

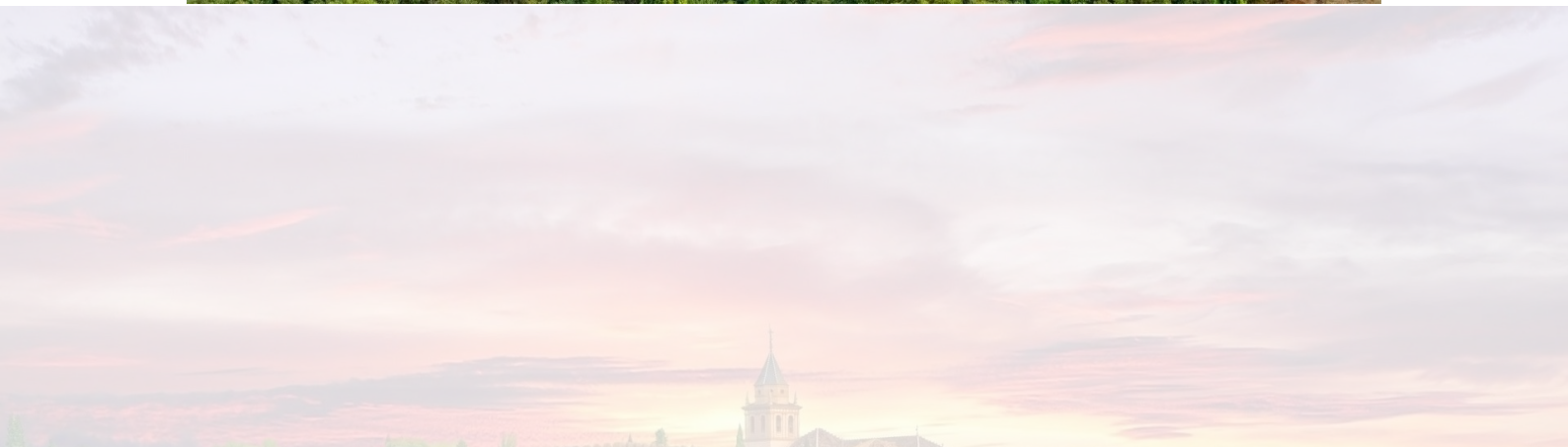


# Ingeniería Informática + ADE

Universidad de Granada (UGR)

**Autor:** Ismael Sallami Moreno

**Asignatura:** Tema 4: Seguridad en Redes (FR)



# Índice

<b>1. Introducción</b>	<b>4</b>
1.1. ¿Qué es la seguridad?	4
1.2. ¿En qué nivel o capa se debe situar la seguridad?	4
1.3. Mecanismos de Seguridad	4
1.4. Ataques de Seguridad	5
<b>2. Cifrado</b>	<b>5</b>
2.1. Cifrado simétrico	6
2.2. Cifrado asimétrico	8
<b>3. Autenticación</b>	<b>10</b>
3.1. Reto-respuesta	10
3.2. Intercambio de Diffie-Hellman	11
3.2.1. Ataque Man-in-the-Middle (MitM)	12
<b>4. Funciones Hash</b>	<b>13</b>
4.1. Funciones Hash (Compendios)	13
4.1.1. Características de los Compendios	13
4.1.2. Ejemplos de Funciones Hash	14
4.1.3. Uso de las Funciones Hash	14
4.1.4. MD5 (Message Digest 5, RFC 1321)	14
4.1.5. SHA-1 (Secure Hash Algorithm 1, NIST 1993)	14
<b>5. Firma Digital y certificados digitales</b>	<b>15</b>
5.1. Firma Digital: Objetivos	15
5.1.1. Firma Digital con Clave Secreta y Asimétrica	15
5.1.2. El concepto de <i>Big Brother</i> en la firma digital	16
5.1.3. Debilidad y Garantía de No Repudio	17
5.1.4. Certificados Digitales	17
5.1.5. Formato de los Certificados	18
5.1.6. Autoridades de Certificación (AC) Reconocidas	18
5.1.7. Campos de un certificado X.509	18
5.2. Relación entre los Mecanismos de Seguridad y los Servicios/Aspectos de Seguridad	19
5.2.1. Confidencialidad	19
5.2.2. Autenticación	19
5.2.3. No Repudio	19
5.2.4. Integridad	20
5.2.5. Disponibilidad	20
<b>6. Protocolos Seguros</b>	<b>20</b>
6.1. Seguridad	20
6.2. Pretty Good Privacy (PGP) – correo electrónico seguro	22
6.3. Transport Layer Security (TLS) / Secure Sockets Layer (SSL)	23

---

6.3.1.	SSL Record Protocol . . . . .	23
6.3.2.	SSL Handshake Protocol . . . . .	24
6.3.3.	Generación de Claves de Sesión . . . . .	24
6.3.4.	SSL Assert Protocol . . . . .	24
6.3.5.	Change Cipher Spec Protocol . . . . .	24
6.3.6.	Protocolos Utilizados con TLS/SSL . . . . .	24
6.4.	IPSec . . . . .	25
6.4.1.	Procedimientos de IPSec . . . . .	25
6.4.2.	Modos de Operación de IPSec . . . . .	26

# 1 Introducción

Una red de comunicaciones se considera **segura** cuando se garantiza la protección de todos los aspectos clave de la seguridad. Sin embargo, es importante destacar que **no existen protocolos ni redes 100 % seguras**.

## 1.1. ¿Qué es la seguridad?

La seguridad en redes engloba múltiples aspectos fundamentales:

- **Confidencialidad/Privacidad:** El contenido de la información debe ser comprensible únicamente para las entidades autorizadas.
- **Autenticación:** Garantiza que las entidades involucradas son quienes afirman ser.
- **Control de accesos:** Los servicios deben estar accesibles solo a entidades autorizadas.
- **No repudio o irrenunciabilidad:** Impide que una entidad niegue haber realizado una acción determinada.
- **Integridad:** El sistema debe detectar cualquier alteración de la información, ya sea intencionada o accidental.
- **Disponibilidad:** Los servicios deben mantenerse operativos, independientemente de la demanda.

## 1.2. ¿En qué nivel o capa se debe situar la seguridad?

La seguridad debe aplicarse en **todas las capas del sistema**. El grado de seguridad siempre estará limitado por el punto más débil de la red.

## 1.3. Mecanismos de Seguridad

Para garantizar los aspectos mencionados, se utilizan los siguientes mecanismos:

- **Cifrado:** Puede ser simétrico o asimétrico.
- **Autenticación con clave secreta:** Utilizando mecanismos de reto-respuesta.
- **Intercambio de Diffie-Hellman:** Permite establecer claves secretas de forma segura.
- **Funciones Hash:** Como el código de autenticación de mensajes (*Hash Message Authentication Code, HMAC*).
- **Firma Digital:** Para garantizar la autenticidad e integridad de los mensajes.
- **Certificados Digitales:** Utilizados para validar identidades.



## 1.4. Ataques de Seguridad

Un **ataque de seguridad** se define como cualquier acción, intencionada o no, que compromete alguno de los aspectos de la seguridad. Entre los ataques más comunes se encuentran:

- **Sniffing:** También conocido como "escuchas" o *husmeo*, vulnera la confidencialidad al interceptar información.
- **Spoofing (phishing):** Suplantación de identidad de entidades legítimas.
- **Man in the Middle:** Ataques de interceptación donde un tercero se posiciona entre las entidades comunicantes.
- **Denegación de Servicio Distribuida (DDoS):** Ejemplo de este ataque es el *flooding* o inundación de solicitudes para colapsar un sistema.
- **Malware:** Incluye amenazas como troyanos, gusanos, *spyware*, puertas traseras (*backdoors*), *rootkits*, ransomware y *keyloggers*.

## 2 Cifrado

**Cifrado de datos:**

- Es un procedimiento diseñado para garantizar la **confidencialidad** de la información.
- Consiste en transformar un **texto llano o claro** ( $P$ ) en un **texto cifrado** ( $C$ ).
- Este proceso se basa en la utilización de un algoritmo de **cifrado/descifrado**, comúnmente representado como  $E_K()$  para cifrado y  $D_{K'}()$  para descifrado.
- La complejidad del proceso radica en la utilización de una **clave de cifrado** ( $K$ ) y una **clave de descifrado** ( $K'$ ), las cuales deben permanecer desconocidas para garantizar la seguridad del sistema.

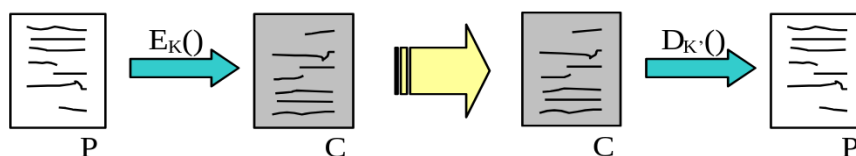


Figura 1: Esquema de cifrado de datos.

## 2.1. Cifrado simétrico

### Cifrado simétrico: Algoritmos de clave secreta

- Utiliza una **única clave** para realizar tanto el cifrado como el descifrado, es decir,  $K = K'$ .
- Un ejemplo representativo es el **DES** (*Data Encryption Standard*), desarrollado por IBM en 1975.

### Enlaces relacionados:

- Estructura Feistel<sup>1</sup>: Explicación de la arquitectura utilizada en muchos algoritmos de cifrado, incluido DES.
- Data Encryption Standard: Detalles técnicos y evolución histórica del algoritmo.

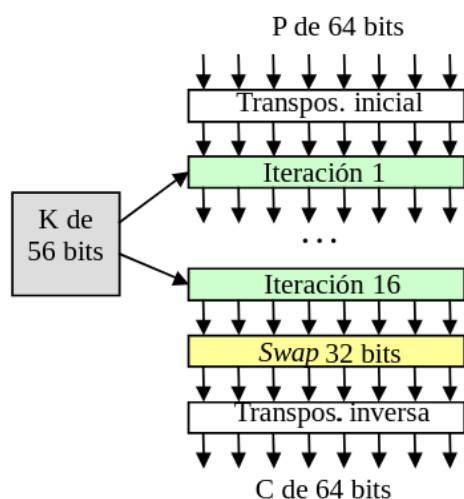


Figura 2: Diagrama A

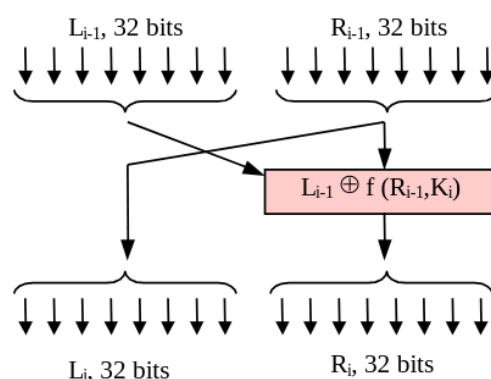


Figura 3: Diagrama B

- Diagrama A:
  - **P de 64 bits** (plaintext) pasa por una **Transpos. inicial** (permutación inicial).
  - El resultado se somete a 16 iteraciones (**Iteración 1** a **Iteración 16**) utilizando una **K de 56 bits** (clave de 56 bits).
  - Después de las iteraciones, ocurre un **Swap 32 bits**.
  - Finalmente, se aplica la **Transpos. inversa** (permutación inversa) para producir el **C de 64 bits** (ciphertext).
- Diagrama B:

<sup>1</sup>Puedes pinchar en los nombres para acceder a los enlaces

- El **entrada de 64 bits** se divide en dos mitades de 32 bits, **Li-1, 32 bits** y **Ri-1, 32 bits**.
  - **Li-1** se combina con el resultado de una función  $f$  aplicada a **Ri-1** y una subclave **Ki** usando la operación XOR ( $\oplus$ ).
  - La salida es dos mitades de 32 bits, **Li, 32 bits** y **Ri, 32 bits**.
- **DES:** Es un esquema de **sustitución monoalfabética**, lo que significa que cada símbolo del texto plano se sustituye por otro, según un esquema fijo basado en la clave.
- **Encadenamiento DES:** Introducido para evitar que DES actúe como un simple algoritmo de sustitución. Este mecanismo encadena los bloques cifrados, asegurando que cada bloque de texto cifrado dependa no solo del bloque actual de texto claro, sino también del bloque anterior de texto cifrado. Esto incrementa significativamente la seguridad del algoritmo.
- Para mejorar la robustez del cifrado, se utilizan variantes como **2DES** y **3DES**.
- **2DES (Double DES):** Consiste en aplicar el algoritmo DES dos veces consecutivas con dos claves diferentes. Aunque incrementa la seguridad en comparación con DES simple, sigue siendo vulnerable a ciertos tipos de ataques, como el ataque de encuentro en el medio (*meet-in-the-middle*).
  - **3DES (Triple DES):** Aplica el algoritmo DES tres veces, generalmente con dos o tres claves diferentes. Este método es mucho más seguro que DES y 2DES, y ha sido ampliamente utilizado en la industria para proteger datos sensibles.

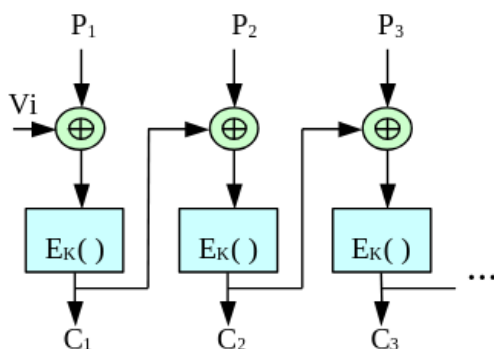


Figura 4: Diagrama A

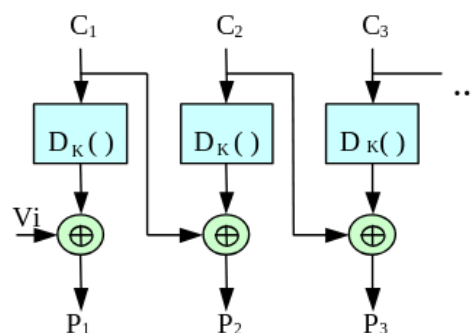


Figura 5: Diagrama B

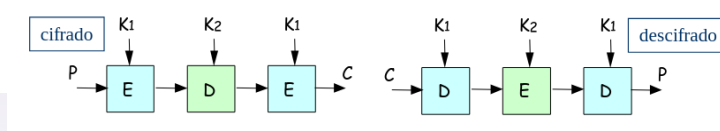


Figura 6: Diagrama 2 y 3DES

**IDEA (International Data Encryption Algorithm, IDEA)**

- Es un algoritmo de cifrado **simétrico**, lo que significa que utiliza la **misma clave** tanto para cifrar como para descifrar.
- Utiliza **claves de 128 bits**, proporcionando un alto nivel de seguridad.
- Diseñado para **operar en tiempo real**, con implementación eficiente en hardware (*Very Large Scale Integration, VLSI*).

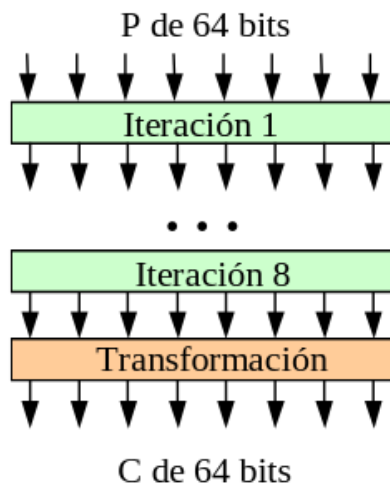


Figura 7: Diagrama A

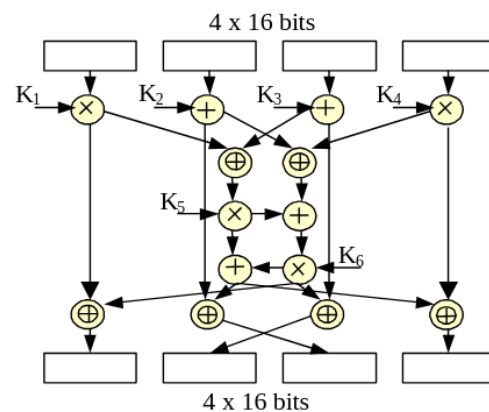


Figura 8: Diagrama B

**2.2. Cifrado asimétrico****Cifrado asimétrico: Algoritmos de clave pública/privada**

- Cada usuario ( $A$ ) posee dos claves distintas:
  - Una **clave pública** ( $K_{PUB_A}$ ).
  - Una **clave privada** ( $K_{PRI_A}$ ).
- Conociendo  $K_{PUB_A}$ , es **imposible deducir**  $K_{PRI_A}$ .
- Las claves son diferentes para las operaciones de cifrado y descifrado:
  - **Cifrado:**  $C = E_{K_{PUB_B}}(P)$ , donde  $P$  es el texto plano.
  - **Descifrado:**  $P = D_{K_{PRI_B}}(C)$ , donde  $C$  es el texto cifrado.
- En caso de enviar  $C = E_{K_{PRI_A}}(P)$ , el receptor puede verificar la **autenticidad** del mensaje al descifrarlo con  $K_{PUB_A}$ .



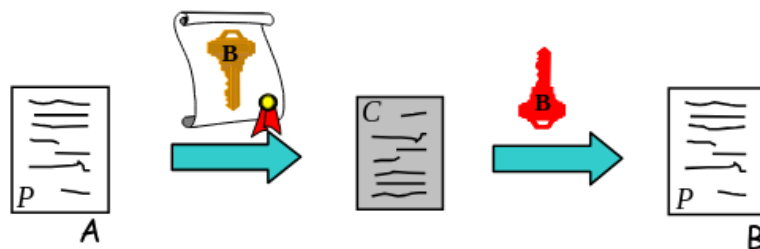


Figura 9: Esquema de cifrado asimétrico.

### RSA (Rivest, Shamir y Adleman)

- Se eligen dos números primos grandes  $p$  y  $q$ , donde  $p, q > 10^{100}$ .
- Calculamos:
  - $n = p \cdot q$ .
  - $z = (p - 1) \cdot (q - 1)$ , utilizando la función de Euler.
- Elegimos un número  $d$ , primo respecto de  $z$ .
- Calculamos  $e$  tal que  $e \cdot d \pmod{z} = 1$ , utilizando el algoritmo de Euclides.
- Las claves generadas son:
  - Clave pública:  $K_{\text{PUB}} = (e, n)$ .
  - Clave privada:  $K_{\text{PRI}} = (d, n)$ .
- Las operaciones de cifrado y descifrado son:
  - **Cifrado:**  $C = P^e \pmod{n}$ .
  - **Descifrado:**  $P = C^d \pmod{n}$ .

### Ejemplo RSA

- Se eligen los valores:
  - $p = 3, q = 11$ .
  - $n = p \cdot q = 33$ .
  - $z = (p - 1)(q - 1) = (3 - 1)(11 - 1) = 2 \cdot 10 = 20$ .
- Elegimos  $d = 7$ , un número primo respecto de  $z$ .
- Calculamos  $e$  tal que  $e \cdot d \pmod{z} = 1$ . En este caso,  $e = 3$ .
- Las claves generadas son:
  - Clave pública:  $K_{\text{PUB}} = (3, 33)$ .
  - Clave privada:  $K_{\text{PRI}} = (7, 33)$ .

**Cifrado y descifrado**

La siguiente tabla muestra el proceso de cifrado y descifrado para varios caracteres. Utilizamos la notación:

- **Cifrado:**  $C = P^e \text{ mód } n$ .
- **Descifrado:**  $P = C^d \text{ mód } n$ .





<u>Simbólico</u>	<u>Numérico</u>	<u>P<sup>3</sup></u>	<u>P<sup>3</sup> mod 33</u>		<u>C<sup>7</sup></u>	<u>C<sup>7</sup> mod 33</u>	<u>Simbólico</u>
S	19	6859	28		13492928512	19	S
U	21	9261	21		1801088541	21	U
Z	26	17576	20		1280000000	26	Z
A	01	1	1		1	01	A
N	14	2744	5		78125	14	N
N	14	2744	5		78125	14	N
E	05	125	26		8031810176	05	E
							
P		C			P		

Figura 10: Ejercicio Resuelto RSA.

**Resultado:** El texto original se recupera perfectamente después del cifrado y descifrado, demostrando la efectividad del esquema RSA.

### 3 Autenticación

#### 3.1. Reto-respuesta

- **Esquema de reto-respuesta:**

- Este método se utiliza para verificar la identidad de una entidad (por ejemplo, un cliente o servidor) sin necesidad de compartir directamente la clave secreta.
- El procedimiento típico es:
  1. La entidad que autentica (por ejemplo, el servidor) envía un **reto** ( $R$ ) a la otra entidad (por ejemplo, el cliente).
  2. El cliente responde cifrando el reto con la clave secreta compartida ( $K$ ), generando la respuesta cifrada  $C = E_K(R)$ .
  3. El servidor descifra la respuesta con la clave compartida y verifica si coincide con el reto original.

- **¿Ataque por reflexión?**

- Un ataque por reflexión ocurre cuando un atacante intercepta un reto enviado por una entidad y lo devuelve como respuesta, intentando hacerse pasar por una entidad legítima.
- Para mitigar este ataque, es importante implementar:

- **Espacios de claves disjuntos:** Utilizar claves diferentes para cada dirección de comunicación. Por ejemplo:
  - ◇  $K_{A \rightarrow B}$ : Clave utilizada para mensajes enviados de  $A$  a  $B$ .
  - ◇  $K_{B \rightarrow A}$ : Clave utilizada para mensajes enviados de  $B$  a  $A$ .
- Esto garantiza que un reto enviado por  $A$  no pueda ser simplemente reflejado de vuelta a  $A$  por  $B$ , ya que  $A$  esperará que el mensaje esté cifrado con  $K_{B \rightarrow A}$ .

**Resumen:** El esquema de reto-respuesta ofrece una manera eficiente de autenticar entidades sin revelar claves secretas. Sin embargo, es crucial implementar contramedidas como los **espacios de claves disjuntos** para evitar ataques por reflexión.

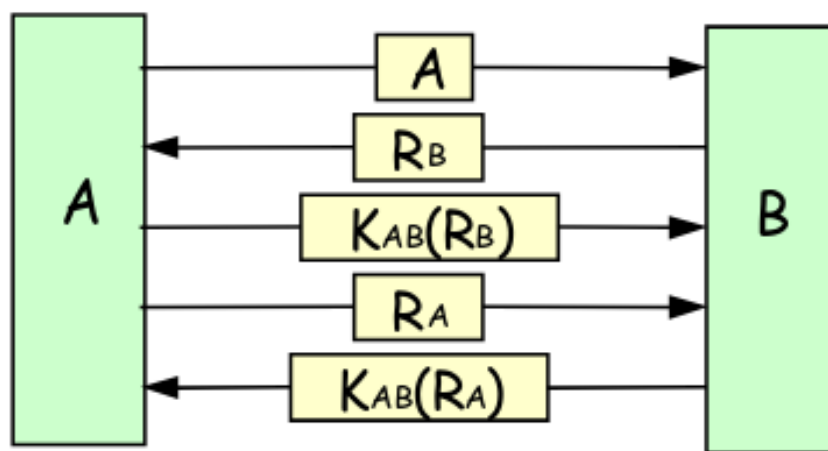


Figura 11: Esquema de reto-respuesta.

### 3.2. Intercambio de Diffie-Hellman

El *intercambio de claves de Diffie-Hellman* es un protocolo de criptografía que permite a dos entidades, A y B, establecer una clave secreta compartida, sin necesidad de intercambiar dicha clave explícitamente. En su lugar, utilizan números públicos y realizan cálculos sobre ellos. A continuación se describen los pasos básicos del protocolo:

1. Ambas partes acuerdan un número primo grande  $p$  y una base  $g$ , los cuales son públicos.
2. La parte A elige un número secreto  $a$  y calcula  $A = g^a \text{ mód } p$ , luego envía  $A$  a la parte B.
3. La parte B elige un número secreto  $b$  y calcula  $B = g^b \text{ mód } p$ , luego envía  $B$  a la parte A.
4. Ambas partes calculan la clave compartida:

- La parte A calcula  $K = B^a \text{ mód } p$ .
- La parte B calcula  $K = A^b \text{ mód } p$ .

Como  $A^b \text{ mód } p = B^a \text{ mód } p$ , ambas partes obtienen la misma clave secreta  $K$ , que pueden usar para cifrar la comunicación.

Este protocolo es seguro debido a la dificultad del problema de calcular los logaritmos discretos, es decir, dado  $g^a \text{ mód } p$  y  $g^b \text{ mód } p$ , es extremadamente difícil calcular  $a$  o  $b$ .

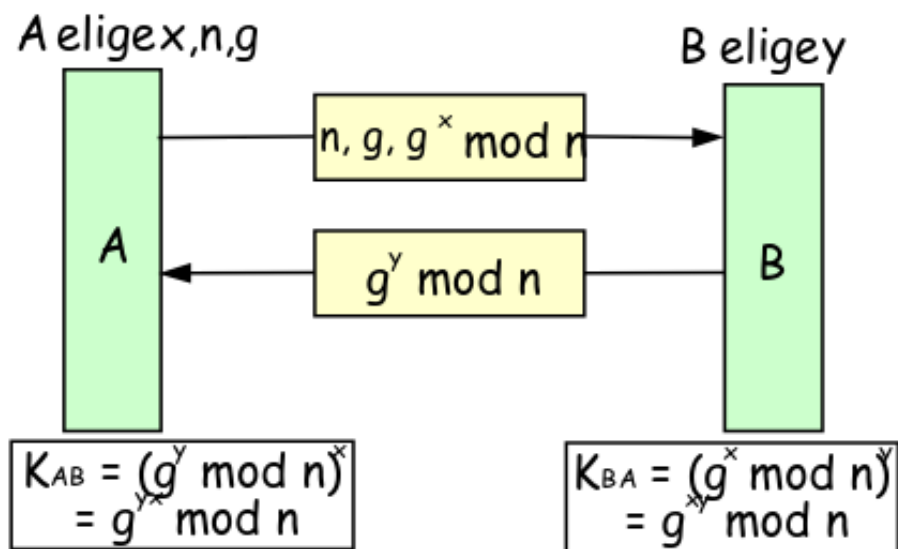


Figura 12: Intercambio de claves de Diffie-Hellman.

### 3.2.1. Ataque Man-in-the-Middle (MitM)

Aunque el intercambio de Diffie-Hellman es seguro, puede ser vulnerable a un *ataque Man-in-the-Middle* (MitM). En este tipo de ataque, un atacante (C) se coloca entre las dos partes A y B, interceptando y alterando los mensajes entre ellas.

1. La parte A envía su valor  $A = g^a \text{ mód } p$  al atacante (C) en lugar de enviarlo directamente a B.
2. El atacante (C) elige su propio valor secreto  $c$ , calcula  $C = g^c \text{ mód } p$  y envía C a la parte B.
3. La parte B, al recibir C, calcula la clave  $K_B = C^b \text{ mód } p$  y la envía de vuelta al atacante.
4. El atacante calcula la clave  $K_A = A^c \text{ mód } p$ , y ahora tiene dos claves secretas:  $K_A$  y  $K_B$ .
5. De esta manera, el atacante puede leer y modificar la comunicación entre A y B sin que ninguna de las partes se dé cuenta.



Este ataque funciona porque A y B creen que están intercambiando información directamente entre sí, pero en realidad están enviando la información al atacante, quien actúa como intermediario.

Para mitigar este tipo de ataque, se pueden usar firmas digitales o certificados, lo que permite a A y B asegurarse de que están comunicándose con la parte correcta.

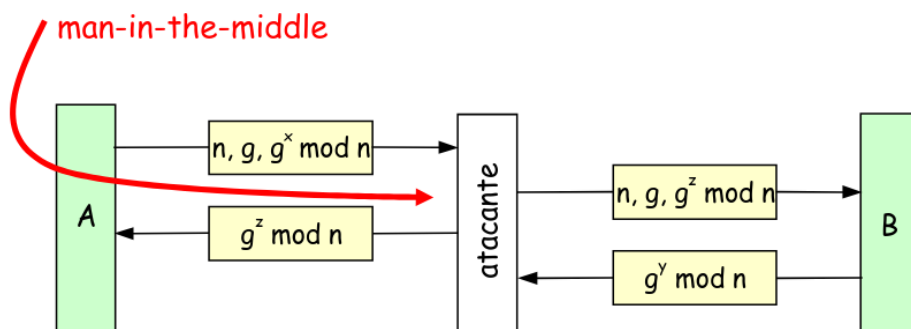


Figura 13: Ataque

## 4 Funciones Hash

### 4.1. Funciones Hash (Compendios)

Las funciones hash son algoritmos matemáticos que transforman un mensaje de longitud variable en una secuencia de longitud fija. Son ampliamente utilizadas en criptografía para garantizar la integridad de los datos y otros aspectos de seguridad.

#### 4.1.1. Características de los Compendios

Las funciones hash tienen las siguientes características:

- **Funciones unidireccionales (irreversibles):** El cálculo de un hash es sencillo, pero es prácticamente imposible obtener el mensaje original a partir de su resumen hash.
- **Texto de entrada (M) de longitud variable:** El tamaño del mensaje de entrada puede variar, pero siempre se transforma en un resumen de longitud fija.
- $M \rightarrow H(M)$ : El mensaje  $M$  se convierte en un hash  $H(M)$ , que tiene una longitud fija (por ejemplo, 256 o 512 bits).
- **Imposibilidad de obtener  $M$  a partir de  $H(M)$ :** Dado un hash  $H(M)$ , es prácticamente imposible encontrar el mensaje  $M$  original.
- **Invulnerabilidad a ataques de colisión:** Es imposible encontrar dos mensajes diferentes,  $M$  y  $M'$ , que produzcan el mismo hash  $H(M) = H(M')$ .

### 4.1.2. Ejemplos de Funciones Hash

Algunos ejemplos de funciones hash populares son:

- MD5
- SHA-1
- SHA-512

### 4.1.3. Uso de las Funciones Hash

Las funciones hash se utilizan principalmente para garantizar la integridad de los datos y la autenticación. Un ejemplo común es el *Hash Message Authentication Code* (HMAC), que combina un mensaje  $M$  con una clave  $K$ :

$$HMAC(K, M) = H(K \parallel M)$$

Para evitar ataques de extensión, se utiliza una versión más segura:

$$HMAC(K, M) = H(K \parallel H(K \parallel M))$$

### 4.1.4. MD5 (Message Digest 5, RFC 1321)

El proceso de generación de un resumen MD5<sup>2</sup> de 128 bits se realiza en los siguientes pasos:

- **Relleno:** El mensaje se rellena con 100..0 hasta alcanzar una longitud máxima de 448 bits.
- **Adición de campo de longitud:** Se añade un campo de 64 bits que contiene la longitud original del mensaje.
- **División en bloques:** El mensaje se divide en bloques de 512 bits.
- **Procesamiento secuencial:** Los bloques se procesan secuencialmente para generar el resumen de 128 bits.

### 4.1.5. SHA-1 (Secure Hash Algorithm 1, NIST 1993)

El proceso de generación de un resumen SHA-1<sup>3</sup> de 160 bits es muy similar al de MD5:

- **Relleno:** El mensaje se rellena con 100..0 hasta alcanzar una longitud máxima de 448 bits.
- **Adición de campo de longitud:** Se añade un campo de 64 bits que contiene la longitud original del mensaje.

<sup>2</sup>Para imágenes de ellos accede a la diapositiva 21 del tema 4

<sup>3</sup>Para imágenes de ellos accede a la diapositiva 22 del tema 4

- **División en bloques:** El mensaje se divide en bloques de 512 bits.
- **Procesamiento secuencial:** Los bloques se procesan secuencialmente para generar el resumen de 160 bits.

## 5 Firma Digital y certificados digitales

### 5.1. Firma Digital: Objetivos

La firma digital tiene varios objetivos clave en el contexto de la seguridad y la autenticación:

- **El receptor pueda autenticar al emisor:** La firma digital permite que el receptor verifique la identidad del emisor.
- **No haya repudio:** El emisor no puede negar la autenticidad del mensaje firmado, ya que solo él podría haber firmado ese mensaje con su clave privada.
- **El emisor tenga garantías de no falsificación (integridad):** La firma garantiza que el mensaje no ha sido alterado desde que fue firmado.

#### 5.1.1. Firma Digital con Clave Secreta y Asimétrica

Existen dos tipos de firmas digitales, dependiendo del sistema de claves utilizado:

- **Firma digital con clave secreta:** Usada en sistemas donde se utiliza una sola clave para cifrar y descifrar. Cabe destacar el ejemplo de Big Brother, que realiza diversas operaciones hashing para la clave secreta de la firma digital.

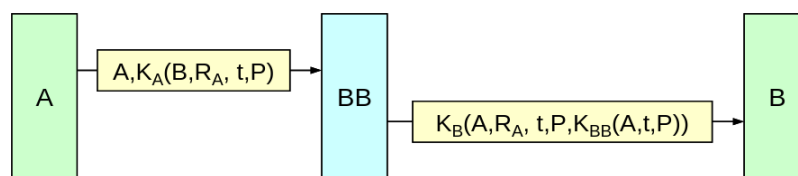


Figura 14: Big Brother

- **Firma digital con clave asimétrica:** Se utilizan dos claves, una pública y otra privada. El emisor firma el mensaje con su clave privada y el receptor verifica la firma con la clave pública del emisor.

### 5.1.2. El concepto de *Big Brother* en la firma digital

El concepto de *Big Brother* en la firma digital se refiere a un escenario donde una autoridad central o entidad de confianza supervisa, regula y controla el proceso de autenticación y verificación de firmas digitales. Esta idea se asemeja a un sistema donde todos los actores confían en una única entidad para garantizar la seguridad e integridad de las operaciones. A continuación, se describen los pasos fundamentales en este esquema:

1. **Generación de claves:** Cada usuario genera un par de claves criptográficas (una clave pública y una clave privada). La clave pública es registrada y almacenada por la autoridad central (el *Big Brother*).
2. **Emisión de certificados:** La autoridad central emite certificados digitales para asociar cada clave pública con la identidad del usuario. Estos certificados son firmados digitalmente por la autoridad, lo que les otorga validez y confiabilidad.
3. **Creación de la firma:** Para firmar un documento, el usuario utiliza su clave privada. Este proceso produce una firma digital única, basada en el contenido del documento y en la clave privada del usuario.
4. **Verificación de la firma:** La parte receptora utiliza la clave pública del usuario para verificar la autenticidad de la firma digital. La validez de la clave pública y su asociación con la identidad del firmante se confirman consultando el certificado digital emitido por la autoridad central.
5. **Control y supervisión:** La autoridad central actúa como un garante del sistema, asegurando que las claves públicas y los certificados estén actualizados y evitando posibles fraudes o duplicaciones.

Este modelo tiene la ventaja de centralizar la confianza, simplificando el proceso para los usuarios finales. Sin embargo, también presenta desafíos significativos, como la dependencia total de la autoridad central, que puede convertirse en un único punto de falla en el sistema.



El proceso de firma digital con clave asimétrica se puede describir mediante un doble cifrado:

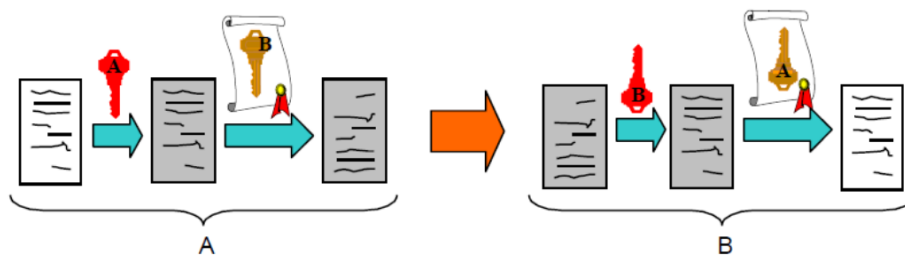


Figura 15: Proceso de firma digital con clave asimétrica.

1. Primero, para proporcionar privacidad, el mensaje  $T$  es cifrado con la clave pública  $K_{pubB}$  del receptor.
2. Luego, para autenticación, se cifra previamente con la clave privada  $K_{priA}$  del emisor.
3. El mensaje firmado se envía como  $K_{pubB}(K_{priA}(T))$ .
4. El receptor, al recibir el mensaje, lo descifra primero con su clave privada  $K_{priB}$ , luego con la clave pública del emisor  $K_{pubA}$ , y finalmente obtiene el mensaje original  $T$ .

### 5.1.3. Debilidad y Garantía de No Repudio

Una de las debilidades del sistema es que para garantizar el *no repudio*, se necesita asegurar la asociación indisoluble entre la "identidad A" y su "clave pública  $K_{pubA}$ ". Esto es necesario para garantizar que el emisor no pueda negar su identidad al haber firmado el mensaje.

Para resolver este problema, se utiliza un *certificado digital*, que garantiza la asociación entre una identidad y una clave pública.

### 5.1.4. Certificados Digitales

Un *certificado digital* es un documento que asocia de manera fidedigna una identidad a una clave pública. El proceso de obtención y verificación de un certificado digital incluye los siguientes pasos:

- El usuario genera su par de claves pública y privada.
- El usuario envía una solicitud a una *Autoridad de Certificación (AC)*, firmada digitalmente, indicando su identidad y su clave pública.
- La AC verifica la firma y, si todo es correcto, emite un certificado digital que contiene:

- La identidad de la AC.
  - La identidad del usuario.
  - La clave pública del usuario.
  - Otros datos, como el período de validez del certificado.
- El certificado es firmado digitalmente por la AC, utilizando su clave privada, para evitar su falsificación.

#### 5.1.5. Formato de los Certificados

El formato más común para los certificados digitales es el estándar *X.509*. Este formato asegura que los certificados sean compatibles entre diferentes sistemas y aplicaciones.

#### 5.1.6. Autoridades de Certificación (AC) Reconocidas

Algunas de las Autoridades de Certificación más reconocidas incluyen:

- ACE ([www.ace.es](http://www.ace.es))
- VeriSign ([www.verisign.com](http://www.verisign.com))
- CAMERFIRMA ([www.camerfirma.es](http://www.camerfirma.es))
- CERES ([www.cert.fnmt.es](http://www.cert.fnmt.es))

#### 5.1.7. Campos de un certificado X.509

<i>Field</i>	<i>Explanation</i>
Version	Version number of X.509
Serial number	The unique identifier used by the CA
Signature	The certificate signature
Issuer	The name of the CA defined by X.509
Validity period	Start and end period that certificate is valid
Subject name	The entity whose public key is being certified
Public key	The subject public key and the algorithms that use it

Figura 16: Campos de un certificado X.509.

## 5.2. Relación entre los Mecanismos de Seguridad y los Servicios/Aspectos de Seguridad

Existen varios mecanismos de seguridad que ayudan a garantizar los diferentes aspectos de la seguridad en los sistemas informáticos. A continuación, se describe la relación entre los mecanismos de seguridad y los servicios de seguridad más importantes:

### 5.2.1. Confidencialidad

La *confidencialidad* se refiere a garantizar que solo las partes autorizadas puedan acceder a la información. Se consigue mediante el uso de cifrado, que puede ser de dos tipos:

- **Cifrado simétrico:** Utiliza una única clave secreta compartida entre el emisor y el receptor para cifrar y descifrar el mensaje.
- **Cifrado asimétrico:** Utiliza un par de claves, una pública y una privada, donde el emisor cifra el mensaje con la clave pública del receptor y el receptor descifra con su clave privada.

Ambos métodos permiten garantizar que la información permanezca confidencial durante su transmisión.

### 5.2.2. Autenticación

La *autenticación* asegura que el receptor pueda verificar la identidad del emisor. Se consigue mediante varios mecanismos, como:

- **Reto-respuesta:** El receptor envía un reto (por ejemplo, una pregunta o desafío) al emisor, quien debe responder correctamente para probar su identidad.
- **Firma digital (Big Brother, doble cifrado):** La firma digital asegura que un mensaje provenga de una fuente específica. En el doble cifrado, el mensaje es cifrado primero con la clave pública del receptor para garantizar la privacidad y luego con la clave privada del emisor para garantizar la autenticidad.

### 5.2.3. No Repudio

El *no repudio* significa que el emisor no puede negar haber enviado un mensaje una vez que ha sido firmado. Este aspecto de la seguridad se logra mediante:

- **Firma digital (Big Brother, doble cifrado):** Al firmar digitalmente un mensaje con la clave privada del emisor, se asegura que solo él pudo haberlo firmado, lo que impide que pueda negar su participación en la transmisión del mensaje.

---

<sup>3</sup>Para más información de un certificado X.509 accede a la diapositiva 28 del tema 4.

### 5.2.4. Integridad

La *integridad* garantiza que el mensaje no haya sido alterado durante la transmisión. Se consigue mediante:

- **Compendios o resúmenes:** Se genera un valor hash del mensaje utilizando funciones hash. Este valor se adjunta al mensaje como un resumen, y cualquier alteración del mensaje cambiará el resumen, lo que permite detectar la manipulación.

### 5.2.5. Disponibilidad

La *disponibilidad* se refiere a asegurar que los servicios y recursos estén disponibles para los usuarios autorizados cuando los necesiten. Sin embargo, los mecanismos de seguridad mencionados anteriormente no proporcionan disponibilidad por sí solos. Para garantizar la disponibilidad, se requieren:

- **Sistemas antiataque:** Para proteger contra ataques de denegación de servicio (DoS) y otros que puedan afectar la disponibilidad.
- **Redundancia:** La redundancia en las líneas de acceso y en los servidores garantiza que, en caso de fallo de un sistema, otro pueda tomar su lugar y continuar brindando los servicios.

## 6 Protocolos Seguros

### 6.1. Seguridad

#### Seguridad Perimetral

La *seguridad perimetral* se refiere a la protección de las redes de comunicaciones de una organización contra accesos no autorizados o ataques desde el exterior. Para ello, se utilizan diversos mecanismos de seguridad, entre los que destacan:

- **Firewalls:** Son dispositivos o software que filtran el tráfico de red entre diferentes zonas, como la red interna y la externa, permitiendo o bloqueando conexiones según un conjunto de reglas predefinidas.
- **Sistemas de detección de intrusiones (IDS):** Son sistemas que monitorean el tráfico de red en busca de patrones o comportamientos sospechosos que puedan indicar un intento de intrusión.
- **Sistemas de respuesta a intrusiones (IRS):** Estos sistemas no solo detectan intrusiones, sino que también responden a ellas de manera automática, por ejemplo, bloqueando el acceso a ciertos recursos o alertando a los administradores.

#### Seguridad Criptográfica en Protocolos

La seguridad en los protocolos de comunicación se puede aplicar en diferentes capas del modelo OSI, cada una con sus propias soluciones de seguridad:



### Capa de Aplicación

En la capa de aplicación se implementan mecanismos de seguridad para proteger las comunicaciones a nivel de la aplicación. Algunos ejemplos son:

- **Pretty Good Privacy (PGP):** Es un protocolo de cifrado utilizado para la protección de correos electrónicos. PGP ofrece confidencialidad mediante cifrado, autenticación mediante firma digital y garantiza la integridad del mensaje mediante un hash.
- **Secure Shell (SSH):** Protocolo utilizado para acceder de manera segura a máquinas remotas, proporcionando cifrado de los datos, autenticación del servidor y del cliente, y confidencialidad en la comunicación.

### Capa de Sesión (Entre Aplicación y Transporte)

En esta capa, la seguridad se implementa mediante protocolos que proporcionan seguridad en la comunicación entre la capa de aplicación y la capa de transporte:

- **Transport Layer Security (TLS):** Es un protocolo criptográfico que proporciona seguridad en las comunicaciones a través de la red. TLS, que anteriormente era conocido como SSL (Secure Sockets Layer), se utiliza en protocolos como HTTPS, IMAPS, SSL-POP y VPN. TLS ofrece:
  - **Confidencialidad:** Mediante una clave secreta negociada entre el cliente y el servidor.
  - **Autenticación:** El servidor se autentica por defecto mediante su clave pública (KPUBLICA).
  - **Integridad:** Utilizando el algoritmo HMAC (Hash-based Message Authentication Code) para asegurar que los datos no han sido modificados durante la transmisión.
- **TLS Handshake:** El proceso de *Handshake* se utiliza para negociar los parámetros de seguridad antes de que los datos sean transmitidos. Esto incluye la negociación de la clave secreta y la autenticación del servidor.
- **TLS Record Protocol:** Después del *Handshake*, el protocolo de registro se encarga de cifrar y autenticar los datos transmitidos.

### Capa de Red

En la capa de red se implementan mecanismos de seguridad para proteger el tráfico de red en su totalidad. Un ejemplo de ello es:

- **IPSec (VPN):** IPSec es un conjunto de protocolos que proporcionan seguridad a nivel de red, protegiendo los datos a través de la encriptación y autenticación. Se utiliza principalmente en redes privadas virtuales (VPN), proporcionando confidencialidad, autenticación de los paquetes de datos y protección contra ataques de modificación.

## 6.2. Pretty Good Privacy (PGP) – correo electrónico seguro

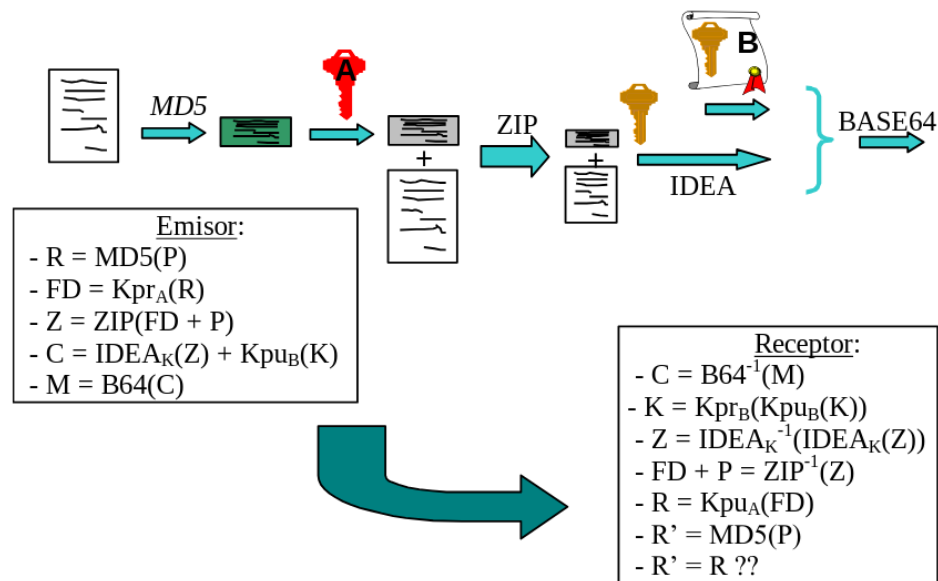


Figura 17: Pretty Good Privacy (PGP).

### Proceso de Cifrado y Descifrado en PGP

PGP (Pretty Good Privacy) es un método de cifrado para la comunicación segura por correo electrónico. A continuación se detalla el proceso de cifrado y descifrado utilizado en PGP.

#### Cifrado por el Emisor

1. **Generación de Resumen:** Se genera un resumen del mensaje original  $P$  usando el algoritmo MD5:

$$R = \text{MD5}(P) \quad (1)$$

2. **Firma Digital:** Se cifra el resumen utilizando la clave privada del emisor  $K_{prA}$ :

$$FD = K_{prA}(R) \quad (2)$$

3. **Compresión:** Se comprimen la firma digital y el mensaje original:

$$Z = \text{ZIP}(FD + P) \quad (3)$$

4. **Cifrado del Mensaje:** Se cifra el mensaje comprimido usando el algoritmo IDEA y una clave de sesión  $K$ . Además, se cifra la clave de sesión usando la clave pública del receptor  $K_{pubB}$ :

$$C = \text{IDEA}_K(Z) + K_{pubB}(K) \quad (4)$$

5. **Codificación Base64:** El mensaje cifrado se codifica en formato Base64 para su transmisión:

$$M = \text{B64}(C) \quad (5)$$

**Descifrado por el Receptor**

1. **Decodificación Base64:** El receptor decodifica el mensaje del formato Base64:

$$C = B64^{-1}(M) \quad (6)$$

2. **Descifrado de la Clave de Sesión:** El receptor descifra la clave de sesión usando su clave privada  $K_{prB}$ :

$$K = K_{prB}(K_{pubB}(K)) \quad (7)$$

3. **Descifrado del Mensaje:** Se descifra el mensaje comprimido usando la clave de sesión:

$$Z = IDEA_K^{-1}(C) \quad (8)$$

4. **Descompresión:** Se descomprime el mensaje para obtener la firma digital y el mensaje original:

$$FD + P = ZIP^{-1}(Z) \quad (9)$$

5. **Verificación de la Firma:** Se descifra la firma digital usando la clave pública del emisor  $K_{puA}$ :

$$R = K_{puA}(FD) \quad (10)$$

6. **Verificación del Resumen:** Se genera un resumen del mensaje recibido y se compara con el resumen cifrado recibido:

$$R' = MD5(P) \quad (11)$$

$$R' \stackrel{?}{=} R \quad (12)$$

**6.3. Transport Layer Security (TLS) / Secure Sockets Layer (SSL)**

El protocolo *Transport Layer Security (TLS)* (anteriormente conocido como *Secure Sockets Layer (SSL)*) proporciona seguridad en las comunicaciones de red a través de la confidencialidad, autenticación e integridad. Se utiliza ampliamente en protocolos como **HTTPS**, **IMAPS**, **SSL-POP** y **VPN**.

**6.3.1. SSL Record Protocol**

El *SSL Record Protocol*<sup>4</sup> encapsula los datos de otros protocolos y ofrece un canal seguro que garantiza:

- **Privacidad:** A través de la encriptación de los datos.
- **Autenticación:** Garantizando que el servidor es quien dice ser.
- **Integridad:** Usando técnicas de hash para asegurar que los datos no sean modificados durante la transmisión.

Este protocolo se encarga de fragmentar, comprimir y cifrar los datos antes de su transmisión a través de la red.

<sup>4</sup>Para ver imágenes sobre este apartado revisa las diapositivas 33 y 34 del tema 4

### 6.3.2. SSL Handshake Protocol

El *SSL Handshake Protocol* es el proceso mediante el cual el cliente y el servidor negocian los parámetros de seguridad antes de la transmisión de datos. Este protocolo realiza las siguientes funciones clave:

- **Negociación del algoritmo de cifrado:** El cliente y el servidor acuerdan qué algoritmo de cifrado utilizar para proteger la comunicación.
- **Negociación de la función hash:** Se selecciona la función hash que se utilizará para garantizar la integridad de los datos.
- **Autenticación del servidor:** El servidor se autentica utilizando un certificado X.509, lo que permite al cliente verificar su identidad.

### 6.3.3. Generación de Claves de Sesión

Durante el proceso de *handshake*, se generan las claves de sesión necesarias para proteger la comunicación. Estas claves se generan de la siguiente manera:

- **Claves aleatorias cifradas con la clave pública del servidor (KPUB\_SERVER):** El cliente genera claves de sesión aleatorias que son cifradas con la clave pública del servidor para asegurar su confidencialidad.
- **Diffie-Hellman:** Alternativamente, se puede utilizar el protocolo Diffie-Hellman para intercambiar de manera segura una clave compartida sin necesidad de transmitirla directamente.

### 6.3.4. SSL Assert Protocol

El *SSL Assert Protocol* es utilizado para informar sobre cualquier error o anomalía detectada durante la sesión, asegurando que las partes involucradas puedan reaccionar ante posibles problemas de seguridad.

### 6.3.5. Change Cipher Spec Protocol

El *Change Cipher Spec Protocol* notifica cualquier cambio en la configuración del cifrado durante una sesión SSL/TLS. Este protocolo se utiliza para asegurarse de que tanto el cliente como el servidor estén de acuerdo en los cambios realizados en los parámetros de cifrado, como el algoritmo o las claves de sesión.

### 6.3.6. Protocolos Utilizados con TLS/SSL

TLS/SSL es utilizado en una variedad de protocolos para asegurar la comunicación. Algunos ejemplos incluyen:

- **HTTPS:** Protocolo de transferencia de hipertexto seguro utilizado para comunicaciones seguras en la web.



- **IMAPS:** Protocolo de acceso a mensajes de internet seguro, utilizado para acceder a correos electrónicos de manera segura.
- **SSL-POP:** Protocolo de oficina de correos seguro, usado para acceder a correos electrónicos en servidores POP.
- **VPN:** Redes privadas virtuales que utilizan TLS/SSL para crear túneles seguros entre redes.

## 6.4. IPSec

El objetivo principal de *IPSec*<sup>5</sup> es garantizar la *autenticación*, *integridad* y, opcionalmente, la *privacidad* a nivel de la capa IP. Este conjunto de protocolos se utiliza para asegurar las comunicaciones a través de redes IP, proporcionando confidencialidad e integridad en los datos.

### 6.4.1. Procedimientos de IPSec

IPSec se compone de tres procedimientos clave:

#### 1. Establecimiento de una Asociación de Seguridad:

- Utiliza el protocolo *IKE (Internet Key Exchange)* especificado en la **RFC 2409**.
- El objetivo es establecer una clave secreta compartida entre los participantes, usando el protocolo *Diffie-Hellman*.
- Incluye una autenticación previa de los participantes, mediante certificados, para evitar el ataque de *hombre en el medio*.
- La asociación de seguridad es *simplex*, lo que significa que tiene un único sentido, es decir, se establece para un único flujo de datos.
- La asociación se identifica mediante la combinación de la IP de origen y un *Security Parameter Index (SPI)* de 32 bits.
- Una de las limitaciones de este procedimiento es que vulnera el carácter *no orientado a conexión* de la capa IP.

#### 2. Garantizar la Autenticación e Integridad de los Datos:

- Se utiliza el protocolo de *Cabeceras de Autenticación* especificado en la **RFC 2401**.
- Este protocolo asegura que los datos transmitidos no hayan sido alterados y autentica a los participantes en la comunicación.

#### 3. (Opcional) Garantizar la Autenticación, Integridad y Privacidad de los Datos:

- Se utiliza el protocolo de *Encapsulado de Seguridad de la Carga* especificado en la **RFC 2411**.
- Este protocolo garantiza tanto la integridad como la confidencialidad de los datos, cifrando la carga útil de los paquetes.

<sup>5</sup>Para ver imágenes sobre este apartado revisa las diapositivas 36 del tema 4

### 6.4.2. Modos de Operación de IPSec

IPSec puede operar en dos modos diferentes, según el alcance de la protección que se desea ofrecer:

- **Modo Transporte:**

- En este modo, la asociación de seguridad se establece entre el host de origen y el host de destino de forma directa, asegurando los datos a nivel de la comunicación entre estos dos puntos finales.

- **Modo Túnel:**

- En el modo túnel, la asociación de seguridad se establece entre dos routers intermediarios, de manera que todo el tráfico entre estos routers se encuentra protegido, creando un *túnel* seguro para los datos que se transmiten a través de la red.