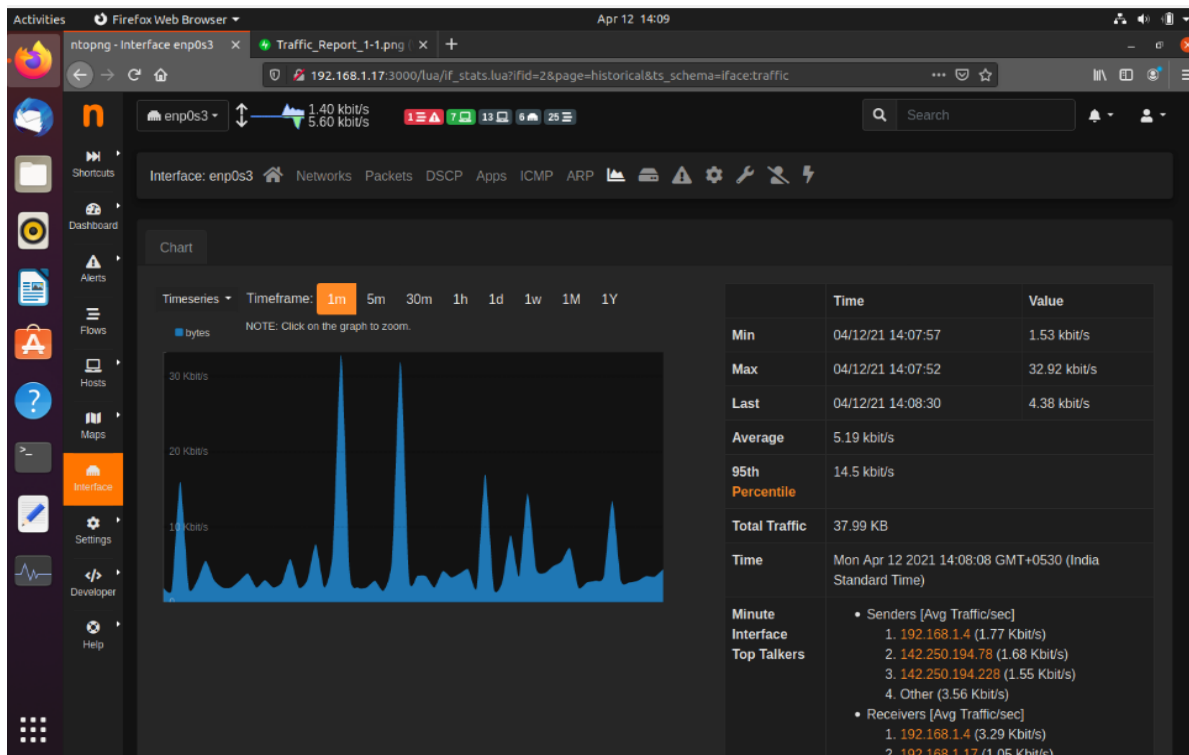


Figure 4.2.11: Other NTOP features

Chapter 5

Summary, Conclusion and Recommendation

5.1 Introduction

Academic institutions like other organizations continue to face threats from random acts of attacks from both script kiddies and professional hackers. By detecting and sometimes reacting to intrusions, Intrusion Detection Systems (IDS) provide a relevant level of security necessary to save institutions from heavy financial losses and reputation resulting from unauthorized system access.

5.2 Recommendation

This project has come up with recommendations that will enhance network security while providing for scalability and effectiveness, and ease of management of the system to the overall network security.

5.2.1 Implementation

The implementation design provides a framework for the modeling of effective intrusion detection systems. Integration of intrusion detection systems with a line of intrusion prevention mechanisms will greatly improve on the system performance.

The advantage with this is that it allows the system to take up the strengths of the different intrusion

prevention mechanisms in the overall network security.

Moreover it also ensures high system scalability since devices can easily be added or removed from the system without affecting its overall performance. Configurations not being tied onto a single device reduces the chances of having a single point of failure.

The system is also effective since it looks at both external attacks and internal attacks that make up one of the most dangerous threats to network security.

It is an adaptable system that can be customized to meet the different needs of different academic institutions. Academic institutions can be of different sizes as regards the number of nodes, different number of users, varying financial backgrounds and sometimes different forms of threats.

By modifying the snort.conf configuration file a specific type of threat can be monitored. As regards the size, for relatively smaller institutions, acquiring dedicated devices such as routers may be beyond their reach and thus an intrusion detection system can be customized and deployed on a single affordable server machine to monitor the whole system. The system can therefore be applicable quite a range of academic institutions.

5.2.2 Good Practices

The practices mentioned below will reduce on the work that could have otherwise been left for the intrusion detection system. When put in place they will therefore improve on the efficiency of the IDS.

1. To reduce the chances of the IDS being compromised, the machine hosting the IDS should be freed from running other services.
2. To reduce on the number of internal attacks, physical security to location of the network devices.

3. Define to users what constitutes bad practices of networked resources.
4. Setup a security policy.

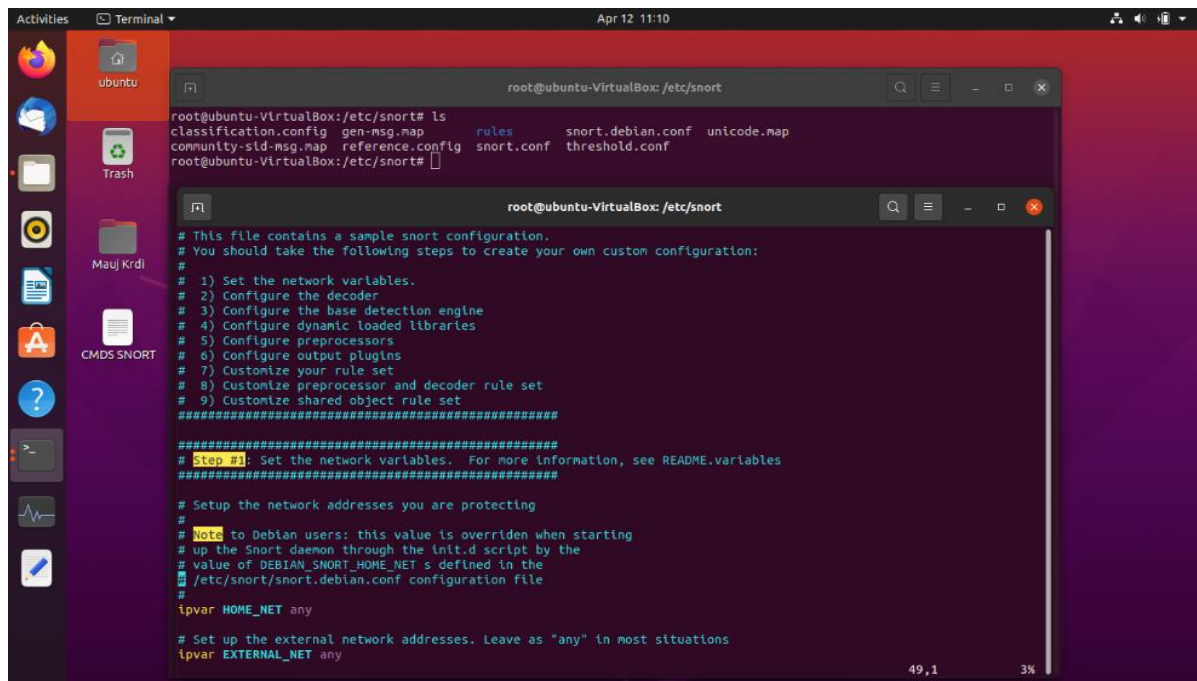
Though these will not directly provide for intrusion detection, they greatly reduce on the possible damages as a result of intrusion. Intrusion detection systems may also be compromised. (See section 2.2.1)

The results have shown that it is more practical to have at least an intrusion prevention systems for an effective intrusion detection, though intrusion detection for networks with critical data such as academic transcripts would require more than just a single line of defense.

The research however leaves several issues on effective intrusion detection for future work, and this include:

1. The need for a standard data exchange between different intrusion detection components
2. A way of combining the different approaches to intrusion detection (NIDS and HIDS) into one integrated system.
3. There is need for better automated responses.

APPENDIX A - SNORT CONFIGURATION FILE



The screenshot shows a terminal window titled "Terminal" with the path "root@ubuntu-VirtualBox: /etc/snort". The terminal displays the output of the command `ls`, listing the following files: `classification.config`, `gen-msg.map`, `rules`, `snort.debian.conf`, `unicode.map`, `community-sid-msg.map`, `reference.config`, `snort.conf`, and `threshold.conf`. Below this, the contents of the `snort.conf` file are shown. The file contains a sample configuration with comments and instructions for setting up Snort. Key sections include a list of steps to create a custom configuration, a section for setting network variables, and a section for setting up network addresses. The file also includes a note for Debian users and a section for setting up external network addresses.

```
root@ubuntu-VirtualBox:/etc/snort# ls
classification.config  gen-msg.map          rules                 snort.debian.conf    unicode.map
community-sid-msg.map reference.config      snort.conf           threshold.conf
root@ubuntu-VirtualBox:/etc/snort#
```

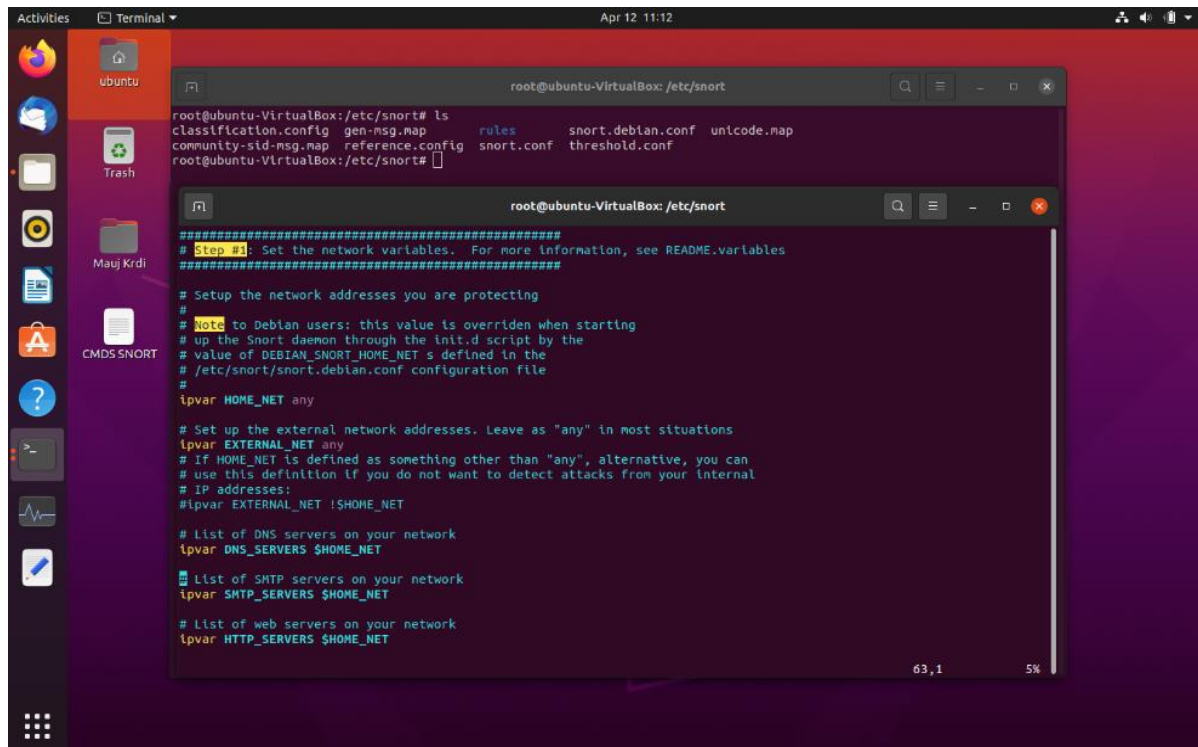
```
#####
# This file contains a sample snort configuration.
# You should take the following steps to create your own custom configuration:
#
# 1) Set the network variables.
# 2) Configure the decoder
# 3) Configure the base detection engine
# 4) Configure dynamic loaded libraries
# 5) Configure preprocessors
# 6) Configure output plugins
# 7) Customize your rule set
# 8) Customize preprocessor and decoder rule set
# 9) Customize shared object rule set
#####

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

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

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
```

49,1 3%



The screenshot shows an Ubuntu desktop environment. On the left is the Dash sidebar with icons for Firefox, Mail, Dash, Trash, and several folders including 'Mauj Krdi' and 'CMD5 SNORT'. The top panel shows 'Activities', 'Terminal', and the date 'Apr 12 11:12'. Two terminal windows are open. The top terminal window shows the command `ls` in the `/etc/snort` directory, listing files: `classification.config`, `gen-msg.map`, `rules`, `snort.debian.conf`, `unicode.map`, `community-sid-msg.map`, `reference.config`, `snort.conf`, and `threshold.conf`. The bottom terminal window shows the contents of `/etc/snort/snort.conf`. It includes a `# Stop 01` comment, instructions for setting network variables, and configuration for `HOME_NET`, `EXTERNAL_NET`, `DNS_SERVERS`, `SMTP_SERVERS`, and `HTTP_SERVERS`.

```
root@ubuntu-VirtualBox:/etc/snort# ls
classification.config  gen-msg.map      rules             snort.debian.conf  unicode.map
community-sid-msg.map reference.config  snort.conf        threshold.conf
root@ubuntu-VirtualBox:/etc/snort#

root@ubuntu-VirtualBox:/etc/snort#
#####
# Stop 01: Set the network variables. For more information, see README.variables
#####
# Setup the network addresses you are protecting
#
# Note to Debian users: this value is overridden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET any

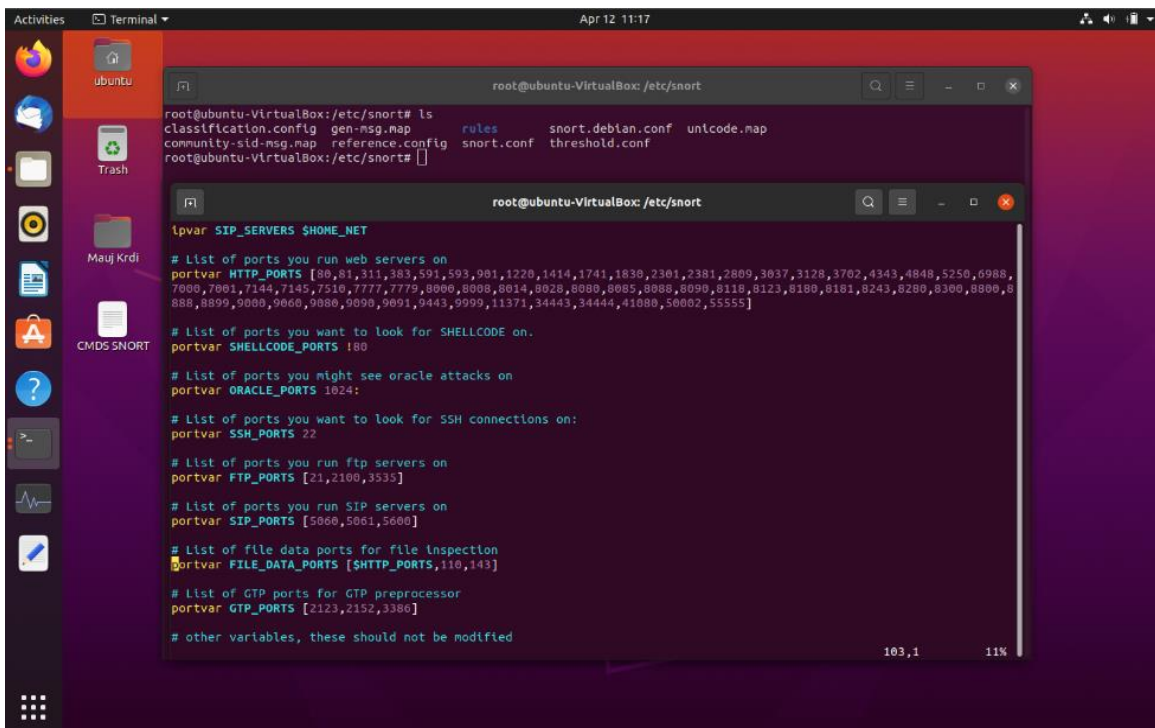
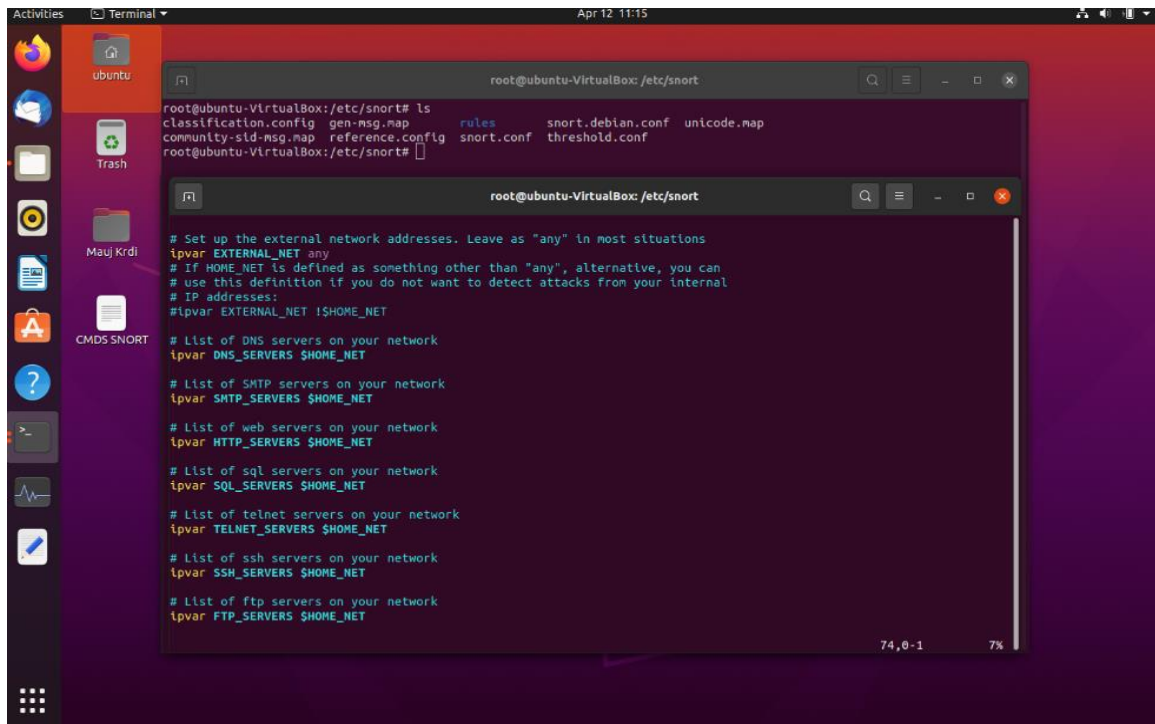
# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET

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

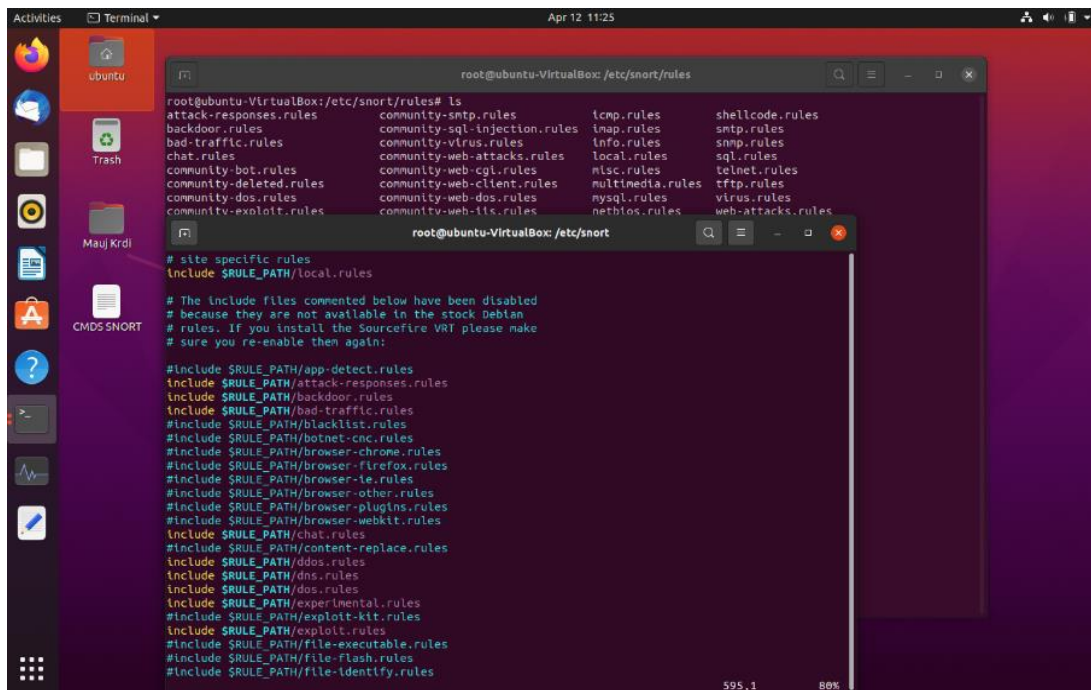
# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

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

63,1 5%
```

APPENDIX B - SNORT RULES FILE

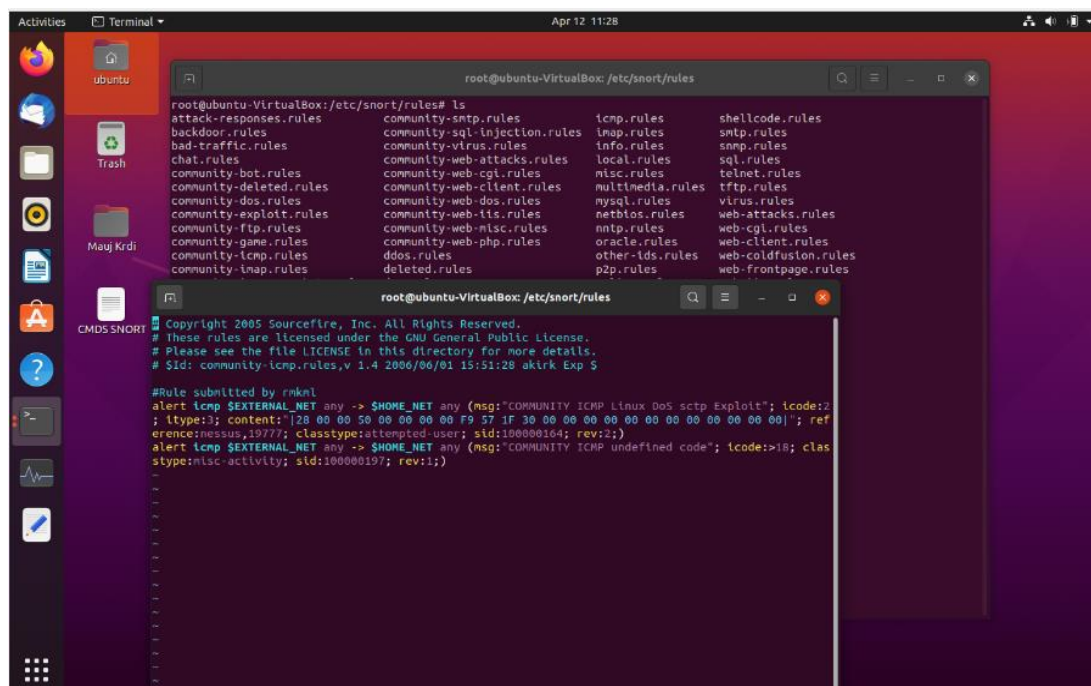


```
root@ubuntu-VirtualBox:/etc/snort/rules# ls
attack-responses.rules  community-sntp.rules      icmp.rules      shellcode.rules
backdoor.rules          community-sql-injection.rules  imap.rules      snmp.rules
bad-traffic.rules       community-virus.rules      info.rules      sql.rules
chat.rules              community-web-attacks.rules  local.rules     telnet.rules
community-bot.rules     community-web-cgi.rules     multimedia.rules  tftp.rules
community-deleted.rules community-web-client.rules    mysql.rules     virus.rules
community-dos.rules     community-web-dos.rules     netbios.rules   web-attacks.rules
community-exploit.rules community-web-tls.rules

# site specific rules
include $RULE_PATH/local.rules

# The include files commented below have been disabled
# because they are not available in the stock Debian
# rules. If you install the Sourcefire VRT please make
# sure you re-enable them again:

#include $RULE_PATH/app-detect.rules
#include $RULE_PATH/attack-responses.rules
#include $RULE_PATH/backdoor.rules
#include $RULE_PATH/bad-traffic.rules
#include $RULE_PATH/blacklist.rules
#include $RULE_PATH/botnet-cnc.rules
#include $RULE_PATH/browser-chrome.rules
#include $RULE_PATH/browser-firefox.rules
#include $RULE_PATH/browser-ie.rules
#include $RULE_PATH/browser-other.rules
#include $RULE_PATH/browser-plugins.rules
#include $RULE_PATH/browser-webkit.rules
include $RULE_PATH/chat.rules
#include $RULE_PATH/content-replace.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/experimental.rules
#include $RULE_PATH/exploit-kit.rules
include $RULE_PATH/exploit.rules
#include $RULE_PATH/file-executable.rules
#include $RULE_PATH/file-flash.rules
#include $RULE_PATH/file-identify.rules
```

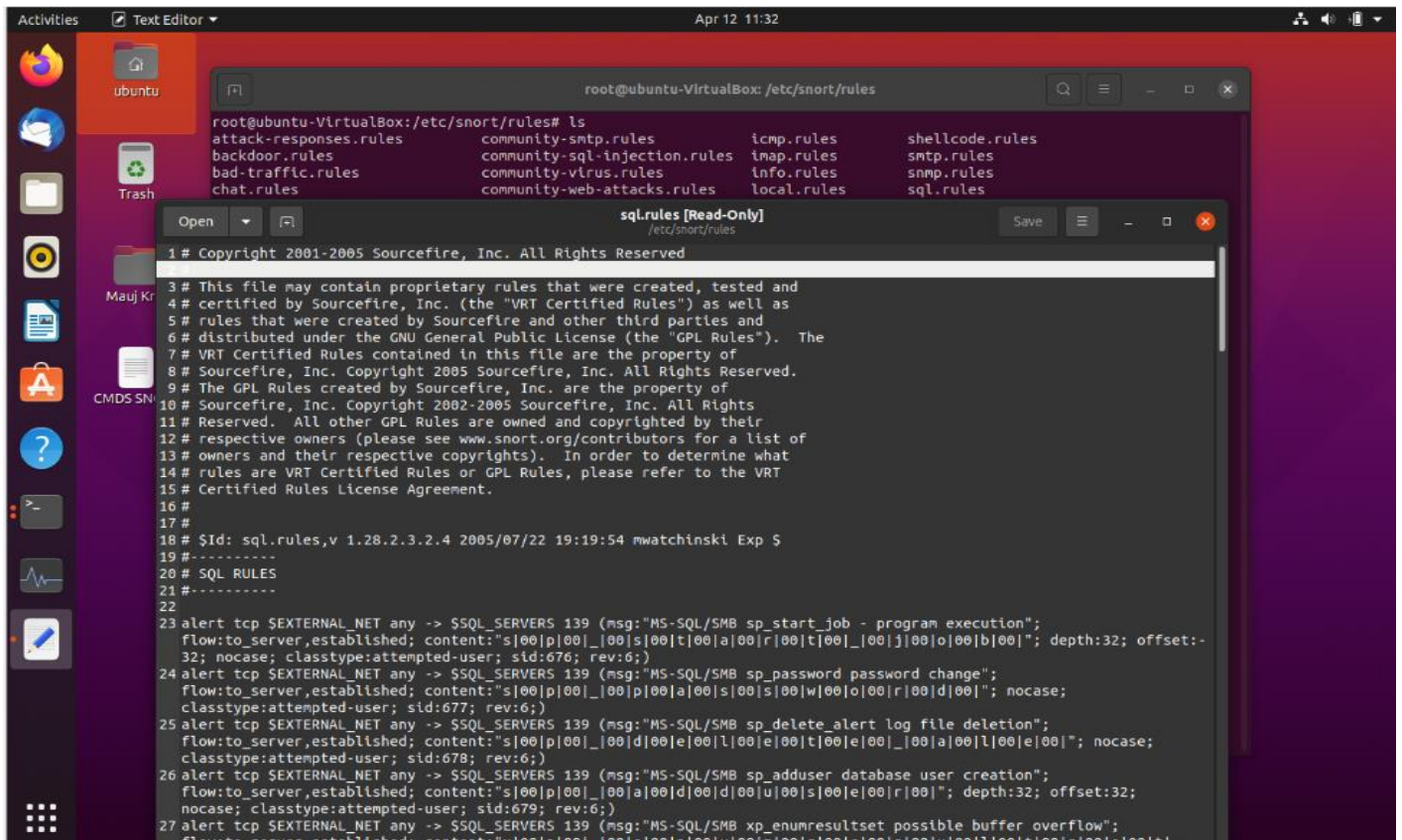


```
root@ubuntu-VirtualBox:/etc/snort/rules# ls
attack-responses.rules  community-sntp.rules      icmp.rules      shellcode.rules
backdoor.rules          community-sql-injection.rules  imap.rules      snmp.rules
bad-traffic.rules       community-virus.rules      info.rules      sql.rules
chat.rules              community-web-attacks.rules  local.rules     telnet.rules
community-bot.rules     community-web-cgi.rules     multimedia.rules  tftp.rules
community-deleted.rules community-web-client.rules    mysql.rules     virus.rules
community-dos.rules     community-web-dos.rules     netbios.rules   web-attacks.rules
community-exploit.rules community-web-tls.rules      nntp.rules      web-cgi.rules
community-ftp.rules     community-web-misc.rules    oracle.rules     web-client.rules
community-game.rules    community-web-php.rules     other-ids.rules  web-coldfusion.rules
community-icmp.rules    deleted.rules               p2p.rules        web-frontpage.rules

Copyright 2005 Sourcefire, Inc. All Rights Reserved.
# These rules are licensed under the GNU General Public License.
# Please see the file LICENSE in this directory for more details.
# $Id: community-icmp.rules,v 1.4 2006/06/01 15:51:28 akirk Exp $

#Rule submitted by rnkml
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"COMMUNITY ICMP Linux DoS sctp Exploit"; icode:2
; type:1; content:"[28 00 00 50 00 00 00 00 F9 57 IF 30 00 00 00 00 00 00 00 00 00 00 00]"; ref
erence:nessus,19777; classtype:attempted-user; sid:100000104; rev:2;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"COMMUNITY ICMP undefined code"; icode:18; clas
stype:isc-activity; sid:100000197; rev:1;)
```

APPENDIX C - SNORT MYSQL DATABASE



References:

1. https://www.cdac.in/login/training_mat/cybersecurity
2. <https://en.wikipedia.org/wiki/Wikipedia/cybersecurity/intrusion-detection-system/>
3. <https://www.geeksforgeeks.org/intrusion-detection-system-ids/>
4. <https://www.fieldengineer.com/>
5. <https://www.securityarchitecture.com/learning/intrusion-detection-systems-learning-with-snort/configuring-snort/>
6. <https://stackoverflow.com/questions/43314482/snort-errors>.
7. <https://github.com/manuvjeet/Project-IDS-Snort-University>
8. <https://www.rapid7.com/blog/post/2016/12/09/understanding-and-configuring-snort-rules/>
9. <https://resources.infosecinstitute.com/topic/snort-rules-workshop-part-one/>
10. <https://www.atlantic.net/vps-hosting/install-ntopng-to-monitor-network-traffic-on-ubuntu-20-04/>
11. <https://stackoverflow.com/questions/3740152/how-to-change-permissions-for-a-folder-and-its-subfolders-files-in-one-step>
12. <https://blog.eccouncil.org/how-do-intrusion-detection-systems-ids-work/>

13. <https://github.com/ipkn/crow/issues/133>