VIVEKANAND COLLEGE FOR ADVANCED COMPUTER AND INFORMATION SCIENCE SURAT

AFFILIATED TO
VEER NARMAD SOUTH GUJARAT UNIVERSITY, SURAT

SEMINAR REPORT
ON

VIRTUAL PRIVATE NETWORKS
An Analysis of the Performance in State-of-the-Art Virtual Private Network solutions in Unreliable NetworkConditions

GUIDED BY:
DR.MEGHA POLISHWALA

PRESENTED BY:
MAYUR VAVADIYA

EXAM NO :
2793

Reg. No. E/4682/SURAT Dt. 15-11-2000

Ph. 70433 65488

(Abhinav Education Trust-Surat) vivekanandsgu@yahoo.in

VIVEKANAND COLLEGE FOR B.C.A • ABHINAV EDUCATION TRUST •

VIVEKANAND COLLEGE FOR ADVANCED COMPUTER & INFORMATION SCIENCE

(Affiliated to the Veer Narmad South Gujarat University)

Near Saroli Bridge, Jahangirpura, Surat-Olpad Road, SURAT-5.

CERTIFICATE

This is to certify that Mr. /Ms **VAVDIYA MAYURKUMAR HARESHBHAI, Exam No 2794** student of final year BCA during year 2023-24 has Submitted seminar report entitled **VPN (. VIRTUAL PRIVATE NETWORK).** His / Her work is found satisfactory.

Guide Name : **Dr. Megha Rana**

Signature of Guide MRana.

Vivekanand College for BCA

Examiner

Date :

Signature of Examiner

[illegible] COLLEGE FOR ADVANCE COMPUTER [illegible]

I/C Principal
Vivekanand College for BCA
SURAT

TABLE OF CONTENTS

Abstract

This study aimed to identify the differences between state-of-the-art VPN solutions on different operating systems. It was done because a novel VPN protocol is in the early stages of release and a comparison of it, to other current VPN solutions is interesting. It is interesting because current VPN solutions are well established and have existed for a while and the new protocol stirs the pot in the VPN field. Therefore a contemporary comparison between them could aid system administrators when choosing which VPN to implement. To choose the right VPN solution for the occasion could increase performance for the users and save costs for organizations who wish to deploy VPNs. With the remote workforce increasing issues of network reliability also increases, due to wireless connections and networks beyond the control of companies. This demands an answer to the question how do VPN solutions differ in performance with stable and unstable networks?

This work attempted to answer this question. This study is generally concerning VPN performance but mainly how the specific solutions perform under unreliable network conditions. It was achieved by researching past comparisons of VPN solutions to identify what metrics to analyze and which VPN solutions have been recommended. Then a test bed was created in a lab network to control the network when testing, so the different VPN implementations and operating systems have the same premise. To establish baseline results, performance testing was done on the network without VPNs, then the VPNs were tested under reliable network conditions and then with unreliable network conditions. The results of that were compared and analyzed.

The results show a difference in the performance of the different VPNs, also there is a difference on what operating system is used and there are also differences between the VPNs with the unreliability aspects switched on. The novel VPN protocol looks promising as it has overall good results, but it is not conclusive as the current VPN solutions can be configured based on what operating system and settings are chosen. With this set-up, VPNs on Linux performed much better under unreliable network conditions when compared to setups using other operating systems.

The outcome of this work is that there is a possibility that the novel VPN protocol is performing better and that certain combinations of VPN implementation and OS are better performing than others when using the default configuration. This works also pointed out how to improve the testing and what aspects to consider when comparing VPN implementations.

Keywords: Virtual Private Network, VPN, WireGuard, IPSec, OpenVPN, Performance, Unreliability, Packet Loss, Delay.

1 INTRODUCTION

A Virtual Private Networks (VPN) is a way to extend a private network through a public network such as the Internet. Users may then use the VPN to access data on the private network through the Internet as if they are directly connected to the private network.

It is understood that using a VPN may reduce performance of the network connection, due to the fact that VPN adds encryption overhead which will increase the latency. The performance is sacrificed to achieve a higher degree of privacy. By examining different VPN implementations, we can find out how much the performance degrades.

According to a study by two companies FlexJobs and Global Workplace Analytic in 2017, more and more employees are working from home: Between 2005 and 2015 telecommuting grew by 115%. An on-site user that is working on corporate network infrastructure, which is controlled and maintained by the corporation will likely have a more stable connection than the remote worker. A remote worker may use different types of connections that are not managed by the remote workers company. This could lead to an unstable network connection as it is not controlled by professionals directly, as they preferrably are on-site. Considering this scenario, VPN can potentially differ in performance if the connection is unreliable. The fact that the amount of remote workers is increasing and that VPN is a multi-billion-dollar industry, which is also projected to grow further according to Khan et al. (2018), begs the question to be answered; how does the performance of certain VPN implementations differ?

This project compares state-of-the-art VPNs and how they differ in performance under normal and unreliable network conditions on different operating systems (OSs). The performance is measured in throughput and degradation is measured in how much, if any, performance was lost after the connection unreliability were introduced.

The results are intended to be used as an aid to system administrators when to evaluate the options for which VPN solution to deploy. The results can be used as a steppingstone to create other VPN experiments and to avoid pitfalls that were discovered in this study. If a system administrator intends to use the same configuration as in this study, then the results can be beneficial without having to do their own testing.

The structure of the rest of this study is as such: Chapter 2 covers general background information about the VPN solutions that are used in the testing of this report and the metrics that are measured when testing the performance of VPNs. Chapter 3 is about the motivation for undertaking this study, why it is important to benchmark VPN performance and especially under unreliable network conditions, this chapter also contains the research question and the objectives that are undertaken to answer the research question. Chapter 4 is about how the plan was created to undertake the objectives and to answer the research question, what tools were used and how the experiment was set up and configured also how the data was handled that was produced by the experiment testing. Chapter 5 is the presentation of the results from the experiment. Chapter 6 explains the conclusions that is drawn from the results.

2 BACKGROUND

This chapter describes some information about key concepts of the study. First the concept of Virtual Private Networks is explained. Then the most popular VPN solutions are mentioned, which are also the VPN solutions tested in this study. Also, other VPN solutions are mentioned. Finally the performance of VPNs is covered to explain how to measure network performance related to VPNs.

2.1 What is a Virtual Private Network?

Virtual Private Networks are defined broadly as a way to extend a private network through the public network such as the Internet (Brown, 1999). A private network is a network that exists in a Local Area Network (LAN). The extension of a private network that is done with the use of VPN technology can be used to access other private networks remotely through the use of a VPN tunnel. A tunnel in networking is a way to send data that is not normally supported by a network protocol by repackaging data in a packet to another protocol. If two routers have tunneling configured it is possible to encapsulate the data to send directly to each other over the Internet and then decapsulate the payload that was sent through the tunnel to send it onwards to the destination.

In addition to tunneling private networks together, other strong aspects of VPN are the capabilities of adding encryption and integrity to the tunnel (Brown, 1999). VPN can be used to keep sensitive data inside the private network and ensure that the data is unchanged during transport. It adds one extra layer of security by not exposing the private network to the public Internet but at the same time being able to access a private network remotely.

Figure 1a is the first example of a VPN type, it is called *host-to-network* or *remote access*. It is used as the name suggests, to connect one computer to a private network, for example a remote worker that needs to access private company files on the company network.

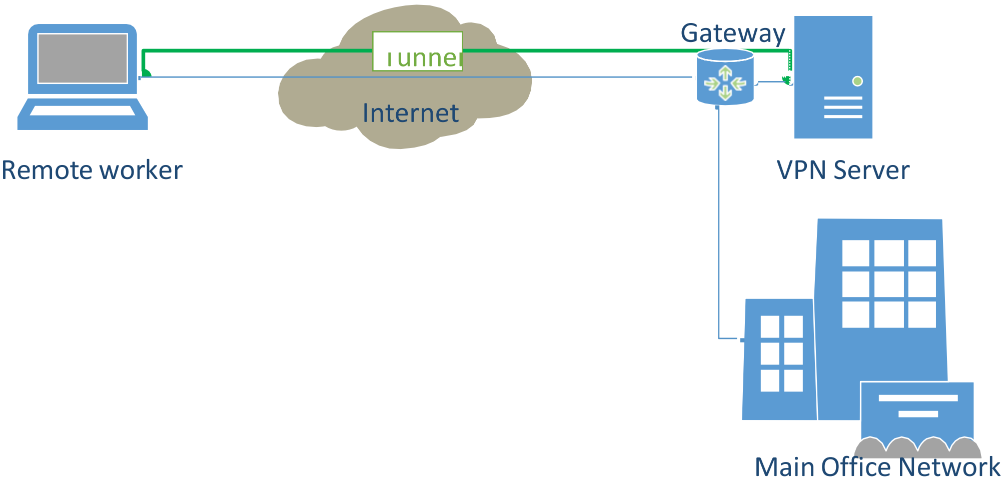


Figure 1a – Remote access VPN type

Figure 1b shows a site-to-site network, which is used to connect the private networks of two branches of the same company to share their local private network and data. It is also possible to utilize this type of VPN between two different companies or organization to collaborate.

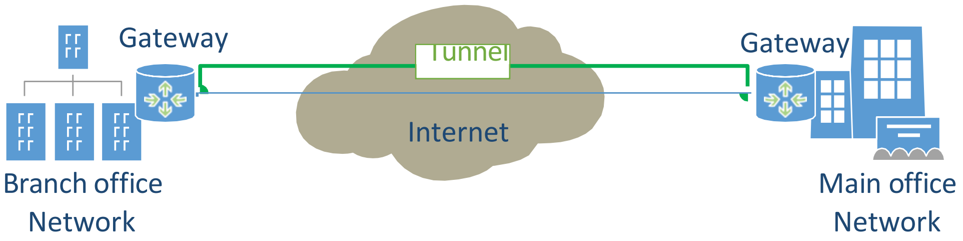


Figure 1b – Site-to-site VPN type

One of the reasons to use a VPN is because it adds security aspects to a network connection. Security in this case means the protection of network traffic regarding confidentiality, integrity and authentication. Confidentiality is achieved through encryption. If someone would get access to the packets sent in the VPN tunnel that is encrypted they would only see encrypted data that is unreadable. Depending on what type of encryption is being used, it could be very hard to decrypt. Integrity is achieved through mechanisms that detects if any packets are tampered with. VPNs make use of a number of cryptographic algorithms and protocols to add for example authentication mechanisms in place to restrict who can use it. There are many different levels of security that can be provided with different VPN applications. Another reason to use VPNs is to access other private networks remotely in the ways stated in Figure 1a and 1b.

2.2 Different VPN solutions

According to a study (Khan et al. 2018) two of the most common VPN solutions are IPSec (IP Security Protocol) and OpenVPN. A new solution named WireGuard is according to the white paper (Donenfeld, 2018) being developed to replace both aforementioned VPN solutions while claiming to have better performance. OpenVPN recently responded that it welcomes new projects such as WireGuard but that OpenVPN and WireGuard have different goals based on their users needs ("The New Cloudflare VPN: What It Is & Isn't | OpenVPN", n.d.).

Table 1 shows some information about the different VPN solutions examined in this study. The market share describes what VPN solution is supported by the top 15 VPNs on the market (Khan et al. 2018). The most supported VPN solution is OpenVPN followed by IPSec. In comparison WireGuard is virtually non-existent as only one of the VPNs tested in the study by Khan et al. (2018) supports WireGuard. All VPN solutions are supported on the three main OSs. OpenVPN consists of around 70.000 lines of code according to their own statement (OpenVPN, 2019) excluding cryptographic primitives while the latest statement from WireGuard is that it is around 4000 lines of code (Donenfeld, 2018). IPSec varies depending on what implementation is used but according to a blog post by Salter (2018) an implementation of IPSec using Strongswan consists of 400.000 lines of code. This is however including the cryptographic primitives.

VPN Solution	Market share	Platform Availability	Lines of code	TCP/UDP
L2TP/IPSec	High	Linux, Windows, macOS	~400.000	Both
OpenVPN	Highest	Linux, Windows, macOS	~70.000	Both
WireGuard	Very low	Linux, Windows, macOS	~4000	UDP only

Table 1 – VPN solution information

2.2.1 OpenVPN

OpenVPN has become the *de facto* standard in VPNs today with more than 50 million downloads since its release in 2001 (OpenVPN, 2019). It uses SSL/TLS for key exchange and encryption. OpenVPN is open source, it is secure by the extensive scrutiny it gets by being accessible to anyone who wishes to review the code. In 2017 an independent review of OpenVPN was performed by Cryptography Engineering (Hopkins and Green, 2019), the results found no major vulnerabilities. It supports both TCP and UDP but defaults at UDP. UDP is faster but does not perform error correction as TCP. OpenVPN is fully functional on Windows, macOS and Linux. There is plenty of ciphers and encryption methods to choose from.

2.2.2 IPSec with L2TP

In 1995, a security group part of the Internet Engineering Task Force (IETF) standardized IPSec in RFC-1825 ("*IP Security Protocol* (IPSec) (1995) – RFC 1825) as a part of the IPv4 suite and is an open standard.

IPSec is a protocol suite, which means its several protocols working together (Cisco support, 2019). The tunneling for the VPN process is done by *L2TP* (Layer 2 Tunneling Protocol) by encapsulating the payload from one point to another. IPSec has the option to add security in a similar way as most other VPN solutions by encrypting and negotiating keys between the points in the tunnel. The main actions that those protocols perform are on data packets. The outcome of this is that now the IP payload has additional security in form of encryption (Ettl, 2004).

One drawback is that setting up and maintaining a VPN using this method is the complexity. IPSec involves certificates and pre-shared private keys. Ferguson and Schneier (2003) conclude that the major drawbacks of IPSec was the complexity of the protocol suite. But it was still better than the alternatives in their research. This is confirmed by Kotuliak et al. (2011). Additionally, the L2TP is as all VPNs very process heavy due to the encryption and decryption involved in this method (Agrawal et al. 2012). Similar to OpenVPN the IPSec protocol is fully available on Windows, macOS and Linux.

2.2.3 WireGuard

WireGuard is the name of a new VPN that aims to replace two of the most widely used VPN solutions, namely OpenVPN and IPSec (Donenfeld, 2018). It claims to be more useful than IPSec while avoiding the complexity of it and by having better performance than OpenVPN. It was originally written for Linux systems but is now available across more platforms. It is also open source like OpenVPN and IPSec, so it also benefits of being open to view for anyone with the knowledge to audit it. It is a design goal to have an overall straight forward configuration like SSH, i.e asymmetric key cryptography.

WireGuard also uses state-of-the-art cryptographic algorithms and protocols such as NOISE, BLAKE2 and Curve25519. All this while only being under 4000 lines of code at the time of the whitepaper release (this is excluding cryptographic primitives). According to the official white paper, Wireguard is currently UDP only (Donenfeld, 2018).

WireGuard has been sent out for review to be added in the Linux kernel (currently V. 5.0.8 as of writing this). One of the reasons why it was not added before has been addressed by the WireGuard developers. The Linux kernel developer Linus Thorvalds has been praising the quality of WireGuard in the Linux kernel mailing list (2018). That could mean that there is a chance that it will be part of the Linux Kernel soon. However, WireGuard is very young, it was announced as a pre-release in 2018 but already its beginning to draw much positive attention, this could imply that this novel VPN will be true to its claims that it is "faster, simpler and leaner" than the other VPN solutions.

2.3 Other VPN solutions

There are other VPN protocols identified in the literature that showed that the majority of VPN services uses one or both of IPSec and OpenVPN. There are some protocols however that are used extensively but were excluded, such as PPTP and SSTP.

PPTP (Point to Point Tunneling Protocol) is excluded because of the security issues it has and Microsoft recommends to use another technology for tunneling ("Microsoft Security Advisory 2743314", 2017). It only uses 128-bit encryption, which is not secure enough today. Even though the number of bits is not the only decisive factor of encryption security. Additionally the authentication methods have been proven to be vulnerable (Robinson, n.d.).

SSTP (Secure Socket Tunneling Protocol) is similar to OpenVPN since they both use SSL but OpenVPN is open source and SSTP is solely owned by Microsoft. Lawas et al. (2016) evaluated the performance of SSTP and Internet Key Exchange Version 2 (IKEv2). IKEv2 is a protocol to establish tunnels similarly as L2TP (both are built into IPSec). The VPN server was running Windows Server 2012 and the VPN clients Windows 8.1. In this research they used a tool called Distributed Internet Traffic Generator (D-ITG) to generate traffic and measure the throughput, jitter and delay. The authors concluded that IKEv2 performed better than SSTP in all cases and recommended future work to expand on VPN solutions and OSs to evaluate.

OpenSSH (Open Secure Shell) is a VPN solution that was excluded due to the low market share. OpenVPN was tested by Coonjah et al. (2015) against an OpenSSH VPN solution. OpenSSH is an open source version of SSH (Secure Shell) and OpenSSH VPN is a tunnel created in SSH. SSH is a protocol to access and operate network services securely over an unsecure network such as the Internet. It is secured with asymmetric cryptography (public and private key). In this study the authors used iPerf to measure the metrics (throughput, jitter and delay) on a full Linux testbed at a real company between different sites. The conclusion of this research was that OpenSSH provided better link utilization over OpenVPN in terms of speed. Meaning that by using the same infrastructure the OpenSSH VPN had higher throughput than OpenVPN in that specific study. The study on the top 15 VPNs by Khan et al. (2018) showed that none of the most popular VPN providers offer OpenSSH.

2.4 Network performance testing

The performance of VPN solutions can be measured in different ways. Two aspects are, what metrics to measure and what tools to measure it with. A study by Nawej & Du (2018) recommends three metrics that are described in detail below. The tool was chosen because it is, according to Arevalo-Cordero et al. (2019) a widely used tool for network measurement and performance tuning, this is confirmed by the usage of the tool in other sources cited in this study. The metrics used when testing networks are: *throughput*, *latency* and *packet loss* and the tool for measuring them is *iPerf*. These concepts are explained briefly below.

Throughput is how much data is sent from one point to another during a certain time frame. Throughput is usually measured in bits per second (bps). It is affected by the whole infrastructure of the channel such as the physical medium (cabling) and computational power among other factors.

Latency is defined as the time it takes to transmit a packet in one direction, for example client to server. In VPN testing latency is in a time value, usually measured in seconds milliseconds (ms).

Packet loss is a metric that refers to how many packets are "lost" i.e did not arrive from source to destination. This can be caused due to congestion on the network for example. It is measured as a percentage of packet lost with respect to packets sent.

Unreliability handling is a description of how the network handles disturbances. These disturbances may be artificial or natural such as high latency or packet loss. It can be quantifiable by limiting of the aforementioned metrics. While many people and companies are getting more aware of online privacy, it is not possible to have stable Internet connection constantly since users are more mobile now (Khan et al. 2018) with the usage of laptops rising and for example smartphones and other ultraportable network connected devices. This puts the company data at risk, because it is being used outside of the premises. A secure VPN is a must in any remote workers toolbox. The remote workers are possibly working on public unreliable WIFI's or on a train that might likely have unreliable Internet due to the fact that it moves, which is why it is important to test how state-of-the-art VPNs handle an unreliable network. The unreliability of a network could be a fluctuating connection such as cellular on a moving train that is traveling far from cities that have good infrastructure or traveling through mountains. It could also be a remote worker, who is working very far away from the office while requiring access to files across the world. This would add delay that is way above the usual packet transfer delay that one might expect while working with files that are on a server in the same building. Therefore it is important to investigate how the current VPNs compare on unreliable networks.

iPerf was identified in related research as the tool to test the performance of VPNs. In a research by Narayan, Brooking and de Vere (2009), the authors used iPerf to measure the throughput on three different VPN solutions. The VPNs used in this research are PPTP, SSL (Secure Sockets Layer) and IPSec. While the study is outdated by now, the measure metrics are still valid for evaluation today. That research concluded that the network performance does indeed depend on what OS, VPN solution and encryption algorithm is chosen.

Network performance is expected to decrease when utilizing a VPN due to several reasons. One reason is the physical distance between the client and server. Another reason is which type of encryption is used in the VPN configuration. More secure and heavier encryption algorithms can use the full CPU power of a VPN server and that leaves less computational resources for the network to use and therefor the network performance decreases.

3 Problem description

Imagine a network administrator is tasked to implement a VPN, there are several questions that arises, among them; which VPN solution is the best for their case i.e what are the needs for that particular situation.

Some organizations may require only the cheapest solution and others, for example the fastest. For organizations that will use VPN in less than perfect conditions, this study aims to help the system or network administrator to solve the problem with how to chose the best VPN solution, where the network connection is unreliable and identify the best VPN for that case.

3.1 Research question

How does the performance differ between state-of-the-art VPN solutions under stable versus unreliable network conditions?

3.2 Motivation

While the remote workforce is increasing, studies on remote workers show that there are plenty of attack vectors for such workers. One of the best ways to protect yourself against a common type of threats is to use a VPN according to Diab, Tohme & Bassil (2008). Sometimes the only way to access private networks is using a VPN. But for remote workers or general VPN users that are connecting from far away to their VPN server and users that are on unreliable networks, the VPN adds additional constraints on their connection. A study by Maskey, Horsmanheimo & Tuomimaki (2014) shows that remote workers that connect to the organization with a roadrunner setup (meaning through a mobile internet connection) suffer from increased latency due to the networks being congested during busy hours (which is the common working hours; approximately 07:00 to 17:00).

This motivates the study to identify how different VPN implementations perform under varying network conditions.

4 Methodology

This section describes in greater detail the steps taken to fulfill the objectives. Step one is tied to a literature analysis to identify the necessary information about how to solve the research question. The validity and ethical aspects of the chosen methodologies are also covered at the end of this section.

4.1 Following the steps

Steps to answer the research question are:

1. Identify what VPN solutions to experiment on, how to control traffic, what metrics to use, with which tools and what data to collect

2. Test the network in the experimental setup without any VPN solution to identify baseline performance

3. Configure and test VPN Solutions on three different operating systems with the network unreliability conditions

4. Analyze and compare the results to see if there is any indication of performance difference between the VPN solutions

4.1.1 Identify

The three VPNs that were identified as suitable solutions to test are OpenVPN, WireGuard and IPSec/L2TP. A very important fact is that the configurations of the VPN solutions are default For all VPN solutions, default configuration settings were kept as far as possible, rather than unifying the settings in respect to, for example, network protocols, cryptographic algorithms, or choices for compression. The decision to not touch the default settings was motivated by the assumption that the VPN solutions' developers would be most qualified to provide sane (secure) settings for their own VPN solutions. Default settings are used in all cases of the VPN tests. Each case is defined as one instance of testing, for example IPSec testing on Windows with Unreliability #1 – delay is one case of testing.

Furthermore as WireGuard is designed to be simple to deploy and use, it has fewer configuration options than IPSec and OpenVPN.

OpenVPN is deployed with completely default settings as that is recommended by the developers and that is enough for this experiment.

The *IPSec* implementation is also deployed with default settings, but from a third hand created script. This script and its configuration settings is described more in the next chapter where a more detailed description of the experimental design is presented.

The *WireGuard* implementation is also deployed with default settings to make the results valid as no other VPN has any configuration than the default, excluding of course the connection settings, which need to be configured for the VPN to work.

Based on the analysis in the background section a collection of relevant software and hardware that was needed for the experiment were identified. The analysis also presented what metrics to use in the testing. The lab set up was chosen to be as simple as possible but still having one server and one client per OS and to be able to control and monitor the network traffic. This was done to minimize outside impact on the network and to have as valid results as possible.

Network metrics that were identified as important to test from the related works from the background chapter while examining VPNs are:

- Throughput – *Megabits per second* (Mbps)
- 400 msec Delay (part 1 of unreliability)
- 1% Packet loss (part 2 of unreliability)

These metrics are also part of the variables in this experiment. Wohlin et al. (2012) states that there are two different variable types in an experiment, independent and dependent variables. The dependent variable is the variable that is being tested on, in this study it is the throughput. The independent variables are the ones that are being manipulated and controlled, in this study the two unreliability aspects, delay and packet loss are the independent variables.

According to a Cisco Support document (Cisco.com, 2019) the maximum recommended delay to use VoIP is 400ms and that will serve as the testing point of the unreliable network condition #1 - delay.

The packet loss value of 1% was chosen because according to Polacky, Pocta & Jarina (2016), an acceptable level of packet loss has to be under 10% in VoIP calls but since VPN adds additional overhead and the retransmission caused during such a high packet loss amount, the results of testing with such a high percentage of packet loss ended in the VPN connection breaking completely. Therefore the packet loss percentage was lowered to 1% as anything above that broke the connection and there was no way of getting consistent results.

The tool used to measure network performance is iPerf3 (https://iperf.fr) version 3.1.3. The reason behind this is that it fulfills the need to test throughput and it contains the data transferred i.e it is possible to see how many packets did not arrive. Since iPerf is available for Windows, Linux and macOS, it is fitting for this experiment as it is being done on all three different OSs. iPerf has been used in similar studies to measure similar performance metrics as in a study by Byun et al. (2013) to analyze the broadband wireless performance in California.

4.1.2 Test the network in the experimental setup

The second step is to test the network in the experimental setup on all the clients without any VPN. The tests are done to establish a baseline result on how the introduction of the unreliability modifications to the network will affect a server-client connection without the VPN. This is done to have something to compare and contrast the VPN testing results to.

4.1.3 Configure and test the three VPN solutions

The testing was designed to run tests, one client at a time. One default iPerf test consists of 10 seconds continuos testing and reporting the throughput with 1 second intervals. How iPerf works is described by the developers as “iPerf works by writing an array of len bytes a number of times. Default is 128 KB for TCP” iPerf User Documentation (n.d) this means that it sends packets through the network with a window size of 128 KB, essentially the buffer of how much data is sent before it is acknowledged by the receiver. iPerf has a very simple architecture with one end being the server and the other end a client, any endpoint can be the server or the client. iPerf also runs in memory, so the disk is not involved at all when measuring with this tool and the data that is being sent is noise (random data). The default test that was used in this study is an upload of data from client to server. Each test was repeated 50 times. The test had the first 1-second interval removed from the data set to give it a chance to establish a proper connection, because the test started immediately after enabling the VPN.

To minimize the chance of interference, while testing one VPN all others were switched off, and both clients and server were restarted and left on standby for 10 minutes to allow for any kind of automatic updates to have a chance to run and the machines to stabilize.

To introduce the network unreliability conditions the chosen router needed to support traffic shaping. The software router pfSense comes with this option as default and is named Traffic Shaper. In the settings it is possible to add any amount of latency and packet loss. The unreliability testing was achieved by utilizing the traffic shaping tool on the pfSense router. The traffic shaper tool was configured to add 400msec delay as one unreliability and 1% packet loss as the other. Since all the data between the VPN clients and server is routed through the router, it applies the traffic shaping both ways. All clients were tested with all three different setups, namely; baseline test which had no unreliability added. Unreliability #1 is the 400msec delay option and unreliability #2 is the 1% packet loss option.

The two degredations, 1% packet loss and 400msec delay, are used as the limit because the focus is on VoIP, and VoIP is a normal use-case for remote workers and it is a network speed sensitive application to be used through VPNs or generally on the Internet. Contrasting to reading a page, delay and packet loss is not as impactful for that since it will just load the page, but streaming VoIP is more sensitive to it.

4.1.4 Analyze and compare

The final objective to be completed for the study to answer the research questions was to collect and process all this data that was collected and analyze it to reach a solid conclusion. In this objective the validity of the experiment will be handled as well to make sure the results are of high quality.

4.2 The Experiment

This section will describe the setup of the experiment in more detail. The network topology of the experiment can be seen in Figure 3. The machines were unplugged from the switch while not being tested to eliminate interference by for example DHCP requests. The experiment test-bed setup consists of one machine acting as a software router running pfSense. It has two NICs (Network Interface Controller) that can each handle speeds up to 1Gbs. Port 1 is connected to the VPN server and port 2 to a switch connecting the client network. A switch between the router and the clients was used in the setup for simplicity with testing, because the router only had two ports and would therefore have to be manually unplugged and plugged whenever another client was tested. If the switch introduced any delay it would have been the same for all clients and VPNs and thus the addition of it is negligible for this experiment.

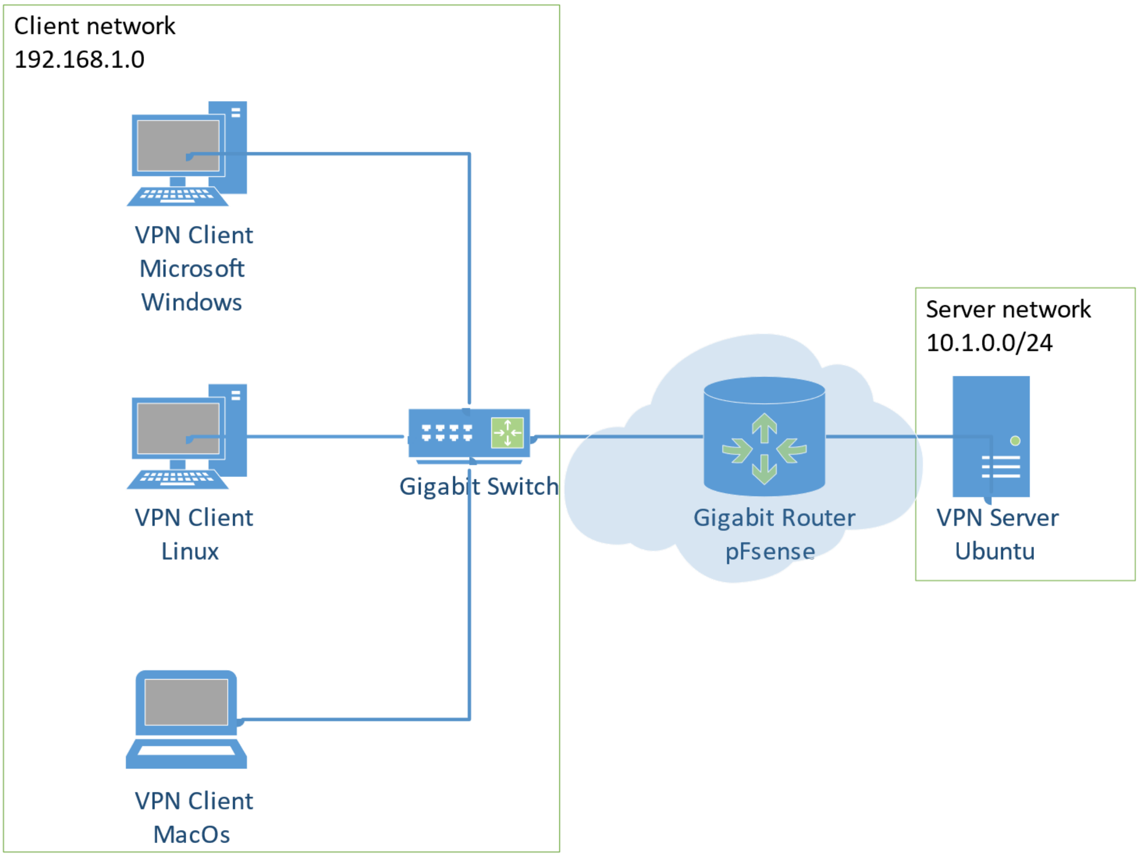


Figure 3 – The experiment test-bed setup

The clients were installed with their respective OS, one Linux Ubuntu, one macOS and one Windows 10. The measurement tool iPerf was installed on all clients and the server. Details of the commands to run the tool can be seen in the appendices B to D depending on what OS it was ran on.

The three VPN solutions OpenVPN, IPSec and WireGuard were installed on all three clients. They were all tested one after another without any unreliability modification first and the results recorded. When those tests were done, the unreliability options were introduced one by one and the results recorded. Windows and macOS natively support IPSec and were only configured to connect to the IPSec VPN server while Linux used LibreSwan (https://libreswan.org/). A table describing the default configuration of the three VPN implementations can be seen in Table 2. It describes important aspects of the VPN to be considered in this study. One of the aspects is the algorithm for the data encryption, the performance of a VPN is affected by what encryption is used. The second one is if compression of the payload is enabled, which by default is disabled on all the tested VPN solutions. Lastly multi-threading, this specifies if the encryption and decryption process is able to be done on multiple CPU cores. Todays computers often have more than one core, so this option gives the VPN the ability to perform parallel encryptions on multiple CPU cores. This would of course increase the performance of the VPN which utilizes this functionality, by encrypting and decrypting faster.

Default configuration	IPSec	WireGuard	OpenVPN
Algorithm for encrypting data	AES-256	ChaCha20	AES-256-GCM
Compression	Yes*	No	No
Multi-threading	Yes	Yes	No

Table 2 VPN Default configuration information
** Only Linux-to-Linux enables compression by default*

The IPSec implementation installed on the server in the experiment is configured with L2TP (Layer 2 Tunneling Protocol) for tunneling and IPSec for encryption. More specifically the tunneling protocol is xl2tpd (https://github.com/xelerance/xl2tpd/releases/tag/v1.3.14) which is a L2TP implementation that is maintained by Xelerance Corporation. The IPSec client is a LibreSwan IPSec implementation with pre-shared key (PSK), username and password. The whole installation was done with Lin Songs (hwdssl2) script (https://github.com/hwdsl2/setup-IPSec-vpn). To be consistent in the experiment, all VPNs were installed and tested with the default configuration. Unexpectedly the default configuration differed in performance between the OSs of the same VPN implementation.

The router in this experiment is a pfSense FreeBSD 11.2-RELEASE-p10 (pfSense 2.4.4-RELEASE-p3) software router. The reason I chose this router is due to the fact that it is possible to shape the traffic directly on the router using the tool that comes with pfSense called dummynet. With dummynet it is possible to add the unreliability aspects of the testing directly between the VPN nodes without having to use a dedicated tool or software on the nodes. Adding a software or a tool on the nodes would mean that it would take from the computational power to shape the traffic instead of the computational resources being available for the VPN. Using iPerf together with dummynet is proven to be reliable for network performance testing as it has been used for many years in researches such as Cole & Thain (2016) and Brassil et al., (2008) where both of these journal papers use the tools in a similar way as in this study to simulate bad network connection with dummynet and using iPerf to send data between nodes. A more detailed technical description of the router can be found in appendix A.

The configuration for Wireguard server is the default configuration that comes with the latest available version 0.0.20190702. More detailed information about all the machines and softare used in the experiment can be found in appendix A.

4.2.1 Testing

The setup is tested with a normal stable network connection and configuration with all network ports and cables in the experimental set up being capable of 1Gb speeds. The operating systems that are chosen in the experiment are Windows 10, Linux Ubuntu and macOS. The VPN solutions are IPSec/L2TP, OpenVPN and WireGuard. The tests are done with iPerf which is available on all three OSs. To test how the VPN solutions differ in unreliable network conditions, the network will be degraded by shaping the traffic to drop a certain percentage of packets or adding delay while performing an iPerf test to see how the VPN implementations handle the unreliability.

To keep the environment consistent between the different OSs the network is using the exact same cables and server settings and no other services are running. When testing the unreliability options on the same OS, the settings are the same, no other services are running and the same hardware is used. No inconcistencies are expected due to the fact that all VPNs are tested on the same OS and the same hardware with the same tool and configuration of the tool.

The tests started without any VPN then the VPN solutions were enabled for the tests and disabled when it was done as seen in Figure 4. This flow chart was repeated three times, one loop for No VPN, one loop for Unreliability #1 and one loop for Unreliability #2.

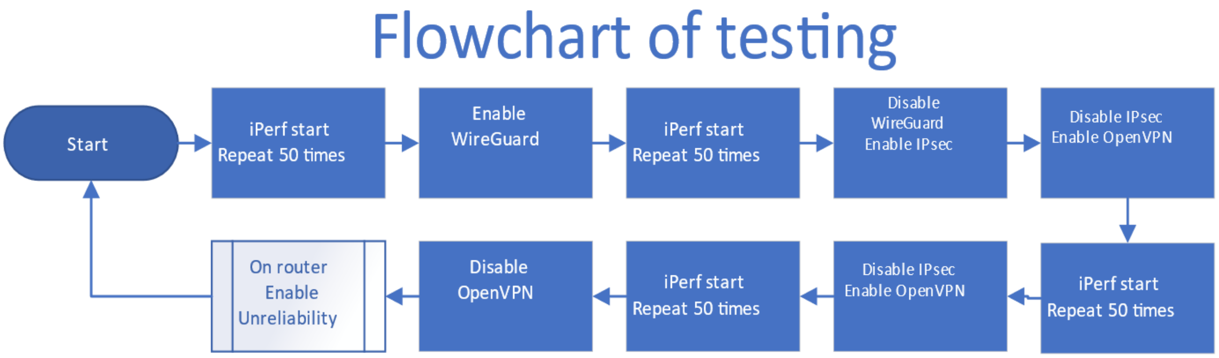


Figure 4 – Flowchart of testing

The automation is built-in with settings on pfSense for the unreliability. Linux tests are done with scripts and crontab. On the macOS, this is done with scripts and workflow (a built in tool for automation with macOS devices). Windows is done with scripts and the Windows task scheduler. For details on this see appendices B through D.

4.3 Data processing

This section explains how the data is processed, from the gathering point, to how its then filtered out and then used.

4.3.1 Data gathering

The data is gathered through iPerf. When running the tool all the results are displayed both while running and a summary of the tests after it is completed. This full test-time is saved into text documents with the use of scripts. All three operating systems and the three different VPN solutions handles the data gathered similarly but the scripts differ, see appendix B through D.

4.3.2 Data filtering

All the combined data is then filtered with scripts into a spreadsheet to retrieve a mean result of throughput of all VPNs on all the OSs. This is done in the same way for the baseline tests and with the two unreliability variables separately.

4.4 Validity threats

For an experiment like in this study there are three validity categories that need their respective threats handled. The three validity categories are internal, external and conclusion validity. Important to point out is that, as I tested only the VPNs default setting, the results may not apply if the different VPN solutions are configured with a goal in mind. A goal could be to configure the settings of a VPN to have higher throughput or higher security. If any changes are made to the VPN configuration, the results may differ from the results in this study.

In addition to validy threat, Wohlin et al. (2012) explains that there are biases to keep in mind when doing experiments . In this study two biases were identified and eliminated, namely Mono-method and Mono-operation bias. Mono-operation bias is eliminated by including more than one independent variable that is tested on. Mono-method bias is eliminated by using more than one test. In this study multiple tests were done for every testing scenario.

1. Internal validity threats

   Threats to the internal validity includes all the third-party software used. It is only a best-case scenario to assume that the measuring tools used worked exactly as they were meant to. This threat was handled by researching other works where these tools and software were used and attempt to use it in the same way and also to follow the official documentation of the tools and software used. Another step to ensure internal validity is to compare the results of this study with other related studies to see if the performance pointed to a direction that is expected.

2. External validity threats

   Threats to external validity exists in this study. If another study aims to recreate a similar set up there could be a difference in the results because operating systems and applications being tested in this study gets updated and with that, improvements are done as well and could affect the performance. One VPN solution that is showing bad throughput could have its performance increased with an update. In contrast, older systems could have different results as well as the OS or VPN solution had a different performance in previous patches.

3. Conclusion validity threats

   A threat to the conclusion validity is that this is a simulation of a real-world problem. The network unreliability is not real but simulated by the router and it is possible that when tested in a real scenario on real hardware and with real network unreliability that the results could be different. Two more threats to conclusion validity that is identified as possible in this study is low statistical power and unreliability of measures. They are both tied to the experiment and are handled by making sure that the tests are repeated many times and that any large measurement inconsistency is omitted in the results.

4.5 Ethical aspects

Ethical aspects are of importance in this subject area because it focuses on a process that regards cyber security and such has added sensitivity to ethical practice. Much of the software used in this study were Open Source or at least free of charge to use. This encourages anyone to build upon this research or to recreate it without having to be hindered by high costs. The usage of the information gathered in this study was not used for anything malicious nor is that recommended at any point. As no other people were involved during all stages of the study there is no anonymity issues to mention more than the related sources that are referenced throughout this study, all of the sources are obtained in a lawful way through public access records and libraries. The results are presented as honestly as possible without intent to deceive.

4.6 Alternative methods

One of the reasons an experiment was chosen as the methodology for this study is due to me having more experience in making experiments from courses I have done in the past. Also by excluding other research methodologies the only logical reason is an experiment.

A litterature analysis was not possible since no other same research has been found on this exact subject area, it is difficult to argue that the research question could have been answered by theoretical research only, without the actual experiment. Because WireGuard is so new there is not enough literature to make a literature study as a methodology to answer the research question of this study.

Interviews are not a suitable method for this project because the research question asks for specifics which are verifiable while interviews are mostly used to retrieve people’s opinions. Also it is hard to find a person to interview that is profecient in all the VPNs that were studied, especially WireGuard because it is so novel and constantly changing. Additionally I have no experience in conducting interviews so it would be an extra validity threat to consider thus making it a bad choice.

5 RESULTS

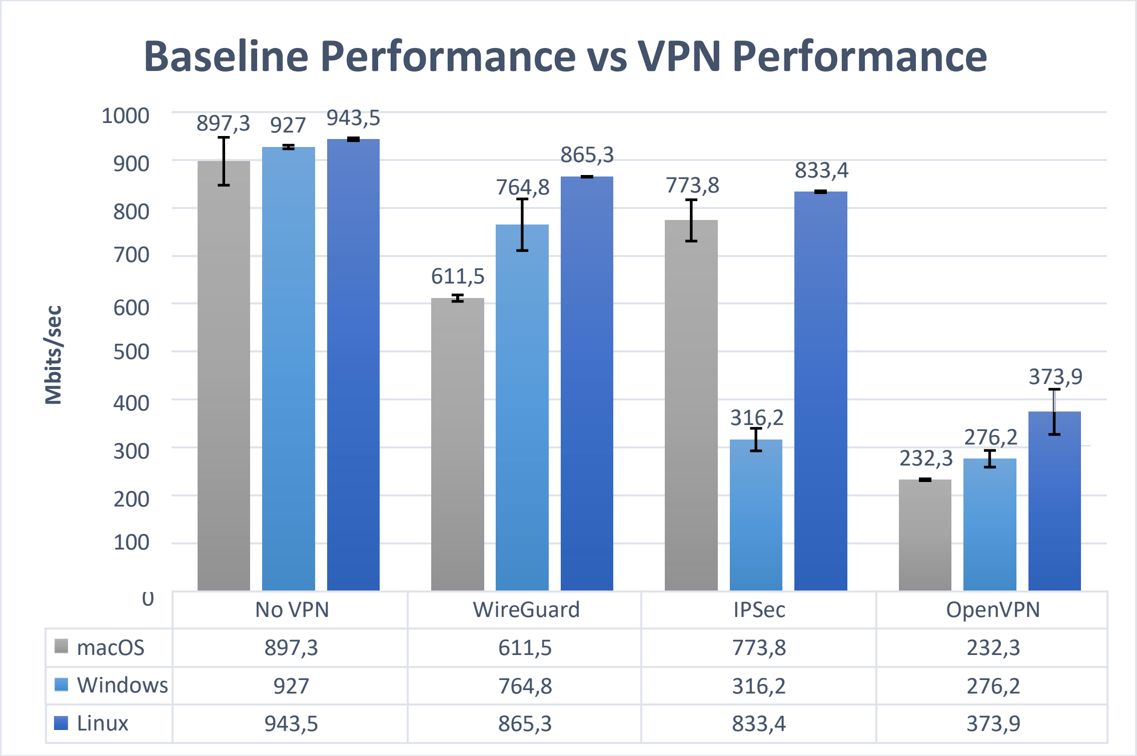
The results of the experiment are presented as recommended in the thesis book by Berndtsson et al. (2008). The results presented are values from when iPerf sends the packets to the server and server receives and presents the values. All the results presented are the mean values of the 50 tests per case. As discussed in the methodology section the tests are on 36 different cases. A table of all 36 individual cases can be seen in table 3.

	No VPN	WireGuard	OpenVPN	IPSec
	No network degradation	No network degradation	No network degradation	No network degradation
Windows	927	764.8	276.2	316.2
Linux	943.5	865.3	373.9	833.4
macOS	897.3	611.5	232.3	773.8
	Unreliability #1 - Delay	Unreliability #1 - Delay	Unreliability #1 - Delay	Unreliability #1 - Delay
Windows	3.9	3.6	1.2	0.7
Linux	48.4	48	4.2	50.6
macOS	5.6	15.8	26.6	22.2
	Unreliability #2 – Packet loss	Unreliability #2 – Packet loss	Unreliability #2 – Packet loss	Unreliability #2 – Packet loss
Windows	94.2	89.8	89.6	80.3
Linux	261.8	167.9	89.9	139
macOS	78.2	104.1	68.4	62.3

Table 3 - Test cases (all values are in Mbit/s)

5.1 Baseline vs VPN performance

The no VPN baseline performance is from testing the connection without any of the unreliability added or VPN enabled. The baseline VPN performance results are based on the performance tests done with iPerf on the VPNs using a reliable network, i.e no unreliability aspect enabled. The error bars in the result graphs is the deviation from the mean. Meaning that the top of the error bar is the furthest point from the mean towards higher throughput and the bottom of the error bar is the lowest throughput that was recorded in the testing for that particular case.



	No VPN	WireGuard	IPSec	OpenVPN
macOS	897,3	611,5	773,8	232,3
Windows	927	764,8	316,2	276,2
Linux	943,5	865,3	833,4	373,9

Figure 5a – Baseline vs VPN Performance

The results of the baseline performance show one obvious trend that OpenVPN is generally slower than the No VPN tests. The performance tests with No VPN were as expected faster than the tests with VPNs enabled. The fast throughput when using IPSec is due to the compression of data that is done with that VPN implementation. Similar results were presented in a study about the impact of compression on VPNs by McGregor & Lee (2002). The IPSec implementation used in this study has compression on by default for Linux and since we are using default values on all OSs, this explains the increase in throughput. Interestingly IPSec performed very poorly on Windows as compression was not enabled on that implementation by default. Important to note is that the result may differ because they use different encryption algorithms with the default settings.

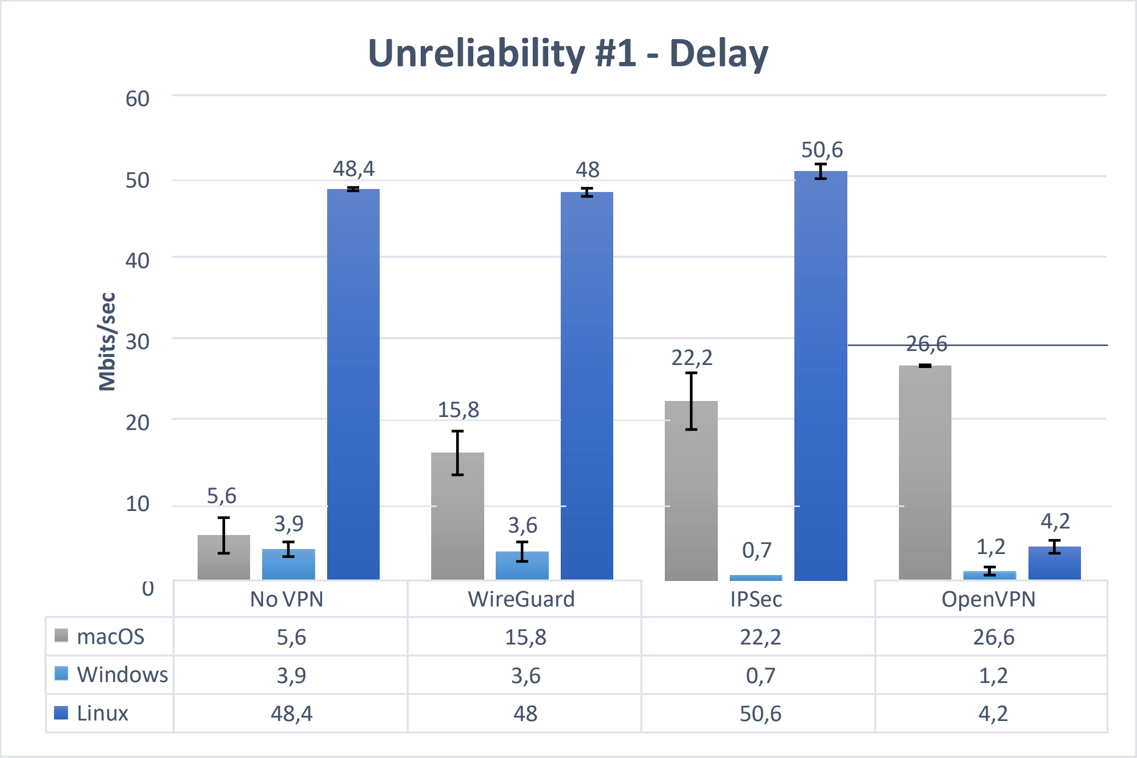
On macOS, the mean throughput without VPN was 897.3 Mbits/sec which is expected of a baseline with this type of lab set up. The worst performing VPN implementation on macOS was OpenVPN with a loss of 74% throughput compared to the no VPN baseline result. WireGuard lost 31% throughput and IPSec lost 13% throughput compared to the no VPN baseline.

Windows had the highest speed of all tests with the No VPN testing of 927 Mbits/sec. It also had OpenVPN as the worst performing VPN at 276.2 Mbits/sec. That is a loss of 70% which is pretty similar results to the macOS implementation of OpenVPN in this comparison. The best performer on Windows was WireGuard with only a 17% loss in throughput. IPSec even though natively supported in Windows performed poorly compared to the other OSs. It got outperformed by 659% compared to the No VPN test.

Linux performed overall very well in this test with only a loss of 8.3% with WireGuard and 11.6% with IPSec over no VPN.

5.2 Unreliability #1 – delay

The unreliability #1 – delay means that the traffic shaper on the router has been enabled to add 400msec delay on the experiment network. This was then tested without any VPN enabled and each of the VPNs one by one.



	No VPN	WireGuard	IPSec	OpenVPN
■ macOS	5,6	15,8	22,2	26,6
■ Windows	3,9	3,6	0,7	1,2
■ Linux	48,4	48	50,6	4,2

Figure 5b - No VPN and VPN Performance under Unreliability #1 - Delay

The results of the first unreliability variable show that Linux handle delays well in all cases except OpenVPN and IPSec. Similarly macOS also performs comparatively well, especially with OpenVPN under delay. Windows really falls behind with the worst results in all scenarios during the delay tests.

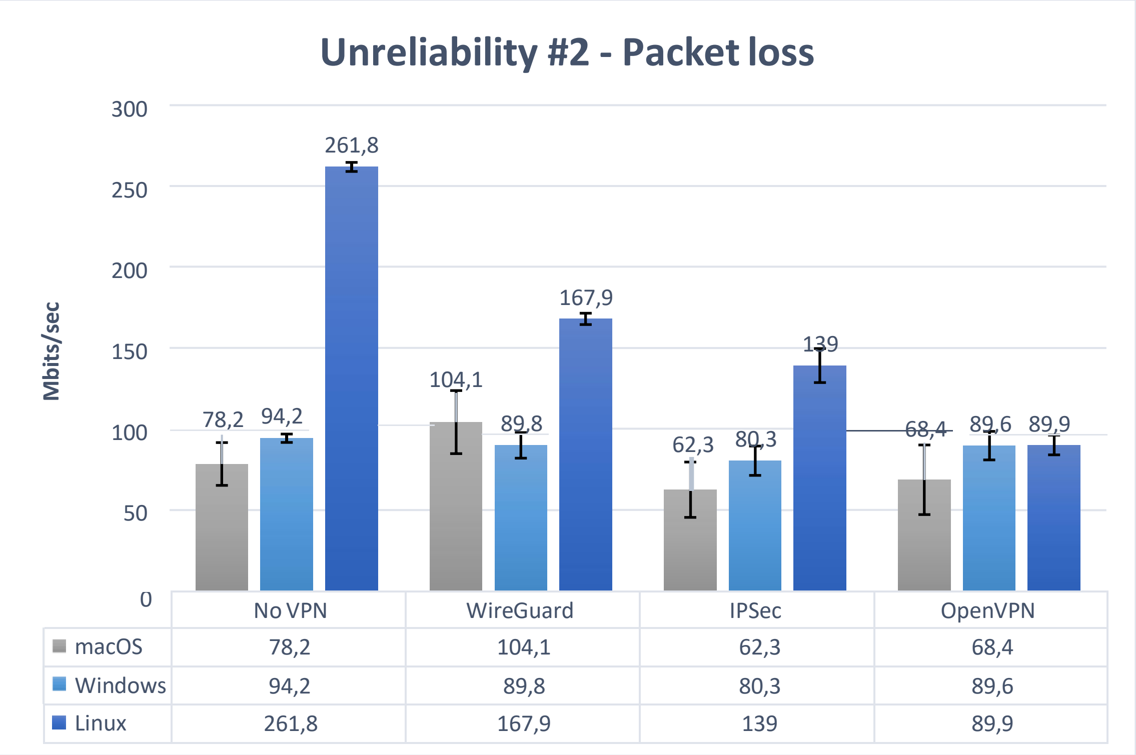
The macOS throughput with this unreliability resulted in a mean of 5.6 Mbits/sec without any VPN and performed better with any VPN than without. OpenVPN on macOS increased the throughput by a considerable 375%. IPSec increased with 296% and WireGuard increased it with 182%. The throughput on macOS showed a higher than average variance in the results in all cases except when using OpenVPN.

The results of Windows with the delay unreliability had a throughput of 3.9 Mbits/sec without any VPN and performed worse with all VPN solutions. WireGuard decreased throughput with 7%. OpenVPN decreased with 69% and IPSec decreased with 82%. The results are very poor on Windows overall with delay enabled. Since its consistent with all VPN solutions and the baseline, this may be because of the way the default network settings are set up on Windows.

The Linux results are by far the best with all VPN solutions except OpenVPN. Without any VPN the throughput was 48.4 Mbits/sec. With IPSec similarly as without any unreliability the throughput increased compared to the No VPN results. In this test it increased with 4%, also WireGuard decreased with 0.8%. OpenVPN on the other hand decreased the throughput in comparison with No VPN by 91% which is a considerable amount.

5.3 Unreliability #2 – packet loss

The unreliability #2 – packet loss aspect means that in a similar way as the delay aspect was enabled on the router in the experiment, but this time a 1% packet loss was added to the network connection.



	No VPN	WireGuard	IPSec	OpenVPN
■ macOS	78,2	104,1	62,3	68,4
■ Windows	94,2	89,8	80,3	89,6
■ Linux	261,8	167,9	139	89,9

Figure 5c – No VPN and VPN Performance under Unreliability #2 – Packet loss

The results from the packet loss unreliability tests show similar results in the way that Linux is generally performing the best while macOS and Windows do not differ much as the error bars overlap in all tests between the two.

macOS mean result is 78.2 Mbits/sec in this test without any VPN. With WireGuard the throughput increased by 33%. With IPSec it decreased with 20% and with OpenVPN it also decreased with 12%.

Windows got 94.2 Mbits/sec without any VPN during the packet loss tests. WireGuard decreased that with 4%. IPSec decreased with 14% and OpenVPN decreased the throughput compared to No VPN with 4%.

Linux with packet loss achieved 261.8 Mbits/sec which is far better than macOS and Windows with the same unreliability factor. With WireGuard it lost 35%. With IPSec the decrease was 46% and OpenVPN decreased the throughput compared to No VPN on Linux with 65%.

6 DISCUSSION

In this study we examined three different VPN solutions on three different OSs in three different network reliability cases to find out how the performance differed. Based on the results, Linux together with IPSec is the best OS/VPN solution and it would be the best implementation to deploy when network is expected to have high latency or experience packet loss. Even though WireGuard is not yet officially released and audited, it produces results that match and sometimes outperform the well-established IPSec and OpenVPN solutions.

6.1 Detailed results discussion

To reiterate, the research question for this paper is:

“How does the performance differ between state-of-the-art VPN solutions under stable versus unreliable network conditions?”

A definitive conclusion is that all implementations have their own advantages and disadvantages. Some VPN solutions perform better on a certain operating system. The most impactful conclusions drawn by the results are presented below.

- The best performing VPN solution for macOS, if the network is reliable, was IPSec followed by WireGuard and worst performing was OpenVPN

  This is due to two reasons. First, IPSec is natively supported in macOS and therefore can be assumed to be well-integrated and optimized for this operating system. Second, due to the compression that is enabled by default on IPSec in macOS. WireGuard showed great performance in terms of throughput speeds overall in this study but is not up to par with IPSec on macOS. When exposing the network to a delay of 400msec OpenVPN performed slightly better but it is insignificant due to the results of IPSec having a bigger spread that overlapped the throughput speed of OpenVPn. 1% packet loss unreliability testing on macOS showed similar results as the delay unreliability tests. OpenVPN slightly outperforms IPSec but the results overlap. Therefore recommended overall best VPN with default settings for macOS in reliable and unreliable networks is IPSec.

- We can see that with any of the unreliability variables in effect, all VPNs and OSs had reduced throughput in comparison with no unreliability

  This is expected since the iPerf tests were done with TCP so the packets had to be resent when they did not arrive in the correct order caused by the network unreliability #1 – delay. Similarly when the packets had to be resent during network unreliability #2 – packet loss. This would result in lower throughput speeds due to spending time on resending packets.

- Linux is fastest with the baseline and no unreliability at 943.5Mbits/sec

  The results of the baseline tests showed that Linux had a highest mean result but macOS had a greater spread (see Figure 5a/b/c error bars) and it overlapped at the top end with the Windows and Linux performance. macOS had the lowest mean average result in the baseline tests. The results are very close however and the difference could be explained by the fact that the network protocols in the operating systems handles the window size of the iPerf tests better in this particular test.

- All VPNs except OpenVPN perform best in Linux during the delay unreliability when comparing the other OSs under delay

  An explanation for this is likely due to the network stack of Linux handling the delay better than Windows and macOS in this case. For example the TCP window scale is dynamic by default and could be the reason why Linux peform better, also TCP retransmission behaviour is different on the OSs which could explain the differences.

- All VPNs perform best in Linux while experiencing packet loss

  This point is related to the previous point and the explanation would be the same i.e differences in the OSs network-stack behaviour. Also important to note is that the results are heavily dependent on the tool (iPerf) and there is a possibility that it is better suited for one OS than the others.

- OpenVPN is not the top performer in any test

  This result is expected. The generally lower performance of OpenVPN compared to the other cases in the baseline tests could be explained by how OpenVPN utilizes single-threading and the other cases make use of multi-threading. All the related works on VPN throughput speeds that were referred to in this study show that OpenVPN is usually slower in terms of throughput in comparison to IPSec and other VPN solutions. Coonjah et al. (2015) compared OpenVPN, IPSec and other VPN solutions and confirms this result.

- Linux performs best in all unreliability tests except one, the unreliability #1 – delay

  As mentioned previously in this section, this could be due to variance in the network stacks of the OSs. Also it could indicate that the testing tool iPerf works differently on Linux than on the other OSs.

7 Conclusion

Based on the results a few recommendations for OS and VPN combination emerged when using the default configuration. They can be seen in Table 6, the VPN in bold/blue background is the overall recommended for the specific network situation as it had the best performance. The recommendation is the following:

- If deploying a VPN service on a reliable network connection in a Windows environment, then the recommendation based on the results in this study is to use WireGuard. If the environment is based on Linux it is similar to Windows recommended to use WireGuard. If the environment consists of macOS devices primarily, the recommendation is to make use of IPSec.

- If deploying VPN on a network connection where 200ms latency is expected, in a Windows environment, then the recommendation based on the results in this study is to use WireGuard. If the environment is based on Linux it is recommended to use IPSec. If the environment consists of macOS devices primarily, the recommendation is to make use of OpenVPN.

- If deploying VPN on a network connection where 1% packet loss is expected the result show that WireGuard performs best on all operating systems.

Network connection	Windows	Linux	macOS
Reliable	WireGuard	WireGuard	IPSec
Delay	WireGuard	IPSec	OpenVPN
Packet loss	WireGuard	WireGuard	WireGuard

Table 6 – General VPN Recommendations

7.1 Future work

In future tests it is recommended to test different hardware for the server, router and clients and also to measure the utilization of computational resources while performing tests. Different hardware could be stronger or weaker CPU. Also the VPN server could be hosted on another OS than Linux Ubuntu Server, for example Windows Server.

Future tests could also measure different values of the metrics chosen, for example higher/lower delay in the network unreliability testing. I would also strongly recommend experimenting on different VPN solutions and also other configurations of these VPNs. In this experiment the default values were used, it could be beneficial to implement different VPN solutions with same encryption ciphers so that the results are more conclusive which protocol or solution performs best in a given scenario.

This test could also be done in a real company on real hardware instead of a simulation for more accurate values of the performance as this test had no real Internet traffic between the server and client and that new setup could provide different results.

Also another metric that was measured in several of the related work was jitter. This would be beneficial to test as well but since my work focused on TCP the jitter was not an issue. Testing UDP in a similar way would be beneficial because there are cases where UDP is to prefer when using VPN.

As mentioned in the results section, the difference in the results could be because the default settings were used for the VPN solutions. That means that they use different encryption ciphers which is one factor that decides how high the throughput is. For future works it is strongly recommended to use the same cipher in all tests to level out the playing field.

Future work would benefit from configuring both server and client operating system to perform similarly in regard to the network stack and VPN solutions. Meaning that the TCP/IP stack would handle buffers and window sizes in the same way and as mentioned, that the VPN solutions use the same encryption methods. It would also be beneficial to perform the test by using iPerf and then adding another tool to verify the accuracy of the tool.

References

Agrawal, H., Dutta, Y., & Malik, S. (2012*). Performance Analysis of Offloading IPSec Processing to Hardware Based Accelerators.* 2012 International Symposium On Electronic System Design (ISED). doi: 10.1109/ised.2012.53

Arevalo-Cordero, S., Gallegos-Segovia, P., Vintimilla-Tapia, P., Bravo-Torres, J., Cedillo-Elias, E., & Larios-Rosillo, V. (2019). *Data traffic management in a hybrid cloud composed of Openstack and Azure*. 2019 IEEE Colombian Conference On Communications And Computing (COLCOM). doi: 10.1109/colcomcon.2019.8809158

Berndtsson, M., Hansson, J., Olsson, B. and Lundell, B. (2008). *Thesis Projects*. London: Springer London.

Brassil, J., McGeer, R., Rajagopalan, R., Bavier, A., Roberts, L., Mark, B., & Schwab, S. (2008). *Improving VPN performance over multiple access links*. 2008 5Th International Conference On Broadband Communications, Networks And Systems. doi: 10.1109/broadnets.2008.4769158

Brown, S. (1999). *Implementing virtual private networks.* New York: McGraw-Hill.

Byun, Y., Narayanan, S., Mott, S., Biba, K., Schwenkler, J., Osborn, R., & Morris, M. (2013). *Wireless Broadband Measurement in California.* 2013 10Th International Conference On Information Technology: New Generations. doi: 10.1109/itng.2013.85

Cisco. (2019). *Security and VPN - Support Documentation.* Retrieved from https://www.cisco.com/c/en/us/tech/security-vpn/index.html

Cole, J., & Thain, W. (2016). *A small network for modeling MPLS*. Southeastcon 2016. doi: 10.1109/secon.2016.7506760

Coonjah, I., Catherine, P. and Soyjaudah, K. (2015). *Performance evaluation and analysis of layer 3 tunneling between OpenSSH and OpenVPN in a wide area network environment.* 2015 International Conference on Computing, Communication and Security (ICCCS).

Diab, W., Tohme, S., & Bassil, C. (2008). *VPN Analysis and New Perspective for Securing Voice over VPN Networks.* Fourth International Conference On Networking And Services (Icns 2008). doi: 10.1109/icns.2008.8

Donenfeld, J. (2018). *WireGuard: Next Generation Kernel Network Tunnel* [Ebook] (1st ed.). Retrieved from http://www.wireguard.com/papers/wireguard .pdf.

Ettl, R. (2004). *Understanding and Configuring IPSec between Cisco Routers* [Ebook] (1st ed.). SANS Institute. Retrieved from https://www.sans.org/reading-room/whitepapers/vpns/understanding-configuring-IPSec-cisco-routers-1356

Ferguson, N., & Schneier, B. (2003). *A Cryptographic Evaluation of IPSec.* Retrieved from https://www.schneier.com/academic/paperfiles/paper-IPSec.pdf

Global Workplace Analytics (2017) *2017 State of Telecommuting in the U.S. Employee Workforce*. [PDF] Available at: https://cdn.thepennyhoarder.com/wp-content/uploads/2017/06/30140000/State_Of_Telecommuting_U.S._Employee_Workforce.pdf [Accessed 3 Aug. 2019].

Hopkins, J. and Green, M. (2019). *OpenVPN 2.4 Evaluation Summary and Report.* [online] Private Internet Access Blog. Available at: https://www.privateInternetaccess.com/blog/2017/05/openvpn-2-4-evaluation-summary-report/ [Accessed 16 Aug. 2019].

IETF *(*1995). *IP Security Protocol (IPSec)* – RFC 1825 *(.* Retrieved 11 July 2019, from https://datatracker.ietf.org/wg/IPSec/

iPerf (n.d). *iPerf User Documentation* Retrieved 11 July 2019, from https://iperf.fr/iperf-doc.php

Khan, M.T., DeBlasio, J., Voelker, G.M., Snoeren, A.C., Kanich, C., & Vallina-Rodriguez, N. (2018). *An Empirical Analysis of the Commercial VPN Ecosystem.* IMC.

Labrosse, A. (2014). *Performance comparison of Internet technologies in the context of in-home live video streaming (MSc).* KTH Royal Institute of Technology.

Lawas, J., Vivero, A. and Sharma, A. (2016). *Network performance evaluation of VPN protocols (SSTP and IKEv2).* 2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN).

Maskey, N., Horsmanheimo, S., & Tuomimaki, L. (2014). *Analysis of latency for cellular networks for smart grid in suburban area.* IEEE PES Innovative Smart Grid Technologies, Europe. doi: 10.1109/isgteurope.2014.7028750

McGregor, J., & Lee, R. (2002). Performance impact of data compression on virtual private network transactions. Proceedings 25Th Annual IEEE Conference On Local Computer Networks. LCN 2000. doi: 10.1109/lcn.2000.891091

Microsoft Security Advisory 2743314. (2017). Retrieved 11 September 2019, from https://docs.microsoft.com/en-us/security-updates/SecurityAdvisories/2012/2743314?redirectedfrom=MSDN

Nawej, C., & Du, S. (2018). Virtual Private Network's Impact on Network Performance. 2018 International Conference On Intelligent And Innovative Computing Applications (ICONIC). doi: 10.1109/iconic.2018.8601281

Narayan, S., Brooking, K. and de Vere, S. (2009). *Network Performance Analysis of VPN Protocols: An Empirical Comparison on Different Operating Systems.* 2009 International Conference on Networks Security, Wireless Communications and Trusted Computing.

OpenVPN. (n.d). *The New Cloudflare VPN: What It Is & Isn't.* [ONLINE] Available at: https://openvpn.net/what-is-cloudflare-vpn/ [Accessed 10 September 2019].

OpenVPN. (2019). *VPN Software & Solutions.* [ONLINE] Available at: https://openvpn.net/. [Accessed 10 April 2019].

Polacky, J., Pocta, P., & Jarina, R. (2016). Influence of packet loss on a speaker verification system over IP network. 2016 26Th International Conference Radioelektronika (RADIOELEKTRONIKA). doi: 10.1109/radioelek.2016.7477365

Robinson, H. *SANS - Information Security Resources.* Retrieved from https://www.sans.org/security-resources/malwarefaq/pptp-vpn [Accessed 10 May 2019].

Salter, J. (2018, 26 August) WireGuard VPN review: A new type of VPN offers serious advantages. [Blog post] Retrieved from https://arstechnica.com/gadgets/2018/08/wireguard-vpn-review-fast-connections-amaze-but-windows-support-needs-to-happen/ [Accessed 10 September 2019].

Thorvalds, L. (2018, August 02). [GIT] Networking [Linux Kernel Mailing List]. Retrieved from https://lists.openwall.net/netdev/2018/08/02/124 Linux Kernel Mailing List. [Accessed 14 May 2019].

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). *Experimentation in Software Engineering.* Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-29043-5

APPENDIX A – Hardware specifications

Machine information	Server	Router
OS	Ubuntu 18.04.2 LTS	FreeBSD 11.2-RELEASE-p10 (pfSense 2.4.4-RELEASE-p3)
CPU	AMD Opteron X3216	Intel Xeon E5620
RAM	8GB	16GB
IP	10.1.0.101/24	192.168.1.39/24
OpenVPN IP	10.8.0.2	
OpenVPN Version	2.4.4	
IPSec IP	10.9.0.2	
IPSec (LibreSwan) Version	3.27	
WireGuard IP	10.10.1.2	
WireGuard Version	0.0.20190702	

Machine information - Clients	Ubuntu	macOS	Windows
OS	Ubuntu 18.04.3 LTS	macOS Mojave 10.14	MS Windows 10 Pro V.1809
CPU	Intel i7-6700K	Intel i5-2,3GHz	Intel i7-6700K
RAM	16GB	8GB	16GB
IP	192.168.1.47	192.168.1.48	192.168.1.38/24
OpenVPN IP	10.8.0.3	10.8.0.4	10.8.0.1
OpenVPN Version	2.4.4	(Tunnelblick) 3.7.9	11.6.0.0
IPSec IP	10.9.0.3	10.9.0.4	10.9.0.1
WireGuard IP	10.10.1.3	10.10.1.4	10.10.1.1
WireGuard Version	0.0.20190701	0.0.20190610 (13)	0.0.17

Referance

www.google.com

www.chatgtp.com