

Al-Najah National University  
Department of Engineering and Information technology  
Computer Network and Information Security

# IPSec Basic Configuration

Mohammed Adnan  
Instructor: Dr. Ahmed Awad



27/3/2021

# 1 Abstract

The purpose of this experiment is to implement a simplified version of IPsec to demonstrate the integration of cryptography algorithms in TCP/IP protocols.

# 2 Introduction

In computing, Internet Protocol Security (IPsec) is a secure network protocol suite that authenticates and encrypts the packets of data to provide secure encrypted communication between two computers over an Internet Protocol network. It is used in virtual private networks (VPNs).

IPsec includes protocols for establishing mutual authentication between agents at the beginning of a session and negotiation of cryptographic keys to use during the session. IPsec can protect data flows between a pair of hosts (host-to-host), between a pair of security gateways (network-to-network), or between a security gateway and a host (network-to-host). IPsec uses cryptographic security services to protect communications over Internet Protocol (IP) networks. It supports network-level peer authentication, data-origin authentication, data integrity, data confidentiality (encryption), and replay protection.

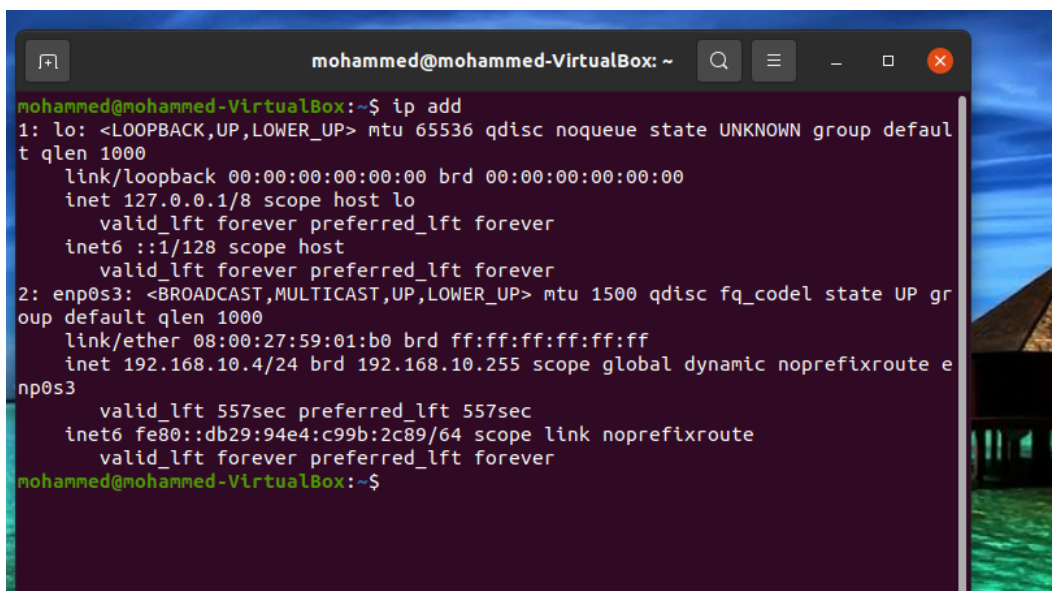
The initial IPv4 suite was developed with few security provisions. As a part of the IPv4 enhancement, IPsec is a layer 3 OSI model or internet layer end-to-end security scheme. In contrast, while some other Internet security systems in widespread use operate above layer 3, such as Transport Layer Security (TLS) that operates at the Transport Layer and Secure Shell (SSH) that operates at the Application layer, IPsec can automatically secure applications at the IP layer. The IPsec is an open standard as a part of the IPv4 suite. IPsec uses the following protocols to perform various functions:

Authentication Headers (AH) provides connectionless data integrity and data origin authentication for IP datagrams and provides protection against replay attacks. Encapsulating Security Payloads (ESP) provides confidentiality, connectionless data integrity, data-origin authentication, an anti-replay service (a form of partial sequence integrity), and limited traffic-flow confidentiality. Internet Security Association and Key Management Protocol (ISAKMP) provides a framework for authentication and key exchange, with actual authenticated keying material provided either by manual configuration with pre-shared keys, Internet Key Exchange (IKE and IKEv2), Kerberized Internet Negotiation of Keys (KINK), or IPSECKEY DNS records. The purpose is to generate the Security Associations (SA) with the bundle of algorithms and parameters necessary for AH and/or ESP operations.

## 3 Procedure

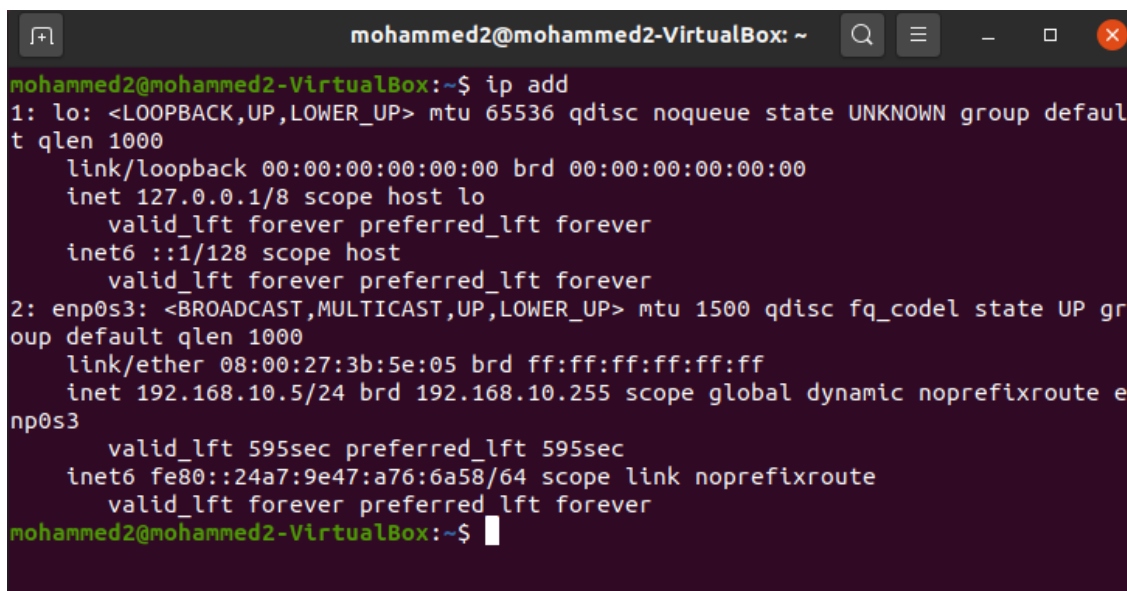
### 3.1 Host-to-Host IPSec Communication

1. By Construct a simple peer-to-peer network using two Linux machines (virtual or standalone). figures 1 2 show the IP address for each host

A terminal window titled 'mohammed@mohammed-VirtualBox: ~' showing the output of the 'ip add' command. The output displays details for the loopback interface 'lo' (127.0.0.1) and the ethernet interface 'enp0s3' (192.168.10.4).

```
mohammed@mohammed-VirtualBox:~$ ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:59:01:b0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.4/24 brd 192.168.10.255 scope global dynamic noprefixroute enp0s3
        valid_lft 557sec preferred_lft 557sec
    inet6 fe80::db29:94e4:c99b:2c89/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
mohammed@mohammed-VirtualBox:~$
```

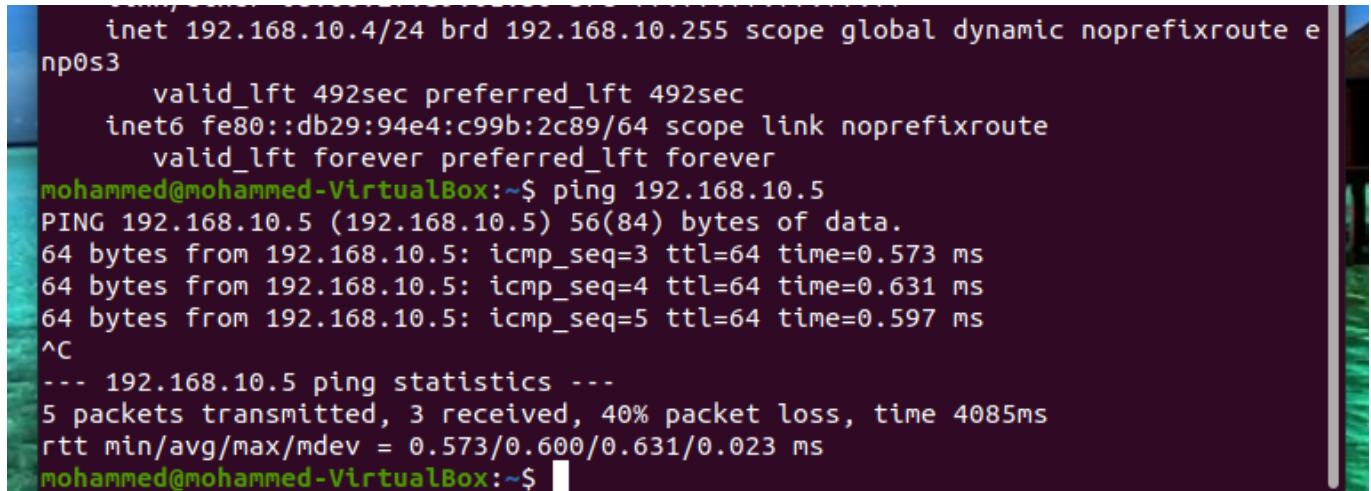
Figure 1: Ip host A address.

A terminal window titled 'mohammed2@mohammed2-VirtualBox: ~' showing the output of the 'ip add' command. The output displays details for the loopback interface 'lo' (127.0.0.1) and the ethernet interface 'enp0s3' (192.168.10.5).

```
mohammed2@mohammed2-VirtualBox:~$ ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:3b:5e:05 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.5/24 brd 192.168.10.255 scope global dynamic noprefixroute enp0s3
        valid_lft 595sec preferred_lft 595sec
    inet6 fe80::24a7:9e47:a76:6a58/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
mohammed2@mohammed2-VirtualBox:~$
```

Figure 2: Ip host B address.

2. To make sure, that each host is reachable by the other, we use the ping command, as shown in figure 3.

A terminal window with a dark background and light-colored text. It shows network configuration for an interface named 'e' (likely eth0). The configuration includes an IPv4 address of 192.168.10.4/24 and an IPv6 address of fe80::db29:94e4:c99b:2c89/64. Below the configuration, a user named 'mohammed' at a host named 'mohammed-VirtualBox' runs the command 'ping 192.168.10.5'. The output shows three successful ping attempts with varying times (0.573 ms, 0.631 ms, 0.597 ms). Finally, the user presses Ctrl-C (^C) to stop the ping, and the terminal displays the 'ping statistics' showing 5 packets transmitted, 3 received, and a 40% packet loss over a total time of 4085ms.

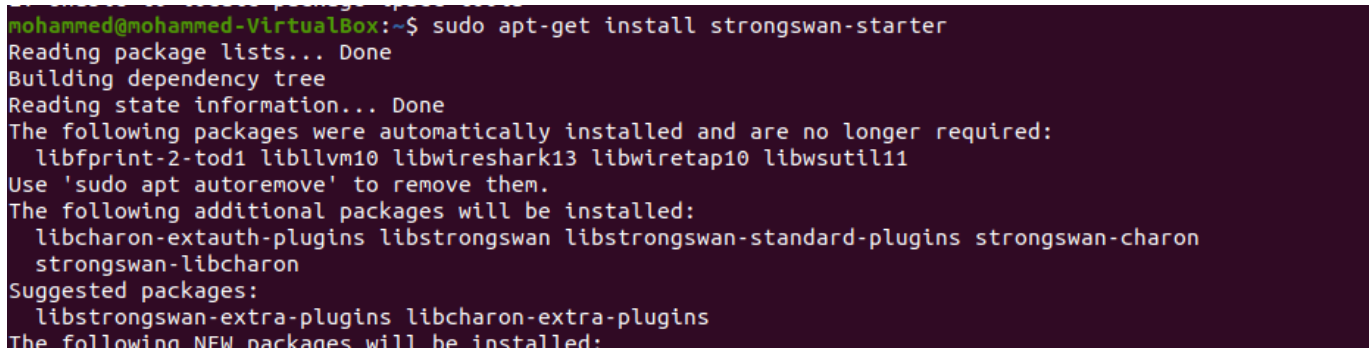
```
inet 192.168.10.4/24 brd 192.168.10.255 scope global dynamic noprefixroute e
np0s3
    valid_lft 492sec preferred_lft 492sec
inet6 fe80::db29:94e4:c99b:2c89/64 scope link noprefixroute
    valid_lft forever preferred_lft forever
mohammed@mohammed-VirtualBox:~$ ping 192.168.10.5
PING 192.168.10.5 (192.168.10.5) 56(84) bytes of data.
64 bytes from 192.168.10.5: icmp_seq=3 ttl=64 time=0.573 ms
64 bytes from 192.168.10.5: icmp_seq=4 ttl=64 time=0.631 ms
64 bytes from 192.168.10.5: icmp_seq=5 ttl=64 time=0.597 ms
^C
--- 192.168.10.5 ping statistics ---
5 packets transmitted, 3 received, 40% packet loss, time 4085ms
rtt min/avg/max/mdev = 0.573/0.600/0.631/0.023 ms
mohammed@mohammed-VirtualBox:~$
```

Figure 3: Ping A to B.

3. **Question:** What are the basic requirements to create host-to-host secure communication with IPsec?

The requirements of a host-to-host connection are minimal, as is the configuration of IPsec on each host. The hosts need only a dedicated connection to a carrier network (such as the Internet) and Red Hat Enterprise Linux to create the IPsec connection.

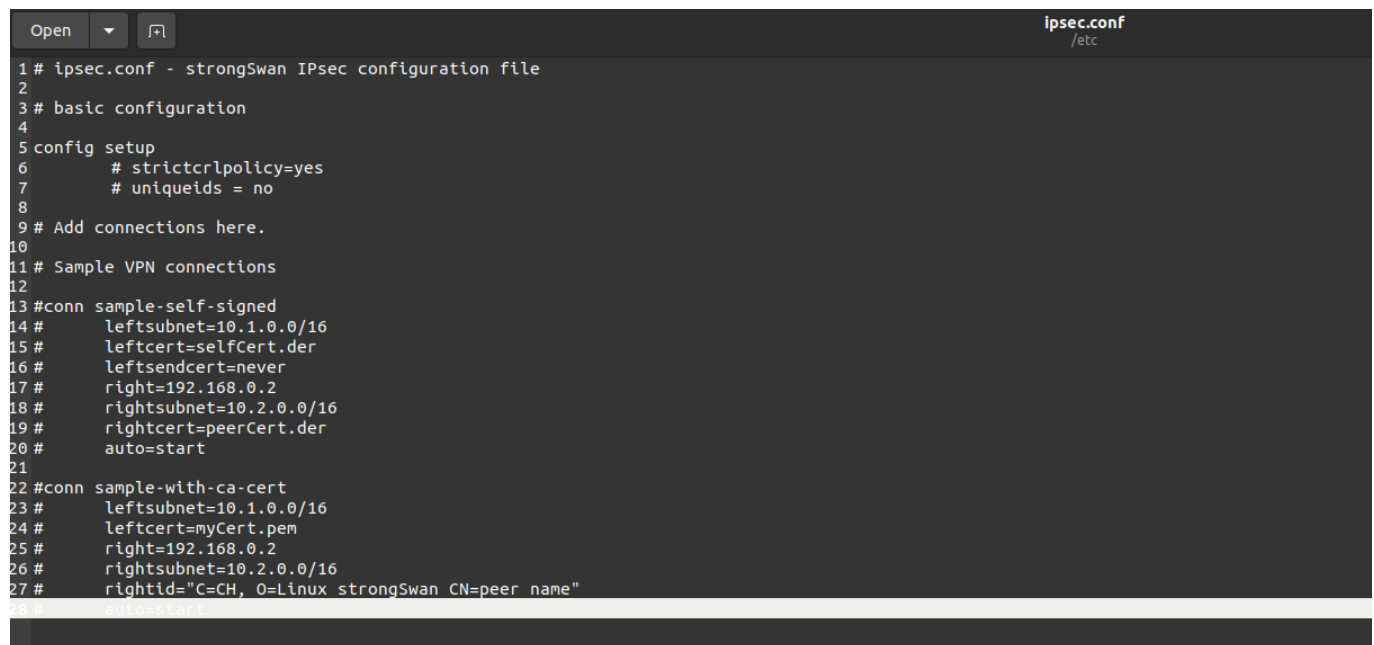
4. Also, installing the IPsec tool with all its needed dependencies using the command `:apt-get install ipsec-tools strongswan-starter` , as shown in figure 4.

A terminal window showing the command 'sudo apt-get install strongswan-starter' being executed. The output shows the package lists being read, the dependency tree being built, and state information being read. It then lists packages that were automatically installed and are no longer required (libfprint-2-tod1, liblvm10, libwireshark13, libwiretap10, libwsutil11) and suggests using 'sudo apt autoremove' to remove them. Next, it lists additional packages that will be installed: libcharon-extra-plugins, libstrongswan, libstrongswan-standard-plugins, strongswan-charon, and strongswan-libcharon. Finally, it suggests installing libstrongswan-extra-plugins and libcharon-extra-plugins. The output ends with 'The following NEW packages will be installed:'.

```
mohammed@mohammed-VirtualBox:~$ sudo apt-get install strongswan-starter
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libfprint-2-tod1 liblvm10 libwireshark13 libwiretap10 libwsutil11
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libcharon-extra-plugins libstrongswan libstrongswan-standard-plugins strongswan-charon
  strongswan-libcharon
Suggested packages:
  libstrongswan-extra-plugins libcharon-extra-plugins
The following NEW packages will be installed:
```

Figure 4: Install ipsec.

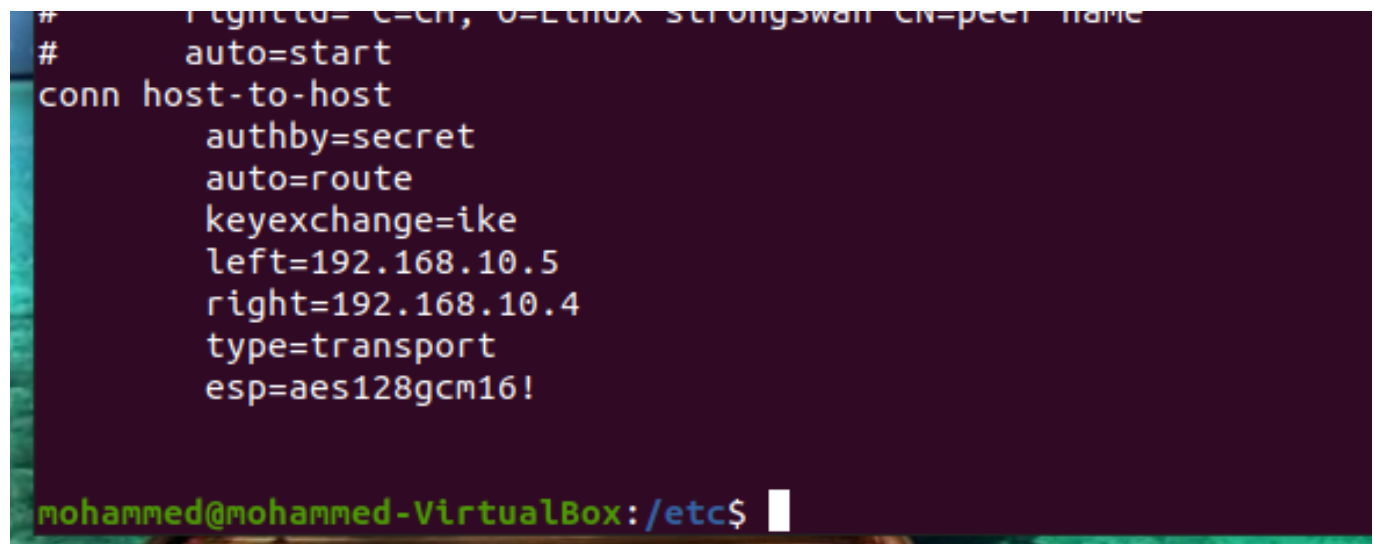
5. After installing it ,we Open the IPsec configuration file ipsec.conf as figure 5 show .

A screenshot of a text editor window showing the contents of the file /etc/ipsec.conf. The window has a title bar with 'ipsec.conf' and '/etc'. The text inside is as follows:

```
1 # ipsec.conf - strongSwan IPsec configuration file
2
3 # basic configuration
4
5 config setup
6     # strictcrlpolicy=yes
7     # uniqueids = no
8
9 # Add connections here.
10
11 # Sample VPN connections
12
13 #conn sample-self-signed
14 #     leftsubnet=10.1.0.0/16
15 #     leftcert=selfCert.der
16 #     leftsendcert=never
17 #     right=192.168.0.2
18 #     rightsubnet=10.2.0.0/16
19 #     rightcert=peerCert.der
20 #     auto=start
21
22 #conn sample-with-ca-cert
23 #     leftsubnet=10.1.0.0/16
24 #     leftcert=myCert.pem
25 #     right=192.168.0.2
26 #     rightsubnet=10.2.0.0/16
27 #     rightid="C=CH, O=Linux strongSwan CN=peer name"
```

Figure 5: ipsec.conf.

6. At the end of the file ipsec.conf,i add the following lines with the IPs i have used in my network (instead of IPX and IPY), as shown in figure 6

A screenshot of a terminal window showing the end of the ipsec.conf file being edited. The text is as follows:

```
#     rightid="C=CH, O=Linux strongSwan CN=peer name"
#     auto=start
conn host-to-host
    authby=secret
    auto=route
    keyexchange=ike
    left=192.168.10.5
    right=192.168.10.4
    type=transport
    esp=aes128gcm16!
```

The prompt at the bottom is 'mohammed@mohammed-VirtualBox:/etc\$'.

Figure 6: Adding the line .

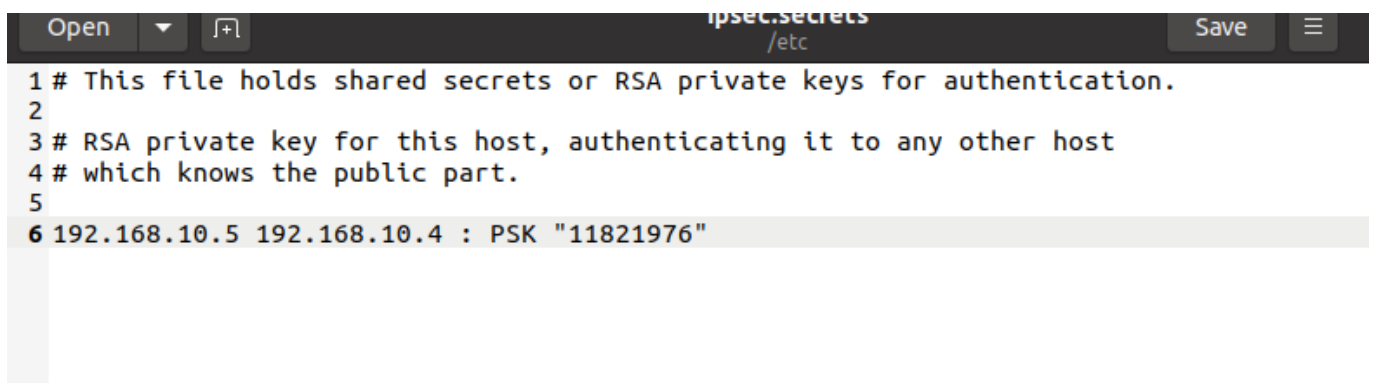
7. **Question:** Explain each item in the above lines you have added to the file **ipsec.conf**.

- (a) **conn host-to-host:** The tunnel has to be established between two hosts.
- (b) **authby=secret:** how the two security host should authenticate each other.
- (c) **auto=route:** route loads a connection and installs kernel traps. If traffic is detected between left-subnet and rightsubnet, a connection is established.
- (d) **keyexchange=ike:** which protocol should be used to initialize the connection both protocols are handled by Charon and connections marked with ike will use IKEv2 when initiating.
- (e) **left=IPX , right=IPY :** Connection descriptions are defined in terms of a left endpoint and a right endpoint. For example, the two parameters leftid and rightid specify the identity of the left and the right endpoint. For every connection description an attempt is made to figure out whether the local endpoint should act as the left or the right endpoint. This is done by matching the IP addresses defined for both endpoints with the IP addresses assigned to local network interfaces. If a match is found then the role (left or right) that matches is going to be considered "local". If no match is found during startup, "left" is considered "local"
- (f) **esp=aes128gcm16! :** comma-separated list of ESP encryption/authentication algorithms to be used for the connection.

8. Add the following line to the file IPsec.secrets as shown in figures 7 and 8, and we made sure to include the IPs I had in my network instead of IPX and IPY

```
1 # This file holds shared secrets or RSA private keys for authentication.
2
3 # RSA private key for this host, authenticating it to any other host
4 # which knows the public part.
5 192.168.10.5 192.168.10.4 :PSK "11821976".
6
```

Figure 7: Ipsec.secret file A .



```
Open  ipsec.secrets  Save  ≡
/etc
1 # This file holds shared secrets or RSA private keys for authentication.
2
3 # RSA private key for this host, authenticating it to any other host
4 # which knows the public part.
5
6 192.168.10.5 192.168.10.4 : PSK "11821976"
```

Figure 8: Ipsec.secret file B.

9. Now restart the IPSec process in each **vm** by **sudo ipsec restart** command, as shown in figures 9 and 10

```
mohammed@mohammed-VirtualBox:/etc$ sudo ipsec restart
[sudo] password for mohammed:
Stopping strongSwan IPsec failed: starter is not running
Starting strongSwan 5.8.2 IPsec [starter]...
mohammed@mohammed-VirtualBox:/etc$
```

Figure 9: Restart ipsec process A.

```
mohammed2@mohammed2-VirtualBox:/etc$ sudo ipsec restart
[sudo] password for mohammed2:
Stopping strongSwan IPsec failed: starter is not running
Starting strongSwan 5.8.2 IPsec [starter]...
mohammed2@mohammed2-VirtualBox:/etc$
```

Figure 10: Restart ipsec process B.

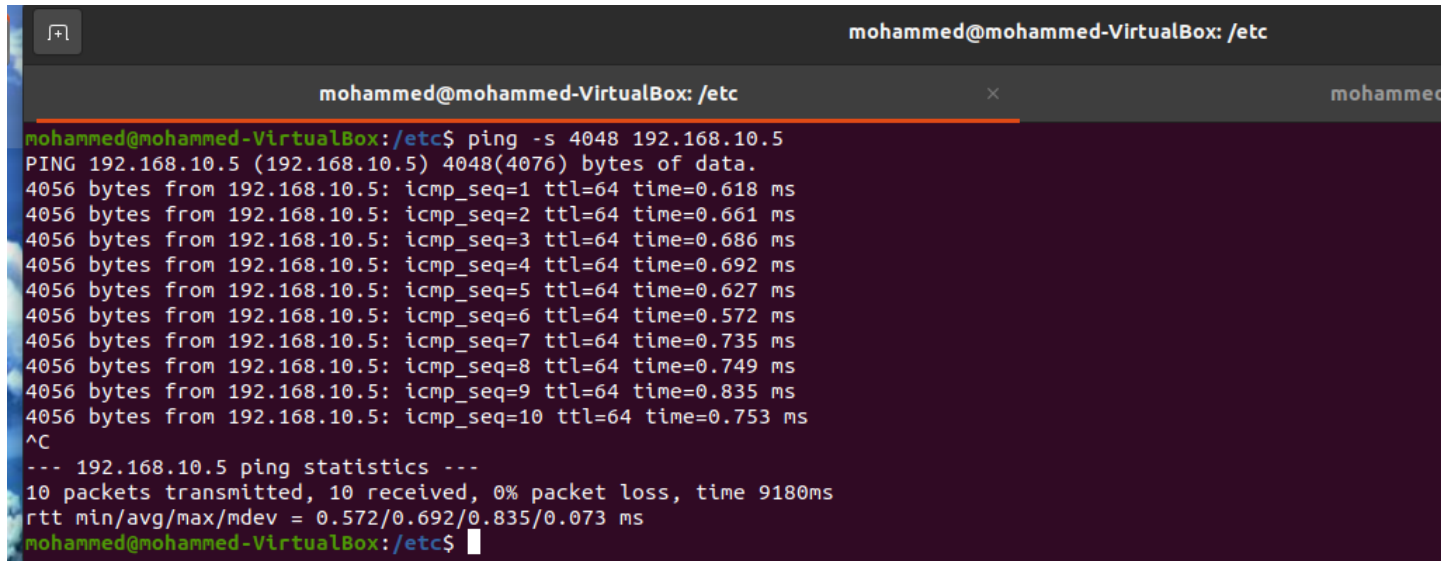
10. **Question :** Why do you need to re-start the IPSec process?  
The restart action was set to only refresh the configuration, to minimize disruption to running tunnels, and also to apply changed configuration
11. The command **IPSec statusall**, as in figure 11, shows the status of the new configuration that has been added..

```
mohammed@mohammed-VirtualBox:/etc$ sudo ipsec statusall
Status of IKE charon daemon (strongSwan 5.8.2, Linux 5.8.0-45-generic, x86_64):
  uptime: 4 minutes, since Mar 22 15:31:09 2021
  malloc: sbrk 1486848, mmap 0, used 504496, free 982352
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 0
  loaded plugins: charon aesni aes rc2 sha2 sha1 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1
  ips-prf gmp agent xcbc hmac gcm drbg attr kernel-netlink resolve socket-default connmark stroke updown eap-mscha
Listening IP addresses:
  192.168.10.4
Connections:
host-to-host: 192.168.10.4...192.168.10.5 IKEv1/2
host-to-host: local: [192.168.10.4] uses pre-shared key authentication
host-to-host: remote: [192.168.10.5] uses pre-shared key authentication
host-to-host: child: dynamic == dynamic TRANSPORT
Routed Connections:
host-to-host{1}: ROUTED, TRANSPORT, reqid 1
host-to-host{1}: 192.168.10.4/32 == 192.168.10.5/32
Security Associations (0 up, 0 connecting):
  none
mohammed@mohammed-VirtualBox:/etc$
```

Figure 11: IPSec statusall.

### 3.2 Testing the Constructed Tunnel

1. After Issue a ping command from the first host to the second host. And Setting the packet size to be 4048 bytes,as shown in figure 12 .

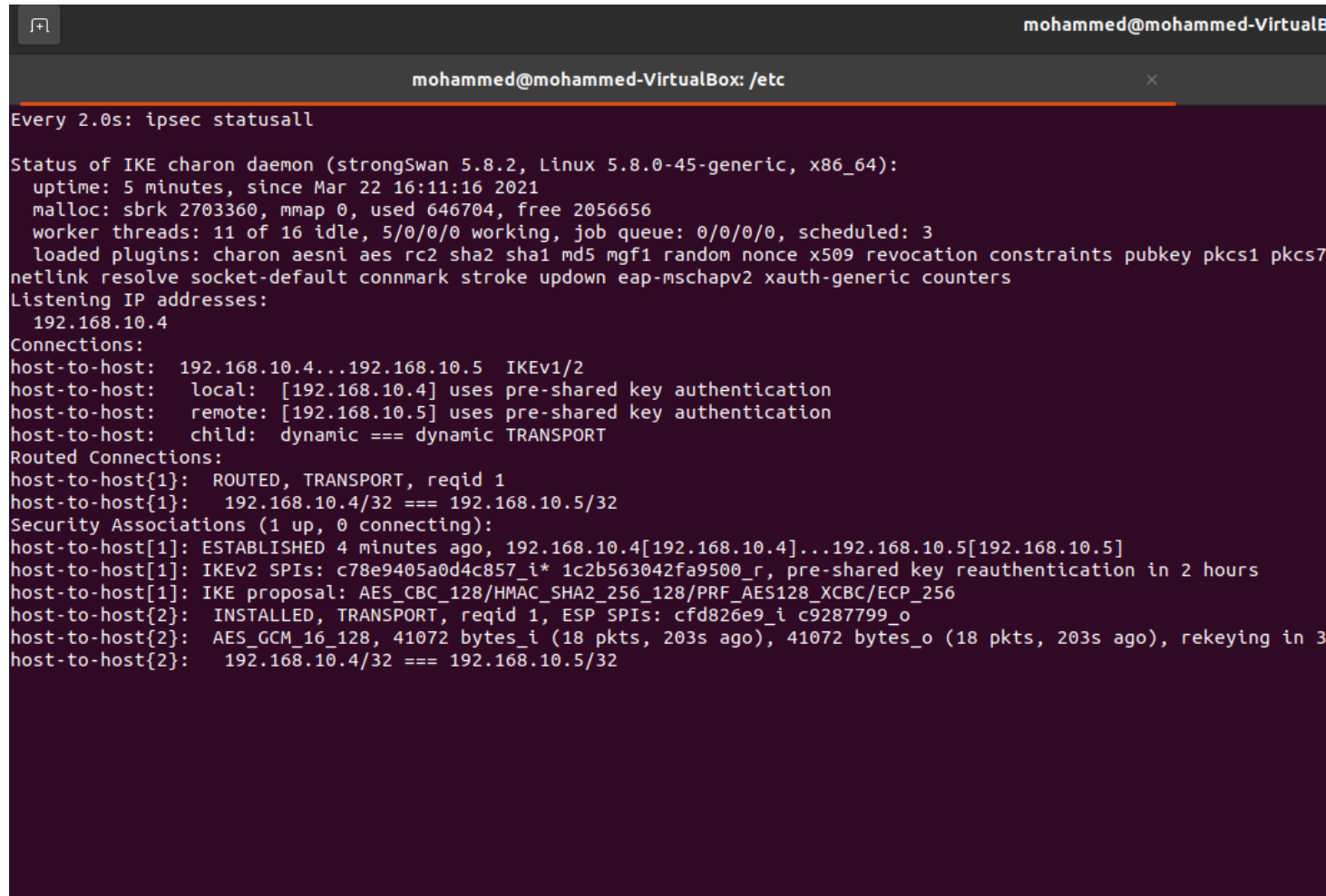
A terminal window titled 'mohammed@mohammed-VirtualBox: /etc' showing a ping command execution. The command is 'ping -s 4048 192.168.10.5'. The output shows 10 successful pings with a packet size of 4048 bytes (displayed as 4076). The ping statistics show 10 packets transmitted, 10 received, 0% packet loss, and a total time of 9180ms. The round-trip time (rtt) statistics are: min/avg/max/mdev = 0.572/0.692/0.835/0.073 ms.

```
mohammed@mohammed-VirtualBox: /etc$ ping -s 4048 192.168.10.5
PING 192.168.10.5 (192.168.10.5) 4048(4076) bytes of data.
4056 bytes from 192.168.10.5: icmp_seq=1 ttl=64 time=0.618 ms
4056 bytes from 192.168.10.5: icmp_seq=2 ttl=64 time=0.661 ms
4056 bytes from 192.168.10.5: icmp_seq=3 ttl=64 time=0.686 ms
4056 bytes from 192.168.10.5: icmp_seq=4 ttl=64 time=0.692 ms
4056 bytes from 192.168.10.5: icmp_seq=5 ttl=64 time=0.627 ms
4056 bytes from 192.168.10.5: icmp_seq=6 ttl=64 time=0.572 ms
4056 bytes from 192.168.10.5: icmp_seq=7 ttl=64 time=0.735 ms
4056 bytes from 192.168.10.5: icmp_seq=8 ttl=64 time=0.749 ms
4056 bytes from 192.168.10.5: icmp_seq=9 ttl=64 time=0.835 ms
4056 bytes from 192.168.10.5: icmp_seq=10 ttl=64 time=0.753 ms
^C
--- 192.168.10.5 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9180ms
rtt min/avg/max/mdev = 0.572/0.692/0.835/0.073 ms
mohammed@mohammed-VirtualBox: /etc$
```

Figure 12: Ping with 4048 packet size.



2. Run the command `watch ipsec statusall` as figure 13 show.



```
Every 2.0s: ipsec statusall

Status of IKE charon daemon (strongSwan 5.8.2, Linux 5.8.0-45-generic, x86_64):
  uptime: 5 minutes, since Mar 22 16:11:16 2021
  malloc: sbrk 2703360, mmap 0, used 646704, free 2056656
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 3
  loaded plugins: charon aesni aes rc2 sha2 sha1 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1 pkcs7
netlink resolve socket-default connmark stroke updown eap-mschapv2 xauth-generic counters
Listening IP addresses:
  192.168.10.4
Connections:
host-to-host: 192.168.10.4...192.168.10.5 IKEv1/2
host-to-host: local: [192.168.10.4] uses pre-shared key authentication
host-to-host: remote: [192.168.10.5] uses pre-shared key authentication
host-to-host: child: dynamic == dynamic TRANSPORT
Routed Connections:
host-to-host{1}: ROUTED, TRANSPORT, reqid 1
host-to-host{1}: 192.168.10.4/32 == 192.168.10.5/32
Security Associations (1 up, 0 connecting):
host-to-host{1}: ESTABLISHED 4 minutes ago, 192.168.10.4[192.168.10.4]...192.168.10.5[192.168.10.5]
host-to-host{1}: IKEv2 SPIs: c78e9405a0d4c857_i* 1c2b563042fa9500_r, pre-shared key reauthentication in 2 hours
host-to-host{1}: IKE proposal: AES_CBC_128/HMAC_SHA2_256_128/PRF_AES128_XCBC/ECP_256
host-to-host{2}: INSTALLED, TRANSPORT, reqid 1, ESP SPIs: cfd826e9_i c9287799_o
host-to-host{2}: AES_GCM_16_128, 41072 bytes_i (18 pkts, 203s ago), 41072 bytes_o (18 pkts, 203s ago), rekeying in 3
host-to-host{2}: 192.168.10.4/32 == 192.168.10.5/32
```

Figure 13: Ipsec statusall command .

3. We conclude that the IPSec connection is still open, also the time still counts, and the security association waits for one of the parties left or right to connect.

4. After used tcpdump to capture ESP packets. We write those packets to a file named **ESP.cap**, as shown in figure 14 .

```
mohammed@mohammed-VirtualBox: ~/Desktop$ cd /home/mohammed/Desktop/
mohammed@mohammed-VirtualBox:~/Desktop$ sudo tcpdump -w ESP.cap
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 4096 bytes
^C16 packets captured
16 packets received by filter
0 packets dropped by kernel
mohammed@mohammed-VirtualBox:~/Desktop$ sudo wireshark &
```

Figure 14: ESP.cap file .

- After Opening the file ESP.cap using Wireshark and selecting a captured ESP, Figure 15 shows the ESP packet and the SPI for these packets was **3263704050**.

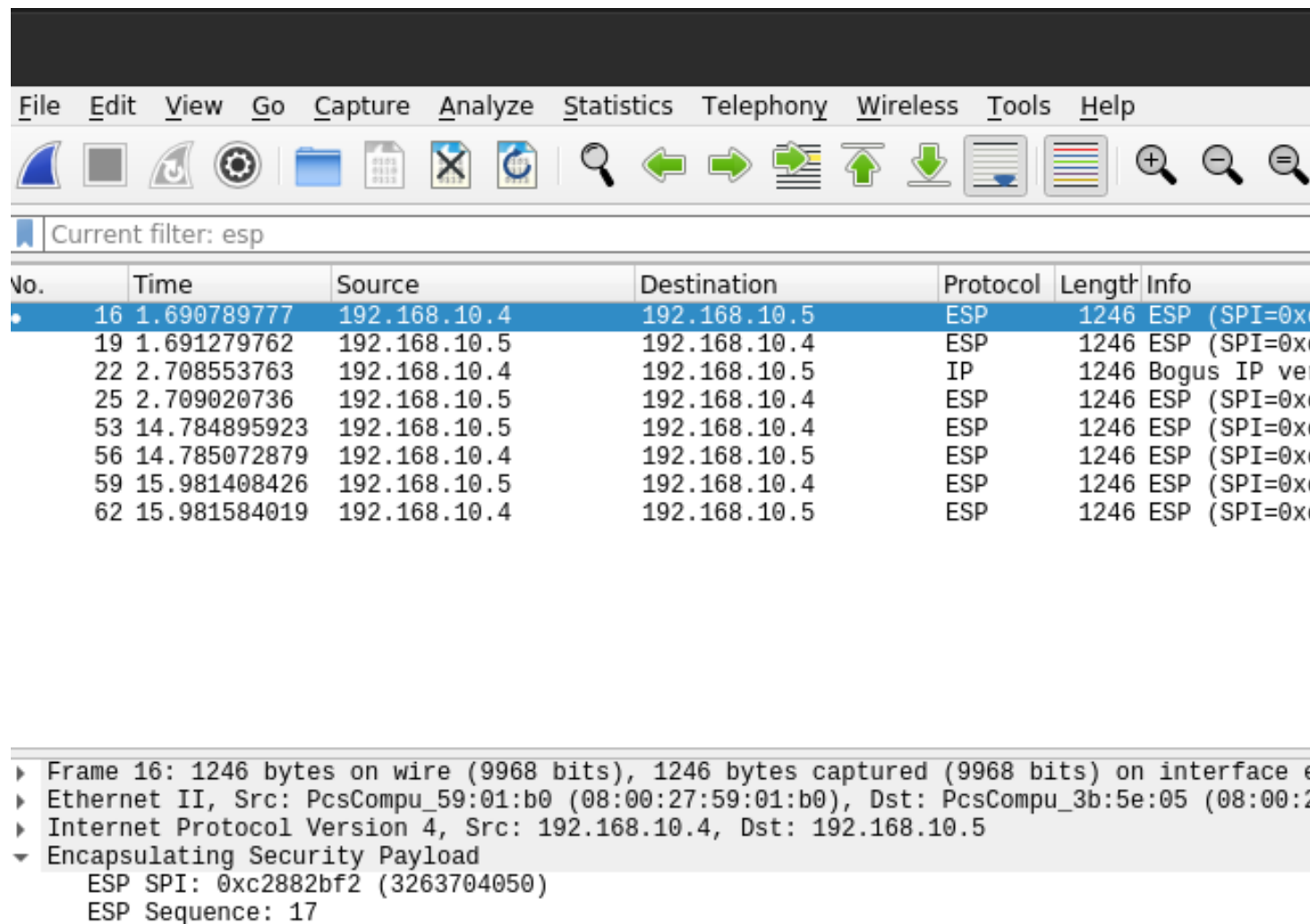
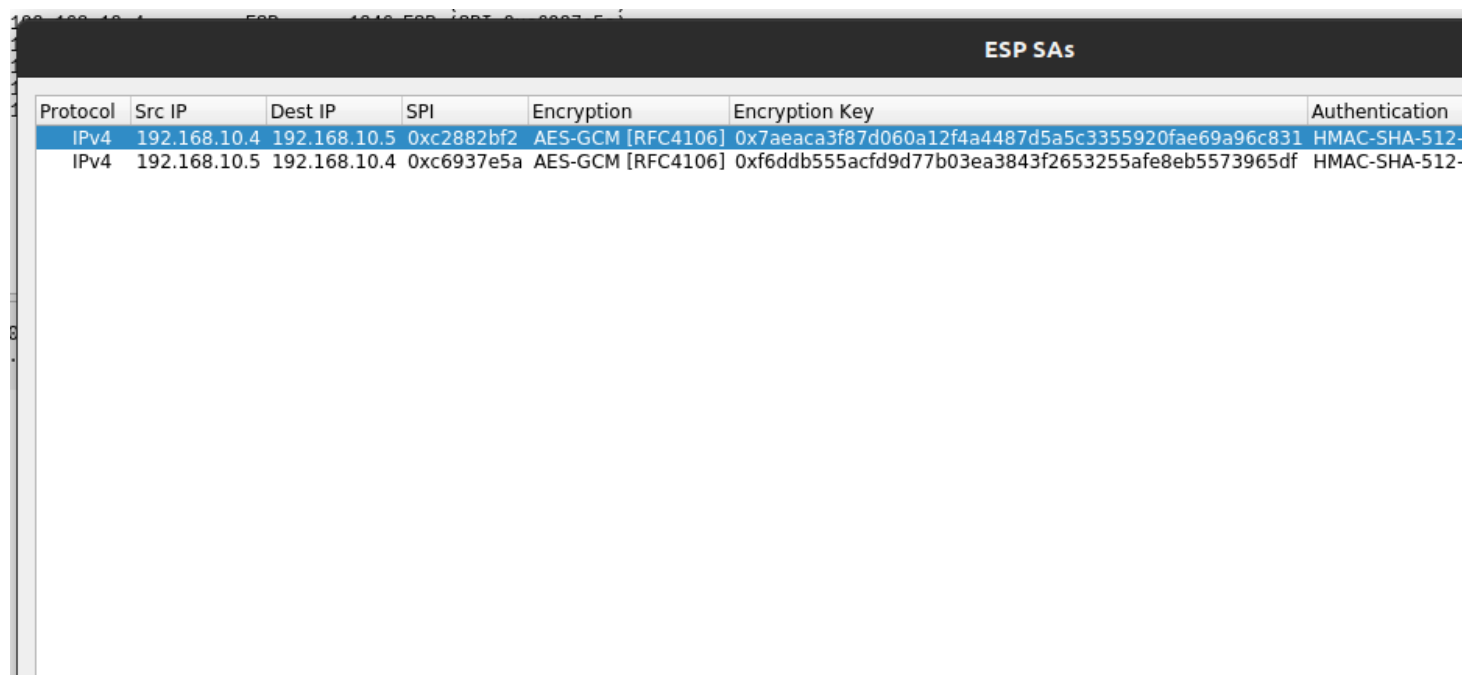


Figure 15: ESP packet.

6. **Question :** Show the Security Associations (SAs) that have been implemented on the first host. How can you do that?

In first we go in preference and we search for **ESP** protocol and we add two entry in ESP SA which is : inbound and outbound as figures 16 and 17 show.



Protocol	Src IP	Dest IP	SPI	Encryption	Encryption Key	Authentication
IPv4	192.168.10.4	192.168.10.5	0xc2882bf2	AES-GCM [RFC4106]	0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831	HMAC-SHA-512
IPv4	192.168.10.5	192.168.10.4	0xc6937e5a	AES-GCM [RFC4106]	0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df	HMAC-SHA-512

Figure 16: Add inbound outbound.

▼ Payload: Security Association (33)

Next payload: Key Exchange (34)

0... .... = Critical Bit: Not Critical

.0000 0000 = Reserved: 0x00

Payload length: 48

▼ Payload: Proposal (2) # 1

Next payload: NONE / No Next Payload (0)

0... .... = Critical Bit: Not Critical

.0000 0000 = Reserved: 0x00

Payload length: 44

Proposal number: 1

Protocol ID: IKE (1)

SPI Size: 0

Proposal transforms: 4

▼ Payload: Transform (3)

Next payload: Transform (3)

0... .... = Critical Bit: Not Critical

.0000 0000 = Reserved: 0x00

Payload length: 12

Transform Type: Encryption Algorithm (ENCR) (1)

Reserved: 00

Transform ID (ENCR): ENCR\_AES\_CBC (12)

Figure 17: SA entry.

7. By doing ping from the second host to the first host, and capture an ESP packet on the first host, the SPI was **3263704050**, as figure 18 shows.

53	14.784895923	192.168.10.5	192.168.10.4	ESP	1246	ESP (SPI=0xc6937e5a)
56	14.785072879	192.168.10.4	192.168.10.5	ESP	1246	ESP (SPI=0xc2882bf2)
59	15.981408426	192.168.10.5	192.168.10.4	ESP	1246	ESP (SPI=0xc6937e5a)
62	15.981584019	192.168.10.4	192.168.10.5	ESP	1246	ESP (SPI=0xc2882bf2)

▶ Frame 56: 1246 bytes on wire (9968 bits), 1246 bytes captured (9968 bits) on interface enp0s3, id 0  
 ▶ Ethernet II, Src: PcsCompu\_59:01:b0 (08:00:27:59:01:b0), Dst: PcsCompu\_3b:5e:05 (08:00:27:3b:5e:05)  
 ▶ Internet Protocol Version 4, Src: 192.168.10.4, Dst: 192.168.10.5  
 ▼ Encapsulating Security Payload  
   ESP SPI: 0xc2882bf2 (3263704050)  
   ESP Sequence: 19

Figure 18: ESP packet.

8. We Change the pre-shared key value in one of the hosts, and restart the IPsec process, and do ping again as figure 19 show .

```

starting strongswan 5.8.2 IPsec [starter]...
mohammed@mohammed-VirtualBox:/etc$ ping -s 4048 192.168.10.5
PING 192.168.10.5 (192.168.10.5) 4048(4076) bytes of data.

```

Figure 19: Ping.

**Question:**Is there any captured packet?  
 No,as shown in figure 20.

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
Current filter: esp						
No.	Time	Source	Destination	Protocol	Length	Info

Figure 20: Captured packet.

### 3.3 SSH With the Constructed Tunnel

1. Figure 21 show how to install **SSH** client in both hosts.

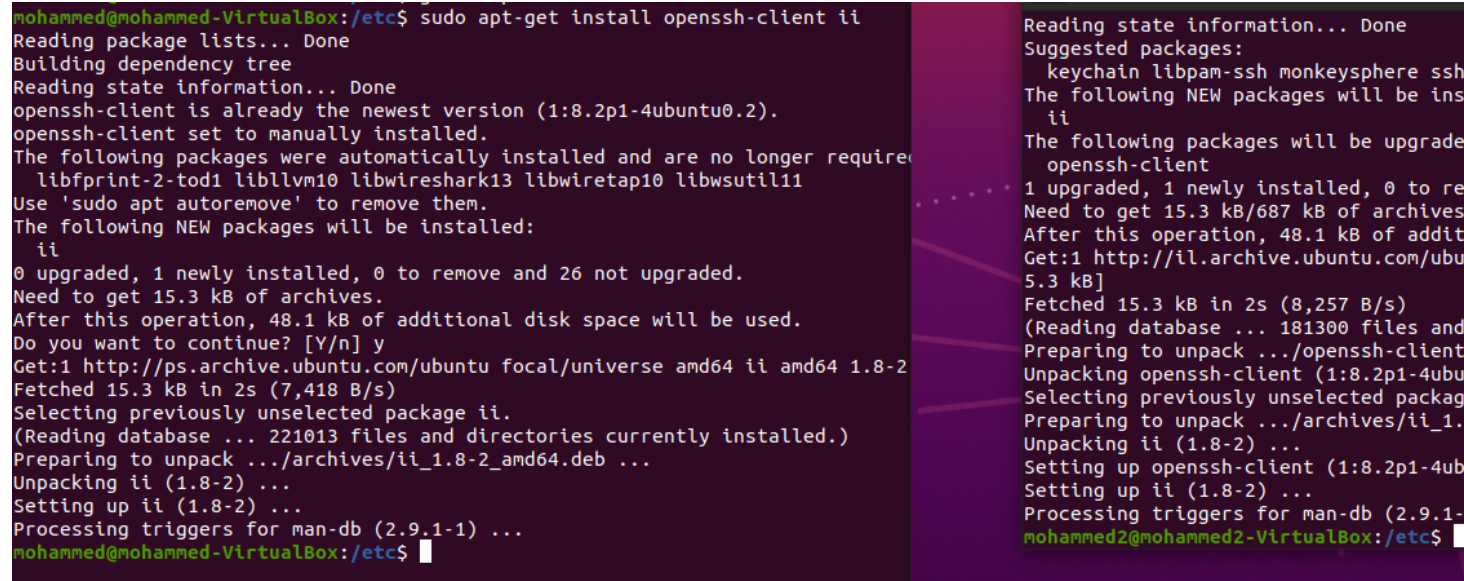
The image shows two terminal windows side-by-side. The left window is titled 'mohammed@mohammed-VirtualBox:/etc\$' and shows the command 'sudo apt-get install openssh-client ii'. The output indicates that 'openssh-client' is already the newest version (1:8.2p1-4ubuntu0.2) and is set to manually installed. It lists several packages that were automatically installed and are no longer required, including 'libfprint-2-tod1', 'liblvm10', 'libwireshark13', 'libwiretap10', and 'libwsutil11'. It then lists the following NEW packages to be installed: 'ii'. The output shows that 0 packages are upgraded, 1 is newly installed, and 0 are to be removed. It states that 15.3 kB of archives are needed, and after this operation, 48.1 kB of additional disk space will be used. It asks if the user wants to continue, and the user responds 'y'. It then shows the download of the package from the Ubuntu archive, followed by unpacking and setting up the 'ii' package. The right window is titled 'mohammed2@mohammed2-VirtualBox:/etc\$' and shows the same command. The output is identical to the left window, showing that the package is already installed and setting it up.

Figure 21: SSH client.

2. Also, figure 22 shows how to install **SSH** server.

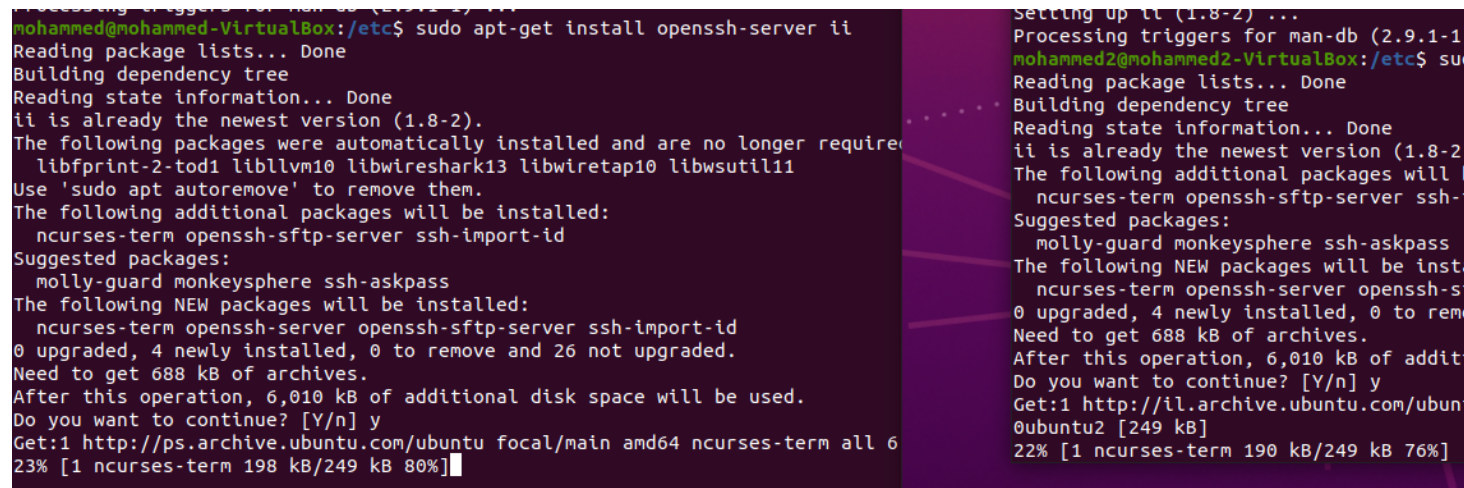
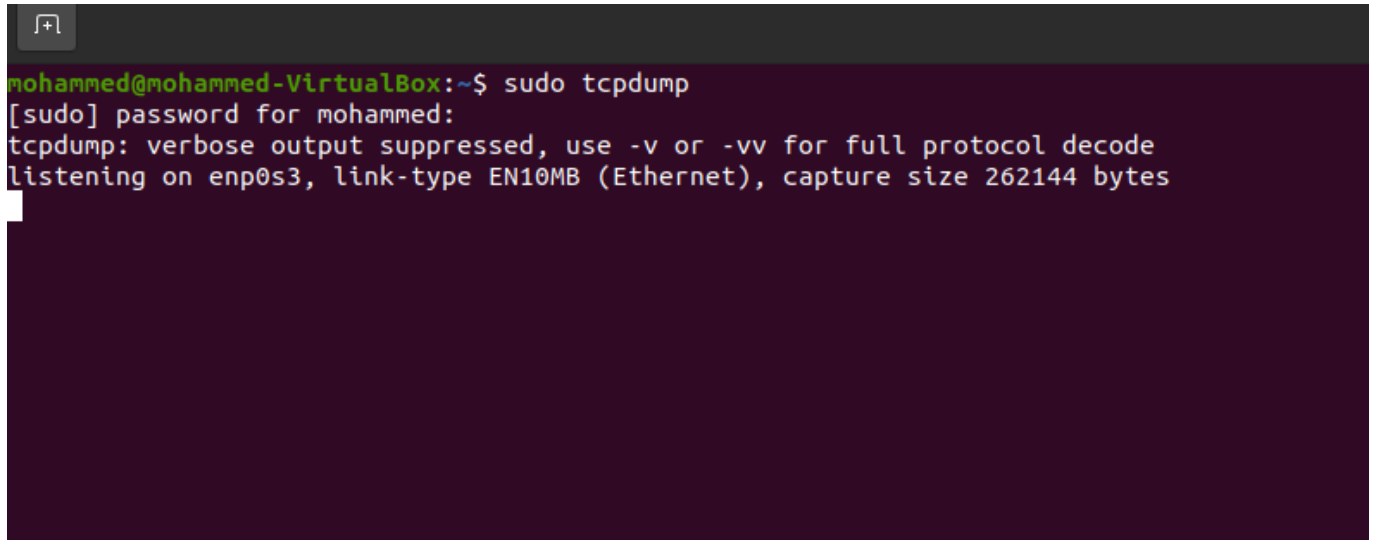
The image shows two terminal windows side-by-side. The left window is titled 'mohammed@mohammed-VirtualBox:/etc\$' and shows the command 'sudo apt-get install openssh-server ii'. The output indicates that 'ii' is already the newest version (1.8-2). It lists several packages that were automatically installed and are no longer required, including 'libfprint-2-tod1', 'liblvm10', 'libwireshark13', 'libwiretap10', and 'libwsutil11'. It then lists the following additional packages to be installed: 'ncurses-term', 'openssh-sftp-server', and 'ssh-import-id'. It suggests packages 'molly-guard', 'monkeysphere', and 'ssh-askpass'. The output shows that 0 packages are upgraded, 4 are newly installed, and 0 are to be removed. It states that 688 kB of archives are needed, and after this operation, 6,010 kB of additional disk space will be used. It asks if the user wants to continue, and the user responds 'y'. It then shows the download of the packages from the Ubuntu archive, followed by unpacking and setting up the 'ii' package. The right window is titled 'mohammed2@mohammed2-VirtualBox:/etc\$' and shows the same command. The output is identical to the left window, showing that the package is already installed and setting it up.

Figure 22: SSH server.

3. By run the TCP dump on one machine and tried to capture ESP packets , the rustle was as figure [23](#) show.



```
mohammed@mohammed-VirtualBox:~$ sudo tcpdump
[sudo] password for mohammed:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Figure 23: Run Tcpdumb.



4. To connect SSH from one machine to another. I used the command `ssh username@host-ip-address` with the password of the host i connecting to. Figures 24 and 25 show how it done.

```
mohammed@mohammed-VirtualBox:/etc$ ssh localhost
The authenticity of host 'localhost (:::1)' can't be established.
ECDSA key fingerprint is SHA256:VjPQrZ9W0CfNQh8eGxa0RCLKVYqFw2Xkrgu0zZvTjQ0.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
mohammed@localhost's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.8.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

19 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

Figure 24: SSH connection.

```
mohammed@mohammed-VirtualBox:~$ ssh mohammed2@192.168.10.5
mohammed2@192.168.10.5's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.8.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

341 updates can be installed immediately.
149 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
mohammed2@mohammed2-VirtualBox:~$
```

Figure 25: SSH connection.

5. When the SSH connection is open in IPSec Tunnel, every move inside the connection made an ESP packet, as I conclude when I tried to analyze one of those packets.

### 3.4 conclusion

IPsec is a group of protocols that are used together to set up encrypted connections between devices. It helps keep data sent over public networks secure as we see in this experiment. IPsec is often used to set up VPNs, and it works by encrypting IP packets, along with authenticating the source where the packets come from as this experiment show.

### 3.5 references

<https://www.secfu.net/2017/12/23/the-ikev2-header-and-the-security-association-payload/>

[https://wiki.wireshark.org/ESP\\_preferences](https://wiki.wireshark.org/ESP_preferences)

<https://phoenixnap.com/kb/ssh-to-connect-to-remote-server-linux-or-windows>

[https://en.wikipedia.org/wiki/IPsecAuthentication\\_header](https://en.wikipedia.org/wiki/IPsecAuthentication_header)