ÍNDICE GENERAL

CAPÍTULO 4

SEGURIDAD EN REDES

4.1 Introducción

La **seguridad** implica garantizar los siguientes aspectos en su **totalidad**:

1. **Confidencialidad/privacidad**: Solo lo pueden **leer** los que estén **autorizados**.
2. **Autenticación**: Soy quien digo ser (**no** hay **suplantación de identidad**).
3. **Control de accesos**: Solo doy **servicio** a los que estén **autorizados**.
4. **No repudio** o irrenunciabilidad: No puedo mentir y decir que **no fui el autor** de algún mensaje o obra.
5. **Integridad**: Detecta **cambios** (intencionales o no).
6. **Disponibilidad**: **Mantiene servicios** independiente de demanda.

Tenemos que dar seguridad en **TODAS las capas**.

Definición 4.1.1 Ataque de seguridad
Acción que afecta a cualquier aspecto de la seguridad.

Algunos tipos de ataque

1. **Sniffing** = vulneración a la **confidencialidad**, escuchas lo que dicen los mensajes (husmear).
2. **Spoofing** (phishing) = suplantación de la identidad de entidades
3. **Man in the middle** = Interceptas conexiones y mensajes.
4. Distributed Denial of Service (**DDoS**) = denegación de servicio (lo saturas) distribuido.
5. **Malware** = troyanos, gusanos, spyware, backdoors, rootkits, ransomware, keyloggers

4.1.1. Mecanismos de seguridad (tipos)

Prevención

- **Autenticación** e **identificación**.
- **Control de acceso**.
- **Separación** (física, temporal, lógica, criptográfica y fragmentación).
- Seguridad en las comunicaciones (**cifrado** e integridad de la información).

Detección

- **IDS** (Intruder Detection System)

Recuperación

- copias de seguridad (**backup**).
- mecanismos de **análisis forense**: averiguar el alcance, las actividades del intruso en el sistema y cómo entró.

4.2 Cifrado (simétrico y asimétrico)

Definición 4.2.1 Cifrar

Transformar la **información** en otro espacio de representación por un **algoritmo** conocido y que, a pesar de conocer el algoritmo, si no se saben la clave de cifrado y descifrado **no es reversible**. Garantiza la **confidencialidad**.

4.2.1. Cifrado simétrico/con clave secreta

Existe **una sola clave para cifrar y descifrar**. Por ejemplo, **DES**.

DES

Es un algoritmo de cifrado de los años 70 desarrollado por IBM. Usa claves de 56 bits, así que a priori la complejidad es de 2^{56}. Se llama **cifrado de bloques** porque salen y entran bloques de bits. Son esquemas de **sustitución monoalfabética**, es decir, coge símbolos de un alfabeto de entrada y te da el texto cifrado sustituyendo uno a uno los símbolos. Los esquemas de este tipo tienen la debilidad que, si sabemos el idoma original, sabes la jerarquía de uso de las letras (probabilidad de usar más vocales que consonantes y así). Por lo tanto, si un símbolo se repite mucho, puedes intuir cuál es esa letra. Para evitar esto, se utiliza **Encadenamiento DES**. Esto cifra dependiendo de la cadena de entrada y la historia previa de cifrado. Existe también **DES doble**, que usa dos cirados de 56 bits, así que es más robusto.

IDEA

Utiliza **claves de 128 bits** y tiene **8 iteraciones**. Al final tiene una transformación. Utiliza subconjuntos de la clave.

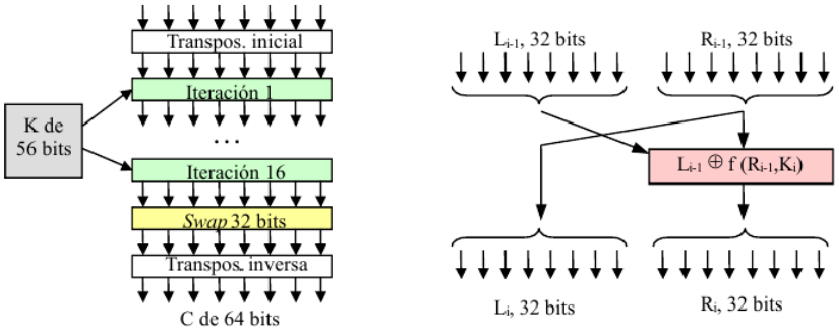

Figura 4.1: Pasos de DES en su estado original

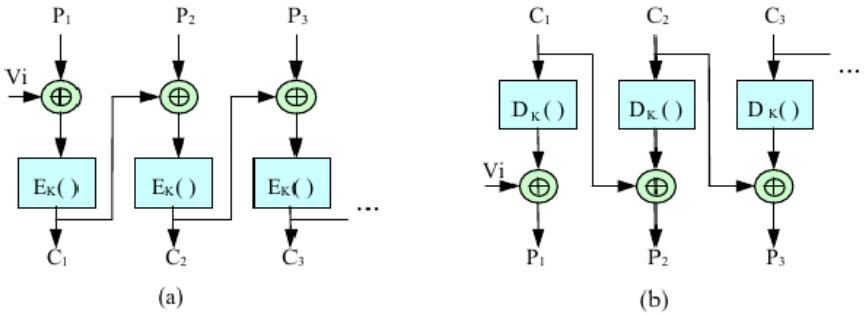

Figura 4.2: Encadenamiento DES con partes de la clave antes o después de salir el bloque.

AES

Cifrado simétrico de claves de 128, 192 o 256 bits. No tiene debilidad hoy por hoy.

4.2.2. Cifrado asimétrico

Cada usuario tiene una **clave pública y privada** y están vinculadas. Conocida la pública, es **computacionalmente imposible** obtener la privada. Ejemplo: A obtiene la clave pública de B. Solo B tiene su privada y es el único que puede transformar el texto cifrado al original. Ver imagen **4.3**.

Aspectos que garantiza

1. La **privacidad**. Si yo cifro con la clave pública que me has dado, cualquiera que intercepte el mensaje es imposible que obtenga la privada.
2. La **autenticidad** si me puedo asegurar que la clave pública de A es A y es inmóvil.

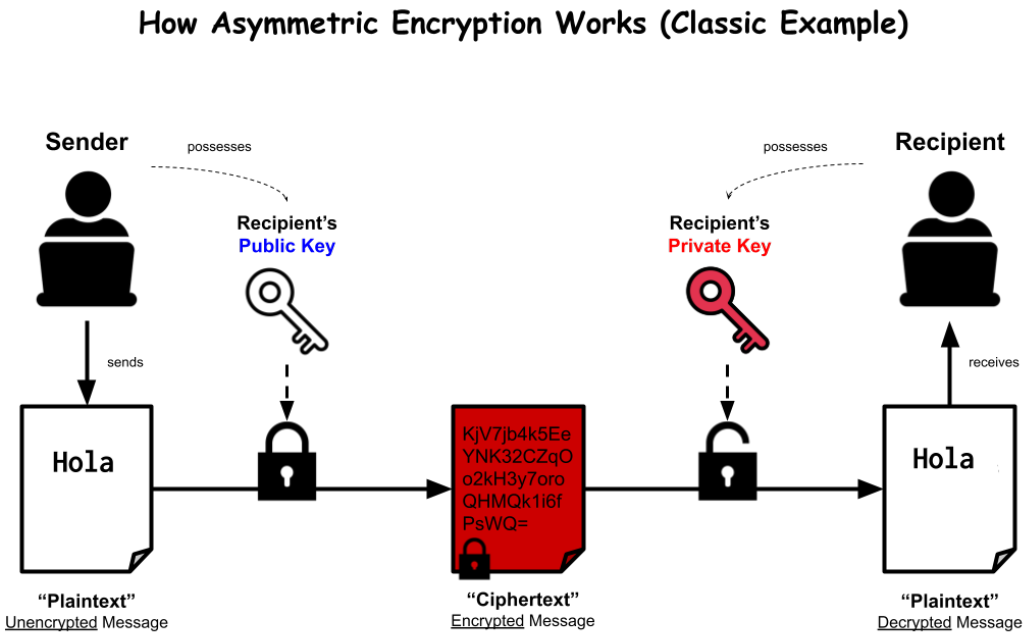

Figura 4.3: Cifrado asimétrico.

RSA

Es un algoritmo que garantiza que teniendo la pública no sacas la privada.

Pasos RSA

1. Elegimos **p** y **q** primos muy grandes.
2. Obtenemos N $(p \cdot q)$ y la función de euler de los primos Z = $(p-1) \cdot (q-1)$
3. Saco un primo de Z llamado d.
4. Calculamos e tal que el resto de dividir $(e \cdot d) \,/ z = 1$
5. Conocido (e,n) (**clave pública**) es imposible conocer (d,n) (**clave privada**)

4.3 Autenticación con clave secreta (reto-respuesta)

Un login y password no tiene garantía de autenticidad (solo sirve para identificarte). Existe un procedimiento llamado **reto-respuesta**. Se basa en la hipótesis de que **existe una clave secreta entre A y B solo conocida por ellos**. Se manda un reto (un número aleatorio) y se exige al receptor que **devuelva el reto** cifrado con la **clave que comparten** los dos.

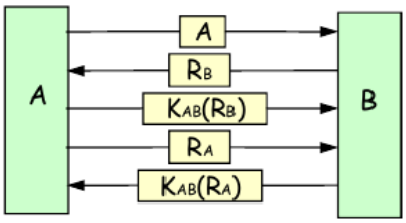

Figura 4.4: Procedimiento reto-respuesta. A le dice a B que es él. B le manda un reto. A le devuelve el reto cifrado con la clave que comparten y el reto de A. B le devuelve el reto cifrado.

Nota ¿Es posible simplificar el procedimiento a 3 mensajes?

Si redujera a solo 3 mensajes hay un ataque de reflexión. C le dice "oye, soy A, toma mi reto" B le dice "¡¡hola a!! ¡¡toma mi reto!!" C abre otra sesión y le manda el primer reto de B a B para que le cifre todo con la clave y se lo manda de vuelta con su primera sesión. Así B cifra su propio reto. Para evitar esta falla de seguridad, se podrían marcas de tiempo y pones milisegundos razonables para recibir una respuesta. También ***nonces****, que son retos que solo se pueden usar una vez.*

4.4 Intercambio de Diffie-Hellman (establecimiento de clave secreta)

Definición 4.4.1 Diffie-Hellman

Método de establecer una **clave secreta** entre dos entidades de un canal no seguro.

Yo elijo x, n y g. Te mando $n, g, g^x mod(n)$ y tú me devuelves $g^y mod(n)$, siendo y un número que tú eliges. En ambos extremos $g^{xy} mod(n)$ es nuestra clave secreta. Con este método, un tercero no podría sacar la clave porque habría que hacer un **logaritmo modular de primos muy grandes**.

El problema, es que es débil ante ataques de **man in the middle**. Una persona podría interceptar las comunicaciones entre los dos, creando así **una clave con cada uno**.

Nota Dilema de dependencia

Para poder autenticarnos, usamos ***clave secreta****. Pero para eso necesitamos usar* ***diffie-hellman****. Pero para usar Diffie-Hellman tengo que* ***autenticarme****.*

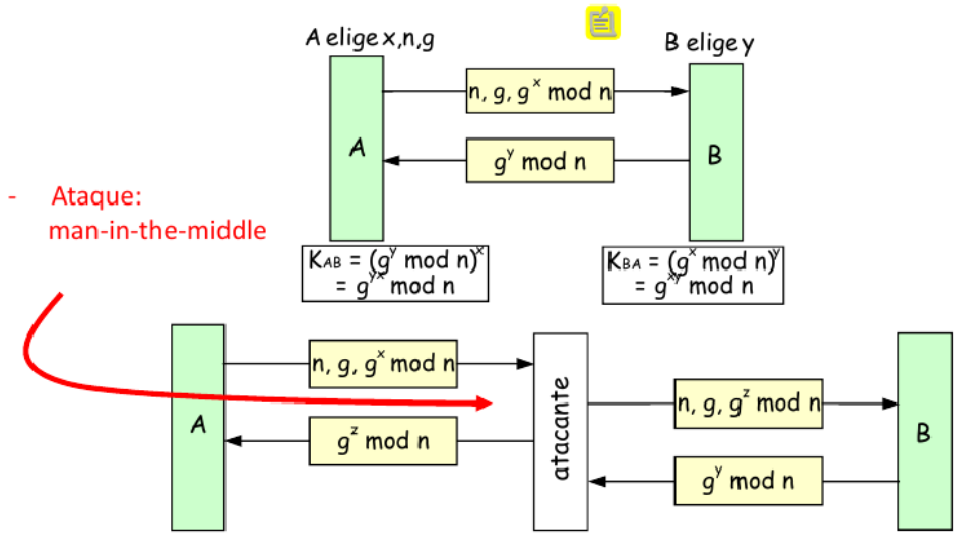

Figura 4.5: Arriba tenemos el funcionamiento normal de diffie-hellman y abajo un ataque man-in-the-middle explotando su vulnerabilidad.

4.5 Funciones Hash. Hash Message Authentication Code (HMAC)

Definición 4.5.1 Funciones hash

Garantizan la **integridad**, puesto que podemos detectar **alteraciones** del mensaje original. Metemos un texto de entrada por un cálculo **unidireccional e irreverrsible**. Esto nos hace **inmunes a ataques de colisión**, porque es imposible sacar el texto original del hash.

Los hash por sí solos no sirven de nada porque nada impide **modificar el mensaje y rehacer el hash**. Para eso se utilizan los **HMAC**. Utilizamos **una clave secreta junto al mensaje original** y evitamos así suplantación de identidad.

4.5.1. MD5

Es un hash de **128 bits**. Tiene una entrada sin límite de longitud.

Pasos de MD5

1. Le añado al texto original **relleno** para que sea múltiplo de 512.
2. Elijo **los primeros 512 bits**.
3. El **resultado** de ese hash **entra al siguiente bloque** (los **128 bits** previos a la siguiente iteración)

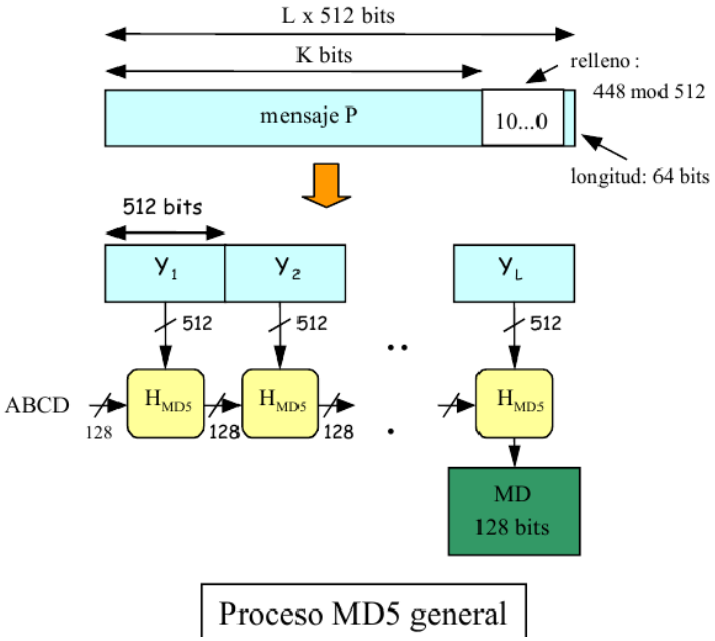

Figura 4.6: Proceso de creación de hash MD5.

4.5.2. SHA-1

Esencialmente igual que MD5, pero tiene **160 bits** en vez de 128. También han salido versiones más nuevas con mayor número de bits. Como es mas largo, **tarda más y es menos vulnerable** a ataques que MD5.

4.6 Firma Digital

Definición 4.6.1 Firma digital
Esencialmente un paralelo digital a una firma real. Su objetivo es que se pueda **autenticar** al emisor, que **no haya repudio** (no renunciar autoría) y que no haya **falsificación por parte del receptor**.

4.6.1. Big brother

Hay una **autoridad de confianza** que comparte la clave de A con A y la clave de B con B y que tiene una **clave secreta** que sólo él conoce. Esta autoridad, en el caso de España sería la fábrica de moneda y timbre o el certificado de dni de la policía.

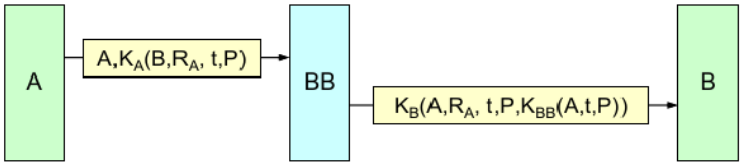

Figura 4.7: Comunicación Big Brother. A le indica a BB (Big brother) que es A, y cifrado con su clave K_A, le da la identidad de B, un reto (R_A), una marca de tiempo (t) y el texto (P). En el mensaje hacia B, lo que está firmado con K_{BB} es la firma digital.

> **Nota Ejemplos de Big brother**
>
> *B, se fía del Big brother, tiene **autenticado** el origen porque lo que recibe solo puede hacerlo el Big brother y tiene que ser entregado en el entorno temporal de la marca de tiempo. **No repudio**. Imagina que A dice que repudia de la firma. B va a juicio y tiene el mensaje que le llegó. Le dice al juez "mira, tengo este numeraco que es el cifrado de A, texto y tiempo". ¿Y quien cifró eso? Pues la policía o la entidad big brother que sea. El juez le reclama al big brother que descifre el KBB. La policía descifra y se ve que A en cierto tiempo le firmó P. Por último, la **integridad**. Imagina que A le dice a B que está haciendo fraude porque ha modificado el contrato. B no puede generar una firma válida sobre un contrato que tenga su modificación (porque dependería del KBB) para que eso que dice A ocurriera. Así que no puede desmostrar que su modificación es la original y se mantiene la integridad.*

4.6.2. Doble cifrado

Para autenticar usando claves públicas y privadas tenemos que tener **100 % seguro que autentica al firmante**. Es decir, tengo que estar seguro de que A se asocia a su clave pública. Esto se consigue con un **certificado digital**. En cuanto al **no repudio**, está firmado por la clave pública de A que autentica a A, así que no se puede rechazar. Y sobre la **fiabilidad**, si se modifica no puedes cifrarlo porque te falta la clave privada de A. Otra debilidad es **falsa denuncia de robo de clave**, en el que insinúas que te robaron la clave y por eso firmaron por ti.

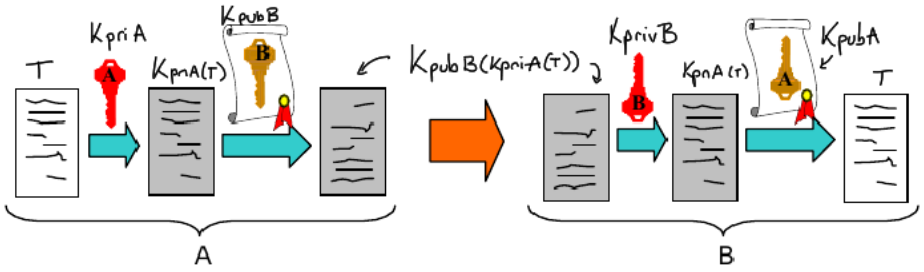

Figura 4.8: Primero firmo con mi clave privada para autenticarme y luego con la pública de B para asegurarme de que solo él lee el mensaje. B descifra su clave pública con su clave privada y luego obtiene el mensaje con la clave pública mía. Lo cual asegura que yo escribí el mensaje y que él no puede reescribirlo.

4.7 Certificados digitales

Definición 4.7.1 Certificado digital
Garantiza la asociación identidad<->clave pública.

Similar al Big Brother, necesitamos **una autoridad de certificación (AC)** que puede ser la policía o similares. En el DNI electrónico por ejemplo, está asociado al chip de tu DNI.

Proceso de emsión de certificado digital

1. El usuario **obtiene claves pública y privada**.
2. Envía una **solicitud firmada digitalmente a la AC** indicando su identidad y su clave pública.
3. **AC comprueba y emite el certificado cifrado** digitalmente con la clave de AC para no falsificarlo.

4.7.1. X.509

Los campos de un certificado X.509 están especificados en la tabla 4.1.

4.8 Protocolos seguros: implementación de mecanismos de seguridad

La seguridad en redes se puede dividir en dos partes:

- **Seguridad perimetral**, que consiste en Firewalls, sistemas de detección de intrusiones y de respuesta y similares.
- **Seguridad en protocolos** que conforman Internet.

Cuadro 4.1: Campos de un certificado X.509

Campo	Explicación
Versión	Número de versión de X.509
Número de serie	Identificador del AC
Firma	Firma del certificado
Entidad emisora	Nombre del AC definido por X.509
Periodo de validez	Fecha de empiezo y final de la validez del certificado
Nombre del individuo	La entidad cuya clave pública está siendo certificada
Clave pública	La clave pública de la entidad certificada y los algoritmos que la usan

La seguridad **hay que ponerla en todos los niveles de protocolos**. Por ejemplo, si IP es vulnerable da igual cuánto asegure a la capa de aplicación.

4.8.1. Capa de aplicación

PGP

PGP ofrece seguridad en correo. Para utilizarlo ambos deben tener la clave pública del otro.

Pasos de PGP

1. **MD5**: Hace que al texto plano le pases un hash MD5.
2. **Clave privada de A**: Cifra el hash con la clave privada de A.
3. **Compresión**: Crea un archivo comprimido de texto plano + hash cifrado.
4. **Cifrado con IDEA y Cifrado con KpubB**: Crea un mensaje en el que por un lado cifra el comprimido con el algoritmo IDEA con una clave secreta Ks. Luego cifra esa clave secreta con la clave pública de B.
5. **Base 64**: Finalmente, este mensaje se manda pasado a BASE64 para convertir todo a texto ASCII.

Pasos del receptor de PGP

1. Deshago Base64.
2. Saco la clave secreta con mi clave privada.
3. Saco el mensaje comprimido con la clave secreta.
4. Descomprimo el mensaje
5. Veo el Hash con la clave pública de A
6. Deshasheo el mensaje original
7. Compruebo que el mensaje en texto plano deshasheado y el mensaje en texto plano del comprimido son el mismo.

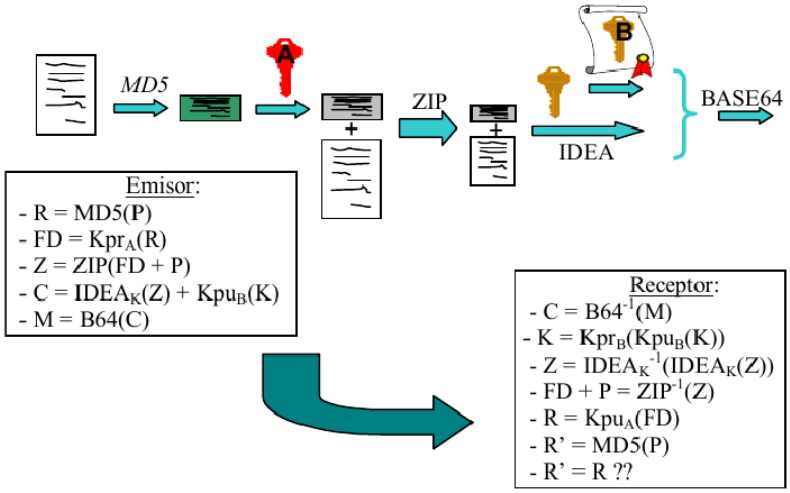

Figura 4.9: Proceso PGP de forma gráfica.

Otros ejemplos

- SSH (Secure shell) permite la conexión remota mediante el uso de cifrado. También se pueden utilizar certificados digitales.
- DNSSEC: DNS con firma de las respuestas.

4.8.2. Capa de transporte

TLS

> **Definición 4.8.1 TLS**
> TLS (Antes ssl) es un protocolo (conjuntos de protocolos) que da **seguridad a nivel de tcp**. Cuando cualquier aplicación que use tcp le ponemos TLS aparece la S. Del http: HTTPS.

Está formado por 4 protocolos, pero vamos a centrarnos en Record y Handshake.

Partes de TLS

- **SSL Handshake Protocol**:
  1. Negocia el algoritmo de cifrado
  2. Negocia la función Hash
  3. Autentica al servidor con X.509 (certificado digital)
  4. El cliente genera claves de sesión (Aleatorias o diffie hellman)
- **SSL Record Protocol**: Encapsula los protocolos y ofrece un canal seguro con **privacidad, autenticacion e integridad.**
  1. Parte en cachitos el mensaje de datos
  2. Lo comprime
  3. Obtiene el hash y a la concatenación de mensaje con hash lo cifra.
  4. Asegura **autenticación** porque está autenticado en el servidor, **integridad** porque usa un hash y **privacidad** porque es imposible descifrar los mensajes sin la clave de sesión y los certificados.
- **SSL Assert protocol**: Informa de errores en la sesión
- **Change Cipher Espec Protocol**: Notifica cambios en el cifrado.

4.8.3. Capa de Red

IPSec

TCP se incrusta en IP así que tenemos que hacer algo para evitar vulnerabilidades. Queremos garantizar que en IP **son quienes dicen ser**, **detectar cambios den datagramas** y **cifrar**. IPSec son 3 procedimientos, uno de ellos opcional:

1. **Establecer asociación de seguridad** con un protoclo llamado IKE (Interket Key exchange) Establecer una Ks (**clave secreta**) con **Diffie-Hellman. Necesito certificados**.
2. **Protocolo de cabeceras de autenticación** (Garantiza dos cosas: **autenticación** e **integridad**). Se hace con HMAC.
3. (Opcional) **Encapsulado de seguridad de la carga**.

Modos de operación

- **Modo túnel**: La asociación se hace entre routers. Solo seguro entre routers.
- **Modo Transporte**: La asociación se hace extremo a extremo. Seguro en todo el proceso.

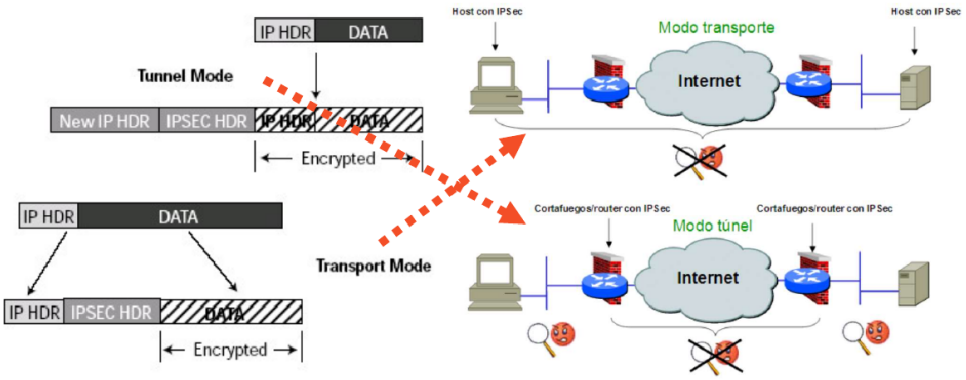

Figura 4.10: Modo túnel y modo transporte de IPSec