



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ



Наташа Спремо, ПР27/2016

Виртуелне приватне мреже

ПРОЈЕКАТ
- Примењено софтверско инжењерство (ОАС) -

Нови Сад, **16.03.2021.**

САДРЖАЈ

1. ОПИС РЕШАВАНОГ ПРОБЛЕМА
2. ОПИС КОРИШЋЕНИХ ТЕХНОЛОГИЈА И АЛАТА
3. ОПИС РЕШЕЊА ПРОБЛЕМА
4. ПРЕДЛОЗИ ЗА ДАЉА УСАВРШАВАЊА
5. ЛИТЕРАТУРА

1. ОПИС РЕШАВАНОГ ПРОБЛЕМА

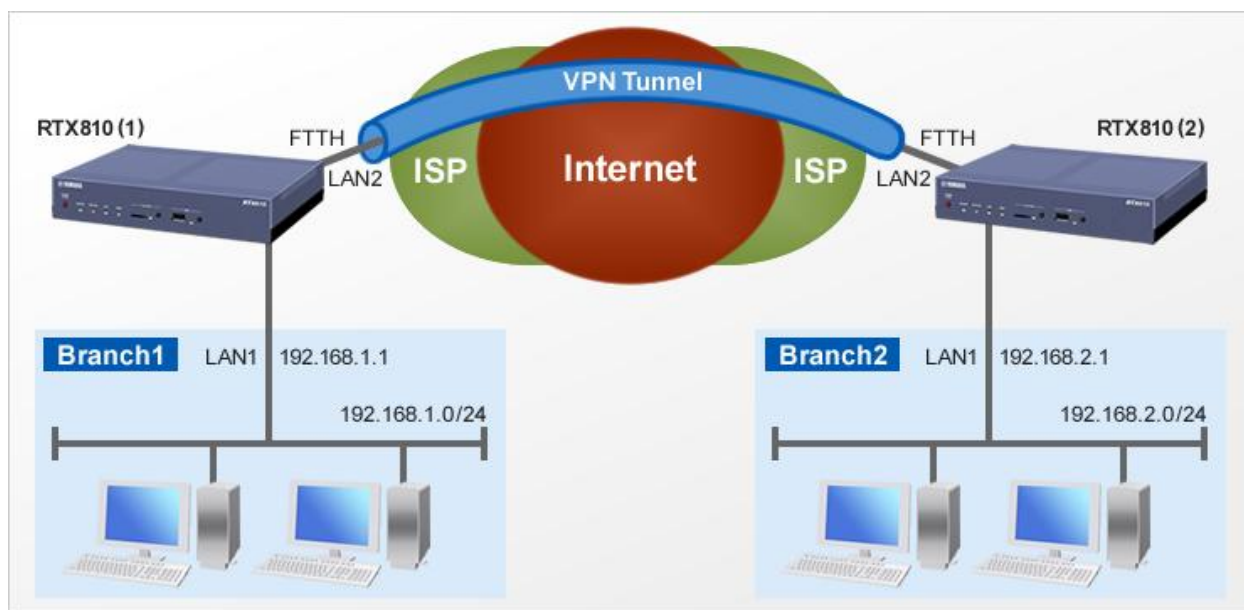
Сваки уређај који је повезан на интернет видљив је интернет провајдерима чије услуге користи корисник. Поред уређаја, провајдери имају приступ свим подацима који саобраћају интернетом. Како би се постигла већа приватност користи се виртуелна приватна мрежа.

Виртуелна приватна мрежа, позната као *VPN (Virtual Private Network)*, је технологија која омогућава сигурно повезивање приватних мрежа у заједничку виртуелну приватну мрежу кроз једну мрежну инфраструктуру. Овакав начин повезивања реализује се „преусмеравањем” путем интернета.

Главна функционалност овакве мреже јесте сигурна комуникација између аутентификованих корисника у чији садржај немају увид спољни посматрачи. Претходно поменути корисници могу бити на географски удаљеним локацијама и несметано користити виртуелну приватну мрежу уколико имају исправне креденцијале.

Предности виртуелне приватне мреже су бројне, неке од којих су брзина, флексибилност, приватност и финансијске погодности.

Принцип рада *VPN*-а уводи појам тунеловања (енг. *tunneling*). **Тунеловање** је саставни део *VPN* мреже и представља пренос пакета података намењених приватној мрежи преко јавне мреже. Рутери јавне мреже немају информацију да преносе пакете који припадају приватној мрежи и *VPN* пакете третирају као нормалан саобраћај. Појам тунел уводи се због тога што су подаци који путују тунелом разумљиви само онима који се налазе на његовом изворишту и одредишту. Почетак и крај тунела се налазе у *VPN* мрежама. Када енкапсулирани пакет стигне на одредиште врши се деенкапсулација и прослеђивање података на коначно одредиште. Цео процес енкапсулације, транспорта и деенкапсулације назива се тунеловање (слика 1).



Слика 1 Дијаграм рада *VPN* мреже

Тренутно постоји широка понуда провајдера виртуелних приватних мрежа на интернету. Приликом одабира треба темељно испитати сигурност и безбедност самог провајдера и услуга које пружа.

Једна од главних функционалности виртуелних приватних мрежа јесте маскирање или промена праве *IP* адресе корисника. Иако *VPN* провајдери предузимају јаке мере да заштите информације корисника, дешавају се грешке попут тзв. *IP leak*-а или грешке при конекцији на сервер.

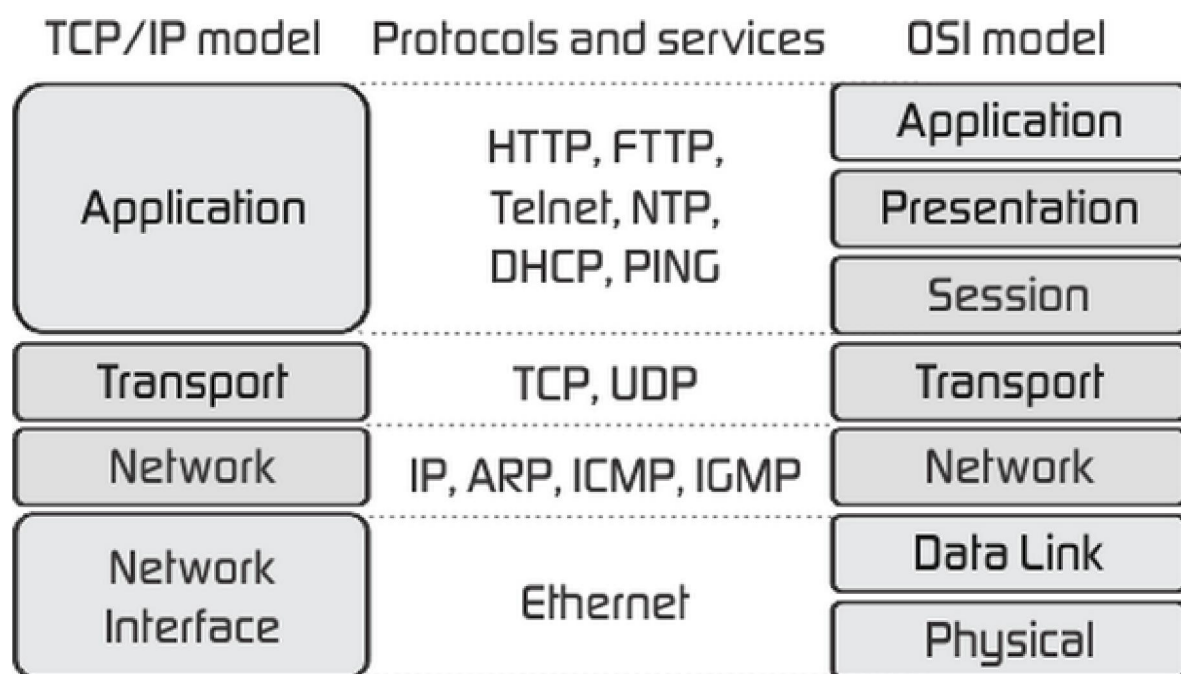
Главна тема пројекта је управо провера валидности услуга које пружа одређен *VPN* провајдер. У сврхе тестирања изабрана је бесплатна понуда компаније *TunnelBear*.

2. ОПИС КОРИШЋЕНИХ ТЕХНОЛОГИЈА И АЛАТА

Пројекат се заснива на две .NET Framework конзолне апликације. Представља основну комуникацију клијента и сервера преко *TCP* конекције, као и добављање *IP* адресе.

2.1 TCP/IP модел

TCP/IP је референтни модел који се користи на интернету. Развијен је пре *OSI* модела, тако да се слојеви ова два модела не поклапају у потпуности (слика 1.1). *TCP/IP* модел чини пет слојева: физички, слој везе, мрежни, транспортни и апликациони. Овај модел се спорадично бави са најнижим слојевима (физичким и слојем везе). Заједно, ова два слоја се третирају као „*host-mreža*” слој. Нагласак се ставља на слој мреже, транспортни и апликациони слој. Мрежни и транспортни слој одговарају слојевима три и четири *OSI* модела. Међутим, код *TCP/IP* на транспортни слој директно се наставља апликациони слој, који обухвата функционалност три вршна слоја *OSI* модела.



Слика 2.1 Разлике *TCP/IP* и *OSI* модела

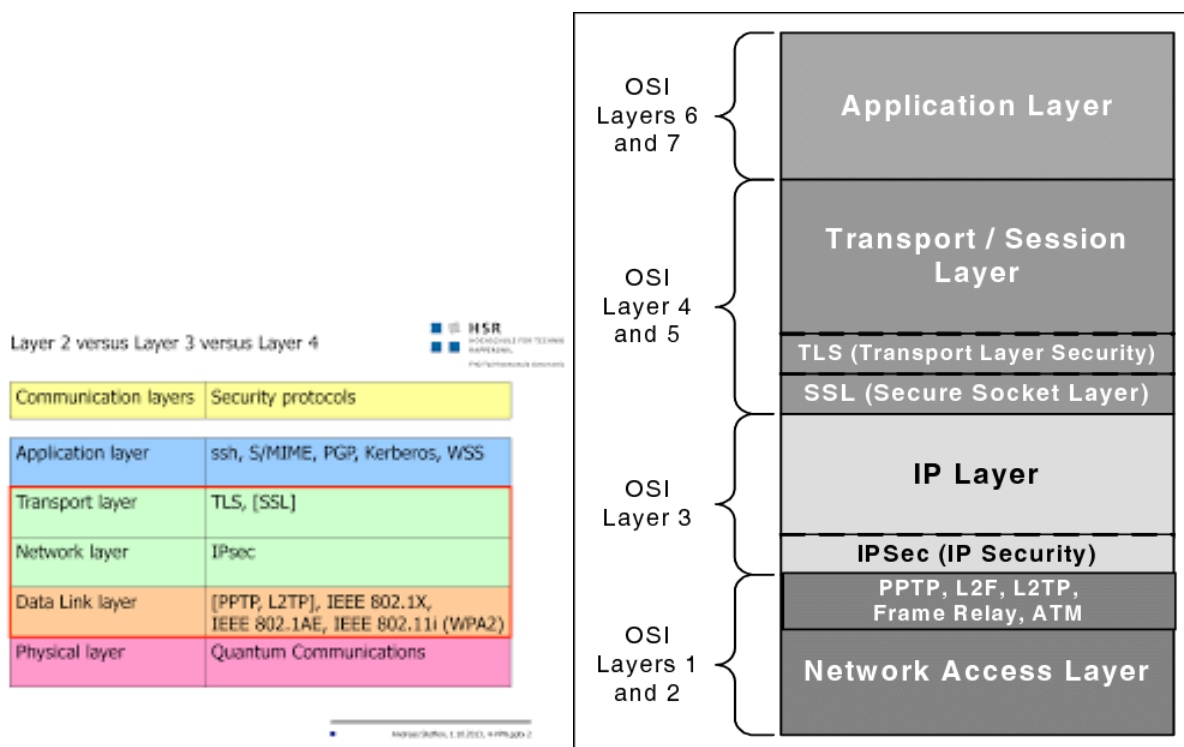
Главни протокол на мрежном нивоу је *IP* (*Internet Protocol*). Интернет слој је одговоран за испоруку пакета од хоста до хоста на интернету. Основни циљ овог слоја јесте рутирање и избегавање загушења на мрежи. Идентификатор који се користи на *IP* слоју *TCP/IP* протокол стека назива се *IP* адресом. Интернет или ***IP* адреса** је 32-битна адреса, која на јединствен начин дефинише везу хоста или рутера на интернет. *IP* адресе су јединствене у смислу да свака адреса дефинише једну и само једну везу на интернет. Два уређаја никада не могу имати исту *IP* адресу.

На транспортном нивоу, *TCP/IP* дефинише два протокола: ***TCP*** и ***UDP***. *TCP* је транспортни протокол конекционог типа који омогућава успостављање поузданог тока бајтова између две удаљене апликације. *TCP* обавља сегментацију тока бајтова на поруке које прослеђује интернет слоју. На страни одредишта, *TCP* реконструише ток бајтова и прослеђује га апликацији. Уз то,

TCP се бави контролом протока како би спречио да брзи предајник претрпа порукама спорог пријемника које он не може да обради.

2.2 Tunnel Bear

За саме услуге виртуелне приватне мреже коришћен је софтвер **Tunnel Bear**. Апликација је доступна за коришћење на готово свим платформама, као што су *Windows, Mac, iOS, Android*. Могућа је употреба и у самом претраживачу (*Chrome, Firefox, Opera*). Ова екстензија омогућава безбедно коришћење *browser-a*, односно енкриптује једино податке на веб претраживачу и може се имплементирати на било коју платформу коју подржавају претходно наведени претраживачи. Софтвер који се користи на овом пројекту јесте *Tunnel Bear* за *Windows* оперативни систем. Примењује се АЕС 256-битна енкрипција и самим тим пружа висок ниво сигурности.



Слика 2.2 Приказ протокола које користи виртуелна приватна мрежа

Постоје различити протоколи уз помоћ којих се остварује *VPN* конекција. На другом нивоу *ISO/OSI* модела користи се *Layer 2 Tunneling Protocol (L2TP)* и већ превазиђен *Point-To-Point Tunneling Protocol (PPTP)*. На трећем нивоу налази се најчешће коришћен и врло популаран вид комуникације под називом *IPsec Tunneling*, док на четвртм постоји *TLS/SSL Tunneling Protocol*.

3. ОПИС РЕШЕЊА ПРОБЛЕМА

При самом покретању апликације покреће се серверска страна на локалној *IP* адреси и чека захтев за конектовање клијента (слике 3.1, 3.2).

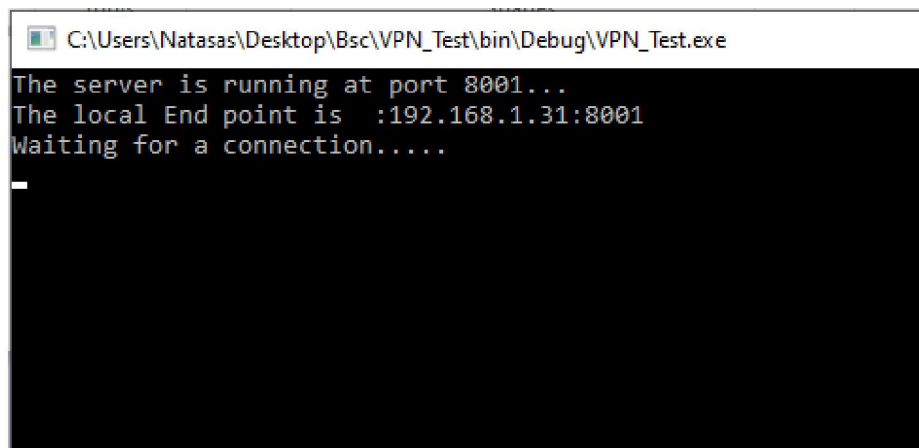
```
try
{
    IPAddress ipAd = IPAddress.Parse("192.168.1.31"); //IPv4 local

    /* Initializes the Listener */
    TcpListener myList = new TcpListener(ipAd, 8001);

    /* Start Listeneting at the specified port */
    myList.Start();

    Console.WriteLine("The server is running at port 8001...");
    Console.WriteLine("The local End point is : " + myList.LocalEndpoint);
    Console.WriteLine("Waiting for a connection.....");
}
```

Слика 3.1 Иницијализација сервера



```
C:\Users\Natasas\Desktop\Bsc\VPN_Test\bin\Debug\VPN_Test.exe
The server is running at port 8001...
The local End point is :192.168.1.31:8001
Waiting for a connection.....
```

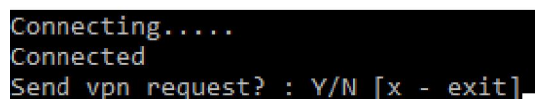
Слика 3.2 Конзолна апликација сервера

Док сервер ослушкује на одређеној адреси и порту, клијент, након покретања, покушава да се повеже са истим подацима (слике 3.3, 3.4).

```
try
{
    TcpClient tcpclnt = new TcpClient();
    Console.WriteLine("Connecting.....");

    tcpclnt.Connect("192.168.1.31", 8001);
}
```

Слика 3.3 Конектовање клијента са већ покренутим сервером



```
Connecting.....
Connected
Send vpn request? : Y/N [x - exit]_
```

Слика 3.4 Конзолна апликација клијента

Када су клијент и сервер успешно повезани, клијент може да шаље захтев да се даља комуникација остварује преко *VPN* конекције уколико је то могуће. Ако *VPN* није укључен, сервер враћа поруку да тренутно није могуће остварити такву везу, међутим ако јесте, тражи нову адресу коју је доделио сам *VPN* провајдер и шаље је клијенту као податак уз помоћ којег се конектује на *VPN* везу (слика 3.5). Након тога искористи нађену адресу тако што покреће сервера на њој који ослушкује за нове клијенте преко *VPN* адресе (слика 3.6).

```
try
{
    var vpn = NetworkInterface.GetAllNetworkInterfaces().First(x => x.Name == "TunnelBear");
    var ip = vpn.GetIPProperties().UnicastAddresses.First(x => x.Address.AddressFamily == AddressFamily.InterNetwork).Address.ToString();

    ASCIIEncoding asen = new ASCIIEncoding();
    s.Send(asen.GetBytes(ip));
    Console.WriteLine("\nSent Acknowledgement");
    OpenVPNServer(IPAddress.Parse(ip));
    break;
}
catch (Exception e)
{
    string str = "VPN is not activated.";
    Console.WriteLine(e.Message);
    ASCIIEncoding asen = new ASCIIEncoding();
    s.Send(asen.GetBytes(str));
    Console.WriteLine("\nSent Acknowledgement");
}
```

Слика 3.5 Добављање *VPN* IP адресе на серверској страни

```
1 reference
public static void OpenVPNServer(IPAddress ipAdr)
{
    try
    {
        /* Initializes the Listener */
        TcpListener myList = new TcpListener(ipAdr, 8001);

        /* Start Listeneting at the specified port */
        myList.Start();

        Console.WriteLine("The server on VPN is running at port 8001...");
        Console.WriteLine("Waiting for a connection.....");

        Socket s = myList.AcceptSocket();
        Console.WriteLine("Connection accepted from " + s.RemoteEndPoint);
    }
}
```

Слика 3.6 *VPN* отварање конекције на серверу

Након тога, клијентска страна преузима поруку и ако добије нову адресу, преко ње поново се конектује на сервер (слике 3.7, 3.7.1). У супротном поново се појављује порука да ли жели послати захтев.


```

Stream stm = tcpclnt.GetStream();

ASCIIEncoding asen = new ASCIIEncoding();
byte[] ba = asen.GetBytes(str);
Console.WriteLine("Transmitting.....");

stm.Write(ba, 0, ba.Length);

byte[] bb = new byte[100];
int k = stm.Read(bb, 0, 100);
string recv = "";
for (int i = 0; i < k; i++)
    recv += Convert.ToChar(bb[i]);
if (recv.ToLower().Equals("x"))
{
    tcpclnt.Close();
    break;
}

Console.WriteLine("VPN IPv4 address: " + recv);
Regex rgx = new Regex(@"((25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9]?)\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9]?)");
MatchCollection result = rgx.Matches(recv);
if(result.Count != 0)
{
    tcpclnt.Close();
    OpenVPNClient(recv);
}

```

Слика 3.7 Пријем поруке на клијентској страни

```

1 reference
public static void OpenVPNClient(string ipAdr)
{
    try
    {
        TcpClient tcpclnt = new TcpClient();
        Console.WriteLine("Connecting.....");

        tcpclnt.Connect(ipAdr, 8001);

        Console.WriteLine("Connected");
        while (true)
        {
            Console.Write("Send message [x - exit]");

            string str = Console.ReadLine();

            Stream stm = tcpclnt.GetStream();

            ASCIIEncoding asen = new ASCIIEncoding();
            byte[] ba = asen.GetBytes(str);
            Console.WriteLine("Transmitting.....");

            stm.Write(ba, 0, ba.Length);

            byte[] bb = new byte[100];
            int k = stm.Read(bb, 0, 100);
            string recv = "";
            for (int i = 0; i < k; i++)
                recv += Convert.ToChar(bb[i]);
            if (recv.ToLower().Equals("x"))
                break;
            Console.WriteLine(recv);
        }
        tcpclnt.Close();
    }
}

```

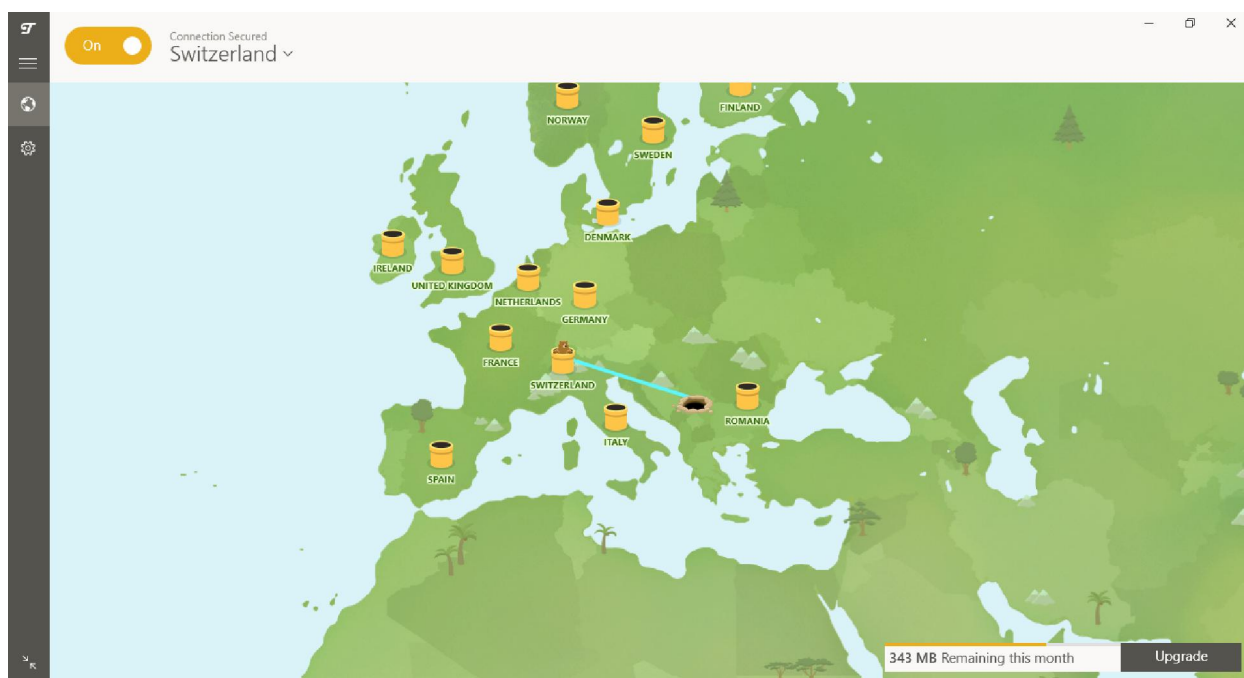
Слика 3.7.1 Отварање *VPN* конекције на страни клијента

Када се успешно повежу обе стране, могу комуницирати преко *VPN* везе, док год клијент не обустави конекцију, при чему се гаси и серверска страна (слика 3.8).

<pre>C:\WINDOWS\system32\cmd.exe The server is running at port 8001... The local End point is :192.168.1.31:8001 Waiting for a connection.... Connection accepted from 192.168.1.31:50156 Request for VPN IP address. Sent Acknowledgement The server on VPN is running at port 8001... Waiting for a connection.... Connection accepted from 10.0.0.60:50157 nova poruka Sent Acknowledgement</pre>	<pre>C:\Users\Natasas\Desktop\Bsc\VPN_Client\bin\Debug\VPN_Client.exe Connecting..... Connected Send vpn request? : Y/N [x - exit]y Transmitting..... VPN IPv4 address: 10.0.0.60 Connecting..... Connected Send message [x - exit] nova poruka Transmitting..... Message succesfully transmitted. Send message [x - exit]</pre>
---	--

Слика 3.8 Сервер – клијент комуникација преко нове адресе

Сам софтвер *VPN* провајдера који је коришћен је приказан на слици 3.9. Интерфејс је интуитиван и једноставан. У горњем левом углу постоји дугме за активирање и деактивирање виртуалне приватне мреже. Када се активира, софтвер аутоматски додели најближи сервер како би конекција била што бржа и квалитетнија. Ипак, ако корисник није задовољан увек може изабрати други сервер који је на располагању и тако „променити” своју локацију. Такође, на андроид уређају апликација функционише исто (слика 3.9.1).



Слика 3.9 Кориснички интерфејс *Tunnel Bear* апликације за *Windows*



Слика 3.9.1 Кориснички интерфејс *Tunnel Bear* апликације за Андроид

4. ПРЕДЛОЗИ ЗА ДАЉА УСАВРШАВАЊА

Конзолна апликација је рађена са циљем утврђивања рада *VPN* мреже. Самим тим апликација је доста једноставна и има пуно простора за унапређивање. С обзиром да је основа, чији део су конектовање на сервер и TCP комуникација, правилно урађена, лако је додати још функционалности. Корисно би било добављање више информација о мрежи, нпр. тачна локација употребљене адресе, оптерећеност сервера или јављање грешке при испаду коришћеног сервера. При слању иницијалног захтева за конектовање на *VPN* мрежу, ако није активирана путем апликације *Tunnel Bear* може се омогућити покретање апликације како би се виртуелна мрежа активирала. Уколико дође до испада или самог деактивирања *VPN* конекције, најбоље би било омогућити кориснику да покуша опет да се повеже или наставак комуникације са сервером на локалној мрежи. Пошто поједини провајдери виртуелне приватне мреже ограничавају проток интернета, тј коришћење саме мреже, биле би корисне додатне информације колико мегабајта је преостало клијенту.

Такође, може се унапредити са конзолне апликације на WPF апликацију, чиме би се омогућило корисницима да интуитивно и лако употребљавају софтвер. Све информације би биле читкије и јасније. С обзиром да је апликацију овог *VPN* провајдера могуће користити и на мобилним уређајима, логика се може пребацити на Андроид платформу уз помоћ *Xamarin* или неке друге платформе.

Основна мана урађеног пројекта јесте зависност софтвера од друге апликације. Тачније, уколико сам корисник не поседује апликацију *VPN* провајдера или не активира виртуелну приватну мрежу из те апликације, сам софтвер не може да обавља функције које су имплементирани. Уколико би се омогућила интеграција ова два софтвера, употреба би била једноставнија.

ЛИТЕРАТУРА

- [1] <https://einfo.biz/vpn-mreza-cemu-sluzi-i-zasto-se-koristi/>
- [2] http://security.hsr.ch/lectures/Information_Security_2/Vorlesungsunterlagen/04-VPN_Notes.pdf
- [3] <https://www.techrepublic.com/article/fix-the-four-biggest-problems-with-vpn-connections/#:~:text=1%3A%20The%20VPN%20connection%20is,the%20most%20common%20VPN%20problem.&text=If%20your%20VPN%20server%20is,Remote%20Access%20service%20is%20running>
- [4] [https://www.calyptix.com/research-2/ssl-vpn-and-ipsec-vpn-how-they-work/#:~:text=IPsec%20VPN%20is%20one%20of,than%20just%20a%20single%20device\).&text=IPsec%20VPNs%20come%20in%20two%20types%3A%20tunnel%20mode%20and%20transport%20mode](https://www.calyptix.com/research-2/ssl-vpn-and-ipsec-vpn-how-they-work/#:~:text=IPsec%20VPN%20is%20one%20of,than%20just%20a%20single%20device).&text=IPsec%20VPNs%20come%20in%20two%20types%3A%20tunnel%20mode%20and%20transport%20mode)
- [5] <https://singipedia.singidunum.ac.rs/preuzmi/41767-virtuelne-privatne-mreze/1846>