

CERTIFICATION OF APPROVAL

Network Denial of Service Defense System (nDos)

By

Muhd. U'mari Bin Zulkifli

A project dissertation submitted to the
Information & Communication Technology Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF TECHNOLOGY (Hons)
(INFORMATION & COMMUNICATION TECHNOLOGY)

Approved by,

(Mr. Low Tan Jung)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

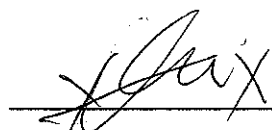
January 2007

£
QA
76.9
A25
M952

1) Computer security
2) Computer networks -- Security measures

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



MOHD U'MARI ZULKIFLI

4060

Abstract

Denial of Service attack is widely spread within virtual world as a malicious act that could give a huge impact in terms of the system performance and financial aspect. Network Denial of Service Defense System is an extension of intrusion detection system which incorporated with detection and prevention capabilities. The architecture of nDos is based on NIPS where it is place inline on the network statefully analyzing packet content and block certain packets that match a signature and alert on others. A NIPS protection is based on the content of packets. The system loads a large array of signatures. These signatures take the form of a string of data characteristic of some particular type of attack. When a data packet enters the network, the IDS/IPS examines that data against its database of signatures. If the data match, then the IDS/IPS takes appropriate action. In the case of an IDS, the intrusion attempt will be logged, whereas, in the case of an IPS, the system can drop the data packet, or even sever the offending machine's connection. Ndos provide web interface for data retrieval and manipulation. The front-end of the system is based on PHP/MySQL hence it could provide statistical analysis for managerial point of view. The back-end of nDos is using snort_inline as detection engine and iptables firewall for traffic prevention mechanism. Once an attack being launch nDos will logged the incident based on rules and configuration and iptables or generic firewall need to determine the traffic state whether to accept or drop the connection. Predefined thresholds value is important for DoS attack where a lot of connections of traffic generated hence when exceed the value the detection engine could identify such an attack. nDos is targeted for educational purpose and small-medium size enterprise because of there is only commercial IPS solution available in the market. Portability and compatibility is an issue where for future recommendation Live CD features could be implemented to provide high compatibility without concern of the OS.

TABLE OF CONTENTS

CERTIFICATION	i
ABSTRACT	ii
LIST OF FIGURE	iii
LIST OF ABBREVIATION	iv
CHAPTER 1	INTRODUCTION	1
	1.1 Background	1
	1.2 Problem Statement	2
	1.3 Objective/Scope of Work	4
CHAPTER 2	LITERATURE REVIEW	7
	2.1 The Architecture	7
	2.2 System Flow	9
	2.3 Strength of Network Intrusion Detection System	10
	2.4 Lightweight IDS Technology	12
	2.5 Linux Apache MySQL PHP (LAMP)	13
	2.6 Communication with Other Database	14
CHAPTER 3	METHODOLOGY/PROJECT WORK	15
	3.1 Procedure Identification	15
	3.2 Development Tools	16
CHAPTER 4	RESULT AND DISCUSSION	19
	4.1 Findings	19
	4.2 Prototyping phase	25
	4.3 nDos architecture	28
	4.4 Prototyping development phase.	31
CHAPTER 5	CONCLUSION	34
	5.1 Conclusion	34
	5.2 Recommendation	35
REFERENCES	36

LIST OF FIGURES

Figure 1.3.3	Division of tasks
Figure 2.1	A DDoS attack in operation.
Figure 2.5	PHP Architecture
Figure 3.1	Phases in Rapid Application Development
Figure 4.1.2	DoS packet analysis
Figure 4.1.3	System performance using 'top'
Figure 4.2.1	nDos login page
Figure 4.2.2	nDos main page
Figure 4.2.3	LAN
Figure 4.2.4	DMZ
Figure 4.2.5	Mixed
Figure 4.3	nDos architecture
Figure 4.4	Snort architecture
Figure 4.5	packet / traffic flow
Figure 4.6	ip address configuration
Figure 4.7	nDos startup
Figure 4.8	Attack logged sort by timestamp
Figure 4.9	Attack detection and logging

LIST OF TABLE

Table 4.1	IP address configuration
Table 4.2	Portion of DoS attack source
Table 4.3	Custom rule created

LIST OF ABBREVIATION

nDos – Network Denial of Service Defense System

PHP – Personal Home Page Hypertext Preprocessor

LAN – Local area network

DMZ – Demilitarized zone

DNAT= Destination Network Address Translation

SNAT- Source Network Address Translation

IPS – Intrusion prevention system

IDS – Intrusion detection system

NIPS – Network intrusion prevention system

NIDS – Network intrusion detection system

CERT ≡ Computer Emergency Response Team

DoS – Denial of Service

DDoS ≡ Distributed denial of service

IP – Internet Protocol

TCP ≡ Transport Control Protocol

UDP – User Datagram Protocol

LAMP ≡ Linux Apache MySQL PHP

CISSP - Certified Information Systems Security Professional

SME ≡ Small to Medium size Enterprise

IT – Information Technology

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

The Network Denial of Service Prevention System (NDos) project focuses on the designing and development of a Network Intrusion Prevention System (NIPS), which targeted to detect, alert and prevent Denial of Service through the network. With current SME network operation, Network Intrusion Prevention System (NIPS) is not widely being implemented. Through the rising of Internet as communication medium and as one method of transaction in doing business, security measure need to be implemented. Network security consists of the provisions made in an underlying computer network infrastructure, policies adopted by the network_administrator to protect the network and the network-accessible resources from unauthorized access and the effectiveness (or lack) of these measures combined together. Securing any network infrastructure is like securing possible entry points of attacks on a country by deploying appropriate defense such as a Network Intrusion Prevention System (NIPS). Most common network administrator usually just updating or patch which known as vulnerabilities for their security measure. Yes that is one of it but there is a lot of other way around to compromise a system. As for the Network Denial of Service Prevention System (NDos) it will protect the infrastructure from Denial of Service variants. Network Denial of Service Prevention System (NDos) is a Network Intrusion Detection System integrated with firewall architecture that could detect the intrusion and prevent the attack by using automated filtering policy. Network administrator could monitor the attack and analyze the log for countermeasure. For simplicity and portability logging and monitoring could be access using web browser.

1.2 PROBLEM STATEMENT

1.2.1 PROBLEM IDENTIFICATION

1.2.1.1 No exposure in terms of computer and network security

Currently in environment, promotion upon the information technology (IT) is moving asynchronously with the action to educate people to cope with the highly rapid development of information technology (IT).

1.2.1.1 Problem in providing network security measure

Without the exposure and lack of knowledge, a lot of network infrastructure being provision by default standard operation. Network security aspect is not a critical part during the implementation of the infrastructure as if it work based on what is outdated.

1.2.1.3 Alert and prevention not concurrently

Network Intrusion Detection System (NIDS) provide alert and logging but it is not a security measure to prevent the attack to the target system. Both actions must be taken simultaneously in order for administrator to secure the system.

1.2.1.4 Monitoring and analyzing time consuming

Another major drawback with current system is where configuration and settings must be done in command-line manner. Monitoring and analyzing the log cannot be automated preview to the administrator.

1.2.2 SIGNIFICANT OF THE PROJECT

With the implementation of Network Denial of Service Prevention System (NDos), it would revolutionize current security aspect for SME and non-profit organization, plus could expose to the world how important it is in the virtual world.

1.2.2.1 Multimedia Supercorridor (MSC) Application

In conjunction with Multimedia Supercorridor (MSC), the entire application environment runs through out the network. Network Denial of Service Prevention System (NDos) is one of the security measures that could be implemented

1.2.2.2 Reliability and mobility of access

Nowadays it is not difficult to get access to the internet even on mobile. Hence allow access to its configuration and interface using web-based interaction thus eliminate any other obstacle for accessing the system

1.2.2.3 Save time analyzing log and activities

Network Denial of Service Prevention System (NDos) could improve the way of forensic analysis by using web-based monitoring and presentation of the captured data.

1.2.2.4 Backup and defense

When a system without Network Denial of Service Prevention System (NDos) is under attack, a chance of the administrator to recover the system to working state is slightly zero. Deploying a network security measure in the system could eliminate the system being attack and restore process could be done flawlessly because it could defend the system during the attack.

1.3 OBJECTIVE AND SCOPE OF STUDY

1.3.1 OBJECTIVE OF THE PROJECT

1.3.1.1 To create an easy configuration system

A system that is easy to configure and operate smoothly without any hassle or technical expertise. Configuration is presented in graphical user interface to ease user navigation.

1.3.1.2 Reliable defense system for DoS variant.

Make use of the blocking capabilities of a firewall with the deep packet inspection of an Intrusion Detection System (IDS), when in place at suitable point could become a reliable defense system.

1.3.1.3 Make use of open source tools and embrace open source to the whole community

Lack of information about open source capabilities to the Malaysian community is one of the points that this project could help. By using open source tools it could create a reliable defense system targeted for SMEs and non-profit government.

1.3.1.4 Enhancing built-in firewall for secure protection

Iptables or ipchains in Linux environment come in a handy use for filtering process during attack. Including an Intrusion Detection System (IDS) with a firewall would result as network defense system.

1.3.2 SCOPE OF STUDY

To ensure the system that will develop meet the requirement and functional as required, several scope of study have to define. The scope of study that arrange by the most important area is explained below in ascending order.

1.3.2.1 Linux environment

In order to do an Intrusion Detection System (IDS) in Linux environment, it is very crucial to understand the architecture of the operating system. Since Linux is different from what the most well-known operating system from Windows, it is important to understand and make use of the Linux environment and secure the operating system for a security measure.

1.3.2.2 Research on firewall

Since the project is concern on DoS variant, a suitable firewall should been implementing for packet filtering. As in Linux environment a suitable firewall should be analyze for its capabilities for filtering DoS variant attack.

1.3.2.3 Alert, Monitor and Prevent

A secure network defense system supposes could alert the user, monitor the network and prevent the attack. There are lots of ways to alert the user nowadays such as mail.

1.3.3 Feasibility of the Project within the Scope and Time Frame

The development of this project involves the research work, interface design, coding and testing. Figure below show the scope of works which are representing in percentages.

- 40% in research work
- 5% in interface design
- 50% in coding
- 5% in testing

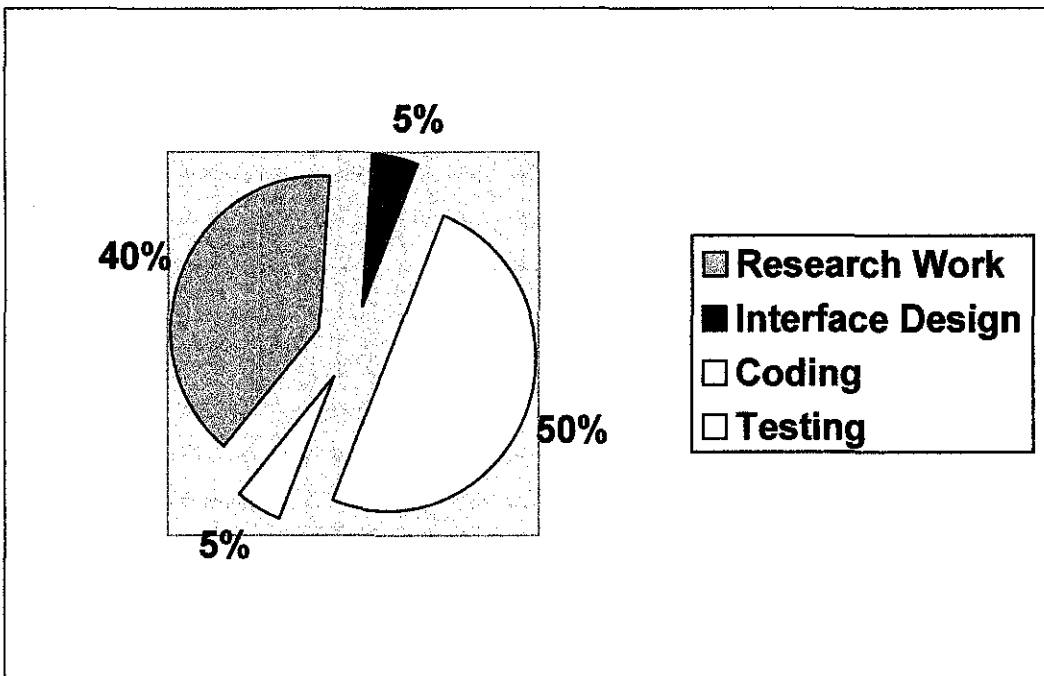


Figure 1.3.3: Division of tasks

The project consume a large amount of time especially in research work and coding because the research work need to be conduct in detail and the development of the application is not a simple ways as it involve difficulties in coding. However, the implementation of is expected to be completed in the given period of time.

CHAPTER 2

LITERATURE REVIEW

In the past decade, we have seen a rapid advancement in information technology applied by SME because of promotion by the government to almost every sector of industries. The Internet growth powered by the World Wide Web accelerates this advancement. However, it was noted that the increasing use of computer communication for many day to day tasks has resulted in a greater reliance on communication networks such as the Internet. The impact of a serious interruption to the operation of the Internet may have far reaching and costly consequences.

2.1 The architecture

With the advancement of Information Communication Technology in Malaysia, it is a must to implement up-to-date security measure for the network infrastructure. As the intrusion techniques become complex day by day, network administrator should not take security aspect as something not important.

According to Abhishek Singh, CISSP (2005), DoS is a Denial of Service to a victim trying to access a resource. In many cases it can be safely said that the attack requires a protocol flaw as well as some kind of network amplification. The motivation for DoS attacks is not to break into a system. Instead, it is to deny the legitimate use of the system or network to others who need its services.

Distributed DoS (DDoS) is the combined effort of several machines to bring down victim. In many cases there is a master machine that launches the attack to zombie machines that are part of a bot network, as shown below in Figure 1;

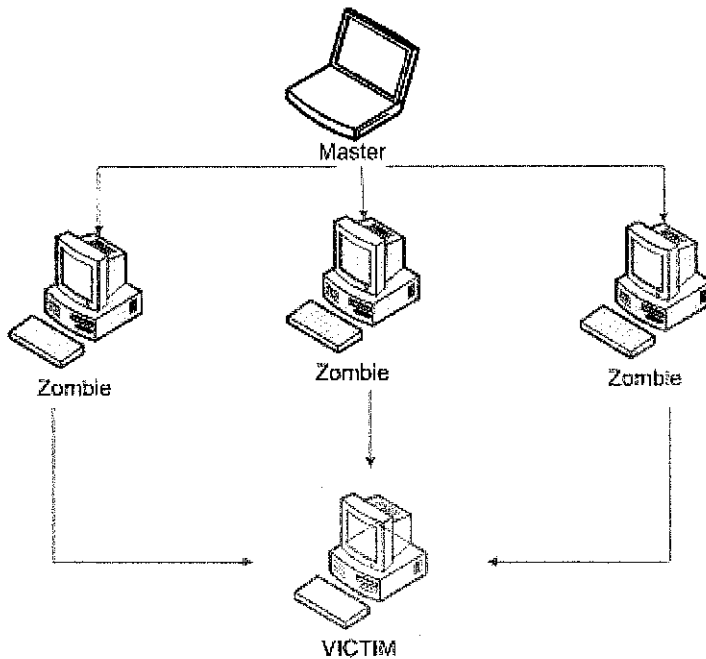


Figure 2.1: A DDoS attack in operation.

Network Denial of Service Prevention System (NDos) is an IPS that could alert and prevent the attack. IPS is any device (hardware or software) that has the ability to detect attacks, both known and unknown, and prevent the attack from being successful. There are five different categories of IPSs that focus on attack prevention at layers that most firewalls are not able to decipher. The five types of IPSs are inline NIPS, application-based firewalls/IPS, layer seven switches, network-based application IPSs, and deceptive applications.

Network Denial of Service Prevention System (NDos) is an inline NIPS. An inline NIPS offers the great capabilities of a regular NIDS with the blocking capabilities of a firewall. As with most NIDS, the user can monitor, in this case protect, many servers or networks with a single device. This can be both a blessing and a curse. If the system were to fail or crash the traffic would not get through the device. (ISS Guard actually fails open when the product crashes). If you are concerned about uptime and SLAs, this might cause a big issue for your network.

2.2 System Flow

So now it is understand that the internet will be use as a medium for business communication and transaction. Hence at certain point it must be secured from intruder. Network Denial of Service Prevention System (NDos) or inline NIDS works like a layer two bridge, sitting between the systems that need to be protected and the rest of the network. All traffic will pass through the inline NIDS and analyze the packet for any vulnerabilities that it is configured to look for such as DoS variant. If a packet contains a piece of information that trips a signature the packet can be forwarded or dropped and either logged or unlogged.

Administrator could monitor and will be alert if there is incident occur. The firewall will analyze the packet and drop any malicious packet transmitted to the Network Denial of Service Prevention System (NDos) based on certain rules. This automated process will set up a secure perimeter. Rely on the firewall, restoration and defense setup could be done before the disaster continue infected the whole system.

2.3 Strength of Network Intrusion Detection System (NIDS)

The network-based IDS have much strength due to its real-time packet capture and analysis functionality that cannot easily perform with host-based IDS alone. Below are some its strengths that make network-based IDS clearly a needed component to any security systems and policy

2.3.1 Cost of Ownership.

The network-based IDS allow strategic deployment at critical access points to view network traffic destined to numerous systems that need to be protected. It does not require software to be loaded and managed on a variety of hosts as does the host-based IDS. Being fewer detection points are required, it makes the cost of management more effective for an enterprise environment.

2.3.2 Evidence Removal.

The network-based IDS uses live network traffic for its attack detection in real-time and a hacker cannot remove this evidence once captured. This captured data not only has the attack in it but information that may help lead to his/her identification. This captured network traffic may also be needed as evidence leading to prosecution. One problem sometimes experienced with host-based IDS is that hackers understand most of the audit log files and that is usually the first place they go to cover their tracks and remove or damage this information.

2.3.3 Real-Time Detection and Response.

The network-based IDS detect malicious and suspicious attacks as they are occurring in true real-time and provide faster response and notification to the attack at hand. Example: A hacker initiating a network based denial of service (DOS) based on TCP can be instantly stopped by having the IDS send a TCP reset to terminate the attack before it crashes or damages a targeted host. Many times with host-based IDS, the notification from a particular log or event may be detected after the damage is already done or not at all if the system has crashed. Having real-time notification allows you to quickly react in a desired fashion. Some may want to allow further penetration so they can gather more information in a surveillance mode while other may opt for immediate termination of the attack.

2.3.4 Malicious Intent Detection.

Network-based IDS can also be very valuable in determining malicious intent. If network-based IDS are placed outside of detection Firewall it can detect attacks intended for resources behind the Firewall, although the firewall may be rejecting these attack attempts. Host-based IDS could not show these rejected attacks because they never hit the Host but are important to know the frequency and types of attacks being thrown at your network.

2.3.5 Operating System Independence.

The network-based IDS are not dependent on the host operating system for its detection source, as is the host-based IDS solution. All of the log information used with based solution requires the operating system functioning properly and not compromised in any way.

2.4 Lightweight IDS Technology

A lightweight intrusion detection system can easily be deployed on most any node of a network, with minimal disruption to operations. Lightweight IDS should be cross-platform, have a small system footprint, and be easily configured by system administrators who need to implement a specific security solution in a short amount of time. They can be any set of software tools which can be assembled and put into action in response to evolving security situations. Lightweight IDS' are small, powerful, and flexible enough to be used as permanent elements of the network security infrastructure.

Snort is perfectly suit to these roles, weighing in at roughly 100 kilobytes in its compressed source distribution. On most modern architectures Snort takes only a few minutes to compile and put into place, and perhaps another ten minutes to configure and activate. Compare this with many commercial NIDS', which require dedicated platforms and user training to deploy in a meaningful way. Snort can be configured and left running for long periods of time without requiring monitoring or administrative maintenance, and can therefore also be utilized as an integral part of most network security infrastructures.

Network Denial of Service Prevention System (NDos) is a modified version of Snort that accepts packets from generic Linux firewall such as iptables, via libipq, instead of libpcap. It then uses new rule types (drop, sdrop, reject) to tell iptables whether the packet should be dropped, rejected, modified, or allowed to pass based on a snort rule set. It has real-time alerting capability and rules based logging to perform content pattern matching and detect a variety of DoS variant attacks.

2.5 Linux Apache MySQL PHP (LAMP)

As a server based operating system, Linux could cater all of Network Denial of Service Prevention System (NDos) requirements. Using Apache web server as presentation layer or user interface to the system and PHP/MySQL database is use for interaction between the back-end and front-end. MySQL as commonly used for data storing and retrieving for logging. While PHP is use to process inputs from user interface and provide outputs during interaction between front-end and back-end. PHP and MySQL are often referred to as the "dynamic duo" of dynamic Web scripting. PHP and MySQL work very well together, in addition to the speed and features of each individual tool. . The operating system and other services need to be secured before Network Denial of Service Prevention System (NDos) implementation could be done.

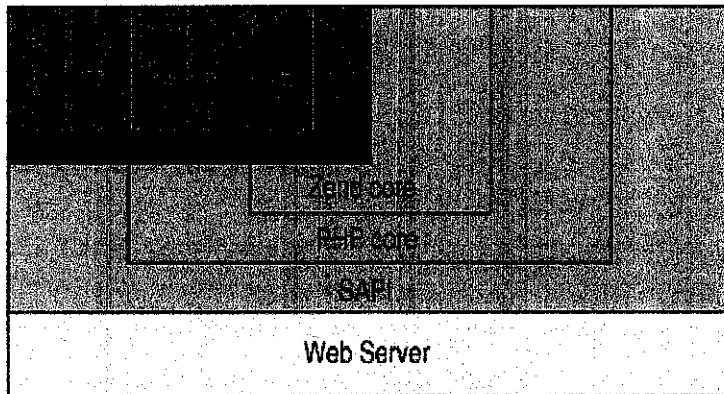


Figure 2.5: PHP Architecture

SAPI currently has server implementations for Apache, Roxen, Java (servlet), ISAPI (Microsoft IIS and soon Zeus), and AOLserver and of course CGI. All of PHP's functions are part of one of the layers with a side facing up in the architecture figure. Most functions such as the MySQL support are provided by an extension. Most extensions are optional. They can be linked into PHP at compile time or built as dynamically loadable extensions that can be loaded on demand.

2.6 Communication with Other Databases

Unlike other scripting languages for Web page development, PHP is open-source, cross-platform, and offers excellent connectivity to most of today's common databases including Oracle, Sybase, MySQL, ODBC (and others) using ADODB. PHP also offers integration with various external libraries which enable the developer to do anything from generating PDF documents to parsing XML.

CHAPTER 3

METHODOLOGY/PROJECT WORK

3.1 Procedure Identification

Throughout this project, Rapid Application Development (RAD) is chosen as the development methodology. RAD is a programming system that enables programmers to quickly build working programs. Historically, RAD systems have tended to emphasize reducing development time, sometimes at the expense of generating efficient executable code.

There are some advantages of using RAD methodology compare to the other traditional sequential development such as waterfall model. The term 'rapid' clearly means as 'fast', shown that the RAD can reducing the development time. Besides, the concept of iteration in RAD allows for corrective actions and encourages refinements. The iteration concept is perfectly ideal with the nature that human can't avoid mistakes and imperfect.

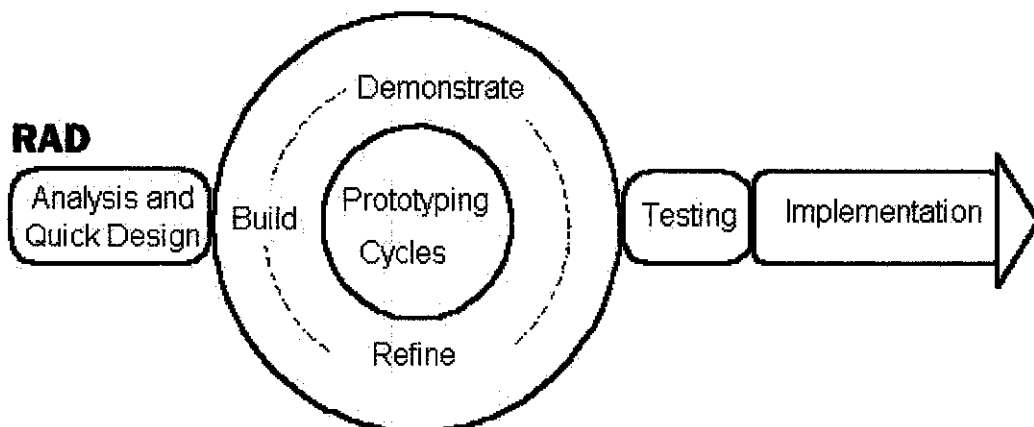


Figure 3.1: Phases in Rapid Application Development

3.2 Development Tools

3.2.1 PHP

PHP (PHP: Hypertext Preprocessor) is an open source, reflective programming language. Originally designed as a high level scripting language for producing dynamic web pages, PHP is used mainly in server side application software.

3.2.2 MySQL

MySQL is an open source RDBMS that relies on SQL for processing the data in the database. MySQL provides APIs for the languages C, C++, Eiffel, Java, Perl, PHP and Python. In addition, OLE DB and ODBC providers exist for MySQL data connection in the Microsoft environment. A MySQL .NET Native Provider is also available, which allows native MySQL to .NET access without the need for OLE DB

3.2.3 Snort Package

Snort is an open source network intrusion detection system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more.

3.2.4 Macromedia Dreamweaver

Macromedia Dreamweaver is a web development tool, created by Macromedia (now Adobe Systems), which is currently in version 8. Initial versions of the application served as simple WYSIWYG HTML editors but more recent versions have incorporated notable support for many other web technologies such as CSS, JavaScript, and various server-side scripting frameworks.

3.2.5 iptables

Packet filtering tool that is used by SmoothWall to provide firewalling capabilities incorporated in each Linux distro. Iptables provides a set of hooks within the Linux kernel for intercepting and manipulating network packets which could perform network address translation, stateful tracking and packet enqueueing to userspace. Set of rules define by the system administrator to deal with network packets. There are three built-in tables, each of which contains some predefined chains. The administrator can create and delete user-defined chains within any table. Filter table is responsible for filtering which accept or block a packet to proceed whether incoming, outgoing or reroute the traffic. Nat table is responsible for rewriting packet addresses or ports. Primarily the first packet in any connection pass through this table to determine how all packets in that connection will be rewritten. Some predefined chains are capable for DNAT (destination-NAT) and SNAT (source-NAT). Mangle table provide such a features for adjusting packet options for example quality of service (QoS). There are four target rules within iptables such as ACCEPT, DROP, QUEUE or RETURN. Those are built-in chains where the packet is diverted for processing. Connection tracking is another important feature built on top of iptables. It allows the kernel to keep track of all logical network connections or sessions through classification using four states such as NEW, ESTABLISHED, RELATED and INVALID. A stateless protocol like UDP could be tracked using these features. The connection tracking information would make packet filtering rules more powerful and easier to manage.

3.2.6 Ethereal

Widely use packet sniffer for network analysis and packet inspection. During initial testing Ethereal is used to analyze a hostile network environment to analyze DoS attack such as udp flooding. The filter features in Ethereal provide information about the packet and yet could classify all of the traffic to meet certain criteria, hence the behavior from such DoS attack could be identify for further analysis.

3.2.7 Putty

Remote access application to configure a machine remotely via telnet, ssh or Xwindows.

CHAPTER 4

RESULT AND DISCUSSION

4.1 FINDING

The project research has been focusing on alert and prevention of denial of service attack, snort_inline, generic firewall, ip_queue module and PHP/MySQL. The finding of this project will be present into:

nDos web-enable denial of service prevention

1. initial testing phase
2. prototyping phase

4.1.1 Initial Testing Phase

Denial of Service attack as mentioned mainly the objective is to consume the victim resources hence no legitimate access available. As for the test bed for nDos projects the system run on **Ubuntu 5.10 (Breezy Badger)** installed in **VMware Workstation 5.5.2**. Memory capacity is limit up to **256MB** in VMware settings. Bridged network environment using virtual Vmnet bridged on one actual or physical network card. A network setting on the test bed is according to the code below:

Table 4.1 : IP address configuration

```
h4x0r@nDoS:~$ sudo dhclient eth0
Password:
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 165.0.6.252
bound to 165.0.6.103 -- renewal in 128040 seconds.
h4x0r@nDoS:~$ route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
165.0.6.0 * 255.255.255.0 U 0 0 0 eth0
default 165.0.6.254 0.0.0.0 UG 0 0 0 eth0
h4x0r@nDoS:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target prot opt source destination

Chain FORWARD (policy ACCEPT)
target prot opt source destination

Chain OUTPUT (policy ACCEPT)
target prot opt source destination
```

IP address assigned by DHCP server and iptables accept all inbound and outbound connection. The node default route is a gateway at 165.0.6.254 which means all the traffic must be passing through that particular node.

To analyze the system against Denial of Service attack, a simple DoS program constructed using C language. The program attack any host by sending payload consist of junk data rapidly on specific port using UDP protocol. The structure or algorithm of the program is accept 2 parameters which is hostname and port number then create a UDP connection. When connection established the program will send the payload rapidly. The payload being sent is based on the size variable defined in the code.

Table 4.2 : Portion of DoS attack source

```
#define PAYLOAD
"0123456789ABCDEFGHIJKLMN0PQRSTUVWXYZ"
#define SIZE 45

for(;;)
{
send(s, PAYLOAD, SIZE, 0);
}
}
```

During the penetration test, the DoS program is launch from other host outside the victim or test system subnet. The attacker machine at 160.0.226.22 launched the attack to 165.0.6.103 which means the packet is route by a gateway 165.0.6.254. On the test system a packet sniffer been used to analyze the attack. Using Ethereal 0.10.12 to capture the packet on eth0 provide huge information on DoS attack behavior and mechanism.

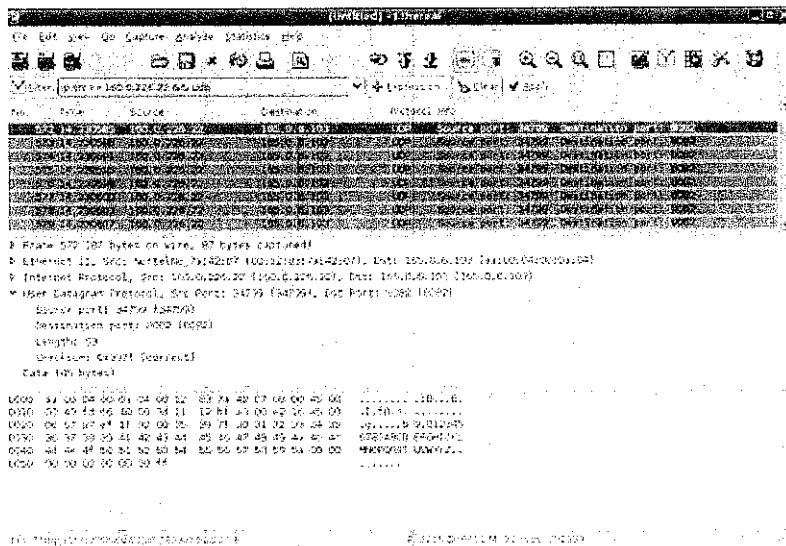


Figure 4.1.2 : DoS packet analysis

4.1.2 Attack analysis

The packets information consists of the protocol which is UDP. The payload data is including the source and destination of the packets. During the attack the average load of the system is around 50 % to 60 %. However the task manager for the host system showed VMware consume almost 90% of the resources. The background process on the victim or test system fluctuates randomly as there are shortage of resources and memory available. The simple basic DoS attack proves that the attack could affect the system performance.

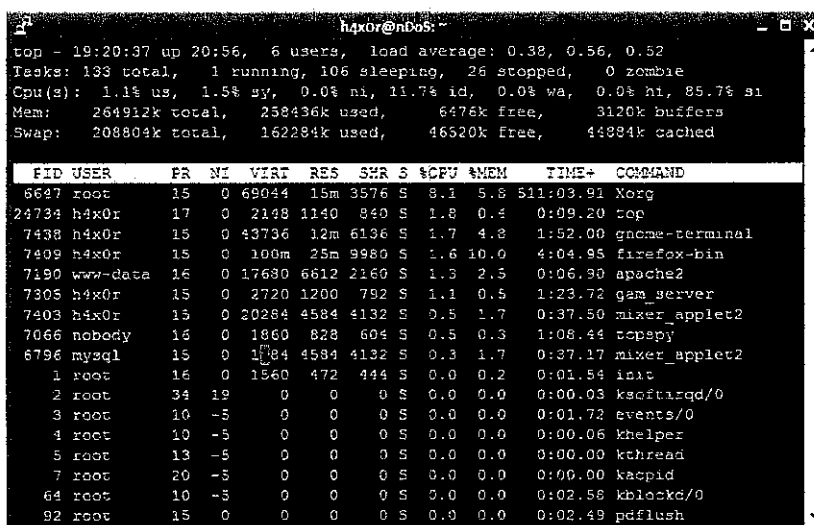


Figure 4.1.3: System performance using 'top'

After analyzing the packet and take the system setup condition itself there are certain variables that could be take as parameters to detect and prevent such DoS attack. A Denial of Service (DoS) attack is a saturation attack, in which a service provided by a computer is overloaded with requests from illegitimate sources, thus preventing the service from performing its intended function for legitimate users. A distributed Denial of Service attack (DDoS) is a DoS attack which is launched from a number of hosts in parallel, so that source of the attack is bolstered both by redundancy and multiplicity. The simulation attack caused the test system overload. The packets must be stop from arriving by using packet filtering application or firewall such as iptables.

4.1.3 Hypothesis

Based on the penetration test certain requirement is needed to consider as solution to solve the problem. An effective DoS/DDoS detection system is capable of detecting and responding to denial-of-service attacks on networked computers in real time. Thresholds value is used to identify logic attacks such as Ping of Death, Land attack and Teardrop. It depend on how accurate the values are set as thresholds do not take into account legitimate surges in traffic caused by benign behavior for example everyone logging in to the system at the same time. The method described above, threshold-based detection, is only one basic method, and there are ways to evade it. Combining that technique with others greatly improves your system's ability to determine benign traffic from an attack.

Multiple detection mechanism is another requirement instead of thresholds because it is not static. Ability to learn the network or statistical anomaly analysis capability could provides a solid view of how the network looks over time, thus it could detect quickly. Statistical anomaly analysis provides more realistic picture of normal network behavior.

Attack coverage is another method where it uses pattern and protocol dissection. The CERT Coordination Center (CERT/CC), a federally funded research and development center operated by Carnegie Mellon University, divides attacks into three classes: bandwidth (or flood) attacks, protocol attacks, and logic attacks. Bandwidth attacks are relatively straightforward attempts to consume resources,

such as network bandwidth or equipment throughput. Protocol attacks take advantage of the inherent design of common network protocols. These attacks use the expected behavior of protocols such as TCP, UDP, and ICMP to the attacker's advantage, and essentially befuddle the victim with specially crafted packets, as it tries to conform to standard protocol practice. Logic attacks exploit specific known vulnerabilities in network software, such as a Web server, or the underlying TCP/IP stack. While the categorization of DoS attacks is not well standardized, effective IDS must be able to detect a DoS attack regardless of the actual means of attack.

Today's networks are extremely heterogeneous, comprised of so many environments, systems and servers that a much more granular approach to detection is necessary. Most of Intrusion Detection System (IDS) using sensor based are limited in scope of coverage for the entire network. The IDS must be able to monitor, with distinct and specialized security policies, each subset of the aggregate network traffic. Granular profiling is necessary in order to learn the normal traffic behavior accurately. Furthermore, only with granular separation will it be feasible to accurately isolate the attacking traffic and take countermeasures with little side effect.

The alert features or alarm function is not just for detecting an attack but it must be accurately raised. DoS and DDoS attacks are extremely fast. Given the tendencies of DoS and DDoS attacks to last for more than a minute or two, what are the number of alarms or alert raised if the IDS alerts every seconds throughout the attack duration. Effective IDS must recognize the sequence of attacks and provide consolidated alerts without losing critical attack information. All of the data is crucial for forensic analysis and for incident response.

Accurate detection is the first step and having the IDS to respond in real-time is a goal. How fast the defense system respond to an attack could make a difference. Response actions are another requirement. The IPS should be able to automatically filtering mountain of packets selectively and in real-time. However the user must be in control and automatic response should be enable upon reliable detection. The core of nDos system which is snort inline achieves several basic features of those

requirements. Several testing phases should be done parallel with the development of nDos in order to produce reliable output.

4.2 Prototyping Phase

nDos login page to provide roles and security for an intrusion prevention system. Username and roles creation is stored using MySQL database. Access credential is a must for a security system. Meanwhile maintaining log and attack traffic user access could be used as key for assigning alert group to a specific attack.

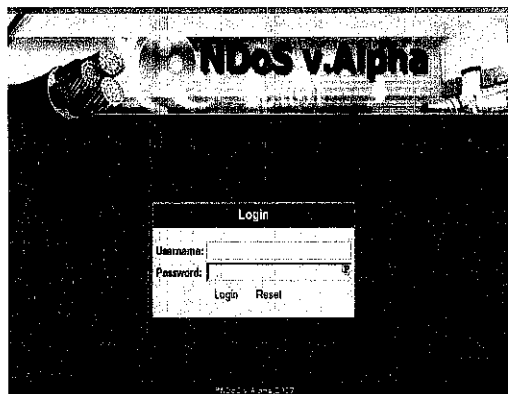


Figure 4.2.1: nDos login page

Status of alert being queried and system status display for current and reliable incident response system. Attack monitoring is on the main page for easy access and is classified based on time, volume and basic TCP/IP protocol. Sensor is a point where an IPS (Intrusion Prevention System) / IDS (Intrusion detection system) analyze the traffic.

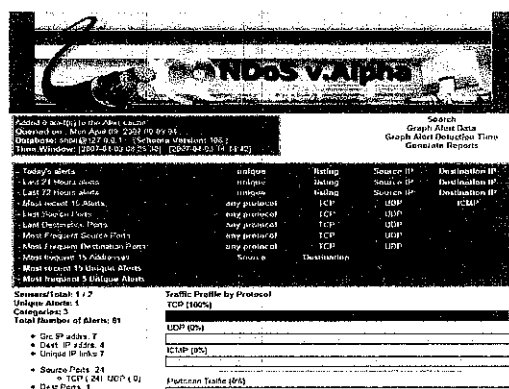


Figure 4.2.2: nDos main page

Sensor placement is important for a network security comparatively with the performance aspect. Several sensors can be place in such way to protect against internal and external attack. For example three type of network configuration which could be combine together as being practice from basic LAN setup in SOHO ,SME and educational institution;

1. LAN
2. DMZ
3. Mixed

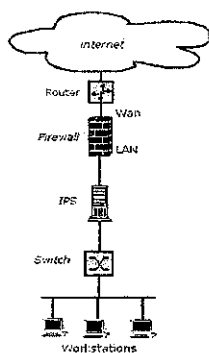


Figure 4.2.3: LAN

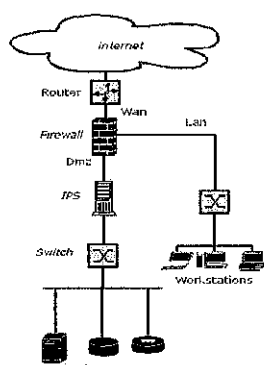


Figure 4.2.4: DMZ

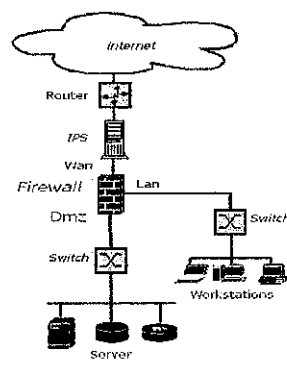


Figure 4.2.5: Mixed

4.2.1 LAN

Detection engine for nDdos could be configure to protect internal and external attack by deployment of a single snort_inline run on a machine that capable of routing traffic from internal and external by utilizing two network interface. Bridging capabilities within Linux environment by using bridge-utils-1.0.0 or forward/prerouting using iptables could provide both protections using single machine for NIPS. A firewall is needed to protect LAN against incoming and outgoing attack. A machine running nDdos could be place behind the firewall.

4.2.2 DMZ

DMZ is a part of the network that is neither part of the internal network nor directly part of the Internet which basically a network sitting between two networks. Sensor is place behind the firewall to protect servers cluster environment where it must be reachable for anyone. The most dangerous intrusions are those in the internal network. Putting a sensor here as shown in the figure will also make you able to detect attacks coming from insiders, nodes within the internal network. In this way the sensor also has a better defense against for instance DoS attacks directed at the IPS itself. The best solution is to apply sensors on both sides of the internal firewall, yielding a complete overview of any attack and whether they penetrate the firewall.

4.2.4 Mixed

Mixed configuration is a blend of basic LAN configuration and DMZ configuration for complex network applied within educational institution or SME. The setup is differ where segregation between LAN where it suppose to be protected against any external attack and DMZ which also need to be protect without degrading servers or cluster high availability. Deployment of sensor should be the first node before all of the traffic flow in and out of the protected network.

4.3 nDos Architecture

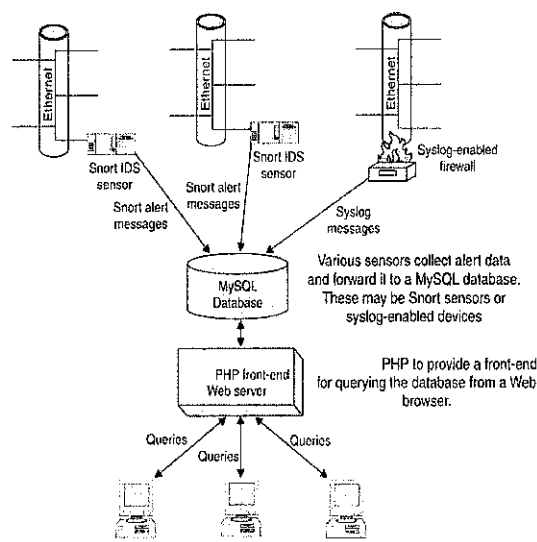


Figure 4.3: nDos architecture

Various sensors deploy within various network for network protection. alert messages logged when incident or intrusion detected by nDos core which is snort_inline, the data is forwarded to MySQL for data analysis and PHP front-end provide essential features for data retrieval and manipulation from web interface.

nDos utilizing snort_inline as the core of the system for detection engine and generic Linux firewall such as iptables or ipfw. As looking into classical approach of an IPS third party firewall could provide the capability for prevention the malicious traffic but as Linux technologies being well explore, a loadable kernel module; ip_queue eliminate extra cost for constructing an IPS. The module is important to redirect all traffic flow through the kernel space into user space for snort_inline detection. Taking advantage of ip_queue module, iptables as Linux generic firewall could become a prevention mechanism for accepting or blocking traffic.

4.3.1 snort_inline architecture

The core of nDos is snort_inline which is a modified version of Snort. It accepts packets from firewall via libipq instead of libpcap which commonly use for packet inspection. Based on predefined rules snort_inline automatically instruct the firewall whether the traffic need to be allow, modified or reject.

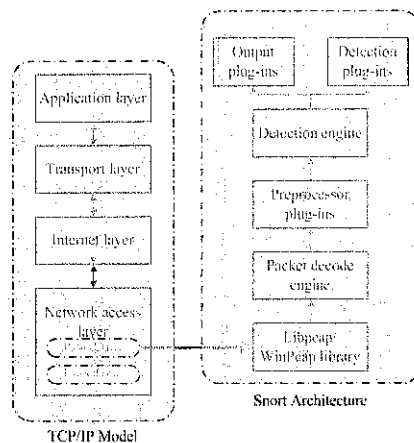


Figure 4.4: Snort architecture

In terms of incident reporting, within snort_inline it already incorporated with various output module which could be choose to suit data logging features for a system. nDos interface using PHP/MySQL hence Snort output module is available for MySQL logging. This feature is important in order to provide statistical analysis for managerial usage.

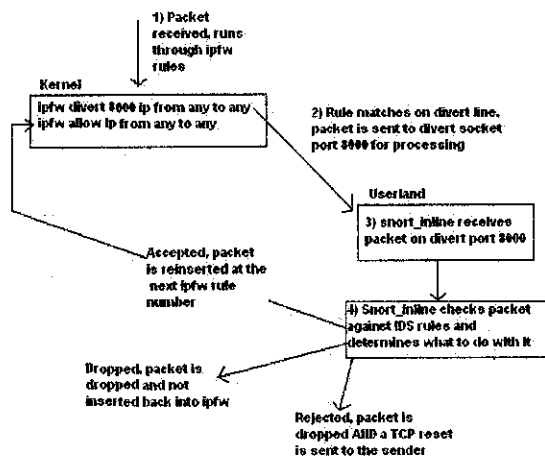


Figure 4.5 : packet / traffic flow

4.3.2 iptables

NIPS need a prevention mechanisms thus nDdos using iptables which is the name of the user space tool by which administrators create rules for the packet filtering and NAT modules. Traffic that flows to kernel space in Linux environment will be redirect to user space to create a stateful firewall using stack. Rules created using iptables could specify which port or network to protect using chain. When a packet is sent to a chain, it is compared against each rule in the chain in order. The rule specifies what properties the packet must have for the rule to match.

4.3.3 ip_queue module

It is a packet handling in userspace for iptables which is distributed with kernel in Linux operating system. a handler, which deals with the actual mechanics of passing packets between the kernel and userspace. The loadable kernel module passing the packet stack for a userspace application to receive, possibly manipulate, and issue verdicts on packets which is snort_inline.

4.4 Prototyping development phase

Snort_inline is already compiled with MySQL output modules, iptables and ip_queue module using VMware on Ubuntu 6.06. Rules created to protect web servers on the mixed network configuration. NDos is set up with LAN ip *165.0.6.72* gateway *165.0.6.254*.

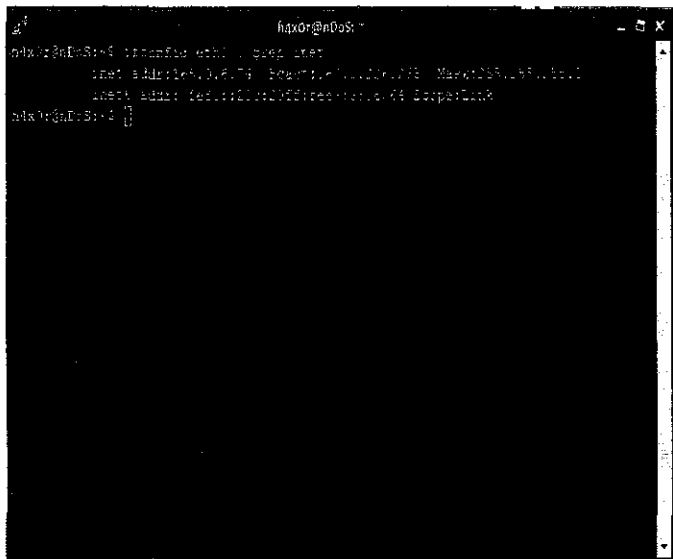


Figure 4.6: ip address configuration

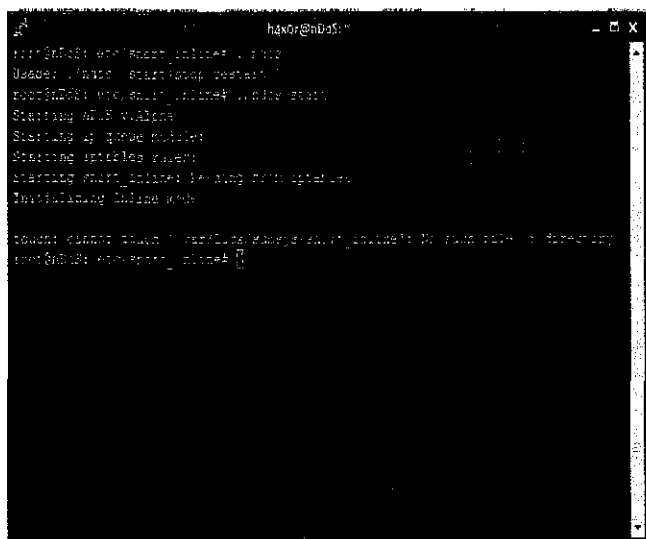


Figure 4.7: nDos startup

Protected network is configured for 165.0.6.0/24 for the whole 254 class C network which is the LAN. A rule created to test the functionality of nDos to protect a web server within the protected network. Any external incoming connection to web server port 80 will be block.

Table 4.3: Custom rule created

```
drop tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
(classtype:attempted-user;msg:"WEB-ATTACKS - Intrusion webserver
attempted");
```

nDns is start by executing a shell script to invoke the iptables rule, ip_queue module and snort_inline to be executed. All incoming traffic flows from the kernel space is load into queue through iptables for snort_inline analysis. Once the system is up and running it is time for attack to determine whether the system could function as it should be.

A simple denial of service attack is launch from 160.0.226.206 which is not a part of protected network. The data and information gather from the attack theoretically should prevent any external intrusion to the protected network

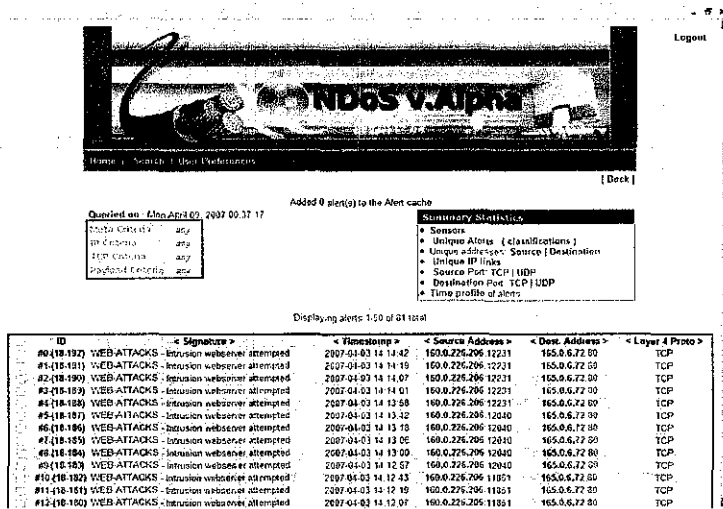


Figure 4.8: Attack logged sort by timestamp

```

haxor@nDoS:~$
[**] [Classification: Attempted User Privilege Gain] [Priority: 1] [TCP] 160.0.2
26.206:12040 -> 168.0.6.72:80
04/09-14:10:10.489360 [**] [100:0] WEB-ATTACKS - Intrusion webserver attempted
[**] [Classification: Attempted User Privilege Gain] [Priority: 1] [TCP] 160.0.2
26.206:12040 -> 168.0.6.72:80
04/09 24:13:42.003185 [**] [100:0] WEB-ATTACKS - Intrusion webserver attempted
[**] [Classification: Attempted User Privilege Gain] [Priority: 1] [TCP] 160.0.2
26.206:12040 -> 168.0.6.72:80
04/09-14:10:55.015696 [**] [100:0] WEB-ATTACKS - Intrusion webserver attempted
[**] [Classification: Attempted User Privilege Gain] [Priority: 1] [TCP] 160.0.2
26.206:12040 -> 168.0.6.72:80
04/09-14:11:01.312602 [**] [100:0] WEB-ATTACKS - Intrusion webserver attempted
[**] [Classification: Attempted User Privilege Gain] [Priority: 1] [TCP] 160.0.2
26.206:12040 -> 168.0.6.72:80
04/09 14:11:07.121967 [**] [100:0] WEB-ATTACKS - Intrusion webserver attempted
[**] [Classification: Attempted User Privilege Gain] [Priority: 1] [TCP] 160.0.2
26.206:12040 -> 168.0.6.72:80
04/09-14:11:19.174867 [**] [100:0] WEB-ATTACKS - Intrusion webserver attempted
[**] [Classification: Attempted User Privilege Gain] [Priority: 1] [TCP] 160.0.2
26.206:12040 -> 168.0.6.72:80
04/09-14:11:40.989494 [**] [100:0] WEB-ATTACKS - Intrusion webserver attempted
[**] [Classification: Attempted User Privilege Gain] [Priority: 1] [TCP] 160.0.2
26.206:12040 -> 168.0.6.72:80
root@nDoS:/etc/snort_inline#

```

Figure 4.9: Attack detection and logging

During the attack take place it is obvious that the core of the system is able to analyze the incoming traffic. Attack prevention and detection functionality achieve through rule created to protect web servers within protected network. Once the attack is detected then based on prior rule created, snort_inline automatically drop the traffic using iptables.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5. CONCLUSION OF THE PROJECT

Network Denial of Service Prevention System (NDos) is a project intend to construct a network security measure based on open source tools. The main objectives are to alert, prevent and analyze Denial-of-Service variants attack. The development of the nDos system is in conjunction with Malaysia government to create an E-government to operate in a paperless environment. However, when talking about online system definitely adequate security measure should be taking into account. Implementation of and intrusion detection or intrusion prevention system is one of the aspect for proper network security. With the used nDos an organization could experience how NIPS benefit them before jump into the commercial NIPS. The web based interface with graphical user interface would ease upon user interaction with the system. Having the system execute using a single command create easy setup and configuration without any complicate SOP. Besides, it also could encourage health awareness to patients whereby they can monitor and keep track their own health history.

5.2 RECOMMENDATION

There are some future upgrade and enhancement that could be implemented to the system. Some of the future upgrade is involving the expansion of scope of nDos system.

Artificial intelligent – for this project, the automation process is still conventional. In future, incorporated with neural networks or artificial intelligent could benefit the system because there are several of hostile network environment that need the system to be in learning state. Plus, detection and prevention with AI could produce a reliable output once the system done regression test.

Portability - the nDos system can be expand through the integration of Live CD which means zero installation and configuration. NDos could be load from a CD or pen drive hence increase the portability and provide on-the-fly configuration. Eliminate the risk applied to the physical hardware or software. Current operating system, files or database leave intact without any modification

Mobility – Combine with cellular technology, nDos could provide real-time alert response via SMS, MMS or 3G. Incident logged to MySQL database could be extract for example Ozeki SMS server is compatible with MySQL database hence data extraction from nDos database could be simplify for SMS system.

REFERENCES

1. Martin Roesch, 1999, *Snort - Lightweight Intrusion Detection for Networks*
<http://www.snort.org/docs/lisapaper.txt>
2. Abhishek Singh, CISSP, 2005, *Demystifying Denial-Of-Service attacks*
<http://www.securityfocus.com/infocus/1853>
3. Peter Jackson.2005, *Detection of Network Threats using Honeypots*
4. Dale Dougherty, 2001, *LAMP: The Open Source Web Platforms*
<http://www.onlamp.com/pub/a/onlamp/2001/01/25/lamp.html>
5. Netfilter/iptables
<http://en.wikipedia.org/wiki/Iptables>
6. Lance Spitzner 2004-01-14 *Problems and Challenges with Honeypots*
<http://www.securityfocus.com/infocus/1757>

APPENDICES

APPENDICES

Snort_inline configuration

```
##### Define Variable : #####

var EXTERNAL_NET !$HOME_NET
var HOME_NET 165.0.6.0/24
var SSH_PORTS 22
var DNS_SERVERS $HOME_NET

var SMTP_SERVERS $HOME_NET
var HTTP_SERVERS $HOME_NET
var SQL_SERVERS $HOME_NET
var TELNET_SERVERS $HOME_NET
var SNMP_SERVERS $HOME_NET

var HTTP_PORTS 80
var SHELLCODE_PORTS !80
var ORACLE_PORTS 1521
var AIM_SERVERS
[64.12.24.0/24,64.12.25.0/24,64.12.26.14/24,64.12.28.0/24,\
64.12.29.0/24,64.12.161.0/24,64.12.163.0/24,205.188.5.0/24,205.188.9.0
/24]

var RULE_PATH /etc/snort_inline/rules

include $RULE_PATH/classification.config
include $RULE_PATH/reference.config

### Configure Preprocessor : ###
config disable_decode_alerts
config checksum_mode: none
config detection: search-method lowmem

#preprocessor perfmonitor: time 60 file
/var/log/snort_inline/perfmon.txt pktcnt 500

#Flow, Frag and stream
preprocessor flow: stats_interval 0 hash 2
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy first detect_anomalies
preprocessor stream4: disable_evasion_alerts,detect_scans
#preprocessor portscan2-ignorehosts: $HOME_NET
#preprocessor portscan2: scanners_max 256 targets_max 1024,
target_limit 5, port_limit 20, timeout 60
preprocessor arpspoof

preprocessor rpc_decode: 111 32771
preprocessor bo
preprocessor telnet_decode

### Configure Output : ###
```

```

output database: alert, mysql, dbname=ndos user=ndos host=localhost
password=nd0s detail=full

output alert_syslog: LOG_AUTH LOG_ALERT LOG_CONS LOG_NDELAY LOG_PERROR
LOG_PID

output alert_fast: snort_inline-fast.log

include $RULE_PATH/attack-responses.rules
include $RULE_PATH/backdoor.rules

include $RULE_PATH/netbios.rules

include $RULE_PATH/rpc.rules
include $RULE_PATH/spyware-put.rules

include $RULE_PATH/web-client.rules
include $RULE_PATH/x11.rules
include $RULE_PATH/web.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules

```

nDos startup

```

#!/bin/bash
#
# chkconfig: 2345 20 40
# description: Snort_inline is an IPS implementation of the popular snort IDS package
# processname: snort_inline
# config: /etc/snort_inline/snort_inline.conf

# Source function library.
#. /etc/init.d/functions

# Source networking configuration.
#. /etc/sysconfig/network

[ -f /usr/local/bin/snort_inline ] || exit 0

start() {
    # Start daemons.
    echo -en 'Starting nDoS v.Alpha\n'
    echo -n $"Starting ip_queue module:"
    lsmod | grep ip_queue >/dev/null || /sbin/modprobe ip_queue;
    echo -e "\t\t\t [ \033[32mOK\033[37m ]"

    echo -n $"Starting iptables rules:"
    iptables -N ip_queue
    iptables -I INPUT -p all -j ip_queue

```

```

#Add new IPTABLES rules here and they will be added into the ip_queue Ruleset
#iptables -I ip_queue -p tcp --dport 80 -j QUEUE

echo -e '\t\t\t\t [ \033[32mOK\033[37m ]'
echo -n $"Starting snort_inline: "
#daemon
    /usr/local/bin/snort_inline -c /etc/snort_inline/snort_inline.conf -Q -N -I \
        /var/log/snort_inline -t /var/log/snort_inline -D

RETVAL=$?
echo
[ $RETVAL = 0 ] && touch /var/lock/subsys/snort_inline
}

stop() {
    # Stop daemons.

    echo -n $"Shutting down snort_inline: "
    killall snort_inline
    echo -ne $"
Removing iptables rules:"
    iptables -F ip_queue
    iptables -D INPUT -p all -j ip_queue
    iptables -X ip_queue
    echo -e '\t\t\t\t [ \033[32mOK\033[37m ]'
    echo -n $"Unloading ip_queue module:"
    rmmod ip_queue
    echo -en '\t\t\t\t [ \033[32mOK\033[37m ]'
        echo -en '\nQuit nDoS\n'
    RETVAL=$?
    echo
    [ $RETVAL = 0 ] && rm -f /var/lock/subsys/snort_inline
}

restart() {
    stop
    start
}

# Arguments passed.
case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
)

```

```
restart)
  restart
  ;;
*)
  echo $"Usage: $0 {start|stop|restart}"
  exit 1
esac

exit $RETVAL
```