IPSec Traffic Overhead Analysis in Dual Stack IPv4/IPv6 Transition Mechanisms

A thesis submitted in fulfillment of the requirements for the degree of

Master of Science

by

Mathias Mujinga

University of Fort Hare
Together in Excellence

Department of Computer Science

Telkom Centre of Excellence

University of Fort Hare

Alice 5700

South Africa

December 2005

Acknowledgements

First and foremost I would like to give my warmest thanks to the Almighty, who was with me showing me the light along the way.

To my mother and father, I can not thank you enough for bringing me into this world. God bless you.

I would also like to thank Professor Hyppolyte N. Muyingi for guiding me along the way and the Telkom Centre of Excellence for financial and technical support.

I want to express my appreciation to my colleagues and the staff of the Department of Computer Science, for their comments and useful criticisms; guys you mean a lot to me.

Last but certainly not least, I want to thank you Pretty Fundiswa Keke for your emotional support in stressful times, God be with you.

Abstract

Internet Protocol version 6 (IPv6) is the next generation of Internet Protocol proposed by the Internet Engineering Task Force (IETF) to supplant the current Internet Protocol version 4 (IPv4). Lack of security below the application layer in IPv4 is one of the reasons why there is a need for a new Internet Protocol. IPv6 has built-in support for the Internet Protocol Security protocol (IPSec). IPSec is an Internet standard that is designed to provide data confidentiality, integrity and the authentication through the use of the Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols. This paper reports work done in evaluating the implications of the compulsory use of IPSec on the traffic performance of the dual-stack IPv4/IPv6 computer network, as this adds additional headers on the packets.

This work was conducted on Windows platforms. The following protocols; ICMP, HTTP, FTP and TFTP were used in the experiment. The findings show that the IPSec traffic overhead is higher on smaller packets as compared to larger packets for all application protocols tested. This ultimately has an impact on the round trip and download times, which seriously affect the quality of service. There was also a significant difference in the overhead, round trip time and download time of different IPSec protocols implemented separately.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

3DES – Triple-DES
AES – Advanced Encryption Standard
AH – Authentication Header
ALG – Application Layer Gateway
ANSI – America National Standards Institute
CA – Certificate Authority
CIDR – Classless Inter-Domain Routing
CRC – Cyclical Redundancy Check
CRL – Certificate Revocation List
DARPA – Defense Advanced Research Projects Agency
DES – Data Encryption Standard
DHCP – Domain Host Configuration Protocol
DHCPv6 – Domain Host Configuration Protocol for IPv6
DiffServ – Differentiated Services
DNS – Domain Name System
DOD – Department Of Defense
DoS – Denial of Service
DSCP – Differentiated Services Code Point
DSI – Dynamic Systems Initiative
DSTM – Dual Stack Tunneling Mechanism
DV – Digital Video
ECC – Error Checking and Correction
ESP – Encapsulating Security Payload
EUI-64 – Extended Unique Identifiers for 64-bits
FTP – File Transfer Protocol
HIDS – Host Intrusion Detection System
HTTP – Hyper-Text Transfer Protocol
IANA – Internet Assigned Numbers Authority
ICMP – Internet Control Message Protocol
ICV – Internet Check Value
IETF – Internet Engineering Task Force
IIS – Internet Information Services
IKE – Internet Key Exchange
IP – Internet Protocol
IPSec – Internet Protocol Security
IPv4 – Internet Protocol version 4
IPv6 – Internet Protocol version 6
ISAKMP – Internet Security Association and Key Management Protocol
ISATAP – Intra-Site Automatic Tunnel Addressing Protocol
ISO – International Standardization Organization
IV – Initialization Vector
L2TP – Layer 2 Tunneling Protocol
LAN – Local Area Network
LLC – Logical Link Control
MAC – Media Access Control

MD5 – Message Digest 5
MIME – Multipurpose Internet Mail Extensions
MSS – Maximum Segment Size
MTU – Maximum Transmission Unit
NAT – Network Address Translation
NAT-PT - Network Address Translation Protocol Translation
NAT-T - Network Address Translation Traversal
NIST – National Institute of Standards and Technology
NLA ID – Next-Level Aggregation Identifiers
NNTP – Network News Transfer Protocol
OSI – Open System Interconnection
OUI – Organizational Unit Identifiers
PHB – Per-Hop behavior
PPP – Point-to-Point Protocol
PPTP – Point-to-Point Tunneling protocol
QoS – Quality of Service
RA – Router Advertisement
RFC – Request for Comment
RS – Router Solicitation
RSA – Rivest, Shamir and Adlerman
RSVP – Resource Reservation Protocol
RTT – Round Trip Time
SA – Security Association
SADB – Security Association Database
SHA1 – Secure Hash Algorithm version 1
SMTP – Simple Mail Transfer Protocol
SPD – Security Policy Database
SPI – Security Parameter Index
SSL – Secure Socket Layer
TCP – Transmission Control Protocol
TCP/IP – Transmission Control Protocol/Internet Protocol
TFTP – Trivial File Transfer Protocol
TLA ID – Top-Level Aggregation Identifiers
TLS – Transport Layer Security
ToS – Type of Service
TTL – Total Time to Live
UDP – User Datagram Protocol
URL – Uniform Resource Locator
UTP – Unshielded Twisted Pair
VN – Virtual Network
VoIP – Voice over IP
VPN – Virtual Private Network

CHAPTER ONE – INTRODUCTION

Chapter one introduces the investigation of the traffic implications of using Internet Protocol Security (IPSec) as a mandatory protocol in IP. This chapter is laid out as follows; the introduction is followed by the background and problem statement, focus and scope of this research. The expected output is then followed by the thesis organization and finally the chapter conclusion.

1.1 Introduction

The Internet Protocol (IP) is the protocol that operates as the backbone of the Internet and networking in general. The initial IP was first published in 1981, in RFC 791 [3] and is now generally known as IPv4. During those days a 32-bit address space was deemed enough in respect of the available Internet demand, but as Internet growth increased year after year the address space became depleted. This and other shortcomings of IPv4 led the Internet Engineering Task Force (IETF) to consider the introduction of a new protocol as early as the early 1990s. IPv4 like any other protocol has its own weaknesses. These include; obviously limited address space, the increasing size of routing tables, the need for end-to-end networking, the lack of support for mobile nodes and the absence of in-built security.

Internet Protocol version 6 (IPv6) is the next generation Internet Protocol proposed by the IETF in RFC 2460 [4], published in 1998 to supplant IPv4. IPv6 is intended to address the above mentioned as well as other weaknesses of IPv4. There is no doomsday transition expected, because IPv6 deployment is transition-rich to prevent isolation of the IPv4 nodes, hence IPv4 and IPv6 will coexist for an extended period of time. This is because of a number of factors, which include lack of urgency since some places like North America still have enough addresses to last for a couple of years. While in places like Asia the need for addresses is critical and there are noticeable developments in the deployment of IPv6. This lack of demand coupled with the cost of upgrading networks to be IPv6 enabled are the main reasons for lack of IPv6 deployment across the world.

There are a number of new features in IPv6; these include those features that were not addressed in the initial design of the IPv4 and some improvements on initial IP features. IPv6 includes the following features; a larger address space, router and host auto-configuration, improved multi-homing, support for mobility, improved security and ultimately improved quality of service (QoS) [1][4]. IPv4 was not designed with an in-built security mechanism below the application layer; hence applications were required to implement their own security, using protocols such as Socket Secure Layer (SSL).

Security has been a hotspot in the Internet community for a long time, because the Internet has become the major component of communication, hence the need for privacy and confidentiality on the wire has grown by the day. IPSec was introduced to provide a more secure IP. IPSec operates at the network layer, in a way that is completely transparent to applications, and much more powerful, because the applications do not need to have any knowledge of IPSec to be able to use it. Encrypted tunnels can be created as well as Virtual Private Networks (VPNs), or simply an encryption between computers. Since there are so many options, IPSec is rather complex. This protocol is mandatory in IPv6 as a way of improving security but is optional in IPv4. The Authentication Header (AH) and the Encapsulating Security Payload (ESP) protocols are the two protocols of IPSec that are applied as extension headers in an IPv6 packet; which means they increase the size of the IP packet. This paper will investigate the cost in terms of frame size and delay performance when transmitting traffic on computer networks, with IPSec enabled on IPv4 and IPv6, on a variety of application protocols. Some of our findings have been presented in [84-85].

1.2 Background and Problem Statement

The Internet community has and still is trying to limit the effects of the dwindling address space. There are a number of mechanisms in place to alleviate this problem, but the reality is that, the temporary solutions will not last for long and they are costly in terms of IP performance and QoS. The other problem which the Internet community is fighting is the growing size of the non-default routing tables. This

problem has a direct impact on the performance of routing IP traffic from source to destination. These problems coupled with other shortcomings of IPv4 including the desperate need for reliable security on the Internet has prompted IETF to introduce IPv6 into the picture, to replace IPv4.

Effective IP security is one feature which the Internet community really needs, but when the current version of IP was designed security was not properly addressed. As time went by and the Internet became a major communication tool security became an issue, the IETF introduced IPSec to solve the security weaknesses in IPv4 in 1998, published in RFC 2401 [2]. IPSec was designed to work on IPv4 and any other future IP protocol, IPv6 included, even if it was introduced after IPv4. The way IPSec provides security services in both IPv4 and IPv6 is slightly different. When used with IPv4, IPSec headers are inserted after the IPv4 header and before the next-layer protocol header. While with IPv6 this is applied in the form of additional extension headers [15]. This obviously increases the overhead of an IP datagram, and since this protocol is mandatory on IPv6 this overhead becomes increasingly significant. Given the processing power needed to encrypt and decrypt the data with the most powerful of cryptographic algorithms, this process requires more processing power from the processing node. Therefore the use of IPSec has serious effects on the performance of a computer and the network to which it is connected. Two main ways, the round trip time and the throughput, might be affected. Although the introduction of IPv6 solves or alleviates the problems of IPv4 to some extent, there are still some schools of thought that are skeptical of its benefits. The performance implications brought about by the compulsory use of IPSec in IPv6 is one area of concern for some. Not much research has been conducted in this area, mainly because IPv6 is a relatively new protocol, with most of its issues still under discussion. Therefore finer details like performance effects have not yet been explored in detail.

There has been some research done on the performance implications of IPSec deployment. In [5] the authors evaluated the performance overheads under a range of different bandwidth and different processors, on throughput and processor; single and dual, when communicating over a secured VPN on IPv4 infrastructure, using a

Linux 2.6.1 kernel. The findings showed that, the overhead differs from one processor type to the other, even when dual processors of the same type were used.

In another work, [6] evaluated the performance of data transmissions with IPv4 and IPv6 networks. The results show that IPSec obviously degrades the network performance in terms of throughput and end-to-end delay for large data transmission and for the actual application. The authors concentrated on Digital Video (DV) transmission as the application. Their results showed that, for large data transmissions, when authentication and encryption are applied, the throughput degrades to 1/9 compared with the throughput without authentication or encryption. As for the DV transmission, the end-to-end throughput showed that, 1/10 of the video information can be successfully transferred from the source node to the destination node to obtain an adequate quality of DV transmission. The two works mentioned above were conducted on Linux 2.6.1 kernel and FreeBSD 2.2.8 with KAME 19990908-stable respectively. [5] considered IPv4 only and concentrated on the impact of IPSec on different processor types. While [6] considered IPv6 but concentrated on one application, DV transmission. This work uses Windows IPv6 and IPSec implementations and evaluates a variety of IP packet sizes over HTTP, FTP and ICMP protocols. The additional frame overhead induced by IPSec on both IPv4 and IPv6 on these protocols was evaluated and its impact on average round trip times and throughput was noted. This is achieved by comparing traffic with IPSec on and IPSec off.

HP also did an IPSec performance evaluation on their HP-UX server [54] that evaluated the following aspects of the server: throughput, CPU utilization and service demand for transferring data with HP-UX IPSec on one-way (single-CPU) rx2600 HP-UX. There findings showed that in general, performance of sending IP data degrades by almost 3% and it significantly affects CPU performance and service demand.

There are three main categories of transition mechanisms to IPv6 namely; dual stack, tunneling and protocol translation. The first two involves encapsulating the IPv6 packet in an IPv4 packet on the source node and transmitting it on an IPv4 network infrastructure and then the destination node decapsulates the packet back from IPv4

to IPv6. Network Address Translation – Protocol Translation (NAT-PT) allows IPv6 nodes to communicate with the IPv4 nodes transparently, using a single IPv4 address. The TCP/UDP ports of the IPv6 nodes are translated into TCP/UDP ports of the registered IPv4 address [10-11].

1.3 Design Considerations and Implementation Decisions

The work proposed here evaluates the cost of using IPSec on the LAN environment. This evaluation will include a comparison between the performance of IPSec on both IPv4 and IPv6. The LAN provides an environment which can be controlled without outside intervention. The evaluation will be conducted using Microsoft Windows® operating systems that include the latest IPv6 and IPSec implementations, which are Windows® 2003 Server Service Pack 1 (SP1) and Windows® XP Professional Service Pack 2 (SP2) [7].

1.3.1 Transition Mechanisms

There are a number of transition mechanisms available in Windows® and these are being used for the experiments, although they are not designed to be used in a production environment, the IPv6 protocol for Windows XP and Windows Server 2003 supports a number of features that make it possible us to set up and test IPSec functionality over an IPv4 routing infrastructure [8]. It should be noted that, security provided by these transition mechanisms is limited in scope and depth [9].

Dual Stack Translation Mechanism (DSTM) was the primary method in the IPv6 experiments; 6to4 in particular. Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) was another method available in Windows. 6to4 is a tunneling addressing mechanism that enables communication between two IPv6 computers that live in an IPv4 environment [12]. ISATAP is a transition mechanism that can be used within a site to connect isolated IPv6/IPv4 dual stack hosts to the IPv6 Internet. It contains the word "automatic' in its name because once an ISATAP server/router has been set up only the clients must be configured to connect to it [13]. 6to4 was chosen because it is applicable on the global Internet and intra-site networks, while ISATAP is mainly

an intra-site transition mechanism. However, some experiments were conducted using ISATAP and it was found to have the same frame overhead impact as 6to4.

1.3.2 Experimentation

The experimental environment is an isolated network, derived from Microsoft's IPSec Setting up the infrastructure [14], and is comprised of IPv4/IPv6 dual stack nodes. The network consists of five computers of which three are servers running Windows 2003 Server SP1 and two clients running Windows XP Professional SP 2. All nodes have Microsoft TCP/IP version 6 protocol stack enabled. IPSec was configured on the domain controller. A third party protocol analyzer is being used for packet capturing and analysis. A detailed implementation of the test bed is given in Chapter 4.

1.4 Expected Output

This research will give an insight into the quantitative traffic expense, which the mandatory use of IPSec will bring into networks, and a model of how and when to use it in a network will be given. The use of IPSec compromises processor and network performance, but when security is of paramount importance performance has to be sacrificed to some extent. This is achieved by screening the traffic to which IPSec has to be applied. Therefore the dilemma which faces the network administrators is that of deciding which traffic to screen and which not to, as this opens security holes in a network. This predicament compromises IPSec as a security mechanism to prevent intrusion in a network, hence the reason why IPSec deployment is concentrated in virtual private networks (VPNs).

A VPN is an extension of a private network that encompasses encapsulated, encrypted, and authenticated links across shared or public networks such as the Internet. A common use for IPSec is to secure the links of a virtual network (VN), creating a VPN. The topology of an IPSec VPN commonly consists of IPSec tunnel mode virtual links, as required by the IPSec architecture when the communicating peers are gateway pairs, or a host and a gateway.

VPN connections typically provide remote access and router-to-router connections to private networks over the Internet. IPSec in the Microsoft® Windows® Server 2003 operating system protects networks from active and passive attacks by securing IP packets through the use of packet filtering, cryptography, and the enforcement of trusted communication. IPSec is useful in host-to-host, VPN, site-to-site (also known as gateway-to-gateway or router-to-router), and secure server scenarios, and can be managed by using Group Policy or scripted by using command-line tools. Knowing when and how to deploy IPSec efficiently will help to save two of the most valued resources in the Internet community; the scarce and expensive bandwidth and computer processing power.

1.5 Thesis Organization

This thesis consists of six chapters, each chapter has an abstract and a core; the first chapter is this introduction, which introduces the problem statement and the guidelines on how to achieve the expected output.

Chapter two introduces the background of IP and its versions. There are two versions of IP; the current version IPv4 which was introduced more than two decades ago and IPv6 which is in the pipeline and in the processes of replacing the current version. Chapter three provides a detailed analysis of security mechanisms in IP. IPSec works on both IP versions. But the main difference is that; it is optional in IPv4, simply because it was introduced later after IPv4 was already defined, while its implementation is compulsory in IPv6.

Chapter four describes in detail how the test bed was implemented, explaining choices made in designing and implementing the test bed. In this chapter we give the structure and the components of the test bed that were used to conduct the experiments. Chapter five gives the experiments that were carried out in this research. It also includes the results and results analysis based on the tests findings. Chapter six summarizes and concludes the thesis, giving recommendations based on results and findings. It also includes possible future work of this research.

CHAPTER TWO – INTERNET PROTOCOLS

This chapter gives a detailed background of the two versions of IP that provide networking services on the Internet. The current version of IP; IPv4 is introduced, and its characteristics, weaknesses and temporary solutions to these weaknesses. This will be followed by a detailed introduction of the next generation Internet protocol; IPv6, concentrating on the improvements on the current version.

2.1 Introduction

The Internet Protocol (IP) is the backbone of the today's Internet; this protocol has been holding the Internet for more than two decade with exceptional results. The Internet is currently based on the Internet Protocol version 4 (IPv4). This standard described in RFC 781 [3], was published in September 1981. At that time, there were only approximately one thousand computers on the Internet, and virtually all were large systems in universities and government offices. Computers designed for single users were almost non-existent. In 1981, the four billion addresses available under the IPv4 standard seemed like an address space so large that, there was no need to be conservative with IP address allocation. As a result early address allocations were not as well distributed as later allocations. But since its inception the Internet has grown and is continuing to grow at phenomenal rate. Today, it is estimated that there are 200 million computers on the Internet and roughly 800 million Internet users.

There are other protocols that work in conjunction with IP to achieve the communication between systems across the globe at those magical speeds. In this chapter a detailed view will be presented of how some of these protocols interact with IP.

2.2 Internet Protocol version 4

Internet Protocol (IP) is the fundamental building block for all control and data communications exchange across and with the Internet. Internet Protocol version four (IPv4) is the current version which is widely used on the Internet today. This section will focus in detail on IPv4 features and how the protocol works.

IP is a universal data delivery protocol across most network types. Data is sent over an IP network in connectionless datagrams. Datagrams use a best effort or connectionless delivery service between the source and destination; connectionless in that no session is established between the source and destination before sending data. Each datagram may take a different route through the network. A datagram is any connectionless protocol message at any protocol layer, while a packet is any protocol message at the network or transport layer. Hence in IPv4 which is a connectionless protocol, the words 'packet' and 'datagram' may sometimes be used interchangeably. A datagram is comprised of the control information and the payload data. The former is used to identify the source and the destination of the data and to manage the datagram while it is in transit over the link or through other system layers. This information is grouped together at the start of the datagram in a header. The IP protocol provides five main functions; basic unit for data transfer, addressing, routing, quality of service (QoS) and fragmentation of datagrams.

In this chapter a detailed insight is given into the core protocols of the Transmission Control Protocol/Internet Protocol (TCP/IP) suite, which includes the above two protocols and a host of others. TCP/IP is the protocol suite that holds the Internet together. Its development was started by the United States Department of Defense (DOD) in 1969 under the Defense Advanced Research Projects Agency (DARPA) which established it as a standard in 1976. The initial model designed was Open System Interconnection Reference model. The TCP/IP protocol suite is now used for communications, whether in voice, video, or data.

The Internet Protocol (IP) is situated at the network layer of the OSI model and is designed to interconnect packet switched communication networks to form an Internetwork. IPv4 uses 32-bit addresses, allowing for a theoretical maximum of 4,294,967,295 unique addresses which, practically is a very small number. It transmits blocks of data called datagrams received from the IP's upper–layer software to and from source and destination hosts. It provides a best effort delivery service between the source and destination; it is connectionless in that it does not establish a session between the source and destination before it transmits its data.

This eventually is dealt with by TCP protocol. This is the layer that is also responsible for the IP protocol addressing [17-18].

2.2.1 IPv4 Header

An IP datagram begins with a common header; there are twelve or more header fields in the IP header, each with a defined function to be performed by any receiving (intermediate/destination) station, as shown on Figure 2-1.

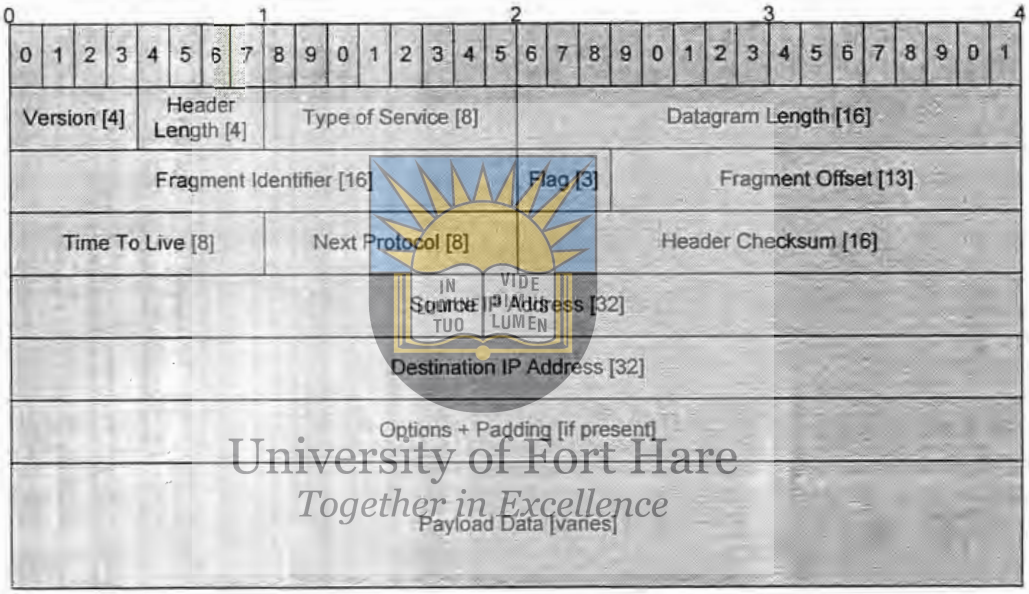


Figure 2-1: IPv4 header format

The functions of each field are explained in the next sections.

2.2.1.1 Version

Version is the first 4-bit field that indicates the version of IP implemented by the network; actually the version number is equal to 4.

2.2.1.2 Internet Header Length

Internet Header Length using 4 bits is the length of the IP header in 32-bit words, and thus points to the beginning of the data. The minimum value for a correct header is 5 and the maximum value should be 64 bytes.

2.2.1.3 Type of Service (ToS)

This field provides an indication of the abstract parameters of the quality of service (QoS) desired. These parameters are to be used to guide the selection of the actual service parameters when transmitting a datagram through a particular network. Several networks offer service precedence, which somehow treats high precedence traffic as more important than other traffic (generally by accepting only traffic above certain precedence at time of high load). The type of service field has been redefined to the Differentiated Services (DiffServ) field.

2.2.1.3.1 DiffServ

DiffServ is basically a method of trying to improving the QoS on the Internet, by giving precedence to traffic. The DiffServ field supersedes the existing definitions of the ToS octet and the IPv6 traffic class octet. In DiffServ, externally observable forwarding treatments at a single node are described by the term per-hop behavior (PHB). Each PHB is represented by a 6-bit value called a Differentiated Services code point (DSCP), all traffic with the same DSCP are awarded the same forwarding treatment. The first 6 bits of the DiffServ field are used as a DSCP to encode the PHB for a packet at each DiffServ node. The remaining 2 bits are currently unused and the DSCP should be treated as an index, and the mapping of DSCPs to PHBs must be configurable [75].

2.2.1.4 Total Length

Total Length is the length of the datagram, measured in octets, including internet header and payload data. This field defines the length of a datagram to be up to 65,535 octets. Such long datagrams are impractical for most hosts and networks. All hosts must be prepared to accept datagrams of up to 576 octets (whether they arrive whole or in fragments). It is recommended that hosts only send datagrams larger than 576 octets if they have assurance that the destination is prepared to accept the larger datagrams. The number 576 is selected to allow a reasonable sized data block to be transmitted in addition to the required header information. For example, this size allows a data block of 512 octets plus 64 header octets to fit in a datagram. The

maximal IP header is 64 octets, and a typical value is 20 octets, allowing a margin for headers of higher level protocols [71].

2.2.1.5 Fragment Identifier

An identifying value assigned by the sender to aid in assembling the fragments of a datagram. 65535 fragments of an original packet are allowed and a path MTU will determine the number of fragments needed to allow easy transmission.

2.2.1.6 Flags

Various control flags mostly to define fragmentation needs. Bit 0: reserved, must be zero. Bit 1: (DF), 0 = May Fragment, 1 = Do not Fragment. Bit 2: (MF), 0 = Last Fragment, 1 = More Fragments.

2.2.1.7 Fragment Offset

This field indicates where in the datagram this fragment belongs. The fragment offset is measured in units of 8 octets (64 bits). The first fragment has offset zero.

2.2.1.8 Time to Live

This field indicates the maximum time the datagram is allowed to remain in the internet system. If this field contains the value zero, then the datagram must be destroyed. This field is modified in IP header processing. The time is measured in units of seconds, but since every module that processes a datagram must decrease the time to live (TTL) by at least one even if it process the datagram in less than a second, the TTL must be thought of only as an upper bound on the time a datagram may exist. The intention is to cause undeliverable datagrams to be discarded, and to bind the maximum datagram lifetime.

2.2.1.9 Protocol

This field indicates the next level protocol used in the data portion of the IP datagram. Each protocol has its specific value, for example TCP has a protocol ID of 6 and UDP has 17.

2.2.1.10 Header Checksum

The checksum checks errors on the header only. Since some header fields change (e.g., time to live), this is recomputed and verified at each point where the internet header is processed. The checksum algorithm is: The checksum field is the 16 bit one's complement of the one's complement sum of all 16-bit words in the header. For purposes of computing the checksum, the value of the checksum field is zero. This is a simple to compute checksum and experimental evidence indicates it is adequate, but it is provisional and may be replaced by a cyclical redundancy check (CRC) procedure, depending on further experience [3].

2.2.1.11 Source Address

The source address is a 32-bit for IPv4 address.

2.2.1.12 Destination Address

The destination address is a 32-bit for IPv4 address.

2.2.1.13 Options

The options may or may not appear in datagrams. They must be implemented by all IP modules (host and gateways). What is optional is their transmission in any particular datagram, not their implementation. In some environments the security option may be required in all datagrams. The option field is variable in length of up to 44 bytes. There may be zero or more options. There are two cases for the format of an option: Case 1: A single octet of option-type. Case 2: An option-type octet, an option-length octet, and the actual option-data octets. The option-length octet counts the option-type octet and the option-length octet as well as the option-data octets.

The option-type octet is viewed as having 3 fields, of which the following indicate the option types: 1 bit copied flag, for options copied to all fragments on fragmentation, 2 bits option class and 5 bits option number. The copied flag indicates if this option is copied into all fragments on fragmentation: 0 = not copied, 1 =

copied. The option classes are: 0 = control, 1 = reserved for future use, 2 = debugging and measurement, 3 = reserved for future use.

2.2.1.14 Padding

The internet header padding is used to ensure that the IP header ends on a 32 bit boundary. The padding is zero or less than 3 bytes because the smallest option (1-byte) requires 3 more bytes of padding [3].

2.3 The OSI Model

The Open System Interconnection (OSI) Reference Model defines a networking framework of TCP/IP protocol suite implementation in seven layers. Control is passed from one layer to the next, starting at the application layer in one station, and proceeding to the bottom layer, over the channel to the next station and back up the hierarchy. This model was developed by the International Standardization Organization (ISO). The layers are shown on Figure 2-2, followed by the description and functions of each layer.

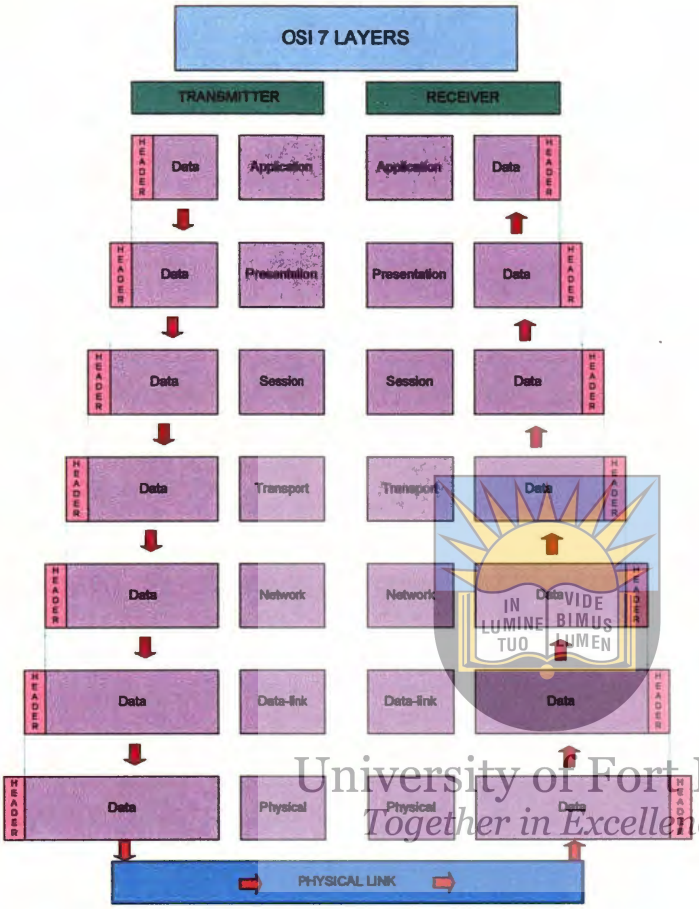


Figure 2-2: The Seven Layers of the OSI Model

- Layer 7: Application – supports applications and end-user processes; provides services such as email, file transfer and real-time applications.

- Layer 6: Presentation – this layer provides independence from different data presentations, by translating from application to network format and vice versa. It formats and encrypts data to be sent across a network, providing freedom from compatibility problems. Also known as syntax layer, it is responsible for formatting, encryption and compression of data. Refer to section 2.2.3 for the importance of data compression in network communication.

- Layer 5: Session – establishes, manages and terminates connections (sessions) between applications. It sets up, coordinates, and terminates conversations,

exchanges and dialogues between the applications at each end. Deals with session and connection coordination.

- Layer 4: Transport – provides transparent transfer of data between end systems, or hosts and is responsible for end-to-end error recovery and flow control. Ensures complete data transfer. Comprised of the following protocols; User Datagram Protocol (UDP) documented in [68] that delivers packets without any flow control and error recovery and Transmission Control Protocol (TCP) documented in [67] that provides delivery guarantees.

- Layer 3: Network – provides switching and routing technologies, creating logical paths, known as virtual circuits for transmitting data from node to node. Responsible for routing, forwarding, addressing, internetworking, error handling, congestion control and packet sequencing

- Layer 2: Data Link – this layer encodes and decodes data packets into bits. It handles errors in the physical layer, flow control and frame synchronization. It is divided into two sub-layers – the Media Access Control (MAC) layer and the Logical Link Control (LLC) layer. The MAC layer controls how a computer on the network gains access to the data and permission to transmit it, while the latter controls frame generation and synchronization and defines bridging, virtual path, for example packet-switched or circuit-switched. Flow control and error checking may be provided by both or one of them. Compression may also occur here for example in point-to-point protocol (PPP) transmission.

- Layer 1: Physical – conveys the electrical impulses, light or radio signal (also known as bit-stream), through the network at the electrical and mechanical level. It provides the hardware means of transmitting and receiving data on a carrier, including defining cables, cards and physical aspects.

The OSI model was modified by the Internet Engineering Task Force (IETF) from a seven layer infrastructure to only four layers.

2.3.1 Compression and Encryption

Compression has to take place before encryption. The reason is that compression relies upon patterns that can be replaced with tokens to make compression possible, while encryption attempts to remove all patterns. Therefore, compression after encryption yields the opposite result. The other issue is packet fragmentation. Encryption without prior compression can lead to packet fragmentation because of the additional header information that encrypted packets must carry. If the process of adding this additional information takes the packet size beyond the path MTU on a given link, fragmentation takes place. That means that instead of a single packet, two packets, each with its own header, now cross the network. The net result is more traffic, more routing, more CPU involvement. Yet fragmentation can be avoided with even a very modest level of compression [76]. Ultimately, compression saves bandwidth and improves QoS of a communication channel.

2.4 TCP/IP Model

TCP/IP is a suite of communication protocols used to connect nodes on the Internet. Unlike the seven layer OSI model, TCP/IP is a four layer model that originated from the OSI model and can be mapped onto the seven layers. The major protocols are TCP and IP together with a number of other protocols. IP is charged with pumping packets into the communication channel, while TCP is charged with making sure they get to the destination by; 1) opening and closing the session, 2) packet management, 3) flow control, and 4) error detection and handling.

Figure 2-3 shows a comparison of the two protocol stacks; OSI and TCP/IP. A collection of compatible protocol layers is referred to as a stack.

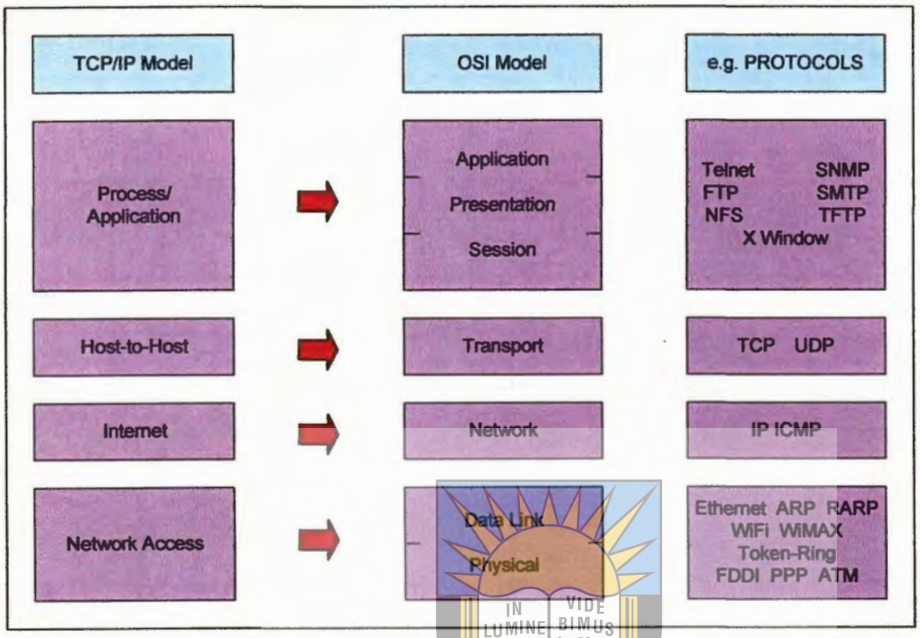


Figure 2-3: A comparison of the OSI and TCP/IP models and the protocols that operate on each layer

- Layer 4: Application – supports application and end-user processes; provides services such as email, file transfer, voice and video.

- Layer 3: Transport – provides a transparent delivery service to the application layer by transferring data between end systems, or hosts and is responsible for end-to-end error recovery and flow control by establishing, maintaining and terminating end-to-end network communication. It ensures complete data transfer.

- Layer 2: Network – is responsible for logical device addressing, forwarding, data packaging, manipulation and delivery, routing, switching, error handling, congestion control and packet sequencing between hosts.

- Layer 1: Network Access – establishes direct connection to physical media and handles data flow control using the mechanisms as in the OSI model under the data link and physical layers. Manages physical addresses of hosts.

2.5 How TCP/IP Works

TCP/IP provides two primary services: best effort packet delivery and reliable byte-stream transport. TCP/IP has several distinguishing features that have led to its popularity. These include:

Network Topology Independence – TCP/IP is used on bus, ring, and star networks. It is used in local-area networks as well as wide-area networks.

Physical Network Hardware Independence – TCP/IP can utilize Ethernet, token ring, or any number of physical hardware variations.

Open Protocol Standard – The TCP/IP protocol suite standard is freely available for independent implementation on any computer hardware platform or operating system. TCP/IP's wide acceptance and the fact that TCP/IP is available on platforms ranging from supercomputers to hand-held devices makes it an ideal set of protocols to unite different hardware and software.

Universal Addressing Scheme – Each computer on a TCP/IP network has an address that uniquely identifies it, so that any TCP/IP enabled device can communicate with any other on the network. Each packet of data sent across a TCP/IP network has a header that contains the address of the destination computer as well as the address of the source computer.

Powerful Client-Server Framework – TCP/IP is the framework for powerful and robust client-server applications that operate in local-area networks and wide-area networks.

Application Protocol Standards – TCP/IP does not just provide the programmer with a method for moving data around a network among custom applications. It also provides the underpinnings of many application-level protocols that implement such common functionality as e-mail and file-transfer capabilities.

2.6 IPv4 Address Space

IPv4 addresses are all 32 bits long, grouped in four bytes. This type of address consists of 4 bytes, which are represented as decimal values separated by periods, as in:

192.96.11.10

In order to ensure uniqueness, IP addresses are assigned in part by the Internet Assigned Numbers Authority (IANA). The IPv4 address scheme uses a hierarchical structure. The bits in an IP address are allocated for Network and Host fields, which specify a network and host number, respectively. IPv4 addresses are divided into five classes, which range from A to E.

IPv4 address classes are distinguished by the high-order bits: 0 for class A, 10 for class B, 110 for class C, 1110 for class D, and 1111 for class E. Two particular network addresses – 0 and those with all 1s – are reserved. Network address 0 is reserved for the originating entity (network or host), and address 255 is used for broadcasts.

- Class A: is used for very large networks (networks with a large number of nodes). This class uses 7 bits for network and 24 bits for Host. The high-order bit is 0 in such an address. There are 126 possible networks each with 16777214 hosts per network.

- Class B: is used for medium-size networks, such as networks that span a large college campus. This class uses 14 bits for network and 16 bits for Host. The two high-order bits are set to 10. This address class is also popular for local-area networks (LANs), particularly if they use subnetting. This class allows for 16384 with 65534 hosts each. The address range contains networks 128.0.0.0 through 191.255.0.0, the network number is in the first two octets.

- Class C: is used for small networks (those with no more than 255 nodes). This class allocates 21 bits for network and only 8 bits for Host. The three high-order bits are 110. Class C networks range from 192.0.0.0 through 223.255.255.0, with

the network number contained in the first three octets. This class allows for nearly 2 million networks with up to 254 hosts.

- Classes D, E and F: are addresses falling into the range of 224.0.0.0 through 254.0.0.0. These are either experimental or are reserved for special purpose use and do not specify any network. IP multicast, which is a service that allows material to be transmitted to many points on an internet at one time, has been assigned addresses from within this range.

The use of subnetting provides additional flexibility in addressing. A subnet is a portion of a network or an internetwork that can be viewed from the outside as a single element. An IP address that uses subnetting has three types of information: network, subnet, and host. Subnets are identified by combining an address with a mask, which is a bit pattern that cancels out unwanted bits, so that only the bits of interest remain.

2.7 IPv4 Weaknesses

IPv4 is a very useful protocol on the Internet, but it has weaknesses which have prompted the Internet community to look for a replacement. At the moment there are various ways of trying to solve the shortcomings of IPv4, but these are temporary and in the long term problems will still persist. Below are various weaknesses of IPv4 and their respective temporary solutions.

- Address Space – IPv4 addresses have been depleting at an alarming rate since the early 1990s, due to the high growth rate which the Internet is experiencing. IPv4 address space supports roughly four billion possible addresses, of which a considerable number is reserved for private network use, loop back addresses, multicast and unspecified future use.

- Size of Routing Tables – the alarming growth rates of the Internet are directly causing an alarming growth of the Internet's non-default routers. Non-default routers are the backbone routers whose routing tables must reflect routing information for every connected network in the world. Some experts argue that

the growing size of the routing tables' scalability is a more urgent reason for migrating to IPv6 than the dwindling address space issue [15].

- End-to-Endness – in trying to reduce the rate at which the IP addresses are running out, the Internet community introduced network address translation (NAT), which completely eliminates the end-to-end connection between source and destination nodes that is so fundamental to most multimedia applications on the Internet [15].

- Security – IPv4 was not built with security in mind, hence the IP Security Protocol (IPSec) that provides security to the IP layer was introduced later on after the introduction and deployment of IPv4, although security mechanisms may be built in all other OSI layers. Although IPSec can be patched in IPv4 protocol stack, its performance can not be compared to that of a built-in protocol.

- Configuration – IPv4 node configuration is a still a difficult task, there are two mechanisms for configuring nodes; static configuration under which nodes are assigned IP addresses that do not change over time and dynamic configuration also referred to as stateless auto-configuration, that gives IP addresses to previously recognized and authorized nodes.

- Mobility – the demand for IP mobility is increasing and IPv4 is not mobile enough to support new applications and services that require mobility.

- Performance – IPv4 header uses options field to specify some form of performance, this varies the size of the header which in turn compromises the processing of the packets on routers and end nodes, as well as the link bandwidth.

2.8 Temporary Solutions

There have been a number of strategies implemented to cope with the unprecedented growth of the Internet, to alleviate the shortcomings of IPv4 and ultimately extend its lifetime. The Internet community owes the extended usability of IPv4 to these techniques; below are some of these strategies.

2.8.1 Network Address Rationing

Rationing is the first step when a shortage in a particular resource is predicted; this also applies to IP addresses. Once it became clear that IPv4 addresses would soon be in short supply, the IETF recommended that the regional Internet registries impose more stringent controls over the assignment of large blocks of IP addresses. This was implemented as early as 1992 through RFC 1366 [72].

2.8.2 IP Subnetting

Subnetting specifies additional bits, apart from the official bits available for network addresses by taking some bits from the node address. Subnetting was never mentioned in the original standard of IP, RFC 791 [3]. RFC 917 [39] was published in 1984 after it was realized that the initial two level addressing was unable to cater for different types of network media. For instance, not all nodes within a network would fit on a single cable. Large organizations spread over more than one physical plant require more than one physical LAN, and popular networking technologies such as Ethernet have a limit on the number of nodes on the same network.

The solution was to provide a means of subdividing Internet addresses so that large networks could appear to have several smaller sub-networks or subnets. IP subnetting does not increase the size of IP address space available, but it does make it possible to use the address space within any network more efficiently. This can be achieved by allocating different subnets in one network to different organizations.

2.8.3 Classless Inter-Domain Routing (CIDR)

IP addresses were defined originally as classful, that is, grouped into classes A to E. Organizations were assigned a whole class network address depending on their size, even if they can not utilize all the available addresses in the network. This method was a waste of IP addresses. Classless Inter-Domain Routing also known as supernetting was introduced through RFC 1519, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", in addition to the official numbers of bits that identify the node address, supernetting takes some bits from the network address.

This allows grouping more than one class C networks and assign them to one organization and which can then be addressed as one network from outside. The main advantages of CIDR are: organizations are allocated no more than exactly the number of IP addresses they need and it dramatically collapses the size of the non-default routing tables [15].

Despite the use of CIDR the non-default routing tables has grown 30-fold since 1992, but the situation would have been worse without it.

2.8.4 Recycling Unused IP Networks

The Internet Assigned Numbers Authority (IANA) appealed for unused IP addresses to be returned in RFC 1917 "An Appeal to the Internet Community to Return Unused IP Networks (Prefixes) to the IANA". This appeal was made because large chunks of IP addresses have been allocated to organizations that no longer exist or that could never use all of their allocations. Several of these pieces have been recovered and re-assigned.

2.8.5 Network Address Translation (NAT)

Network Address Translation allows for the use of private networks which have minimal or no connectivity to the global Internet. Only nodes which need Internet connectivity are assigned unique IP addresses, such as, email servers. This helps to serve the address space. Due to the deployment of NAT, end-to-end network connectivity is broken, and networked devices cannot be located by legitimate applications and services. Many applications that utilize peer-to-peer connections cannot work well today. Examples include voice over Internet Protocol (VoIP), video and secure collaborations, all of which have varying degrees of difficulty with working well in a NAT'ed network. IPv6 removes these obstacles, and enables applications and services to be easily developed and deployed. IPv6 will make applications just work without awkward network configurations, management tasks, or server deployments.

2.9 Permanent solutions

Internet Protocol version six (IPv6) is regarded by many as the permanent solution to the shortcomings of IPv4. There is no doubt that IPv4 contributed a lot to the Internet but updating it has become increasingly more problematic. The predicted depletion of address space and the increasing size of the non-default routing tables are the main alarming reasons for the introduction of a new protocol.

The rest of this chapter will introduce certain aspects of IPv6, with special focus on security mechanisms. IP security is provided by a set of two protocols, Authentication Header (AH) and Encapsulating Security Payload (ESP).

2.10 Internet Protocol version 6

Internet Protocol version 6 (IPv6) is a new version of the Internet Protocol, designed as the successor to IP version 4 (IPv4). The changes from IPv4 to IPv6 fall primarily into the following five categories:

- Expanded Addressing Capabilities

  The most obvious distinguishing feature of IPv6 is its use of much larger addresses. The size of an address in IPv6 is 128 bits, which is four times larger than an IPv4 address. A 32-bit address space allows for 2^{32} or 4,294,967,296 possible addresses. A 128-bit address space allows for 2^{128} or 3.4×10^{38} possible addresses.

  In the late 1970s when the IPv4 address space was designed, it was unimaginable that it could be exhausted. However, due to changes in technology and an allocation practice that did not anticipate the recent explosion of hosts on the Internet, the IPv4 address space was consumed to the point that by 1992 it was clear a replacement would be necessary.

  With IPv6, it is even harder to conceive that the IPv6 address space will be consumed. To help put this number in perspective, a 128-bit address space provides 6.5×10^{23} addresses for every square meter of the Earth's surface.

The relatively large size of the IPv6 address is designed to be subdivided into hierarchical routing domains that reflect the topology of the modern-day Internet. The use of 128 bits allows for multiple levels of hierarchy and flexibility in designing hierarchical addressing and routing that is currently lacking on the IPv4-based Internet. [52]

- Header Format Simplification

  Some of the IPv4 header fields have been dropped such as header checksum and fragment related fields or have been made optional, to reduce the common-case processing cost of packet handling and to limit the bandwidth cost of the IPv6 header.

- Improved Support for Extensions and Options

  Changes in the way IP header options are encoded allow for more efficient forwarding, less stringent limits on the length of options, and greater flexibility for introducing new options in the future.

- Flow Labeling Capability

  A new capability is added to enable the labeling of packets belonging to particular traffic "flows" for which the sender requests special handling, such as non-default quality of service or "real-time" service.

- Authentication and Privacy Capabilities

  Extensions to support authentication, data integrity, and (optional) data confidentiality are specified for IPv6. [19]

- Auto-configuration Capability

  One important goal for IPv6 is to support node Plug and Play. That is, it should be possible to plug a node into an IPv6 network and have it automatically configured without any human intervention. IPv6 supports stateful auto-configuration and stateless auto-configuration.

1. **Stateful** – This type of configuration requires the use a Dynamic Host Configuration Protocol for IPv6 (DHCPv6) server for the installation and administration of the nodes. The DHCPv6 server keeps a list of nodes to which it supplies configuration information. It also maintains state information so the server knows how long each address is in use, and when it might be available for reassignment.

2. **Stateless** – This type of configuration is suitable for small organizations and individuals, where each host determines its addresses from the contents of received router advertisements. Using the IEEE EUI-64 standard to define the network ID portion of the address, it is reasonable to assume the uniqueness of the host address on the link.

IEEE EUI-64 is a standard defined by IEEE for 64-bit extended unique identifier (EUI-64), which is a concatenation of the 24-bit company ID value on the network card by the IEEE Registration Authority and a 40-bit extension identifier assigned by the organization with that company ID assignment.

Regardless of how the address is determined, the node must verify that its potential address is unique to the local link. This is done by sending a neighbor solicitation message to the potential address. If the node receives any response, it knows that the address is already in use and must determine another address [14].

2.10.1 IPv6 Header Format

In comparison to IPv4 header that consists of 12 compulsory fields, IPv6 has only 8 compulsory fields. This design feature was introduced to improve header processing along the transmission route. Figure 2-4 shows IPv6 header fields.

2.10.1.1 Version

Version is a 4-bit Internet Protocol version number, 6 for IPv6.

2.10.1.2 Traffic Class

The 8-bit Traffic Class field in the IPv6 header is available for use by originating nodes and/or forwarding routers to identify and distinguish between different classes or priorities of IPv6 packets. This is similar to the Type of Service field in IPv4 packet header. Traffic class is used in DiffServ purposes.

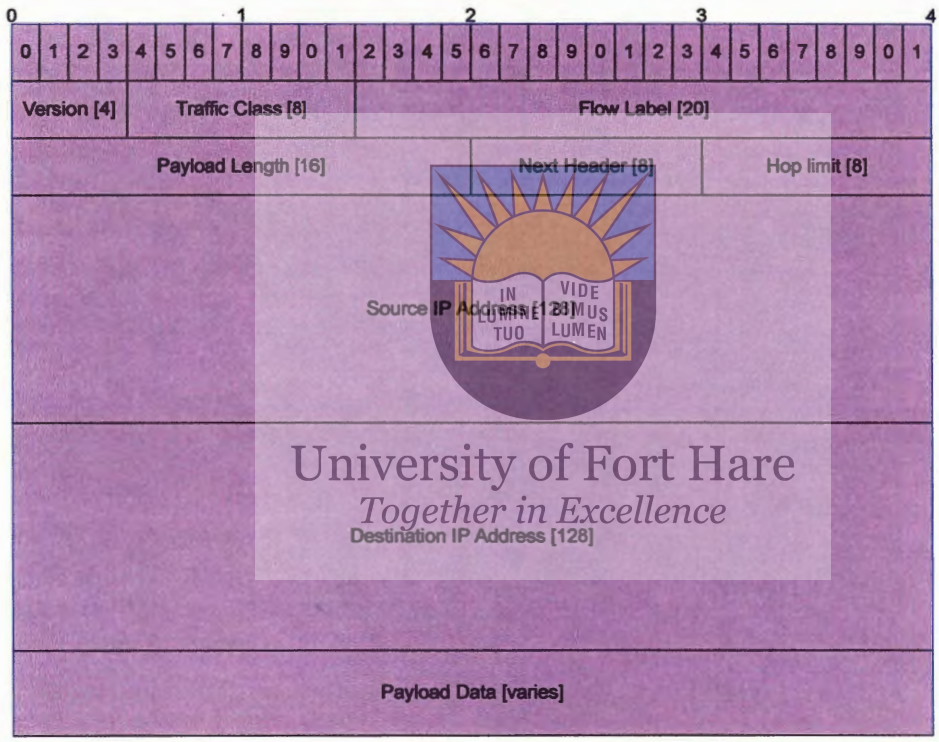


Figure 2-4: IPv6 header format

2.10.1.3 Flow Label

The 20-bit Flow Label field in the IPv6 header may be used by a source to label sequences of packets as a flow, for which it requests special handling by the IPv6 routers, such as non-default quality of service or "real-time" service. Hosts or routers that do not support the functions of the Flow Label field are required to set the field to zero when originating a packet, pass the field on unchanged when forwarding a packet, and ignore the field when receiving a packet.

2.10.1.4 Payload Length

This 16-bit unsigned integer is the length of the IPv6 payload, i.e., the rest of the packet following this IPv6 header, in octets. Any extension headers present are considered part of the payload, i.e., included in the length count.

2.10.1.5 Next Header

Next Header is an 8-bit selector that identifies the type of header immediately following the basic 40 bytesIPv6 header. It uses the same values as and is similar to the IPv4 Protocol field.

2.10.1.6 Hop Limit

This is an 8-bit unsigned integer that is decremented by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero. This is similar to the TTL field of the IPv4 packer header.

2.10.1.7 Source Address

Source Address is a 128-bit address of the originator of the packet.

2.10.1.8 Destination Address

Destination Address is a 128-bit address of the intended recipient of the packet, possibly not the ultimate recipient, if a Routing header is present.

2.10.2 IPv6 Extension Headers

In IPv6, optional Internet-layer information is encoded in separate headers that may be placed between the basic IPv6 header and the upper layer header in a packet. There are a maximum of six such extension headers, each identified by a distinct Next Header value. An IPv6 packet may carry zero, one, or more extension headers, each identified by the Next Header field of the preceding header.

Extension headers are not examined or processed by any node along a packet's delivery path, until the packet reaches the node (or each of the set of nodes, in the

case of multicast) identified in the Destination Address field of the IPv6 header. Therefore, extension headers must be processed strictly in the order in which they appear in the packet; a receiver must not, for example, scan through a packet looking for a particular kind of extension header and process that header prior to processing all preceding ones.

The exception referred to in the preceding paragraph is the Hop-by-Hop Options header, which carries information that must be examined and processed by every node along a packet's delivery path, including the source and destination nodes. The Hop-by-Hop Options header, when present, must immediately follow the IPv6 header. Its presence is indicated by the value zero in the Next Header field of the IPv6 header.

A full implementation of IPv6 includes implementation of the following extension headers and their corresponding next header values are shown in brackets:

- Hop-by-Hop Options (0)
- Routing (43)
- Fragment (44)
- Destination Options (60)
- Authentication (51)
- Encapsulating Security Payload (50)

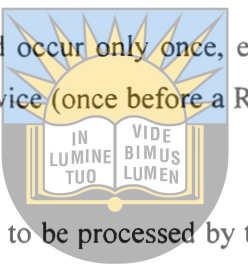
2.10.3 Extension Header Order

When more than one extension header is used in the same packet, it is recommended that all headers appear in the following order:

- IPv6 header
- Hop-by-Hop Options header

- Destination Options header*
- Routing header
- Fragment header
- Authentication header
- Encapsulating Security Payload header
- Destination Options header*
- upper-layer header

Each extension header should occur only once, except for the Destination Options header, which should occur twice (once before a Routing header and once before the upper-layer header).



* The first header has options to be processed by the first destination that appears in the IPv6 Destination Address field plus subsequent destinations listed in the Routing header. The second header has options to be processed only by the final destination of the packet [4].

2.10.4 Maximum Packet Lifetime

Unlike IPv4, IPv6 nodes are not required to enforce maximum packet lifetime. That is the reason the IPv4 "Time to Live" field was renamed "Hop Limit" in IPv6. In practice, very few, if any, IPv4 implementations conform to the requirement that they limit packet lifetime, so this is not a change in practice. Any upper-layer protocol that relies on the internet layer (whether IPv4 or IPv6) to limit packet lifetime ought to be upgraded to provide its own mechanisms for detecting and discarding obsolete packets.

2.10.5 Maximum Upper-Layer Payload Size

When computing the maximum payload size available for upper-layer data, an upper-layer protocol must take into account the larger size of the IPv6 header relative

to the IPv4 header. For example, in IPv4, the TCP's maximum segment size (MSS) option is computed as the maximum packet size (a default value or a value learned through Path MTU Discovery) minus 40 octets (20 octets for the minimum-length IPv4 header and 20 octets for the minimum-length TCP header). When using TCP over IPv6, the MSS must be computed as the maximum packet size minus 60 octets, because the minimum-length IPv6 header (i.e., an IPv6 header with no extension headers) is 20 octets longer than a minimum-length IPv4 header. [19]

2.11 IPv6 Transition Mechanisms

Transition from one protocol to another is supposed to be a well planned process which might stretch for decades depending on the complexity of the protocols involved. Pv6 is no exception, and the authorities involved have been planning and designing ways of smoothly introducing IPv6 without a doomsday for more than a decade now. Therefore, the transition period will be considerably long, in which case IPv4 and IPv6 will interoperate on our networks.

Network designers recommend deploying IPv6 at the edge first and then moving towards the network core to reduce the cost and operational impacts of the integration. The key strategies used in deploying IPv6 at the edge of a network involve carrying IPv6 traffic over the IPv4 network, allowing isolated IPv6 domains to communicate with each other before the full transition to a native IPv6 backbone. It is also possible to run IPv4 and IPv6 throughout the network, from all edges through the core, or to translate between IPv4 and IPv6 to allow hosts communicating in one protocol to communicate transparently with hosts running the other protocol. All techniques allow networks to be upgraded and IPv6 deployed incrementally with little or no disruption of IPv4 services. A wide range of techniques have been identified and implemented, basically falling into three categories:

- **Dual stack**, to allow IPv4 and IPv6 to co-exist in the same devices and networks
- **Tunneling**, to avoid order dependencies when upgrading hosts, routers, or regions

- **Translation**, to allow IPv6-only devices to communicate with IPv4-only devices

Usually these techniques are all used in combination [23][74].

2.11.1 Dual Stack Techniques

A dual stack technique is an implementation of the TCP/IP suite of protocols that includes both an IPv4 Internet layer and an IPv6 Internet layer. This is the mechanism used by IPv6/IPv4 nodes so that communication with both IPv4 and IPv6 nodes can occur. Dual stack implementation also has a separate implementation of Host-to-Host layer protocols, TCP and UDP protocols for IPv4 and IPv6. Therefore dual stack should not be confused with dual IP layer which contains a single implementation of Host-to-Host layer protocols such as TCP and UDP. All upper layer protocols in a dual IP layer implementation can communicate over IPv4, IPv6, or IPv6 tunneled in IPv4, Figure 2-5 shows these two models.

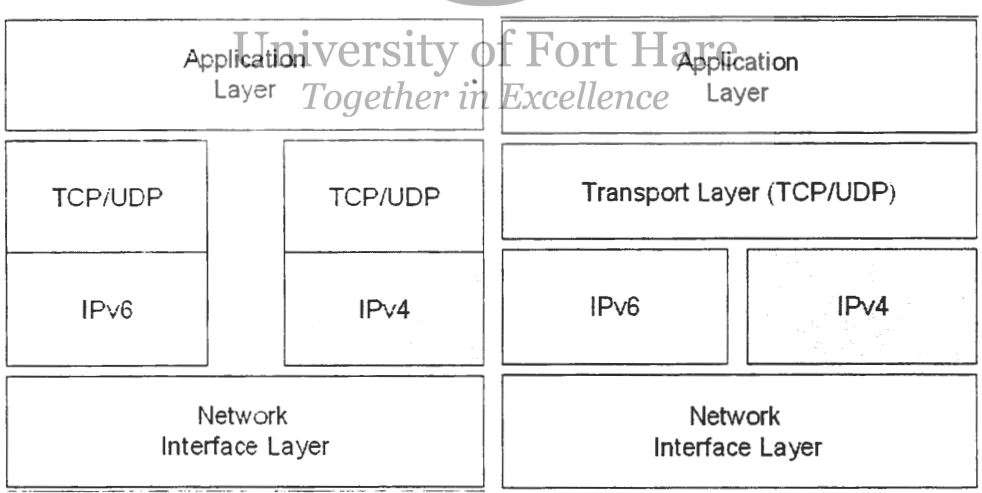


Figure 2-5: Dual stack and Dual IP layer models

Although these architectures are different they function in terms of providing functionality for IPv6 transition.

2.11.2 Tunneling Techniques

This technique involves tunnels that encapsulate the IPv6 traffic within the IPv4 packets, and are primarily for communication between isolated IPv6 sites or connection to remote IPv6 networks over an IPv4 backbone. All tunneling mechanisms require that the endpoints of the tunnel run both IPv4 and IPv6 protocol stacks, that is. endpoints must run in dual-stack mode.

The techniques include using manually configured tunnels, semi-automatic tunnel mechanisms such as tunnel broker services, and fully automatic tunnel mechanisms such as 6to4 for the WAN and intra-site automatic tunnel addressing protocol (ISATAP) for the campus environment. The following are the main tunneling techniques in use [20]:

2.11.2.1 Manually configured

The primary use of a configured tunnel is to provide stable and secure connections for regular communication between two edge routers, or between an end system and an edge router, or for connection to remote IPv6 networks. The edge routers and end systems used as tunnel endpoints must be dual-stack devices. Manual tunnels are used between two points and require configuration of both the source and destination addresses of the tunnel, whereas automatic tunnel mechanisms need to be only enabled and are more transient. Because each tunnel is independently managed, the more tunnel endpoints one has, the more tunnels one needs, and the greater the management overhead. As with other tunnel mechanisms, network address translation (NAT) is not allowed along the path of the tunnel [21].

2.11.2.2 Semi-automated

In this category there is only one technique, that is, a tunnel broker. A tunnel broker is a web server that receives requests from clients, and then generates the tunnel and sends back tunnel information to the client. The tunnel broker model introduces a server which acts as a relay between the customer and the tunnel server(s) at the ISP. First, the client needs to authenticate at the tunnel broker. If this is successful it

establishes the tunnel between the client and the ISPs tunnel endpoint(s). This procedure can also be used for billing purposes [22].

2.11.2.3 Automatic

The simplest way to introduce compatibility between different versions of IP is to have both versions running in the same machine. This enables the machine to handle both types of traffic. Unlike the manually configured tunnels discussed above there are techniques that automatically configure tunnels.

The IPv6 transition mechanisms include a technique for hosts and routers to dynamically tunnel IPv6 packets over IPv4 routing infrastructure. IPv6 nodes that utilize this technique are assigned special IPv6 unicast addresses that carry an IPv4 address in the low order 32-bits.

2.11.2.3.1 6to4

6to4 tunneling is a technique where the tunnel endpoint is determined by the globally unique IPv4 address embedded in a 6to4 address and it allows IPv6 packets to be transmitted over an IPv4 network. 6to4 is defined in RFC 3056. 6to4 tunnels are used for less-permanent, transient connectivity and are useful when two hosts wish to exchange IPv6 traffic but some portion of the network between those hosts only supports IPv4. When used by an individual host, that host must have IPv4 connectivity and a global IPv4 address, and the host is responsible for encapsulation of outgoing IPv6 packets and decapsulation of incoming 6to4 packets. 6to4 does not facilitate interoperation between IPv4-only hosts and IPv6-only hosts.

A 6to4 IPv6 address is a combination of the unique routing prefix 2002::/16 and a globally unique 32-bit IPv4 address, it follows the format: 2002:{IPv4}::{IPv4} on Windows and 2002:{IPv4}::1 on Linux, where {IPv4} is a hexadecimal representation of the 32-bit IPv4 address. For example, 2002:c060:bf0::c060:bf0 on Windows with the IPv4 32-bit address being 192.96.11.240. 6to4 tunnels are configured between border routers, or between a border router and a host. [25-26]

2.11.2.3.2 6over4 (RFC 2529): Deprecated

Another mechanism that does not require registration of IPv6-information is the 6over4 mechanism, which implements transporting of IPv6 over a multicast-enabled IPv4 network, instead of Ethernet or FDDI. Its main drawback is the need for an existing multicast infrastructure. If this is not available, setting it up involves as much effort as setting up a configured IPv6 tunnel directly.

The goal of 6over4 is to use an existing IPv4 domain with multicast support to create a virtual link-layer for IPv6 hosts. This method can be used to enable IPv6 traffic between isolated hosts, within an IPv4 cloud. It can also be implemented on a router to enable external IPv6 connectivity for these hosts. 6over4 uses special IPv6 link-layer addresses which are bound to a virtual interface on the node. Packets sent to this interface are automatically encapsulated in IPv4. The address can for example have the link local scope prefix "FE80::/64". The last 32 bits are then copied from an IPv4 address of the node, e.g. "FE80::D5E1:3CB5/64". The limitations of 6over4 are similar to native IPv6, except if used in conjunction with link-layer protocols like ATM, which have complex multicast implementations. In such cases the performance might be poorer than with native IPv6 [27].

2.11.2.3.3 ISATAP

Like some other tunneling mechanisms, ISATAP uses a special address format to enable IPv6 traffic on links that are not connected to an IPv6 capable router. The protocol embeds IPv4 addresses as follows:

< 64-bit global or local prefix >"0000:5EFE"< 32 bit IPv4 address >

Every IPv4 address assigned to a link is seen as a link-layer address for ISATAP. Unlike for example 6over4, multicast support is not required on links where this transition protocol is used. It is also possible to use IPv4 addresses from the private address space defined in RFC 1918. Nevertheless, mechanism for anycast and multicast emulation on these links is subject of future work. ISATAP automatic tunnel interfaces may be configured over multiple underlying links with diverse maximum transmission units (MTUs) [28].

2.11.2.3.4 Teredo

Microsoft is also involved in the IPv6 development process and supports IPv6 in the newer versions of its operating systems. Christian Huitema from the Microsoft Corporation published a draft [45] which describes a method to establish IPv6 connectivity for clients that are located behind NAT. Some tunneling mechanism like 6to4 can only be used in combination with NAT if both services are combined in one box. This problem is faced by the Teredo service. It implements a very complex solution that introduces Teredo clients, servers and relays. Therefore it should be used only as a last resort if there are no other possibilities of connecting the IPv6 clients to the IPv6 Internet. The solution is based on the encapsulation of IPv6 packets in UDP, as this protocol is accepted by NAT in most cases. The Teredo service uses a special 32-bit prefix, which will have to be assigned by IANA, and the Teredo service UDP port, which is 3544. Because of the complexity of the service and the fact that additional entities (the Teredo server and relay) are necessary, the security considerations are manifold. An active Teredo implementation might open vulnerabilities regarding UDP port 3544 and also increase the risk for DOS or DDOS attacks [29].

The following types of NAT are defined:

2.11.2.3.4.1 Cone NAT

A cone NAT is a NAT in which the NAT translation table entry stores a mapping between an internal address and port number and an external address and port number. Once the NAT translation table entry is in place, inbound traffic to the external address and port number from any source address and port number is allowed and translated.

2.11.2.3.4.2 Restricted NAT

A NAT in which the NAT translation table entry stores a mapping between an internal address and port number and an external address and port number, for either specific source addresses or specific source address and port numbers. An inbound packet that matches the NAT translation table entry for the external destination

address and port number from an unknown external address or port number is silently discarded.

2.11.2.3.4.3 Symmetric NAT

A NAT that maps the same internal address and port number to different external addresses and ports, depending on the external destination address (for outbound traffic).

Teredo can work over only cone and restricted but not over symmetric NAT [51].

2.11.3 Translation Techniques

Translation is primarily intended for later stages of co-existence, to allow continued use of legacy devices (such as printers) when newer devices weaned of any IPv4 become available. This approach should only be used to enable IPv6-only devices to talk to IPv4-only devices, with the goal of avoiding the substantial operational costs of application incompatibilities, single points of failure, and synchronization of infrastructure components with application deployments.

2.11.3.1 Network Address Translation-Protocol Translation

Network Address Translation-Protocol Translation (NAT-PT) is an IETF RFC 2766 [10] specification that will help the Internet make the transition from an IPv4 environment to an IPv6 environment. During the transition, IPv6-only and IPv4-only hosts will coexist and will need to communicate. NAT-PT is the IPv4-to-IPv6 protocol translator needed during this coexistence phase.

A NAT-PT router sits on the edge of an IPv6 network. The router has a pool of IPv4 addresses, which are used to dynamically assign IPv4 addresses to IPv6 nodes during transactions in an IPv6-to-IPv4 session. During the Internet's transition phase, the router performs both network address translation (NAT) as well as protocol translation (PT). One of the disadvantages to NAT-PT is that some applications carry the IP address (the network address) as part of the packet payload. Because the NAT-PT router does not 'snoop' the payload, the router may not be aware of the embedded

IP address. In this case, the router may need the services of an application layer gateway (ALG). The ALG will look at the packet payload and translate the IP addresses embedded inside the packet. An ALG is needed for applications such as domain name system (DNS) and file transfer protocol (FTP) [73].

When a host node transmits a large volume of data to another host node, the data is sent contiguously in the form of IP packets. For these IP packets, data should not be fragmented while it is sent from the source node to the destination node. However, the difference in IP header length of both protocols may exceed the Maximum Transmission Unit (MTU) of the translator due to the link on the boundary of an IPv4 network and IPv6 network. Therefore, although a relay node does not usually fragment IP packets in IPv6, the translator performs fragmentation instead of the source node.

2.12 IPv6 on Windows Platforms

Microsoft is one of the leading organizations in introducing IPv6 to the Internet community. Like most organizations, they are continuing with extensive research in providing IPv6 capabilities in their products. Microsoft's most refined version of IPv6 protocol support is provided in Microsoft Windows Server 2003 family and Windows XP with Service Pack 1 and Service Pack 2 and the Advanced Networking Pack for Windows XP. This section will explain how this protocol is implemented on the above mentioned platforms.

The IPv6 protocol for the Windows Server 2003 family is not a dual IP layer. The IPv6 protocol driver, Tcpip6.sys, contains a separate implementation of TCP and UDP and is sometimes referred to as a dual-stack implementation.

2.12.1 Compatibility Addresses

The following addresses are defined to aid in the coexistence of IPv4 and IPv6 nodes:

2.12.2 IPv4-compatible addresses

The IPv4-compatible address, 0:0:0:0:0:0: *w.x.y.z* or ::*w.x.y.z* (where *w.x.y.z* is the dotted decimal representation of a public IPv4 address), is used by IPv6/IPv4 nodes that are communicating with IPv6 over an IPv4 infrastructure. When the IPv4-compatible address is used as an IPv6 destination, the IPv6 traffic is automatically encapsulated with an IPv4 header and sent to the destination using the IPv4 infrastructure.

2.12.3 IPv4-mapped addresses

The IPv4-mapped address, 0:0:0:0:0:FFFF:*w.x.y.z* or ::FFFF:*w.x.y.z*, is used to represent an IPv4-only node to an IPv6 node. It is used only for internal representation. The IPv4-mapped address is never used as a source or destination address of an IPv6 packet. The IPv6 protocol for the Windows Server 2003 family does not support the use of IPv4-mapped addresses. The IPv4-mapped address is used by some IPv6 implementations when acting as a translator between IPv4-only and IPv6-only nodes.

2.12.4 6over4 addresses

6over4 addresses comprise of a valid 64-bit unicast address prefix and the interface identifier ::*WWXX*:*YYZZ* (where *WWXX*:*YYZZ* is the colon-hexadecimal representation of *w.x.y.z*, a unicast IPv4 address assigned to an interface). An example of a link-local 6over4 address based on the IPv4 address of 192.96.11.10 is FE80::836B:45C. 6over4 addresses are used to represent a host when using the automatic tunneling mechanism defined in RFC 2529.

2.12.5 6to4 addresses

6to4 addresses are based on the prefix 2002:*WWXX*:*YYZZ*::/48 (where *WWXX*:*YYZZ* is the colon hexadecimal representation of *w.x.y.z*, a public IPv4 address assigned to an interface). 6to4 addresses are used to represent a site when using the automatic tunneling mechanism defined in [26].

2.12.6 ISATAP addresses

Intra-site Automatic Tunnel Addressing Protocol (ISATAP) addresses are composed of a valid 64-bit unicast address prefix and the interface identifier ::0:5EFE:*w.x.y.z* (where *w.x.y.z* is a unicast IPv4 address assigned to an interface). An example of a link-local ISATAP address is FE80::5EFE:192.96.11.10. ISATAP addresses are used to represent a host when using the automatic tunneling mechanism defined in the Internet draft titled ISATAP [28].

2.12.7 Teredo addresses

Teredo addresses use the prefix 3FFE:831F::/32. An example of a Teredo address is 3FFE:831F:CE49:7601:8000:EFFF:62C3:FFFE. Beyond the first 32 bits, Teredo addresses are used to encode the IPv4 address of a Teredo server, flags, and the encoded version of a Teredo client's external address and port. Teredo addresses are used to represent a host when using the automatic tunneling mechanism defined in the Internet draft [30].

2.12.8 IPv6 over IPv4 Tunneling

IPv6 over IPv4 tunneling is the encapsulation of IPv6 packets with an IPv4 header so that IPv6 packets can be sent over an IPv4 infrastructure. Within the IPv4 header:

- The IPv4 Protocol field is set to 41 to indicate an encapsulated IPv6 packet.
- The Source and Destination fields are set to IPv4 addresses of the tunnel endpoints.

The tunnel endpoints are either manually configured as part of the tunnel interface or are automatically derived from the sending interface, the next-hop address of the matching route, or the source and destination IPv6 addresses in the IPv6 header.

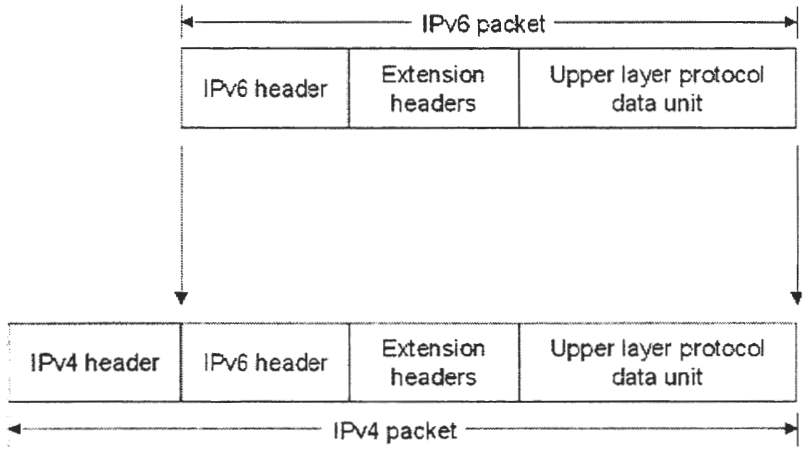


Figure 2-6: IPv6 packet tunneled through IPv4 infrastructure

For IPv6 over IPv4 tunneling shown in Figure 2-6, the IPv6 path maximum transmission unit (MTU) for the destination is typically 20 less than the IPv4 path MTU for the destination. However, if the IPv4 path MTU is not stored for each tunnel, there are instances where the IPv4 packet will need to be fragmented at an intermediate IPv4 router. In this case, IPv6 over IPv4 tunneled packet must be sent with the Do not Fragment flag in the IPv4 header set to 0.

2.13 Windows Tunneling Configurations

There are three tunneling configurations with which to tunnel IPv6 traffic between IPv6/IPv4 nodes over an IPv4 infrastructure as defined by RFC 2893 [31]:

- Router-to-Router
- Host-to-Router or Router-to-Host
- Host-to-Host

IPv6 over IPv4 tunneling only describes an encapsulation of IPv6 packets with an IPv4 header so that IPv6 nodes are reachable across an IPv4 infrastructure. Unlike tunneling for the Point-to-Point Tunneling Protocol (PPTP) and Layer Two Tunneling Protocol (L2TP), there is no exchange of messages for tunnel setup, maintenance, or termination.

2.13.1 Router-to-Router

In the router-to-router tunneling configuration, two IPv6/IPv4 routers connect two IPv4 or IPv6 infrastructures over an IPv4 infrastructure. The tunnel endpoints span a logical link in the path between the source and destination. The IPv6 over IPv4 tunnel between the two routers acts as a single hop. Routes within each IPv4 or IPv6 infrastructure point to the IPv6/IPv4 router on the edge. For each IPv6/IPv4 router, there is a tunnel interface representing the IPv6 over IPv4 tunnel and routes that use the tunnel interface.

Examples of this tunneling configuration are:

- An IPv6-only test lab that tunnels across an organization's IPv4 infrastructure to reach the IPv6 Internet.
- Two IPv6-only routing domains that tunnel across the IPv4 Internet.
- A 6to4 router that tunnels across the IPv4 Internet to reach another 6to4 router or a 6to4 relay router.

2.13.2 Host-to-Router and Router-to-Host

In the host-to-router tunneling configuration, an IPv6/IPv4 node that resides within an IPv4 infrastructure creates an IPv6 over IPv4 tunnel to reach an IPv6/IPv4 router. The tunnel endpoints span the first segment of the path between the source and destination nodes. The IPv6 over IPv4 tunnel between the IPv6/IPv4 node and the IPv6/IPv4 router acts as a single hop. On the IPv6/IPv4 node, a tunnel interface representing the IPv6 over IPv4 tunnel is created and a route is added using the tunnel interface. The IPv6/IPv4 node tunnels the IPv6 packet based on the matching route, the tunnel interface, and the next-hop address of the IPv6/IPv4 router.

In the router-to-host tunneling configuration, an IPv6/IPv4 router creates an IPv6 over IPv4 tunnel across an IPv4 infrastructure to reach an IPv6/IPv4 node. The tunnel endpoints span the last segment of the path between the source node and destination node. The IPv6 over IPv4 tunnel between the IPv6/IPv4 router and the IPv6/IPv4 node acts as a single hop. On the IPv6/IPv4 router, a tunnel interface representing the IPv6 over IPv4 tunnel is created and a route (typically a subnet

route) is added using the tunnel interface. The IPv6/IPv4 router tunnels the IPv6 packet based on the matching subnet route, the tunnel interface, and the destination address of the IPv6/IPv4 node.

Examples of host-to-router and router-to-host tunneling are:

- An IPv6/IPv4 host tunnels traffic across an organization's IPv4 infrastructure to reach the IPv6 Internet.
- An ISATAP host tunnels traffic across an IPv4 network to an ISATAP router to reach the IPv4 Internet, another IPv4 network, or an IPv6 network.
- An ISATAP router tunnels traffic across an IPv4 network to reach an ISATAP host.

2.13.3 Host-to-Host

In the host-to-host tunneling configuration, an IPv6/IPv4 node that resides within an IPv4 infrastructure creates an IPv6 over IPv4 tunnel to reach another IPv6/IPv4 node that resides within the same IPv4 infrastructure. The tunnel endpoints span the entire path between the source and destination nodes. The IPv6 over IPv4 tunnel between the IPv6/IPv4 nodes acts as a single hop.

On each IPv6/IPv4 node, an interface representing the IPv6 over IPv4 tunnel is created. Routes might be present to indicate that the destination node is on the same logical subnet defined by the IPv4 infrastructure. Based on the sending interface, the optional route, and the destination address, the sending host tunnels the IPv6 traffic to the destination.

Examples of this tunneling configuration are:

- IPv6/IPv4 hosts that use ISATAP addresses to tunnel across an organization's IPv4 infrastructure
- IPv6/IPv4 hosts that use IPv4-compatible addresses to tunnel across an organization's IPv4 infrastructure.

2.13.4 Configured Tunnels

A configured tunnel requires manual configuration of tunnel endpoints. In a configured tunnel, the IPv4 addresses of the tunnel endpoints are not derived from addresses that are encoded in the IPv6 source or destination addresses or the next-hop address of the matching route. Typically, router-to-router tunneling configurations are manually configured. The tunnel interface configuration, consisting of the IPv4 addresses of the tunnel endpoints, must be manually specified along with the static routes that use the tunnel interface.

2.13.5 Automatic Tunnels

An automatic tunnel is a tunnel that does not require manual configuration. Tunnel endpoints are determined by the use of logical tunnel interfaces, routes, and source and destination IPv6 addresses.

2.14 Drawbacks of IPv6

Migrating to IPv6 does have drawbacks but most of these drawbacks are on the business side. Implementing IPv6 will require careful planning, a thorough review of the network's architecture and a detailed migration plan. Training will be needed for architectural and operational personnel, and interoperability with existing IPv4 implementations will need to be maintained for quite a while. All these issues will ultimately cost companies a great deal of money and unless the need to migrate is really necessary organizations will be reluctant to consign funds on migration. This is one of the reasons why migration to IPv6 has been moving very slowly for the past decade. Some authors [79] has pointed out that the issues with the scalability of IPv4 routing do exist in IPv6 as the IPv6 routing/addressing architecture is pretty much the same as IPv4. But the auto-configuration feature of IPv6 is partly addressing the problem. IPv4 will be around for a long time and will still be the protocol of choice for many implementations, due to the well developed IPv4 infrastructure. However, as legacy equipment becomes fully depreciated and network upgrades can be financially justified, the potential benefits of IPv6 need to be considered to see if it has a role in the overall technology strategy.

CHAPTER THREE – INTERNET PROTOCOL SECURITY

This chapter introduces the security mechanisms developed to provide security services to the IP, Internet Protocol security (IPSec) to be more specific. A detailed discussion will be given of the protocols that provide security services in IPSec together with the algorithms implemented in those protocols.

3.1 Introduction

One of the weaknesses of the original Internet Protocol is that it lacks any sort of general purpose mechanism for ensuring the authenticity and privacy of data as it is passed over the internetwork. Since IP datagrams must usually be routed between two devices over unknown networks, any information in them is subject to being intercepted and even possibly changed. With the increased use of the Internet for critical applications, security enhancements were needed for the IP. To this end, a set of protocols called IP Security or IPSec was developed.

In this section a brief description of IPSec concepts and protocols is provided. It begins with an overview of IPSec, including a discussion of the history of the technology and defining standards. A description of the main components and protocols of the IPSec suite, and its different architectures and methods for implementation is given. Furthermore an actual description of how IPSec works, beginning with a description of the two IPSec modes (transport and tunnel) and how they differ. Then security associations and related constructs such as the Security Parameter Index (SPI) are described. The last three topics cover the three main IPSec protocols: IPSec Authentication Header (AH), IPSec Encapsulating Security Payload (ESP) and the IPSec Internet Key Exchange (IKE).

3.2 Internet Protocol Security

3.2.1 Introduction

IP Security (IPSec) is a framework of open standards for ensuring secure private communications over IP networks in which IP datagrams may be authenticated and

encrypted when they are exchanged between two nodes and is defined in RFC2401 [32], based on standards developed by the Internet Engineering Task Force (IETF). IPSec is a fairly large collection of technologies that encompasses network and security protocols, cryptographic algorithms, and recommendations. It is a set of extensions to the IP protocol family that provides cryptographic security services. These services allow for authentication, integrity, access control and confidentiality. IPSec provides similar services as SSL, but at the network layer, in a way that is completely transparent to applications, and much more powerful. This is so because the applications do not have to have any knowledge of IPSec to be able to use it. Any IP protocol can be used over IPSec [16].

IPSec is not a single protocol, but rather a set of services and protocols that provide a complete security solution for an IP network. These services and protocols combine to provide various types of protection. Since IPSec works at the IP layer, it can provide these protections for any higher layer TCP/IP application or protocol without the need for additional security methods, which is a major strength. Some of the kinds of protection services offered by IPSec include:

- Encryption of user data for privacy.
- Authentication of the integrity of a message to ensure that it is not changed en route.
- Protection against certain types of security attacks, such as replay attacks.
- The ability for devices to negotiate the security algorithms and keys required to meet their security needs.
- Two security modes, tunnel and transport, to meet different network needs.

IPSec was initially developed with IPv6 in mind, but has been engineered to provide security for both IPv4 and IPv6 networks, and its operation in both versions is similar. There are some differences in the datagram formats used for AH and ESP depending on whether IPSec is used in IPv4 and IPv6, since the two versions have different datagram formats and addressing.

When two devices, either end user hosts or intermediate devices such as routers or firewalls engage in secure communications, a secure path is setup between the devices that may traverse many insecure intermediate systems. To accomplish this, the following tasks need to be performed:

1. There must be an agreement on a set of security protocols to use, so that each device sends data in a format the other can understand.
2. There must be a decision on a specific encryption algorithm to use in encoding data.
3. There must be an exchange of keys that are used to "unlock" data that has been cryptographically encoded.
4. Once this background work is completed, each device must use the protocols, methods and keys previously agreed upon to encode data and send it across the network.

3.2.2 How IPSec Works

IPSec secured links are defined in terms of Security Associations (SAs). Each SA is defined for a single unidirectional flow of data, and usually from one single point to another, covering traffic distinguishable by some unique selector. All traffic flowing over a single SA is treated the same. Some traffic may be subject to several SAs, each of which applies some transform. Incoming packets can be assigned to a particular SA by the three defining fields, (Destination IP address, Security Parameter Index (SPI), security protocol). The SPI can be considered a cookie that is handed out by the receiver of the SA when the parameters of the connection are negotiated. The security protocol must be either AH or ESP. Since the IP address of the receiver is part of the triple, this is a guaranteed unique value. They can be found from the outer IP header and the first security protocol header (which contains the SPI and the security protocol) [16].

3.3 Overview of IPSec Architecture

The objectives of the IPSec security architecture are as follows:

- Data origin authentication and data integrity: It should not be possible to send an IP packet with a forged source address or to modify the destination address of an IP packet without detection by the receiver. It also should not be possible for the content of a packet to be modified without detection by the receiver.

- Replay protection: It should not be possible for a packet to be replayed at a later time without detection by the receiver. This service is provided by the sequence number fields found in both IPSec protocols. They provide anti-replay protection for the packet. The sequence number is a 32-bit, incrementally increasing number starting from 1 that indicates the packet number sent over the SA for the communication. The sequence number cannot repeat for the life of the quick mode security association. The receiver checks this field to verify that a packet for an SA with this number has not already been received. If one has been received, the packet is rejected.
- Confidentiality: It should not be possible to read the content of transmitted IP packets. Furthermore, IPSec should provide protection from traffic flow analyses through the use of encryption algorithms.

Another important objective of the IPSec architecture is to give sender, receiver and all gateway nodes the option of deciding which security requirements they want to apply to specific IP traffic flows according to the local security policy. The sending entities (host and gateway systems) secure IP packets according to a local security policy and receiving entities discard packets that have inadequate protection because they lack confidence in these packets. The main concepts of the architecture are defined in RFC 2401 [2]. These include the concept of Security Association (SA), the conceptual Security Associations Database (SADB), the Security Policy as well as the conceptual Security Policy Database (SPD) [40]. RFC 2401 also gives an overview of the two fundamental security protocols of the architecture, Authentication Header (AH) and Encapsulating Security Payload (ESP), both of which are specified in separate RFCs. Both protocols can be operated in one of two possible operation modes, transport mode or tunnel mode. Other RFCs specify details on the use of specific cryptographic operations with AH and ESP:

- Data Encryption Standard (DES) and other block ciphers in cipher block chaining mode are provided for encryption.
- Data origin authentication and integrity protection for IP packets is realized using a Hash Message Authentication Code (HMAC) construction, with Message Digest 5 (MD5), Secure Hash Algorithm 1 (SHA1), Advanced Encryption

Standard (AES) and RACE Integrity Primitives Evaluation Message Digest 160-bit message digest algorithm (RIPEMD-160) provided as cryptographic hash functions.

The two complementary protocols; Internet Security Association and Key Management Protocol (ISAKMP) and Internet Key Exchange (IKE) are used to negotiate the keys needed for the cryptographic operations above and perform entity authentication.

In this context a security association (SA) can be interpreted as a type of 'simplex connection' that provides specific security services to the traffic it carries. The term 'connection' is used with quotation marks because the Internet protocol realizes a connectionless and thus a stateless service for all participating entities. However, the respective peer entities require certain state knowledge to implement the security services, e.g. in respect of the security mechanisms and keys used. This state is administered through security associations established, deployed and dissolved between peer entities. As an SA is basically only used for one communication direction, two security associations are always needed for bidirectional communication.

The security services negotiated for an SA are supplied by exactly one of the two security protocols – Authentication Header (AH) or Encapsulating Security Payload (ESP). A triple, which consists of a Security Parameter Index (SPI), an IP destination address and the identifier of one of the two security protocols (AH or ESP), provides the unique SA identification for each system.

A security association can basically be established between the following entities:

- host to host
- host to gateway
- gateway to gateway

Two conceptual databases should exist in each system for the administration and specification of security associations. The Security Association Database (SADB) contains the SAs active in a system at any given time, and the Security Policy Database (SPD) defines which security services are being applied to which IP

packets and how this application should be executed. The SPD therefore identifies which security associations need to be established between which peer entities for which data streams and specifies the parameters to be negotiated for these associations.

These databases are thus called 'conceptual databases' since no 'real' database technology (e.g. relational database with general query language SQL) is required. The security architecture does not define how the databases are implemented nor are any general database concepts normally used in practice.

A security association is basically operated in one of the following two modes:

- Transport mode can only be used between the end points of a communication relationship, i.e., between two hosts — or, if a data stream is destined directly for a gateway system (e.g. in the case of network management applications) — between a host and a gateway system.
- Tunnel mode is most commonly used between gateways, or at an end-station to a gateway, the gateway acting as a proxy for the hosts behind it.

The difference between the two protocol modes is that the transport mode only adds a security-specific protocol header and possibly a trailer whereas the tunnel mode totally encapsulates the protected IP packets with a new IP header. This encapsulation of IP packets enables gateway systems to protect certain data streams on behalf of other entities. As a result, these systems can provide uniform protection to entire subnetworks, but not within the individual subnets.

The security protocol Authentication Header [33] implements the security service of data origin authentication with re-play protection for IP packets. As the name indicates, the protection is provided through a protocol header inserted between the IP protocol header and the user data of the IP packet. In contrast, the security protocol Encapsulating Security Payload [34] offers optional data origin authentication with packet replay protection as well as optional confidentiality for a packet with an AH transmitted user data. Selection of at least one of the two optional security services is necessary for an SA. The protocol implementation involves an

additional protocol header and a trailer with the user data of the IP packet encapsulated between them.

The two protocols, Internet Security Association Establishing security Key Management associations Protocol (ISAKMP) and Internet Key Exchange (IKE) serve as the basis for the negotiation and establishment of security associations.

The ISAKMP protocol [35] defines a generic framework for entity authentication, key exchange and the negotiation of parameters for security associations. It does not identify a specific authentication protocol, but instead stipulates the 'language' for defining such protocols by specifying fundamental things as packet formats, retransmission timers, message construction requirements, etc. In principle, ISAKMP is defined as a general protocol for the authentication and negotiation of security parameters independently of IPSec. Use of ISAKMP for IPSec is explained in detail in RFC 2407 [43] separately from the protocol definition.

The Internet Key Exchange (IKE) protocol specified in RFC 2409 [41] defines a specific authentication and key exchange protocol that conforms to ISAKMP and may be used for various purposes, although in practice its main usage is the negotiation of security associations for IPSec. This negotiation takes place in two phases: During the first phase an IKE SA is established that defines how other security associations for protecting specific data streams should be negotiated between the two peer entities during the second phase.

Both security protocols AH and ESP carry a sequence number that provides replay protection for IP packets. This sequence number is initialized with the value 0 when an SA is established and incremented by 1 with each packet sent. A new key is needed to replace the session key agreed for the SA before a wraparound of the 32-bit long sequence number occurs. Because this sequence number is included in the computation of the MAC, any modification of it by an attacker will be detected.

The receiver of an IPSec-protected packet always verifies that the sequence number contained in the packet is within a range of acceptable numbers. This range is called a 'sliding window'. The reason an entire window is used is that in the Internet the arrival order of IP packets can change during transmission of IP packets via different routes, even during normal operation. Therefore, later packets may possibly arrive at

the receiver sooner than packets that were sent earlier. If receivers were to insist upon a strict sequence of the numbers, they would have to discard IP packets that unintentionally end up in the wrong sequence, which would result in an unnecessary reduction of the data throughput in the Internet. Consequently, a receiver only accepts IP packets if they are not 'too old', i.e., if newer IP packets with significantly higher sequence numbers have not already been accepted [36][38].

3.4 IPSec Protocols

IPSec is comprised of two core protocols and a number of other protocols and services that support these core protocols. These protocols are the ones that actually do the work of encoding information to ensure security. The core protocols are Authentication Header (AH) and Encapsulating Security Payload (ESP). AH and ESP need the support of the following other protocols and services for them to work:

- **Encryption/Hashing Algorithms:** AH and ESP are generic and do not specify the exact mechanism used for encryption. This gives them the flexibility to work with a variety of such algorithms, and to negotiate which is used as needed. Two common ones used with IPSec are Message Digest 5 (MD5) and Secure Hash Algorithm 1 (SHA-1).

- **Security Policies and Associations, and Management Methods:** Since IPSec provides flexibility in letting different devices decide how they want to implement security, some means is required to keep track of the security relationships between devices. This is done in IPSec using constructs called security policies and security associations, and by providing ways to exchange security association information.
- **Key Exchange Framework and Mechanism:** For two devices to exchange encrypted information they need to be able to share keys for unlocking the encryption. They also need a way to exchange security association information. In IPSec, a protocol called the Internet Key Exchange (IKE) provides these capabilities.

The above mentioned functions are explained in detail in the following sections.

3.4.1 Authentication Header

AH provides authentication, integrity, and replay protection but not confidentiality. This protocol provides authentication services for IPSec. It allows the recipient of a message to verify that the supposed originator of a message was in fact the one that sent it. It also allows the recipient to verify that none of the data in the datagram has been changed by any intermediate devices en route. It also provides protection against replay attacks, where a message is captured by an unauthorized user and re-sent. AH comes after the basic IP header and it is identified as protocol ID 51. It contains cryptographic hashes of the data and identification information. There are several different RFCs giving a choice of actual algorithms to use in the AH, however they all must follow the guidelines specified in RFC 2402 [33]. The primary difference between the authentications provided by ESP and AH is the extent of the coverage. Specifically, ESP does not protect any IP header fields unless those fields are encapsulated by ESP tunnel mode. Figure 3-1 shows the AH headers protected by both modes; transport and tunnel. Figure 3-2 shows the format of an AH header.

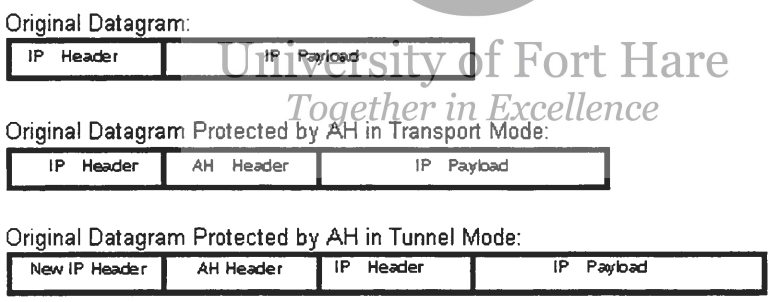


Figure 3-1: AH header in Transport and Tunnel modes

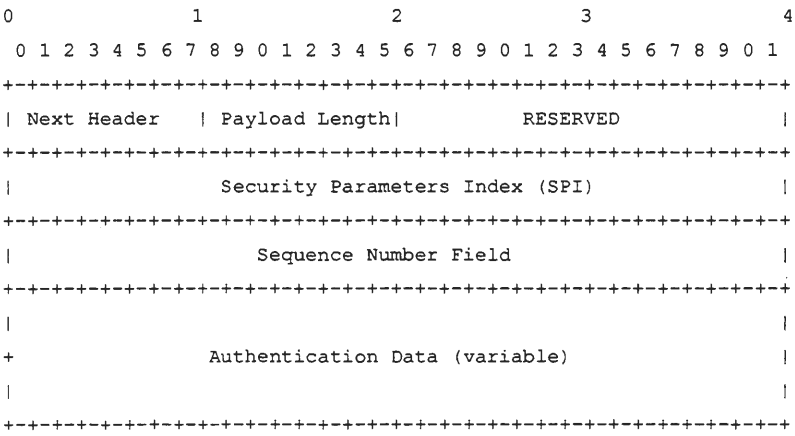


Figure 3-2: AH header format

The following sections give brief descriptions of each field in the AH header.

3.4.1.1 Next Header

The Next Header is an 8-bit field that identifies the type of the next payload after the Authentication Header. The value of this field is chosen from the set of IP Protocol Numbers defined in RFC 1700 [42] from the Internet Assigned Numbers Authority (IANA). For example, ESP header is 50 and 60 is Destination Options header.

3.4.1.2 Payload Length

This 8-bit field specifies the length of AH in 32-bit words (4-byte units), minus "2". All IPv6 extension headers, as per RFC 1883, encode the "Header Extension Length" field by first subtracting 1 (64-bit word) from the header length (measured in 64-bit words). AH is an IPv6 extension header. However, since its length is measured in 32-bit words, the "Payload Length" is calculated by subtracting 2 (32 bit words). In the "standard" case of a 96-bit authentication value plus the 3 32-bit word fixed portion, this length field will be "4". A "null" authentication algorithm may be used only for debugging purposes. Its use would result in a "1" value for this field for IPv4 or a "2" for IPv6, as there would be no corresponding Authentication Data field.

3.4.1.3 Reserved

This 16-bit field is reserved for future use. It must be set to zero.

3.4.1.4 Security Parameters Index (SPI)

The SPI is an arbitrary 32-bit value (random number) that, in combination with the destination IP address and security protocol (AH), uniquely identifies the Security Association for this datagram.

3.4.1.5 Sequence Number

This unsigned 32-bit field contains a monotonically increasing counter value. It is mandatory and is always present even if the receiver does not elect to enable

the anti-replay service for a specific SA. Processing of the Sequence Number field is at the discretion of the receiver, because it depends on whether the receiver wants to process anti-replay protection or not. Since the sender always assumes anti-replay to be enabled.

3.4.1.6 Authentication Data

This is a variable-length field that contains the Integrity Check Value (ICV) for this packet. The ICV is computed by the ESP packet minus the Authentication data. The length of the field is specified by the authentication function selected. This field is optional and is included only if the authentication service has been selected for the SA in question. The field must be an integral multiple of 32 bits in length by using padding [33].

3.4.2 Encapsulating Security Payload

ESP provides authentication, integrity, replay protection, and confidentiality of the data that is, it secures everything in the packet that follows the header and it is identified in the IP header as protocol ID 50. Replay protection requires authentication and integrity. ESP header allows for rewriting of the payload in encrypted form. The ESP header does not consider the fields of the IP header before it and therefore makes no guarantees about anything except the payload in a tunnel mode the original IP header is also authenticated. The various types of ESP applicable must follow RFC2406 [34]. An ESP header can also provide authentication for the payload, but not the outer header. Confidentiality can be used with or without authentication or integrity. Similarly, one could use authentication/integrity with or without confidentiality. A complete implementation of ESP uses two algorithms, one for authentication and the other for encryption. Figure 3-3 shows the ESP headers protected by both modes; transport and tunnel and Figure 3-4 shows the format of an ESP header.

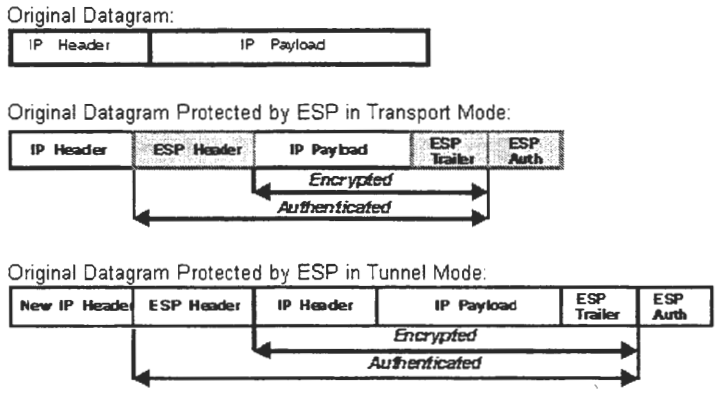


Figure 3-3: ESP header in Transport and Tunnel modes

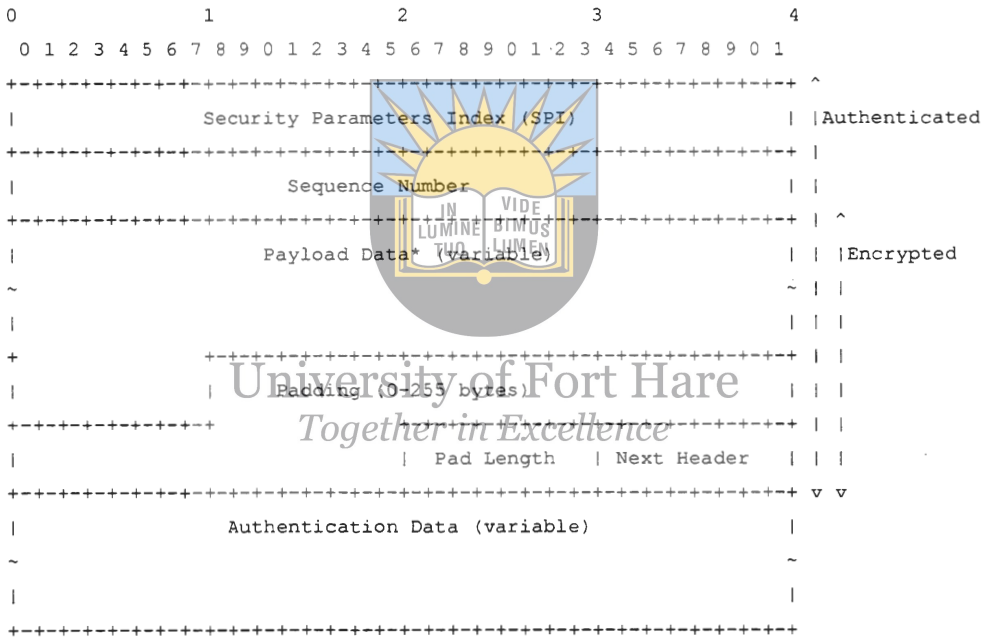


Figure 3-4: ESP packet format

Below is a description of each field in the ESP protected packet.

3.4.2.1 Security Parameters Index

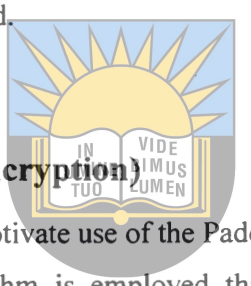
The SPI is an arbitrary 32-bit value that, in combination with the destination IP address and security protocol (ESP), uniquely identifies the Security Association for this datagram.

3.4.2.2 Sequence Number

This unsigned 32-bit field contains a monotonically increasing counter value (sequence number). It is mandatory and is always present even if the receiver does not elect to enable the anti-replay service for a specific SA.

3.4.2.3 Payload Data

Payload Data is a variable-length field containing data described by the Next Header field. The Payload Data field is mandatory and is an integral number of bytes in length. If the algorithm used to encrypt the payload requires cryptographic synchronization data, e.g., an Initialization Vector (IV), then this data may be carried explicitly in the Payload field.



3.4.2.4 Padding (for Encryption)

Several factors require or motivate use of the Padding field.

1. If an encryption algorithm is employed that requires the plain text to be a multiple of some number of bytes, e.g., the block size of a block cipher, the Padding field is used to fill the plaintext (consisting of the Payload Data, Pad Length and Next Header fields, as well as the Padding) to the size required by the algorithm.
2. Padding also may be required, irrespective of encryption algorithm requirements, to ensure that the resulting cipher text terminates on a 4-byte boundary. Specifically, the Pad Length and Next Header fields must be right aligned within a 4-byte word, as illustrated in the ESP packet format figure above, to ensure that the Authentication Data field (if present) is aligned on a 4-byte boundary.
3. Padding beyond that required for the algorithm or alignment reasons cited above may be used to conceal the actual length of the payload, in support of (partial) traffic flow confidentiality. However, inclusion of such additional padding has adverse bandwidth implications and thus its use should be undertaken with care.

The sender may add 0-255 bytes of padding. Inclusion of the Padding field in an ESP packet is optional, but all implementations must support generation and consumption of padding.

1. For the purpose of ensuring that the bits to be encrypted are a multiple of the algorithm's block size (first bullet above), the padding computation applies to the Payload Data exclusive of the IV, the Pad Length, and Next Header fields.
2. For the purposes of ensuring that the Authentication Data is aligned on a 4-byte boundary, the padding computation applies to the Payload Data inclusive of the IV, the Pad Length, and Next Header fields.

3.4.2.5 Pad Length

The Pad Length field indicates the number of pad bytes immediately preceding it. The range of valid values is 0-255, where a value of zero indicates that no Padding bytes are present. The Pad Length field is mandatory.

3.4.2.6 Next Header

The Next Header is an 8-bit field that identifies the type of data contained in the Payload Data field, e.g., an extension header in IPv6 or an upper layer protocol identifier. The Next Header field is mandatory.

3.4.2.7 Authentication Data

The Authentication Data is a variable-length field containing an Integrity Check Value (ICV) computed over the ESP packet minus the Authentication Data. The length of the field is specified by the authentication function selected [34].

3.5 IPSec Modes

IPSec functionality is applied depending on whether the endpoint doing the IPSec encapsulation is the original source of the data or a gateway:

3.5.1 Transport mode

Transport mode is used by a host that is generating the packets. In transport mode, the security headers are added in front of the transport layer headers, before the IP header is added to the packet. Protecting a packet in transport mode modifies the following original end-to-end IP header fields; next protocol ID and datagram length. Figure 3-5 shows the IP datagram protected using the IPSec in transport mode.

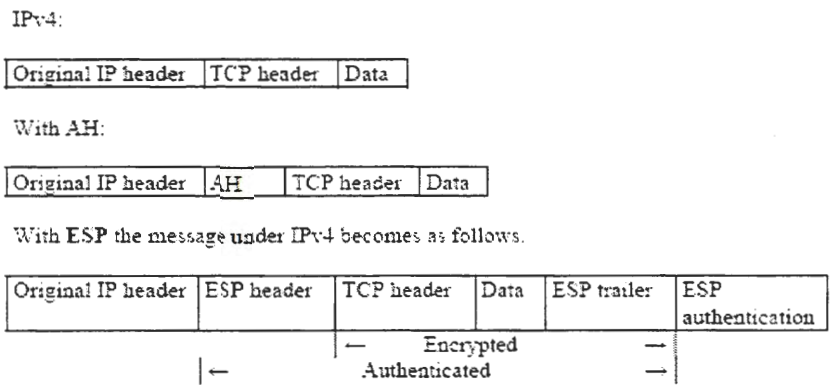


Figure 3-5: IP datagram protected in transport mode

3.5.2 Tunnel mode

Tunnel mode is used when the end-to-end IP header is already attached to the packet, and one of the ends of the secure connection is only a gateway. In this mode, the AH and ESP headers are used to cover the entire packet including the end-to-end header, and a new IP header is added to the packet that covers just the hop to the other end of the secure connection. Figure 3-6 shows the IP datagram protected using the IPSec in tunnel mode [37].

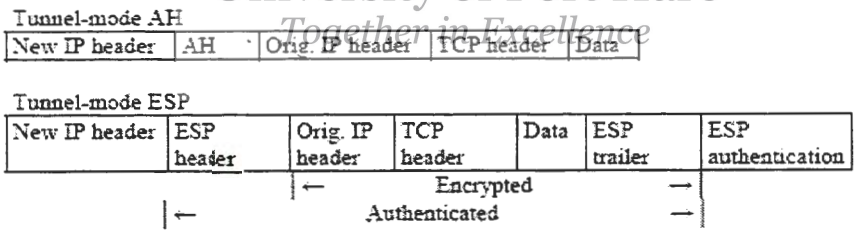


Figure 3-6: IP datagram protected in tunnel mode

3.6 Other Protocols and Services

IPSec needs the support of other protocols and services which a brief description will be given in this section.

3.6.1 Security Policies

A security policy is a rule that is programmed into the IPSec implementation that tells it how to process different datagrams received by the device. For example, security policies are used to decide if a particular packet needs to be processed by

IPSec or not; those that do not bypass AH and ESP entirely. If security is required, the security policy provides general guidelines for how it should be provided, and if necessary, links to more specific detail.

Security policies for a device are stored in the device's Security Policy Database (SPD).

3.6.2 Security Association

A Security Association (SA) is a set of security information that describes a particular kind of secure connection between two devices. An SA is needed for both the implementation of the IP ESP header and of the IP AH header. A SA is defined by three parameters. These are:

- Destination Address

  This is the address of the device for which the SA is established for.

- Security Parameters Index

  Is an arbitrary 32-bit number that is chosen to uniquely identify a particular SA for any connected device. The SPI is placed in AH or ESP datagrams and thus links each secure datagram to the security association. It is used by the recipient of a transmission so it knows what SA governs the datagram.

- Security Policy Identifier

  Specifies whether this association is for AH or ESP. If both are in use with this device they have separate SAs.

The security protocol headers do not contain information pertaining to the cryptographic algorithms and the associated parameters. These representations are achieved through the transmission of a Security Parameter Index (SPI).

The SPI at least contains the algorithm, algorithm mode and the keys used with the algorithm. In the ESP header case, certain sizes for determining synchronization and initialization of the encryption/decryption algorithm are also needed for the SPI. In addition, the SPI contains sensitivity level of data, for example, Secret or Unclassified for systems meant to provide multi-level security. The sending host uses the sending user ID and Destination Address to select an SA and hence SPI value.

The receiving host uses SPI value and Destination Address to distinguish the association. Hence, an AH implementation will always be able to use the SPI and the Destination Address to determine the security association and related security configuration data for all valid incoming IP packets. An SA is normally one-way. An authenticated communication between two hosts will have two SPI for both directions [35]. A device's security associations are contained in its Security Association Database (SAD).

3.6.3 Key Management

IPSec mandates support for two separate methods of cryptographic key and SA management: manual and automatic.

- Manual Key Management

  This is the simplest form of key management and involves each IPSec connection to be configured manually on both hosts. While this is suitable in small static situations, it is unsuitable in larger deployment scenarios due to scalability problems.

- Automatic Key and SA Management

  Larger deployment scenarios call for an Internet-standard, scalable and automated SA and key management protocol. This is provided by Internet Key Exchange (IKE). IKE is required to allow for use of anti-replay features of AH and ESP and to facilitate on-demand creation of SAs [5].

3.6.3.1 Internet Key Exchange

IPSec, like many secure networking protocol sets, is based on the concept of a shared secret. Two devices that want to send information securely encode and decode it using a piece of information that only they know. Before either AH or ESP can be used, however, it is necessary for the two devices to exchange the "secret" that the security protocols themselves will use. The primary support protocol used for this purpose in IPSec is called Internet Key Exchange (IKE). IKE is defined in RFC 2409 [41], and is one of the more complicated of the IPSec protocols to comprehend. IKE is considered a "hybrid" protocol because it combines and supplements the functions

of three other protocols. The first of these is the Internet Security Association and Key Management Protocol (ISAKMP). The other two protocols are Oakley and Skeme. All these protocols are explained briefly below. IKE comprises of two framework protocols, called ISAKMP and Oakley, which are very briefly described below

3.6.3.2 ISAKMP

This protocol provides a framework for exchanging encryption keys and security association information. It operates by allowing security associations to be negotiated through a series of phases. ISAKMP is defined in RFC 2408 [35]. ISAKMP is a generic protocol that supports many different key exchange methods that combines features from two key exchange protocols:

- **Oakley:** Describes a specific mechanism defined in RFC 2412 [44] for exchanging keys through the definition of various key exchange modes. Most of the IKE key exchange process is based on OAKLEY.
- **Skeme:** Describes a different key exchange mechanism than OAKLEY. IKE uses some features from SKEME, including its method of public key encryption and its fast re-keying feature.

IKE does not strictly implement either OAKLEY or SKEME but takes bits of each to form its own method of using ISAKMP.

Since IKE functions within the framework of ISAKMP, its operation is based on the ISAKMP phased negotiation process. There are two phases:

- **ISAKMP Phase 1:** The first phase is a setup stage where two devices agree on how to exchange further information securely. This negotiation between the two units creates a security association for ISAKMP itself; an ISAKMP SA. This security association is then used for securely exchanging more detailed information in Phase 2.
- **ISAKMP Phase 2:** In this phase the ISAKMP SA established in Phase 1 is used to create SAs for other security protocols. Normally, this is where the parameters for the real SAs for the AH and ESP protocols would be negotiated [42].

The ISAKMP security association negotiated during Phase 1 includes the negotiation of the following attributes used for subsequent negotiations:

- An encryption algorithm to be used.
- A hash algorithm.
- An authentication method, such as authentication using previously shared keys.
- A Diffie-Hellman group. Diffie-Hellman is a public-key cryptography mechanism. In this method, instead of encrypting and decrypting with the same key, data is encrypted using a public key knowable to anyone, and decrypted using a private key that is kept secret. A Diffie-Hellman group defines the attributes of how to perform this type of cryptography. Four predefined groups derived from OAKLEY are specified in IKE and provision is allowed for defining new groups as well.

Note that even though security associations in general are unidirectional, the ISAKMP SA is established bi-directionally. Once Phase 1 is complete, then, either device can set up a subsequent SA for AH or ESP using it.

3.7 IPSec Algorithms

Authentication is performed by hash functions in cryptography while encryption is done by block ciphers. IPSec uses block ciphers for encryption and hash function for authentication. A block cipher is an encryption function for fixed-sized blocks, currently block ciphers has a block size of 128 bits. These blocks take the 128-bit plaintext and generate a 128-bit ciphertext as the result. Block ciphers are reversible, that is, there is a decryption function that takes the 128-bit ciphertext and decrypts it to the original 128-bit plaintext, and hence the plaintext and ciphertext are always of the same size. Block ciphers are used mainly to encrypt information. IPSec usually uses them.

Block ciphers can either use symmetric and asymmetric keys. The Symmetric key works by encrypting data using a single key that can also be used to decrypt that data. The following are some of the symmetric key block ciphers DES, Triple DES (3DES) and AES. The Asymmetric key is a process where by each party has a public and private key. What is encrypted with the public key can only be decrypted with

the private key. The public keys are exchanged by both parties and data is encrypted by the source with the destinations public key. The data can then only be read by the destination or someone with access to the destinations private key. Rivest, Shamir and Adleman (RSA) and Error Checking and Correction (ECC) are some of the available asymmetric key block ciphers.

A hash function is a function that takes as input an arbitrarily long string of bits or bytes and produces a fixed size result which is irreversible. It is usually used for digital signatures for authentication, which is also its use in IPSec. Hash functions are also known as message digest functions, and the hash result is also known as the digest, or the fingerprint. The commonly used hash functions in IPSec are MD5 and SHA-1.

The IPSec protocol suite does not define the authentication and encryption algorithms used in implementations; these are defined in separate RFCs. In this section we will discuss commonly used algorithms involved in IPSec implementation.

3.7.1 DES

Data Encryption Standard (DES) is a block cipher that uses a key size of 56 bits and small block sizes of 64 bits and encrypts them, which makes it unsuitable for use in today's fast computers and large amounts of data. DES was developed in 1975 by Walter Tuchman who was the head of the DES research team at IBM and it was standardized in 1981 by American National Standards Institute (ANSI). The small key size problem is solved by 3DES, but there is no known fix for the small block size.

3.7.2 3DES

Triple DES or 3DES is a block cipher built from three DES encryptions in sequence, this solves the problem of small key size in DES. The first encryption is encrypted with a second key, and the resulting cipher text is again encrypted with a third key. Three 56-bit keys are used, instead of one, for an overall key length of 192 bits derived from three 56-bit DES key plus 8 parity bits per DES key. DES is slow by current standards and 3DES is one-third the speed of DES.

3.7.3 AES

The Advanced Encryption Standard (AES) also known as Rijndael is a block cipher adopted as an encryption standard by the US government. It was designed by Vincent Rijmen and Joan Daemen in 1998 who named it Rijndael, their proposal was selected from a group of five algorithms after National Institute of Standards and Technology (NIST) had asked for proposals from the cryptographic community and received 15 proposals. AES was adopted by NIST in November 2001 after a 5-year standardization process. AES has a fixed block size of 128 bits and a key size of 128, 192 or 256 bits [37].

3.7.4 MD5

Message Digest algorithm 5 (MD5) is a 128-bit hash function developed by Ronald Rivest in 1991[24][45] that was further developed from MD4. MD5 splits the message into blocks of 512 bits, with the last block padded and the length of the message is included. It splits the 128-bit state into four words of 32- its each.

According to the analysis of MD5 conducted by Joseph D. Touch [55] it appeared that since MD5 processes the whole packet, its implementation on software currently achieves only 85 Mbps and 256 Mbps on hardware, which basically under utilize the high performance software and hardware available these days that have much higher data rates. He recommended reconsideration in using MD5 as the default algorithm for IPSec data integrity.

3.7.5 SHA-1

The Secure Hash Algorithm (SHA) was designed by the National Security Agency of USA and standardized by National Institute of Science and Technology (NIST), the first version is often called SHA-0, which was later improved to SHA-1 after a weakness was found in SHA-0. It is a 160-bit hash function based on MD4 and it shares common features with MD5, but more conservative in design. SHA-1 is two to three times slower than MD5 [24].

3.8 IPSec on Windows

IPSec is the long-term direction for secure networking. It provides aggressive protection against private network and Internet attacks through end-to-end security. The only computers that must know about IPSec protection are the sender and receiver in the communication. In Windows XP and the Windows Server 2003 family, IPSec provides the ability to protect communication between workgroups, local area network computers, domain clients and servers, branch offices (which might be physically remote), extranets, and itinerant clients.

Windows uses 3DES as the primary encryption algorithm and DES for fallback purposes if 3DES is not supported, which is the case in Windows 2000, if the High Encryption Pack or Service Pack 2 (or later) is not installed. Computers running Windows XP and the Windows Server 2003 family support 3DES and do not require installation of the High Encryption Pack.

Microsoft introduced the new Security Configuration Wizard, which is bundled with Windows Server 2003 service pack 1. The Wizard works in conjunction with security policies. The resulting security policies can be applied to any server on the network, allowing for consistency and stability of the security settings on all servers. The security policies are created based on a baseline server. Once the security policy is created, it can be applied to the baseline server, or any other server in the organization.

Microsoft uses Diffie-Hellman groups to determine the length of the base prime numbers that are used during the key exchange process. Group 2048 (high) is stronger (more secure) than Group 2 (medium), which is stronger than Group 1 (low). Group 1 provides 768 bits of keying strength, Group 2 provides 1024 bits, and Group 2048 provides 2048 bits. Strong Diffie-Hellman groups combined with longer key lengths increase the computational difficulty of determining a secret key.

Generally breaking a key of particular cryptographic algorithm requires a lot of money to build a computer with such power. For instance, according to RFC 3766 [80] to exhaust the 3DES (112 bits) key space of 2^{112} you need US$1 trillion and it will take about 33 million years.

For enhanced security, it is recommended not to use Diffie-Hellman Group 1. For maximum security, Group 2048 is suggested whenever possible. Group 2 may be used when required for interoperability with Windows 2000 and Windows XP. Diffie-Hellman Group 2048 is provided only with the Windows Server 2003 family [14]. This research we used Group 2 for interoperability between Windows Server 2003 and Windows XP.

3.9 Attacks Protected by IPSec

IPSec protects data so that an attacker finds it extremely difficult or impossible to interpret it. The level of protection provided is determined by the strength of the security levels specified in the IPSec policy structure.

IPSec has a number of features that significantly reduce or prevent the following attacks:

- Sniffers (lack of confidentiality)

  The Encapsulating Security Payload (ESP) protocol in IPSec provides data confidentiality by encrypting the payload of IP packets.

- Data modification

  IPSec uses cryptography-based keys, shared only by the sending and receiving computers, to create a cryptographic checksum for each IP packet. Any modification to the packet data alters the checksum, which indicates to the receiving computer that the packet has been modified in transit.

- Identity spoofing, password-based, and application-layer attacks

  IPSec allows the exchange and verification of identities without exposing that information to interpretation by an attacker. Mutual verification (authentication) is used to establish trust between the communicating systems and only trusted systems can communicate with each other. After identities are established, IPSec uses cryptography-based keys, shared only by the sending and receiving computers, to create a cryptographic checksum for each IP packet. The cryptographic checksum ensures that only the computers that have knowledge of the keys could have sent each packet.

- Man-in-the-middle attacks

  IPSec combines mutual authentication with shared, cryptography-based keys.

- Denial-of-service attacks

  IPSec uses IP packet filtering methodology as the basis for determining whether communication is allowed, secured, or blocked, according to the IP address ranges, IP protocols, or even specific TCP and UDP ports [14].

3.10 Default IPSec Policies

Because the configuration of the IPSec rules can be somewhat complex, and because many environments will have the same level of need for their IPSec implementations, Microsoft includes three IPSec policies with Windows Server 2003. These policies are worth looking at before going to the effort of creating new policies. The three policies are Client (Respond Only), Server (Request Security) and Secure Server (Require Security). Each contains certain rules that define exactly how that policy affects IPSec traffic, but they can be summarized simply as follows:

When Client (Respond Only) is used, a Windows Server 2003 system will allow and provide for an IPSec connection if a client system requests it. All other connections will be allowed without IPSec. When Server (Request Security) is used, the Windows Server 2003 system will ask all incoming client connections to use IPSec. If the client is able to use IPSec, then it does precisely that.

If the client is not able to use IPSec, for example if the connection originates from an operating system that does not have an IPSec client, then the connection is till permitted to continue, though obviously without the security provided by IPSec. As the name suggests, when the Secure Server (Require Security) policy is used, all incoming requests to the server must be able to use IPSec encryption. If they cannot, then that connection is refused. The only exception to the rules that make up the Secure Server (Require Security) policy is that ICMP traffic is allowed to connect without encryption [14].

3.11 Weaknesses of IPSec

In IPv6 networks IPSec can be deployed from end-to-end, data may be encrypted along the entire path between the source node and the destination node. In IPv4 networks, IPSec is typically nowadays deployed between border routers of separate networks and for Virtual Private Networks (VPN).

The IPSec protocols are an excellent step in the right direction for Internet security. If correctly implemented and configured, the protocols could provide e-business and organizations like defense with the ability to take advantage of the speed and reach of the Internet without being as prone to the dangers of attack in an unpoliced environment. Leasing dedicated secure lines from telecommunications providers is prohibitively expensive, whereas dial-up to the internet is relatively cheap. By incorporating application and transport layer security measures such as Firewall, SSL and SSH along with IPSec, it would seem that a highly robust secure network could be created at some cost, that still needs to be evaluated.

In security circles there is a property called the weakest link property which states that; security architecture is only as strong as its weakest link. Ultimately IPSec is as strong as its encryption, hashing and key exchange algorithms and the weaknesses of these components are inherently weaknesses of IPSec. It goes without saying that the complexity of a security system does not measure its strength; hence complexity is the worst enemy of security.

Even if there is a secure tunnel between hosts for a specific type of traffic, and if the hosts itself is compromised from a separate unprotected connection, then all protected data will be available to the attacker. The sensible placement and monitoring of secure links created with IPSec is critical. IPSec is simply a tool and must be combined with other security measures such as Host Intrusion Detection Systems (HIDS), good key management, well configured firewalls, Socket Secure Layer (SSL) in case of HTTP and many others security mechanisms [46].

Applying IPSec to multicast is challenging and is a topic still under discussion. This is because an IPSec SA is identified by the triple protocol of (either ESP or AH), SPI, and destination address. The problem is the destination address, in case of multicast which points to more than one host.

IPSec is not just to protect the confidentiality of the data, but also to assure the authenticity of the sender and the integrity of the data (that it has not been changed in transit). The problem with Network Address Translation (NAT) is that NAT must change information in the packet headers in order to do its job. The first problem is that NAT changes the IP address of the internal computer to that of the NAT device. The IKE protocol used by IPSec embeds the sending computer's IP address in its payload, and this embedded address does not match the source address of the IKE packet (which is that of the NAT device). When these addresses do not match, the receiving computer will drop the packet.

In addition, NAT is not able to use the port numbers in TCP and UDP headers to multiplex packets to multiple internal computers when those headers have been encrypted by ESP.

The IPSec working group of the IEEE has created standards for NAT-Traversal (NAT-T) that are defined in RFC 3947 and RFC 3948 [81-82]. NAT-T is designed to solve the problems inherent in using IPSec with NAT. NAT-T adds a UDP header that encapsulates the ESP header. It sits between the ESP header and the outer IP header. This gives the NAT device a UDP header containing UDP ports that can be used for multiplexing IPSec data streams. NAT-T also puts the sending computer's original IP address into a NAT-OA (Original Address) payload. This gives the receiving computer access to that information so that the source and destination IP addresses and ports can be checked and the checksum validated. This also solves the problem of the embedded source IP address not matching the source address on the packet [83].

Ultimately the use of IPSec through NAT is a topic still under discussion and before deploying such a set up it is advisable to assess the security risks associated with it and taking into account the security needs of your network.

The use of compression might also bring about another weakness in IPSec, especially the encryption aspect of IPSec. If encryption takes place before compression, the patterns that are added to the plaintext to obtain a ciphertext by encryption might be removed by compression algorithms [76].

CHAPTER FOUR – TEST BED IMPLEMENTATION

This chapter details the approach and design that was used in carrying out this research, from the point of view of the implementation. It lists the tools that were used and why they were chosen. A detailed description of the security policies and algorithms used in implementing IPSec will be given together with all other components that are necessary in IPSec deployment.

4.1 Introduction

The research being undertaken involves setting up an experimental test bed, on which all the tests to be done will be done conducted. The initial design and approach used in creating this test bed was adapted from the Microsoft Help and Support Center utility on Microsoft Windows™ Server 2003. This was used as the basis for the infrastructure even though there are major differences. There are two setup infrastructures used in this test bed. The first was the IPv4 and IPv6 setup, followed by the IPSec infrastructure setup, since the research involves investigating the performance implications of IPSec on both IPv4 and IPv6 protocol stacks.

4.2 Implementation Tools

A number of tools were used which include software and hardware. Therefore, this section provides a brief description and explanation of each piece of hardware and software used.

4.2.1 Software

The software used in this research comprises first, of the testing platform which is the Microsoft Windows™ Operating Systems; Windows™ 2003 Server Service Pack 1 and Windows™ XP Professional Service Pack 2. The Windows platforms were chosen because Windows is a widely used desktop operating system. It is believed that the Windows operating systems consists of more than 95% of desktop computers on the Internet. And the research on Windows will give grounds for comparison with other operating systems. A network analyzer was also used and packet capture

software called Finisar Surveyor 5.5. The sections that follow will give a detailed description of software packages used in this research.

4.2.1.1 Windows™ XP Professional

Microsoft Windows™ XP Professional Service Pack 2 is the latest workstation operating system offering from Microsoft (at the time of writing). This platform provides production-quality developments in the technologies used in this research, that is, IPv6 and IPSec [50].

4.2.1.2 Windows™ Server 2003

Microsoft Windows™ Server 2003 Service Pack 1 is the latest Windows™ server operating system. This option was chosen from among other server operating systems such as Microsoft Windows™ NT 4.0 and Microsoft Windows™ 2000 because it offers latest developments in Microsoft's IPv6 and IPSec technologies.

4.2.1.3 Finisar Surveyor 5.5

Surveyor is a network monitoring and packet capturing tool by Finisar Corporation. It provides the following functions that allows for network data analysis.

- Capture – Captures data from a network and places it in system memory space (buffer) on an analyzer device. Surveyor allows for the creation and saving of capture filters that direct analyzer devices to capture only the information to be viewed and analyzed.
- Capture View – It allows for the observation of the data in a way that is useful for network analysis and enables the creation and saving of filters to view only the information to be analyzed. The data can be viewed in numerous ways and from different perspectives. Display of the data can be either as graphical charts or row-and-column tables.
- Filter – Surveyor allows one to create and save capture/display filters to collect/display only the information to be viewed and analyzed.

- Save – Move captured data from a capture buffer to a storage device on the Surveyor host PC. Surveyor enables the storage of captured data onto your hard drive for later viewing, analysis, or transmission.
- Record counter information – Surveyor enables the capture of all byte, frame, and error counter values compiled during the capture or transmission of data.
- Monitor – Real-time views for data seen on a network segment. This real-time data can be viewed in numerous ways and from different perspectives. Display of the real-time data can be either graphical charts or row-and-column tables [60].

4.2.2 Hardware

There were two main pieces of hardware that were used namely; computers and a switch, since this research involved setting up a network.

4.2.2.1 Computers

The isolated network for testing purposes consists of five computers each with a specific function which will be explained in detail in the following sections. There are two client computers and three server computers. Detailed functions of these computers will be given in section 4.3.

4.2.2.2 Switch

The D-Link DES-1008D is a Dual-Speed 8-port 10/100Mb Ethernet/Fast Ethernet NWay auto-negotiating switch. It is designed to eliminate unnecessary traffic, and relieve congestion by delivering dedicated bandwidth for each of the eight ports. The DES-1008D can be deployed with multiple high-speed servers for shared bandwidth to 10Mbps or 100Mbps workgroups. With the highest bandwidth at 200Mbps (100Mbps in full-duplex mode), any port can provide workstations with a congestion-free data pipe for simultaneous access of the server.

The DES-1008D complies with IEEE 802.3 10BASE-T and 802.3u 100BASE-TX, which specify NWay auto-negotiation, Flow control in half-duplex mode. The DES-1008D combines dynamic memory allocation with store-and forward switching to ensure that the buffer is effectively allocated for each port, while

controlling the data flow between the transmit and receive nodes to guarantee against all possible packet loss [61].

4.2.2.2.1 NWay Auto-negotiation

NWay™ is a technology used with Ethernet networking devices (such as router and switch) to automatically negotiate the highest possible common transmission speed between two devices. It was developed by National Semiconductor in 1994 for the IEEE 802.3u 100BASE-T working group in response to the networking industry's need for a mechanism to handle the connections between devices with varying connection speeds.

The NWay protocol (also known as auto-negotiation or auto-sensing) is defined in Clause 28 of the D4 draft of the ANSI/IEEE STD 802.3 MAC Parameters, Physical Layer, Medium Attachment Units and Repeater for 100 Mbps Operation [62]. NWay technology was chosen as the basis for this mechanism due to its simplicity, low cost, flexibility, interoperation with the installed base, and adaptability to future technologies [63].

4.3 Experimentation test bed

The experimentation test bed is a small network with five computers connected together by the D-Link DES-1008D switch using unshielded twisted pair (UTP) category 5 cables. The structure of our network, that is, the experimental test-bed is shown in Figure 4.1.

The computers have the following hardware specifications;

- Intel Pentium 4 CPU
- 2.8 GHz clock speed
- 512 MB of RAM
- 100 GB HDD

4.3.1 Functions of nodes

4.3.1.1 Domain Controller

A domain controller is a server that is running a version of the Windows™ Server 2003 Standard Edition operating system and has an Active Directory installed. Domain controllers store data and manage user and domain interactions, including user logon processes, authentication, and directory searches.

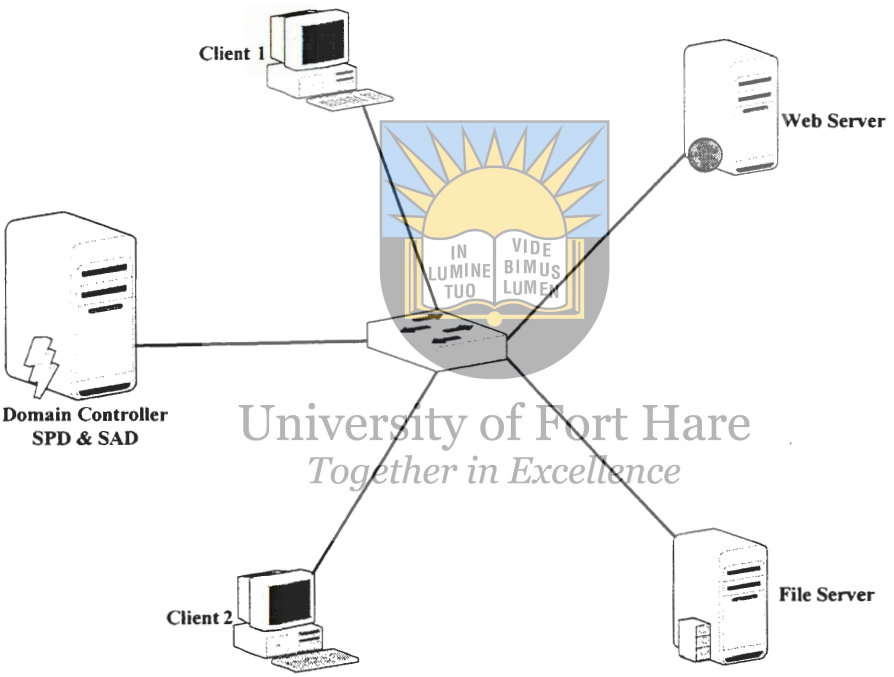


Figure 4-1: Experimental Test-bed Setup

4.3.1.1.1 Active Directory Users and Computers

The Active Directory is a central component of the Windows™ platform, which provides the means to manage the identities and relationships that make up network environments. This service is simpler to manage in Windows™ Server 2003. The Active Directory provides a single point of management for Windows™-based user accounts, clients, servers, and applications. It also helps organizations integrate systems not using Windows™ with Windows™-based applications, and Windows™-compatible devices, thus consolidating directories and easing

management of the entire network operating system. The successful operation of an Active Directory forest depends on clients and services being able to locate domain controllers.

The success of a domain controller location depends on the registration of information in the Domain Name System (DNS) and the availability of that information. Active Directory uses DNS to locate networked computers by resolving computer names to IP addresses.

4.3.1.1.2 Domain Name System

The DNS is a hierarchical, distributed database that contains mappings of DNS domain names to various types of data, such as IP addresses. DNS enables the location of computers and services by user-friendly names. This service is installed on the domain controller.

4.3.1.1.3 Certificate Authority

A certificate is a proof of validity for a public key issued by a certificate authority (CA). In addition to the key itself, a certificate contains information about the subject the certificate is issued to, when the certificate is valid, and the organization that issued the certificate. Windows™ Server 2003 Certificate Services supports the following two types of CA:

- Enterprise;
- Stand-alone.

Enterprise and stand-alone CA can be configured as either Root CA or Subordinate CA. Subordinate CA can further be configured as either Intermediate CA or Issuing CA. Before CA infrastructure is created there is the need to determine the type or types of CA that are to be used, and define the specialized roles that each CA is to assume.

Enterprise CA is integrated with Active Directory. They publish certificates and certificate revocation lists (CRL) to Active Directory. Enterprise CA use information stored in Active Directory, including user accounts and security groups, to approve or deny certificate requests. Enterprise CA use certificate templates. When a

certificate is issued, the enterprise CA uses information in the certificate template to generate a certificate with the appropriate attributes for that certificate type.

A Stand-alone CA does not require an Active Directory and does not use certificate templates. If a stand-alone CA is used, all information about the requested certificate type must be included in the certificate request. By default, all certificate requests submitted to stand-alone CA are held in a pending queue until a CA administrator approves them. A stand-alone CA can be configured to issue certificates automatically upon request, but this is less secure and is usually not recommended, because the requests are not authenticated.

Since an Active Directory is being used and the CA is needed to issue certificates automatically, an enterprise root CA is chosen.

4.3.1.1.4 Root CA

A root CA is the CA that is at the top of a certification hierarchy and must be trusted unconditionally by clients in your organization. All certificate chains terminate at a root CA. Whether you use enterprise or stand-alone CA, you need to designate a root CA.

4.3.1.1.5 Subordinate CA

A CA that is not a root CA is considered subordinate. The first subordinate CA in a hierarchy obtains its CA certificate from the root CA [49].

The domain controller acts as the IPSec gateway that stores the Security Policy Database (SPD) and the Security Association Database (SAD).

4.3.1.2 Internet Information Services Server

IIS server provides the web server service in the intranet to test secure web traffic. This service is runs on a computer using Microsoft Windows™ Server 2003 operating system; hence the use of the Internet Information Services (IIS) 6.0.

IIS 6.0 is a powerful Microsoft Web server that provides a highly reliable, manageable, and scalable Web application infrastructure for all versions of Windows™ Server 2003. IIS helps organizations increase Web site and application

availability while lowering system administration costs. IIS 6.0 supports the Microsoft Dynamic Systems Initiative (DSI) with automated health monitoring, process isolation, and improved management capabilities. Designed for intranets, the Internet, and extranets, IIS 6.0 makes it possible for organizations of all sizes to quickly and easily deploy powerful Web sites and applications [47].

4.3.1.3 File Server

File server is a computer running a Microsoft Windows™ Server 2003 operating system; this computer provides file and print sharing services to the intranet clients. File servers help manage access to files such as data files and network-accessible applications.

4.3.1.4 Clients

There are two computers in this infrastructure which acts as clients. These computers run the Microsoft Windows™ XP Professional operating system. The clients will be communicating across the network and accessing different resources from the network servers.

4.4 IPv6 implementation

Most people predict that IPv6 will be deployed at the edge of the network and gradually migrate inwards toward the core. The combination of deploying IPv6 transition technologies and migrating applications to be IPv6-capable is the key to getting started. This can easily and cheaply be done today. Migrating to an IPv6 infrastructure is a manageable, cost-effective, and gradual process when the organization makes well-defined, practical, and achievable plans and allows the network traffic to drive the upgrade schedule [50].

Microsoft Windows™ Server 2003 and Windows™ XP Professional operating systems have the latest IPv6 support on all Windows™ platforms; as a result the IPv6 protocol stack was installed on all the computers on the network. After installing IPv6 protocol stack on a Windows™ computer, it acquires additional network interfaces, each corresponding to the enabled IPv6 transition mechanisms that are supported. The computers end up being IPv6/IPv4 dual stack nodes, that is,

they have both IPv4 and IPv6 implemented. A node is IPv6-enabled if it has an IPv6 interface configured.

For coexistence to occur, the largest number of nodes (IPv4 or IPv6 nodes) can communicate using an IPv4 infrastructure, an IPv6 infrastructure, or an infrastructure that is a combination of IPv4 and IPv6. True migration is achieved when all IPv4 nodes are converted to IPv6-only nodes. However, for the foreseeable future, practical migration is achieved when as many IPv4-only nodes as possible are converted to IPv6/IPv4 nodes. IPv4-only nodes can communicate with IPv6-only nodes only when using an IPv4-to-IPv6 proxy or translation gateway.

4.4.1 IPv6 Support on Windows™

The IPv6 protocol for the Windows Server 2003 Family, Windows XP with SP1, Windows XP with SP2, and Windows CE .NET is a production-quality implementation capable of supporting a set of key scenarios and can be installed and uninstalled as a network protocol through the Network Connections folder. No official support for IPv6 protocol in Windows XP Professional without service pack or prior platforms is provided by Microsoft.

The key to moving applications to be IPv6-capable lies in IPv6 transition technologies, which allow IPv6 traffic to be encapsulated and sent over existing IPv4 networks such as the Internet and private intranets. The dominant IPv6 transition technologies used to support moving applications to IPv6 are the following:

- 6to4 for computers and devices that have public IPv4 addresses.
- Teredo for computers and devices that have private IPv4 addresses.
- Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) for enterprises that want to control how IPv6 is deployed on a private intranet.

Microsoft released an IPv6 Technology Preview for Windows™ 2000 in 2000 to provide developers with initial experience working with the new protocol [48]. This was continued to be provided in the later versions of Windows™ operating systems namely; Windows™ XP. Production-quality support for the IPv6 protocol stack is provided in Windows™ XP Service Pack 2, which includes support for the key IPv6

transition technologies mentioned above and an IPv6 host-based firewall (Windows Firewall).

For equivalent functionality for computers running Windows™ XP with Service Pack 1, hence it is necessary to download and install the Advanced Networking Pack for Windows™ XP. Windows™ Server 2003, released in March 2003, supports a production-quality IPv6 protocol stack, plus IPv6 support for some advanced networking services such as Domain Name System (DNS). Windows™ Server 2003 Service Pack 1 includes Windows™ Firewall. Other IPv6-capable platforms include Windows™ CE, Windows™ XP Embedded, SmartPhone, and others.

Microsoft plans to continue to deliver significant enhancements to IPv6 in the future releases of its products. The next release of the Windows operating system, codenamed "Longhorn," will natively support IPv6. A single protocol stack containing both IPv4 and IPv6 will replace the current dual-IP stack architecture. IPv6 will be installed and enabled by default, and will be the preferred Internet layer transport. The entire suite of networking services including DNS, Dynamic Host Configuration Protocol (DHCP), Point-to-Point Protocol (PPP), and Internet Protocol security (IPSec) will be IPv6-capable.

Longhorn is a movement that will spark a new wave of innovation for the entire industry. It is advertised as the most secure, highest quality release of Windows ever, and offers breakthrough capabilities for supporting communications, connectivity, mobility, management, and entertainment scenarios.

Additionally, Microsoft is working on delivering IPv6 support in its enterprise applications and services along with Longhorn. Products such as SQL Server, Microsoft Office, and others will have IPv6 support in the Longhorn time frame [50].

4.4.2 Installing IPv6

Windows Server 2003 Family, Windows XP with SP1 and Windows XP with SP2 come with IPv6 protocol stack disabled by default. This protocol can be installed using either the command line or the Network Connections folder but Windows XP uses only the command line option.

4.4.2.1 Command Prompt Installation

To install IPv6 protocol through the command prompt the following command is used:

```
ipv6 install
```

This creates all the IPv6 interfaces supported by the operating system of which some are enabled and some disabled by default and now available on Windows XP Professional SP2 only. Windows Server 2003 SP1 uses the network connections folder method. . To uninstall IPv6 the command is used:

```
ipv6 uninstall
```

4.4.2.2 Network Connections Installation

The Network connections folder installation method that is available in Windows Server 2003 Family, Windows XP with SP1 and Windows XP with SP2, installs IPv6 protocol like any other protocol using a GUI interface by adding Microsoft IPv6 Developer Edition under protocols. Configuration and all other settings use the command prompt through either the 'ipv6' command family or 'netsh interface ipv6' command family. Microsoft says it will discontinue the use of 'ipv6' commands in favor of 'netsh interface ipv6' commands for all platforms.

4.4.2.3 Different types of addresses

Many addresses are configured on one interface. IPv6 has a concept called 'scope', and has three types of addresses according to the scope: Link-local, Site-local, and Global address. One interface can have all these addresses, which are used for different purposes.

The Link-local address is the first address configured when an interface is attached to the network, but it is effective only on the same link. In other words, communication with a link-local address does not go beyond the router. But a link-local address is not used for data communications; it is used for control functions. A site-local address is limited for use in a site, for internal communication in a company. But usually, IPv6 connected computers use global address.

4.4.2.4 Automatically created addresses

When a connection is made to global IPv6 Internet with Windows XP, the computer automatically creates a Global IPv6 Address for the connected network interface immediately. The computer does not ask any server. The only information it gets from others is "Network Prefix," the number of network the computer resides on. It combines that with "Interface ID," which the computer creates on its own. 64-bit Network Prefix and 64-bit Interface ID makes 128bit Global IPv6 Address.

The information used by the computer for making an Interface ID is a MAC address, in the case of Ethernet adapter. A MAC address is usually 48bit address shown in hexadecimal form. The first 24-bit is vendor ID, registered to IEEE. The latter 24-bit is a unique ID or serial number assigned by the vendor. So a MAC address is supposed to be unique in the world.

A MAC address is 48 bits, 16 bits short for Interface ID, which is 64 bits. So some tricks are used, like inserting 'fffe' in the middle. A network prefix is obtained using a link-local address, first 64 bits are always 'fe:/64.' A computer just connected to the network first creates a Link-Local Address by combining 'fe:/64' and Interface ID computed from the MAC address of the connected interface. It confirms that the same address is not used on the same link, just in case. Then it becomes ready to communicate.

On the other hand, a router sends a message called Router Advertisement (RA) periodically. This message contains network prefix information. The connected computer receives this information and combines this with Interface ID to form Global IP Address.

A computer sends a message called the Router Solicitation (RS) to request prefix information. At this point, the computer has no knowledge of the router address, but it uses a multicast address 'ff02::2,' which any routers are required to receive. The computer sends RS message with this as destination address, and the nearest router sends back RA.

The IPv6 protocol for the Windows Server 2003 family supports the following automatic tunneling technologies:

- 6to4 is enabled by default.
- ISATAP is enabled by default.
- IPv6 Automatic Tunneling is disabled by default.
- 6over4 is disabled by default.
- Teredo is enabled by default. It is only supported for Windows XP (SP1 and later) when the Advanced Networking Pack for Windows XP is installed.

4.4.3 Configuring IPv6

There are two methods that can be used to configure IPv6 protocol on Windows, these are:

- Automatic configuration using stateless addresses
- Manual configuration

4.4.3.1 Automatic configuration

The IPv6 protocol for Windows XP supports address auto-configuration for stateless addresses as defined in RFC 2462 [53]. IPv6 hosts and routers automatically create unique link-local addresses for all LAN interfaces that appear to be Ethernet interfaces. IPv6 hosts use received Router Advertisement messages to automatically configure:

- A default router.
- The default setting for the Hop Limit field in the IPv6 header.
- The determination of whether the node should use a stateful address configuration protocol, such as Dynamic Host Configuration Protocol for IPv6 (DHCPv6), for addresses and other configuration requirements. The IPv6 protocol for Windows XP does not support DHCPv6 or any other stateless address configuration protocol.
- The timers used in Neighbor Discovery processes.
- The maximum transmission unit (MTU) of the local link.

- The list of network prefixes that are defined for the link. Each network prefix contains both the IPv6 network prefix and its valid and preferred lifetimes. If indicated, a network prefix is combined with the interface identifier to create a stateless IPv6 address configuration for the receiving interface. A network prefix also defines the range of addresses for nodes on the local link.

Additionally, the IPv6 protocol for Windows XP automatically configures:

- IPv4-compatible addresses on an automatic tunneling interface.
- 6to4 addresses on a 6to4 tunneling interface for all public IPv4 addresses that are assigned to the computer.
- Intra-site Automatic Tunnel Addressing Protocol (ISATAP) addresses on an automatic interface for all IPv4 addresses that are assigned to the computer.
- Routes to off-link prefixes (if the off-link address prefix is advertised by a router running Windows XP).

4.4.3.2 Types of auto-configuration

There are three types of auto-configuration:

4.4.3.2.1 Stateless

Configuration of addresses is based on the receipt of Router Advertisement messages. These messages include stateless address prefixes that require that hosts not use a stateful address configuration protocol.

4.4.3.2.2 Stateful

Configuration is based on the use of a stateful address configuration protocol, such as DHCPv6, to obtain addresses and other configuration options. A host uses a stateful address configuration when it receives Router Advertisement messages that do not include address prefixes and require that the host use a stateful address configuration protocol. A host will also use a stateful address configuration protocol when there are no routers present on the local link.

4.4.3.2.3 Both

Configuration is based on the receipt of Router Advertisement messages. These messages include stateless address prefixes and require that hosts use a stateful address configuration protocol.

For all auto-configuration types, a link-local address is always configured.

4.4.3.3 Auto-configuration process

The address auto-configuration process for an IPv6 node occurs as follows:

1. A tentative link-local address is derived, based on the link-local prefix of FE80::/64 and the 64-bit interface identifier.
2. Duplicate address detection is performed to verify the uniqueness of the tentative link-local address.
3. If duplicate address detection fails, manual configuration must be performed on the node.
4. If duplicate address detection succeeds, the tentative link-local address is assumed to be unique and valid. The link-local address is initialized for the interface. The corresponding solicited-node multicast link-layer address is registered with the network adapter.

An IPv6 host, address auto-configuration continues as follows:

1. The host sends a Router Solicitation message.
2. If no Router Advertisement messages are received, then the host uses a stateful address configuration protocol to obtain addresses and other configuration parameters. The IPv6 protocol for Windows XP does not support the use of a stateful address configuration protocol.
3. If a Router Advertisement message is received, the configuration information that is included in the message is set on the host.
4. For each stateless auto-configuration address prefix that is included:
    - The address prefix and the appropriate 64-bit interface identifier are used to derive a tentative address.

- Duplicate address detection is used to verify the uniqueness of the tentative address.
    - If the tentative address is in use, the address is not initialized for the interface.
    - If the tentative address is not in use, the address is initialized. This includes setting the valid and preferred lifetimes based on information included in the Router Advertisement message.
        - If it is specified in the Router Advertisement message, the host uses a stateful address configuration protocol to obtain additional addresses or configuration parameters.

4.4.3.4 Manual Configuration

You can manually configure IPv6 addresses and routes by using the ipv6.exe command-line tool. Manual configuration might be required in a network that has multiple IPv6 network segments within which the IPv6 routers are not configured to send router advertisements [14]..

4.4.3.5 6to4 Configuration

6to4 is an address assignment and router-to-router automatic tunneling technology that is used to provide unicast IPv6 connectivity between IPv6 sites and hosts across the IPv4 Internet. 6to4 hosts do not require any manual configuration and create 6to4 addresses through standard IPv6 address auto-configuration mechanisms. 6to4 routers, however, require additional processing logic for encapsulation and decapsulation and, depending on the implementation, might require additional configuration.

6to4 uses the global address prefix 2002:WWXX:YYZZ::/48, where 2002 is a Top-Level Aggregation Identifiers (TLA ID) reserved for 6to4 addresses and WWXX:YYZZ is the Next-Level Aggregation Identifiers (NLA ID), which corresponds to the colon-hexadecimal representation of a public IPv4 address (w.x.y.z) assigned to the host or site. The full address of a 6to4 node is

2002:WWXX:YYZZ:[SLA ID]:[Interface ID]. Where SLA ID is the Site-Level Aggregation Identifier.

Within an IPv6 intranet, local IPv6 routers advertise a 2002:WWXX:YYZZ:[SLA ID]::/64 prefix so that hosts can create an auto-configured 6to4 address. 64-bit prefix routes within the IPv6 intranet are used to deliver traffic between 6to4 hosts on separate subnets. Additionally, a 2002::/16 route is used to tunnel IPv6 traffic to other 6to4 hosts outside the intranet. All 6to4 traffic that does not belong within the site is forwarded by the routing infrastructure to the 6to4 router on the border of the IPv6 intranet. 6to4 traffic for another site, received by the 6to4 router, is encapsulated in an IPv4 header and sent to the destination IPv4 address, which corresponds to the public IPv4 address embedded in the NLA ID portion of the destination IPv6 address.

After 6to4 traffic is received by the IPv4 destination (a 6to4 router), it is decapsulated and forwarded to the appropriate node by the routing infrastructure of the destination IPv6 intranet. RFC 3056 [26] defines the following terms:

- 6to4 host

  An IPv6 host that is configured with at least one 6to4 address.

- 6to4 router

  An IPv4/IPv6 router that forwards 6to4-addressed traffic between 6to4 hosts within a site and other 6to4 routers (or 6to4 relay routers) on an IPv4 internetwork, such as the Internet.

- 6to4 relay router

  Is an IPv4/IPv6 router that forwards 6to4-addressed traffic between 6to4 routers and hosts on a native IPv6 test bed known as the 6bone.

When you use 6to4 hosts, an IPv6 routing infrastructure within 6to4 sites, a 6to4 router at site boundaries, and a 6to4 relay router, the following types of communication can occur:

- A 6to4 host can communicate with another 6to4 host within the same site.

This type of communication is available through the IPv6 routing infrastructure, which provides reachability to all hosts within the site.

- A 6to4 host can communicate with 6to4 hosts in other sites across the IPv4 Internet.

  This type of communication occurs when a 6to4 host forwards IPv6 traffic that is destined to a 6to4 host in another site to the local site 6to4 router. The local site 6to4 router encapsulates the IPv6 traffic with an IPv4 header and sends it to the 6to4 router at the destination site on the Internet. The 6to4 router at the destination site removes the IPv4 header and forwards the IPv6 packet to the appropriate 6to4 host.

- A 6to4 host can communicate with hosts on the 6bone.

  This type of communication occurs when a 6to4 host forwards IPv6 traffic that is destined for a 6bone host to the local site 6to4 router. The local site 6to4 router encapsulates the IPv6 traffic with an IPv4 header and sends it to a 6to4 relay router that is connected to both the IPv4 Internet and the 6bone. The 6to4 relay router removes the IPv4 header and forwards the IPv6 packet to the appropriate 6bone host.

All of these types of communication use IPv6 traffic without the requirement of obtaining either a direct connection to the 6bone or an IPv6 global address prefix from an ISP.

4.5 6to4 Support in Windows XP and Windows Server 2003 Family

Support for 6to4 hosts and 6to4 routers is provided by the 6to4 component that is part of the IPv6 protocol for Windows XP and Windows Server 2003 family. If there is a public IPv4 address assigned to an interface on the host and a global prefix is not received in a router advertisement, the 6to4 component:

- Automatically configures 6to4 addresses on the 6to4 Tunneling Pseudo-Interface for all public IPv4 addresses that are assigned to interfaces on the computer.

- Automatically creates a 2002::/16 route that forwards all 6to4 traffic with the 6to4 Tunneling Pseudo-Interface (interface index 3). All traffic forwarded by this host to 6to4 destinations is encapsulated with an IPv4 header.
- Automatically performs a DNS query to obtain the IPv4 address of a 6to4 relay router on the Internet.

The results of the 6to4 component auto-configuration vary depending on the configuration of the host. Any host that is running the IPv6 protocol for Windows XP and is configured with an IPv4 public address is automatically configured as a 6to4 host. A 6to4 host can perform its own tunneling and reach 6to4 hosts on the Internet, 6to4 hosts in other sites, or hosts on the 6bone [8].

By default, the IPv6 protocol for Windows™ XP configures link-local addresses for each interface that corresponds to each installed Ethernet network adapter.

The 6to4 configuration focuses mainly on the IPv6 connectivity between IPv6 hosts within one site. This configuration requires only the installation of the IPv6 protocol on at least two nodes on the same network segment (also known as a link or subnet) with no intermediate routers.

4.5.1 ISATAP Configuration

ISATAP is an address assignment and host-to-host, host-to-router, and router-to-host automatic tunneling technology that is used to provide unicast IPv6 connectivity between IPv6 hosts across an IPv4 intranet. ISATAP can be used for communication between IPv6/IPv4 nodes on an IPv4 network. ISATAP addresses use the locally administered interface identifier ::0:5EFE:w.x.y.z where:

•The 0:5EFE portion is formed from the combination of the Organizational Unit Identifier (OUI) that is assigned to the Internet Assigned Numbers Authority (IANA) (00-00-5E), and a type that indicates an embedded IPv4 address (FE).

•The w.x.y.z portion is any unicast IPv4 address, which includes both public and private addresses.

The ISATAP interface identifier can be combined with any 64-bit prefix (including 6to4 prefixes) for IPv6 unicast addresses.

4.5.2 ISATAP Support in Windows XP

By default, the IPv6 protocol for Windows XP automatically configures the ISATAP address of FE80::5EFE:w.x.y.z on the Automatic Tunneling Pseudo-Interface for each IPv4 address that is assigned to the node. This ISATAP address, based on the link-local prefix of FE80::/64, allows two hosts to communicate over an IPv4 network by using each other's link-local ISATAP addresses.

The use of link-local ISATAP addresses allows IPv6/IPv4 hosts on an IPv4 intranet to communicate with each other, but not with other IPv6 hosts that are outside of the site. To communicate with IPv6 hosts that are outside of the site, the following additional configuration is required:

- A host must receive a router advertisement from a router, typically the site border router that contains a global address prefix.
- A host must have a default route that points to an ISATAP address that corresponds to the intranet interface of the site border router.

The site border router is the router between the intranet and the Internet or 6bone. A site border router is most often a 6to4 router that is connected to the Internet. After receiving the router advertisement from the site border router, additional ISATAP addresses that are based on the global prefix are automatically configured.

For example, if the site is connected to the 6bone and a host (configured with the IPv4 address of 10.40.1.29) receives the global prefix of 3000::/64 in a router advertisement, the ISATAP address of 3000::5EFE:10.40.1.29 is automatically configured. Without a global address prefix and a 6bone connection, a site can use a 6to4-based global address prefix, connecting through the IPv4 Internet to other 6to4 sites, 6to4 hosts, and the 6bone. If the site is using the 6to4 address prefix of 2002:836B:1:5::/64 (based on the public address of 131.107.0.1 and an SLA ID of 5), the ISATAP address of 2002:836B:1:5:0:5EFE:10.40.1.29 is automatically configured.

The following illustration shows two ISATAP hosts communicating across the Internet even when each site is using the 192.168.0.0/16 private address space internally. Two hosts using the same IPv4 address in separate sites might have the

same ISATAP-derived interface ID, but they still have globally unique IPv6 addresses because of the unique subnet ID that is based on a unique IPv4 public address.

By using the transition technologies of 6to4 and ISATAP, IPv6/IPv4 nodes can communicate across private IPv4 intranets and the Internet through IPv4-encapsulated IPv6 traffic. When 6to4 and ISATAP are used together, 6to4 provides the first 64 bits of the address (the subnet prefix) and ISATAP provides the last 64 bits of the address (the interface ID) [8].

4.5.3 Teredo Configuration

Teredo, also known as IPv4 network address translator (NAT) traversal for IPv6, provides address assignment and host-to-host automatic tunneling for unicast IPv6 connectivity across the IPv4 Internet when IPv6/IPv4 hosts are located behind one or multiple IPv4 NATs. To traverse IPv4 NATs, IPv6 packets are sent as IPv4-based User Datagram Protocol (UDP) messages. 6to4 provides a similar function as Teredo; however, 6to4 router support is required in the edge device that is connected to the Internet. 6to4 router functionality is not widely supported by IPv4 NATs. Even if the NAT were 6to4-enabled, 6to4 would still not work for configurations in which there are multiple NATs between a site and the Internet.

Teredo resolves the issues of the lack of 6to4 functionality in modern-day NATs or multi-layered NAT configurations by tunneling IPv6 packets between the hosts within the sites. In contrast, 6to4 uses tunneling from the edge device. Tunneling from the hosts presents another issue for NATs: IPv4-encapsulated IPv6 packets are sent with the Protocol field in the IPv4 header set to 41, which is the IPv6 protocol ID. Most NATs only translate TCP or UDP traffic and must either be manually configured to translate other protocols or have an installed NAT editor that handles the translation. Because Protocol 41 translation is not a common feature of NATs, IPv4-encapsulated IPv6 traffic will not flow through typical NATs. Therefore, the IPv6 packet is encapsulated as an IPv4 UDP message, containing both IPv4 and UDP headers.

UDP messages can be translated by most NATs and can traverse multiple layers of NATs. It is important to note that Teredo is designed as a last resort transition technology for IPv6 connectivity.

If native IPv6, 6to4, or ISATAP connectivity is present between communicating nodes, Teredo is not used. As more IPv4 NATs are upgraded to support 6to4 and IPv6 connectivity becomes ever-present, Teredo will be used less and less, until eventually it is not used at all [8].

4.5.3.1 Teredo components

The components of Teredo include the following:

Teredo client

An IPv6/IPv4 node that supports a Teredo tunneling interface through which packets are tunneled either to other Teredo clients or to nodes on the IPv6 Internet (through a Teredo relay).

Teredo server

An IPv6/IPv4 node that is connected to both the IPv4 Internet and the IPv6 Internet. The role of the Teredo server is to assist in the initial configuration of Teredo clients and to facilitate the initial communication between either different Teredo clients or between Teredo clients and IPv6-only hosts.

Teredo relay

An IPv6/IPv4 router that can forward packets between Teredo clients on the IPv4 Internet and IPv6-only hosts.

Teredo host-specific relay

An IPv6/IPv4 node that has an interface and connectivity to both the IPv4 Internet and the IPv6 Internet and can communicate directly with Teredo clients over the IPv4 Internet, without the need for an intermediate Teredo relay. The connectivity to the IPv4 Internet can be through a public IPv4 address or through a private IPv4 address and a neighboring NAT. The connectivity to the IPv6 Internet can be through a direct connection to the IPv6 Internet or through an IPv6 transition technology such as 6to4.

4.5.3.2 Teredo addresses

Teredo address consists of the following parts:

The first 32 bits are for the Teredo prefix, which is the same for all Teredo addresses. The Internet Assigned Numbers Authority (IANA) has not yet defined this prefix, although the prefix 3FFE:831F::/32 is used for initial deployment.

The next 32 bits contain the IPv4 public address of the Teredo server that assisted in the configuration of this Teredo address.

The next 16 bits are reserved for Teredo flags. The only defined flag is the high-order bit known as the Cone flag. The Cone flag is set when the NAT connected to the Internet is a cone NAT.

The next 16 bits store an obscured version of the external UDP port that corresponds to all Teredo traffic for this Teredo client. When the Teredo client sends its initial packet to a Teredo server, the source UDP port of the packet is mapped by the NAT to a different, external UDP port. All Teredo traffic for the host uses the same external, mapped UDP port. The external port is obscured by exclusive ORing (XORing) the external port with 0xFFFF. For example, the obscured version of the external port 5000 in hexadecimal format is EC77 (5000 equals 0x1388, and 0x1388 XOR 0xFFFF equals 0xEC77). Obscuring the external port prevents NATs from translating it within the payload of the packets that are being forwarded.

The last 32 bits store an obscured version of the external IPv4 address that corresponds to all Teredo traffic for this Teredo client. Just like the external port, when the Teredo client sends its initial packet to a Teredo server, the source IP address of the packet is mapped by the NAT to a different, external address.

Obscuring the external address prevents NATs from translating it within the payload of the packets that are being forwarded.

4.5.3.3 How Teredo works

For two computers running the Advanced Networking Pack for Windows XP, the most crucial Teredo processes are those used for initial configuration and communication with a peer in a different site. Initial configuration for Teredo clients

is accomplished by sending a series of Router Solicitation messages to Teredo servers. The responses are used to derive a Teredo address and determine whether the client is behind a cone, restricted, or symmetric NAT. If the Teredo client is behind a symmetric NAT, then it cannot function. The type of NAT the Teredo client has discovered from the display of the **netsh interface ipv6 show teredo** command can be seen. Based on the received Router Advertisement messages, the Teredo client constructs its Teredo address from the following:

The first 64 bits are set to the value included in the Prefix Information option of the received router advertisement. The 64-bit prefix advertised by the Teredo server consists of the Teredo prefix (32 bits) and the public IPv4 address of the Teredo server (32 bits).

The next 16 bits are either 0x8000 (cone NAT) or 0x0 (restricted NAT).

The next 16 bits are set to the obscured external UDP port number that is included in a special Teredo header in the router advertisement.

The last 32 bits are set to the obscured external IP address that is included in a special Teredo header in the router advertisement.

4.5.4 Teredo Support on Windows

The Advanced Networking Pack for Windows XP includes Teredo client and Teredo host-specific relay functionality. When the Advanced Networking Pack for Windows XP is installed, the Teredo host-specific relay functionality is automatically enabled if a global IPv6 address has been assigned. A global address can be assigned from a Router Advertisement message that is received from a native IPv6 router, an ISATAP router, or a 6to4 router. If there is no global address configured, Teredo client functionality is enabled [8].

4.6 IPSec implementation

IPSec is an Internet standard that is designed to provide data confidentiality, integrity and authentication through the use of the Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols and its support is built-in into IPv6. IPSec was designed to be independent of underlying network topologies and

provides transparent services to the application layer. In addition, IPSec is independent of cryptographic algorithms. This allows for seamless integration of new and stronger encryption algorithms into the IPSec architecture.

4.6.1 IPSec Components

Windows XP and Windows 2003 Server IPSec is comprised of the following components; IPSec Policy Agent service, Internet Key Exchange and IPSec driver.

4.6.2 IPSec Policy Agent Service

The purpose of the IPSec Policy Agent is to retrieve policy information and pass it to other IPSec components that require this information to perform security services on a Windows Server 2003 operating system, and it appears as IPSec services in the list of system services in the Services console.

The IPSec Policy Agent has the following functions:

- It retrieves the appropriate IPSec policy, if one has been assigned from the Active Directory, if the computer is a domain member, or from the local registry, if the computer is not a member of a domain.
- Polls for changes in policy configuration.
- Sends the assigned IPSec policy information to the IPSec driver.

If the computer is a member of a domain, policy retrieval occurs when the system starts, at the interval specified in the IPSec policy, and at the default Winlogon polling interval. You can also manually poll Active Directory for policy using the `gpupdate /target:computer` command.

The following are additional aspects of IPSec policy behavior for a computer that is a member of a domain:

- If IPSec policy information is centrally configured for computers that are domain members, the IPSec policy information is stored in Active Directory and cached in the local registry of the computer to which it applies.

- If the computer is temporarily not connected to the domain and policy in which it is cached, new policy information for that computer replaces old, cached information when the computer reconnects to the domain.
- If the computer is a stand-alone computer or a member of a domain that does not use Active Directory for policy storage, the IPSec policy is stored in the local registry.

The IPSec Policy Agent starts automatically at system start time. If there are no IPSec policies in the Active Directory or the registry, or if the IPSec Policy Agent cannot connect to the Active Directory, the IPSec Policy Agent waits for policy to be assigned or activated.

4.6.3 Internet Key Exchange

Before secured data can be exchanged, a security agreement between the two computers must be established. In this security agreement, called a security association (SA), both nodes agree on how to exchange and protect information. To build this agreement between the two computers, the IETF has established a standard method of security association and key exchange resolution named Internet Key Exchange (IKE) which:

- Centralizes security association management, reducing connection time.
- Generates and manages shared, secret keys that are used to secure the information.

This process does not only protect communication between computers, it also protects remote computers that request secure access to a corporate network. In addition, this process works whenever the negotiation for the final destination computer (endpoint) is performed by a security gateway.

4.6.4 IPSec driver

The IPSec driver receives the active IP filter list from the IPSec Policy Agent, and then attempts to match every inbound and outbound packet against the filters in the list. When a packet matches a filter, it applies the filter action. When a packet does

not match any filters, the packet is passed back without modification to the TCP/IP driver to be received or transmitted.

If the filter action permits transmission, the packet is received or sent with no modifications. If the action blocks transmission, the packet is discarded. If the action requires the negotiation of security, main mode and quick mode SAs are negotiated.

The negotiated quick mode security association (SA) and keys are used with both outbound and inbound processing. The IPSec driver stores all current quick mode SAs in a database. The IPSec driver uses the Security Parameters Index (SPI) field to match the correct SA with the correct packet. SPI is a four byte field present in every IPSec header that is used to identify a particular security policy and this value is the same for the rest of the session. This means the SPI value changes when the same policy is used in another transmission.

When an outbound IP packet matches the IP filter list with an action to negotiate security, the IPSec driver queues the packet and then notifies Internet Key Exchange (IKE), which begins security negotiations with the destination IP address of that packet. If several outbound packets are going to the same destination and match the same filter before IKE has finished the negotiation, then only the last packet sent is saved. After a successful negotiation is completed, the IPSec driver on the sending computer:

1. Receives the SA containing the session key from IKE.
2. Locates the outbound SA in its database, and inserts the SPI from the SA into the AH or ESP header.
3. Signs the packets (and encrypts them if confidentiality is required).
4. Sends the packets to the IP layer to be forwarded to the destination computer.

If the negotiation failed, the IPSec driver discards the packet.

When an IPSec-secured inbound packet matches a filter in the IP filter list, the IPSec driver:

1. Receives the session key, SA, and SPI from IKE.
2. Locates the inbound SA in its database by destination address and SPI.

3. Checks the signature and, if required, decrypts the packets.

4. Matches the IP packet searches for a matching filter in the filter list to ensure that no traffic, other than what was agreed upon during the negotiation, is received.

5. Sends packets to the TCP/IP driver to pass to the receiving application.

When an unsecured IP packet is received, the IPSec driver searches for a matching filter in the filter list. If a match occurs and the filter action for that filter either requires IP security or blocks the packet, then the packet is discarded [56].

4.7 IPSec Configuration

An IPSec policy consists of one or more rules that determine IPSec behavior. Each IPSec rule contains the following configuration items. Included also are configuration items we created to deploy IPSec on our network.

4.7.1 IP Filter Lists

A filter list contains one or more predefined packet filters that describe the types of traffic to which the configured filter action for this rule is applied. One IP filter list is created that has the following properties; it matches all IP packets from the source IP address to any IP address for all protocols, except ISAKMP. Windows Sever 2003 allow the filtering of all protocols except IKE because it is not yet supported. It provides the option of exempting certain protocols by editing registry keys.

There is a registry key called `NoDefaultExempt` that is edited as follows to exempt protocols from IPSec filtering;

- A value of 0 specifies that multicast, broadcast, Resource Reservation Protocol (RSVP), Kerberos, and ISAKMP traffic are exempt from IPSec filtering. This is the default filtering behavior for Windows 2000 and Windows XP. This setting must be used only if necessary, for compatibility with an existing IPSec policy or Windows 2000 and Windows XP behavior.
- A value of 1 specifies that Kerberos and RSVP traffic are not exempt from IPSec filtering, but multicast, broadcast, and ISAKMP traffic are exempt.

- A value of 2 specifies that multicast and broadcast traffic are not exempt from IPSec filtering, but RSVP, Kerberos, and ISAKMP traffic are exempt.
- A value of 3 specifies that only ISAKMP traffic is exempt from IPSec filtering. This is the default filtering behavior for Windows Server 2003.

The value 3 was used for `NoDefaultExempt` registry entry on the domain controller, because it offers the maximum protection, since some denial of service (DoS) attacks are using broadcast, multicast, RSVP and Kerberos. Windows Server 2003 provides improved DoS avoidance that can be affected through ISAKMP protocol [14].

4.7.2 Filter Actions

A filter action includes the type of action required, that is, permit, block, or negotiate security for packets that match the filter list. For the Negotiate Security filter action, the negotiation data contains one or more security methods that are used during IKE negotiations and other IPSec settings. Each security method determines the security protocol (such as AH or ESP), the specific cryptographic and hashing algorithms, and session key regeneration settings used. We created the following filter actions:

1. AH-MD5: a filter action with AH protocol only and using Message Digest 5 (MD5) as the integrity algorithm.
2. AH-SHA1: a filter action with AH protocol only and using Secure Hash Algorithm 1 (SHA1) as the integrity algorithm.
3. ESP-MD5&3DES: has ESP protocol only, using MD5 as the integrity algorithm and 3DES as the encryption algorithm.
4. ESP-SHA1&3DES: has ESP protocol only, using SHA1 as the integrity algorithm and 3DES as the encryption algorithm.
5. ESP-MD5: a filter action with ESP protocol only and using MD5 for data integrity without an encryption algorithm.
6. ESP-SHA1: a filter action with ESP protocol only and using SHA1 for data integrity without an encryption algorithm.

7. BOTH1: this filter action has both protocols applied with the following options; AH data integrity uses MD5 algorithm, ESP data integrity uses SHA1 algorithm and ESP data confidentiality uses 3DES.

8. BOTH2: this filter action has both protocols applied with the following options; AH data integrity uses SHA1 algorithm, ESP data integrity uses MD5 algorithm and ESP data confidentiality uses 3DES.

All filter actions were set to accept insecure communications but will always respond with secure communication.

4.7.3 Authentication Methods

Each IPSec security policy defines at least one authentication method. Each authentication method defines the requirements for how identities are verified in communications to which the associated policy applies. The two communicating computers must have at least one common authentication method or communication will fail. Creating multiple authentication methods increases the chance that a common method between two computers can be found.

Only one authentication method can be used between a pair of computers, regardless of how many are configured. If you have multiple rules that apply to the same pair of computers, it is necessary to configure the authentication methods list in those rules to enable the pair to use the same method. For example, if a rule between a pair of computers specifies only Kerberos for authentication and filters only TCP data and, in another rule, specifies only certificates for authentication and filters only UDP data, authentication will fail. One or more authentication methods are configured (in order of preference) and used for authentication of IPSec peers during main mode negotiations. Windows Server 2003 provides three authentication methods namely;

- Kerberos V5 protocol

  This rule applies to computers that are validated by a Windows 2000 or Windows Server 2003 Active Directory domain or a trusted Active Directory domain. This method was used for authentication because an Active Directory was used to apply IPSec on the domain members.

- Public Key Certificate

This method uses a certificate issued from a specified root certification authority (CA).

- Pre-shared Key

  This method allows you to specific a string that will be used for authentication.

4.7.4 Tunnel Setting

A tunnel end point specifies whether the traffic is tunneled and, if it is, the IP address of the tunnel endpoint. For outbound traffic, the tunnel endpoint is the IP address of the IPSec tunnel peer. For inbound traffic, the tunnel endpoint is a local IP address. The tunnel setting has the option of either using the tunnel or not. The tunnel endpoint option was not used, because the tests are based on a point-to-point IPSec implementation without any intermediate gateways. Therefore the transport mode was used which does not specify a tunnel.

4.7.5 Security Policies

Respective security policies were created for every filter action we defined and assigned these policies on the respective organizational units which contain the accounts of computers on the network on the domain controller. After running a set of tests the policy was changed until all experiments were conducted.

4.7.6 IPSec Algorithms

Windows IPSec implementation provides a choice of two hash functions and two encryption algorithms. MD5 and SHA1 are the available hash functions, while DES and 3DES are the available encryption algorithms.

4.8 IPSec weaknesses on Windows

Windows Server 2003 provides the latest support for IPv6 and for some of its security mechanisms on Windows platforms. However, it is important to be aware of the limitations of Windows Server 2003's implementation of IPv6 when it comes to security features. Microsoft's Help and Support documentation [14] state that the implementation of IPSec that comes with Server 2003 IPv6 is not recommended for production use. This is because it uses static keying and does not provide for

updating of keys when sequence numbers are reused, and thus does not provide the level of security necessary for mission-critical communications. IPSec implementation that comes with Windows Server 2003's IPv6 does not support ESP data encryption, which means it fails to provide for data confidentiality. However, the use of ESP with null encryption uses the ESP header, only data origin authentication and data integrity services are provided. Note that even though data encryption is not provided, a full ESP header is applied, which allows for the performance of one part of the ESP performance tests, that is, frame overhead. But time related tests such as round trip times will not be realistic, since lower processing times will be recorded due to the absence of a complete encryption process in IPv6 transmissions. These weaknesses only apply to IPSec in IPv6 but not in IPv4 protocol which does support data encryption.

CHAPTER FIVE: RESULTS AND ANALYSIS

This chapter gives a detailed analysis of the experiments that were conducted in this research. It will go into detail of what tests were done, how they were done and why they were done. It will also show the graphical analysis of all the results.

5.1 Tests Conducted

The tests that were carried out will be explained in detail in this section. There are two main sections of tests; firstly those done to determine the traffic overhead and those done to determine the delay using the round-trip times and download times in the case of HTTP, FTP and TFTP. ICMP tests were conducted on IPv4 protocol and IPv6 protocol and FTP, TFTP and HTTP tests on IPv4 only.

Windows IIS 6.0 for IPv6 has the following limitations on HTTP functionality, as a result we could not run HTTP and FTP tests on IPv6.

- Site routing is limited to host headers only. Sites cannot be configured to route on an IPv6 address or on a combination of an IPv6 address and host header. This limitation also affects sites that are configured to route based on an IPv4 address. When IPv6 installed, sites that are already specifically configured for IPv4 site-based routing do not respond to requests that come in over IPv6.
- File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP), and Network News Transfer Protocol (NNTP) services are not supported. IPv6 is supported only for the WWW service. The FTP, SMTP, and NNTP services do not have IPv6 support in IIS 6.0 [78].

5.1.1 Frame Overhead

Frame overhead are additional bits that are added to a digital signal at the sending end or intermediary gateways of a digital link. Frame overhead in this case is the frame size overhead. For instance, in the IPSec frame overhead, it is basically the additional bits added by IPSec headers to the original packet meant for transmission to provide security services. Snapshots of the captured packets are given in Appendix B. The frame overhead induced was determined by applying different IPSec

protocols and algorithms in our first set of experiments. This made it possible to determine the percentage increase in the frame size due IPSec implementation. This ultimately means that more processing time is needed in processing packets with IPSec enabled by network nodes. An increase in frame size also requires a larger buffer size to achieve the same quality of service (QoS) before the overhead.

Equation 5-1 gives the percentage overhead for a given packet size, in our case the packet sizes are mentioned in every section that discusses a particular test. In this equation, x is the frame size with IPSec enabled and y is the frame size with IPSec disabled.

$$\text{Percentage Frame Size Overhead} = 100\left(\frac{x-y}{y}\right) \quad (5\text{-}1)$$

5.1.2 Round Trip Time

Round Trip Time (RTT) is a measure of the time it takes for a packet to travel from a computer, across a network to another computer, and back. RTT is computed by the sending side recording the clock when it transmits a packet, and then records the clock again when an acknowledgment or a reply arrives. Subtracting the two values, yields a single estimate of the round trip time. A collection of these values for a set period of time give the average RTT. RTT shows the IP network total path delay that can be caused by many factors such as; packet capturing delay, queuing delay, switching/routing delay and light-speed propagation delay.

Packet capture delay is the time required to receive the entire packet before processing and forwarding it through the router. This delay is determined by the packet length and transmission speed. Using short packets over high-speed trunks can easily shorten the delay but potentially decreases network efficiency.

Latency is normally composed of queue delay, path delay and transmission delay. Path delay is the delay in transmitting frames between end nodes.

Queue delay is delay required at the input and output ports of a packet switch due to the statistical multiplexing nature of IP networks and to the asynchronous nature of packet arrivals. This delay is a function of the traffic load on a packet switch, the

length of the packets, and the statistical distribution over the ports. Designing very large router and link capacities can reduce but not completely eliminate this delay.

Switching/routing delay is the time the router takes to switch the packet. This time is needed to analyze the packet header, check the routing table, and route the packet to the output port. This delay depends on the architecture of the route engine and the size of the routing table. Hence additional headers increase this type of delay. New IP switches can significantly speed up the routing process by making routing decisions and forwarding the traffic via hardware as opposed to software processing [70].

Propagation delay is the time required for a digital signal to travel from one point to another, usually from source node to the destination node in a computer network. It largely depends on the underlying network topology. Increased RTT induced by additional overhead due to more processing time and larger frame sizes affects the QoS of the network, especially in real-time applications such as video conferencing and voice over IP (VoIP).

Transmission delay is the time it takes for the TCP Layer to transmit the segment onto the communications link. This is determined purely by the TCP segment's size and the bandwidth of the link.

Let t_i be the return trip time of trip i and n be the number of trips recorded, 50 in the experiments, therefore the average RTT for a particular packet size is computed using equation 5-2.

$$\text{Average RTT} = \frac{1}{n}\sum_{i=1}^{n} t_i \qquad (5\text{-}2)$$

5.1.3 Download Time

The download time is the time it takes to download a file or a webpage from the remote server or computer to the local computer. We calculated this by subtracting the time the first frame of the downloaded file is received from the server and the time the last frame is received.

Let t_s be the time the request for a web page is sent from the client computer to the web server, and t_r be the time the last fragment is received on the client from the web server. This process is performed 50 or more times, which is represented by n, therefore the average of the difference of the two times is recorded as the download time for that particular web page size. Equation 5-2 shows the formula we used to calculate download times

$$\text{Download time} = \frac{1}{n}\sum_{i=1}^{n}(t_{r_i} - t_{s_i}) \tag{5-3}$$

5.2 Applications/Protocols Tested

5.2.1 ICMP

Internet Control Message Protocol (ICMP), documented in RFC 792 [64], is a required protocol tightly integrated with IP. ICMP messages, delivered in IP packets, are used for out-of-band messages related to network operation. ICMP uses IP as if ICMP were a higher-level protocol, that is, ICMP messages are encapsulated in IP datagrams. However, ICMP is an integral part of IP and must be implemented by every IP module. This protocol was chosen for its simplicity and because it is the most used protocol in a network for delivering network maintenance messages and error messages. ICMP is also not dependent on either TCP or UDP.

5.2.1.1 PING

The Ping is an application that tests host responses over a network connection. Ping uses the network layer to send packets to a remote address. If there are network connectivity problems or the host has problems, the ping will fail, indicating a problem exists. Additional tests may be needed at that point to determine the cause of the problem.

The Ping is an application on OSI layer 7 designed to use echo request and echo reply to calculate and display round trip times for each request/reply pair to a host over a network. Ping utilizes raw ICMP sockets to send packets to a remote host, calculates the difference in time between transmission and receipt of the packet and

produces printed, human-readable results of these measurements in milliseconds for each request/reply pair. Any delay anywhere in the chain will create what is referred to as propagation delay or more simply latency.

5.2.2 HTTP

The HyperText Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems used to transfer data across the Internet. HTTP has been in use by the World-Wide Web global information initiative since 1990. The current version of HTTP is HTTP/1.1. The HTTP protocol is a request/response protocol. A client sends a request to the server in the form of a request method, Uniform Resource Locator (URL) and protocol version, followed by a Multipurpose Internet Mail Extensions (MIME) like message containing request modifiers, client information, and possible body content over a connection with a server. The server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta information, and possible entity-body content [65]. MIME is a specification for formatting non-ASCII messages so that they can be sent over the Internet, in addition, MIME supports messages in character sets other than ASCII. We used HTTP to test how IPSec performs on TCP, since most of these files may require to be protected using IPSec.

Currently HTTP secures its packets using HTTPS, through Socket Secure Layer (SSL). But the difference between SSL and IPSec is that, higher layer security services, in this case SSL, protect a single protocol, while IPSec on network layer protects all protocols provided they are specified on configuration. IPSec can provide some security services in the background with no impact on user. SSL develops something at user level without changing underlying operating system. But layer 3 means the goal of the IPSec is to develop something within the operating system that does not require changes to the application so it can provide security to diverse range of application protocols.

Transport Layer Security (TLS) is a standard protocol that is used to provide secure Web communications on the Internet or intranets. It enables clients to authenticate servers or, optionally, servers to authenticate clients. It also provides a secure

channel by encrypting communications. TLS is the latest and a more secure version of the SSL protocol.

5.2.3 FTP

File Transfer Protocol (FTP) is a procedure used to upload and download files to and from your FTP server. FTP is a special way to login to another Internet site for the purposes of retrieving and/or sending files and is the best way of sending files from one computer to another over the Internet using TCP. The objectives of FTP are; to promote sharing of files, to encourage indirect or implicit use of remote computers, to shield a user from variations in file storage systems among hosts and to transfer data reliably and efficiently.

FTP requires two programs, a server program, and a client program. The client program connects to an FTP sever on the Internet, after the user has provided authentication information or after logging in anonymously if the server is configured to allow anonymous logins. Once connected, the FTP server sends a welcome message to the client over the open network connection, usually on TCP port 21 by default. The user then enters commands to upload or download files [66]. Securing FTP connection is commonly done using user login/authentication information.

5.2.4 TFTP

Trivial File Transfer Protocol (TFTP) is a simple UDP-based protocol used for transferring files between computers. TFTP is used where user authentication and the need to view directories on remote computers are not required. Its two major uses are to bootstrap diskless machines that are being installed over the network and to install images that reside in firmware. This type of communication also might need to be protected to improve network security to avoid intruders.

The TFTP file interface is very simple, providing no access control or security. It provides its own reliable delivery by using a simple stop-and-wait acknowledgment system. In this system, the file transfer is delayed until an acknowledgment is received from the destination computer. Any transfer begins with a request to read or write a file, which also serves to request a connection, usually on UDP port 69 by

default. The request consists of a single packet sent if the server grants the request, the connection is opened and the file is sent in fixed length blocks of 512 bytes. The TFTP header consists of a 2 byte opcode field and the block number. Opcode field indicates whether the packets is a read request, write request, error, acknowledgement or data packet and block number identifies a particular block [63]. A packet of less than 512 bytes signals termination of a transfer and it has file size limit of 32 MB.

5.3 Frame Structure

5.3.1 IPv4

The Maximum Transmission Unit (MTU) is the largest size of an IP datagram which may be transferred using a specific data link connection. The minimum size of an IPv4 Ethernet frame is 64 bytes, which includes the minimum frame payload of 46 bytes plus the Ethernet header of 18 bytes including 4 bytes for CRC. The maximum frame size for IPv4 Ethernet is 1518 bytes, given by an Ethernet MTU of 1500 bytes plus the Ethernet header. Therefore if the frame payload is less than 46 bytes, padding bytes will be added and if the frame payload is more than 1500 the packet will be fragmented. The largest IPv4 payload is 65535 bytes, which means an IP packet of 65535 bytes will be fragmented into smaller packets of 1500 bytes, generally in 8-byte chunks for them to be transmitted across an IPv4 network. And an IP packet of more than 65535 bytes will not be transmitted due to lack of fragment identification. An IPv4 fragment of 1518 bytes is made up of the following components:

- 14 bytes – Typical Ethernet header
- 20 bytes – Typical IP header
- 1480 bytes – IP payload
- 4 bytes – CRC

Table 5-1: IPv4 packet headers order

Ethernet header	IPv4 header	IP payload	Ethernet Trail
14	20	variable	4

Table 5-1 shows the order of headers in an IPv4 packet and the IP payload may include any other headers such as the transport protocol being used by the packet and the application protocols headers.

5.3.2 IPv6

An IPv4 Ethernet frame using 6to4 has a minimum size of 86 bytes, comprised of 68 bytes of frame payload plus 18 bytes of Ethernet header including 4 bytes of CRC. It was noticed that fundamental differences occurred on how IPSec affects IPv4 and IPv6, as the IPv6 results will show in this section. The default MTU size for IPv6 packets on an Ethernet is 1280 octets [4]. Therefore a fragmented packet has a frame size of 1318 bytes for each fragment except the last fragment that have different amount of payload data. An IPv6 fragment frame of 1318 bytes is made up of the following:

- 14 bytes – Ethernet header and Trail
- 20 bytes – IPv4 header
- 40 bytes – IPv6 header
- 8 bytes – Fragment header
- 1232 – Payload data
- 4 bytes – CRC

Table 5-2: Generic order of headers in an IPv6 packet

Ethernet header	IPv4 header	IPv6 header	Extension headers	Data	Ethernet Trail
14	20	40	variable	variable	4

Table 5-2 shows the order of headers in an IPv6 packet and the order of extension headers if more than one exists is documented in IPv6 specification RFC 2460. The fragment header in IPv6 is an extension header that replaces the fragment offset in IPv4, which identifies each fragment when packets are fragmented. In IPv6 packets

are fragmented at sources only, hence nodes need to learn the MTU before transmitting packets, since intermediary nodes must not fragment any transmission. The fragment header size, like all other extension headers in IPv6 is added on top of the 40 bytes minimum size of an IPv6 header.

5.4 IPSec Transform Sets

An IPSec transform specifies a single IPSec security protocol (either AH or ESP) with its corresponding security algorithms and mode. There are two IPSec modes (tunnel and transport) and each is suitable for a particular network setup. Tunnel mode is most commonly used between gateways, or at an end-station to a gateway, the gateway acting as a proxy for the hosts behind it. Transport mode is used between end-stations or between an end-station and a gateway, if the gateway is being treated as a host. Since the tests are based on a single network, that is, point to point communication within a site, it was decided to use the transport mode for the tests. Therefore all IPSec transform sets are based on transport mode IPSec that protects the IP payload only with the IP header visible. A transform set is a combination of individual IPSec transforms designed to enact a specific security policy for protecting a particular traffic flow. Table 1 shows the available IPSec transform sets in Windows Server 2003.

Windows IPSec implementation provides two integrity algorithms: SHA1 and MD5 and two encryption algorithms: 3DES and DES as explained in the previous chapter. Hence all our IPSec transform sets are based on these four algorithms and in transport mode. There are 24 possible IPSec transform sets in Windows IPSec implementation all of which were tested on IPv4 and IPv6.

Table 5-3: IPSec transform sets in Windows Server 2003

Transform Set	AH algorithm	ESP Algorithm
AH Only		
1	AH-MD5	None
2	AH-SHA1	None
ESP Only		
3	None	ESP-SHA1
4	None	ESP-MD5
5	None	ESP-SHA1-3DES
6	None	ESP-MD5-3DES
7	None	ESP-SHA1-DES
8	None	ESP-MD5-DES
AH and ESP Encryption		
9	AH-MD5	ESP-DES
10	AH-MD5	ESP-DES
11	AH-SHA1	ESP-DES
12	AH-SHA1	ESP-DES
AH and ESP Integrity		
13	AH-MD5	ESP-SHA1
14	AH-MD5	ESP-MD5
15	AH-SHA1	ESP-SHA1
16	AH-SHA1	ESP-MD5
BOTH		
17	AH-MD5	ESP-SHA1-3DES
18	AH-MD5	ESP-MD5-3DES
19	AH-SHA1	ESP-SHA1-3DES
20	AH-SHA1	ESP-MD5-3DES
21	AH-MD5	ESP-SHA1-DES
22	AH-MD5	ESP-MD5-DES
23	AH-SHA1	ESP-SHA1-DES
24	AH-SHA1	ESP-MD5-DES

There are five main categories of transform sets as shown in Table 5-3:

- AH Only

  This transform set specifies an AH protocol algorithm only with none for ESP protocol.

- ESP Only

  This transform set specifies an ESP protocol algorithm(s) only with none for AH protocol.

- AH and ESP Encryption

  This transform set specifies the AH protocol algorithm and an ESP encryption algorithm without ESP integrity algorithm.

- AH and ESP Integrity

  This transform set specifies the AH protocol algorithm and an ESP integrity algorithm without ESP encryption algorithm.

- BOTH

  This transform set specifies the AH protocol algorithm and both ESP encryption and integrity algorithms. Therefore this is a full implementation of IPSec and both refer to both protocols applied fully in an IPSec transform set.

It was discovered that in some cases different sets give the same results.

5.5 Test Considerations

The tests were done on both IP protocols; IPv4 and IPv6. For all tests plain benchmarking tests were used, that is, performance tests were performed without enabling IPSec and these results were used as a baseline for comparing with IPSec tests results as outlined in [58]. IPv6 tests were carried out using 6to4 because it is applicable on the global Internet and intra-site networks, while ISATAP is mainly an intra-site transition mechanism. However preliminary experiments were conducted using ISATAP and it became evident that, it has the same frame overhead impact as 6to4. Each test was run twice consecutively for a period of 50 seconds and the second run was considered for both overhead tests and RTT in the case of ICMP and download times for HTTP, FTP and TFTP.

5.6 Overhead on ICMP

The first protocol that was measured with IPSec frame overhead was ICMP using the ping application, on the following sizes of IP payload; 1 byte, 8 bytes, 32 bytes, 128

bytes, 512 bytes, 2048 bytes, 8192 bytes, 32768 bytes and 65500 bytes, of which 1 to 512 bytes are non-fragmented packets and the rest are fragmented. The ping program was started on one client sending to the other client, while capturing the communication using the protocol analyzer on the sending computer. This procedure was conducted with IPSec off first and IPSec on, using different transform sets on IPv4 and IPv6.

5.6.1 IPSec Overhead Tests on IPv4

The observations made after capturing packets of different sizes, is that, AH applied on it's own adds an additional 24 bytes to each packet sent on IPv4, even if the packet is fragmented only 24 bytes are added on the first fragment. These 24 bytes are made of the following sections of the AH header, as captured by our protocol analyzer.

- 1 byte – Next Header
- 1 byte – Payload Length
- 2 bytes – Reserved
- 4 bytes – Security Parameters Index
- 16 bytes – Authentication Data

Table 5-4 shows the structure of the frame protected by AH protocol in transport mode, the sizes are in bytes and Table 5-5 shows the AH header sections.

Table 5-4: Ethernet frame protected by AH in transport mode

Ethernet header	IPv4 header	AH header	Data	Ethernet Trail
14	20	24	variable	4

Table 5-5: AH header format

Next header	Payload length	Reserved	SPI	Authentication data
1	1	2	4	16

The additional bytes due to IPSec headers are the same for all packet sizes, fragmented and non-fragmented. The overhead is the same irrespective of the hash function used, that is, MD5 or SHA1.

'ESP Only' transform sets add an overhead of 36 bytes using both properties (integrity and encryption) of ESP. This figure is divided into the following sections as shown in Table 5-6 and Table 5-7:

- Security Association Identifier – 4 bytes
- Sequence Number – 4 bytes
- Opaque Transform Data (28 bytes) – excluding the initial packet size

The overhead for applying ESP integrity only is 28 bytes, made up of:

- Security Association Identifier – 4 bytes
- Sequence Number – 4 bytes
- Opaque Transform Data (20 bytes) – excluding the initial packet size

Table 5-6: Ethernet frame protected by ESP in transport mode

Ethernet header	IPv4 header	ESP header	Data	Ethernet Trail
14	20	28 or 36	variable	4

Table 5-7: ESP packet format

Security Association Identifier	Sequence Number	Opaque Transform Data
4	4	20 or 28

Note that the Security Association Identifier and the Sequence Number are displayed as clear data. The additional bytes added by IPSec headers using different IPSec transforms sets are shown in Table 5-8.

Table 5-8: IPSec frame overhead bytes

Transform Set Description	Additional Bytes
AH only	24
ESP Integrity only	28
ESP Integrity and Encryption	36
AH and ESP Integrity	48
AH and ESP Encryption	52
AH and ESP	60

The overhead for using both IPSec protocols is the summation of the overhead added by each protocol. For instance, the overhead of AH and ESP fully implemented is 60 bytes (24 AH and 36 ESP).

Figure 5-1 shows the impact of each of the two security protocols enabled separately and both fully implemented. It compares the percentage of the overhead discussed above on the original packet transmitted with three different IPSec transform sets namely; AH only, ESP only and AH and ESP both applied.

The overhead starts low on packet sizes of 1 byte, because some overhead bytes are used as padding bytes, which reduces the overall overhead of IPSec headers on the packets that need padding. All packets of less than 18 bytes require padding for them to be transmitted over the network and they all have a frame size of 64 bytes. That means the percentage of IPSec overhead increases from 1 byte and reaches the maximum on the 18 bytes packet. Then it starts to fall on packet sizes of more than 18 bytes. By comparison the overhead of 18 bytes packets using the transform sets shown in Figure 1 is as follows; AH only gives 38%, ESP only gives 56% and both protocols give 94%. This shows that there is quite a significant increase in overhead when using ESP only compared to AH only on IPv4 on small packets.

The overhead of using both protocols on 18 bytes packets is 94%, which is the aggregate of AH and ESP overheads. The overhead falls sharply as packet size increases to the level of about 0.04% for packets of 65000 bytes, because IPv4 IPSec headers are applied on the first fragment only. Hence the additional overhead bytes are constant and they are spread over an increasing common denominator, that is, the cumulative size of all fragments. This trend applies to all IPSec transform sets.

Understandably AH only has a lower overhead as compared to ESP only, which in turn has a lower overhead compared to both protocols applied. Applying ESP only instead of AH only increases the overhead by 7.64% on average across all the packet sizes we experimented with. While using both protocols instead of ESP only increases the overhead by 15.67% on the average and using both protocols instead of AH only increases the overhead by 23.31%.

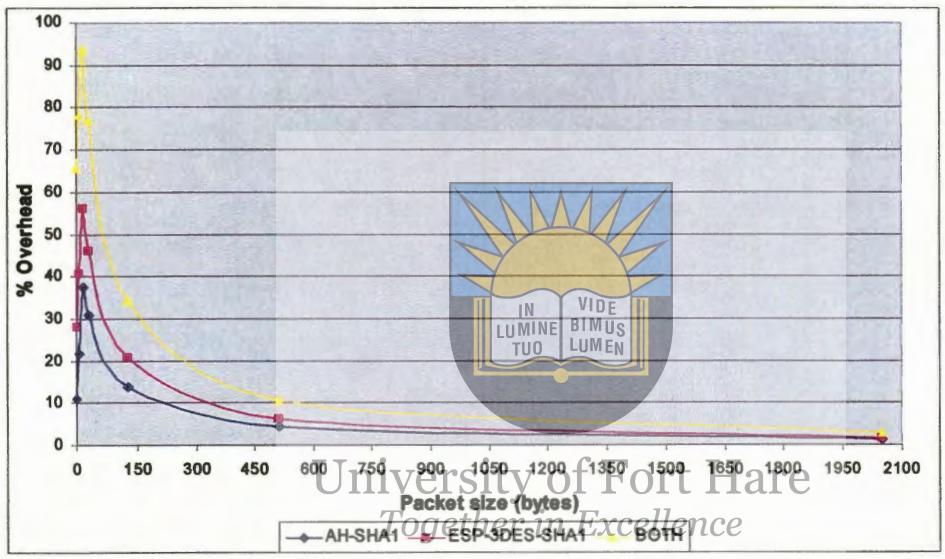


Figure 5-1: Impact of AH, ESP and BOTH IPSec transform sets on IPv4 overhead

The use of ESP protocol only in an IPSec transform set gives two options of IPSec transform sets, namely; ESP with integrity only and a complete ESP that has both integrity and encryption. Figure 5-2 shows the impact of these ESP transform sets.

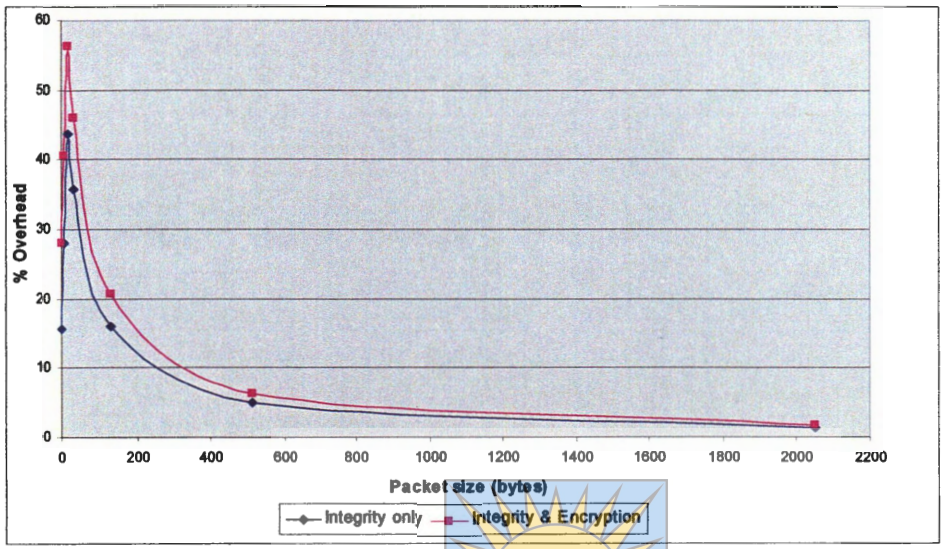


Figure 5-2: Impact of ESP transform sets on IPv4 overhead

ESP integrity using either MD5 or SHA1 algorithms results in the same overhead impact. The same applies for a complete ESP implementation that has integrity and encryption, irrespective of the algorithms used the overhead is the same on ICMP packets. ESP integrity only overhead is relatively lower than that of ESP with integrity and encryption.

Applying ESP integrity and encryption increases the overhead by almost 12% compared to ESP integrity only on the smallest packet size that does not need padding, 18 bytes. On the average the overhead increases by 5% from packet sizes of 1 byte to 65500 bytes. 65535 bytes was used in the experiments because, this is the maximum IP packet that can be sent using the ping application. For all different IPSec transform sets on IPv4 the impact of the overhead relatively is the same on very large packets, for example, in the experiments done the impact of the overhead is almost constant at 0.04% across all different transform sets with a packet size of 65500 bytes.

The avoidance of applying IPSec on small packets is recommended based on the findings of the experiments.

5.6.2 IPSec Overhead on IPv6

IPSec overhead on IPv6 adds the same number of bytes (for IPSec headers) as IPv4 on all IPSec transform sets as given in Table 5-3. Figure 5-3 shows the percentage impact of overhead on ICMP packets of different sizes. As was the case on IPv4, AH only, has a lower overhead as compared to ESP only, which in turn has a lower overhead compared to when both protocols are applied. Unlike IPv4, there is no increase in the overhead on very small packets, because no padding bits are necessary when using IPv6, since the minimum frame payload of IPv6 Ethernet exceeds the IPv4 Ethernet minimum payload.

The impact of the overhead is higher from the first packet size of 1 byte and it gradually decreases as the packet size increases. The first significant difference of how IPSec affects IPv6 as compared to IPv4 is that, IPSec headers are applied on all fragments, while on IPv4 they were applied on the first fragment only. This causes the overhead to have a more significant impact on IPv6. On fragmented packets the IPSec protocol headers are applied on every fragment on IPv6 protocol, hence the percentage overhead does not decrease sharply as it does on IPv4. Consequently, the overhead on large packets on IPv6 is higher than that on IPv4.

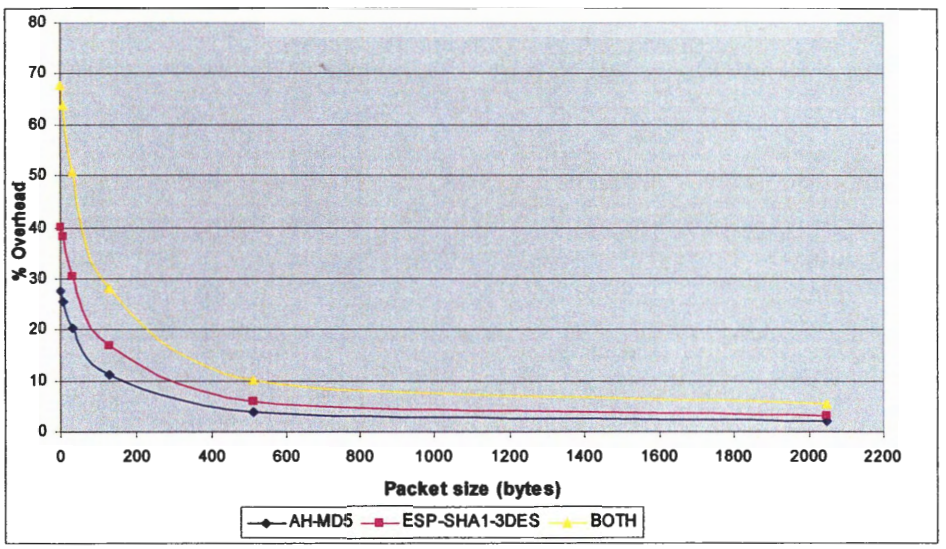


Figure 5-3: Impact of AH, ESP and BOTH IPSec transform sets on IPv6 overhead

Secondly, unlike the constant overhead percentage on packets of 65500 bytes on IPv4 using different transform sets; IPv6 has a constant overhead for each transform set from 32768 bytes and larger. For example, AH only has 1.85%, ESP only has 2.77% and both protocols together have 2.62%. The largest IP payload allowed was used on ICMP protocol of 65500 bytes since 65535 bytes is not accepted on Windows. Applying ESP only instead of AH only increases the overhead by 5.77% on the average across all the packet sizes. While using both protocols instead of ESP only increases the overhead by 11.82% on average and using both protocols instead of AH only increases the overhead by 17.59%.

Figure 5-4 shows the impact of two ESP only implementations of IPSec; ESP with integrity only and ESP with both integrity and encryption. A non-significant difference of at least 1% in the two settings can be noticed on the last point (on 2048 bytes), which is relatively higher than the difference in IPv4 protocol of 0.3% shown in Figure 5-2 due to the above mentioned fact that IPSec headers applied on every fragment in IPv6.

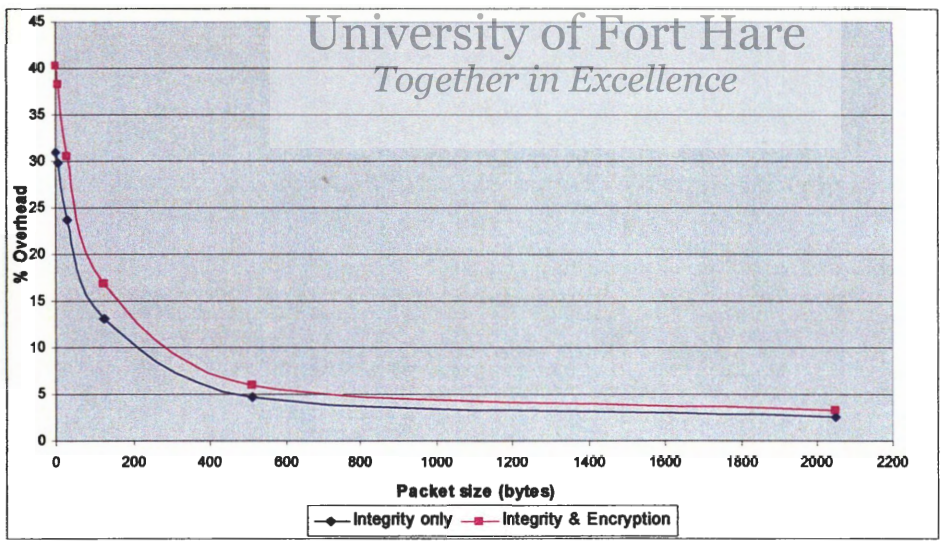


Figure 5-4: Impact of ESP transform sets on IPv6 overhead

A comparison was also done on how the IPSec overhead impacts on IPv4 and IPv6, using the default three transform sets; AH only, ESP only and both protocols. Figure 5-5 illustrates this comparison.

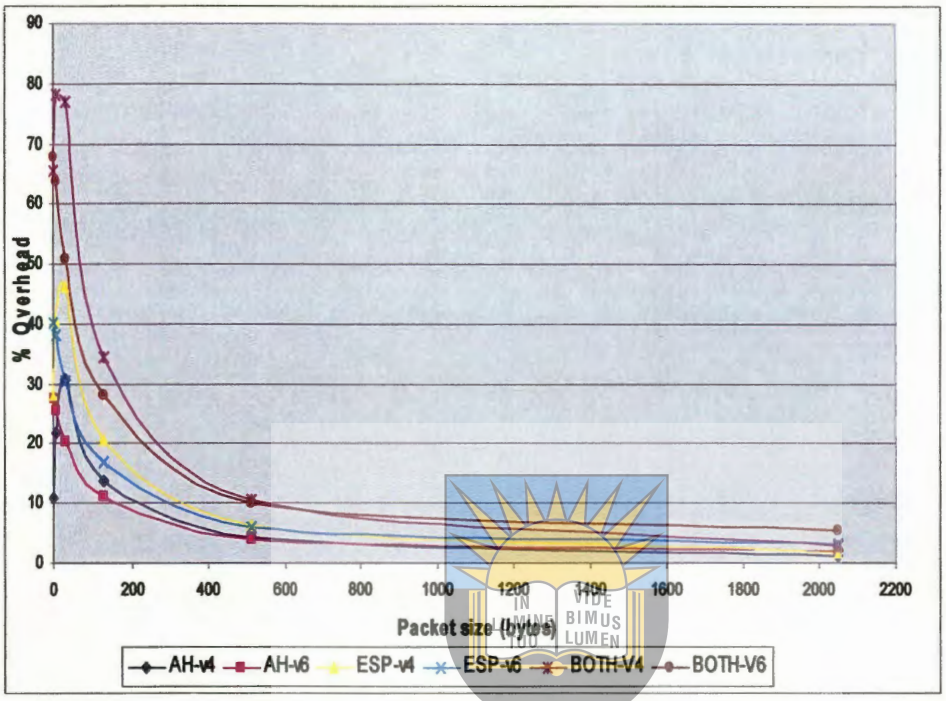


Figure 5-5: Comparison of IPSec transform sets on IPv4 and IPv6 overhead

The comparison of IPSec overhead impact on IPv4 and IPv6 gave the trend shown in Figure 5-5. The overhead has a higher impact on IPv4 as compared to IPv6 for smaller packets that are not fragmented. Comparing IPv4 and IPv6 average values shows that IPSec has a generally higher overhead on IPv4 than IPv6. This can be attributed to the fact that IPSec headers' sizes are the same on both IP protocols, while the frame sizes are different on these IP protocols for the same packet sizes. IPv6 fragmented packets have a higher overhead as compared to IPv4, because in IPv4 the IPSec header is applied on the first fragment only, while in IPv6 it is applied on each fragment.

It was observed that, the maximum frame size of 1518 bytes is never exceeded in IPv4 even after turning IPSec on. The additional overhead of using IPSec headers will occupy part of the payload data, hence reducing the payload of each fragment on fragmented IPv4 packets, this differs from how fragmented packets are handled in IPv6. In IPv6 the same payload data size is maintained and the additional bytes added by IPSec headers increases the frame size directly.

IPSec headers sizes are constant for both IPv4 and IPv6, but the way these headers are applied is different. Therefore their effects are also different on both protocols. For instance, on IPv4 the overhead is applied on the first fragment only when a packet needs fragmentation. This obviously means the burden of the overhead is higher on IPv6 than it is on IPv4. On both IP protocols, there is no difference in the frame size when using either SHA1 or MD5 for data integrity and DES or 3DES for data confidentiality.

5.7 Round Trip Time on ICMP

From the RTT values recorded when testing IPSec overhead, the average RTT was computed for each packet size using all the IPSec transform sets defined in Table 5-3. Figure 5-6 shows the average RTT of ICMP messages on IPv4; the comparison includes the RTT of packets with IPSec disabled. Firstly, for all transform sets the RTT increases proportionally as packet size increases, this is because a larger packet obviously needs more time to reach it's destination across the network and come back to the source and possibly requires more time to implement IPSec process. The graph shows that there is almost no difference between the RTT of transmissions with IPSec disabled and IPSec with AH only. The latter offers security advantage over a non-protected connection.

There is also no significant difference between the RTT of IPSec with ESP only and IPSec with both headers implemented, especially on smaller packets, but the difference becomes slightly significant on very large packets. As a matter of fact the average RTT of ESP only and both protocols is almost twice the average RTT of IPSec disabled and AH only especially on large packets. Generally this also gives a security advantage to using both protocols instead of ESP only. But the huge difference of about 120% between IPSec with both protocols and IPSec disabled on 65500 bytes packet size, suggests that a longer processing time is required when an encryption algorithm is used in an IPSec transform set, which ultimately decreases the throughput of a communication channel.

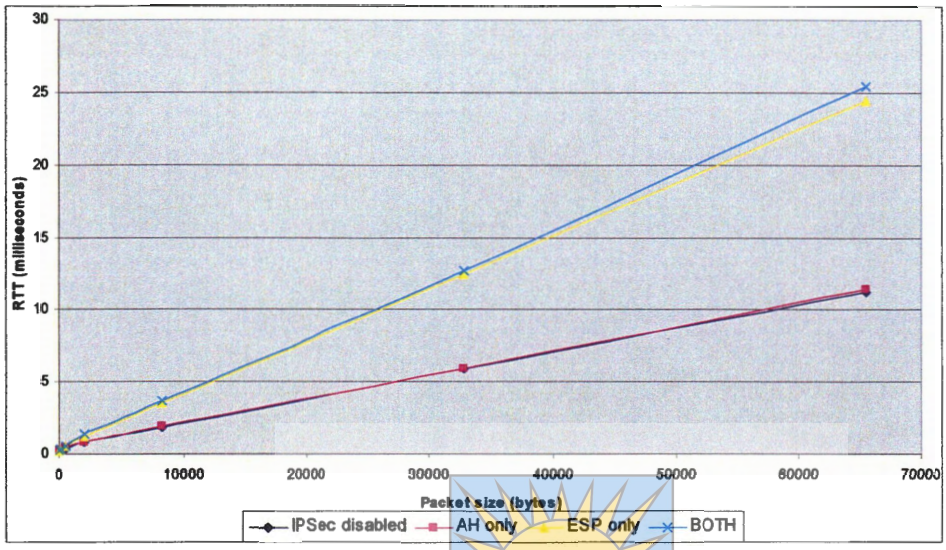


Figure 5-6: Impact of IPSec on ICMP Average RTT on IPv4

The average RTTs for IPv6 shows a different picture from those of IPv4, as shown in Figure 5-7. There is a gradual increase in RTT from transmission without IPSec to AH only, ESP only and finally with both headers applied. The trend portrayed in IPv4 of a huge difference between transform sets with an encryption algorithm and those without an encryption algorithm does not apply in IPv6. This is because the Microsoft Server 2003 IPSec has ESP with null encryption in IPv6 while IPv4 ESP has fully functional encryption process. Even though this is the case ESP only and both IPSec protocols has a longer RTT as compared to AH only and IPSec disabled transmissions of about 20%. It would be interesting to investigate the impact of Windows Server 2003 IPSec implementation with full encryption.

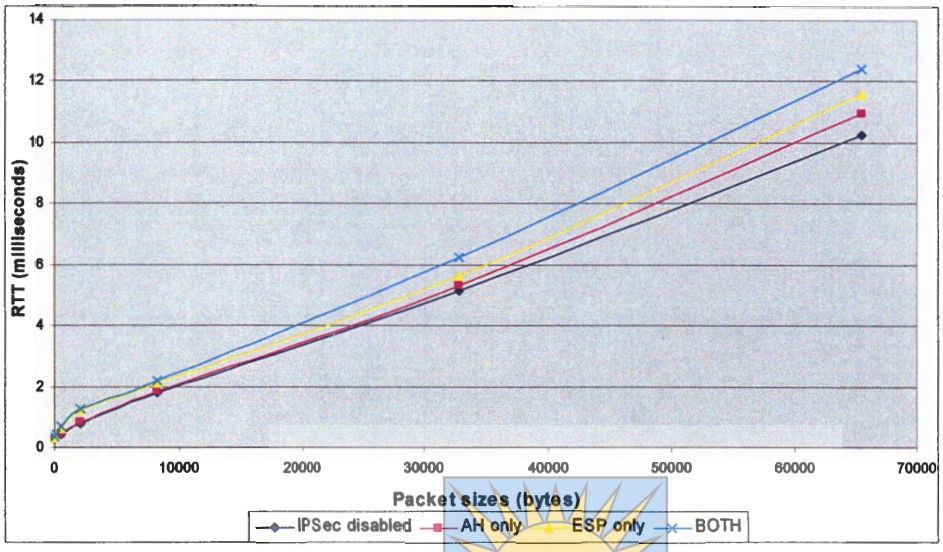


Figure 5-7: Impact of IPSec on ICMP Average RTT on IPv6

5.8 Frame-size Overhead on HTTP

Frame overhead tests of IPSec on HTTP were conducted, using different webpage sizes that contain only text. The webpage sizes considered are; 1 KB, 5 KB, 20 KB, 60 KB and 100KB. This experiment was conducted by capturing the HTTP requests on the client machine that was requesting webpages from the web server. HTTP packets use TCP as the transport protocol and IP as the network layer protocol. Packets that are larger than 1500 bytes are fragmented. The first fragment which has the HTTP header is packed in a typical maximum Ethernet frame of 1518 bytes made up of:

- 14 bytes – Typical Ethernet header
- 20 bytes – Typical IP header
- 20 bytes – Typical TCP header
- 228 bytes – HTTP header
- 1232 bytes – Data after TCP header (the payload)
- 4 bytes – CRC

For the rest of other fragments the HTTP header size is added to the payload.

AH only adds an overhead of 24 bytes, ESP only adds 36 bytes and with both protocols applied the overhead is cumulative as was the case in ICMP. The first comparison shows the percentage overhead of the three main security policies; AH only, ESP only and both applied in Figure 5-8.

The graph shows the impact of IPSec using the following transform sets; AH with MD5 or SHA1, followed by ESP with both integrity and encryption and lastly both protocols applied. The overhead of AH is lower than that of ESP and both protocols, because AH uses only one cryptographic algorithm to provide it's security services while ESP uses two, one for integrity and the other for encryption.

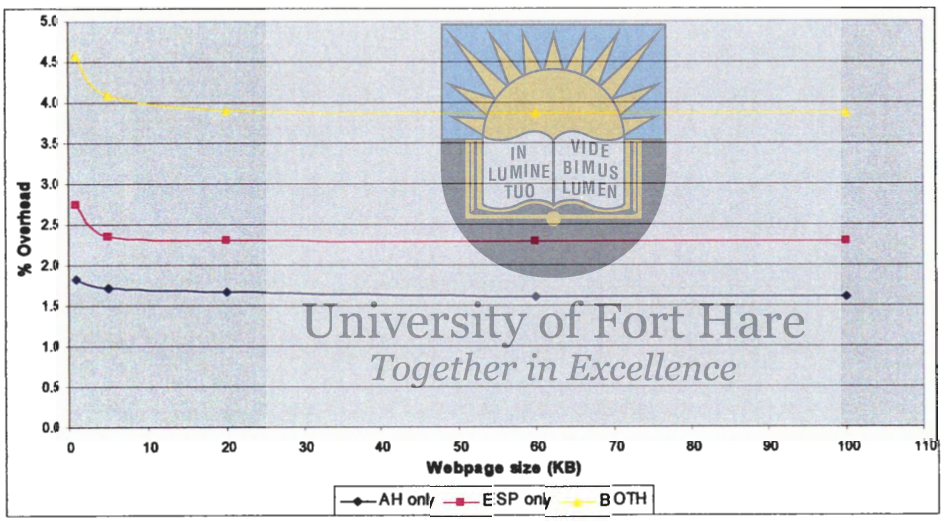


Figure 5-8: Impact of IPSec on HTTP overhead

The overhead is also slightly higher on smaller webpages as compared to larger pages. For AH the overhead starts higher for a webpage of 1 KB and slightly drops for 5 KB webpages but for ESP and for both protocols it drops sharply because they add a larger overhead than AH. On the same note ESP's overhead is lower than that of both protocols applied together.

On the average, the overhead increases by almost 1 percentage point when ESP is used instead of AH and it further increases on the average by at least 1.5 percentage points. The overhead increases by a margin of 2.5 percentage points when both protocols are used instead of AH only. Given the fact that, the overhead for using AH

only is 1.6% and that for using both protocols is 3.3%, it follows that there is a security advantage to using both protocols in HTTP compared to any other IPSec transform set.

The overhead starts higher on smaller packets but it drops as packet size increases and becomes almost constant. This is because IPSec protects every fragment on fragmented packets, therefore the same overhead is spread evenly on all the fragments and on non-fragmented packets the impact is relatively higher because non-fragmented packets have frame sizes smaller than the frame size of a fragment.

Comparing the two ESP only transform sets' implementations of IPSec gives the picture in Figure 5-9.

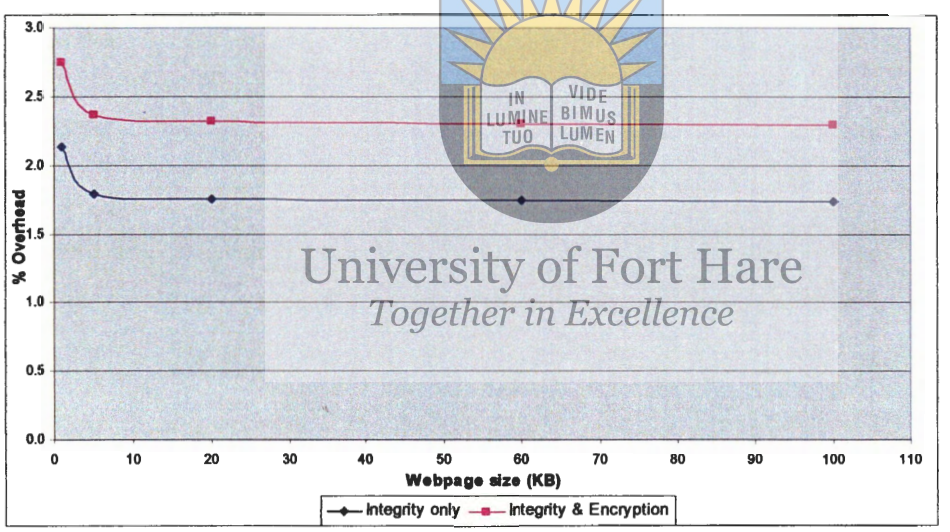


Figure 5-9: Impact of ESP transform sets on HTTP overhead

Figure 5-9 shows that there is small difference in the IPSec overhead when using the two ESP only transform sets on HTTP. ESP with integrity and encryption induces an additional 0.6% on the overhead as compared to ESP with integrity only.

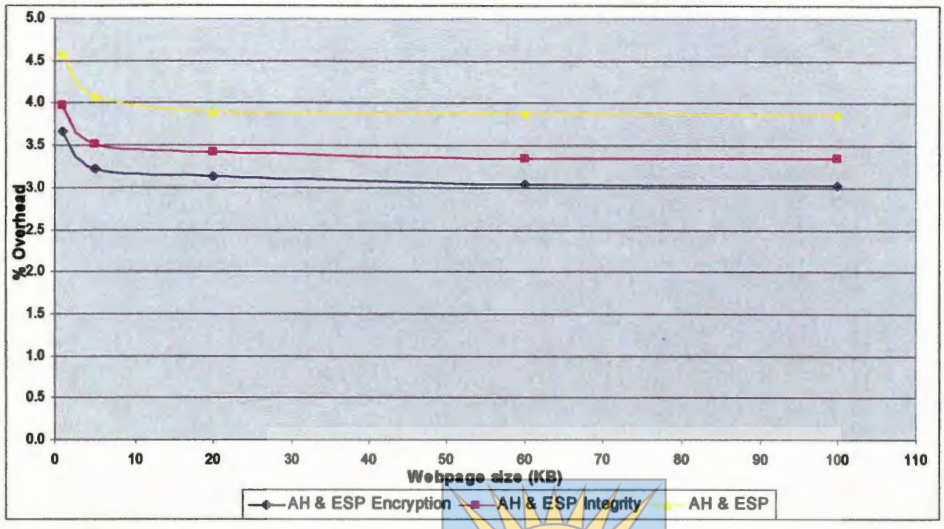


Figure 5-10: Impact of IPSec transform sets using on HTTP overhead

Figure 5-10 compares the impact of IPSec transform sets in the category of a full implementation of both protocols. There are three different implementations namely, AH and ESP encryption, AH and ESP integrity, and AH and ESP integrity and encryption. The graph shows that, there is no huge differences in the overhead when these transform sets are applied separately. The overall overhead increases by almost 1 percentage point from the least expensive transform set to the most expensive set.

5.9 Download Time on HTTP

A comparison was made of the time it takes to download web pages of different sizes, first with IPSec disabled then with different IPSec transform sets. The webpage sizes used are as follows; 1 KB, 5 KB, 20 KB, 60 KB and 100 KB. This test was conducted by requesting these web pages from the web server and capturing the packets between the server and the requesting client. The protocol analyzer was running on the client. The time the first frame is received and the time the last frame is received and the difference between the two was recorded was taken as the download time. Even though the times were not uniform the trend for different IPSec transform sets was relatively clear.

There was a slight difference between the download times of AH only transform sets using MD5 and SHA1. The average download time of all the webpage files tested

shows that SHA1 has more average download time than MD5 by 1 second. Figure 5-11 shows the download times for different IPSec transform sets, with different protocols. The AH only transform set uses SHA1 algorithm, the ESP only transform set uses 3DES and SHA1 and both protocols transform set uses MD5 for AH and 3DES and SHA1 for ESP. These transform sets are compared against the download times of web pages with IPSec disabled.

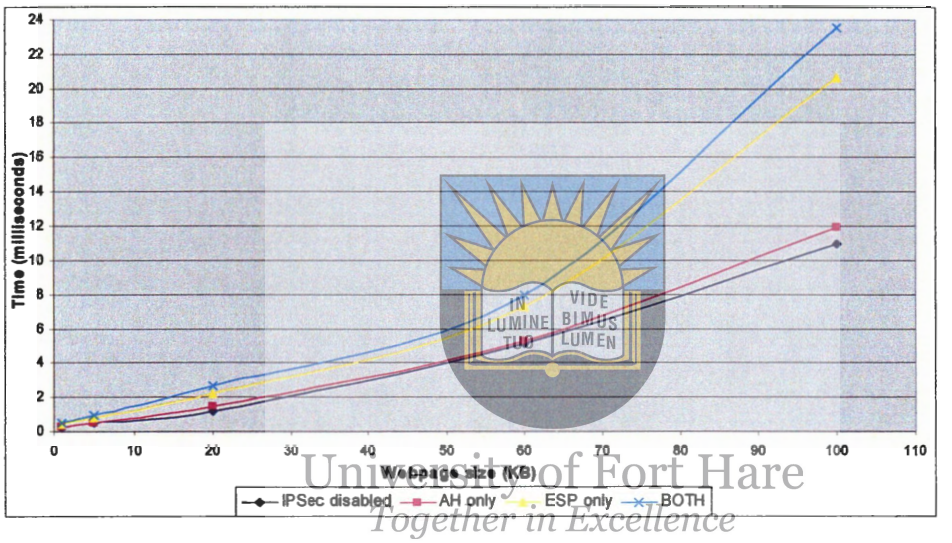


Figure 5-11: Impact of different IPSec protocols on HTTP download times

The download time increases exponentially from IPSec without encryption to IPSec with encryption. There is a small difference in the download times between the connections without IPSec and IPSec using the AH only transform set. There is a difference of 1.2 milliseconds in the average download times of the two sets. Similarly, there is also a small difference between ESP only and both protocols transform sets' download times. For all transform sets the download times increase gradually from webpage size of 1 KB to 60 KB, but beyond this size the download times increase sharply; almost exponentially for IPSec disabled and AH only and almost proportionally to the square page size in ESP only and both protocols.

According to the survey conducted by T. Sullivan [59], the average size of the web pages on the Internet is 60 KB. Figure 5-11 confirms the argument that, as webpage sizes exceeds 60 KB it takes longer to download from the server, hence performance is compromised. The assumption is made here that the propagation delay in the link

is negligible compared to processing and queuing delays. Therefore in the case where IPSec is enabled using ESP only or both IPSec protocols, the cost in terms of performance is very significant for pages above 60 KB. The other reason why the download time increases exponentially especially on large packets is the use of an encryption algorithm, which was found to increase server response time by 10 to 50 times [77] because in the transform sets without encryption the times do not increase that much. For connection through multiple intermediate nodes, delays should increase for bigger packets.

Figure 5-12 shows the download times obtained when using the IPSec transform sets available when using ESP only. The graph confirms the trend shown in Figure 5-11, of significant increase of download times when encryption is used especially on large packets.

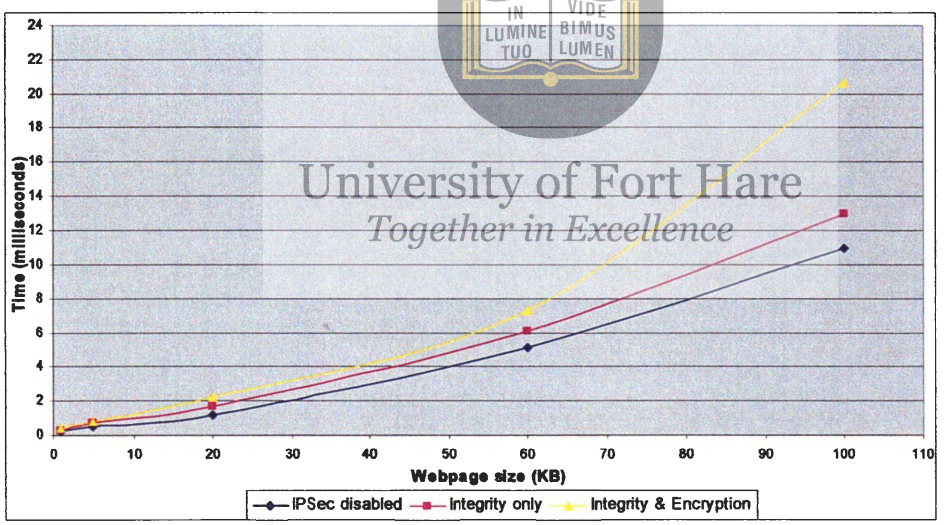


Figure 5-12: Impact of ESP on HTTP download times

The ESP integrity only transform set was using MD5 algorithm, which has a slightly lower average download time than SHA1. The download times of ESP integrity only are slightly higher than those of AH only using the same algorithm, if we compare ESP integrity only in Figure 5-12 and AH only in Figure 5-11.

Figure 5-13 compares the download times of different IPSec transform sets with both protocols applied. The download times without IPSec encryption are significantly

lower compared to encryption turned on, as can be seen in the graph. This is because encryption generally requires more processing time on the encrypting system, in our case on the web server.

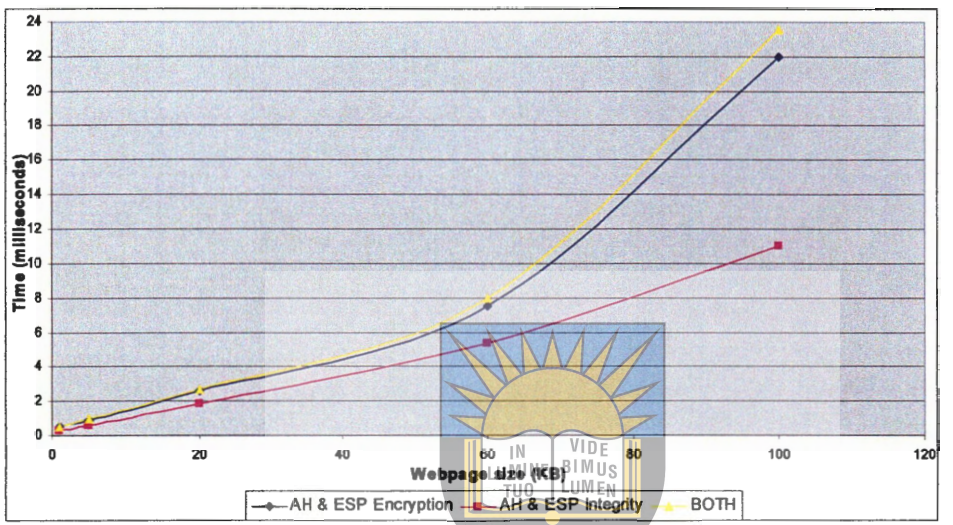


Figure 5-13: Impact of IPSec on HTTP download times using both protocols

This delays the transmission of a webpage request to the requesting client. There is a slight increase in download time from ESP encryption applied and both IPSec protocols fully implemented.

5.10 Overhead on FTP

The tests included frame overhead tests of IPSec on FTP, using different clear text file sizes. File sizes were considered for HTTP and FTP unlike frame sizes used for ICMP because of the different nature of these protocols used. ICMP is used by an application (ping) that sends data directly from source to destination. While HTTP and FTP are used to send data form source to destination using webpages and files respectively. The following sizes were considered: 1 KB, 20 KB, 100 KB, 1 MB, 5 MB and 10 MB, but in this section only sizes up to and including 1 MB will be shown because the trend beyond 1 MB is constant. This experiment was conducted by capturing the FTP file requests on the client machine that was requesting files from the FTP server using a web browser.

The breakdown of an FTP packet fragment is packaged in a typical Ethernet frame of 1518 bytes is:

- 14 bytes – Typical Ethernet header
- 20 bytes – Typical IP header
- 20 bytes – Typical TCP header
- 1460 bytes – Data after TCP header (the payload)
- 4 bytes – CRC

The first comparison was of the impact of AH only, ESP only and both protocols transform sets on FTP, which is shown in Figure 5-14. AH only adds an overhead of 24 bytes, ESP only adds 36 bytes and with both protocols applied the overhead is also cumulative as was the case in ICMP and HTTP.

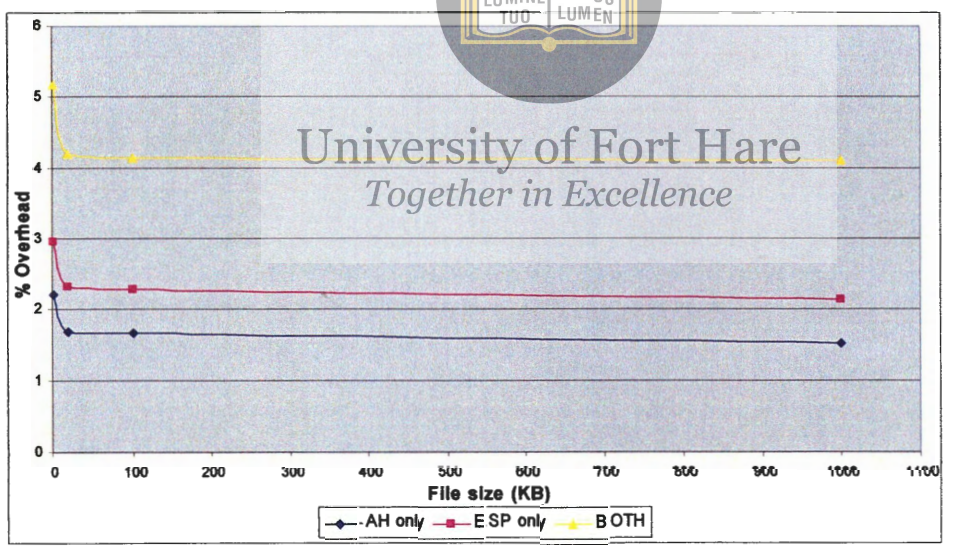


Figure 5-14: Comparison of different IPSec protocols on FTP overhead

The transform sets used in Figure 5-14 use the following algorithms; AH with MD5 or SHA1, ESP with both integrity and encryption and lastly both protocols applied. For all transform sets the overhead drops sharply from the file size of 1 KB to 20 KB, this is because the frame size of a 1 KB FTP file (1138 bytes, of IPSec both protocols transform set) is far lower than the fragment frame size, 1514 bytes for a 20 KB FTP file, hence the percentage impact of 24 bytes (AH only) is higher on 1

KB compared to 20 KB fragments. This trend applies to all the three transform sets on the graph. On the average AH only adds an overhead of almost 2%, ESP only adds an overhead of almost 2.5% and both protocols adds an average overhead of 4.5%, that is, the cumulative overhead of AH and ESP. The small difference between these values suggests that, there is also security advantage in using both protocols compared to other IPSec transform sets.

The overhead of fragmented packets is almost constant as file sizes increase, because the IPSec headers are applied on every transmitted fragment. AH only overhead is lower than that of ESP only and both protocols as was the case on HTTP, because AH only uses only one cryptographic algorithm to provide it's security services while ESP uses two, one for integrity and the other for encryption.

When comparing the overhead of using different ESP transform sets of integrity only and integrity with encryption, we get the trend shown in Figure 5-15. On average ESP integrity only adds an overhead of almost 2 percent on FTP and ESP with both integrity and encryption adds an overhead of about 2.5 percent.

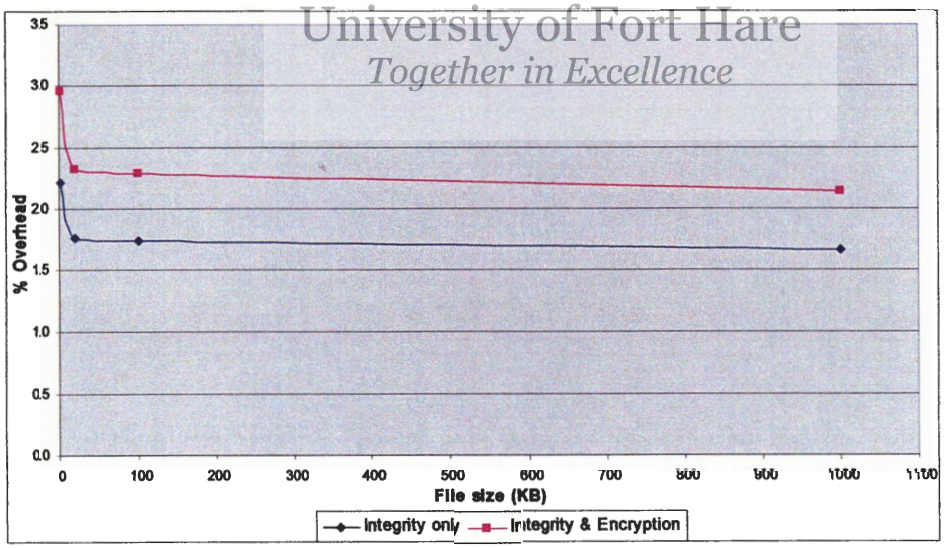


Figure 5-15: Impact of ESP transform sets on FTP overhead

The same trend shown in Figure 14 of the overhead remaining almost constant on fragmented packets applies also on ESP only transform sets shown in Figure 5-15.

Figure 5-16 compares the overhead added by using different IPSec transform sets that utilize both IPSec protocols.

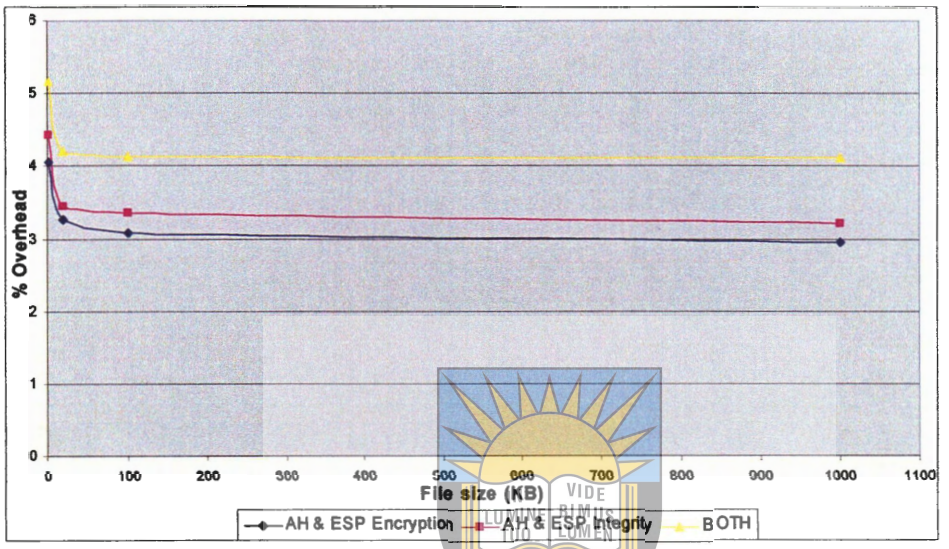


Figure 5-16: Impact of IPSec using both protocols on FTP overhead

Figure 5-16 shows a slightly insignificant increase in overhead when both IPSec protocols are fully implemented as compared to both protocols with partial ESP implementation. The overhead increases by almost 1%. It was noted that there is a difference in download times when using DES for encryption and 3DES in ESP protocol, but this difference is really insignificant.

5.11 Download Times on FTP

FTP download time tests were more uniform with a clearly visible trend than that of HTTP tests. The following file sizes were used for this test: 1 KB, 20 KB, 1 MB, 5 MB and 10 MB. Download time for all the frames of a particular download was recorded, that is, for the first frame and the last frame and computed the difference, which we recorded as the download time.

First, the times of IPSec disabled and IPSec using different protocols as shown in Figure 5-17 were compared.

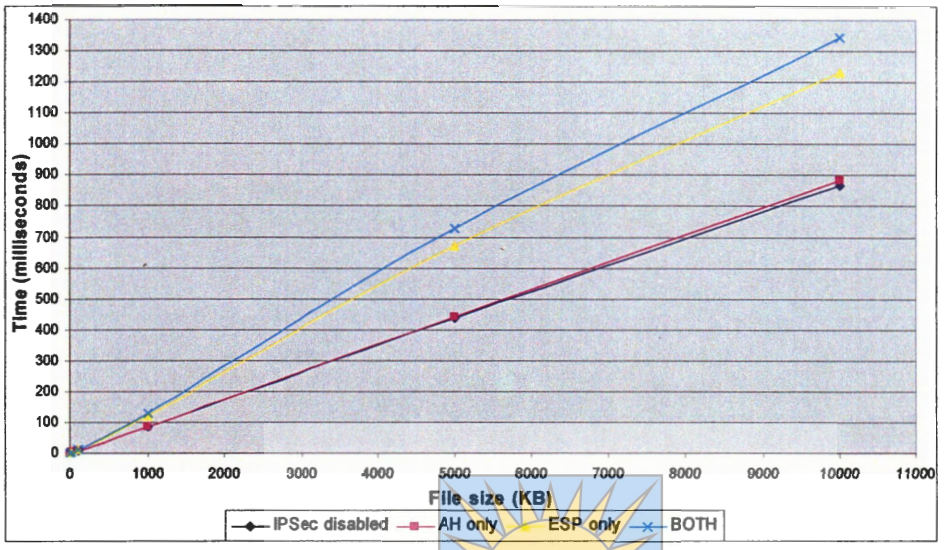


Figure 5-17: Impact of IPSec on FTP download times

Figure 5-17 shows that there is almost no difference in download times between IPSec AH only and IPSec disabled. There is a difference of 4 milliseconds in the average values of these two sets, which is very insignificant. The other two transform sets; ESP only and both protocols do have a small difference in their download times but it is more significant than the first two sets. Unlike the trend in HTTP download times, FTP download times increase linearly as the packet size increases. But still there are significant differences between IPSec transform sets with ESP encryption and those without ESP encryption, confirming that ESP encryption needs more processing on the processing node.

Figure 5-18 compares the download times of FTP using ESP only transform sets. In ESP integrity only we used SHA1 algorithms and full ESP uses SHA1 and 3DES. There is a huge increase of 43% for the range of file sizes used, between ESP with integrity only, and ESP with both integrity and encryption, this is a very significant performance cost consideration when deploying IPSec.

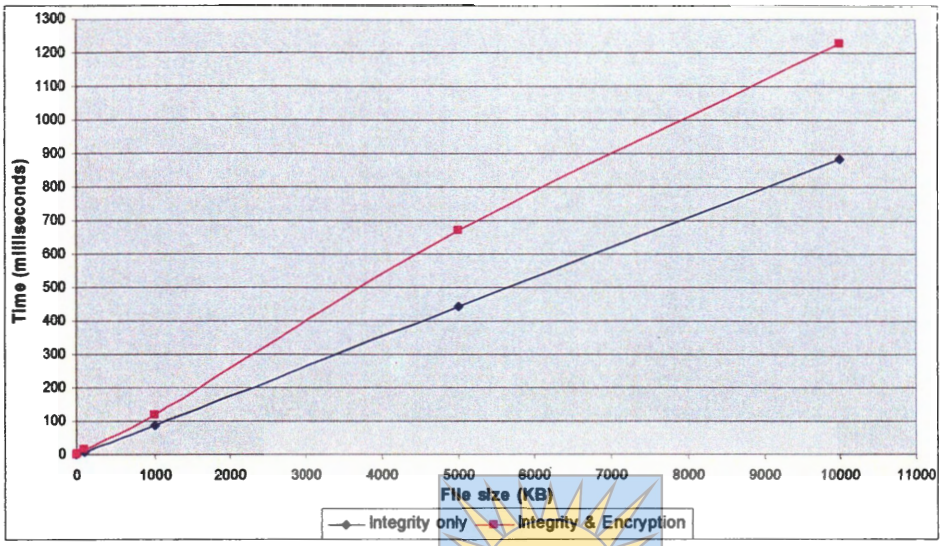


Figure 5-18: Impact of ESP transform sets on FTP download times

When comparing download times that use a full implementation of IPSec with different algorithms, the trend shown in Figure 5-19 appears.

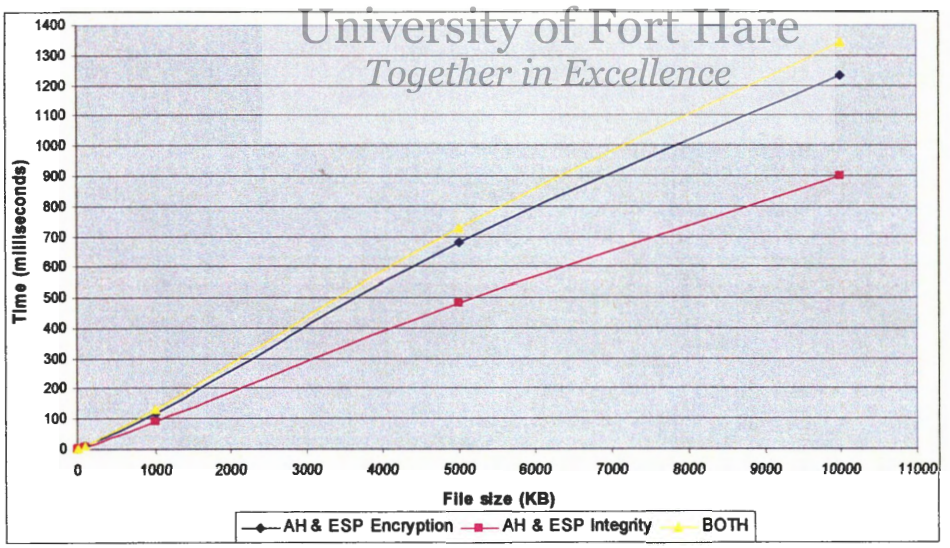


Figure 5-19: Impact of IPSec using both protocols on FTP download times

The algorithms used in these transform sets are; AH uses SHA1 through out, ESP encryption uses 3DES and integrity uses MD5. The transform set without encryption also has a lower download time compared to those with ESP encryption and

download times increase as file sizes increase. This means encryption increases the server response time linearly.

5.12 Overhead on TFTP

The impact of the IPSec frame overhead on TFTP was also tested, using different clear text file sizes. The following sizes were used: 4 bytes, 14 bytes, 200 bytes, 1 KB, 20 KB, 100 KB, 1 MB and 10 MB, but the graphs will only show sizes up to and including 1 KB because the trend beyond 1 KB is the exactly the same as that for 1 KB. This experiment was conducted by capturing the TFTP file requests from a third party TFTP server. The breakdown of a TFTP block that consists of 512 bytes of data is as follows:

- 14 bytes – Typical Ethernet header
- 20 bytes – Typical IP header
- 8 bytes – Typical UDP header
- 2 bytes – Opcode
- 2 bytes – Block number
- 512 bytes – Data
- 4 bytes – CRC

The above fields are present in each block transmitted from the TFTP server except the data field which varies with the file size. The initial read request form the TFTP client consists of a different header altogether, which consists of a read request opcode, the file name, which varies in size and the mode, which can be a string of either "netascii", "octet", or "mail". The client specifies the mode in which it can read the data. [69].

IPSec headers added on TFTP traffic are of the same size as those added on FTP. For the packet sizes tested, 4 bytes was chosen because it is small enough to require padding, 14 bytes is the size that does not need padding, at which the overhead is at its maximum. Theoretically the overhead starts low on packet sizes of 1 byte, in this case 4 bytes, because some overhead bytes are used as padding bytes, which reduces

the overall overhead of IPSec headers on the packets that need padding. All packets of less than 14 bytes require padding to be transmitted over the network and they all have a frame size of 64 bytes. This means the percentage of IPSec overhead increases from 4 bytes and reaches the maximum on 14 bytes packet. Then it starts to fall on packet sizes of more than 14 bytes.

This trend does not apply on packets of higher than 512 bytes nor on those divisible by 512, as the overhead will be higher if a packet is fragmented due to the overhead in the last fragment which distorts the proportionality of the overhead on these two fragment; hence the average overhead becomes higher than the minimum obtained on blocks of 512 bytes. This trend is shown in Figure 5-20; the following sizes were used: 4 bytes, 14 bytes, 200 bytes, 512 bytes, 530 byte, 1024 bytes, and 1040 bytes.

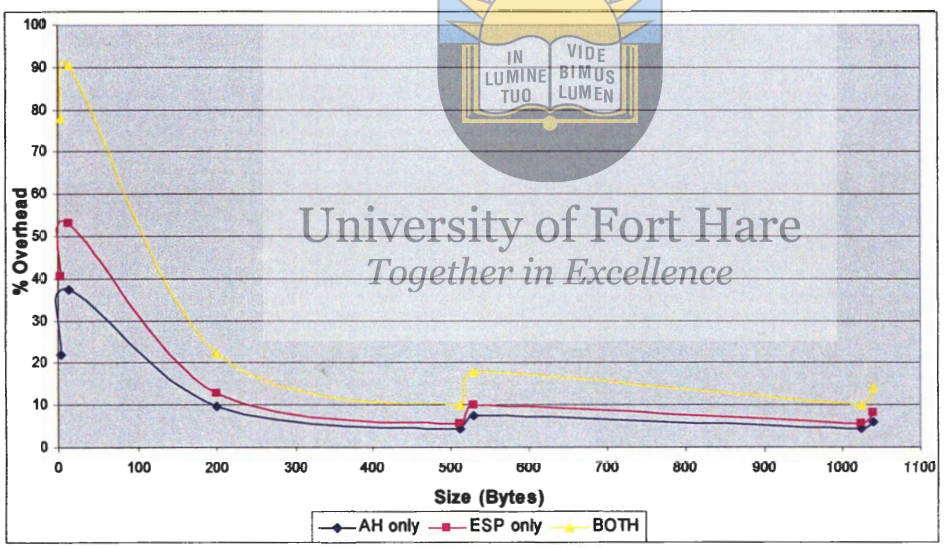


Figure 5-20: Impact of IPSec on TFTP overhead on file sizes divisible and not divisible by 512

Figure 5-20 shows the trend of IPSec overhead for packets divisible and not divisible by 512 bytes. The graph has peaks and troughs from 512 bytes onwards. When both protocols were considered, any packet larger than 512 but less than 1024 will have an overhead of more than the minimum mentioned above of 9.96% obtained on 512 bytes and 1024 bytes. In these experiments 530 bytes and 1040 bytes were used, as these points have a higher overhead as shown on Figure 5-20. This means that the higher the overhead the closer the size of the last fragment is to 512 bytes; otherwise

if it is divisible by 512 the overhead is at its lowest. And on the same note, the overhead is also lower towards the minimum the fewer the number of blocks or fragments there are in a transmission of blocks not divisible by 512. There also seems to be a security advantage in using both IPSec protocols as compared to either one protocol as it induces only about 8% additional overhead from AH only to both protocols on a 1040 bytes packet size.

5.13 Download Times on TFTP

These tests included measuring the download times for TFTP. Figure 5-21 shows the impact of IPSec on download time for packet sizes divisible by 512 with IPSec disabled and IPSec enabled. The download times increase very slowly from 4 bytes (theoretically from 1 byte) to 512 bytes due to the fact that, the processing time for one frame of a full block or less is almost the same. But the time increases linearly for blocks of 512 bytes, as can be seen on the last two points in Figure 5-21. The results suggest that, there is a delay when downloading blocks of 512 bytes, and this delay increases exponentially when IPSec is enabled.

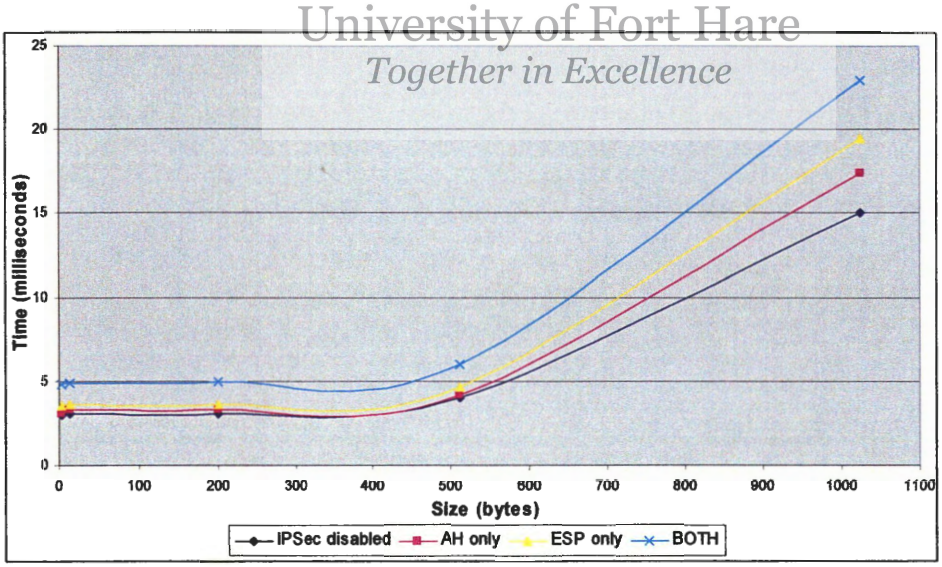


Figure 5-21: Impact of IPSec on TFTP download times for file sizes divisible by 512

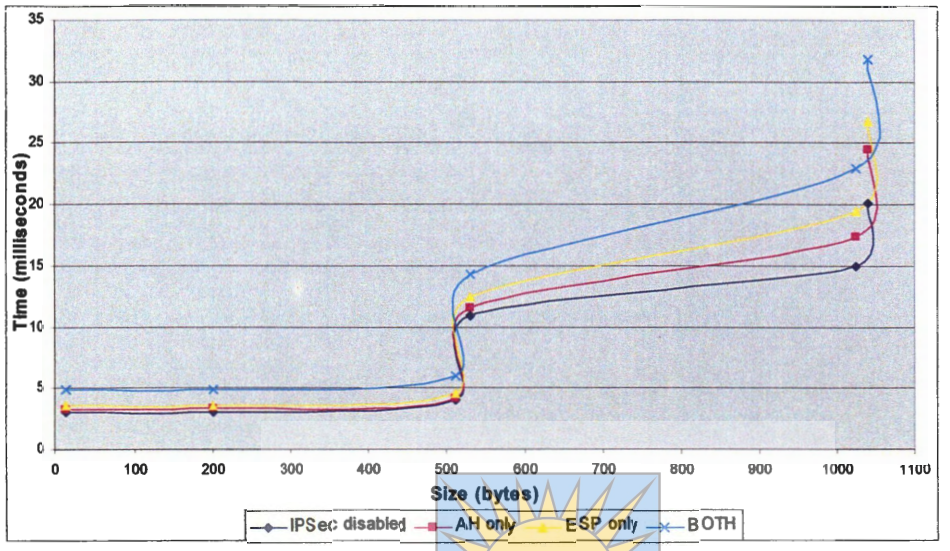


Figure 5-22: Impact of IPSec on TFTP download times for file sizes divisible and not divisible by 512

Figure 5-22 compares TFTP download times with IPSec disabled compared to IPSec enabled on packets divisible and by those not divisible by 512 using the same file sizes as used in the overhead tests. File sizes of 530 and 1040 bytes distorts the gradual increase in download times, hence an increase of more than 200% (using both protocols) in download times from 512 bytes to 530 bytes because 530 bytes is made up of two blocks and the increase is very important between 2 frames of similar size but one being slightly bigger than 512 bytes. The increase in download time (using both protocols) from 1024 bytes to 1040 bytes is very much lower at 40%.

Comparing FTP (TCP) and TFTP (UDP) IPSec overhead, Figures 5-16 and 5-20 for the same file size of 1 KB, shows that the IPSec overhead on UDP is twice that of IPSec on TCP. Comparing the UDP and TCP download times for a file size of 1 KB shows that UDP takes 22.935 milliseconds compared to 0.246 milliseconds for TCP. The difference between these two figures applies in the same proportion to UDP and TCP traffic with IPSec disabled. These results show that, the performance of TCP applications is less affected by IPSec as compared to those that run on UDP.

5.14 Conclusion

The results confirmed that the use of additional headers in any communication channel will obviously increase a certain amount of traffic overhead on the network. But this overhead depends on a number of factors, such as how applications treat additional headers and how services that add additional headers apply these headers on different application protocols. For instance, in IPv4 ICMP, IPSec headers were applied on the first fragment only, while in IPv6 ICMP they were added on each fragment that is transmitted on Windows implementation. This causes the overhead of IPSec to be higher in IPv6 fragmented packets than it is in IPv4. But the effect of this higher overhead in IPv6 is not a true reflection of the overall overhead because of the fact that encapsulated IPv6 frames are generally larger than IPv4 frames, and this results in a lower overhead being recorded. On the same note, different IPSec transform sets that provide different security services add different sizes of IPSec headers depending on the protocols used; that is, whether AH only, ESP only or both protocols.

Generally the overhead induced by additional headers and the eventual subsequent fragmentation can be reduced by using compression techniques, that might reduce the number of fragments needed to transmit large packets. Compression on the other hand might also reduce the time needed to transmit data from one network point to the other, that is, propagation delay.

The drawback in using data compression is that the compression and decompression processes may take a lot of memory and processor time [76]. This coupled with the use of encryption will ultimately increase propagation delay. Hence using faster computers (processors) in cases where a lot of compression and encryption processes are used significantly reduces propagation delay which in turn improves QoS.

Encryption and decryption processes require significant amount of processing time as well and organizations are increasingly requiring more privacy even on the web server, which results in data encryption for privacy often coupled with data compression to save storage space. There is an urgent need to develop adaptive optimal encryption-compression algorithms that are suitable for different applications that need IPSec security services. One research has shown that web server encryption slowed the server response time by 10 to 50 times [77]. This was

shown in these experiments on HTTP and FTP download times, where the times increased exponentially when using encryption especially on large packets. The other reason for the increase in the download times is the additional IPSec headers which provided security services. Fortunately encryption and compression technology is improving rapidly, thus reducing performance costs of these technologies.

Considering the results of some related work, [6] showed that the use of IPSec degrades the network performance in terms of throughput and end-to-end delay for data transmission. Our work also shows that there is considerable delay when comparing data transmission with IPSec disabled and IPSec enabled, based on the RTT and download time were measured. [5] also showed that there are significant performance implications when IPSec is deployed on single processors as compared to a dual processor computer.

These experiments also compared the impact of IPSec on TCP and UDP, using FTP and TFTP protocols respectively. Considering the trends shown in Figures 5-16 and 5-20 using the same file size of 1 KB, show that the IPSec overhead on UDP is twice that of IPSec on TCP. Comparing the UDP and TCP download times for a file size of 1 KB shows that UDP takes 22.935 milliseconds compared to 0.246 milliseconds for TCP. But these figures are not entirely due to IPSec, because even with IPSec disabled UDP is still slower than TCP. When IPSec is applied UDP becomes relatively even slower than TCP. These results show that, the performance of TCP applications are less affected by IPSec as compared to those that run on UDP.

CHAPTER SIX – CONCLUSION AND FURTHER WORK

This chapter concludes the work done to evaluate the impact of IPSec traffic on IPv4/IPv6 dual stack system. It reports on the successes and limitations of the system tested. It also gives guidelines on how best to implement IPSec based on the findings here. Lastly it suggests the possible future extensions of the research.

6.1 Introduction

The research goals were basically to evaluate how the compulsory use of IPSec in IPv6 would impact on the IPv4/IPv6 dual stack traffic. The evaluation involved setting up an IPv4/IPv6 dual stack network, consisting of five computers of which two were clients and three were servers running Windows XP Professional SP 2 and Windows Server 2003 SP1 respectively. The experiments involved determining the overhead of additional IPSec headers and their effect on RTT and download times. From the test results guidelines will be given on how to implement IPSec efficiently without wasting network bandwidth.

These experiments showed that, generally the impact of IPSec overhead on all protocols tested is higher for smaller packets as compared to larger packets for both IPv4 and IPv6, except in cases where the packet is small enough to require padding bits in IPv4. This is because part of IPSec overhead is used to replace the padding bits that were otherwise needed in the absence of IPSec. The impact on the transmission delay appears significant with large packets. It was also observed that, the use of both IPSec protocols increases the overhead cumulatively as the overhead of each protocol.

6.2 Limitations of this research

This research is limited to a specific system or set up. Firstly, the experiments conducted were based on a single site, that is, intra-site communication. Due to the fact that intra-site tests were used the IPSec mode most suitable for intra-site communications was tested, which is the transport mode. Inter-site experiments are a possible further research area, where IPSec protected traffic can be transmitted from one network to the other using an intelligent switch or a router as a gateway. In this

case it will be appropriate to consider both modes; transport and tunnel. Secondly, the tests were conducted on Microsoft operating systems, which by design do not support IPv6 IPSec ESP encryption. Even though IPv6 has null encryption an encryption algorithm can be specified, which adds an overhead but no encryption services are provided on the packets.

These results also apply to the Intel Pentium 4 processors. However the impact of IPSec traffic overhead on different types of processors was not explored.

6.3 General guidelines for implementing IPSec

Part of the expected output of this research are recommendations based on the tests conducted on the system, and on which QoS features of the network can be achieved with the best possible security. This section will give an insight on how best to use IPSec, but these are just general guidelines which differ greatly from one network set up to the other. But where security is very important, network performance and quality of service can be compromised without hesitation, unless the service being provided is rendered unusable, as in the case in real-time applications such as video conferencing in networks were bandwidth capacity is greatly limited.

The overhead of processing IPSec based packet of small size from the source to the destination node may become more significant, especially when a network is large. This is due to serious delay problems, in multiple and heterogeneous (multiple MTU and multiple layer 3 intermediary) networks. Also the processing overhead will increase further in any intermediate node (switch or router) in the case of multiple access links, Ethernet, and especially in a transport mode.

Firstly, the use of IPSec on very small packets is unnecessary; unless the data is highly sensitive in terms of security and only if the application can be the ultimate weakest link in the network security. From the experiments conducted, it was observed that, for IPv4 ICMP the highest overhead is obtained on packet size of 18 bytes and for IPv6 it is obtained on 1 byte packet sizes. After that, the overhead decreases gradually as the packet size increases for both IPv4 and IPv6. The same trend applies to HTTP, FTP and TFTP. Therefore, the larger the packet size the less the impact of the IPSec overhead as this overhead will be spread over a larger packet size. On the same note, the IPSec overhead on IPv6 ICMP is significantly larger than

that on IPv4 ICMP on large packet sizes because in IPv6 the IPSec headers are added on each fragment, while on IPv4 they are applied once for all fragments. Therefore using large IPv6 ICMP packets without IPSec would be preferable as control messages need to travel faster. The RTT for ICMP and download time for FTP increases linearly as packet size increases, while the download time for HTTP and TFTP increases exponentially.

The use of either DES or 3DES in IPSec encryption results in the same overhead bits being added to the packet which ultimately will have the same frame-size overhead impact. Therefore, there is a security advantage in using 3DES rather than DES in Windows Server 2003 IPSec, since 3DES provides stronger security than DES. But they differ on the RTT and download time, in that 3DES increases the RTT and download time because being more secure than DES it is implemented differently than the latter. But the difference between the two is very insignificant, hence it was not documented and considering that 3DES is more secure, there is no valid reason in using DES when 3DES is supported.

The tests also showed that, irrespective of the integrity algorithm used in both AH and ESP protocols, between SHA1 and MD5, the IPSec header size does not change for any IPSec transform sets tested. Hence the percentage overhead for both algorithms is the same. But they affect the round trip time and download time differently. The use of SHA1 increases the round trip time and download time, because SHA1 is a 160-bit function while MD5 is a 128-bit hash function, though it takes an insignificantly longer time to process. Therefore, there is a security advantage in using SHA1 rather than MD5 for authentication.

A comparison of IPSec overhead impact on IPv4 and IPv6 showed the overhead has a higher impact on IPv4 as compared to IPv6 for smaller packets that are not fragmented. This can be attributed to the fact that IPSec headers' sizes are the same on both IP protocols, while the frame sizes are different on these IP protocols for the same packet sizes. IPv6 fragmented packets have a higher overhead as compared to IPv4, because in IPv4 the IPSec header is applied on the first fragment only, while in IPv6 it is applied on each fragment.

There is also no significant difference between the RTT of IPSec with ESP only and IPSec with both headers implemented, especially on smaller packets, but the

difference becomes slightly significant on very large packets. As a matter of fact the average RTT of ESP only and both protocols is almost twice the average RTT of IPSec disabled and AH only especially on large packets. Generally this also gives a security advantage to using both protocols instead of ESP only. This trend suggests that a longer processing time is required when an encryption algorithm is used in an IPSec transform set, which ultimately decreases the throughput of a communication channel. Hence it is quite important to apply encryption on selected traffic that really need to be encrypted in order to improve the performance of network especially on the transmission that can do without encryption.

6.4 Further Work

From the limitations mentioned in Section 6.2 possible extensions of this project can be suggested. Firstly, experiments across networks using both tunnel and transport IPSec modes is an area that needs further work. Since the IPSec overhead, round trip time and download time are likely to increase when a communication channel is moving across networks. This is due to additional devices such as routers and switches that might increase the delay.

Another area of further research is the impact of IPSec on IPv6 HTTP and FTP. The current implementation of Windows IIS has limitations that affect the functionality of HTTP and FTP on IPv6. On the same note, IPv6 ESP encryption when available is another area of interest on Windows Server 2003.

This research provides a benchmark for comparing IPSec frame-size overhead on IPv4/IPv6 dual stack in a Windows environment and Linux environment. Therefore, the same research on a Linux operating system is another area of further research. This research was submitted to an International Journal of Network Security (IJNS) [86] and the reviewers recommended the tests to be conducted also in Linux environment. Hence a follow up which will extend this research to a Linux environment is being envisaged.

6.5 Conclusion

When lack of security becomes enough of a nuisance users begin to weigh its benefits against having no security at all. Hence, to achieve the maximum security

possible with IPSec, one needs also to implement other security principles such as intrusion detection systems and firewalls.

Our results showed that IPSec definitely adds a significant overhead which ultimately impacts on traffic delay. Therefore there is a need to find ways of reducing this overhead. The following are mechanisms that can help alleviate this burden.

The use of compression significantly reduces the size of a packet, which ultimately reduces the number of fragments on fragmented packets. The use of effective compression algorithms should be coupled with powerful and faster computers that are capable of processing compression operations without or with as little performance implications as possible. Fortunately, such computers are already available. There is also an urgent need to develop adaptive optimal encryption-compression algorithms that are suitable to different applications that need IPSec security services and which do not compromise the strength of the security provided.

The findings also showed that there are relative advantages in terms of frame-size overhead and RTT in using AH only compared to ESP only or both protocols. This is due to the absence of an encryption algorithm that needs a longer processing time to perform it's functions. Therefore where necessary the use of AH only is recommended in terms of improving the QoS.

Another solution for reducing the frame-size overhead and possibly the delay might be the use of IPSec for authentication and encryption of the first fragment only of data transmission, which is currently the case in IPv4 ICMP. This implementation significantly reduces the overhead and RTT. But security risks of such an implementation on specific applications and protocols must be evaluated in order to eliminate all security vulnerabilities. This might be achieved by implementing strict measures for dropping suspicious packets, without unnecessarily increasing significantly the rate at which dropped packets need to be retransmitted.

Increasing the MTU of IPv6 packets, which in turn reduces the number of fragments for a particular transmission might also prove to be another solution to reducing the frame-size overhead. But this is a long term solution due to the nature of its implementation which might need a change in a number of standards and legacy network devices that conform to the current standards. But in a nutshell this solution improves the performance and QoS of IP. This solution goes hand in hand with the

previous mentioned solution of using efficient compression algorithms to reduce the size of frames.

Security is critical; be it security for data in transit or security of data storage. But efficient security mechanisms come at a price; degradation in performance and QoS. For data in transit, a larger bandwidth might be required due to the overhead added by security mechanisms and delay is increased. As for data storage, the time required to access the data is increased. Ultimately to obtain the best security possible is costly, as high performance computers are needed to process the operations of the most secure security systems. Failure to provide these resources might result in vulnerabilities in the security system and ultimately performance and QoS are affected.

Therefore, before making a decision on implementing a security system in-depth research needs to be conducted to evaluate the implications of such a security mechanism and to decide how best it can be implemented in order to obtain the best security and with the available resources, as well as ascertain whether new resources are needed.

REFERENCES

[1] N. Doraswamy and D. Harkins: "IPSec: The New Security Standard for the Internet, Intranets, and Virtual Private Networks", Prentice Hall PTR, ISBN: 0-13-011898-2, 1999.

[2] S. Kent and R. Atkinson "Security Architecture for the Internet Protocol" RFC 2401, November 1998.

[3] DARPA Internet Protocol Specification "Internet Protocol" RFC 791, September 1981.

[4] S. Deering and R. Hinden "Internet Protocol, Version 6 Specification" RFC 2460, December 1998.

[5] J. Ronan, *et al.* "Performance Implications of IPSec Deployment" Telecommunications Software & Systems Group (TSSG), Waterford Institute of Technology, Ireland, *available online*: http://w3.tmit.bme.hu/ips2004/papers/ips2004_002.pdf, *last accessed May 2004.*

[6] S. Ariga, *et al.* "Performance Evaluation of Data Transmission Using IPSec over IPv6 Networks", *available online*: http://www.isoc.org/inet2000/cdproceedings/1i/1i_1.htm, *last accessed June 2004.*

[7] "IPv6 Configurations and Test Lab for Windows XP", Windows XP White Paper, http://mail.cat.or.th/ipv6/ipv6configs.doc, *last accessed June 2004.*

[8] "IPv6 Transition Technologies", Microsoft Corporation, October 2005, *available online:*http://www.microsoft.com/windowsserver2003/techinfo/overview/ipv6coexist.mspx, *last accessed November 2005*

[9] D. Shinder [Windows Security], "IPv6: Windows Server 2003 Supports a More Secure IP – Sort of", *available online:* http://www.windowsecurity.com/articles/Windows_Server_2003_IPv6.html, *last accessed June 2005.*

[10] G. Tsirtsis and P. Srisuresh "Network Address Translation - Protocol Translation (NAT-PT)" RFC 2766, February 2000.

[11] C. Bouras *et al.* "The deployment of IPv6 in an IPv4 world and transition strategies" Internet Research: Electronic Networking Applications and Policy Journal, 2003 Volume: 13 Number: 2 Page 86-93.

[12] B. Carpenter and K. Moore "Connection of IPv6 Domains via IPv4 Clouds"

RFC 3056, February 2001.

[13] F. Templin *et al.* "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)" draft-ietf-ngtrans-isatap-20.txt, January 2002.

[14] Microsoft Windows 2003 Server, Help and Support Documentation.

[15] P. Loshin "IPv6: Theory, Protocol, and Practice" Second Edition, Morgan Kaufmann Publishers, ISBN: 1-55860-810-9, 2003.

[16] "13 - Using IPSec (IP Security Protocol)", *available online:* http://www.synack-hosting.com/13-Using_IPSec.pdf, *last accessed October 2005.*

[17] Matthew G. Naugle "Illustrated TCP/IP", Wiley Computer Publishing, John Wiley & Sons, Inc., ISBN: 0471196568, 1998.

[18] A. Farrel "The Internet and its Protocols – A Comparative Approach", Morgan Kaufmann, ISBN: 1-55860-913-X, 2004

[19] S. Deering and R. Hinden "IPv6 Specification" RFC 2460, December 1998

[20] "The ABCs of IP Version 6", CISCO, *available online:* http://www.ieng.net/application/pdf/en/us/guest/products/iosswrel/c1127/cdccont_0900aecd8018e369.pdf, *last accessed November 2005.*

[21] R. Gilligan and E. Nordmark, "Transition Mechanisms for IPv6 Hosts and Routers" RFC 2893, August 2000.

[22] A. Durand "Tunnel Broker" RFC 3053, January 2001.

[23] M. Blanchet and F. Parent "IPv6 transition mechanisms", Viagenie May 2000, *available online:* http://www.viagenie.qc.ca/en/ipv6/presentations/IPv6-transition-mechanisms_v1.pdf, *last accessed October 2005.*

[24] N. Ferguson and B. Schneier "Practical Cryptograhpy" Wiley Publishing, Inc., ISBN: 0-471-22357-3, 2003.

[25] "Interconnecting IPv6 Domains Using Tunnels", CISCO, *available online:* http://www.cisco.com/en/US/tech/tk872/technologies_design_guide09186a00800c98f6.shtml, *last accessed October 2005.*

[26] B. Carpenter and K. Moore "Connection of IPv6 Domains via IPv4 Clouds" RFC 3056, February 2001

[27] B. Carpenter and C. Jung "Transmission of IPv6 over IPv4 Domains without Explicit Tunnels" RFC 2529, March 1999.

[28] F. Templin, T. Gleeson, M. Talwar, D. Thaler, "Intra-Site Automatic Tunnel

Addressing Protocol (ISATAP)", draft-ietf-ngtrans-isatap-12 (Status: WORK IN PROGRESS), January 24th, 2003.

[29] C. Huitema, "Teredo: Tunneling IPv6 over UDP through NATs". draft-ietf-ngtrans-shipworm-08, (Status: WORK IN PROGRESS), September 17th, 2002.

[30] C. Huitema "Teredo: Tunneling IPv6 over UDP through NATs" (draft-huitema-v6ops-teredo-05), 5 April 2005.

[31] I. Guardini "Migrating from IPv4 to IPv6-planning an effective IPv6 transition" Global IP Summit 2000, *available online:* http://carmen.ipv6.tilab.com/papers/globalIPsummit-v6trans/home.html, *last accessed December 2005.*

[32] S. Kent and R. Atkinson "Security Architecture for the Internet Protocol" RFC2401, November 1998.

[33] S. Kent and R. Atkinson "IP Authentication Header" RFC2402, November 1998.

[34] S. Kent and R. Atkinson "IP Encapsulating Security Payload" RFC2406, November 1998.

[35] D. Maughan *et al.* "Internet Security Association and Key Management Protocol (ISAKMP)" RFC 2408, November 1998.

[36] G. Schafer "Security in Fixed and Wireless Networks", John Wiley and Sons, Inc., ISBN: 0-470-86370-6, December 2003.

[37] B. Schneier "Applied Cryptography: Protocols, Algorithms and Source Code in C", John Wiley & Sons, Inc., ISBN: 0-471-11709-9, 1996.

[38] D. Schnitzer "Applied IPv6 Security", *available online:* www.schnitzer.at/dominik/uni/ipv6sec.pdf , *last accessed September 2005.*

[39] J. Mogul "Internet Subnets" RFC 917, October 1984.

[40] N. Dunbar "IPSec Networking Standards - An Overview", Information Security Technical Report, Vol. 6, No. 1, 1 March 2001, Page 35-48.

[41] D. Harkins and D. Carrel "The Internet Key Exchange (IKE)" RFC 2409, November 1998.

[42] J. Reynolds and J. Postel "Assigned Numbers" RFC 1700, October 1994.

[43] D. Piper "The Internet IP Security Domain of Interpretation for ISAKMP"

RFC 2407, November 1998.

[44] H. Orman "The OAKLEY Key Determination Protocol" RFC 2412, November 1998.

[45] R. Rivest "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[46] D. Clark "Vulnerability's of IPSEC - A discussion of possible weaknesses in IPSEC implementation and protocols", SANS Institute 2002.

[47] "Internet Information Services" Windows Server 2003 Technology Centers, *available online:*http://www.microsoft.com/WindowsServer2003/iis/default.mspx, *last accessed October 2005.*

[48] "Windows 2000 Virtual Private Networking Scenario" Microsoft Technical Services, *available online:*http://www.microsoft.com/windows2000/techinfo/howitworks/communications/remoteaccess/w2kvpnscenario.asp, *last accessed October 2005.*

[49] "Defining CA Types and Roles" Microsoft Windows Server 2003 TechCenter, *available online:*http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/DepKit/1b28424c-8c62-44b6-a24f8ea06ac5832b.mspx, *last accessed October 2005.*

[50] L. Huang "Microsoft IPv6 Strategy", Microsoft Corporation, *available online:* http://www.usipv6.com/6sense/2005/may/07.htm, *last accessed October 2005.*

[51] "Teredo Overview" Microsoft Windows Server 2003 TechCenter, *available online:*http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/teredo.mspx, *last accessed October 2005.*

[52] "Introduction to IP Version 6" Microsoft Corporation August 2005, *available online:*http://www.microsoft.com/windowsserver2003/technologies/ipv6/introipv6.mspx, *last accessed October 2005.*

[53] S. Thomson and T. Narten "IPv6 Stateless Address Auto-configuration" RFC 2462, December 1998.

[54] "HP-UX IPSec Performance and Sizing White Paper" Hewlett-Packard Development Company, *available online:*http://docs.hp.com/en/6092/ipsecperf.pdf, *last accessed October 2005.*

[55] Joseph D. Touch "Performance Analysis of MD5" Applications, Technologies, Architectures, and Protocols for Computer Communication – Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication, SIGCOMM/ACM, 1995.

[56] "IPSec Components" Microsoft Windows Server 2003 TechCenter, *available online:*http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/ServerHelp/53054aad-48e3-446a-ae28-f727637f15a5.mspx.

[57] J. Klaue and A. Hess "On the Impact of IPSec on Interactive Communications" Technical University Berlin, *19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Workshop 17, International Workshop on Security in Systems and Networks – SSN (to be incorporated).*

[58] M. Kaeo and T. Van Herck "Methodology for Benchmarking IPSec Devices" Internet-Draft (draft-ietf-bmwg-ipsec-meth-00), November 2005, Expires: May 5, 2006, *available online:* http://www.ietf.org/internet-drafts/draft-ietf-bmwg-ipsec-meth-00.txt, *last accessed November 2005.*

[59] T. Sullivan "How much is too much?", *available online:*http://www.pantos.org/atw/35654.html, *last accessed November 2005.*

[60] Finisar Surveyor 5.5 Help Files.

[61] DLink® Dual-Speed 8-port 10/100 Ethernet Switch, Product Description, *available online:* http://www.d-link.co.za/des-1008d.html, *last accessed 17 November 2005.*

[62] **802.3u-1995** IEEE Standards for Local and Metropolitan Area Networks: Supplement to Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications: Media Access Control (MAC) Parameters, Physical Layer, Medium Attachment Units, and Repeater for 100 Mbps Operation, ISBN 1-55937-542-6.

[63] B. Bunch "An Introduction to Auto-Negotiation", February 1995, *available online:*http://www.scyld.com/NWay.html#7.0, *last accessed November 2005.*

[64] J. Postel "Internet Control Message Protocol" RFC 792, September 1981.

[65] R. Fielding *et al.* "Hypertext Transfer Protocol – HHTP/1.1" RFC 2616, June 1999.

[66] J. Postel and J. Reynolds "File Transfer Protocol" RFC 959, October 1985.

[67] Information Sciences Institute University of Southern California "Transmission Control Protocol" RFC 793, September 1981.

[68] J. Postel "User Datagram Protocol" RFC 768, August 1980.

[69] K. Sollins "The TFTP Protocol (Revision 2)", RFC 1350, July 1992.

[70] International Engineering Consortium (IEC) "Voice Quality (VQ) in Converging Telephony and Internet Protocol (IP) Networks", *available online:* http://www.iec.org/online/tutorials/voice_qual/topic05. html, *last accessed November 2005.*

[71] J. Postel "The TCP Maximum Segment Size and Related Topics", RFC 879, November 1983.

[72] E. Gerich "Guideline for Management of IP Address Space", RFC 1366, October 1992.

[73] R. Murty, Intel Corporation "Accelerating the Transition to IPv6 Using the Intel® Internet Exchange Architecture" Intel Developer UPDATE Magazine, September 2002.

[74] M. Nakajima and N Kobayashi "IPv4/IPv6 Translation Technology", December 2003, *available online:* http://www.fujitsu.com/downloads/MAG/vol40-1/paper21.pdf, *last accessed November 2005.*

[75] Z. Wang "Internet QoS: Architectures and Mechanisms for Quality of Service", Morgan Kaufmann Publishers, ISBN: 1-55860-608-4, 2001.

[76] N.J. Yeager and R.E. McGrath "Web Server Technology: The Advanced Guide for World Wide Web Information Providers", Morgan Kaufmann Publishers, ISBN: 1-55860-376X, 1996.

[77] E. Bina, R. McCool, V. Jones and M. Winslett "Secure Access to Data Over the Internet", Proceedings of the 3rd International Conference on Parallel and Distributed Information Systems, p. 99-102, ISBN:0-8186-6401-0, 1994.

[78] "How IIS 6.0 Supports IPv6 (IIS 6.0)" Microsoft Windows Server 2003 TechCenter, *available online:* http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/f5e56a2d-0b75-4cfa-810d-61badf87e40f.mspx, *last accessed November 2005.*

[79] M. Morrow and K. Vijayananda "Developing IP-Based Services: Solutions

for Service Providers and Vendors", Elsevier Science, USA, ISBN: 1-55860-779-X, 2003.

[80] H. Orman and P. Hoffman "Determining Strengths For Public Keys Used For Exchanging Symmetric Keys", RFC 3766, April 2004.

[81] T. Kivinen *et al.* "Negotiation of NAT-Traversal in the IKE", RFC 3947, January 2005.

[82] Huttunen *et al.* "UDP Encapsulation of IPSec ESP Packets", RFC 3948, January 2005.

[83] D. Shinder "NAT Traversal (NAT-T) Security Issues", *available online:* http://www.windowsecurity.com/articles/NAT-Traversal-Security.html, *last accessed December 2005.*

PUBLICATIONS AND PRESENTATIONS

[84] "Performance Analysis of IPSec in IPv6 Transition Mechanisms", Southern African Telecommunication Networks and Applications Conference (SATNAC) 6 – 9 September 2004, George, South Africa.

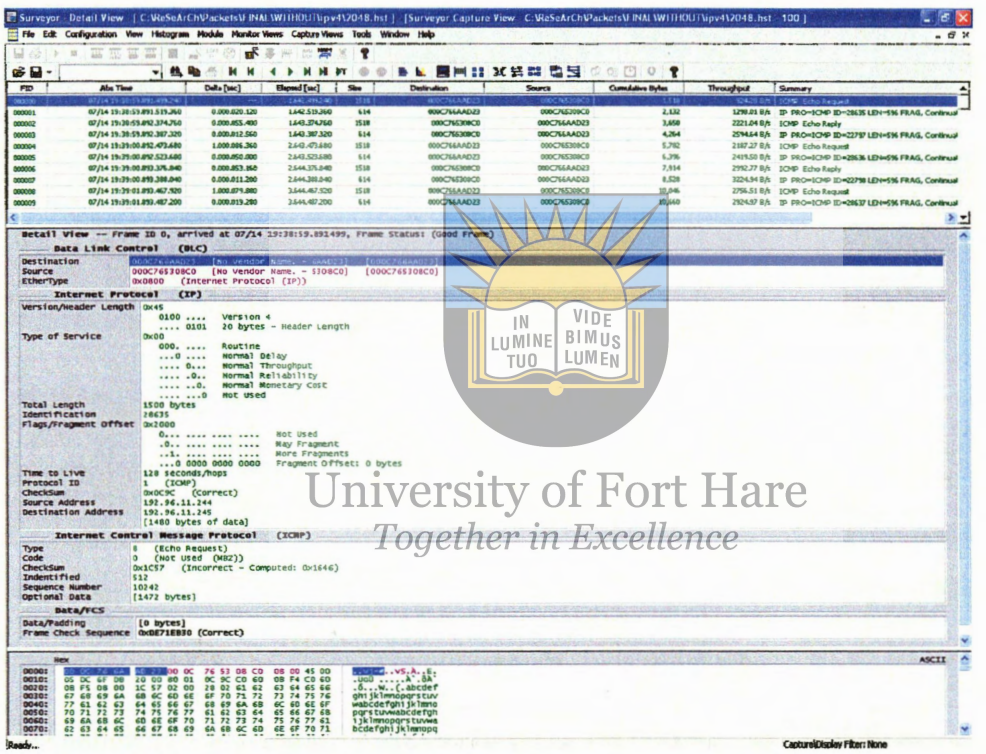
[85] "IPSec Overhead Analysis in Dual Stack IPv4/IPv6 Transition Mechanisms", The 8th International Conference on Advanced Communication Technology (ICACT, IEEE Communications Society) 20 -22 February 2006, Dangwon Do, Korea.

[86] "IPSec Traffic Overhead Analysis in Dual Stack IPv4/IPv6 Transition Mechanisms – An Analytical Study" International Journal of Network Security, *(Submitted)*

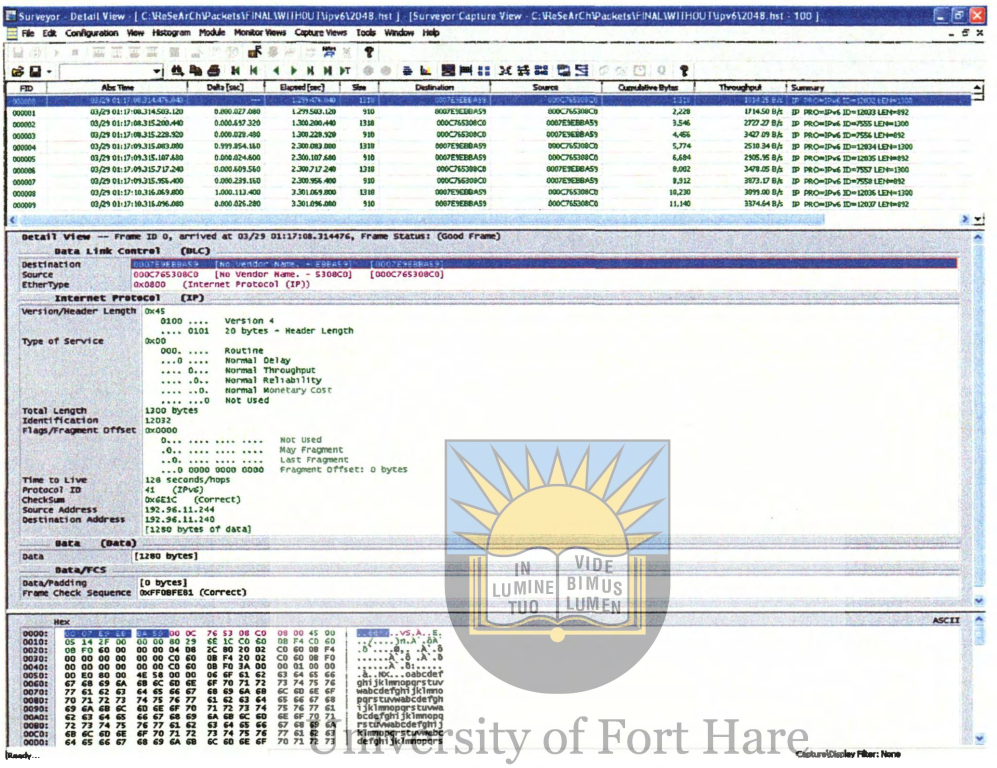
APPENDIX A: FINISAR SURVEYOR 5.5 SCREEN SHOTS

Section A: Snapshots of traffic with IPSec disabled

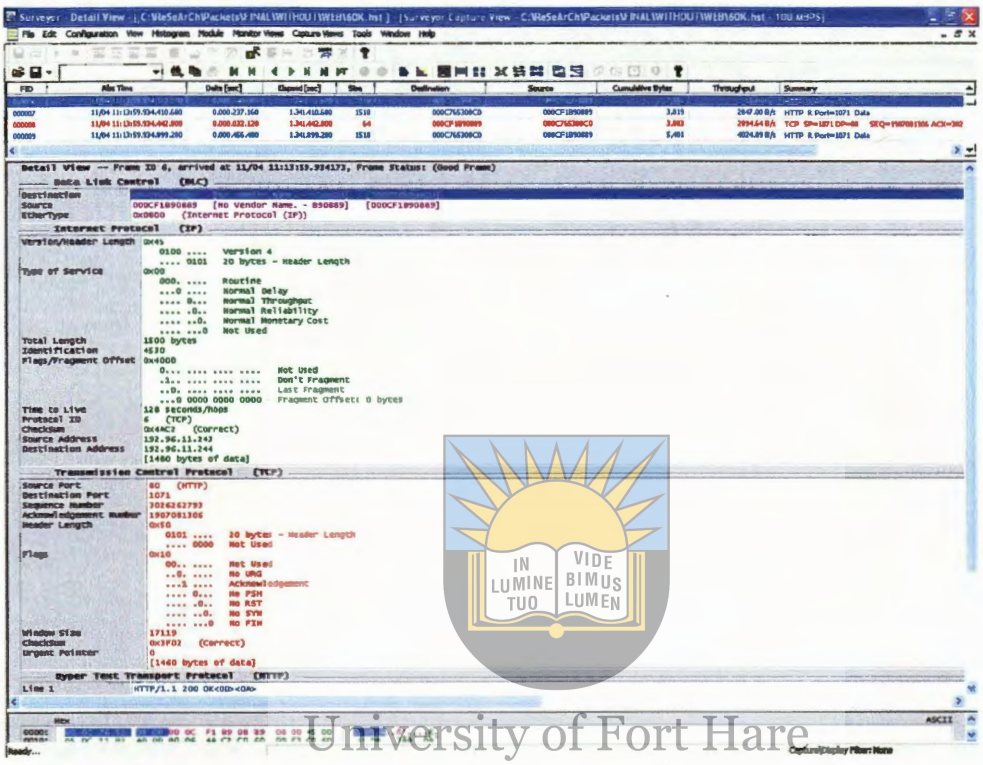
1. ICMP IPv4



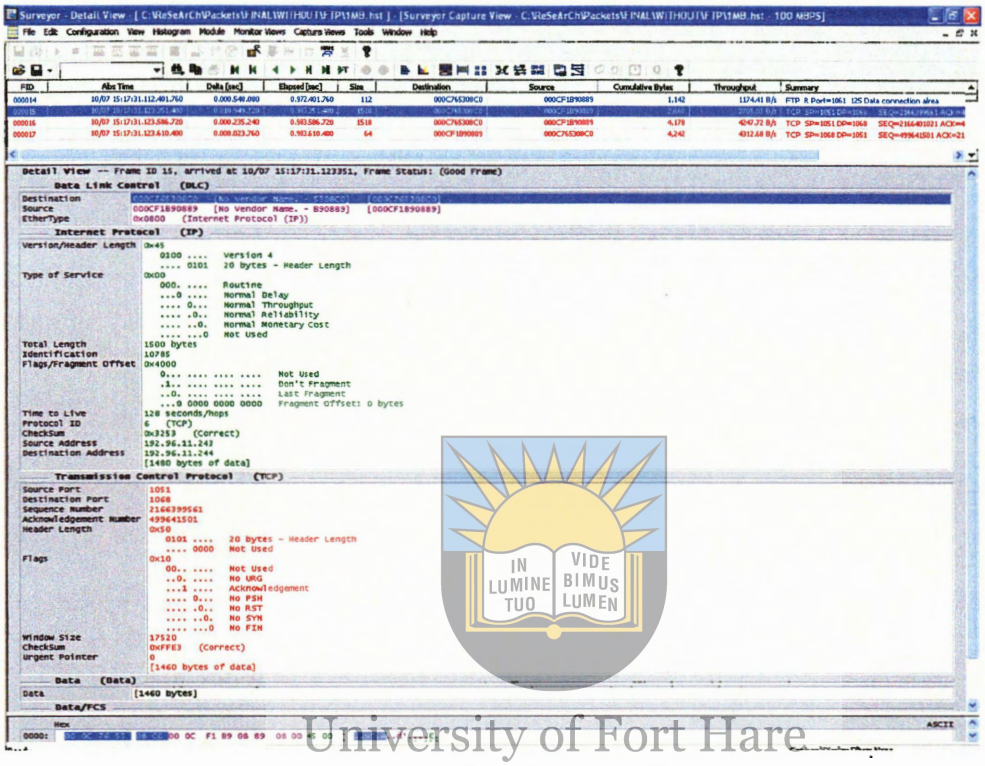
2. ICMP IPv6



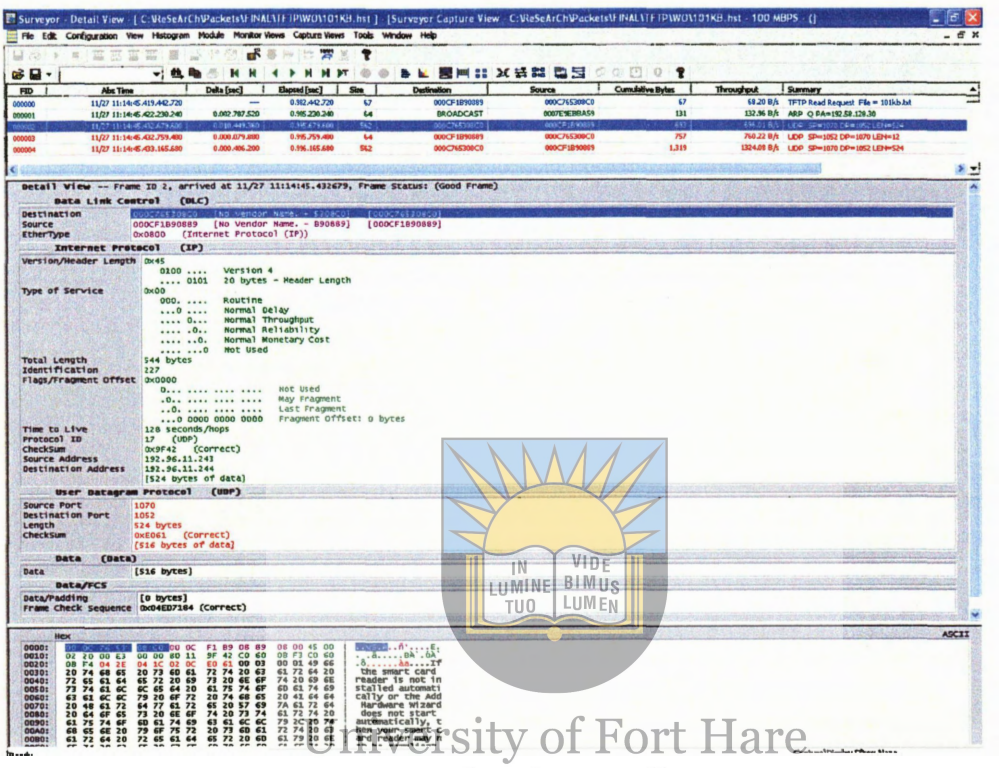
3. HTTP



4. FTP

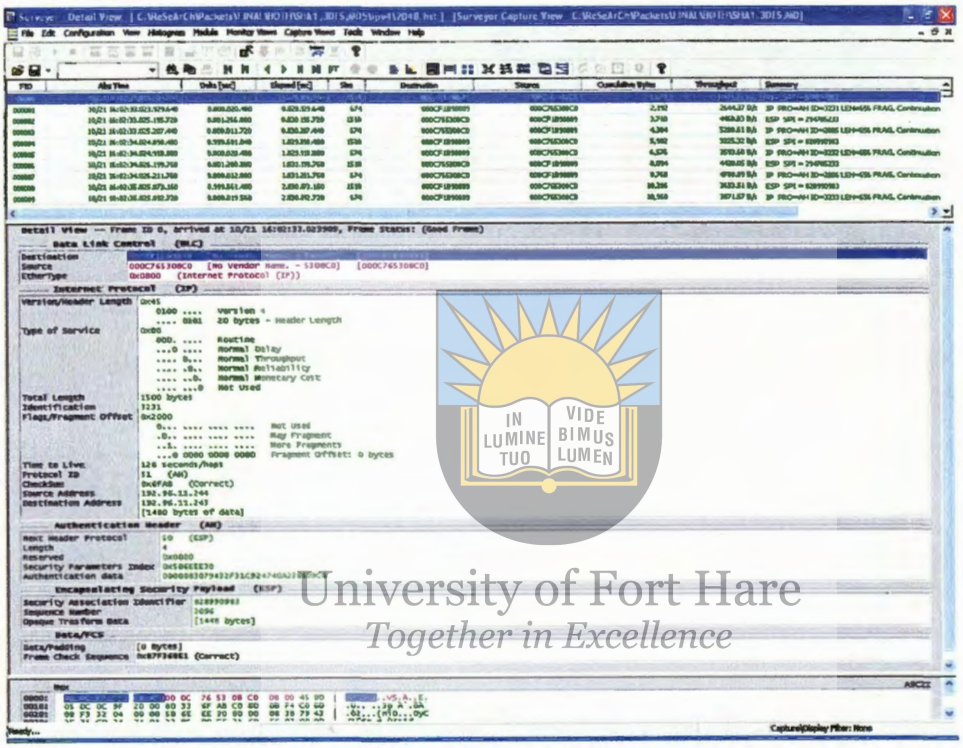


5. TFTP

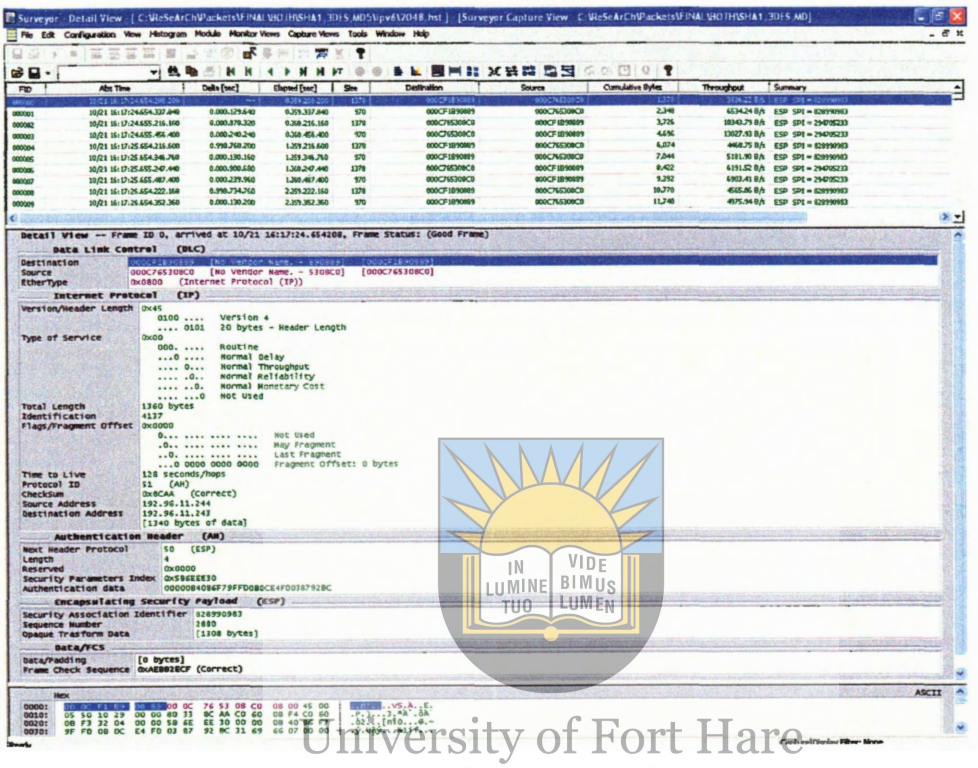


Section B: Snapshots of traffic with IPSec enabled using both protocols

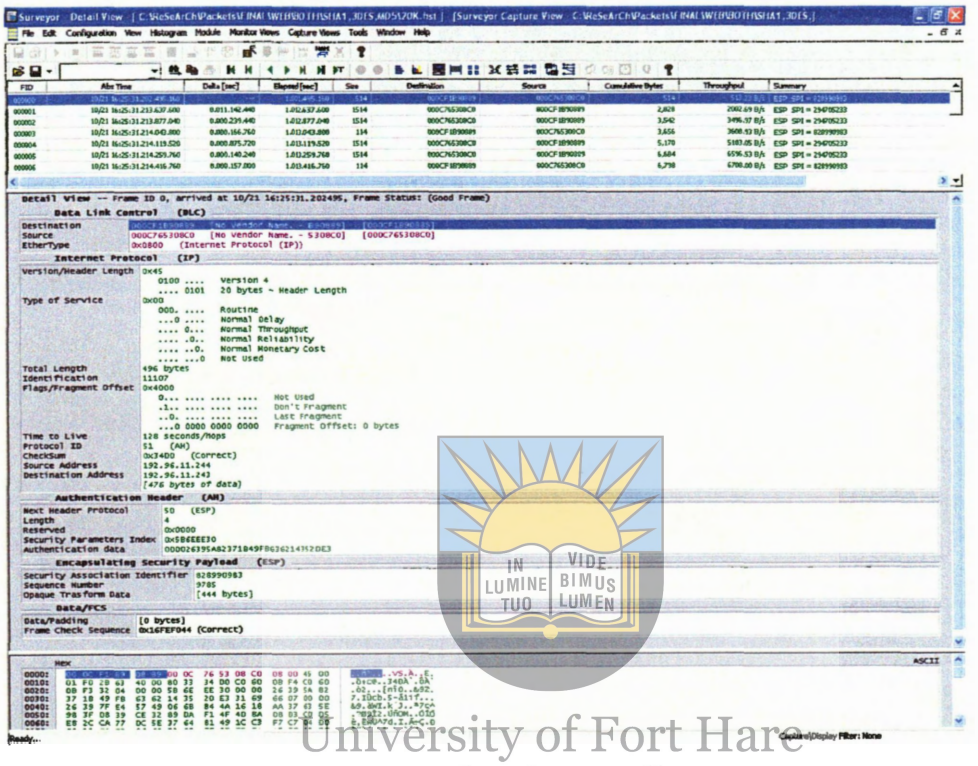
1. ICMP IPv4



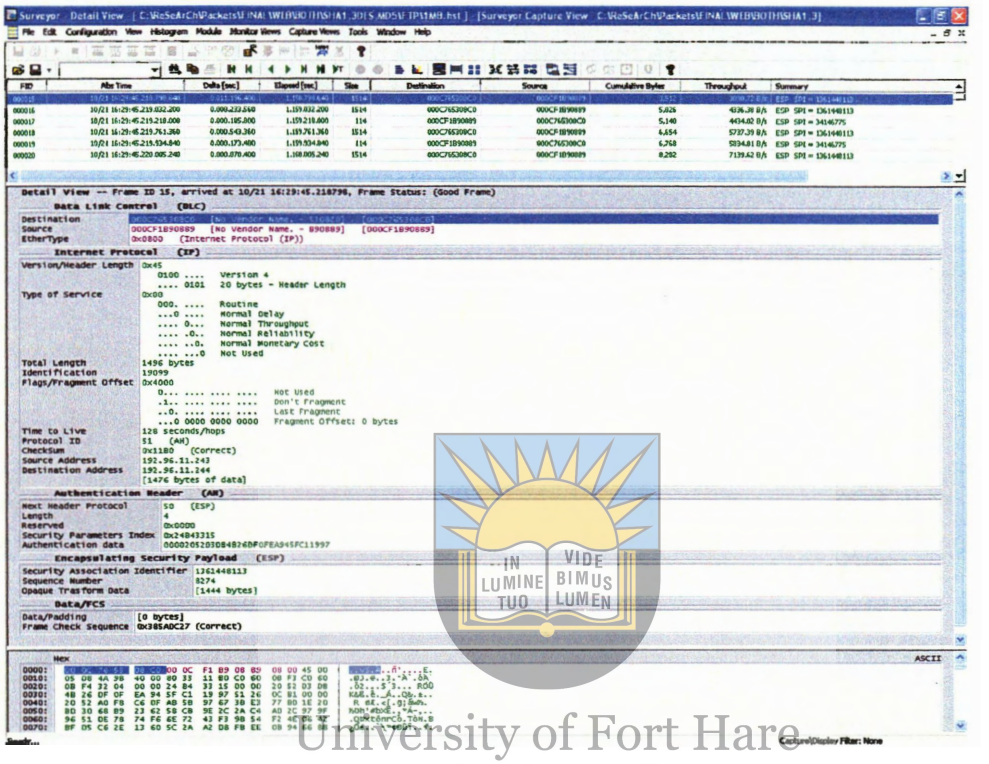
2. ICMP IPv6



3. HTTP



4. FTP



5. TFTP