ELEC-C7420 Basic Principles in Networking
Part II: Security

Assignment V: IPsec & VPN

Members: Beatriz Glaser
Thuy Määttä

1. Goals of the experiment

The goal of this experiment is to learn how to establish secure communication between a server and client, as well as get a deeper understanding of VPN, its implementation and benefits. It is important to familiarise ourselves with IPsec and VPN, because of their current significance and popularity on the field. Being able to establish secure communication is essential for the exchange of any information between individuals, companies and other purposes. Additionally, this experiment further solidifies our skills using Wireshark and VPN, which are tools commonly used in today's market.

2. Experimental setup

2.1. Setting up port forwarding

We first set up a port forwarding. We used ngrok as a tool to create a secure tunnel, port forwarding with ngrok, which allows the external client to access the host server script without revealing the host's IP address and port. The server script can then be run locally.

2.1.1. Download the ngrok from https://ngrok.com/
2.1.2. Follow the instruction to install and setup ngrok
2.1.3. From Terminal/Command Prompt, setup the port forwarding

```
Book-Pro ~ % ngrok tcp 5500
```

In this case, we chose the port number: 5500 from local host
2.1.4. Ngrok gave the information to implement in the client-socket

```
Session Status                online
Account                       sonnsons.mimi@gmail.com (Plan: Free)
Version                       3.2.2
Region                        Europe (eu)
Latency                       26ms
Web Interface                 http://127.0.0.1:4040
Forwarding                    tcp://2.tcp.eu.ngrok.io:13728 -> localhost:5500

Connections                   ttl     opn     rt1     rt5     p50     p90
                              11      0       0.01    0.00    222.30  1066.08
```

2.2. Implementing server-client socket script

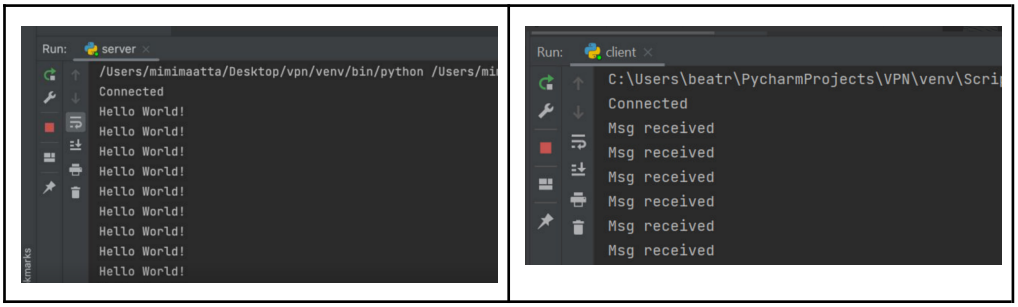
2.2.1. Server script
Localhost on server: '127.0.0.1'
Port: 5500

2.2.2. Client script
Client host is the ngrok generated host name: '2.tcp.eu.ngrok.io'
Port is the generated ngrok port number: '13728'

3. Results

3.1. Establishing basic server-client connection

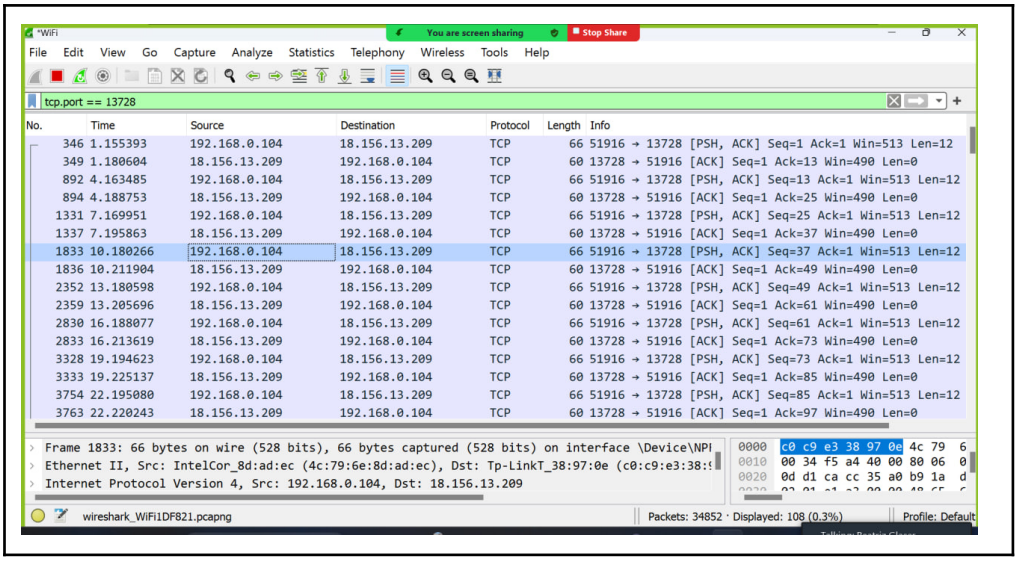
Once the server and client scripts are completed following the steps described above (paying close attention to the respective addresses and ports), the client should be able to successfully connect to the server. For this experiment, this is done with two different machines. The server and client programs are shown by the following figures, respectively.



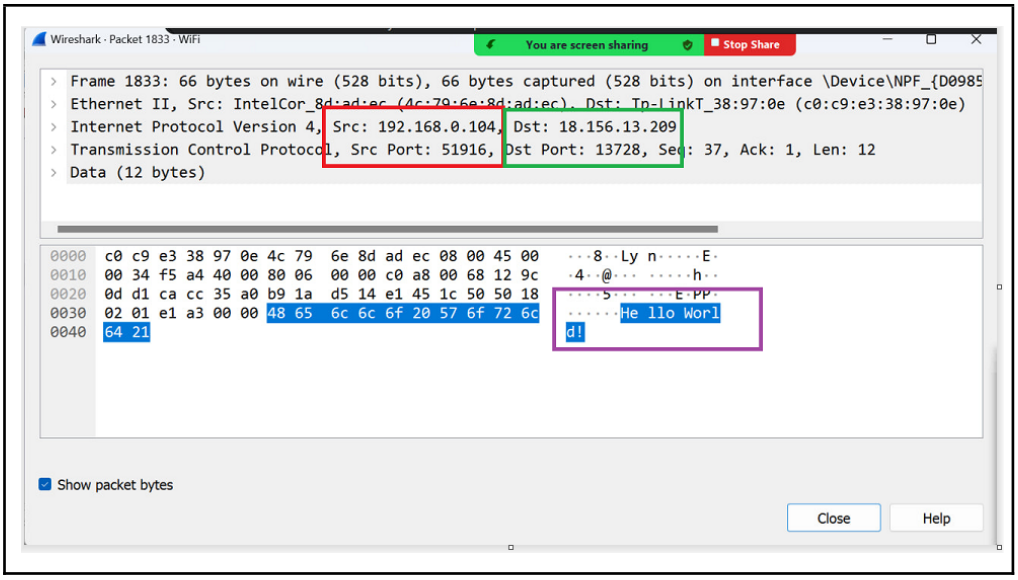
As depicted by the images, once the connection between client and server is established, both print out a "connected" message. The client then begins to send "Hello World!" messages to the server on a loop with 3 second time breaks. Every time the server receives the message, it informs the client, which then prints out the "Msg received" message. The scripts to both server and client can be found on the annex section of this report.

3.2. Capturing connection using WireShark

The server-client connection can be seen using the WiFi capture in WireShark. The following image shows the message transition. We filtered ("tcp.port == 13728") the packets shown by the ngrok port to filter out any irrelevant transitions happening on the wifi network.



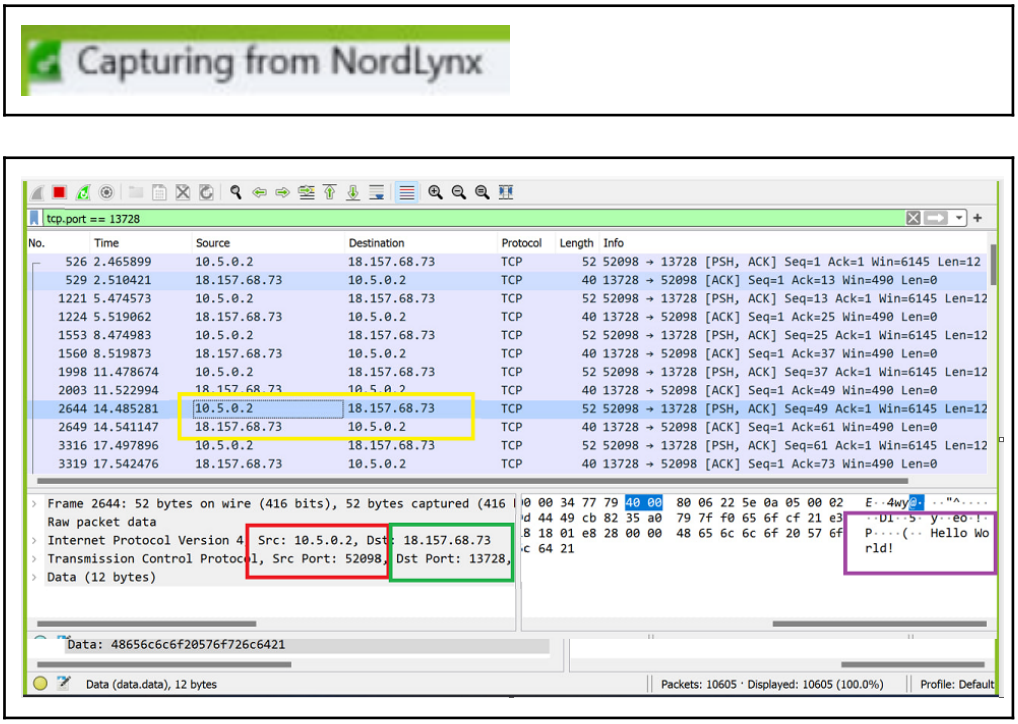
From the image we can also observe that the transactions are happening twice every three seconds. The first is the “Hello World!” message from client (IP address ending in 104) to server (ngrok address ending in 209), and the second is the “Msg received” message from server to client. This loop happens every three seconds (1, 4, 7, 10…) as indicated by the script. The following image further depicts the message highlighting the source and destination addresses and ports, as well as the message.



3.3. Connecting using VPN

3.3.1. Using NordLynx capture

We then connected the client machine to NordVPN. The server-client connection is performed the same way as described in 3.1, however, when checking the transmissions using WireShark we got a different result. When using NordVPN, we gain access to the NordLynx capture which shows the transmissions happening within the new IP client address and the server.

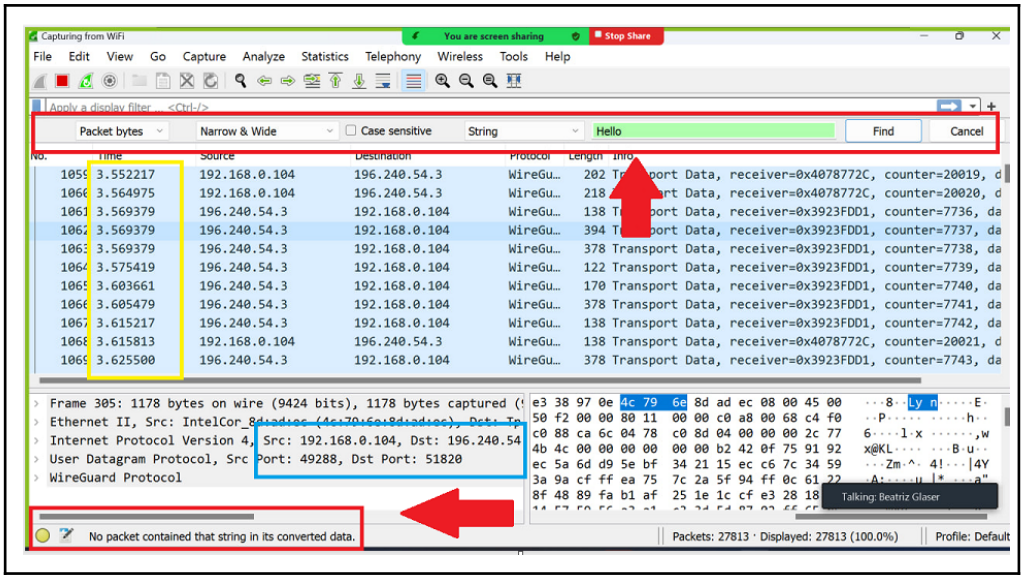


As we can observe from the image, the source IP address and port are different, but the destination remains the same. We can also see that the exchange is happening every three seconds, exactly like before. The VPN connection is masking the client IP address.

3.3.2. Using WiFi capture

The NordLynx capture is only available exclusively to the client computer that is using NordVPN, hence why the messages and destination address are shown. However, we get a very different result when looking at the WiFi capture (which is available to anyone connected to the router).

Now with the VPN, the WiFi capture is being overflowed with transmissions between the client machine (...104) and the new IP address. These transmissions happen multiple times per second and all of it is encrypted. You cannot see any packets destined to the server port (13728), nor can you filter out any “Hello World!” message. The attempt to filter out is indicated by red arrows on the following image.



From this we can conclude that the tunnelling and protection provided by the VPN makes it impossible for any external parties to use applications such as WireShark to identify and read the communication happening between client and server. The packets and transmission itself are unidentifiable, as the only communication seen is between the client's machine and the secure IP address appointed by the NordVPN.

Interestingly enough, when using NordLynx we observed a different source IP address (10.5.0.2), than the one seen on the WiFi capture (196.240.54), even though we had not changed the VPN connection on the NordVPN app. After different tries/tests and research, we noticed that the address shown by NordLynx refers to a private IP address and the WIFI capture to a public one. Screenshots of our findings are shown in the annex section.

Lastly, to get a better understanding of the process, while doing the experiment we also got to learn about different VPN protocols (such as OpenVPN, WireGuard, SSL).

4. Conclusion

In this experiment we took our "server-client" implementation knowledge to the next level by making the communication between them secure. Throughout the process we also gained a better understanding of what VPN is and how to use it to our advantage when it comes to networking. The transferring of information and data securely is crucial to safeguard people- and companies' privacies.

5. Annex

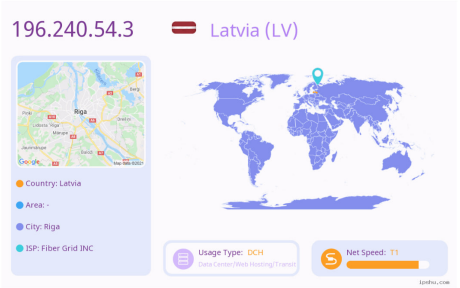
5.1. Server python script

```python
import socket
import time


socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_ip = '127.0.0.1'
port = 5500
socket.bind((server_ip, port))
socket.listen()
conn, addr = socket.accept()
print('Connected')
#print("Client from address {} connected".format(addr))

while True:
    print(conn.recv(1024).decode('utf-8'))
    conn.send('Msg received'.encode('utf-8'))
```

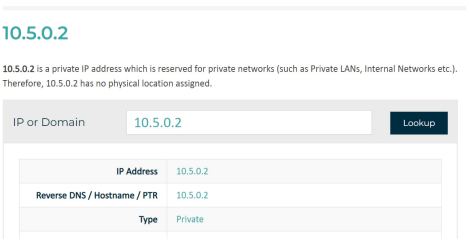
5.2. Client python script

```python
import socket
import time


socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
ip = '2.tcp.eu.ngrok.io'
port = 13728
socket.connect((ip, port))
print('Connected')

while True:
    socket.send("Hello World!".encode('utf-8'))
    print(socket.recv(1024).decode('utf-8'))
    time.sleep(3)
```

5.3. IP addresses observed when using VPN



https://en.ipshu.com/ipv4/196.240.54.3



https://whatismyip.live/ip/10.5.0.2