VPN con IPsec

Seguridad en Redes de Ordenadores

Departamento de Teoría de la Señal y Comunicaciones y Sistemas Telemáticos y Computación (GSyC)

Abril de 2018

Contenidos

Contenidos

Introducción

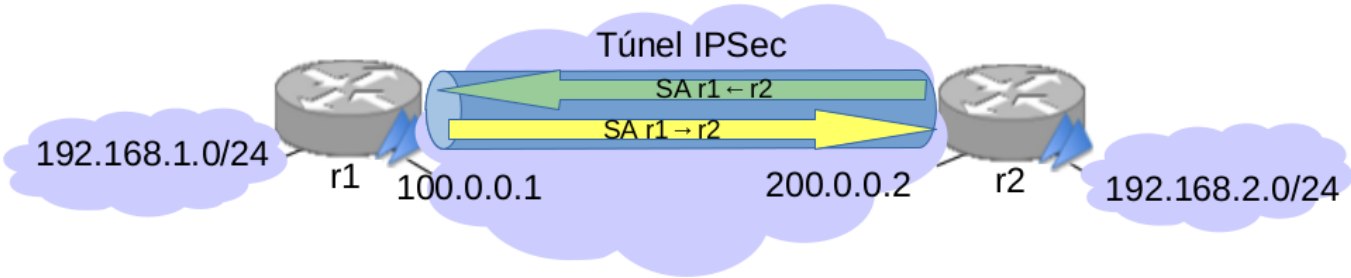
- Utilizaremos **strongSwan** para establecer la configuración de IPsec que proporciona cifrado y autenticación entre servidores y clientes.
- *strongswan* es un demonio para el **establecimiento de claves** a través del protocolo IKE que permite negociar y establecer asociaciones de seguridad (SA) y políticas de seguridad (SP). El demonio `charon` se encarga de gestionar la configuración IKEv2 dentro de *strongswan*.
- El tráfico IPsec lo gestiona la implementación de IPsec del sistema operativo.
- Ficheros de configuración:
  - **/etc/ipsec.conf**: configuración de las comunicaciones IPsec.
  - **/etc/ipsec.secrets**: lista de secretos (pre-shared keys) y nombres de las claves privadas.
  - **/etc/ipsec.d**: directorio donde se almacenan los certificados y las claves privadas.
  - **/etc/strongswan.conf**: configuración global.
- Una comunicación IPsec queda determinada por los dos puntos finales que intervienen en la comunicación IPsec. Los puntos finales se denominan `left` y `right`. Según el punto donde se esté realizando la configuración estos dos lados quedan definidos de la siguiente forma:
  - left: es el lado local, donde se está almacenando la configuración actual (`left=local`).
  - right: es el lado remoto (`right=remote`), con el que se desea establecer la comunicación.

Contenidos

Contenidos

Configuración túnel ESP entre dos máquinas

- Configuración de un túnel ESP para establecer la comunicación entre las máquinas de la subred 192.168.1.0/24 y la subred 192.168.2.0/24 a través de una red no segura.



- La configuración se realizará en las máquinas extremos del túnel: r1 y r2. Será necesario definir 2 SA, una para cada sentido de la comunicación.

ipsec.conf (I)

- Contiene la información de configuración de IPsec dividida en 3 secciones.
- Los comentarios se indican con el carácter #.
- as opciones de cada sección deben estar sangradas hacia la derecha:
  - config setup: parámetros generales de configuración

```
config setup
    charondebug="all"
```

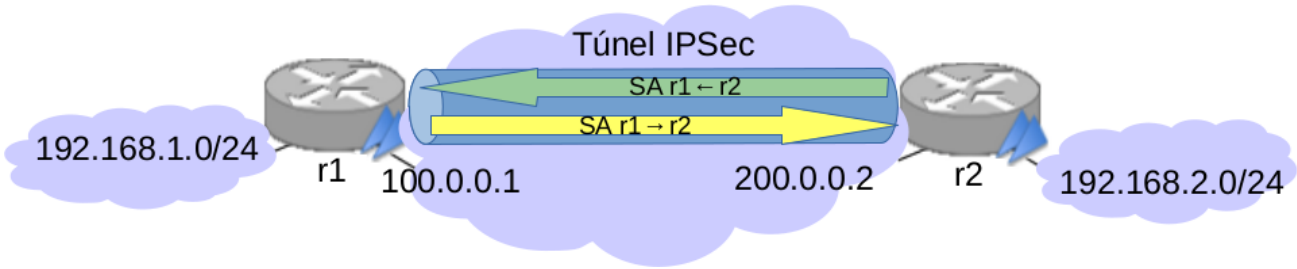
  - conn: parámetros de una conexión particular. Los parámetros comunes a todas las conexiones se definen bajo la sección `conn %default`. A continuación se pueden configurar tantas conexiones como sean necesarias, cada conexión se le asigna un nombre, por ejemplo `conn net-net`:

```
conn %default
    ikelifetime=60m                 # tiempo de vida de IKE SA
    rekeymargin=3m                  # tiempo de espera para renegociar
    keyingtries=1                   # número de intentos para negociar con IKE
    keyexchange=ikev2               # IKEv1/IKEv2
    ike=aes128-sha256-modp3072!     # cifrado-integridad-grupoDH para IKE
    esp=aes128-sha256-modp3072!     # cifrado-integridad-grupoDH para ESP

 conn net-net
    # Configuración específica del túnel ESP entre dos máquinas
    # ...
```

ipsec.conf (II)

- Cada extremo del túnel IPsec tendrá un fichero de configuración del túnel y quedará definido en la sección `conn` determinada. Los parámetros a definir estarán definidos dependiendo del lado `left` o `right` del túnel:
  - left: parámetros asociados al lado local
  - right: parámetros asociados al lado remoto
- Supongamos que tenemos una configuración en modo túnel ESP entre 2 routers intermedios, `r1` y `r2`, sus ficheros `ipsec.conf` tendrán la siguiente sección `conn` que define los parametros de esta comunicación entre ambos. Esta sección se le ha asignado el identificador `net-net`:



En `ipsec.conf` de **r1**:

```
conn net-net
        left=100.0.0.1
        leftcert=r1Cert.pem
        leftid=@r1
        leftsubnet=192.168.1.0/24
        right=200.0.0.2
        rightid=@r2
        rightsubnet=192.168.2.0/24
        type=tunnel
        auto=add
```

En `ipsec.conf` de **r2**:

```
conn net-net
        left=200.0.0.2
        leftcert=r2Cert.pem
        leftid=@r2
        leftsubnet=192.168.2.0/24
        right=100.0.0.1
        rightid=@r1
        rightsubnet=192.168.1.0/24
        type=tunnel
        auto=add
```

- En `r1` debe estar almacenado su certificado de clave pública `r1Cert.pem` en la carpeta `/etc/ipsec.d/certs` y de forma análoga en `r2`, su certificado de clave pública `r2Cert.pem` debe estar en la carpeta `/etc/ipsec.d/certs`.

ipsec.secrets

- Este fichero contiene el nombre de la clave privada RSA que identifica la máquina local y que deberá estar almacenado en la carpeta `/etc/ipsec.d/private`. También puede contener las claves compartidas entre ambos extremos si quisieran utilizar pre-shared key.
- Cuando se usa RSA: las dos máquinas que intervienen en la configuración del túnel deben tener en este fichero el nombre del fichero donde se encuentra su clave privada:
  - En `r1` el fichero `/etc/ipsec.secrets` debe contener el nombre del fichero donde se encuentra la clave privada de `r1`:

    ```
    : RSA r1Key.pem
    ```
  - En `r2` el fichero `/etc/ipsec.secrets` debe contener el nombre del fichero donde se encuentra la clave privada de `r2`:

    ```
    : RSA r2Key.pem
    ```

  Nótese que entre el carácter ":" y la palabra RSA hay un espacio.

Contenidos

Arranque del túnel ESP

- Se inicia `ipsec` en los dos extremos:

  ```
  ipsec start
  ```

- En el caso de túnel entre dos routers, cualquier extremo puede comenzar el establecimiento del túnel para ello se ejecuta en uno de los extremos del túnel la activación de la conexión con el identificador que se le había asignado en el fichero `ipsec.conf` (en nuestro ejemplo `net-net`):

  ```
  ipsec up net-net
  ```

- A partir de ese momento comienza la negociación de los algoritmos de cifrado y autenticación entre los dos extremos usando el protocolo IKEv2 y si se configurarán los SA y SP.

Contenidos

Estado del túnel ESP

- El estado en cada extremo se puede comprobar con el siguiente comando:

```
r1:~# ipsec status
Security Associations:
     net-net[1]: ESTABLISHED 31 seconds ago, 100.0.0.1[r1]...200.0.0.2[r2]
     net-net{1}:  INSTALLED, TUNNEL, ESP SPIs: cb3dc960_i cfb9dfca_o
     net-net{1}:    192.168.1.0/24 === 102.168.2.0/24

r2:~# ipsec status
Security Associations:
     net-net[1]: ESTABLISHED 3 minutes ago, 200.0.0.2[r2]...100.0.0.1[r1]
     net-net{1}:  INSTALLED, TUNNEL, ESP SPIs: cfb9dfca_i cb3dc960_o
     net-net{1}:    192.168.2.0/24 === 102.168.1.0/24
```

Transform (xfrm)

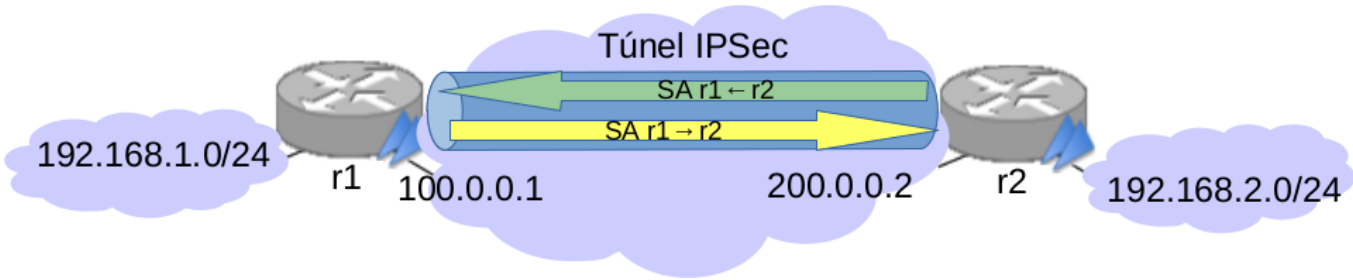
- `xfrm` (transform) es un framework utilizado para manipular paquetes a la entrada y salida de una máquina atendiendo a reglas IPsec, denominadas políticas de seguridad (Security Policy, SP).
- Una SP se aplica a un paquete si coinciden los selectores de la política con los parámetros del paquete (direcciones IP origen/destino, puertos, etc), en cuyo caso debe procesarlo IPsec.
- Para que lo procese IPsec es necesario consultar la SA que determina los parámetros de seguridad que le aplican a dicho paquete.
- xfrm gestiona SAD (Security Asociation Database) y la SPD (Security Policy Database).

Políticas (xfrm)

- **Política in**: se recibe un paquete dirigido a la dirección IP del router.
  - De la cabecera se obtienen SPI, las direcciones IP y el protocolo que se está usando ESP/AH. Con esta información se busca en SAD, donde debería existir SA.
    - Si no existe SA se provoca un error.
    - Si existe, el SA indica qué algoritmos se necesitan para descifrar/autenticar el paquete y se procede a ello.
  - A partir de SA se obtiene la política correspondiente SP en SDP para comprobar si el mecanismo de seguridad especificado en la política es el que se tenía que aplicar al paquete.
- **Política out/fwd**: el router genera/reenvía un paquete.
  - Se busca en SPD para ver si cumple los selectores de la política (IP origen, IP destino, etc) y saber si es necesario que lo procese IPsec, si se descarta o si se envía en claro.
  - Si se necesita aplicar IPsec la máquina consultará SAD.
    - Si existe SA se aplicarán los parámetros de seguridad de SA.
    - Si no existe SA, se avisará a IKE para que establezca SA.

Security Association Database (SAD) (I)

- Para el intercambio de información entre 2 máquinas deben existir al menos 2 SAs, uno para cada sentido.
- Cada SA contiene la información necesaria para gestionar un paquete en IPsec: el SPI, algoritmo de cifrado, clave de cifrado, algoritmo de autenticación, clave de autenticación.



Security Association Database (SAD) (II)

- Por ejemplo para ver SAD en `r1`:

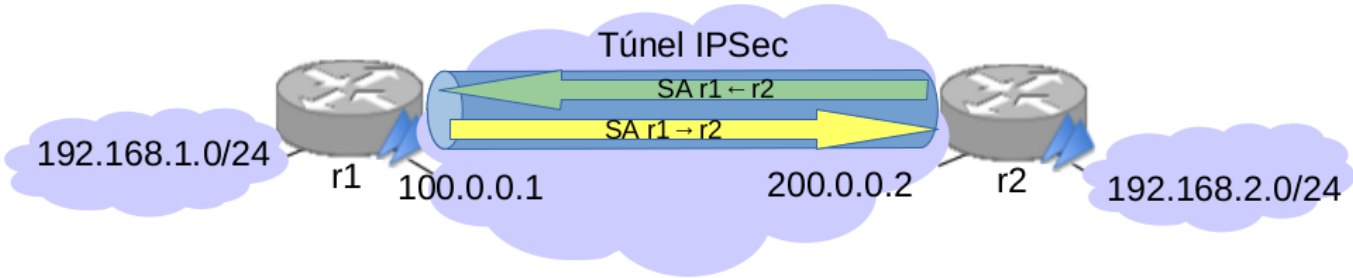
```
r1:~# ip xfrm state
src 100.0.0.1 dst 200.0.0.2
        proto esp spi 0xcfb9dfca reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256) 0x972feeb4fa34659af67b1cb1a2f098
                8331b61c5f2f16961782ad6202db5cd62e 128
        enc cbc(aes) 0xf024505f8f456a6a0f78a6cb8c5386b4

src 200.0.0.2 dst 100.0.0.1
        proto esp spi 0xcb3dc960 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256) 0x91aa030b659b33a4d829b4aee2945f
                266a050a750323171e31fc3d063544de52 128
        enc cbc(aes) 0x782ef28cdbbfd53e5272a1316af9eb6f
```

- La SA muestra toda la información necesaria para descifrar y autenticar el paquete, por ejemplo para el sentido 200.0.0.2 hacia 100.0.0.1:
  - Algoritmo de cifrado AES-CBC, clave: 0x782ef28cdbbfd53e5272a1316af9eb6f
  - Algoritmo de integridad HMAC-SHA256, clave: 0x91aa030b659b33a4d829b4aee2945f266a050a750323171e31fc3d063544de52
- El identificador reqid 1 debe ser igual al que se muestre en SP almacenada en SPD. Este identificador relaciona la SA con la SP.

Security Association Database (SPD)(I)

- En SPD se encuentran las reglas para aplicar a los paquetes a la entrada, salida y reenvío.



- SPD en r1 para la configuración de un túnel IPsec entre `r1` y `r2` debería contener las siguientes entradas para gestionar los paquetes que requieren IPsec:
  - **Política in**: paquete destinado al propio r1, que debe venir de la 192.168.2.0/24 e ir dirigido a la 192.168.1.0/24. Con origen en la 200.0.0.2 y destino 100.0.0.1.
  - **Política out**: paquete creado por r1 que va de la 192.168.1.0/24 a la 192.168.2.0/24. Con origen en la 100.0.0.1 y destino 200.0.0.2.
  - **Política fwd**: paquete que hay que reenviar en r1 y va de la 192.168.2.0/24 a la 192.168.1.0/24. Con origen en la 100.0.0.1 y destino 200.0.0.2.

Security Association Database (SPD)(II)

- Por ejemplo SDP en `r1`:

```
r1:~# ip xfrm policy

src 192.168.2.0/24 dst 192.168.1.0/24
        dir fwd priority 1859 ptype main
        tmpl src 200.0.0.2 dst 100.0.0.1
                proto esp reqid 1 mode tunnel

src 192.168.2.0/24 dst 192.168.1.0/24
        dir in priority 1859 ptype main
        tmpl src 200.0.0.2 dst 100.0.0.1
                proto esp reqid 1 mode tunnel

src 192.168.1.0/24 dst 192.168.2.0/24
        dir out priority 1859 ptype main
        tmpl src 100.0.0.1 dst 200.0.0.2
                proto esp reqid 1 mode tunnel
```

El identificador reqid 1 debe ser igual al que se muestre en SA almacenada en SAD. Este identificador relaciona la SP con la SA.

Tabla de encaminamiento para IPsec

- En Linux existen por defecto 3 tablas de encaminamiento (*policy routing tables*) que se identifican por su `<idtable>`:
  - **Tabla 255 o local**: contiene las rutas de broadcast (*scope link*) y las locales (*scope host*). Prioridad 0 (la más alta).
  - **Tabla 254 o main**: tabla de encaminamiento que se muestra al ejecutar `route`. Prioridad 32766.
  - **Tabla 253 o default**. Prioridad 32767. Tabla especial para un tratamiento de post-procesado.
- Las tablas se consultan por su prioridad para buscar una regla que encaje con las características del paquete que se desea encaminar.
- Strongswan crea una tabla, la tabla 220 con prioridad 220, para insertar las reglas de encaminamiento para IPsec.
- Para consultar estas tablas se usa el comando `ip`:

```
ip route list table <idtable>
```

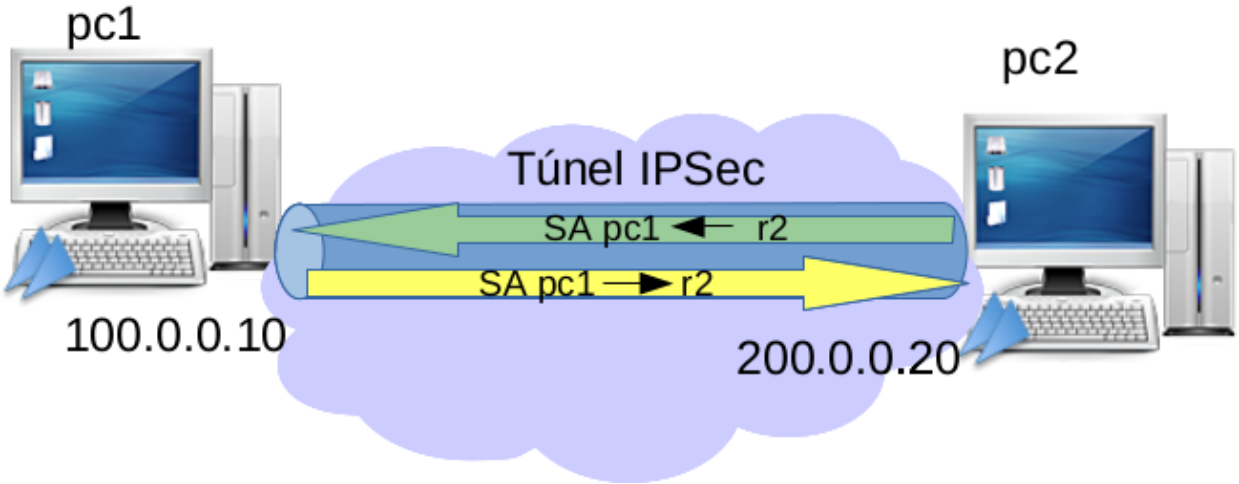
- Al arrancar IPsec se insertará en la tabla 220 la/s regla/s que permiten saber qué paquetes necesitarán IPsec.

Contenidos

Contenidos

ipsec.conf

- La configuración entre los dos extremos es similar al ejemplo anterior, salvo que en uno de los extremos la comunicación sólo es posible desde una máquina concreta.



En `ipsec.conf` de `pc1`:

```
conn host-host
        left=100.0.0.10
        leftcert=pc1Cert.pem
        leftid=@pc1
        right=200.0.0.20
        rightid=@pc2
        rightcert=pc2Cert.pem
        type=transport
        auto=add
```

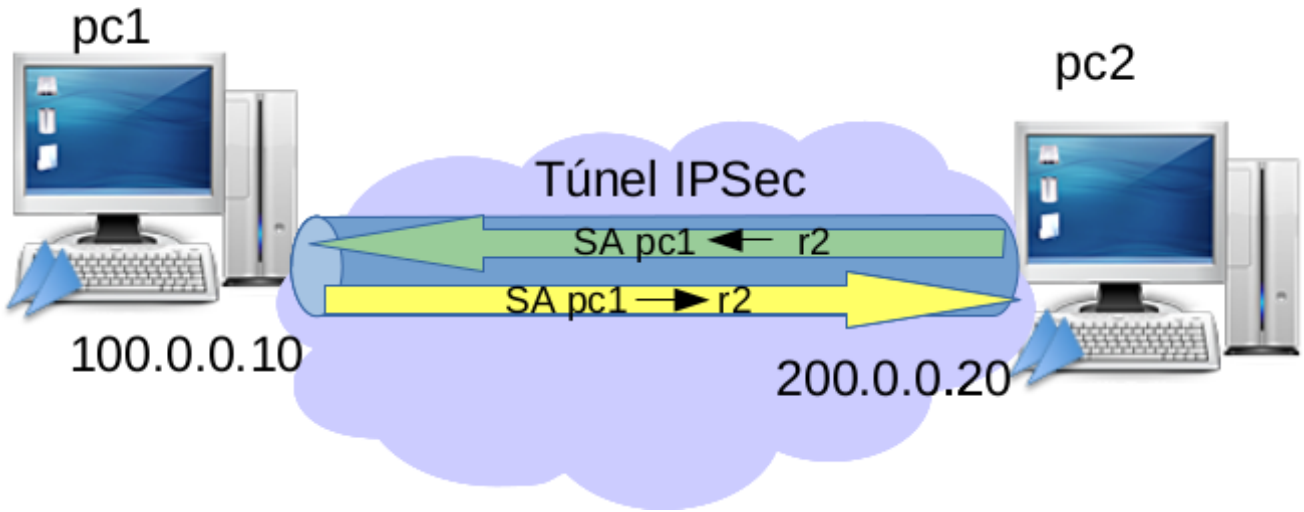
En `ipsec.conf` de `pc2`:

```
conn host-host
        left=200.0.0.20
        leftcert=pc2Cert.pem
        leftid=@pc2
        right=100.0.0.10
        rightid=@pc1
        rightcert=pc1Cert.pem
        type=transport
        auto=add
```

- En `pc1` debe estar almacenado su certificado de clave pública `pc1Cert.pem` en la carpeta `/etc/ipsec.d/certs` y de forma análoga en `pc2`, su certificado de clave pública `pc2Cert.pem`debe estar en la carpeta `/etc/ipsec.d/certs`.

Restringir el protocolo y puerto

- En la configuración anterior se puede restringir el protocolo y puerto usado para IPsec, por ejemplo para que únicamente se use en conexiones TCP desde el puerto 1111 de pc1 y hacia el puerto 2222 de pc2:



En `ipsec.conf` de pc1:

```
conn host-host
        leftprotoport=tcp/1111
        left=100.0.0.10
        leftcert=pc1Cert.pem
        leftid=@pc1
        rightprotoport=tcp/2222
        right=200.0.0.20
        rightid=@pc2
        rightcert=pc2Cert.pem
        type=transport
        auto=add
```

En `ipsec.conf` de pc2:

```
conn host-host
        leftprotoport=tcp/2222
        left=200.0.0.20
        leftcert=pc2Cert.pem
        leftid=@pc2
        rightprotoport=tcp/1111
        right=100.0.0.10
        rightid=@pc1
        rightcert=pc1Cert.pem
        type=transport
        auto=add
```

Contenidos

Arranque y estado de la comunicación

- Se inicia `ipsec` en los dos extremos:

  ```
  ipsec start
  ```

- En el caso de túnel entre dos máquinas, lo puede iniciar cualquiera de los dos extremos. Para ello se ejecuta en un pc el comando para la activación de la conexión con el identificador que se le había asignado en el fichero `ipsec.conf` (en nuestro ejemplo `host-host`):

  ```
  ipsec up host-host
  ```

- A partir de ese momento comienza la negociación de los algoritmos de cifrado y autenticación entre los dos extremos usando el protocolo IKEv2 y si se configurarán los SA y SP.
- Al igual que en ejemplo anterior, se utilizarán los mismos comandos para comprobar el establecimiento de los parámetros de IPsec.

Contenidos

Creación de claves y certificados

- Utilizaremos las herramientas `ipsec pki` para crear las claves RSA y los certificados necesarios para proporcionar la autenticación con IKEv2.
- Las claves y certificados se almacenarán en las siguientes carpetas:
  - `/etc/ipsec.d/private`: claves locales privadas.
  - `/etc/ipsec.d/certs`: certificados de clave pública locales.
  - `/etc/ipsec.d/cacerts`: certificados de autoridades de certificación.

Certificado de una autoridad de certificación (CA)

- Creación de una pareja de claves RSA de 4098 bits y el certificado autofirmado de la autoridad de certificación.

```
$ cd /etc/ipsec.d/
$ ipsec pki --gen --type rsa --size 4096  --outform pem   > private/myCAKey.pem
$ chmod 600 private/myCAKey.pem

$ ipsec pki --self --ca --lifetime 3650 \
    --in private/myCAKey.pem --type rsa \
    --dn "C=ES, O=myCA, CN=My Root CA" \
    --outform pem \
    > cacerts/myCACert.pem
```

- Imprime de forma legible el contenido del certificado autofirmado `myCACert.pem`

```
$ ipsec pki --print --in /etc/ipsec.d/cacerts/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 May 20 21:43:26 2016, ok
           not after  May 18 21:43:26 2026, ok (expires in 3644 days)
serial:    22:27:e4:ce:4d:12:65:0c
flags:     CA CRLSign self-signed
authkeyId: 18:24:45:8b:f5:0a:da:05:b7:83:39:b0:76:1f:96:41:ec:2a:50:d3
subjkeyId: 18:24:45:8b:f5:0a:da:05:b7:83:39:b0:76:1f:96:41:ec:2a:50:d3
pubkey:    RSA 4096 bits
keyid:     ee:df:fd:15:86:7a:35:c9:8f:83:49:a1:15:3a:ff:70:85:54:d6:9a
subjkey:   18:24:45:8b:f5:0a:da:05:b7:83:39:b0:76:1f:96:41:ec:2a:50:d3
```

Certificado de r1 firmado por CA (I)

- Creación de una pareja de claves RSA de 2048 bits y el certificado de r1 firmado por la autoridad de certificación.
  - Crea la pareja de claves y almacena la clave privada y pública en el fichero `r1Key.pem`:

```
$ cd /etc/ipsec.d/
$ ipsec pki --gen --type rsa --size 2048 --outform pem > private/r1Key.pem
$ chmod 600 private/r1Key.pem
```

  - Extrae la clave pública y genera un certificado firmardo por la CA almacenándolo en `r1Cert.pem`. En el parámetro CN debe especificarse el nombre completo de la máquina FQDN, ya que el certificado queda asociado a la identidad de la máquina.

```
$ ipsec pki --pub --in private/r1Key.pem --type rsa | \
    ipsec pki --issue --lifetime 730 \
    --cacert cacerts/myCACert.pem \
    --cakey private/myCAKey.pem \
    --dn "C=ES, O=myCA, CN=r1" \
    --san r1 \
    --flag serverAuth --flag ikeIntermediate \
    --outform pem > certs/r1Cert.pem
```

Certificado de r1 firmado por CA (II)

- Imprime de forma legible el contenido del certificado firmado `r1Cert.pem`

```
$ ipsec pki --print --in /etc/ipsec.d/certs/r1Cert.pem
cert:      X509
subject:   "C=ES, O=myCA, CN=r1"
issuer:    "C=ES, O=myCA, CN=My Root CA"
validity:  not before May 21 10:18:26 2016, ok
           not after  May 21 10:18:26 2018, ok (expires in 725 days)
serial:    ec:f9:09:9b:36:72:b3:a5
altNames:  r1
flags:     serverAuth
authkeyId: 18:24:45:8b:f5:0a:da:05:b7:83:39:b0:76:1f:96:41:ec:2a:50:d3
subjkeyId: e5:72:0f:60:c9:68:0f:21:f9:de:83:eb:ad:5a:9d:ff:4a:98:99:84
pubkey:    RSA 2048 bits
keyid:     12:ba:41:a8:19:64:85:69:e2:3d:df:98:5c:86:2b:d0:72:3e:33:18
subjkey:   e5:72:0f:60:c9:68:0f:21:f9:de:83:eb:ad:5a:9d:ff:4a:98:99:84
```

Contenidos

Descifrar paquetes con Wireshark

- Se pueden descifrar/autenticar paquetes con Wireshark si se le proporcionan las claves para poder realizar esas operaciones.
- Dependiendo del tipo de paquete que se quiera descifrar, es necesario introducir información en Wireshark que le permita realizarlo.

Descifrar paquetes ISAKMP

- Claves en IKEv2: La información se obtiene del fichero de log correspondiente al demonio que se encarga de establecer la sesión IKE (`/var/log/charon.log`) y hay que introducirla en:
  Wireshark → Edit → Preferences → Protocols → ISAKMP
  - **Initiator's SPI (8 bytes)**= 14FC7C5DE1A5A77D
    ```
    generating rule 0 IKE_SPI
                  => 8 bytes @ 0x40288ac8
                  <1>    0: 14 FC 7C 5D E1 A5 A7 7D
    ```
  - **Responder's SPI (8 bytes)**= C350EC54BCC32D3C
    ```
    generating rule 1 IKE_SPI
                  => 8 bytes @ 0x40288ad0
                  <1>     0: C3 50 EC 54 BC C3 2D 3C
    ```
  - **SK_ei (16 bytes)**: Sk_ei secret = 35278270A460316DDE5BACBFB2B076D9
    ```
    Sk_ei secret => 16 bytes @ 0x40289bc0
        0: 35 27 82 70 A4 60 31 6D DE 5B AC BF B2 B0 76 D9
    ```
  - **SK_er (16 bytes)**: Sk_er secret = D61E8D5E16E6A926241AFE263201D5C2
    ```
    Sk_er secret => 16 bytes @ 0x40289bc0
                0: D6 1E 8D 5E 16 E6 A9 26 24 1A FE 26 32 01 D5 C2
    ```
  - **Encryption algorithm**: AES-CBC-128 [RFC3602]
  - **SK_ai (32 bytes)**: Sk_ai secret=DD357538E467C9775C4624D004BCBF8513ADF1C39C180E4348971742636C8C0D
    ```
    Sk_ai secret => 32 bytes @ 0x4028bb18
       0: DD 35 75 38 E4 67 C9 77 5C 46 24 D0 04 BC BF 85
      16: 13 AD F1 C3 9C 18 0E 43 48 97 17 42 63 6C 8C 0D
    ```
  - **SK_ar (32 bytes)**: Sk_ar secret=CDBF395A5DDE8C926F9CC9FCC612490A9DCD97DEB352DD7D2D2C36E981AC27CD
    ```
    Sk_ar secret => 32 bytes @ 0x4028bb18
       0: CD BF 39 5A 5D DE 8C 92 6F 9C C9 FC C6 12 49 0A
      16: 9D CD 97 DE B3 52 DD 7D 2D 2C 36 E9 81 AC 27 CD
    ```
  - **Integrity algorithm**: AES-CBC-128 [RFC3602]

Descifrar paquetes IPsec: SA1

- Claves ESP: La información se obtiene del SA establecido (consultando SAD) y hay que introducirla en: Wireshark → Edit → Preferences → Protocols → ESP

```
r1:~# ip xfrm state
src 100.0.0.1 dst 200.0.0.2
        proto esp spi 0xcfb9dfca reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256) 0x972feeb4fa34659af67b1cb1a2f0988331b61c5f2f16961782ad6202db5cd62e 128
        enc cbc(aes) 0xf024505f8f456a6a0f78a6cb8c5386b4

src 200.0.0.2 dst 100.0.0.1
        proto esp spi 0xcb3dc960 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256) 0x91aa030b659b33a4d829b4aee2945f266a050a750323171e31fc3d063544de52 128
        enc cbc(aes) 0x782ef28cdbbfd53e5272a1316af9eb6f
```

- Parámetros de SA desde 100.0.0.1 a 200.0.0.2:
  - Protocol: IPv4
  - Src IP: 100.0.0.1
  - Dest IP: 200.0.0.2
  - SPI: 0xcfb9dfca
  - Encryption algorithm: AES-CBC [RFC3602]
  - Encryption key (16 bytes): 0xf024505f8f456a6a0f78a6cb8c5386b4
  - Authentication algorithm: HMAC-SHA-256-128 [RFC4868]
  - Authentication key (32 bytes): 0x972feeb4fa34659af67b1cb1a2f0988331b61c5f2f16961782ad6202db5cd62e

Descifrar paquetes IPsec: SA2

- Claves ESP: La información se obtiene del SA establecido (consultando SAD) y hay que introducirla en: Wireshark → Edit → Preferences → Protocols → ESP

```
r1:~# ip xfrm state
src 100.0.0.1 dst 200.0.0.2
        proto esp spi 0xcfb9dfca reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256) 0x972feeb4fa34659af67b1cb1a2f0988331b61c5f2f16961782ad6202db5cd62e 128
        enc cbc(aes) 0xf024505f8f456a6a0f78a6cb8c5386b4

src 200.0.0.2 dst 100.0.0.1
        proto esp spi 0xcb3dc960 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256) 0x91aa030b659b33a4d829b4aee2945f266a050a750323171e31fc3d063544de52 128
        enc cbc(aes) 0x782ef28cdbbfd53e5272a1316af9eb6f
```

- Parámetros de SA desde 200.0.0.2 a 100.0.0.1:
  - Protocol: IPv4
  - Src IP: 200.0.0.2
  - Dest IP: 100.0.0.1
  - SPI: 0xcb3dc960
  - Encryption algorithm: AES-CBC [RFC3602]
  - Encryption key (16 bytes): 0x782ef28cdbbfd53e5272a1316af9eb6f
  - Authentication algorithm: HMAC-SHA-256-128 [RFC4868]
  - Authentication key (32 bytes): 0x91aa030b659b33a4d829b4aee2945f266a050a750323171e31fc3d063544de52