Journal of Theoretical and Applied Information Technology
31st January 2021. Vol.99. No 2
© 2021 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DESIGNING AND IMPLEMENTING A SECURED SMART NETWORK THAT CAN RESIST NEXT-GENERATION STATE SURVEILLANCE

[1]HAITHAM AMEEN NOMAN, [2]SALMAN IMAD SALMAN AL SATARY, [3]OMAR SAMI WASEF AL-QURAINI, [4]MOHAMED ABDULKADER YOUSFI, [5]QUSAY AL-MAATOUK

[1,2,3,4] Princess Sumaya University for Technology, Jordan
[5] Asia Pacific University of technology and innovation, Malaysia
E-mail: [1] h.ani@psut.edu.jo,[2] satarysalman@gmail.com, [3] omarsquraini@gmail.com, [4] m.yousfi96e@gmail.com [5] qusay@staffemail.apu.edu.my

ABSTRACT

The vast number of internet services have already become an essential entity in people's lives. Although these increasingly growing services have made communication between people easier, yet they deemed to be a double-edged sword, that might expose people privacy to serious threats like intercepting private messages by hackers and governments. This paper aims to develop and implement a specific encrypted end-to-end encrypted network tunnel that protects the user's privacy from being violated by any prying eyes. The designed network comprises Nine VPS (Virtual Private Servers), in which each client and server within this network would implement OpenVPN technology. The route through this Network will be determined by a random algorithm. One of these selected OpenVPN servers receives a particular message from the client, it would only recognize the preceding OpenVPN node that sent this message, so that it will disguise the identity of the original sender. The network has shown a high level of anonymity that can protect the identity of its users along with their data from being monitored by third parties. Moreover, the traffic speed will not be affected significantly.

Keywords: *OPENVPN, VPN, Anonymity, Privacy, Internet surveillance*.

1. INTRODUCTION

Nowadays it is estimated that people send about Twenty-Two billion texts message per day and around Three hundred and fifty billion queries for Google searches per day [1]. These vast and valuable information are exchanged in the world with simplicity and does require only a couple of touches on the devices' screens. Although, this leap in technology has brought a significant development to our world by making it look smaller, yet at the same time, this critical information can be monitored by third parties like Internet service providers and governments as revealed by Snowden and Mark Klein [2].

As we are entering into the digital era, people will have to learn to adapt to the change of digitizing most of the things in life. One of the limitations is the insufficient knowledge on multi-factor authentication method and the implementation method of it [3]. The valuable resources that are stored and process by a normal computer should be protected by a certain type of protection to avoid unauthorized access. But to do this kind of security into its maximum potential, the security mechanism should put human factors such as ease of use and accessibility into considerations as well as the current security mechanisms for most computer systems have completely neglected the importance of human factors in security [4].

A heated debate among law enforcement, security enthusiasts, hacker communities, and human rights activists has recently arisen regarding the question whether online anonymity is acceptable and in which form. From a technical perspective, anonymous communications on the Internet can be achieved at different steps using vast types of technologies such as special Internet browsers, proxy servers, VPN [5].

For users who seek the ultimate solution when it comes to privacy, one of the most notable anonymity-granting technologies is The Onion Router (Tor). Tor is a browser that ensures a level of confidentiality through linking a network of computers, which provide layers of encryption between the user and the information source, making it quite difficult for someone to trace back both sides [6].
Although Tor is functionally neutral, since anonymity can be applied in multiple ways and shaped according to distinct purposes, there are two common appropriations of this specific technology: first, as a resource to circumvent political repression especially in highly repressive contexts to exercise freedom of speech; and second, as a new way to engage in illegal activity, taking advantage of online anonymity to escape law when committing crimes [7].

In fact, Tor is widely known for its illegal uses, and websites on this network are generically referred to as the Dark Net. Despite the variety of contents available through Tor, emphasis is often given only to crypto markets such as Silk Road, the infamous online drug marketplace which connected thou-sands of sellers and buyers using Tor to preserve their identities from 2011 to 2013 [8].

Additionally, an anonymity-granting resource that is commonly available and widely used are VPN services, which are able to change the user's original IP address with another one in another location, typically offering multiple geographical locations around the world to the user to choose. As a result, tracking the user will lead to the IP address not of their computer, but of a server provided by the VPN. On the other hand, data protection through VPN services has one primary advantage from a privacy point of view [9].

The advantage is that all the information shared by the user regardless of the applications is immediately encrypted and dispatched through a secure tunnel established by the VPN server. This is because of the centralization of information by VPN companies; however, this service alone is not considered fully secure. In WLAN security, public networks provide Wi-Fi to the connected clients without the need to enter a passphrase. On addition to their lack of authentication, such networks do not encrypt the data that exchanged between the devices and the network Wi-Fi. These data can hold critical information like username and passwords of servers, chat and visited websites. People can use VPN to provide another layer of security to the connected devices by encrypting their traffic over the public and private networks as well [10, 11].

1.1 Privacy Vs Anonymity

Since the dawn of history, the concept of Privacy as a social and legal issue has been a concern of social scientists and philosophers. Privacy has been recognized as a fundamental human right in the United Nations Declaration of Human Rights, on Civil and Political Rights (PI/EPIC Privacy International, Electronic Privacy Information Center, 1999).

In developed societies and countries, privacy must be protected at any cost. With the advent of modern information and communication technologies systems, the lights have been shed on privacy more than any other time. According to a survey presented in Business Week, privacy and anonymity are the fundamental issues of concern for most Internet users, ranked above issues like usability, fraud, spam. According to the same survey, 78 percent of online users prefer to use the Internet more and 61 percent of non-users would start using the Internet, if privacy policies and practices were disclosed [12].

The concept of privacy may be understood as seclusion in the sense that everyone has the right to remain undisturbed. The second important factor deals with the threats that each anonymity type is effectively coping with. This paper presents a novel technology that provides both privacy and anonymity to the users. In this paper, we will discuss the different technologies that people can utilize in order to hide their identity and protect their data from being monitored then we will discuss the comparison between these technologies in terms of advantages and disadvantages before lastly moving to explain our own methodology that provide the user with high level of anonymity [13, 14].

2. ANONYMITY TECHNOLOGIES

This section discusses the different technologies that can be used to provide both anonymity and confidentiality to the internet users. The internet users will need to download a specific Software to connect to a specific network that will be directing their traffic to the internet anonymously.

2.1 Tor Network

The researcher Chaum in 1981 was the first among the other pioneers who wrote about internet privacy and anonymous communication methodology [15]. Chaum's idea was to hide the relationship between the input and output message by relying on cryptography in order to prevent eavesdroppers and attackers from prying, monitoring and revealing users' data [16].

In 1996 the concept of onion routing was introduced for the first time by the researcher Goldschlag. This concept relies on a two-way communication channel with little latency. The implemented concept is based on a group of connected routers (Also known as Onion routing) [9].

These routers are connected to each other and the data transfer from one to another in a secured connection. This concept was finally implemented and named Tor project and released in 2004. Onion Routing is a distributed overlay network, designed to provide a layer of anonymity to both TCP and UDP based applications like web browsing, SSH, and instant messaging

Tor Networks' clients use Tor browser Software to access the network, the browser then in turns, automatically builds a designated path through the network and establish what called a circuit. Each specific node (or "onion router" in this circuit knows its predecessor and successor, but no other nodes in the circuit. Traffic flows down the circuit in fixed-sized cells, and unwrapped by asymmetric key at each node in the onion network [17].

In a Nutshell, Tor is a complex private network that provides anonymity to its users by encapsulating and routing the traffic through multiple nodes randomly, so that the message originator identity along with the generated traffic are kept concealed [18]. The first generation of Tor Network comprises three entities:

Onion Routers (OR): These are the routers that responsible of forwarding the traffic in the network.

Client or Onion Proxy (OP): It sends traffic through the network.

Directory Servers: They provide signed directories in the network and its routers (The user updates this knowledge periodically via HTTP).

The first generation of Tor network design is depicted in Figure 1.

The second generation of Onion routing was ratified to address the limitation of the previous generation by adding the following improvements [9]:

- Improved forward secrecy: Instead of predetermining the circuit by the client's browser in the previous design, it would initiate session keys with each hop in the circuit.
- TCP streams circuit sharing: The previous generation would have a separate circuit for each application-level request, yet in this particular design, everything will operate on the same circuit.
- Leaky-pipe circuit topology: The client that initiated the circuit can redirect the traffic to a different node partway through the circuit.

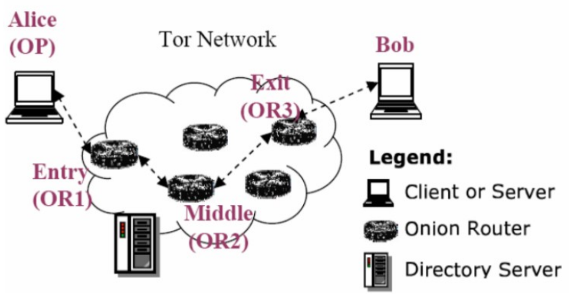


Figure 1: Tor Network Design

- Congestion control: earlier designs did not address congestion; on the other hand, the newer design allows for a little control of the traffic without relying on a centralized monitor.

- End-to-end integrity checking: The original onion routers did not have any integrity checking on the data being sent which was added in the second generation.

2.2 I2P Technology

I2P is an anonymous network, that uses a two-way tunnel to encrypt the routed data. I2P has implemented its communication network and protocol stack on top of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) [6]. I2P ensures that participants can send and receive messages with other participants securely and anonymously.

The protocol made its debut for Freenet, but then it expanded to become an independent project. I2P is based on garlic routing, which is similar to onion routing, since both routing techniques send data through multiple peers in order to hide the sender's and receiver's address [6].

I2P uses a structure similar to the VPN to communicate in the network. When a router in the I2P network is initialized, multiple outgoing and incoming tunnels are created within the router. The outgoing tunnel are controlled by the servers, meanwhile the incoming tunnels are controlled by the clients.

I2P uses the network database (NetDB) to store information on theI2P network. The NetDB is implemented as a DHT and is propagated via so-called flood fill routers using the Kademlia algorithm, making I2P a de-centralized network. For a router to start participating in the I2P network it requires a part of the NetDBin which the information resides it needs to communicate to another participant.
There are two types of tunnels; "Exploratory Tunnel" that are low bandwidth tunnels designed to let the routers test the other tunnels on addition to sending database query data.

The other tunnel is the "Client Tunnel" which is a high bandwidth tunnel that used to send normal data like HTTP, IRC and other services. Each of those services has an associated set of tunnels to transport the data.

All tunnels are made to look the same in order to prevent people from knowing which tunnel is being used to send data. Tunnels in I2P are unidirectional, meaning that different Tunnels are used for transmitting and receiving messages. For instance, if Alice wants to communicate with Bob - who is hosting a service - she establishes a pool of Outbound and Inbound Tunnels. Alice does this by querying the NetDB for a set of peers and obtains their Router Info. Alice then establishes an encrypted connection with the first hop and sets up the first part of the Outbound Tunnel. Through this first hop, she will send the messages required to create Tunnels with the second and third router and completing the Outbound Tunnel. Alice repeats this until she has a set of Inbound and Outbound Tunnels called Tunnel Pools.

2.3 FREENET

Freenet is a decentralized peer-to-peer network architecture that can be used to create a secure and private global file storage system. Each user in the network provides a specific node in which data can be stored in. Additionally, in order to add a new file into the network, the user would need to send out a request message that contains the file itself along with a global unique ID (GUID) through the network. Consequently, the file can migrate or be replicated into other nodes [19].
Technically, the GUID keys are calculated using the SHA-1 hashing algorithm and in order to ensure that the file is not modified, a signed-subspace key (SSK) is used based on asymmetric. On the other hand, routing is done in a similar way to a Tor network. Technically, each node would only know the adjacent node addresses and the GUID (file) which they possibly have whether the node actually has the file or simply knows a node that does have the file.

It cannot be determined, if a node requests a file that is not registered in the adjacent nodes, it will send to the adjacent nodes, one by one.
This process would make the file owner and requester effectively anonymous since each node would act as the file owner and request in the chain of requests.

2.4 Riffle Technology

The system was developed by MIT and the École Polytechnique Fédérale de Lausanne in Switzerland. It's based on the same encryption mechanism of onion-encryption system in Tor. The system wraps messages in layers of encryption as they travel through the anonymizing network to disguise and conceal the route the network users have taken. Similar to Tor, Riffle runs connections through a mix network of nodes, passing packets from system to system to conceal the identity of the original source. What distinguish Riffle from Tor is that the former has extra defenses to potentially prevent spies from unmasking its users [2].

Protecting anonymized users from being identified is a major concern all round because these networks are used by whistleblowers, journalists, government workers and people trying to evade censorship blocks, where unmasking them could lead to imprisonment or more severe punishment. Last year, researchers at Carnegie Mellon University apparently found a way to deanonymize sections of the Tor network by using a series of infected nodes that ratted out the network's users [13].

Riffle as much like Tor also uses onion encryption that consists of clients with many servers in between these servers. These servers are called mixnet, when a mixnet receives messages, it removes a layer of encryption and then shuffles the date before sending it out. Mixnet makes this network resistant to traffic analysis [2].

2.5 Technologies Comparison

Table 1 highlights the difference between the proposed network technology and different explained implemented projects:

Table1: Anonymity Technologies Comparison

Technology	Advantages	Disadvantages
Tor Network	• Distributed Network • Open Source • Continuous support	• Unsecure Exit Node • Fingerprinting circuit is possible • Traffic Analysis attack
Freenet	• Anonymous file sharing • Hiding owner's files	• Unscalable • Node spoofing attack is possible
I2P	• Traffic analysis is very hard	• Slow speed • Waste resources on fake tunnels
Raffle	• Faster than Tor • More resistant to Traffic Analysis	• Unsecure Exit Node

3. DESIGN REQUIREMENTS

The following design requirements have been set for this project:

1- The user shall access the internet through a designated nested server using a specific browser.
2- The browser shall be running under any of the following platforms (Windows, Linux, MAC, Android, and IOS).
3- The browser shall have a feature to bypass firewall restrictions on the network by changing the source port of the traffic.
4- The network shall run OpenVPN with a strong encryption algorithm on Linux servers to redirect traffic.
5- Connecting to the internet via the nested network shall not drop the connection speed by more than 30%.
6- The number of Linux servers used in this project shall be at least 9.
7- The number of client devices who will run the specific browser shall be at least 3.
8- Time4vps server subscription per month, which is 4 Euro per server.

3.1 Analysis of Design Requirements

Since OpenVPN will encapsulate all the data going out of the client's machine; A browser will not be necessary to establish the connection with the server. For that reason, it was decided to let the user access the network by running two scripts that will connect the user with the anonymity network. The client will access the network by only running two scripts; these two scripts will require no more than one command to automatically set up the connection and build the whole path of routers to connect the user to the internet through the anonymity network.

On addition to providing anonymity, the network provides the user the ability to bypass restrictions on firewall. The OpenVPN client will encapsulate the traffic by an HTTP format and then to be sent through port 443 to the OPENVPN server. The server on the other end will be listening on port 443. It will then decapsulate the packet into the original OpenVPN UDP packet. The firewall will let the traffic pass through it as soon as it notices the traffic under HTTPS format. This exchange of information will transport the packets among the other network nodes, in which each node is running OPENVPN server. The network is a one-way network, meaning that a packet will be passed to the servers in a one direction. The Network shifts in a way similar to a train track as it routes the packets either to a different server or outside of the network which is chosen randomly.
However, the track can never get stuck because it always ends up at a particular exit node which that will route it to the outside network. In the design, there will be three entry nodes. The clients will connect to any of these three entry nodes randomly and the network will be able to scale up to any number of clients as long as the servers have RAM to route the new client. A random algorithm was developed to provide the network with anonymity as the entry point will be chosen randomly on addition to all subsequent server routes.

This is done for the sake of hiding the path that the packets would take eventually.

4. DESIGN APPROACH

As explained earlier, the purpose of this research is to create a private network that hides the identity of internet user along with its data. The internet user will access this network through any preferred browser, yet before doing so, the user should run a script that to set the proper connection, encryption and route. We designed our network to consist of Nine servers, each of these servers runs OPENVPN open-source VPN service. To explain the topology of the network, the following Software components have been used:

4.1 Tunneling Protocol

Tunneling protocol is an important part of this project, as it is the infrastructure of the whole network, for it will decide the way that the traffic will be encrypted, how the traffic will be routed, and how the servers and clients will interact with each other, and how they will be configured. OpenVPN network uses ports 1190, 1191, and 1192 and so on are usually blocked in many firewalls. This issue can be solved using the following approaches:

- Configuring OpenVPN to use an open port

OpenVPN can be configured to use port 443, the firewall will recognize that the packets sent through this port are not SSL packets and will immediately drop them.

- Tunneling OpenVPN data through DNS

This method will require to send the OpenVPN packets through the DNS protocol. However, we will need to a DNS server to implement this approach. Additionally, tunneling data through a DNS tunnel will be very slow; since it will stuff data inside DNS queries that can hold small size of data.

- Tunneling OpenVPN through an SSL tunnel

Port 443 is most likely open in any firewall, and since port 443 is used for SSL communication. This method will convert an OpenVPN packet into an SSL packet. In that case, a firewall will not be able to block the packet as it will appear as an SSL packet. A tool dubbed “Stunnel” was used to create an SSL tunnel. Stunnel works by encrypting all packets sent to the OpenVPN port and then forwarding them to port 443. On the server side, Stunnel will listen to port 443, it will then decrypt all packets and then forward them to the OpenVPN port.

The reason behind using OPENVPN is because it is open-source protocol that uses TLS protocols, and is easy to configure while maintaining strong security that has customizable symmetric encryption algorithms like AES, Blowfish, or Camelia and asymmetric encryption algorithms. OPENVPN harness the security strength of IPsec and the OpenSSL which made it the ideal choice for this project. OpenVPN also uses two virtual channels to communicate between client and server. Both of these channel’s encryption and authentication are determined differently. Since we are going to connect a group of OPENVPN servers, we encountered a technical problem. OpenVPN does not support server to server connection. Because of this, it was necessary to come up with an approach to let two servers running OpenVPN to connect with each other. Since OpenVPN works on a specific port number over mostly UDP [20], one port number was assigned for the server process, meanwhile another port number for the client process. This allows the VPS server to work as both an OpenVPN client and server; however, when the VPS server was configured to work as both an OpenVPN client and server.

The server was assigned with the public IP address of the followed server that will be connected to. Because of this, it will not allow SSH connection on its public IP address. This issue was solved by adding a rule to the IP table in the server, in order to allow SSH connections to be established to its original public IP address.

4.2 Routing Design

Our routing design is established on client’s side before routing the traffic. This is similar to how Tor does its routing; however, it provides a problem of having a static route that can be analyzed, since the client will have a unique path. Additionally, we tend to randomize the path of the packets, so that it’s continuously routed between nodes, unless the random algorithm chooses to exit the network.

This would solve the problem of having a static predictable path. The router data can only travel one way to avoid the looping address problem and to avoid the data getting stuck inside the network. The network routing algorithm can be illustrated in Figure 2.

The user will connect to any of these selected servers randomly. Each of these servers can be considered as entry nodes, or gateway to the network. For instance, Server 1 will be connected to the Eight other servers as a client. These Eight servers will handle the passing packets from the entry servers. Meanwhile, Server 2 will be connected to the Six other servers as a client, and server 3 will be connected to the Seven other servers as a client as well. The other Six servers will act as a client to a total number of servers equal to the total amount of servers minus the server's number.

The topology will act in a similar fashion to a train rail, meaning that the traffic incoming to a server will be routed to the same gateway until a certain amount of time passes. The Python script "dbth.py" will be used to generate a random number and based on generated number, it will choose a gateway, in which it will route all traffic through

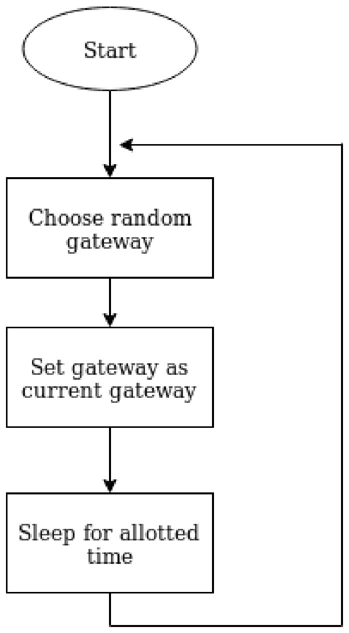


Figure 2: Routing Algorithm

Afterward, the code will sleep for an allotted time and then back into choosing a random gateway to route the traffic through accordingly. It can be noticed from Figure 3 illustration the three entry nodes servers are connected to Eight different servers in which each of them is running OPENVPN service to route the traffic in one direction.

As illustrated in Figure 2, the client has direct access to only three servers. Those servers will be the entry point for the client to connect to nested connected OPENVPN servers.

4.3 User Interface and Setup

The user interface and setup were designed to be usable and friendly for all types of users. The user will need to run a python file that will be downloaded by the user called "clientSetup.py" then the user will need to go to instruction folder.

Afterward, the user will need to run another Python file called "ui.py" along with the command "sudo OpenVPN --config client.conf", the user will then get connected to the network.

As for the "ui.py" script, it runs the random algorithm to pick the entry server. Subsequently, it checks if the server's configuration, key, and certificate files are available, if they are not, it requests them from the server by sending a request file, , where it will retrieve the key and certificate from the server's client.

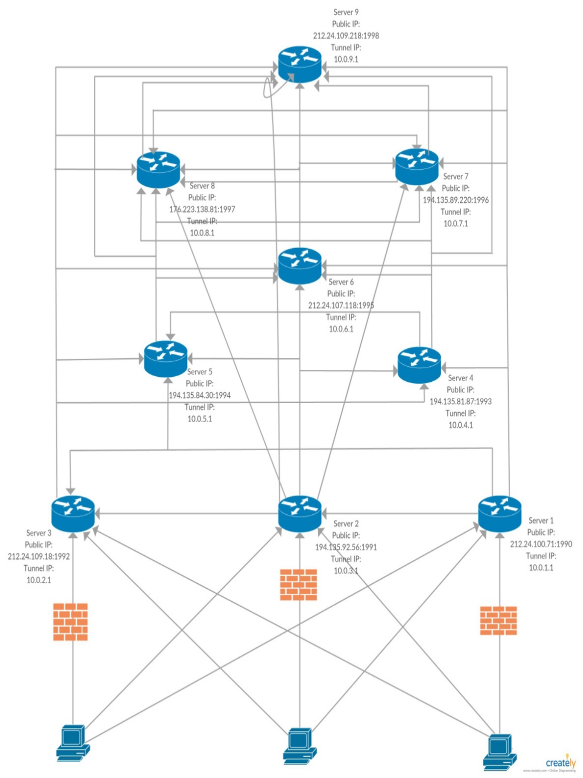


Figure 3: Network Design

Transferring the configuration files of the clients from the server to the client is an important issue that must be done securely. if the client's files were to be compromised then all the traffic from the user to that server will be decrypted leaving the connection vulnerable.

4.4 Server Interface

We wrote a server interface script dubbed “Si.py”. The script listens for incoming clients’ connections on port 5000.
Threads are opened for any client that connects to it. Afterward, it waits for the client to send a certificate request that will be used to generate a client certificate.

4.5 Firewall Bypassing Feature

OPENVPN server uses UDP protocol over ports 1190, 1191, and 1192 which can easily be blocked by any ISP leaving our network unavailable for the users. For that reason, we decided to come up with a solution to overcome this issue. The technique benefits from Port 443. This port in most cases is allowed in firewall, since it is used for SSL communication. Our technique will convert an OpenVPN packet into an SSL packet so that firewalls will not be able to block the packet as it will appear as an innocuous SSL packet.

A tool dubbed “Stunnel” was used to create an SSL tunnel. Stunnel is a proxy designed to add TLS encryption functionality to existing clients and servers without any changes in the programs' code. Its architecture is optimized for security, portability, and scalability (including load-balancing), making it suitable for large deployments. Stunnel uses the OpenSSL library for cryptography, so it supports whatever cryptographic algorithms are compiled into the library.

One could use stunnel to provide a secure SSL connection to an existing non-SSL-aware SMTP mail server. Assuming the SMTP server expects TCP connections on port 25, one would configure stunnel to map the SSL port 465 to non-SSL port 25. On the other hand, a mail client connects via SSL to port 465. Network traffic from the client initially passes over SSL to the stunnel application, which encrypts/decrypts traffic and forwards unsecured traffic to port 25 locally. The mail server would see a non-SSL mail client. In the customized network, we decided to use this tool by encrypting all packets sent to the OpenVPN port and then forwarding them to port 443.
On the server side, Stunnel will listen to port 443, it will then decrypt all packets and then forward them to the OpenVPN port. Both Figure 4 and Figure 5 illustrate the mechanism of the tool Stunnel.

On the server’s side, Stunnel will be waiting for packets to be received on port 443. As soon as the tool Stunnel receives the passing packets, it will decrypt those packets and then forward them to the port that OpenVPN is listening on. Figure 5 illustrates the server side of the tool.

4.6 IP Tables

To control the network, IP tables need to be configured, this is done in the server setup. Enabling IP Forward in the Ip Tables is essential to allow the traffic to pass through the servers . Also, using IPMASQ feature to masquerade the clients so that they will not be assigned special IP addresses, instead they will all rely on the server’s IP address.

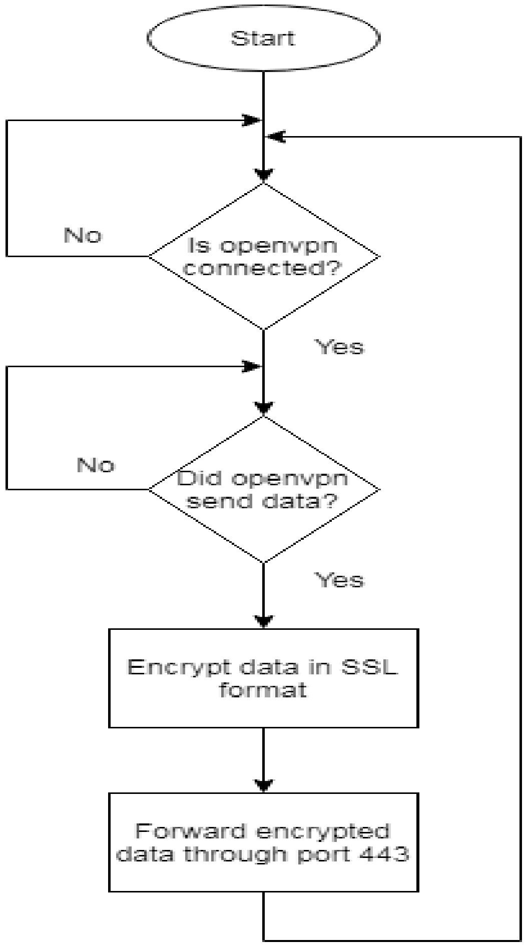

Start
Is openvpn connected?
No
Yes
Did openvpn send data?
No
Yes
Encrypt data in SSL format
Forward encrypted data through port 443

Figure 4: Stunnel Tool Workflow

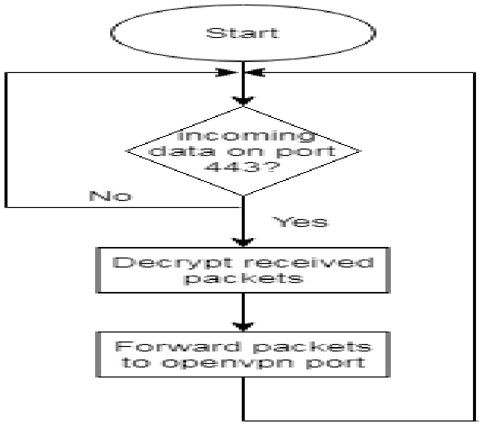


***Figure 5:** Stunnel Server Side Workflow*

5 RESULTS

This section discusses the testing phase of the network privacy. As explained previously, the Software is made up of three different parts: User to the server, server to server, and finally server to the outside of the network. The first connection will be made through the OpenVPN connection between the users as a client to the server on the other side.

This connection will provide the user with an IP address and route its traffic through the server. The second phase will be established between two servers in which each server runs OPENVPN server. This connection will provide the user with extra anonymity by routing the traffic through multiple OPENVPN servers randomly. The third and last connection will be between the server and the destination of the user to get the traffic back to the user. All the passing traffic will be encrypted to provide confidentiality and anonymity to the users.

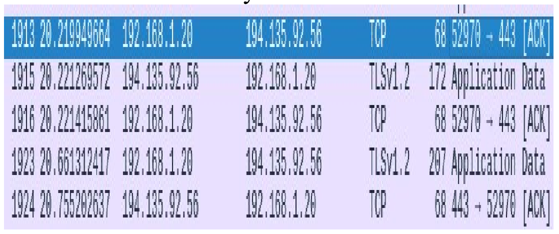
To test the network, it was decided to conduct four different experiments in order to test the network performance, security, and capability of evading firewalls as listed below:

5.1 Firewall Test

We decided to run the system to try bypassing Fortinet firewall, without using Stunnel tool and solely depending on OPENVPN tunneling.

As expected, the firewall blocked the connection to the network, since it prevents OPENVPN ports. As an alternative approach, we decided to run Stunel to circumvent the firewall by tunneling the traffic through SSL (specifically port 443) and the connection was successfully initiated.

As shown in Figure 6, Wireshark was used to analyze the traffic, where it shows, packets were encapsulated and tunneled through port 443 and the connection successfully established with the server.



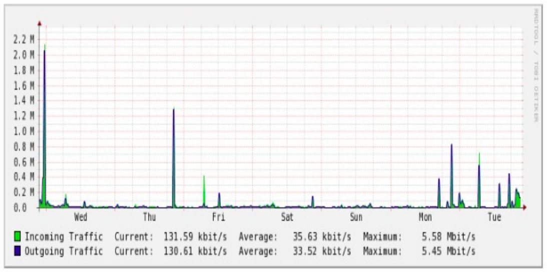
```
1913 20.21994964  192.168.1.20     194.135.92.56    TCP       68 52970 → 443 [ACK]
1915 20.22126957  194.135.92.56    192.168.1.20     TLSv1.2  172 Application Data
1916 20.22141586  192.168.1.20     194.135.92.56    TCP       68 52970 → 443 [ACK]
1923 20.66131241  192.168.1.20     194.135.92.56    TLSv1.2  207 Application Data
1924 20.75520263  194.135.92.56    192.168.1.20     TCP       68 443 → 52970 [ACK]
```

Figure 6: Wireshark Analysis

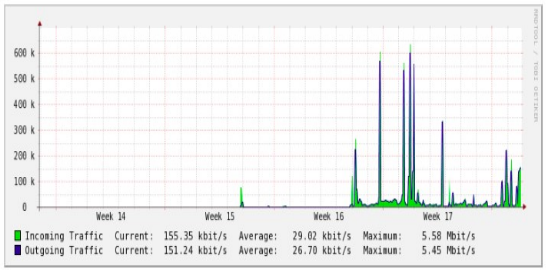
5.2 Traffic Distribution

As can be seen from the following Figures, the entry servers 1, 2, and 3 received the most distributed traffic from the users, this is because all the clients of the network will need to access these servers before accessing the whole network. Figure 7 illustrates

Server 1 traffic distribution on both monthly and yearly basis



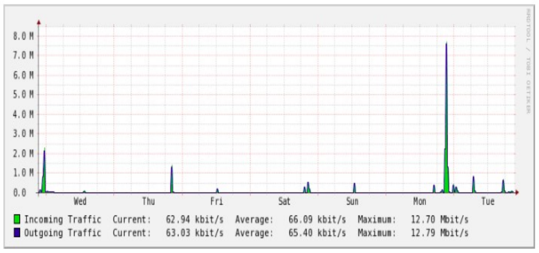
Monthly



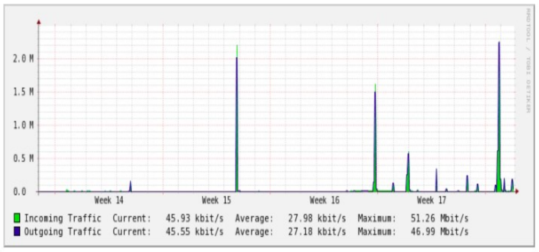
Yearly

***Figure 7:** Entry Server Traffic Distribution*

On the other hand, we can see in Figure 7 the traffic on non-entry Server is less than the entry one, for this example we tool particularly Server 5.



Monthly

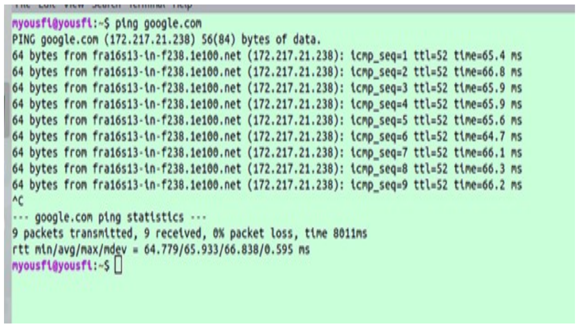


Yearly

***Figure 8:** Non-Entry Server Traffic Distribution*

5.3 Ping Test

Another approach we used to test the performance of the internet connection is to harness the simple functionality of "Ping" utility. Basically, we calculated the required time the packet needs to reach Google servers when using our private network. In order to conduct the test, we had to calculate the time required to reach Google without using our private network. As can be noticed in Figure 8, ICMP packet has an average time of 65.9 millisecond to reach Google's server.

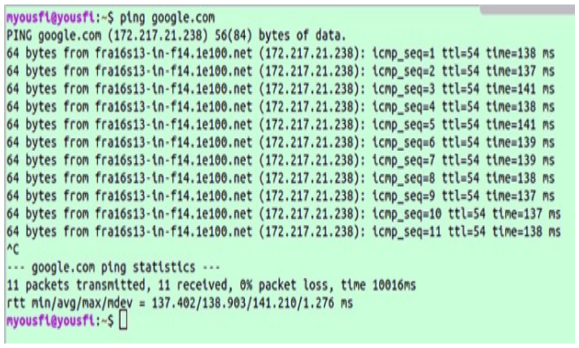


```
myousfi@yousfi:~$ ping google.com
PING google.com (172.217.21.238) 56(84) bytes of data.
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=1 ttl=52 time=65.4 ms
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=2 ttl=52 time=66.8 ms
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=3 ttl=52 time=65.9 ms
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=4 ttl=52 time=65.9 ms
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=5 ttl=52 time=65.6 ms
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=6 ttl=52 time=64.7 ms
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=7 ttl=52 time=66.1 ms
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=8 ttl=52 time=66.3 ms
64 bytes from fra16s13-in-f238.1e100.net (172.217.21.238): icmp_seq=9 ttl=52 time=66.2 ms
^C
--- google.com ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8011ms
rtt min/avg/max/mdev = 64.779/65.933/66.838/0.595 ms
myousfi@yousfi:~$
```

***Figure 9:** Average Time Needed To Reach Google Without Using The Private Network*

On the other hand, we conducted a second similar test but this time with using our network. The test shows, the average time that ICMP took to reach Google's servers was138.9 ms. this is 47.7% of the original ping speed. This indicates that, there was a 52.3% ICMP drop.

The major reason behind the speed drop was the fact that the traffic is routed multiple times between multiple servers, this means that the packets must first travel to the entry server. That took an average time of 92.47 millisecond to go to the server and back. Figure 9 illustrates the results.



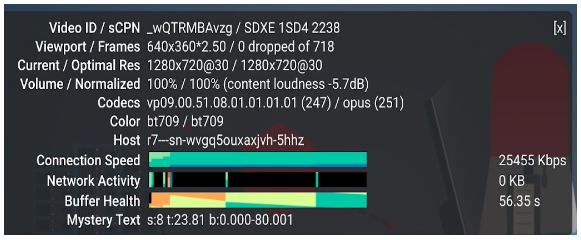
```
myousfi@yousfi:~$ ping google.com
PING google.com (172.217.21.238) 56(84) bytes of data.
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=1 ttl=54 time=138 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=2 ttl=54 time=137 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=3 ttl=54 time=141 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=4 ttl=54 time=138 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=5 ttl=54 time=141 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=6 ttl=54 time=139 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=7 ttl=54 time=139 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=8 ttl=54 time=138 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=9 ttl=54 time=137 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=10 ttl=54 time=137 ms
64 bytes from fra16s13-in-f14.1e100.net (172.217.21.238): icmp_seq=11 ttl=54 time=138 ms
^C
--- google.com ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10016ms
rtt min/avg/max/mdev = 137.402/138.903/141.210/1.276 ms
myousfi@yousfi:~$
```

***Figure 10:** Average Time Needed To Reach Google Using The Private Network*

The most significant speed constraint is the geographical area between all the servers, the client, and the destination. This is a tradeoff between the performance and security.

5.4 NERD Tool Test

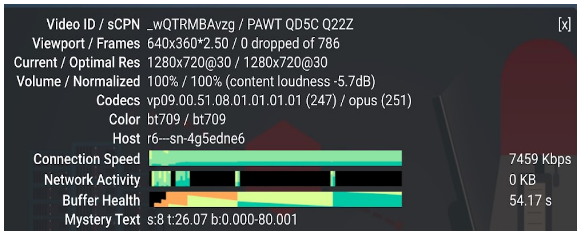
Nerdtool allows the researcher to display many different things directly on top of your desktop; weather reports, headline news. Nerdtool is capable of displaying shell outputs with ANSI escaped formatting, but can also manipulate the final output's colors, size, and font. This test was chosen since it gives us real-time testing to the network and it shows how the bandwidth changes. Figure 11 shows the network statistics while not connected to the custom network. What to note here is that the connection speed is 25.45 Mbps.



```
       Video ID / sCPN  _wQTRMBAvzg / SDXE 1SD4 2238                         [x]
      Viewport / Frames 640x360*2.50 / 0 dropped of 718
  Current / Optimal Res 1280x720@30 / 1280x720@30
    Volume / Normalized 100% / 100% (content loudness -5.7dB)
                 Codecs vp09.00.51.08.01.01.01.01 (247) / opus (251)
                  Color bt709 / bt709
                   Host r7---sn-wvgq5ouxaxjvh-5hhz
       Connection Speed                                              25455 Kbps
       Network Activity                                              0 KB
          Buffer Health                                              56.35 s
           Mystery Text s:8 t:23.81 b:0.000-80.001
```

Figure 11: Users Are Not Connected Through The Network

When the user is connected to the network, the connection speed is 7.45 Mbps which is 28% of the original connection speed, a 72% speed drop as can be noticed in Figure 12



```
       Video ID / sCPN  _wQTRMBAvzg / PAWT QD5C Q22Z                         [x]
      Viewport / Frames 640x360*2.50 / 0 dropped of 786
  Current / Optimal Res 1280x720@30 / 1280x720@30
    Volume / Normalized 100% / 100% (content loudness -5.7dB)
                 Codecs vp09.00.51.08.01.01.01.01 (247) / opus (251)
                  Color bt709 / bt709
                   Host r6---sn-4g5edne6
       Connection Speed                                              7459 Kbps
       Network Activity                                              0 KB
          Buffer Health                                              54.17 s
           Mystery Text s:8 t:26.07 b:0.000-80.001
```

Figure 12: Users Are Connected Through The Network

6. CONCLUSION AND FUTURE WORK

In conclusion, it can be noticed that our private network routes traffic between the deployed servers randomly, which causes the IP address of the client to change according to the exit point of the network. We used Nine servers in our network where each runs OPENVPN. Another advantage we would like to shed the light on is the scalability of our private network.

As a matter of fact, we designed the network in a way that facilitate the addition of extra servers. The network will also encrypt the data to hide the user's traffic. As for future work, one major addition that can speed up the network and provide a larger range of scalability is by changing the network to an SDN which gives the network more routing capabilities.

In consequence, as more servers added to the network the more the privacy will be enhanced. This is simply because it will be harder to predict the traffic path that travels through many nodes.
Keeping Stunnel feature always enabled can allow the users to bypass firewalls as mentioned before. Another major upgrade for the project is by changing the encryption algorithm from AES 128 to AES 256 for better encryption. Enhancing the graphical user interface and simplify it is essential as that would let normal clients to connect to the network easily.

A graphical user interface that can have a range of customizable settings such as, max hop count that limits the number of times the traffic will be routed, a manageable key files system that can be updated whenever needed.

REFERENCES

[1] M. Herrmann and C. Grothoff, "Privacy-implications of performance-based peer selection by onion-routers: a real-world case study using I2P," in *International Symposium on Privacy Enhancing Technologies Symposium*, 2011, pp. 155-174: Springer.
[2] A. Kwon, D. Lazar, S. Devadas, and B. Ford, "Riffle: An efficient communication system with strong anonymity," *Proceedings on Privacy Enhancing Technologies,* vol. 2016, no. 2, pp. 115-134, 2016.
[3] D. C. JIANLI, M. AL-RASHDAN, and Q. AL-MAATOUK, "SECURE DATA STORAGE SYSTEM," *Journal of Critical Reviews,* vol. 7, no. 3, p. 2020, 2019.
[4] Y. S. CHUEN, M. AL-RASHDAN, and Q. AL-MAATOUK, "GRAPHICAL PASSWORD STRATEGY," *Journal of Critical Reviews,* vol. 7, no. 3, p. 2020, 2019.
[5] N. P. Hoang and D. Pishva, "Anonymous communication and its importance in social

networking," in *16th International Conference on Advanced Communication Technology*, 2014, pp. 34-39: IEEE.

[6] T. Minárik and A.-M. Osula, "Tor does not stink: Use and abuse of the Tor anonymity network from the perspective of law," *Computer Law & Security Review,* vol. 32, no. 1, pp. 111-127, 2016.

[7] E. Jardine, "Privacy, censorship, data breaches and Internet freedom: The drivers of support and opposition to Dark Web technologies," *new media & society,* vol. 20, no. 8, pp. 2824-2843, 2018.

[8] X. Wang and Y. Zhang, "NCIS 2011."

[9] D. Kurniawan, M. T. Alrashdan, Q. Al-Maatouk, and A. Z. Al-Othmani, "SECURED SOFTWARE EMBEDDED SYSTEM AGAINST DATA BREACHES ON AUTOMOTIVE INDUSTRY," *International Journal of Management (IJM),* vol. 11, no. 11, 2020.

[10] H. A. Noman, S. A. Noman, and Q. Al-Maatouk, "WIRELESS SECURITY IN MALAYSIA: A SURVEY PAPER," *Journal of Critical Reviews,* vol. 7, no. 4, p. 2020, 2019.

[11] M. A. Hussain, M. T. Alrashdan, and Q. Al-Maatouk, "SECURE BIO-RFID SYSTEM IN ORGANIZATIONS," *International Journal of Management (IJM),* vol. 11, no. 11, 2020.

[12] E. Kavakli, C. Kalloniatis, and S. Gritzalis, "Addressing privacy: matching user requirements with implementation techniques," in *7th Hellenic European Conference on Computer Mathematics and its Applications (HERCMA 2005), Athens, Greece*, 2005.

[13] H. A. Noman, S. M. Abdullah, and H. I. Mohammed, "An automated approach to detect deauthentication and disassociation dos attacks on wireless 802.11 networks," *International Journal of Computer Science Issues (IJCSI),* vol. 12, no. 4, p. 107, 2015.

[14] S. Gritzalis, "Enhancing web privacy and anonymity in the digital era," *Information Management & Computer Security,* 2004.

[15] R. A. Haraty and B. Zantout, "The TOR data communication system: A survey," in *2014 IEEE Symposium on Computers and Communications (ISCC)*, 2014, pp. 1-6: IEEE.

[16] P. C. Johnson and A. Kapadia, "From Chaum to Tor and Beyond: A Survey of Anonymous Routing Systems," 2007.

[17] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," Naval Research Lab Washington DC2004.

[18] T. Sardá, S. Natale, N. Sotirakopoulos, and M. Monaghan, "Understanding online anonymity," *Media, Culture & Society,* vol. 41, no. 4, pp. 557-564, 2019.

[19] I. Clarke, S. G. Miller, T. W. Hong, O. Sandberg, and B. Wiley, "Protecting free expression online with freenet," *IEEE Internet Computing,* vol. 6, no. 1, pp. 40-49, 2002.

[20] M. Feilner, *OpenVPN: Building and integrating virtual private networks*. Packt Publishing Ltd, 2006.