ECOLE POLYTECHNIQUE DE BRUXELLES

IRELE - MA1 Electrical Engineering

ELEC-H417

Report Lab 4
Virtual Private Network (VPN)

Authors :

Amaury ARICO

Alexis BOLLENGIER

Emmeran COLOT

Sefa GÖNEN

Professor :

Jean-Michel DRICOT

Assistant :

Navid LADNER

Academic year :

2024 - 2025

Contents

1

Mission 1 - Clear traffic observation

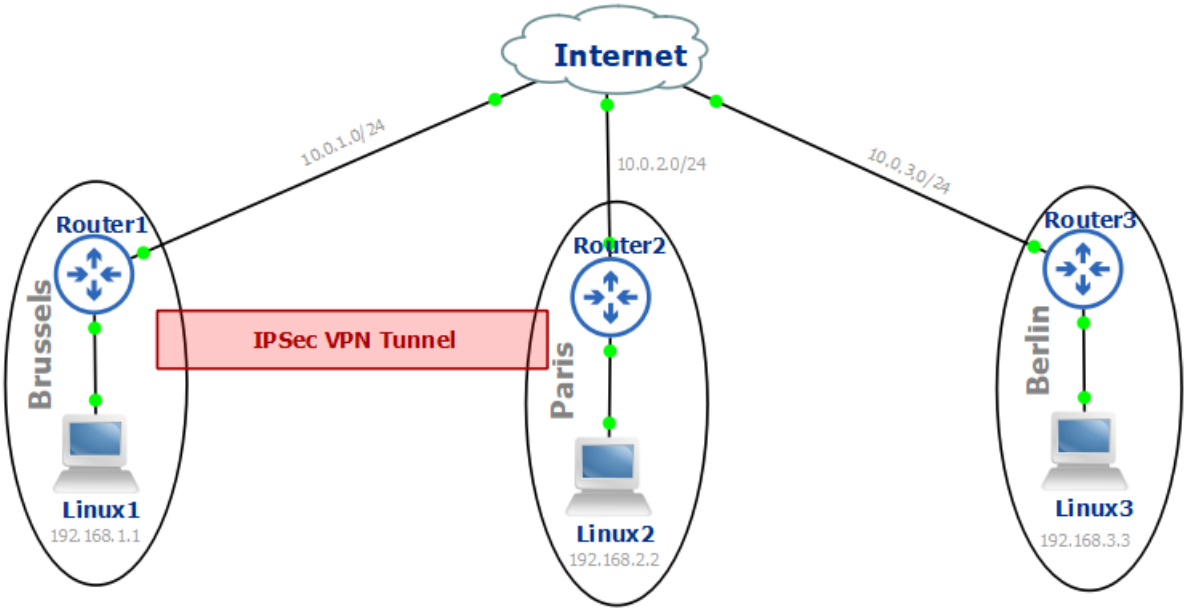

Figure 1.1: Topology of the network for the lab

1.1 Question - TCP 3-way handshake procedure

TCP 3-way handshake procedure

When you send your "Hello world!" message, you can observe 7 packets in Wireshark. Put your Wireshark capture in the report and highlight the payload value (i.e. the cleartext message). Could you explain why and what are all these packets ?

The payload [1] is at the end (*layer 7* data) of the 4^{th} packet. To ensure that the message is received, it is sent in **TCP** and this is the reason of the number of packet sent (7) despite the fact that the content of the message was only 28 bytes.

Let's analyze each request:

1. **Linux1 → Linux2**, wanting to instantiate the connection with the **SYN** flag raised (for synchronization)

[1] The payload was sent from Linux1 (*192.168.1.1*) to Linux2 (*192.168.2.2*) and contained "Network architecture is fun"

2. **Linux2 → Linux1**, acknowledging (**ACK** flag + *Ack=1*) the reception of the synchronization (**SYN** flag) request
3. **Linux1 → Linux2**, acknowledging (**ACK** flag + *Ack=1*) the acknowledgment of Linux2
4. **Linux1 → Linux2**, pushing the payload (**PSH** flag) and still acknowledging the reception of the 2^{nd} packet (**ACK** flag + *Ack=1*)
5. **Linux1 → Linux2**, informing it of the finish of the transmission (**FIN** flag). It is necessary because if the payload is too long, it would have needed multiple push requests. This packet still acknowledges the last received packet (**ACK** flag + *Ack=1*)
6. **Linux2 → Linux1**, acknowledging the reception of the 4^{th} packet (**ACK** flag + *Ack=29*)
7. **Linux2 → Linux1**, acknowledging the reception of the 5^{th} packet (**ACK** flag + *Ack=30*)

To finish this analysis, let's just recall how the *acknowledgment number* is determined: when sending a packet of *sequence number N* and payload of length *M*, the expected acknowledgment response number is $M+N$. This number corresponds also to the sequence number of the next packet to send.

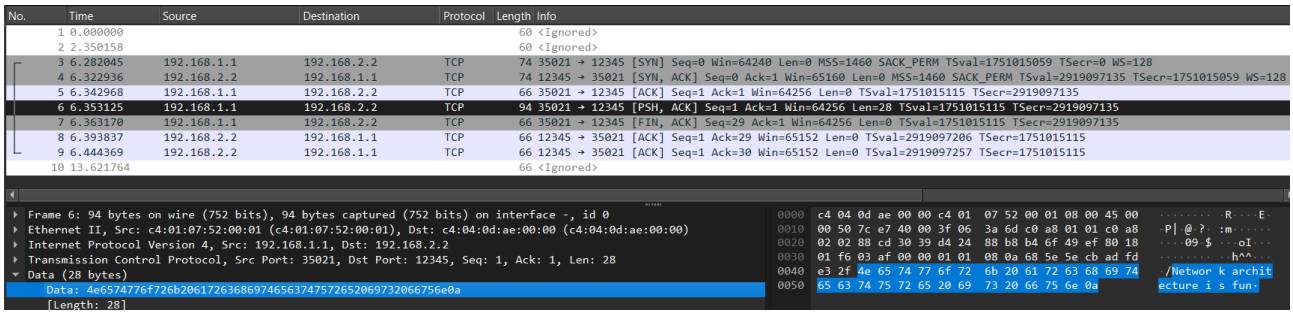

Figure 1.2: Wireshark capture of an *echo* request

1.2 Question - Sequence and acknowledgement number

Sequence and acknowledgement number

Explain what are the sequence number and acknowledgement number flags used for.

The explanation at the end of the previous question can be studied more in depth:
The **sequence number** is a value linked to the sender side[2] that indicates the packet that he is sending. This number is in fact a pointer that is set to 0 at the start of the communication.
The **acknowledgment number** is a value linked to the receiver side that indicates the next packet's sequence number that the receiver expects to receive. The calculation has been explained in the previous section (seq + length).
It can be seen on the Wireshark screenshot **??** that Linux1 send the packet with sequence number 29 without waiting for the acknowledgment number 29 to avoid waiting twice the travel time. In the case this acknowledgment never arrived, Linux1 would have sent the packet with sequence number 1 again.

[2] both PCs are senders and receivers

2 Mission 2 - VPN configuration

2.1 Question - Cryptographic parameters

Cryptographic parameters

Explain briefly each of these policy lines (i.e. What is AES ? What is a hash function and what could it be used for in this case ? What does pre-shared key means ? What is Diffie-Hellman ? What is the liftetime used for ?)

- **AES encryption:** transform the data in *cyphertext* in a way that is resistant to brute-force, making it unreadable for anyone intercepting it. It uses the same key for encryption and decryption and is based on *XORing*, substituting, shifting and mixing operations.
  It is here used to make sure that the data cannot be read.
- **SHA hash function** *(or Secure Hash Algorithm)***:** a hash function converts a message in a fixed-size output in a deterministic way. SHA has the particularity that even if a bit of the original data is flipped, the output of the hash is entirely different.
  It is here used to make sure that the data has not been modified during transmission.
- **Pre-shared key authentication:** verifies identity using a shared secret key known by both parties. This is used during the initial setup of the communication.
- **Group 2 Diffie-Hellman:** allows to share keys over an unsecured channel. The choice of group 2 is a good compromise between security and computation time.
  Note that it is not used to share the authentication key as it is already known by the two correspondents.
- **Lifetime of 86400:** Sets the duration of the security association to 24 hours. After this time, both parts must check the identity of the other again (for example by sending the authentication key).

2.2 Bonus question - Why AES

Why AES

There is two family of encryption algorithms (symmetric and asymmetric). Which family does AES belong to ? Why do we use this family in our case, what is the main benefit ?

As mentioned before, it is a symmetric encryption method, meaning that the same key is used on both sides (for encryption and decryption). The benefit of it is that it is faster to decrypt/encrypt and easier to use than asymmetric encryption.

3

Mission 3 - Encrypted traffic observation

3.1 Question - Wireshark Analysis

Wireshark Analysis

Put both Wireshark captures of your communications with Paris and Berlin in your report and highlight in both the secret message send (encrypted or not). Put them in parallel and explain the differences between the message that used the VPN (aka. towards Paris) and the one that didn't (aka. towards Berlin).

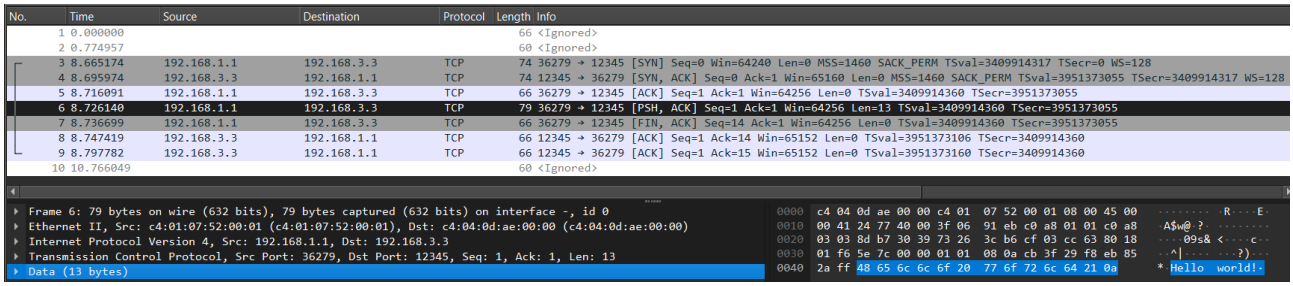

Figure 3.1: **Berlin**: Wireshark capture of the *echo* request without the VPN

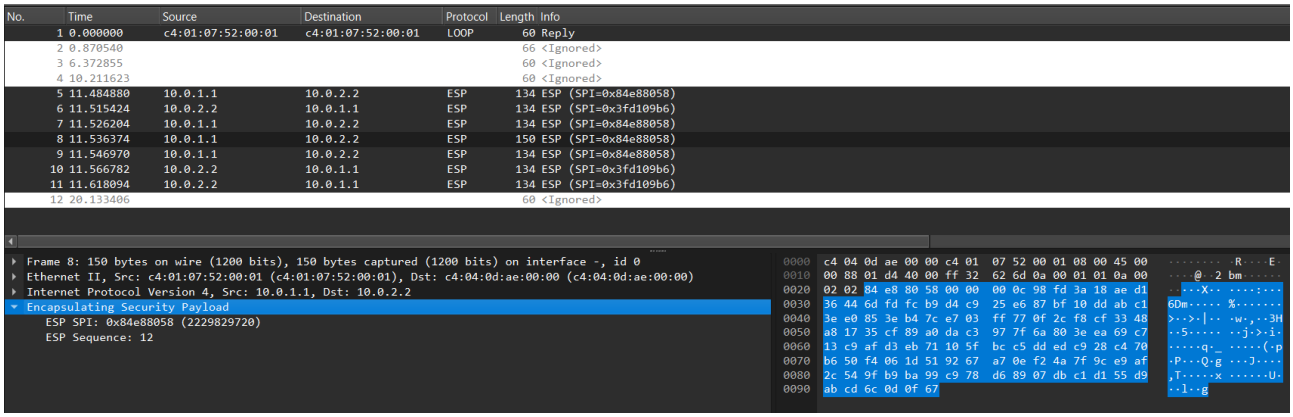

Figure 3.2: **Paris**: Wireshark capture of the *echo* request through the VPN channel

The main difference between the two situations here is that it is impossible for us to retrieve the payload [1] when the payload is sent through the VPN channel as it is encrypted.

Because the **IPSec** has been configured in tunnel mode, even the destination has been encrypted between Router1

[1] "Hello world!"

and Router2, which means that we cannot be sure of the destination of the packet in fig **??**.
Finally the length of the packages in fig **??** is smaller as the use of a VPN will add new headers and new trailers:

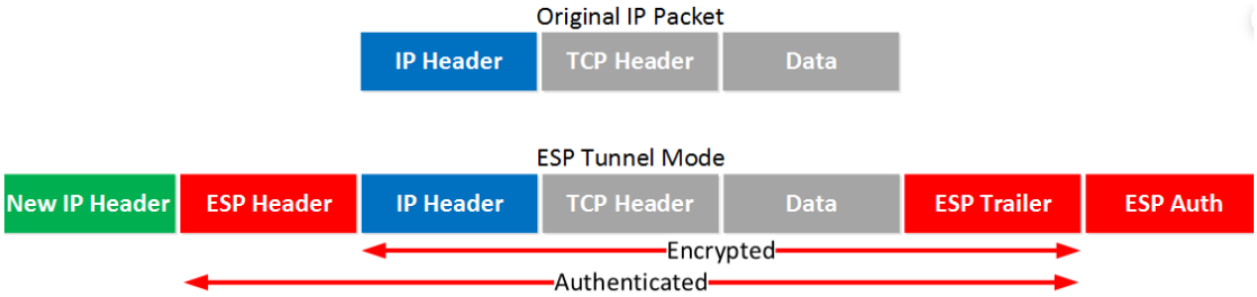

Figure 3.3: Wireshark capture of the *echo* request through the VPN channel

3.2 Question - VPN Uses

VPN Uses

What are the usages of VPN ?

The usage of a VPN make the content inside of the sent packages unreadable and unchangeable by any malicious actor thanks to the protection (encryption + hash). The big difference between a VPN and a "classical" encrypted channel is that it is the whole packet that is encrypted here. This means that even the emitter/receiver addresses are hidden. This implies that is used for security reasons, for example when using a public Wi-Fi.
It can also be used to bypass firewalls as the destination in the header is just the exit of the VPN channel and not the real destination of the packet.

3.3 Bonus question - VPN Security

VPN Security

Is VPN really a good way to secure your communication ? Why ? If it is not the case, what could you do instead (or in addition) ?

The provider of the VPN can still see what you are sending in it and might store/sell this data so you must trust the provider. Once the packets have exited the VPN and are on their way to their real destination, they are no longer protected.
A way to prevent it is to encrypt the packets you send through the VPN with an *end to end encryption* (E2EE) such that the content of the packet stays encrypted until the last hop on the network.