

Assignment 6 - IPsec

Section 1: Goals of the Experiment

The goal of this experiment is to learn how IP security protects data on the network layer by comparing Wireshark captures with and without an implemented VPN application. The steps to achieving this goal is to construct a website that is locally hosted, acting as the server, connect a client to the server and capture the communication between client (through VPN) and server using Wireshark.

Section 2: Experiment Setup

Step 1: Download a VPN tunneling application: TunnelBear (Figure1)

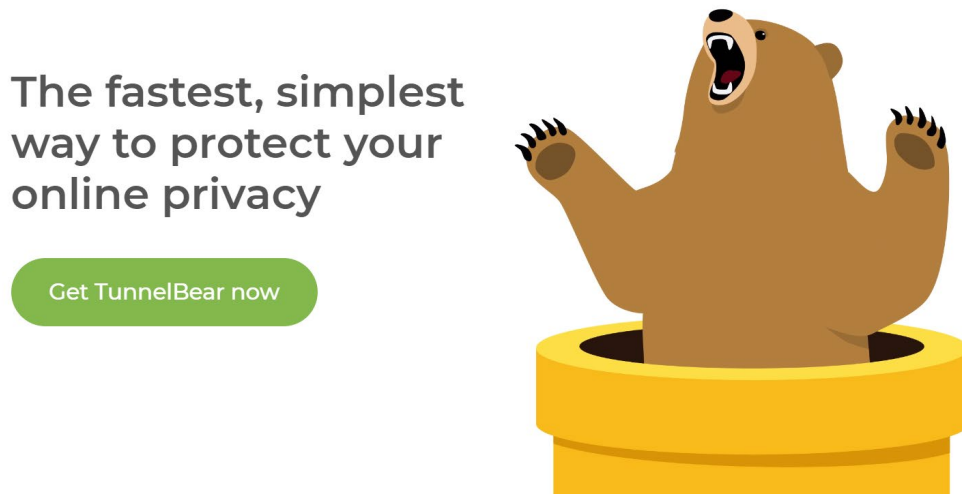
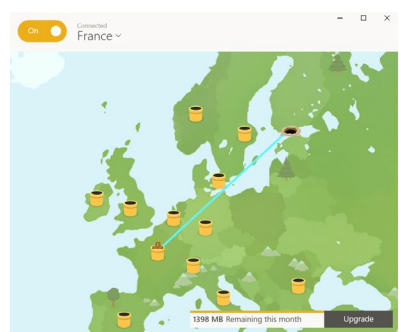


Figure1 : The website to download TunnelBear

An example when browsing from a secure tunnel in France.(Figure 2)

 <p>Figure 2a: Using TunnelBear to switch the location to France.</p>	<pre>ter 區域連線* 1: : Media disconnected cific DNS Suffix . : ter Wi-Fi: cific DNS Suffix . : Address. : 2001:999:32:37:81d0:4d Address. : 2001:999:32:37:2948:a5 Address. : 2001:999:32:37:f954:96 6 Address. : fe80::81d0:4d87:9e76:a : 172.20.10.8 : 255.255.255.240 y : fe80::891:cd35:6302:e9 172.20.10.1</pre> <p>Figure 2b: Internal IP address (Private)</p>	<p>Details for 62.210.99.169</p> <p>IP: 62.210.99.169 Decimal: 1053975465 Hostname: 62-210-99-169.rev.poneytelecom.eu ASN: 12876 ISP: Free SAS Organization: ONLINE SAS Services: Suspected network sharing device Type: Broadband Assignment: Static IP Blacklist: Click to Check Blacklist Status Continent: Europe Country: France 🇫🇷 Latitude: 48.8582 (48° 51' 29.52" N) Longitude: 2.3387 (2° 20' 19.32" E)</p> <p>Figure 2c: External IP address (Public)</p>
--	--	--

Step 2: Access the website we made in assignment 5 without using VPN.

Sarah Tse(748537)

I-Hsin Lin(754961)

This step is included to show the difference between browsing normally and with a VPN. First we ran the program using command line as seen in Figure 3 on the laptop acting as the server. The code can be found in Appendices A-C.

```
C:\Users\sarah\AppData\Local\Programs\Python\Python37-32\Assignments - Copy>py app.py
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 951-006-426
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [07/Apr/2019 14:54:35] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [07/Apr/2019 14:54:36] "GET /favicon.ico HTTP/1.1" 404 -
```

Figure 3: Running web server

The client was able to access the web page by typing in server's internal IP address since both the server and client were on the same network (aalto open).

Wireless LAN adapter Wi-Fi:

```
Connection-specific DNS Suffix . : 
IPv6 Address. . . . . : 2001:708:150:10::2998
Link-local IPv6 Address . . . . . : fe80::6dca:20bc:46e:ea06%20
IPv4 Address. . . . . : 10.100.2.213
Subnet Mask . . . . . : 255.255.192.0
Default Gateway . . . . . : fe80::32f7:dff:fe80:4000%20
                             10.100.0.1
```

C:\Users\sarah\AppData\Local\Programs\Python\Python37-32\Assignments>

← → ↻ ① 不安全 | 10.100.2.213:5000

★ Bookmarks 我的最愛 Google 103@gmail.com

How old are you?

submit

Figure 4a: IP address of server

Figure 4b: Client accessing web page

Step 3: Use Wireshark to capture traffic.

While the client was accessing the site, we ran Wireshark on the server laptop and received TCP and HTTP packets. The capture is normally what we would see when browsing a website without an SSL certificate.

No.	Time	Source	Destination	Protocol	Length	Info	TX Power	RSSI
1	0.000000	10.100.2.213	40.67.248.104	TLSv1.2	97	Application Data		
2	0.066264	40.67.248.104	10.100.2.213	TLSv1.2	179	Application Data		
3	0.114465	10.100.2.213	40.67.248.104	TCP	54	2860 → 443 [ACK] Seq=44 Ack=126 Win=256 Len=0		
4	1.542758	10.100.38.106	10.100.2.213	TCP	66	49909 → 5000 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1		
5	1.543042	10.100.2.213	10.100.38.106	TCP	66	5000 → 49909 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 S...		
6	1.544848	10.100.38.106	10.100.2.213	HTTP	540	GET / HTTP/1.1		
7	1.546142	10.100.38.106	10.100.2.213	TCP	54	49909 → 5000 [ACK] Seq=1 Ack=1 Win=65536 Len=0		
8	1.546351	10.100.2.213	10.100.38.106	TCP	71	5000 → 49906 [PSH, ACK] Seq=1 Ack=487 Win=254 Len=17 [TCP segment of ...		
9	1.546829	10.100.2.213	10.100.38.106	HTTP	692	HTTP/1.0 200 OK (text/html)		
10	1.561190	10.100.38.106	10.100.2.213	TCP	54	49906 → 5000 [ACK] Seq=487 Ack=657 Win=254 Len=0		
11	1.563072	10.100.38.106	10.100.2.213	TCP	54	49906 → 5000 [FIN, ACK] Seq=487 Ack=657 Win=254 Len=0		
12	1.563155	10.100.2.213	10.100.38.106	TCP	54	5000 → 49906 [ACK] Seq=657 Ack=488 Win=254 Len=0		
13	7.181176	10.100.2.213	40.67.248.104	TLSv1.2	97	Application Data		
14	7.249186	40.67.248.104	10.100.2.213	TLSv1.2	179	Application Data		
15	7.385737	10.100.2.213	40.67.248.104	TCP	54	2867 → 443 [ACK] Seq=44 Ack=126 Win=253 Len=0		
16	7.520537	10.100.38.106	10.100.2.213	HTTP	685	POST / HTTP/1.1 (application/x-www-form-urlencoded)		

Apply a display filter ... <Ctrl>/>

Hypertext Transfer Protocol

Line-based text data: text/html (17 lines)

```
<!DOCTYPE html>\n
<html>\n
<head>\n
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR01XCMTq3Xlpa34MD+dh/1fQ784/j6CY/13TQ0hckL...
```

Figure 5: Normal Browsing Wireshark Capture

Sarah Tse(748537)

I-Hsin Lin(754961)

Step 4: Access the website again with VPN.

We kept the website running but the client accessed the site with the VPN but when putting in the external IP address of the server the site was unreachable. With further research, we found that this is because locally hosted websites are inaccessible to computers outside of the network it is being hosted on. This is because when the client sends the packet with the external IP and port number, the router isn't configured to accept the packet at the port. To solve this issue, we utilized ngrok, an application that "exposes local servers behind NATs and firewalls to the public internet over secure tunnels," to forward the client request to the server.



This site can't be reached

86.50.147.0 took too long to respond.

Try:

- [Checking the connection](#)
- [Checking the proxy and the firewall](#)
- [Running Windows Network Diagnostics](#)

ERR_CONNECTION_TIMED_OUT

Reload

```
ngrok by @inconshreveable

Session Status      online
Session Expires     7 hours, 59 minutes
Version             2.3.25
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://be86e567.ngrok.io -> http://localhost:5000
Forwarding           https://be86e567.ngrok.io -> http://localhost:5000

Connections         ttl    opn    rt1    rt5    p50    p90
                   0      0      0.00   0.00   0.00   0.00
```

Figure 6b: Ngrok application

To start the port forwarding, we ran ngrok on the server laptop and entered the command "ngrok http 5000". The application will assign a temporary uniform resource locator (URL) that the client can use to access the website. Once we fixed this process, the client laptop was able to access the website outside of the network with the VPN as seen in Figure 6e.

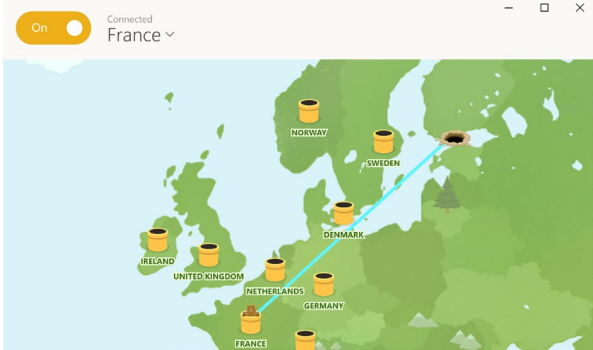


Figure 6c: Turning on TunnelBear VPN on client laptop

whats my ip

Web Images Videos News Answer

Your IP address is 62.210.99.18 in [Paris, Ile-de-France, France \(75010\)](#)

Figure 6d: Client IP address with VPN

Sarah Tse(748537)
I-Hsin Lin(754961)

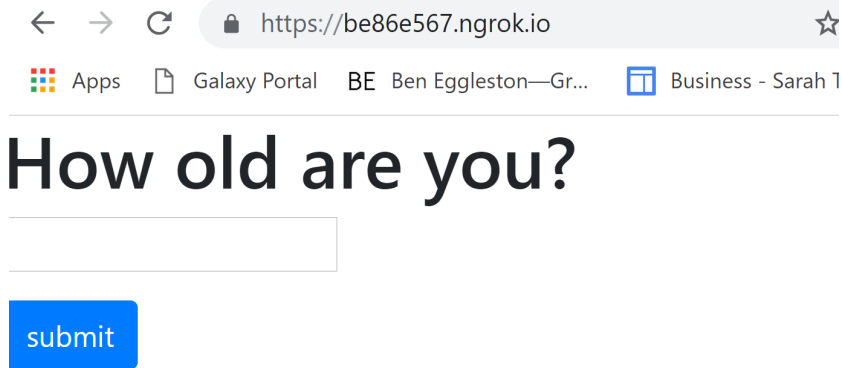


Figure 6e: Client accessing website with VPN after port forwarding

Step 5: Use Wireshark to capture traffic

While the client was connected to the server, we ran Wireshark on the server side but there were no packets that had an IP address that matched the client's VPN assigned IP address. Objectively, we were looking for a packet that has the VPN assigned IP of the client (62.210.99.18) and the external IP of the server (86.50.147.0) to show that the client successfully connected to the server. Figure 7a is a screen capture of the Wireshark trace on the server side while the client was accessing the website.

A screenshot of the Wireshark network traffic capture interface. The top section shows a list of captured packets with columns for No., Time, Source, Destination, Protocol, Length, and Info. The bottom section shows the details of the selected packet (Frame 30), including Ethernet II, Internet Protocol Version 6, Transmission Control Protocol, and Transport Layer Security.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.100.2.213	3.120.198.117	TLSv1.2	117	Application Data
2	0.035338	3.120.198.117	10.100.2.213	TLSv1.2	117	Application Data
3	0.075698	10.100.2.213	3.120.198.117	TCP	54	10430 → 443 [ACK] Seq=64 Ack=64 Win=257 Len=0
4	0.180028	10.100.2.213	74.125.140.189	UDP	65	61261 → 443 Len=23
5	0.231377	74.125.140.189	10.100.2.213	UDP	62	443 → 61261 Len=20
6	5.184603	2603:1026:7:14::2	2001:708:150:10::2998	TLSv1.2	109	Application Data
7	5.185626	2603:1026:7:14::2	2001:708:150:10::2998	TLSv1.2	1157	Application Data
8	5.185858	2001:708:150:10::2998	2603:1026:7:14::2	TCP	74	10405 → 443 [ACK] Seq=1 Ack=1119 Win=253 Len=0
9	5.186249	2603:1026:7:14::2	2001:708:150:10::2998	TLSv1.2	109	Application Data
10	5.226768	2001:708:150:10::2998	2603:1026:7:14::2	TCP	74	10405 → 443 [ACK] Seq=1 Ack=1154 Win=253 Len=0
11	5.544989	108.177.15.189	10.100.2.213	UDP	82	443 → 52648 Len=40
12	5.559142	10.100.2.213	108.177.15.189	UDP	70	52648 → 443 Len=28
13	6.067743	2603:1026:7:16::2	2001:708:150:10::2998	TLSv1.2	117	Application Data
14	6.107658	2001:708:150:10::2998	2603:1026:7:16::2	TCP	74	10516 → 443 [ACK] Seq=1 Ack=44 Win=258 Len=0
15	6.381035	216.58.201.238	10.100.2.213	UDP	82	443 → 50402 Len=40
16	6.381036	216.58.201.238	10.100.2.213	UDP	61	443 → 50402 Len=19

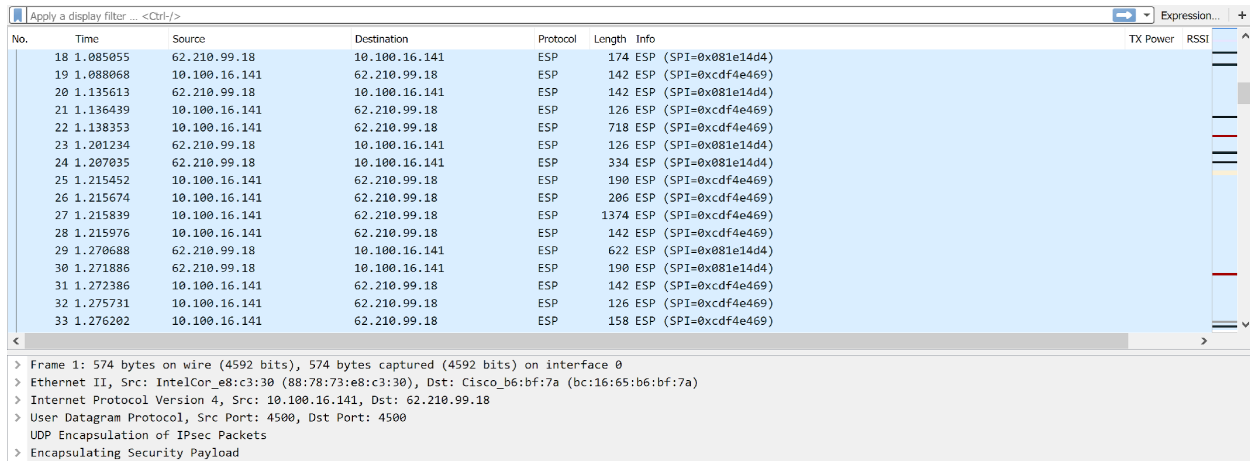
> Frame 30: 102 bytes on wire (816 bits), 102 bytes captured (816 bits) on interface 0
> Ethernet II, Src: Cisco_fa:40:00 (30:f7:0d:fa:40:00), Dst: IntelCor_e8:c3:30 (88:78:73:e8:c3:30)
> Internet Protocol Version 6, Src: 2a03:2880:f00a:8:face:b00c:0:2, Dst: 2001:708:150:10::2998
> Transmission Control Protocol, Src Port: 443, Dst Port: 10397, Seq: 1, Ack: 33, Len: 28
> Transport Layer Security

Figure 7a: Server side Wireshark capture

On the client side, we also ran Wireshark and only saw communication between the VPN server(62.210.99.18) and the client's internal IP address(10.100.16.141). Figure 7b does show that the Wireshark trace captured the Encapsulated Secure Payload(ESP) packets but only between the VPN server and client.

Sarah Tse(748537)

I-Hsin Lin(754961)



No.	Time	Source	Destination	Protocol	Length	Info
18	1.085055	62.210.99.18	10.100.16.141	ESP	174	ESP (SPI=0x081e1d4d)
19	1.088068	10.100.16.141	62.210.99.18	ESP	142	ESP (SPI=0xcdf4e469)
20	1.135613	62.210.99.18	10.100.16.141	ESP	142	ESP (SPI=0x081e1d4d)
21	1.136439	10.100.16.141	62.210.99.18	ESP	126	ESP (SPI=0xcdf4e469)
22	1.138353	10.100.16.141	62.210.99.18	ESP	718	ESP (SPI=0xcdf4e469)
23	1.201234	62.210.99.18	10.100.16.141	ESP	126	ESP (SPI=0x081e1d4d)
24	1.207035	62.210.99.18	10.100.16.141	ESP	334	ESP (SPI=0x081e1d4d)
25	1.215452	10.100.16.141	62.210.99.18	ESP	190	ESP (SPI=0xcdf4e469)
26	1.215674	10.100.16.141	62.210.99.18	ESP	206	ESP (SPI=0xcdf4e469)
27	1.215839	10.100.16.141	62.210.99.18	ESP	1374	ESP (SPI=0xcdf4e469)
28	1.215976	10.100.16.141	62.210.99.18	ESP	142	ESP (SPI=0xcdf4e469)
29	1.270688	62.210.99.18	10.100.16.141	ESP	622	ESP (SPI=0x081e1d4d)
30	1.271886	62.210.99.18	10.100.16.141	ESP	190	ESP (SPI=0x081e1d4d)
31	1.272386	10.100.16.141	62.210.99.18	ESP	142	ESP (SPI=0xcdf4e469)
32	1.275731	10.100.16.141	62.210.99.18	ESP	126	ESP (SPI=0xcdf4e469)
33	1.276202	10.100.16.141	62.210.99.18	ESP	158	ESP (SPI=0xcdf4e469)

> Frame 1: 574 bytes on wire (4592 bits), 574 bytes captured (4592 bits) on interface 0
> Ethernet II, Src: IntelCor_e8:c3:30 (88:78:73:e8:c3:30), Dst: Cisco_b6:bf:7a (bc:16:65:b6:bf:7a)
> Internet Protocol Version 4, Src: 10.100.16.141, Dst: 62.210.99.18
> User Datagram Protocol, Src Port: 4500, Dst Port: 4500
 UDP Encapsulation of IPsec Packets
 > Encapsulating Security Payload

Figure 7b: Client Side Wireshark trace

Section 3: Results and Conclusion

In addition to the experiment detailed above, we tried running the server through the VPN to see if that would make a difference but got the same results as the client VPN. The Wireshark traces show no communication between the client (with VPN IP address) and server. This is due to the limited functions of Wireshark on a Windows computer since we can only track traffic coming in and out of the laptop. After searching for some information, we found that since Wireshark uses WinPcap(windows packet capture) on Windows, and some third-party (or even some standard) VPN software doesn't work with WinPcap. This might be the problem. We suspect that with a Linux based operating system that has "monitor" mode we could possibly track more traffic on the network. Despite this problem, we were still able to see the ESP packet on the client side and learned how secure the using VPN can be.

Section 4: Appendices

Appendix A - Unsecure Backend Python Code (app.py)

```
from flask import Flask, render_template, request
app = Flask(__name__)

@app.route("/", methods=['GET', 'POST'])
def send():
    if request.method == 'POST':
        age = request.form['age']

        return render_template('age.html', age=age)

    return render_template('index.html')

if __name__ == "__main__":
    app.run(debug=True, host='0.0.0.0')
```

Appendix B - Index.html

```
<!DOCTYPE html>
<html>
<head>
```

Sarah Tse(748537)

I-Hsin Lin(754961)

```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap
.min.css" integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
<title>Page Title</title>
</head>
<body>

<h1>How old are you?</h1>
<form method = "POST" action="/">
    <div class = "form-group">
        <input type = "text" name="age">
    </div>
    <input class="btn btn-primary" type="submit" value="submit">
</form>
</body>
</html>
```

Appendix C - age.html

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap
.min.css" integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
<title>Page Title</title>
</head>
<body>

    <h1> Your age is {{ age }}</h1>
</body>
</html>
```