Evaluation of VPNS

MUSTAFA AL-ABAYCHI

JOANNA ELLVIN
Abstract
This thesis evaluated the performance of four different virtual private networks (VPNs): IP security (IPsec), OpenVPN, SSH port forwarding and SSH using virtual interfaces. To evaluate these VPNs, three comparative performance tests were carried out in which the maximum throughput of each VPN was measured. In every test, a specific parameter was varied to observe how it affected the VPNs throughput. The parameters varied were the type of transport layer protocol used, the encryption algorithm used and whether the VPN used compression or not.

The results showed, among others, that when TCP traffic was transferred through the VPN and AES-128 was used as encryption algorithm in a Gigabit Ethernet network, the throughput for SSH port forwarding was 168 Mbit/s, 165 Mbit/s for IPsec, 95.0 Mbit/s for SSH using virtual interfaces and 83.3 Mbit/s for OpenVPN. These results are to be compared to the throughput measured when no VPN was used, 940 Mbit/s.

Three conclusions are drawn from the results of the performance tests. The first conclusion is that the throughput of a VPN depends on the technology the VPN solution is based on, the encryption method that is used and the type of data that is sent over the VPN. The second conclusion is that IPsec and SSH port forwarding are the most effective VPNs of the ones compared in this thesis, while OpenVPN and SSH using virtual interfaces are less effective. Lastly, it is concluded that although the different parameters affected the throughput of each VPN, the relation between the VPNs is the same in almost every test. In other words a VPN that performs well in one test performs well in every test.

Keywords: Virtual Private Network (VPN), IP Security (IPsec), Evaluation, Performance, Secure Shell (SSH), OpenVPN, Performance evaluation, Performance tests
Abstract


Utifrån resultaten av prestandatesterna dras tre slutsatser. Först fastställs att en VPNs genomströmmning beror på tekniken den bygger på, krypteringsalgoritmen den använder och vilken typ av trafik som transporteras genom den. Den andra slutsatsen som görs är att de två effektivaste VPNen av de som jämförts i detta arbete är IP-säkerhet och SSH portvidarebefordring, medan OpenVPN och SSH med virtuella nätverksgränssnitt är mindre effektiva. Till sist fastslås att även om parametrarna påverkar genomströmmningen av varje VPN så är relationen mellan VPNen densamma i nästan varje test. Med andra ord så har en VPN, som har hög prestanda jämfört med de andra i ett test, även hög prestanda i de andra testerna.

Nyckelord: Virtuellt privat nätverk, IP-säkerhet, Utvärdering, Prestanda, Genomströmmning, SSH, OpenVPN, Utvärdering av prestanda, Prestandatester
Table of Contents

Abstract............................................................................................................. i

Abstract........................................................................................................... ii

Table of Contents............................................................................................ iii

Table of Figures................................................................................................. v

1. Introduction ................................................................................................. 1
  1.1 Definition of a VPN.................................................................................. 1
    1.1.1 Site-to-site VPNs............................................................................. 2
    1.1.2 Remote Access VPNs....................................................................... 2
  1.2 Problem ................................................................................................. 3
  1.3 Objective .............................................................................................. 3
  1.4 Benefits and Ethical Considerations....................................................... 3
  1.5 Method ................................................................................................. 4
  1.6 Limitations............................................................................................. 5
  1.7 Outline ................................................................................................... 5

2 Background .................................................................................................... 7
  2.1 Considerations in designing VPNs........................................................ 7
    2.1.1 Tunnelling ...................................................................................... 7
    2.1.2 Authentication and Cryptography Algorithms............................... 8
  2.2 Existing Solutions .................................................................................. 9
    2.2.1 IPSec ............................................................................................ 9
    2.2.2 OpenVPN .................................................................................... 13
    2.2.3 SSH and its VPNs ....................................................................... 14
    2.2.4 GRE ........................................................................................... 17
    2.2.5 PPTP .......................................................................................... 18
    2.2.6 L2TP ........................................................................................... 19

3 Methods ......................................................................................................... 21
  3.1 Identifying the Problem ....................................................................... 21
  3.2 Choices of solutions to be evaluated..................................................... 22
  3.3 Research Methods ............................................................................... 22
  3.4 iPerf: Throughput Measurement Tool ............................................... 22
  3.5 Experiment Environment and Parameters........................................... 23
    3.5.1 Test Parameters............................................................................ 24

4 Results ......................................................................................................... 27
  4.1 Reference Measurement ..................................................................... 27
  4.2 Transferring data over TCP/UDP ......................................................... 27
  4.3 Different encryption algorithms ......................................................... 28
  4.4 Activating/Deactivating Compression ................................................ 29

5 Analysis ....................................................................................................... 31
  5.1 The reference measurement ................................................................ 31
  5.2 Environment with TCP versus UDP .................................................... 31
  5.3 Environment with different encryption algorithms............................ 31
  5.4 Environment with and without Compression ..................................... 31
  5.5 General Analysis of the Used VPNs ................................................... 32

6 Conclusions ............................................................................................... 35
  6.1 Discussion ........................................................................................... 35
  6.2 Future work ......................................................................................... 35
References ................................................................. 37
Appendix ................................................................. 40
Table of Figures

Figure 1. Site-to-site VPN scenario. .............................................................................................................. 2
Figure 2. Remote access VPN scenario........................................................................................................ 3
Figure 3. Illustration of a tunnelled packet. .................................................................................................. 7
Figure 7. IPSec ESP header, tunnelling mode [2]. ....................................................................................... 11
Figure 8. OpenVPN packet header. ............................................................................................................. 14
Figure 10. IP packet after leaving the source SSH VPN gateway.............................................................. 17
Figure 11. PPTP encapsulation levels [32]. ................................................................................................. 18
Figure 12. iPerf report when sending UDP traffic ...................................................................................... 23
Figure 13. iPerf report when sending TCP traffic......................................................................................... 23
Figure 14. The test bed used in every experiment ..................................................................................... 24
Figure 15. Packet loss (%) at different sending rates. .................................................................................. 25
Figure 16. Throughput obtained by transferring data with no encapsulation or encryption, IP, with only encapsulation, IPIP, and with both encapsulation and encryption, the VPNs........................................ 27
Figure 17. UDP throughput different VPNs.................................................................................................. 28
Figure 18. The throughput obtained by sending data over TCP in different VPNs........................................ 28
Figure 19. Throughput of the defined VPNs using different encryption algorithms.................................. 29
Figure 21. Cost of compression function when it runs on a compressed file.............................................. 30
Figure 22. The process of tunnelling a packet in an OpenVPN gateway.................................................... 32
Figure 23. The tunnelling process of a packet in an IPSec gateway............................................................ 33
Figure 24. Processing of a packet sent from the client to the resource in SSH.......................................... 34
1. Introduction

As many other aspects, the aspect of exchanging information was affected by the emergence of the Internet. Different groups of users of the Internet started to exploit this technology. Some groups, for example commercial and political organizations, prefer to have private networks instead of using the public Internet to guarantee the security of their information [3]. But using these private networks makes it difficult to exchange information when having offices in different geographical locations. To connect different locations with each other without depending on the Internet naturally comes with high expenses and a decreased level of security. Organizations have to install their own networks along the path, which explains the reason for the high costs, and they have no sustainable infrastructure like the one they would have if they used the insecure Internet. For organizations that have employees that travel to different countries, it is impossible to establish a secure connection with these employees using such private networks.

To solve the problem of sustainability, decrease the expenses and make it possible to connect with employees at any geographical location, the technology of virtual private networks, VPNs, was developed. This technology uses the Internet as an underlying infrastructure, but in a secure and confidential way [3, 4].

1.1 Definition of a VPN

The definition of a VPN is a network that is accessible via the public network by some specific users but not by others [5]. Through time, VPNs have been used in different ways [6]. The first way is called trusted VPNs. These trusted VPNs use no encryption and their traffic takes the same path each time. This path is decided and maintained by a company, for example an ISP. The client trusts these companies to keep the client’s packets unreachable by any attacker. The client must trust the company to keep their traffic secret from any third party as well as rely on the company not having any untrustworthy employees tampering with their traffic in an unwanted way. These trusted VPNs are therefore risky.

The other method is secure VPNs. It is based on encryption, which ensures integrity and confidentiality of all traffic. Traffic flows may take different paths but on the other hand they are secure. Besides encryption, a main concept of VPNs is putting the data that is going to be sent over the Internet in an IP packet, and then making this IP packet a payload of another IP packet. This process is called encapsulation. The new packet is addressed to an entry node, a gateway, which can extract the encapsulated packet, and forward it to the destination inside the private network. The extracting of the packet is called the decapsulation. The IP packet will be encrypted when entering the public network and decrypted when leaving the public network.

Virtual private networks are useful in many circumstances. In spite of having a common goal, that is to gain access to an internal network remotely and
securely [7], VPNs can be categorized into two main usage scenarios. These scenarios are site-to-site VPNs and remote access. Below is a definition of these scenarios.

1.1.1 Site-to-site VPNs

In site-to-site VPNs, shown in Figure 1, there are two sites in two or more geographical locations [7]. Clients in each site want to access resources in the other site. Commonly this type of VPNs is used by companies, which want to connect their offices distributed at different locations to work as an intranet. To make it possible, each site is provided with one or more gateways. These gateways are responsible for the tunnelling and authentication on both sides of the tunnel. The clients of the network in the two sites are unaware of the tunnel. Only the gateways are aware of it. Figure 1 is an illustration of the site-to-site VPN scenario.

Gateways in site-to-site VPNs have to be configured to fulfil the security policies of VPNs. They also have to be able to identify the other end of the tunnel and even the subnets behind the tunnel. Furthermore, gateways that form two ends of a tunnel have to be consistent so that they are able to establish a connection and let the traffic flow [7]. That can for instance mean the receiving router can decrypt what the sending router has encrypted.

Figure 1. Site-to-site VPN scenario.

1.1.2 Remote Access VPNs

In remote access VPNs, there is one gateway to the server and the clients are many, spread on many different geographical sites, possibly in different countries or continents. An example of a remote access VPN is shown in Figure 2 below. The biggest advantage in this scenario, compared with the previous one, is that clients will be able to access the server wherever they are. That makes it for example possible for an employee to dial a telephone number of the company he works in via the local network even when not present at the company. The conversation is then secure and there is no need to pay for it as any other call. An additional advantage with remote access VPNs is that they do not require unit conversion when being in a country that uses different units in their communication infrastructure than the units used in the country where the private network is located [7].

There are some considerations when implementing Remote Access VPNs. Having many clients deployed in different networks makes it more likely to be attacked. Therefore security is an important issue, both at the client and the
gateway side, where the authentication has to be verified. Different clients also have different access to the resources in the intranet.

![Remote access VPN scenario.](image)

Figure 2. Remote access VPN scenario.

Some studies have previously been done to measure the performance of VPNs. Two of these studies are *Performance Comparison of IPsec and TLS based VPNs* [8] and *Network Performance Analysis of VPN Protocols: An Empirical Comparison on Different Operating Systems* [9]. In [9], the authors focused on evaluating different private networks in different operating systems. In this thesis project different VPN solutions will be evaluated in defined scenarios.

### 1.2 Problem

The VPN solutions IPsec, SSH port forwarding and the two VPNs that use virtual interface, OpenVPN and SSH VPN operate differently from each other. The underlying technology for each of them may lead to a higher performance in one and a lower performance in another.

The main problem in this thesis project is stated as: which of the VPN solutions IPsec, OpenVPN, SSH port forwarding and SSH using virtual interfaces gives higher throughput when transferring data using fixed parameters? Sets of fixed parameters that rule the tests are presented in Chapter 4 of this report. The problem does not stop at that point. Further investigation about what underlying technology determines the obtained results, and why that technology leads the results in the direction they were led will also be considered.

### 1.3 Objective

The main goal is to investigate and evaluate technologies and protocols for secure VPNs and find the technology that is most efficient for a specific scenario. The result of the project is expected to be of general interest since only little previous work has been done on the evaluation of different VPN solutions.

### 1.4 Benefits and Ethical Considerations

The thesis project is supposed to be of general interest, due to the wide use of VPNs. The results may be used as a reference when choosing which VPN to implement.
Two points are defined to be related to environmental benefits in this thesis project. Firstly, since VPNs provide a secure channel for transmitting data, they also help to avoid travels that only aim to bring the data physically to ensure the security. Reducing travels means reducing CO2 emissions. Making the transmission more convenient by defining the suitable technology to use may mean that this technology will be used more often and thus less travels are made.

Secondly, using a VPN with high throughput lets the network components finish their job earlier. That is related to current issue of green networking [10, 11]. By green networking, we mean letting the network components be passive when they are not in use. Using a high throughput technology leads to a faster transfer, letting the network components go back to the passive mode and thus much energy is saved.

Some benefits of the work are even related to ethical issues. This kind of work can help people who live in countries where they cannot express their political or religious opinions. Evaluating the throughput of this technology may improve their ability to reach and transmit the information and make it more convenient.

Another ethical consideration is to keep information of this kind of work away from being accessed by e.g. terrorist organizations, which may use the evaluation of VPNs to find a good way to avoid being reached by governments that fight them.

1.5 Method

In this section, a summary of the approach that was taken to perform this thesis project is presented. More details about how this thesis project was carried out are presented in section 3 of this paper.

The thesis project was initiated by identifying the problems that are related to the subject. Problems concerning subjects like security, round-trip delay, overheads and throughput were defined at this stage. When defining the problems, it was taken into consideration that they are related to VPNs that are commonly used today.

After presenting the list of the potential problems to be investigated, we studied them to identify the one that was more prioritized than the others and thus should be further investigated in this project. Whether a problem is a subject of many other previous studies or not was taken into consideration when identifying the prioritized problem. Another aspect that ruled the chosen prioritized problem is the scope of the problem, whether it covers more or less properties of a VPN.

The next part was to specify the research method that was used in this thesis project and the appropriate tool with which the tests were performed. This choice was made after studying the existing scientific research methods and matching the one that fits for the problem to be investigated in this thesis project.
Finally we designed the environment in which the tests were carried out, and defined the interesting parameters that ruled the tests to make it possible to answer our identified problem. The measurements were carried out in a laboratory where the involved machines were in a closed environment. That was to assure that no external factors could affect the results. The machines were connected to each other using Ethernet. Two of the machines were the gateways, corresponding to the edges of the tunnel. The other two were the hosts, one host behind each gateway.

To measure the throughput a program called iPerf was used. This program helped us also in generating two types of traffic, namely UDP and TCP. Besides measuring the throughput of these traffic types, throughput was measured for two types of data, compressed data and uncompressed data. The VPNs were once configured to compress the traffic in the tunnel and once not to compress it. That aimed to find out the effect of compressing already compressed on the throughput and the potential gain obtained by compressing regular, uncompressed data. Another configuration parameter that was modified to several values was the encryption algorithm. We measured the throughput of traffic sent over the VPNs using different encryption algorithm.

1.6 Limitations
The thesis project deals with experimental measurements of throughput. The experiments were limited to be done over Ethernet only. That is because Ethernet is a stable environment. Conclusions are both clearer and easier to draw when using Ethernet compared to Wi-Fi.

Gateway platforms were limited to the operating system Linux. Linux is an open source operating system. It is mostly therefore Linux was chosen to be the operating system of the gateways.

The experiments were limited to be done in a laboratory environment. In real life new parameters may appear. These parameters are difficult to control and are sometimes difficult to define. The thesis project is therefore limited to a laboratory environment.

1.7 Outline
This section is a presentation of how the rest of this report is organized.

Chapter 2 aims to define the usage scenarios in which VPNs are useful and presents different solution to be used in these scenarios. The chapter is initialized with some consideration in designing VPNs and presenting the basic components of them. The chapter ends with selecting VPN solutions to evaluate.

In Chapter 3, the methods used in this thesis project are presented. That covers how we defined the problem to answer in this project, the tools used and the experimental environment where the tests were performed.

The results of the tests are presented and analysed in Chapter 4 and Chapter 5.
respectively.

Chapter 6 includes the conclusions drawn of this thesis project and suggestions of future works.
2 Background
In order to present the existing VPN technologies and how these technologies can be exploited, this section is divided into three parts. The first part will describe factors to be considered when constructing a VPN. The second part describes the conceivable usage scenarios of VPNs. Lastly, a section that discusses different solutions of VPNs is presented.

2.1 Considerations in designing VPNs
There are different types of VPNs, as presented in section 2.2 in this thesis report. Although these are used for different purposes, they share three common components essential for a VPN: tunnelling, authentication and data integrity and confidentiality. These three aspects all need to be considered in every implementation of a VPN, despite what type it belongs to. It is here the difference between VPN solutions exists. Therefore it is important to understand these concepts to further be able to differentiate between VPN solutions. To fully understand these important concepts, the following section presents them in detail.

2.1.1 Tunnelling
Tunnelling is one of the foundations of the VPN technology. The concept of a tunnel is what makes VPNs possible. Encapsulating a packet as payload to an external packet is the essence of creating tunnels. The new packet is transported over a medium, commonly the Internet, to a destination where decapsulation occurs and the original packet is revealed. An illustration of that is shown in Figure 3. The original IP address in the figure is found in the inner IP-header in the figure. The outer IP-header includes the IP address of the gateway that decapsulate the packet. During the transport, the tunnelled packet is perceived as a normal payload of the protocol in which it is encapsulated [12]. There are two main reasons that make tunnelling an important concept to VPNs. The first reason is its provision of a means to hide the private source and destination IP addresses used by the hosts of the tunnel endpoints [12]. It is necessary to hide these IP addresses from routers in the Internet, since they do not know how to route private addresses. The second reason tunnelling is important is because of its ability to provide security to an otherwise insecure protocol, usually by encryption mechanisms.

Tunnelling can occur at any layer of the OSI model, but in VPNs it is most common to occur in layer 2 or layer 3. Some examples of layer 2 tunnelling protocols are PPTP, L2TP and L2F. An example of a layer 3 tunnelling protocols is IPsec [7].

| Outer IP-header | Inner IP-header | Payload |

Figure 3. Illustration of a tunnelled packet.
2.1.2 Authentication and Cryptography Algorithms

Since VPNs are established over the Internet, which is an unreliable medium, it is essential for VPNs to provide secure transfer of data. Otherwise, ineligible users could read, control and affect the data transported through them. Since every packet in the Internet is routed independently, data security must be provided to each individual packet [7]. When it comes to data security, there are two things of concern. The first is, keeping the data unreadable from a third party, i.e. confidentiality, and the second is assuring that the received packet is unmodified by a third party, i.e. integrity [4, 7].

Authentication algorithms

Authentication algorithms are such algorithms that make it possible to verify that the alleged source of data is the one with whom we really are communicating. Authentication also covers that the packet has not been modified by a malicious part, so that the data sent is identical to the data received [13].

Authentication is provided by hash algorithms. A hash algorithm is a function that takes a message with a non-fixed length as parameter and returns a message with a fixed length [4]. Each message has a unique output from the hash function. This output is called a hash value or a fingerprint.

When one side of a tunnel sends a message to the other side, the fingerprint is attached to the message. The receiving side calculates the fingerprint using the message as its input. If the generated fingerprint is identical with the received one, the message is accepted, otherwise, it is dropped. But since there are several common algorithms to generate these fingerprints, such as the Secure Hash Algorithm (SHA) and Message Digest 5 (MD-5), an attacker can easily success in having the corrupted message accepted. This is done by changing both the message and the corresponding fingerprint after revealing what hash algorithm was used to generate that specific fingerprint. This causes a problem for using hash algorithms in their purest form [4].

To solve that problem, message authentication codes (MACs) are used. A MAC is a shared key, that is a key that both the sender and the receiver possess, that is attached to each hash function input. Accordingly, when the receiving side gets the message, it attaches the MAC to it and generates a fingerprint. Since it is a characteristic of hash algorithms that it is impossible to conclude the original message using the fingerprint, it is impossible for the attacker to calculate the MAC. Although MACs provide a strong authentication mechanism, there exists an even stronger mechanism. This mechanism is called HMAC algorithm, which stands for Hash Message Authentication Code [4].

Encryption algorithms

Cryptography is the science of developing algorithms that convert readable data, into incomprehensive data to the public, called encrypted data or ciphers. That must be followed by the ability to convert the encrypted data back into the original clear data. The processes are called encryption and
decryption respectively. Decryption will only be feasible by those who are expected to have the accessibility to read the data [7].

There are two types of cryptographic algorithms, symmetric and asymmetric encryption algorithms. In the first type, symmetric encryption, a single key is shared between the two sides of the tunnel. Besides sharing the same key, the symmetric encryption algorithms have the following characteristics [4]:

- They result in compact ciphers.
- They are fast.
- The issue of key management and distribution is complex.

The second type, asymmetric encryption algorithms that is also known as public-key algorithms, is used both for acquiring encryption and authentication. The concept of asymmetric algorithms is based on having a private key that is kept secret by a single user, and a public key, which can be used by anyone. Messages that are encrypted by the public key can only be decrypted by the private key and vice versa. Besides having two different keys, the asymmetric algorithms have the following characteristics [4]:

- They are slow.
- Their ciphers are not compact.
- There is no complexity in the distribution and management of the keys like the complexity that exists in symmetric encryption algorithms, thanks to the public keys.

An example application in using asymmetric encryption algorithms to acquire authentication is the Digital Signature. In this mechanism, a public hash function is used to generate a signature that is unique to the sent message. RSA is, on the other hand, an example where an asymmetric algorithm is used to encrypt plain text into a cipher text using a public key [4].

Both symmetric and asymmetric algorithms may be used in building VPNs, but symmetric encryption algorithms are more common because of their speed compared with asymmetric algorithms. A widely used algorithm in implementing VPNs is the Advanced Encryption Standard AES [4, 7].

2.2 Existing Solutions
In this part we highlight the technology behind each of the following solutions and what specifications they have.

2.2.1 IPsec
This section presents the details of IPsec. Firstly, the security protocols Authentication Header, AH, and Encapsulating Security Payload, ESP, are presented. Then the concept of security associations is presented, followed by IPsec databases SA and key management techniques. Then authentication of IPsec hosts is described and lastly an example of IPsec packet processing is given.

Security Protocols AH and ESP
The two security protocols used by IPsec are AH and ESP. Both of the protocols protect packets against replay attacks [14]. They also provide data integrity and secure authentication by means later described, but only ESP provides confidentiality. Since confidentiality is highly desired, IPsec VPNs are commonly implemented with the ESP protocol [4].

AH provides data integrity and data origin authentication. It is able to operate in two different modes called transport mode and tunnel mode [4]. The two modes differ in the way the data is encapsulated. In tunnel mode, a new IP header and the AH header are added to the packet, while in transport mode the AH header is added in between the IP header and the next protocol header. Authentication of IPsec packets in AH tunnel mode is performed on the whole packet except on the fields in the new IP packet header which are mutable, such as the Time-to-Live (TTL) field. In transport mode, the authentication is performed on the same values as in tunnel mode, except that there exists no new IP header and therefore it occurs on the original IP fields. The following figures, Figure 4 and Figure 5, are simplification of how a packet that uses AH protocol in transport respectively tunnelling mode looks like. The lightly shadowed areas are authenticated and the packet is dropped if an attacker modifies them.

![Figure 4. IPsec AH packet in transport mode](image)

ESP provides all characteristics that AH provides and the additional property of data confidentiality. Therefore ESP is more common than AH. Including the ESP header, the ESP trailer and the ESP AUTH to the packet provides data confidentiality and data authentication. Figure 6 shows the transport mode variant of an ESP packet. The lightly shadowed parts denote authenticated parts of the packet. The darkly shadowed parts denote the encrypted and authenticated parts of the packet.

![Figure 6. IPsec ESP header, transport mode](image)
In the tunnelling mode, the header of ESP includes a new IP header where the public IP address is placed, then the ESP header and the user packet, that is the original IP header, then the other protocol headers that are used in the private network and the payload. At the end an ESP trailer is placed and an ESP ICV, that is used for authentication [4]. Figure 7 illustrates an ESP packet in the tunnelling mode.

Figure 7. IPsec ESP header, tunnelling mode [2].

The encrypted parts of the ESP packet are, as shown in Figure 7 the user packet and the ESP trailer. To encrypt these parts of the ESP packet, symmetric encryption algorithms are often used. Authentication of the ESP packet is performed on the ESP user packet, the ESP trailer and the ESP header. Methods used for authentication are commonly the two HMAC algorithms SHA-HMAC-96 and MD5-HMAC-96 [4].

Security Associations
Another important concept of IPsec is that of Security Associations (SAs). Each SA is identified by an SPI and defines how IPsec shall protect traffic flow in one direction of the VPN tunnel. SAs contain information about the SA lifetime, cryptographic algorithm, security protocol and security protocol mode. To protect traffic flow in both directions of the VPN tunnel, at least two SAs are needed, one in each direction. It is also possible for a traffic flow to be protected by the use of more than two SAs as one of the protocols ESP or AH may be selected for authentication and the other for encryption.

IPsec Databases
The fourth element of IPsec is the three databases called the Security Policy Database (SPD), the Security Association Database (SAD or SADB) and the Peer Authorization Database (PAD). Together they ensure that the process of IPsec traffic runs correctly. The SPD specifies the inbound and outbound traffic that should be protected or bypassed by IPsec. The SAD contains an entry for each SA and information related to it. The third database, the PAD, specifies how a host shall be authenticated and which hosts the IPsec gateway is allowed to negotiate SAs with [4].

SA and Key Management Techniques
To establish and manage SAs, IPsec provides two different methods. The first one is Manual SA and Key Management and the second one is Automated SA and Key Management. These are provided through the IKE protocol. Although IPsec supports these two different methods, only Automated SA and Key Management through the IKE protocol is used in practice, since the manual configuration method does not scale. Because of this, the automated management method will be the only method described here.
As mentioned above, the automated SA and key management method make use of the IKE protocol. This protocol exists in two versions: IKE Version 1 (IKEv1) and IKE Version 2 (IKEv2). Because IKEv2 is the most recent, secure and most efficient one, it is the protocol which will be the focus of the following description of Automated SA and Key Management, although much of it applies to IKEv1 too [4].

To establish a SAs between two IPsec hosts, IKEv2 defines a negotiation between the IPsec hosts in two phases [15]. Phase 1 consists of two messages called request messages because they are sent by the initiator of the VPN, and two response messages sent by the IPsec host to which the initiator wants to establish the VPN. With these messages, the IPsec peers negotiate an IKE SA, which defines the way the two IPsec hosts communicate and an IPsec SA and decides upon a session key [15]. They also authenticate each other and establish the cryptographic algorithms to be used. The purpose of phase 2 is to use IKEv2 request and response messages to negotiate more IPsec SAs if needed [4].

**Authentication of IPsec hosts**

The authentication between the two IPsec hosts in IKEv2 negotiations may be performed by using one of three techniques: pre-shared keys, encrypted nonce or digital signatures. Authentication using pre-shared keys implies that both IPsec hosts use an identical key configured on them for authenticating the other host. When using an encrypted nonce as authentication method, both IPsec hosts need to be in possession of the other host's public key before the IKE negotiation. Then an IPsec host may authenticate the other host by using that host's public key to encrypt a nonce, which then only can be decrypted by that host's private key [4]. The third alternative, digital signatures, is the most secure authentication method IPsec offers, but it is also the most complex one [15].

To use authentication with digital signatures, both IPsec hosts need to possess the other host's public key before the IKE negotiation starts. Also, the hosts need to be sure that the public key received is in fact the one of the other host and not one of an attacker. This is ensured by the use of signed digital certificates, which are associations between a host's public key and information about the hosts identity [4, 16]. A digital certificate is created for both IPsec hosts by a Certificate Authority (CA), which is a third party trusted by both hosts. These certificates are signed by the CA, and then exchanged between the two IPsec hosts.

The hosts can in this way be sure that the public key received in the certificate is indeed the one of the other IPsec host. The actual authentication is then performed by an IPsec host by encrypting, i.e. signing hashed data with its private key to create a digital signature. The host then sends this digital signature to the other IPsec host. The receiving host then verifies the identity of the sender by decrypting the digital signature with the sender's public key. If the decryption is successful, the receiving host can be sure of the identity of the sender.
**IPsec Packet Processing**

So far, this section has described the technologies behind IPsec, now it will conclude with an example of IPsec processing between two IPsec gateways. When a packet from the inside local network, destined for the outside, arrives at an IPsec gateway, the gateway needs to decide whether to protect the packet with IPsec, discard the packet or forward it without IPsec protection. The gateways make this decision by checking the SPD, which as previously mentioned includes information about which traffic to protect with IPsec or not. If the SPD states that the packet needs IPsec protection, the gateway checks for an IPsec SA in the SDA. Depending on whether a corresponding SA exists, one of two actions will be taken by the IPsec gateway. If one or more SA exists, the packet will be encapsulated according to the information in the SA or SAs and then forwarded to the other IPsec gateway. If no SA exists and automated SA and key management is used, an IKE negotiation is started with the other IPsec peer.

After a successful negotiation, at least one IPsec SA exists in the SPD. Now, the packet will be encapsulated by ESP or AH, depending on what the information in the corresponding SA states. If there is more than one corresponding SA to the packet, the packet is encapsulated multiple times according to the specifications in the SAs, until all SAs have been used. When the encapsulation process is finished, the IPsec protected packet is forwarded on the outside interface.

When a packet protected by IPsec arrives at the outside interface of an IPsec gateway, the ESP or AH headers are removed. This is done by a check of the IPsec packet header for the SPI to find the SAD and then reference the correct SA in it. In case the packet was protected by multiple AH and or ESP headers, this process is repeated until the IPsec protection is removed. The packet is then forwarded, unprotected, out of the inside interface to the local network.

**2.2.2 OpenVPN**

OpenVPN is a software based VPN solution. It listens to the network devices TUN/TAP, the fundamental technologies that lie behind OpenVPN. TUN/TAP are virtual interfaces that are accessible by all applications and users, which makes OpenVPN simple to implement [17]. The difference between the interface TUN and the interface TAP can be summarized as, when using the interface TUN the connection appears as if it was point-to-point connected with the other end of the tunnel, whereas the use of the TAP interface results in a virtual hardware bridge [17].

The open source project of developing TUN/TAP aimed to enable tunnelling of IP traffic on Linux platforms. Today, this software is compatible with Linux, Windows, Mac OS X and other operating systems, making OpenVPN convenient to use in these widely used platforms. Since the interfaces of TUN/TAP act in layer 2 and layer 3 in the OSI model, OpenVPN acts in both these layers. Acting in these layers gives OpenVPN the property of encapsulating packets such as Ethernet Frames and IPX packets. This is an
uncommon property in other VPN solutions [17].

An OpenVPN packet is shown in Figure 8. As stated before, the link layer is included in the packet and is encrypted and encapsulated in the packet. The shadowed part of the packet in Figure 8 is the encrypted part of the packet.

Besides being easy to configure, OpenVPN is flexible. The fact that OpenVPN runs in the user space makes OpenVPN simple. This simplicity gives OpenVPN more security [17]. OpenVPN is also described as well structured and that gives it additional simplicity. It is also compatible with all standardized encryption methods [17]. This compatibility is obtained, thanks to OpenVPNs dependency on the OpenSSL library [18].

Authentication in OpenVPN may be performed in two different modes. The first mode is called static key, and the second is Transport Layer Security, TLS. In the first mode, a pre-shared key is generated and used in the tunnel endpoints before establishing the tunnel. The generated key includes four independent keys. These are: send HMAC, receive HMAC, encrypt and decrypt [19]. HMAC is defined in section 2.1.2 of this report.

The second authentication mode, TLS, requires consequent authentication certificates on both sides of the tunnel [18].

The main drawbacks with OpenVPN are some security issues and that most of the other platforms except computers, do not support it [8, 17]. However, OpenVPN has an application for Android operating system, released in 2012. The security issues are due to the dependence on TLS, which methods of key exchange are weak and therefore may expose OpenVPN users to man-in-the-middle attacks [20].

2.2.3 SSH and its VPNs

This section provides an overview of the Secure Shell (SSH) protocol, followed by descriptions of the two SSH VPNs implemented in this work: SSH port forwarding and SSH using virtual interfaces (referred to as SSH VPN in the rest of this report).

The SSH protocol described in RFC 4251 [21], offers secure, encrypted communication channels between an SSH server and an SSH client. SSH was originally designed to replace insecure programs for remote login and file transfer such as telnet, rlogin and ftp [7, 22]. Although SSH commonly is used for remote login and file transfer, it can also be used in different ways to create a VPN.

There are two versions of SSH: SSH1 and SSH2. SSH2 is the latest version with better scalability and stronger security and integrity check of data compared to SSH1, which has major security vulnerabilities [7, 23]. For these
reasons, the description of SSH will only focus on the SSH2 protocol, although much of it accords to SSH1 too.

To establish an SSH tunnel, the client machine has to be authenticated by the SSH server. SSH uses asymmetric keys for this purpose [7]. The asymmetric keys need to be generated by both the SSH client and the server and their respective public keys have to be exchanged between them before authentication. As the client initiates an SSH session with the server, it generates a random number, the session key. Also it selects a symmetric encryption algorithm for providing confidentiality to the traffic that will be transferred between the client and server.

The next step is that the client encrypts the session key with its private key and the public key of the server before sending it to the server [7]. Upon receiving the encrypted session key, the server decrypts it using its private key and the public key of the client machine. If the decryption was successful, the client machines identity has been verified. After the authentication phase, both the client and server possess the session key. This key will be used by them, together with the symmetric encryption algorithm previously selected by the host, to encrypt the data they exchange.

When the client machine has been authenticated and the secure tunnel has been established during the authentication phase of the client, the user needs to be authenticated. SSH may perform this authentication in different ways, but the recommended, most secure way is Rivest-Shamir-Adleman (RSA) key-based authentication [24]. To use RSA authentication, the user must generate a public- and private RSA key pair and provide the SSH server with the generated public key. The server is then able to verify the identity of the user by encrypting messages with the users public key, which are only able to be decrypted by the users private key and therefore only readable by the user [24].

SSH does not only authenticate the SSH client user and machine to the server, as just described, but also the SSH server to the SSH client. This is done by the use of a cache stored in the client. The cache is used in such a way that every time a connection is made to an SSH server with an IP address never connected to before, the SSH client acknowledges the new connection and saves the public key of the server in the cache. Every subsequent connection, the client checks for the servers IP address in its cache and compares the public key received from it with the one stored. If the keys match, the identity of the server is verified [23].

SSH provides integrity to the data transferred in the tunnel by the use of MACs such as MD5 and SHA-2 [25]. The exact MAC algorithm used varies and is decided upon during establishment of the tunnel between the SSH client and server [26].

2.2.3.1 SSH Port Forwarding
The basic concept behind SSH port forwarding is to establish a secure tunnel between the local computer to the SSH server, which then forwards the traffic
to the computer that runs the service the local computer is requesting [23]. The traffic forwarded must run on top of TCP and hence have a specified port number. Services that conform to this requirement are for example HTTP, SMTP or FTP, which without the security provided by SSH, sends their traffic in clear text, visible to everyone on the public Internet [22].

Figure 9 illustrates a packet format and shows that the TCP connection is unsecure and that only the application layer data, the payload in the figure, is secured. The parts that are not shadowed are unsecured. The figure shows also that, in SSH, there is not an encapsulated IP header. That is because all traffic received on port x is forwarded to a single machine. When the packet is received by the gateway, the payload is extracted, encrypted and encapsulated in a new packet. It is then forwarded to the local resource machine that is a part of the local network, thus the payload is sent to that machine in clear text with the local gateway as the source IP address.

![Figure 9. IP packet after leaving the source SSH port forwarding gateway.](image)

The following is an example of the process of SSH port forwarding. To use SSH port forwarding for establishing a VPN, the client needs an SSH client software, such as Putty for Windows or OpenSSH for UNIX [7]. The client chooses a service to secure with VPN as mentioned earlier. HTTP will be used in this example. Then, instead of sending its HTTP traffic to the Internet, the local computer must be configured to send it through the SSH tunnel. The user on the local machine performs this configuration and specifies a port on it, the forwarding port. The user specifies as well the destination machine and a port on that machine.

The destination machine is the one that runs the desired resource, HTTP in this case, and the port is the dedicated port for that service, port 80 for HTTP [7]. The SSH client then starts a secure connection to the intermediate machine (the SSH server).

After the connection is established, the SSH client listens to traffic destined for the forwarded port on the local machine [7]. When the local machine receives data from an application on the forwarded port, it now has to send it through the SSH tunnel instead. This is done by the SSH application process, which puts the data in the payload field of the SSH packet header and sends it through the tunnel to the SSH intermediate computer [26]. The intermediate computer now checks the MAC of the SSH packet to verify its integrity [26]. Finally the intermediate computer forwards the SSH packet to the destination port, port 80, on the destination computer. In this way, the HTTP data has securely reached its destination through SSH port forwarding.
2.2.3.2 SSH VPN

SSH VPN operates in a similar way as OpenVPN in the sense of using virtual interfaces to communicate with the other end of the tunnel. In contrast to a VPN which uses SSH port forwarding, this type of VPN does not limit the traffic transferred over the VPN to only use the TCP protocol. Neither does it limit the users of the VPN to reach only the service running on a specific port.

Using the TUN interface means that the link layer header will be a requirement to be included in the packet, because the two gateways have a point-to-point connection. The header is therefore encapsulated and encrypted in the SSH packet. The packet can be illustrated as in Figure 10, where the MAC address representing the link layer header is included in the shadowed part of the figure. Shadowed parts denote the encrypted parts of the packet.

<table>
<thead>
<tr>
<th>IP header</th>
<th>TCP</th>
<th>MAC</th>
<th>IP header</th>
<th>TCP/UDP</th>
<th>Payload</th>
</tr>
</thead>
</table>

Figure 10. IP packet after leaving the source SSH VPN gateway.

Having a point-to-point connection between the source gateway and the destination gateway means that all the traffic that is in the source gateway pass through the virtual interface. Afterward, the encryption and tunnelling occur. At the virtual interface in the destination gateway, the decapsulated packet appears as a packet received from a neighbour host, that is, a host that is one hop from it.

2.2.4 GRE

Generic Routing Encapsulation, GRE, is a tunnelling protocol developed by Cisco. It has the characteristic that it can be carrier for, not only unicast IP packets, but also multicast and broadcast IP packets and non-IP packets. That is, GRE is capable to carry an "arbitrary network layer protocol over an arbitrary network layer protocol" [27, 28].

The authentication and confidentiality of GRE is obtained by relying on IPsec. As GRE depends on IPsec to gain security, IPsec may also be dependent on GRE, because IPsec cannot send multicast packets. Accordingly, the combination of these two protocols provides a means for a VPN connection to authenticate. It also makes the connection confidential and capable to carry additional types of packets.

The GRE protocol takes the network header, encapsulates it into the GRE header and both will be encapsulated into some other protocol to be carried through the network [29]. That means that GRE adds two additional headers, which means additional overhead of the packet size.

GRE tunnel endpoints are stateless. That means that they do not save any information about the state and the availability of the other endpoint. That results that if one tunnel endpoint is down, the other endpoint is not aware of that. Consequently, the active endpoint does not remove the interface that connects it to the disabled endpoint from the routing table [30]. The fact that
GRE is stateless, is considered a drawback, since the change in the network topology does not result in dynamic changes in the choice of the next-hop or the interface [30].

2.2.5 PPTP

Point-to-Point Tunnelling Protocol, PPTP, is a tunnelling protocol developed by Microsoft Cooperation and a number of other companies. The protocol is an extension of the link layer protocol PPP that acts in the data link layer [31]. The main idea of PPTP is encapsulating PPP packets into IP datagrams and sending them over the Internet [32]. The packets that are sent using PPP from a client to the gateway are tunnelled into IP datagrams so that they can be moved across an IP network [32].

The encapsulation process is deployed into several levels. The first level is encapsulation of the payload into a TCP/IP, IPX or NetBEUI packet. The second level is to encapsulate these packets into an IP packet addressed with the local IP address that is valid in the internal network. At the top of these packets, the PPP protocol establishes the connection between the local host and the remote host.

When arriving to the local gateway, all the previous protocol headers are encrypted and encapsulated into a GRE header. The GRE has an IP header on its top that is addressed to the remote gateway. The external packet is a PPP packet. This external packet is used to establish a connection between the two endpoints of the tunnel. Figure 11 shows the levels of this process. The shaded part of the figure shows the encrypted PPP packets that are created by PPTP [32]. At the destination side of the tunnel, the remote gateway extract the PPP packet, so that the protocol that is used for transferring the packet is switched back to a link layer protocol, that is, what now is decrypted PPP.

<table>
<thead>
<tr>
<th>PPP Delivery Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Header</td>
</tr>
<tr>
<td>GRE Header</td>
</tr>
<tr>
<td>PPP Header</td>
</tr>
<tr>
<td>IP Header</td>
</tr>
<tr>
<td>TCP Header</td>
</tr>
<tr>
<td>Data</td>
</tr>
</tbody>
</table>

Figure 11. PPTP encapsulation levels [32].

PPTP requires bi-directional connections in form of TCP/IP sessions to make it possible for each endpoint in the tunnel to update the information about the status and availability of the other endpoint. There are other several signalling messages that are exchanged thanks to the establishment of this TCP/IP session.
The authentication algorithm is not limited to a specific one when using PPTP. Instead PPTP sends a negotiation frame to negotiate what authentication algorithm to use [33]. However, since Microsoft is one of the developers of this tunnelling protocol, there are some authentication and encryption methods compatible in gateways that run Windows NT Server version 4.0 and Windows NT Workstation version 4.0 [32].

PPTP is described as vulnerable when it comes to security. In the website of Microsoft, an article published in 2013 states that PPTP is as insecure as a plaintext [34]. It is also blamed to have vulnerabilities and "major security weaknesses" in Virtual Private Networks: technologies and solutions by R. Yuan [7].

2.2.6 L2TP
Layer 2 Tunnelling Protocol, L2TP, is a protocol developed by IETF, the Internet Engineering Task Force, and is defined in RFC 2661 [35]. It is an open source VPN that acts in layer 2 [7]. In this tunnelling protocol, the client side of the tunnel is called L2TP Access concentrator, LAC, and the server side of the tunnel is called L2TP Network Server, LNS. To establish a connection between the LAC and the LNS there are two types of messages being exchanged [35]. The first type is Control Messages. The messages of this type are provided a reliable transport service, whereas the messages of the second message type, Data Messages, are provided an unreliable transport service via UDP. The control messages carry the negotiation terms of the tunnel and establish it, and the data messages carry the encapsulated PPP frames [7, 35].

The L2TP header and the data messages that contain the payload, which often are encapsulated PPP frames but may be other packets, are sent as UDP datagrams.

Using UDP as the carrier of the L2TP packets adds some characteristics to L2TP. Using UDP means not requiring a proper arrival of the packets and it does not retransmit lost packets, which is preferable in some applications. Another characteristic that is a result of using UDP is that L2TP is unsuitable solution when one, or both, of the tunnel sides NAT [20, 35]. That is because the UDP datagrams are sent from an arbitrary port and received from the port 1701 on both sides. That is, if machine A initiates a session with machine B, it sends from an arbitrary port from its own machine to the port number 1701 in B. B replies from an arbitrary port to A’s port number 1701 [35]. This property may be a limitation of L2TP based VPNs.

When it comes to security issues, L2TP does not provide protection to its tunnels [7, 20, 35]. The only authentication that is provided by L2TP is during the establishment of the tunnel. This authentication is an available option if one or both the sides of the tunnel, i.e. the LAC and the LNS, want to authenticate each other. To exploit this advantage, a secret must be shared between the LAC and the LNS. Once the establishment is completed, an outsider may corrupt the tunnel by adding packets to it [35]. However, when L2TP encapsulates a protocol packet, it gets its security mechanisms,
including the encryption and the authentication methods [7].

To provide authentication and confidentiality, IPsec may be used. In this scenario, the packets appear like UDP/IP packets in the carrier media [35]. When using IPsec to gain security the protocol is known as L2TP/IPsec. Using L2TP/IPsec results in high overhead, due to the size of the encapsulated header. This issue may be solved though, by using compression methods [7].
3 Methods
This chapter describes the methods used during the thesis project. It starts with identifying the problem, followed by the research method used in this thesis project. A short description of iPerf, which is a tool used in the experiments, is presented. The chapter ends with a description of the experiments and the environment in which they are done.

3.1 Identifying the Problem
In order to identify the problem, we started reading books and articles. The books and the articles supplied us with knowledge about what different technologies exist to implement VPNs. Listing the VPN solutions and what they rely on gives us the possibility to choose what VPNs will be included in our experiments.

Studying the technology area helps us to specify some problems that can be studied. The problems we listed were:

- **Security**: A potential problem is evaluating the authentication algorithms to experiment whether or not a VPN solution is solid enough to ensure that users that do not have the right to access the network can never access it. Another security issue is evaluation of encryption algorithms that are used in the chosen VPN solutions. Previous work is done in this area [36, 37].
- **Round-trip delay**: The time it takes for the packet to reach its destination plus the time it takes to send a response back.
- **Overheads**: VPN solutions add overheads to the packets. The size of the overheads differs from a solution to another. Overheads cause more packets to be sent over the tunnel and deplete the tunnel capacity. They result in longer time to completely transmit a data block. The overheads are injected in the packet at the expense of the data to be transmitted.
- **Throughput**: Throughput is the number of bits of data that actually reaches the receiver over the tunnel within a second. Throughput is measured in Mbit/s, that is, throughput takes into concern both time and the amount of data that is sent.

The common way of using VPNs is to rely on a software-based solution. Many of these software-based solutions are free to install. However, as it is known, software-based solutions operate slower compared with solutions that rely on additional hardware. The most common problem in the most used VPN type is therefore time-related.

Among the presented problems above, overheads, round-trip delay and throughput are problems that affect the time it takes to send data. The one of them that is most comprehensive is throughput. That is because it takes in concern both time and the amount of data that is sent during the time unit. Therefore, the problem to be answered in this thesis project concerns the effect of the VPN technology on the throughput.
3.2 Choices of solutions to be evaluated
The VPNs we choose to evaluate are IPsec, OpenVPN, SSH port forwarding and SSH VPN. The reason we chose to evaluate IPsec and OpenVPN is mainly because they are popular and well used and therefore it would be interesting to investigate if they are the best alternatives out there or if there are better alternatives and the users should start using these alternatives instead [8]. SSH port forwarding, which is not common because of the possibility of using illegal protocols in it, is chosen to represent an uncommon VPN [15]. By choosing it beside IPsec and OpenVPN, we have a point of reference between a more used and less used solutions.

SSH VPN shares some concepts with SSH port forwarding, and other concepts with OpenVPN. Therefore, it is interesting to include it in order to compare it with both and find out what factors rule the throughput.

3.3 Research Methods
Given that the throughput is a numerical value, it is clear that we have to do measurements. That means that we have some variables that we change to find the relationship between that variable and one or more of VPN characteristics. The experimental research method is the scientific method that describes the method followed in our project [38].

3.4 iPerf: Throughput Measurement Tool
As mentioned, the desired data to collect is throughput. A common way of collecting such data and measuring network performance is with the use of a packet generator tool. One such tool and the tool used in this work is iPerf [39]. iPerf is chosen to collect and analyse data because it is freely available, easy to use and because of its ability to measure the desired metric throughput. To make measurements with iPerf, one computer runs iPerf in client mode and another runs it in server mode and the iPerf client connects to the iPerf server and sends traffic to it. Some features of iPerf are its ability to report packet loss when sending UDP traffic, measure jitter and measure traffic for a specific period of time [39]. iPerf is supported by many platforms, including MAC OS X, Windows and Linux. The version of iPerf used in this thesis project is version 2.0.5 for Mac OS X.

Another property that iPerf has is that one can control the sending rate over the network when using UDP. By sending rate, we mean the amount of data that leave the transmitter to the receiver. It does not necessarily reach the receiver successfully. If the sending rate is too high compared with the network capacity, we get a high packet loss. To investigate the throughput we have to find the sending rate at which packet loss starts. More details about how we determined this limit in this thesis project are presented in section 3.5.1. A measurement in iPerf by sending UDP datagrams over the network results in the table shown in Figure 12. When running this example, IPsec VPN was activated.
The figure is a screenshot in the machine that corresponds to the client edge in the link. The IP-address 10.1.0.10 that is shown in the figure belongs to the server edge of the link. The data that is sent is taken from the file uc.txt. The parameter -u means that we specify UDP traffic to be sent and the parameter -b fixes the sending rate of data. In this example, we chose the sending rate to be 170 Mbit/s.

The shadowed part of the figure is the server report. Here, iPerf reports that the bandwidth is 150 Mbit/s. This number is the segment of data that actually reached the server edge of the link, i.e. what we define to be our throughput. The packet loss is shown in the figure to be 11%.

When sending TCP traffic, there is no need to specify a sending rate. The highest capacity of the link is reported and since it is TCP, there is no packet loss. In Figure 13, we show the report of a TCP throughput measurement in iPerf. IPSec run in this example as well.

### 3.5 Experiment Environment and Parameters

To evaluate the four VPN technologies previously presented, an environment for performing the three different performance comparative experiments is constructed.

The experiments are carried out in the NSLab at KTH Kista. The test bed used in every experiment is simulating a site-to-site VPN scenario and shown in Figure 14. As Figure 14 shows, the environment consists of three Gigabit Ethernet networks: 10.1.0.0/24, 10.2.0.0/24 and 192.168.0.0/24. The two class A networks are representing the two sites respective private networks, while the class C network is a public network, which represents the unreliable transport medium on top of which the VPNs are constructed, i.e. the Internet.
The two gateways GW1 and GW2 in Figure 14 are Intel Atom D525 1.80 GHz computers running Ubuntu Server version 14.04.1. Two physical ports are used on each gateway, both of which are Gigabit Ethernet ports.

Both private networks each have one host, which connects to their respective network via Gigabit Ethernet ports. The host at network 10.1.0.0/24 is named host1 and the host at network 10.2.0.0/24 is named host2. host1 is the iPerf traffic source, the iPerf client, and host2 is the iPerf server. Both hosts are MacBook Pro Early 2011 computers running MAC OS X Yosemite version 10.10, each equipped with one 2.3 GHz Intel Core i5 processor.

3.5.1 Test Parameters
In order to figure out how a specific parameter affects the performance of the VPNs, a number of parameters are fixed and a single parameter is modified. Next is a description of each set of experiments.

Parameter1: Transport Layer Protocol
The first set of experiments aims to discover the maximum throughput of the different VPN solutions when transporting traffic over different transport layer protocols.

The parameters in this scenario are set as follow:
• The compression is inactivated in the tunnels.
• The encryption algorithm that is used is AES-128.
• The parameter that is changed is the transport layer protocol, by switching between the TCP and UDP protocols.

The traffic sent is UDP datagrams of 1470 bytes and the buffer size of the client and server is 9.00 KB respective 192 KB. When performing UDP tests like these with iPerf it reports the amount of lost packets for every test. When packet loss is 0%, the VPN solution managed to transfer UDP packets so that the throughput is equal to the sending rate.

Our approach to find the throughput for the VPNs in UDP traffic scenario was to send data at different sending rates. Each test of a selected sending rate is iterated ten times before the average packet loss in per cent is calculated. The
first time this average reaches 1%, the throughput is defined to be exactly below the sending rate that generated that packet loss. This process is shown in Figure 15.

The figure shows an example of how the throughput of IPsec was decided by analysing the collected data of sending UDP traffic. The average packet loss exceeded 1% at the sending rate 153 Mbit/s, the right red marked cell. The average packet loss at the sending rate 152 Mbit/s was less than 1%, the left red cell. Therefore, the throughput of IPsec was determined to be 152 Mbit/s when sending UDP traffic.

This test procedure is followed for all VPN solutions except SSH port forwarding, since it is not able to transfer UDP traffic in its standard configuration as described in chapter 2.2.3.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.74%</td>
<td>2.98%</td>
<td>2.45%</td>
<td>2.81%</td>
<td>4.33%</td>
<td>5.11%</td>
<td>11.40%</td>
</tr>
<tr>
<td>2</td>
<td>0.00%</td>
<td>0.00%</td>
<td>1.13%</td>
<td>1.02%</td>
<td>3.14%</td>
<td>1.86%</td>
<td>1.69%</td>
<td>5.41%</td>
<td>4.78%</td>
<td>11.02%</td>
</tr>
<tr>
<td>3</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>1.77%</td>
<td>1.54%</td>
<td>3.52%</td>
<td>3.69%</td>
<td>4.41%</td>
<td>11.64%</td>
</tr>
<tr>
<td>4</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.75%</td>
<td>1.46%</td>
<td>0.83%</td>
<td>2.59%</td>
<td>3.72%</td>
<td>4.41%</td>
<td>11.39%</td>
</tr>
<tr>
<td>5</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>1.44%</td>
<td>2.80%</td>
<td>2.46%</td>
<td>3.89%</td>
<td>3.08%</td>
<td>11.52%</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.01%</td>
<td>0.94%</td>
<td>0.56%</td>
<td>0.32%</td>
<td>2.30%</td>
<td>5.66%</td>
<td>5.45%</td>
<td>12.21%</td>
</tr>
<tr>
<td>7</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>1.08%</td>
<td>0.00%</td>
<td>3.18%</td>
<td>3.28%</td>
<td>5.05%</td>
<td>10.15%</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>1.70%</td>
<td>2.53%</td>
<td>3.20%</td>
<td>3.37%</td>
<td>5.20%</td>
<td>11.66%</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.00%</td>
<td>0.00%</td>
<td>1.07%</td>
<td>1.16%</td>
<td>2.54%</td>
<td>2.45%</td>
<td>3.39%</td>
<td>5.18%</td>
<td>11.48%</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.80%</td>
<td>1.62%</td>
<td>3.01%</td>
<td>1.28%</td>
<td>3.55%</td>
<td>6.12%</td>
<td>12.02%</td>
<td></td>
</tr>
</tbody>
</table>

Average: 0.00% 0.00% 0.19% 0.45% 1.99% 1.79% 2.55% 4.03% 4.88% 11.45%

Figure 15. Packet loss (%) at different sending rates.

**Parameter 2: Encryption algorithm**

The second set of experiments aims to investigate how different types of encryption algorithms and different key sizes affect the throughput of the VPN solutions.

The parameters in this scenario are set as follow:
- The transport layer protocol is TCP.
- The compression is inactivated in the tunnels.
- The parameter that is changed is the type of encryption algorithm used.

The following three different encryption algorithms in CBC-mode are used:
- Blowfish with a key length of 128 bits.
- AES with key lengths of 128, 192 and 256 bits.
- 3DES, with its key length of 168 bits.

**Parameter 3: Compression**

The third and last set of experiments aims to investigate the effect of compression on the throughput for every chosen VPN technology.

The parameters are as follow:
- The transport layer protocol is TCP.
• The encryption algorithm that is used is AES-128.
• The parameter that is changed is compression.

In each VPN solution compression is activated respectively deactivated. When the compression is activated so that the tunnels compress data, two types of data are sent:
  • Compressed data in form of a zip file. The original data is a video with the extension .mp4, that is already compressed data, of 38.1 MB. The new size after additional compression, by converting to a zip file, is 37.9 MB.
  • Uncompressed data in form of a file of 136 MB. The file consists entirely of zeros. Containing only zeros means that the data will be compressed efficiently in the tunnel.

By deactivating compression, the content of the sent data is no longer of importance. Therefore, only one type of data is sent over the tunnel that does not compress data.
4 Results
This chapter presents the results of the tests carried out in this work and described in Chapter 3. The chapter is divided into three sections, each presenting the result from a specific test, i.e. the test where different encryption algorithms are used, where compression is activated respective deactivated and where TCP versus UDP traffic is transferred. Because all the tests are performed in a static environment, the standard deviation of them is low. The coefficient of variation, which is the quotient of the standard deviation divided by the average of the throughput, lies between 0-9%. Such a low standard deviation in respect to the average is insignificant to the final results and therefore not shown in any of the following figures or discussed any further. For more details about the accuracy of the results, see Appendix.

4.1 Reference Measurement
The following figure, Figure 16, presents the result of measuring the throughput without using VPN. The data that is sent over IP is neither encapsulated nor encrypted, and the data that is sent over IPIP is only encapsulated. The results show the effect of the VPN function components, encapsulation and encryption.

In these tests, the encryption algorithm that is used in the VPNs is AES-128 and the traffic sent is TCP traffic.

The throughput of the data sent over IP is 939.7 Mbit/s. The corresponding throughput over IPIP is 927.9 Mbit/s. The highest throughput obtained in conjunction with using a VPN, that is encrypted data, is 167.6 Mbit/s in SSH port forwarding.

![Figure 16. Throughput obtained by transferring data with no encapsulation or encryption, IP, with only encapsulation, IPIP, and with both encapsulation and encryption, the VPNs.](image)

4.2 Transferring data over TCP/UDP
Figure 17 below shows the throughput that IPsec, OpenVPN and SSH VPN had when transferring UDP traffic. SSH port forwarding is not included in the
figure because there is no possibility to transfer UDP traffic over it, as discussed in section 2.3.3, and therefore such a test could not be carried out. In UDP, IPsec results in more than 150 Mbit/s in throughput, whereas OpenVPN and SSH VPN give less than 100 Mbit/s.

Figure 17. UDP throughput different VPNs.

Figure 18 presents the throughput, for each VPN, that is obtained by transferring TCP traffic through the VPNs. Around 165 Mbit/s throughput is generated by IPsec and SSH port forwarding. OpenVPN and SSH VPN give the throughput 83.3 and 95.0 Mbit/s respectively.

Figure 18. The throughput obtained by sending data over TCP in different VPNs.

4.3 Different encryption algorithms
The results of the measured throughput of the VPNs using different encryption algorithms are shown in Figure 19. The figure shows how a VPN solution tends to perform compared with the other three VPN solutions. It also shows the performance, measured in throughput, of different algorithms in the same VPN solution. The results of this measurement varies between little more than 230 Mbit/s, obtained in SSH port forwarding and IPsec using Blowfish, and around 35 Mbit/s, obtained in SSH VPN and OpenVPN using 3DES.
4.4 Activating/Deactivating Compression

The figure below, Figure 20, shows for every VPN, whether or not we get higher throughput by using compression, when sending data over VPNs. In OpenVPN and SSH port forwarding, the throughput increases by activating the compression, whereas it decreases in IPsec and SSH VPN.
In the following figure, Figure 21, the throughput of each VPN is depicted in two different cases. The first case, the dark blue column in the figure, is the throughput when compression is deactivated and uncompressed data is sent through the VPN. The other case, shown in the lighter blue column, is when compression is activated and the traffic transferred through the VPN already is compressed. The purpose of this figure is to show the difference of the measured throughputs in the two scenarios in the same VPN solution and thereby showing the cost of using compression without being able to compress the data any further. All VPNs show dramatically fall in throughput when sending compressed data in the tunnel that compresses the traffic, except OpenVPN, which keeps almost the same throughput.

Figure 21. Cost of compression function when it runs on a compressed file.
5 Analysis
In this chapter we present our analysis of the results that are presented in the previous chapter. The chapter discusses the results of the respective measurement parameter and ends with general analyses of the results.

5.1 The reference measurement
Figure 16 showed us that the encryption is the most costly operation in a VPN. Transferring data over IPIP gave insignificantly lower throughput than the throughput obtained by transferring data over IP. In contrast, when sending the encrypted data using a VPN, the throughput sinks dramatically.

5.2 Environment with TCP versus UDP
As expected, IPsec had higher throughput than OpenVPN and SSH VPN. That is because of the virtual interfaces that are used in OpenVPN and SSH VPN. By comparing the results from the VPNs that were obtained by sending data over TCP and over UDP we can see that the throughput when sending UDP traffic is close to the throughput obtained by sending TCP traffic.

5.3 Environment with different encryption algorithms
OpenVPN and SSH get their encryption algorithms from the same library, OpenSSL. However, OpenVPN uses TLS, Transport Layer Security, but SSH VPN does not. That can be the reason behind the slightly higher throughput obtained in SSH VPN.

In the same environment we notice that IPsec and SSH port forwarding result in similar throughput, except when we use the 3DES encryption algorithm, where a significant difference is found. It was difficult to define the reason to this difference, since both SSH and IPsec use OpenSSL as a cryptographic library. They were, furthermore similar regarding the other measured encryption algorithms.

The encryption algorithm that is the fastest in our measurements is Blowfish, whereas the slowest one is 3DES.

5.4 Environment with and without Compression
The result of enabling compression in the VPNs shows that in some VPNs, we got higher throughput by using compression whereas we got lower throughput when we used it in other VPNs. The reason of obtaining these results is considered to be beyond the scope of this thesis project. However, it is important to know what maximum throughput a specific VPN has during optimal conditions, i.e. when the properties that benefit its throughput are enabled.

OpenVPN and SSH port forwarding have higher throughput when they use compression than they do when they are not using it. In contrast, IPsec and SSH VPN result in lower throughput when they use compression.

IPSec, SSH VPN and SSH port forwarding all showed a decrease in throughput when trying to compress already compressed data, but OpenVPN did not. According to the documentation on the official website of OpenVPN[40], this behaviour depends on the compression algorithm that
was used in the experiments in this thesis project and is commonly used in combination with OpenVPN. This algorithm, which is called LZO, may enable a property of detecting whether the data is already compressed. This is done by taking samples of the transmitting data and checking whether it is already compressed. If that was the case, compression is deactivated. This operation is repeated periodically and the compression is deactivated as long as the data is defined to be unsuitable to compress.

5.5 General Analysis of the Used VPNs

By looking generally at the results we see a significant difference between the two VPNs that use virtual interfaces, OpenVPN and SSH VPN, and the two that do not, IPsec and SSH port forwarding. By using the virtual interface TUN, OpenVPN and SSH VPN add additional costs in form of time to the execution of packet processing. The reason is that these generate more IP table lookups than the ones that do not use virtual interfaces. That is because an additional interface means additional lookups. Each time a packet reaches an interface, the interface gets information from the IP table about where to forward the packet in the next step, as shown in Figure 22.

![Figure 22. The process of tunnelling a packet in an OpenVPN gateway.](image)

In Figure 22 the green arrows denote to the path of a packet whereas the black arrows denote the lookup operations. A packet reaches the OpenVPN gateway in its interface, eth0. The first operation is to make a lookup in the IP table to know how to handle the packet. That step is clarified in the figure by the black arrow number one. The second step, green arrow flagged with number two, is to pass the packet to the virtual interface, which in turn makes a lookup in the IP table to know where to forward the packet, black arrow number three. The green arrows number five and six represent the packet forwarding to the OpenVPN software processing it and sending it to the interface eth1. These steps mean changing from kernel to user space once in each direction. The interface eth1 makes a lookup in the IP table, black arrow
number six and the packet leaves the gateway.

Processing a packet as shown in Figure 22 means 3 lookups in the IP table and changing between kernel and user space. These extra IP table lookups are one of the reasons we see a noticeable difference between the two VPN solutions that use virtual interfaces and the two that do not.

IPsec avoid switching between the two spaces, kernel and user space, and has less IP table lookups. That is because IPsec runs in the kernel and has no virtual interface. An illustration in Figure 23 represents a packet that is received in interface eth0. The interface makes a lookup, denoted by a black arrow, number one. The packet is then forwarded to StrongSwan that is the software over which IPsec runs operating in the kernel. That is represented in the green arrow number two. In number three the interface eth1 receives the packet from the software. Lastly, the exit interface, eth1, makes a lookup in the IP table before the packet leaves the gateway. The black arrow, number four in the figure, denotes the last lookup.

**Figure 23.** The tunnelling process of a packet in an IPsec gateway.

SSH port forwarding, unlike IPsec, switches the packets between kernel and user space. Switching between these modes costs time. But in spite of that we see that SSH yielded the same results as IPsec and sometimes evenbetter results than IPsec. The theory we formulated about this is summarized in Figure 24, where the green frame denotes the network layer header, the blue frame denotes the transport layer header, the white frame with red border denotes unencrypted data and the red frame denotes encrypted data.

The theory is that the client in SSH port forwarding addresses packets to the local gateway. When a packet arrives at the local gateway, its network layer header is removed from the packet because it has reached the destination to which it is addressed. Then the transport layer header is also removed and the data is sent to SSH, since the packet uses the port number that addresses SSH. The encryption process is time consuming. But because of the extraction of the packet data in the local gateway, the encryption is performed only on the smaller data part of the packet. In IPsec, a larger part, which includes all layers down to the network layer, is encrypted. The reason that SSH only needs to encrypt the application layer data is that the other side of the tunnel
knows nothing about the client who requests the data. The communication is only between the gateways. That makes SSH avoid attaching additional encrypted layers in the packet. Therefore SSH port forwarding yields high throughput results despite switching between kernel and user mode.

Figure 24. Processing of a packet sent from the client to the resource in SSH.
6 Conclusions
This chapter presents a discussion about the thesis project, how it was performed, the validity of the results and conclusions drawn by these results. It ends with a section that suggests future work in this area.

6.1 Discussion
In this thesis project we measured the throughput in different VPNs to compare them with each other. Each VPN is also compared with itself when varying some parameters to examine how that affects the throughput of the VPN. The parameters that are varied in the VPNs are: activating and deactivating compression, using different encryption algorithms and using TCP and UDP traffic. As a comparative reference, we measured the throughput of the traffic sent in plain IP and the throughput that is tunnelled using the IPIP protocol.

The tests in this project were done in a closed environment, so that no external factors have affected the results. The tests were not dependent on theoretical information that can mislead the results.

The thesis project concludes that the choice of what VPN solution to be used depends on what property the user prioritizes. If the priority is high throughput and the user gives less concern to security, the best solution then is to use Blowfish. However, it is stated in RFC 2451 that the algorithm contains weak keys [41]. That means that it is not the best solution if the security is of high concern. Even 3DES is not recommended by Cisco to use if it is not absolutely necessary [42]. We choose not to go into further details about the properties of each encryption algorithm used in this work because that would be beyond the scope of this report.

The tests have shown that whether TCP or UDP is used as transport protocol does not have any significant effect on the throughput. Therefore the user does not need to consider what type of traffic shall be transferred over the VPN in regard to the TCP or UDP when choosing VPN.

The maximum throughput was obtained by SSH port forwarding when compression was activated. SSH port forwarding is a good solution when a single source is desired to be reached at the other end of the tunnel, which is a limitation for this solution. The next highest throughput was obtained when using IPsec with compression disabled. This solution does not have the restriction that SSH port forwarding has, i.e. the whole private network at the other side of the tunnel can be reached.

6.2 Future work
In this thesis project we chose four different VPN solutions to evaluate and compare in terms of throughput performance: IPsec, OpenVPN, SSH VPN and SSH port forwarding. As presented in chapter 2 in this report, there exist other VPN solutions, for example GRE or L2TP, which may have proven to be more efficient than the ones evaluated if they had been included in this thesis project. To produce a more comprehensive evaluation of different VPN
solutions, these solutions as well as other VPN solutions not mentioned here, can as a suggestion be evaluated in a similar way to the one in this project.

Based on the results we got, the encryption is the most costly factor in a VPN. That means that can improve a throughput of a VPN by developing faster encryption algorithms.

When speaking about VPNs, security is of great importance. The security is not only how to improve an encryption algorithm that is secure enough, but also suggesting new user-friendly solutions for how the pre-shared key can be shared and how to authenticate users that have different authentication to a local network. An existing solution is RADIUS that provides a centralized authentication for users who use a specific network [43]. However there is still space for more studies about the efficiency of these types of methods, how to improve them and suggest some new.

Another suggestion for future work is to evaluate some other metric of the VPNs than the one in this thesis project, throughput. For example, could the security they provide be evaluated to find out if they differ in any way and from those results rank the VPNs in terms of security level?

Lastly, this thesis project shows that the SSH port forwarding has as high throughput as IPsec. The question that arises by this result and could be a subject of a future work is whether this will be the case even when loading the gateways with multiple requests.
References

Appendix

In this appendix we present the coefficient variation of the tests. The coefficient variation has not exceeded 9% in any test. The coefficient of variation is defined by the formula:

\[
\text{Coefficient variation} = \frac{\text{Standard deviation}}{\text{Average}}
\]

For UDP, where the approach of calculating the throughput was different from the other measurements, we present the average packet loss in per cent and the standard deviation of that average, and not the coefficient variation of UDP measurements.

Table 1. The coefficient of variation in the reference tests.

<table>
<thead>
<tr>
<th>IP</th>
<th>IP-IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 2. The coefficient of variation in the second test scenario, TCP traffic.

<table>
<thead>
<tr>
<th></th>
<th>Coefficient of variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPsec</td>
<td>4.2</td>
</tr>
<tr>
<td>OpenVPN</td>
<td>8.4</td>
</tr>
<tr>
<td>SSH port forwarding</td>
<td>1.6</td>
</tr>
<tr>
<td>SSH VPN</td>
<td>4.7</td>
</tr>
</tbody>
</table>

Table 3. The average packet loss and standard deviation of the sending rate that is defined as the throughput in the second test scenario, UDP traffic.

<table>
<thead>
<tr>
<th></th>
<th>Average packet loss (%)</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPsec</td>
<td>0.45</td>
<td>0.49</td>
</tr>
<tr>
<td>OpenVPN</td>
<td>0.39</td>
<td>0.51</td>
</tr>
<tr>
<td>SSH VPN</td>
<td>0.86</td>
<td>0.82</td>
</tr>
</tbody>
</table>
Table 4. The coefficient variation in the third test scenario.

<table>
<thead>
<tr>
<th></th>
<th>AES 128</th>
<th>AES 192</th>
<th>AES 256</th>
<th>Blowfish</th>
<th>3DES</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPsec</td>
<td>4.2</td>
<td>2.5</td>
<td>3.2</td>
<td>6.0</td>
<td>2.2</td>
</tr>
<tr>
<td>OpenVPN</td>
<td>8.4</td>
<td>3.8</td>
<td>8.6</td>
<td>2.4</td>
<td>0.1</td>
</tr>
<tr>
<td>SSH port forwarding</td>
<td>1.6</td>
<td>4.1</td>
<td>3.0</td>
<td>3.1</td>
<td>0.1</td>
</tr>
<tr>
<td>SSH VPN</td>
<td>4.7</td>
<td>6.1</td>
<td>5.3</td>
<td>5.2</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 5. The coefficient of variation in the fourth test scenario.

<table>
<thead>
<tr>
<th></th>
<th>Compression activated</th>
<th>Compression deactivated</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Compressed data</td>
<td>Uncompressed data</td>
</tr>
<tr>
<td>IPsec</td>
<td>2.0</td>
<td>3.9</td>
</tr>
<tr>
<td>OpenVPN</td>
<td>5.8</td>
<td>2.6</td>
</tr>
<tr>
<td>SSH port forwarding</td>
<td>1.6</td>
<td>3.7</td>
</tr>
<tr>
<td>SSH VPN</td>
<td>3.9</td>
<td>4.4</td>
</tr>
</tbody>
</table>