2.1. Generacion de certificados en pc3, r1 y r4

NOTA: A LO LARGO DE LA PRACTICA SE PODRA APRECIAR QUE LOS SPI VAN CAMBIANDO. ESO SE DEBE A QUE NO HICE LA PRACTICA SEGUDIA, SINO EN DIFERENTES DIAS.

1. Indica los nombres de los ficheros de certificados y claves privadas que has generado y en que carpetas los has almacenado en cada una de las maquinas.

Creamos lab-ipsec la carpeta my-rsa-certs: mkdir ~/lab-ipsec/my-rsa-certs
dentro cremos tres carpetas: cacerts, certs, private
+ dentro de cacerts:
$ ipsec pki --gen --type rsa --size 4096 --outform pem > myCAkey.pem
$ sudo chmod 600 myCAkey.pem
$ ipsec pki --self --ca --lifetime 3650 --in myCAKey.pem --type rsa --dn "C=ES, O=myCA, CN=My Root CA" --outform pem > myCACert.pem
$ ipsec pki --print --in myCACert.pem

+ dentro de la carpeta creada private (mkdir private):
$ ipsec pki --gen --type rsa --size 2048 --outform pem > r1Key.pem
$ ipsec pki --gen --type rsa --size 2048 --outform pem > r4Key.pem
$ ipsec pki --gen --type rsa --size 2048 --outform pem > pc3Key.pem
$ sudo chmod 600 * (cambiamos los permisos a todos los ficheros que hay en el directorio, que es lo que queremos)
$ ipsec pki --pub --in r1Key.pem --type rsa | ipsec pki --issue --lifetime 730 --cacert /home/david/lab-ipsec/my-rsa-certs/cacerts/myCACert.pem --cakey /home/david/lab-ipsec/my-rsa-certs/cacerts/myCAKey.pem --dn "C=ES, O=myCA, CN=r1" --san r1 --flag serverAuth --flag ikeIntermediate --outform pem > r1Cert.pem
$ ipsec pki --pub --in r4Key.pem --type rsa | ipsec pki --issue --lifetime 730 --cacert /home/david/lab-ipsec/my-rsa-certs/cacerts/myCACert.pem --cakey /home/david/lab-ipsec/my-rsa-certs/cacerts/myCAKey.pem --dn "C=ES, O=myCA, CN=r4" --san r4 --flag serverAuth --flag ikeIntermediate --outform pem > r4Cert.pem
$ ipsec pki --pub --in pc3Key.pem --type rsa | ipsec pki --issue --lifetime 730 --cacert /home/david/lab-ipsec/my-rsa-certs/cacerts/myCACert.pem --cakey /home/david/lab-ipsec/my-rsa-certs/cacerts/myCAKey.pem --dn "C=ES, O=myCA, CN=pc3" --san pc3 --flag serverAuth --flag ikeIntermediate --outform pem > pc3Cert.pem

Los certificados aunque los habia generado en la carpeta private, los he movido a otra llamada certs.

Una vez generado todo lo necesario, he almacenado lo siguiente:
--> r1: certs --> r1Cert.pem
        private--> r1Key.pem
        cacerts--> myCACert.pem

--> r4: certs --> r4Cert.pem
        private--> r4Key.pem
        cacerts--> myCACert.pem

--> pc3: certs --> pc3Cert.pem

```
            private--> pc3Key.pem
            cacerts--> myCACert.pem

================================================================================
================================================================================
==========
```

2. Incluye el resultado de imprimir de forma legible cada uno de los certificados que has creado.

```
david@alonsod:~/lab-ipsec/my-rsa-certs/certs$ ipsec pki --print --in
r1Cert.pem
cert:      X509
subject:  "C=ES, O=myCA, CN=r1"
issuer:   "C=ES, O=myCA, CN=My Root CA"
validity:  not before Apr 06 13:04:02 2018, ok
           not after  Apr 05 13:04:02 2020, ok (expires in 729 days)
serial:    01:0e:c3:1c:98:d6:ad:69
altNames:  r1
flags:     serverAuth iKEIntermediate
authkeyId: 91:f5:e2:36:bc:b7:a7:d0:d4:50:dc:5f:74:41:a4:ac:5a:aa:25:e8
subjkeyId: 2b:df:68:61:64:39:51:21:ee:81:37:7c:c1:0a:c8:5d:ac:be:7e:69
pubkey:    RSA 2048 bits
keyid:     05:0f:39:36:ae:52:4b:b9:af:28:19:bb:11:c1:54:35:99:60:35:9d
subjkey:   2b:df:68:61:64:39:51:21:ee:81:37:7c:c1:0a:c8:5d:ac:be:7e:69

david@alonsod:~/lab-ipsec/my-rsa-certs/certs$ ipsec pki --print --in
r4Cert.pem
cert:      X509
subject:  "C=ES, O=myCA, CN=r4"
issuer:   "C=ES, O=myCA, CN=My Root CA"
validity:  not before Apr 07 11:38:47 2018, ok
           not after  Apr 06 11:38:47 2020, ok (expires in 729 days)
serial:    08:cd:d4:74:f0:da:7f:52
altNames:  r4
flags:     serverAuth iKEIntermediate
authkeyId: 91:f5:e2:36:bc:b7:a7:d0:d4:50:dc:5f:74:41:a4:ac:5a:aa:25:e8
subjkeyId: ef:5d:6b:ed:d3:61:72:97:bd:17:5d:49:60:f6:82:2f:11:c8:90:34
pubkey:    RSA 2048 bits
keyid:     c2:45:3e:b1:7e:0a:b1:4f:32:e4:42:b9:ee:eb:10:4e:97:28:a3:18
subjkey:   ef:5d:6b:ed:d3:61:72:97:bd:17:5d:49:60:f6:82:2f:11:c8:90:34

david@alonsod:~/lab-ipsec/my-rsa-certs/certs$ ipsec pki --print --in
pc3Cert.pem
cert:      X509
subject:  "C=ES, O=myCA, CN=pc3"
issuer:   "C=ES, O=myCA, CN=My Root CA"
validity:  not before Apr 07 11:39:00 2018, ok
           not after  Apr 06 11:39:00 2020, ok (expires in 729 days)
serial:    3b:52:98:80:a6:1a:e8:7e
altNames:  pc3
flags:     serverAuth iKEIntermediate
authkeyId: 91:f5:e2:36:bc:b7:a7:d0:d4:50:dc:5f:74:41:a4:ac:5a:aa:25:e8
subjkeyId: 85:07:c0:58:58:9b:dd:1a:2c:8c:a3:0f:fa:8c:a2:34:34:77:b2:38
```

```
pubkey:    RSA 2048 bits
keyid:     95:cc:a8:44:bf:68:61:a4:12:4f:20:14:3c:cc:2c:2e:c3:9d:0b:3a
subjkey:   85:07:c0:58:58:9b:dd:1a:2c:8c:a3:0f:fa:8c:a2:34:34:77:b2:38

david@alonsod:~/lab-ipsec/my-rsa-certs/cacerts$ ipsec pki --print --in
myCACert.pem
cert:      X509
subject:  "C=ES, O=myCA, CN=My Root CA"
issuer:   "C=ES, O=myCA, CN=My Root CA"
validity:  not before Apr 06 12:40:36 2018, ok
           not after  Apr 03 12:40:36 2028, ok (expires in 3649 days)
serial:    5f:8e:c7:7d:e9:88:1e:d3
flags:     CA CRLSign self-signed
subjkeyId: 91:f5:e2:36:bc:b7:a7:d0:d4:50:dc:5f:74:41:a4:ac:5a:aa:25:e8
pubkey:    RSA 4096 bits
keyid:     a3:34:da:36:d5:0e:ed:ab:db:6a:1c:0e:6a:78:3c:46:5f:e8:a2:b7
subjkey:   91:f5:e2:36:bc:b7:a7:d0:d4:50:dc:5f:74:41:a4:ac:5a:aa:25:e8

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++
```

2.2. ESP en modo tunel entre r1 y r4

1. Incluye en la memoria los ficheros que has configurado en cada uno de los extremos del tunel.

dentro del directorio ~/lab-ipsec/r1/etc se encuentra el primer fichero que tenmos que modificar: ipsec.conf
Esta es la configuracion para la mquina r1:

```
config setup
      # strictcrlpolicy=yes
      # uniqueids = no
      charondebug="all"

# Add connections here.

conn %default
      ikelifetime=24h
      rekeymargin=3m
      keyingtries=1
      keyexchange=ikev2
      ike=aes128-sha256-modp3072!
      esp=aes128-sha256-modp3072!

# Sample VPN connections

conn net-net
      left=100.18.2.1
      leftcert=r1Cert.pem
      leftid=@r1
      leftsubnet=10.18.1.0/24
      right=100.18.4.4
```

```
      rightid=@r4
      rightsubnet=10.18.2.0/24
      type=tunnel
      auto=add
```

modificamos tambien el mismo fichero pero en la maquina r4:

```
config setup
      # strictcrlpolicy=yes
      # uniqueids = no
      charondebug="all"

# Add connections here.
conn %default
      ikelifetime=24h
      rekeymargin=3m
      keyingtries=1
      keyexchange=ikev2
      ike=aes128-sha256-modp3072!
      esp=aes128-sha256-modp3072!

# Sample VPN connections
conn net-net
      left=100.18.4.4
      leftcert=r4Cert.pem
      leftid=@r4
      leftsubnet=10.18.2.0/24
      right=100.18.2.1
      rightid=@r1
      rightsubnet=10.18.1.0/24
      type=tunnel
      auto=add
```

++
++
++++++++++

2.2.1. Gestion de SAs usando IKEv2

1. Inicia un captura de trafico en r1(eth1) para capturar el intercambio de mensajes IKE (recuerda usar la opcion -s 0 para que se capturen los paquetes completos) y guarda los paquetes capturados en el fichero ipsec-00.cap. Arranca IPsec en los extremos y activa la configuraci´on del tunel desde r1 para que comience el intercambio de mensajes IKE. Comprobaras que los extremos han acordado la SA que van a usar. Interrumpe la captura y cargala en Wireshark para analizarla.

$ arrancamos en r1 la captura en r1-eth1: tcpdump -i r1-eth1 -s 0 -w ipsec-00.cap
$ un vez lanzada la captura, lanzamos ipsec(en r1 y r4): ipsec start
$ ahora en r1 lanzamos la conexion net-net que habiamos configurado previamente en ipsec.conf: ipsec up net-net
$ vemos el estado de cada extremo (en r1 y r4):

```
*** r1 ***
root@alonsod:~/lab-ipsec# ipsec status
Security Associations (1 up, 0 connecting):
     net-net[1]: ESTABLISHED 71 seconds ago,
100.18.2.1[r1]...100.18.4.4[r4]
     net-net{1}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: c0bbf522_i
cecca69d_o
     net-net{1}:   10.18.1.0/24 === 10.18.2.0/24

*** r4 ***
root@alonsod:~/lab-ipsec# ipsec status
Security Associations (1 up, 0 connecting):
     net-net[1]: ESTABLISHED 57 seconds ago,
100.18.4.4[r4]...100.18.2.1[r1]
     net-net{1}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: cecca69d_i
c0bbf522_o
     net-net{1}:   10.18.2.0/24 === 10.18.1.0/24
```

==========

2. Indica que campos puedes ver en cada uno de los mensajes IKE capturados. Explıcalos.

El tercer paquete que aparece en la captura se corresponde con el primer mensaje ike. Podemos encontrar los siguientes campos:

```
Initiator SPI: 3f628fb6aade7fc9
Responder SPI: 0000000000000000
Next payload: Security Association (33)
Version: 2.0
Exchange type: IKE_SA_INIT (34)
Flags: 0x08 (Initiator, No higher version, Request)
Message ID: 0x00000000
Length: 576
.
.
.
```

Podemos observar que el campo Responder aun esta a cero ya que se trata del primer mensaje de IKE. Tambien podemos destacar la version utilizada, o el tipo de mensaje: IKE_SA_INIT (34), que nos indica que se trata de un tipo de mensaje de establecimiento de un tunel con IKE.
En la captura podremos ver que el mensaje 4 es la respuesta de r4, y podemos observar cambios como que Responder SPI ahora ya no es todo ceros: Responder SPI: 5b49b0b65141a6c7
El quinto mensaje se trata de un mensaje de solicitud IKE AUTH: Exchange type: IKE_AUTH (35). Este mensaje va cifrado por la claves que se han derivado de la negociación IKE SA INIT.
El septimo paquete se trata de la respuesta al tipo de mensaje IKE_AUTH. En los flags se puede observar tambien que se trata de una respuesta.

==========
==========

3. Utilizando Wireshark podras descifrar los paquetes IKEv2 si configuras en Wireshark las claves que se estan utilizando para el intercambio de mensajes IKEv2. Para ello deberas tomar las claves que se encuentran en el fichero de logs de r1: /var/logs/charon.log. Localiza en ese fichero las claves: Sk ei
(16 bytes), Sk er (16 bytes), Sk ai (32 bytes) y Sk ar (32 butes).
Con la captura que has realizado cargada en Wireshark selecciona en el men´u
Edit -> Preferences -> Protocols -> ISAKMP -> IKEv2 Decryption Table Edit y copia allı los valores sin dejar espacios en blanco.

$ Initiator SPI: 3f628fb6aade7fc9
$ Responder SPI: 5b49b0b65141a6c7
$ Sk_ei:2C15B8905B1BF1B6678101EB5F954D7E
$ Sk_er:D3DB9D9DF475897C87D54828A7F31CF1
$ Encryption algorithm: AES-CBC-128 [RFC3602]
$ Sk_ai:DA3BB7D1197BF698730952089EAB542222D6BE586005F818F6269BFC3B5D873B
$ Sk_ar:7AB79CC8E28F87CF7DCD9BD02183305FD6F4A2B2AEE5466B6CCC870121D4092C
$ Integrity algorithm: HMAC_SHA_256_128 [RFC4868]

Comprobamos que lo tenemos bien:
david@alonsod:~/.config/wireshark$ cat ikev2_decryption_table
This file is automatically generated, DO NOT MODIFY.
3f628fb6aade7fc9,5b49b0b65141a6c7,2c15b8905b1bf1b6678101eb5f954d7e,d3db9d9df475897c87d54828a7f31cf1,"AES-CBC-128 [RFC3602]",da3bb7d1197bf698730952089eab542222d6be586005f818f6269bfc3b5d873b,7ab79cc8e28f87cf7dcd9bd02183305fd6f4a2b2aee5466b6ccc870121d4092c,"HMAC_SHA2_256_128 [RFC4868]"

a) En el mensaje IKE_AUTH que envıa r1 a r4:

1) Fıjate como cada payload contiene un campo Next Payload que indica lo que viene a continuacion. Indica todos los payloads que aparecen y cual es el campo Next Payload del ultimo.

Next payload: Encrypted and Authenticated (46)
Next payload: Identification - Initiator (35)
Type Payload: Identification - Initiator (35)
Type Payload: Certificate (37)
Type Payload: Notify (41) - INITIAL_CONTACT
Type Payload: Certificate Request (38)
Type Payload: Identification - Responder (36)
Type Payload: Authentication (39)
Type Payload: Security Association (33)
Type Payload: Traffic Selector - Initiator (44) # 1
Type Payload: Traffic Selector - Responder (45) # 1
Type Payload: Notify (41) - MOBIKE_SUPPORTED
Type Payload: Notify (41) - ADDITIONAL_IP4_ADDRESS
Type Payload: Notify (41) - MULTIPLE_AUTH_SUPPORTED
Type Payload: Notify (41) - EAP_ONLY_AUTHENTICATION

El ultimo Next Payload: Next payload: NONE / No Next Payload (0)

2) Como se identifica al extremo initiator

Aparece tal cual el nombre de r1:
Identification Data:r1
      ID_FQDN: r1

3) Que certificado se incluye

vemos que se trata del certificado de la CA:
Certificate Data (id-at-commonName=r1,id-at-organizationName=myCA,id-at-countryName=ES)
Viene informacion del numero de serie, fecha de validez, el firmante, etc

4) Como debe identificarse el otro extremo (responder).

El responder es r4, segun vemos en la captura:
Identification Data:r4
      ID_FQDN: r4

5) Como el otro extremo (responder) va a autenticar a initiator.

Se consulta SAD usando la información del paquete <SPI,dirIPDestino,Protocolo>, se descifra/autentica el paquete y se consulta SPD para saber si existıá una polıítica asociada a dicho paquete.

6) En la SA que se propone indica el protocolo IPsec que se va a utilizar, el SPI y los algoritmos que envıa r1 a r4 para confidencialidad y autenticacion.

Protocolo IPSec que se va a utilizar: Protocol ID: ESP (3)
SPI Size: 4
Proposal transforms: 3
SPI: c0bbf522
Type Payload: Transform (3)
      Transform Type: Encryption Algorithm (ENCR) (1)
      Transform ID (ENCR): ENCR_AES_CBC (12)
      Transform IKE2 Attribute Type (t=14,l=2) Key-Length : 128
Type Payload: Transform (3)
      Transform Type: Integrity Algorithm (INTEG) (3)
      Transform ID (INTEG): AUTH_HMAC_SHA2_256_128 (12)
Type Payload: Transform (3)
      Transform Type: Extended Sequence Numbers (ESN) (5)
      Transform ID (ESN): No Extended Sequence Numbers (0)

7) Indica que selectores de trafico se envıan, tanto para initiator como responder.

En initiator:
Type Payload: Traffic Selector - Initiator (44) # 1
      Number of Traffic Selector: 1
      Traffic Selector Type: TS_IPV4_ADDR_RANGE (7)

Protocol ID: Unused
Selector Length: 16
Start Port: 0
End Port: 65535
Starting Addr: 10.18.1.0
Ending Addr: 10.18.1.255

En responder:
Type Payload: Traffic Selector - Responder (45) # 1
Number of Traffic Selector: 1
Traffic Selector Type: TS_IPV4_ADDR_RANGE (7)
Protocol ID: Unused
Selector Length: 16
Start Port: 0
End Port: 65535
Starting Addr: 10.18.2.0
Ending Addr: 10.18.2.255

b) En el mensaje IKE_AUTH que envıa r4 a r1:

1) Como se identifica al extremo responder

Vamos a la cabecera Type Payload: Identification - Responder (36), la desplegamos y vemos lo siguiente:
Identification Data:r4
ID_FQDN: r4

Aparece como r4 de manera directa.

2) Que certificado se incluye

El de la CA es el que se incluye:
Certificate Data (id-at-commonName=r4,id-at-organizationName=myCA,id-at-countryName=ES)

3) Como el otro extremo (initiator) va a autenticar a responder

Igual que en punto 5 del apartado anterior:
Se consulta SAD usando la información del paquete <SPI,dirIPDestino,Protocolo>, se descifra/autentica el paquete y se consulta SPD para saber si existıá una polıt́ica asociada a dicho paquete.

4) En la SA que se propone indica el protocolo IPsec que se va a utilizar, el SPI y los algoritmos que envıa r1 a r4 para confidencialidad y autenticacion. Indica que diferencias ves con respecto a lo que initiator envio en su propuesta de SA.

Type Payload: Security Association (33)
Type Payload: Proposal (2) # 1
Protocol ID: ESP (3)
SPI Size: 4
Proposal transforms: 3
SPI: cecca69d

```
            Type Payload: Transform (3)
                  Transform Type: Encryption Algorithm (ENCR) (1)
                  Transform ID (ENCR): ENCR_AES_CBC (12)
                  Transform IKE2 Attribute Type (t=14,l=2) Key-Length :
128
            Type Payload: Transform (3)
                  Transform Type: Integrity Algorithm (INTEG) (3)
                  Transform ID (INTEG): AUTH_HMAC_SHA2_256_128 (12)
            Type Payload: Transform (3)
                  Transform Type: Extended Sequence Numbers (ESN) (5)
                  Transform ID (ESN): No Extended Sequence Numbers (0)
```

La unica diferencia que veo es que ha cambiado el SPI

5) Indica que selectores de trafico se envıan, tanto para initiator como responder.

```
Type Payload: Traffic Selector - Initiator (44) # 1
      Traffic Selector Type: TS_IPV4_ADDR_RANGE (7)
      Protocol ID: Unused
      Selector Length: 16
      Start Port: 0
      End Port: 65535
      Starting Addr: 10.18.1.0
      Ending Addr: 10.18.1.255

Type Payload: Traffic Selector - Responder (45) # 1
      Traffic Selector Type: TS_IPV4_ADDR_RANGE (7)
      Protocol ID: Unused
      Selector Length: 16
      Start Port: 0
      End Port: 65535
      Starting Addr: 10.18.2.0
      Ending Addr: 10.18.2.255
```

==========

4. Consulta SAD y SPD en cada uno de los extremos e incluye en la memoria la informaci´on relevante para el tunel IPsec. Fıjate en los valores SPI e indica si estos valores coinciden con los de la SA negociada previamente.

En r1:

```
root@alonsod:~/lab-ipsec# ip xfrm state
src 100.18.2.1 dst 100.18.4.4
        proto esp spi 0xcac84d29 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256)
0x65dbcbd41d83012b70b96661907ab72796d2a9801c4df0994c1279c2e208fdf0 128
        enc cbc(aes) 0x12df41085d4587ee90993257cec0ccc9
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
src 100.18.4.4 dst 100.18.2.1
```

```
        proto esp spi 0xcd5ee06d reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256)
0xcc5be9fb5b8f663a82bbb7abc753e46fd75db301e28aff6bba399e8b1c58b837 128
        enc cbc(aes) 0x847f5f689496a7897063f92eac3d39f5
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000

En r4:
root@alonsod:~/lab-ipsec# ip xfrm state
src 100.18.4.4 dst 100.18.2.1
        proto esp spi 0xcd5ee06d reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256)
0xcc5be9fb5b8f663a82bbb7abc753e46fd75db301e28aff6bba399e8b1c58b837 128
        enc cbc(aes) 0x847f5f689496a7897063f92eac3d39f5
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
src 100.18.2.1 dst 100.18.4.4
        proto esp spi 0xcac84d29 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256)
0x65dbcbd41d83012b70b96661907ab72796d2a9801c4df0994c1279c2e208fdf0 128
        enc cbc(aes) 0x12df41085d4587ee90993257cec0ccc9
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000


================================================================================
================================================================================
==========

5. Consulta la tabla de encaminamiento de la maquina en r1 y r4. ¿Que
ocurre con los paquetes enviados a las direcciones internas desde la
sucursal1 a la sucursal2? Justifica tu respuesta consultando la tabla
de encaminamiento 220 (creada por strongswan).

en r1:
root@alonsod:~/lab-ipsec# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use
Iface
default         100.18.2.2      0.0.0.0         UG    0      0        0
r1-eth1
10.18.1.0       *               255.255.255.0   U     0      0        0
r1-eth0
100.18.2.0      *               255.255.255.0   U     0      0        0
r1-eth1

root@alonsod:~/lab-ipsec# ip route list table 220
10.18.2.0/24 via 100.18.2.2 dev r1-eth1  proto static  src 10.18.1.1


en r4:
root@alonsod:~/lab-ipsec# route
Kernel IP routing table
```

```
Destination     Gateway         Genmask         Flags Metric Ref    Use
Iface
default         100.18.4.3      0.0.0.0         UG    0      0        0
r4-eth0
10.18.2.0       *               255.255.255.0   U     0      0        0
r4-eth1
100.18.4.0      *               255.255.255.0   U     0      0        0
r4-eth0

root@alonsod:~/lab-ipsec# ip route list table 220
10.18.1.0/24 via 100.18.4.3 dev r4-eth0  proto static  src 10.18.2.4
```

Vemos que para que se comunique la sucursal 1 con la 2 se usaran la direccion 100.18.2.2

++

2.2.2. Comunicacion usando IPsec (ESP en modo tunel)

1. Explica detalladamente que es lo que crees que ocurrirıa en r1 cuando:
pc1 le envıa un datagrama IP a pc2
pc2 le envıa un datagrama IP a pc1

En el apartado anterior hemos usado IKE, por lo que en principio los mensajes iran cifrados. El trafico saliente lo dejara pasar pero el entrante comprobara si tiene alguna SA asociada. De no ser asi lo descartaria.

==

2. Inicia las siguientes capturas de trafico:
En r1(eth1) y guarda los paquetes capturados en el fichero ipsec-01.cap :

```
root@alonsod:~/lab-ipsec# tcpdump -i r1-eth1 -s 0 -w ipsec-01.cap
tcpdump: listening on r1-eth1, link-type EN10MB (Ethernet), capture size
262144 bytes
^C9 packets captured
13 packets received by filter
0 packets dropped by kernel
```

En r4(eth0) y guarda los paquetes capturados en el fichero ipsec-02.cap :

```
root@alonsod:~/lab-ipsec# tcpdump -i r4-eth0 -s 0 -w ipsec-02.cap
tcpdump: listening on r4-eth0, link-type EN10MB (Ethernet), capture size
262144 bytes
^C13 packets captured
13 packets received by filter
0 packets dropped by kernel
```

En pc2 y guarda los paquetes capturados en el fichero ipsec-03.cap

```
root@alonsod:~/lab-ipsec# tcpdump -i pc2-eth0 -s 0 -w ipsec-03.cap
tcpdump: listening on pc2-eth0, link-type EN10MB (Ethernet), capture size
262144 bytes
^C10 packets captured
10 packets received by filter
0 packets dropped by kernel
```

hacemos el ping a pc2 desde pc1: (para saber la ip de pc2, lanzamos un xterm pc2 y hacemos ifconfig para ver cual es)

```
root@alonsod:~/lab-ipsec# ping -c 3 10.18.2.20
PING 10.18.2.20 (10.18.2.20) 56(84) bytes of data.
64 bytes from 10.18.2.20: icmp_seq=1 ttl=62 time=8.54 ms
64 bytes from 10.18.2.20: icmp_seq=2 ttl=62 time=9.31 ms
64 bytes from 10.18.2.20: icmp_seq=3 ttl=62 time=9.47 ms

--- 10.18.2.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 8.545/9.113/9.477/0.414 ms
```

Explica el contenido que puedes ver de los paquetes ESP que se corresponden con los paquetes echo request/echo reply capturados y como varıa el campo ESP Sequence.

Como ya se habian enviado mensajes con anterioridad, el numero ESP Sequence no empieza en cero, sino en 4.
p.e: captura: ipsec-01.cap
ESP SPI: 0xcbb14921 (3417393441)
ESP Sequence: 4

ESP SPI: 0xc5e4fa22 (3320117794)
ESP Sequence: 4

==========
==========
==========

3. ¿Con que TTL crees que se habran recibido los paquetes en pc2. Compruebalo observando la captura ipsec-03.cap. Explica el resultado

De primeras no es necesario ir a la captura ipsec-03.cap, ya que cuando hemos hecho el ping desde pc1, aparece cual es el TTL. En este caso es 62, ya que ha atravesado dos routers antes de llegar a pc2 (64 - 2 = 62)
Si nos metemos en la captura ipsec-03.cap, en la cabecera de IPv4, podremos ver lo siguiente:
Time to live: 62, lo cual ya adelantabamos dos lineas mas arriba.

==========
==========

4. Utilizando Wireshark podras descifrar los paquetes ESP si configuras en Wireshark las claves que se estan utilizando para el intercambio de mensajes ESP. Para ello deberas tomar las claves que has mostrado en SAD. Con la captura ipsec-01.cap cargada en Wireshark selecciona en el menu: Edit -> Preferences -> Protocols -> ESP marca la opcion de descifrar y autenticar paquetes y edita la configuracion para copiar allı los valores de cada uno de los SAs que muestra la SAD de r1 sin dejar espacios en blanco

```
root@alonsod:~/lab-ipsec# ip xfrm state
src 100.18.2.1 dst 100.18.4.4
        proto esp spi 0xccc63553 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256)
0x3d816e606fb96f5c46389291f9e96086bce24a78f2e194eeea628333e5d2985c 128
        enc cbc(aes) 0x5418ed7691eae584905a28295bba595f
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
src 100.18.4.4 dst 100.18.2.1
        proto esp spi 0xcfc77e47 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256)
0xf9bce8cd4b35ab031d544709d3dbaffa90b2de9494687f6d8b95cc786967bc39 128
        enc cbc(aes) 0xaafde3d64126272690e95e7bf0935e55
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
```

Si miramos en el fichero de esp_sa,veremos lo siguiente:

```
david@alonsod:~$ cd .config/wireshark/
david@alonsod:~/.config/wireshark$ cat esp_sa
# This file is automatically generated, DO NOT MODIFY.
"IPv4","100.18.2.1","100.18.4.4","0xccc63553","AES-CBC
[RFC3602]","0x5418ed7691eae584905a28295bba595f","HMAC-SHA-256-128
[RFC4868]","0x3d816e606fb96f5c46389291f9e96086bce24a78f2e194eeea628333e5d
2985c"
"IPv4","100.18.4.4","100.18.2.1","0xcfc77e47","AES-CBC
[RFC3602]","0xaafde3d64126272690e95e7bf0935e55","HMAC-SHA-256-128
[RFC4868]","0xf9bce8cd4b35ab031d544709d3dbaffa90b2de9494687f6d8b95cc78696
7bc39"
```

==========
==========

5. Explica el contenido de la cabecera ESP de los paquetes que se corresponden con los paquetes echo request/echo reply capturados, observa como ahora puedes ver todos los campos.

Si nos fijamos, aparecen por cada echo request, dos echo reply, ya que uno de ellos esta sin descifrar y el siguiente lo encontramos en claro.

```
ESP SPI: 0xcfc77e47 (3485957703)
ESP Sequence: 4
ESP IV: 3ab04477f9b232eb2f94f2298291d3c1
Pad: 0102030405060708090a
ESP Pad Length: 10
Next header: IPIP (0x04)
```

==========

6. ¿Observando la captura puedes saber si IPsec esta trabajando en modo tunel o en modo transporte?

Lo podemos saber. Mirando la captura, si vemos que aparecen dos cabeceras ip, estamos en modo tunel (como en este caso).
En cambio, si aparace solo una direccion ip, estamos en modo transporte (saldria la direccion publica directamente. En el modo tunel aparecen las direcciones publicas y las del tunel)

==========

7. ¿Que crees que ocurrira si se envıa trafico TCP/UDP de pc1 a pc2? ¿Por que? Inicia una captura nuevamente en r1(eth1) y guarda los paquetes capturados en el fichero ipsec-04.cap. Compruebalo utilizando nc para lanzar un cliente y un servidor en dichas maquinas, primero con UDP y despues con TCP. Examina la captura e indica que ves.

Como las comunicaciones utilizan el tunel, cualquier tipo de comunicacion ira cifrada.

He tenido que meter nuevos valores de ESP ya que se han reiniciado.

```
root@alonsod:~/lab-ipsec# ip xfrm state
src 100.18.2.1 dst 100.18.4.4
        proto esp spi 0xce88cbcc reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256)
0xa2e56e8b953cbb7d998f5536bbd6ddaedea791da0ce0c9f991298efa1ac6fe3d 128
        enc cbc(aes) 0x8064afc79244df93486a22815ed743be
        anti-replay context: seq 0x0, oseq 0x2, bitmap 0x00000000
src 100.18.4.4 dst 100.18.2.1
        proto esp spi 0xc59cbc79 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256)
0x6009b7347163c1a176cc9448f1f1b2f61511a8ea0586693a9fc62dd5cc9ea907 128
        enc cbc(aes) 0xc909563658e6dccc952eb57b9560bb0b
        anti-replay context: seq 0x4, oseq 0x0, bitmap 0x0000000f
```

Aparecian cifrados los paquetes pero al meter la configuracion actualizada de ESP, aparecen en claro.

==========
==========
==========

8. Inicia una captura nuevamente en r1(eth1) y guarda los paquetes capturados en el fichero ipsec-05.cap. Interrumpe IPsec en r1 e interrumpe la captura de paquetes. Utiliza Wireshark para visualizar los paquetes capturados explica que ocurre.

Al igual que en el apartado anterior, he lanzado un server y un client pero en este caso UDP
primero: hacemos captura en r1 --> tcpdump -i r1-eth1 -s 0 -w ipsec-05.cap
segundo: lanzamos el server UDP --> nc -l -u -p 7777
tercero: lanzamos el cliente UDP --> nc -u 10.18.1.10 7777

Si tenemos bien configurada la informacion ESP en wireshark, deberiamos poder ver los mensajes UDP.

++++++++++
++++++++++
++++++++++

2.3. ESP en modo transporte entre pc3 y r4

1. Realiza la configuracion en los ficheros que sean necesarios para permitir esta comunicacion. Incluye estos ficheros en la memoria. No olvides guardar en la maquina real los ficheros que modifiques, de esta forma tendras una copia de seguridad.

Para que funcione ipsec en modo transporte, lo que voy a hacer es modificar tres ficheros:

$ en pc3: pc3/etc/ipsec.conf

```
# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
      # strictcrlpolicy=yes
      # uniqueids = no
      charondebug="all"

# Add connections here.

conn %default
      ikelifetime=24h
      rekeymargin=3m
      keyingtries=1
      keyexchange=ikev2
      ike=aes128-sha256-modp3072!
```

```
        esp=aes128-sha256-modp3072!

# Sample VPN connections

conn host-host
      leftprotoport=tcp/3333
      left=100.18.7.30
      leftcert=pc3Cert.pem
      leftid=@pc3
      rightprotoport=tcp/4444
      right=100.18.4.4
      rightid=@r4
      rightcert=r4Cert.pem
      type=transport
      auto=add


#conn sample-self-signed
#      leftsubnet=10.1.0.0/16
#      leftcert=selfCert.der
#      leftsendcert=never
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightcert=peerCert.der
#      auto=start

#conn sample-with-ca-cert
#      leftsubnet=10.1.0.0/16
#      leftcert=myCert.pem
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightid="C=CH, O=Linux strongSwan CN=peer name"
#      auto=start
```

$ pc3/etc/ipsec.secrets : ## Esto es bastante importante ya que sino al intentar arrancar host-host nos dice que no encuentra la clave privada de pc3 (LOGICO!!) ##

```
: RSA pc3Key.pem
```

$ y en r4: r4/etc/ipsec.conf

```
# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
      # strictcrlpolicy=yes
      # uniqueids = no
      charondebug="all"

# Add connections here.
```

```
conn %default
      ikelifetime=24h
      rekeymargin=3m
      keyingtries=1
      keyexchange=ikev2
      ike=aes128-sha256-modp3072!
      esp=aes128-sha256-modp3072!

# Sample VPN connections
# para este apartado podemos comentar conn net-net
#conn net-net
      #left=100.18.4.4
      #leftcert=r4Cert.pem
      #leftid=@r4
      #leftsubnet=10.18.2.0/24
      #right=100.18.2.1
      #rightid=@r1
      #rightsubnet=10.18.1.0/24
      #type=tunnel
      #auto=add

conn host-host
      leftprotoport=tcp/4444
      left=100.18.4.4
      leftcert=r4Cert.pem
      leftid=@r4
      rightprotoport=tcp/3333
      right=100.18.7.30
      rightid=@pc3
      rightcert=pc3Cert.pem
      type=transport
      auto=add

#conn sample-self-signed
#      leftsubnet=10.1.0.0/16
#      leftcert=selfCert.der
#      leftsendcert=never
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightcert=peerCert.der
#      auto=start

#conn sample-with-ca-cert
#      leftsubnet=10.1.0.0/16
#      leftcert=myCert.pem
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightid="C=CH, O=Linux strongSwan CN=peer name"
#      auto=start
```

LO QUE DEBE SALIR SI HEMOS HECHO TODO BIEN:

```
$ EN PC3
root@alonsod:~/lab-ipsec# ipsec start
Starting strongSwan 5.3.5 IPsec [starter]...
```

```
$ EN R4:
root@alonsod:~/lab-ipsec# ipsec start
Starting strongSwan 5.3.5 IPsec [starter]...

$ EN PC3
root@alonsod:~/lab-ipsec# ipsec up host-host
initiating IKE_SA host-host[1] to 100.18.7.30
generating IKE_SA_INIT request 0 [ SA KE No N(NATD_S_IP) N(NATD_D_IP)
N(HASH_ALG) ]
sending packet: from 100.18.4.4[500] to 100.18.7.30[500] (576 bytes)
received packet: from 100.18.7.30[500] to 100.18.4.4[500] (609 bytes)
parsed IKE_SA_INIT response 0 [ SA KE No N(NATD_S_IP) N(NATD_D_IP)
CERTREQ N(HASH_ALG) N(MULT_AUTH) ]
received cert request for "C=ES, O=myCA, CN=My Root CA"
sending cert request for "C=ES, O=myCA, CN=My Root CA"
authentication of 'r4' (myself) with RSA_EMSA_PKCS1_SHA256 successful
sending end entity cert "C=ES, O=myCA, CN=r4"
establishing CHILD_SA host-host
generating IKE_AUTH request 1 [ IDi CERT N(INIT_CONTACT) CERTREQ IDr AUTH
N(USE_TRANSP) SA TSi TSr N(MOBIKE_SUP) N(ADD_4_ADDR) N(MULT_AUTH)
N(EAP_ONLY) ]
sending packet: from 100.18.4.4[4500] to 100.18.7.30[4500] (1616 bytes)
received packet: from 100.18.7.30[4500] to 100.18.4.4[4500] (1568 bytes)
parsed IKE_AUTH response 1 [ IDr CERT AUTH N(USE_TRANSP) SA TSi TSr
N(AUTH_LFT) N(MOBIKE_SUP) N(NO_ADD_ADDR) ]
received end entity cert "C=ES, O=myCA, CN=pc3"
  using certificate "C=ES, O=myCA, CN=pc3"
  using trusted ca certificate "C=ES, O=myCA, CN=My Root CA"
checking certificate status of "C=ES, O=myCA, CN=pc3"
certificate status is not available
  reached self-signed root ca with a path length of 0
authentication of 'pc3' with RSA_EMSA_PKCS1_SHA256 successful
IKE_SA host-host[1] established between 100.18.4.4[r4]...100.18.7.30[pc3]
scheduling reauthentication in 86164s
maximum IKE_SA lifetime 86344s
CHILD_SA host-host{1} established with SPIs c06d741f_i c80c106a_o and TS
100.18.4.4/32[tcp/4444] === 100.18.7.30/32[tcp/3333]
connection 'host-host' established successfully

comprobamos que esta todo bien:

en r4:
root@alonsod:~/lab-ipsec# ip xfrm state
src 100.18.4.4 dst 100.18.7.30
        proto esp spi 0xc80c106a reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0x902c14d57a93af3db37f7193e511994582136d6f65dd2b70bd2c2826f8600188 128
        enc cbc(aes) 0xe15bbc94ea1e786228ff6d4448982248
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.4.4/32 dst 100.18.7.30/32
src 100.18.7.30 dst 100.18.4.4
        proto esp spi 0xc06d741f reqid 1 mode transport
```

```
        replay-window 32
        auth-trunc hmac(sha256)
0x32960aeb56b2effc883a4be571a1252c2c77e4cdb26f804186475f7e46e583a1 128
        enc cbc(aes) 0x7c18249c2d33e22e48a72f65cc1dc249
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.7.30/32 dst 100.18.4.4/32
```

==========

2. Inicia una captura en pc3 para guardar su contenido en el fichero ipsec-06.cap y establece la configuracion ESP en modo transporte entre ambas maquinas. Ejecuta un ping desde la maquina pc3. Interrumpe la captura y explica su contenido.

$ HACEMOS EL PING EN PC3 HACIA R4:

```
root@alonsod:~/lab-ipsec# ping 100.18.4.4
PING 100.18.4.4 (100.18.4.4) 56(84) bytes of data.
64 bytes from 100.18.4.4: icmp_seq=1 ttl=62 time=16.5 ms
64 bytes from 100.18.4.4: icmp_seq=2 ttl=62 time=1.31 ms
64 bytes from 100.18.4.4: icmp_seq=3 ttl=62 time=0.207 ms
64 bytes from 100.18.4.4: icmp_seq=4 ttl=62 time=0.311 ms
64 bytes from 100.18.4.4: icmp_seq=5 ttl=62 time=0.193 ms
^C
--- 100.18.4.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4031ms
rtt min/avg/max/mdev = 0.193/3.718/16.567/6.438 ms
```

En la captura podemos ver el inicio de la comunicacion a traves del modo transporte, con los mensajes IKE correspondientes (IKE_SA_INIT y IKE_AUTH)
A continuacion vemos los tipicos mensajes echo request y echo reply de un ping (en este caso entre pc3 y r4)

==========

3. Observa el contenido de SAD y SDP en pc3 y explica su contenido

Vemos que aparecen los spi, se indica el modo transporte, el tipo de cifrado que se utiliza, etc.
Tambien se ven las direcciones ip de pc3 y r4.

$ EN PC3:

```
root@alonsod:~/lab-ipsec# ip xfrm state
src 100.18.7.30 dst 100.18.4.4
        proto esp spi 0xc7f2ff23 reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0xb06cdb2d15983cf13e4303757b704b4ff88a1497c254076d3c1d69db45be818d 128
```

```
        enc cbc(aes) 0x8c07bd48293725c298064e92f073a9db
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.7.30/32 dst 100.18.4.4/32
src 100.18.4.4 dst 100.18.7.30
        proto esp spi 0xc55f0e6e reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0xe1acb8a5511b506a6546585dad491dcab93a75a062dce476fb682598c234cbd0 128
        enc cbc(aes) 0x34f5799b29caece9ca90a4db9234215b
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.4.4/32 dst 100.18.7.30/32


root@alonsod:~/lab-ipsec# ip xfrm policy
src 100.18.4.4/32 dst 100.18.7.30/32 proto tcp sport 4444 dport 3333
        dir in priority 2816
        tmpl src 0.0.0.0 dst 0.0.0.0
                proto esp reqid 1 mode transport
src 100.18.7.30/32 dst 100.18.4.4/32 proto tcp sport 3333 dport 4444
        dir out priority 2816
        tmpl src 0.0.0.0 dst 0.0.0.0
                proto esp reqid 1 mode transport
src 0.0.0.0/0 dst 0.0.0.0/0
        socket in priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket out priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket in priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket out priority 0
src ::/0 dst ::/0
        socket in priority 0
src ::/0 dst ::/0
        socket out priority 0
src ::/0 dst ::/0
        socket in priority 0
src ::/0 dst ::/0
        socket out priority 0

$ EN R4:
root@alonsod:~/lab-ipsec#  ip xfrm state
src 100.18.4.4 dst 100.18.7.30
        proto esp spi 0xc55f0e6e reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0xe1acb8a5511b506a6546585dad491dcab93a75a062dce476fb682598c234cbd0 128
        enc cbc(aes) 0x34f5799b29caece9ca90a4db9234215b
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.4.4/32 dst 100.18.7.30/32
src 100.18.7.30 dst 100.18.4.4
        proto esp spi 0xc7f2ff23 reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0xb06cdb2d15983cf13e4303757b704b4ff88a1497c254076d3c1d69db45be818d 128
```

```
        enc cbc(aes) 0x8c07bd48293725c298064e92f073a9db
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.7.30/32 dst 100.18.4.4/32


root@alonsod:~/lab-ipsec# ip xfrm policy
src 100.18.7.30/32 dst 100.18.4.4/32 proto tcp sport 3333 dport 4444
        dir in priority 2816
        tmpl src 0.0.0.0 dst 0.0.0.0
                proto esp reqid 1 mode transport
src 100.18.4.4/32 dst 100.18.7.30/32 proto tcp sport 4444 dport 3333
        dir out priority 2816
        tmpl src 0.0.0.0 dst 0.0.0.0
                proto esp reqid 1 mode transport
src 0.0.0.0/0 dst 0.0.0.0/0
        socket in priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket out priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket in priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket out priority 0
src ::/0 dst ::/0
        socket in priority 0
src ::/0 dst ::/0
        socket out priority 0
src ::/0 dst ::/0
        socket in priority 0
src ::/0 dst ::/0
        socket out priority 0
```

==========

4. Inicia una captura en pc3 para guardar su contenido en el fichero ipsec-07.cap y establece la configuracion ESP en modo transporte entre ambas maquinas. Usando nc ejecuta un servidor TCP en r4 en el puerto 4444 al que se conecta un cliente desde la maquina pc3 y el puerto 3333. Escribe algo desde el cliente para que se transmita al servidor. Interrumpe la captura y responde las siguientes cuestiones:

a) ¿Que campos observas en los paquetes ESP?

PARA LA CAPTURA ipsec-07.cap, los valores de spi eran los siguientes:
PC3:

```
root@alonsod:~/lab-ipsec# ip xfrm state
src 100.18.7.30 dst 100.18.4.4
        proto esp spi 0xc7f2ff23 reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0xb06cdb2d15983cf13e4303757b704b4ff88a1497c254076d3c1d69db45be818d 128
        enc cbc(aes) 0x8c07bd48293725c298064e92f073a9db
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
```

```
        sel src 100.18.7.30/32 dst 100.18.4.4/32
src 100.18.4.4 dst 100.18.7.30
        proto esp spi 0xc55f0e6e reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0xe1acb8a5511b506a6546585dad491dcab93a75a062dce476fb682598c234cbd0 128
        enc cbc(aes) 0x34f5799b29caece9ca90a4db9234215b
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.4.4/32 dst 100.18.7.30/32

root@alonsod:~/lab-ipsec# ip xfrm state
src 100.18.4.4 dst 100.18.7.30
        proto esp spi 0xc55f0e6e reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0xe1acb8a5511b506a6546585dad491dcab93a75a062dce476fb682598c234cbd0 128
        enc cbc(aes) 0x34f5799b29caece9ca90a4db9234215b
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.4.4/32 dst 100.18.7.30/32
src 100.18.7.30 dst 100.18.4.4
        proto esp spi 0xc7f2ff23 reqid 1 mode transport
        replay-window 32
        auth-trunc hmac(sha256)
0xb06cdb2d15983cf13e4303757b704b4ff88a1497c254076d3c1d69db45be818d 128
        enc cbc(aes) 0x8c07bd48293725c298064e92f073a9db
        anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
        sel src 100.18.7.30/32 dst 100.18.4.4/32
```

Lanzamos el servidor(r4): nc -l -p 4444
Ahora lanzamos el cliente(pc3): nc -p 3333 100.18.4.4 4444

Si vemos la captura, vemos que solo aparecen mensajes ESP. Metiendo los valores adecuados conseguimos descifrar el contenido de los paquetes.

Este es el resumen de lo que he añadido en wireshark:

```
"IPv4","100.18.7.30","100.18.4.4","0xc7f2ff23","AES-CBC
[RFC3602]","0x8c07bd48293725c298064e92f073a9db","HMAC-SHA-256-128
[RFC4868]","0xb06cdb2d15983cf13e4303757b704b4ff88a1497c254076d3c1d69db45b
e818d"
"IPv4","100.18.4.4","100.18.7.30","0xc55f0e6e","AES-CBC
[RFC3602]","0x34f5799b29caece9ca90a4db9234215b","HMAC-SHA-256-128
[RFC4868]","0xe1acb8a5511b506a6546585dad491dcab93a75a062dce476fb682598c23
4cbd0"
```

a) ¿Qué campos observas en los paquetes ESP?
Antes de descifrar los paquetes, vemos que en la cabecera de ESP aparece el numero de secuencia y los SPI.

b) Descifra el contenido de ESP para ver exactamente qué es lo que se envıá. ¿Qué diferencias observas con respecto al modo túnel?

A diferencia del modo tunel, lo que ocurre ahora es que no aparecen dos cabeceras IP, sino solo una.

c) ¿Con qué TTL se reciben los paquetes IP en r4? ¿Por qué?
El TTL es 62, lo cual es logico ya que atraviesa dos routers antes de llegar al destino que es r4. Si fuese modo tunel en vez de modo transporte, lo que ocurriria es que habria dos TTL, uno con la ip del tuenl y otra con la ip publica. En ese caso solo se reduciria solo una de ellas.

++

2.4. AH en modo transporte entre r1 y r4

1. ¿Qué crees que ocurrirá si se envıá tráfico de pc1 a pc2? ¿Por qué? Inicia una captura nuevamente en r1(eth1) y guarda los paquetes capturados en el fichero ipsec-08.cap. Compruébalo utilizando ping entre dichas máquinas. Examina la captura e indica qué ves.

hay que cambiar la linea de esp por esta --> ah=sha256-sha384!

para evitar olvidar lo que habia antes, en vez de borrarlo lo he comentado. Lo que he comentado son las siguientes lineas
en r1: #esp=aes128-sha256-modp3072!, # leftsubnet=10.18.1.0/24, # rightsubnet=10.18.2.0/24, # type=tunnel

en r4: # esp=aes128-sha256-modp3071, # leftsubnet=10.18.2.0/24, # rightsubnet=10.18.1.0/24, # type=tunnel

la nueva configuracion de r1: ipsec.conf

```
# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
      # strictcrlpolicy=yes
      # uniqueids = no
      charondebug="all"

# Add connections here.

conn %default
      ikelifetime=24h
      rekeymargin=3m
      keyingtries=1
      keyexchange=ikev2
      ike=aes128-sha256-modp3072!
      #esp=aes128-sha256-modp3072!
      ah=sha256-sha384!

# Sample VPN connections

conn net-net
```

```
      left=100.18.2.1
      leftcert=r1Cert.pem
      leftid=@r1
      # leftsubnet=10.18.1.0/24
      right=100.18.4.4
      rightid=@r4
      rightcert=r4Cert.pem
      # rightsubnet=10.18.2.0/24
      # type=tunnel
      type=transport
      auto=add

#conn sample-self-signed
#      leftsubnet=10.1.0.0/16
#      leftcert=selfCert.der
#      leftsendcert=never
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightcert=peerCert.der
#      auto=start

#conn sample-with-ca-cert
#      leftsubnet=10.1.0.0/16
#      leftcert=myCert.pem
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightid="C=CH, O=Linux strongSwan CN=peer name"
#      auto=start
```

y la de r4: ipsec.conf

```
# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
      # strictcrlpolicy=yes
      # uniqueids = no
      charondebug="all"

# Add connections here.
conn %default
      ikelifetime=24h
      rekeymargin=3m
      keyingtries=1
      keyexchange=ikev2
      ike=aes128-sha256-modp3071
      # esp=aes128-sha256-modp3071
      ah=sha256-sha384!

# Sample VPN connections
conn net-net
      left=100.18.4.4
      leftcert=r4Cert.pem
```

```
      leftid=@r4
      # leftsubnet=10.18.2.0/24
      right=100.18.2.1
      rightid=@r1
      rightcert=r1Cert.pem
      # rightsubnet=10.18.1.0/24
      # type=tunnel
      type=transport
      auto=add

conn host-host
      leftprotoport=tcp/4444
      left=100.18.4.4
      leftcert=r4Cert.pem
      leftid=@r4
      rightprotoport=tcp/3333
      right=100.18.7.30
      rightid=@pc3
      rightcert=pc3Cert.pem
      type=transport
      auto=add

#conn sample-self-signed
#      leftsubnet=10.1.0.0/16
#      leftcert=selfCert.der
#      leftsendcert=never
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightcert=peerCert.der
#      auto=start

#conn sample-with-ca-cert
#      leftsubnet=10.1.0.0/16
#      leftcert=myCert.pem
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightid="C=CH, O=Linux strongSwan CN=peer name"
#      auto=start
```

No funciona el ping entre pc1 y pc2 ya que la configuracion que hemos hecho es de maquin a maquin (notese que hemos comentado las lineas que anuncian las subredes)
Por tanto, el unico ping que deberia funcionar (usando ah en modo transporte) es entre r1 y r4.

==========
==========
==========

2. Explica, sólo partiendo de los datos que aparecen en la captura, por qué se puede saber que la configuración IPsec realizada está en modo túnel AH.

Voy a hacer la captura entre r1 y r4, ya que entre pc1 y pc2 no es. Ahora podemos ver que se esta utilizando AH en modo transporte. Lo podemos saber gracias a que aparece una cabecera que lo indica (Authentication Header), y sabemos que se esta utilizando el modo transporte ya que no aparecen dos cabeceras ip como en el caso de modo tunel.

==========
==========
==========

3. Explica por qué crees que no se puede realizar un ataque de reproducción con esta configuración.

Porque cada paqute lleva un numero de secuencia, que hace inviable que se produzcan ataques de reproduccion. En el caso de intentar modificicar el paquete, lo mas probable es que se produzcan fallos en cuanto a la autenticacion.