DESIGN AND IMPLEMENTAION OF FIREWALL TO INSPECT TRAFFIC IN ENCRYPTED VPN TUNNELS

A Thesis in
Electrical Engineering

Presented to the Faculty of the University
of Missouri–Kansas City in partial fulfillment
of the requirements for the degree

MASTER OF SCIENCE

by

BHANU PRAKASH PANCHAKARLA

B.TECH., K.L UNIVERSITY, AP, India, 2014

Kansas City, Missouri
2019

© 2019

BHANU PRAKASH PANCHAKARLA

ALL RIGHTS RESERVED

DESIGN AND IMPLEMENTAION OF FIREWALL TO INSPECT TRAFFIC IN ENCRYPTED VPN TUNNELS

Bhanu Prakash Panchakarla Candidate for the Master of Science Degree

University of Missouri–Kansas City, 2019

ABSTRACT

For the past 30 years, security is the greatest factor in internet and even in intranet. Starting from world war 1, we are striving to improve the security and now in 2019, we are confident enough to send the data security privately in public lines. We have built various kinds of firewalls which can inspect traffic at layer 2, 3, 4 and 7 of OSI model. These firewalls are robust enough and always have high availability build inside in case of failover events. We have various kinds of advanced cyphers which can pack the data tight enough so that no one can see or modify them. We use internet as private lines for sending data to others with the help of VPN tunnels. This change in technology made our life easy and cost effective. This technology helps us to be geo-independent, platform independent and resource independent. However, in most of the situations, we need monitoring over network to prevent attacks on our network. If the traffic is completely encrypted with latest algorithms, it's not possible to monitor that. So, this thesis works presents a view and demonstration on how to monitor the traffic over encrypted tunnels and block it if necessary.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the School of Computing and Engineering, have examined a thesis titled "Design and Implementation of Firewall to Inspect Traffic in Encrypted VPN Tunnels," presented by Bhanu Prakash Panchakarla candidate for the Master of Science degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Deep Medhi, Ph.D., Committee Chair
Department of Computer Science & Electrical Engineering

Cory Beard, Ph.D.
Department of Computer Science & Electrical Engineering

Zhu Li, Ph.D.
Department of Computer Science & Electrical Engineering

CONTENTS

ILLUSTRATIONS

TABLES

Tables Page

ACKNOWLEDGEMENTS

I would like to thank my academic adviser Dr. Deep Medhi for his guidance and support during the entire thesis work.

I would like to thank my parents, friends and NETREL lab members for their support.

I would like to thank University of Missouri - Kansas City for providing me an opportunity to work on this thesis report.

CHAPTER 1

INTRODUCTION

For the past 30 years, security is the greatest factor in internet and even in intranet. Starting from world war 1, we are striving to improve the security and now in 2018, we are confident enough to send the data security privately in public lines. We have built various kinds of firewalls which can inspect traffic at layer 2, 3, 4 and 7 of OSI model. These firewalls are robust enough and always have high availability build inside in case of failover events. We have various kinds of advanced cyphers which can pack the data tight enough so that no one can see or modify them. We use internet as private lines for sending data to others with the help of VPN tunnels. This change in technology made our life easy and cost effective. This technology helps us to be geo-independent, platform independent and resource independent.

1.1 Motivation for the Work

For the question “Is Alice able to send data to Bob securely?”, the answer is yes. But to what extent Alice/Bob can keep it secure in this real world. Do you think the systems used by them are 100% under their control? No one can answer this question after recent “Wanna Cry” attack. The below story is the real-world scenario I faced at my work, and this gave motivation of the work.

Let’s listen to the story of Alice and Bob.

Alice: Hello Bob, did you got my latest cryptic picture sent last night? We have to present it today.

Bob: Yes Alice, I got it. It is secure as you sent it using Diffie Hellman key exchange algorithm.

(After 10 mins, Eve is presenting the same picture in the class.)

Alice: How can Eve get our presentation? Is our line secure?

Bob: Yes Alice, it is secure, we are using Diffie Hellman key exchange algorithm to transmit it. Let’s call Sniff, he is expert in analyzing our network traffic. He will find what’s happening in our network.

(After an hour, Sniff analyzed the entire traffic and made a report)

Sniff: Yah bob, I see a lot of traffic in your network. The entire traffic is encrypted, and I see your certificates in them.

Alice: Hey Sniff, can you decrypt and check what is going in that? I have the private keys you required.

Sniff: Yo man, you are using Diffie Hellman key exchange algorithm, it’s not possible decrypt it offline.

Alice: Let’s do it on online traffic. Can you do that?

Sniff: Is it possible? I haven’t seen such thing till now.

Moral of the story: having high level of security will troll you. Having low level of security will mole you. Leaving the traffic not analyzed will curse you.

1.2 Objective of the Work

The main objective of this thesis is to scan the online traffic passing via secured tunnels formed various protocols like SSL, TLS, VPN's etc. By scanning the traffic passing via these tunnels, we can block unauthorized and suspicious activities.

1.3 Summary of the Work

Below are the contributions of this work:

- We proposed a new type of firewall which can scan the traffic inside encrypted tunnels.
- These firewalls are built at Layer 2 and Layer 3 models of OSI.
- This work we compared traditional firewalls with the new proposed firewalls.
- We computed the roundtrip time, response time and speed test using these firewalls at various geo locations using GENI platform.

1.4 Organization

This thesis is organized as follows: Chapter 2 presents a literature survey on network security which deals with various layers of OSI, chapter 3 gives overview on type of firewalls, types of encryptions we have in practice, types of VPN's we use in real world and their implementation, OpenVPN, IPfilters and their functionality. Chapter 4 deals with proposed methodology which involves integration of encryption

and decryption with firewalls which leads to inspection of traffic in OpenVPN. Chapter 5 follows with simulation and results which are derived from proposed methodology which include roundtrip time, speed test, latency etc. Chapter 6 concludes this thesis and provides some future work and impartments to this work.

CHAPTER 2

LITERATURE SURVEY

In [1], authors compared throughput vs MTU of OpenVPN and IPSec tunnels by enabling and disabling AES-NI instruction set which is hardware based. By observing these results in [1], we can conclude that having hardware supported encryption instruction set will add advantage to increase the throughput of the appliance. Calculation of latencies caused by encryption and decryption is also an important factor, but, in [1] no where they mentioned about this. In [2], authors compared OpenVPN and OpenSSH interns of file transfer and throughput, but they forgot basic purpose of OpenVPN. OpenVPN is wholly used to secure our entire interested network traffic which include HTTP, HTTPS, FTP, TCP, UDP and SSH, but OpenSSH cannot full fill the needs of VPN.

In [5], authors evaluated the performance of OpenVPN on end user mobile platform as client and evaluated the performance of OpenVPN interns of throughput with various encryption algorithms. But, this client device has very limited resources when compared to computer in terms of memory, CPU and hardware instruction set described in [1]. Also, in [5] authors haven't evaluated the performance based on latencies caused encryption and decryption of traffic by OpenVPN which is performed in this thesis. In [6], authors compared the performance of OpenVPN server on home-based routers with various encryption standards and different transport layer protocols. These results laid foundation for measuring the performance of throughput and latencies in this work.

While studying about OpenVPN core structure, dismantling each of it and understanding its integration with OpenSSL is very important. For that, [3] and [4] helped me to understand the core implementation of OpenVPN and integration of TLS 1.2 with OpenSSL which is a core functionality of OpenVPN. In [7], [8], [9], [10] and [11] authors described various encryption standards and compared their performance with other encryption standards.

2.1 Encryption at Various OSI Layers

Open System Interconnection or OSI model is developed in 1984 by International Organization of Standardization for having a standard way of connection between multi vendors. It is a 7-layer structure in which each layer has its own significance. As per initial design, there is no encryption at each layer. But with current development and technology, each layer is secured with its own encryption protocols. Encryption at each layer is described below.

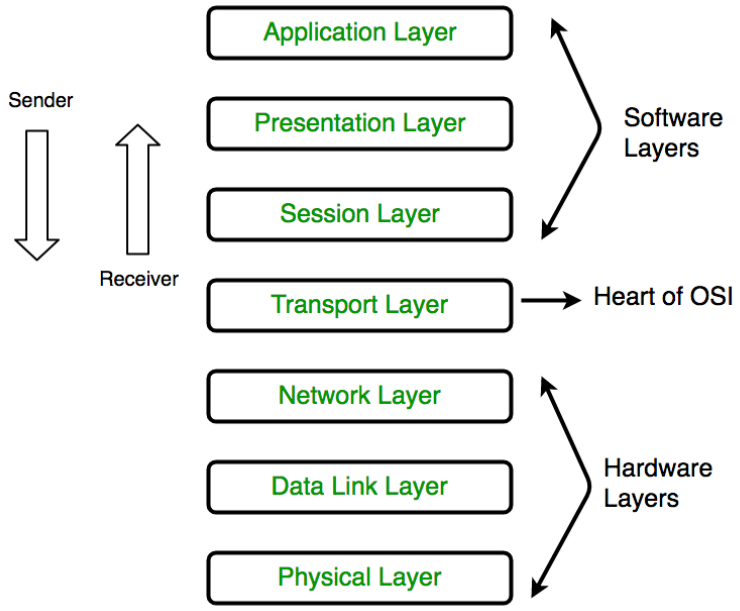


Fig 1 Basic OSI Layers

At layer 7, we have all our applications which can use their own protocols to protect the data. At layer 6, all general protocols at layer 7 can be encrypted with various encryption standards such as SSL, TLS, SSH, etc. At layer 1 and layer 2, there are encryption protocols such as 802.1x, WAP, WPA2, PAP, CHAP, ECAP etc., which will provide authentication, authorization and encryption. This thesis falls under encryption of layer 2 and 3 as OpenVPN can operate on both layers.

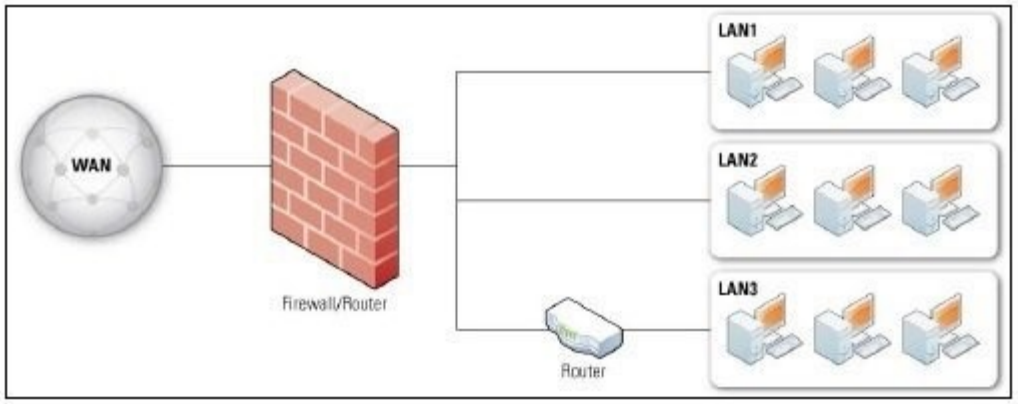
CHAPTER 3

FIREWALLS AND VPN

In this chapter, we are going to discuss on concepts of firewalls, VPN's and OpenVPN in-depth. In firewalls section, are going to explore what is a firewall, types of firewalls, encryption techniques and various models. In VPN section, we are going to see what a VPN is, types of VPN and types of VPN protocols. In OpenVPN section, we are going to discuss about OpenVPN basics, layer 3 and layer 4 protocols in OpenVPN, Virtual interface modes and architecture of OpenVPN.

3.1 Introduction to Firewalls

Firewall is a piece of software or hardware that allows only authorized access and blocks unauthorized to network elements or applications. Firewall can be implemented as a software or hardware or combination of both. Industrial standard firewalls are mostly combination of hardware and software which allows fast process of incoming traffic. Usually firewalls provide single block point, i.e., all the networks in an organization can be divided into blocks called as zones, traffic from one zone to another can be routed via same firewall and policies can be implemented according to as per access requests i.e., from internet to inside of an organization, traffic can be blocked and from inside to outside, it can be allowed.



Simple Routed Network with Firewall Device

Source: National Institute of Standards and Technology

http://csrc.nist.gov/publications/nistpubs/800-41-Rev1/sp800-41-rev1.pdf

Fig 2. Basic Firewall Architecture.

3.1.1 Classification of Firewalls

Usually firewall allows only specified traffic which can be inspected based on IP address, port status, target application, direction of flow of traffic or URL based. Remaining all traffic is considered as unauthorized and it will be blocked. As the remaining all traffic is blocked, it can prevent hackers from accessing the unauthorized data. More sophisticated firewalls can block the traffic based on session count also which is commonly known as screen count.

Based on design of firewall, they are categorized as 3 types by National Institute of Standards and Technology (NIST).

1. Packet filters: This will inspect only layer 3 and layer 4 of source and destinations. If we add rule for inbound traffic, we should add rule for outbound reverse also.

2. Stateful Inspection: This inspects traffic based on layer 3 and layer 4 of both source and destinations. On other hand, when there is inbound traffic, it will auto create rule for outbound reverse traffic.
3. Proxys: These firewalls will hide source or destination details based on configuration so that these details can't be revealed to outside.

Based on implementation of firewalls, they are classified as 4 types.

1. Network Layer firewalls: These firewalls will inspect traffic based on layer 3 and layer 4 only i.e., based on source and destination IP's/port details. It will not inspect the details on each packet.
2. Application layer firewalls: These firewalls rather than inspecting layer 3 or 4, they also inspect based on application, cyphers used etc., also.
3. Proxy firewalls: These will inspect traffic based on URL's, applications.
4. Unified threat Management: These firewalls will do deep packet inspection and will try to match the signatures with known threats.

All the above-mentioned firewalls are packed as single piece of hardware or software. Among these, vendors like Cisco, Juniper, Palo Alto, etc., but these are commercial. On other hand, we have some Linux distros which support these. Best example for Linux firewalls is PFSense, iptables, IPfilters, Uncomplicated firewalls (UFW) etc. These firewalls rather than simply inspecting the traffic, they also provide programmability. In this thesis, I use IPfilters package and the details are discussed in next chapter.

3.1.2 Encryption and Encryption standards

In modern days, sending a plain data over a network will expose the content to middle man. So, it can be sent over a cryptic manner. Cypher is nothing but a procedure or set of algorithms to perform encryption or decryption of data so that it can be sent over network to maintain integrity and confidentiality. When a cypher algorithm is applied to a plain text, the resultant encrypted text is called as cypher text.

In real world, there are various types of encryptions which provide various levels of security. These security features depend on source and destination compatibilities like system resources, network connectivity, memory allocations, etc. Depending on size of data being encrypted, encryptions are categorized in two ways.

1. Stream Ciphers: In these methods, input data is crypted in continuous stream passion.
2. Block ciphers: In these methods, input data is taken in form of blocks and encrypted

Based on type of key used for encryption and description, encryptions are again divided into two types.

1. Symmetric encryption: In these methods, same key is used to encrypt and decrypt the data on both source and destination sides. In symmetric ciphers, we must share the key beforehand between source and destination. If any third party knows the key, they will be able to intercept the data.
2. Asymmetric encryption: In these methods, we use different keys to encrypt and decrypt the data. For asymmetric data, it is not necessary to share the key

beforehand. So, these provide high security when compared to symmetric ciphers. We can use asymmetric ciphers to share symmetric cipher keys which maintain confidentiality in over internet

Even though cyphers are categorized as 2 basic ways above, based on standards and way of functionality, they are categorized as below which are mutually exclusive.

1. Encryption algorithm standards
2. Hashing Algorithms
3. Digital Signature standards
4. Public-key infrastructure Standards (PKI)
5. Wireless Standards

These algorithms define the way plain text to be encrypted or hashed and may use symmetric or asymmetric keys. The strength of the encryption depends on the size of the key used.

3.2 Introduction to VPN

VPN stands for Virtual Private Network allows users or clients to access the data encrypted through internet or intranet. VPN uses a concept called as tunnel or commonly known as VPN tunnel. Whenever user wants to use a VPN, they have to create a tunnel to the VPN server This tunnel can be encrypted various standards of cyphers and thus this data is secured.

3.2.1 Types of VPN

VPN's can be basically categorized as two types based on how user access the VPN which are explained below

3.2.1a Remote Access VPN

In this type of VPN, there will be only end users connected to VPN servers and access the services securely. These services may be like web data, files, API's etc. Examples for remote access VPN are HTTPS, SSH, SFTP, OPENVPN in tunnel mode, etc.

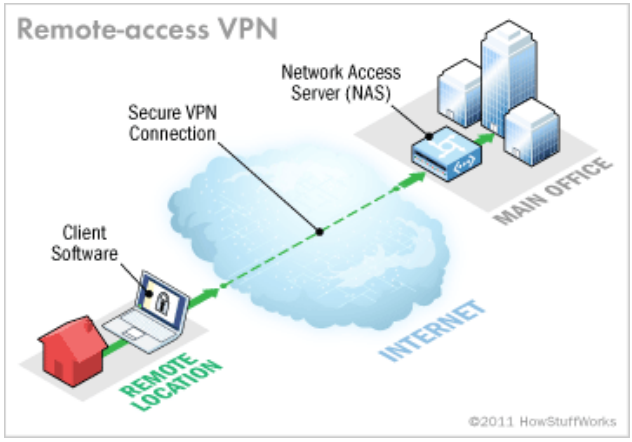


Fig 3. Remote-access VPN

Remote Access VPN is useful for business users as well as home users. Best example for this VPN usage is, business clients. When a business client is travelling or working from home, in order to access corporate resources securely with remote-access VPN. Best corporate software used are Cisco Any connect VPN, IPSec, PPTP, SSTP, etc.

Best example other that corporate users is home users using various kinds of VPN services like VPN unlimited, IPVANISH, NordVPN etc. These services provide remote access VPN and using these, end users will create an encrypted tunnel to the provider servers using public internet. After creating private tunnels, users will point

default gateway to the tunnel destination IP so that all the traffic will pass through the tunnel.

3.2.1b Site – to – Site VPN

In remote access VPN, tunnel is formed between end user host machine whereas, in sit-to-site VPN, they use virtual bridge and it is formed between 2 routers so that they will appear next to each other and they can exchange direct routes also. In site-to-site VPN's, we have intranet site-to-site VPN and extranet site-to-site VPN. Intranet site-to-site VPN means, VPN created between same corporate offices, whereas for extranet, VPN is created between 2 different corporates. These router nodes can be geographically separated, and they can use internet or private lines for communication.

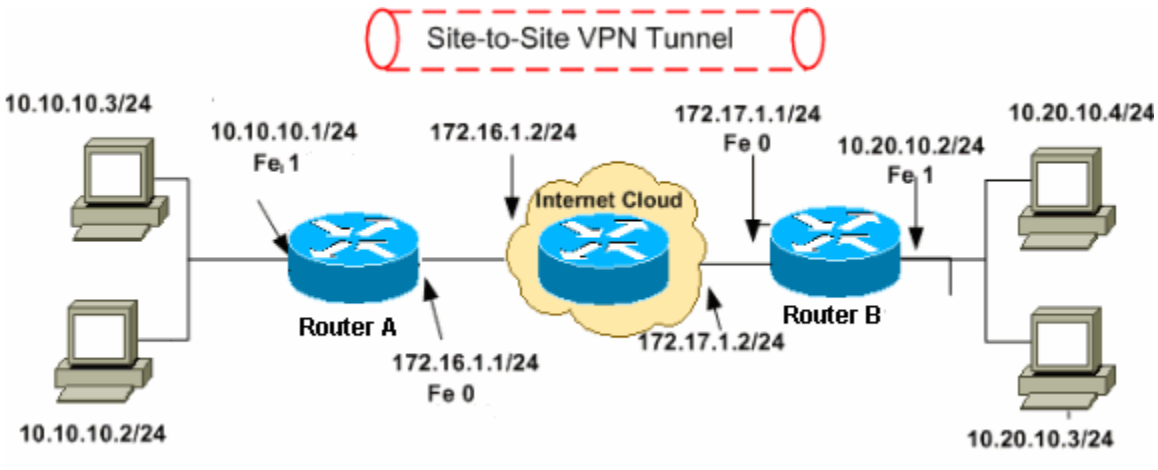


Fig 4. Site-to-Site VPN

Since Site-to-site VPN is based on Router-to-Router communication, in this VPN type one router acts as a VPN Client and another router as a VPN Server. The communication between the two routers starts only after an authentication is validated between the two.

3.2.2 Types of VPN Protocols

The above two VPN types are based on functionality of VPN. Based on encryption method and how they operate, there are various VPN protocols described below.

3.2.2a Internet Protocol Security or IPSec:

IPSec or Internet Protocol Security is used over layer 3 network and it uses authentication and encryption methods to keep the data secure. IPSec works in 2 modes as 1) Transparent mode and 2) Tunneling mode. Transparent mode is nothing but site-to-site mode and tunneling mode is remote access VPN. The authentication method could be via presaged key or cert based or both.

3.2.2b. Layer 2 Tunneling Protocol (L2TP):

L2TP or Layer 2 Tunneling Protocol is a tunneling protocol that is usually combined with another VPN security protocol like IPSec to create a highly secure VPN connection. L2TP creates a tunnel between two L2TP connection points and IPSec protocol encrypts the data and handles secure communication between the tunnel.

3.2.2c. Point – to – Point Tunneling Protocol (PPTP):

PPTP or Point-to-Point Tunneling Protocol creates a tunnel and encapsulates the data packet. It uses a Point-to-Point Protocol (PPP) to encrypt the data between the connection. PPTP is one of the most widely used VPN protocol and has been in use since the time of Windows 95. Apart from Windows, PPTP is also supported on Mac and Linux.

3.2.2d. Secure Sockets Layer (SSL) and Transport Layer Security (TLS):

SSL (Secure Sockets Layer) and TLS (Transport Layer Security) create a VPN connection where the web browser acts as the client and user access is restricted to specific applications instead of entire network. SSL and TLS protocol are most commonly used by online shopping websites and service providers. Web browsers switch to SSL with ease and with almost no action required from the user, since web browsers come integrated with SSL and TLS. SSL connections have https in the beginning of the URL instead of http.

3.2.2e. OpenVPN:

OpenVPN is an open source VPN that is useful for creating Point-to-Point and Site-to-Site connections. It uses a custom security protocol based on SSL and TLS protocol. More details will be explained in next chapter.

3.3 Introduction to OpenVPN

OpenVPN is an opensource software which implement VPN technology to create site to site or P2P connection between two locations or systems. This software uses OpenSSL Library to perform encryption and decryption and derives much of its crypto capabilities from this library. On a standard, it uses SSL/TLS for key exchange with its custom security protocols so that key will be secure while it is being transferred over internet.

3.3.1 Authentication Methods

OpenVPN allows end users to authenticate using certificates, username/password or pre-shared secret keys. It has the capabilities to enable multiple

users to use same credentials at the same time and provide the access. Certificate based authentication is the more secure and fastest whereas username/password authentication is easiest configuration available in the setup. We can also combine various authentication methods together to authenticate single session which increases security further. This means, we can combine certificate-based authentication with username/password to keep it more secure without compromising the system delays.

3.3.2 Layer 3, 4 Protocols

OpenVPN can operate on both IPv4 and IPv6 and can also interchange IPv4 to IPv6 from LAN to WAN if necessary. OpenVPN can be operated on both TCP and UDP with custom listening ports. However, port 1154 on TCP or UDP is the default port for this OpenVPN protocol. As we feel TCP is more reliable protocol over internet, here UDP is gives high performance over TCP as TCP adds lot of headers and acknowledgment packets which cause delays in communication. On other hand, to use TCP, we should have excessive bandwidth so that there will be less retransmissions. If not, the performance will fall, and this is called as “TCP meltdown problem”. To make UDP connections reliable, OpenVPN uses internal acknowledgments methods inbuilt data packets.

3.3.3 Virtual Interface Modes

OpenVPN can operate in two modes commonly known as tunneled mode and bridged mode. In tunnel mode, OpenVPN creates a layer 3 interface commonly known as tunnel which is associated with IP address. This IP address is obtained from DHCP server of OpenVPN server. The gateway for this tunnel is nothing but the tunnel on

OpenVPN server. By default, all the routes are pointed to this tunnel so that all the traffic will enter this interface. After traffic hitting this interface, OpenVPN software will encrypt this traffic and send it via normal physical interface. In bridged mode, OpenVPN will create a layer 2 interface called as bridge. All the layer 2 traffic is pointed to this interface so that all the traffic entering this interface will encrypted and sent to bridged interface of OpenVPN server via physical interface. The main difference between tunnel and bridged interface is, tunneled interface can't handle layer 2 frames directly. Whenever a layer 2 traffic hit tunnel interface, layer 2 info will be removed and then encrypted. The main advantage of bridged mode is, a local area network can be split between multiple geographical interfaces and these systems will consider they are next to each other on same switch. The main difference between private lines and OpenVPN is, private lines are too costly, whereas OpenVPN interfaces operate over internet by encrypting the data.

3.3.4 Architecture

OpenVPN server starts listening on the assigned TCP/UDP port. When a user tries to connect to initially client will send TLS header with client certificate and requests one from server side. So, server will send the server certificate and try negotiation for the cyphers suits so that they can stand on one. When cypher suite is selected, they will exchange their private and public keys which will be used to exchange the symmetric key for data encryption. When all the authentications are completed, server and client will two different channels known as control channel and data channel. Usually, this control channel is used for negotiating the key parameters

such as negotiation key for data encryption, tunnel MTU size, speed, data size sent, etc., whereas data channel is only used to send data in encrypted manner. To maintain integrity for both control channel and data channel, OpenVPN uses HMAC authentication so that if there is any tampering in control plane or data plane, it can be identified. Finally, these two channels are multiplexed and sent via physical interface.

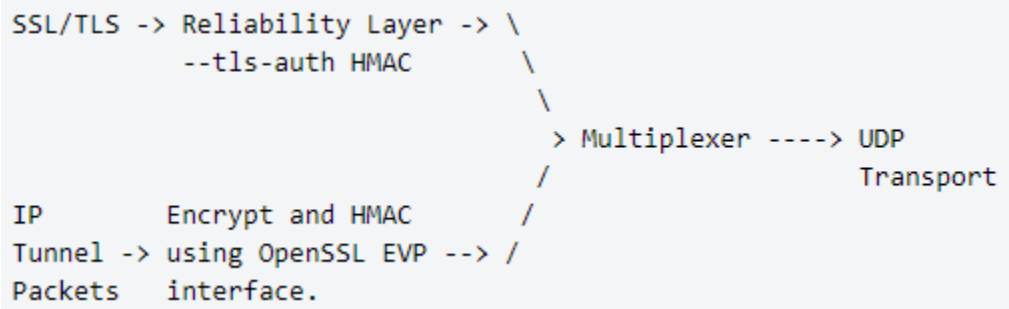


Fig 5. Multiplexing of two channels in OpenVPN

On other hand, OpenVPN server separates each client connection my maintaining a unique session ID. This session ID is visible in control plane. Whenever a client wants to retrieve the closed session, client will use this session ID to retrieve the connection which are stored in server database. A sample sniff of above conversation is shown in below Wireshark snip.

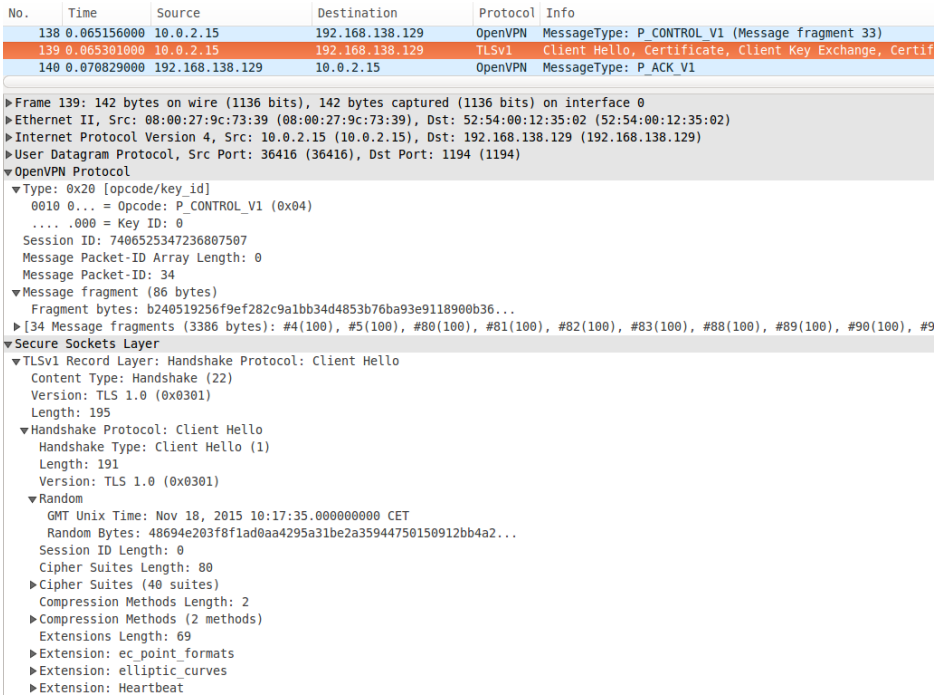


Fig 6. OpenVPN packets in Wireshark

CHAPTER 4

METHODOLOGY

4.1 Proposed Methodology

Usually all VPN traffic is encrypted over tunnel and it is not visible to middle man. The main aim of this thesis is to interpret traffic going over an encrypted VPN tunnel in the middle of path and block the traffic depending the requirement. There are various key exchange algorithms like RSA, CSC, YAK, Diffie-Hellman etc., which uses asymmetric keys structures to exchange symmetric keys for encryption. However, with some key exchanges like RSA we can post decrypt the traffic if we have traffic from starting point to end which includes key exchange data also. However, with some key exchange algorithms like Diffie Hellman, we can't decrypt the traffic post the session even if we sniffer data from start point to end. Also, it is not possible to interpret the traffic over live network if it is encrypted with this exchange algorithms. With this proposed mythology, we can decrypt and interpret the traffic on live wire.

In this method, we will install a custom firewall at the point of interest to intercept the VPN traffic. This custom firewall is designed to intercept VPN traffic. To intercept the OpenVPN traffic, it is tightly packed with OpenVPN package and NETFilters package. In Linux operating system, all the device level programs will be running in Kernel space of the operating system, so as the network interfaces. At Kernel level, network traffic needs to pass through certain blocks shown in below figure.

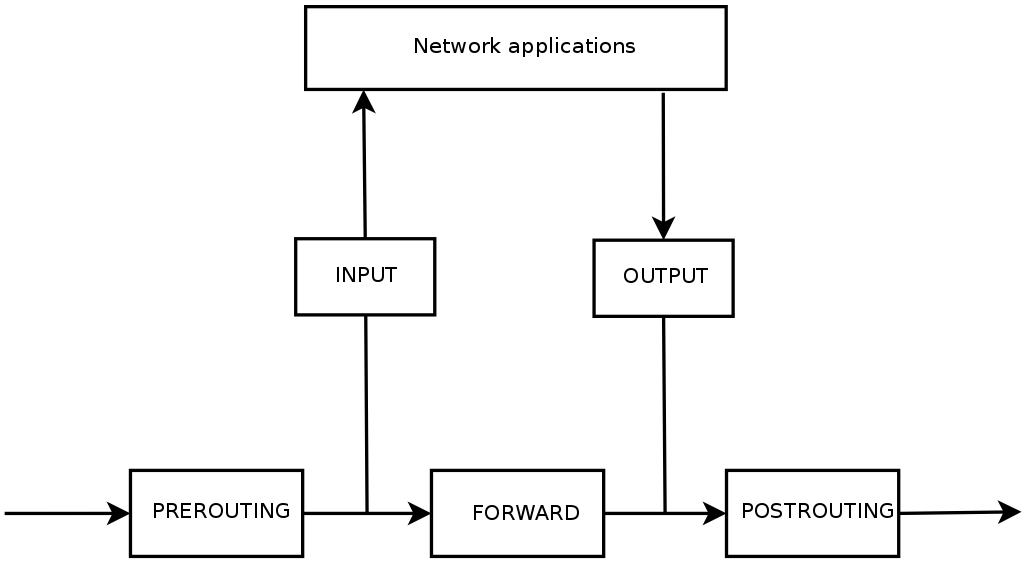


Fig 7 NetFilters Architecture in Linux Kernel

NetFilter has the capability to edit live traffic frames at each layer of above block and can reroute as necessary. So, in this firewall, I snipped the traffic at pre-routing block. If it matches the VPN source and destination, then I will take it out of kernel memory and send it to user space to do processing on that. After sending t to user space memory, it will be handled by different program which will interact with OpenVPN decryption libraries and pre-shared certificate database with credentials if necessary. When a packet reaches this user space, it will be analyzed and all session information like session ID, source IP, destination IP, syphers suits etc. and then it will generate its own public key exchange parameters to and send them to other ends looking like a middle man. All these are done using OpenVPN libraries itself which are called in custom C program. These libraries output is in turn redirected to two tunnel interfaces as required and iptables package will take care of filtering traffic here. Once the traffic is identified as trusted, it will be encrypted with openvpn libraries and sent to other tunnel. The below diagram describes with an example.

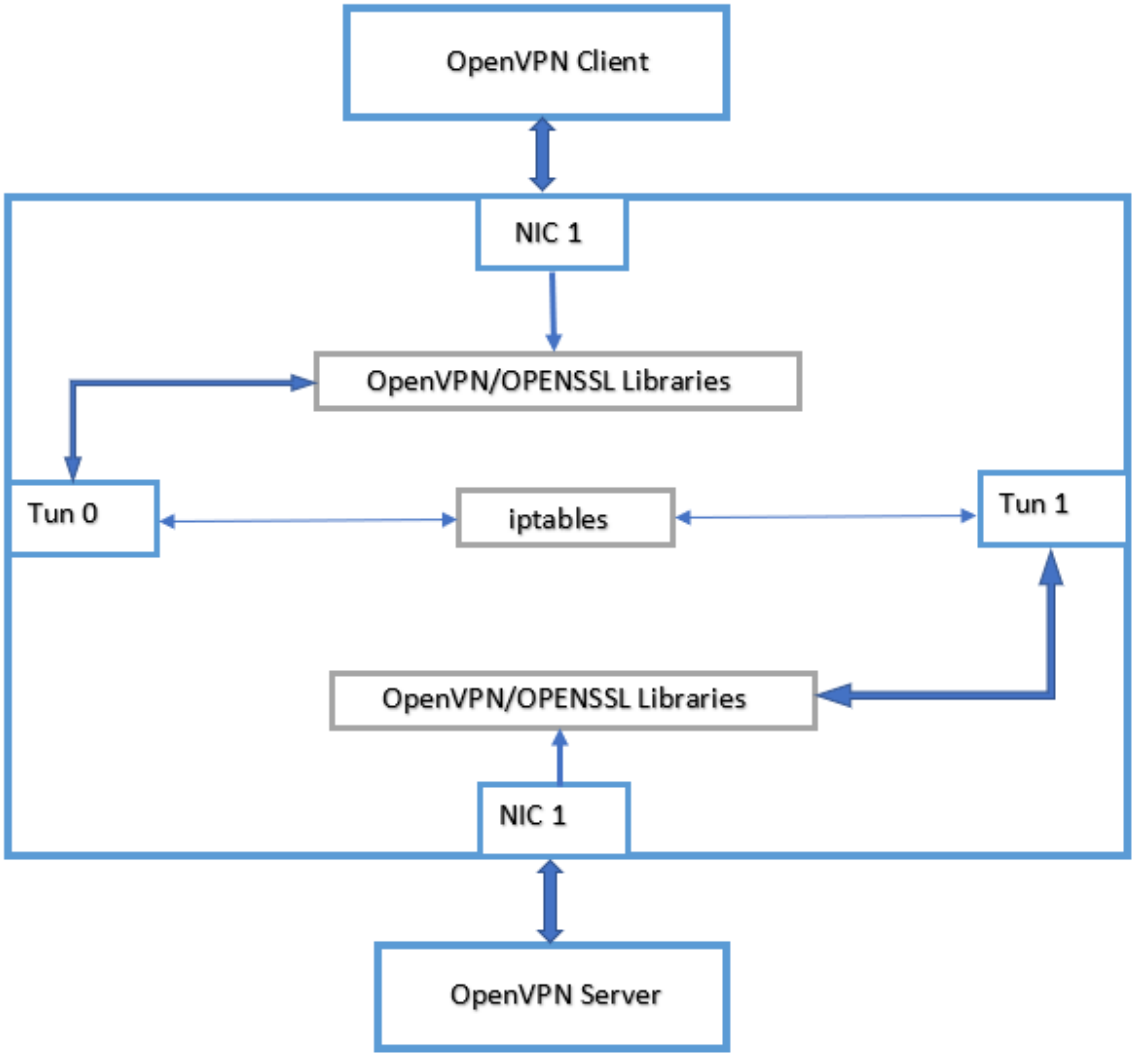


Fig 8. Proposed Methodology Architecture

The above diagram shows how the proposed methodology works with traffic flow. Whenever a client initiates a connection, as our firewall is in middle of the path, it will receive on one of its NIC say “NIC1”. Using NETFilters, this data is snipped at pre-routing table and then sent to OpenVPN/OpenSSL libraries. These libraries will act like an OpenVPN server and will use certificate of actual OpenVPN server database to decrypt the traffic and forward it to tunnel 0. From tunnel 0, they are routed to tunnel 1.

While this routing happens, as this traffic is decrypted, iptables will inspect this traffic and take necessary action like drop or allow. On this traffic is allowed by iptables, it will reach tunnel 1 and then it is again taken care by OpenSSL/OpenVPN libraries to encrypt it same certificate that actual server uses and transmit it to another interface where OpenVPN server is connected.

There are various key exchange algorithms like RSA which will allow decryption of data when we perform packet capture from start point to end point of session. On other hand, we have other key exchange algorithms like Diffie-Hellman algorithms which will not allow decryption of traffic even we have post encrypted data from start point to end point. This firewall setup allows us to analyze the encrypted traffic even if it is encrypted with such hard encryption techniques and can store decrypted data for post analysis.

4.2 Expected Results

Expected results for this work are mainly throughput from client to server and latencies from end to end. In order to measure throughput, I used “wget” package in Linux and custom socket programming to measure the total bytes sent per second. Here, the server that host the file is nothing but the OpenVPN server. Also, I varied the MTU size of the tunnel interfaces on client and server side and measured the throughput. These results show how MTU affects the throughput while data is getting encrypted and decrypted.

We have different components in this setup like client, gateway, firewall, encryption na decryption components in firewall, iptables component in the firewall,

OpenVPN server and internet. All these layers add their own latencies. So, measuring the latencies at each component is very deficult. For doing it, I used “TCPDUMP” tool on Linux and snipped the traffic on each interface of each device which includes tunnel interfaces on proposed firewall. When a packet passes from one component to other, it is recorded in sniffer with the timestamp. By comparing the time stamps, we can measure the latencies caused at each layer.

4.3 GENI Test Bed

For this thesis work, GENI Portal played an important role. I used GENI environment to create all the systems required at 10 different locations all over the United States. All these systems are allocated with same RAM and number of cores, but the processor type is different depending on location and allocation. All these locations are having at least 1Gbps of shared internet connection. As this is shared internet connection with other resources, the network speed of all the systems is limited to 100Mbps intentionally on the network interface cards of all allocated systems. The architecture of each location in GENI is shown in below diagram.

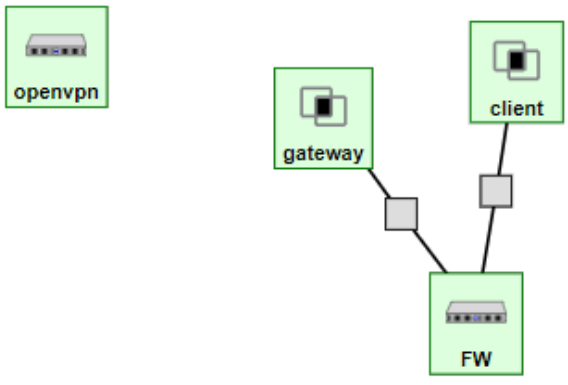


Fig 9. A view in GENI slice

CHAPTER 5

RESULTS

As mentioned earlier, for each location, client system is connected to firewall which contains our programs, scripts. All internet traffic of client system is forwarded to this firewall. In turn, all the traffic received from client is forwarded gateway by this firewall. The detailed network structure with IP is shown below.

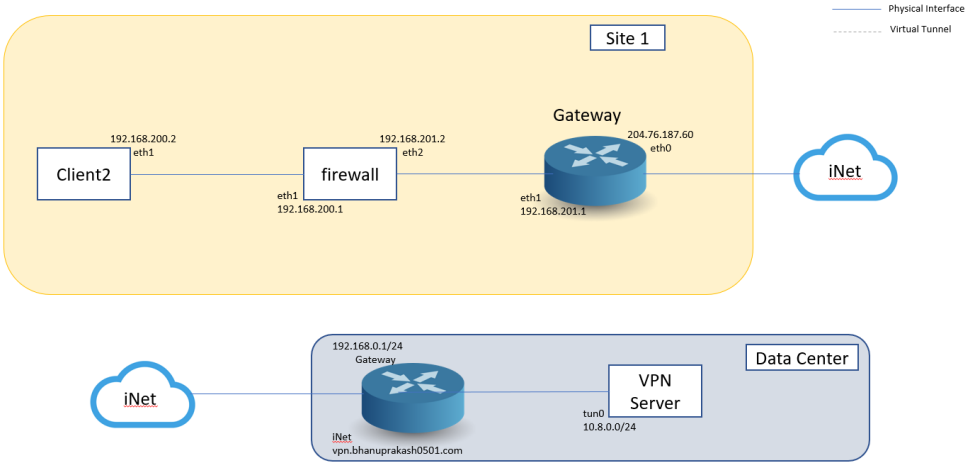


Fig 10a. Client Server Architecture over Internet before connecting to OpenVPN server

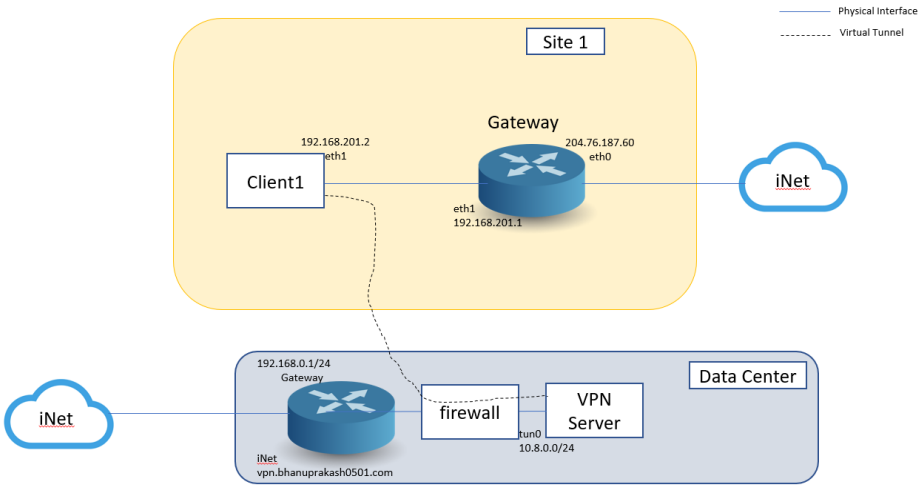


Fig 10b. Client Server Architecture over Internet after connecting to OpenVPN server

5.1 Geographic Locations of GENI Testbed Locations

Here is the map showing the locations chosen for doing this experiment. All the systems are allocated on InstaGENI of various universities.

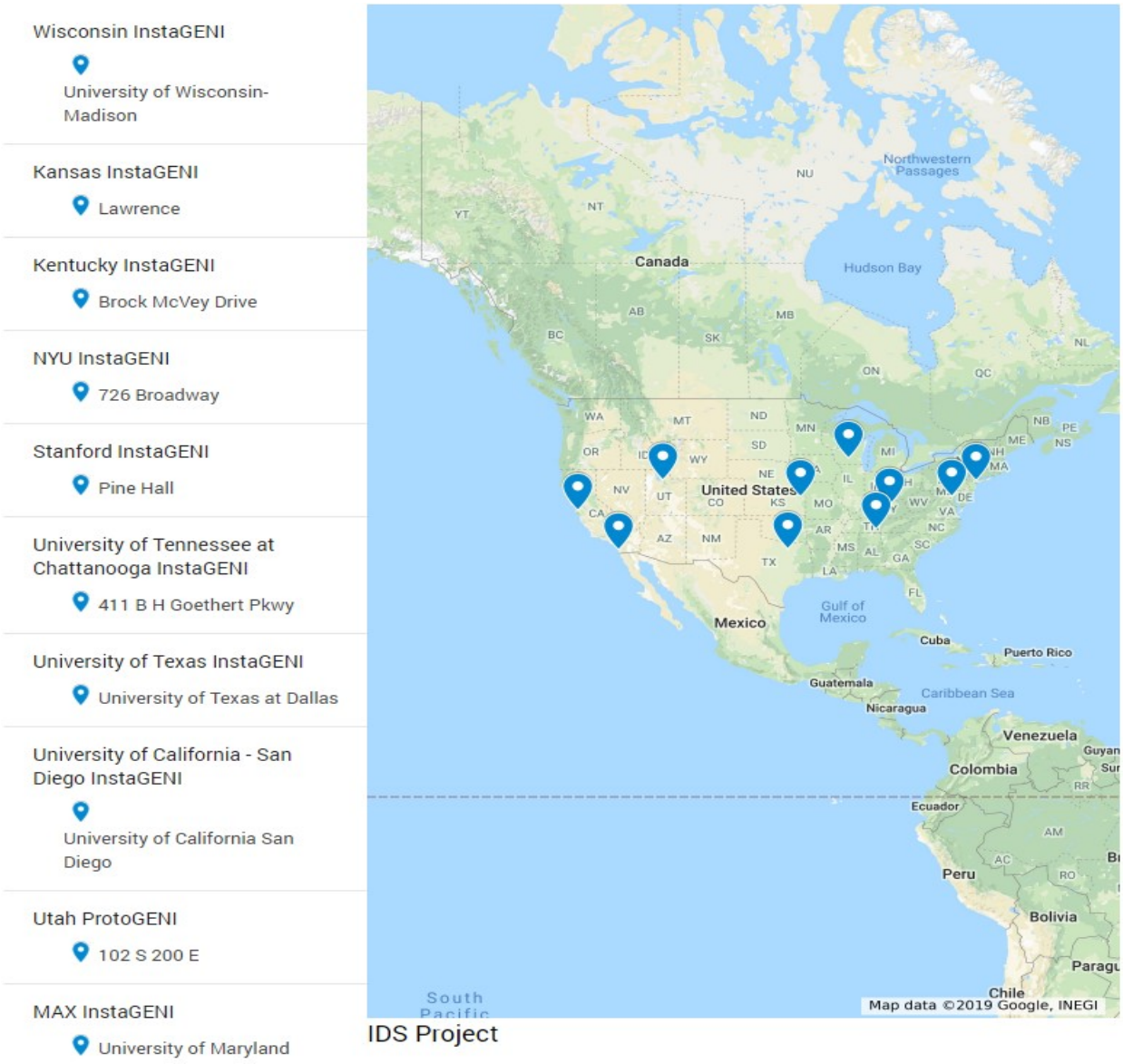


Fig 11. Geo Locations of GENI Testbed

5.2 Traceroute information

As mentioned in the above diagram, when we traceroute to 8.8.8.8 public IP from client machine, it should pass through firewall and then gateway which is shown in below output when VPN is not connected.

```
bpyv6@client:~$ traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  FW-link-0 (192.168.200.1)  0.751 ms  0.632 ms  0.526 ms
 2  gateway-link-1 (192.168.201.1)  2.250 ms  2.124 ms  2.027 ms
 3  pc1.instageni.wisc.edu (128.104.159.20)  2.485 ms  2.327 ms  2.149 ms
 4  128.104.159.1 (128.104.159.1)  2.423 ms  2.816 ms  4.091 ms
 5  r-uwmadison-hub-et-1-3-0-159.uwsys.net (143.235.40.0)  2.494 ms  2.374 ms  2.313 ms
 6  peer-wrips-600w.uwsys.net (143.235.42.5)  13.725 ms  12.567 ms  12.896 ms
 7  72.14.218.180 (72.14.218.180)  12.795 ms  12.777 ms  12.898 ms
 8  108.170.244.1 (108.170.244.1)  19.719 ms 108.170.243.193 (108.170.243.193)  19.542 ms  19.705 ms
 9  209.85.250.53 (209.85.250.53)  13.949 ms 108.170.229.31 (108.170.229.31)  20.286 ms 216.239.47.193 (216.239.47.193)  20.089 ms
10  google-public-dns-a.google.com (8.8.8.8)  19.968 ms  19.864 ms  19.987 ms
bpyv6@client:~$
```

Fig 12. Traceroute without OpenVPN Server

When the client is connected to OpenVPN server, then the traffic will pass via VPN tunnel. At this point when we trace the route to 8.8.8.8, firewall and gateway will no be shown. This traceroute is shown when site 1 client is connected to site 10 OpenVPN server.

```
bpyv6@client:~$ traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  10.8.0.1 (10.8.0.1)  56.882 ms  56.656 ms  56.472 ms
 2  pc3.instageni.washington.edu (128.95.190.18)  56.290 ms  56.114 ms  55.932 ms
 3  128.95.190.253 (128.95.190.253)  57.970 ms  57.786 ms  57.524 ms
 4  lo0--1.uwcr-atg-1.infra.washington.edu (198.48.66.1)  56.819 ms  56.669 ms  56.539 ms
 5  10.132.1.75 (10.132.1.75)  56.251 ms  56.117 ms  55.934 ms
 6  ae0--4011.icar-sttl1-2.infra.pnw-gigapop.net (209.124.190.134)  55.740 ms  51.402 ms  51.199 ms
 7  72.14.223.77 (72.14.223.77)  51.502 ms  51.331 ms  51.550 ms
 8  108.170.245.113 (108.170.245.113)  51.260 ms  51.256 ms  51.045 ms
 9  209.85.254.167 (209.85.254.167)  51.159 ms 209.85.254.93 (209.85.254.93)  51.357 ms 216.239.63.189 (216.239.63.189)
10  google-public-dns-a.google.com (8.8.8.8)  51.194 ms  51.025 ms  50.848 ms
bpyv6@client:~$
```

Fig 13. Traceroute with OpenVPN Server

5.3 Throughput test

On all these servers of GENI portal, we limited the interface speed to 100Mbps. So, without VPN, we are getting 99~100Mbps speed when we did a perform speed test to local server at the same location. Client at each location is connected to all 10 OpenVPN servers at different locations without firewall actively interpreting the traffic and performed throughput test. The results are shown below are average of 20 runs shown in Mbps.

Table 1. Throughput calculation without proposed firewall (Values in Mbps)

Client\Server	site 1	site 2	site 3	site 4	site 5	site 6	site 7	site 8	site 9	site 10
site 1	87	73	64	68	71	62	58	69	69	51
site 2	71	89	65	68	62	69	70	63	63	72
site 3	69	59	88	72	51	52	61	62	68	69
site 4	73	70	66	85	69	65	58	57	71	67
site 5	67	63	50	65	89	61	63	69	63	61
site 6	68	58	59	70	60	88	69	64	61	67
site 7	63	65	71	59	64	66	86	68	66	55
site 8	70	64	70	54	62	55	72	89	61	67
site 9	67	60	68	59	69	50	63	59	90	66
site 10	55	67	67	61	62	66	67	63	67	87

Also, I installed the firewall to interpret the OpenVPN traffic and preformed speed test. The speed tests got little bit degraded due to encryption and decryptions performed in firewall itself to inspect the traffic. The speed tests are shown in below table (Mbps).

Table 2. Throughput calculation with proposed firewall (Values in Mbps)

Client\Server	site 1	site 2	site 3	site 4	site 5	site 6	site 7	site 8	site 9	site 10
site 1	89	74	70	71	72	65	60	70	71	57
site 2	73	91	65	70	64	70	71	62	65	74
site 3	69	60	90	72	55	51	66	65	72	71
site 4	75	72	65	88	70	71	60	61	75	68
site 5	70	63	53	65	92	65	68	71	64	63
site 6	72	63	64	72	61	87	70	67	63	68
site 7	62	67	74	61	63	68	88	69	65	58
site 8	74	63	71	55	67	58	73	91	62	67
site 9	70	64	70	61	70	51	63	62	92	68
site 10	63	71	68	65	68	66	70	64	71	89

5.4 Throughput difference With and Without Firewall:

The difference between these two tables is due encryption, decryption and inspection of traffic inside firewall. This difference is shown in below table (Mbps)

Table 3. Throughput difference caused by proposed firewall. (Values in Mbps)

Client\Server	site 1	site 2	site 3	site 4	site 5	site 6	site 7	site 8	site 9	site 10
site 1	2	1	6	3	1	3	2	1	2	6
site 2	2	2	0	2	2	1	1	-1	2	2
site 3	0	1	2	0	4	-1	5	3	4	2
site 4	2	2	-1	3	1	6	2	4	4	1
site 5	3	0	3	0	3	4	5	2	1	2
site 6	4	5	5	2	1	-1	1	3	2	1
site 7	-1	2	3	2	-1	2	2	1	-1	3
site 8	4	-1	1	1	5	3	1	2	1	0
site 9	3	4	2	2	1	1	0	3	2	2
site 10	8	4	1	4	6	0	3	1	4	2

Visual view of above table is shown in below graph.

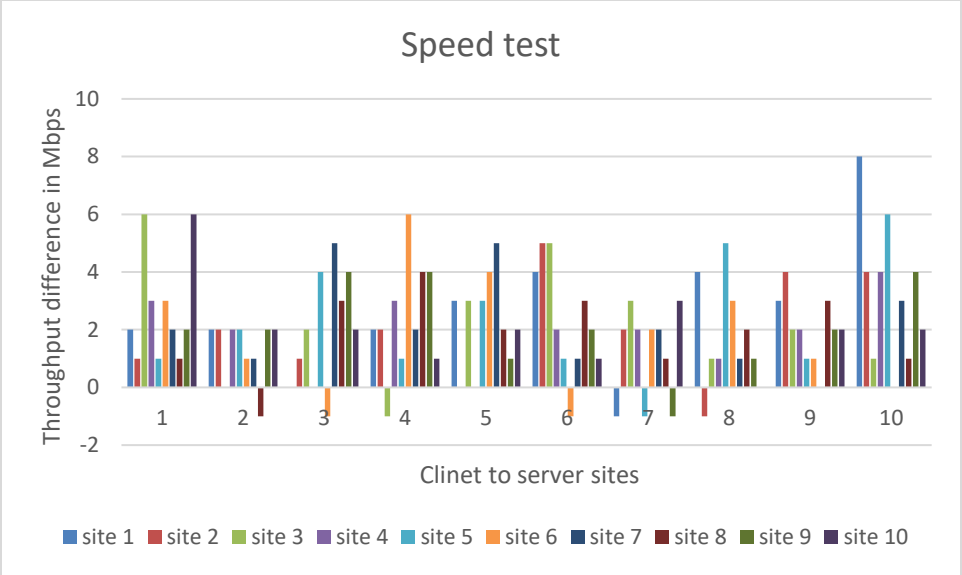


Fig 14. Graph for throughput difference caused by proposed firewall. (Values in Mbps)

5.5 Latencies:

Latencies are one of the important parameters to determine the network performance. In this experiment, latencies are calculated at each point of the node. Each site's client is connected to OpenVPN server on other sites. After connecting, performed a curl to www.google.com website and at the same time, tcpdump is captured on client tunnel interface, both tunnels of firewall and OpenVPN tunnel. After analyzing this tcpdump data, time difference between TCP initial SYN and SYN+ACK packet is noted as latency. Latencies calculated on OpenVPN server are purely due to internet as this data is going out to google.com and coming back over internet. Also, time difference between SYN on TUN0 and TUN1 of firewall is considered as latencies due to firewall as iptables firewall is the only thing inspecting at this moment. Time difference between SYN of ETH1 and tun0 on firewall is considered as

encryption and decryption performed by my script. All this info from client to server at same location is shown in below table (milli seconds).

Table 4. Latencies Calculated without Internet (Values in milli sec)

	client	firewall encryption/decryption	firewall iptables latency	OpenVPN delay
site 1	22.459	0.776	0.051	18.798
site 2	23.061	0.765	0.055	20.598
site 3	22.459	0.743	0.051	20.148
site 4	24.086	0.81	0.061	21.514
site 5	21.913	0.721	0.055	19.541
site 6	27.128	0.757	0.052	24.587
site 7	30.397	0.805	0.058	27.748
site 8	21.451	0.641	0.049	19.244
site 9	32.471	0.735	0.051	30.186
site 10	19.605	0.709	0.055	17.254
Average	24.503	0.7462	0.0538	21.9618

Also, latencies on clients of all sites to www.google.com when connected to all OpenVPN severs a tabulated in below table (milli seconds)

Table 5. Latencies calculated with internet (Values in milli sec)

	site1	site2	site3	site4	site5	site6	site7	site8	site9	site10
site1	22.459	83.458	78.674	79.481	83.157	89.482	85.247	68.568	120.917	92.631
site2	79.248	23.061	81.782	87.791	89.421	92.526	90.175	75.218	116.375	72.429
site3	71.482	80.621	22.459	90.476	82.374	97.647	92.482	78.341	110.846	70.562
site4	72.731	85.367	88.637	24.086	84.245	100.674	110.824	92.428	115.219	69.268
site5	79.625	84.468	90.182	89.845	21.913	80.286	90.931	65.486	104.237	73.238
site6	83.127	85.628	86.724	98.647	84.627	27.128	91.824	61.137	119.681	82.691
site7	81.624	88.127	89.428	104.551	92.138	89.534	30.397	85.394	145.375	92.289
site8	62.628	78.534	72.259	90.278	70.815	59.724	81.328	21.451	115.962	75.165
site9	115.751	110.428	112.871	115.57	108.359	110.84	157.352	118.762	32.471	118.638
site10	61.627	67.682	70.015	73.267	78.421	80.764	90.958	80.694	105.357	19.605

5.6 Latency vs MTU:

MTU stands for maximum transmission unit at layer 2 of OSI model. When we change the MTU size on all interfaces including tunnels, there is an effect on latencies on firewall and OpenVPN server as encryption and decryption payload is increased. On other hand, even with change in MTU size, the latencies due to iptables module are remain constant as they inspect only layer 3/4 IP information and payload doesn't matter to this. These latencies are shown in below graph.

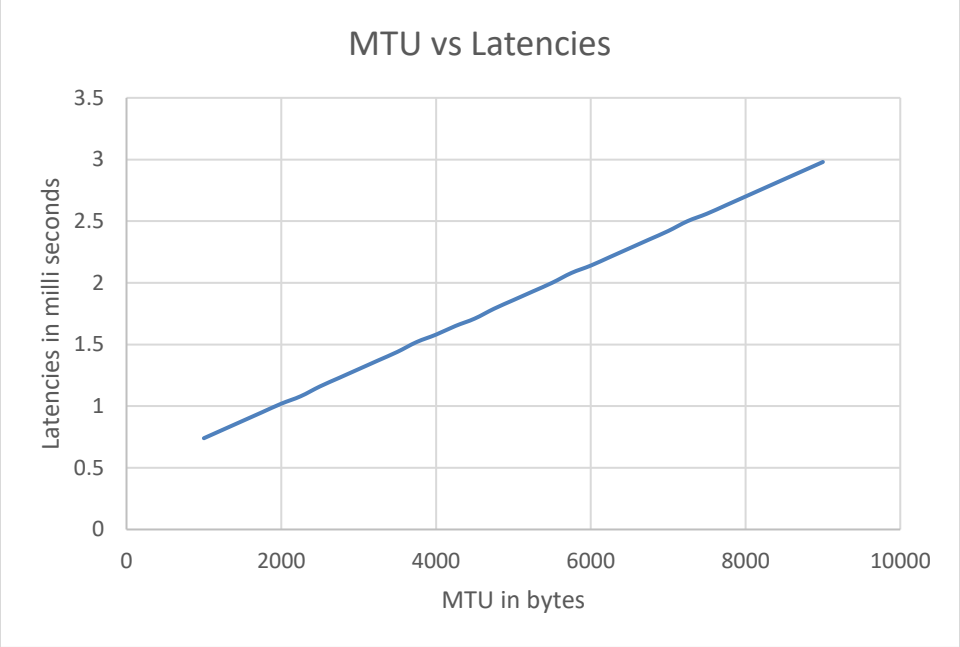


Fig 15. MTU vs Latencies caused by firewall

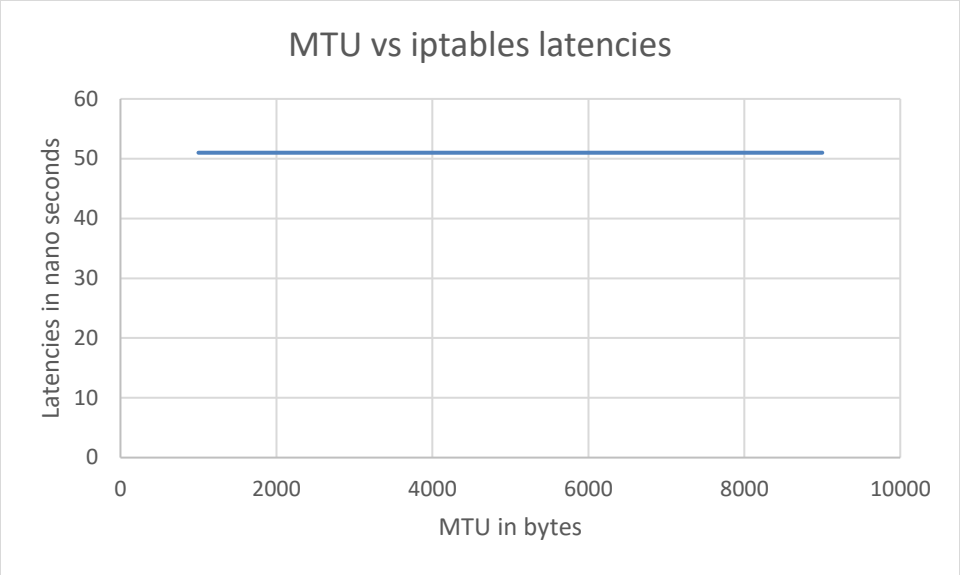


Fig 16. MTU vs Latencies caused by iptables firewall.

CHAPTER 6

FUTURE SCOPE

This firewall design can be implemented in many ways such as sniffers to monitor encrypted traffic, as a standalone firewall, antivirus to block viruses over tunneled interfaces, vulnerable commands over encrypted tunnels, reverse proxies for secure applications, etc. Also, this proposed model can be used to monitor encrypted traffic at various layers such as WPA, 802.1x, PAP, CHAP, EAP, etc.

Appendix

Documentation on OpenVPN

OpenVPN is an opensource VPN sever with source code hosted in GitHub. Even if you search for documentation of OpenVPN source code, we can find plenty of documentation related to configuration files of client and server but not source code. Below are some of the difficulties I faced with OpenVPN while doing this thesis.

- Source code documentation is available at https://build.openvpn.net/doxygen/
- This document is divided into 4 modules namely driver module, network interface module, control channel module and data channel module.
- Whole structure of all modules, classes and header files are listed graphically represented at https://build.openvpn.net/doxygen/dir_68267d1309a1af8e8297ef4c3efbcdba.html
- In above link, if we dig each module, it will describe the function calls, arguments of each function, return values and various dependent packages.
- The directory structure of the source code is documented at this URL: https://build.openvpn.net/doxygen/files.html
- Each function module can be called in C program individually, but we need to include the whole package before calling it. Otherwise, compilation will fail due to missing dependency packages.

REFERENCE LIST

[1] D. Lacković and M. Tomić, "Performance analysis of virtualized VPN endpoints," 2017 40th *International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, 2017, pp. 466-471

[2] I. Coonjah, P. C. Catherine and K. M. S. Soyjaudah, "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)*, Pamplemousses, 2015, pp. 1-4.

[3] J. R. Maria Navin, P. Suresh, and K.R. Pradeep. (2013). "Implementation of OpenSSL API's for TLS 1.2 Operation", 2013 *International Journal of Advanced Computer Research,* 3(12) S. No. 3, pp. 179-183.

[4] L. Daniel, E. Poll and J. de Ruiter, "Inferring OpenVPN State Machines Using Protocol State Fuzzing," *IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, London, 2018, pp. 11-19..

[5] J. Qu, T. Li, and F. Dang, "Performance Evaluation and Analysis of OpenVPN on Android", *Fourth International Conference on Computational and Information Sciences* 2012.

[6] Michael Hall and Raj Jain, "Performance analysis of openvpn on a consumer grade router, " cse.wustl.edu, Nov 2008.

[7] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, April 2008. https://tools.ietf.org/html/rfc5246

[8] NIST FIPS PUB 186-2, "Digital Signature Standard", *National Institute of Standards and Technology, U.S. Department of Commerce*, 2000.

[9] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006. https://tools.ietf.org/html/rfc4346

[10] "Specification for the Advanced Encryption Standard (AES)" *National Institute of Standards and Technology*, *U.S. Department of Commerce,* FIPS 197. November 26, 2001.

[11] B. Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed., Published by John Wiley & Sons, Inc. 1996.

VITA

Bhanu Prakash Panchakarla was born on Jan 5th, 1993 in Andhra Pradesh, INDIA. He received his Undergraduate degree from K.L University, Vaddeswaram, INDIA