INTRUSION DETECTION SYSTEM

by

Abdul Hadi Bin Mohamad

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Technology (Hons)
(Information Technology)

December 2004

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Intrusion Detection System (IDS)

by

Abdul Hadi Bin Mohamad

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Technology (Hons)
(Information Technology)

Approved by

____

(Mr. Low Tang Jung)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
DECEMBER 2004

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.

[illegible]

ABDUL HADI BIN MOHAMAD

ABSTRACT

An Intrusion detection system is generally considered to be any system designed to detect attempts compromise the integrity, confidentiality or availability of the protected network and associated computer systems. Intrusion Detection System (IDS) aims to detect attempted compromises by monitoring network traffic for indications that an attempted compromise is in progress, or an internal system is behaving in a manner which indicates it may already be compromised. A host based IDS (HIDS) monitors a single system for signs of compromise.

The vast majority of worms and other successful cyber attacks are made possible by vulnerabilities in a small number of common operating system services. Attackers are opportunistic. They take the easiest and most convenient route and exploit the best-known flaws with the most effective and widely available attack tools. They count on organizations not fixing the problems, and they often attack indiscriminately, scanning the Internet for any vulnerable systems. The easy and destructive spread of worms, such as Blaster, Slammer, and Code Red, can be traced directly to exploitation of unpatched vulnerabilities.

ACKNOWLEDGEMENT

Bismillah ar-Rahmani Ar-Raheem

In the Name of Allah, The Most Compassionate, the Most Merciful

First and foremost I would like to recite my greatest gratitude to the Most Merciful Allah for giving me the opportunity in completing this manuscript on time and without much hassle or problem. Without His observance in giving me the chance in finishing the report, there might be major problem which can resulted in delay of turning in the report in the time constrain.

In completing this preliminary report, there are some people that had been the backbone of the activities done in the complete of this text. I wouldn't have been able to finish up without their assistance, encouragement, and support either in terms of material, or spiritual. With this I would like to put some credit to them who has helped me through this time duration. They are as listed as beneath:

1. **Mr. Low Tan Jung** – my supervisor (for giving me the guidelines and ways in producing a good output and full support in terms of knowledge input along this project)

2. **Universiti Teknologi PETRONAS** – all UTP staff (for the full cooperation and providing me very convenient places to continue the project with the provided utilities)

3. **Parents and families** (for giving the full moral support in completing the report in addition to the consultation they have given during my troubled times)

4. **Friends** (as they have been there for me during the good and bad times as we stay together under the same varsity)

ABBREVIATION AND NOMENCLATURES

NIDS	:	Network Intrusion Detection System
GUI	:	Graphical User Interface
FTP	:	File Transfer Protocol
ICMP	:	Internet Control Message Protocol
SME	:	Small Medium Enterprise
SDLC	:	System Development Life Cycle
PC	:	Personal Computer
UDP	:	User Datagram Protocol
TCP	:	Transfer Control Protocol
IP	:	Internet Protocol
I/O	:	Input/Output
SUID	:	Stream Unique Identifier
HTTP	:	Hyper Text Transfer Protocol
CGI	:	Computer Graphic Interface
DOS	:	Disk Operating system
IRC	:	Internet Relay Chat
DNS	:	Domain Name Server
LAN	:	Local Area Network
WAN	:	Wide Area network

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 BACKGROUND OF STUDY

This report is written as a pre-requisite for undergraduate students to complete the studies. Intrusion Systems is chosen as my title for the final year project as it could help me to enhance and in-depth with all the theories I have learnt during my years in study.

There is a need for Intrusion Detection System in current local area networks to report security incidents and protect the network against intrusions from the Internet. The rates of intrusions and security incidents have increased dramatically in the last few years. CERT/CC, the Coordination Centre of the Computer Emergency Response Team at Carnegie-Mellon University, has recorded a doubling of both software vulnerabilities and reported security incidents every year since 1999. In the first nine months of this year, 1820 software vulnerabilities were reported to CERT. During the same period, 34754 incidents were reported (CERT 2001).

This study is so important as to add more research and knowledge to the most crucial network problems exist nowadays. These problems need serious attention. Those limitations exist needs serious R&D to enhance existing system. Leakage is never allowed in any Intrusion Detection System.

Although there a lot of Intrusion Detection Systems currently exist, this system still differs from others. Of course, the main characteristics or criteria of IDS still the same, differences is there. This systems developed for those organization or small network that needs a security system that is less expensive but still efficient. The simplicity of this system makes it easy to be implemented and cheaper to purchase. Furthermore, those small organizations will definitely needs a security system that could cater their needs of security.

1.2 PROBLEM STATEMENT

As computer systems and the Internet have grown in size, complexity and usage the demands placed upon those responsible for ensuring the continued operation and security of these systems has also grown. This has lead to a demand for automated systems for detecting malicious activity on both individual hosts and networks. In line with our capitalistic society users crave for a system that could really protect their network. Especially those institutions that need a high level of security etc financial institution. This has lead to the development of a range of Intrusion Detection Systems. Some of these systems are available as free open source applications, while others are offered as commercial products.

Figure 1 below shows the virus attacks around the world monitored by a group of researcher from their official website (www.dshield.org) at 11/8/2004.

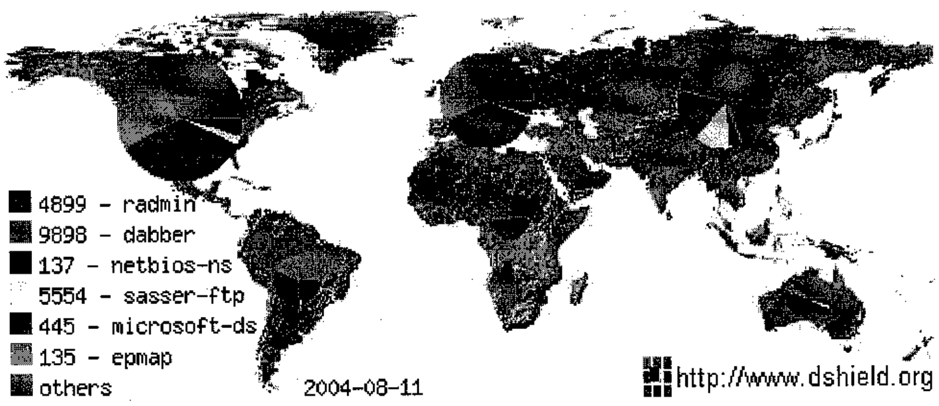
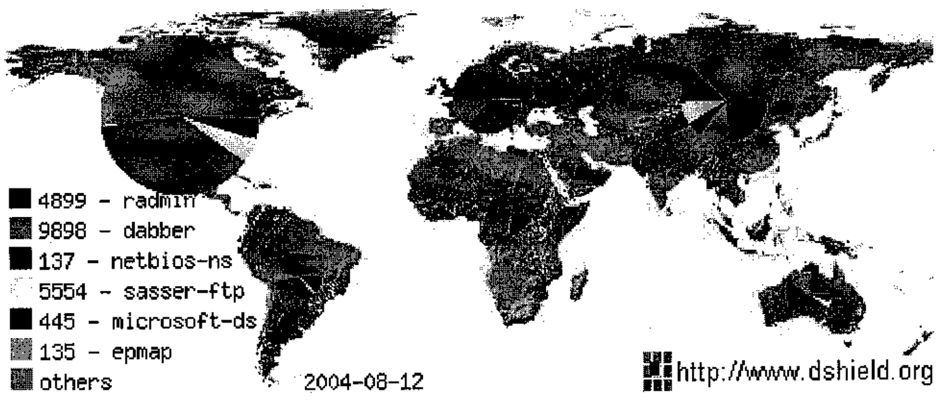


Figure 2 shows the same attack on the next day (12/8/3004).



From both figures shown above, we could see how fast and hazard these attacks could spread. From quite a tiny size of attack, it grew immensely to a huge number of attacks just in 24 hours period. And this needs very serious attention.

With the existing of current IDS technology, these kinds of attacks still increase day by day. Which in other words, what we could argue is, whether those systems is applicable or efficient as it should be? Does it cater the needs of security? Or is it just another system that adds cost to an organization. From here, we could see that there is a lot of room of improvements needed. Whether in the development of IDS or in the development of human understanding in malicious activity and IDS technology.

The other problem statement is that Firewall is just not sufficient. Thus, when installing a firewall, the first thing it does is stops ALL communication. The firewall administrator then carefully adds "rules" that allow specific types of traffic to go through the firewall. For example, a typical corporate firewall allowing access to the Internet would stop all UDP and ICMP datagram traffic, stops incoming TCP connections, but *allows* outgoing TCP connections. This stops all incoming connections from Internet hackers, but still allows internal users to connect in the outgoing direction.

A firewall is simply a fence around our network, with a couple of well chosen gates. A fence has no capability of detecting somebody trying to break in (such as digging a hole underneath it), nor does a fence know if somebody coming through the gate is allowed in. It simply restricts access to the designated points.

1.3 OBJECTIVE AND SCOPE OF STUDY

Because of the increasing occurrences of network intrusions, IDS technologies have developed further in the last few years. There have been a number of developments in the networking area, e.g. move to switched networks and emergence of new security protocols, like IPsec. Some of these technologies have made the protection of network traffic easier, but the use of encryption has also made it more difficult to detect patterns in the IP packets. Enhancements are being made into network protocols, e.g. ICMP Traceback message, and functionality is being added to network equipment, to turn network devices into 'active network' so that the intruders can be located and the intrusion contained.

The main objectives of this project are to develop a prototype of Intrusion Detection system that is efficient and needs less cost. As known, current IDS technology is expensive and needs a lot of training for staff before being implemented. This project will try to cater these problems.

In this research, I will focus on the development of an Intrusion Detection system using an open sourcecode. Visual Basic 6 is the major tool used to develop the system. Time frame of 6 months might be inadequate, everything have to be on schedule. Every sources and time given have to be fully utilized as this title would consume a lot of time in researching and developing the system.

The other project objective and scope tries to "detect" and "block" malicious activity in the network where the system is implemented. Detecting and blocking is the essential criteria of any IDS. As this is the major role of efficient IDS. Alert handling or legal act taken after detecting and blocking will be beyond the scope. This project tries to focus and research in detail to these two objectives and scope.

Currently SME companies do not generally have the expertise to install and run their own IDS system. An outsourced, fully managed solution may be applicable where a firewall system is not adequate. In the future networks are expected to develop into the same direction as present day 'active networks', where protocols and the infrastructure offer more protection from intrusions.

Early intrusion attacks and, consequently, intrusion detection expertise were developed in Unix environment. The Lawrence Livermore National Laboratories in the U.S. published a guide for Unix system administrators describing techniques and actions in case their systems were targeted in an intrusion attack or used to attack other host systems (Pienarczyk et al. 1994). Unix has traditionally been used in the universities, because of the free licenses from AT&T. A number of research projects in the last ten to fifteen years have concentrated on Unix based systems and related operating system research, as an example Kumar (1995).

An IDS also allows a company to efficiently manage its incident analysis resources by centralizing its attack records and by giving the analyst a quick and easy way to spot new trends and patterns and to identify threats to the network across multiple network segments. This report will also try to give the reader some insight into the benefits of running an IDS, from both incident analyst and corporate views.

CHAPTER 2

LITERATURE REVIEW

Research into intrusion detection up to four years ago has been summarised by Axelsson (1998). There has been a marked shift from host-based intrusion detection systems to network-based systems but the difficulty of tracking transactions in a fast network with encrypted traffic has lead into a hybrid system of IDS. There is a need for both policy based and anomaly based detection. Axelsson prefers non-realtime detection and accurate reporting of events that happened, rather than the IDS issuing an immediate but vague warning (1998, pp.12-13).

In his later research Axelsson (2000, pp. 8-9) classifies ID systems into three categories, based on the type of intrusion the system most easily detects:

- well known intrusions with a static pattern
- generalisable intrusions, exploiting flaws in the attacked system
- unknown intrusions

Axelsson has dictated a trend towards increasing security and interoperability of the IDS, and a trend towards distributed intrusion detection in larger environments. The number of intrusion detection systems capable of active response is increasing. Axelsson points out that security concerns about the intrusion detection system itself raise the possibility of the attacker exploiting an active response to effect a denial of service attack on the system being monitored, or an innocent third party' (Axelsson 2000, p. 13).

According to Amoroso (Intrusion Detection Systems (IDS) Part I - (network intrusions; attack symptoms; IDS tasks; and IDS architecture, 2003)

"Intrusion detection is a process of identifying and responding to malicious activity targeted at computing and networking resources". He mentioned Intrusion Detection System (abbreviated as IDS) is a defense system, which detects hostile activities in a network. The key is to detect and possibly prevent activities that may compromise system security, or a hacking attempt in progress including reconnaissance/data collection phases that involve for example, port scans. One key feature of intrusion detection systems is their ability to provide a view of unusual activity and issue alerts notifying administrators and/or block a suspected connection. He also stated that IDS tools must be capable of distinguishing between insider attacks originating from inside of an organization (coming from own employees or customers) and external ones (attacks and the thread posed by hackers).

Currently, there exist a lot of intrusion detection systems available to use. One of the most known systems is Snort. 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. Snort has three primary uses. It can be used as a straight packet sniffer like tcpdump(1), a packet logger (useful for network traffic debugging, etc), or as a full blown network intrusion detection system. Although Snort has all these capabilities, but still it needs long hour of training to really master the system. The training must be conducted by a train professional that have an expertise in the system.

CHAPTER 3

METHODOLOGY

3.1 WATERFALL MODEL

There are a lot of methodologies that I could choose. Each methodology has its own pro's and con's. For this project, I decided to use the Waterfall model which is best suited to this project. This methodology was selected as it suits most to this project. Other methodology has also been taken into consideration such as the Spiral model and RAD model. But seeing into complexity and cyclic process of the methodology makes me decided to choose this methodology. Coping to the time frame given, I need to have a simple methodology yet efficient and practical to use in completing the project. As mention earlier, the time given t complete the project is not sufficient, so everything must go on schedule and as smooth as possible.

The Waterfall Model is a software development model first proposed in 1970 by W.W.Royce, in which development is seen as flowing steadily through the phases of requirements analysis, design, implementation, testing (validation), and integration and maintenance.

Methodology plays a vital role in completing any project. Waterfall model is used as the methodology to plan and manage the system development process for this project. All the phases in the system development life cycle (SDLC) applies to this model in order to develop the project. The waterfall model consists of 4 important phases that I will undergo:

a. Planning
b. Analysis
c. Design
d. Implementation & Testing

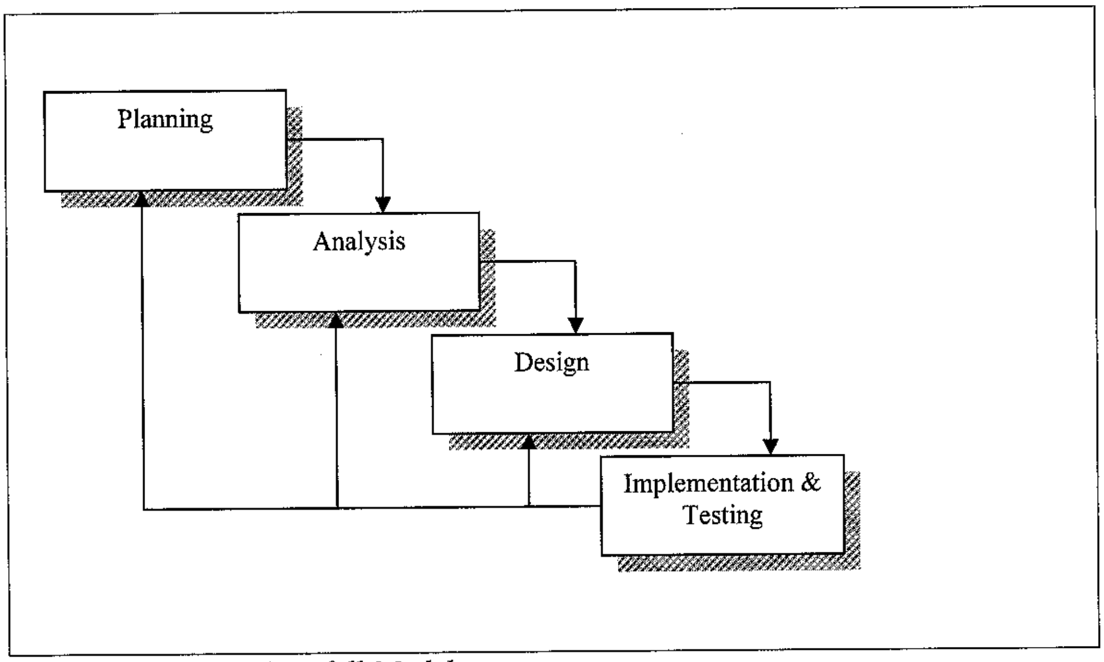


Figure 3 Waterfall Model

3.1.1 Planning

System planning begins with a formal proposal or request for the project. In this phase, the purpose is to identify clearly the nature and scope of the business opportunity or problem by performing preliminary investigation or also called as feasibility study. The outcome from this study is project scope. This preliminary investigation is a critical step since the outcome will affect the entire development process.

At this phase, the project started with the request from the lecturers to submit the project proposal. As discussed with the supervisor, this topic was selected since it is an interesting topic to discover. During this stage, proposal was sent to the FYP committee for approval. Scope of study was also established during this period.

3.1.2 Analysis

The purpose of this phase is to understand the requirements and build a logical model for the system. As implemented in this project, this is the phase of doing research and analysis. Facts, information, data, and findings were collected as much as possible during this stage.

Research on network intrusion and detection were done aggressively. Several projects regarding the topic were also analyzed in order to come out with a good product analysis. At this stage, the preliminary report was sent to the supervisor as required. The end product of this system analysis phase is the system requirement, which identify the design requirements for the project. The author has also analyzed a few existing IDS system. Most of them are developed using an open source. Eg. Snort. This system was developed using UNIX. From observation and analysis of the system, the author identified a few characteristics that need an improvement.

3.1.3 Design

In this phase, all necessary outputs, inputs, interfaces, and processes will be identified. The tools needed for the design phase are prepared and installed. Some of the tools are free downloaded from Internet such as Java Script. The development will start with installation of those software and hardware needed to run the program. Each software was initially analyzed for its compatibility with the system. The codes were thoroughly checked and debug in order to have a bug free code.

3.1.4 Implementation and Testing

During implementation phase, the system is constructed. The code are written, tested, and documented, and the system is installed. At this stage, the full system will be implemented.

Development of the system required most of time and effort. The development phase has to be implemented according to the plan designed earlier in the planning phase. But, some changes did occurred to suit the problems arised. Coding and debugging consume a lot of time. The code has to be debugged again and again.

Testing was done after the installation of the system. Testing phase is very important as it will show whether the systems is functioning as required. Testing was done a few times to avoid any missing data. The systems was re-tested a couple of time. Correction and enhancement implemented according to the test results.

Then, the system is delivered as required by the Academic Services committee. At this phase, the system will be presented to the examiners in order to check whether it meets the objectives and user expectations. The objective of this phase is to deliver a completely functioning and documented information system. If the system does not meet the requirement and expectation, it has to be enhanced again. During this phase, the system evaluation will be conducted to determine whether the system operates properly and if costs and benefits are within expectations.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 TRADITIONAL NETWORK SECURITY MODEL

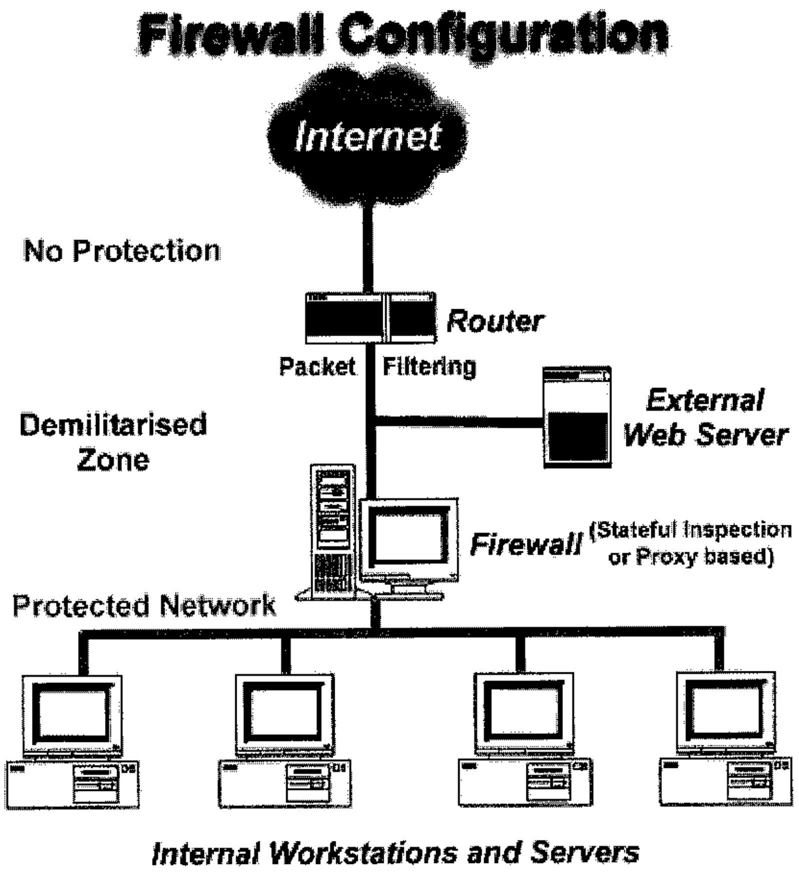


Figure 4 Traditional Network Security Model

In the diagram we see a typical firewall configuration. Packets from the Internet are filtered to allow connection only to the firewall (and if present the external web server). It is at this point or gateway that all security checking is aimed, access lists allowing only certain hosts out (and usually none in) prevents unauthorized access to our internal network. The granularity of these access lists is where all the expertise is in building good firewalls e.g. "allow only web requests from the sales department PC's out to our suppliers' web host between 9am and 5pm". We could see that no Intrusion Detection system is applied to the system.

4.2 THE NEED OF INTRUSION DETECTION AFTER NETWORK SECURITY

A common misunderstanding of network security is that firewalls recognize attacks and block them. This is not true.

Firewalls are simply a device that shuts off everything, then turns back on only a few well-chosen items. In a perfect world, systems would already be "locked down" and secure, and firewalls would be unneeded. The reason we have firewalls is precisely because security holes are left open accidentally.

In summary, a firewall is not the dynamic defensive system that users imagine it to be. In contrast, an IDS *is* much more of that dynamic system. An IDS *does* recognize attacks against the network that firewalls are unable to see.

For example, in April of 1999, many sites were hacked via a bug in ColdFusion. These sites all had firewalls that restricted access only to the web server at port 80. However, it was the web server that was hacked. Thus, the firewall provided no defense. On the other hand, an intrusion detection system would have discovered the attack, because it matched the signature configured in the system.

Another problem with firewalls is that they are only at the boundary to our network. Roughly 80% of all financial losses due to hacking come from inside the network. A firewall at the perimeter of the network sees nothing going on inside;

it only sees that traffic which passes between the internal network and the Internet.

Some reasons for adding IDS to firewall are:

- Double-checks misconfigured firewalls.
- Catches attacks that firewalls legitimate allow through (such as attacks against web servers).
- Catches attempts that fail.
- Catches insider hacking.

Hackers are much more capable than we think; the more defenses we have, the better. And they still won't protect us from the determined hacker. They will, however, raise the bar on determination needed by the hackers.

4.3 MAIN TASKS OF INTRUSION DETECTION SYSTEM

The main task of intrusion detection systems is defense of a computer system by detecting an attack and possibly repelling it. Detecting hostile attacks depends on the number and type of appropriate actions. Intrusion prevention requires a well-selected combination of "baiting and trapping" aimed at both investigations of threats. Diverting the intruder's attention from protected resources is another task. Both the real system and a possible trap system are constantly monitored. Data generated by intrusion detection systems is carefully examined (this is the main task of each IDS) for detection of possible attacks (intrusions).

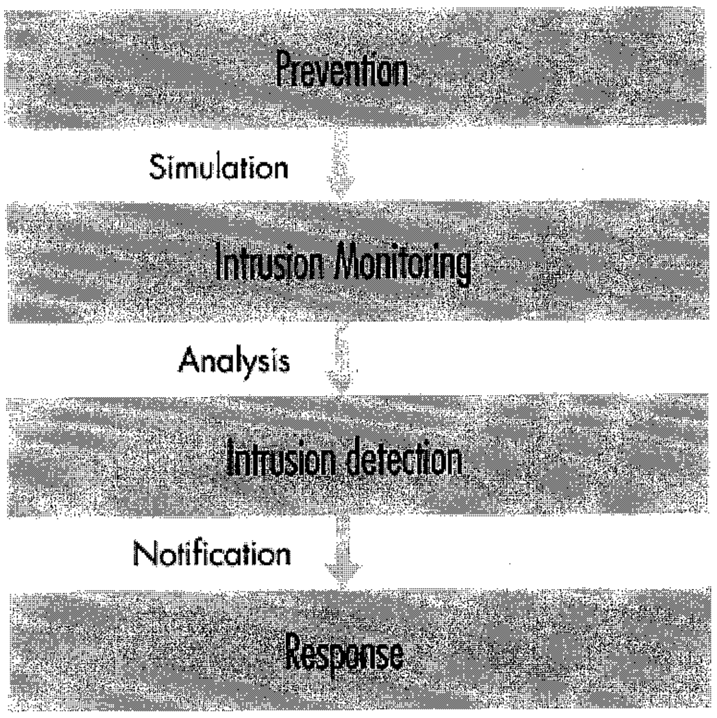


Figure 5. Intrusion detection system activities

Once an intrusion has been detected, IDS issues alerts notifying administrators of this fact. The next step is undertaken either by the administrators or the IDS itself, by taking advantage of additional countermeasures (specific block functions to terminate sessions, backup systems, routing connections to a system trap, legal infrastructure etc.) – following the organization's security policy . An IDS is an element of the security policy.

Among various IDS tasks, intruder identification is one of the fundamental ones. It can be useful in the forensic research of incidents and installing appropriate patches to enable the detection of future attack attempts targeted on specific persons or resources.

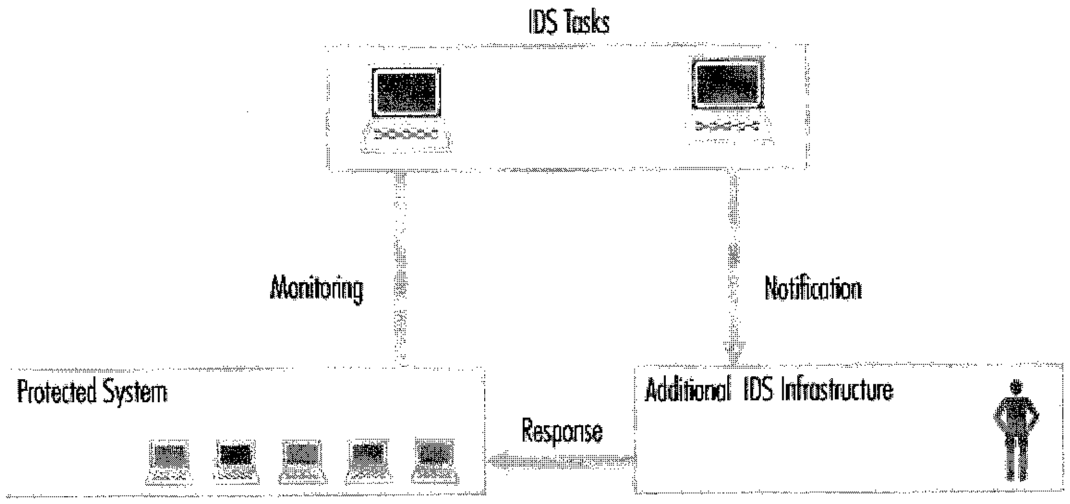


Figure 6. Intrusion detection system infrastructure

4.4 CLASSES OF ATTACK

Categorizing Attacks in Two Dimensions

	Point of Origin	
	Internal user	External User
Denial of Service	Annoying	Annoying
Increased privilege	Moderately serious	Serious risk
Superuser privilege	Very serious	Disaster

Table above provides a convenient way of looking at attack categories. We can see that threats generally are divided between *interna*l and *external* points of origin. Inside the table are relative indications of the seriousness of the consequences. If an internal user obtains privileges belonging to other users, we usually can rectify the situation and perhaps take legal action. When someone outside our network is able to gain super user access into one of our nodes, we have a catastrophic breakdown in security somewhere. Also, because so many ways to hide one's identity from the outside exist, the chances of catching the intruder are slim.

4.4.1 Internal Threats

Statistics from the FBI Crime Lab consistently show that the majority of computer crime occurs from inside. True, as more people connect to the internet, he threats from outside increase. Today, most crimes still are committed by insiders. Or at least outside criminals assisted by insiders. The thefts of millions of dollars from a major U.S banks was launched from Russia, but collision from insider makes the task easier. Although some companies do not like to think of their employees, contractors, or business partners as potential criminals, historical data encourages them to do so. Below are some of the threats that an insider poses to internal systems.

a) Internal Denial-of-Service Attack

Recently, a number of NT systems at the University of Texas were hounded by a denial-of-service attack against the IP stack delivered with NT. The attack was a variant of the Teardrop UDP attack that was possible because of a bug in NT. By sending certain type of UDP datagrams, an adversary could cause the systems to crash. Because UDP packets are often blocked by screening routers or firewalls, this threat was unlikely from outside sources. Someone with access to one of the UT labs launched the attack internally.

Users with accounts on various company servers or on university systems pose threats because they already have access to the system. When they are able to establish a login session on a computer, a number of denial-of-service attacks rare possible:

- Consume all of the disk space in the /tmp directory of UNIX systems to slow or crash the system (depending on how that particular version of UNIX handles this condition)
- Write a program to consume all available resources such as all of the memory buffers allocated for sockets
- Fill up the printer queue directory
- Create a number of concurrent I/O bound processes that thrash the disk repeatedly

They really don't need an account on a system to cause problems. Physical or network access is sufficient for locking all accounts with failed login attempts until the lockout threshold is hit for each account. If the systems permit remote logins from other nodes inside the enterprise, failed login attacks are possible even when physical access is not granted.

Most environments run a large number of client-server applications. The telnet is a well known example. However, numerous proprietary client-server protocols are running throughout the enterprise, and each of these are also susceptible to denial-of-service attacks. For example, it is unlikely that many legacy applications are performing adequate authentication of packets received. Forged IP addresses and packets can find their way into listening servers and cause denial-of-service attacks. If the servers are design to accept connections from any internal node, it's easy to create packets, flood the server with them and thus render the server useless. In general, the closer they can get to running on the systems directly, the more damage they can potentially do.

b) Internal Privilege Escalation

UNIX and NT systems both provide ways for users to gain increased privileges through program execution. NT uses its access rights mechanism, and UNIX relies on the now familiar SUID or SGID concepts. Even if the privileged program does not give the users access to everything on the system, even a little privilege boost can help. For one thing, if the average UNIX user can gain privileges of the mail group by exploiting a SGID mail program, then that user will have access to the mail spool directory. Denial-of-service attacks or worse are then possible. Privileged programs are compromised in a number of ways:

- The program does not check buffer limits and is subject to a buffer overflow attack
- The program does not check input parameters and is tricked into executing one of the parameters as a command
- The program makes invalid assumptions about its environment
- The program is tricked into operating on a different resource because of poor programming practices.

c) Internal Superuser Privileges

The biggest threat to a system is when a user gains superuser or complete administrative privileges. The same kind of attacks and problems mentioned previously apply for root or administrator privileged programs. Buffer overflow attacks, data driven attacks, spoofed resources, and spoof network packets have all been exploited by normal users to gain privileged access to the systems.

Will a fire wall prevent these privileges escalations from happening? If the network attack is like the test.cgi attack and the Web sever is running as root or administrator, then the firewall will not help.

4.4.2 External Threats

When we have a publicly visible systems, as almost everyone does today, there is always a threat that someone can find a away into our systems. A system in the perimeter network is always the first one to be hit. When someone attacks our systems, the result could be denial of service. For example, our web server could be slowed considerably if it is hit with a denial-of-service attack. If someone manages to gain a login shell as a normal user, this represents the next level of severity in threats. Naturally, if someone obtains complete control over a system by gaining root or super user privileges, and this adversary is a remote unaffiliated with our enterprise, this represents the worst threat.

a) External Denial-of-Service Threats

Publicly visible network addresses are nearly impossible to defend from all denial-*of-service* attacks. If our web server allows arbitrary users to connect, someone can write a program to generate a large number of http transactions with our server as the target. The net result is a flooded web server. Most web server is not design to detect or defend against these attacks, although this is precisely the only place to adequately defend against such a threat.

A firewall or screening router is also not going to be of much help here because it is difficult to state the packet filtering rule for this condition. For example, a large number of http packets with bad data from a single source address are hard to distinguish from a large number of well-formed http packets unless our filter is smart enough to know the details of the http protocol and partially assembled packets. To really solve the problem, the component that has the highest semantic view of the packets, in other words the web server itself, must implement this form of application-level security. If the web server detects a series of bogus packets or even good packets from the same address in a fixed interval of time, the server could notify the firewall to block incoming traffic from that address. Of course, the clever denial-of-service attacker would just forge a series of IP addresses to avoid detection.

b) External Privilege Escalations

- This class of attacks is becoming less frequent as knowledge of security problems spreads. A remote user can escalate privileges in two different ways:
- A program that does not permit logins is running on the target node but is accepting network connections (such as web server).
- The remote user is able to gain access to the systems via a login, or in other words, a network program is listening for external connections.

An example of the former is, the web server daemon. Poor CGI programming practices can permit remote users to execute arbitrary command on the system, albeit only with the privileges of the web server daemon. A rather worse example surfaced in 1997 with some implementation of FTP.

An FTP client can issue a command to the FTP server that requests multiple files at once. The client issuing the *mget** command is asking the server to send all files in the current directory of the server. Unfortunately, some FTP client implementations did not bother to check that the files sent by the server were only those included in the current directory. A user in the home directory who then FTP to a malicious server and executes the *mget** command could find many other files being added to the home directory. The server could push viruses or Trojan Horses to the receiving clients because of this bug.

4.5 UNDERSTANDING MALICIOUS ACTIVITIES

Malicious traffic ranges from simple probing and scanning to denial-of-service. Some types of attack are quite easily distinguishable, while others can be quite obscure and hard to differentiate from legitimate traffic.

4.5.1 Scanning

Traffic design to map our network and find security weaknesses is by far the most common type of malicious activity. This traffic, called *scanning,* takes several form: network scanning, port scanning, ping scanning, vulnerability scanning, and etc.all of the different types of scanning can be grouped into two broad types based upon purpose: "**network scanning"** and "**port scanning".**

The purpose of scanning is to determine what systems or services our system is running. Scans can be broad, with the intent of mapping out our entire network and services, or they can be specifically targeted to determine whether any of our computer systems are susceptible to a particular vulnerability.

4.5.2 Network Scanning

Network scanning generally seeks to determine several things: what IP addresses are actively used, what services are running on those active systems and what specific versions of the services are running. To determine what IP addresses are in use, attackers use one of the two techniques --- ICMP Echo Request and TCP/UDP (User datagram Protocol) connections.

Network scanning cannot be effectively prevented, but it can be hindered somewhat. ICMP (Internet Control message Protocol) is used for most types of network scanning to speed up the determination of whether the target host should be scanned in more detail. Blocking the ICMP packets will slow down some scanning. ICMP blocking is usually accomplished by filtering ICMP packets at either the border router or the firewall. By blocking ICMP, we force the remote user to perform a blind connect. A *blind connect* is simply an attempted TCP connection used to determine whether the host is active. Forcing a remote system to use blind connects significantly slows down the scanning process.

4.5.3 Port Scanning

An attacker is rarely interested in just a list of the systems we are running. Port scanning is the process of attempting to connect to ports in order to see if a device is running. Most popular port scanners attempt to make detection more difficult by randomizing the order of the ports tested for. Opening ports in a random order was sufficient to fool many of the earliest IDS software programs. This randomization doesn't affect many intrusion detection programs anymore because this randomization technique is well known.

One other way an attacker does is by manipulating the SOCKS port on user's machine. SOCKS is a system that allows multiple machines to share a common Internet connection. The reason that attackers scan for this is because a large percentage of users misconfigure SOCKS.

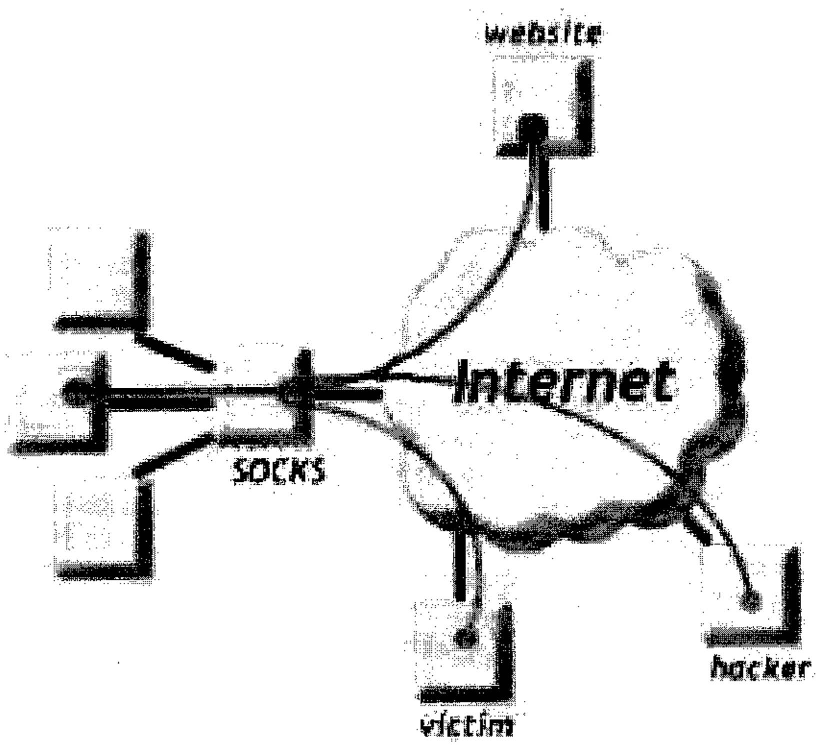


Figure 7

Many products support SOCKS. A typical product for home users is WinGate, which is installed on a single machine that actually connects to the Internet. All the other machines within the home connect to the Internet through the machine running WinGate.

A misconfigured SOCKS permits arbitrary the sources and destinations. Just as it allows internal machines access to the Internet, it will allow external machines to access the internal home network. Most importantly, it may allow a attacker access to other Internet machines through your system. This allows the attacker to hide his/her true location.

IRC chat servers will often scan clients for open SOCKS servers. They will kick off such people with a message indicating how to fix the problem. If you receive such a message, then you can check the client to see if it is a WinGate bot performing such a check. A false-positive may occur if an application is temporarily unavailable. In this case, it will look like your internal machines are "attacking" the SOCKS server.

4.5.4 Brute Force

Probably, the most easily detected type of attack, and the most inelegant, is a brute force attack. *Brute Force attacks* are commonly used in an attempt to discover username and password combinations for exposed services. Brute force attacks usually cause a noticeable increase in traffic and a high number of connections.

Brute force attack is a good example of the type of activity that can occur legitimately, as well as in attack. A legitimate user trying several passwords because he or she forgot her password would look very similar. The biggest differentiator in most cases is the volume. A true attacker will likely try hundreds of passwords, whereas a legitimate user will usually try only a handful.

4.5.5 Buffer Overflows

In basic terms, buffer overflow works by sending data that exceed the amount of space provided in memory for the data. The extra data overflows into other memory areas of the computer system. If the data being sent is carefully crafted by an attacker, the data can be use to effectively 'patch' a running system with the attacker's desired program code. Overflows are generally used to effectively either to create denial-of-service or to inject external code into the systems. The additional code usually provides a backdoor for the attacker to enter the system.

4.5.6 Applications Attacks

Application attacks, as the name implies, targeted at specific applications. An attack against a web server will connect to the web service and use the application protocol (in case of web, HTTP) to perpetrate the attack.

Each specific application attack is unique in nature and thus requires an equally specific detector. General detectors for application attacks are difficult to construct. Specific detectors for each attack are usually straightforward to create.

4.5.7 Denial of Service

Denial-of-service attacks are designed to interfere with legitimate user's ability to use their computer systems and services. Denial-of-service usually takes one of two broad forms. They either use *flooding* to exceed capacity, or they *exploit a bug* of some sort to crash specific services.

From technical perspective, flooding attacks are easiest to execute. A typical flood might use several systems with connections to high bandwidth to inundate a target host with millions of packets. Successful flooding requires access to enough systems and bandwidth to exceed the capacity of the target. This becomes more difficult to arrange as the target's capacity increases.

Denial-of-service attacks that exploit bugs are a little more difficult to construct, but require only single system with reasonable connectivity to execute the attack. Once the tool for exploiting particular bugs has been written, the attack software usually propagates quickly among attackers. It is only a matter of time before a program for executing particular DOS attacks is readily available.

Both flooding and denial-of-service attacks produce patterns that can be readily detected on the network. The same detectors that detect port scans can be used to detect flooding attacks. Bug-based denial-of-service attacks are typically unique and require their own signature.

4.5.8 Disinformation Attacks

This is one of the sneakiest uses of redirection. Domain name System (DNS) spoofing is an example of such an attack. Many DNS server do not authenticate DNS resolutions responses. Any DNS query response sent to a vulnerable server is cached as if the server had requested the information. As an example, if I want to redirect mail between Company A and Company B to my own mail server, Company C, I can send a spoofed MX address with my own mail server IP address (C) to Company A's DNS server. Any time Company A sends an email to Company B, the email will be redirected to my mail server. My mail server can then be configured to forward a copy of the mail to the real Company B mail server. This attack is perpetrated using a single spoofed UDP DNS query reply packet.

4.5.9 Worms

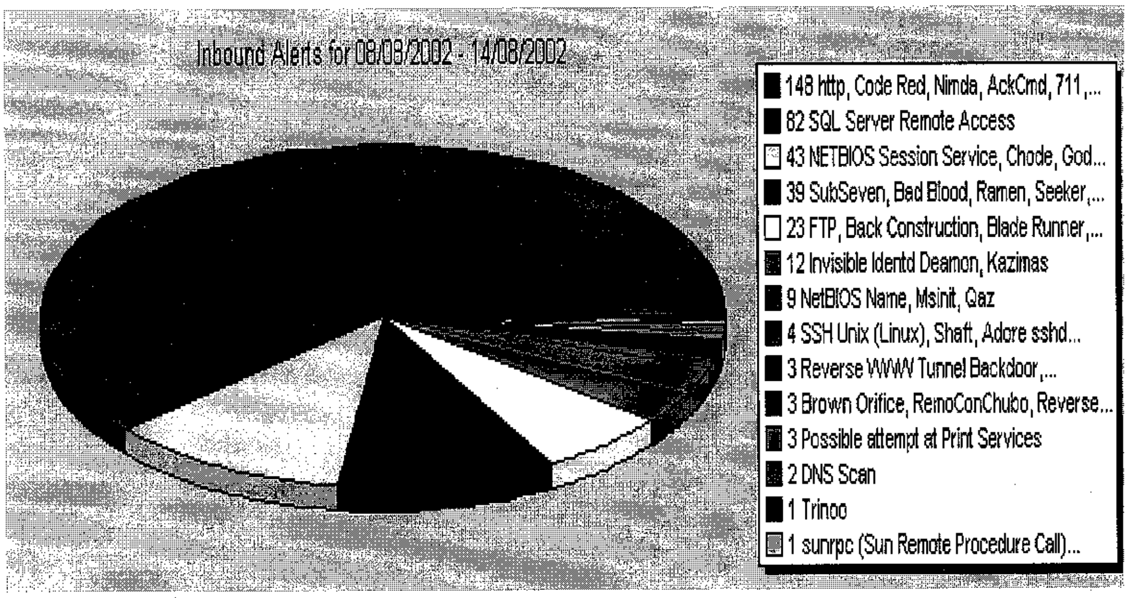


Figure 8

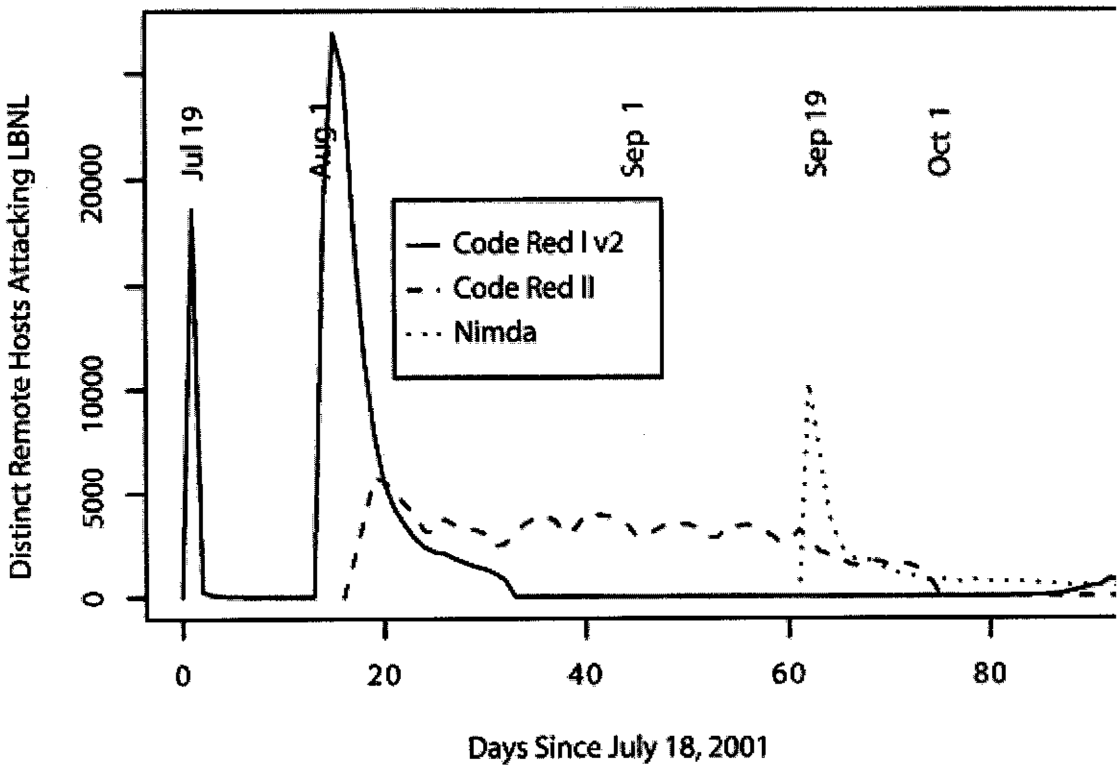


Figure 9. Onset of Code Red I v2, Code Red II, and Nimda: Number of remote hosts launching confirmed attacks corresponding to different worms, as seen at the Lawrence Berkeley National Laboratory. Hosts are detected by the distinct URLs they attempt to retrieve, corresponding to the IIS exploits and attack strings. Since Nimda spreads by multiple vectors, the counts shown for it may be an underestimate.

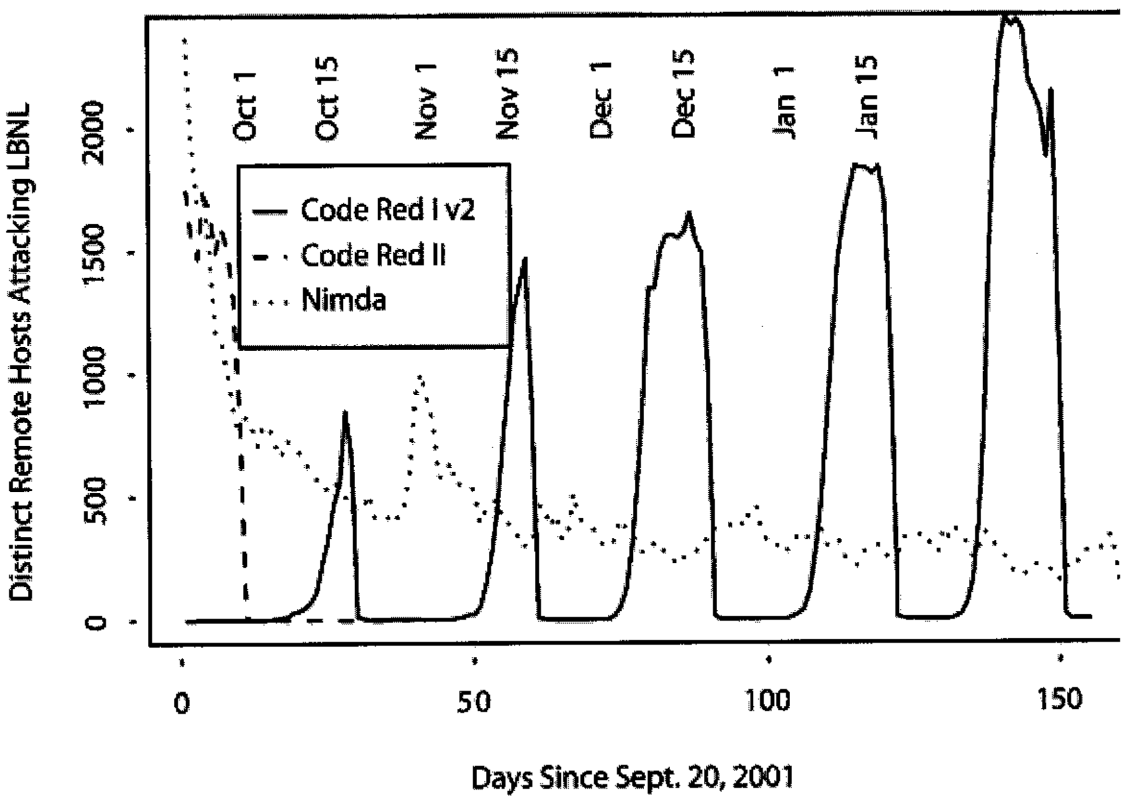


Figure 10 . The endemic nature of Internet worms: Number of remote hosts launching confirmed attacks corresponding to different worms, as seen at the Lawrence Berkeley National Laboratory, over several months since their onset. Since July, 139,000 different remote Code Red I hosts have been confirmed attacking LBNL; 125,000 different Code Red II hosts; and 63,000 Nimda hosts. Of these, 20,000 were observed to be infected with two different worms, and 1,000 with all three worms. (Again, Nimda is potentially an underestimate because we are only counting those launching Web attacks.)

For purposes of attack classification, worms such as Nimda and Code Red, or the more recent SQLSnake, can be best thought of as automated combination attacks. These worms start by scanning for vulnerable hosts and then proceed to use an application attack to compromise the system. After the system is compromised, the worms transfer a copy of their code into the target system. This newly compromised computer then begins the cycle over by starting to scan for more vulnerable hosts to propagate.

Worms such as Nimda and Code Red presents two issues for intrusion detection. During the initial few days of the worm propagation, the number of alerts generated is staggering. We might see in excess of tenfold our normal alerts, often as much as 100-fold. The second problem for IDS in relation to these worms is that the alerts never completely go away. After the initial few weeks, when the majority of sites have patched their systems to stop the spread, we will still receive a steady background flow of alerts. This continual flow of alerts is due to the fact that it is almost impossible for worms like these to completely die out. These worms are designed to infect default installations of software and operating systems. Given the size and diversity of the Internet, there are always a smattering vulnerable system attached and functioning at any given time. Couple of existence of vulnerable systems with a lack of appropriate technical expertise at far too many companies attached to the internet and the result is an inability to rid the Internet of worms once they are loose.

4.6 DEPLOYMENT OF INTRUSION DETECTION SYSTEM

4.6.1 Network Hosts

Even though network intrusion detection systems have traditionally been used as probes, they can also be placed on hosts (in non-promiscuous mode). Take for example a switched network where an employee is on the same switch as the CEO, who runs Win98. The windows machine is completely defenseless, and has no logging capabilities that could be fed to a traditional host-based intrusion detection system. The employee could run a network-based password cracker for months without fear of being caught. A NIDS installed like virus scanning software is the most effective way to detect such intrusions.

4.6.2 Network Perimeter

IDS is most effective on the network perimeter, such as on both sides of the **firewall**, near the **dial-up** server, and on links to **partner** networks. These links tend to be low-bandwidth (T1 speeds) such that an IDS can keep up with the traffic.

4.6.3 WAN Backbone

Another high-value point is the corporate WAN backbone. A frequent problem is hacking from "outlying" areas to the main corporate network. Since WAN links tend to be low bandwidth, IDS systems can keep up.

4.6.4 Server Farms

Serves are often placed on their own network, connected to switches. The problem these servers have, though, is that IDS systems cannot keep up with high-volume traffic. For extremely important servers, you may be able to install dedicate IDS systems that monitor just the individual server's link. Also, application servers tend to have lower traffic than file servers, so they are better targets for IDS systems.

4.6.5 LAN Backbones

IDS systems are impractical for LAN backbones, because of their high traffic requirements. Some vendors are incorporating IDS detection into switches. A full IDS system that must reassemble packets is unlikely to keep up. A scaled-down system that detects simpler attacks but can keep up is likely to be a better choice.

4.7 RESPONSE ACTIVITIES

After detecting any of the malicious attacks on our network, now we come to the phase where we have to response to the situation. The principal elements of response may include notification, blocking, isolation, or resort to law enforcement.

4.7.1 Notification

Most response plans stipulate notification steps for informing the affected systems and their administrators and users. A primary purpose of notification is to allow the affected individual to respond and recover from the attack as quickly as possible. Recovery is aided significantly by obtaining as much detail as possible about everything that occurred during the attack.

4.7.2 Blocking

One broad response to an attack is block traffic from the attack source. Blocking is always accomplished through a rule in the firewall. At a simple level, we should consider placing at least one IDS sensor outside the blocking mechanism. This gives us the ability to continue tracking the activities of the attacker. If the attacker has compromised additional systems in our network, for example, the attacker might attempt to communicate with those systems when the connectivity with the target of the attack is lost. The action taken by the attacker after implementing the blocking can provide valuable information to assisting the investigation of the detected security incident and recovery of any affected systems.

4.7.3 Isolation

If we want to allow the attacker to continue their activities for purposes of study, or gathering evidence for legal prosecution, we have to isolate the activity as much as possible to prevent harm to other systems. An attacker can cause additional harm to our systems as well as other systems on the Internet if not contained. As in the case of notification, additional information gathering mechanisms potentially provided by the IDS systems can significantly enhance this isolation process.

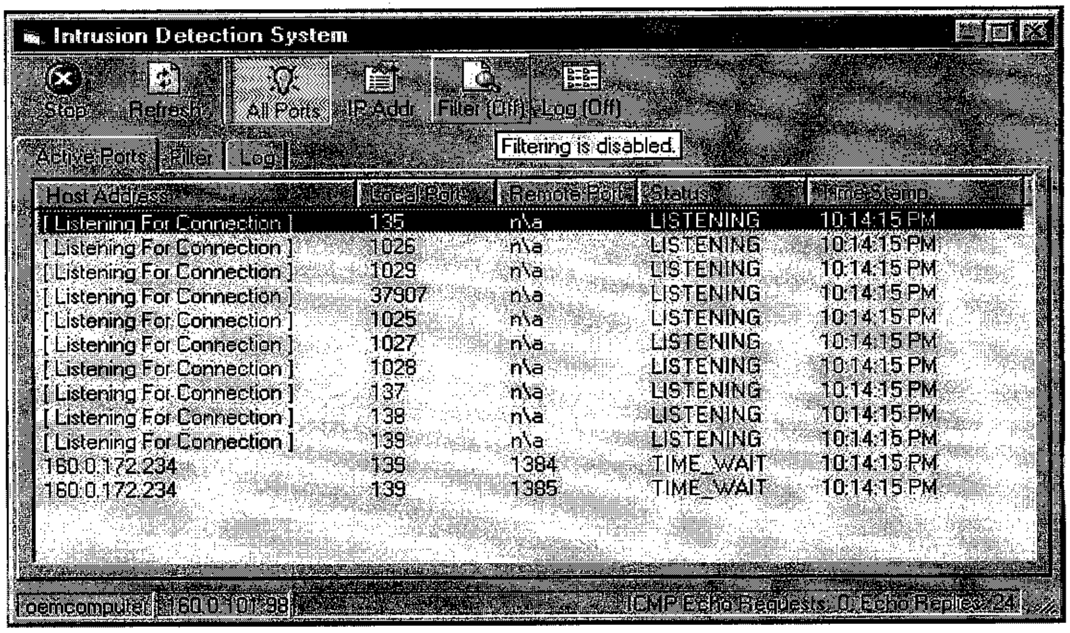
4.7.4 Law enforcement/Internet body communication

For some attacks, our response plan may indicate that we are to provide law enforcement or Internet groups such as Computer Emergency Response Team (CERT) with the information gathered by our IDS systems. If law enforcement involved, the data must be handled in a specific manner in order to be useful as legal evidence. The details of handling requirements vary somewhat from state to state. In most cases, the actual system drives are usually used as the primary source of evidence rather than primary logs, although this may partly due to the lack of widely use IDS as mush as anything else. We should consult legal counsel to insure our procedures meet with local legal requirements.

There are several organizations with which we could consider sharing our attack data. The internet is a global network. The attack being used against our network is almost definitely being use against others as well. By sharing information, we can assist proactively finding and preventing attacks. Several groups have been set up to take attack information and make it accessible to others to help prevent attacks.

DShield.org (www.dshield.org) collects firewall and IDS systems logs and uses them to analyze what sort of attacks are occurring and where they are coming from. DShield.org will even contact the sources of malicious activity on our behalf if we wish it to. DShield.org provides an updated “top 20” source of attacks that can be used to provide an ongoing blocking of the current worst abusers of security. It also provides several tools and clients for making submission of log information quick and easy.

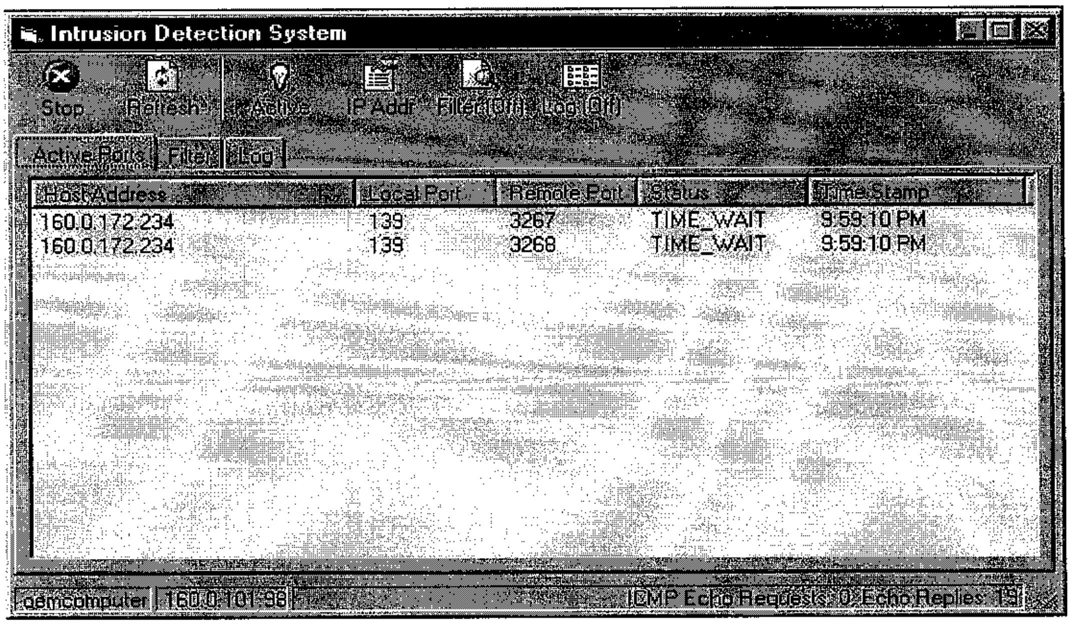
4.8 SNAPSHOT OF WORKING PRODUCT



Host Address	Local Port	Remote Port	Status	Time Stamp
[Listening For Connection]	135	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	1026	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	1029	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	37907	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	1025	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	1027	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	1028	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	137	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	138	n\a	LISTENING	10:14:15 PM
[Listening For Connection]	139	n\a	LISTENING	10:14:15 PM
160.0.172.234	139	1384	TIME_WAIT	10:14:15 PM
160.0.172.234	139	1385	TIME_WAIT	10:14:15 PM

oemcomputer | 160.0.101.98 | ICMP Echo Requests: 0, Echo Replies: 24

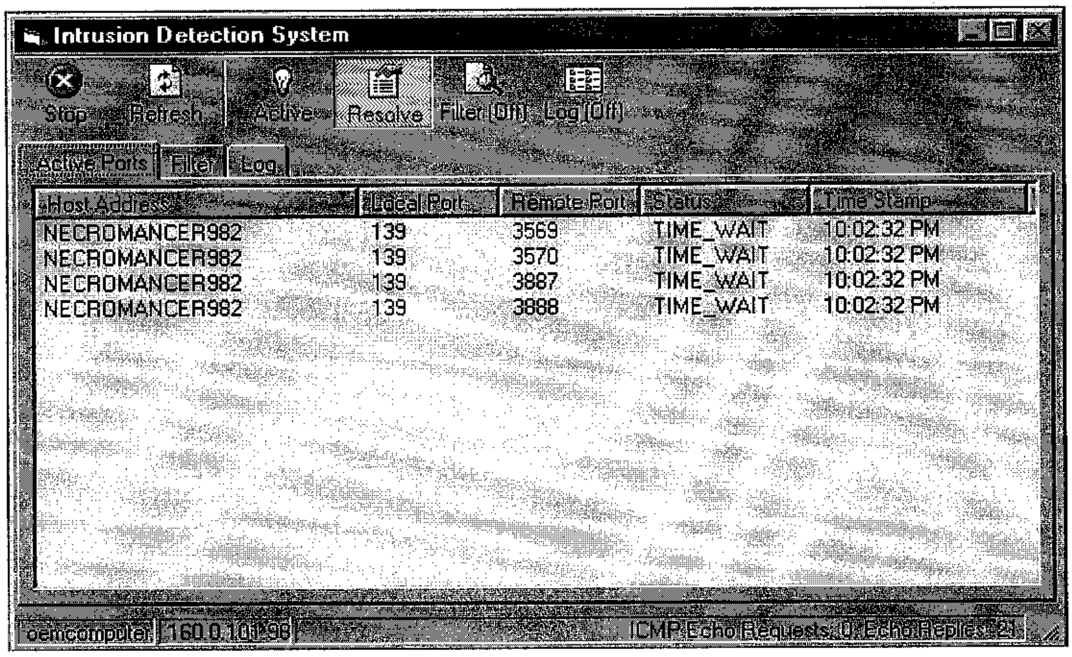
Figure 11. Main page showing the IDS is detecting all connection that is currently running in the network.



Host Address	Local Port	Remote Port	Status	Time Stamp
160.0.172.234	139	3267	TIME_WAIT	9:59:10 PM
160.0.172.234	139	3268	TIME_WAIT	9:59:10 PM

oemcomputer | 160.0.101.98 | ICMP Echo Requests: 0, Echo Replies: 19

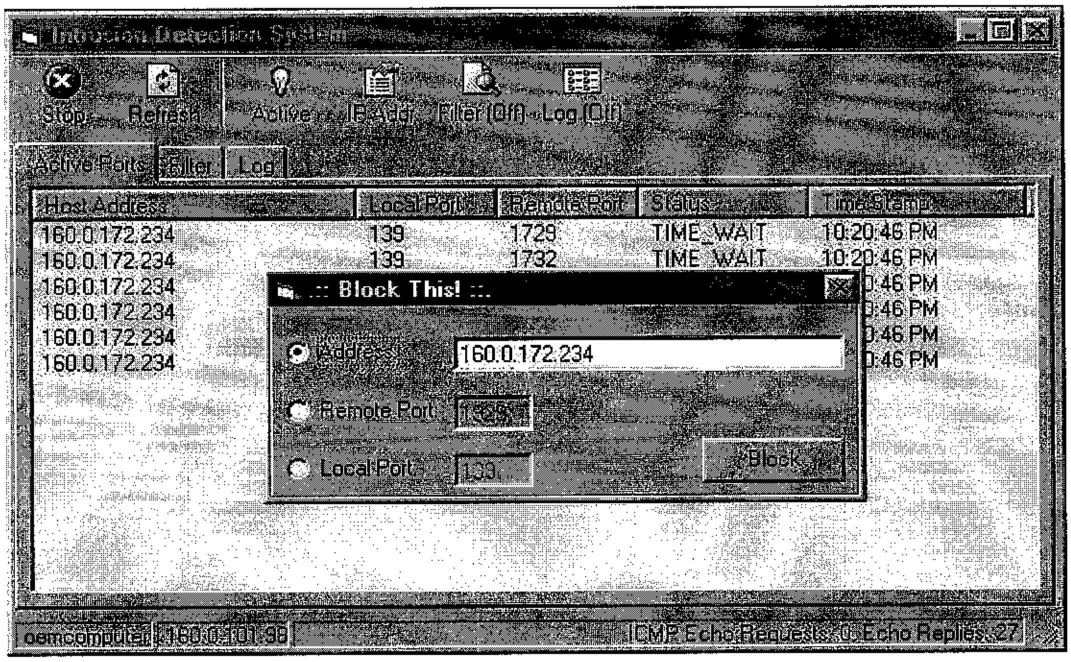
Figure 12. Main page showing only an active connection detected by the IDS.



Host Address	Local Port	Remote Port	Status	Time Stamp
NECROMANCER982	139	3569	TIME_WAIT	10:02:32 PM
NECROMANCER982	139	3570	TIME_WAIT	10:02:32 PM
NECROMANCER982	139	3887	TIME_WAIT	10:02:32 PM
NECROMANCER982	139	3888	TIME_WAIT	10:02:32 PM

```
oemcomputer | 160.0.101.98 | ICMP Echo Requests: 0, Echo Replies: 21
```

Figure 13. The Host name of the active IP address detected will be shown after the resolve button is clicked.



Host Address	Local Port	Remote Port	Status	Time Stamp
160.0.172.234	139	1729	TIME_WAIT	10:20:46 PM
160.0.172.234	139	1732	TIME_WAIT	10:20:46 PM
160.0.172.234				0:46 PM
160.0.172.234				0:46 PM
160.0.172.234				0:46 PM
160.0.172.234				0:46 PM

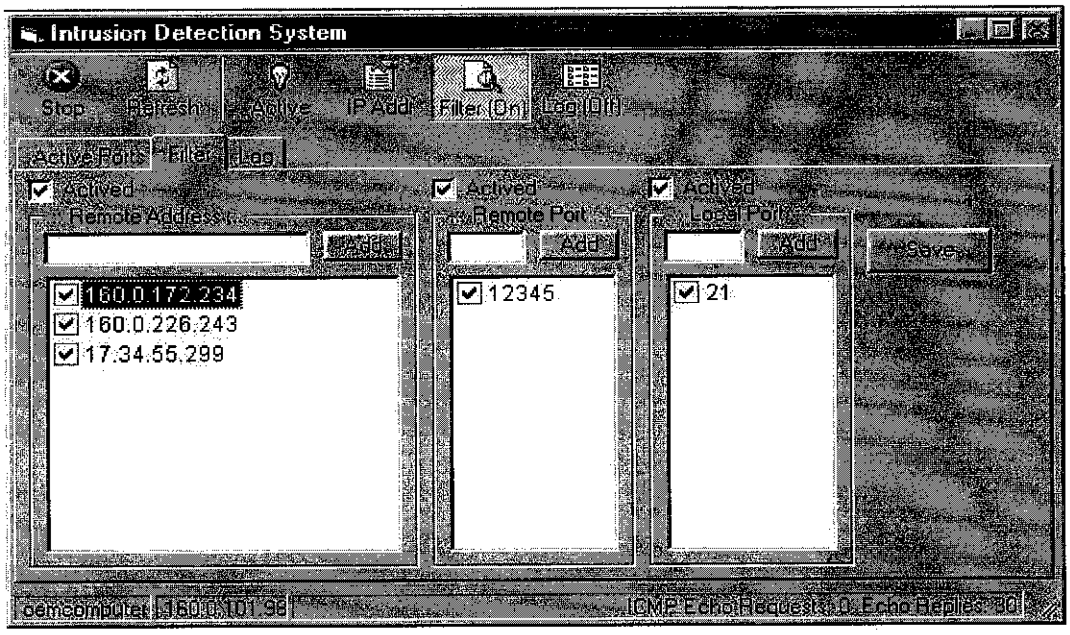
[✓ Address] 160.0.172.234

Remote Port [illegible]

Local Port 139

```
oemcomputer | 160.0.101.98 | ICMP Echo Requests: 0, Echo Replies: 27
```

Figure 14. IP address/hostname detected that tries to flood the network will be block by the systems administrator.



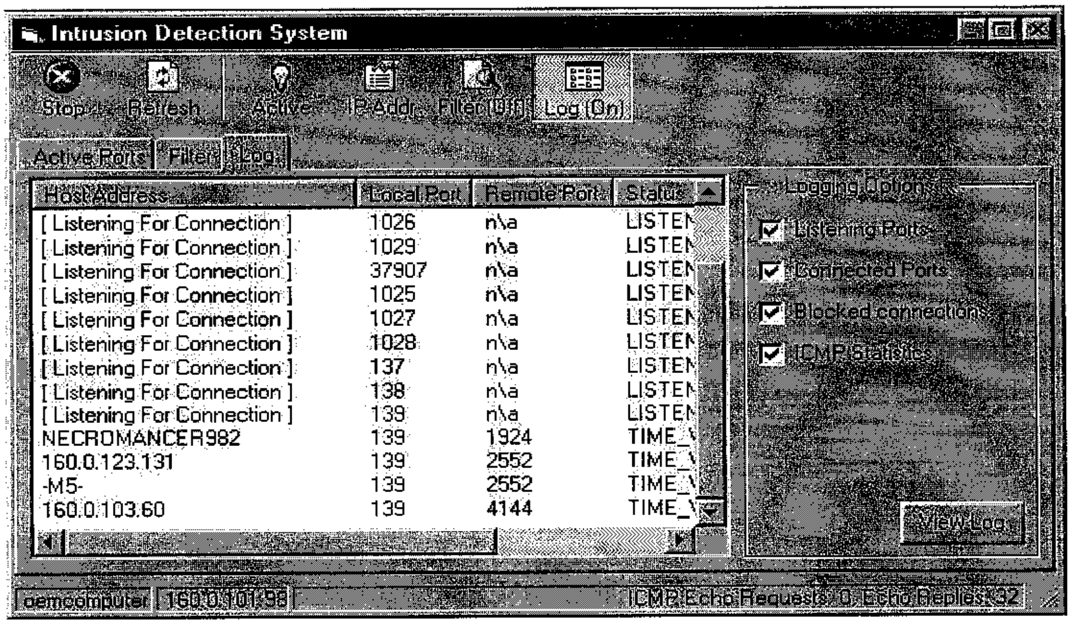
☑ 160.0.172.234
☑ 160.0.226.243
☑ 17.34.55.299

☑ 12345

☑ 21

oemcomputer | 160.0.101.98 | ICMP Echo Requests: 0, Echo Replies: 30

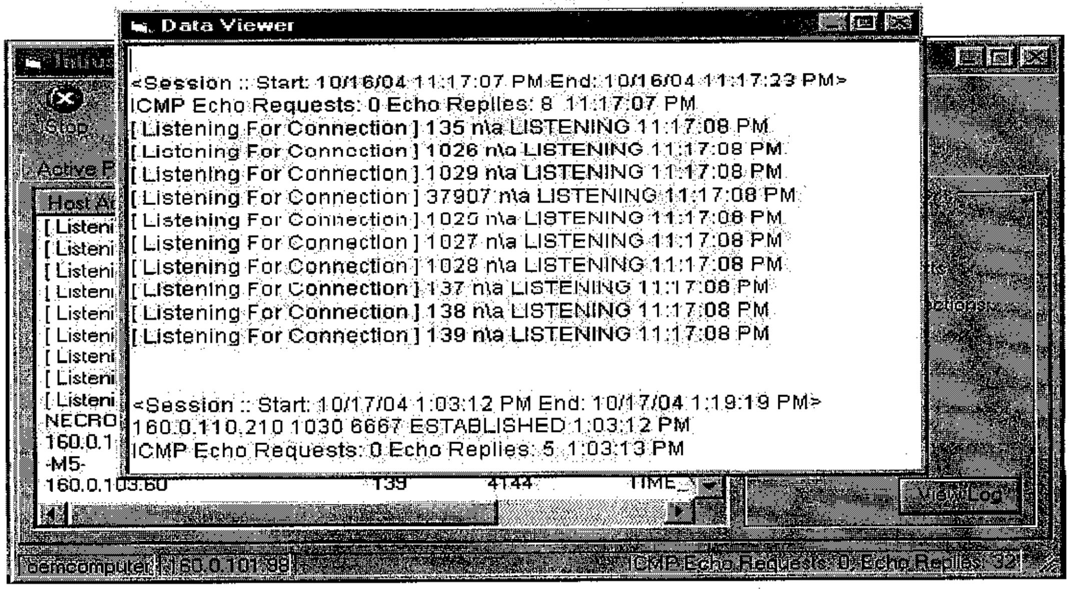
Figure 15. The blocked IP address/host will be listed in the Filter section. In this section Systems Administrator could manually insert the threatening IP address and also re-allow particular IP addresses that have been blocked.



```
Host Address                    Local Port   Remote Port   Status
[ Listening For Connection ]    1026         n\a           LISTEN
[ Listening For Connection ]    1029         n\a           LISTEN
[ Listening For Connection ]    37907        n\a           LISTEN
[ Listening For Connection ]    1025         n\a           LISTEN
[ Listening For Connection ]    1027         n\a           LISTEN
[ Listening For Connection ]    1028         n\a           LISTEN
[ Listening For Connection ]    137          n\a           LISTEN
[ Listening For Connection ]    138          n\a           LISTEN
[ Listening For Connection ]    139          n\a           LISTEN
NECROMANCER982                  139          1924          TIME_W
160.0.123.131                   139          2552          TIME_W
-M5-                            139          2552          TIME_W
160.0.103.60                    139          4144          TIME_W
```

oemcomputer | 160.0.101.98 | ICMP Echo Requests: 0, Echo Replies: 32

Figure 16. All transaction/detection will be recorded in the Log section.



```
<Session :: Start: 10/16/04 11:17:07 PM End: 10/16/04 11:17:23 PM>
ICMP Echo Requests: 0 Echo Replies: 8  11:17:07 PM
[ Listening For Connection ] 135 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 1026 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 1029 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 37907 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 1025 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 1027 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 1028 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 137 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 138 n\a LISTENING 11:17:08 PM
[ Listening For Connection ] 139 n\a LISTENING 11:17:08 PM


<Session :: Start: 10/17/04 1:03:12 PM End: 10/17/04 1:19:19 PM>
160.0.110.210 1030 6667 ESTABLISHED 1:03:12 PM
ICMP Echo Requests: 0 Echo Replies: 5  1:03:13 PM
```

Figure 17. Log file is saved for further inspections and can be viewed by clicking the View Log button.

4.9 SYSTEM TESTING

In ensuring the IDS system is working as required, testing has to be implemented.

One Web Server was installed and set up for the testing purposes. This web server will run a portal. In this case, the author has developed a simple portal called UTP Industrial Internship Training portal. This portal will be used by UTP students that are currently undergoing their Industrial Internship. All the latest important information will be uploaded in the portal. This portal is very important for those students as this will be their main source to get the latest information from the academic services department.

The IDS system is installed in the web server that runs the portal. The IDS will always monitor the data request from user. This is where the IDS systems will play his part.

Portal is one of the most vulnerable and easiest systems to be hacked. Differs from critical systems such as e-banking that have a higher level of security.

One of the most common type of malicious activity is Brute Force attack (refer page 22). The IDS system will always list the entire request from users. From the request connection detected by the IDS, systems administrator could determine whether that request is from a normal user or from other people with bad intentions.

Systems admin could verify whether the request is from a deterministic hacker. One simple example is in the Log In page. If a normal user that forgot his/her password, normally they will try a few possible passwords in their mind. This will normally run for 3 to 5 times before they give up. But, for a deterministic hacker, they will try for thousands of times until they get the correct password. This attack will cause a noticeable increase in traffic and a high number of connections.

This large volume of request will be detected by the IDS system. When the threat is detected, the systems admin could block or terminate the connection instantly.

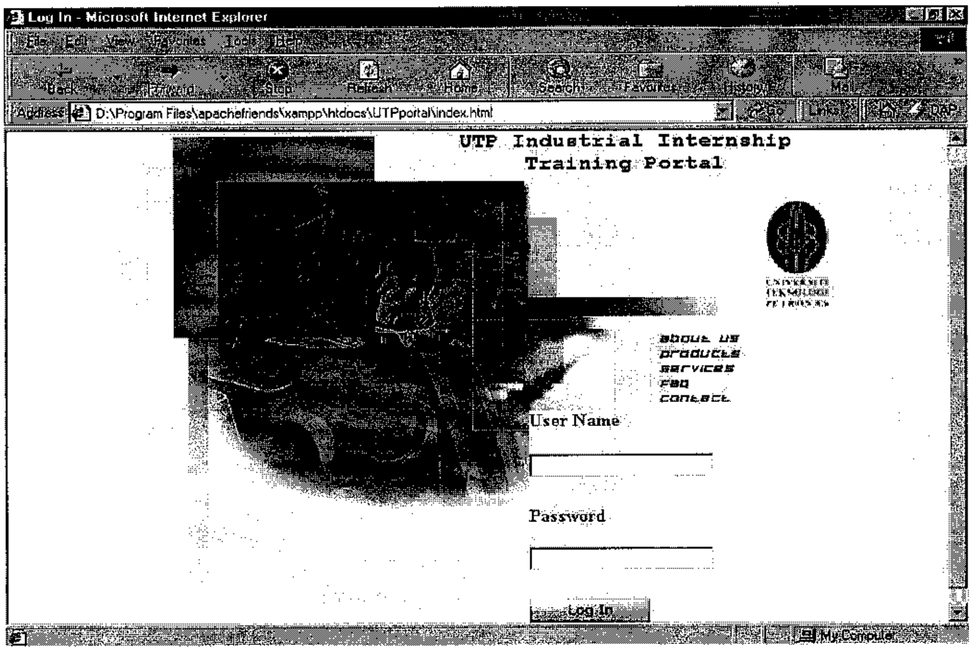
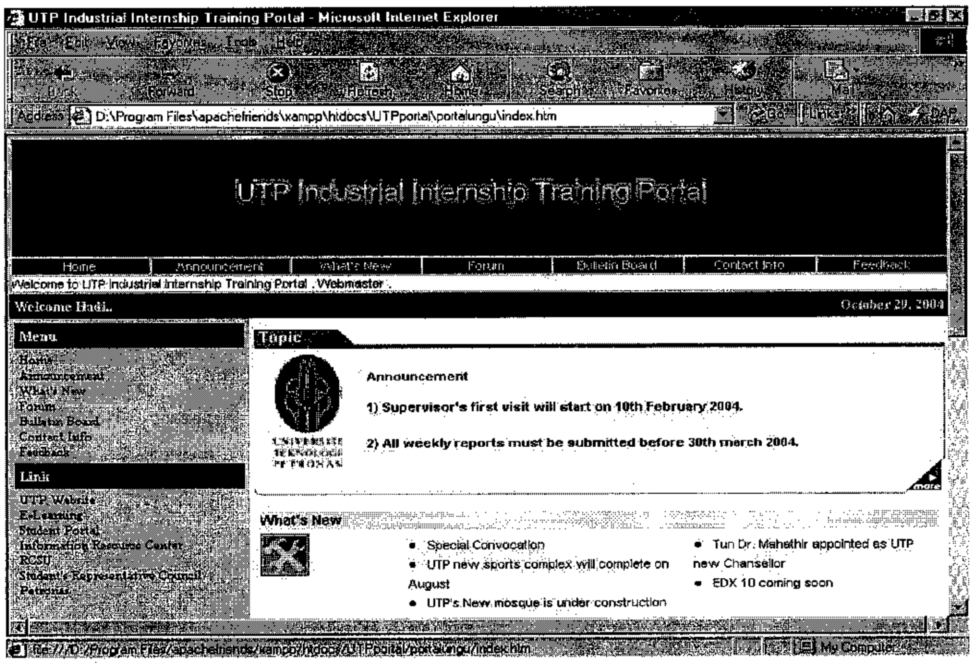

Log In - Microsoft Internet Explorer
Address D:\Program Files\apachefriends\xampp\htdocs\UTPportal\index.html
UTP Industrial Internship Training Portal
about us
products
services
faq
contact
User Name
Password
Log In
My Computer

Figure 18. Log In page


UTP Industrial Internship Training Portal - Microsoft Internet Explorer
Address D:\Program Files\apachefriends\xampp\htdocs\UTPportal\portalungu\index.htm
UTP Industrial Internship Training Portal
Home
Announcement
What's New
Forum
Bulletin Board
Contact Info
Feedback
Menu
Link
UTP Website
E-Learning
Student Portal
Information Resource Center
RCSU
Student's Representative Council
Petronas
Topic
What's New
more

Welcome to UTP Industrial Internship Training Portal . Webmaster .

Welcome Hadi..

October 29, 2004

Announcement

1) Supervisor's first visit will start on 10th February 2004.

2) All weekly reports must be submitted before 30th march 2004.

- Special Convocation
- UTP new sports complex will complete on August
- UTP's New mosque is under construction
- Tun Dr. Mahathir appointed as UTP new Chansellor
- EDX 10 coming soon

Figure 19. Portal main page after a successful log in.

4.10 CHALLENGES TO EFFECTIVE INTRUSION DETECTION SYSTEM

Intrusion Detection technology has quite a way to go to achieve a plug and play implementation. There are still many challenges to achieve effective intrusion detection systems. Fortunately, these challenges can be overcome with some work. The major challenges facing IDS include the following:

4.10.1 Alert Handling

Easily the biggest challenge faced by most organization is alert handling. Until an intrusion detection system is properly tuned to a specific environment, there can be literally thousands of alerts generated on a daily basis. Unfortunately, because we can't determine whether an alert is false or positive until after the alert has been investigated, we must sort through all of the alerts. The expertise and manpower required to handle alerts can be quite daunting.

4.10.2 False Alerts

Most of the intrusion detection systems generate a large number of false alerts. Ratios of four, five pr even ten false alerts for every real alert are quite common.

4.10.3 Evasion

An increasing numbers of attackers' understands the shortcomings of some of the intrusion detection technology, such as signature based IDS. An attackers understands the weaknesses, their attacks are design to bypass detection.

4.10.4 Unknown Attacks

Although IDS is reasonably good at finding known attacks, new and unknown attacks are not well detected by most intrusion detection systems, if they are detected at all.

4.10.5 Architectural Issues

Technology such as switches, Gigabit Ethernet and encryption make network-based intrusion detection much more challenging.

4.10.6 Resource Requirements

Successfully implementing Intrusion detection requires a non trivial investment in resources. The time investment required to properly utilize intrusion detection is substantial. The dollar cost to implement intrusion detection systems can be kept reasonably low by using open source solutions such as Snort, but those savings are usually offset in the time to invest to master and maintain the intrusion detection systems. Using commercial product for intrusion detection systems will usually reduce the time commitment required but by no means eliminates it.

In addition to these current challenges, there are several areas of intrusion detection in which improvement would significantly enhance the value and usefulness of intrusion detection.

4.10.7 Reporting

Consolidated and truly useful reporting from most IDS packages is noticeably lacking.

4.10.8 Visualization

Tools for visualizing activity in process to enhance understanding and responses would be useful.

4.10.9 Correlation

Tighter correlation of activities between various sensors and actual network conditions would yield many benefits such as reduced false alerts, better understanding of attack severity and increased detection.

4.10.10 Vulnerability Assessment

Cross-referencing attack information with current systems configurations and vulnerabilities allows us to determine severity and ramifications of attacks much better.

4.10.11 Data Mining

Better analysis of existing data gathered can detect many attacks that currently go undetected.

4.11 MEASUREMENT CRITERIA

After a few weeks of research and studies, I managed to clarify the criteria needed to develop good intrusion detection systems. These criteria are gathered from various research studies, research paper books, the internet and also my understanding of the network security environment.

4.11.1 Ability to Identify Attacks

The main performance requirement of a IDS is to detect intrusions. However the definition of an intrusion is currently unclear. In particular, many vendors and researchers appear to consider any attempt to place malicious traffic on the network as an intrusion.

In reality a more useful system will log malicious traffic and only inform the operator if the traffic posses a serious threat to the security of the target host. Snort is tending towards this direction with the use of alert classification ranging from 1 to 10. With 1 representing a point of interest only and 10 representing a major threat to security.

4.11.2 Known vulnerabilities and attacks

All IDSs should be capable of detecting known vulnerabilities. However research (Allen 2000), (NSS 2001) indicates that many commercial IDS fail to detect recently discovered attacks. On the other hand if a vulnerability or attack is known all systems should be patched, or workarounds applied thus the need for an IDS to detect these events will be removed. Unfortunately the reality is that many systems are not patched or upgraded as vulnerabilities are discovered.

4.11.3 Unknown attacks

This must be the most important feature of any IDS. It is the IDS that can detect attacks that are not yet known which will justify its implementation. New vulnerabilities are discovered every day. By its very nature these are also the most difficult attacks to detect.

4.11.4 Relevance of attacks

This refers to the ability of the IDS to identify the relative importance of any attack. For an example, given the use of a windows exploit on a UNIX system is not of high importance. However if the alert is raised, and the analyst must investigate every alert, a mechanism should be available to distinguish the relevance of different alerts.

4.11.5 Stability, Reliability and Security

Any IDS should be able to continue consistently operate in all circumstances. The application and operating system should be capable of running for years without segmentation faults or memory leakage.

An important function of IDS is to consistently report identical events in the same manner. One disadvantage of a product using signature recognition is the ability of different users to configure different alerts to provide different messages. Thus traffic on one network may trigger a different alert to the same traffic on another system of the same type. A number of efforts are currently underway to solve this problem. Both security focus and CVE provide databases of known vulnerabilities, and exploits targeting them.

The system should also be able to withstand attempts to compromise it. If an attacker can identify IDS on a network it could prove to be a valuable asset. It is also possible the attacker will attempt to disable the system using DoS or DDoS techniques. The system should be able to withstand all of these types of attack.

4.11.6 Information provided to analyst

The information provided to the analyst when alert is raised should be enough to clearly identify the reason the event causing the event to be raised, and the reason this event is of interest. It should also provide links to vulnerability databases, such as bugtraq or CVE to assist the analyst in determining the relevance and appropriate reaction to a particular alert.

4.11.7 Identify target and source

The alert should also identify the source of the alert and the target system. Further information such as a whois or DNS lookup on a IP address would be also be beneficial.

4.11.8 Severity, potential damage

Identification of the potential severity of an attack. Some alerts are triggered by events to relate to information gathering, such as port scanning. Although this information may be relevant if a more serious attack in launched the volume of scanning that occurs on the internet makes it impractical to investigate every time a network is scanned. On the other hand indication that a local host has been compromised by a Trojan should be given higher priority.

4.11.9 Outcome of attack (Success or failure)

Another useful (although currently non existent) feature of an IDS should be to indicate the outcome of an attack. In most cases, alert simply indicates that an attempt has been made. It is then the responsibility of the analyst to search for correlating activity to indicate the outcome of the attack. If an IDS were to present the analyst with a list of other alerts generated by the target host, and a summary of other (non alert) traffic the evaluation of the outcome could be greeted accelerated.

4.11.10 Legal validity of data collected

The legal validity of the data collected by any IDS is of extreme importance if any legal will be taken against the attacker. A disturbingly large number of systems do not collect the actual network packets; instead they simply record their own interpretation of events. A more robust system will also capture and store the network traffic, as well as raising the alert.

4.11.11 Manageability

One of the greatest risks of IDS is that once the system is implemented it will not be utilized to its full capabilities. Often the reason for this is due to the complexity of configuring and maintaining the system. It is also important that IDS can be optimized for a particular network. There is no point in monitoring for web server exploits if there is not a web server on the network.

4.11.12 Ease or complexity of configuration

Unfortunately the usability of a system is usually inversely proportional to the flexibility and customizability of that system. The desire for flexibility can configurable of the system will be determined by the users of the system, the network in which it will be operating and the level of functionality required from the system.

If the system is to be maintained by a network administrator who is also responsible for standard network management he or she is unlikely to have the time available to optimize and configure the system so usability will be a primary consideration. On the other hand if an intrusion analyst if employed specifically to manage intrusion detection a more complex system with greater functionality may be desired.

4.11.13 Possible configuration options

The IDS should be capable of being optimized for the systems on the network. As mentioned earlier there is no point in performing http analysis if a web server is not operating on the network under inspection. The level of traffic on the network will also determine the intensity of analysis performed. A simple system suitable for a single network segment with low traffic will be able to combine the sensor and analysis functions within the single unit. A network with high levels of traffic may need to separate the sensor and analysis functions across different hosts.

There are also a number of other configuration options that may apply to particular situations. For example in some situations the IDS (i.e. analyst) may not be allowed to view the contents of packets on the network. In this case it should be possible to configure the DIS to only examine (and store) the header information from the packets.

4.11.14 Scalability and Interoperability

a) Scalability

Most organizations grow and expand over time. As they expand so do their supporting infrastructure, include computer networks. Any IDS should be capable of expanding with the network. As new network segments are added new IDS may also be needed. Will it be possible to consolidate the reports from multiple IDS into a single user interface? Another important question will be the storage of this information. If a small network is monitored data storage may be possible in flat files. However as the amount of data collected grows it may be necessary to transfer this data storage into a database.

b) Interoperability

Research has proven that the most effective intrusion detection requires correlating information from a range of sources. This includes NIDS, HIDS, system logs, firewall logs and any other information sources available. At the time of writing the Intrusion Detection Working Group (IDWG) had submitted a number of documents defining standards for communication between IDSs. It is expected that these will be released as RFCs in the near future.

Once these standards are implemented any IDS using the standard protocols will be able to communicate with and other IDS. This will enable an organization to implement a range of IDS from different vendors and still maintain interoperability.

4.11.15 Vendor support

The level of vendor support required in an implementation will be determined by the skill levels of the staff implementing the system. However as staff turnover rates are common in the IT industry it is worthwhile considering the level of support that is available from the vendor.

4.11.16 Signature updates

Any signature based IDS is dependant upon it signatures to detect intrusions. The ability of these systems to detect new or even modified intrusions has been shown to be poor (Allen 2000). In order for these systems to be effective updated signatures must be available as new vulnerabilities and exploits are discovered.

Many signature based systems now allow the operator to create their own signatures. This can allow the system to monitor for new alerts as they are discovered without relying on the vendor to supply updates. However monitoring vulnerabilities and writing signatures as they occur is a demanding task. Consider the amount of traffic on bugtraq in a single day.

CHAPTER 5

CONCLUSION

Selecting and implementing IDS is a challenging task. There are a number of factors to be considered, and these factors will change from situation to situation. In order to ensure a successful implementation an organization should determine its requirements and then locate a system that meets them.

Experienced security professionals realize the value of the triad prevention, detection and response (Smaha and Winslow, 1994). One of the best defenses is to build formidable preventive mechanisms. However, in practice, prevention alone is insufficient. Programs bugs and human errors have resulted in numerous security breaches in the past.

A security policy must be monitored for violations. That is, we want to detect any security breaches that are caused by configuration problems or slack policies. Finally, because security solutions must scale, it must be possible to define automated responses to security incidents. Care is, of course, needed. We do not want a response policy that tries to terminate all of the processes running on behalf of the perpetrator, especially if this affects availability of resources that are crucial to us.

Intrusion detection can be extremely valuable tool when implemented correctly. Understanding the practical limitations as well as the capabilities of the technology will enable us to achieve the best results. Understanding the history of intrusion detection helps to reinforce what we can expect to gain from intrusion detection. As the technology improves, we can harness the increase in capabilities to the best advantage.

REFERENCES

1. Ron Gula, How to Handle and Identify Network Probes, April 1999, www.securitywizards.com [Local Copy] Required Reading.
2. Hobbit, The FTP Bounce Attack, The original paper on the subject. http://www.insecure.org/nmap/ hobbit.ftpbounce.txt Reference.
3. Fyodor, Remote OS detection via TCP/IP Stack Finger Printing. Written: October 18, 1998 Last Modified: April 10, 1999. http://www.insecure.org/nmap/nmap-fingerprinting-article.html Required Reading.
4. solar designer, Designing and Attacking Port Scan Detection Tools, Phrack Magazine, Volume 8, Issue 53 July 8, 1998, article 13 of 15, www.phrack.com or http://phrack.infonexus.com/ search.phtml?view&article=p53-13 Recommended Reading.
5. List of ports used by Trojans, http://www.simovits.com/nyheter9902.html Reference.
6. www.dshield.org
7. http://citeseer.ist.psu.edu/Security/IntrusionDetection/
8. www.sans.org
9. www.mandrakesoft.com/products/mnf/features
10. http://vsftpd.beasts.org

11.www.robertgraham.com/pubs/network-intrusion-detection.html#1.1

12.www.snort.org

13.http://bastille-linux.org

14.www.tripwire.org