A NEW APPROACH FOR THE SCALABLE INTRUSION DETECTION IN HIGH-SPEED NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÜMİT BURAK ŞAHİN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2007

Approval of the thesis:

A NEW APPROACH FOR THE SCALABLE INTRUSION DETECTION IN HIGH-SPEED NETWORKS

Submitted by **ÜMİT BURAK ŞAHİN** in partial fulfillment of the requirements for the degree of Master of Science in **Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, **Graduate School of Natural and Applied Sciences** _______________

Prof. Dr. İsmet Erkmen
Head of Department, **Electrical and Electronics Engineering** _______________

Dr. Şenan Ece (Güran) Schmidt
Supervisor, **Electrical and Electronics Engineering Dept.** _______________

Examining Committee Members

Prof. Dr. Hasan Güran
Electrical and Electronics Eng. Dept., METU _______________

Dr. Şenan Ece (Güran) Schmidt
Electrical and Electronics Eng. Dept., METU _______________

Assoc. Prof. Dr. Özgür Barış Akan
Electrical and Electronics Eng. Dept., METU _______________

Assist. Prof. Dr. Cüneyt Bazlamaçcı
Electrical and Electronics Eng. Dept., METU _______________

Ahmet Fethi Ayhan (MSc)
TURK TELEKOM _______________

Date: December 3, 2007

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Ümit Burak Şahin

Signature :

ABSTRACT

A NEW APPROACH FOR THE SCALABLE INTRUSION DETECTION
IN HIGH-SPEED NETWORKS

Şahin, Ümit Burak
M.Sc., Department of Electrical and Electronics Engineering
Supervisor: Şenan Ece (Güran) Schmidt

December 2007, 144 pages

As the networks become faster and faster, the emerging requirement is to improve the performance of the Intrusion Detection and Prevention Systems (IDPS) to keep up with the increased network throughput. In high speed networks, it is very difficult for the IDPS to process all the packets. Since the throughput of IDPS is not improved as fast as the throughput of the switches and routers, it is necessary to develop new detection techniques other than traditional techniques. In this thesis we propose a rule-based IDPS technique to detect Layer 2-4 attacks by just examining the flow data without inspecting packet payload. Our approach is designed to work as an additional component to existing IDPS as we acknowledge that the attacks at Layer 5 and above require payload inspection. The rule set is constructed and tested on a real network to evaluate the performance of the system.

Keywords: Intrusion Detection and Prevention Systems, High Speed Network, Layer 2-4 Attacks, Flow-based Detection

ÖZ

YÜKSEK HIZLI AĞLARDA ÖLÇEKLENEBİLİR ATAK TESPİT VE ENGELLEME SİSTEMİ YAKLAŞIMI

Şahin, Ümit Burak
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Danışmanı: Şenan Ece (Güran) Schmidt

Aralık 2007, 144 sayfa

Ağ iletişim hızı sürekli olarak arttıkça, Atak Tespit ve Engelleme Sistemlerinin (ATES) performansının da ağ performansı doğrultusunda arttırılması öncelikli ihtiyaç haline gelmiştir. Yüksek hızlı ağlarda, geleneksel ATES'lerinin çalışma prensibi ile tüm paketlerin incelenmesi çok zordur. ATES'lerinin performansının artışı, anahtar, yönlendirici gibi ağ cihazlarının performansının artışı kadar hızlı olamadığı için, geleneksel yöntemlerden farklı yeni yöntemler ile atak tespiti ve engellemesi yapılması gerekmektedir. Bu tez çalışmasında, ağdaki tüm paketlerin açılarak incelenmesine gerek kalmayacak şekilde, trafiğin akış bilgileri incelenerek, 2.-4. katman atakların tespit edilebilmesi ve engellenmesi için kural tabanlı ATES tekniği önerilmiştir. 5. ve üstü katmanlarda gerçekleşen atakların tespiti için paket incelemesi gerektiğinden, yaklaşımımız mevcut ATES'e ek ağ bileşeni olarak çalışacak şekilde tasarlanmıştır. Gerçek ağda kural seti oluşturulmuş ve sistem performansının değerlendirilmesi için testler yapılmıştır.

Anahtar Kelimeler: Atak Tespit ve Engelleme Sistemleri, Yüksek Hızlı Ağlar, 2-4 Katman Ataklar, Akış Tabanlı Tespit

ACKNOWLEDGMENTS

I would like to thank Dr. Şenan Ece (Güran) Schmidt for her creative ideas, valuable supervision and patience. Her support on this thesis work increased my motivation to the top level and her guidance encouraged me to complete this thesis.

I want to thank my managers in SEKOM for their support on this research. I also want to thank my colleagues for their continuous assistance.

I would also like to thank my parents and my sister Serpil for giving me encouragement and patience during this thesis and all kind of supports during my whole education.

Finally, I have very special thanks for my dear wife İnci, for her precious support and patience during this thesis.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

AAA	Authentication, Authorization, and Accounting
ADSL	Asynchronous Digital Subscriber Line
ANN	Artificial Neural Network
AOL	American Online
AS	Autonomous System
ASCII	American Standard Code for Information Interchange
ASIC	Application Specific Integrated Circuit
ATM	Asynchronous Transfer Mode
CIPS	Coordinated Intrusion Prevention System
CPU	Central Processing Unit
DHCP	Dynamic Host Configuration Protocol
DLL	Data Link Layer
DoD	Department of Defense
DoS	Denial of Service
DDoS	Distributed Denial of Service
DPI	Deep Packet Inspection
DMZ	Demilitarized Zone
DNS	Domain Name System
EBCDIC	Extended Binary Coded Decimal Interchange Code
FD	Flow Detection
FPGA	Field Programmable Gate Array
FTP	File Transfer Protocol
GrIDS	Graph-Based Intrusion Detection System
ICMP	Internet Control Message Protocol
IDES	Intrusion Detection Expert System

IDS	Intrusion Detection System
IDPS	Intrusion Detection and Prevention System
IETF	Internet Engineering Task Force
IOS	Internetworking Operating System
IP	Internet Protocol
IPS	Intrusion Prevention System
IPTV	Internet Protocol Television
IT	Information Technology
LAN	Local Area Network
MAC	Media Access Control
MAN	Metropolitan Area Network
MIDS	Multiple Intrusion Detection System
MSN	Microsoft Network
NADIR	Network Anomaly Detection and Intrusion Reporter
NAT	Network Address Translation
NBAR	Network Based Application Recognition
NIC	Network Interface Card
NID	Network Intrusion Detector
NSM	Network Security Monitor
NTP	Network Time Protocol
OC	Optical Carrier
PCF	Partial Completion Filter
PDA	Personal Digital Assistant
PFW	Parallel Firewall
PpS	Packet per Second
RHPNIDS	Rule-Based High-Performance Network Intrusion Detection System
RDP	Remote Desktop Protocol
RRD	Round Robin Database
QoS	Quality of Service
RAM	Random Access Memory
SANS	SysAdmin, Audit, Networking, and Security

SMS	Short Message Service
SNMP	Simple Network Management Protocol
SONET	Synchronous Optical Networking
SSL	Secure Sockets Layer
SVM	Support Vector Machine
SYN	Synchronize
TCAM	Ternary Content Addressable Memory
TCP	Transmission Control Protocol
TLS	Transport Layer Security
ToS	Type of service
TTL	Time to Live
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network
VoIP	Voice over IP
VPN	Virtual Private Network
WAN	Wide Area Network
WiMAX	Worldwide Interoperability for Microwave Access

CHAPTER 1

INTRODUCTION

An intrusion detection system (IDS) inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. An IDS detects unwanted manipulations to computer systems, both through the Internet and Intranet. The manipulations may take the form of attacks by crackers. An IDS can detect many types of malicious network traffic and computer usage that can not be detected by a conventional firewall. These include network attacks against vulnerable services, data driven attacks on applications, host based attacks such as privilege escalation, unauthorized logins and access to sensitive files, and malware (viruses, trojan horses, and worms).

An Intrusion Prevention System (IPS) detects the attacks similar to an IDS. An IPS is the next security layer that combines the protection of firewalls with the monitoring ability of an IDS to protect networks with the analysis necessary to make the proper decisions on the fly. IPS are designed to sit inline with traffic flows and prevent attacks in real-time. IDS appears to be much easier to implement into a network using port mirroring. Since IDS are passive, they are inherently fail-open. IPS must be introduced into the entire network infrastructure, not simply by port mirroring on a network segment. Network traffic saturation must be considered to ensure the additional IPS network traffic does not bring down the network. IPS may be considered as a system that further supports the firewall-to-IDS protection methodology. Therefore, IPS can be named as IDPS.

There are two kinds of intrusion detection and prevention systems (IDPS) categorized according to detection methods: *misuse detection* and *anomaly detection*. In misuse detection, the IDPS analyzes the information it gathers and compares it to large databases of *attack signatures*. Frequent updates are needed. False positives are the menace for the network. IDPS gathers information using deep packet inspection. Deep packet inspection (DPI) is a form of computer network packet filtering that examines the data and/or header part of a packet as it passes an inspection point, searching for non-protocol compliance, viruses, spam, intrusions. Examining data part of a packet causes latency on the network. In addition, this method increases the load of IDPS that causes network traffic saturation. Essentially, the IDPS looks for a specific attack that has already been documented. Similar to a virus detection system, misuse detection software is only as good as the database of attack signatures. To address the threats posed by network-based attacks in high speed networks, misuse detection method is not so appropriate. In anomaly detection, the system administrator defines the baseline, or normal, state of the network's traffic load, protocol, and typical packet size. The anomaly detector monitors network segments to compare their state to the normal baseline and look for anomalies.

Rapid increase of Internet users throughout the world has resulted in exponential growth of Internet traffic in wide area networks (WANs). Internet applications have become more bandwidth intensive as multimedia content becomes indispensable. On the other hand, despite the accurate detection capability, throughput of intrusion detection and prevention systems does not improve as fast as that of network infrastructure. The backbone and global Internet connections reach 10Gbps capacities. At these speeds, the traffic analysis of packet payloads by intercepting the traffic forces the limits of the buffers and memories of computer architectures with current capacities. Mirroring the traffic to another link, and storing it for further analysis of the packet payload requires vast amount of storage area. For example for a 622 Mbps link, it requires a 400 Terabytes storage area for 1 hour data when the link is 100% utilized.

Detection and prevention of attacks and undesirable traffic with high precision are not so easy. Data payload inspection in high speed networks is almost impossible. Currently, the most of the intrusion detection methods look for specific signatures in the IP packet payload. This signature based detection methods have scalability problems to work at backbone rates slowing down the response time of the network. Also, inspection of payloads violates the user data privacy. More important than these, these methods are useless for the encrypted data. IDPS can only analyze the encrypted packets if it knows the encrypted packet policies and attributes. However, this will affect the throughput of the IDPS. When data processing cannot keep pace with the speed and the throughput of networks, it is often the case that the packets not analyzed on time are dropped. These dropped packets may have hostile data with attack signatures, which causes a high false negative rate in the IDS. In another case, if IDPS is being used, communication will be interrupted because of the system's bad performance. In addition, some attackers can overload the network with large number of packets. Therefore, it is necessary to develop efficient intrusion detection techniques.

In this thesis, a new approach is proposed for the efficient intrusion detection and prevention for high speed networks. We observe that the attacks in the network are composed of attacks that act at Layer 5 and above and attacks that act at Layer 2 to Layer 4. IDPS perform DPI by checking a long list of signatures (in the order of thousands of signatures) for all of the packets regardless of the network layer the attack operates. Acknowledging that the attacks at Layer 5 and above require DPI, we propose an additional component that we call *Detection Engine* to existing IDPS which analyzes and detects the Layer 2-Layer 4 attacks and informs the IDPS to take the required actions. It is no longer necessary for the IDPS to check the related signatures to the attacks that can be detected by the Detection Engine and hence the network can respond faster. However, the Detection Engine is not enough to detect all the attacks by itself and the attacks that cannot be detected by it are detected by the IDPS.

The Detection Engine works with a set of *rules* that describe the behaviour of the attacks in time at Layer 2- Layer 4 *flow* level. We construct the rules by performing a first characterization of the attacks in a small scale laboratory network by using the attacks that we generate artificially. We collect flow level data from different network devices and then correlate this data to intrusions that are detected by a commercial IDPS to deduce the rules. These rules are then refined in a large scale network with real traffic and real intrusions following the same approach of correlating flow data and IDPS results. We evaluate the performance of our approach in the large scale network with real traffic. Our performance metrics are the accuracy of our Detection Engine as well as the improvement of the IDPS performance. Our experiments show that the Detection Engine is 96% accurate with respect to the commercial IDPS and the response time of the network is improved around 30%.

Note that as the Detection Engine does not perform DPI it can eliminate the attacks even if they are encrypted. Although traditional IDPS, that is using misuse detection, have to be up to date to detect new attacks, the solution in this study has *day zero protection*. Since this solution is flow and rule based, it can eliminate a new attack. For signature based systems, an attack and its variants must be defined in the database. Since the behaviors of an attack and variants are the same, it is easy to detect and prevent variants of attacks.

The thesis is organized as follows:

In Chapter 2, first, an introduction to high speed networks is given and then, network security and issues of providing network security as high speeds is introduced. Next, IDPS are reviewed in detail. Finally, the Network Tools that are used to develop intrusion detection technique in the solution in this thesis are introduced.

In Chapter 3, the procedure that we follow to construct the flow-based rules for the Detection Engine is described. The results of our performance tests for our approach are presented and discussed in Chapter 4. A conclusion and a discussion on future work are included in Chapter 5.

CHAPTER 2

INTRUSION DETECTION AND PREVENTION SYSTEMS FOR HIGH SPEED NETWORKS

Intrusion detection and prevention systems are an important component of defensive measures protecting computer systems and networks from abuse. In the 1980s, most intruders were experts, with high levels of expertise and individually developed methods for breaking into systems. Nowadays less expert intruders can make more sophisticated attacks using readily available intrusion tools and exploit scripts. On the other hand, throughput of intrusion detection and prevention systems does not improve as fast as that of network infrastructure. It causes bottleneck on network. Therefore, we can say IDPS is a rate determining object of network throughput. The research in this study is motivated for reducing the load on the IDPS without giving up target network security. Using IDPS more efficiently increases the network throughput. We first present an overview of high speed networks. Then, we present network security and literature survey on IDPS techniques. Finally, the attack generation tools and traffic analysis tools that are used to develop intrusion detection technique in the solution in this thesis are introduced.

2.1 HIGH SPEED NETWORKS

“Never underestimate the bandwidth of a station wagon full of tapes hurtling down the highway.

Andrew S. Tanenbaum

Rapid increase of Internet users throughout the world has resulted in the exponential growth of Internet traffic in wide area networks (WANs). Worldwide Internet traffic, which is continuing to double every year, was 180 Petabits per day in 2002 [1]. To meet the increased demand for bandwidth, optical systems have been widely deployed in the backbone network [2]. Global IP Traffic, which is continuing to grow at a compound annual growth rate of 50% from 2006 to 2011, is forecasted to reach to 29 Exabyte’s per month in 2011 from 4.5 Exabyte’s per month in 2006 [3]. The estimations are summarized in Figure 1. In the figure, consumer segment includes fixed IP traffic generated by households, university populations, and internet cafes. Business segment includes all fixed IP WAN or Internet traffic generated by organizations (including government). Mobility segment includes mobile data and Internet traffic generated by handsets, notebook cards, Wi-Fi hotspots, WiMAX. In addition to more users, Internet applications have become more bandwidth intensive as multimedia content becomes indispensable. Online video has grown rapidly. In North America, online video traffic has jumped from 7 percent of traffic in 2005 to 18 percent in 2007 [4]. In response to this remarkable application development, many service providers are upgrading their network capacity.

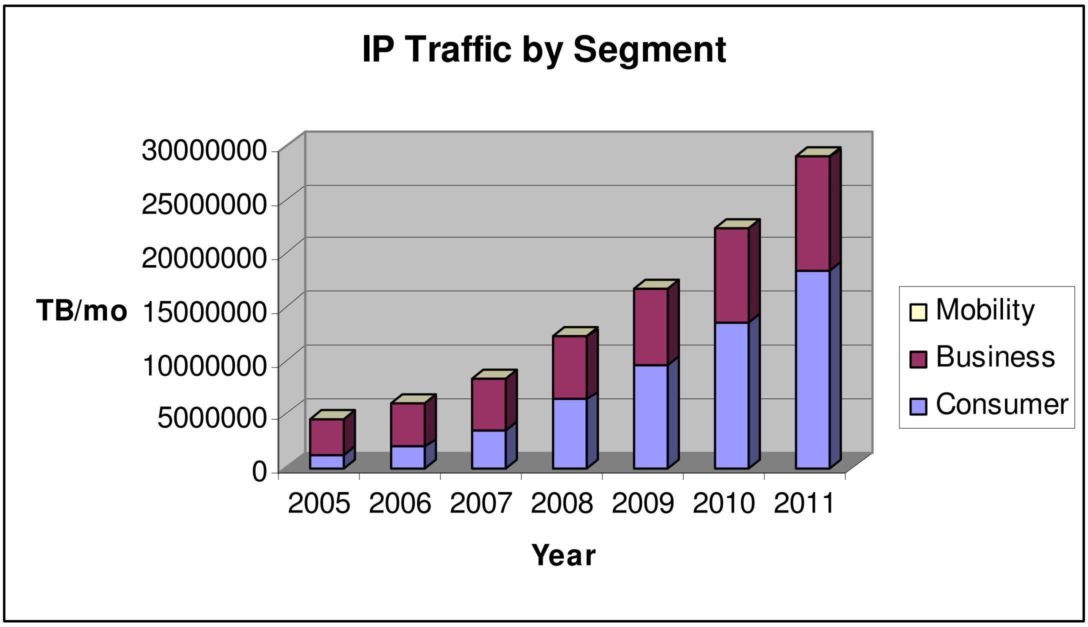


Figure 1 Worldwide IP Traffic by Segment [3]

The Internet revolution has led to a massive increase in data traffic. This trend is set to continue; over the next few years and it is likely that 95% of all communication traffic will shift to data. The need to support high bandwidth traffic has required that equipment performance grow at an exponential rate. WAN equipment which only operated at speeds of OC-48 (2.5Gbps) in 2000, now runs at speeds up to OC-768 (40Gbps) [5][6].

Residential networks deploy asynchronous digital subscriber line (ADSL) and cable modems to provide subscribers with broadband access. Connecting the broadband access network and the broadband backbone network is the metropolitan area network (MAN). Originally, the MANs were designed and deployed to handle voice traffic. Data requirements arose only as an afterthought [7].

Due to the simplicity and cost effectiveness of the protocol, Ethernet has been widely deployed in the enterprise local area network (LAN) environment to handle the data traffic. Standard interfaces are readily available for 10/100/1000 Mbps Ethernet and the 10-Gbps Ethernet standard. Because of Ethernet's relative simplicity and the economies of scale from the existing Ethernet installed based, infrastructure equipment costs for Ethernet are significantly less than for frame relay and asynchronous transfer mode (ATM). For example, Ethernet is much easier to learn and deploy than SONET and ATM technologies. With suitable rate-limiting functions to manage the available resources and with sufficiently large trunk capacity, Ethernet can provide rapid bandwidth on demand. For networking, it is important to remove the inter-working issues between platforms and environments in order to make service provisioning and activation simpler. Because of its IP friendly nature, Ethernet eliminates a layer of complexity (e.g., ATM and SONET) from WAN access, thus reducing configuration requirements. Integrating and interfacing end customer information technology (IT) systems are relatively simple with Ethernet metro services. All of the above advantages coupled with the developed Gigabit Ethernet WAN capability have well positioned Gigabit Ethernet as a technology to break the bandwidth bottleneck in the MAN environment [8].

Despite the slowdown in the deployment of telecommunication equipment in the early 2000s, the demand for bandwidth has not stopped increasing. The explosion of video traffic has dictated new requirements on the network, which for a long time has been carrying voice traffic [9].

Today, video is no longer restricted to our TV sets. Cell phones, PDAs, wireless home entertainments systems, laptops/desktops and other personal electronic devices all have video capabilities offering the user an enhanced multimedia experience, anywhere, any time. Moreover, delivering video services has shifted from broadcast only to on demand content (Video-on-Demand, Free-on-Demand, IPTV, etc.), peer to peer video conferences and Internet video downloads (e.g. Google videos, YouTube.com, music video clips on iTunes.com, etc.), thus putting more pressure on the network to provide any-to-any high-bandwidth connectivity with high quality of service. Youtube.com is just the beginning of online video. Online video is growing rapidly. Internet backbone is a potential bottleneck for new contents.

Furthermore, Enterprise Local Area Networks are now being interconnected over 10-Gbps Ethernet interfaces, storage arrays are running over 10-Gbps Fiber Channel interfaces and carriers' routers are now equipped with 40-Gbps interfaces, thus increasing the transport network's bandwidth capacity becomes more imperative than ever.

Ethernet technology has evolved into multi-gigabit bandwidth with applications extending from LAN, MAN, and even to WAN. Telecom carriers are faced with fast growing Internet data traffic that exceeds traditional voice traffic. The simple protocol of Ethernet can offer telecom carriers many advantages in terms of simplified network architecture and substantial equipment cost reduction. The dynamic bandwidth allocation flexibility makes many new business applications possible.

2.2 NETWORK SECURITY

"The only system which is truly secure is one which is switched off and unplugged, locked in a titanium lined safe, buried in a concrete bunker, and is surrounded by nerve gas and very highly paid armed guards. Even then, I wouldn't stake my life on it…."

Gene Spafford, Director, Computer Operations, Audit, and Security Technology (COAST), Purdue University

Over the past few years, Internet-enabled business, or e-business, has drastically improved companies' efficiency and revenue growth. E-business applications such as e-commerce, supply chain management, and remote access enable companies to streamline processes, lower operating costs, and increase customer satisfaction. Such applications require mission-critical networks that accommodate voice, video, and data traffic, and these networks must be scalable to support increasing numbers of users and the need for greater capacity and performance. However, as networks enable more and more applications and are available to more and more users, they become ever more vulnerable to a wider range of security threats. To combat those threats and ensure that e-business transactions are not compromised, security technology must play a major role in today's networks.

In the past, closed networks were designed using closed network topology. Closed network is implemented in a corporate environment and provides connectivity only to known sites and clients without connecting to public networks. Therefore it was thought to be secure network.

The networks of today are designed with availability to the Internet and public networks, which is a major requirement. Most of today's networks have several access points to other networks both public and private; therefore, securing these networks has become fundamentally important and difficult.

With the development of large open networks there has been a huge increase in security threats in the past 20 years. Not only have hackers discovered more vulnerabilities but the tools used to hack a network have become simpler and the technical knowledge required has decreased. More dangerous and easy to use threat capabilities are shown in Figure 2. There are downloadable applications available that require little or no hacking knowledge to implement. There are also applications intended for troubleshooting a network that when used improperly can pose severe threats. Techniques used for attacks are becoming much more sophisticated. Signatures left by attacking software are becoming much more difficult to detect through either operations analysis or through signature-based detection systems. Dynamic mutation of the attacking tools compounds the problem of detection [10]. US-CERT publishes information on a wide variety of vulnerabilities. Number of vulnerabilities' types published by CERT is shown in Figure 3.

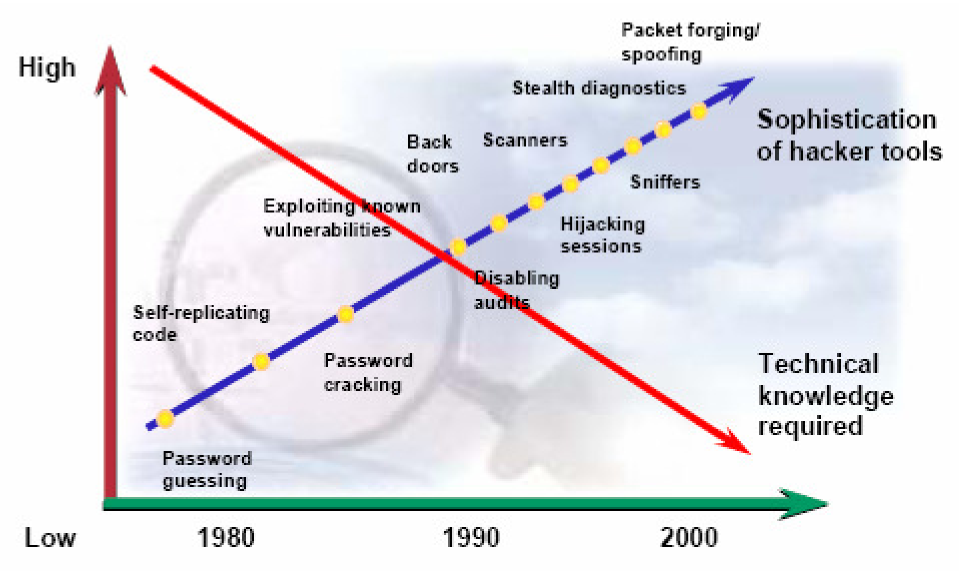


Figure 2 Threat Capabilities [11]

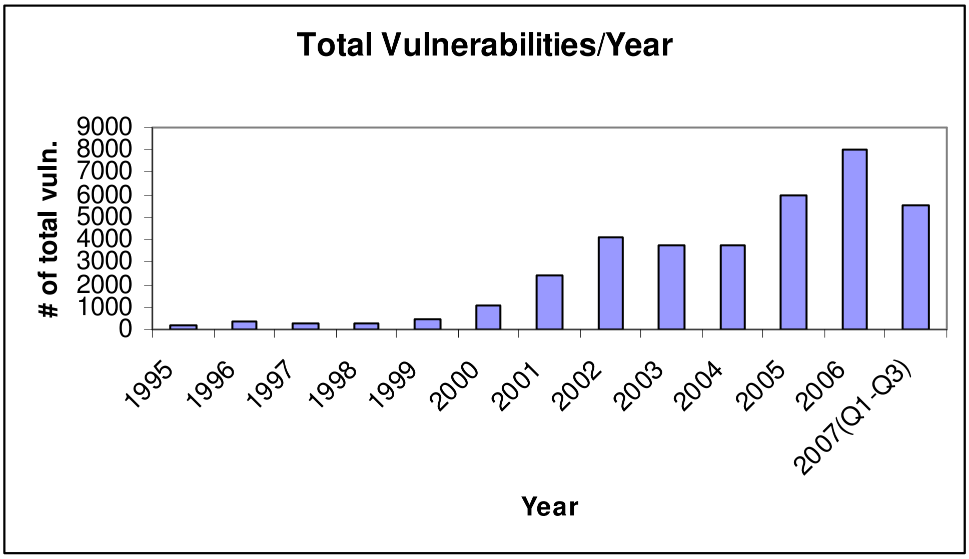


Figure 3 Total Vulnerabilities Cataloged [12]

Security must be a fundamental component of any e-business strategy. As enterprise network managers open their networks to more users and applications, they also expose these networks to greater risk. The result has been an increase in business security requirements. The Internet has radically shifted expectations of companies' abilities to build stronger relationships with customers, suppliers, partners, and employees. E-business requires mission-critical networks that accommodate ever-increasing constituencies and demands for greater capacity and performance. These networks also need to handle voice, video, and data traffic as networks converge into multi service environments.

2.2.1 Network Security Policy

The network security policy is the core of the network security process. Every company should have a written network security policy. At a minimum, that policy should fulfill the following objectives:

- Analyze the threat based on the type of business performed and type of network exposure.
- Determine the organization's security requirements.
- Document the network infrastructure and identify potential security breach points.
- Identify specific resources that require protection and develop an implementation plan.

After setting appropriate policies, a company or organization must methodically consider security as part of normal network operations. This process could be as simple as configuring routers to not accept unauthorized addresses or services, or as complex as installing firewalls, IDPS's, centralized authentication servers, and encrypted virtual private networks (VPNs). Network security is a continuing process:

- **Secure:** This includes hardening your network systems by installing security devices such as firewall, IDPS, VPN, and AAA (authentication, authorization, and accounting) servers, vulnerability patching.
- **Monitor:** To ensure that a network remains secure, it is important to monitor the state of security preparation. Network vulnerability scanners can

proactively identify areas of weakness, and IDPS's can monitor and respond to security events as they occur.

- **Test**: Testing security is as important as monitoring. Without testing the security solutions in place, it is impossible to know about existing or new attacks.
- **Improve:** Monitoring and testing provides the data necessary to improve network security. Administrators and engineers should use the information from the monitor and test phases to make improvements to the security implementation as well as to adjust the security policy as vulnerabilities and risks are identified.

The importance of network security has been significantly increasing in the past few years. However, the increasing complexity of managing security policies particularly in enterprise networks poses real challenge for efficient security solutions. Network security perimeters such as Firewalls, IPSec gateways, IDPS's operate based on locally configured policies. Yet these policies are not necessarily autonomous and might interact between each other to construct a global network security policy. Due to manual, distributed and uncoordinated configuration of security policies, rules conflicts and policy inconsistencies are created, causing serious network security vulnerabilities. In addition, enterprise networks continuously grow in size and complexity, which makes policy modification, inspection and evaluation nightmare. Addressing these issues is a key requirement for obtaining provable security and seamless policy configuration. In addition, with growth in network speed and size, the need to optimize the security policy to cope with the traffic rate and attacks is significantly increasing. The constant evolution of policy syntax and semantics make the functional testing of these devices for vulnerability penetration is a difficult task [13][14].

A security policy is a formal statement of the rules by which people who are given access to an organization's technology and information assets must abide. It is essentially a document summarizing how the corporation will use and protect its computing and network resources [15].

Security policies provide many benefits and are worth the time and effort needed to develop them. Developing a security policy:

- Provides a process for auditing existing network security.
- Provides a general security framework for implementing network security.
- Defines which behavior is and is not allowed.
- Helps determine which tools and procedures are needed for the organization.
- Helps communicate consensus among a group of key decision makers and define responsibilities of users and administrators.
- Defines a process for handling network security incidents.
- Enables global security implementation and enforcement. Computer security is now an enterprise-wide issue, and computing sites are expected to conform to the network security policy.
- Creates a basis for legal action if necessary.

The following are some of the key policy components [15]:

- Statement of authority and scope specifies who sponsors the security policy and what areas the policy covers.
- Acceptable use policy specifies what the company will and will not allow regarding its information infrastructure.
- Identification and authentication policy specifies what technologies, equipment, or combination of the two the company will use to ensure that only authorized individuals have access to its data.
- Internet access policy specifies what the company considers ethical and proper use of its Internet access capabilities.
- Campus access policy specifies how the campus users will use the company's data infrastructure.
- Remote access policy specifies how remote users will access the company's data infrastructure.
- Incident handling procedure specifies how the company will create an incident response team and the procedures it will use during and after an incident.

There are three primary characteristics of a good security policy [15]:

- Most important, the policy must be enforceable and it must apply to everyone.
- The policy must be capable of being implemented through system administration procedures and through the publication of acceptable-use guidelines or other appropriate methods.
- The policy must clearly define the areas of responsibility and the roles of users, administrators, and management.

2.2.2 Network Security Devices Overview

Securing network infrastructure is like securing possible entry points of attacks on a country by deploying appropriate defense. Network security starts from authenticating any user. In network security, AAA stands for authentication, authorization and accounting. Authentication refers to the confirmation that a user who is requesting services is a valid user of the network services requested. Authorization refers to the granting of specific types of service to a user, based on their authentication, what services they are requesting, and the current system state. Accounting refers to the tracking of the consumption of network resources by users. This information may be used for management, planning, billing, or other purposes. Once user is authenticated, firewall enforces access policies such as what services are allowed to be accessed by the network users. A firewall is a system or group of systems that enforces an access control policy between two or more networks. Though effective to prevent unauthorized access, firewall fails to check potentially harmful contents such as computer worms being transmitted over the network. An intrusion detection and prevention system (IDPS) helps detect and prevent such malware. IPS also monitors for suspicious network traffic for contents, volume and anomalies to protect the network from attacks such as denial of service. Communication between two hosts using the network or two site to site networks could be encrypted to maintain privacy. A VPN is an encrypted connection between private networks over a public network such as the Internet. Management, monitoring and reporting is the important components to use these devices efficiently. In Figure 4, the components of network security are simulated as the ones in a bank.

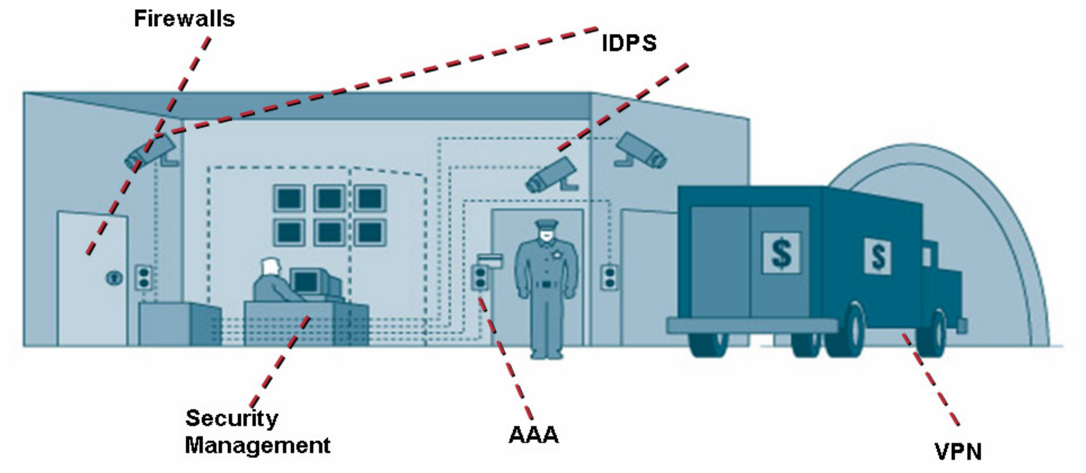


Figure 4 Components of Network Security

2.2.3 Network Attacks and Undesired Traffic

It is important to understand the nature of potential attacks on network security. Network attacks and malicious traffic can infiltrate network and cause real damage to business by causing a huge drain on network resources and security. Protecting against these attacks and going on the offensive against "bad" IP addresses are the main aims. The preparation against attacks and their prevention requires knowledge and understanding of possible threats which networks and information systems are facing.

Undesired traffic is generally limited to the delivery of spam and traffic created by worms, botnets, and other malicious attacks. In some countries, this definition can expand to such traffic as non-local VoIP or video streaming services, which are squelched to create a market for the in-house services of the same type. In some organizations, Peer-to-peer applications such that E-donkey, Emule, Gnutella, Kazaa and instant messaging applications such that MSN Messenger, Yahoo Messenger, AOL, Skype are considered as undesired traffic.

A brief introduction to the nature and methodology of a typical computer attack is described. Computer attacks generally follow a five step approach as described below [16].

- **Reconnaissance:** Before launching an attack, attackers conduct detailed reconnaissance to collect information about their prey. This process typically involves using low-technology reconnaissance, general web searches, the "whois" database, and the Domain Name System (DNS).
- **Scanning:** The attacker, equipped with information about the infrastructure of the victim's network, begins scanning the victim's systems looking for vulnerabilities and openings. At the end of this phase the attacker will have gained valuable information about the victim's network, including lists of phone numbers with modems, addresses of live hosts, network topology, open ports, and firewall rule sets. There are a number of powerful freely available network scanners on the web for that purpose.
- **Gaining access:** If the attacker is a legitimate user of the system, then most likely he/she will attempt to gain access using operating system and application attacks. If the attacker is an outsider, then the attack is most likely to be through the network.
    - **Operating System and Application attacks:** This approach depends on the skill of the attacker with simple inexperienced attackers, usually referred to as "script kiddies," utilizing prepackaged exploits to more advanced attackers using highly systematic approaches. Generally, variations of the operating system buffer overflow attacks are used to gain root access to the target. In addition, password guessing is used as an entry point to log in to the target.
    - **Network attacks**: Network sniffers are usually utilized by attackers to collect Data Link Layer (DLL) information from all computers on the same subnet. A sniffer is a program that gathers traffic data from the network. In addition, other techniques like spoofing and session

hijacking are used typically with a freely available tool called Netcat. In some cases, attackers are not interested in gaining access to the network, but would just like to prevent legitimate users from accessing its resources. In this case, the attackers would launch a DoS attack to consume the resources of the network and computers, especially web servers.

- **Maintaining Access:** Now that the attackers have gained access to the target system, they need to maintain this access. Many techniques are utilized here, based on malicious software such as Trojan horses, Backdoors, and RootKits. A Trojan horse is a program that looks like it has a benign or beneficial purpose, but is actually implementing some malicious function. A backdoor is a malicious computer program or particular means that provide the attacker with unauthorized remote access to a compromised system exploiting vulnerabilities of installed software and bypassing normal authentication. A Backdoor works in background and hides from the user. A RootKit is a tool that allows an attacker to maintain superuser access on a machine by modifying system software.
- **Covering Tracks:** Some of the main techniques used by hackers to hide their tracks are Backdoor programs and RootKits. Beyond that, attackers will also attempt to modify system logs and create covert channels, which are hidden communication paths used to transmit data so that the victim will not see the data.

There are various types of attacks. The type used often depends on the motive and targets of the attacker. The types of threats can be classified for instance according to the following list:

i) Unstructured threats: These threats primarily consist of random hackers using various common tools, such as malicious shell scripts, password crackers, credit card number generators, and dialer daemons. Although hackers in this category may have malicious intent, many are more interested in the intellectual challenge of cracking safeguards than in creating havoc.

ii) **Structured threats:** These threats are created by hackers who are more highly motivated and technically competent. Typically, such hackers act alone or in small groups to understand, develop, and use sophisticated hacking techniques to penetrate unsuspecting businesses. These groups are often involved in the major fraud and theft cases reported to law enforcement agencies. Occasionally, such hackers are hired by organized crime, industry competitors, or state-sponsored intelligence collection organizations.

iii) **External threats:** These threats consist of structured and unstructured threats originating from an external source. These threats may have malicious and destructive intent, or they may simply be errors that generate a threat.

iv) **Internal threats:** These threats typically involve disgruntled former or current employees. Although internal threats may seem more ominous than threats from external sources, security measures are available for reducing vulnerabilities to internal threats and responding when attacks occur.

There are four types of network attacks:

a) **Reconnaissance attacks:** An intruder attempts to discover and map systems, services, and vulnerabilities. Reconnaissance refers to the overall act of learning info about a target network by using publicly available info and applications such that DNS queries, ping sweeps, port scans, sniffing. Reconnaissance is the unauthorized discovery and mapping of systems, services, or vulnerabilities. Characteristics of the applications running on the hosts can lead to specific info on that application such that version, patches etc.

   Reconnaissance attacks can consist of the following:

   - **Packet sniffers:** A packet sniffer is a software application that uses a network adapter card in promiscuous mode to capture all network packets.
   - **Port scans and Ping sweeps:** Port scans and ping sweeps are typically applications built to run various tests against a host or device in order to identify vulnerable services. The information is gathered

by examining IP addressing and port or banner data from both TCP and UDP ports.

- **Internet Information Queries:** DNS queries can reveal such information as who owns a particular domain and what addresses have been assigned to that domain. Ping sweeps of the addresses revealed by the DNS queries can present a picture of the live hosts in a particular environment. After such a list is generated, port scanning tools can cycle through all well-known ports to provide a complete list of all services running on the hosts discovered by the ping sweep. Finally, the hackers can examine the characteristics of the applications that are running on the hosts. This step can lead to specific information that is useful when the hacker attempts to compromise that service.

b) **Access attacks:** An intruder attacks networks or systems to retrieve data, gain access, or escalate access privileges. Network access attacks are attacks in which an intruder gains unauthorized access to resources on a network and uses this access to carry out any number of unauthorized or even illegal activities. As soon as this access has been gained, it can even be used to carry out denial of service attacks against networks.

Network access can be further subdivided into two categories:

- **Data access:** In a data access attack, an attacker gains unauthorized access to the data contained on devices sitting on a network. The attacker can just as easily be an internal user as an outside one. Privileged data often is available to users who have access to a certain kind of information on the network, but for whom this access is limited. These users do not have the necessary privileges to access certain confidential information. However, they gain this access by illegally increasing their privilege level. This is known as privilege escalation. Unauthorized data retrieval involves reading, writing, copying, or moving files that are not intended to be accessible to the intruder.

- **System access:** System access is a more aggravated form of network access attack in which an attacker gains access to system resources and devices. This access can include running programs on the system and using its resources to do things as commanded by the attacker. The attacker can also get access to network devices such as cameras, printers, and storage devices.

Access attacks can consist of the following:

- **Password attacks:** Password attacks can be implemented using several methods, including brute-force attacks, Trojan horse programs, IP spoofing, and packet sniffers.
- **Trust exploitation:** Although it is not an attack in itself, trust exploitation refers to an individual’s taking advantage of a trust relationship within a network. It is not a name of an attack, but it is a description of how an attack works. As its name suggests, it is when a trusted source on a network takes advantage of its position. This vulnerability applies to systems as well so that if one device on a segment is compromised, it can lead to other systems being compromised on the same segment. This vulnerability is a key reason that SAFE divides the network into logical groups based on access needs. SAFE is a security methodology developed by Cisco and partner products. SAFE uses a defense-in-depth and modular approach to security design.
- **Port redirection:** Port redirection is a specific case of trust exploitation. Essentially, this is a tunneling type of attack. In this case, an attacker uses a compromised host to relay traffic passed through an open port on a firewall or in a router's access lists that would normally be denied.
- **Man-in-the-middle attacks:** A man-in-the-middle attack requires that the attacker have access to network packets that come across the

network. Such attacks are often implemented using network packet sniffers and routing and transport protocols. The possible uses of such attacks are theft of information, hijacking of an ongoing session to gain access to your internal network resources, traffic analysis to derive information about your network and its users, denial of service, corruption of transmitted data, and introduction of new information into network sessions.

c) **Denial of service (DoS) attacks** [17]: An intruder attacks network in a way that damages or corrupts computer system or denies people access to networks, systems, or services. DoS attacks are undoubtedly a very serious problem in the Internet, whose impact has been well demonstrated in the computer network literature. The main aim of a DoS attacks is the disruption of services by attempting to limit access to a machine or service instead of subverting the service itself. This kind of attack aims at rendering a network incapable of providing normal service by targeting either the network's bandwidth or its connectivity

   DoS attacks can be classified into five categories based on the attacked protocol level:

   - **Network Device Level:** DoS attacks in the Network Device Level include attacks that might be caused either by taking advantage of bugs or weaknesses in software or by trying to exhaust the hardware resources of network devices. One example is buffer overrun error in the password checking routine.
   - **OS Level:** Attacks take advantage of the ways operating systems implement protocols. One example of this category is the Ping of Death attack.
   - **Application Level:** Attacks try to settle a machine or a service out of order either, by taking advantage of specific bugs in network applications that are running on the target host or by using such applications to drain the resources of the victim. One example is the

finger bomb. A malicious user could cause the finger routine to be recursively executed on the hostname, potentially exhausting the resources of the host.

- **Data Flood:** An attacker attempts to use the bandwidth available to a network, host or device at its greatest extent, by sending massive quantities of data and so causing it to process extremely large amounts of data. One example is, an attacker could attempt to use up the available bandwidth of a network by simply bombarding the targeted victim with normal, but meaningless packets with spoofed source addresses.
- **Protocol Feature Attacks:** For example several attacks exploit the fact that IP source addresses can be spoofed. Several types of DoS attacks have focused on DNS, and many of these involve attacking DNS cache on name servers. An attacker who owns a name server may coerce a victim name server into caching false records by querying the victim about the attacker's own site. A vulnerable victim name server would then refer to the rogue server and cache the answer.

Distributed Denial of Service (DDoS) attacks are the "next generation" of DoS attacks on the Internet. UDP and TCP SYN flooding, Internet Control Message Protocol (ICMP) echo request floods, and ICMP directed broadcasts (also known as smurf attacks) are similar. Victims of DDoS attacks experience packet flooding from many different sources, possibly spoofed IP source addresses that bring their network connectivity to a grinding halt. In the past, the typical DoS attack involved a single attacker's attempt to flood a target host with packets. With DDoS tools, an attacker can conduct the same attack using thousands of systems. A DDoS attack uses many computers to launch a coordinated DoS attack against one or more targets. Using client/server technology, the perpetrator is able to multiply the effectiveness of the DoS significantly by harnessing the resources of multiple unwitting accomplice computers, which serve as attack platforms. The DDoS attack is the most advanced form of DoS attacks. It is distinguished from other attacks by its ability to deploy its weapons in a ‘‘distributed’’ way over the Internet and to aggregate these forces to create lethal traffic.

A DDoS Attack is composed of four elements:

- The real attacker.
- The handlers or masters, which are compromised hosts with a special program running on them, capable of controlling multiple agents.
- The attack daemon agents or zombie hosts who are compromised hosts that are running a special program and are responsible for generating a stream of packets towards the intended victim. Those machines are commonly external to the victim’s own network, to avoid efficient response from the victim, and external to the network of the attacker, to avoid liability if the attack is traced back.
- A victim or target host.

d) **Worms, viruses, and Trojan horses:** Malicious software is inserted onto a host in order to damage a system, corrupt a system, replicate itself, or denies services or access to networks, systems, or services.

The primary vulnerabilities for end-user workstations are worm, virus, and Trojan horse attacks.

- A worm executes arbitrary code and installs copies of itself in the infected computer’s memory, which infects other hosts.
- A virus is malicious software that is attached to another program to execute a particular unwanted function on a user's workstation.
- A Trojan horse is different only in that the entire application was written to look like something else, when in fact it is an attack tool.

Application-layer attacks can be implemented using several different methods:

- One of the most common methods is exploiting well-known weaknesses in software commonly found on servers, such as Sendmail, PostScript, and FTP. By exploiting these weaknesses, attackers can gain access to a computer with the permissions of the account running the application, which is usually a privileged, system-level account.
- Trojan horse program attacks are implemented using programs that an attacker substitutes for common programs. These programs may provide all the functionality that the normal program provides, but also include other features that are known to the attacker, such as monitoring login attempts to capture user account and password information. These programs can capture sensitive information and distribute it back to the attacker. They can also modify application functionality, such as applying a blind carbon copy to all e-mail messages so that the attacker can read all of your organization's e-mail. One of the oldest forms of application-layer attacks is a Trojan horse program that displays a screen, banner, or prompt that the user believes is the valid login sequence. The program then captures the information that the user enters and stores or e-mails it to the attacker. Next, the program either forwards the information to the normal login process (normally impossible on modern systems) or simply sends an expected error to the user (for example, Bad Username/Password Combination), exits, and starts the normal login sequence. The user, believing that he or she has incorrectly entered the password (a common mistake experienced by everyone), re-enters the information and is allowed access.
- One of the newest forms of application-layer attacks exploits the openness of several new technologies: the HTML specification, web browser functionality, and HTTP. These attacks, which include Java applets and ActiveX controls, involve passing harmful programs across the network and loading them through a user's browser.

The following are some measures to take to reduce risks for application-layer attacks:

- Read operating system and network log files or have them analyzed: It is important to review all logs and take action accordingly.
- Subscribe to mailing lists that publicize vulnerabilities: Most application and operating system vulnerabilities are published on the Web by various sources.
- Keep your operating system and applications current with the latest patches: Always test patches and fixes in a non-production environment. This practice prevents downtime and keeps errors from being generated unnecessarily.
- Use IDPS's to scan for known attacks, monitor and log attacks, and in some cases, prevent attacks: The use of IDPS's can be essential to identify security threats and mitigate some of those threats. In most cases, it can be done automatically.

The results of attack categories are defined as follows:

- **Corruption of Information:** any unauthorized alteration of files stored on a host computer or data in transit across a network.
- **Disclosure of Information:** the dissemination of information to anyone who is not authorized to access that information.
- **Theft of Service:** the unauthorized use of computer or network services without degrading the service to other users.
- **Denial of Service:** the intentional degradation or blocking of computer or network Resources.

2.3 INTRUSION DETECTION AND PREVENTION SYSTEMS

In the last decade, the networking revolution has finally come of age. More than ever before, we see that the Internet is changing computing as we know it. The possibilities and opportunities are limitless; unfortunately, so too are the risks and chances of malicious intrusions. Intrusion is a set of actions that attempt to compromise the integrity, confidentiality, or availability of any resource on a computing platform. Intrusion is the act of a person or proxy attempting to break into or misuse one's system in violation of an established policy [18]. In addition, intrusion threat is the potential possibility of a deliberate unauthorized attempt to access information, manipulate information, or render a system unreliable or unusable. With this perspective, there are different aspects to an intrusion, each of which is significant to a full analysis and response. These aspects include [19]:

- **Risk:** Accidental or unpredictable exposure of information, or violation of operations integrity due to the malfunction of hardware or incomplete or incorrect software design.
- **Vulnerability:** A known or suspected flaw in the hardware or software or operation of system that exposes the system to penetration or its information to accidental disclosure.
- **Attack:** A specific formulation or execution of a plan to carry out a threat.
- **Penetration:** A successful attack - the ability to obtain unauthorized (undetected) access to files and programs or the control state of a computer system.

Intrusion detection has been an active field of research for about three decades, starting in 1980, with the publication of John Anderson's Computer Security Threat Monitoring and Surveillance, which was one of the earliest papers in the field. Dorothy Denning's seminal paper, "An Intrusion Detection Model," published in 1987, provided a methodological framework that inspired many researchers and laid the groundwork for commercial products.

It is very important that the security mechanisms of a system are designed so as to prevent unauthorized access to system resources and data. Intrusion attempts or a threat are the potential possibility of a deliberate unauthorized attempt to access information, manipulate information or render a system unreliable or unusable [20].

An intrusion detection and prevention system inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. If an intrusion occurs, an alarm is generated. The alarm can be in several forms such as a line in the logs, an email or a Short Message Service (SMS) to the system administrator.

There are several ways to categorize an IDPS:

- **Misuse detection vs. anomaly detection:** in misuse detection, the intrusion detection and prevention system analyzes the information it gathers and compares it to large databases of attack signatures. Essentially, the intrusion detection and prevention system looks for a specific attack that has already been documented. Like a virus detection system, misuse detection software is only as good as the database of attack signatures that it uses to compare packets against. In anomaly detection, the system administrator defines the baseline, or normal, state of the network's traffic load, breakdown, protocol, and typical packet size. The anomaly detector monitors network segments to compare their state to the normal baseline and look for anomalies.
- **Network-based vs. host-based systems:** in a network-based system, the individual packets flowing through a network are analyzed. The system can detect malicious packets that are designed to be overlooked by a firewall's simplistic filtering rules. In a host-based system, the system examines at the activity on each individual computer or host.
- **Passive system vs. reactive system:** in a passive system, the intrusion detection and prevention system detects a potential security breach, logs the information and signals an alert. In a reactive system, the intrusion detection and prevention system responds to the suspicious activity by logging off a

user or by reprogramming the firewall to block network traffic from the suspected malicious source.

Though they both relate to network security, an IDPS differs from a firewall in that a firewall looks out for intrusions in order to stop them from happening. The firewall limits the access between networks. An IDPS evaluates a suspected intrusion once it has taken place and signals an alarm or prevent it. An IDPS also watches for attacks that originate from within a system.

Computer attacks happen every day. The operating systems, software, and networking stacks we use have many vulnerabilities, some of which are intrinsic to them. Attackers scan our systems and networks for these vulnerabilities and break into the system, eventually getting access to private data, such as bank accounts and credit card information. Attackers also use the machines they break into to launch new attacks on other machines. IDPSs are designed to detect and respond to intrusions and alert security officers to any possible attack on networks and systems.

In computer and network security, standard approaches to intrusion detection and response attempt to detect and prevent individual attacks. Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) are real-time software for risk assessment by monitoring for suspicious activity at the network and system layer.

Building a completely secure system seems like a way to handle intrusions. However, in practice it is not possible, because of many reasons. Software market is growing rapidly and bug free software is still a dream. Using cryptographic methods also have own problems, such that passwords can be cracked, entire crypt-systems can be broken.

2.3.1 Techniques

IDPS that monitor computer systems and networks, analyze them for signs of security policy violation, and respond accordingly, is based on one of the following approaches:

- Anomaly Detection Systems
- Misuse Detection Systems
- Hybrid Systems

2.3.1.1 Anomaly Detection Systems

Anomaly detection systems flag observed activities that deviate significantly from the established normal usage profiles as anomalies, i.e., possible intrusions. For example, the normal profile of a server may contain the averaged frequencies of some system commands used in his or her login sessions. If for a session that is being monitored the frequencies are significantly lower or higher, then an anomaly alarm will be raised. The main advantage of anomaly detection is that it does not require prior knowledge of intrusion and can thus detect new intrusions. The main disadvantage is that it may not be able to describe what the attack is and may have high false positive rate. Anomaly detection technique assumes that all intrusive activities are necessarily anomalous [21]. However, we find two different possibilities:

- Anomalous activities that are not intrusive are flagged as intrusive (false positive)
- Events are not flagged intrusive, though they actually are (false negative)

The main issues in detection systems are the selection of threshold levels so that the percentage of false positives and false negatives are inconsiderable. Anomaly detection relies on detecting behaviors that are abnormal with respect to some normal standards. Many anomaly detection systems and approaches have been developed to detect the faint signs of DDoS attacks [22]. Although detection of novel attacks and not needing for priori knowledge of security flows are the advantages of anomaly detection, it has disadvantages. High false alarm rates due to the fixed user profile distribution assumptions and it is less effective in dynamic environments.

2.3.1.2 Misuse Detection Systems

Misuse detection systems use patterns of well known attacks or weak spots of the system to match and identify known intrusions. For example, a signature rule for the "guessing password attack" can be "there are more than 4 failed login attempts within 2 minutes". The main advantage of misuse detection is that it can accurately and efficiently detect instances of known attacks. The main disadvantage is that it lacks the ability to detect the truly innovative (i.e., newly invented) attacks.

The concept behind misuse detection schemes is that there are ways to represent attacks in the form of a pattern or a signature so that even variations of the same attack can be detected. This means that these systems are not unlike virus detection systems, they can detect many or all known attack patterns, but they are of little use for as yet unknown attack methods. An interesting point to note is that anomaly detection systems try to detect the complement of bad behavior but misuse detection systems try to recognize known bad behavior.

High detection accuracy and very low false alarm rate are the advantages of misuse detection. However, building the signatures is difficult and time consuming process. Also, misuse detection method can not detect unknown intrusions.

2.3.1.3 Hybrid Detection Systems

Hybrid detection systems are using both of the methods, anomaly detection and misuse detection. Although, most of the commercial products were using misuse detection methods, they are using hybrid detection system nowadays.

As the networks become faster and faster, the emerging requirement is to improve the performance of the network intrusion detection/preventions systems to keep up with the increased network throughput. In high-speed networks, it is very difficult for the traditional IDPS to capture all the packets, not to mention the complicated intrusion analysis. When data processing cannot keep pace with the speed and the throughput of networks, it is often the case that the packets not analyzed in time are dropped. These dropped packets may have hostile data with attack signatures, which causes a high false negative rate in the intrusion detection/prevention system. In addition, some attackers can overload the network with large numbers of packets so as to bring on denial of service attack or ID evasion. Therefore, it is necessary to develop efficient intrusion detection techniques [23].

2.3.2 Historical Survey and Academic Solutions

Though the public awareness of the whole area of "intrusion detection'' seems to have been more recent, it is certainly not a new area of inquiry. In fact, it has been an area of concern for most of what we know of modern computers. There have been a number of important milestones in the brief history of Intrusion Detection Systems.

In the mid to late 1960's, as time sharing systems emerged, controlling access to computer resources became a concern. In 1970's, The Department of Defense (DoD) Ware Report pointed out the need for computer security [24]. A number of systems were designed and implemented using security kernel architectures in the mid 1970's. In the late 1970's, Tiger Teams began to evaluate the security of various systems. Tiger teams are also known as red teams, ethical hackers, penetration testers, and intrusion testers. They are people hired to demonstrate vulnerabilities in systems by exploiting vulnerabilities.

In 1980, Anderson first proposed that audit trails should be used to monitor threats. The importance of such data had not been comprehended at that time and all the available system security procedures were focused on denying access to sensitive data from an unauthorized source [25]. In 1983, "The Department of Defense Trusted Computer System Evaluation Criteria" the "orange book" was published and provided a set of criteria for evaluating computer security control effectiveness [26]. In 1987, an abstract model of an Intrusion Detection Expert System (IDES) was presented by Denning. Her paper was the first to propose the concept of intrusion detection as a solution to the problem of providing a sense of security in computer systems [27]. The Internet Worm program, which infected thousands of machines and disrupted normal activities for several days, was detected in 1988 [28].

In the next decade, a new concept was introduced by Heberlein, named NSM (Network Security Monitor). It is now called Network Intrusion Detector (NID). Instead of examining the audit trails of a host computer system, suspicious behavior was detected by passively monitoring the network traffic in a LAN [29]. In 1992, the intrusion detection model proposed by Denning was refined and IDES prototype system was created. This system was designed to detect intrusion attempts with adaptation to gradual changes in behavior to minimize false alarms [30]. In 1993, a different idea NADIR (Network Anomaly Detection and Intrusion Reporter) was introduced. The audit data from multiple hosts were collected and aggregated in order to detect coordinated attacks against a set of hosts [31]. An improved version of IDES, called NIDES (Next-generation Intrusion Detection Expert System) [32] was introduced. In the mid 1990's, The use of autonomous agents was suggested in order to improve the scalability, maintainability, efficiency and fault tolerance of an IDS [33]. The implementation of GrIDS (A Graph-Based Intrusion Detection System) addressed the scalability deficiencies in most contemporary intrusion detection systems [34]. An innovative approach to intrusion detection was offered, by incorporating informational retrieval techniques into intrusion detection tools [35].

In the last decade, a novel approach is using Hidden Markov Models (HMM) to detect complex Internet attacks [36]. Bayesian paradigm for designing intrusion detection systems was introduced. Bayes' rule provides a means of combining competing intrusion detection methods such as anomaly detection and pattern recognition. Bayesian methods present evidence of intrusion as probabilities, which are easy for human fraud investigators to interpret [37]. Anomaly intrusion detection can be conducted based on linear prediction and Markov chain model. Linear prediction is employed to extract features from system calls sequences of the privileged processes which are used to make up of the character database of those processes, and then the Markov chain model is founded based on those features [38]. A Radial-basis-function neural network detector for Distributed-Denial-of-Service (DDoS) attacks in public networks based on statistical features estimated in short-time window analysis of the incoming data packets is presented and evaluated [39]. A Rule-Based High-Performance Network Intrusion Detection System (RHPNIDS) is constructed for high-speed network which improves the system performance based on advanced approaches to both data collection and data analysis [40]. Coordinated Intrusion Prevention System (CIPS), includes Parallel Firewall (PFW), Flow Detection (FD) and Multiple Intrusion Detection System (MIDS) against large-scale or coordinated intrusions [41]. Multi-threaded implementations of signature-based Network Intrusion Detection System (NIDS) sensors are constructed. A number of novel designs for a multi-threaded NIDS sensor are presented in detail [42]. Two data mining methodologies-Artificial Neural Networks (ANNs) and Support Vector Machine (SVM) and two encoding methods-simple frequency-based scheme are used to detect potential system intrusions in [43]. Pattern matching is one of critical parts of Network Intrusion Detection Systems (NIDS). A multi-pattern matching method should efficiently handle a large number of patterns with a wide range of pattern lengths and noncase-sensitive pattern matches. [44].

Especially, in the last decade, a lot of research studies are being done to cover the emerging need for security analysis techniques that can keep up with the increased network throughout.

2.3.3 IDPS Implementations for High Speed Networks

The netflow based anomaly IDS in [45] aims for the detection of ping sweeps, DoS attacks and port scans using netflow data taken from routers and switches. In addition, statistical traffic modeling is used in [45]. Scalable attack detection refers scaling behavioral network detection to very high speeds [46]. Wide variety of DoS and scanning attacks are examined and it is shown that several categories (bandwidth based, claim-and-hold, port-scanning) can be scalably detected. In addition to existing approaches for scalable attack detection, a novel data structure called partial completion filters (PCFs) that can detect claim-and-hold attacks scalably in the network is examined.

Besides the software implementations for IDPS, there are also hardware implementations. String content matching is a major component of intrusion detection, constituting 46% of execution time (Figure 5) [47][35]. It is a key to achieve good performance for the signature based intrusion detection and prevention systems in network security strategies.

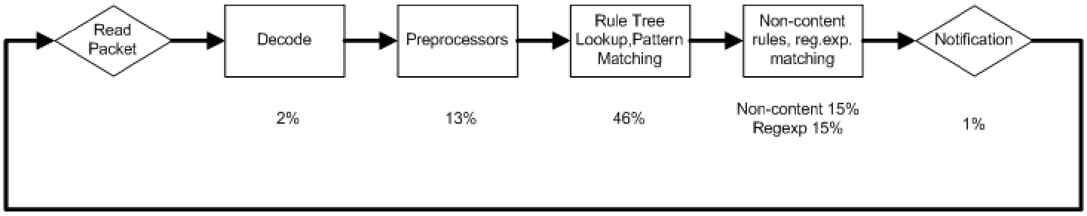


Figure 5 Components of Intrusion Detection [47]

The idea of using reconfigurable resources along with a conventional processor has lead to research in the area of reconfigurable computing. The main goal is to take advantage of the capabilities and features of both resources. While the processor deals with all the general purpose computations, the reconfigurable hardware acts as a specialized co-processor that deals with specialized operations where the parallelism, regularity of computation, like repeated comparisons of strings, can be exploited by creating custom operators, pipelines, and interconnection pathways. With a dedicated FPGA (Field Programmable Gate Array) or ASIC (Application Specific Integrated Circuit), a TCAM-based (Ternary Content Addressable Memory) co-processor can match correlative patterns and multiple patterns, thus improving the overall performance of Network Intrusion Detection and Prevention Systems.

Almost all of the commercial products use ASIC architecture in their hardware design. Despite these hardware architecture techniques, the performance of systems using pattern matching is still lower than that of backbone devices.

2.3.4 Benchmarking of the Products in the Market

Traditional network-based IDS products account for 57% of spending in 1Q04, with host IDS/IPS next at 27% and in-line network based IDS/IPS closing in on host at 16%. In 2007 the numbers will be dramatically different, due to strong growth in the in-line IPS hardware market: 48% traditional network based, 24% host, and 28% in-line network based as seen in Figure 6 [48], Infonetics Research is the premiere international market research and consulting firm specializing in data networking and telecom).

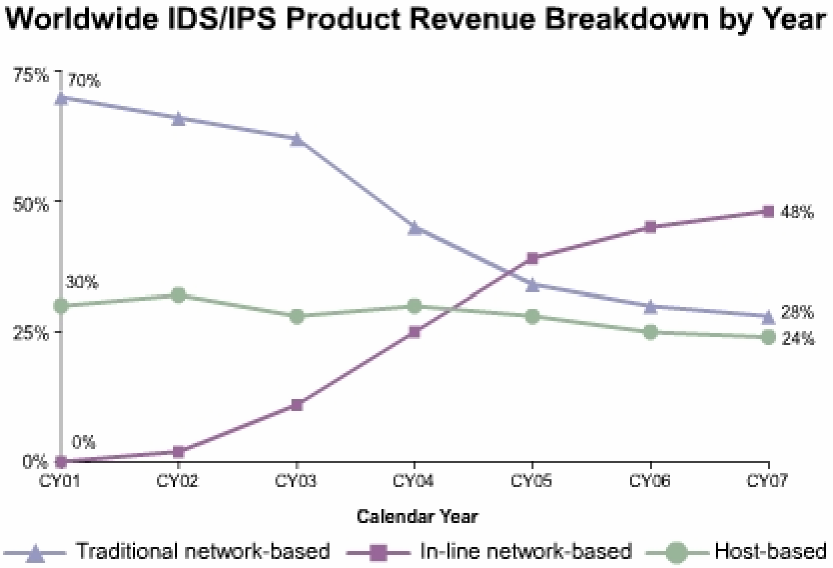


Figure 6 Worldwide IDS/IPS Product Revenue Breakdown by year [48]

Intrusion Detection and Prevention Products tracks network-based intrusion detection and prevention hardware and software, and host-based intrusion detection and prevention products. Network based IDS/IPS hardware grows faster than all other categories as seen in Figure 7.

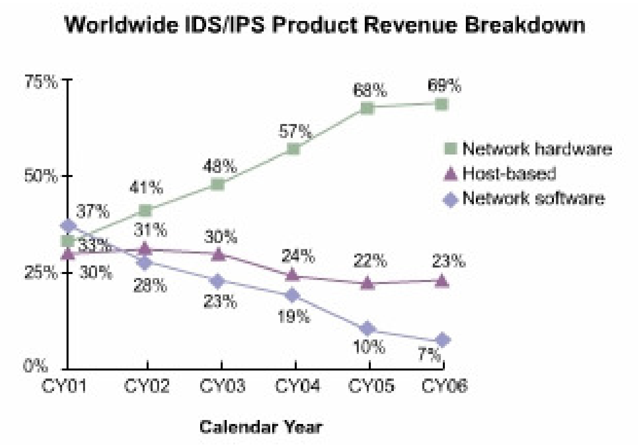


Figure 7 Worldwide IDS/IPS Product Revenue Breakdown [48]

Worldwide network security revenue is forecast to hit $5.3 billion in 2010. VPN and firewall appliances and software make up 87% of network security revenue in 1Q07; IDS/IPS makes up the balance at 13%. So it is about $600M. The worldwide network based IDS/IPS product manufacturer revenue is shown in Figure 8 and total revenue is shown in Figure 9 [48].

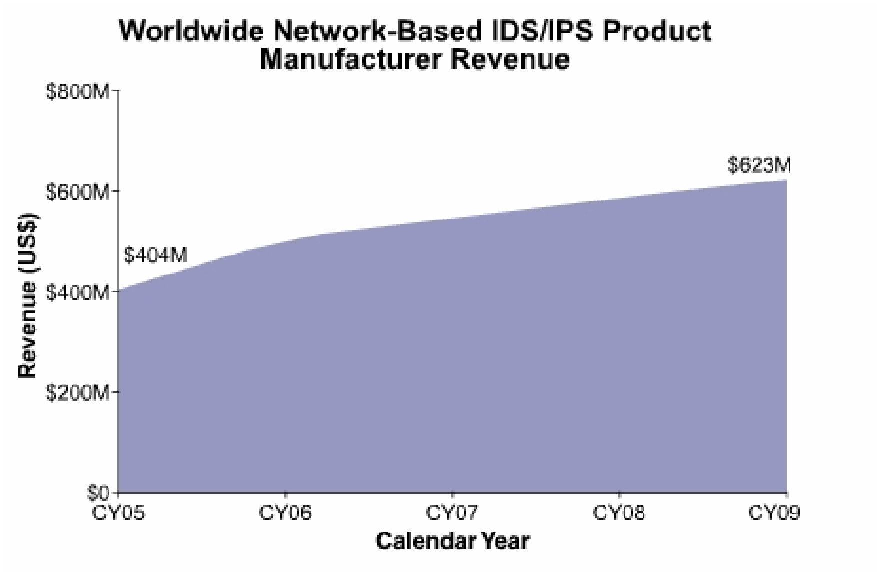


Figure 8 Worldwide IDS/IPS Product Manufacturer Revenue [48]

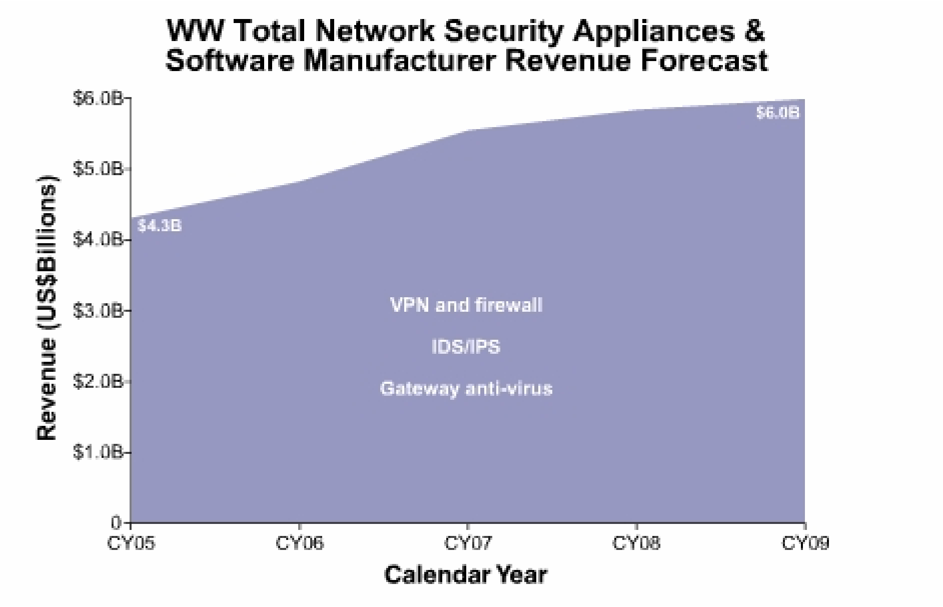


Figure 9 Worldwide Total Network Security Appliances & Software Manufacturer Revenue Forecast [48]

According to the Gartner reports and Infonetics reports, there are five leading IPS vendors: ISS, Cisco, McAfee, 3Com Tipping Point, Juniper. They provide security products and services that preemptively protect enterprise organizations against Internet threats. There are different intrusion prevention models. Throughput changes in between 10Mbps to 15Gbps and maximum concurrent connections are about 1 million. Vendors usually achieve these values in the laboratory environment by using large packets, 1518 bytes. Therefore, the throughput of the devices is not so high by considering the packet per second values. On the other hand, the performance tests are done using packets with low attack ratio. In real networks, packet size distributions [49] and the attack packet ratio vary.

2.4 TRAFFIC ANALYSIS AND MONITORING TOOLS

The Network Tools that are used to develop intrusion detection technique in the solution in this thesis are introduced in this section. Traffic analysis and control tools, network monitoring tools, security tools and attack generation tools are briefly described. Traffic analysis and network monitoring are necessary and important part of contemporary advanced network management and administration. Attack generation tools are important to simulate an attack and determine a general network protection procedure.

2.4.1 Traffic Analysis Tools

"Traffic Analysis Tools" is a collective term used to describe a variety of software-based analytical procedures and methodologies that support different aspects of traffic and transportation analyses. Traffic analysis tools include methodologies such as sketch-planning, travel demand modeling, traffic signal optimization, and traffic simulation. They are used around the world for troubleshooting, analysis, software and protocol development, and education. They work like a protocol analyzer. In common industry usage, a sniffer (with lower case "s") is a program that monitors and analyzes network traffic, detecting bottlenecks and problems. Using this information, a network manager can keep traffic flowing efficiently. A sniffer can also be used legitimately or illegitimately to capture data being transmitted on a network. A network router or a switch reads every packet of data passed to it, determining whether it is intended for a destination within the router's own network or whether it should be passed further along the Internet. A router, switch with a sniffer, however, may be able to read the data in the packet as well as the source and destination addresses. Sniffers are often used on academic networks. There are different kinds of sniffer programs that can be used.

Ethereal

Ethereal, an Open Source Software released under the GNU General Public License is used as sniffer program to capture all of the incoming and outgoing packets.

Ethereal is used by network professionals for troubleshooting, analysis, software and protocol development, and education. It has all of the standard features that are expected in a protocol analyzer. Its open source license allows talented experts in the networking community to add enhancements. It runs on all popular computing platforms, including Unix, Linux, and Windows [50].

Like other protocol analyzers**,** Ethereal**'s** main window shows 3 views of a packet. It shows a summary line, briefly describing what the packet is. A packet details display is shown, allowing you to drill down to exact protocol or field that you interested in. Finally, a hex dump shows exactly what the packet looks like when it goes over the wire.

In addition, Ethereal has some features that make it unique. It can assemble all the packets in a TCP conversation and show the ASCII (or EBCDIC, or hex) data in that conversation. Display filters in Ethereal are very powerful; more fields are filterable in Ethereal, and the syntax to create filters is rich. As Ethereal progresses, expect more and more protocol fields to be allowed in display filters. Packet capturing is performed with the pcap library which is application programming interface for packet capturing.

2.4.2 Flow Based Traffic Analysis Tools

Flow-based measurements are the major traffic information sources at the network level. The high-speed wide area network environment with non-trivial topology (possibly dynamically changing) transporting wide spectrum and significant amount of traffic is the object of interest.

2.4.2.1 Netflow

Netflow is a Cisco Internetworking Operating System (IOS) application that provides statistics on packets flowing through the routing devices in the network [51]. Netflow is an embedded instrumentation within Cisco IOS Software to characterize network operation. It is critical to understand how the network is behaving including:

- Application and Network Usage

- Network productivity and utilization of network resources
- The impact of changes on the network
- Network anomaly and security vulnerabilities
- Long term compliance issues

Cisco IOS Netflow helps to understand who, what, when, where, and how network traffic is flowing. Netflow is completely transparent to the existing network.

IP flow is defined as unidirectional stream of packets between a given source and destination. Each packet data is forwarded within a router or switch is examined for a set of IP packet attributes. These attributes are the IP packet identity or fingerprint of the packet and determine if the packet is unique or similar to other packets. Traditionally, an IP Flow is based on a set of 5 and up to 7 IP packet attributes. IP packet attributes used by Netflow:

- IP Source address
- IP destination address
- Source Port
- Destination Port
- Layer 3 protocol type
- Class of Service
- Router or switch interface

FLOW-TOOLS

Flow-tools are library and collections of programs used to collect, send, process and generate reports from Netflow data [52]. The tools are used together on a single server. For large deployments, tools can be distributed to multiple servers. Flow data is collected and stored by default in host byte order.

Typical flow analysis information found in a Netflow data record includes:

- Source and destination IP address
- Source and destination TCP/User Datagram Protocol (UDP) ports

- Type of service (ToS)
- Packet and byte counts
- Start and end timestamps
- Input and output interface numbers
- TCP flags and encapsulated protocol (TCP/UDP)
- Routing information (next-hop address, source autonomous system (AS) number, destination AS number, source prefix mask, destination prefix mask)

Some of the programs included in the flow-tools distribution are:

flow-capture - Collect, compress, store, and manage disk space for exported flows from a router.

flow-report - Generate reports for Netflow data sets. Reports include source/destination IP pairs, source/destination AS, and top talkers. Over 50 reports are currently supported.

flow-send - Send data over the network using the Netflow protocol.

flow-receive - Receive exports using the Netflow protocol without storing to disk like flow-capture.

flow-gen - Generate test data.

flow-dscan - Simple tool for detecting some types of network scanning and Denial of Service attacks.

flow-merge - Merge flow files in chronoligical order.

flow-split - Split flow files into smaller files based on size, time, or tags.

FLOWSCAN

FlowScan analyzes and reports on Internet Protocol (IP) flow data exported by routers [53]. Consisting of Perl scripts and modules, FlowScan binds together a flow collection engine, a high performance database (Round Robin Database - RRD), and a visualization tool (RRDtool). FlowScan produces graph images that provide a continuous, near real-time view of the network border traffic.

FlowScan analyzes and reports on Netflow format data (indigenous to Cisco routers) collected using CAIDA's cflowd flow tool. FlowScan examines flow data and maintains counters reflecting what was found. Counter values are stored using RRDtool, a database system for time-series data. Finally, FlowScan uses visualization capabilities of both RRDtool and other front-ends to report on the processed flow data.

RRDTool

RRDtool (Round Robin Database tool) is a system to store and display time-series data (e.g. network bandwidth, machine-room temperature, server load average, or even the height of surfing waves on La Jolla Shores) [54]. It stores the data in a very compact way, aggregating at stepwise coarser granularity as it archives further back in time, so as to maintain manageable archive size, RRDtool presents useful graphs by processing the data to enforce a certain data density. RRDtool can be used either via simple wrapper scripts (from shell or Perl) or via user-friendly frontends that poll network devices. Tobi Oetiker developed part of RRDtool during his summer 1999 sabbatical with CAIDA. He continues to support it from his home institution, the Swiss Federal Institute of Technology.

2.4.2.2 NBAR

NBAR can determine which protocols and applications are currently running on a network. NBAR includes the Protocol Discovery feature that provides an easy way of discovering application protocols operating on an interface so that appropriate QoS policies can be developed and applied. With Protocol Discovery, networkers can discover any protocol traffic supported by NBAR and obtain statistics associated with that protocol [55].

Mission critical applications including ERP and workforce optimization applications can be intelligently identified and classified using Network Based Application

Recognition (NBAR). Once these mission critical applications are classified they can be guaranteed a minimum amount of bandwidth, policy routed, and marked for preferential treatment. Non-critical applications including Internet gaming applications and MP3 file sharing applications can also be classified using NBAR and marked for best effort service, policed, or blocked as required.

2.4.3 Network Monitoring Tools

There are many tools that provide affordable and easy-to-use network monitoring solutions. The simple network management protocol (SNMP) forms part of the internet protocol suite as defined by the Internet Engineering Task Force (IETF) [56]. SNMP is used by network management systems to monitor network-attached devices for conditions that warrant administrative attention. It consists of a set of standards for network management, including an Application Layer protocol, a database schema, and a set of data objects.

SNMP exposes management data in the form of variables on the managed systems, which describe the system configuration. These variables can then be queried (and sometimes set) by managing applications.

SNMP agents on the switches and router send traps to a SNMP trap collector [57] when something goes wrong. The SNMP Agents on the devices are configured by the way of its notification and target MIBs to send traps to the collector on Collector-Analyzer machine in case of events, such as interface is going up or down, signaling congestion to adjust failure detection. General Public License program SNMP Trap Collector saves all incoming SNMP traps to an SQL database (PostgreSQL) and then converts them to readable events which can be viewed and acknowledged via a web browser.

Syslog is a standard for forwarding log messages in an IP network. The syslog protocol is a client - server type protocol: the syslog sender sends a small textual message (less than 1024 bytes) to the syslog receiver. The receiver is commonly called "Syslogd", "Syslog daemon" or "Syslog server". Syslog messages can be sent via UDP and/or TCP. Often the data is sent in cleartext; however, an SSL wrapper such as Stunnel, sslio or sslwrap can be used to provide for a layer of encryption through SSL/TLS.

Syslog is typically used for computer system management and security auditing. While it has a number of shortcomings, Syslog is supported by a wide variety of devices and receivers across multiple platforms. Because of this, Syslog can be used to integrate log data from many different types of systems into a central repository. Syslog is now standardized within the Syslog working group of the IETF [58].

Syslog is a method to collect messages from devices to a server running a Syslog daemon. Logging to a central Syslog server helps in aggregation of logs and alerts. A Syslog service simply accepts messages, and stores them in files or prints them according to a simple configuration file. This form of logging is the best available for devices because it can provide protected long-term storage for logs. This is useful both in routine troubleshooting and in incident handling.

2.4.4 Security and Traffic Control Tools

As the traffic on the network is monitored and analyzed, packets must be filtered by security tools and shaped according to their priorities.

2.4.4.1 Iptables

Netfilter is a framework that provides a set of hooks within the Linux kernel for intercepting and manipulating network packets [59]. The best-known component on top of netfilter is the firewall which filters packets, but the hooks are also used by other components which perform network address translation, stateful tracking and packet enqueueing to user space. The name Netfilter also refers to the name of the project that provides a set of firewalling tools for Linux. These components are usually loadable kernel modules, although the project also offers a set of userspace tools and libraries. Backward compatibility support for ipchains has been recently removed.

A packet filter is a piece of software which looks at the *header* of packets as they pass through, and decides the fate of the entire packet. It might decide to **deny** the packet (ie. discard the packet as if it had never received it), **accept** the packet (ie. let the packet go through), or **reject** the packet (like deny, but tell the source of the packet that it has done so).

Iptables is the name of the user space tool by which administrators create rules for the packet filtering and NAT modules. While technically iptables is merely the tool which controls the packet filtering and NAT components within the kernel, the name iptables is often used to refer to the entire infrastructure, including netfilter, connection tracking and NAT, as well as the tool itself. iptables is a standard part of all modern Linux distributions.

2.4.4.2 Ebtables

The ebtables program is a filtering tool for a bridging firewall [60]. The filtering is focused on the Link Layer Ethernet frame fields. Apart from filtering, it also gives the ability to alter the Ethernet MAC addresses and implement a brouter. The ebtables utility enables basic Ethernet frame filtering on a Linux bridge, logging, MAC NAT and routing. It only provides basic IP filtering; the full-fledged IP filtering on a Linux Bridge is done with iptables. The so-called bridge-nf code makes iptables see the bridged IP packets and enables transparent IP NAT. The firewalling tools iptables and ebtables can be used together and are complementary. Ebtables tries to provide the bridge firewalling that iptables cannot provide, namely the filtering of non-IP traffic. Although {ip,ip6,arp}tables can see bridged traffic, the only way to have full access to the Ethernet header is by using ebtables.

Main features of ebtables are:

- Usage analogous to iptables.
- Ethernet filtering.
- MAC NAT: ability to alter the MAC Ethernet source and destination address. This can be useful in some very strange setups (a real-life example is available).
- Brouting: decide which traffic to bridge between two interfaces and which traffic to route between the same two interfaces. The two interfaces belong to

a logical bridge device but have their own IP address and can belong to a different subnet.

- Pass packets to userspace programs, using netlink sockets (the ulog watcher).

The capabilities of ebtables are:

- Ethernet protocol filtering.
- MAC address filtering.
- Simple IP header filtering.
- ARP header filtering.
- 802.1Q VLAN filtering.
- In/Out interface filtering (logical and physical device).
- MAC address nat.
- Logging.
- Frame counters.
- Ability to add, delete and insert rules; flush chains; zero counters.
- Brouter facility.
- Ability to atomically load a complete table, containing the rules you made, into the kernel. See the man page and the examples section.
- Support for user defined chains.
- Support for marking frames and matching marked frames.

2.4.4.3 Traffic Control

Packet-switched networks differ from circuit based networks in one very important regard. A packet-switched network itself is stateless. A circuit-based network (such as a telephone network) must hold state within the network. IP networks are stateless and packet-switched networks by design; in fact, this statelessness is one of the fundamental strengths of IP.

The weakness of this statelessness is the lack of differentiation between types of flows. In simplest terms, traffic control allows an administrator to queue packets differently based on attributes of the packet. It can even be used to simulate the behaviour of a circuit-based network. This introduces statefulness into the stateless network.

When the kernel has several packets to send out over a network device, it has to decide which ones to send first, which ones to delay, and which ones to drop. This is the job of the packet scheduler, and several different algorithms for how to do this "fairly" have been proposed.

With Linux QoS subsystem it is possible to make very flexible traffic control. Traffic control is the name given to the sets of queuing systems and mechanisms by which packets are received and transmitted on a router [61]. This includes deciding which (and whether) packets to accept at what rate on the input of an interface and determining which packets to transmit in what order at what rate on the output of an interface.

In the overwhelming majority of situations, traffic control consists of a single queue which collects entering packets and dequeues them as quickly as the hardware (or underlying device) can accept them. This sort of queue is a FIFO.

Common traffic control solutions

- Limit total bandwidth to a known rate.
- Limit the bandwidth of a particular user, service or client.
- Maximize TCP throughput on an asymmetric link; prioritize transmission of ACK packets.
- Reserve bandwidth for a particular application or user as shown in Figure 10.
- Prefer latency sensitive traffic.
- Managed oversubscribed bandwidth.
- Allow equitable distribution of unreserved bandwidth.
- Ensure that a particular type of traffic is dropped.

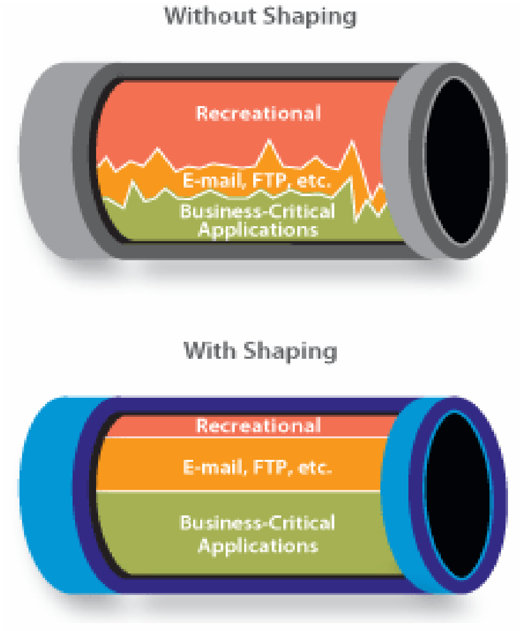


Figure 10 Traffic Shaping [61]

When properly employed, traffic control should lead to more predictable usage of network resources and less volatile contention for these resources. The network then meets the goals of the traffic control configuration. Bulk download traffic can be allocated a reasonable amount of bandwidth even as higher priority interactive traffic is simultaneously serviced. Even low priority data transfer such as mail can be allocated bandwidth without tremendously affecting the other classes of traffic.

In a larger picture, if the traffic control configuration represents policy which has been communicated to the users, then users (and, by extension, applications) know what to expect from the network.

2.4.5 Attack Generation Tools

If you want to stop hackers from invading your network, first you've got to invade their minds. The goal of the ethical hacker is to help the organization take preemptive measures against malicious attacks by attacking the system himself; all the while staying within legal limits. This philosophy stems from the proven practice of trying to catch a thief, by thinking like a thief. As technology advances and organization depend on technology increasingly, information assets have evolved into critical components of survival. The definition of an Ethical Hacker is very similar to a Penetration Tester.

Backtrack [62] and Phlak [63] are used for penetration tests. They are optimized to be used by security penetration testers. Currently they consist of more than 300 different up-to-date tools which are logically structured according to the work flow of security professionals. New technologies and testing techniques are merged in these tools.

BackTrack is the most top rated Linux live distribution focused on penetration testing [62]. Backtrack and PHLAK are known as ethical hackers. Backtrack is based on a Slackware Linux distribution. PHLAK is a Linux distribution based on Morphix [63]. There are more than 300 tools to penetrate to the networks. Metasploit, port scanners, vulnerability scanners, offline/online password attack tools, hash collision, fuzzers, arp spoofing, ip spoofing, dns spoofing tools, man in the middle attack tools, Sql injection, blind sql injection, Oracle Metacoretex, OSScanner are used. Current hacker tools and technologies warrant reengineering to address cyber crime and homeland security [64]. The Metasploit Project is an open source computer security project which provides information about security vulnerabilities and aids in penetration testing. The goal is to provide useful information to people who perform penetration testing, IDS signature development, and exploit research. This site was created to fill the gaps in the information publicly available on various exploitation techniques and to create a useful resource for exploit developers. The tools and information on this project are provided for legal security research and testing purposes only [65][66][67].

CHAPTER 3

THE PROPOSED DETECTION ENGINE

"If you know the enemy and yourself, you need not fear the result of a hundred battles. If you know yourself but not the enemy, for every victory gained you will also suffer a defeat. If you know neither the enemy nor yourself, you will succumb in every battle."

Sun Tzu in the Art of War

IDPS performs deep packet inspection on all packets including clear (non-malicious, non-intrusion packets) packets as well as packets that belong to L2-L4 attacks. Majority of the processing overhead and resource consumption of the IDPS come from analyzing the intrusion packets and performing necessary actions when an intrusion is detected. Our approach is based on the hypothesis that majority of the all of the common L2-L4 attacks can be detected by inspecting the flow level characteristics without performing deep packet inspection. We note that, deep packet inspection is required for L5-L7 type of attacks which do not have L2-L4 reconnaissance phase. We expect that, detecting the attacks by using simple flow information before IDPS have to examine them decreases the resource consumption of the IDPS hence provides a scalable solution for IDPS for high-speed networks. Detected attacks can be dropped automatically if the intrusion detection server in our approach is on the path of the intrusion traffic (in in-line mode). Otherwise, server makes denial requests to the switch, router or firewall.

The approach in this thesis is implemented and tested using two different networks. A laboratory network is used for preliminary characterization of the attacks and building a first set of rules. Layer 2-4 attacks and undesired traffic behaviors are first characterized in the laboratory network to detect through the complete network traffic using both flow-based and payload inspection based methods such as flows, logs, sniffer and SNMP traps. The discovered characteristics include the usage of TCP and UDP ports at the same time or one after another, connection duration, IP packet size, number of connection, and flow time. Following this first stage, the same characterization method is used for a real network to enhance the characterization and the rule set obtained from the laboratory network. The simplified final rule set is obtained by merging the rules for different types of attacks that expose similar characteristics. This final rule set is tested in the same real network, for experiments and performance evaluation.

Note that, payload inspection is not the component of the intrusion detection technique in our approach. It is only used in the laboratory networks for analysis and verification of the attacks' characteristics. The netflow based anomaly IDS in [34] aims for the detection of ping sweeps, DoS attacks and port scans using netflow data taken from routers and switches. In addition, statistical traffic modeling is used in [45]. In our approach, detection is based on not only using netflow data but also logs and traps. Statistical information gathered from netflow data is correlated with the syslogs and traps. Our approach is entirely general and can be applied to any backbone network. It is also possible to add new attack types to our database. Rather than attempting to detect of all kinds of L2-L4 attacks, certain types of attacks (such as DoS attacks, uncomplicated DDoS attacks, scans, worms, poisoning, spoofing, protocol anomaly attacks, password attacks) are the focus in the solution in this thesis. These attacks affect the performance of the IDPS and network access, since they are the most frequent attacks and the target of them is stopping services.

3.1 FLOW ANALYSIS AND SMALL LABORATORY NETWORK

Analyzing attack behavior in a real network is not so easy. Exposing attacks' flow patterns among the flows of the real network is like looking for a needle in the haystack. When we try to sniff the network, we get about 40 MB file that includes 90 KB packets just for a minute in a real network consisting of 200 clients and 1 Mbps wide area connection for internet. Using this huge amount of data from the real network, could lead to misclassification of the legal and illegal characteristic of the attackers' packets. Therefore, we choose to use a laboratory stage as a starting point of this study. Our aim is to analyze the attacker behavior first in a small laboratory network, then in a larger network. We have to examine just about 40 MB file that includes one hour traffic of the small network.

3.1.1 Flow Patterns

In the small laboratory network, a set of network attacks are staged and examined to extract a preliminary characterization for these attacks. Our aim is identifying any flow pattern that is specific before the attacks or undesired traffic generated or while attack is happening. Small laboratory network and the staged attacks are used to determine a general working procedure on the network and to determine the procedure to compose rule database. Attacks are investigated using netflow analyzer, Syslog server, SNMP collector and sniffer in the laboratory stage small network. Attack by attack all of the parsed logs are examined. A *flow pattern* consists of a set of IP packets and the related devices in the network. The flow pattern is defined using source/destination IP address, number of bytes and packets associated with an IP flow, number of aggregated flows, protocol, TCP/UDP source/destination port, packet length, timeout value, VLAN, MAC address. In addition, the network device and interface information are included in the flow pattern. We first extract the flow patterns of Layer 2-4 attacks. Information gathered from the flows and captured packets on the network are used to expose the attacks' behavior and so flow pattern. Packets are observed in a specific timeslot that the staged attacks are generated in and flow patterns are represented using the exported netflow data. Netflow version 9 flow-record format is described in Appendix A. A given flow goes through a set of *states* that we define as reconnaissance, scanning, gaining access, attack, maintaining access and covering tracks in time. Flow patterns of attacks are expressed using rules based on these states. For instance, we observe that if a packet destined to a port 9996 is following a packet destined to 445, it is most probably a worm. If a packet destined to port 445 is seen in the network, flow pattern is supposed to be in reconnaissance state. Following packets destined to 445 does not change the state. If a packet destined to 9996 is arrived, the flow changes to an attack state.

A rule set is prepared that reflects attack behaviors using flow pattern usage of TCP and UDP ports at the same time or one after another, connection duration, IP packet size, number of connection, flow time, SNMP traps.

3.1.2 Laboratory Network Topology

We set the test bed in Figure 11 to determine the characteristics of the attack and undesired traffic. Characterization is not for the specific design of network topology. Network traffic flow collection, logs, trap collection are important for characterization, but not the specific topology used in our study.

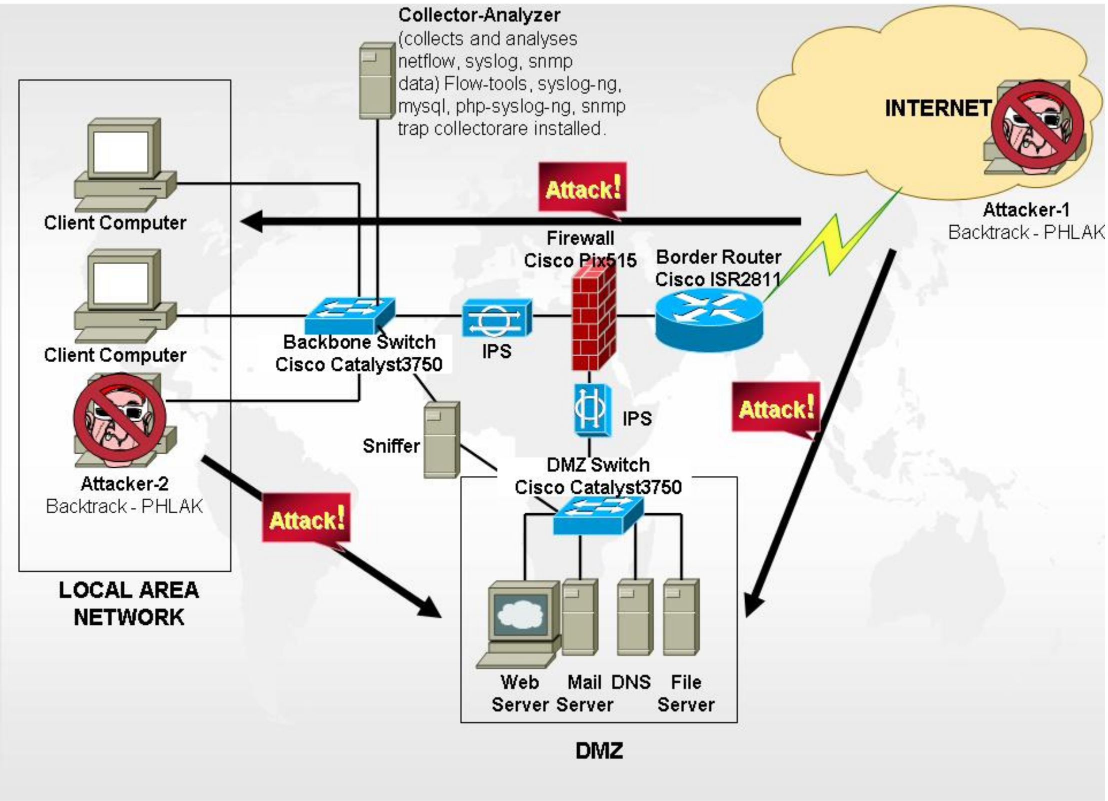


Figure 11 Small Laboratory Network Topology

The small laboratory network consists of a border router, backbone switch, DMZ switch, firewall, intrusion detection and prevention system (IDPS), servers and client computers. Topology design guides of the SANS Institute are used to generate the topology that most of the networks are designed on is for our small laboratory network. Intrusion prevention systems are positioned to analyze the traffic that passes through the firewall. Switches are used to connect servers and clients. Border router is used to make internet connection. Next, we discuss the devices in our laboratory network in detail.

According to the IDPS’s logs, attacks’ behaviors are examined from the stored data in the Collector-Analyzer and Sniffer Server. Sniffer is just used to examine the packet payload and the signature of the IDPS as a part of the verification mechanism. All of the devices and computers are synchronizing their time with NTP server. This is important to track all data, logs that is related with an attack or undesired traffic. To synchronize all data, NTP (Network Time Protocol) server is installed on the Sniffer Server.

Firewall:

The firewall is positioned to separate the network into three segments, internal, DMZ (Demilitarized Zone) for servers and outside. In a network, the most vulnerable hosts to attack are those that provide services to users outside of the LAN, such as e-mail, web and DNS servers. Due to the increased potential of these hosts being compromised, they are placed into their own sub-network. We use the firewall in this topology to collect syslog data. The syslog collected from the firewall depends on the filter configuration since the logs are generated as packet matches to a filter. The access filters defined on the firewall in our topology are shown in Table 1. SANS Institute suggests to deny all packets that are sourced from outside and DMZ to the inside network unless there are servers giving worldwide global service [68]. However, the related filters are defined to permit packets to analyze the packets passing through the firewall.

Table 1 Access Filter Rules

From	To	Destination Service	Action
Inside	Outside	Any	permit
Outside	Inside	Any	permit
Inside	webserver	http,https	permit
Inside	mailserver	smtp, http, pop3, imap, https	permit
Inside	DNS	Domain	permit
Outside	webserver	http,https	Permit
Outside	mailserver	smtp, http, pop3, imap, https	Permit
Outside	DNS	Domain	Permit
DMZ	Inside	Any	Permit
DMZ	Outside	smtp, http, https	Permit

Active-standby stateful firewalls are used for perimeter security. NAT (Network address translation) is configured on the firewalls. NAT is masquerading IP addresses for sharing internet access if IT administrators of the network do not have enough public IP addresses. In addition, IP NAT translation logs are used to track the IP flows. Firewall is configured to send logs in syslog format to the Collector-Analyzer.

Border router:

All of the interfaces of the border router are configured to export the flow traces which are supplied by netflow. In addition, SNMP traps are enabled on the border router. SNMP traps enable an agent on the router or switch to notify the management station of significant events by way of an unsolicited SNMP message. Link status, memory status, CPU status, alert status are traced and traps are generated if there is any problem. For traps that are supported by the router and switch in specific MIBs, refer to the Appendix B. In the configuration of border router, the IP address of the Collector is defined as the destination of the exported netflow data and also as the destination of SNMP traps.

Backbone Switch:

Backbone switch is configured to route the traffic between VLANs (Virtual LAN) and outside network. A VLAN is a group of hosts with a common set of requirements that communicate as if they are attached to the same wire, regardless of their physical location. It can be thought as a broadcast domain. For instance, each department of a university can be associated to different VLAN's. Number of VLANs configured on the switches is not important. It can be just a default VLAN or many VLANs. Collecting the netflow data passing through VLAN or VLAN's is the important operation for us. In the configuration of backbone switch, the IP address of the Collector is defined as the destination of the exported netflow data and also as the destination of SNMP traps. Although it seems that outgoing traffic can be monitored on the router, netflow data on the switch is important to examine the traffic between internal VLANs.

DMZ Switch:

DMZ Switch is configured to send netflow data and SNMP traps to Collector-Analyzer.

Collector-Analyzer, Sniffer:

To collect the data, log and analyze them, two computers are used, which we call Collector-Analyzer and Sniffer. Collector-Analyzer collects syslog, netflow data and SNMP traps directed to it by router, switches, firewall and IDPS and performs correlation of this data.

Sniffer Server captures the packets passing through backbone switch and DMZ switch to verify the attacks detected by our solution are not false positive. Since sniffer software has to run on a server which has high volume disks and Collector-Analyzer needs high memory, two servers are used for different functions. Sniffer Server is connected to the local area backbone switch and DMZ switch. All of the traffic that is passing through the switches is mirrored to the Sniffer Server's ethernet port. Scripts to parse the captured packets are running on Sniffer Server. Sniffer is just used to examine the packet payload and the signature of the IDPS as a part of the verification mechanism

The Collector-Analyzer is used for collecting three different data in the small laboratory network. The processes on the Collector-Analyzer trigger that scripts and get the output of them to analyze. The Collector-Analyzer stores netflow data, syslog data and SNMP traps. On the border router, on the backbone switch and on the DMZ switch netflow feature is enabled which exports flow information about the routed traffic gathered from IP headers of the packets switched through these ports. All data (only some fields of packet headers) going through the router and the switch are cached in device and flows are grouped together into "netflow export" UDP datagram for exporting to collector-Analyzer. By analyzing netflow data, we can identify the cause of congestion, determine the class of service (CoS) for each user and application, and identify the source and destination network for traffic. The netflow data is directed to the Collector-Analyzer. General Public License program

Flow-Tools is setup on Collector-Analyzer to capture and store the data for further analysis. In addition, Syslog-ng, MySQL and Php-syslog-ng are installed on the Collector-Analyzer machine and used to collect the syslog data from the firewall and IDPS. Php-syslog-ng is a front-end graphical user interface for viewing Syslog-ng messages logged to MySQL in real time. Moreover, General Public License program SNMP Trap Collector saves all incoming SNMP traps to an SQL database (PostgreSQL) and then converts them to readable events which can be viewed and acknowledged via a web browser. SNMP traps are generated on the border router, switch, firewall and IDPS. We have the facility for SNMP agents on the switches and router to send traps to a SNMP trap collector when something goes wrong. The SNMP Agents on the devices are configured by the way of its notification and target MIBs to send traps to the collector on Collector-Analyzer machine in case of events, such as interface is going up or down, CPU, memory overload, signaling congestion to adjust failure detection. On the firewall, syslog feature is enabled to send syslogs to Syslog server on Collector-Analyzer.

IDPS:

We use two different leading commercial signature based intrusion detection systems. Their percentage in the market is about 30% (each about ~15%) according to Gartner report in May 2006. They are configured to examine the packets but not to block. In their database about 2300+ signatures are defined and enabled. The event logs of the intrusion prevention systems are monitored on the vendor specific management screen. In addition, we get the detection logs of these two commercial products to the Collector-Analyzer to analyze our syslog data, SNMP traps, netflow data and sniffed data. All of the signatures are logged in syslog format and sent to the Collector-Analyzer. Commercial IDPS's are not the component of the intrusion detection technique in this approach. They are used for analysis and verification of the attacks' characteristics. When a new attack is detected by commercial product, it triggers the process on Collector-Analyzer to analyze the flows related with this attack.

There are four ethernet ports on the IDPS to support in-path (inline) deployments. Each two are paired as a segment to work in inline (in-path) mode. One of the ports of a segment is connected to the inside, the other to the outside. One segment of commercial product IDPS is positioned between the backbone switch and firewall. Another segment is positioned between the DMZ switch and firewall. Time is synchronized among the devices in the network to provide log synchronization. According to the IDPS's logs, attacks' behaviors are examined from the stored data in the Collector-Analyzer and Sniffer.

The network throughput of the firewall is not constant in the network. Since the firewall is installed on the way of Internet and DMZ, the main traffic is passing through it. However, there is still multicast and broadcast traffic and little unicast traffic on the switches. The maximum traffic seen on the firewall is listed in Table 2. Properties of the devices are listed in Table 3.

Table 2 Maximum Traffic seen on the Firewall in the Small Laboratory Network

Firewall Interface	**Direction In**	**Direction Out**
Inside	1620 Kbps	470 Kbps
Outside	780 Kbps	340 Kbps
DMZ	940 Kbps	580 Kbps

Table 3 Small Laboratory Network Device/Software

Device/ Software	Feature	Model	Functionality	Settings	Notes
Border Router	Test Bed Component	Cisco ISR2811	Connects wide area network to local area network. It is located on the perimeter.	12.4.3a enterprise feature set Fast Ethernet serial interfaces	All of the interfaces are configured to export the flow traces which are supplied by netflow. SNMP Traps and syslog are enabled.
Firewall	Test Bed Component	Cisco PIX515	Perimeter Security. Stateful packet filtering.	Version 7.2.1. Four ethernet ports	Device is configured to send SNMP traps and syslog
Backbone Switch	Test Bed Component	Cisco Catalyst 3750	Layer 3 switch. Interconnection between VLAN's. Access control lists are applied.	12.2.25 IP Services feature set 48 port 10/100/1000 switch	Device is configured to send SNMP traps, syslog and netflow.
DMZ Switch	Test Bed Component	Cisco Catalyst 3750	Layer 2 mode. Servers are connected to the DMZ zone through this switch.	12.2.25 IP Services feature set 48 port 10/100/1000 switch	Device is configured to send SNMP traps, syslog and netflow.
Sniffer	Test Bed Component Software	Pentium IV 3.06 GHz. 1 GB ram computer.	Captures the packets passing through backbone switch and DMZ switch.	OS Debian 3.1. Ethereal is installed.	Just for checking the signature based IDPS functionality.
Collector-Analyzer	Test Bed Component Software	Pentium IV 3.06 GHz. 2 GB ram computer.	Collects syslog, netflow and SNMP traps. Makes correlation and optimization.	OS Centos 4. Flow-tools, Syslog-ng, SNMP trap collector are installed. syslog, SNMP, netflow data correlation is done using perl scripts on this machine	Perl scripts and MySQL are used to make correlation.

Table 3 (continued)

Device/ Software	Feature	Model	Functionality	Settings	Notes
Attacker-1	Test Bed Component Software	Standard computer (Backtrack attack tool is used)	Simulate simple attacks	Backtrack is a bootable CD.	It consists of more than 300 different up-to-date tools which are logically structured according to the work flow of security professionals.
Attacker-2	Test Bed Component Software	Standard computer (Backtrack attack tool is used)	Simulate simple attacks	Backtrack is a bootable CD.	It consists of more than 300 different up-to-date tools which are logically structured according to the work flow of security professionals.
IDPS	Analyzer Component	Commercial Product	Signature based IDPS. About +2300 signatures are enabled.	Latest operating system and signatures are installed.	

3.1.3 Attackers and Attack Staging

To simulate the attackers, we use attacker tools and two different computers, one in inside network, other one in the outside network. There are two attackers which are located on the internal and the external segments. The attacks are destined to the victim computers located in DMZ and in internal side. Although the internal or external attacker's mission is the same, we can encounter different logs and so different characteristics. For instance, an external attack can trigger the firewall to send a trap, but we can only discover the same kind of attack starting from inside network and destined to inside network by using the netflow data taken from the switches. Although the external attack can be detected using the netflow data gathered from router, and the logs of firewall, same kind of internal attack can be detected using netflow data gathered from the switches as the internal attack goes through only switches.

Note that, we do not simulate an attacker located in the DMZ. DMZ contains Web Servers, Mail Servers, Domain Name Servers, File Systems, Database Servers, Authentication Servers and it is physically closed to the unknown computers. The operating systems, patches, security tools on the servers located in Demilitarized Zone are controlled by system administrators systematically. In addition, the usage of the Ethernet interfaces of DMZ without permission can be controlled. Hence unlike the external segments with uncontrolled machines or the internal segment which can be used by guest computers, the only way for an attacker to be located in the DMZ is first hacking a computer there. Hence, if the intrusion packets destined to the DMZ are prevented, the existence of an attacker in that zone is impossible. So, an attacker is not simulated in DMZ.

Frequently encountered network attacks are generated to examine the behavior of these attacks such as worm, trojan, DoS, DDoS, portscanner, spoofing, SNMP attack, DNS attack, memory leak vulnerability, TCP UDP floods etc. Running an exhaustive test of all known attacks is very hard. Equivalence partitioning is one of the methods to realize tests [69]. According to this approach, laboratory tests are realized by attempting a representative subset of attacks from each category. There are three main types of Layer 2-4 attacks: reconnaissance, exploits and DoS. Reconnaissance attacks refer to the overall act of learning info about a target network by using publicly available info and applications such that DNS queries (domain owner, assigned IP's), ping sweeps (identify live hosts from the assigned IP's), ports scans (identify services running on the live hosts). In the exploits attack, the intruders use known or unknown hidden features or bugs to gain access to the system. In the DoS attacks, the intruder attempts to crash a system, jam the network links, overload the CPU, exhaust the system resources, or fill up the storage. In the DoS attacks, the intention of the intruder is not trying to retrieve sensitive information, but to simply prevent the system from being usable, jam the network links or crash routers to make the network inaccessible.

Backtrack and PHLAK (Professional Hacker's Linux Assault Kit) are used to generate the attacks. Our attack tools simulates the following Reconnaissance attacks: ping sweeps, network scans, port scans, IP address queries, domain name queries, operating system scans, vulnerability scans, man in the middle attacks, the following exploits attacks: SQL injection, blind SQL injection, Oracle Metacoretex, offline/online password attack tools, metasploit, worms and finally the following DoS attacks; flooding a network, disrupting a server by sending more requests than it can possibly handle, preventing a particular individual from accessing a service, disrupting service, ICMP floods, smurf attack, teardrop attack, IRC attack, SYN floods, P2P attack, mailbomb, arppoison, tcpreset, udpstorm. Staged attacks are injected in the clear-text traffic. About 6% percentage of total traffic is the attacks.

3.1.4 Collecting Data and Constructing the Rules

It is seen that flow pattern of the victim and the attacker are important characteristics to detect Layer 2-4 attacks. Using this information, and the logs of a commercial intrusion detection and prevention system we can identify the attack flows. We collect and analyze data in the laboratory network for four days.

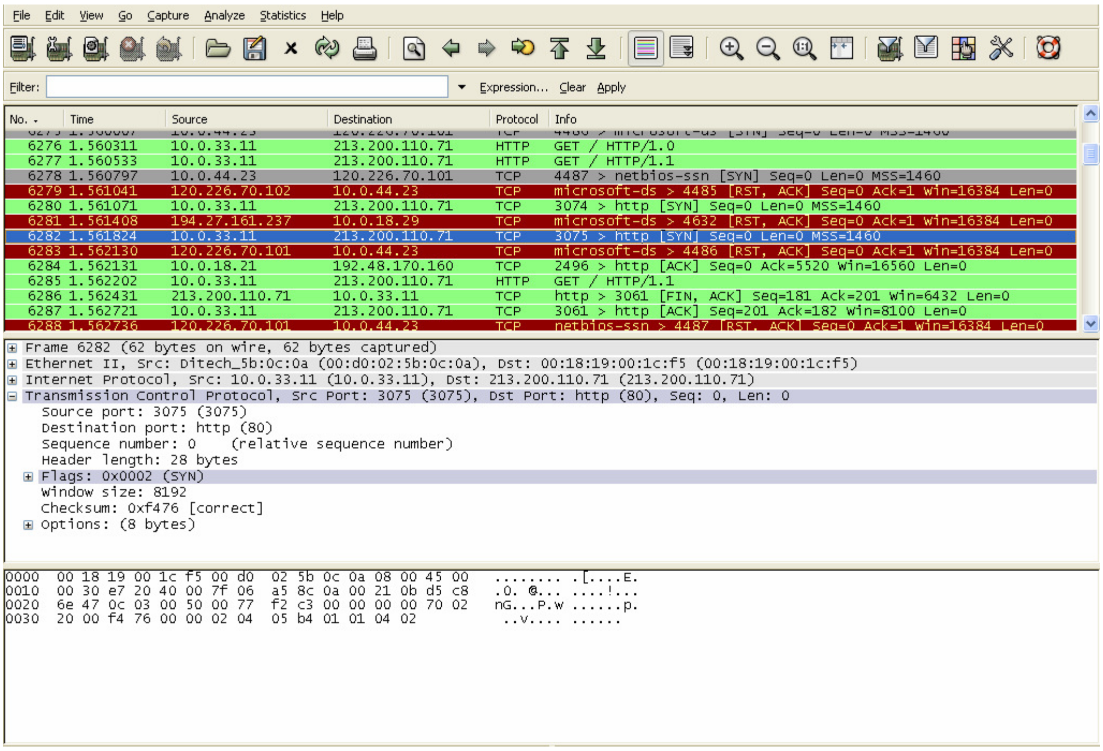
Collecting Data:

The traffic passing through the network contains attacks in clear traffic. While a sample attack is created by our attack generation tools on the attacker PC to the victim PC, the traffic is mirrored to the Sniffer and also the flows are exported to the netflow analyzer from the switch. We collect the syslogs from the firewall and the servers. Netflow data and SNMP traps are taken from border router, backbone switch and DMZ switch. On the other hand commercial intrusion prevention system's logs are taken. All of the logs and alerts are collected and analyzed on the Collector-Analyzer. Perl scripts and MySQL database is used to correlate, to parse and to merge the syslogs, netflow and SNMP data with referring to the commercial products' logs. Flow patterns such as the usage of TCP and UDP ports, connection duration, attacker's behavior before it attacks, and victim's behavior after it is attacked, IP packet size distributions and flow time are examined.

Port mirroring, also known as roving analysis port, is a method of monitoring network traffic that forwards a copy of each incoming and outgoing packet from one port of a network switch to another port where the packet can be studied. Like port mirroring, using VLAN mirroring packets from one VLAN or more VLANs are copied to the specified port on the switch. We are using port or VLAN mirroring as a diagnostic tool and debugging feature, especially when fending off an attack. We configure mirroring by assigning a port that is connected to the IDPS and internal VLANs especially the server VLAN from which to copy all packets and assign another port where those packets will be sent.

Ethereal is working as a packet capturer and also as a protocol analyzer on the port receiving the mirrored data to monitor each segment separately. Data is stored on the Sniffer Server. Ethereal captures and evaluates the data by examining the headers and payloads of the packets without affecting the client on the original port. Ethereal is not the component of the intrusion detection technique in this approach. It is for analysis and verification of the attacks' characteristics. Signatures of the intrusion detection systems and the payloads are compared. Marking the IP address of an attacker and tracking the attacker behavior is done using sniffer software and PERL scripts written on the Sniffer Server. The processes on the Collector-Analyzer trigger that scripts and get the output of them to analyze and verify whether the packet that is supposed to be an intrusion because of the rules is really an attack or not.

A scene from Ethereal software is in Figure 12.



```
No. -  Time      Source           Destination      Protocol  Info
6275 1.560007    10.0.44.23       120.226.70.101   TCP       4486 > microsoft-ds [SYN] Seq=0 Len=0 MSS=1460
6276 1.560311    10.0.33.11       213.200.110.71   HTTP      GET / HTTP/1.0
6277 1.560533    10.0.33.11       213.200.110.71   HTTP      GET / HTTP/1.1
6278 1.560797    10.0.44.23       120.226.70.101   TCP       4487 > netbios-ssn [SYN] Seq=0 Len=0 MSS=1460
6279 1.561041    120.226.70.102   10.0.44.23       TCP       microsoft-ds > 4485 [RST, ACK] Seq=0 Ack=1 Win=16384 Len=0
6280 1.561071    10.0.33.11       213.200.110.71   TCP       3074 > http [SYN] Seq=0 Len=0 MSS=1460
6281 1.561408    194.27.161.237   10.0.18.29       TCP       microsoft-ds > 4632 [RST, ACK] Seq=0 Ack=1 Win=16384 Len=0
6282 1.561824    10.0.33.11       213.200.110.71   TCP       3075 > http [SYN] Seq=0 Len=0 MSS=1460
6283 1.562130    120.226.70.101   10.0.44.23       TCP       microsoft-ds > 4486 [RST, ACK] Seq=0 Ack=1 Win=16384 Len=0
6284 1.562131    10.0.18.21       192.48.170.160   TCP       2496 > http [ACK] Seq=0 Ack=5520 Win=16560 Len=0
6285 1.562202    10.0.33.11       213.200.110.71   HTTP      GET / HTTP/1.1
6286 1.562431    213.200.110.71   10.0.33.11       TCP       http > 3061 [FIN, ACK] Seq=181 Ack=201 Win=6432 Len=0
6287 1.562721    10.0.33.11       213.200.110.71   TCP       3061 > http [ACK] Seq=201 Ack=182 Win=8100 Len=0
6288 1.562736    120.226.70.101   10.0.44.23       TCP       netbios-ssn > 4487 [RST, ACK] Seq=0 Ack=1 Win=16384 Len=0
```

```
⊞ Frame 6282 (62 bytes on wire, 62 bytes captured)
⊞ Ethernet II, Src: Ditech_5b:0c:0a (00:d0:02:5b:0c:0a), Dst: 00:18:19:00:1c:f5 (00:18:19:00:1c:f5)
⊞ Internet Protocol, Src: 10.0.33.11 (10.0.33.11), Dst: 213.200.110.71 (213.200.110.71)
⊟ Transmission Control Protocol, Src Port: 3075 (3075), Dst Port: http (80), Seq: 0, Len: 0
    Source port: 3075 (3075)
    Destination port: http (80)
    Sequence number: 0    (relative sequence number)
    Header length: 28 bytes
  ⊞ Flags: 0x0002 (SYN)
    Window size: 8192
    Checksum: 0xf476 [correct]
  ⊞ Options: (8 bytes)
```

```
0000   00 18 19 00 1c f5 00 d0  02 5b 0c 0a 08 00 45 00   ........ .[....E.
0010   00 30 e7 20 40 00 7f 06  a5 8c 0a 00 21 0b d5 c8   .0. @... ....!...
0020   6e 47 0c 03 00 50 00 77  f2 c3 00 00 00 00 70 02   nG...P.w ......p.
0030   20 00 f4 76 00 00 02 04  05 b4 01 01 04 02          ..v.... ......
```

Figure 12 Ethereal Monitor

Constructing the rules

Reconnaissance attacks collect information about target network before the attack and the suffered victim's flows are passing through the network after. As we deepen the examination of the flows, it is seen that not only an attack is happening at a specific time but also there are some unusual flows that happen before and after an attack happens. Attacker's behavior is defined by the flows that are for reconnaissance and victim’s behavior is defined by the involuntary flows after it is attacked.

When we catch an attack packet using commercial IDPS or sniffer, we get the timestamp of the packet and examine the traffic of victim and attacker in the interval of 10 minutes, 5 minutes before the attack happens and 5 minutes after the attack happens. This 10 minute interval is the starting value to analyze and find an optimum value for the detection accuracy. If we encounter an unusual flow of the attacker or the victim in that interval, we extract the flow pattern and the state changes for that attack as described in Section 3.1.1 to construct a set of rules. For the attacks that can be detected using this information refer to the Appendix C.

All of the flows, logs and traps are stored in a database in Collector-Analyzer. When the sample attack is generated, the flows, logs, traps as well as the CPU and memory usage levels in the devices. 5 minutes before the attack happens and 5 minutes after the attack happens, adding up to a 10 minute total interval, are taken and inserted in another table in the database on Collector-Analyzer. This table contains all of the logs, traps and flows related to the attack. Collector-Analyzer correlates this data and makes state based rule. On the other hand, the logs of IDPS and captured packets are used to verify the rule. The framework for the analysis and correlation method in the small laboratory network is stated in Figure 13.

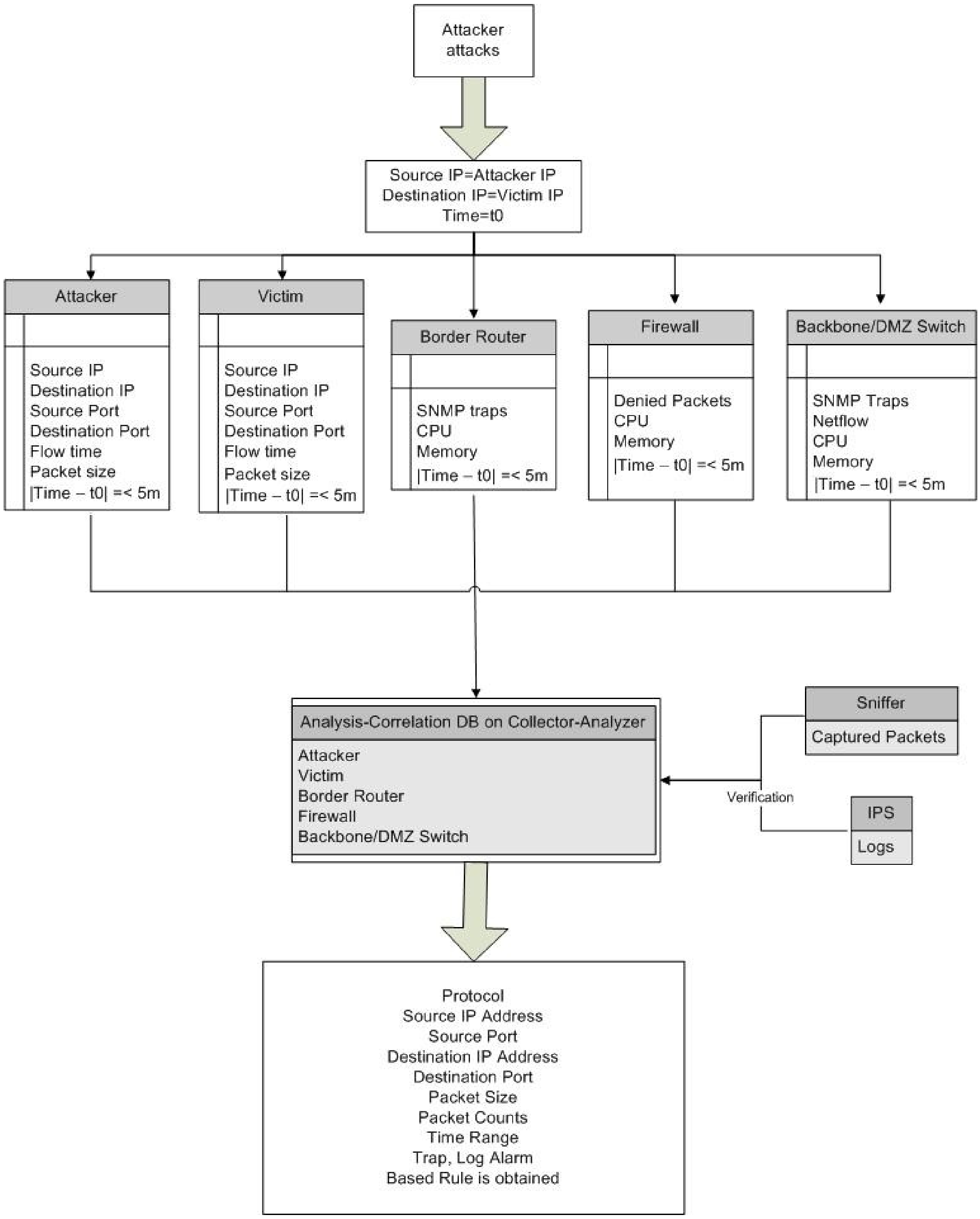


Figure 13 Analysis and Correlation Method

An example of a rule is given in Figure 14. In this example, when a flow that is destined to TCP port 445 on any Windows Operating System installed computer is detected by the Engine, the source IP address, destination IP address and the time information is collected. Normally, this port is used for SMB (Server Message Block) over TCP. SMB protocol is used for file sharing in Windows OS. In the time interval of two minutes, the attacker's and victim's traffic are observed by the Engine. In this interval, it can be said that there is an uncertain attack flow that is being observed. If a new flow destined to victim's TCP port 9996 or attacker's TCP port 5554 at least once or flows destined to 445 at least for 100 times from the victim's address is encountered, it can be said that it is a worm. If not, uncertain attack flow becomes clear text. Hundred is selected to achieve accuracy. Flows destined to port TCP 9996 are checked, because of the worm's characteristic determined in previous observations. When a worm is infected on a machine, worm will open TCP port 9996 and 5554 for malicious activities. Note that, for this type of attack, two minutes is found to be enough to detect it and it is indicated in the rule.

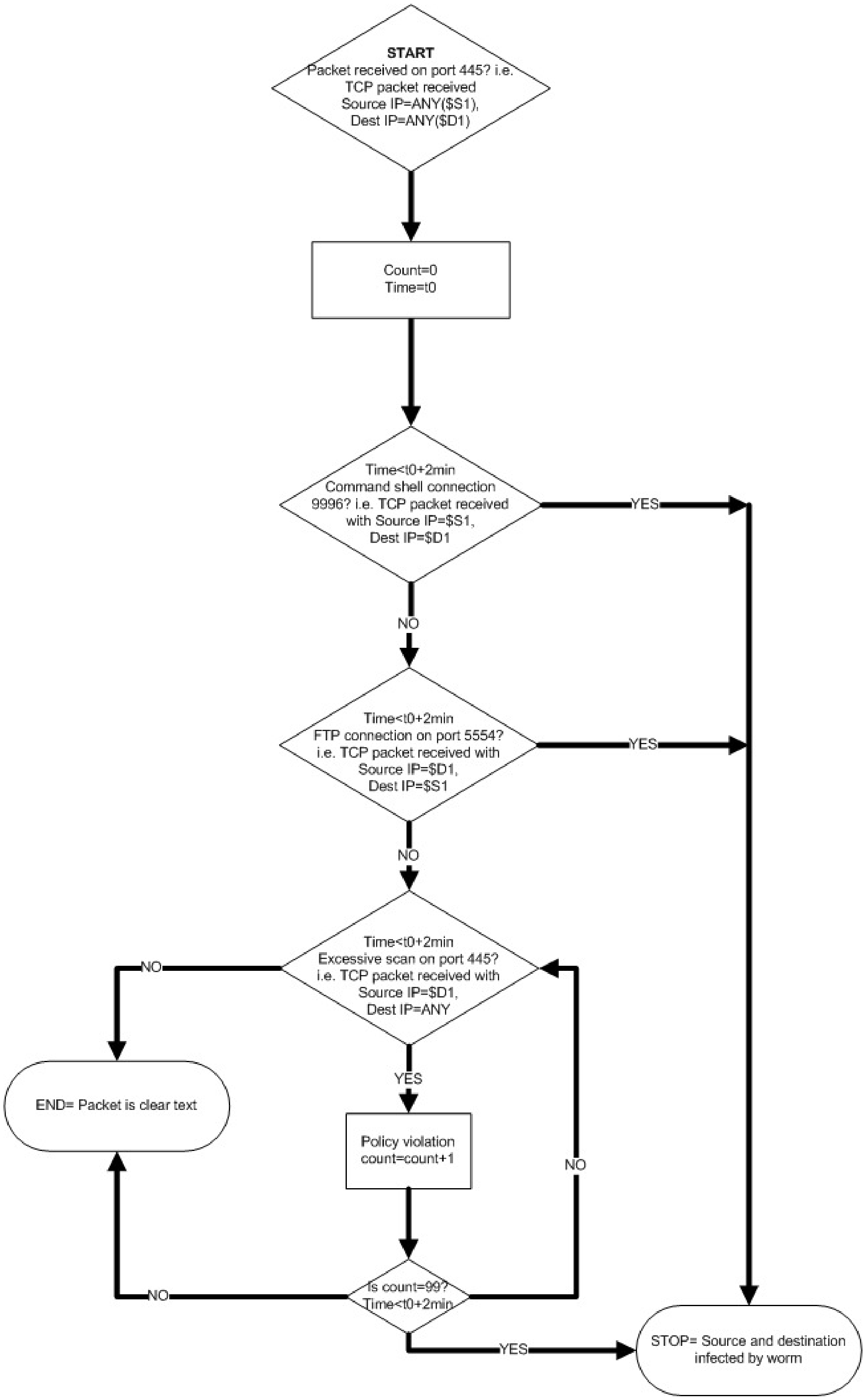


Figure 14 Sample Rule

Database structure and the Detection Engine

We correlate all of the syslog data, netflow data and SNMP traps with respect to the commercial intrusion prevention system's log files. Perl scripts are written that are parsing all of the related data stored in the Sniffer and the Collector-Analyzer at each time an attack occurs. A Perl script running on Collector-Analyzer parses the related files on itself, and triggers the script running on Sniffer, and gets the result of that script to analyze with its results. The system collects the logs and flows for about two weeks and attack by attack all of the parsed logs are examined. At the end of this examination, a *Rule Database* is constructed that reflects attack behaviors.

The Rule Database consists of two kinds of tables, *Attack Rule Tables* and *Uncertain Attack Tables*. Let attack rule A_j be defined with a sequence of FN_j flows. Let $F_{j,i}$ show the flow pattern of the i^{th} flow of A_j where $1 \leq i \leq FN_j$. For all attack rules A_j, if $i \leq FN_j$,$F_{j,i}$ is stored in Attack Rule Table T_i. The number of flows stored in T_i is denoted by TS_i. If $F_{j,i}$ is stored at the k^{th} position in T_i, then $T_i[k]= F_{j,i}$ where $k \leq TS_i$. Let f be the flow pattern of i^{th} flow of a possible attack defined by A_j that could be in the reconnaissance state. If f is stored at the q^{th} position in Uncertain Attack Table UT_i, then $UT_i[q]= f$ where $q \leq UTS_i$ and UTS_i is the size of UT_i. Note that the Attack Rule Tables do not change as long as the rules are not updated while the Uncertain Attack tables contain information related to potential attacks that are under watch at a given time. Let N be the longest flow sequence in the system such that, for all A_j, $FN_j \leq N$. In our study $N = 3$. However, number of flows that indicates an intrusion can increase as new kinds of intrusions appear. In this condition, the tables can be modified. See Figure 15 for the structure of the rule database and the structure of the records of the flow patterns that are stored for each attack.

Flow analysis is performed by our *Detection Engine*. When a flow with flow pattern f passes through the network, the information about the flow is delivered to the Detection Engine. The Detection Engine searches for f in all T_i where $1 \leq i \leq N$. If there is a match $f = F_{j,q}$ in T_{q}, it notes the rule number j and the flow number q that is matched. Otherwise it does not take any action for this flow. An attack rule A_j is matched if all flows that match its flow pattern sequence $F_{j,1}$ to $F_{j,FNj}$ are detected in the network.

The Detection Engine does not give the final decision about an attack as long as the related flow pattern sequence is not completed. From the detection of a flow pattern that matches $F_{j,1}$, until detection of the final flow pattern to match $F_{j,FNj}$, the Detection Engine stores the matching flow patterns in Uncertain Attack Tables UT_1 to UT_{FNj-1}. The continuity of the flow pattern sequence for the uncertain attacks is maintained by checking if the flow pattern matching $F_{j,(q-1)}$ is already in $UT_{(q-1)}$ when the flow pattern matching $F_{j,q}$ is received. Rules can have different expectancy durations. Flows in the Uncertain Attacks tables are cleared as the time to live values of the flows are reached. The flowchart for the Detection Engine is shown in Figure 16.

Note that the rules for different attacks can have common flow sequences. If the incoming flow matches with a flow of two different rules, we have to process this flow for each type of rules. Rules are merged to decrease the common flows of the rules and hence decrease the number of rules. The merged rules cover one attack and its' variants and its' similar attacks. For instance, we can use 4 distinct rules for web server, mail server, RDP, telnet server password attack. However, the rules are merged and processed as a single rule. Their common behavior is making connection to a specific port, 10 times per minute.

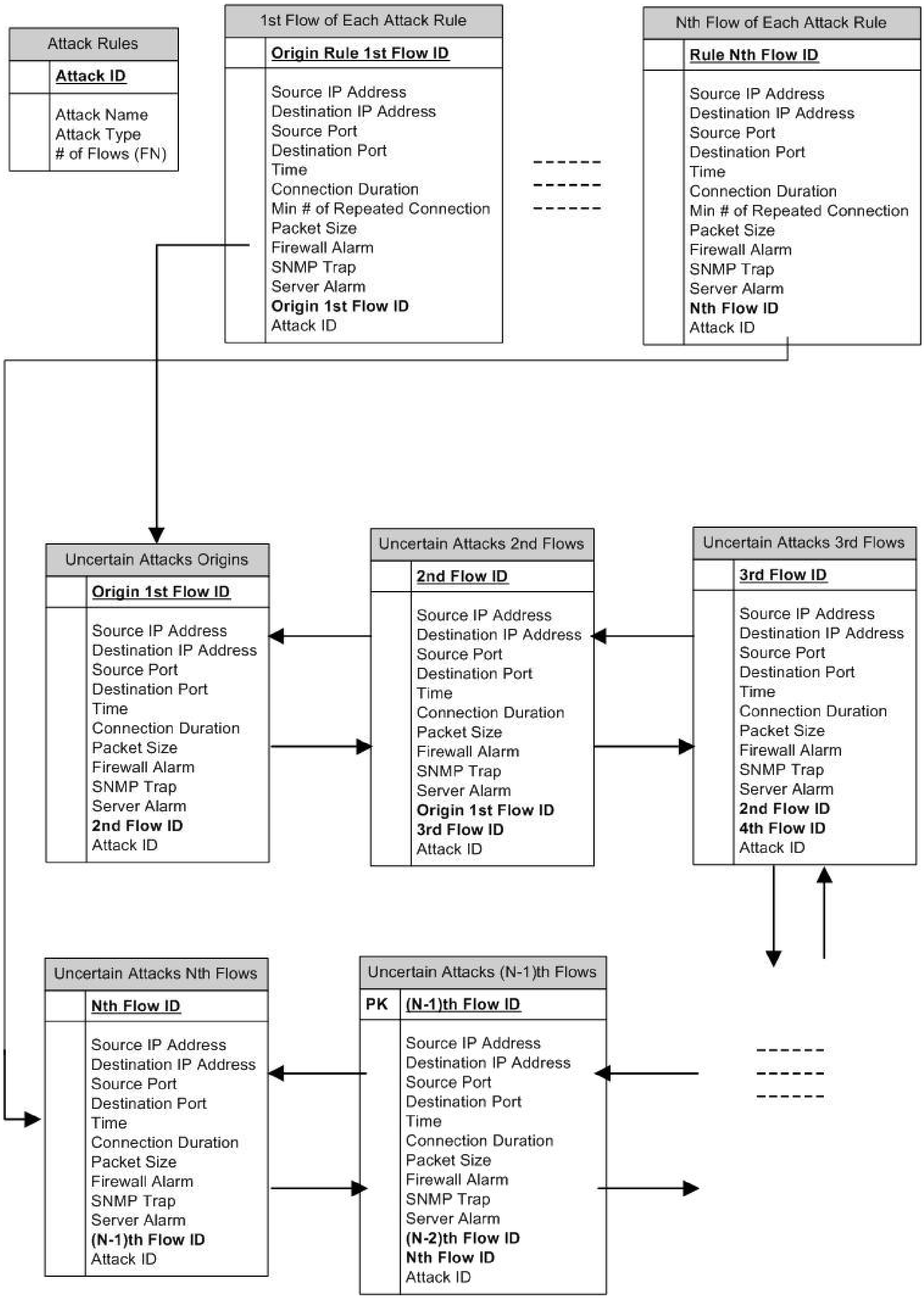


Figure 15 Database Model

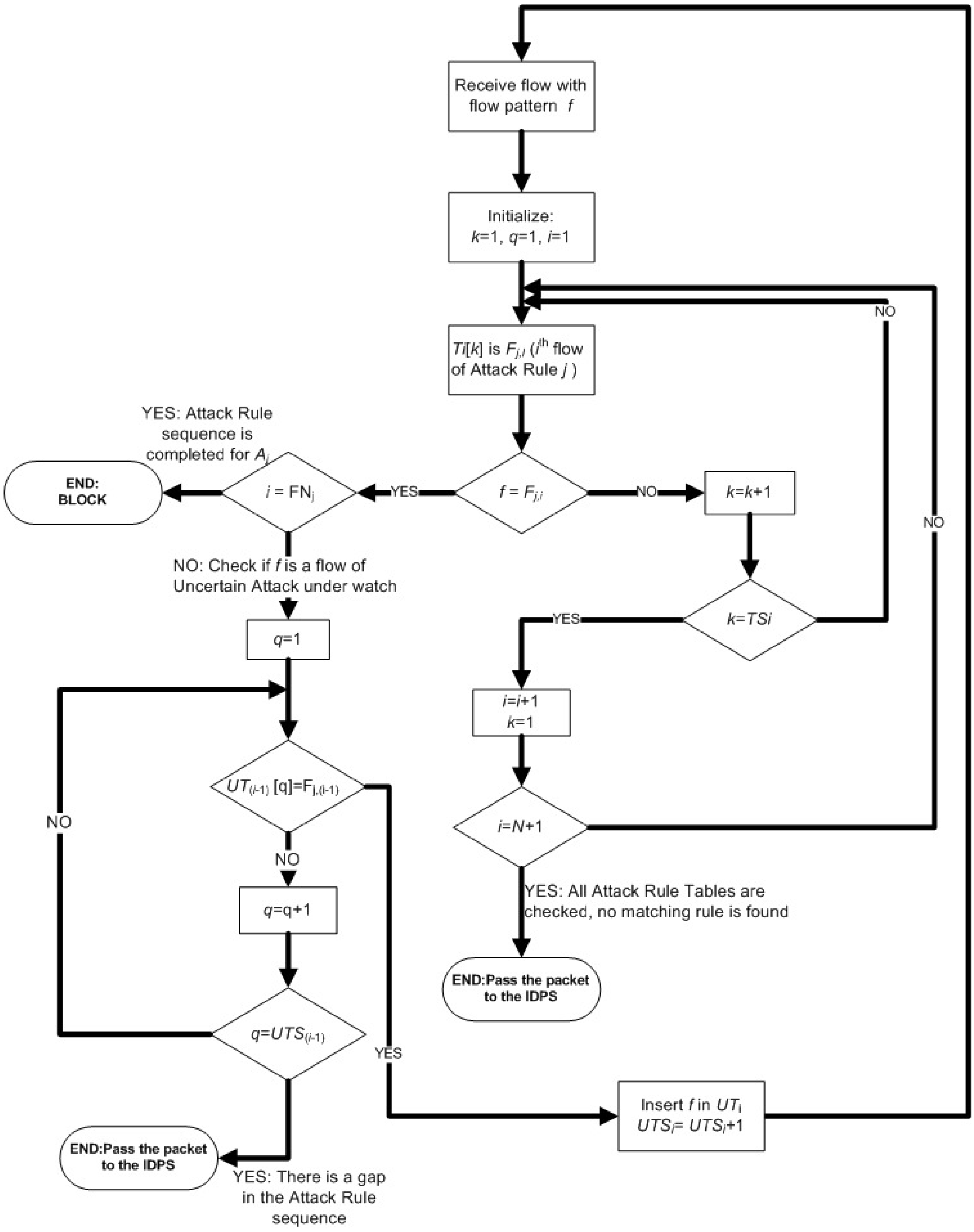

Receive flow with flow pattern f
Initialize: k=1, q=1, i=1
$Ti[k]$ is $F_{j,i}$ (i^{th} flow of Attack Rule j)
$f = F_{j,i}$
NO
k=k+1
k=TSi
NO
NO
YES
YES
i = FN$_j$
YES: Attack Rule sequence is completed for A_j
END: BLOCK
NO: Check if f is a flow of Uncertain Attack under watch
q=1
$UT_{(i-1)}$ [q]=F$_{j,(i-1)}$
NO
q=q+1
q=$UTS_{(i-1)}$
NO
YES
YES: There is a gap in the Attack Rule sequence
END:Pass the packet to the IDPS
i=i+1
k=1
i=N+1
YES: All Attack Rule Tables are checked, no matching rule is found
END:Pass the packet to the IDPS
Insert f in UT_i
UTS_i= UTS_i+1

Figure 16 Engine’s Flowchart

Merging rules also provides adaptivity to our Engine which does not exist in signature based IDPS. Assume that, if a new worm broke out, putting the Internet on high alert for contamination, it would, of course, be necessary to take a specific action, apply a specific patch, close a specific port on the firewall, router or switch, or add a specific signature to all IDS devices. However, if we took specific actions that prevented only this worm and not the 100 similar worms soon to be developed, we would be in violation of this virtue [70]. For example, On August 4, 2001 Code Red II appeared. Code Red II is not a variant of the original Code Red worm. Although it uses the same injection vector it has a completely different payload. It pseudo-randomly chose targets on the same or different subnets as the infected machines according to a fixed probability distribution, favoring targets on its own subnet more often than not. Additionally, it used the pattern of repeating 'X' characters instead of 'N' characters to overflow the buffer. Although its payload is different, its characteristic is similar to the previous one [71]. For instance, we observe that, if 60 packets destined to port UDP 69 are following a packet coming to port UDP 135 or 137, it is most probably a worm. On the other hand, if 60 packets destined to port TCP 445 are following a packet coming to port UDP 135 or 137, it is again most probably a worm. Therefore we do not need to analyze each rule separately. If we observe a packet destined to port UDP 137 or 139 of an IP address, then we have to observe the packets from that IP address and count the packets either destined to port UDP 69 or port TCP 445.

The list of rules extracted in the laboratory stage is in Appendix C. From the logs produced by the commercial IDPS logs and sniffer files, we verify that, this system can effectively detect and deny the types of network intrusions which we focus on. Total amount of false positives and false negatives ratio compared to full functional IPS is about 4%. This system is reducing the load on the intrusion detection and prevention systems and so response time without giving up target network security.

3.2 FINE TUNING NETWORK

In the second step, the rules generated using the gathered data from the small laboratory network are refined on a large real network with a similar topology and the same devices such as firewall and IDPS. Real traffic hence real attacks are used in this second step. Real attacks are investigated using the rule set constructed in the first stage. The sniffed packets, the netflow data, syslogs and the IDPS logs file are compared using the same methodology in the laboratory network for fine tuning of the attack flows determined at first step. In the small laboratory network, collected logs are analyzed to extract the attack's characteristic. On the other hand, in the fine tuning real network, real traffic and attacks are analyzed to improve the attack characteristics.

The topology in Figure 17 is used to realize the fine tuning network. As in small laboratory network test bed, real network consists of a border router, backbone switch, DMZ switch, firewall, IDPS, servers and client computers. We use the same two different leading commercial signature based intrusion detection systems we used for the laboratory network for 3 weeks in the large real network. This network has 42 Mbps total WAN connection: 34 Mbps ATM and 8 Mbps satellite links are used for Internet. The network throughput of the firewall is not constant in the network. The maximum traffic seen on the firewall is listed in Table 4.

Table 4 Maximum Traffic seen on the Firewall in the Fine Tuning Network

Firewall Interface	Direction In	Direction Out
Inside	41 Mbps	2.3 Mbps
Outside	37 Mbps	5.4 Mbps
DMZ	11 Mbps	2.7 Mbps

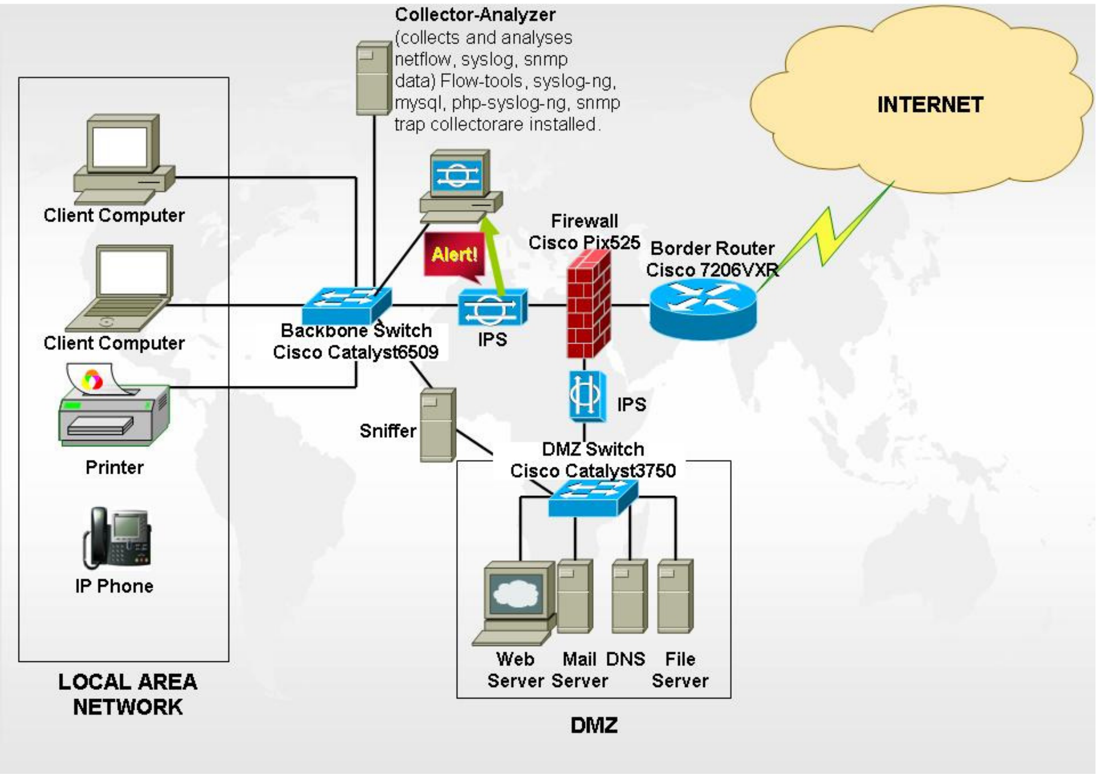


Figure 17 Fine Tuning Network Topology

The feature, model, functionality of the devices used in this topology is given in Table 5. Although the topology of larger real network seems like the topology of small laboratory network, the models and performance of the devices differ.

Table 5 Fine Tuning Network Device/Software

Device/ Software	Feature	Model	Functionality	Settings	Notes
Border Router	Test Bed Component	Cisco 7206VXR	Connects wide area network to local area network. It is located on the perimeter.	12.4.9T advanced security feature set Gigabit Ethernet and Asynchronous Transfer Mode (ATM) and Packet over SONET/SDH (POS)	All of the interfaces are configured to export the flow traces which are supplied by netflow. SNMP Traps and syslog are enabled.
Firewall	Test Bed Component	Cisco PIX525	Perimeter Security. Stateful packet filtering.	Version 7.2.1. Four ethernet ports	Device is configured to send SNMP traps and syslog
Backbone Switch	Test Bed Component	Cisco Catalyst 6509	Layer 3 switch. Interconnection between VLAN's. Access control lists are applied.	12.2.18 IP Services feature set 9 slot modular switch, about 4000 clients are connected through this switch. About 240 gigabit ethernet ports are connected.	Device is configured to send SNMP traps, syslog and netflow.
DMZ Switch	Test Bed Component	Cisco Catalyst 3750	Layer 2 mode. Servers are connected to the DMZ zone through this switch.	12.2.25 IP Services feature set 48 port 10/100/1000 switch	Device is configured to send SNMP traps, syslog and netflow.
Sniffer	Test Bed Component Software	Pentium IV 3.06 GHz. 1 GB ram computer.	Captures the packets passing through backbone switch and DMZ switch.	OS Debian 3.1. Ethereal is installed.	Just for checking the signature based IDPS functionality.

Table 5 (continued)

Device/ Software	Feature	Model	Functionality	Settings	Notes
Collector-Analyzer	Test Bed Component Software	Pentium IV 3.06 GHz. 2 GB ram computer.	Collects syslog, netflow and SNMP traps. Makes correlation and optimization.	OS Centos 4. Flow-tools, Syslog-ng, SNMP trap collector are installed. Syslog, SNMP, netflow data correlation is done using perl scripts on this machine	Perl scripts and MySQL are used to make correlation. Optimization is the main issue in this step.
Servers	Test Bed Component Software	Linux based servers	Mail, Web	OS Centos 4.	Servers are configured to send logs to Collector-Analyzer machine.
IDPS	Analyzer Component	Commercial Product	Signature based IDPS. About +2300 signatures are enabled.	Latest operating system and signatures are installed.	

Layer 2-4 information of each packet, netflow, CPU, memory information of border router, backbone switch and DMZ Switch, denied packets, CPU and memory information of firewalls are stored in the database on Collector-Analyzer. When a rule gathered from the laboratory network phase is matched or IDPS log alerts a new attack, flow is inserted in an uncertain attack table on database. Time intervals determined in the small laboratory network is used in this part of study. However, as the different variants of attacks and new attacks are detected in this step, it is seen that this value is not ideal to catch an attack with low false positive and negative ratio. On the other hand, using high interval values makes the database run slow. Interval value is set for each type of attack. In addition, some different reconnaissance flows that are for the same attacks defined in small laboratory stage are discovered. These flows are inserted in the rule database to decrease false positive and false negative values. For a new attack that is triggered by IPS logs, a new table called by attack's name or id is created on the database. This table includes the information of attacker, victim, router, switch and firewall to analyze and correlate in time interval of 4 minutes, 2 minutes before the attack happens and 2 minutes after the attack happens. For analysis, time interval value, 4 minutes, is used. Because in the small laboratory network, it is seen that 10 minutes time interval is unnecessary, and 4 minutes interval is enough to make fair analysis. Analysis and correlation of this table is done and an existent rule is reorganized or a new rule is added to the rule database. Time values and reconnaissance flows are changed to reorganize and improve the rules. Flow analysis is described in the next section.

Fine Tuning method in large laboratory network is stated in Figure 18.

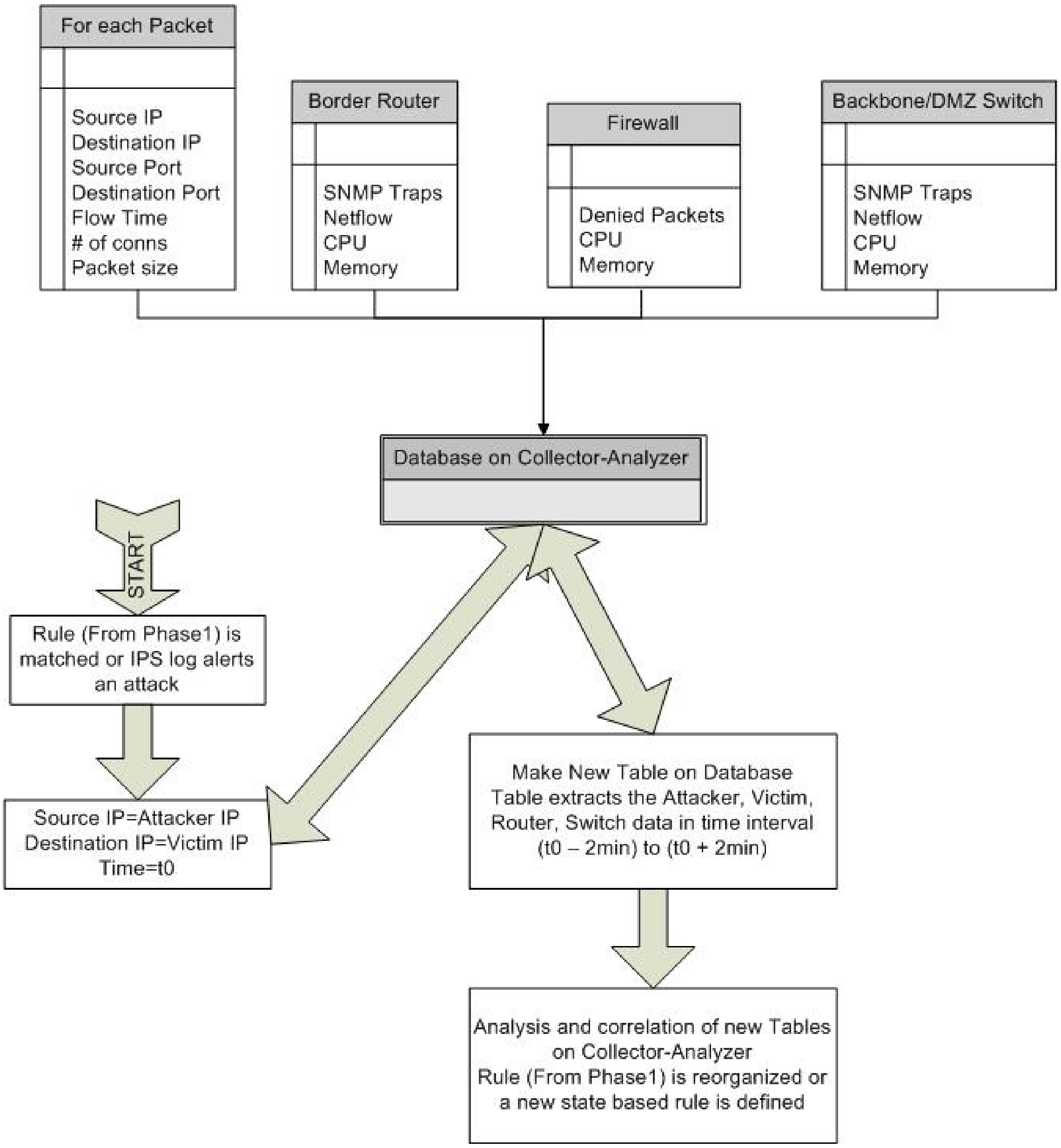

For each Packet
Source IP
Destination IP
Source Port
Destination Port
Flow Time
# of conns
Packet size
Border Router
SNMP Traps
Netflow
CPU
Memory
Firewall
Denied Packets
CPU
Memory
Backbone/DMZ Switch
SNMP Traps
Netflow
CPU
Memory
Database on Collector-Analyzer
START
Rule (From Phase1) is matched or IPS log alerts an attack
Source IP=Attacker IP
Destination IP=Victim IP
Time=t0
Make New Table on Database
Table extracts the Attacker, Victim, Router, Switch data in time interval (t0 – 2min) to (t0 + 2min)
Analysis and correlation of new Tables on Collector-Analyzer
Rule (From Phase1) is reorganized or a new state based rule is defined

Figure 18 Fine Tuning Method

Combined rule set is consisting of about 40 rules. To test in real network, the Engine is connected to the network in passive mode as in Figure 19. At the same time, commercial IDPS is running in in-path (inline) mode. It is just monitoring network, not dropping any connection. Logs of the IDPS are compared with the logs of the Engine. According to the logs of IDPS and packet payloads, our engine was not so successful. About %10 false positive and %15 false negative alerts are produced by the Engine during the first observations of fine tuning phase. The system is observed for four days in the small network and for ten days in the large real network. Experiments are done in another ten days. The analysis and experiments took three weeks totally. We change the time periods of the rules and we made some modifications on the rule set. The rules were modified and tested until the Engine has zero false negative packets with referring to the commercial products' logs. For instance, in the flow analysis and small laboratory network we observe that, if 60 packets destined to port UDP 69 or TCP 445 are following a packet coming to port UDP 135 or 137, it is most probably a worm. However, in the fine tuning network as we try to detect these kinds of worms with this rule, the number of packets expected caused a problem. Although the commercial products' give alert, the proposed Engine did not detect the worm. Since the worm is detected by commercial product, there is not false negative for the network. But we can say that there is false negative for the proposed Engine. As we decrease the value to 20, then the worms are detected by the Engine. However, this time regular TFTP connections are detected as a worm. So, false positives are seen. With referring to the commercial products' logs, the value is set to 40 to decrease false positive and false negative probability for that rule to 0%.

3.3 INTEGRATING THE DETECTION ENGINE WITH IDPS

The computer with hostname Sniffer is used as our Detection Engine. The Detection Engine operates in one of two modes:

- Passive listener mode: The Engine processes the data on behalf of the network devices. Packets are not passing through the Engine.
- Transparent bridge mode (Inline mode): Packets are passing through the Engine. Engine can detect and drop the intrusions by itself. If the Engine announces an illegal traffic, it can deny the connection using Ebtables

running on the same server. Engine is important component, since the failure of Engine interrupts the network connections.

It is important to select the right operational mode for the Engine because it impacts how we integrate it into the existing network and how the Engine handles traffic. It can only operate in one mode at a time.

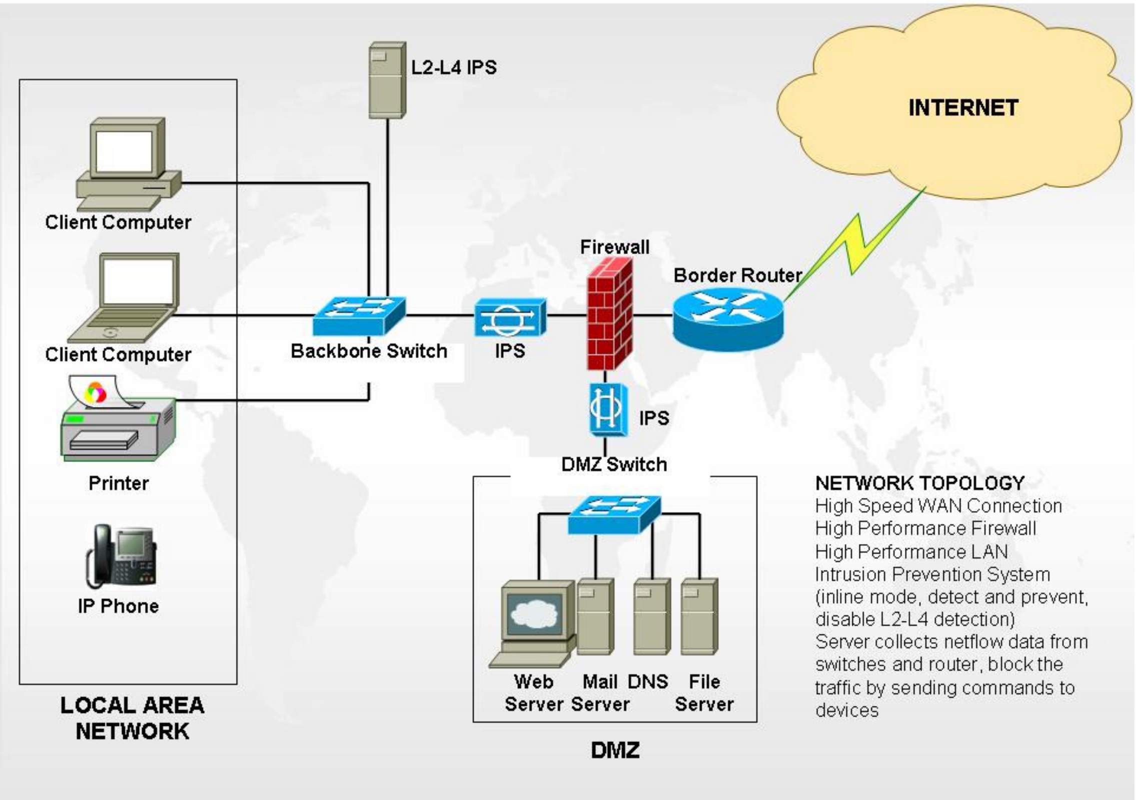

L2-L4 IPS
INTERNET
Client Computer
Client Computer
Printer
IP Phone
LOCAL AREA NETWORK
Backbone Switch
IPS
Firewall
Border Router
IPS
DMZ Switch
Web Server
Mail Server
DNS
File Server
DMZ
NETWORK TOPOLOGY
High Speed WAN Connection
High Performance Firewall
High Performance LAN
Intrusion Prevention System
(inline mode, detect and prevent, disable L2-L4 detection)
Server collects netflow data from switches and router, block the traffic by sending commands to devices

Figure 19 Engine in Passive Mode

The feature, model, functionality of the devices used in this topology is same with the given ones in Table 3. In this topology, for this improvement of the action functionality of the Engine, Sniffer Server and Collector-Analyzer are not needed to be used. The fine tuned rules and the detection engine is running on the server called L2-L4 IDPS. The hardware specs of this server are almost same as the ones of the

Sniffer Server. It has 1GB extra memory. OS Debian 3.1 is running on the server. Flow-tools, Syslog-ng, SNMP trap collector are installed to collect data. Syslog, SNMP, netflow data correlation is done using perl scripts on this machine. Perl scripts and MySQL are used to make detection. Rules are located in the database.

In Passive Listener Mode, router, switch and firewall must be set up to send flow data, syslog data, SNMP traps to the Engine. The server then processes the data on behalf of these network devices.

Although there are two gigabit Ethernet NICs (Network Interface Card) on the L2-L4 IDPS server, one of them is being used to run in Passive Listener Mode. IP address is defined on the port that is connected to the network. All data from router, switch and firewall is directed to this IP address. If the Engine discovers illegal traffic, it connects to one of the network device and writes an access-list to drop the packets. Depending on the way of intrusion, access-lists are written on a first device that the attack first appears. Access-lists can be written on a router, switch or firewall. According to the rule, access-list can be permanent or a time based access-list.

Remote connection to a network device is done using Expect package on the Debian server. The Expect package provides a program that performs programmed dialogue with other interactive programs.

In Transparent Bridge mode as shown in Figure 20, the server is connected to the network using both of the two gigabit NIC's, LAN1 and LAN2 ports. Although the server gets the flow data, syslog data and SNMP traps from the router, switch and firewall, it mainly collects the information of the flows by sniffing the traffic that is passing through its LAN ports. However, it is just using flow headers of the packets, such as source/destination IP addresses, TCP/UDP ports, time stamps of the packets.

Firewall and L2-L4 IDPS functions are running on the same server. L2-L4 IDPS process is following the firewall process for an incoming packet from outside. It is just for security and device processor optimization. There is no need to analyze packets that will be dropped by firewall. Therefore, an incoming packet that is not acceptable by the firewall is not sent to L2/L4 IDPS process. Processor and memory is not running unnecessarily for these kinds of packets. Therefore L2-L4 IDPS process is operating between the firewall and IDPS server in the network.

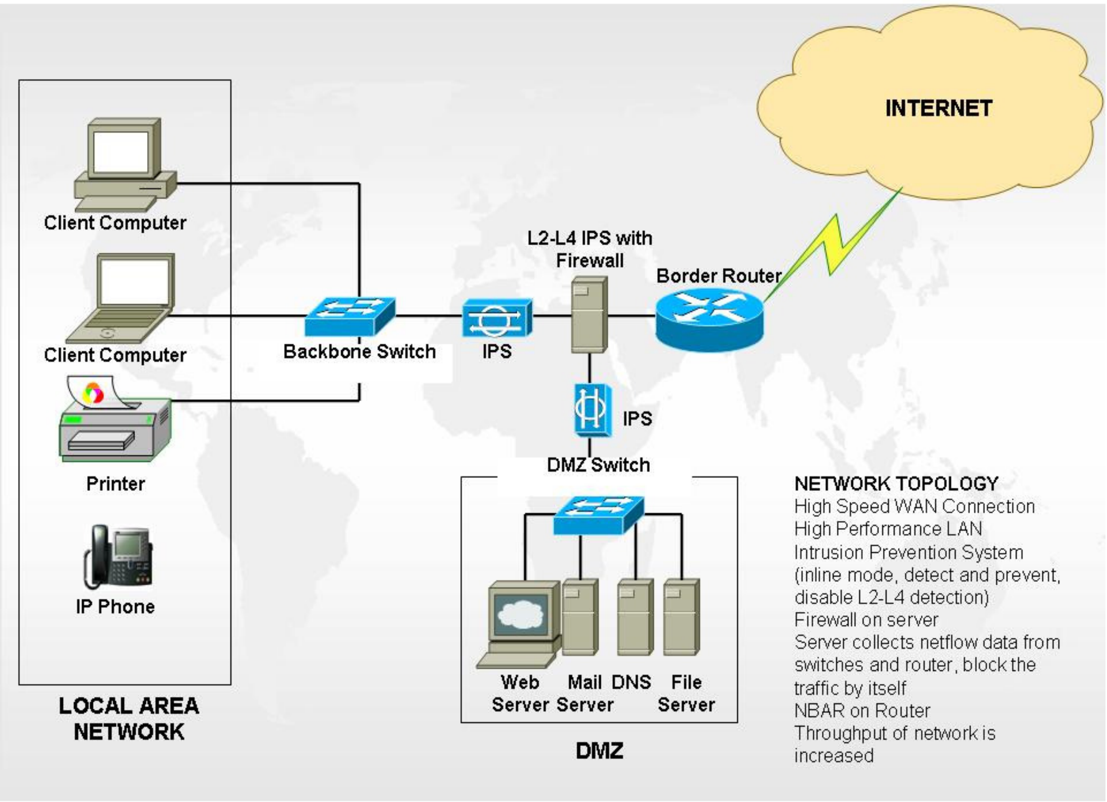


Figure 20 Engine in Transparent Bridge Mode

The feature, model, functionality of the devices used in this topology is same with the given ones in passive listener mode. In addition to the tools specified in the previous mode, ebtables and iptables are used to enable firewall functionality on the server. If the packets passing through this server are hostile, either the Engine can modify the firewall rules on itself or it modifies the filters on switches or router or firewall using expect scripts.

A bridge is a network device that connects two network segments of any network type (Ethernet, token ring etc.) transparently to form one subnet. Transparency means that we do not have to tell any component (computer, application etc.) that there is a new device between them. So, no configuration on them is needed.

A bridge is a way to connect two Ethernet segments together in a protocol independent way. Packets are forwarded based on Ethernet address, rather than IP address (like a router). Since forwarding is done at Layer 2, all protocols can go transparently through a bridge.

The Linux bridge code implements a subset of the ANSI/IEEE 802.1d. A Linux bridge is more powerful than a pure hardware bridge because it can also filter and shape traffic. The combination of bridging and firewalling is done using ebtables.

Using Ebtables [72] all of the Layer 2-4 data that is passing through the server is collected. Flow data and SNMP traps are taken from the switch and syslog data is taken from the front firewall. All data correlation is done by the Engine and analyzed using the Rule set. If the Engine announces illegal traffic, it can deny the connection using Ebtables running on the same server. Configuration examples are given in Appendix D.

Firewalls are typically implemented as routers, but our system does not have to be that way. Bridging packet filters have a number of advantages, and we can add them to the network at a later stage without changing the configuration of the network components' IP addresses and routes.

On the other hand, TC is used to control traffic as described in background [61]. When the kernel has several packets to send out over a network device, it has to decide which ones to send first, which ones to delay, and which ones to drop. This is the job of the packet scheduler, and several different algorithms for how to do this "fairly" have been proposed.

CHAPTER 4

EXPERIMENTS AND RESULTS

The Engine is configured to detect unwanted manipulations to computer systems using the rule set. The rule set is organized and optimized to detect the attacks with low false positive and false negative ratio as described in Chapter III. The Engine is integrated with commercial IDPS in the network as a complementary component. While the Engine is working, it decreases the load, CPU and memory consumption of commercial IDPS in the network. To analyze the impact of the engine on the network performance, we use the large scale network that we used for the fine tuning of the rules in Chapter III. Note that, our approach for constructing the Engine is independent of the topology and it just requires devices that are able to collect and process the necessary information. The traffic that we observe during the experiments is real traffic generated by public clients and servers. Note that, the rule set can differ from network to network. Hence, before putting the Engine in production, it will be better to run in analysis passive listener mode, to analyze and regulate the rules for that network. For the regulation, rules that are determined after the fine tuning phase will be used.

We conduct a series of experiments and perform an analysis on the response time, CPU and memory usage of the commercial IDPS and also on the false negatives and false positives of our method measured with respect to the commercial IDPS.

Response time is the elapsed time between the end of an inquiry or demand on a computer system and the beginning of a response; for example, the length of the time between an indication of the end of an inquiry and the display of the first character of the response at a user terminal.

Experiments are done on the large scale network. Traffic is not generated by the generators. It is real traffic. The experiment results are collected and averaged over a week. The experiments are repeated three times. First experiments are made on the network without any intrusion detection and prevention device for a week. Then the same experiments are made for another week using just commercial IDPS with full active signatures. The last experiments are made using commercial IDPS with integrated Engine for a week. Note that, the traffic characteristics of the network do not change over three weeks. Average number of connections, flow connection distribution, and the network throughput is consistent for each recurring week. We use two different leading commercial signature based intrusion detection systems in the large real network. Their percentage in the market is about 30% (each about ~15%) according to Gartner report in May 2006.

A response time measurement tool is written in Perl and is used to analyze the response time of most popular sites on the web in Turkey which are located on high available servers (www.milliyet.com.tr, www.youtube.com etc [73]). The tool script is running as a client. It sends a request and gets the whole response data. This tool's capabilities are:

- using test methods tool can check servers (health check),
- measuring response time of web server
- measuring response time of ftp server
- measuring response time of smtp server
- measuring response time of pop3 server
- measuring response time of sql inquiries

Test Results

Each experiment configuration –no IDPS, full functional IDPS and our proposed flow-based intrusion detection Engine integrated with IDPS- is tested for a week on the large scale real network. The results of the experiments are presented next.

Response time:

The response time is measured for different scenarios over a week and the following figures for each hour is obtained averaging the measurements taken for that specific hour of the day over a week period.

As seen in Figure 21-27, the response time is the longest while we are using full functional IDPS. When the Engine is integrated to commercial IDPS, there is still latency as compared to the design without any IDPS. However, it is not as high as the latency in the design with full function IDPS. When the Engine is used in the network, the number of packets processed with payload inspection is less compared to the full functional IDPS over the same amount of traffic. Therefore, response time for same amount of data is less than using full functional IDPS when our proposed Engine is installed.

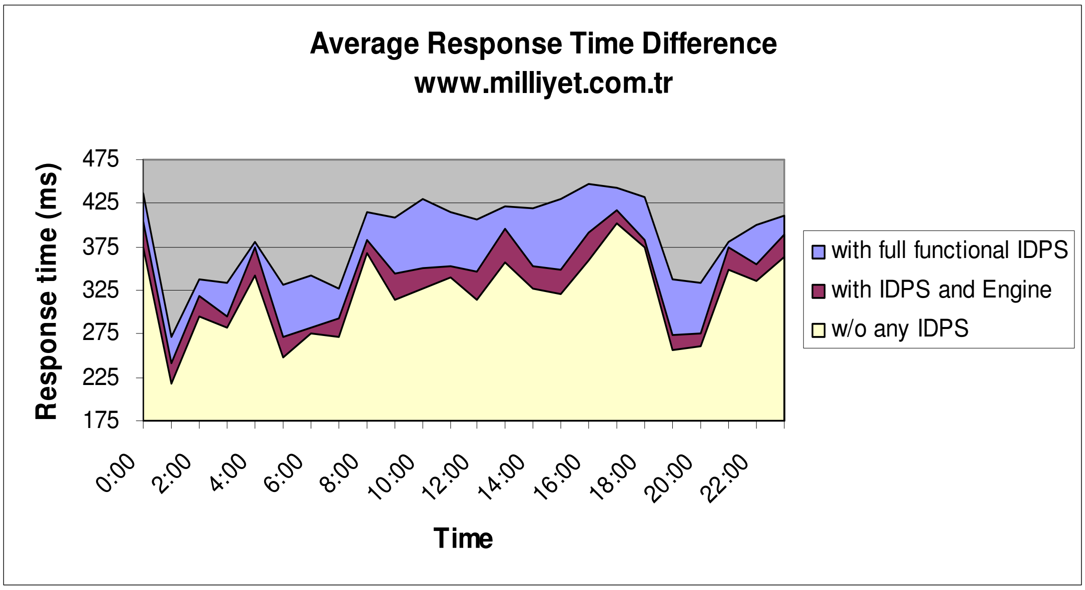


Figure 21 Average Response Time Difference (www.milliyet.com.tr)

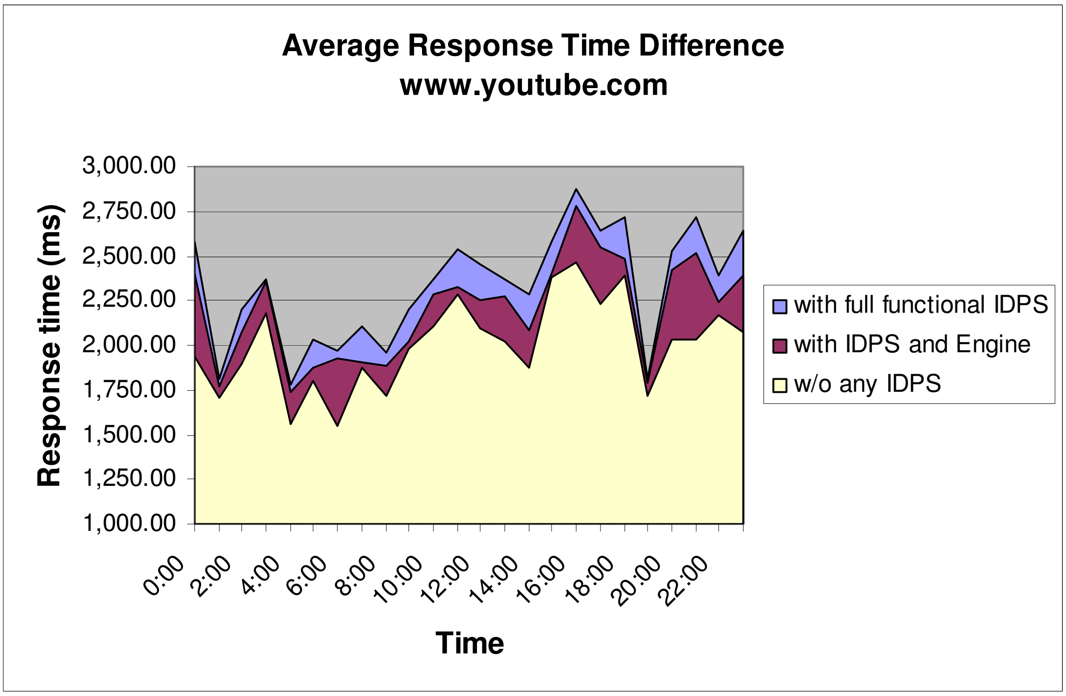


Figure 22 Average Response Time Difference (www.youtube.com)

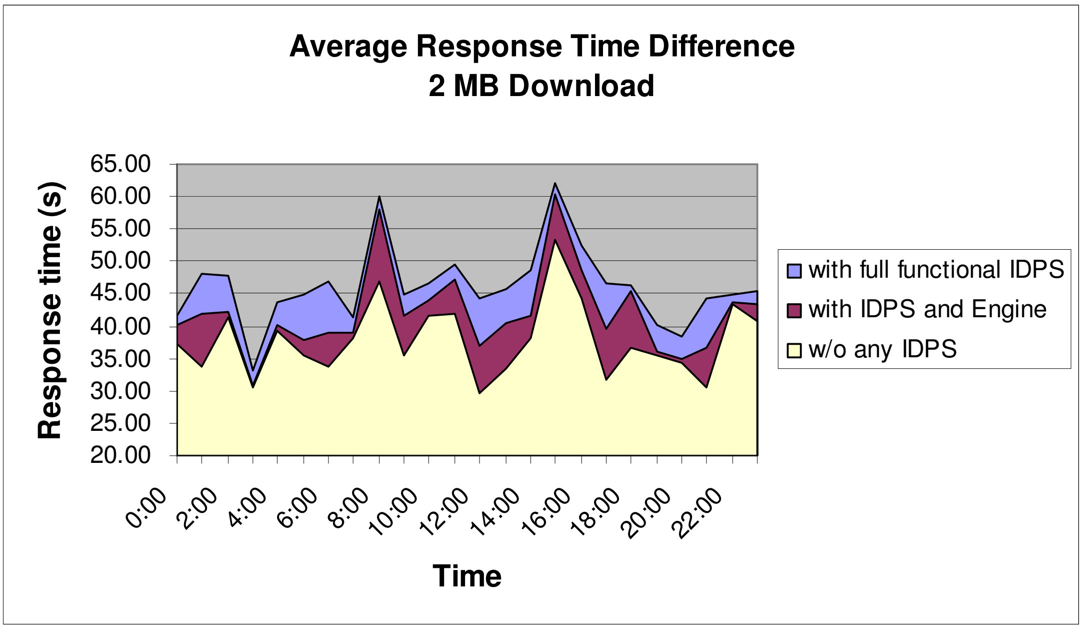


Figure 23 Average Response Time Difference (2MB download)

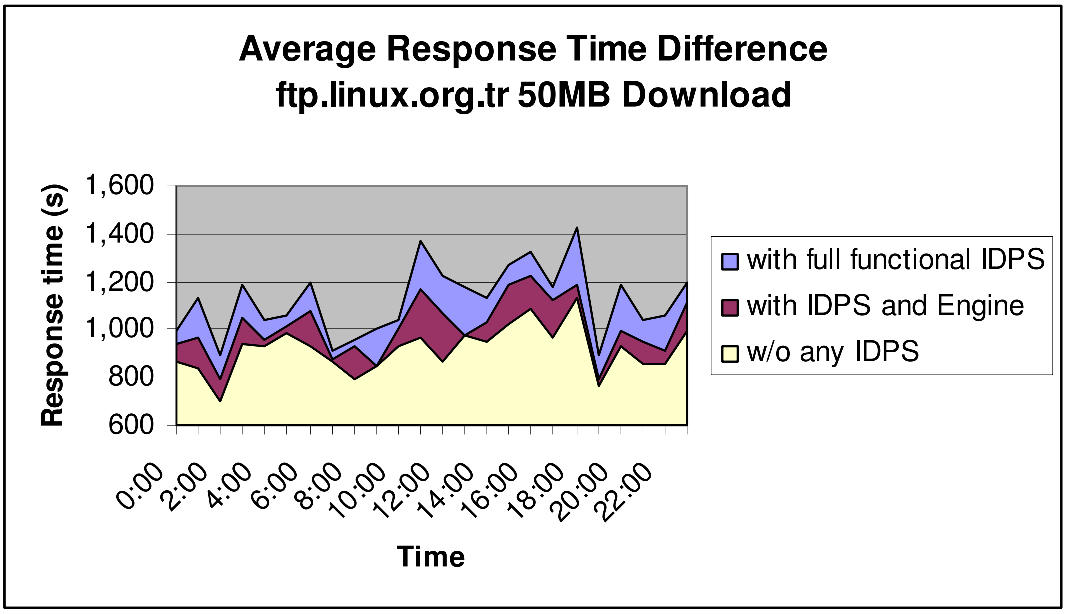


Figure 24 Average Response Time Difference (50 MB FTP)

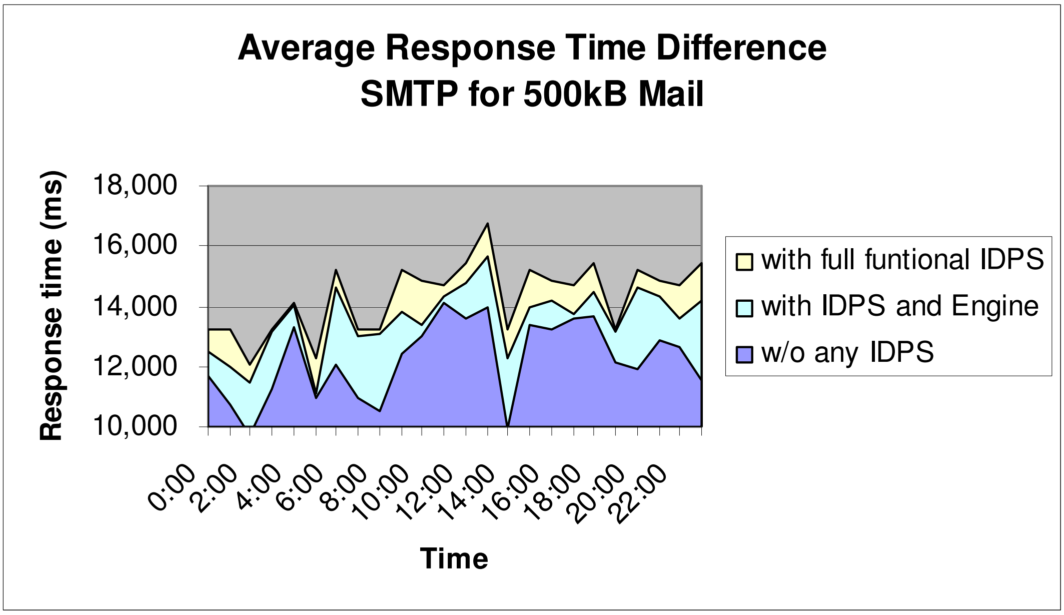


Figure 25 Average Response Time Difference (SMTP for 500kB Mail)

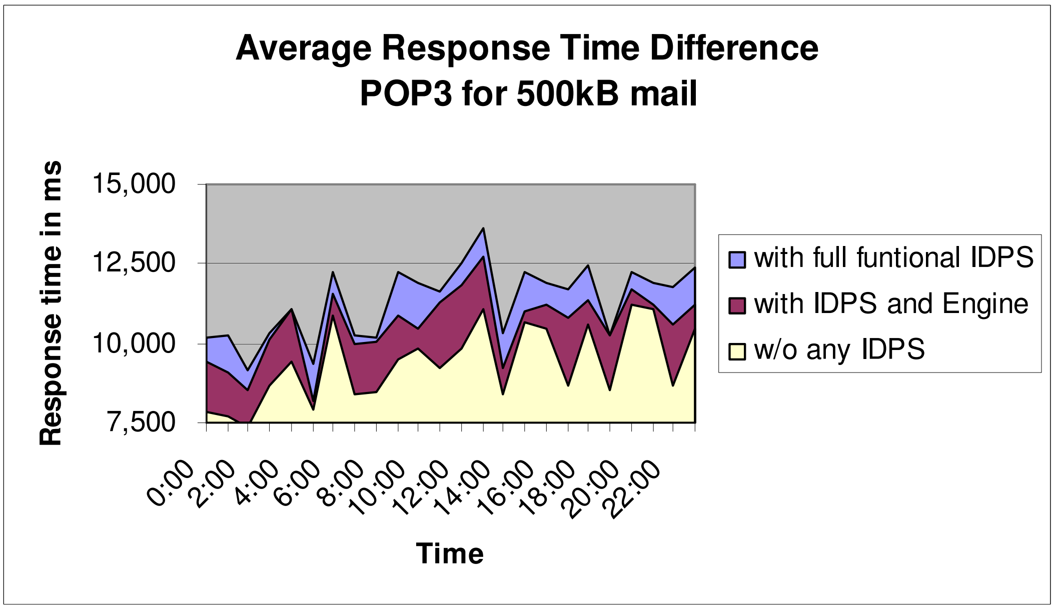


Figure 26 Average Response Time Difference (POP3 for 500kB Mail)

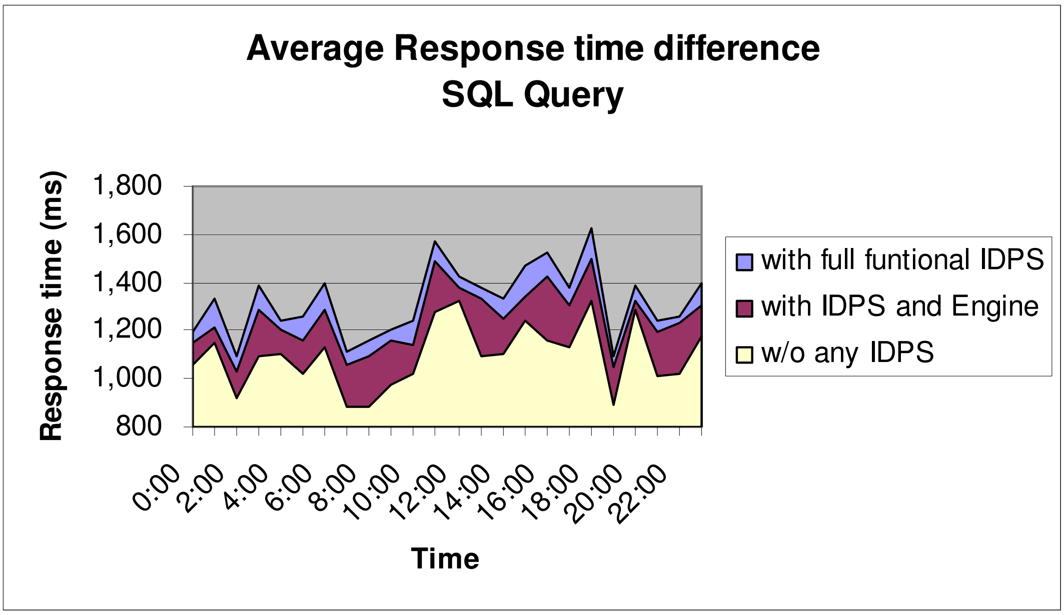


Figure 27 Average Response Time Difference (SQL Query)

For the response time analysis, the response time improvement must be considered. We define the response time improvement (RTI) for a daily traffic as follows:

$$\mathrm{RTI} = \left[1 - \frac{\text{[Area (with ips and Engine) - Area (without ips)]}}{\text{[Area(with full functional ips) - Area (without ips)]}}\right].$$

The RTI calculated in these experiments are presented in Table 6.

Table 6 Response Time Improvement

TEST	RTLI
www.milliyet.com.tr	36.51%
www.youtube.com	33.84%
2 MB Download	35.45%
ftp.linux.org.tr (50 MB Download)	31.41%
SMTP for 500 kB Mail	31.68%
POP3 for 500 kB Mail	31.61%
SQL Query	30.20%

CPU and memory usage:

We examine the commercial IDPS product's performance (number of attacks, CPU and memory consumption) when it is used in full functional mode and when it is integrated with our proposed Engine.

Figure 28 and Figure 29 show the memory and CPU usage during the tests. Although the CPU time and memory usage calculation of commercial IPS in our real network does not give very distinct difference, it is shown that when we use L2-L4 IPS tool, number of attacks that the commercial IPS need to catch decreases about 25% (See Figure 30) although the amount of data passing through the device is same. The reason for this decrease is that, L2-L4 attacks are being detected by the Engine. Each day, about 10K attacks are detected using this system. The attack types are summarized in Table 7.

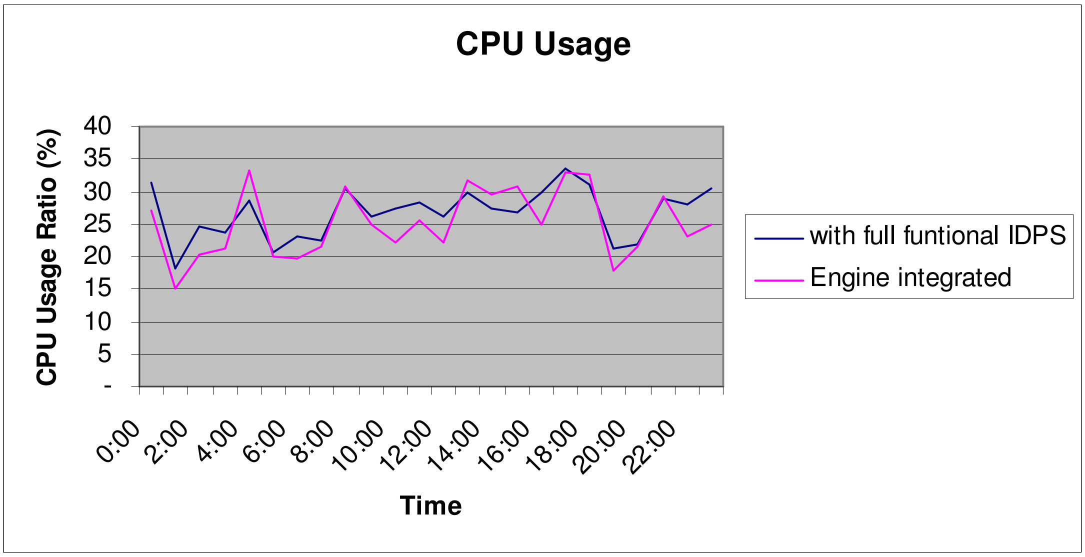


Figure 28 CPU Usage of Commercial IDPS

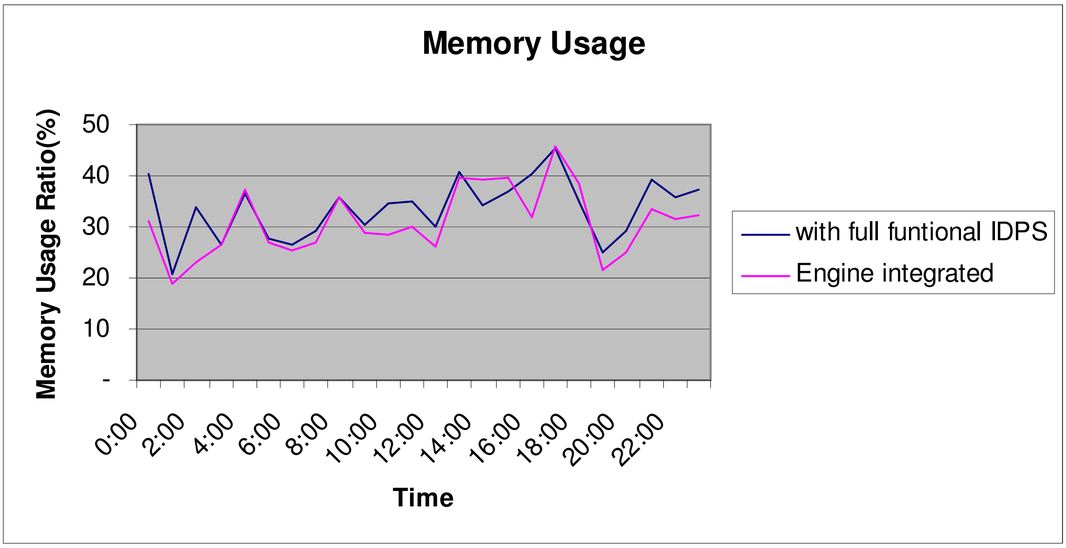


Figure 29 Memory Usage of Commercial IDPS

Despite decreasing number of attacks on the commercial IDPS, CPU and memory consumption is almost the same. Since the traffic on the IDPS is saturated, the throughput does not decrease as we reduce the load over the IDPS. However, the amount of data passing through the network during the same time period increases. It is the reason that the response time of connections are decreasing.

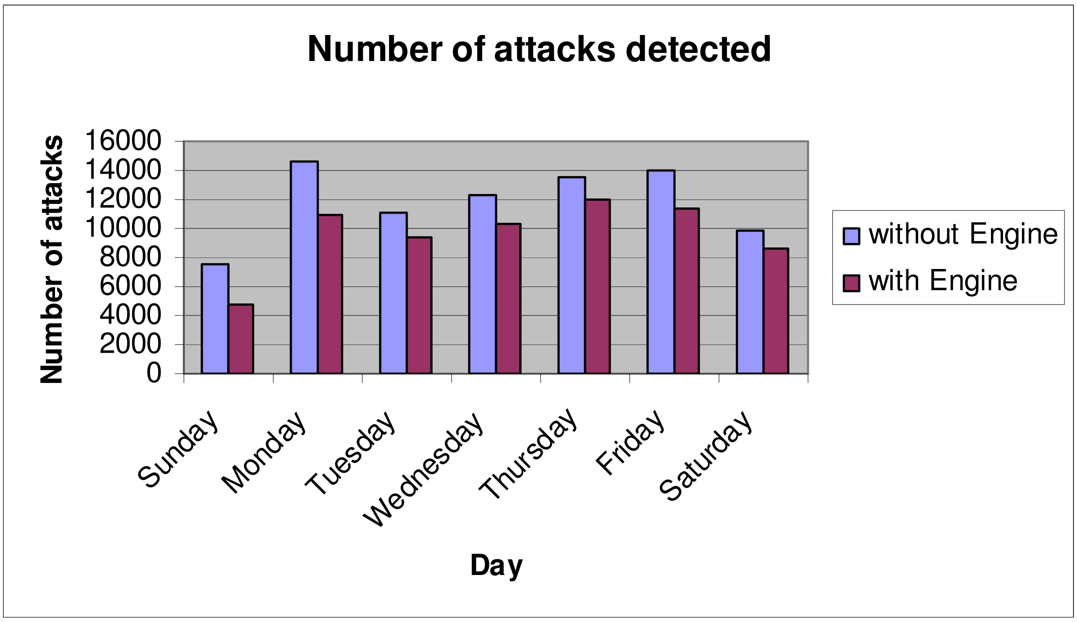


Figure 30 Number of attacks on commercial IDPS per day

Table 7 Attack Statistics

Description	**Percentage**
SQL Buffer Overflow	32%
W32 Worm Propagation	25%
Apache/IIS Buffer Overflow	10%
Windows RPC Messenger Service pop-up spam	7%
Windows Buffer Overflow	4.2%
Windows RPC Bind Request Buffer Overflow	3.2%
DLL Windows File Download	2.2%
Remote Management Exploit Attempt	1.7%
Real-time Virus-scan Overflow Attempt	1.7%
VNC Network Scanning Activity	1.4%
Other	8%

CPU and memory consumption of the proposed Engine is analyzed. It is seen that the Engine with this hardware configuration can handle 90 Mbps Internet traffic (Figure 31).

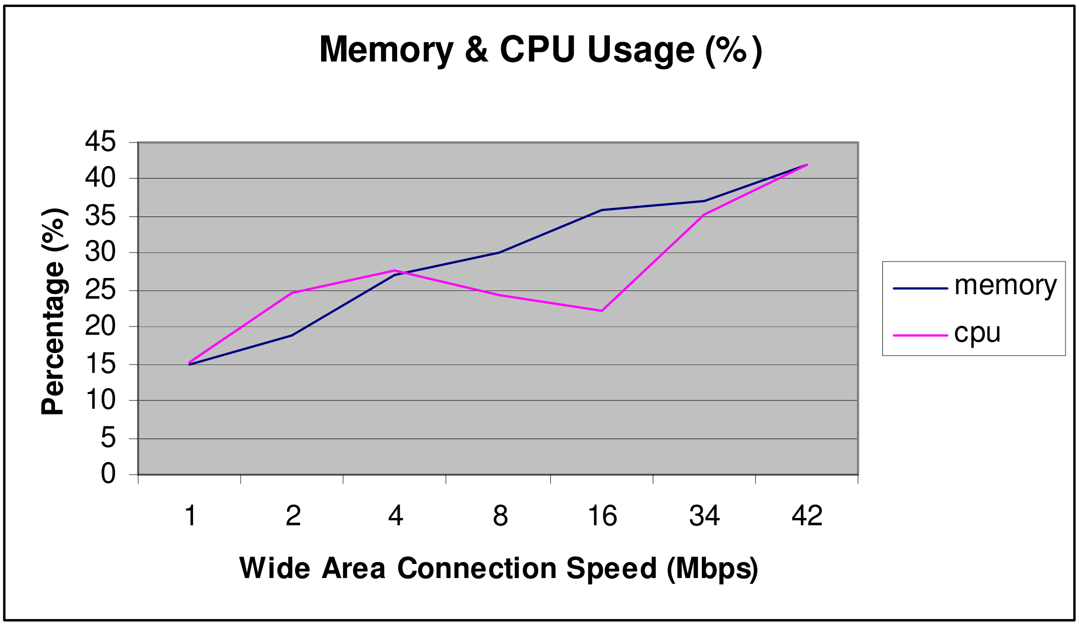


Figure 31 Memory and CPU Usage of the Proposed Engine

Success Rate of the proposed Engine integrated with IDPS

As seen in Figure 32, total amount of false positives and false negatives ratio compared to full functional IPS is about 4%.

Proposed Engine is installed on another network which has similar network topology with 2 Mbps Internet connection. Applications running on this network are different than the ones running on the previous network, because of the company's application and user profiles. False negatives and false positives ratio are analyzed on this network and same results are taken using the same rule set.

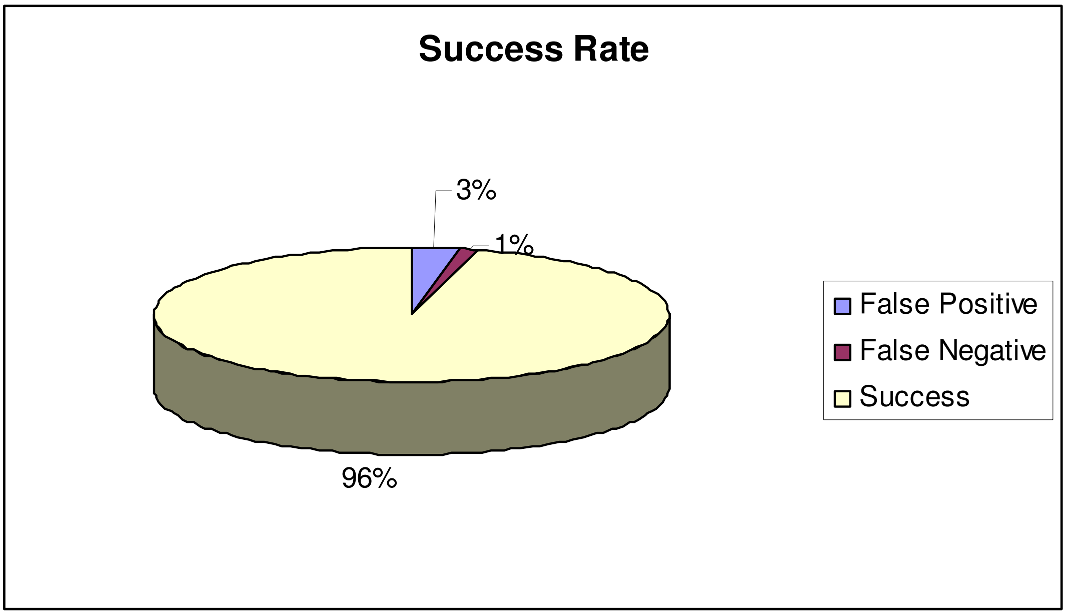


Figure 32 Success Rate of the proposed Engine integrated with IDPS

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this thesis, we present our new approach to scalable intrusion detection for high-speed networks. Currently, the most accurate intrusion detection methods look for specific signatures in the IP packet payload. Hence, the signature based detection methods have scalability problems to work at backbone rates. In addition packet inspection is useless for encrypted data and it violates privacy.

We use flow data as well as network monitoring, management and control data such as logs and SNMP traps collected from the devices in the network -router, switch, firewall, server- to construct a set of rules that describe the behaviour of the attack flows in time. These rules are then used by our Detection Engine to detect Layer 2-4 attacks decreasing the load on the IDPS and improving the performance of the network. Experiments in a real network show that our Detection Engine is 96% accurate with respect to the commercial IDPS and it improves the response time of the network by about 30%.

Throughput of intrusion detection and prevention systems does not improve as fast as that of high speed network infrastructure. Therefore, we can say IDPS is a rate determining object of network. Our engine is reducing the load on the intrusion detection and prevention systems without giving up target network security. Using IDPS more productive increases the network throughput.

A very important contribution of our approach is demonstrating that the solutions reducing the load on the intrusion detection and prevention systems increase the whole throughput of network. Besides the studies to find new techniques or improve present techniques of the intrusion detection and prevention systems, cooperative tools that is reducing the load on the intrusion detection and prevention systems without giving up target network security can be studied.

Our future work includes further performance analysis of our approach under different scenarios such as starting with the same basic rule set in different networks and observe how the rule set can be fine-tuned to the specific network conditions. We will perform experiments that involve encrypted data and day zero protection issues. Another future study is the analysis that includes the improvements on cooperative tools to intrusion detection systems which can be achieved by using the experiences in anomaly based intrusion detection research.

REFERENCES

[1] Telecommunications Industry Association, TIA White Paper: Fiber Optic Network Capacity and Utilization, 2002.

[2] Ferreira, P., Lehr, W., and McKnight, L. Optical networks and the future of broadband services, *Technological Forecasting & Social Change* 69: 741–758, 2002.

[3] Cisco Systems White Paper: Global IP Traffic Forecast and Methodology, 2007.

[4] Cisco Systems White Paper: Exabyte Era, 2007.

[5] Chen, D.Z., Xia, T. J., Wellbrock, G., Mamyshev, P., Penticost, S., Grosso, G., Puc, A., Perrier, P., and Fevrier, H. New Field Trial Distance Record of 3040 km on Wide Reach WDM With 10 and 40 Gb/s Transmission Including OC-768 Traffic Without Regeneration, *Journal Of Lightwave Technology*, 25(1): 28-37, 2007.

[6] Altera White Paper: The Evolution of High-Speed Transceiver Technology, Nov 2002.

[7] Chenga, J. Z., Yua, H., Sincoskieb, W. D. Meeting the broadband access infrastructure demands: The promise of Gigabit Ethernet, *Technological Forecasting & Social Change* 72: 1–10, 2005.

[8] Metroethernet Forum [Online]. Available: http://www.metroethernetforum.org, November 2007.

[9] NORTEL, White Paper: Evolution towards volume deployable 40-Gbps networks, 2007.

[10] Tront, J. G. and Marchany, R. C. Meeting Internet Security: Intrusion Detection & Prevention, Proceedings of the 37th Hawaii International Conference on System Sciences, 2004.

[11] Cisco Systems: Securing Cisco IOS Networks Student Guide, 2004.

[12] CERT Coordination Center (CERT/CC) [Online]. Available: http://www.cert.org/stats, November 2007.

[13] Al-Shaer, E., Hamed, H. Taxonomy of Conflicts in Network Security Policies, *IEEE Communications Magazine*, 44(3): 134-141, 2006.

[14] Al-Shaer, E. Network Security Policies: Verification, Optimization and Testing, Talk at Georgia Institute of Technology, Atlanta, October 2005.

[15] Site Security Handbook (RFC 2196) [Online]. Available: http://www.ietf.org/rfc/rfc2196.txt, November 2007.

[16] Labib, K. Computer Security and Intrusion Detection. ACM Crossroads special issue on Computer Security, Issue 11.1, Fall 2004.

[17] Douligeris, C., and Mitrokotsa, A. DDoS attacks and defense mechanisms: classification and state-of-the-art, *Computer Networks* 44: 643–666, 2004.

[18] Lundin, E. and Jonsson, E. Privacy versus Intrusion Detection Analysis. Proceedings of the Second International Workshop on Recent Advances in Intrusion Detection, W. Lafayette, IN, 1999.

[19] Sundaram, A., An Introduction to Intrusion Detection, Crossroads: The ACMStudent Magazine, 2, 4, Hyperlink: acm.org/Crossroads, 1996.

[20] Anderson, J. P. Computer Security Threat Monitoring and Surveillance, Technical report, James P Anderson Co., Fort Washington, Pennsylvania, April 1980.

[21] Kemmerer, R.A., Vigna, G. Intrusion detection: a brief history and overview, Reliable Software Group, Computer Science Department, University of California Santa Barbara, Supplement to Computer Security and Privacy, pp.27–30, 2002.

[22] Mirkovic, J., Prier, G., and Reiher, P. Attacking DDoS at the source, Proceedings of ICNP 2002, Paris, France, pp. 312–321, 2002.

[23] Yang, W., Fanga, B., Liub, B., and Zhanga, H. Intrusion detection system for high-speed network, *Computer Communications* 27: 1288-1294, 2004.

[24] Ware, W. H. Security Controls for Computer Systems: Report of Defense Science Board, Task Force on Computer Security. Santa Monica, CA: The Rand Corporation, 1979.

[25] Anderson. J. P. Computer Security Threat Monitoring and Surveillance. Technical Report, James P Anderson Co., Fort Washington, Pennsylvania, 1980.

[26] Department of Defense (DoD) Trusted Computer System Evaluation Criteria (TCSEC) (DoD 5200.28-STD 1985). Fort Meade, MD: Department of Defense, 1985.

[27] Denning, D., Edwards, D., Jagannathan, R., Lunt, T., and Neumann, P. A Prototype IDES: A Real-Time Intrusion Detection Expert System, Computer Science Laboratory, SRI International. 1987.

[28] Spafford, E. H. The Internet Worm Program: An Analysis (CSD-TR-823). West Lafayette, IN: Purdue University, 1988.

[29] Heberlien, T., Mukherjee, B., Levitt, K.N., Dias, G., and Mansur, D. Towards Detecting Intrusions in a Networked Environment. Proceedings of the Fourteenth Department of Energy Computer Security Group Conference, pp. 1747-1765, 1991.

[30] Lunt, T.F., Tamaru, A., Gilham, F., Jagannathan, R., Neumann, P., Javitz, H., Valdes, A., and Garvey, T. A realtime intrusion detection expert system (IDES) - final technical report. Computer Science Laboratory, SRI Intemational, Menlo Park, Califomia, 1992.

[31] Hochberg, J., Jackson, K., Stallings, C., McClary, J.F., DuBois, D., and Ford, J. NADIR An Automated System for Detecting Network Intrusion and Misuse. *Computers and Security* 12(3):235-248, 1993.

[32] Javitz, H.S., Valdes, A., Lunt, T.F., Tamaru, A., Tyson, M., and Lowrance, J. Next generation intrusion detection expert system (NIDES); 1: Statisticalalgorithms rationale; 2: Rationale for proposed resolver. Technical Report A016, SRIInternational, 333 Ravenswood Avenue, Menlo Park, CA 94025, March 1993.

[33] Crosbie, M., and Spafford, E. Defending a Computer System Using Autonomous Agents. Proceedings of the Eighteenth National Information Systems Security Conference, Baltimore, MD, 1995.

[34] Chen, S.S., Cheung, S., Crawford, R., Dilger, M., Frank, J., Hoagland, J., Levitt, K., Wee, C., Yip, R., and Zerkle, D., GrIDS-- A Graph-Based

Intrusion Detection System for Large Networks, The 19th National Information Systems Security Conference, Baltimore, MD, 1996.

[35] Anderson, R. and Khattak, A. The Use of Information Retrieval Techniques for Intrusion Detection, Proceedings of RA ID, Louvain-la-Neuve, Belgium, 1998.

[36] Ourston, D., Matzner, S., Stump, W., and Hopkins, B. The Prediction Role Of Hidden Markov Model In Intrusion Detection, CCECE 2003 - CCGEI 2003, Montreal, May 2003.

[37] Scott, S.L. A Bayesian paradigm for designing intrusion detection systems, *Computational Statistics & Data Analysis,* 45(1): 69-83, 2004.

[38] Yin, Q., Shen, L., Zhang, R., and Li, X. A new intrusion detection method based on behavioral model, Intelligent Control and Automation, WCICA 2004, Fifth World Congress on Volume 5, pp. 4370 – 4374, 15-19 June 2004.

[39] Gavrilis, D., and Dermatas, E. Real-time detection of distributed denial-of-service attacks using RBF networks and statistical features, *Computer Networks and ISDN Systems,* 48(2): 2005.

[40] Yang, W., Fang, B., Liu, B., and Zhang, H. Intrusion detection system for high-speed network, *Computer Communications,* 27(13): 1288-1294, 2004.

[41] Jin, H., Yang, Z., Sun, J., Tu, X., and Han, Z. CIPS: Coordinated Intrusion Prevention System, Information Networking, Volume 3391: 89-98, 2005.

[42] Haagdorens, B., Vermeiren, T., and Goossens, M. Improving the Performance of Signature-Based Network Intrusion Detection Sensors by Multi-threading, Information Security Applications, Volume 3325: 188-203, 2005.

[43] Chen, W., Hsu, S., Shen, H. Application of SVM and ANN for intrusion detection, *Computers and Operations Research,* 32(10): 2617 – 2634, 2005.

[44] Kim, S. Pattern Matching Acceleration for Network Intrusion Detection Systems, Embedded Computer Systems: Architectures, Modeling, and Simulation, Volume 3553:289-298, 2005.

[45] Pao, T., Wang, P., NetFlow Based Intrusion Detection System, Proceedings of the 2004 IEEE International Conference on Networking, Sensing & Control Taipei, Taiwan, pp. 731-736, March 21-23, 2004.

[46] Kompella, R.R., Singh, S., and Varghese, G., On Scalable Attack Detection in the Network, IEEE/ACM *Transactions On Networking*, 15(1): 14-25, 2007.

[47] Derek, L., Vijay, S., Design Alternatives for a High-Performance Self-Securing Ethernet Network Interface, Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International

[48] Infonetics Research [Online]. Available: http://www.infonetics.com, November 2007.

[49] Caida [Online]. Available: http://www.caida.org/analysis/AIX/plen_hist/, November 2007.

[50] Ethereal [Online]. Available: http://www.ethereal.com, November 2007.

[51] Cisco Netflow [Online]. Available: http://www.cisco.com/warp/public/732/Tech/netflow, November 2007.

[52] Flow-tools [Online]. Available: http://www.splintered.net/sw/flow-tools/, November 2007.

[53] Plonka, D., Flowscan: A network traffic flow reporting and visualization tool, USENIX LISA, pp. 305–317, December 2000.

[54] RRDtool [online]. Available: http://oss.oetiker.ch/rrdtool/, November 2007.

[55] NBAR [Online]. Available: http://www.cisco.com/go/nbar, November 2007.

[56] SNMP [Online]. Available: http://www.ietf.org/rfc/rfc1157.txt, November 2007.

[57] SNMP Trap Collector [Online]. Available: http://sourceforge.net/projects/ trapcollector/, November 2007.

[58] Syslog [Online]. Available: http://www1.tools.ietf.org/html/draft-ietf-syslog-protocol-19, November 2007.

[59] Iptables [Online]. Available: http://www.netfilter.org, November 2007.

[60] Ebtables [Online]. Available http://ebtables.sourceforge.net/, November 2007.

[61] Traffic Control [Online]. Available http://www.rns-nis.co.yu/~mps/linux-tc.html, November 2007.

[62] Backtrack [Online]. Available: http://www.remote-exploit.org/backtrack.html, November 2007.

[63] PHLAK [Online]. Available: http://www.phlak.org, November 2007.

[64] Skaggs, B., Blackburn, B., Manes, G., Shenoi, S. Network Vulnerability Analysis, Proceedings of MWSCAS-2002. pp. 493-495, Aug 2002.

[65] Metasploit [Online]. Available http://www.metasploit.com, November 2007.

[66] Hilley, S. Anti-forensics with a small army of exploits, *Digital Investigation* 4: 13–15, 2007.

[67] Security Tools [Online]. Available: http://sectools.org, November 2007.

[68] SANS Institute White Paper, "DMZ Lab Security Policy", 2006.

[69] Puketza, N., Mukherjee, B., Olsson, R.A., and Zhang, K. Testing Intrusion Detection Systems: Design Methodologies and Results from an Early Prototype, Proc., National Computer Security Conference (NCSC), Baltimore, MD, pp. 1-10, Oct. 1994.

[70] Day, K., Inside the Security Mind: Making the Tough Decisions, Prentice Hall, pp. 73-105, 2003.

[71] CodeRed [Online]. Available http://www.unixwiz.net/techtips/CodeRedII.html, November 2007.

[72] Ebtables [Online]. Available http://ebtables.sourceforge.net/, November 2007.

[73] Alexa the Web Information Company [Online]. Available http://www.alexa.com, November 2007.

APPENDIX A

Netflow Version 9 Field Type Definitions

Field Type	Value	Length (bytes)	Description
IN_BYTES	1	N (default is 4)	Incoming counter with length N x 8 bits for number of bytes associated with an IP Flow.
IN_PKTS	2	N (default is 4)	Incoming counter with length N x 8 bits for the number of packets associated with an IP Flow
FLOWS	3	N	Number of flows that were aggregated; default for N is 4
PROTOCOL	4	1	IP protocol byte
SRC_TOS	5	1	Type of Service byte setting when entering incoming interface
TCP_FLAGS	6	1	Cumulative of all the TCP flags seen for this flow
L4_SRC_PORT	7	2	TCP/UDP source port number ie : FTP, Telnet, or equivalent
IPV4_SRC_ADDR	8	4	IPv4 source address
SRC_MASK	9	1	The number of contiguous bits in the source address subnet mask ie: the submask in slash notation
INPUT_SNMP	10	N	Input interface index; default for N is 2 but higher values could be used
L4_DST_PORT	11	2	TCP/UDP destination port number ie: FTP, Telnet, or equivalent
IPV4_DST_ADDR	12	4	IPv4 destination address
DST_MASK	13	1	The number of contiguous bits in the destination address subnet mask ie: the submask in slash notation
OUTPUT_SNMP	14	N	Output interface index; default for N is 2 but higher values could be used
IPV4_NEXT_HOP	15	4	IPv4 address of next-hop router
SRC_AS	16	N (default is 2)	Source BGP autonomous system number where N could be 2 or 4

Appendix A (continued)

Field Type	Value	Length (bytes)	Description
DST_AS	17	N (default is 2)	Destination BGP autonomous system number where N could be 2 or 4
BGP_IPV4_NEXT_HOP	18	4	Next-hop router's IP in the BGP domain
MUL_DST_PKTS	19	N (default is 4)	IP multicast outgoing packet counter with length N x 8 bits for packets associated with the IP Flow
MUL_DST_BYTES	20	N (default is 4)	IP multicast outgoing byte counter with length N x 8 bits for bytes associated with the IP Flow
LAST_SWITCHED	21	4	System uptime at which the last packet of this flow was switched
FIRST_SWITCHED	22	4	System uptime at which the first packet of this flow was switched
OUT_BYTES	23	N (default is 4)	Outgoing counter with length N x 8 bits for the number of bytes associated with an IP Flow
OUT_PKTS	24	N (default is 4)	Outgoing counter with length N x 8 bits for the number of packets associated with an IP Flow.
MIN_PKT_LNGTH	25	2	Minimum IP packet length on incoming packets of the flow
MAX_PKT_LNGTH	26	2	Maximum IP packet length on incoming packets of the flow
IPV6_SRC_ADDR	27	16	IPv6 Source Address
IPV6_DST_ADDR	28	16	IPv6 Destination Address
IPV6_SRC_MASK	29	1	Length of the IPv6 source mask in contiguous bits
IPV6_DST_MASK	30	1	Length of the IPv6 destination mask in contiguous bits
IPV6_FLOW_LABEL	31	3	IPv6 flow label as per RFC 2460 definition
ICMP_TYPE	32	2	Internet Control Message Protocol (ICMP) packet type; reported as ((ICMP Type*256) + ICMP code)
MUL_IGMP_TYPE	33	1	Internet Group Management Protocol (IGMP) packet type
SAMPLING_INTERVAL	34	4	When using sampled Netflow, the rate at which packets are sampled ie: a value of 100 indicates that one of every 100 packets is sampled
SAMPLING_ALGORITHM	35	1	The type of algorithm used for sampled Netflow: 0x01 Deterministic Sampling ,0x02 Random Sampling
FLOW_ACTIVE_TIMEOUT	36	2	Timeout value (in seconds) for active flow entries in the Netflow cache

Appendix A (continued)

Field Type	Value	Length (bytes)	Description
FLOW_INACTIVE_TIMEOUT	37	2	Timeout value (in seconds) for inactive flow entries in the Netflow cache
ENGINE_TYPE	38	1	Type of flow switching engine: RP = 0, VIP/Linecard = 1
ENGINE_ID	39	1	ID number of the flow switching engine
TOTAL_BYTES_EXP	40	N (default is 4)	Counter with length N x 8 bits for bytes for the number of bytes exported by the Observation Domain
TOTAL_PKTS_EXP	41	N (default is 4)	Counter with length N x 8 bits for bytes for the number of packets exported by the Observation Domain
TOTAL_FLOWS_EXP	42	N (default is 4)	Counter with length N x 8 bits for bytes for the number of flows exported by the Observation Domain
Vendor Proprietary	43		
IPV4_SRC_PREFIX	44	4	IPv4 source address prefix (specific for Catalyst architecture)
IPV4_DST_PREFIX	45	4	IPv4 destination address prefix (specific for Catalyst architecture)
MPLS_TOP_LABEL_TYPE	46	1	MPLS Top Label Type: 0x00 UNKNOWN 0x01 TE-MIDPT 0x02 ATOM 0x03 VPN 0x04 BGP 0x05 LDP
MPLS_TOP_LABEL_IP_ADDR	47	4	Forwarding Equivalent Class corresponding to the MPLS Top Label
FLOW_SAMPLER_ID	48	1	Identifier shown in "show flow-sampler"
FLOW_SAMPLER_MODE	49	1	The type of algorithm used for sampling data: 0x02 random sampling. Use in connection with FLOW_SAMPLER_MODE
FLOW_SAMPLER_RANDOM_INTERVAL	50	4	Packet interval at which to sample. Use in connection with FLOW_SAMPLER_MODE
Vendor Proprietary	51		
MIN_TTL	52	1	Minimum TTL on incoming packets of the flow
MAX_TTL	53	1	Maximum TTL on incoming packets of the flow
IPV4_IDENT	54	2	The IP v4 identification field
DST_TOS	55	1	Type of Service byte setting when exiting outgoing interface

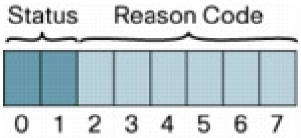
Appendix A (continued)

Field Type	Value	Length (bytes)	Description
IN_SRC_MAC	56	6	Incoming source MAC address
OUT_DST_MAC	57	6	Outgoing destination MAC address
SRC_VLAN	58	2	Virtual LAN identifier associated with ingress interface
DST_VLAN	59	2	Virtual LAN identifier associated with egress interface
IP_PROTOCOL_VERSION	60	1	Internet Protocol Version Set to 4 for IPv4, set to 6 for IPv6. If not present in the template, then version 4 is assumed.
DIRECTION	61	1	Flow direction: 0 - ingress flow, 1 - egress flow
IPV6_NEXT_HOP	62	16	IPv6 address of the next-hop router
BPG_IPV6_NEXT_HOP	63	16	Next-hop router in the BGP domain
IPV6_OPTION_HEADERS	64	4	Bit-encoded field identifying IPv6 option headers found in the flow
Vendor Proprietary	65		
Vendor Proprietary	66		
Vendor Proprietary	67		
Vendor Proprietary	68		
Vendor Proprietary	69		
MPLS_LABEL_1	70	3	MPLS label at position 1 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
MPLS_LABEL_2	71	3	MPLS label at position 2 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
MPLS_LABEL_3	72	3	MPLS label at position 3 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
MPLS_LABEL_4	73	3	MPLS label at position 4 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.

Appendix A (continued)

Field Type	Value	Length (bytes)	Description
MPLS_LABEL_5	74	3	MPLS label at position 5 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
MPLS_LABEL_6	75	3	MPLS label at position 6 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
MPLS_LABEL_7	76	3	MPLS label at position 7 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
MPLS_LABEL_8	77	3	MPLS label at position 8 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
MPLS_LABEL_9	78	3	MPLS label at position 9 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
MPLS_LABEL_10	79	3	MPLS label at position 10 in the stack. This comprises 20 bits of MPLS label, 3 EXP (experimental) bits and 1 S (end-of-stack) bit.
IN_DST_MAC	80	6	Incoming destination MAC address
OUT_SRC_MAC	81	6	Outgoing source MAC address
IF_NAME	82	N	
IF_DESC	83	N (default specified in template)	Full interface name ie: "'FastEthernet 1/0"
SAMPLER_NAME	84	N (default specified in template)	Name of the flow sampler
IN_ PERMANENT _BYTES	85	N (default is 4)	Running byte counter for a permanent flow
IN_ PERMANENT _PKTS	86	N (default is 4)	Running packet counter for a permanent flow
* Vendor Proprietary*	87		
FRAGMENT_OFFSET	88	2	The fragment-offset value from fragmented IP packets

Appendix A (continued)

Field Type	Value	Length (bytes)	Description
FORWARDING STATUS	89	1	Forwarding status is encoded on 1 byte with the 2 left bits giving the status and the 6 remaining bits giving the reason code.  Status is either unknown (00), Forwarded (10), Dropped (10) or Consumed (11). Below is the list of forwarding status values with their means. Unknown • 0 Forwarded • Unknown 64 • Forwarded Fragmented 65 • Forwarded not Fragmented 66 Dropped • Unknown 128, • Drop ACL Deny 129, • Drop ACL drop 130, • Drop Unroutable 131, • Drop Adjacency 132, • Drop Fragmentation & DF set 133, • Drop Bad header checksum 134, • Drop Bad total Length 135, • Drop Bad Header Length 136, • Drop bad TTL 137, • Drop Policer 138, • Drop WRED 139, • Drop RPF 140, • Drop For us 141, • Drop Bad output interface 142, • Drop Hardware 143, Consumed • Unknown 192, • Terminate Punt Adjacency 193, • Terminate Incomplete Adjacency 194, • Terminate For us 195

APPENDIX B

MIBs SUPPORTED IN IMAGE 12.4.3a

ADSL-DMT-LINE-MIB
ADSL-LINE-MIB
ATM-MIB
BRIDGE-MIB
CISCO-AAA-SERVER-MIB
CISCO-AAA-SESSION-MIB
CISCO-AAL5-MIB
CISCO-ACCESS-ENVMON-MIB
CISCO-ADSL-DMT-LINE-MIB
CISCO-ALPS-MIB
CISCO-ASPP-MIB
CISCO-ATM-EXT-MIB
CISCO-ATM-PVCTRAP-EXTN-MIB
CISCO-BSC-MIB
CISCO-BSTUN-MIB
CISCO-BULK-FILE-MIB
CISCO-BUS-MIB
CISCO-CALL-APPLICATION-MIB
CISCO-CALL-HISTORY-MIB
CISCO-CAR-MIB
CISCO-CAS-IF-MIB
CISCO-CDP-MIB
CISCO-CIRCUIT-INTERFACE-MIB
CISCO-CLASS-BASED-QOS-MIB
CISCO-COMPRESSION-SERVICE-ADAPTER-MIB
CISCO-CONFIG-MAN-MIB
CISCO-DATA-COLLECTION-MIB
CISCO-DIAL-CONTROL-MIB
CISCO-DIST-DIRECTOR-MIB
CISCO-DLCSW-MIB
CISCO-DLSW-EXT-MIB
CISCO-DLSW-MIB
CISCO-DOT11-CONTEXT-SERVICES-MIB
CISCO-DSL-CPE-MIB

CISCO-DSP-MGMT-MIB
CISCO-DSPU-MIB
CISCO-EMBEDDED-EVENT-MGR-MIB
CISCO-ENTITY-ASSET-MIB
CISCO-ENTITY-EXT-MIB
CISCO-ENTITY-VENDORTYPE-OID-MIB
CISCO-ENVMON-MIB
CISCO-FLASH-MIB
CISCO-FRAME-RELAY-MIB
CISCO-FRAS-HOST-MIB
CISCO-FTP-CLIENT-MIB
CISCO-GPRS-GTP-MIB
CISCO-H323-TC-MIB
CISCO-HSRP-EXT-MIB
CISCO-HSRP-MIB
CISCO-ICSUDSU-MIB
CISCO-IETF-ATM2-PVCTRAP-MIB
CISCO-IETF-ATM2-PVCTRAP-MIB-EXTN
CISCO-IETF-IP-FORWARD-MIB
CISCO-IETF-IP-MIB
CISCO-IETF-NAT-MIB
CISCO-IF-EXTENSION-MIB
CISCO-IMAGE-MIB
CISCO-IP-STAT-MIB
CISCO-IPMROUTE-MIB
CISCO-IPSEC-FLOW-MONITOR-MIB
CISCO-IPSEC-MIB
CISCO-IPSEC-POLICY-MAP-MIB
CISCO-ISDN-MIB
CISCO-ISDNU-IF-MIB
CISCO-LEC-DATA-VCC-MIB
CISCO-LEC-EXT-MIB
CISCO-LECS-MIB
CISCO-LES-MIB
CISCO-MEMORY-POOL-MIB
CISCO-MOBILE-IP-MIB
CISCO-MVPN-MIB
CISCO-NBAR-PROTOCOL-DISCOVERY-MIB
CISCO-NETFLOW-MIB
CISCO-NTP-MIB
CISCO-OSPF-MIB
CISCO-OSPF-TRAP-MIB
CISCO-PIM-MIB
CISCO-PING-MIB
CISCO-POP-MGMT-MIB
CISCO-PPPOE-MIB
CISCO-PROCESS-MIB

Appendix B (continued)

CISCO-PRODUCTS-MIB
CISCO-QLLC01-MIB
CISCO-QUEUE-MIB
CISCO-RAS-MIB
CISCO-RMON-CONFIG-MIB
CISCO-RMON-SAMPLING-MIB
CISCO-RSRB-MIB
CISCO-RTTMON-MIB
CISCO-SAA-APM-MIB
CISCO-SDLLC-MIB
CISCO-SIP-UA-MIB
CISCO-SMI
CISCO-SNADLC-CONV01-MIB
CISCO-SNAPSHOT-MIB
CISCO-SNMP-TARGET-EXT-MIB
CISCO-STACKMAKER-MIB
CISCO-STUN-MIB
CISCO-SYSLOG-MIB
CISCO-TC
CISCO-TCP-MIB
CISCO-VLAN-IFTABLE-RELATIONSHIP-MIB
CISCO-VLAN-MEMBERSHIP-MIB
CISCO-VOICE-ANALOG-IF-MIB
CISCO-VOICE-ATM-DIAL-CONTROL-MIB
CISCO-VOICE-COMMON-DIAL-CONTROL-MIB
CISCO-VOICE-DIAL-CONTROL-MIB
CISCO-VOICE-DNIS-MIB
CISCO-VOICE-ENABLED-LINK-MIB
CISCO-VOICE-FR-DIAL-CONTROL-MIB
CISCO-VOICE-IF-MIB
CISCO-VOICE-NUMBER-EXPANSION-MIB
CISCO-VOICE-URI-CLASS-MIB
CISCO-VPDN-MGMT-EXT-MIB
CISCO-VPDN-MGMT-MIB
CISCO-VSIMASTER-MIB
CISCO-VTP-MIB
CISCO-WRED-MIB
DIAL-CONTROL-MIB
DLSW-MIB
DNS-SERVER-MIB
DS1-MIB
DS3-MIB
ETHERLIKE-MIB
EVENT-MIB
EXPRESSION-MIB

Appendix B (continued)

FUNI-MIB
HC-RMON-MIB
HDSL2-SHDSL-LINE-MIB
IF-MIB
IGMP-STD-MIB
IMA-MIB
INT-SERV-GUARANTEED-MIB
INT-SERV-MIB
IP-FORWARD-MIB
IPMROUTE-STD-MIB
ISDN-MIB
LAN-EMULATION-CLIENT-MIB
MIP-MIB
MPLS-LDP-MIB
MPLS-LSR-MIB
MPLS-TE-MIB
MPLS-VPN-MIB
MSDP-MIB
NOTIFICATION-LOG-MIB
NOVELL-IPX-MIB
NOVELL-RIPSAP-MIB
OLD-CISCO-APPLETALK-MIB
OLD-CISCO-CHASSIS-MIB
OLD-CISCO-CPU-MIB
OLD-CISCO-DECNET-MIB
OLD-CISCO-FLASH-MIB
OLD-CISCO-INTERFACES-MIB
OLD-CISCO-IP-MIB
OLD-CISCO-MEMORY-MIB
OLD-CISCO-NOVELL-MIB
OLD-CISCO-SYSTEM-MIB
OLD-CISCO-TCP-MIB
OLD-CISCO-TS-MIB
OSPF-MIB
OSPF-TRAP-MIB
PIM-MIB
Q-BRIDGE-MIB
RFC1213-MIB
RFC1231-MIB
RFC1243-MIB
RFC1315-MIB
RFC1381-MIB
RFC1382-MIB
RFC1406-MIB
RMON-MIB
RMON2-MIB
RS-232-MIB

Appendix B (continued)

RSVP-MIB
SMON-MIB
SNA-SDLC-MIB
SNMP-FRAMEWORK-MIB
SNMP-NOTIFICATION-MIB
SNMP-PROXY-MIB
SNMP-TARGET-MIB
SNMP-USM-MIB
SNMP-VACM-MIB
SONET-MIB
SOURCE-ROUTING-MIB
TCP-MIB
UDP-MIB
VRRP-MIB
XGCP-MIB

APPENDIX C

ATTACK RULES

Name	**Excessive Denies to a Particular Port on the Same Host**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	Victim-X	Port-X	Any	>1400	40	High CPU or memory consumption	5
Name	**Same Source Causing Excessive Denies on a Particular Port**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	Any	Port-X	Any	Any	40	—	5
Name	**Same Source, Same Destination, Same Port**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	Victim-X	Port-X	Any	Any	10	—	10

Appendix C (continued)

Name					Ping of Death				
Protocol	Source IP Address	Source Port	Destination IP Address	Destination Port	Device	Packet Size (bytes)	Counts	Host Alarm	Time Range (in secs)
ICMP	Attacker-X	Any	Victim-X	Any	Any	>2048	10	—	10
Name					**Smurf Attack, Ping Flood**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
ICMP	Any	Any	Victim-X	Any	Any	Any	100	—	10
Name					**Finding Server (open to outside) and the Service on it**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	A range of Network IP Addresses (in order, or disordered)	A range of Ports (in order or disordered) (ports < 1024 and special service ports, ie RDP 3389)	Firewall	<1400	40	—	10

Appendix C (continued)

Name	**Distributed Finding Server (open to outside) and the Service on it**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	A range of Network IP Addresses (in order, or disordered)	A range of Ports (in order or disordered) (ports < 1024 and special service ports, ie RDP 3389)	Firewall	<1400	200	—	10
Name	**Invite of Death**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	Victim-X	5060	Any	Any	40	—	10
Name	**E-mail Bomb**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	Victim-X	25	Any	>1000	40	—	60
Name	**Spam**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	Victim-X	25	Any	<250	60	—	60

Appendix C (continued)

Name	**P2P File Sharing**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	Victim-X	3000-4000/6346-6347/6881-6889/6969	Any	Any	40	—	60
Name	**Backdoor (Rootkits, Trojans)**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	Victim-X	Any	Any	Any	1	—	—
Any	Victim-X	Any	Any	Any	Any	Any	40	# of connection	10
Name	**Worm**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	Victim-X	445	Any	Any	1	—	—
Any	Attacker-X	Any	Victim-X	5554/9996	Any	Any	1	—	10
Any	Victim-X	Any	Any	445	Any	Any	40	—	10
Name	**Trojan**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
TCP	Attacker-X	Any	Victim-X	110/143	Any	Any	1	—	—
Any	Victim-X	Any	Attacker-X	5555	Any	Any	1	—	10

Name					IM				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	Victim-X	1863/5190-5193/6660-6669/7000	Any	Any	10	—	60

Name					Pop-up Spam				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
UDP	Attacker-X	Any	Any	1026-1029	Any	Any	10	—	60

Name					Password Attack				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
TCP	Attacker-X	Any	Victim-X	21,22,23,25, 80,110,143, 443,3389,8080	Any	Any	10	—	60
								Wrong Password	

Name					Buffer Overflow				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
TCP	Attacker-X	Any	Victim-X	Port-X	Any	Any	10	—	60
								High Memory Consumption	

Name					DNS Buffer Overflow				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
TCP/UDP	Attacker-X	Any	Victim-X	53	Any	>1000	20	▬ High Memory Consumption	60
Name					**TFTP Buffer Overflow**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
TCP	Attacker-X	Any	Victim-X	69/6970	Any	>1400	10	▬ High Memory Consumption	60
Name					**File/Registry Modification**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	A range of Network IP Addresses (in order, or disordered)	A range of Ports (in order or disordered) (ports < 1024 and special service ports, ie RDP 3389)	Firewall	<1400	40	▬	60
Any	Attacker-X	Any	Victim-X	Port-X	Any	Any	1	▬ File/registry modification	

Appendix C (continued)

Name					**SQL Slammer**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
UDP	Attacker-X	Any	Any	1434	Any	376	10	—	60
Name					**Worm**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Any	Any	Victim-X	Any	Any	Any	1	—	10
TCP/UDP	Any	Any	Victim-X	135/137/139	Any	Any	1	—	
TCP/UDP	Victim-X	Any	Any	69/135/137/445/ 6129/6970	Any	Any	40	—	
Name					**Firewall Overwhelming**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	A range of Network IP Addresses (in order, or disordered)	A range of Ports (in order or disordered) (ports < 1024 and special service ports, ie RDP 3389)	Firewall	>1400	40	—	10
								High CPU or memory consumption trap	

Appendix C (continued)

Name	**IM File Sharing**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	Any	1863/5190/6660-6669/7000	Any	Any	1	—	60
Any	Attacker-X	Any	Any	>1024	Any	>1400	100	—	
Name	**IPSec VPN Overwhelming**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
ESP/AH	Attacker-X	Any	Any	Any	Firewall	Any	100	—	60
UDP	Attacker-X	Any	Any	500	Firewall	Any	100	—	
Name	**Reverse DNS Lookup and Taking Control of Servers**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
UDP	Attacker-X	Any	A range of Network IP Addresses (in order, or disordered)	53	Any	Any	100	—	60
UDP	Attacker-X	Any	DNS	53	Any	Any	100	—	
TCP	Attacker-X	Any	Victim-X	22-23	Any	Any	10	—	
								Wrong Password	

Appendix C (continued)

Name	**Unauthorized DHCP Servers**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
UDP	Network	68	Broadcast	67	Any	Any	100	—	30
UDP	Attacker-X	67	Network	68	Any	Any	100	—	
Name	**Web Server Attack**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	A range of Network IP Addresses (in order, or disordered)	80	Firewall	Any	40	—	20
UDP	Attacker-X	Any	DNS	53	Any	Any	1	—	
TCP	Attacker-X	Any	Victim-X	20-21	Any	Any	10	—	
Name	**SNMP Guess Password**								
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
ICMP (for traceroute)	Attacker-X	Any	A range of Network IP Addresses (in order, or disordered)	Any	Any	Any	40	—	20
UDP	Attacker-X	Any	Victim-X	161	Any	Any	20	—	

Appendix C (continued)

Name					**Trojan**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	A range of Network IP Addresses (in order, or disordered)	80	Firewall	Any	40	—	20
TCP	Attacker-X	Any	Victim-X	80	Any	Any	1	—	
TCP	Attacker-X	Any	Victim-X	1,963	Any	Any	10	—	
Name					**Worm**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	Victim-X	80	Any	Any	1	—	20
TCP	Victim-X	Any	Victim-Y	1521-2484/3872-3891	Any	Any	40	—	
Name					**Arp Spoofing**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
					Switch		100	New Mac-Address	20
Name					**TCP Syn Flooding**				
Protocol	**Source IP Address**	**Source Port**	**Destination IP Address**	**Destination Port**	**Device**	**Packet Size (bytes)**	**Counts**	**Host Alarm**	**Time Range (in secs)**
Any	Attacker-X	Any	Victim-X	Any	Firewall	Any	100	—	60
								Syn Overflows from Victim	

APPENDIX D

Expect, SNMP Trap, Ebtables Configuration Examples

Expect:

```
#!/usr/bin/expect -f
set ip [lindex $argv 0]
set user [lindex $argv 1]
set password [lindex $argv 2]
set adminpass [lindex $argv 3]
set cmd1 [lindex $argv 4]
set cmd2 [lindex $argv 5]
set cmd3 [lindex $argv 6]
spawn /usr/bin/telnet $ip
expect ": "
send "$user\r"
expect ":"
send "$password\r"
expect ">"
send "ena\r"
expect ":"
send "$adminpass\r"
expect "#"
send "$cmd1\r"
expect "#"
send "$cmd2\r"
expect "#"
send "$cmd3\r"
expect "#"
send "q\r"
interact
```

SNMP Trap:

```
snmp-server trap-source GigabitEthernet0/1
snmp-server enable traps snmp authentication linkdown linkup coldstart warmstart
snmp-server enable traps vrrp
snmp-server enable traps ds1
snmp-server enable traps casa
snmp-server enable traps tty
snmp-server enable traps eigrp
snmp-server enable traps flash insertion removal
snmp-server enable traps srp
snmp-server enable traps ds3
snmp-server enable traps envmon
snmp-server enable traps isdn call-information
snmp-server enable traps isdn layer2
snmp-server enable traps isdn chan-not-avail
snmp-server enable traps isdn ietf
snmp-server enable traps ima
snmp-server enable traps bgp
snmp-server enable traps cnpd
snmp-server enable traps config-copy
snmp-server enable traps config
snmp-server enable traps entity
snmp-server enable traps fru-ctrl
snmp-server enable traps event-manager
snmp-server enable traps frame-relay multilink bundle-mismatch
snmp-server enable traps frame-relay
snmp-server enable traps frame-relay subif
snmp-server enable traps hsrp
snmp-server enable traps ipmulticast
snmp-server enable traps msdp
snmp-server enable traps mvpn
snmp-server enable traps ospf state-change
snmp-server enable traps ospf errors
snmp-server enable traps ospf retransmit
snmp-server enable traps ospf lsa
snmp-server enable traps ospf cisco-specific state-change nssa-trans-change
snmp-server enable traps ospf cisco-specific state-change shamlink interface-old
snmp-server enable traps ospf cisco-specific state-change shamlink neighbor
snmp-server enable traps ospf cisco-specific errors
snmp-server enable traps ospf cisco-specific retransmit
snmp-server enable traps ospf cisco-specific lsa
snmp-server enable traps pim neighbor-change rp-mapping-change invalid-pim-message
snmp-server enable traps pppoe
snmp-server enable traps cpu threshold
snmp-server enable traps rsvp
```

```
snmp-server enable traps syslog
snmp-server enable traps l2tun session
snmp-server enable traps aaa_server
snmp-server enable traps atm subif
snmp-server enable traps isakmp policy add
snmp-server enable traps isakmp policy delete
snmp-server enable traps isakmp tunnel start
snmp-server enable traps isakmp tunnel stop
snmp-server enable traps ipsec cryptomap add
snmp-server enable traps ipsec cryptomap delete
snmp-server enable traps ipsec cryptomap attach
snmp-server enable traps ipsec cryptomap detach
snmp-server enable traps ipsec tunnel start
snmp-server enable traps ipsec tunnel stop
snmp-server enable traps ipsec too-many-sas
snmp-server enable traps rtr
snmp-server enable traps rf
snmp-server host 192.168.11.100 version 2c <removed>
```

Ebtables:

```
ebtables -P FORWARD DROP
ebtables -A FORWARD -p IPv4 -j ACCEPT
ebtables -A FORWARD -p ARP -j ACCEPT
ebtables -A FORWARD -p LENGTH -j ACCEPT
ebtables -A FORWARD --log-level info --log-ip --log-prefix EBFW
ebtables -P INPUT DROP
ebtables -A INPUT -p IPv4 -j ACCEPT
ebtables -A INPUT -p ARP -j ACCEPT
ebtables -A INPUT -p LENGTH -j ACCEPT
ebtables -A INPUT --log-level info --log-ip --log-prefix EBFW
ebtables -P OUTPUT DROP
ebtables -A OUTPUT -p IPv4 -j ACCEPT
ebtables -A OUTPUT -p ARP -j ACCEPT
ebtables -A OUTPUT -p LENGTH -j ACCEPT
ebtables -A OUTPUT --log-level info --log-ip --log-arp --log-prefix EBFW -j DROP
```

This is a basic filter configuration which will only let frames made by the protocols IP version 4 and ARP through.

```
ebtables -t nat -A PREROUTING -i eth0 -p ipv4 --ip-dst 192.168.0.0/19 -j dnat --to-dst $MAC_ADDR_OF_ETH1 --dnat-target  ACCEPT
```

This command is a nat rule.