*Master Thesis*
*Computer Science*
*Thesis no: MCS-2003:18*
*June 2003*


# SharkNet

## Cooperation with service providers outside the secure infrastructure

**Vendela Normark**

Department of
Software Engineering and Computer Science
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

This thesis is submitted to the Department of Software Engineering and Computer Science at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. The thesis is equivalent to 20 weeks of full time studies.

**Contact Information:**
Author:
Vendela Normark
Address: Älgbacken 1, 372 34 Ronneby
E-mail: vendela.normark@home.se

External advisor:
Lars Anglert
Ericsson AB
Address: Ölandsgatan 1, Box 518, 371 23 Karlskrona
Phone: +46 455 395000

University advisor(s):
Per Mellstrand / Bengt Carlsson
Department of Software Engineering and Computer Science

Department of
Software Engineering and Computer Science
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

Internet : www.bth.se/ipd
Phone : +46 457 38 50 00
Fax : + 46 457 271 25

# ACKNOWLEDGEMENTS

I would like to thank the following persons for support and criticism of my work;
**Lars Anglert**, Karlskrona, extern supervisor at Ericsson, for assisting in discussions and clarifying the terminology,
**Bengt Carlsson**, Blekinge Institute of Technology, for keeping a critical eye at the writing and finally
**Per Mellstrand**, Blekinge Institute of Technology, for supervising the technical parts of the thesis work.

Ronneby, June 2003

Vendela Normark

# ABSTRACT

This master thesis presents how the authentication is handled in two frequently used protocols. It is a study of the authentication procedure in IPsec and TLS where the techniques have been compared based on facts from literature and practical tests. The results in this thesis are to be used as part arguments for continuous development of cooperation between operators using Ericsson´s charging system and content providers.

**Keywords:** Security, authentication, IPsec, TLS

# CONTENTS

# 1 INTRODUCTION

With the intention to increase choice and profits, many telecom operators today choose to cooperate with external suppliers of services and secondary products. Some of these collaborations require the external providers to be able to access parts of the infrastructure e.g. orders for payment. Can this cooperation be accomplished without jeopardizing the company's security? Most companies have reached a certain level of security awareness and have taken actions to protect their company from malicious intruders. But how could this "sharknet"[1] be combined with the possibility to cooperate with external partners?

## 1.1 Purpose and Scope

This thesis intend to investigate whether it is possible to let a *Content Provider* (CP) charge the prepaid customers of Ericsson's *Charging System* (CS) in real-time through a web based communication. The communication channel should be easy and quick to set up, be secure for both parts and be easy to close down if any hesitation should appear about it's credibility. The dialogue between the end user and the content provider will not be considered in this thesis, a charging request from a content provider is in this thesis always considered to be correct.

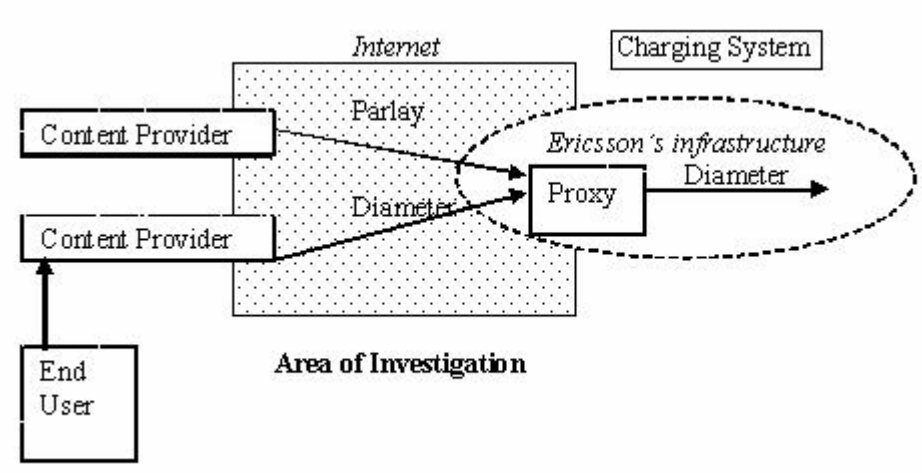


*Figure 1.1 SharkNets area*

[1] An analogy of the secured company as a bay separated from the insecure Internet sea with the type of net used to keep sharks away from bathing beaches

Figure 1.1 clarifies the investigation area for this thesis work. The *End User* contacts the content provider to order a product or service. The content provider makes a connection to the charging system through a web based channel using the Parlay [17] or Diameter [15] protocol, charges the end user's prepaid account and delivers the product. The protocol used inside the charging system is Diameter.

A research question has been formulated to focus upon:

- *How should the content provider be authenticated?*

The question will be answered by information found in literature and additional facts from the results of the practical tests conducted. The aim is to describe how the authentication is handled in the IPsec and TLS protocols. There will also be a discussion concerning who should issue the certificates – the operator or a trusted third party?

## 1.2 About this Report

This report starts with a clarification of some basic practical security terms and descriptions of the two protocols IPsec and TLS. After that the tests with results are presented and discussed. The report ends with the presentation of conclusion based on the referenced facts and practical test results.

When the term operator is used in this report the term refers to a telecom operator using Ericsson´s charging system.

# 2 THE BASIC MECHANISMS

A chain is only as strong as its weakest link and a security system is no exception to this saying. In order to manage secure protocols and products the underlying techniques must be functional and secure. Some of these underlying mechanisms will be presented here.

The chapter is based on information found in [5], [7], [8], [9], [13] and [14].

## 2.1 Key

In computer and information security a key is a secret value and there are various techniques to derive, distribute and use these keys. Different use of keys requires different key lengths in order to prevent a successful attack.

For further details see [5].

## 2.2 Random Number Generator

Since almost every computer security system using cryptography need random numbers for keys, unique values in protocols etcetera, the system is dependent on the randomness of these numbers. A random number is supposed to be unpredictable, irreproducible and have the equal probability of occurring as all other numbers. Random numbers can be generated using hardware, software or a combination of both.

For further details see [8].

## 2.3 Symmetric Encryption

In symmetric encryption the same key is used for both encrypting and decrypting the information. This method is fast and reliable but only as long as the key is really kept secret. Distributing symmetric keys can be a risk of exposure and is known to be hard to manage in a secure way.

Two frequently used concepts in encryption are *Plaintext* and *Ciphertext*. Plaintext is the information in its original form before encryption, and ciphertext is the same information in its encrypted form.

For further details see [7].

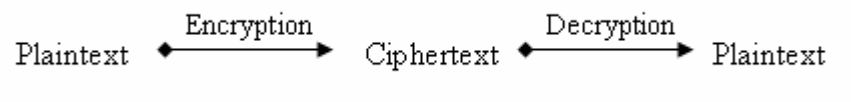


*Figure 2.1 Encryption and decryption*

## 2.4 Asymmetric Encryption

In an asymmetric encryption there are two keys involved: one openly distributed public key used for encryption and one private key used for decrypting the message. Asymmetric encryption is more time consuming than symmetric, but key distribution is easier to handle with this technique. Asymmetric encryption is also known as *Public Key Encryption.*

For further details see [7].

## 2.5 One-Way Hash-Function

The hash function is a method where variable sized input generates fixed sized output. The function is known to be irreversible and collision resistant. This means there should be no way to reverse the function and the outcome is unique. It should not be possible to find two different inputs that generate the same output. Hash functions are often used as a base for algorithms or in combination with other security tools such as encryption.

For further details see [5].

## 2.6 Message Authentication Code

A *Message Authentication Code* (MAC) is used to ensure that the contents of a document or message have not been changed.
The most commonly used MAC is the *Hashed Message Authentication Code* (HMAC). HMAC takes a shared secret key and a message as input and outputs a fixed number of bits. The bits created are sent with the message, and the receiver calculates a HMAC of his own with the message and the same shared key. If the created bits match the attached, the content should not have been changed.

For further details see [9].

## 2.7 Certification

A certificate is used to ensure the correctness of the stated identity of the certificate holder and to connect that identity with a key. Certificates can be of different classes depending on the actual verification done in order to establish the ordering party's true identity. The supplier of a certificate is often a trusted third party who charges the holder for the certificate. The supplier's good name is the insurance for the credibility.

For further details see [14].

## 2.8 Digital Signature

When authentication is performed using a *Digital Signature*, a hashed value is signed with a private key. The signed hash is sent in plaintext together with the certificate holding the public key, see 2.4, and is verified by the receiving party. Alternative ways to use digital signatures are for example with preshared key. Digital signatures assure both the identity of the sender and the integrity of the message.

For further details see [14].

# 3 IP SECURITY PROTOCOL

One of the protocols used in the empirical tests is IPsec. This chapter is an overview of the protocol explaining some of the basic terms.

The chapter is based on information found in [1], [2], [3] and [4].

IPsec is a protocol, or actually a set of protocols, working in the network layer in the Internet Protocol stack. It provides cryptographic security services offering *authentications, integrity, access control* and *confidentiality*. IPsec is transparent in relation to the application, allowing the use of any IP protocol over it. To make it even more flexible, the protocols in IPsec are designed to be algorithm-independent.

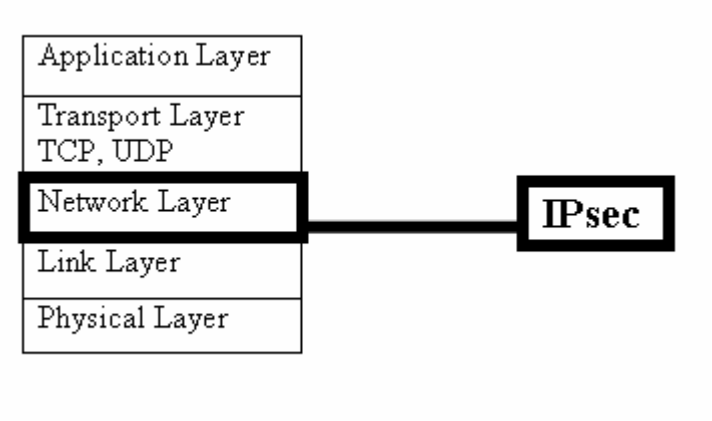


*Figure 3.1 The place of the IPsec in the Internet Protocol stack*

Three of the protocols in IPsec are *Authentication Header* (AH), *Encapsulating Security Payload* (ESP) and *Internet Key Exchange* (IKE) protocol. The decisions made about which protocols to use in a specific communication are defined in a *Security Policy* (SP). There has to be one SP defining outgoing traffic and one for incoming and the policies are stored in a *Security Policy Database* (SPD). This SPD is then consulted for each package passing in or out.

## 3.1 Preparing a Communication

IPsec uses shared secret keys for the different protocols and before starting up a communication there has to be an authentication and distribution of these keys. IPsec supports both manual and automatic key distribution.

Manual management of keys means that a person manually configures each system with keying material and security association management data. These techniques will likely work best in a small, static environment or when only selected communication channels need to be secured. The risk of deficient maintenance in a manual managed system grows with a larger or dynamic system because there are more associations to keep in record. A bad maintenance might lead to exposing the system to risks.

Automated key management is supported by different key-exchanging-protocols in IPsec, and the default protocol is *Internet Key Exchange* (IKE). Except from the actual key-exchange IKE also includes a choice of common algorithms and how often the keys are to be replaced.

After proper authentication and decisions based on keys and algorithms, all this information is gathered in a *Security Association* (SA). Each SA can only support one protocol, AH or ESP, and one direction. This means that setting up a communication will at least require the making of two security associations. All manufactured security associations are stored in a *Security Association Database* (SAD).

## 3.2 Protocols

There are two main protocols in IPsec, AH and ESP, to be used one at a time or in combination. In short the difference between them is that **AH provides authenticity** and **ESP provides confidentiality** (both authentication and encryption). The protocols can be used in either transport mode or tunnel mode. Which protocol to use depends on what type of security is wanted for the issue.

### 3.2.1 Authentication Header

The IP *Authentication Header* (AH) provides connectionless integrity, data origin authentication and an anti-reply service. To verify that the information sent have not been modified, and to ensure the sender´s identity, AH calculates an *Integrity Check Sum* of the all parts of package that are supposed to go unchanged during the transfer, including most of the header. The check sum is generated by a hashed message authentication code. This checksum is then used in the new AH header in addition to *next header, payload length, security parameter index* and *sequence number*. The sequence number assures that each package will not be sent repeatedly without the receiver specifically asking for it e.g. in the case of a lost package. The new AH header is inserted after the original IP header.

| IP Header | AH | Payload |
|---|---|---|

*Figure 3.2 IP-package using AH-protocol in transport mode*

For further details see [2].

### 3.2.2 Encapsulating Security Payload

The *Encapsulating Security Payload* (ESP) protocol provides confidentiality. ESP adds both a new header and a tail to the package and encrypts the parts after the header. The header consists of *security parameter index, sequence number,* and the tail consists of *padding, pad length* and *next header*.

| IP Header | ESP head | Payload | ESP tail |
|---|---|---|---|

*Figure 3.3 IP-package using ESP-protocol in transport mode*

ESP may also provide authentication using a checksum like AH, but since the checksum is only calculated over the payload and not the IP header, it cannot guarantee anything about the sender. When using authentication an extra unencrypted field is added at the end of the tail.

For further details see [3].

## 3.3 Operation Modes

IPsec has two different operation modes supporting both AH and ESP. They can be used themselves or in a combination. The transport mode supports the communication between two hosts while the tunnel mode supports traffic between host and network, and between two different networks. The figure 3.4 shows a combination of using both tunnel and transport mode. In this illustration the tunnel has been set up between the two gateways and a transport connection can thereby be managed between host A and B.

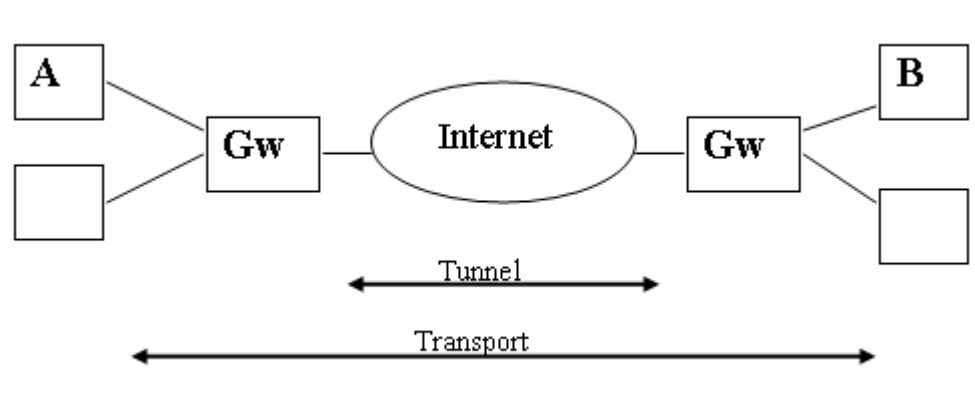


*Figure 3.4 Combination of tunnel and transport mode*

A host must support both transport and tunnel mode but a gateway is only required to support tunnel mode.

For further details see [1].

### 3.3.1 Transport Mode

The transport mode can only support the communication between two hosts, so-called end-to-end communication. In this mode the security headers are added to the package between the IP header and the payload.

| IP Header | AH | Payload |
|---|---|---|

*Figure 3.5 IP-package using AH-protocol in transport mode*

For further details see [1].

### 3.3.2 Tunnel Mode

In the tunnel mode the protocol adds an extra IP header in front of the package to be sent and is used when at least one end of the secure connection is a gateway. The security header appears between the outer IP header, specifying the IPsec processing destination, and the inner IP header specifying the ultimate destination for the package.

| IP Header 2 | ESP head | IP Header | Payload | ESP tail |
|---|---|---|---|---|

*Figure 3.6 Example of a package in tunnel mode using ESP*

For further details see [1].

# 4 TRANSPORT LAYER SECURITY

The other protocol used in the empirical tests is TLS. This chapter is an overview of the protocol and starts with a clarification of the terms TLS and SSL.

The chapter is based on information found in [6], [10], [11] and [12].

*Secure Socket Layer* (SSL) is a standard originally developed by Netscape to secure Web transactions on the Internet. The protocol was later adopted by the *Internet Engineering Task Force* (IETF) and renamed *Transport Layer Security* (TLS). TLS 1.0 is equivalent to SSL 3.0.

TLS is implemented between the transport layer and the application layer and is composed by two protocols: the TLS *Record Protocol* and the TLS *Handshake Protocol.*

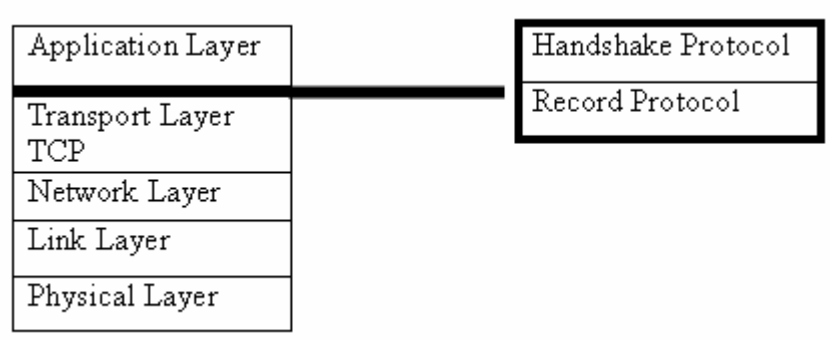


*Figure 4.1 The place of TLS in the Internet Protocol stack*

The protocols in TLS are supposed to offer a confidential pipe between the participating client and server, and also to provide a possibility for them to authenticate themselves. A connection is divided into two phases: the handshake and the data transfer.

## 4.1 Preparing a Communication

Setting up a communication with TLS always includes a client and a server, and it starts with a handshake. In this initial state the terms for the data-exchange shall be specified in these steps:

*Hello Message*
First the client and server exchange information to reach an agreement upon what cryptographic and compression algorithms to use, and share random numbers for generation of keys. In this step the communication is also given a unique session id that allows the reuse of keys for a certain period of time.

*Certificate and Key Exchange*
In the next step the server, and sometimes also the client, identify themselves using a certificate. The keys are derived both at the server and the client side.

*Change Cipher*
When the certificate steps are concluded a cipher change message will be exchanged that informs both parts that it is time to change to the symmetric key and start the real transmission. A check will also be conducted to reassure that no tampering has been made.

If a client reconnects to a server running TLS before the session has expired the client sends the session ID to indicate it wants to resume. The server can then reopen the communication with the use of the keys derived earlier.

For further details see [10] and [12].

## 4.2 Protocols

### 4.2.1 TLS Handshake

The handshake protocol is a combination of four different protocols, see figure 4.2, and is placed right below the application.

| Handshake | Alert | Application Data | Change Cipher spec |
|---|---|---|---|

*Figure 4.2 The four protocols in the handshake protocol*

*Handshake Protocol*
This is the protocol defining all the steps and the conditions in the handshake described in 4.1.

*Alert Protocol*
The alert messages convey the severity of the message and a description of the alert. The alert can be of two types: closure alert or error alert, where the former is an assurance that both server and client know that the connection is ending, and the latter is used for error reporting.

*Application Data Protocol*
The application data protocol is the protocol controlling the data sent from the overlying application.

*Change Cipher Specification*
This protocol consists of a single message that signals transitions in ciphering strategies like described under *Change Cipher* in 4.1

For further details see [6] and [12].

### 4.2.2 Record Protocol

The data sent in a TLS connection are packed in records able to hold $2^{14}$ bytes unencrypted data. This record, and the application data protocol in 4.2.1, provides the information that is necessary for the receiving implementation to interpret the record.

| Type | Version | Length | Payload |
|---|---|---|---|

*Figure 4.3 TLS record*

*Type* indicates the type of the message, alert, application data, handshake or change cipher specification. The *version* field is a check to assure that each side agrees on the same TLS version. *Length* is the size of the data and finally the *payload* is where the data is placed. There can be a MAC at the tail of the package to. Packages larger than the allowed $2^{14}$ bytes need to be either fragmented or compressed with the compressing algorithm agreed upon during the handshake.

For further details see [12].

# 5 AUTHENTICATION

This chapter describes how the authentication is handled in connections using different authentication alternatives. The chapter consists of some background information from literature and also a presentation of the practical tests made for the thesis. The test results will be compared and combined with the presented facts from literature and summarized with a conclusion. A discussion concerning who issues the certificates is also included – is it the charging system operator or a trusted third party?

The presentation of the existing circumstances concerning the charging system and the requirements for the tests expunge from discussions with Lars Anglert, advisor at Ericsson in Karlskrona.

## 5.1 Background

Authentication is the process of identifying an individual, usually based on a username, password or certificate. In security systems authentication is distinct from *authorization*, which is the process of giving individuals access to system objects based on their identity. **Authentication ensures that the individual is who he or she claims to be**, but says nothing about the access rights of that individual.

The thesis will look into the supposed communication between a content provider and an operator. The communication will be in real-time why **performance** is of great matter. It is cooperation with an external partner in a different security domain and the communication will take place over an untrustworthy media. This calls for a strict **security policy** to be enforced with focus put on authentication. The operator wishes to be able to **revoke** the cooperation on a very short notice if dissatisfaction should appear for some reason. Even if the cooperation is of the satisfactory kind there should be a regular update on confidence to assure that only content providers of current interest has access to the charging system. All transactions between the providers and the operators are to be kept on record to guarantee a possible backward check if irregularities should be discovered.

## 5.2 Available Alternatives

Authentication may be performed at various layers of the Internet Protocol stack, see table 5.1.

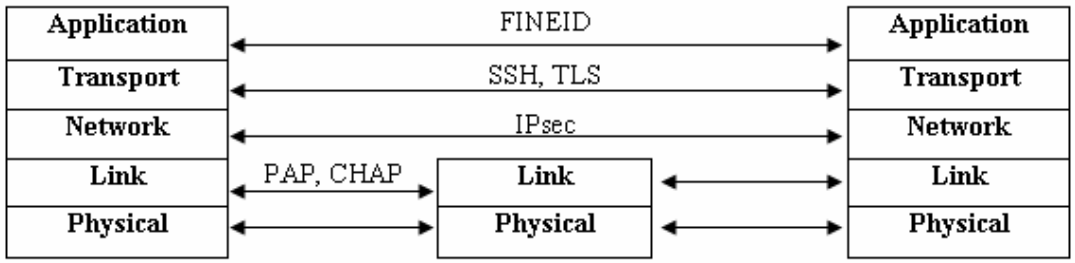


*Table 5.1 Authentication at different layers in the Internet Protocol stack [16]*

This chapter compares the authentication procedure in IPsec and TLS. They were chosen because they are two of the most commonly used security protocols and they are well defined in publicly available material. Both have been around for some time and there are not widely presented flaws in the present versions, suggesting they are secure alternatives. The use of IPsec and TLS is supported by the protocols Parlay and Diameter.

## 5.3 IPsec

There are four different authentication methods allowed in *Internet Key Exchange* (IKE) in IPsec; authentication with *Digital Signature*, two forms of *Public Key Encryption* and one method using *Preshared Keys.* The authentication procedure is a two phase task where the first phase has two alternatives, a full negotiation called *Main Mode* and a quicker setup, with fewer messages sent, called *Aggressive Mode.* Aggressive mode is considered to be less secure because the increased amount of data transferred in each package can provide a presumed eavesdropper more information than the main mode. Aggressive mode is not compulsory according to IPsec standard and will not be used in the practical tests.

Main mode, illustrated in figure 5.1, sends three pairs of messages. In these messages it is settled what algorithms to use, data for generating keys is exchanged and finally an authentication is made in the agreed way.

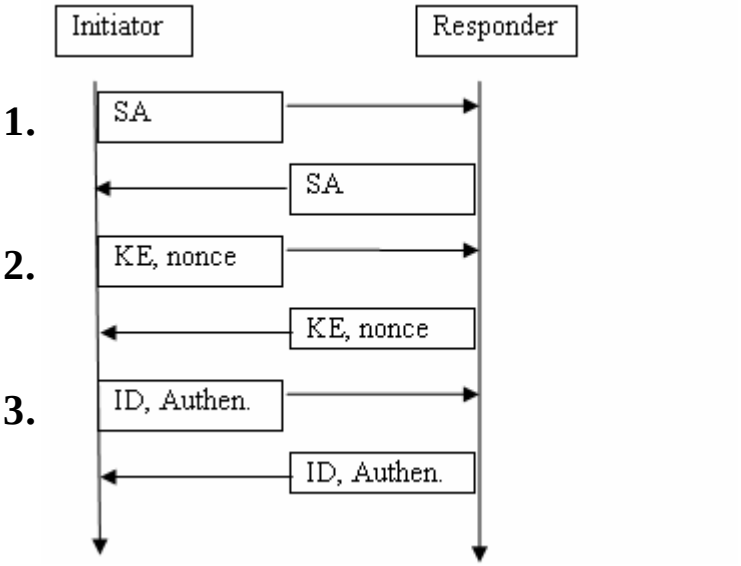


*Figure 5.1 Creating a new SA in main mode*

The *Security Association* (SA) in the first messages in figure 5.1 contains information about what algorithms that are supported by the communicating party. In the second step KE is the *Key Exchange Data,* information to act as foundation for creating keys. *Nonce* is a *Number used Once*, a random value used for key generating. The ID payload is a package sent in the third step containing information about which authentication method, protocol and port to use.

Following the conclusion of main mode the second phase begins, negotiating in *Quick Mode.* The security association produced in phase one lay ground for setting up a new SA to be used during the communication. All messages sent during this mode are encrypted, which is why more information can be sent in each transfer.

### 5.3.1 Digital Signature

The use of digital signatures is supported in both main and aggressive mode and the nonces used are sent in plaintext. The authentication is performed by identifying the signed and verified hashed value of the entire ID payload.

For further details see [4].

### 5.3.2 Public Key Encryption

Public key encryption is supported by both main and aggressive mode and is based on the condition that the parties are aware of each others public keys. The nonces are sent encrypted and the authentication is made by a plain hash value, not signed and verified like in the case with digital signatures (5.3.1).

For further details see [4].

### 5.3.3 Revised Mode of Public Key Encryption

Revised public key encryption is described to have significant advantages to ordinary public key encryption. The difference from the previous method is that the dialogue here uses symmetric encryption directly after the first message.

For further details see [4].

### 5.3.4 Preshared Keys

This authentication method was not described as detailed as the others in the research material. Like in the case with public key encryption the hashed value of the ID payload is directly authenticated but here the nonces are sent in clear.

For further details see [4].

## 5.4 TLS

Authentication in TLS is handled by certificates. In so-called asymmetric authentication only the server identifies itself but in case of symmetric authentication the client must identify itself too using a client certificate.

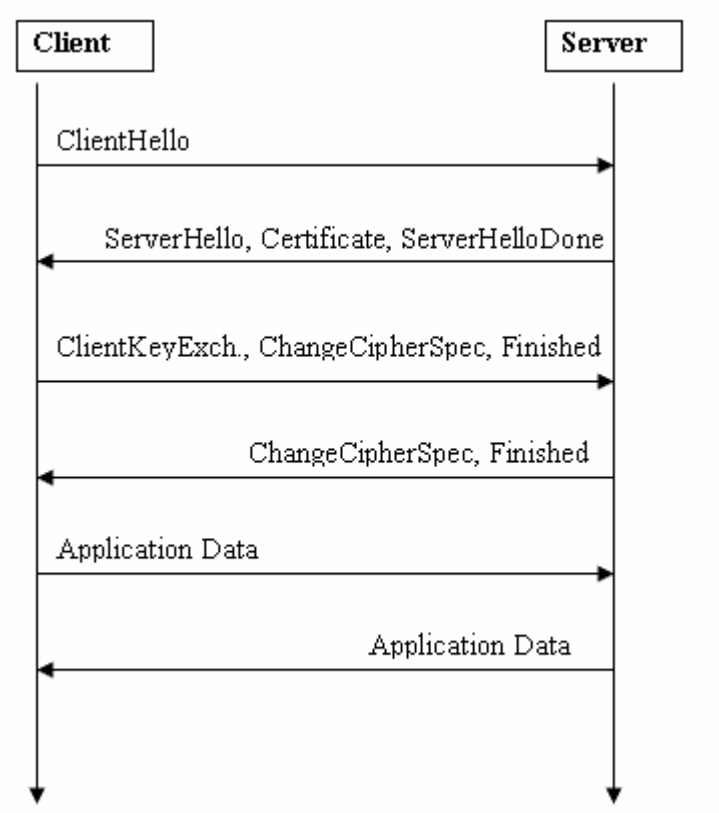


*Figure 5.2 Full handshake in TLS, asymmetric authentication*

Figure 5.2 shows a *Full Handshake*, the authentication procedure when a new connection is set up between a server and a client. If the client reconnects to the

server before a settled time has expired there can be a simplified version of the handshake containing just four messages instead of the presented six in a full handshake.

The content of the different messages has been presented earlier in chapter 4.1 and further details can be found in [12].

## 5.5 Certification

A certificate can be issued either by one of the communicating parties or by a trusted third party. The advantages of a *third party issued certificate* are the shared trust, the simplicity and the reduced work load for the certificate holders. In this case the certificate is ordered and paid for from a certification authority that both parties trust.

The advantages of a *self-issued certificate* are reduced issuing-costs and the possibility for the parties themselves choosing information that secure the identity of the certificate holder.

To attain information on what arguments to put in the adjustment for third party or self-issued certificates, contact was made with a certificate holder well-known to the students at *Blekinge Institute of Technology* (BIT): DAT[2].

BIT's webmail[3] uses a homemade certificate that makes the login procedure pass through an alert notice for each login attempt claiming a non-trustworthy certificate. DAT explains their choice of a self-issued certificate with an immediate need for a certificate when setting up the webmail, and after that it has not been changed. Before the next release of the webmail there will be a discussion of what type of certificate to use.

## 5.6 To Terminate Cooperation

*Access Control List* (ACL) is a technique combining two lists; one *user list* and one *access control list*, where the first combines users and certificates and the latter specifies the operational rights for each certificate. If there is a wish to revoke a client, a change can be made in the ACL so that the client still has a valid certificate but has no access rights. The technique is simple, fast and a trustworthy method. ACL can be used both in IPsec and TLS.

For further details see [12].

[2] DATorenheten – the unit responsible for computer services at BIT
[3] Remote access to the school mail account

## 5.7 Practical Test

A number of practical tests have been conducted during the work of this thesis. The tests where conducted for two purposes; to give a deeper knowledge of IPsec and TLS and also to produce measurable data to act as a foundation for the conclusions to be drawn. The tests were conducted in three steps. First a connection between two computers was arranged, where one computer act as server and the other as client. In the next step various settings of IPsec were made, and also with TLS. In the last step a small file was transferred between the two computers. The file transfer was studied from two aspects, the time consumed and the packages transferred. The methods of measurement are described in 5.7.1.1 and 5.7.1.2.

The connection used during the tests is described in 5.7.1. The results from the configuration part are presented in 5.7.2 and 5.7.3 in describing and evaluating texts, reflecting the facts from research literature. The results from file transfer are presented in diagrams and tables with additional comments in 5.7.4 and 5.7.5.

### 5.7.1 Setting up the Connection

Ericsson's charging system is used by different operators in different environments concerning hardware and software. This brought about that no specific environment was required to use as the base for this investigation.

The tests were started in an already existing environment in the *Security Engineering Laboratory* (Seclab) at BIT in Ronneby, because the environment was known to have an IPsec version included. The intention was to use Windows (win) XP at both server and client but this had to be reconsidered. There was no web-server included in the used version of XP and trying to find a web-server meeting all the qualification needed for the tests proved to be hard. There are many web-servers that are free of charge publicly available at the Internet, but none was found that was compatible with win XP and also supports the use of TLS.

A new installation was made with the *Operating System* (OS) win 2000 Server at the computer acting as server.

| **Computer** | **A – the server** | **B – the client** |
|---|---|---|
| **CPU** | Intel Pentium 4<br>2,80 GHz | Intel Pentium 4<br>2,80 GHz |
| **Memory** | 512 MB | 512 MB |
| **OS** | Microsoft Windows 2000 Server, 5.00.2195 | Microsoft Windows XP Professional Version 2002, Service Pack 1 |

*Table 5.2 Computers used in the test*

Both win XP and win 2000 Server has an included version of IPsec and both support the use of TLS.

A web-server was installed at computer A, it was *Internet Information Service* (IIS) included in the OS that supports the use of TLS. To use TLS a certificate has to be installed. The certificate used during these tests was a server certificate using 1024 bits key length. To be able to use the included version of IPsec no extra preparations had to be made.

The tests have been conducted in a closed environment in order to limit the possible sources of disturbance aiming for results as reliable as possible. In the tests have been used the existing 100Mbit connection at the *Local Area Network* (LAN) in the Seclab. It is a switched LAN, meaning the device that filters and forwards packages between the LAN segments typically works at a lower level than for example routers. Lower level in these circumstances generally means higher speed but fewer granularities. The connection set up for the tests is illustrated in figure 5.3.

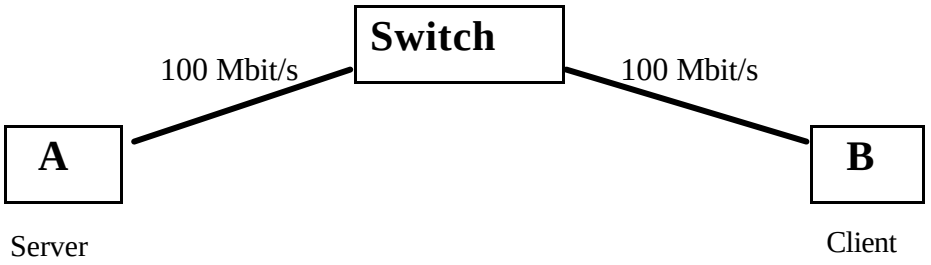


*Figure 5.3 The test setup*

The test procedure was to send a small file between two computers, measure the time for the transfer and analyze the packages sent. The transferred file was a small text file of 16 bytes and was made accessible for the client through the web-server.

The tests have been conducted with different configurations in IPsec and using TLS. Additional tests have also been conducted with no authentication, to have a comparable value. In each test case the client B has made the connection with the server A and has transferred the file back to B. All the measurements have been made at B.

#### *5.7.1.1 Tools Used for Time Measurement*

*Cygwin* was installed at the client computer to make it possible to use certain UNIX programs and commands in the Windows environment. Cygwin supports the use of *wget*, a tool that retrieves files from the World Wide Web. Wget was used in the tests in combination with the UNIX command *time*. The time command obtains the time used for completing a certain task, in this case the time for the client to connect to the server, downloading the file and saving it. The results from the time command are presented in milliseconds.

#### *5.7.1.2 Tools Used for Package Analyse*

To analyse the package transfer for each connection *Ethereal* was installed at the client. Ethereal is a package sniffing tool publicly available and free of charge. This tool was used because it was previously known to provide very detailed and clear information of the packages transferred. Using a familiar tool limits both the startup time and also the potential risk of wrong measurements. The only disadvantage experienced using Ethereal was that no simple way was found to save the results in text format, complicating the possibility to bring the produced files outside the Seclab.

### 5.7.2 Setting up IPsec in Practice

It is well documented in research literature that IPsec is difficult to set up due to the many configuration alternatives. The IPsec versions in win 2000 Server and win XP use IKE, and have a relatively limited choice of settings.

#### *5.7.2.1 Authentication Methods*

There where three authentication methods allowed in the used IPsec versions; *Kerberos*[4], *Certificate* and *Preshared Keys*. The default setting was Kerberos.

The authentication method used in the tests was preshared keys because it seemed to be the most flexible alternative. Kerberos requires both the communicating parties to be in the same domain and the need of a trusted third party. Using a certificate requires issuing certificates for both server and client. The tests aimed to compare several different alternatives of settings, and using preshared keys made it easy to quickly change the conditions for the identification by changing the keys. The used keys had a key size of 128 bits.

#### *5.7.2.2 Ping Tests*

When learning how the different IPsec settings worked between the client and server ping tests where conducted. Each test was made by using the command *ping*, sending four packages of 32 bytes each between the two computers and awaiting a reply. This kind of test is commonly used to determine whether a specific IP address is accessible or not.

In the reply from a sent ping three different types of messages were displayed; *reply*, *negotiating security* or *time out*. In the case of *reply* the connection worked fine, the both parties where able to connect. *Negotiating security* was usually the answer from the party requiring the higher level of security if an asymmetric security setting had been made. When using mismatching preshared keys at the client and server in transport mode, both parties displayed *negotiating security*. In some cases *negotiating security* came prior to a *reply* when setting up a new connection. *Time out* was displayed at the party trying to connect to a higher security level than it supported.

[4] Authentication protocol using a trusted third party to distribute keys, for further details see [13]

The experienced advantages of ping tests over connections to the webpage in this initial state were both a quicker and clearer result. When connecting to the webpage the page was in most cases already available in the cache, meaning a connection seemed to be possible when it was really not. To have a reliable result the cache had to be emptied before each test, which is the reason the ping tests were quicker. When connecting to the webpage, all that could be seen was whether or not the page could be displayed. There was no information about negotiating security or time out which is why the ping tests have to be considered as a more detailed and clear alternative.

#### *5.7.2.3 Reflections*

The default settings for IPsec in the two used OS differ, which is why one has to be careful when choosing the predefined alternatives for AH respectively ESP. Setting up a connection using tunnel mode requires a *remote access* alternative to be used. This is not explicitly stated in research material. The tunnel worked fine even though mismatching keys where used in server and client. Once the tunnel was defined it seemed like no authentication check was made. When applying a new tunnel, a minor delay was noticed, indicating that this was the first connection through the tunnel.

It took some time to get all the settings right in IPsec but when it was done it was perceived as a quick and transparent method.

### 5.7.3 Setting up TLS in Practice

Setting up TLS in the used IIS was a very fast procedure compared to the settings made in IPsec. It was possible to choose if to require TLS-connections or if other traffic also should be allowed to the server. To connect to the server when using TLS one had to use *https://* instead of *http://* in the URL address.

#### *5.7.3.1 Reflections*

The experienced difficulties when using TLS were the issuing and installment of the required certificates. The information available in the Internet on how to make *certificate requests*[5] and certification authorities has not been applicable for the Windows environments used during tests. The certificate used in the tests was a server certificate that expunge from a certificate request generated in the server. The certificate request was signed by a certification authority created in an earlier project by one of the advisors.

Using a third party issuer or having better knowledge in issuing ones own certificates would solve the problem, so from Ericsson's perspective this problem can be ignored, though it should be mentioned for future testing.

[5] A foundation necessary to create a certificate

### 5.7.4 Time Comparison

The following section is a presentation of the results of measuring the connection time together with some observations. The values are gathered using a combination of the *time* command and *wget* when making a transfer of a small file from the server to the client. The test has been conducted with six different setups, four with IPsec, one using TLS and one with no authentication to have a comparable value. Each transfer has been measured five times and calculated into a presented mean value in order to limit the possibility of disturbance influencing the values.

Values presented as *first connection* are data measured in the initial connection after resetting the security parameters. Such connections are likely to include full security negotiating, main mode and quick mode for IPsec and full handshake for TLS. *Following connections* are reconnections made in a rapid sequence after the initial connection.

The following data should be viewed while considering a reasonable fault margin due to possible sources of errors.

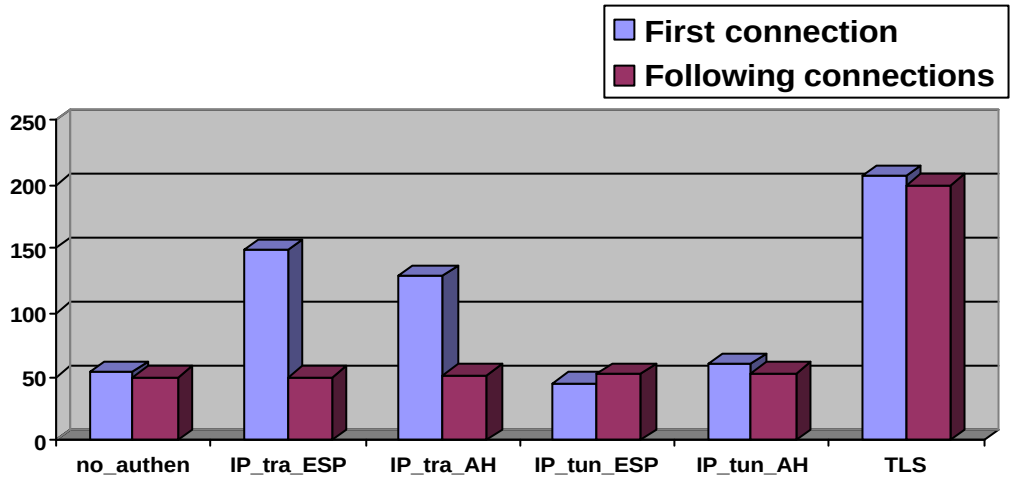


*Figure 5.4 Test results presented in ms*

| Authentication | First Connection | Following Connections |
|---|---|---|
| **no_authen** | 54 | 50 |
| **IP_tra_ESP** | 149 | 50 |
| **IP_tra_AH** | 129 | 51 |
| **IP_tun_ESP** | 45 | 52 |
| **IP_tun_AH** | 60 | 53 |
| **TLS** | 207 | 200 |

*Table 5.3 Test results presented in ms*

#### *5.7.4.1 IPsec Transport Mode ESP/AH*

The differences in values of the first transfer compared to the values from reconnecting are noticeable. In the first connection the security negotiation was made in both main mode and quick mode, see 5.3. In the following connections no difference was noticed compared to the values of no authentication.

Additional tests were made using only quick mode for the same connection. The results of these showed that the times were only slightly greater than the mean values. The difference was not large enough to be reliable due to the fault margin.

#### *5.7.4.2 IPsec Tunnel Mode ESP/AH*

The tests using tunnel mode have not shown any time difference compared to the tests using no authentication. No increased initial value has been noticeable in tunnel mode even though setting up the connection repeatedly.

No measurable differences have been noticed between AH and ESP in either tunnel or transport mode during tests.

#### *5.7.4.3 TLS*

TLS uses considerably longer time for the test and have no noticeable difference between the first setup and the mean value as it was when using IPsec transport mode. The previously presented scenario with a client reconnecting to the server using just a short handshake does not clearly distinguish itself in the results. It seems like the client uses just as much time when reconnecting. Several measurements have been made using different time intervals aiming to get a larger initial value.

Another interpretation of the observed result is that all the shown values are made from initial contacts. The results pointed in that direction even though five measurements made in a rapid sequence is likely to show at least some reconnection. No time value has been found in literature to indicate for how long time the client can reconnect to the server.

The tests with TLS have been made using asymmetric authentication which means that only the server identifies itself. The test results for asymmetric authentication showed TLS to be so much slower than IPsec, this lead to a decision together with the advisor not to look at symmetric authentication for TLS. The symmetric authentication uses more packages when setting up the initial contact and is therefore presumed to be even more time consuming than asymmetric, making it of no interest for this thesis.

#### *5.7.4.4 Summary of the Time Comparison*

The tests results show different values for IPsec transport mode, IPsec tunnel mode and TLS connections. IPsec has shown significant lower time values than TLS during the tests. According to the presented values, IPsec in tunnel mode is the quickest alternative.

### 5.7.5 Package Analyse

The following section is a presentation of the results from the package analyse. The values are gathered using Ethereal. Both the time measurement presented in 5.7.4 and the package analyse have been made from the same transfers using the same six configurations.

In some measurements the produced Ethereal file has shown two *Address Resolution Protocol* (ARP) packages. The ARP packages have been sent in order for the client to find the address to the server. These packages have been excluded in the presented data because they are irrelevant to the task.

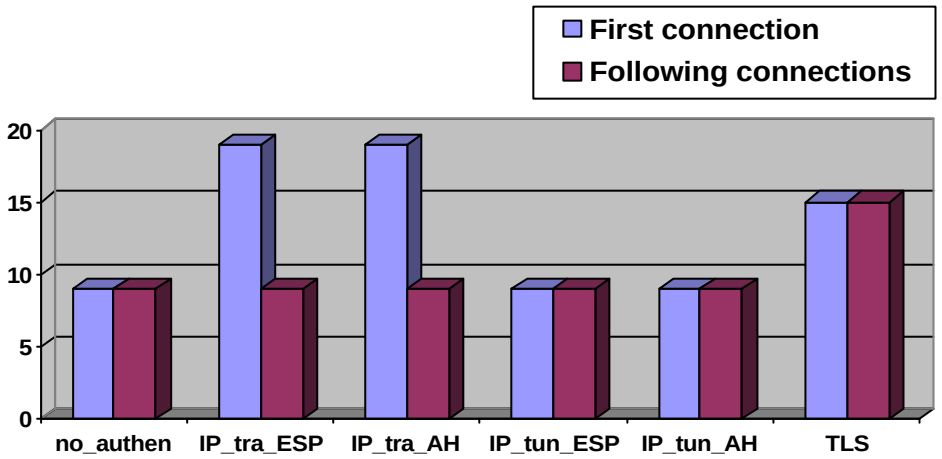


*Figure 5.5 Test results in number of packages sent*

| Authentication | First Connection | Following Connections |
|---|---|---|
| **no_authen** | 9 | 9 |
| **IP_tra_ESP** | 19 | 9 |
| **IP_tra_AH** | 19 | 9 |
| **IP_tun_ESP** | 9 | 9 |
| **IP_tun_AH** | 9 | 9 |
| **TLS** | 15 | 15 |

*Table 5.4 Test results in number of packages sent*

#### *5.7.5.1 IPsec Transport Mode ESP/AH*

The results for IPsec transport mode shows the first setup negotiating in both main mode and quick mode. In the following connections the extra header and, in occurring cases, tail has been noticed inside the packages sent but no extra packages has been generated. Both the main mode and the quick mode negotiation were clearly displayed in the produced Ethereal file and confirmed the research materials descriptions of the setup procedure.

In the tests using only quick mode, four extra packages were displayed instead of the ten extra presented for full negotiation.

#### *5.7.5.2 IPsec Tunnel Mode ESP/AH*

No extra packages were noticed in the first setup connection using tunnel mode even though measurements were repeated. This brings support to the assumption made in 5.7.2.3, that the tunnel is arranged already when applying the settings. In transport mode the ESP packages were evidently in the produced Ethereal file but they were not visible using tunnel mode.

#### *5.7.5.3 TLS*

In all the tests with TLS the displayed handshake procedure was identical to the full handshake presented in 5.4. This supports the theory that all the measured connections show initial values. Further reading was made in order to find support for the results and this was found:

"*There is no requirement that either client or server agree to rehandshake. An implementation which does not wish to rehandshake can simply ignore the message.*" [12].

The conclusion is that the implementation did not support the use of rehandshake and all the presented values for TLS in this test are from full handshake.

#### *5.7.5.4 Summary of the Package Analyse*

This package analyse has provided clear results to be determined that the test results using TLS are from full handshake. Both in the case with TLS and IPsec transport mode the displayed results were identical to the negotiating described in literature. In the case with IPsec tunnel mode no differences were noticeable compared to the case using no authentication. This was probably the result of the measurements took place on a computer participating in the tunnel.

### 5.7.6 Sources of Error

For the time values there are different aspects to keep in mind. The results of the *time* command are presented in milliseconds, just like the test results presented in 5.4. This might bring about that the rounded values can differ more or less from the actual value which might be of higher accuracy. To avoid this source of error have been presented mean values of several measurements and there has also been a note to the reader to keep a reasonable fault margin.

Another possible source of error in time measurements is other processes working in the background slowing down the retrieving of the file. To limit this source of error, several measures have been made to ensure that the values are kept within a supposed interval. The risk of other disturbing activity over the network can be excluded as the package analyse displayed all the traffic at the network.

The risk for errors in the tests when counting the packages is to be considered as low. The package analyse has been made with a widely used tool recommended by people working with security.

## 5.8 Summary

The results from the package analyse has been helpful when analysing the time results presented. It was clear that the TLS implementation did not support rehandshake, which is why the presented values are considered as being from the initial full handshakes.

Comparing the values from the initial negotiation shows TLS to be a much more time consuming alternative even though TLS sends fewer packages, only six compared to the ten in IPsec. IPsec has the quickest first connection.

When reconnecting using IPsec no difference is noticeable compared to the connection with no authentication. When negotiating in a rehandshake TLS uses four packages instead of the six in a full handshake, and is likely to be less time consuming. But it is difficult to see how TLS could possibly match the presented values of IPsec even with two packages less. This can of course not be backed up with actual data since the implementation used does not support rehandshake. The values presented from the initial contact indicate each TLS package to be time consuming. This makes it likely to conclude that rehandshake will provide larger time values than in the reconnecting case using IPsec. IPsec has also the quickest reconnection.

The results of the tests presented in this chapter are to be considered as reliable because of several reasons. There has been an extensive pilot study of both IPsec and TLS before structuring the tests. The tests have been made carefully and have been properly documented. The tests presented in this report have been made in a closed environment to limit the sources of errors. The equipment used in the tests has not shown any indications of not being in working order. Repeated measurements of the same connection have shown values within a limited interval. Support from the reference literature has been found for all test results except for in the case with tunnel mode in IPsec.

### 5.8.1 Future work

For further practical investigations in the area it is recommended to use **another test environment**. It has been my experience that there is none or very deficient support for making this kind of tests using Windows products.

*"In theory there is no difference between theory and practice – in practice there is"*
Yogi Berra

# 6 CONCLUSION

The question this report aim to answer is how to authenticate the content provider taking in account three conditions; security level, performance and revocation.

The question about **revocation** shall be considered as answered. It has been suggested to use an *Access Control List* (ACL). Using an ACL answer to the request to terminate cooperation on short notice without having to await the expiration of certificate or similar time bound shared trust. Access control list can be used in both IPsec and TLS.

Regarding the initial conditions for **performance,** TLS does not seem to be a good alternative. The presented results show there is much overhead in time compared to the IPsec alternatives. The test results presented for IPsec does not show significant overhead and therefore IPsec shall be considered to be the most suitable choice looking at performance.

Looking at **security,** TLS has not shown any reason during tests for not recommending it. The same is true for IPsec transport mode which has not shown any negative aspects in security matters. In line of the tests IPsec tunnel mode is not recommendable for use in cooperation with an external partner due to the fact that it was possible to make connection using mismatching keys.

Summarizing these facts leads to a recommendation to use IPsec transport mode, it is a quick and secure alternative, but such a connection can only be made from host to host, see chapter 3.3. This fact does not answer to the condition set in chapter 1, that the connection between the operator and the content provider will be over a web-based channel.

The tested alternatives did not in any case answer to all the listed requirements which is why the conclusion made from this work is that **none of the tested alternatives are a good solution for the problem presented.**

## 6.1 Future Work

No satisfying solutions have been found but the material presented should be enough to be able to disregard from some solutions in further investigations.

Ericsson has been looking at several alternatives of who should have the responsibility for authentication in a presumed cooperation. One alternative was to let the content provider handle the authentication towards the charging system and that has been the condition for this work.

Another alternative discussed was that the end user makes the contact with the charging system in order to pay the content provider. When using this approach the authentication can be handled in an already well-tried and functional way. This alternative is the most likely to be used, which is why it is hard to see if there can be any further work of use to Ericsson to follow upon this thesis

# 7 REFERENCES

[1] Kent, S; Atkinsson, R. (November 1998). *Security Architecture for the Internet Protocol*. RFC2401

[2] Kent, S; Atkinsson, R. (November 1998).*IP Authentication Header (AH)*. RFC2402.

[3] Kent, S; Atkinsson, R. (November 1998).*IP Encapsulating Security Payload (ESP)*. RFC2406.

[4] Harkins, D; Carrel, D. (November 1998). *The Internet Key Exchange (IKE)*.RFC2409.

[5] Various writers (fall 2002). Course material from DVD006 *Risk analysis and security* https://idenet.bth.se/

[6] Dierks, T, Allen, C. (January 1999). *The TLS Protocol Version 1.0*. RFC2246

[7] Anderson, Ross, *Security engineering – A guide to building dependable distributed systems*, New York: John Wiley & Sons, Inc (2001), ISBN 0-471-38922-6.

[8] Eastlake, D, Crocker, S, Schiller J. (December 1994). *Randomness Recommendations for Security*. RFC1750

[9] Rubin, Aviel D. *White-Hat Security Arsenal – Tackling the Threats*, USA: Addison-Wesley (2001), ISBN 0-201-71114-1.

[10] Henriksson, Samuel. 2001. *Utvärdering av IPSec och SSL*. ISSN 1402-1617 / ISRN LTU-EX—01/037—SE. http://epubl.luth.se/1402-1617/2001/037/index-en.html

[11] Levin, Christian. (2002). *Security Evaluation of Ericssons PrePaid system*.MEE0116

[12] Rescorla, Eric. *SSL and TLS: Designing and Building Secure Systems*. USA: Addisson-Wesley (2000), ISBN 0-201-61598-3

[13] Schnier, Bruce. *Secrets & Lies – Digital Security in a Networked World*, New York: John Wiley & Sons, Inc (2001), ISBN 0-471-25311-1

[14] VeriSign (2000). *Digital ID℠ Introduction* http://www.verisign.com/support/tlc/per/whitepaper.htm

[15] Diameter webpage, http //www.diameter.org/

[16] http://www.tml.hut.fi/Opinnot/T-110.402/2002/Luennot/titu20021016.pdf

[17] Parlay webpage, http://www.parlay.org/specs/index.asp