Göteborg University
School of Economics and Commercial Law
Department of Informatics

Master thesis, IA7400
Spring Semester 2000

# IPsec, the Future of Network Security?

Anders Dahlgren & Oskar Jönsson

## Abstract

This master thesis deals with Internet security in general and IPsec in particular. Traffic and transactions over the Internet are risky, and credit card numbers are easily stolen and abused. IPsec has been developed to ensure security and integrity and we discuss how IPsec works and where in the computer network it resides. In order to come up with any conclusions and results, we did a literature research of the Internet Engineering Task Force's (IETF) Request For Comments (RFC) documents on the Internet. The few books available are also built upon the RFCs. The study found that IPsec will provide security for all kinds of computer traffic without, or with little, human interaction. IPsec can replace current application security techniques like PGP etc. Since IPsec is built in a modular way it is future proof and it is easy to add new cryptographic methods when such are developed and proven more secure.

| | | |
|---|---|---|
| *Supervisors:* | *Birgitta Ahlbom* | *(Department of Informatics, Göteborg)* |
| | *Isac Antblad* | *(Ericsson Mobile Data Design AB, Göteborg)* |

# Preface

It has been a privilege for both of us to be able to write our Master thesis at Ericsson Mobile Data Design AB (ERV). We would like to thank the staff that has helped us whenever we needed their assistance.

Thank you also, Isac Antblad, our supervisor at ERV, and of course Birgitta Ahlbom, our instructor at the Department of Informatics at the University of Gothenburg.

One last thank you to Roger Simon from Sistech, who provided a lot of valuable introductory information as well as an interesting lecture near the end of the thesis writing.

# Table of Contents

# 1 Introduction

As the number of Internet users is heavily increasing, and the way the Internet is used is getting more and more diverse, the issue of security becomes more important than ever before. The Internet with its main protocol suite TCP/IP was originally designed for handling military communication during a nuclear war; today the various ways to use it are almost unlimited. Quite recently, the e-commerce phenomenon has flourished, the number of companies providing commercial services and online shopping via their web sites increases every day, and online banking is becoming more and more common among ordinary people.

With this kind of activity taking place, the need for security arises. To use a commercial online service in a comfortable way, you want to be absolutely sure that nobody can monitor your transactions, pick up your password or credit card number and abuse it. At the same time, you do not want to have to remember numerous passwords and PIN-codes, or have to bring your digipass wherever you go.

The aim of this thesis is to examine the structure and functionality of IPsec, and in what ways the use of IPsec will result in increased network security. We will also provide an overview of TCP/IP and the topic of network security in general. The thesis has been written in co-operation with Ericsson Mobile Data Design AB (ERV).

## 1.1 Background

Ericsson Mobile Data Design AB develops systems for future mobile data and telecommunications. They are currently working on the next generation of mobile communication, GPRS, which will complement the current standard, GSM [35]. The major feature of GPRS is that it is packet switched instead of circuit switched, which means that data are sent in packets in the same way as in traditional computer networks. This development has spurred new interest in secure communications since nobody wants their telephone conversations to be tapped.

One solution to this problem would be IPsec, which is a new technique for securing IP networks developed by the IETF. It has not yet been standardized in any way, but it is regarded as a standard by the security community. IPsec is part of the next version of the TCP/IP protocol suite, IPv6, which will replace today's version IPv4 within the years to come. The introduction of IPv6 will result in more people using IPsec, which in turn should lead to improved network security.

## 1.2 Purpose

The main purpose of this master thesis is to examine the structure and functionality of the various protocols in the IPsec suite. We will also examine its benefits and drawbacks and provide a brief description of TCP/IP and some of its associated security issues.

To be able to understand the new features introduced with IPsec, and its major advantages, it is significant to have an idea about how it works, and which components that are included in the suite. To make this easier we have included a brief description of TCP/IP and network security in general. When you understand why security is needed it is time to investigate the main features of IPsec and the structure and functionality of the protocols included. Thus, our first question is:

1. *What are the security services provided by IPsec?*

When you understand how IPsec works, and have got an idea of its main components and their technical capabilities you also need to know the advantages and disadvantages, to be able to decide whether it is a good or bad choice for a security solution. This is why we ask our second question:

2. *What are the advantages and disadvantages of IPsec?*

After examining the structure and main components of IPsec, its advantages and disadvantages, you may wonder what makes IPsec so special? There are several other security techniques available that are based on similar ideas, using the same encryption algorithms. The third question is similar to the second one, but its purpose is to point out what is specific to IPsec. Thus our third and final question is:

3. *What makes IPsec different from existing security techniques?*

## 1.3 Restrictions

Network security is a very complex and constantly changing area. We started to read about IPsec only, but were forced to widen our focus since there was a lot of necessary pre-understanding to be able to fully get the idea of IPsec. IPsec is a coming security standard and uses various forms of encryption, key exchange methods and hash algorithms to perform its tasks. We found it impossible to describe every part of this huge area, and decided to try to cover the most essential parts to make it understandable to the reader. To make this a little bit easier we also included some information about the most common algorithms used in IPsec.

## 1.4 Target group

This is a university master thesis, and hence it is written for an academic audience. But due to its rather technological material it may even be of interest for all kinds of people working with, or interested in network security and the coming security standard for IPv6, IPsec.

## 1.5 Disposition

The structure of this master thesis is as follows:

In chapter 1, **Introduction**, we give a brief description of what the thesis is about, its purpose and for what audience it is written.

In chapter 2, **Method**, we describe the method used to gather information about the problem area. We also give a brief description of the IETF.

In chapter 3, **Internet Background**, we describe how the Internet has emerged from a military network to be publicly available. We also present a timewindow to illustrate its development.

In chapter 4, **Overview of TCP/IP**, we describe the structure and functionality of the OSI-model and the TCP/IP suite.

In chapter 5, **Network Security**, we give an introduction to network security and illustrate some of the various problems that occur when computers are connected in network environments.

In chapter 6, **Encryption**, we present a summary of some of the most widely used encryption algorithms and explain the differences between symmetric and asymmetric encryption.

In chapter 7, **IP Security**, we describe the structure and functionality of IPsec, and its main components; Security Association (SA), Authentication Header (AH) and Encapsulating Security Payload (ESP).

In chapter 8, **Key Management and Key Exchange**, we describe the ideas behind key management and key exchange. We also present the various protocols used to form the base for the Internet Key Exchange.

In chapter 9, **Internet Key Exchange**, we give a good description of the Internet Key Exchange (IKE); the key exchange protocol set to default for IPsec.

In chapter 10, **Discussion**, we present our thoughts and opinions about IPsec. Why it is to be used at all, its advantages and disadvantages. We would also discuss security in general and finally draw conclusions.

# 2 Method

The method used during the writing of this thesis is a pure literature study, although in various forms. We started out by searching for suitable books, but found few. We continued by browsing the web for articles regarding IPsec, encryption, TCP/IP and network security. Finally we signed up for the available IPsec mailing lists found on the IETF web site to be able to follow the ongoing discussions and have the opportunity to ask questions. “Rapporter och uppsatser” [2] was used for guidance during the writing of the thesis.

IPsec is designed to be used with the Internet Protocol (IP), so it seemed to be a good start to begin our work by reading manuals about the OSI-model and TCP/IP structure and functionality. To understand the design of IPsec it is necessary to have a good knowledge of how IP works, how packets are built and processed and so forth. We continued by reading various IPsec related RFCs and articles, and then material regarding encryption and general computer and network security.

IPsec is a rather new phenomenon, hence there are quite few books written about it, and the ones we found were merely collections of the IETF RFCs. Since IPsec is developed by an IETF working group, the main source of information regarding IPsec has been the IETF RFC archive found on the organization’s web-site (http://www.ietf.org/rfc.html). This means that all original information about IPsec is in electronic form. For information about the IETF and a definition of the term RFC see below.

## 2.1 IETF

The Internet Engineering Task Force (IETF) is not a company, but a large open international community of network designers, operators, vendors, users, and researchers concerned with the Internet and the technology used on it. It is loosely self-organized and open to any interested individual. Participation may be by online contribution, attendance at face to face sessions, or both. Anyone from the Internet community who has the time and interest is urged to participate in IETF meetings and any of its online working group discussions. Participation is by individual technical contributors, rather than by formal representatives of organizations. [9]

The IETF group was formally formed from the IAB (Internet Activities Board) in 1986. The organization consists of a network of designers, operators, vendors, and researchers who make technical and other contributions to the engineering and evolution of the Internet and its technologies. [9]

The actual work of the IETF is done in working groups, organized into areas (e.g. routing, security, transport etc.). Much of the work is handled via mailing lists since the IETF only holds three meetings per year. The working groups are grouped into area groups managed by an Area Director (AD).

The IETF meeting is not a conference, although there are technical presentations. The IETF is not a traditional standards organization, although many specifications are produced that become standards. There is no membership in the IETF. Anyone may register for and attend any meeting. The closest thing there is to being an IETF member is being on the IETF or working group mailing lists, this is where the best information about current IETF activities and focus can be found.

The Internet Assigned Numbers Authority (IANA) is the central coordinator for the assignment of unique parameter values for Internet protocols. The IANA is chartered by the Internet Society (ISOC) to act as the clearinghouse to assign and coordinate the use of numerous Internet protocol parameters. It is IANA who is in charge of all unique parameters on the Internet such as IP addresses.

## 2.2 Description of a Working Group

The primary activities of the IETF are performed by committees known as working groups. There are currently more than 100 working groups. Working groups tend to have a narrow focus and a lifetime bounded by the completion of a specific set of tasks, although there are exceptions. The IETF working groups are collected together into areas, where each have a separate focus. For example, the security area deals with the development of security-related technology. There are currently eight areas in the IETF but the number changes from time to time. [9]

In many areas, the Area Directors have formed an advisory group or directorate. These comprise experienced members of the IETF and the technical community represented by the area. The specific name and the details of the role for each group differ from area to area, but the primary intent is that these groups assist the Area Director with the review of specifications produced in the area. [9]

The rapid advances in communication technology have increased the need for security in the Internet. The purpose of the IP Security Protocol Working Group is to develop mechanisms to protect client protocols of IP. They aim to develop a security protocol in the Network Layer to provide cryptographic security services that will support combinations of authentication, integrity, access control, and confidentiality. [8]

## 2.3 Request for Comments and Internet Drafts

Originally, Request for Comments (RFCs) were just what the name implies and sent as messages between the ARPANET architects about how to resolve certain problems. Over the years, RFCs became more formal and reached the point where they were being cited as standards, even when they were not.

Internet Drafts are working documents of the IETF. Any group or individual may submit a document for distribution as an Internet Draft. These documents are valid for six months, and may be updated, replaced or obsoleted at any time. Guidelines require that an expiration date appear on every page of an Internet Draft. It is not appropriate to use Internet Drafts as reference material or to cite them, other than as "working drafts" or "work in progress". [9]

Both RFCs and Internet Drafts are often very technical documents written by experts and hence quite difficult to read and understand. They often require the reader to possess a great deal of knowledge about the subject, which in most cases (luckily enough) is available in another RFC or Internet Draft. To be compatible with most computer platforms, the documents are written in pure ASCII, even the pictures, which does not make it any easier to read them.

# 3 Internet Background

Most of the information in this chapter is taken from [4]**.** In August 1969, the Information Processing Techniques Office (IPTO), part of the Defense Advanced Research Projects Agency (DARPA) funded a research and development project to study techniques for providing a robust, reliable, vendor-independent packet switched network. The ARPANET, as it came to be called, was intended to form a part of the structure for the US armed forces during the cold war. Its decentralized design came from the urge for the network to survive nuclear attacks. If one node was taken out, the traffic through that node was automatically redirected through other nodes, which would result in the data reaching its destination anyway. Many of the techniques used in data communications of today were developed in the ARPANET.

The experiment with the ARPANET was successful and many of the organizations attached to it began to use it in their daily data communications. Six years later, in 1975, the ARPANET converted from its original experimental network to form an operational network. The responsibility for the administration of the network was given to the Defense Communications Agency (DCA). The development of the ARPANET however did not stop just because it was being used as an operational network. The basic TCP/IP protocols we are using today were developed after the network was operational.

The American military adopted the TCP/IP protocols as standards in 1983 and all hosts connected to the network were required to convert to the new protocols. To ease this conversion, DARPA funded Bolt, Beranek and Newman (BBN) to implement TCP/IP in Berkeley UNIX (BSD), and thus began the joining of UNIX and TCP/IP. About the time that TCP/IP was adopted as a standard, the term *Internet* came into common usage.

In 1983, the old ARPANET was divided into MILNET, part of the Defense Data Network (DDN), and a new and smaller ARPANET [3]. The term *Internet* was used to refer to the entire network, which consisted of both MILNET and ARPANET. In 1990, the ARPANET ceased to formally exist, but the Internet is today larger than ever and encompasses many networks worldwide.

A sign of the network's success is the confusion that surrounds the term *Internet.* Originally it was used only as a name of the network built upon the Internet Protocol. Now *internet* is a generic term used to refer to an entire class of networks. An internet (lowercase i) is any collection of separate physical networks, interconnected by a common protocol, to form a single logical network. The Internet (uppercase I) is the worldwide collection of interconnected networks, which grew out of the original ARPANET, that uses Internet Protocol (IP) to link the various physical networks into a single logical network.

Because TCP/IP is required for Internet connection, the large number of new organizations recently added to the Internet has spurred interest in TCP/IP. As more organizations become familiar with TCP/IP, they see that its power can be applied in other network applications. In the UNIX community, the Internet protocols are often used for local area networking, even when the local network is not connected to the larger Internet.

In 1991, the National Science Foundation (NSF) lifted the restrictions on commercial use of the Internet. With more unknown people connected, and increased commercial use of the Internet, the need for security mechanisms providing access control services and communication security services increased. As TCP/IP was designed with focus on good networking features and functionality, security never became a strong suite.

During the past decade, reports of network-based attacks and exploitations of bugs and design limitations have grown dramatically [28]. More recently, the use and proliferation of downloadable, executable content, such as those provided by Java applets and Active X controls, have opened even more possibilities to attack networked computer systems and Internet sites.

## 3.1 Time Window for Internet 1966-2000

| | |
|---|---|
| 1966 | The first plans for ARPANET are made |
| 1974 | Vinton Cerf and Robert Kahn publishes the specifications for TCP, Transmission Control Protocol |
| 1981 | France Télécom releases Minitel all over France |
| 1983 | EARN, European Academic and Research Network is initiated |
| 1984 | The Domain Name System, DNS, is introduced and the number of host computers rises above 1,000. The term Cyberspace is created by WilliamGibson in the novel “Neuromancer”. |
| 1988 | New countries, among others Sweden, SE, join the NSFNET |
| 1989 | RIPE, Reseaux IP Europeens, is formed to coordinate an all European network |
| 1990 | The ARPANET ceases to exist |
| 1992 | The Internet Society, ISOC, is formed. The number of host computers rises above 1,000,000. Jean Armour Polly coins the term “surfing the Internet”. |

| | |
|---|---|
| 1993 | In February, Mosaic, the world's first wide-spread browser is finished. In April, CERN decides that the www-technology shall be free for everyone<br><br>The ARPANET/Internet celebrates its 25th anniversary and the first online shopping centers are put into business |
| 1994 | The registration of domains is no longer free of charge |
| 1995 | Internet-telephony is becoming more common and American phone-operators demand that the congress stops the technology |
| 1996 | La Fête de L'Internet takes place in France for the first time. |
| 1998 | It is now possible for Americans to download postal stamps for use with the US postal service. |
| 1999 | The Internet celebrates its 30th anniversary. 153.2 million people are connected to the Internet (Jan. 1999) |
| 2000 | IDC predicts that there will be 243 million Internet users. |

[29]

# 4 Overview of OSI and TCP/IP

This section provides an overview of the general OSI reference model, which is used in almost every discussion regarding computer networking. After a brief description of its structure and various layers, we introduce TCP / IP, its layers and its most important protocol, the Internet Protocol (IP).

## 4.1 The OSI Reference Model

The International Standards Organization (ISO) has developed an architectural model called Open Systems Interconnect Reference Model, or just OSI. Its purpose is to provide an easier way to describe the structure and functionality of data communication protocols. The terms defined and used in this model are widely known and used by the data communications community. It is so well known that discussing data communications without it is difficult. [4]

The OSI Reference Model contains seven different layers, stacked on top of each other. Each layer in the layer stack represents a function, which is to be performed when data is transferred between cooperating applications across an intervening network [4]. The protocols are like a pile of building blocks and because of this appearance, the structure is often called a *stack* or *protocol stack*.

A layer in the OSI model does not define a single protocol, but defines a data communications function that may be performed by any number of protocols. Therefore, each layer may contain more than one protocol, where each of these protocols provides a service suitable to the function of that layer [5]. For example, a file transfer protocol (FTP) and an electronic mail protocol (SMTP) both provide user services, and both are part of the Application Layer.

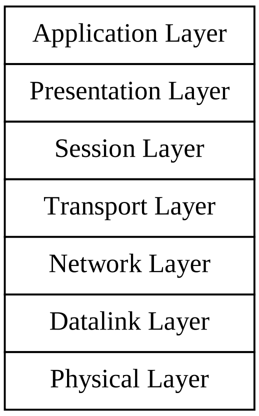

*Figure 4.1: The OSI model.*

Every protocol communicates with its so-called *peer*. A peer is an implementation of the same protocol on the same layer, but on a remote system. For example, the local mail protocol is the peer of a remote mail protocol. This communication must be standardized if it is to be successful. Each protocol is only concerned with communicating to its peer and not with the layers above or below it. Therefore there must also be an agreement on how to pass data between the layers on a single computer. This is because every layer is just involved in sending data from a local application to the equivalent application on the remote computer. The layers above, or the *upper* layers, rely on the *lower* layers to transfer the data over the network. Data is passed down the stack from one layer to the next, until it is transmitted over the network (the physical cable) by the Physical Layer protocols. Then at the remote end, the data is passed back up the stack to finally reach the receiving application. [4]

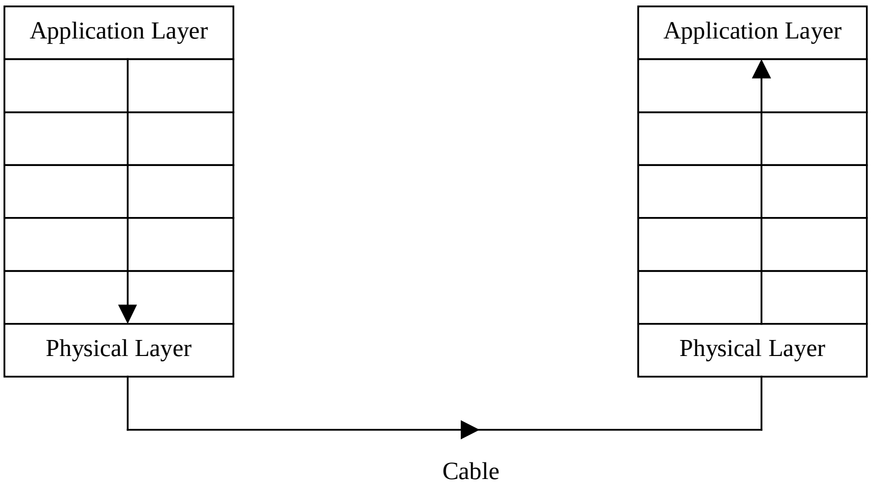

*Figure 4.2: Data passing from one host to another.*

The individual layers do not need to know how the layers above and below them function; they only need to know how the data between them need to be passed. This technology, with isolation of the network communications functions in different layers, minimizes the impact that technological change has on the entire protocol suite. This provides that new applications can be added without changing the physical network, and also that new network hardware can be installed without rewriting the application software [4].

Although the OSI model is very useful, the TCP/IP protocols do not exactly match its structure. When discussing TCP/IP, the OSI layers are used in the following way [4]:

### 4.1.1 Application Layer

The Application Layer is located at the top of the protocol hierarchy. This is where the user-accessed network processes reside. Any network process that occurs above the Transport Layer is a TCP/IP application.

### 4.1.2 Presentation Layer

This layer provides standard data presentation rules for how data is represented. It is required for cooperating applications to be able exchange data.

### 4.1.3 Session Layer

The Session Layer is, as the Presentation Layer, not identifiable as a separate layer in the TCP/IP protocol hierarchy. This layer manages the sessions, or connections, between the cooperating applications. In TCP/IP, this function largely occurs in the Transport Layer, located below the Session Layer. For TCP/IP, the terms *socket* and *port* are used to describe the path over which cooperating applications communicate.

### 4.1.4 Transport Layer

The Transport Layer guarantees that the receiver gets the data exactly as it was sent originally. This function is performed by the Transmission Control Protocol (TCP) in the TCP/IP hierarchy. TCP/IP does however offer a second Transport Layer service, the User Datagram Protocol (UDP), which does not perform the end-to-end reliability checks as is required in TCP.

### 4.1.5 Network Layer

This layer manages connections across the network and isolates the upper layer protocols from the details of the underlying network. In the TCP/IP hierarchy it is the Internet Protocol (IP), which does this isolation.

### 4.1.6 Data Link Layer

The Data Link Layer handles the delivery of data across the underlying physical network. In most cases IP can make use of existing data link protocols.

### 4.1.7 Physical Layer

This layer defines the characteristics of the hardware needed to carry the data transmission signal. This includes things such as voltage levels, number and location of interface pins. TCP/IP does not define physical standards and instead makes use of existing standards.

## 4.2 TCP/IP Structure

To understand TCP/IP fully, a more closely matching architectural model must be used. The layered OSI model can somewhat be used, but there is no universal agreement about how to describe TCP/IP with layers. The layered TCP/IP model is composed with mostly three to five layers (the OSI model has seven). The four-layered model below is based on the description mentioned in “TCP/IP Illustrated” [5]. To make it even more complicated, the Network Layer is sometimes referred to as the *Internet Layer*, and the Link Layer is sometimes called the *Network Access Layer*.

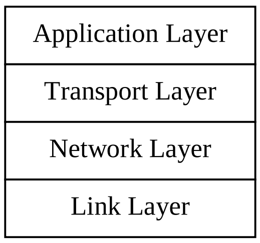

*Figure 4.3: TCP/IP Structure.*

The data is passed down the stack when it is sent to the network and up the stack when it is received from the network, just as in the OSI model. Each layer adds control information as the data is passed down the stack to the underlying network. This control information is called *header* and is placed in front of the data before it is transmitted to the lower layer and it is used to ensure proper delivery. This occurs in every layer and the addition of the delivery information is called *encapsulation*. When the data is received, the layer strips off the layer specific header before the data is passed on to the layer above.

Each layer has its own independent data structures and a layer is unaware of the data structures used by the layers above and below it. In reality however, the data structures are designed to be compatible with data structures on other layers, but the terminology often differs.

### *4.2.1 Application Layer*

At the top of the TCP/IP protocol stack is the Application Layer. This layer includes all processes that use the Transport Layer protocols to deliver data. There are many application protocols and most of them provide user services and new services are added to this layer all the time. Some of the most well known application protocols used today are TELNET, SMTP and FTP. [4]

### 4.2.2 Transport Layer

The layer above the Network Layer is the host-to-host Transport Layer, or just Transport Layer for short. The two most important protocols in the Transport Layer are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). TCP provides a service for reliable data delivery with end-to-end error detection and correction, which IP does not have. UDP on the other hand provides a connectionless datagram delivery service. Both these protocols deliver data between the Application Layer and the Network Layer and the programmers can choose whichever service to choose. [4]

### 4.2.3 Network Layer

The Internet Protocol is the heart of TCP/IP and is also the most important protocol in the Network Layer, which also is called the Internet Layer. IP provides the basic packet delivery on which TCP/IP networks are built. The protocols in the layers above and below this layer all use the Internet Protocol to deliver data. All TCP/IP data flows through IP, incoming and outgoing, regardless of its final destination.

#### 4.2.3.1 Internet Protocol

The Internet Protocol is the building block of the Internet. Its functions include the following:

- Defining the datagram, which is the basic unit of transmission in the Internet.
- Defining the Internet addressing scheme.
- Moving data between the Link Layer and the host-to-host Transport Layer.
- Routing datagrams to remote hosts.
- Performing fragmentation and reassembly of datagrams.

Since IP is a so-called connectionless protocol it does not exchange control information (called a handshake) to establish an end-to-end connection before transmitting the data. The connection-oriented protocols exchange control information before transmission in order to verify that the remote system is ready to receive data. When the handshake is successful, the systems are said to have established a connection. If a connection-oriented service is needed other layers' protocols are invoked to provide this. [4]

IP is often said to be an unreliable protocol, since it does not provide error detection and error recovery. Instead, IP relies on other layers to provide this.

IP can however be relied on to deliver data to the connected network, but it does not check whether the data was correctly received. This check is provided by protocols in other layers when needed. [5]

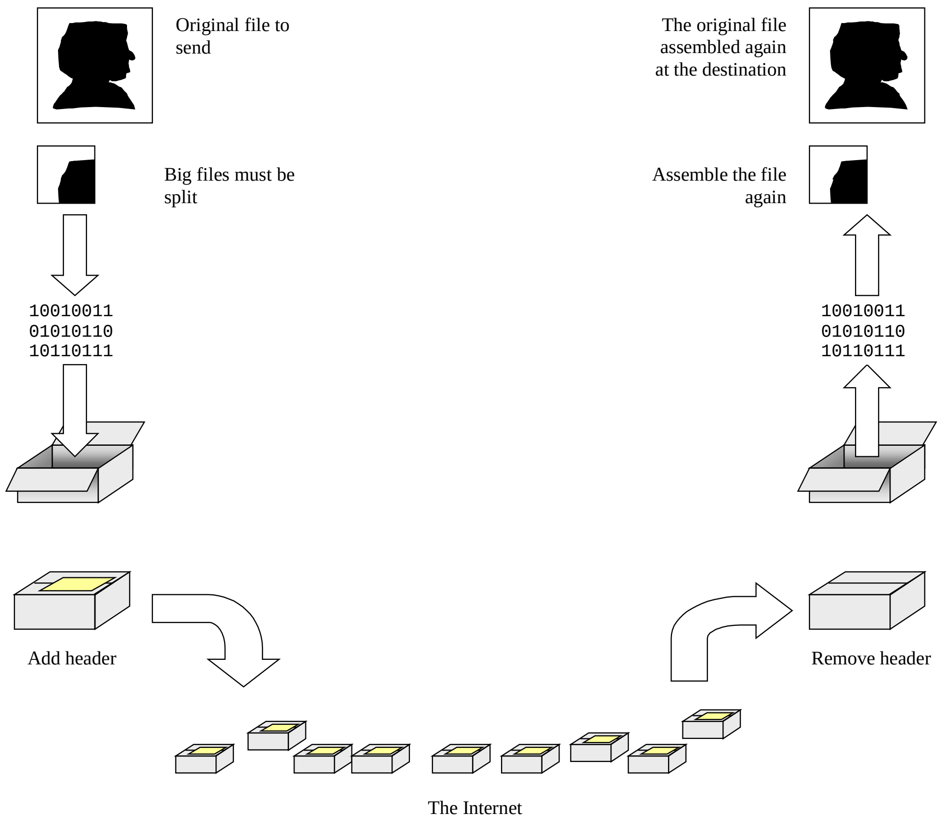

*Figure 4.4: Basic packet switched network.*

#### 4.2.3.2 The Datagram

The TCP/IP protocols were originally built to transmit data over a packet switched network. A *packet* is a block of data with a sort of label on it carrying the information necessary to deliver it to its destination. A packet switching network uses the addressing information to switch the packets from one physical network to another in order to bring them to their final destination. The packets travel independently of other packets on the network, just like ordinary letters in the mail.

The *datagram* is the packet format defined by the Internet Protocol. The first five or six 32-bit words of the datagram are control information. This is called the packet *header* and it is by default five words long (20 bytes). The sixth word is optional. The header contains the necessary information to deliver the packet to its destination.

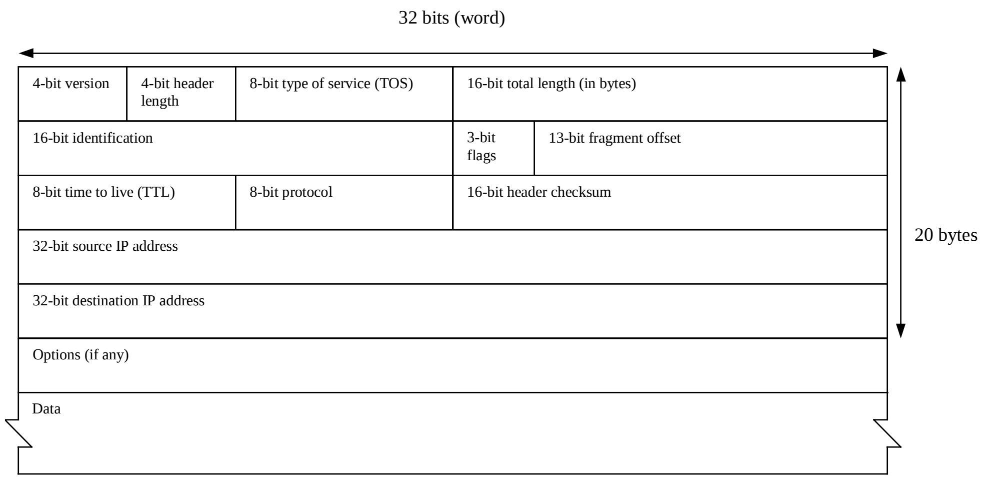

*Figure 4.5: The IP header.*

The Internet Protocol delivers the datagram by checking the Destination Address in the fifth word of the header. The 32-bit IP Destination Address identifies the destination network or a specific host on that network. If the destination address is a local host, the packet is delivered directly to the destination. If not, the packet is passed to a gateway whose task is to switch packets between different physical networks, e.g. a corporate network and the Internet. [4]

### 4.2.4 Link Layer

The Link Layer is the lowest layer of the TCP/IP hierarchy and is also known as the Network Access Layer. The protocols provide means for the system to deliver data to the other devices on a directly attached network [3]. The Link Layer must know the details of the underlying network in order to format the data to make it compliant with the network constraints. The TCP/IP Link Layer can encompass the functions of all three lower layers of the OSI reference model. As new hardware technologies emerge, new Link Layer protocols has to be developed in order to be able to use the new hardware. Consequently, there is a Link protocol for each physical network standard. [4]

# 5 Network Security

In this section we will discuss computer security in general and network security in particular. We present various threats for a networking environment as well as some threats that are specific to networks based on IP, such as spoofing and sniffing. We conclude by discussing the issue of application specific security and the advantages and disadvantages of implementing security at different layers in the stack.

## 5.1 Security in General

Eavesdropping on communications over data networks is easier than to tap a normal telephone conversation. The chance exists even if you think that no one is interested in what you are doing, you cannot be entirely sure. Any information (e-mail, web traffic, TELNET etc.) passing over a network, sensitive or not, can be eavesdropped at. Unless you have taken precautions when you access any network service, your password or confidential information may be stolen. It may then be used to gain illicit access to systems you have access to. This could result in your bank accounts being emptied or other issues, not really financially related, being abused. [20]

Many network services involve a remote login where the user is prompted for his or her account ID and password. If this information is sent over the network without encryption, which is often the case, the message can be intercepted and read by others [20]. The risk is there when you are using programs to log in over a network. Many popular programs used to log in to services or to transfer files (such as TELNET and FTP) send your username and password and then your data over the network without first encrypting them.

Until recently, it has been far too complicated and expensive for home systems and small businesses to employ secure login systems. However, an increasing number of products provide this security feature without fancy hardware, using cryptographic techniques. An example of such a technique is Secure Shell (SSH), which is both freely and commercially available for a variety of platforms. Many products (including SSH-based ones) encrypt the data before it is passed over the network. [20]

## 5.2 Security Threats in the Network Environment

To know that you are using a secure network, you want to be confident of three things [21]:

- That the person you are communicating with really is that person.
- That no one can eavesdrop on your communication.
- That the communication you have received has not been altered in any way during transmission.

Information security has traditionally been considered to have three fundamental objectives. This is commonly referred to as the CIA of computer security [30]:

- *Confidentiality* involves ensuring that the information is not revealed to unauthorized persons. The value of much data relies on it being kept secret from prying eyes.
- *Integrity* ensures consistency of the data, preventing that unauthorized parties create, alter or destroy data.
- *Availability* ensures that legitimate users are not denied access to information and resources that they should be granted.

There is a fourth objective called *legitimate use*, which ensures that resources are not used by unauthorized persons or in unauthorized ways. It is easy to understand that such a person would probably end up in a position from which further violations were possible [1]. Unfortunately, the architecture of modern large IP-based networks makes these four objectives difficult to ensure.

A security threat is a condition or event with potential to harm network resources in the form of destruction, disclosure, fraud etc. Network security threats include impersonation, eavesdropping, denial-of-service, packet replay and packet modification. Network threats encountered can be classified in one of the three following categories, fundamental threats, primary enabling threats and underlying threats.

### 5.2.1 Fundamental Threats

The fundamental threats directly reflect the four security objectives described above. *Information leakage* involves revealing information to an unauthorized person or entity. This might involve direct attacks, such as eavesdropping or wiretapping, or subtler types of information observation. Any occurrence of unauthorized creation, modification, or destruction of data causes *Integrity violation*. When the legitimate access to information or other resources are deliberately impeded, it is said to cause *denial-of-service*. This might involve, for example, making a resource unavailable to legitimate users by heavily loading the network with illegitimate, unsuccessful access attempts. *Illegitimate use* is when a resource is used by an unauthorized person or in an unauthorized way. An example might involve an intruder penetrating a computer system, using that system either as the basis of theft of telecommunications services, or as a point for penetrating another system.

### 5.2.2 *Primary Enabling Threats*

These threats may give the intruder unauthorized access to a system. The primary enabling threats consist of *penetration* and *planting* threats. The main penetration threats are:

#### 5.2.2.1 Masquerade

An entity (person or system) pretends to be a different one from what he really is. This is the most common way of penetrating a security barrier.

#### 5.2.2.2 Bypassing Controls

An attacker exploits system flaws or security weaknesses, in order to acquire unauthorized rights or privileges.

#### 5.2.2.3 Authorization Violation

A person authorized to use a system for one purpose uses it for another, unauthorized purpose. This is also known as an *insider* threat.

#### 5.2.2.4 Physical Intrusion

An intruder gains access by going around the set up physical barriers (gateways etc.).

The planting threats give access to a system by built programs or subroutines:

#### 5.2.2.5 Trojan Horse

A Trojan horse is a piece of software that contains an invisible part which, when executed, might open security barriers. These are, as the name implies, something fun or interesting on the outside, but conceals the Trojan inside.

#### 5.2.2.6 Trapdoor

This is a built in feature in a system or system component that by providing specific input data allows the security policy to be violated. An example is a login system, which allows bypass of the usual password checks when a specific username is given.

#### 5.2.2.7 Service Spoofing

When a false system or system component is used to trick legitimate users or systems into voluntarily giving up sensitive information, such as username and password, it is called service spoofing.

### 5.2.3 *Underlying Threats*

Underlying threats such as *eavesdropping*, where information is revealed by monitoring communications, and traffic analysis where information is leaked through observation of communication patterns, may enable the more fundamental threats. Some other underlying threats are *theft* where a security-critical item such as an ID card is stolen, *indiscretions by personnel* where an authorized person reveals information for money or by carelessness.

## 5.3 IP Security Threats

The strength of IP is also its weakness. The way IP routes packets makes large IP networks vulnerable to a range of security risks [21]:

- *Spoofing*, in which one machine on the network masquerades as another.
- *Sniffing*, in which an eavesdropper listens in on transmission between two other parties.
- *Session hijacking*, in which a sophisticated attacker employing both these mentioned techniques takes over an established communications session and masquerades as one of the two communicating parties.

### 5.3.1 *Spoofing*

The first difficulty IP networks pose is that it is hard to know from where information really originates. A technique called IP spoofing takes advantage of this weakness. The spoofing technique is based on the way in which IP packets are made up. The difficulty with this from a security perspective, is that source IP addresses in IP packets are easily changed. A spoofing attack makes a packet coming from one machine appear to come from somewhere else. [21]

### 5.3.2 *Sniffing*

Sniffing is a technique that is possible in Ethernet-based IP networks. Ethernet LANs make up a large part of most networks, since the technology has the advantages of being cheap, universally available, well understood and easy to expand. It has the disadvantage of making sniffing easy.

In most Ethernet-based LANs, packets are available to every node in the network, since it is a so-called broadcast network. Conventionally, each node's Network Interface Card (NIC) only listens and responds to packets specifically addressed to it. It is relatively easy however, to put an Ethernet NIC on what is called promiscuous mode, meaning it can collect every packet that passes on the wire. There is no way to detect such a NIC, because it does not do anything to the packets when it picks them up. [21]

A special type of software, called a *sniffer*, can take advantage of this feature of the Ethernet technology. Such a tool can record all the network traffic passing by. A sniffer is a valuable tool for any network technician, but in the hands of someone who wants to listen in on sensitive communications, a sniffer is a powerful eavesdropping tool. [21]

### 5.3.3 Session Hijacking

The fact that you have identified the person with whom you are talking once, does not mean that you can depend on IP to ensure that it will be the same person through the rest of the session. You need a scheme that authenticates the data's source throughout the transmission, since someone may take the other part's place, without you really knowing. [21]

### 5.3.4 The Man-in-the-middle

The most obvious solution to the problem of IP security threats is the use of encryption technologies that conceal and authenticate the data passed in IP packets. But there are complications doing this.

To use encryption, you first have to exchange encryption keys. These are used with encryption algorithms to scramble and to unscramble data. Exchanging those keys unprotected might easily defeat the whole purpose, since they could be intercepted and open up yet another attack. This is referred to as the man-in-the-middle attack [24]. A sophisticated attacker could actually work his way into such a key exchange, in a system that left has the way open. Early in the process, he could plant his own key, so that, while you believed that you were communicating with one party's key, you would actually be using a key known to the man-in-the-middle. [21]

## 5.4 Security Services

In the computer communications context, the main security safeguards are known as *security services*. It defines five generic security services [7]:

- *Authentication* services deal with proof of identity. The ISO security architecture defines two forms of authentication: *peer entity authentication* that is used between peers in a system to prove that an entity is who it claims to be, and *data origin authentication*, which is used to verify the source of a given block of data.

- *Access control* services, as the name implies, offer a means of controlling access to a given system or system resource. This service generally makes use of peer entity authentication (to authenticate the entity requesting resource usage), then applies some rule-based mechanism to allow or deny access to the requested resource. It may also require the use of other security services (e.g., confidentiality, data integrity, or non-repudiation, discussed below) when invoked remotely. Access control provides protection of computing and networking resources against unauthorized access (authorization violation and denial-of-service).

- *Confidentiality* services are intended to protect information against unauthorized disclosure.

- The *data integrity services* are designed to ensure that data is not altered during storage or transmission. Altered messages may cause people, or systems, to take inappropriate actions. These services also protect against unauthorized modification, insertion and replay of data, which may cause integrity violation and denial-of-service.

- *Non-repudiation* services are designed to ensure that participants in a communication or transaction cannot later repudiate that communication or transaction. This can be done in either one of two ways. First with *proof of origin* providing the recipient with information which makes it impossible for the sender to later claim not to have sent the data to the recipient. The other way is *proof of delivery*, which makes it impossible for the recipient to later deny receipt of the data. Non-repudiation often requires the involvement of a trusted third party or cryptographic techniques.

## 5.5 Application Specific Security

Today there exist techniques to secure communications over the Internet, but most of them are made for a specific software application. Generally, these techniques employ powerful new encryption algorithms to overcome to security problems.

Examples of application specific security techniques:

- Pretty Good Privacy (PGP)/Web-of-Trust technology encrypts email.

- Secure Socket Layer (SSL) is a browser-based authentication and encryption between the browser and the server that protects commercial web traffic.

While these techniques have their respective strengths and niches, they are limited to specific uses. This does not address the challenges faced by large enterprises and the average Internet Service Provider (ISP) that may never know precisely what applications may be running tomorrow over the networks they are building today.

## 5.6 IP Security – At Which Layer?

There are different protocols designed to secure the traffic at various layers in the network. Exactly how this is done depends on the security requirements of the application and it is up to the user to decide where in the stack security should be implemented. Irrespective of this, the following basic services have to be provided [3]:

- Key management.
- Confidentiality.
- Non-repudiation.
- Integrity/authentication.
- Authorization.

It is possible to provide some or all of the security services mentioned above, it depends on where in the stack security is implemented. Sometimes, it makes sense to provide some capabilities on one layer and other capabilities at another. Listed below are some advantages and disadvantages of providing security at the various layers in the stack:

### *5.6.1 Application Layer*

Application Layer security has to be implemented in the end hosts. Providing security at the Application Layer has the following advantages:

- Executing in the context of the user enables easy access to user information such as private keys.
- Complete access to the data the user wants to protect. This simplifies the task of providing services such as non-repudiation.
- An application can be executed without having to depend on the operating system to provide these services. Normally applications have no way of controlling what gets implemented in the operating system.

- Applications understand the data and can provide appropriate security.

The drawback of implementing security in the Application Layer is that the security mechanisms have to be designed specifically for each application. Existing applications have to be modified to provide security, and as each application defines its own security mechanisms, there is a greater risk of someone making a mistake and hence opening up security holes for attacks.

To implement security in an application, you integrate the application with a system providing the desired security mechanisms. Examples of such systems are PGP and SSL. These systems are Application Layer protocols that are capable of key negotiation and other security services. The applications are enhanced to call into these systems to make use of their security mechanisms. One example is the e-mail clients that use PGP to provide e-mail security. In this case, the e-mail clients are extended with the following capabilities:

- Ability to look up public keys in a local database that corresponds to a particular user.

- Ability to provide security services such as encryption and decryption, non-repudiation, and authentication for e-mail messages.

When security needs are specific and lower layers can not be depended on to provide the security services, applications should design their own security mechanisms. One example of this is non-repudiation; it is difficult for lower layers to provide non-repudiation services, as they do not have access to the data.

### 5.6.2 Transport Layer

Providing security at the Transport Layer has a definite advantage over the Application Layer security, as it does not require modifications to each application. Security services are seamlessly provided to existing applications. However, the issue of obtaining the user context makes it complicated. In order to provide user specific services, assumptions are made that a single user is using the system. Like Application Layer security, security on the Transport Layer can only be implemented on an end system.

Transport Layer Security (TLS) is protocol specific. It is a protocol that provides security services such as authentication, integrity, and confidentiality on top of TCP. As the security mechanism is transport protocol specific, security services such as key management may be duplicated for each transport protocol.

The World Wide Web currently provides security services using TLS. However, if security services were implemented at the Network Layer, they can replace the services provided by TLS. Another limitation of TLS as it is currently defined is that the applications still need modification to request the needed security services from the Transport Layer.

### 5.6.3 Network Layer

Implementing security at the Network Layer has many advantages. First, the key negotiation overheads decrease considerably. This is because multiple transport protocols and applications can share the key management infrastructure provided by the Network Layer. If security is implemented at lower layers, fewer applications require modifications. In contrast, if security is implemented at higher layers, each application has to design its own security mechanism. Also, security is provided seamlessly for any transport protocol.

The main disadvantage of Network Layer security is the difficulty in handling issues such as non-repudiation of data. This is better handled in higher layers. It is more difficult to exercise control on per user basis on a multi-user machine when security is implemented at the Network Layer. However, mechanisms can be provided to perform user-based security on end hosts. On the routers, there is no context of user and this problem does not arise.

### 5.6.4 Data Link Layer

If there is a dedicated link between two hosts or routers and all the traffic needs to be encrypted, hardware devices can be used for encryption. The primary advantage of this solution is speed. However, this alternative is not scalable and works well only on dedicated links. Moreover, the two nodes involved in communication have to be physically connected.

This model is useful in Automatic Teller Machines (ATMs) where all the machines are connected via dedicated links to a central office. If ATMs were connected to an IP network instead of dedicated secure links, the Data Link Layer security would not suffice and the implementation of security would have to move up one layer.

# 6 Encryption

Cryptography is formally the art of encoding data in a way that only the intended recipient can decode it, and know that the message is authentic and unchanged. Strong encryption is not a technical standard, it means that your encryption cannot be broken by current known methods within feasible time without the data being outdated. It can be used to protect your sensitive data against organized crime, government and multinational corporations, all instances with virtually unlimited resources. [23]

Strong encryption brings many possible applications into daily life. Different applications that require privacy, trust, and access control should all use strong encryption methods when possible. Applications include things like transfer of electronic money, secure communications, passwords, and many more. It is in people's own interest that different legal/medical/personal data about their person stay confidential to the instances that have a permit to keep the databases. [23]

## 6.1 Different Types of Cryptosystems

Some cryptographic methods rely on the secrecy of the algorithms used in the cipher, Security through Obscurity (see below). These ciphers are only of historical interest and are not adequate for a real-world situation. However, this does not mean that there are not companies still using the method.

Most modern encryption algorithms use a key to control the encryption and decryption. The message can only be decrypted if the key matches the one that was used to encrypt it. The key used for decryption can be different from the one used in encryption, and this divides the algorithms in symmetric (secret-key) and asymmetric (public-key) classes.

### *6.1.1 Security Through Obscurity*

Many software manufacturers hide bugs that impair the security of programs, or even entire operating systems, without knowing whether some outsider has already found and exploited these bugs [25]. Hiding account passwords in binary files or scripts with the presumption that nobody will ever find it, is another prime case of Security through Obscurity (STO) [26].

The only proper course for a software manufacturer is to issue a software update as soon as possible after a problem is found, and to inform all customers that the update must be installed to correct an existing security problem. Until more manufacturers understand that Security through Obscurity is a fallacy, you should consider that popular computer operating systems, applications, and cryptography programs are presently compromised [27]. Do not rely on the security features of these systems.

One exception to the above are Open Source operating systems such as Linux and FreeBSD, and cryptography programs such as GNU Privacy Guard. Because the developers of these systems publish all of their source code for others to read, they can not rely on Security through Obscurity. The publication of source code actually improves security because the program or operating system can be peer-reviewed by anyone who cares to read it. Many security bugs that are overlooked in other operating systems have been caught and repaired in Linux, because of its extensive peer-review process.

### *6.1.2 Symmetric Algorithms*

Symmetric algorithms also called secret-key algorithms, are the more traditional form of cryptography and use the same key for both encryption and decryption. The key is not to be leaked to outside enemies, hence the name. Furthermore, it must be sufficiently random and should be changed often. Different symmetric algorithms use different key lengths, usually a longer key results in higher security. Secret-key cryptography not only deals with encryption, but it also deals with authentication [24].

Symmetric algorithms can be further divided into two categories: stream ciphers, which take and encrypt one bit of plaintext at a time, and block ciphers, which take a number of bits and encrypt them as a single block. Most ciphers belong to the block cipher class. Symmetric algorithms are generally faster than asymmetric ones and use a much shorter key.

The main problem with secret-key cryptosystems is getting the sender and receiver to agree on the secret key without anyone else finding out [24]. This requires a method by which the two parties can communicate without fear of eavesdropping.

#### 6.1.2.1 Data Encryption Standard

The Data Encryption Standard (DES) is a strong cipher which encrypts a block of 64 bits at a time using a 56 bit key (56 + 8 parity checks = 64) resulting in 64 encrypted bits [24].

DES encryption itself consists of many rounds of different transformations and permutations, which are linear and easy to reverse. The critical encryption is done using S-boxes. The S-boxes, or substitution boxes, are sets of highly non-linear functions, implemented in DES as a set of lookup tables of four rows and 16 columns. The S-boxes encrypt four bits at a time, so encrypting is done in 16 rounds. After the S-boxes the results are still permutated.

The complicated substituting, permutating, XORs and shifts were chosen to have some useful properties:

- The same algorithm works for both encryption and decryption.
- The simple operations make the algorithm very fast.

DES is normally used in Cipher Block Chaining (CBC) or Cipher Feedback (CFB) mode. In CBC mode a plaintext block is first XORed with the previous ciphertext block and then encrypted to obtain the ciphertext. In CFB mode the previous ciphertext block is encrypted and then XORed with the plaintext to get the ciphertext.

There are numerous software applications and C libraries with the DES encryption routines widely available, although exporting DES from the USA is regulated by the NSA (National Security Agency).

According to RSA Laboratories, when implemented entirely in software, DES is at least 100 times faster than RSA (RSA is described in section 6.1.3.3). Implemented in hardware, it may outperform the RSA algorithm by 1,000 or even 10,000 times. This is due primarily to the fact that the DES S-boxes are simple table-lookup functions, while RSA depends on very large integer arithmetic. A fast 486 PC can encrypt about 400 KB per second using DES. [23]

### Security of DES

There are two known ways to break DES. The first requires an exhaustive search of the keyspace, which consists of $2^{56}$ (about $7.2*10^{16}$) possible keys (brute force). If you can test one million keys every second, it should take about 2,000 years to go though the keyspace. With special hardware and/or networked machines, testing can be done magnitudes faster. A chip could be designed that does a billion tests per second, reducing the needed time to two years. It is said that you can buy a dedicated machine with special decrypting hardware that can break DES in a couple of hours with brute force, by spending one million US dollars. [23]

The other more recent method is called differential cryptoanalysis. This method reduces the number of keys that must be tested, but it requires that you have $2^{47}$ chosen plaintexts encrypted with the key you are trying to recover. Since it is highly unlikely that anyone will agree to encrypt $2^{47}$ chosen plaintexts with their secret DES key, this attack is unfeasible in practice. [24]

Because the DES algorithm is based on the mysterious S-boxes, which are just constants without any known connections, it has lead to some rumors that there is a trapdoor in the algorithm.

The overall consensus is that DES, when used properly, is secure against all but the most powerful organizations (like NSA, governments, big supercomputing/parallel computing companies, etc.). Proper use means avoiding known weak keys (weak keys are result of the key being split to 16 pieces, one for each round of encryption). [24]

The short key is the main known risk in DES. Given all these points, using simple DES for top-secret data is not a good idea anymore, but is sufficient for everyday use.

#### 6.1.2.2 Triple DES

If DES with a single key is not sufficiently secure for a given application, it can be made more secure by encrypting more than once with different keys. It has been proven that multiple encryptions do actually improve the security of DES [24], and it is thought that triple-encryption with DES (3DES) is about equivalent to single-encryption with a 112-bit key [6]. 3DES usually uses two different keys. First the data is DES encrypted with first key, then decrypted with the second key, and then encrypted with the first key again. Triple DES is almost three times slower than DES. A fast 486 PC can encrypt about 150 KB per second. [23]

##### Security of 3DES

At a rate of one million keys per second, an exhaustive search of $2^{112}$ keys would require about $1.65*10^{20}$ years to complete. Since the universe is estimated to be only about $10^{10}$ years old that is probably long enough for most purposes. [23]

3DES has been proven much more secure than conventional DES, and is a good alternative for current designs. There still remain the same rumors that concern DES; is there a trapdoor? But as is, it is a better alternative to DES and with current machine technology makes brute force attacks not viable [24]. Applications using DES can easily be converted to use 3DES instead, if the slight effect on speed is not critical.

### *6.1.3 Asymmetric Algorithms*

In traditional cryptography, both the sender and receiver use the same secret key for encryption and decryption. This method is known as secret-key or *symmetric* cryptography. The big challenge here is to exchange this secret key without anyone else finding this secret key out. If a person gets hold of this key he or she may read, modify, and forge all messages encrypted with that key. The generation, transmission and storage of keys is called *key management* and all cryptosystems must deal with key management issues. It is difficult to provide a secure key management system, especially in open systems with a large number of users. [24]

To solve this key management problem, Whitfield Diffie and Martin Hellman introduced the concept of public-key cryptography in 1976 [3]. In these public-key cryptosystems each person gets a pair of keys, one public key and one private key. The public key is published and known by all, while the private key remains secret. Using this technique there is no need for exchanging private keys since all communication involve only public keys. Anyone using an *asymmetric* system, as these are called, can send a confidential message by just using public information, but the message can only be decrypted with the private key, a key only the intended receiver has. Public-key cryptography can also be used for authentication with digital signatures, and not only privacy using encryption. [24]

The private key is always mathematically linked to the public key in a public-key cryptosystem and therefore there is a possibility to derive the secret key from the public key. To defend attacks of this kind the derivation is made as hard as possible where the attacker is required to factor a very large number in order to derive the secret key. [24]

#### 6.1.3.1 Encryption

When the sender wishes to send a secret message to a receiver, she looks up the receiver's public key in a directory, and uses it to encrypt the message, which she then sends over the network. No one listening to the transmission can read the message, since only the receiver's private key can be used to decrypt the message. [24]

#### 6.1.3.2 Digital Signatures

To add a signature to a message, the sender does computation with both her private key and the message itself. The output is called a digital signature and is attached to the message. The receiver verifies the signature by a computation involving the message, the signature and the sender's public key. If the result is correct, corresponding to a mathematical relation, the signature is genuine. If not, the message have been altered in some way. [24]

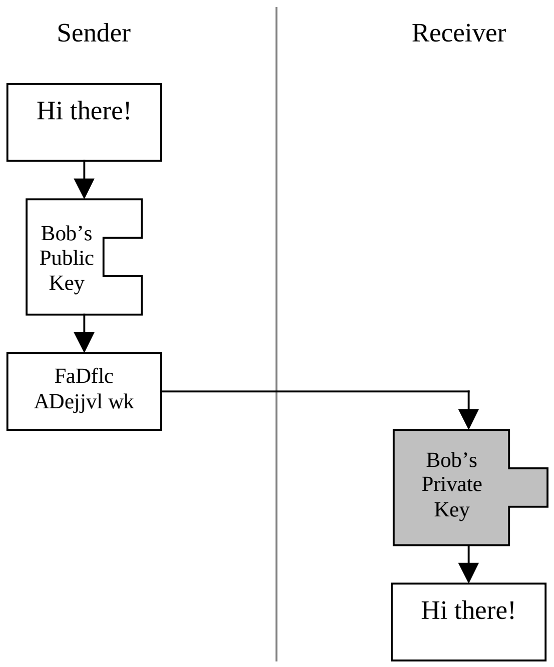

*Figure 6.1: Use of asymmetric encryption and decryption.*

### 6.1.3.3 RSA

RSA is a public-key cryptosystem for both encryption and authentication. It was invented in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman (RSA). RSA is the most widely used public-key cryptosystem today and has often been called a de facto standard [23].

RSA works as follows: take two large primes, $p$ and $q$, and find their product $n = pq$. Choose a number, $e$, less than $n$ and relatively prime to ($p$-1)($q$-1), and find its inverse, $d$, mod ($p$-1)($q$-1), which means that $ed = 1$ mod ($p$-1)($q$-1); $e$ and $d$ are called the public and private exponents, respectively. The public key is the pair ($n$, $e$); the private key is $d$. The factors $p$ and $q$ must be kept secret, or destroyed. [24]

It is presumably difficult to obtain the private key $d$ from the public key ($n$, $e$). If one could factor $n$ into $p$ and $q$, however, then one could obtain the private key $d$. Thus, the entire security of RSA is predicated on the assumption that factoring is difficult; an easy factoring method would break RSA. [3] There are many applications today using RSA, most notably PGP and SSH.

#### RSA Based Encryption

Suppose Alice wants to send a private message, $m$, to Bob. Alice creates the ciphertext $c$ by exponentiating: $c = m^e$ mod $n$, where $e$ and $n$ are Bob's public key. To decrypt, Bob also exponentiates: $m = c^d$ mod $n$, and recovers the original message *m*; the relationship between *e* and *d* ensures that Bob correctly recovers *m*. Since only Bob knows *d*, only Bob can decrypt the encrypted message. [24]

### RSA Based Authentication

Suppose Alice wants to send a signed document *m* to Bob. Alice creates a digital signature *s* by exponentiating: $s = m^d \bmod n$, where *d* and *n* belong to Alice's key pair. She sends *s* and *m* to Bob. To verify the signature, Bob exponentiates and checks that the message *m* is recovered: $m = s^e \bmod n$, where *e* and *n* belong to Alice's public key. [24]

Encryption and authentication take place without any sharing of private keys: each person uses only other people's public keys and his or her own private key. Anyone can send an encrypted message or verify a signed message, using only public keys, but only someone in possession of the correct private key can decrypt or sign a message.

### Speed of the RSA Algorithm

RSA operations are all based on series of multiplications. In practical applications, it is common to choose a small public exponent for the public key. Entire groups of users can use the same public exponent. This makes encryption faster than decryption and verification faster than signing. [24]

There are many commercially available hardware implementations of RSA, and there are frequent announcements of newer and faster chips. The fastest current RSA chip has a throughput greater than 600 Kbits per second with a 512-bit modulus, implying that it performs over 1,000 RSA private-key operations per second.

By comparison, DES is much faster than RSA. In software, DES is generally at least 100 times as fast as RSA. In hardware, DES is between 1,000 and 10,000 times as fast, depending on the implementations. RSA will probably narrow the gap a bit in coming years, as it finds growing commercial markets, but will never match the performance of DES.

### Security of the RSA Algorithm

The security of RSA depends of factoring being difficult. There are several methods to try factoring, but as long as the keys are long enough there is small risk of having your RSA encoded message broken. 384 bits can be broken relatively easily, 512 bits is probably insecure and breakable by major governments, 768 bits is probably relatively safe, 1024 bits should be secure for decades according to today's information. 2,048 bits will most probably remain safe for a long time. [24]

Another way to break RSA is to find a technique to compute e$^{th}$ roots mod $n$. Since $c=m^e$, the e$^{th}$ root of $c$ is the message $m$. This attack would allow someone to recover encrypted messages and forge signatures even without knowing the private key. This attack is not known to be equivalent to factoring. No methods are currently known that attempt to break RSA in this way. [24]

RSA is very vulnerable to chosen-plaintext attacks, and a good guess can reveal the used key. It is also advisable to include some random data (at least 64 bits) to the encrypted plaintext.

#### 6.1.3.4 Diffie-Hellman

The Diffie-Hellman key agreement protocol (also called exponential key agreement) was developed by Whitfield Diffie and Martin Hellman in 1976. The protocol enables two parties to generate a shared and secret key, without having to exchange the secret key over an insecure medium. It is based on public key cryptography, which means that it makes use of a private and a public key. The new secret key is derived from the own private key and the other party's public key [21]. It is generally considered to be secure when sufficiently long keys and proper prime generators are used [23]. A working Diffie-Hellman algorithm can be found in Appendix A.

The protocol has two system parameters $p$ and $g$. They are both public and may be used by all the users in a system. Parameter $p$ is a prime number and parameter $g$ (usually called a generator) is an integer less than $p$, with the following property: for every number $n$ between 1 and p-1 inclusive, there is a power $k$ of $g$ such that $g^k = n$ mod $p$. [24]

This is how it works [22]:

1. Alice and Bob agree on a large prime $n$ and on $g$ such that $g$ is primitive modulo $n$. These two integers are public.

2. Alice randomly chooses a large integer $a$, which she keeps secret, and computes her public value $A=g^a$ mod $n$. Bob does the same thing and generates $b$ and $B=g^b$ mod $n$.

3. Alice sends $A$ to Bob and Bob sends $B$ to Alice.

4. Alice computes $K_{AB}=B^a$ mod $n$. Bob computes $K_{BA}=A^b$ mod $n$. $K_{AB}=K_{BA}=g^{ab}$ mod $n$ is the secret key shared by Alice and Bob and will be used as a key.

A person who intercepts and listens to the communication between Alice and Bob knows $g$, $n$, $A=g^a$ mod $n$ and $B=g^b$ mod $n$, which does not enable him to compute $g^{ab}$ mod $n$. For that he would first have to compute the discrete logarithm of $A$ or $B$ so as to recover $a$ or $b$, which will take a long time [3].

## Security of Diffie-Hellman

Diffie-Hellman is sensitive to the choice of the strong prime and the generator [23]. The size of the secret exponent is also critical to the security. Conservative advice is to make the random exponent twice as long as the intended session key.

The protocol depends on the discrete logarithm problem for its security [3]. It assumes that it is computationally infeasible to calculate the shared secret key $k=g^{ab} \bmod p$ given the two public values $g^a \bmod p$ and $g^b \bmod p$ when the prime $p$ is sufficiently large. Maurer has, under certain assumptions, shown that breaking the Diffie-Hellman protocol is equivalent to computing discrete logarithms.

The Diffie-Hellman key exchange is vulnerable to a man-in-the-middle attack. The man in the middle intercepts all messages exchanged between the parties and modifies the keys in the messages. This results in the man in the middle knowing the secret key and the other two parties not being aware of this. The man in the middle will decipher the message using the corresponding key and then cipher it with the other before sending it further. The two parties will think their communication is secure, but it is not. [3]

In this attack, an opponent Ingrid intercepts Alice’s public value and sends her own public value to Bob. When Bob transmits his public value, Ingrid substitutes it with her own and sends it to Alice. Ingrid and Alice thus agree on one shared key and Ingrid and Bob agree on another shared key. After this exchange, Ingrid simply decrypts any messages sent out by Alice or Bob, and then reads and possibly modifies them before re-encrypting with the appropriate key and transmitting them to the other party. You will see how this works below [22]:

1. Alice sends her public value $A=g^a \bmod n$ to Bob. Ingrid the interceptor replaces this public value by hers. Bob thus receives $I=g^i \bmod n$.

2. Bob sends his public value to Alice; Ingrid also replaces this value by hers.

3. Alice generates the secret $K_{AI}=I^a \bmod n$. Ingrid generates the same secret by computing $A^i \bmod n$.

4. Bob generates the secret $K_{BI}=I^b \bmod n$. Ingrid generates the same secret by computing $B^i \bmod n$.

Both Alice and Bob believe that they share the same secret key, but they do in fact share the same secret as Ingrid; it is not a secret anymore. This vulnerability is present because Diffie-Hellman key exchange does not authenticate the participants. Possible solutions include the use of digital signatures and other protocol variants.

The authenticated Diffie-Hellman key agreement protocol, or Station-to-Station (STS) protocol, was developed by Diffie, van Oorschot, and Wiener in 1992 to defeat the man-in-the-middle attack on the Diffie-Hellman key agreement protocol [24]. The immunity is achieved by allowing the two parties to authenticate themselves to each other by the use of digital signatures and public key certificates.

Roughly speaking, the basic idea is as follows. Prior to execution of the protocol, the two parties Alice and Bob each obtain a public/private key pair and a certificate for the public key. During the protocol, Alice computes a signature on certain messages, covering the public value $g^a$ mod $p$. Bob proceeds in a similar way. Even though Ingrid is still able to intercept messages between Alice and Bob, she cannot forge signatures without Alice's private key and Bob's private key. Hence, the enhanced protocol defeats the man-in-the-middle attack.

In recent years, the original Diffie-Hellman protocol has been understood to be an example of a much more general cryptographic technique, the common element being the derivation of a shared secret value (i.e., key) from one party's public key and another party's private key. The parties' key pairs may be generated anew at each run of the protocol, as in the original Diffie-Hellman protocol. The public keys may be certified, so that the parties can be authenticated and there may be a combination of these attributes.

### *6.1.4 Cryptographic Hash Functions*

A cryptographic hash function generates a fixed-size hash value from a message of any length. The idea is to generate a hash value that cannot be used to trace back the original message. The typical applications include things like secret numbers on ATM cards etc. [23]

#### 6.1.4.1 Message Digest Algorithm 5

Message Digest Algorithm 5 (MD5) is a secure hash algorithm developed at RSA Data Security, Inc. It can be used to hash an arbitrary length byte string into a 128-bit value. MD5 is in wide use, and is considered reasonably secure. MD5 processes the input text in 512-bit blocks, divided into 16 32-bit sub-blocks. The output is a set of four 32-bit blocks, which are concatenated to a single 128-bit hash value. [24]

##### Security of MD5

It has been reported recently that MD5 has potential weaknesses in it, and that it is breakable in some cases. It is also said that one could build a special-purpose machine costing a few million dollars to find a plaintext matching a given hash value in a few weeks. Still, MD5 is considered to be relatively secure and good enough for most purposes. [24]

#### 6.1.4.2 Secure Hash Algorithm

Secure Hash Algorithm (SHA), also Secure Hash Standard (SHS) was published by the US Government. It produces a 160-bit hash value from an arbitrary length string and is structurally similar to MD5 [23]. It is roughly 25% slower than MD5 but may be more secure, because it produces message digests that are 25% longer than those produced by the MD functions. [24]

Security of SHA

Since SHA has a longer (160-bit) hash value it is more resistant to brute force attacks than MD5.

## 6.2 Symmetric versus Asymmetric Cryptography

The best applications combine different cryptosystems to increase the security level. The main advantage of public-key (asymmetric) cryptography is that it is both more secure and easier at the same time. In a secret-key system, the key must be transmitted to the intended receiver in order to be able to decrypt the messages again. A serious issue is that there is a chance that someone may discover the key while it is transmitted. [24]

Public-key systems have another major advantage; they can provide a method for digital signatures and authentication where secret-key systems would have to require a third party to do this. A disadvantage of using public-key cryptography for encryption is that the speed is reduced. The speed we are talking about here is the speed for the full transmission, including decryption and encryption. [24]

The best solution for encryption is to use a combination of both public- and secret-key systems. This to both get the security advantages of public-key systems and the speed of secret-key systems. Such a protocol is referred to as a digital envelope. The public-key system is here used to encrypt the secret key, which is transmitted and used for further encryption and decryption. [24]

In some situations secret-key cryptography is sufficient and there is no need for public-key cryptography. Examples of this are where secret keys can be distributed manually directly between persons, or in a single-user environment. In general, public-key cryptography is best suited and mostly only needed in an open multi-user environment. Public-key cryptography is not meant to replace secret-key cryptography, but to make it more secure. Secret-key cryptography still remains important and is subject to continuous study and research. [24]

# 7 IP Security

The Network Layer (the IP Layer in the case of the TCP/IP stack) is the lowest layer within the layered communications protocol stack model that can provide end-to-end security [7]. The security protocols in the Network Layer provide connectionless integrity, data origin authentication, protection against replay attacks, and confidentiality for all upper-layer application data carried in the payload of an IP datagram. This without requiring modification of existing applications [4]. The security solutions are based upon the open framework of IP Security Architecture (IPsec), defined by the IPsec Working Group of the IETF. It is called a *framework* because it provides a stable and lasting base for providing Network Layer security. IPsec can make use of today's cryptographic algorithms, but as newer and more powerful algorithms become available these can later be used. IPv6 implementations are required to support IPsec, and IPv4 implementations are strongly recommended to do so. [12]

The principal IPsec protocols are [12]:

- *IP Authentication Header (AH)* which provides data origin authentication, data integrity, and replay protection.
- *IP Encapsulating Security Payload (ESP)* provides data confidentiality, data origin authentication, data integrity, and replay protection.
- *Internet Security Association and Key Management Protocol (ISAKMP)* provides a method for automatically setting up Security Associations and managing their cryptographic keys.
- *Internet Key Exchange (IKE)* that performs the key exchanges.

It is important to understand how these protocols interact with each other in order to be able to implement and use IPsec. This is shown in figure 7.1 [18]:

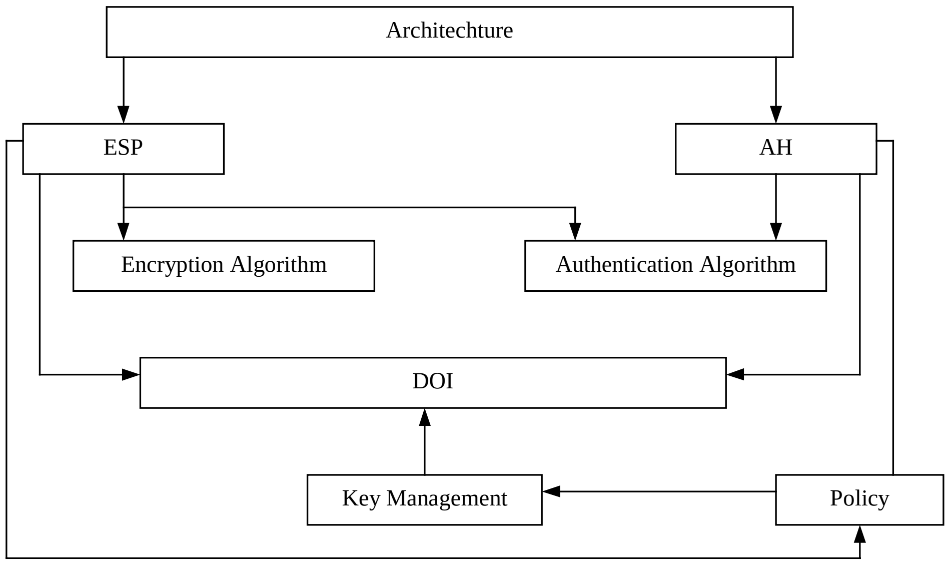

*Figure 7.1: IPsec Roadmap.*

IPsec is not a single protocol, it is an entire suite of protocols. As default it is required for the host to provide confidentiality using ESP and data integrity using either AH or ESP. The AH and the ESP documents define the protocol, the payload header format and the services they provide [13][14]. They also define the rules that apply to packet processing. They do not however include the transforms used to provide it. The transform is used to transform the data to secure it. It includes the algorithm, key sizes and how they are derived, the transformation process, and any algorithm-specific information. It is important that these values are specific in order to enable them to communicate.

IKE is used for generating the keys for the IPsec protocols. It is further used to negotiate keys for other protocols that need keys. IKE’s payload format is generic. It can thus be used to negotiate keys for any protocol and not necessarily only for IPsec. The parameters that are negotiated are documented in a separate document called the IPsec Domain of Interpretation (DOI) [15].

An important component is the policy protocol. The policy is important; it determines if two entities will be able to communicate with each other, and if, what transforms to use. If no policy is declared, or if it is not specific enough, the entities might not be able to communicate. The issues with policy are representation and implementation. Representation deals with definition of policy, storage and retrieval. The IETF is still working on defining policy standards (1999). The implementation addresses the application of policy for actual communication. [3]

## 7.1 IPsec Implementation

IPsec can be implemented in the end hosts, the gateways/routers or both. Where it should be implemented depends on the security requirements of the users. The host-to-host implementation is used, and most useful, when security is desired end-to-end. When security over a part of a network is desired, router implementation is desirable. This example includes VPNs and intranets.

### *7.1.1 Host Implementation*

The host is the device where the packet is originating. A host implementation has the following advantages [3]:

- Provides security end to end.
- Ability to implement all modes of IPsec security.
- Provides security on a per flow basis.
- Ability to maintain user context for authentication in establishing IPsec connections.

Host implementation can be classified into:

1. Implementation that is integrated with the operating system. We call it host implementation.
2. Implementation that is a shim between the network and the Data Link Layer of the protocol stack. This is called the Bump in the Stack (BITS) implementation.

#### 7.1.1.1 OS Integrated

As IPsec is a Network Layer protocol it may be implemented as part of the Network Layer. IPsec needs services of the Internet Layer to construct the IP header. A couple of advantages with implementing IPsec in the OS [12]:

- As IPsec is tightly integrated into the Network Layer, it can avail the network services such as fragmentation, and user context (sockets). This enables the implementation to be very efficient.
- It is easier to provide security services per flow (such as web transactions) as the key management, the base IPsec protocols, and the Network Layer can be seamlessly integrated.
- All IPsec modes are supported.

### 7.1.2 Bump in the Stack

There is a major drawback with OS integrated solutions for VPNs and intranets. On the end hosts, the implementation has to be dependent of features provided by the OS vendors. This might limit the capabilities to provide advanced solutions. To overcome this, IPsec is implemented as a shim between the network and Data Link Layer.

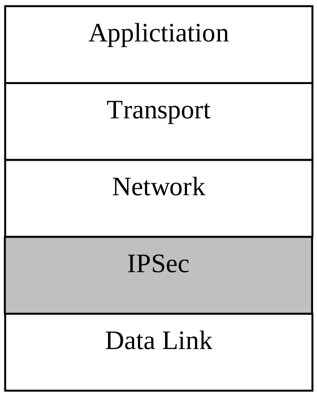

*Figure 7.2: Bump in the Stack (BITS) implementation*

### 7.1.3 Router Implementation

The router implementation provides security for a packet over a part of a network. It can thus be used to build VPNs. Router implementation has the following advantages:

- Ability to secure packets flowing between two networks over a public network, such as the Internet.
- Ability to authenticate and authorize users entering the private network. This feature is used by many organizations today and was previously only available with directly dialed up modem connections.

## 7.2 IPsec Modes

IPsec provides four possible combinations of modes and protocols: AH in Transport Mode, AH in Tunnel Mode, ESP in Transport Mode and ESP in Tunnel Mode [3]. AH in Tunnel Mode is not used as it protects the same data that AH in Transport Mode does. The AH and ESP headers do not change between Tunnel or Transport Mode. It is just a difference in what they are protecting; that is an IP packet or an IP payload.

### 7.2.1 Transport Mode

In this case, the AH and ESP protect the transport header. In this mode AH and ESP intercept the packets flowing from the Transport Layer into the Internet Layer and provide the configured security. [12]

Example: Hosts A and B have been configured so that all Transport Layer packets flowing between them should be encrypted. Transport Mode of ESP is used. If the requirement were just to authenticate the Transport Layer packets, Transport Mode of AH would be used. [3]

When security is not needed TCP and UDP packets flow into the Internet Layer without first having to encrypt or authenticate. The Internet Layer adds the IP header and calls into the Data Link Layer.

| | | |
|---|---|---|
| IP | TCP hdr | Data |

*Figure 7.3: Ordinary TCP/IP packet.*

When security is enabled in the Transport Layer, the Transport Layer packets flow into the IPsec component. The component is implemented as part of the Network Layer (when integrated with the OS). Then, the AH or ESP, or both, headers are added by the IPsec component, and the part of the Network Layer that adds the Network Layer header is invoked. When both AH and ESP headers are used the ESP header should be applied first. If the packet is first protected using AH and then ESP, the data integrity is applicable only for the transport payload as the ESP header is added later on:

| | | | | |
|---|---|---|---|---|
| IP | ESP | AH | TCP hdr | Data |

*Figure 7.4: Confidentiality is applied after integrity.*

This is not wanted, since the data integrity should be calculated over as much data as possible. If the packet is protected using AH after it has been protected by ESP, the data integrity applies to the ESP payload that in turn contain the transport payload. [12]

| | | | | |
|---|---|---|---|---|
| IP hdr | AH hdr | ESP hdr | TCP hdr | Data |

*Figure 7.5: AH provides integrity to the ESP that in turn encrypts the transport payload.*

### 7.2.2 Tunnel Mode

Tunnel Mode is normally used when the ultimate destination is different from the security termination point. It is also used when a router provides security services for packets it is forwarding. In the case of Tunnel Mode, IPsec encapsulates an IP packet with IPsec headers and adds an outer IP header. [12]

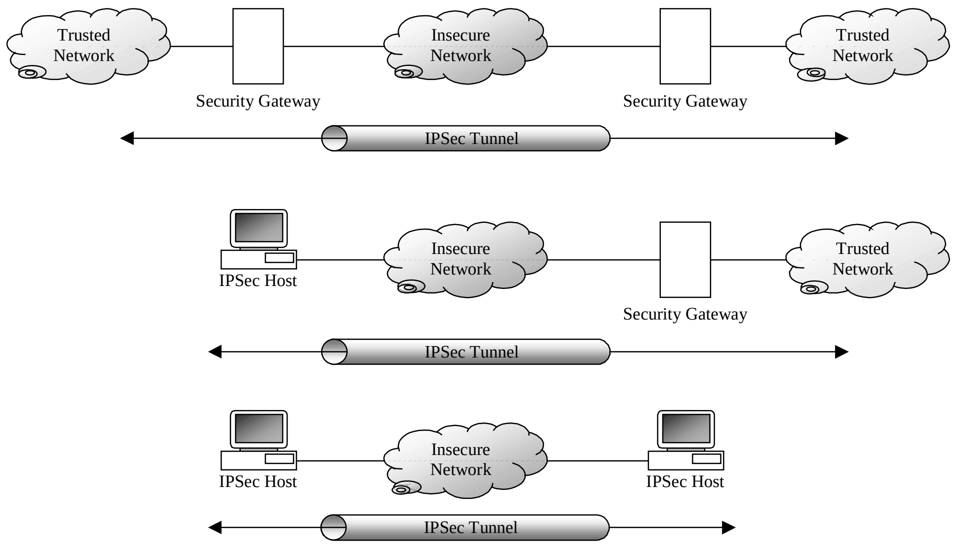

*Figure 7.6: Different Tunnel Modes.*

Thus an IPsec Tunnel Mode packet has two IP headers; inner and outer. The inner header is constructed by the host, and the outer header by the security device service (either host or router) [3].

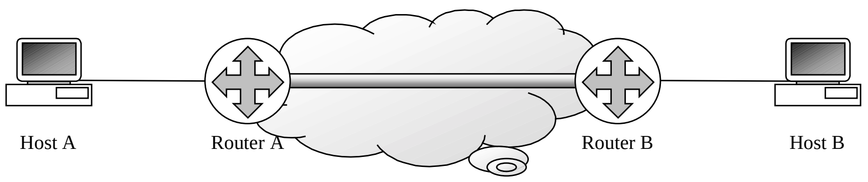

*Figure 7.7: Tunnel example.*

IPsec does also support nested tunnels; that is to tunnel a tunneled packet. It is important that the inner header is completely encompassed by the outer header. Nested tunnels are difficult to build and maintain and should only be used when absolutely necessary.

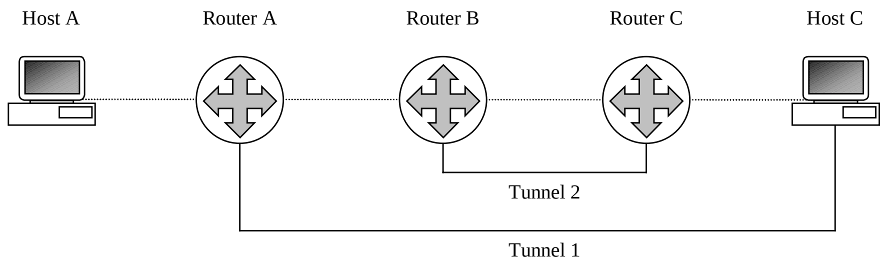

*Figure 7.8: Nested tunnels.*

## 7.3 Security Associations

Security Associations (SAs) form the basis of IPsec, and can be viewed as contracts between the two communicating entities [3]. They determine the IPsec protocols used for securing the packets, the transforms, the keys and the duration for which the keys are valid to name a few. Any IPsec implementation always builds an SA database (SAD) that maintains the SAs that the IPsec protocol uses to secure packets. [12]

The SA is a one way logical connection between two systems, uniquely identified by the following triplet <Security Parameter Index, IP Destination Address, Security Protocol>. [12]

### 7.3.1 Security Parameter Index

The Security Parameter Index (SPI) is just a 32-bit value used to identify different SAs with the same destination address and security protocol [3]. The SPI is carried in the header of the security protocol, and is generally selected by the destination system during the establishment of the SA. The SPI value is used to index into the receiving SAD and fetch the appropriate SA.

### 7.3.2 IP Destination Address

This is just the standard IP address for the destination host.

### 7.3.3 Security Protocol

The security protocol can either be AH or ESP. Since SAs are one way only, multiple SAs have to be initiated and used for bi-directional connections, one in each direction. Say, if host A and B are communicating securely using ESP, host A will have two SAs, one for incoming and one for outgoing packets. The same applies for host B. The $SA_{out}$ of host A and the $SA_{in}$ of the host B will share the same cryptographic parameters such as keys. The same goes for the other two. The SAs are also protocol specific. That is, there is one SA for each protocol. If two hosts are communicating using ESP and AH, there is an SA for each host and protocol, four SAs in total. [12]

#### 7.3.3.1 Security Policy Database

There is another component in the IPsec architecture called the Security Policy Database (SPD). It works in conjunction with the SAD in processing packages. The policy is an important component in IPsec, since it defines the security communications characteristics between two entities. It specifies what protocols to use in what modes and the transforms to use. Further more, it defines how the IP packets are treated, if the traffic must go through IPsec processing, if some packets must be discarded etc. [12]

#### 7.3.3.2 Security Association Database

The Security Association Database (SAD) contains parameter information about each SA, such as AH or ESP algorithms, sequence numbers, protocol and SA lifetime. For outbound processing, an SPD entry points to an entry in the SAD. That is, the SPD determines which SA is to be used for a given packet. For inbound processing the SAD is consulted to determine how the packet must be processed. [12]

### *7.3.4 SA Management*

The two most important tasks of SA management are creation and deletion of SAs. The management can either be manual or through an Internet standard key management protocol such as IKE. The SA management requires an interface for the user applications (which include IKE) to communicate with the kernel to manage the SAD. [3]

#### 7.3.4.1 Creation

The creation process consists of two steps: negotiating the parameters of the SA and updating the SAD with the created SA. It is possible to use manual keying, and when doing so, the two sides agree on the parameters of the SA offline (e.g. by phone or e-mail). The process of allocating the SPI and negotiation of parameters is all manual. This process is error prone, cumbersome and insecure and the SAs never expire when using manual keying. When using a stable and reliable key management protocol, the use of manual keying is questionable.

In an environment where IPsec is deployed, the SAs are created through an Internet standard key management protocol such as IKE. The IPsec kernel invokes IKE when the policy mandates that the connection should be secure and it cannot find the SA. IKE then negotiates the SA with the destination or intermediate host/router, depending on the policy, and creates the SA. Once the

SA is created and added to the SAD, secure packets start flowing between the two hosts.

#### 7.3.4.2 Deletion

The deletion of an SA may occur for one of the following four reasons:

- The lifetime has expired.
- The keys are compromised.
- The number of bytes encrypted/decrypted or authenticated using the SA has exceeded a certain threshold set by the policy.
- The other end requests that the SA be deleted.

The SAs can be deleted manually or by using IKE. To improve security and reduce the risk of someone breaking into the system, it is important to renew the keys at a regular basis. IPsec does not provide the refreshing of the keys, instead the existing SA has to be deleted and a new one negotiated/created. Once the SA is deleted, the SPI it was using can be reused.

To prevent the communication from stalling when an SA is deleted, the new one is negotiated before the old one expires. During a short duration of time the two SAs are coexisting and both can be used, although the new one is preferable in most cases.

### *7.3.5 Parameters*

The SA maintains the context of a secure communication between two entities. The SA stores both protocol-specific and generic fields. Some of the fields are used for outbound processing, some for inbound processing, and some for both, depending on the usage of the field. [12]

#### Sequence Number

The sequence number is a 32-bit field used in outbound processing. It is part of both AH and ESP headers and is used to detect replay attacks by the destination. When the SA is established this field is set to zero, and incremented by one every time the SA is used to secure a packet. The SA is normally renegotiated before this field overflows since it is unsafe to send more than four giga (4,000,000,000) packets using the same keys.

#### Sequence Number Overflow

This field is used in outbound processing and is set when the sequence number overflows. The policy determines if the SA can still be used to process additional packets.

#### Antireplay Window

This parameter is used in inbound processing. One major concern in networks today is replay attacks. In such an attack applications get bombarded with replay packets. This risk is overcome since IPsec is detecting these packets.

#### Lifetime

This limits the duration of how long the SA can be used. Beyond the lifetime, the SA is unusable. The lifetime is defined in terms of bytes that has been secured using this SA, or the duration of time, or both. When the lifetime has expired, the SA can no longer be used. To prevent interruption of the communication when the lifetime is expiring, there are two different types of lifetimes, soft and hard ones. The soft lifetime warns the kernel that the SA is about to expire, and permits the negotiation of a new SA before hard lifetime expires.

#### Mode

IPsec protocols can be used either in Tunnel or Transport Mode. The payload is processed differently depending on the value of this field. This field is set to Tunnel Mode, Transport Mode or a wild card. The wild card indicates that the SA can be used either for Tunnel or Transport Mode.

#### Tunnel Destination

For IPsec Tunnel Mode, this indicates the tunnel destination, that is the destination IP address of the outer header.

#### PMTU Parameters

When IPsec is used in Tunnel Mode, it has to maintain the PMTU information so that it can fragment the packets accordingly. As a part of the PMTU field, the SA maintains two values, the PMTU and the aging field.

### *7.3.6 Security Policy*

The Security Policy determines the security services afforded to a packet. The policy is stored in the Security Policy Database (SPD), which is indexed by selectors and contains the information on the security services offered to an IP packet. [3]

The Security Policy is consulted for both inbound and outbound packet processing of the IP packets. Two different Security Policies can be used; one for each bound, providing an asymmetric policy. However the key management protocol always negotiates bi-directional SAs. In practice, the tunneling and nesting will be mostly symmetric.

The security policy requires policy management to add, delete and modify policies. The SPD is stored in the kernel and IPsec implementations should provide an interface to manipulate the SPD. The management of the SPD is implementation specific and there is no standard defined.

The selectors below are used to determine the security services afforded to a packet. They are all extracted from the Network and Transport Layers headers.

### Source Address

The source address can be a wild card, an address range, a network prefix or a specific host. Wild card is useful and used when the policy is the same for all the packets from a host. The network prefix and address range is used for security gateways providing security to hosts behind it and to build VPNs. A specific host is used either on a multi-homed host or in the gateways when an individual host's security requirements are specific.

### Destination Address

The destination address can also be a wild card, an address range, network prefix or a specific host as above. The first three are used for hosts behind secure gateways. The destination address field used as a selector is different from the destination address used to look up SAs in the case of tunneled IP packets. In case of tunneled packages, the destination IP address of the outer header can be different from that of the inner header when the packets are tunneled. However, the policy in the destination gateway is set based on the actual destination and this address is used to index into the SPD.

### Name

The name is used to identify a policy tied to a valid user or system name. These include a DNS name, X.500 Distinguished Name or other name types defined in the IPsec DOI [15]. However, the name field is only used during IKE negotiation, not during packet processing. It can not be used as a selector during packet processing as there are presently no way to tie an IP address to a name.

### Protocol

Specifies the transport protocol whenever the transport protocol is accessible. When ESP is used the transport protocol is not accessible. Then a wild card is used.

### Upper Layer Ports

In cases where there is session-oriented keying, the upper layer ports represent the source and destination ports to which the policy is applicable. The wild card is used when the ports are inaccessible.

## 7.4 IPsec Processing

Here the processing of the IPsec packets, both inbound and outbound, will be discussed. The processing is split between the two bounds.

### *7.4.1 Outbound*

In outbound processing, the Transport Layer packets flow into the Internet Layer. The Internet Layer consults the SPD to determine the security services afforded to this packet. The input into the SPD is the selectors mentioned earlier. The output is [12]:

- Drop the packet, the packet is not processed, but dropped.
- Bypass security, the Internet Layer adds the IP header to the payload and dispatches the IP packet.
- Apply security, if an SA is already established, the pointer to it is returned. If no SA is established, IKE is invoked to do it.

After the SA is established (if no SA was before) it processes the packets by adding the appropriate AH and ESP headers. It is important that the processing is done in the right order.

| | | | | |
|---|---|---|---|---|
| IP Header | ESP | IP Header | AH | Network Payload |

*Figure 7.9: Nested packet format.*

### *7.4.2 Inbound*

Inbound processing differs from outbound. On the receipt of the IP packet, if the packet does not contain any IPsec headers, the security layer checks the policy to determine how to process the packet. It indexes the SPD using the selector fields. The output will be either discard, apply or bypass [12]:

- Discard, drop the packet.
- Apply, but no SA is established, drop packet.
- If none of the above, the packet is passed up to the next layer for further processing.

If the packet contains IPsec headers, it will be processed by the IPsec layer. The IPsec layer extracts the SPI, source address and destination address from the IP datagram. It indexes into the SAD by using the tuple (SPI, dst, protocol). The protocol value is either AH or ESP and the packet is handled in the appropriate layer (AH or ESP) according to this. After the protocol payload is processed, the policy is consulted to validate the payload. The selectors are used to retrieve the policy. [3]

Once the IPsec layer validates the policy, it strips off the IPsec header and passes the packets to the next layer, which is either a Transport Layer or a Network Layer.

### 7.4.3 Fragmentation

IPsec does not fragment or reassemble packets. On outbound processing, the transport payload is processed and then passed on to the Internet Layer for further processing. On inbound processing, the IPsec layer receives a reassembled packet from the Internet Layer.

However, as IPsec does add an IPsec header, it impacts the PMTU length. If IPsec does not participate in PMTU discovery, the Internet Layer ends up fragmenting a packet as the addition of the IPsec header increases the length of the IP datagram beyond the PMTU. [12]

## 7.5 Encapsulating Security Payload

Unless stated otherwise, the source for this section and its subsections is [14]. The Encapsulating Security Payload (ESP) is a protocol header that is inserted into an IP datagram to provide confidentiality, data origin authentication, antireplay and data integrity services to the standard Internet Protocol. ESP may be applied in two different modes. The first mode inserts the ESP header between the IP header and the upper-layer protocol header (e.g., a TCP or UDP header) or it may be used to encapsulate an entire IP datagram.

An encryptor provides the confidentiality in ESP and an authenticator provides the integrity. The encryptor and the authenticator use the same specific algorithm, which is determined by the corresponding components of an ESP Security Association (SA). As the base ESP definition and its actual algorithms are separated, ESP is seen as a generic and extensible security mechanism. ESP can optionally provide protection from antireplay attacks, but it is actually up to the recipient if this is to be done or not. The sender always inserts a unique sequence number, increasing by every packet sent and it is up to the recipient to check it.

### 7.5.1 The ESP Header

Regardless of the mode of the current ESP it immediately follows the IP header. In IPv4, the ESP header immediately follows the IP header (and any options). The protocol field will be 50 to indicate that following the IP header is an ESP header. In IPv6 however, the placement of the ESP header depends on the presence of extension headers. The ESP header is always inserted after the extension headers, which can change the route to the destination. As it is desirable to protect the destination options it should be inserted before these. If extension headers are present, the next header field preceding the ESP header is set to 50 to indicate that the following will be an ESP header. In cases where no extension headers are present, the next header field in the IPv6 header is set to 50. **[3]**

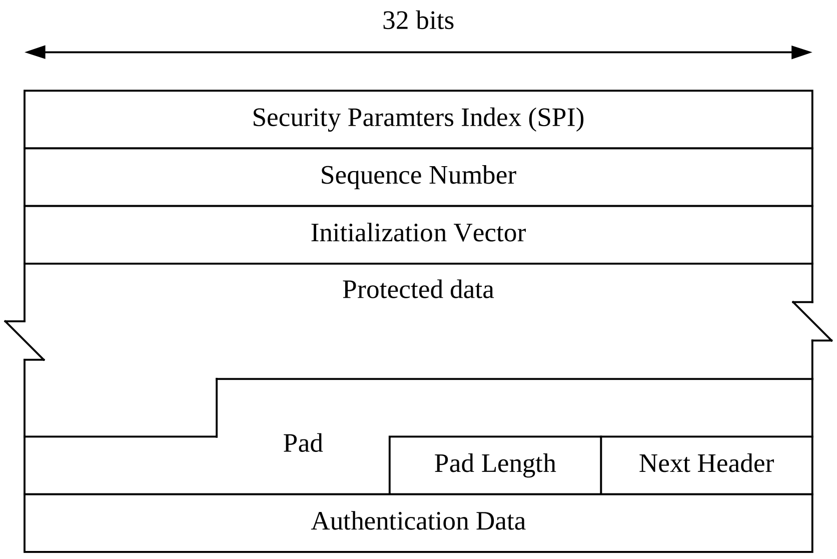

*Figure 7.10: The ESP header and trailer.*

The ESP header itself and the Security Association determine what follows it. This can be an upper-layer protocol such as TCP or an another IP header. ESP provides both confidentiality and integrity to the packet it is protecting.

As the ESP header is an IPsec header it contains an SPI field. This value combined with the destination address and protocol in the preceding IP header identifies the appropriate Security Association to use in the processing of the packet. The SPI itself is a number and is selected by the destination, usually during an IKE exchange. The SPI is authenticated but not encrypted. This is necessary because the SPI is used to identify the encryption algorithm and the key used to decrypt the packet.

The sequence number provided in ESP to fight antireplay attacks is authenticated but not encrypted. This is because we want to determine if a packet is a duplicate and drop it if it is, before decrypting it, saving valuable resources. The actual data being protected by ESP is contained in the data field payload. The length of the payload field depends on the length of the data. The protected data field is also used to contain any Initialization Vector that an encryption algorithm may require.

Padding to maintain boundaries is used in ESP. This is because certain modes of encryption algorithms may require that the input to the cipher be a multiple of its block size. This is just what the padding accomplishes. The pad length field simply defines how much pad has been added so that the recipient can restore the actual length of the payload data. The pad length field is mandatory, so even if there is no pad in the payload, the pad length field must indicate that. [3]

The following field in the header indicates the type of data that is contained in the payload data field. If ESP is applied in Tunnel Mode, the value will be four, indicating IP-in-IP. If ESP is applied in Transport Mode, this value will indicate the type of upper-layer protocol that follows, for example TCP would be six.

The authentication data field is used to hold the result from the data integrity check done on the ESP packet. The length of the field depends on the authentication algorithm employed by the SA used to process this packet. If there is no authenticator specified in the SA, used to process an ESP packet, there is no authentication data field.

### 7.5.2 ESP Modes

The ESP header is applied to an IP packet in either one of two ways; Transport or Tunnel Mode. The difference between the two is what ESP actually is set to protect.

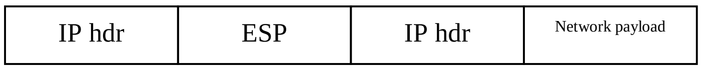

*Figure 7.11: ESP in Tunnel Mode.*

The ESP header is inserted between the IP header and upper-layer protocol header on an IP packet in Transport Mode. In Tunnel Mode the entire protected IP packet is encapsulated in the ESP header and in turn a new IP header is added to that. [3]

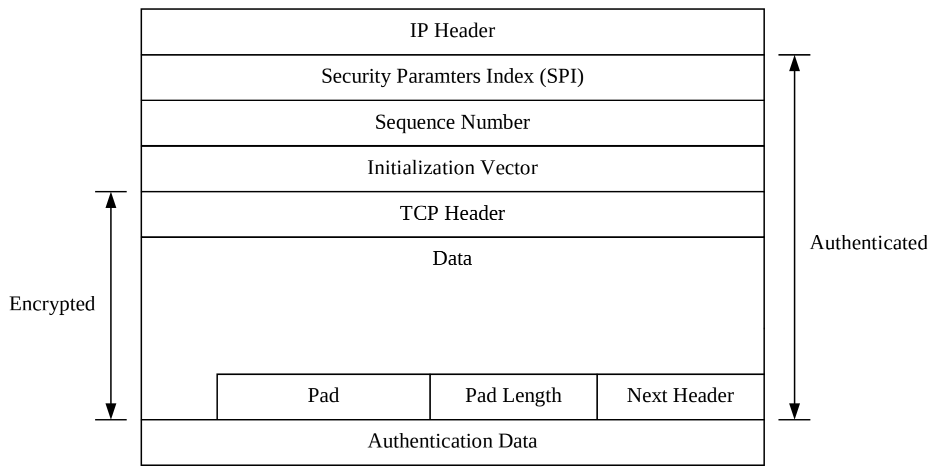

*Figure 7.12: ESP header in Transport Mode. Next Header is set to TCP.*

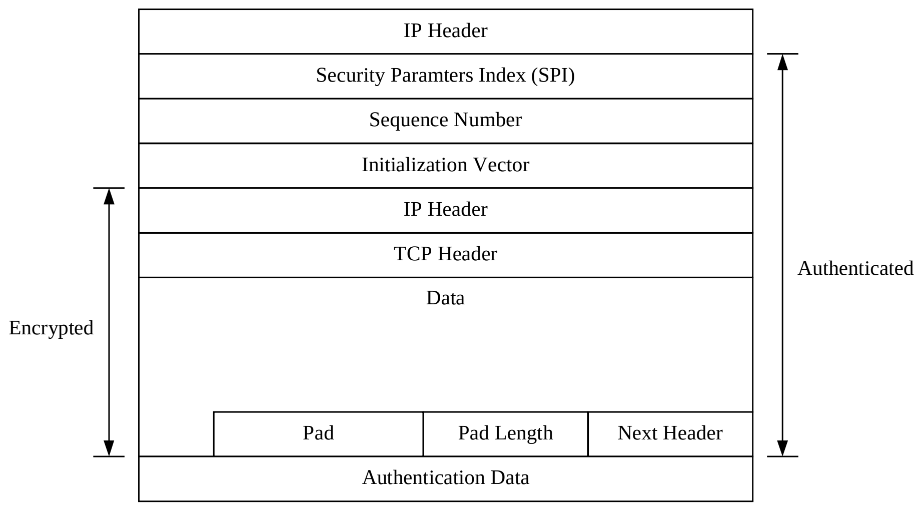

*Figure 7.13: ESP header in Tunnel Mode. Next Header is set to IP-in-IP.*

### 7.5.3 ESP Processing

The processing of an IP packet with ESP partly depends on the mode of the employed ESP. The cipher text is authenticated in either mode, but the authenticated plain text is not encrypted. This means that for outbound packets encryption occurs first, and for inbound packets authentication occurs first.

#### 7.5.3.1 Outbound Processing

During Transport Mode over IPv4, the ESP header is inserted directly after the IP header in an outbound IP packet. The protocol field of the IP header is copied into the next header field of the ESP header to indicate what is coming. Then the remaining fields of the ESP header are filled in. The SPI field is assigned the SPI from the SA in the SAD used to process the packet, and the sequence number is set to the next value in the sequence. Then the pad is inserted and its value assigned, the pad length value is also assigned. Last, the protocol field of the IP header is set to 50, which indicates that the payload is ESP. The rules are similar for IPv6 processing, except for the insertion of the header. In the case of IPv6, the ESP header is inserted after any extension header that may be modified en route. [3]

For Tunnel Mode applications, the ESP header is prepended to the IP packet. The next header field of the ESP header is assigned the value four if it is encapsulating an IPv4 packet and a value of 41 if it is encapsulating an IPv6 packet. The remaining fields are filled in the same manner as in Transport Mode. Then a new IP header is prepended to the ESP header and the appropriate fields are filled in; the source address is the device itself, which is applying the ESP. The destination address is taken from the SA used to apply ESP, the protocol is set to 50 to indicate ESP, and the rest are filled in according to local IP processing.

The next steps are identical, regardless of mode. The packet, from the beginning of the payload data to the next header field, is encrypted using the cipher indicated by the appropriate SA. Then the packet, from the ESP header, through the encrypted ciphertext, to the ESP trailer, is authenticated using the authenticator in the appropriate SA. The result of the authenticator is then inserted into the authentication data field of the ESP trailer. The final step in outbound processing is to recompute the checksum of the IP header that precedes the ESP header. [3]

#### 7.5.3.2 Inbound Processing

There is no way for he receiver to know whether a received packet is in Tunnel Mode or Transport Mode without processing it. Based upon the SA used to process the packet, the receiver will know what it should be, but until it is decrypted there is practically no way to know what ESP is protecting. This is a good thing, since any person doing traffic analysis would not know either. [3]

If a received IPsec packet is a fragment, it must be retained until all fragments have been collected. A fragmented IPsec packet cannot be processed since it would fail the data integrity check.

The first thing the recipient of an ESP packet does is check whether a matching SA exists to process it. This is a basic IPsec requirement and not particular to

ESP. If no SA exists, the packet must be dropped. Inbound processing can begin only if a matching SA exists. Once a valid SA has been identified, it will be used to process the packet. First, the sequence number is checked. If it is valid, that is, it is not a duplicate and is not to the right of the sequence number window contained in the SA, processing proceeds. [3]

Since ESP authenticates ciphertext and not plaintext, the next thing to do is authenticate the packet. The entire ESP packet, minus the authentication data, is passed along with the appropriate key to the authentication algorithm from the SA. If the resulting digest matches the data contained in the Authentication Data Field, taking into account any truncation that the authentication algorithm may require, the packet is authenticated. The next step is decryption. The ESP packet, from the beginning of the payload data to the next header field, is decrypted using the key and cipher algorithm from the SA.

After successfully passing authentication and decryption checks, a preliminary validity check of the resulting packet can be made. If the SA used to process the packet dictates that only ESP packets in a particular mode, either Transport or Tunnel Mode, can be processed, the packet must be checked for compliance. If the packet does not correspond to the required mode it must be dropped.

Now, the packet can be rebuilt without the ESP header. For Transport Mode, the next header field from the ESP header is copied into the protocol field of the IP header, and a new IP checksum is computed. For Tunnel Mode, the outer IP header and the ESP header can merely be thrown away, the decapsulated packet is what is needed. At this point another validity check has to be made. If Tunnel Mode, the IPsec SA may require that packets it processes may only be for a particular host, and/or for a particular port or protocol. If the packet does not correspond to the required address and/or port and/or protocol dictated by the SA, it must be dropped. [3]

Now, a reconstructed and validated packet can be forwarded for further processing. If it is a Transport Mode packet, it is passed up to a higher layer protocol, like TCP or UDP, for processing. If it is a Tunnel Mode packet, it is reinserted into the IP processing stream and forwarded on to its ultimate destination (which may be the same host).

If the reconstructed and validated packet is a fragment, it may be necessary to hold on to this packet until all fragments are received, reconstructed, and validated, and all the fragments have been reassembled. This would be necessary if IPsec was being applied by a network entity, in Tunnel Mode, or behalf of another host, and the SA that was used to process the packet(s) dictated that only packets for a particular port are allowed. Because any fragments would not have that information in them the only way, besides retaining fragments and reassembling a whole packet, would be to forward all fragments on to the destination. Because the destination would not have the SA information, it can not know if the reconstructed packet was valid or not. For processing speed reasons, it is recommended that network entities (such as routers) applying IPsec to transient traffic not retain decrypted and validated fragments if it is not required. Note that this case is different from receiving fragmented IPsec packets that must always be retained until a complete IPsec packet can be reassembled. [3]

## 7.6 Authentication Header

Unless stated otherwise, the source for this section and any subsections is [13]. The Authentication Header (AH) is used to provide data integrity, data origin authentication and optional limited antireplay services to IP. AH provides everything that ESP provides except confidentiality. AH does not encrypt any portion of the protected IP datagram. [3]

Since AH does not provide confidentiality it does not require a cipher algorithm. It requires an authenticator though. AH defines the method of protection, the placement of the header, the authentication coverage, and inbound and outbound processing rules. It does not, however, define the authentication algorithm to use. Like its sibling protocol ESP, AH does not mandate antireplay protection. The use of antireplay services happens only at the recipient side, and only if chosen. There is no way for the sender to know if the recipient will check the sequence number. Therefore, the sender must always assume that the recipient employs antireplay services.

AH can be used to protect an upper-layer protocol (Transport Mode) or an entire IP datagram (Tunnel Mode), just like ESP. The AH header immediately follows an IP header. AH is an IP protocol and an AH-processed IP packet is just another IP packet. Therefore AH can be used stand alone or in conjunction with ESP. It can protect a tunneling protocol or it can be used to tunnel packets itself.

The data integrity that AH provides is subtly different than that provided by ESP; AH authenticates portions of the outer IP header.

### *7.6.1 The AH Header*

AH is another IP protocol that has been assigned the number 51. This means that the protocol field of an AH-protected IPv4 datagram will be 51 indicating that following the IP header is an AH header. In case of IPv6, the value of the next header field depends on the presence of extension headers. In the absence of extension headers, the next header field in the IPv6 header will be 51. In the presence of extension headers prior to the AH header, the next header field in the extension header immediately proceeding the AH header is set to 51. The rules for inserting the AH header in IPv6 is similar to those described for ESP. When AH and ESP are protecting the same data, the AH header is always inserted after the ESP header. The AH header is much simpler than the ESP header because it does not provide confidentiality. There is no trailer as there is no need for padding and a pad length indicator. There is also no need for an Initialization Vector. [13]

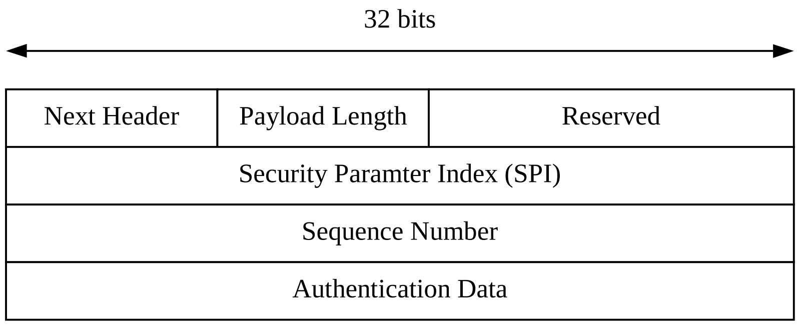

*Figure 7.14: The AH header.*

The *next header* field indicates what follows after the AH header. In Transport Mode it will be the value of the upper-layer protocol being protected, for instance UDP or TCP. In Tunnel Mode it will be the value four indicating IP-in-IP (IPv4) encapsulation or 41 for IPv6 encapsulation.

The *payload length* field indicates the length of the header itself in 32-bit words minus two. The reserved field is not used and must be set to zero.

The *SPI* field contains the SPI which, along with the destination address of the outer IP header, is used to identify the Security Association used to authenticate this packet.

The *sequence number* is a monotonically increasing counter that is identical to that used in ESP.

The *authentication data* field is a variable length field that contains the result of the integrity checking function. AH does not define an authenticator, but there are two mandatory-to-implement authenticators: HMAC-SHA-96 and HMAC-MD5-96. Like ESP, these are keyed MAC functions whose output is truncated to 96 bits. No public key authentication algorithms (like RSA) have been defined for use with AH. This is due to the cost; public key algorithms are too slow for bulk data authentication. In certain situations, such as network bootstrapping, AH is not used for bulk data protection and this limitation may not apply.

### 7.6.2 AH Modes

AH can be used in either Transport or Tunnel Mode, just like ESP. The difference is the data being protected, either an upper-layer protocol or an entire IP datagram. In either case, AH also authenticates immutable portions of the outer IP header.

#### 7.6.2.1 Transport Mode

When used in Transport Mode, AH protects end-to-end communication. The communication endpoint must also be the IPsec endpoint. The AH header is inserted into the datagram to protect it by placing it immediately after the IP header (and any options) and before the upper-layer protocol to protect.

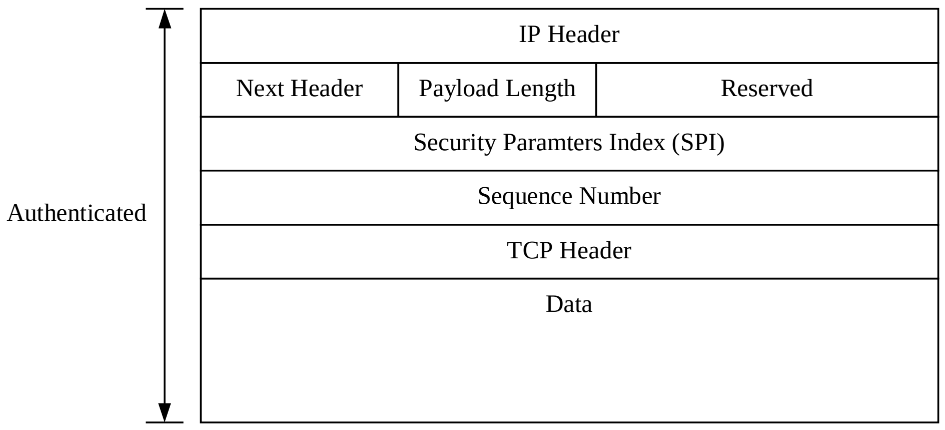

*Figure 7.15: AH in Transport Mode. Next Header is set to TCP.*

#### 7.6.2.2 Tunnel Mode

When used in Tunnel Mode, AH encapsulates the protected datagram and an additional IP header is added in front of the AH header. The internal IP datagram contains the original addressing of the communication and the outer IP datagram contains the addresses of the IPsec endpoints. Tunnel Mode can be used as a replacement to Transport Mode for end-to-end security, but since there is no confidentiality and therefore no protection against traffic analysis, there is really no point. AH is just used to guarantee that the received packet was not modified in transit, that it was sent by the party claiming to have sent it, and optionally that it is a fresh, non-replayed packet. [13]

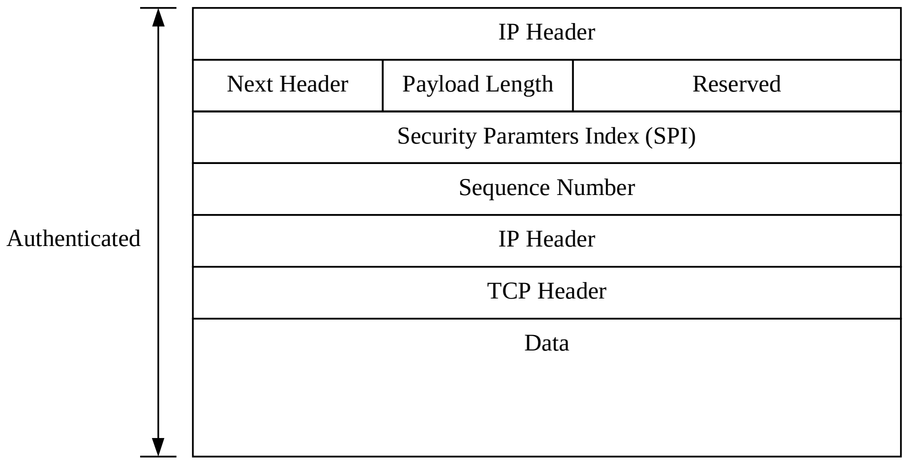

*Figure 7.16: AH in Tunnel Mode. Next Header is set to IP-in-IP.*

### 7.6.3 AH Processing

When an outbound packet matches an SPD entry denoting protection with AH, the SAD is queried to see whether a suitable SA exists. If there is no SA, IKE can be used to dynamically create one. If there is an SA, AH is applied to the matched packet in the mode dictated by the SPD entry. If there is an SPD bundle, the order of application depends on the protocols involved. AH always protects ESP, but not the other way around. [3]

#### 7.6.3.1 Outbound Processing

When an outbound SA is created the sequence number is initialized to zero. Prior to construction of an AH header using this SA, the counter is incremented. This guarantees that the sequence number in each AH header will be a unique, non-zero, and monotonically increasing number.

The remaining fields of the AH header are filled with their appropriate value, the SPI field is assigned the SPI from the SA. The next header field is assigned the value of the type of data following the AH header. The payload length is assigned the number of 32 bit words minus two. The authentication fields of an IPv4 header that are not included in the authenticating Integrity Check Value (ICV) are the shaded fields in figure 7.17:

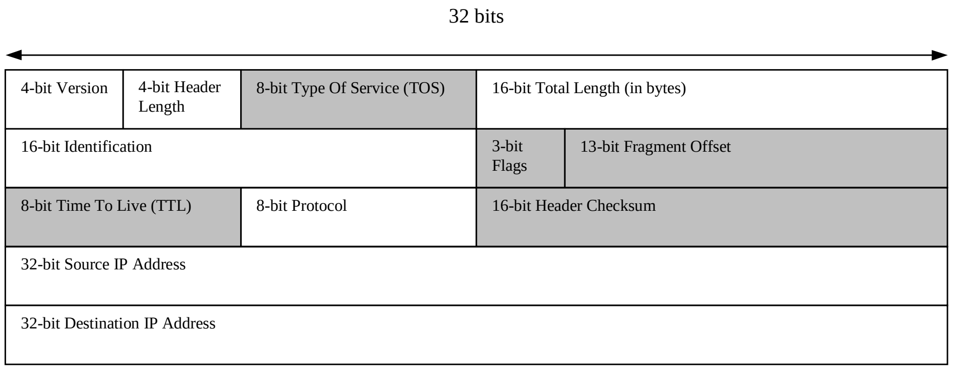

*Figure 7.17: Custom IP-header. The shaded fields that are not authenticated.*

Padding may be needed depending on the requirements of the authenticator or for alignment reasons. The padding must be entirely of the value zero, and its size is not included in the payload length and is not transmitted with the packet.

The AH header must be a multiple of 32 bits for IPv4 and 64 bits for IPv6. If the output of the MAC is such that this requirement is not met, the AH header must be padded. There is no requirement on the value of the pad, but it must be included in the ICV calculations and the pad size must be reflected in the payload length. The mandatory-to-implement authenticators are properly aligned, so no padding is needed when using HMAC-MD5-96 or HMAC-SHA-96.

The ICV is calculated by passing the key from the SA and the entire IP packet (including the AH header) to the algorithm identified as the authenticator in the SA. Since the mutable fields have been zeroed out they will not be included in the ICV. The ICV value is then copied into the authentication data field of the AH and the mutable fields in the IP header can be filled in according to IP processing.

AH processing is now complete and the AH-protected IP packet can be output. Depending on the size of the packet, it might be fragmented prior to placing it on the wire or it might be fragmented in transit by routers between the two IPsec peers. This is not a problem and is taken care of during inbound processing.

#### 7.6.3.2 Inbound Processing

Reassembly may be required prior to AH inbound processing if the protected packet was fragmented prior to its receipt. This is a fairly obvious requirement since the ICV check will fail unless it is done on exactly the same data from which the ICV was generated. AH therefore imposes this requirement on reassembly to guarantee that the inbound packet resembles the outbound packet the peer sent. A fully formed, AH-protected IP packet can then be passed on to AH inbound processing. [3]

The first thing to do when processing any IPsec packet is to find the SA that was used to protect it, and AH is no different than ESP in this respect. The SA is, again, identified by the destination address of the IP header, the protocol number (in this case 51), and the SPI from the AH header. If no matching SA is found, the packet is discarded.

Once the SA is found, a sequence number check is made. This step is optional but the cost-to-benefit ratio is so low that there is really no reason not to perform an antireplay check. The antireplay check determines whether this packet is new or received too late. If it fails this check it is discarded.

The ICV must now be checked. First, the ICV value in the authentication data field of the AH header is saved and that field is zeroed. All mutable fields in the IP are also zeroed. If the authenticator algorithm and payload length are such that implicit padding is required to bring the size of the data authenticated up to the requirements of the algorithm, implicit padding is added. This implicit padding must contain the value zero. The authenticator algorithm is then applied to the entire packet and the resulting digest is compared to the saved ICV value. If they match, the IP packet has been authenticated; if they do not match the packet must be discarded.

Once the ICV has been verified, the sequence number of the sliding receive window can be advanced if necessary. This concludes AH processing. The saved IP header can then be restored, remember that the mutable fields were zeroed out and this would prevent further processing, and the entire authenticated datagram can be passed to IP processing.

## 7.7 The IPsec DOI

IKE defines how security parameters are negotiated and shared keys are established for other protocols, but it does not define *what* to negotiate. That is up to the Domain of Interpretation (DOI) document. The purpose of a DOI document is to define a number of things: a naming scheme for DOI-specific protocol identifiers, the contents of the situation field of the ISAKMP SA payload, the attributes that IKE negotiates in a Quick Mode and any specific characteristics that IKE needs to convey. For instance, the IPsec DOI defines new fields in the ISKAMP ID payload, in effect overloading it, and new values of possible identities. This is necessary to convey selector information used to constrain negotiated IPsec SAs. [15]

The attributes defined in the IPsec DOI are those required to be part of an IPsec SA. Separate attribute spaces for AH and ESP are not necessary since the proposal and transform payloads in ISAKMP already allow for separate specification of the protocol. The task for the DOI is merely to define what the various protocols are that can be negotiated. In the case of IPsec, it is AH, ESP, and the attributes necessary. Any DOI that uses IKE must designate the IKE document as the source for attribute information when negotiating in phase one. The IPsec DOI is no different. [15]

The only difference in attributes between AH and ESP is the cipher. Each one needs an authenticator, and they both need lifetime attributes. So the negotiated attribute space for AH and ESP are one. In the case of AH, the cipher attribute space is just not negotiated. One confusing thing about negotiating SAs with the IPsec DOI is that AH requires both a transform attribute and an authenticator attribute. The currently defined transforms are AH_MD5 and AH_SHA. There are also authenticator attributes for HMAC-MD5 and HMAC-SHA. There is a transform attribute for HMAC-MD5 but also one for Key Pad Data Key (KPDK), the technique used in the deprecated AH specification [10]. For completeness, and backward compatibility, the IPsec DOI can negotiate the deprecated IPsec transforms, and this is the way to do it for AH. For ESP you negotiate a cipher and do not specify an authenticator [11]. Looked at from a historical and future perspective, there might be new ways to use both MD5 and SHA as MACs. [15]

# 8 Key Management and Key Exchange

This chapter will present an overview of key management and key exchange, and give some examples of methods used to perform a key exchange over IP. The methods presented are Oakley, SKEME and ISAKMP, and the understanding of them is important to be able to profit from the information in the next chapter about the Internet Key Exchange.

## 8.1 Key Management

Key management deals with the secure generation, distribution, and storage of keys. Secure methods of key management are of great importance. Once a key is randomly generated, it must remain secret to avoid unfortunate mishaps, such as impersonation. In practice, most attacks on public key systems will probably be aimed at the key management level, rather than at the cryptographic algorithm itself, since it is harder to calculate a key than to steal it.

Users must be able to securely obtain a key pair suited to their efficiency and security needs. There must be a way to look up other people's public keys and to publicize one's own public key. Users must be able to legitimately obtain others' public keys, or otherwise an intruder can either change public keys listed in a directory or impersonate another user. Certificates are used to prevent this from happening, and these certificates must not be forgeable.

## 8.2 Key Exchange

The tricky part with symmetric encryption is the key exchange, as it is the secret key that has to be transmitted over the network. In order to keep the transmission secure you need to decide how the encryption will take place, what kind of encryption algorithms to use and so on.

There are two important facts regarding key exchange [21]:

- Key exchange is fundamentally a complicated business.
- Key exchange gets more complicated as the group of communicating parties increases.

## 8.3 Key Exchange Protocols Developed for IP

IKE, is a hybrid protocol [17] and it is based on a framework defined by the Internet Security Association and Key Management Protocol (ISAKMP) [16]. IKE implements parts of two other key management protocols, Oakley and SKEME. In addition to these, IKE defines two exchanges of its own.

### 8.3.1 Oakley

It was from Oakley that IKE borrowed the idea of different modes, each producing a similar result, an authenticated key exchange, through the exchange of information at different speeds. In Oakley, there was no definition of what information to exchange with each message and the modes were examples of how Oakley could be utilized to achieve a secure key exchange. IKE on the other hand codified the modes into exchanges and by narrowing the flexibility of the Oakley model, IKE limits the wide range of possibilities that Oakley allows, still general, but in a well-defined manner. [19]

### 8.3.2 SKEME

SKEME, which also is a key exchange protocol, defines a type of authenticated key exchange in which the parties make use of public key encryption to authenticate each other and share components of the exchange. Each side encrypts a random number in the public key of the peer and both random numbers (after decryption) contribute to the ultimate key to be used in further transmissions. An optional Diffie-Hellman exchange can be done along with the SKEME share technique for so-called *Perfect Forward Secrecy* (PFS). To refresh the existing key a rapid exchange can take place. This exchange can take place without the requirement of the exchange of public keys. IKE borrows this technique directly from SKEME for one of its authentication methods (authentication with public key encryption) and IKE also borrows the notion of rapid key refreshment without PFS. [17]

### 8.3.3 ISAKMP

ISAKMP was developed by researchers at the National Security Agency (NSA), which used to be a super-secret organization whose existence even was denied by the United States government. As the NSA has come out of the shadows its considerable expertise in cryptography and security has been put to visible use. ISAKMP is one such output.

ISAKMP defines how two peers communicate. It defines how messages in the communication are constructed and also the transitions they go through in order to secure their communications. ISAKMP provides means to authenticate a peer, to exchange information for a key exchange, and to negotiate security services. However ISAKMP does not define how an authenticated key exchange is done. It either does not it define the attributes necessary for establishment of Security Associations. This task is left to other documents, namely a key exchange document (such as the Internet Key Exchange) and a Domain of Interpretation (such as the Internet IP Security Domain of Interpretation).

#### 8.3.3.1 Message and Payloads

The exchanged messages in an ISAKMP-based key management protocol are made up an ISAKMP header and together with chained ISAKMP payloads.

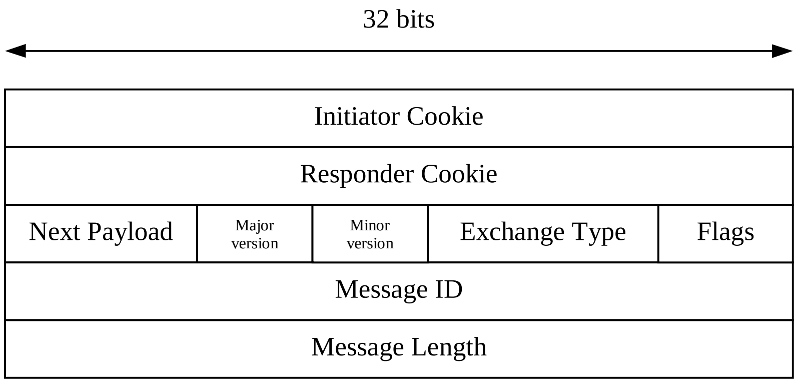

*Figure 8.1: The ISAKMP header.*

### The ISAKMP Header

The initiator and responder cookies and are used with the *Message ID* to identify the state of the ISAKMP exchange in progress. The *Next Payload* field indicates which ISAKMP payload that directly follows the header. The *Major* and *Minor Number* fields identifies the ISAKMP versions. *Exchange Type* identifies the specific type of ISAKMP used in the exchange. The *Flags* field indicates further relevant information to the recipient of the message. The flags are represented by a bit-mask where three fields have been defined so far. The three defined flags are encryption flag, commit flag, which indicates if a peer wants a notification when the exchange is complete, and authentication-only flag used to add key recovery. The other five flags allows further growth as new techniques are developed. The length of the message, including header, is stored in the *Message Length* field.

There are 13 distinct payloads defined in ISAKMP standard. They all begin with a generic header, which is the same for all 13.

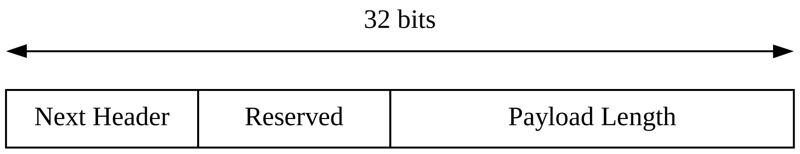

*Figure 8.2: The ISAKMP generic header.*

The *Next Header* field specifies the type of ISAKMP payload that follows the current payload. The payload length field denotes the entire length of an ISAKMP payload, including this header. The reserved field is currently not used and must be set to zero.

Some payloads are dependent on each and cannot exist on their own. For example, a proposal payload always encapsulates a transform payload. The proposal payload in turn is always is encapsulated with a Security Association payload.

ISAKMP defines payloads for certain exchanges. Some of these are generic, like hash digests or pseudo-random nonces, or specific, like certificates or Security Associations. The generic payloads are all identical and differ only in their payload identifier (which is actually in the next payload field of the preceding payload).

#### 8.3.3.2 Exchanges and Phases

There are two separate phases of negotiation described in ISAKMP. Establishment of an authenticated and secure channel between the peers is established in the first phase. The authenticated and secure channel is then used in the second phase to negotiate security services for a different protocol, such as IPsec.

An ISAKMP Security Association is established during the phase-one exchange. This SA has SA has similarities to an IPsec SA, but it is decidedly different. This SA is then used to authenticate subsequent phase-two exchanges. There is actually no limit to the number of ISAKMP SAs two peers can share, but practically, only one will do.

In the phase-two exchange Security Associations for other protocols are established. Since the ISAKMP SA is already authenticated in the first phase, it can be used to all messages in a phase-two exchange.

Further ISAKMP describes five exchanges, which in turn are neither strictly phase-one nor phase-two exchanges. Each of the exchanges has slightly different goals and accomplishes these goals in different number of steps depending on the exchange.

The first step of any ISAKMP exchange is the exchange of cookies that reside in the ISAKMP header. Each cookie is unique to the remote peer and also to the particular exchange. The cookies' purpose is to guarantee uniqueness and prevent the replay of old packets into a new stream. Another purpose is to provide some anticlogging protection. The cookie is the result of hashing a unique identifier of the peer and some timestamp. Each cookie is bound to the remote peer as a result of this, and it is easy to check that the cookie is the one given to that peer.

The cookie exchange takes place in the first two messages exchanged between the peers. A unique cookie for the initiator cookie field is created and inserted into the ISAKMP header. As the first message is from the initiator of the protocol, the responder cookie field is set to zero. When the message has been received and processed, the responder generates a cookie that is unique to the initiator and the exchange. This is copied to the responder cookie portion of the header and the initiator cookie is copied into the initiator cookie field. After these exchanges an SA for each peer is created. The two cookies identify the SA in the same manner as the SPI identifies the IPsec SA.

# 9 Internet Key Exchange

Unless stated otherwise, the source for this section and any subsections is [17]. The Internet Key Exchange (IKE) is the key management protocol included with IPsec, and also the standard key exchange protocol to be used by ISAKMP. It is a hybrid protocol, integrating the SKEME protocols with a subset of the Oakley key exchange scheme, and operating within a framework defined by the Internet Security Association and Key Management Protocol (ISAKMP). This framework defines the protocol language, for example packet formats, retransmission timers and message construction requirements [3]. IKE is used to negotiate, and derive keying material for ISAKMP and other Security Associations in a secure and authenticated manner. The final result of an IKE exchange is an authenticated key and agreed-upon security services; this constitutes an IPsec Security Association. IKE is not intended for IPsec only; it is generic enough to negotiate security services for any other protocol. It does not claim conformance or compliance with the entire Oakley protocol suite, nor is it dependent in any way on the Oakley protocol. Likewise, it does not implement the entire SKEME protocol, only the method of public key encryption for authentication, and its concept of fast re-keying using an exchange of nonces. This protocol is not in any way dependent on the SKEME protocol.

IKE supports Perfect Forward Secrecy (PFS) which works like this: if someone gets hold of the secret key, it only gives this person access to the data encrypted with this one key, not the rest. This means that IKE, when using PFS, may not derive new keys from old keying material that have been used earlier to protect a transmission of data. Thus, using PFS is a bit slower than not using it, but in return it should also result in a higher level of security.

Oakley and SKEME each define a method to establish an authenticated key exchange. This includes payloads construction, the information payloads carry, the order in which they are processed and how they are used. While Oakley defines modes, ISAKMP defines phases. The relationship between the two is very straightforward and IKE presents different exchanges as modes, which operate in one of two phases.

IKE is a general-purpose security exchange protocol that can be used for policy negotiation and establishment of authenticated keying material for a variety of needs. To be able to use IKE, you have to specify what it is being used for. This is done in a specification called a Domain of Interpretation (DOI). For IPsec, this DOI defines how IKE negotiates IPsec SAs [15].

Key exchange is a closely related service to the SA management; when you need to create an SA you also need to exchange keys. IKE delivers them as an integrated package, wrapped up together.

IKE provides the following services [21]:

- A way to agree on which protocols, algorithms and keys to be used (negotiation services).
- A way to ensure from the beginning of the exchange that you are talking to whom you think you are talking to (primary authentication services).
- A way to manage those keys after they have been agreed upon (key management).
- A way to exchange material for generating those keys safely.

## 9.1 Manual Key Exchange

IPsec compliant systems support manual key exchange as well as automatic. That is, if you for some reason would like to use face to face key exchange for certain situations, you still can. This is rather impractical though, especially for larger enterprises, so IKE will probably do most of the work for most people.

## 9.2 IKE Phases

IKE functions in two phases. In phase one, the two IKE peers establish a secure, authenticated channel with which to communicate. This is called the IKE Security Association (SA) and is needed to be able to perform IKE [21]. In phase two, those peers negotiate general-purpose SAs, on behalf of services such as IPsec or any other service, which needs key material and/or parameter negotiation. An IKE peer is an IPsec compliant node capable of establishing IKE channels and negotiating SAs. It may be a desktop computer or a security gateway that negotiates security services for you.

IKE defines two phase-one exchanges, one phase-two exchange, and two extra exchanges for proper maintenance of its Security Association. For the phase-one exchanges, IKE uses the identity protect exchange, and aggressive exchange from the base ISAKMP document and calls them Main Mode and Aggressive Mode, respectively. But unlike the ISAKMP exchanges, IKE has fully defined the exchange, the contents of all payloads and the steps taken to process them. For phase two, IKE defines a Quick Mode exchange. This negotiates security services for protocols other than IKE, primarily IPsec, but then again IKE is generic enough for a Quick Mode exchange to establish the necessary security services. The two other exchanges defined by IKE are an information exchange in which IKE peers can communicate error and status information to each other, and a new group exchange that allows IKE peers to negotiate the use of a new Diffie-Hellman group among themselves.

The two phase-one exchanges, Main Mode and Aggressive Mode, accomplish the same thing: the establishment of a secure and authenticated communications channel (the IKE SA) and authenticated keys used to provide confidentiality, message integrity, and message source authentication to the IKE communications between the two peers. All other exchanges defined in IKE have an authenticated IKE SA as a prerequisite. Therefore a phase-one exchange, either Main Mode or Aggressive Mode, must be performed before any other exchange can take place.

The parameters, encryption algorithm, hash algorithm, authentication method, and Diffie-Hellman group, are referred to as a protection suite. Protection suites are negotiated as a unit by exchanging ISAKMP SA payloads. Each attribute in the protection suite is contained in transform payloads. In addition to the mandatory attributes, there are some optional attributes that also may be negotiated. One of these attributes is lifetime, which determines for how long the IKE SA will exist. The longer an IKE SA exists, the greater the risk of leakage of its key, so an implementation is encouraged to include lifetimes in the protection suites offered to peers.

A hash algorithm is negotiated as part of a protection suite, but its use is usually in HMAC form. IKE uses an HMAC version of the hash algorithm as a Pseudo Random Function (PRF) to generate a seemingly random bitstream. It is possible to negotiate other PRFs with IKE, but usually an HMAC version of the negotiated hash algorithm is used. The encryption algorithm and hash algorithm attributes are straightforward. They determine which algorithm will be used for message encryption and authentication.

The attribute with the most impact on the IKE exchange is the authentication method used. The other attributes will determine the contents of payloads and how messages are protected, but the authentication method determines which payloads are exchanged and when they are exchanged. An IKE exchange may actually change depending on the authentication method negotiated by the two peers. The acceptable methods of authentication are:

- Pre-shared keys.
- Digital signatures using the Digital Signature Algorithm (DSA).
- Digital signatures using the Rivest-Shamir-Adleman (RSA) algorithm.
- Two similar methods of authentication via exchange of encrypted nonces.

These attributes are negotiated between the peers as part of the first messages they exchange. These are the external and visible characteristics of the IKE SA. But each side also maintains some secret information that will not be visible to a casual observer (or an active attacker) of an IKE exchange. This secret information is what is authenticated and what is used to protect IKE messages and also derive keys for other security services.

Phase-one exchanges are authenticated by each side computing a hash that only they could know. Due to the characteristics of hash functions, it is extremely difficult to invert them, the hash digest itself is not a secret and passing it in the clear is not considered a security breech. Since it is impossible to determine the inputs to a hash function from the output, and the same inputs will always produce the same output, the production of the appropriate digest authenticates the peer. The computation of the hash is identical regardless of the authentication method negotiated, but the initiator and responder compute their respective digests differently.

Regardless of the phase-one exchange, the protection suites offered by the initiator in this first message to the responder is unauthenticated and an attacker could modify it down to its lowest or weakest protection suite. The responder would reluctantly accept it and neither party would be aware of that they could have been communicating more securely. By including the entire SA payload in the authenticating hash, such an attack is prevented.

The IKE SA differs from an IPsec SA in that it is bi-directional. There are specific roles assumed by the participants of an exchange. Specifically, one party is the initiator and the other is the responder, but once the SA has been established it may be used to protect both inbound and outbound traffic. Also, regardless of who initiated the phase-one exchange that established the IKE SA, either party may initiate a phase-two exchange and protect it with the IKE SA. The cookies in the ISAKMP header are not swapped if the responder in phase one becomes the initiator in phase two, since the cookie pair is used to identify the appropriate SA in the IKE SAD.

All ciphers in IKE must be in CBC mode and therefore require an Initialization Vector (IV) in addition to a key. The initial IV is generated by hashing the two Diffie-Hellman public values together, and after each successive encryption and decryption the IV becomes the last ciphertext block processed. In this way, the IV chains cipher operations together and runs from bock to block.

## 9.3 IKE Modes

Oakley provides three modes of exchanging keying information and setting up SAs, two for IKE phase-one exchanges and one for phase-two exchanges. Main Mode and Aggressive Mode each accomplishes a phase-one exchange, and must only be used in phase one. The Quick Mode accomplishes a phase-two exchange, and must only be used in phase two. Except for these three modes, there is something called the New Group Mode. This mode does not really belong to either phase one or phase two. It follows phase one, but serves to establish a new group, which can be used in future negotiations. New Group Mode must only be used after phase one.

1 a. Main Mode accomplishes a phase-one IKE exchange by establishing a secure channel.

1 b. Aggressive Mode is another way of accomplishing a phase-one exchange. It is a little bit simpler and a bit faster than Main Mode, but does not provide identity protection for the negotiating nodes, as they must transmit their identities before having negotiated a secure channel through which to do so.

2. Quick Mode accomplishes a phase-two exchange by negotiating an SA for general-purpose communications. This is done under the protection of the IKE SA, which was created from a phase-one exchange.

With the use of IKE phases, an implementation can accomplish very fast keying when necessary. A single phase-one negotiation may be used for more than one phase-two negotiation. Additionally a single phase-two negotiation can request multiple Security Associations. With these optimizations, an implementation can see less than one round trip per SA as well as less than one Diffie-Hellman exponentiation per SA. Main Mode for phase one provides identity protection. When identity protection is not needed, Aggressive Mode can be used to reduce round trips even further. It should also be noted that using public key encryption to authenticate an Aggressive Mode exchange would still provide identity protection.

### *9.3.1 Main Mode Exchange*

Main Mode uses six messages, in three round trips, to establish the IKE SA. These three steps are SA negotiation, a Diffie-Hellman exchange and an exchange of nonces, and the authentication of the peer. The features of Main Mode are identity protection and the full use of the ISAKMP's negotiation capabilities. Identity protection is important when the peers wish to hide their identities. The authentication method can influence the composition of payloads and even their placement in messages, but the intent and purpose of the messages, the steps taken in Main Mode, remain regardless.

### *9.3.2 Aggressive Mode Exchange*

The purpose of an Aggressive Mode exchange is the same as a Main Mode exchange, the establishment of an authenticated Security Association, and keys, which IKE can then use to establish Security Associations for other security protocols. The major difference is that Aggressive Mode takes half the number of messages as Main Mode. By limiting the number of messages, Aggressive

Mode also limits its negotiating power and also does not provide identity protection.

In an Aggressive Mode exchange, the initiator offers a list of protection suites, his Diffie-Hellman public value, his nonce and his identity, all in the first message. The responder replies with a selected protection suite, his Diffie-Hellman public value, his nonce, his identity, and an authentication payload. For pre-shared key and encrypted nonce authentication this would be a hash payload, for signature-based authentication it would be a signature payload. The initiator sends his authentication payload as the final message.

The rich negotiation capabilities of IKE are constrained in Aggressive Mode, because the initiator must provide his Diffie-Hellman public value and his nonce in the first message. Therefore, he can not offer different Diffie-Hellman groups in different protection suites. In addition, if he wishes to do authentication using the revised method of encrypted nonces, he can not offer multiple protection suites with differing encryption or hash algorithm options.

Aggressive Mode is pretty limited, so what are its uses then? For remote access situations where the address of the initiator cannot be known in advance, and both parties desire to use pre-shared key authentication, this is the only exchange possible to establish the IKE SA. It is also useful in situations where the initiator knows the policy of the responder to create the IKE SA more quickly. If an employee wishes to remotely access the corporate resources of his company he will most likely know the policies under which access is granted, and therefore, the full power of IKE negotiation is not necessary.

### 9.3.3 Quick Mode Exchange

Once an IKE SA is established via Main Mode or Aggressive Mode exchange, it can be used to generate SAs for other security protocols such as IPsec. This is done via a Quick Mode exchange, which in turn is done under the protection of a previously established IKE SA. Multiple Quick Modes can be done with a single Main Mode ore Aggressive Mode exchange.

In a Quick Mode exchange the two peers negotiate the characteristics of, and generate keys for, an IPsec Security Association. The IKE SA protects a Quick Mode exchange by encrypting it and authenticating the messages. The messages are authenticated via the PRF function, most likely the HMAC version of the negotiated hash function. This authentication provides data integrity protection as well as data source authentication; upon receipt we will know that the message could only have come from our authenticated peer and that the message did not change in transit.

## 9.4 Establishing a Secure Channel

IKE uses the concept of a Security Association, but the physical construct of an IKE SA is different from an IPsec SA. The IKE SA defines the way in which the two peers communicate, which algorithm to use for encrypting IKE traffic, how to authenticate the remote peer etc. The IKE SA in then used to produce any number of IPsec SAs between the peers. The IPsec SAs may optionally have Perfect Forward Secrecy of the keys and also the peer identity. More than one pair of IPsec SAs may be created at once using a single IKE exchange, and a single IKE SA may perform any number of such exchanges.

The IKE protocol is performed by each party that will be performing IPsec; the IKE peer is also the IPsec peer. That is, to create IPsec SAs with a remote entity you speak IKE to that entity, not to a different IKE entity. The protocol is request-response type with an initiator and a responder. The initiator is the party that is instructed by IPsec to establish some SAs, it initiates the protocol to the responder.

- To establish an IKE SA, the initiating node proposes six things:
- Encryption algorithm (to protect data).
- Hash algorithms (to reduce data for signing).
- An authentication method (for signing data).
- Information about a group over which to do a Diffie-Hellman exchange.
- A Pseudo Random Function (PRF) used to hash values during the key exchange for verification purposes (this is optional, you can also just use the hash algorithm).
- The type of protection to use (ESP or AH).

A Pseudo Random Function (PRF) is just another name for a hash function. You can use the PRF both for authentication purposes and to generate additional key material as a randomizer.

### *9.4.1 How It Is Done*

**1.** The SPD of IPsec is used to tell IKE what to establish, but does not say anything about how this should be done. How IKE actually establishes the IPsec SAs is based on its own policy settings, defined in protection suites. A protection suite must at least define the encryption algorithm, the hash algorithm, the Diffie-Hellman group, and the method of authentication to be used. The policy database is then a list of all protection suites weighted in order of preference. This negotiation is the first thing that the two IKE peers do, since the specific policy suite they agree upon will dictate how the remainder of the communication is done.

**2.** IKE always uses a Diffie-Hellman exchange to establish the shared secret between the two peers, though there are other possible ways of doing this. The Diffie-Hellman exchange act is not subject to negotiation, but the parameters to use are. This exchange and the establishment of a shared secret is the second part of the IKE protocol.

**3.** The next step is an authentication of the Diffie-Hellman shared secret, and therefore, authentication of the IKE SA itself. This is done to guarantee that the remote peer, actually is someone you trust.

## 9.5 IKE Security

Confidentiality is assured by the use of a negotiated encryption algorithm. Authentication is assured by the use of a negotiated method: a digital signature algorithm; a public key algorithm that supports encryption; or, a pre-shared key. The confidentiality and authentication of this exchange is only as good as the attributes negotiated as part of the ISAKMP Security Association.

Repeated re-keying using Quick Mode can consume the entropy of the Diffie-Hellman shared secret. Because of this, it is recommended to set a limit on Quick Mode exchanges between exponentiations.

Perfect Forward Secrecy (PFS) of both keying material and identities is possible with this protocol. If this is desired, an ISAKMP peer must establish only one non-ISAKMP Security Association (e.g. IPsec Security Association) per ISAKMP SA. PFS for keys and identities is accomplished by deleting the ISAKMP SA (and optionally issuing a delete message) upon establishment of the single non-ISAKMP SA. In this way a phase-one negotiation is uniquely tied to a single phase-two negotiation, and the ISAKMP SA established during phase-one negotiation is never used again.

The strength of a key derived from a Diffie-Hellman exchange using any of the groups defined here depends on the inherent strength of the group, the size of the exponent used, and the entropy provided by the random number generator used. Due to these inputs it is difficult to determine the strength of a key for any of the defined groups. The default Diffie-Hellman group (number one) when used with a strong random number generator and an exponent no less than 160 bits is sufficient to use for DES. Groups two through four provide greater security. Implementations should make note of these conservative estimates when establishing policy and negotiating security parameters.

Note that these limitations are on the Diffie-Hellman groups themselves. There is nothing in IKE that prohibits using stronger groups, nor is there anything that will dilute the strength obtained from stronger groups. In fact, the extensible framework of IKE encourages the definition of more groups; use of elliptical curve groups will greatly increase strength using much smaller numbers.

For situations where defined groups provide insufficient strength, New Group Mode can be used to exchange a Diffie-Hellman group which provides the necessary strength. It is incumbent upon implementations to check the primality in groups being offered and independently arrive at strength estimates.

It is assumed that the Diffie-Hellman exponents in this exchange are erased from memory after use. In particular, these exponents must not be derived from long-lived secrets like the seed to a pseudo-random generator.

IKE exchanges maintain running Initialization Vectors (IVs) where the last ciphertext block of the last message is the IV for the next message. To prevent retransmissions, or forged messages with valid cookies, from causing exchanges to get out of sync, IKE implementations should not update their running IV until the decrypted message has passed a basic sanity check and has been determined to actually advance the IKE state machine; i.e. it is not a retransmission.

While the last roundtrip of Main Mode, and optionally the last message of Aggressive Mode, is encrypted it is not, strictly speaking, authenticated. An active substitution attack on the ciphertext could result in payload corruption. If such an attack corrupts mandatory payloads it would be detected by an authentication failure, but if it corrupts any optional payloads, e.g. notify payloads chained onto the last message of a Main Mode exchange, it might not be detectable.

# 10 Discussion

In this section we will provide a general discussion about network security, why it is needed and why it has become so relevant lately. We continue by discussing the benefits and drawbacks of IPsec and then we give an example of applied network security, e-commerce. Finally, we will summarize our conclusions and answer the research questions asked in the introduction to the thesis.

## 10.1 Network Security

Due to the rapid growth of the Internet and computing in general the computer security has never been more important. Computer security is a process that will guarantee that data can be stored in a computer and later retrieved only by the persons authorized to do so. If a person succeeds in accessing the data without authorization the security is violated. One of the major issues in network security is the balance between convenience and security. As mentioned in the introduction, you want a secure connection without having to enter numerous passwords, installing security applications etc.

The process of computer security requires choosing an appropriate security policy and maintaining it. The security is never finally established. It is a dynamic process that requires understanding security issues, keeping up to date on them and monitoring the computer system with the tools available.

Establishing security in a computer system should start from the system itself. There is little point in setting up network security, locally or to and from larger networks as the Internet, if the computer system itself is vulnerable to security attacks.

Computer security and the possible holes in it exist in many forms that can be roughly divided into four categories. For the first, the physical security of the system means limiting the access to the location of the system. Secondly, the software that is used in the system may do or can be instructed to do things that they should not. Thirdly, the improper combination of hardware and software may seriously flaw the system security. Finally, there is the ubiquitous human factor. There is little use of perfect hardware and software if the users do not understand that a login name backwards is bad choice for a password.

While only using a corporate network or a private home network, security is normally already protected in some way, or not needed. Security becomes an issue the moment you connect your computer or network to a public network, such as the Internet. It is impossible to know who is listening in on any traffic passing in or out from your network, the only way to protect oneself from abuse is to use available techniques for secure network connections. The world's largest public network, the Internet, is gaining more users from all over world every day, and some of those are potential exploiters of security flaws. Network security is also a psychological issue; more people would probably use the Internet if they knew that their connection was protected, and in turn the expected boom in e-commerce may become reality.

## 10.2 Advantages of IPsec

One of the major advantages gained by using IPsec is that it is a completely invisible technology. The user does not need to worry about it; it is protecting the system even though the user does not notice it. Since IPsec is the security standard set to default in coming IPv6, everyone using the new version of the TCP/IP stack, will get the security features as well. For instance, when upgrading to a future version of Microsoft Windows, you will automatically get an updated TCP/IP stack, which in turn supports IPsec.

The next premium feature of IPsec is its location in the TCP/IP layered model. IPsec is located in the Internet Layer, which means that all traffic, either inbound or outbound, must pass the IPsec protection suite. Further, it is the only protocol that can secure all and any kind of Internet traffic. IPsec also allows per flow or per connection security and thus allows for very fine-grained security control.

The drawback of most other security systems is that they most often are application specific, that is, they can be used for one purpose only, while IPsec protects everything at the same time. There are several application specific software packages available for various purposes, PGP, SSH, SSL to mention a few. After writing your e-mail, you use the receiver's public PGP key to encrypt the message before sending it. Some of these methods will most likely become obsolete with the introduction of IPsec.

Another positive feature of IPsec is that it is possible to change for future needs. The various algorithms used for encryption, hashing and so forth can be changed when new ones are proven better and more secure. Unfortunately, this feature poses another problem described below.

Of course, IPsec is not the final solution to all security problems there are, but it is a major breakthrough because of its integration with IPv6 and its design to be invisible and therefore easy to use.

## 10.3 Disadvantages of IPsec

One of the larger problems with IPsec today is the compatibility issue. Some large software manufacturers are known for their desire not to comply with existing standards and might instead develop standards of their own.

One of the advantages mentioned above, the possibility to change the algorithms, is also considered a disadvantage. This is due to the fact that some of the algorithms used as default have already been broken. Even though new ones are available, it is up to the administrators to make sure that they actually use them. Further more, with common security techniques, if the negotiation fails, there will be no communication at all. In the case of IPsec, if the best security can not be negotiated, IKE will try to negotiate the second best and so on. This may result in that less than the best possible security is used, because one of the peers is set to negotiate to low security.

Another problem is increased processor load. The encryption and decryption of all information flowing in and out of a computer is pretty hard work. Many have argued that the use of IPsec will force users to upgrade their hardware. This problem is easily overcome by using an IPsec compliant network interface card, which has a built-in processor designed for this task only. In this way, the computer's main processor will not have to deal with the IPsec specific work.

## 10.4 Applied Network Security

In this section we will provide a discussion about the practical use of network security. One example is the subject of personal integrity. To feel comfortable while using a public network, such as the Internet, you want any personal information to be kept safe, without others having the possibility to use or abuse it. Abuse can vary from pretty innocent jokes such as sending e-mails using another person's address, to illegal ones e.g. abusing stolen credit card numbers or empty bank accounts. Another major subject of network security is the increased use of the Internet for commercial purposes, as you will se below:

### *10.4.1 E-Commerce*

One thing not many people have missed is the talk about e-commerce, or *Electronic Commerce*. Everyone seems to talk about e-commerce and there are new shopping sites starting every month, everything can and must be done on the Internet. The Internet revolution was a thing we saw a couple of years ago, and now e-commerce will be the next thing. Many of the largest retail stores and mail order companies look upon the Internet market as a supplemental market to ordinary mail order. This has to be considered both an opportunity and a possible hazard. [33] An unsuccessful investment in a web site may result in that the ordinary mail order is affected negatively.

There is also a problem with introducing a new and easier ways of doing business, like this. People are used to the face to face way of doing business are sometimes afraid of using their credit card number in forms on the web. Younger people are not as reluctant and they believe more in the technology they are using.

In the year of 2000, HUI (Handelns Utredningsinstitut) predicts that a quarter of all retail stores will provide the possibility to sell through Internet, and this will correspond to three percent of the total sales. The Internet market in Sweden will have a turnover of 2.4 billion SEK, which will correspond to 0.7 percent of retail trade's total turn over, which in turn is estimated to 325 billion SEK [31].

E-commerce is seen as a way of increasing the size of the marketplace and attracting new possible customers, as it now is possible to reach a global market. Small companies can compete under the same conditions as bigger companies. Another aspect of e-commerce is that it is cheaper than an ordinary business, since you do not need sales personnel or expensive stores.

As the market is getting more global and *“electrified”* an issue about security is raised. Since everything is done electronically there is a potential risk with proving that you are the right person. Today you only need to have a card number to buy things on the Internet. There are numerous ways to secure transactions today. One of the most widely used is SET (Secure Electronic Transaction), which was developed by Visa and MasterCard. If you use SET, the bank will guarantee the payment between you and the retailer. [34] The market for e-commerce is too big to be neglected for all involved parties (banks, Visa, MasterCard and retailers), even though someone has to pay when a card has been misused.

Many people are afraid of leaving information about themselves at web sites. According to Netsurvey, 77 percent of the asked persons answered "yes, sometimes" to the question if they would refrain from doing business on the Internet if they have to give up personal information in the process. 14 percent would always leave information. [32] 22 percent of the web surfers are afraid that the information gathered might violate their personal integrity. 54 percent do not want to give up information about themselves to unknown people or organizations and 14 percent even consider the time it takes to fill in a form to be too long. [32]

IPsec might be the solution to these security problems, since it protects all connections including web, FTP, TELNET, mail etc. It is not really necessary to implement IPsec just to make web shopping secure. There is already a secure method for web traffic called Secure Socket Layer (SSL), that provides an encrypted connection from the host to the server without the user’s involvement. SSL provides a secure enough connection with the correct key-lengths. If the data is stored insecurely on the server, not even IPsec, or any equivalent method, may help to provide security. [36]

## 10.5 Conclusions

In this part we will present our actual conclusions by answering the three research questions we asked initially:

*1. What are the security services provided by IPsec?*

IPsec is a protocol suite consisting of the Encapsulating Security Payload (ESP) and the Authentication Header (AH). They have the capability to encrypt and authenticate any kind of traffic passing through the TCP/IP stack. IPsec also has the possibility to change the algorithms used for encryption and hashing to keep up to date with the standards that has proven most secure.

IPsec is equipped with a technique for safe negotiation of encryption keys, the Internet Key Exchange (IKE). IKE is capable of handling the negotiation of keys, key lifetimes, encryption algorithms etc. in a secure and authenticated way.

*2. What are the advantages and disadvantages of IPsec?*

The major advantage of IPsec is its invisibility to the user, and the possibility to change the algorithms used for encryption, hashing, authentication etc. Another positive feature is the integration of IPsec in IPv6, which means that everyone using the new version of the TCP/IP stack will get the security services automatically.

One advantage of IPsec is also a disadvantage. The possibility to change algorithms mentioned above may result in that not the best encryption is used. A second disadvantage is the issue of compatibility; this problem will probably be solved since IPsec has been adopted as an industry standard.

*3. What makes IPsec different from existing security techniques?*

IPsec employs existing cryptographic methods and can make use of new ones as they are proven to be more secure than the ones currently used. The main difference between IPsec and existing security techniques is that IPsec is not application specific, that is, it can be used in conjunction with all existing applications, without having to modify the application. This is possible due to the fact that IPsec is protecting IP, which is the only protocol in the Internet Layer, and all traffic passing up or down the stack has to go through the security checks.

The use of IPsec is completely invisible to the users; they will not have to change the way in which they use their computers. In comparison, to use PGP to encrypt your e-mails you will have to lookup the recipient's public key and encrypt the message, before sending it. This extra step is not needed while using IPsec.

Another advantage of IPsec is that it is developed by the IETF and will be integrated in the next version of IP, IPv6. This will result in that when you buy the next version of Microsoft Windows, you will automatically have a new TCP/IP stack installed which supports IPsec. The development of IPsec has been performed in an open manner, that is, a lot of different people without commercial interest in the development, has reviewed the proposals. Several major vendors have also adopted IPsec as an industry standard, which probably will lead to increased compatibility.

# 11 References

## 11.1 Books, Journals and Papers

**[1]** Axelsson, S. (2000). *Aspects of the Modelling and Performance of Intrusion Detection.* Göteborg: Chalmers Reproservice.

**[2]** Backman, J. (1998). *Rapporter och uppsatser.* Lund: Studentlitteratur.

**[3]** Doraswamy, N. & Harkins, D. (1999). *IPsec – The New Security Standard for the Internet, Intranets, and Virtual Private Networks.* Upper Saddle River, NJ: Prentice Hall, Inc.

**[4]** Hunt, C. (1992). *TCP/IP Network Administration.* Sebastopol, CA: O'Reilly & Associates, Inc.

**[5]** Stevens, R. (1998). *TCP/IP Illustrated: the protocols.* Reading, MA: Addison Wesley Longman, Inc.

**[6]** Schneier, B. (1994). *Applied Cryptography.* John Wiley & Sons Inc.

**[7]** Sjögren, A. (1999). *IPSec end-to-end security: will it change the design of future networks?* Master thesis project performed at Department of Teleinformatics, KTH.

## 11.2 Electronic Documents

**[8]**. IP Security Protocol Charter.
http://www.ietf.org/html.charters/ipsec-charter.html
2000-03-20

**[9]** Malkin, G. (1994). RFC1718 - The Tao of IETF -- A Guide for New Attendees of the Internet Engineering Task Force.
http://www.ietf.org/rfc/rfc1718.txt

**[10]** Atkinson, R. (1995). RFC1826 - IP Authentication Header.
http://www.ietf.org/rfc/rfc1826.txt

**[11]** Atkinson, R. (1995). RFC1827 - IP Encapsulating Security Payload (ESP).
http://www.ietf.org/rfc/rfc1827.txt

**[12]** Kent, S. & Atkinson, R. (1998). RFC2401 - Security Architecture for the Internet Protocol.
http://www.ietf.org/rfc/rfc2401.txt

**[13]** Kent, S. & Atkinson, R. (1998). RFC2402 - IP Authentication Header.
http://www.ietf.org/rfc/rfc2402.txt

**[14]** Kent, S. & Atkinson, R. (1998). RFC2406 - IP Encapsulating Security Payload (ESP).
http://www.ietf.org/rfc/rfc2406.txt

**[15]** Piper, D. (1998). RFC2407 - The Internet IP Security Domain of Interpretation for ISAKMP.
http://www.ietf.org/rfc/rfc2407.txt

**[16]** Maughan, D. & Schertler, M. & Schneider, M. & Turner, J. (1998). RFC2408 - Internet Security Association and Key Management Protocol (ISAKMP).
http://www.ietf.org/rfc/rfc2408.txt

**[17]** Harkins, D. & Carrel, D. (1998) RFC2409 - The Internet Key Exchange (IKE).
http://www.ietf.org/rfc/rfc2409.txt

**[18]** Thayer, R. & Doraswamy, N. & Glenn, R. (1998). RFC2411 - IP Security Document Roadmap.
http://www.ietf.org/rfc/rfc2411.txt

**[19]** Orman, H. (1998). RFC2412 - The OAKLEY Key Determination Protocol.
http://www.ietf.org/rfc/rfc2412.txt

**[20]** Guttman, E. & Leong, L. & Malkin, G. (1999). RFC2504 - Users' Security Handbook.
http://www.ietf.org/rfc/rfc2504.txt

**[21]** Timestep, understanding the IPSec protocol suite.
http://www.timestep.com/downloads/ipsec.pdf
2000-01-14

**[22]** IPsec – a technical overview.
http://www.hsc.fr/ressources/veille/ipsec/papier/papier.html.en
2000-01-27

**[23]** Frösen, J. (1995). Practical cryptosystems and their strength
http://www.tcm.hut.fi/Opinnot/Tik-110.501/1995/practical-crypto.html
2000-01-24

**[24]** RSA Laboratories' Frequently Asked Questions About Today's Cryptography, Version 4.1
http://www.rsasecurity.com/rsalabs/faq/
2000-02-09

**[25]** Jargon. Security through Obscurity.
http://www.wins.uva.nl/~mes/jargon/s/securitythroughobscurity.html
2000-02-03

**[26]** Comp-security FAQ.
http://www.vacia.is.tohoku.ac.jp/~s-yamane/articles/hacker/comp-security-FAQ/secur3.html
2000-02-03

**[27]** Jalkanen, J. & Mättö, M. & Perttunen, J. (1998). Security through Obscurity.
http://haa.iki.fi/luennot/sec98/reports/4/
2000-02-03

**[28]** CERT. (1999, August 11). CERT/CC Statistics 1988-1999.
http://www.cert.org/stats/cert_stats.html
2000-02-24

**[29]** Teldoks årsbok 2000
http://www.teldok.org/tda2000/130kap5-6.pdf
2000-02-22

**[30]** Commission of the European Communities. (1991). Information Technology Security Evaluation Criteria (ITSEC). Version 1.2.
http://www.itsec.gov.uk/docs/pdfs/formal/ITSEC.PDF
2000-04-15

**[31]** Svidén, H. (1999). Dyster prognos för Internethandeln.
http://nyheter.idg.se/display.pl?ID=990418-cs1
2000-05-08

**[32]** Svidén, H. (2000). Fortsatt lågt förtroende för näthandel.
http://nyheter.idg.se/display.pl?ID=000502-CS7
2000-05-08

**[33]** Wallström, M. (2000). En fjärdedel av e-inköpen misslyckas.
http://nyheter.idg.se/display.pl?ID=000308-cs18
2000-05-08

**[34]** VISA. Betala på Internet.
http://www.visa.se/betala/index.asp
2000-05-08

## 11.3 Unpublished Documents

**[35]** Ericsson Mobile Data Design AB. (1996). The competence center for mobile data design.

## 11.4 Personal Communication

**[36]** Simon, R. (rs@sconseil.com) (2000, May, 03). Oral information regarding IPsec and its practical usability.

# 12 Table of Figures

# List of Abbreviations

| | |
|---|---|
| AD | Area Director |
| AH | Authentication Header |
| ATM | Automatic Teller Machine |
| BBN | Bolt, Beranek and Newman |
| BITS | Bump in the Stack |
| BSD | Berkeley Software Design |
| CBC | Cipher Block Chaining |
| CFB | Cipher Feedback |
| CIA | Confidentiality, Integrity, Availability |
| DARPA | Defense Advanced Research Projects Agency |
| DCA | Defense Communications Agency |
| DDN | Defense Data Network |
| DES | Data Encryption Algorithm |
| DNS | Domain Name System |
| DOI | Domain of Interpretation |
| DSA | Digital Signature Algorithm |
| EARN | European Academic Research Network |
| ESP | Encapsulating Security Payload |
| FTP | File Transfer Protocol |
| GNU | GNU’s not UNIX |
| HMAC | Hash Message Authentication Code |
| HUI | Handelns Utredningsinstitut |
| IANA | Internet Assigned Numbers Authority |
| ICV | Integrity Check Value |
| IETF | Internet Engineering Task Force |
| IKE | Internet Key Exchange |
| IP | Internet Protocol |
| IPTO | Information Processing Techniques Office |
| ISAKMP | Internet Security Association and Key Management Protocol |
| ISO | International Standardization Organization |
| ISOC | Internet Society |
| ISP | Internet Service Provider |
| IV | Initialization Vector |
| KPDK | Key Pad Data Key |
| LAN | Local Area Network |
| MAC | Message Authentication Code |
| MD5 | Message Digest Algorithm 5 |
| MTU | Maximum Transmission Unit |
| NIC | Network Interface Card |
| NSA | National Security Agency |
| NSF | National Science Foundation |
| OS | Operating System |

| | |
|---|---|
| OSI | Open Systems Interconnect |
| PFS | Perfect Forward Secrecy |
| PGP | Pretty Good Privacy |
| PMTU | Path Maximum Transmission Unit |
| PRF | Pseudo Random Function |
| RFC | Request for Comments |
| RSA | Rivest, Shamir, Adleman |
| SA | Security Association |
| SAD | Security Association Database |
| SET | Secure Electronic Transaction |
| SHA | Secure Hash Algorithm |
| SHS | Secure Hash Standard |
| SKEME | Secure Key Exchange Mechanism |
| SMTP | Simple Mail Transfer Protocol |
| SPD | Security Policy Database |
| SPI | Security Parameter Index |
| SSH | Secure Shell |
| SSL | Secure Socket Layer |
| STO | Security through Obscurity |
| STS | Station-to-Station |
| TCP | Transmission Control Protocol |
| TLS | Transport Layer Security |
| TOS | Type of Service |
| TTL | Time to Live |
| UDP | User Datagram Protocol |
| VPN | Virtual Private Network |

# Appendix A

## Sample Diffie-Hellman Algorithm

```
#include <stdio.h>

/* Usage: dh base exponent modulus */

typedef unsigned char u;
u m[1024],g[1024],e[1024],b[1024];
int n,v,d,z,S=129;

a(u *x,u *y,int o){
   d=0;
   for(v=S;v--;){
          d+=x[v]+y[v]*o;x[v]=d;d=d>>8;
   }
}

s(u *x){
   for(v=0;(v<S-1)&&(x[v]==m[v]););v++;
   if(x[v]>=m[v])a(x,m,-1);
}

r(u *x){
   d=0;
   for(v=0;v<S;){
          d|=x[v];x[v++]=d/2;d=(d&1)<<8;
   }
}

M(u *x,u *y){
   u X[1024],Y[1024];
   bcopy(x,X,S);
   bcopy(y,Y,S);
   bzero(x,S);
   for(z=S*8;z--;){
          if(X[S-1]&1){
                 a(x,Y,1);s(x);
          }
          r(X);
          a(Y,Y,1);
          s(Y);
   }

}

h(char *x,u *y){
   bzero(y,S);
   for(n=0;x[n]>0;n++){
          for(z=4;z--;)
                 a(y,y,1);
          x[n]|=32;
          y[S-1]|=x[n]-48-(x[n]>96)*39;
   }
}

p(u *x){
   for(n=0;!x[n];)n++;
   for(;n<S;n++)

   printf("%c%c",48+x[n]/16+(x[n]>159)*7,48+(x[n]&15)+7*((x[n]&15)
>9));
   printf("\n");
}

main(int c,char **v){
   h(v[1],g);
   h(v[2],e);
   h(v[3],m);
   bzero(b,S);
   b[S-1]=1;
   for(n=S*8;n--;){
          if(e[S-1]&1)
                 M(b,g);
          M(g,g);r(e);
   }p(b);
}
```