Tema 5: Seguridad en redes

Bibliografía:

[Kurose11] Secciones 1.6, 8.1, 8.2 (excepto la introducción) y 8.2.1

Objetivos

- Describir los fundamentos de seguridad en un sistema de redes de computadores
  - Explicar las aplicaciones de la criptografía, más allá de la confidencialidad
  - Explicar las técnicas para proteger la integridad de los mensajes
  - Cómo conseguir autenticación
- Seguridad en la práctica: sockets seguros

Índice

Propiedades (I)

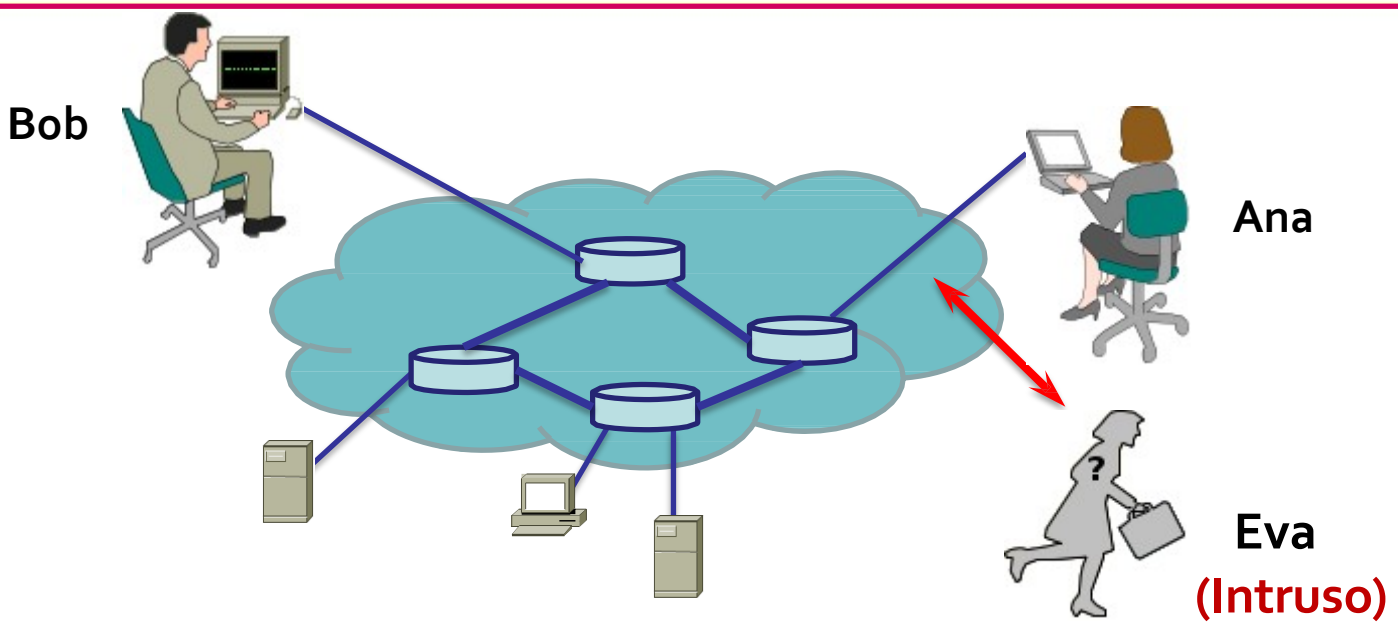
- **Propiedades deseables en una comunicación segura:**
  - **Confidencialidad**: el contenido del mensaje disponible únicamente para el emisor y el receptor
    - Solución: mediante cifrado
      - El emisor cifra el mensaje
      - El receptor descifra el mensaje
  - **Autenticación**: emisor y receptor quieren confirmar la identidad de la otra parte
    - Solución: mediante técnicas criptográficas

Propiedades (II)

- **Integridad del mensaje:** contenido del mensaje inalterado durante la transmisión (accidental o intencionadamente)
  - Solución: técnicas adicionales al control de errores empleado en los niveles de transporte y enlace de datos

- **Control de acceso y disponibilidad:** recursos de la red accesibles y disponibles solo para los usuarios legítimos
  - Solución: restricción en el control de acceso con mecanismos de autenticación, cortafuegos, etc.

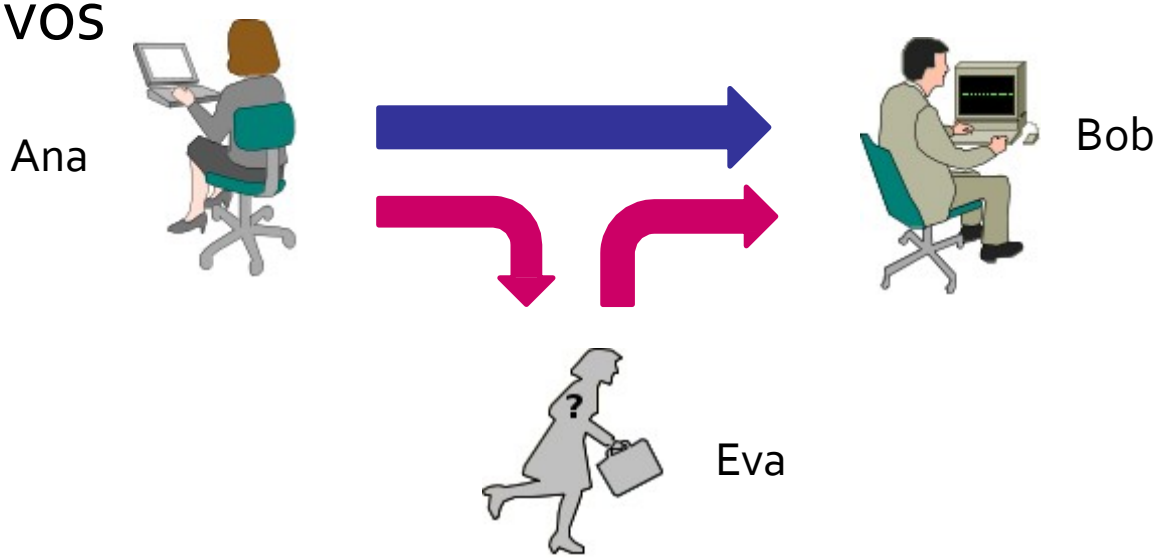
Buenos y malos



- Ana y Bob desean comunicarse de "**forma segura**"
- Ana y Bob pueden ser:
  - 2 routers que quieren intercambiar sus tablas de encaminamiento
  - Dos servidores DNS
  - Un cliente y un servidor que quieren establecer una conexión de transporte segura:
    - Transacciones bancarias on-line
    - Comercio electrónico
    - ...

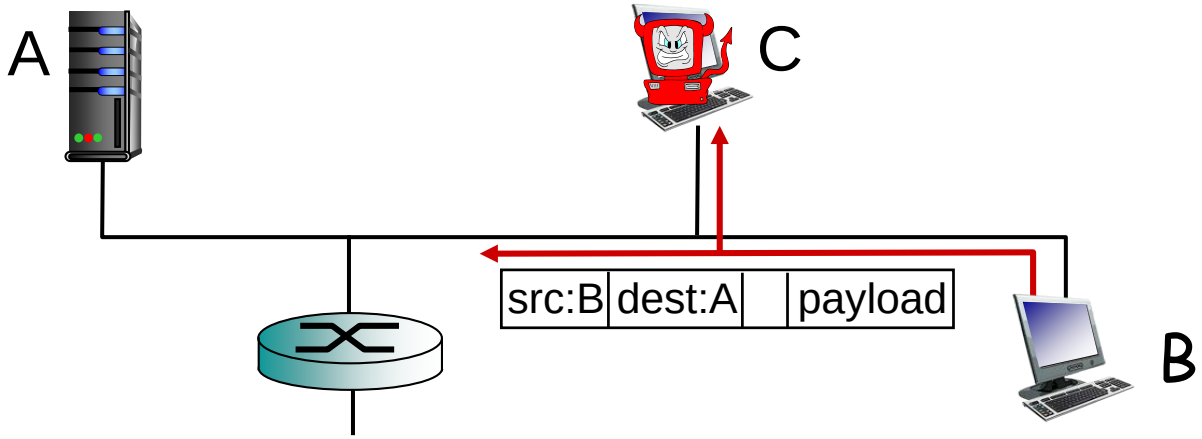
Tipos de *malos*

- **Intruso pasivo**
  - Puede escuchar y recoger información que circula por la red: contraseñas, información sobre tarjetas de crédito, etc.
- **Intruso activo**
  - Puede modificar mensajes, eliminarlos y/o añadir otros nuevos



Acciones de los *malos*

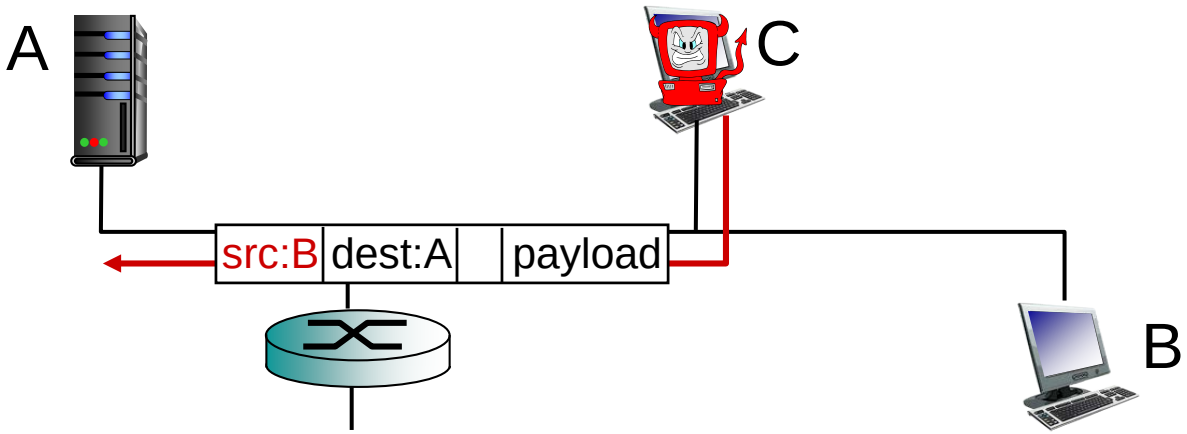
- Examen de paquetes *(Packet sniffing)*
  - Una interfaz de red lee/graba todos los paquetes (incluyendo passwords!) que van por el medio



  - El software Wireshark utilizado en la prácticas es un *husmeador de paquetes* (sw libre)

Acciones de los *malos*

- Suplantación IP *(IP spoofing)*
  - Mandar un paquete con una dirección fuente falsa

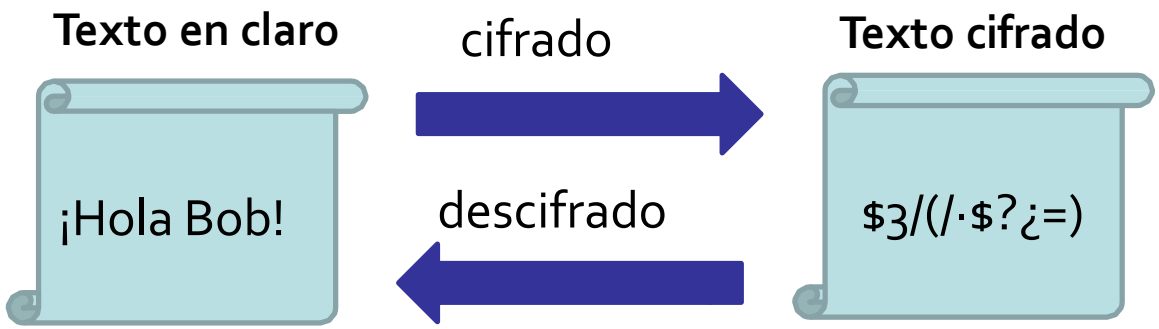


Acciones de los *malos:* resumen

- **¿Qué puede hacer un *tipo malo*?**
  - **Espiar:** interceptar mensajes
  - **Inserción** de mensajes en la conexión
  - **Suplantación:** puede falsear *(spoof)* la dirección origen en paquetes (o cualquier campo en el paquete)
  - **Secuestro *(hijacking)*:** "hacerse cargo" de la conexión mediante la eliminación del emisor o receptor, insertándose él mismo en su lugar
  - **Denegación de servicio**: impedir que el servicio sea utilizado por otros (por ejemplo, por sobrecarga de los recursos)

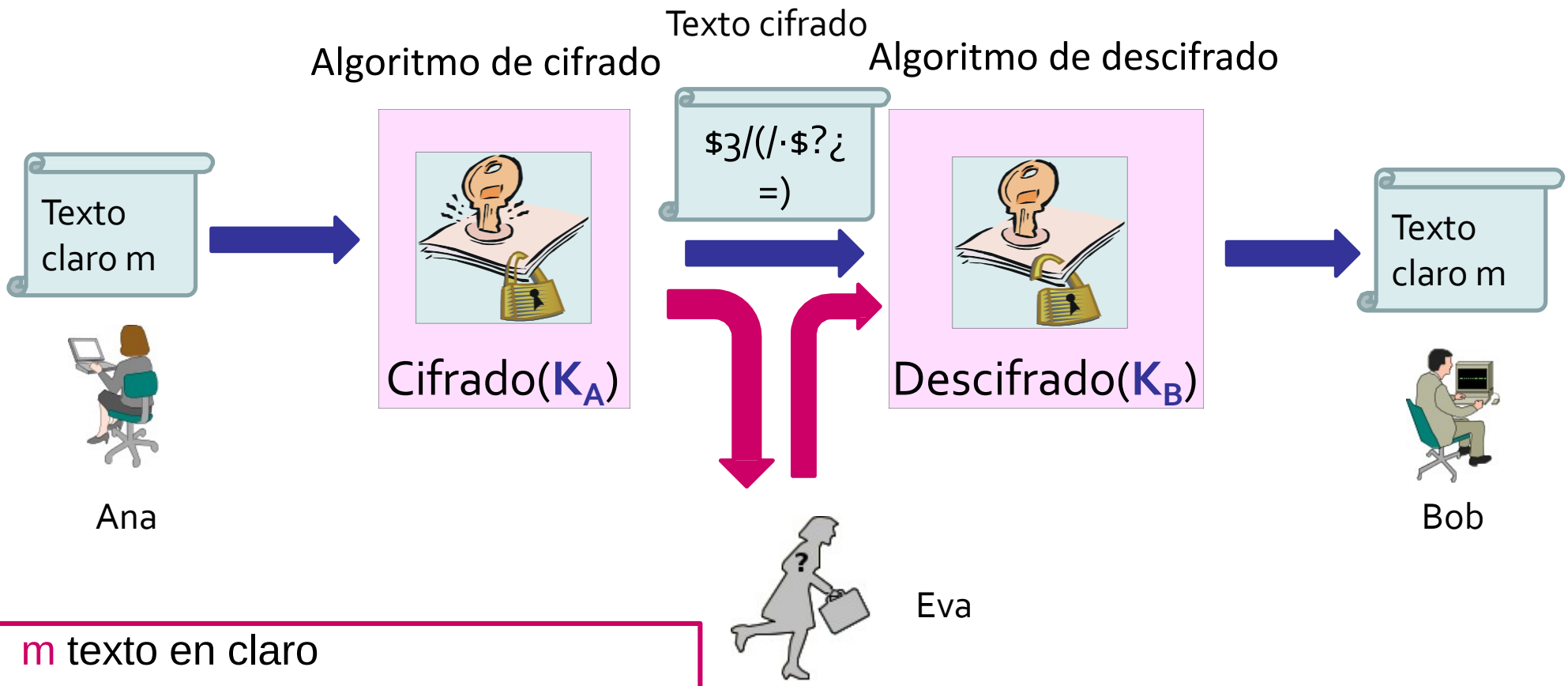
Índice

- **Criptografía:** ciencia y arte de modificar los datos para que solo puedan ser conocidos por su emisor y el receptor deseado
  - Datos modificados → texto cifrado
  - Texto sin modificar → texto nativo o texto en claro



Componentes criptográficos

Para cifrar o descifrar se requiere un algoritmo y una o más claves (K_A, K_B)



m texto en claro
$K_A(m)$ texto cifrado con la clave K_A
$m = K_B(K_A(m))$

- **Cifrados de sustitución:** sustituir una cosa por otra
  - **Cifrado monoalfabético:** sustituye una letra por otra

```
Texto plano:   abcdefghijklmnopqrstuvwxyz
Texto cifrado: mnbvcxzasdfghjklpoiuytrewq
```

Ejemplo: Texto en claro: **hola bob.alice**
Texto cifrado: **akgm nkn.mgsbc**

- ¿K?
- Caso particular: **cifrado de César**
  - Sustitución cíclica de las letras del alfabeto
  - ¿K?

Cifrados de sustitución

- **Cifrado Monoalfabético**
  - Cifrado de César.
    - Texto claro
    - Cifrado K=11

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	x	y	z
l	m	n	o	p	q	r	s	t	u	v	x	y	z	a	b	c	d	e	f	g	h	i	j	k

  - "Hola clase" → "Saxl nxlep"
- **Cifrado Polialfabético**
  - Combinación de cifrados monoalfabéticos con un patrón cíclico
    - Texto claro
    - Cifrado K=11
    - Cifrado K=4

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	x	y	z
l	m	n	o	p	q	r	s	t	u	v	x	y	z	a	b	c	d	e	f	g	h	i	j	k
d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	x	y	z	a	b	c

  - Ejemplo patrón $C_1C_1C_2C_1C_2$: "Hola clase" → "Saol fxlvp"

Cómo romper un cifrado monoalfabético

- Ataque de solo texto cifrado
  - Solo se tiene acceso al texto cifrado
  - Dos aproximaciones:
    - Fuerza bruta: búsqueda a través de todo el espacio de claves
    - Análisis estadístico

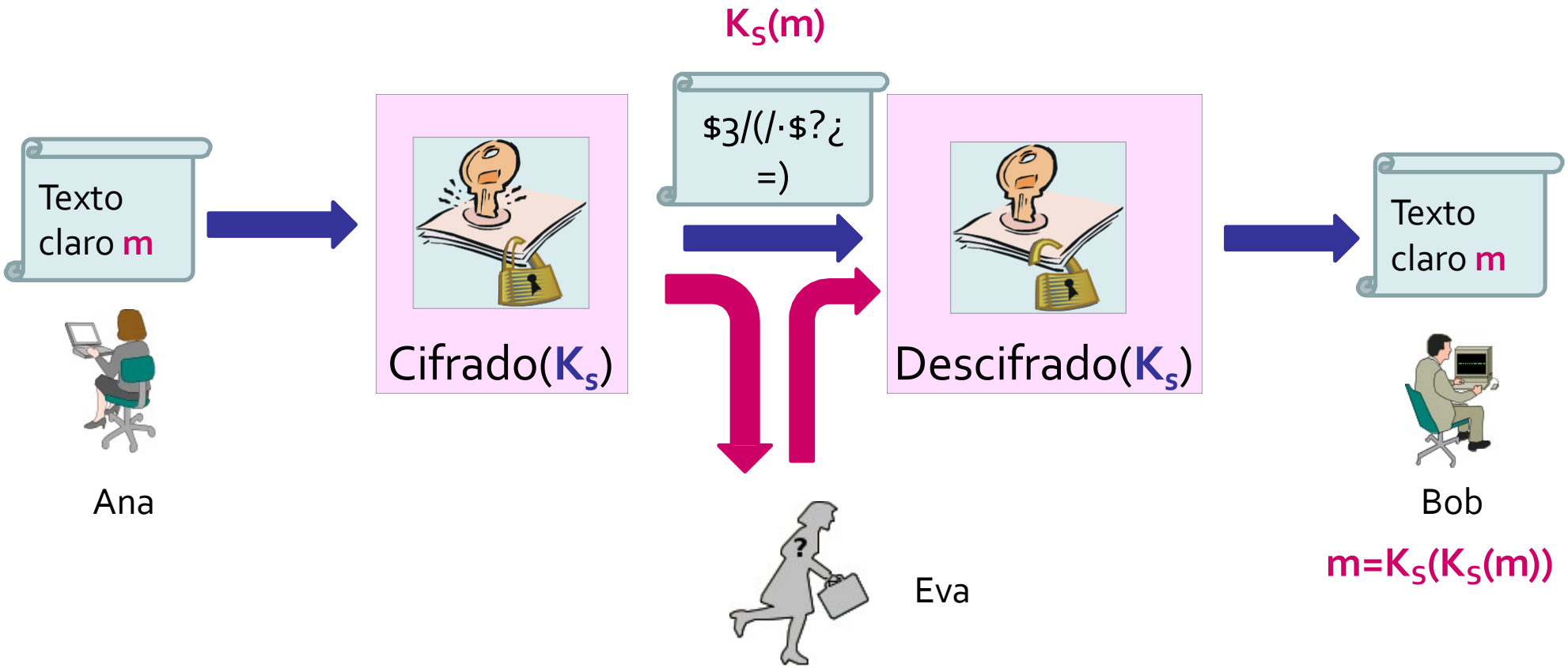
- Ataque de texto en claro conocido
  - Se tiene acceso a un texto en claro que se corresponde con el cifrado
  - En un cifrado monoalfabético pueden identificarse partes de palabras repetidas
- Ataque de texto en claro seleccionado
  - El intruso puede obtener texto cifrado para el texto en claro que elija.

Tipos de criptografía

- Al emplear criptografía
  - El algoritmo suele ser conocido, incluso un estándar
  - Solo las claves son secretas
- Criptografía de clave simétrica
  - Solo una clave
- Criptografía de clave pública
  - Dos claves
- Funciones *hash* criptográficas
  - *Sin claves, ¿qué utilidad puede tener?*

Índice

Cifrado de clave simétrica o secreta



- Bob y Ana comparten la misma clave K_S
  - Ej., la clave puede ser el patrón de sustitución en un cifrado de sustitución
- Problema: ¿cómo se puede transmitir la clave de forma segura?

Cifrado de bloques

- Los algoritmos de clave simétrica trabajan habitualmente con bloques de tamaño fijo de k bits (por ej. 64 bits)
- Realizan una correspondencia 1-a-1: k bits de texto en claro corresponden a k bits de texto cifrado.
- Ejemplo con k=3:

Claro	Cifrado	Claro	Cifrado
000	110	100	011
001	111	101	010
010	101	110	000
011	100	111	001

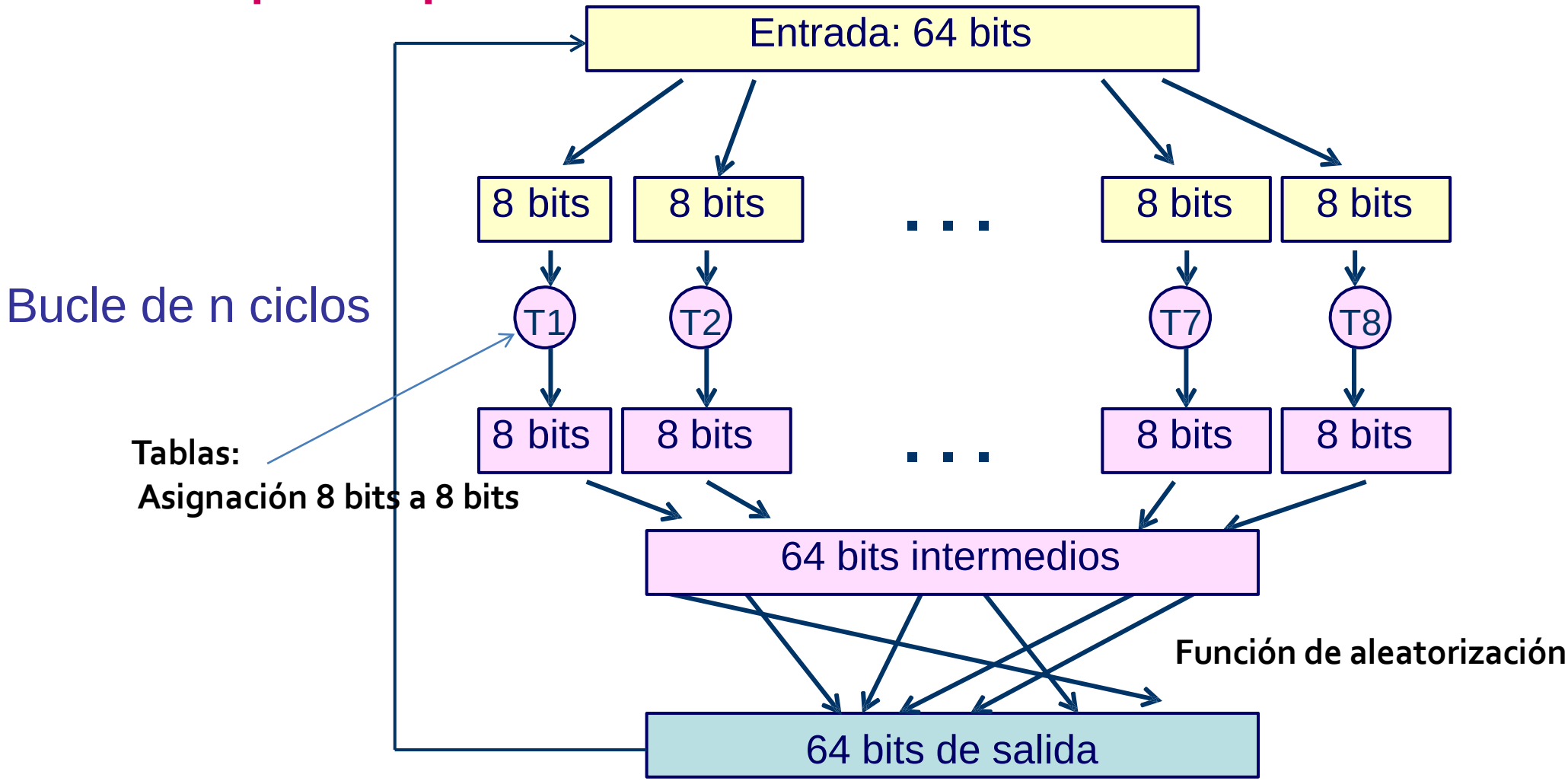
¿Cuál sería el cifrado para 010110001111? ➔ **101000111001**

Cifrado de bloques

- ¿Cuántas combinaciones hay para k=3?
  - ¿Cuántas entradas en la tabla diferentes con 3 bits?
  - ¿Cuántas permutaciones posibles para las diferentes entradas?
  - 40.320 (2^3!), no muchas
- En general, se emplean bloques de mayor tamaño (para k=64 bits, 2^{64}! permutaciones posibles)
  - Problema: Para bloques de 64 bits sería necesaria una tabla con 2^{64} entradas de 64 bits
  - En su lugar, se emplean funciones que simulan la generación de tablas aleatoriamente permutadas.

Cifrado de bloques

Función prototipo



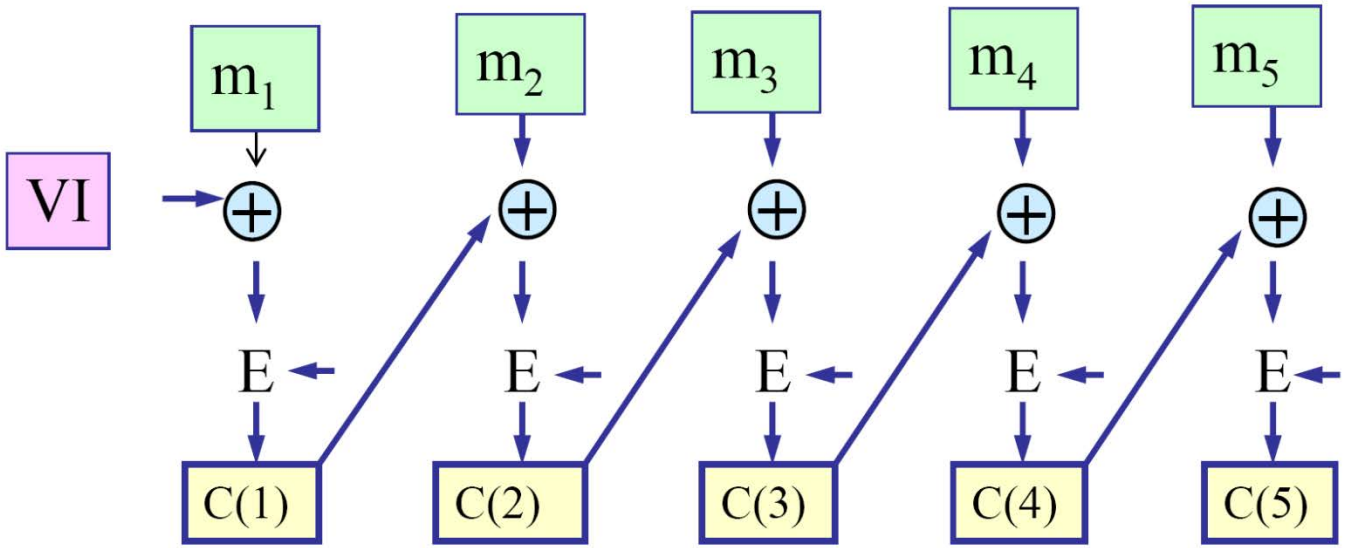
Cifrado de bloques

- ¿Por qué n ciclos?
  - Si hay un solo ciclo, un bit de entrada afecta como mucho a 8 bits de salida.
  - En un segundo ciclo, los 8 bits afectados quedan diseminados e introducidos en múltiples bloques de sustitución.
  - ¿Cuántas rondas?
    - ¿Cuántas veces se necesita barajar las cartas?
    - Menos eficiente conforme va aumentando *n*.

Cifrado de mensajes largos

- Evitar la repetición de bloques
  - Ejemplo: "GET " ó "HTTP/1.1"
  - Con el método anterior, el mismo bloque de texto en claro producirá siempre el mismo texto cifrado
- **CBC (*Cipher Block Chaining*)**
  - El emisor genera un **IV** (vector de inicialización), c(0), bloque aleatorio y lo envía al receptor (en claro).
  - Antes de cifrar el primer bloque, calcula la XOR del bloque a cifrar con el vector **IV** :
    - $c(1)=K_s(m(1)\ XOR\ c(0))$
  - Para cada bloque posterior, aplica a los datos la XOR con el anterior bloque cifrado **antes de cifrarlo**:
    - $c(i)=K_s(m(i)\ XOR\ c(i-1))$

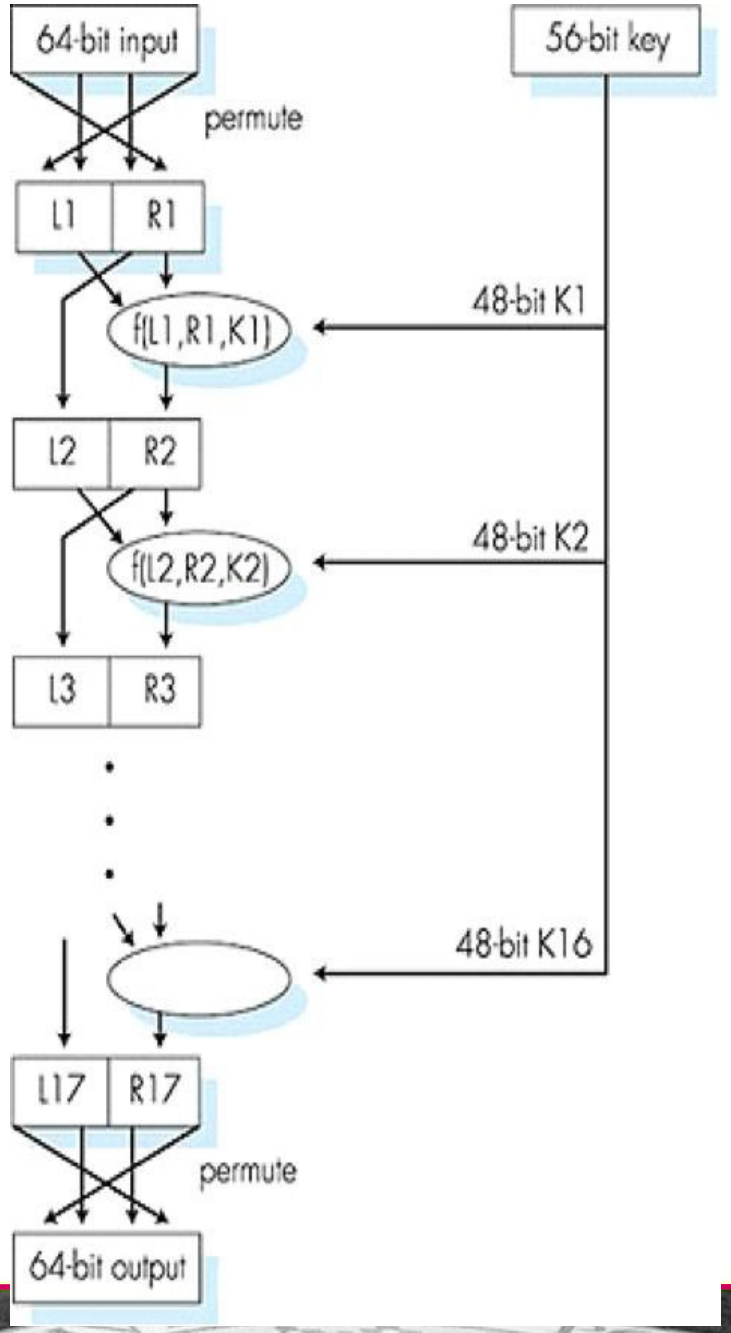
Encadenamiento de bloques cifrados (CBC)



- **Vector de inicialización (VI), c(0)**
  - No necesita ser secreto
  - Se cambia para cada mensaje (o sesión)
    - Se garantiza que, incluso si se envía repetidamente el mismo mensaje, el texto cifrado será completamente diferente cada vez.

Estándares de clave simétrica

- Establecidos por el **NIST** *(National Institute of Standards and Technology)*
- DES (*Data Encryption Standard*) (1993)
  - Bloques 64 bits, clave 56 bits.
  - Cifrado de bloques con CBC (encadenado)
  - Operación:
    - Permutación inicial y final
    - 16 vueltas idénticas aplicando una función, utilizando cada vez una clave distinta de 48 bits



Estándares de clave simétrica (DES y 3DES)

- ¿Cuán seguro es DES?
  - A causa de las mejoras de los procesadores , actualmente se puede averiguar la clave en menos de un día (fuerza bruta).
- Haciendo DES más seguro:
  - **3DES**:
    - Encripta 3 veces con 3 claves diferentes
    - Clave 168 bits (3x56), se usa como 3 claves de 56 bits

Estándares de clave simétrica (AES)

- AES *(Advanced Encryption Standard)*
  - Reemplaza a DES en Nov. 2001
  - Procesa datos en bloques de 128 bits
  - Claves de 128, 192 o 256 bits
  - En 2003 el gobierno de EEUU aprueba su uso para cifrar información clasificada.
  - Desencriptado con fuerza bruta:
    - Suponiendo una máquina que desencriptara DES en un segundo, para desencriptar AES tardaría 149 billones de años.

Clave simétrica: conclusiones

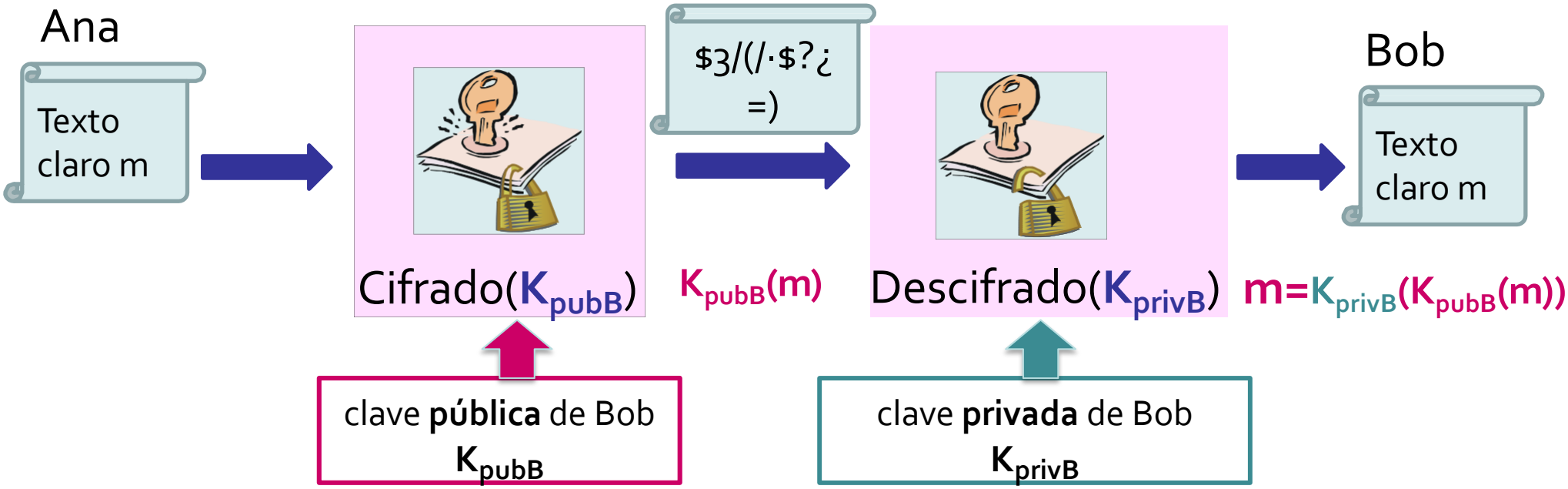
- Permite:
  - Conseguir **confidencialidad** en los mensajes que se transmiten
  - Garantizar la **integridad** de los mensajes
  - Realizar **autenticación**
- Problema:
  - Distribución de una clave común al emisor y al receptor a través de un canal inseguro
- Solución:
  - **Criptografía de clave pública**

Índice

1. Seguridad en las comunicaciones
2. Principios de criptografía
   i. Criptografía de clave simétrica
   ii. Criptografía de clave pública
3. Integridad de los mensajes
4. Autenticación de terminal
5. Conexiones TCP seguras: SSL

Clave pública: componentes

- Criptografía **asimétrica**
- Dos claves distintas
  - Pública: disponible para todo el mundo (K_{pub})
  - Privada: la conoce solo su dueño (K_{priv})
- Cifrado: $m' = K_{pub}(m)$
- Descifrado: $m=K_{priv}(K_{pub}(m))$



Clave pública: autenticación

- Algunos de los algoritmos presentan una interesante propiedad:

$$\underbrace{K_{pubB}(K_{privB}(m))}_{\text{Primero la clave pública, seguido de la privada}} = m = \underbrace{K_{privB}(K_{pubB}(m))}_{\text{Primero la clave privada, luego la pública}}$$

- Cifrar con K_{privB} permite conseguir autenticación y no-repudiación de mensajes (no confidencialidad)
  - **Firmas digitales** (se verán más adelante)

Clave pública: fundamentos

- Mucho **más moderna** que la criptografía simétrica
  - Primera publicación **Diffie-Hellman**, 1976
- … y mucho más **costosa computacionalmente**
  - Algoritmos basados en asimetrías de problemas matemáticos complejos:
    - **RSA (Rivest,Shamir,Adelson algorithm):** aritmética modular de números primos
    - **El Gamal:** problema del logaritmo discreto
    - …
  - Se utiliza en muchos casos para transmitir claves de sesión secretas entre dos sistemas
  - También para conseguir autenticación y no repudio de mensajes (firmas digitales)

Introducción a RSA

- Un mensaje es un patrón de bits
- Un patrón de bits puede ser representado por un número entero
- Por lo tanto, cifrar un mensaje equivale a cifrar un número
- Ejemplo:
  - M=10010001 puede ser representado por el número 145
  - Para cifrar M, se cifrará el número equivalente para obtener un número nuevo (el texto cifrado)

RSA

- Se escogen *p* y *q*, números primos muy grandes (*p·q* del orden de 1024 bits).
- Sea *n=p·q* y *z=(p-1)·(q-1)*
- Se elige *e*, *(e<n)*, de forma que no contenga ningún factor común con *z* (e y z son coprimos, mcd(e,z)=1)
- Se elige *d* tal que *e·d* mod *z* = *1* (*e·d-1* divisible exactamente por z)
- Clave pública = (*n*, *e*)
- Clave privada = (*n*, *d*)
- Para cifrar un mensaje *m*: $c = m^e \bmod n$
- Para descifrar c: $m = c^d \bmod n$

Ejemplo: RSA

- Cifrado del mensaje m:

  $c = m^e \bmod n$

- Descifrado del mensaje c:

  $c^d \bmod n = (m^e \bmod n)^d \bmod n = m^{e \cdot d} \bmod n =$

  $= m^{(e \cdot d \bmod z)} \bmod n = m^1 \bmod n = m$

Según aritmética modular

$[(a \bmod n) + (b \bmod n)] \bmod n = (a+b) \bmod n$

$[(a \bmod n) - (b \bmod n)] \bmod n = (a-b) \bmod n$

$[(a \bmod n) \cdot (b \bmod n)] \bmod n = (a \cdot b) \bmod n$

$(a \bmod n)^d \bmod n = a^d \bmod n$

Según teoría de números

Si p y q son primos y

Si $n = p \cdot q$ y

Si $z = (p-1) \cdot (q-1)$

entonces

$x^y \bmod n = x^{(y \bmod z)} \bmod n$

- Para romper el cifrado, hay que averiguar p y q factorizando n.
  - No existen algoritmos **conocidos** para factorizar rápidamente un número.

Clave pública: conclusiones

- El cálculo de este cifrado es muy costoso computacionalmente, por lo que se limita a pequeños bloques de datos:
  - Intercambio de claves secretas
  - Autenticación de terminales
  - Firmas digitales
- DES es como mínimo 100 veces más rápido que RSA
  - En la práctica, se utiliza RSA para intercambiar la claves simétrica
  - Una vez intercambiada, se emplea criptografía de clave simétrica

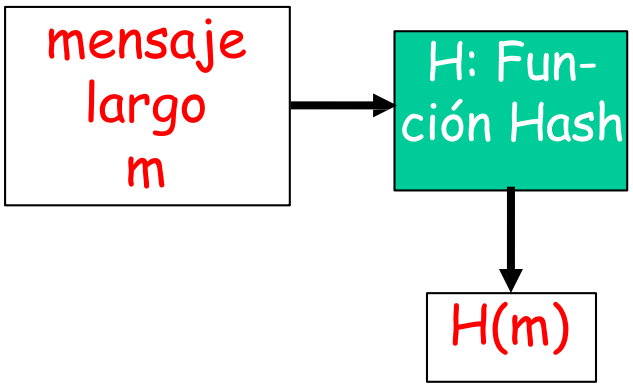
Índice

1. Seguridad en las comunicaciones
2. Principios de criptografía
3. Integridad de los mensajes
   i. Funciones hash
   ii. Código de autenticación (MAC)
   iii. Firma digital
   iv. Certificación y distribución de claves
4. Autenticación del punto terminal
5. Conexiones TCP seguras: SSL

- Aún cuando la confidencialidad no sea un requisito básico, pueden haber otras consideraciones de seguridad:
  - **Integridad**: Garantizar que el mensaje no se ha modificado en tránsito, accidental o intencionadamente
  - **Autenticación**: Verificar que el origen del mensaje es realmente quién dice ser
- Existen dos tipos de soluciones, basadas en clave secreta o basadas en clave pública

i. Resumen del mensaje. Función Hash

- Cifrar todo el mensaje puede resultar costoso (cálculo y almacenamiento)
- **Solución eficiente**: cifrar un bloque de bits pequeño obtenido a partir del documento (resumen del mensaje), mediante una función (función hash, **H**)

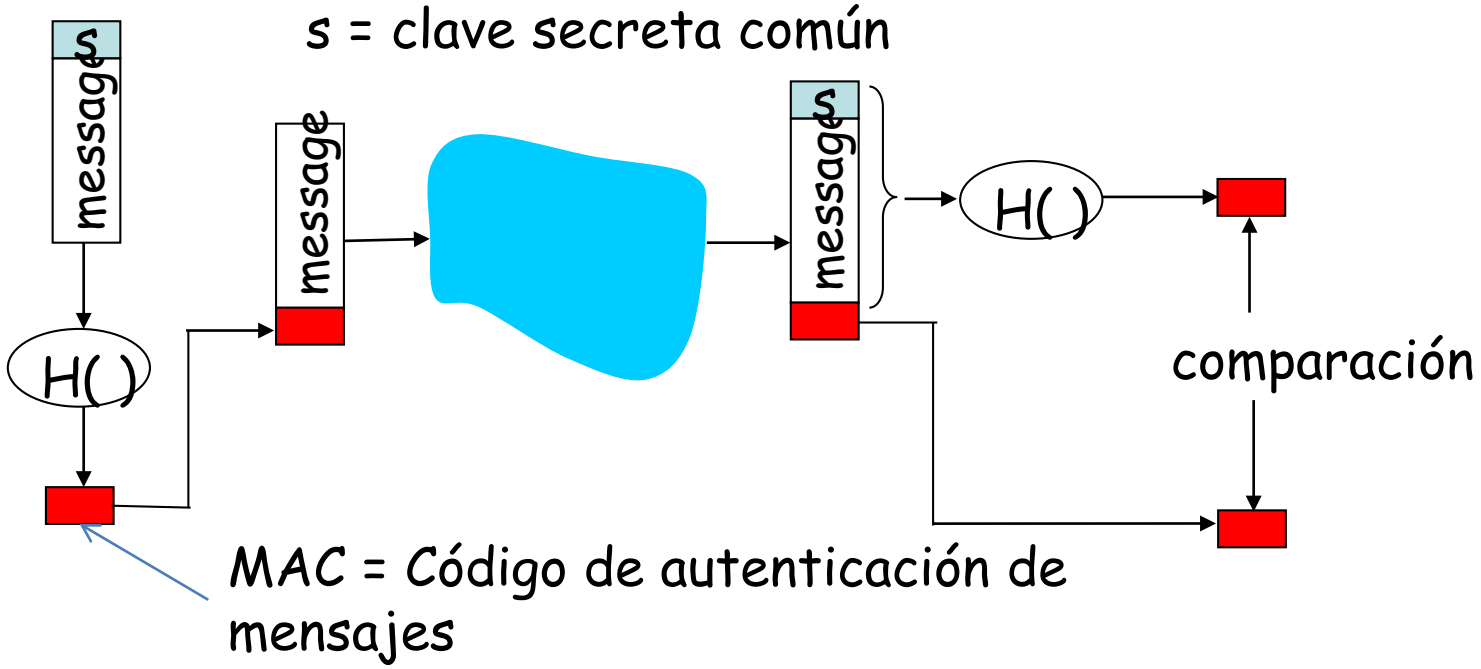


- **Propiedades deseables de H(m)**
  - **Salida de longitud fija**
  - Fácil de calcular (computacionalmente)
  - **Irreversible**
  - No se pueden encontrar (computacionalmente) dos mensajes distintos que den el mismo resultado (**Resistencia a la colisión**)
- Idea similar al *checksum* o a los CRC
- Ejemplos: MD5 (128 bits), SHA-1 (160 bits),SHA-2 (224,256,384 y 512 bits) y SHA-3

ii. Código de autenticación (MAC)

- Para garantizar además de la integridad del mensaje que su origen es veraz ➔ Código de autenticación (MAC)
- Se basa en la existencia de una clave secreta común, ***s***, que solo conocen emisor y receptor
- Se genera un MAC (*Message Authentication Code*) que se añade al mensaje
- El receptor evalúa el MAC
- Ejemplo: *HMAC* (*Hash-based MAC*)
  - Empleado por OSPF *(Open Shortest Path First)* (difusión de tablas de encaminamiento), IPSec y TLS (Transport Layer Security)

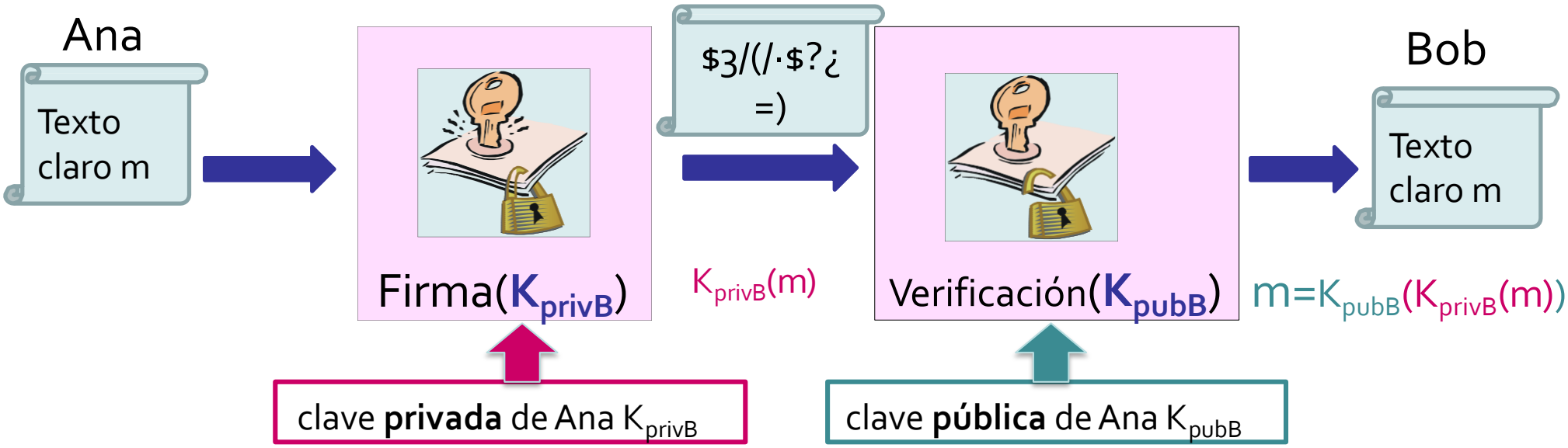
ii. Código de autenticación (MAC)



- **HMAC (Hash-based Message Authentication Code)**:
  - El emisor genera un MAC como el *Hash* (MD5 o SHA-1) del mensaje y de la clave, y lo añade al mensaje.
- Se autentifica el emisor, se verifica la integridad del mensaje (Autenticación de mensaje)
- **No hay cifrado**
- **Problema de difusión de la clave**

iii. Firmas digitales

- Basadas en clave pública ➔ muy costoso
- Permiten demostrar:
  - Quién generó la información
    - Impiden la repudiación del mensaje
  - Que la información no se ha modificado



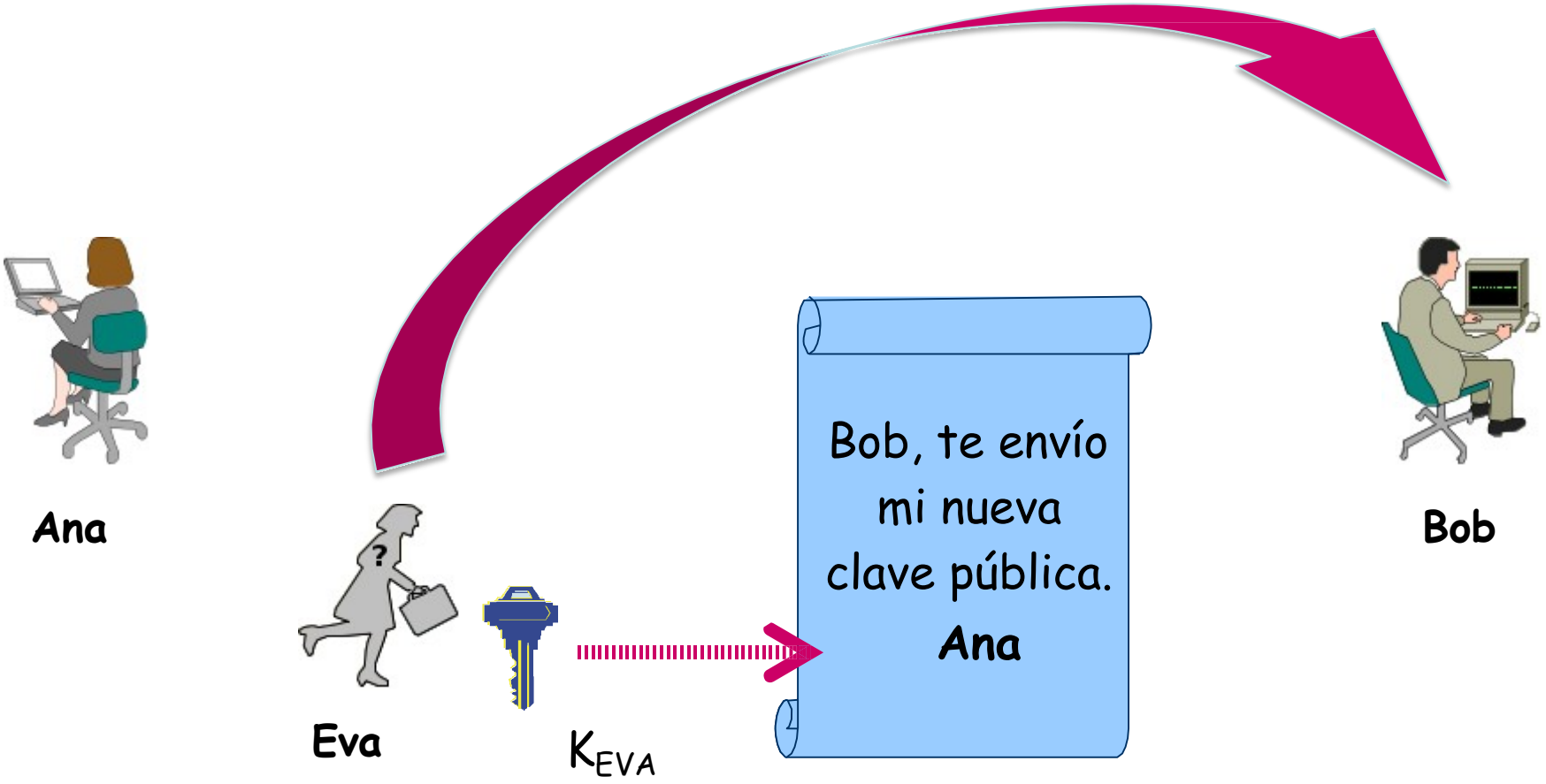
- Problema → Muy costoso computacionalmente

Firmas digitales con resumen del mensaje

- Para aligerar el coste computacional:
  - Se aplica una función *hash* y se obtiene un resumen del mensaje
  - El resumen se cifra con la clave privada del emisor:

    $H(m)=K_{pubA}(K_{privA}(H(m)))$
- Es el procedimiento más frecuente de firma digital

- ¿Cómo estar seguros de que una clave pública es la correcta?



Certificados digitales

- Sirven para resolver el problema de administrar las claves públicas y para que la identidad del dueño no pueda ser falsificada
  - La identidad del usuario es asegurada por un tercero: la **autoridad certificadora (AC)**
- Partes:
  - Una clave pública
  - La identidad de un implicado
  - La autoridad certificadora

Certificados digitales

Emisión del certificado

clave **privada** de AC

Clave pública
de Bob Pérez

K_{bob}

Clave pública
de Bob Pérez

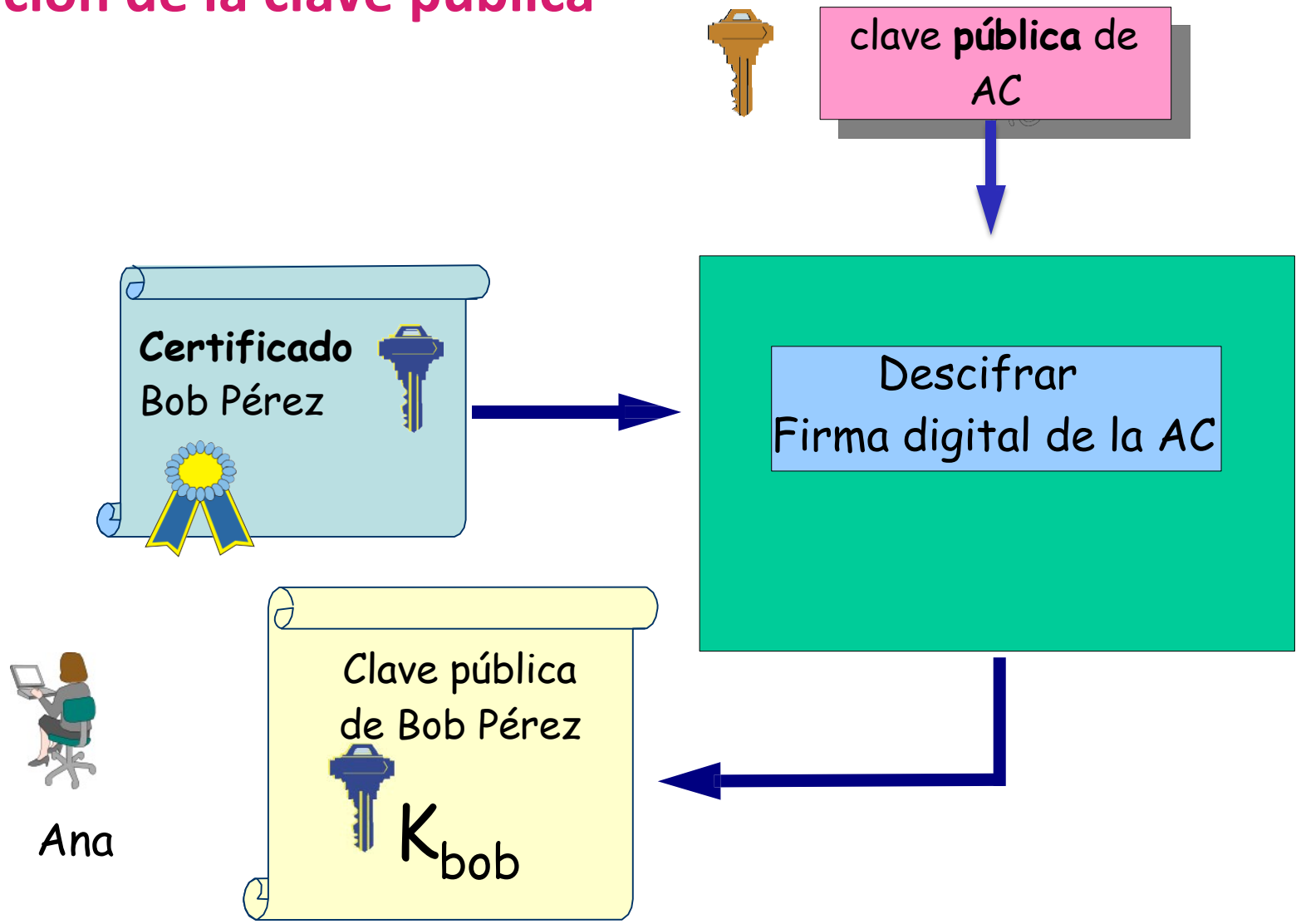
K_{bob}

Firma digital de la AC

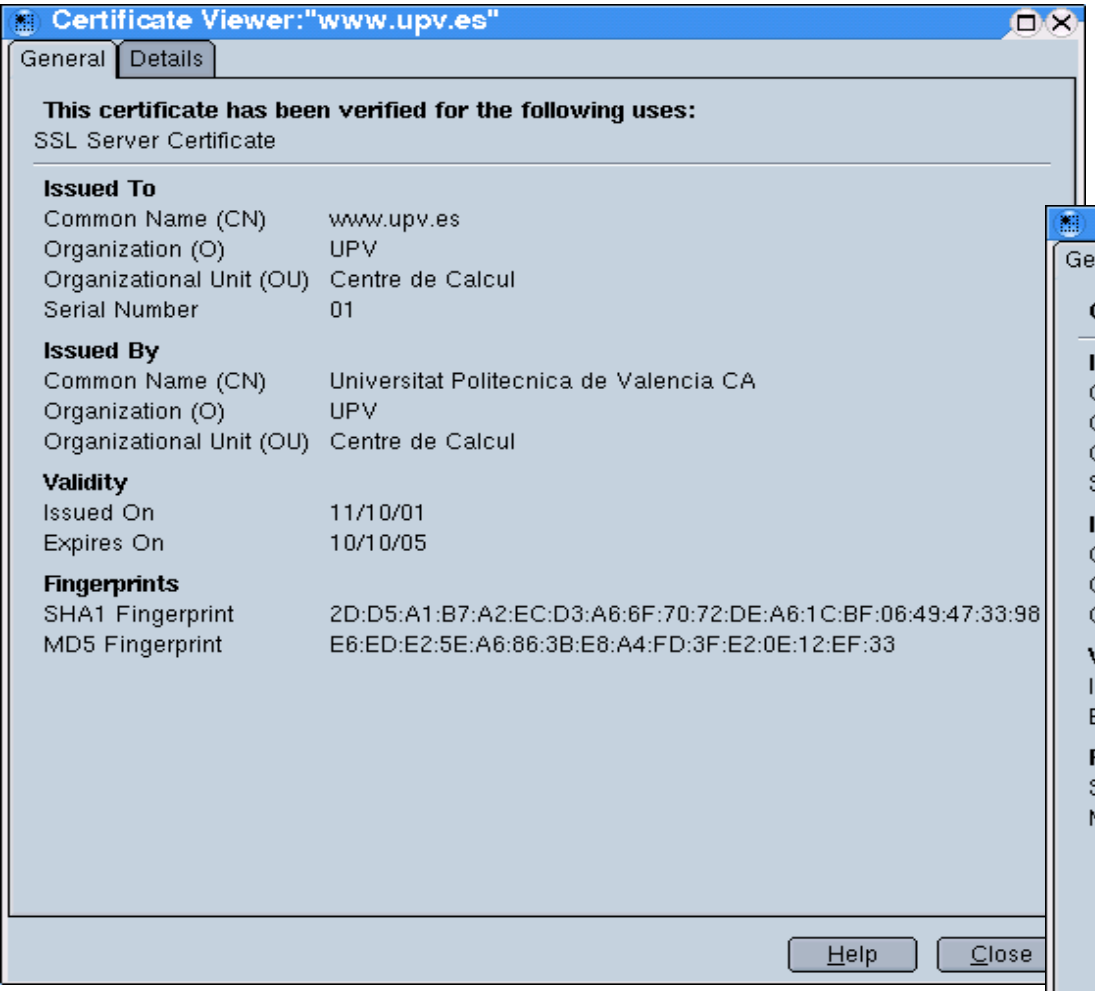
Bob

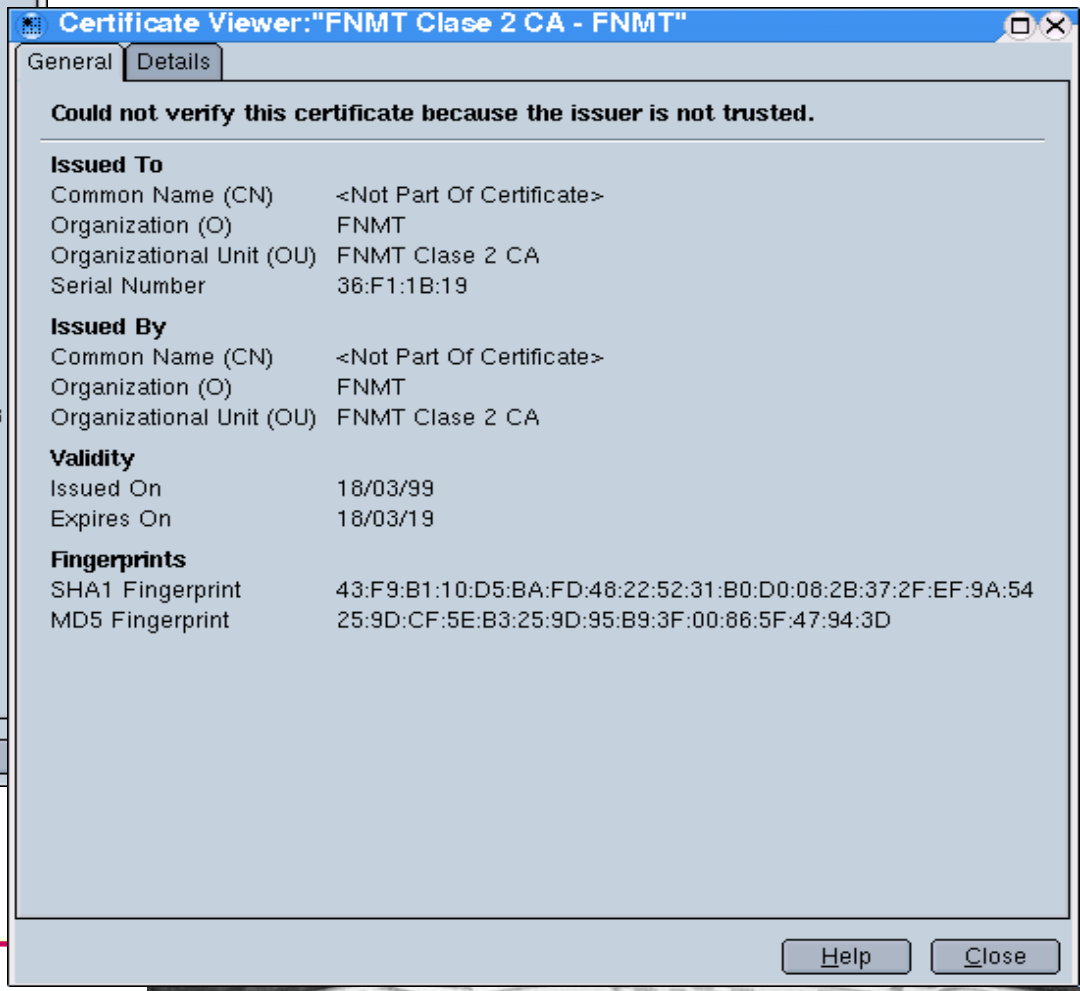
Certificado
Bob Pérez

Obtención de la clave pública



Certificados de un navegador





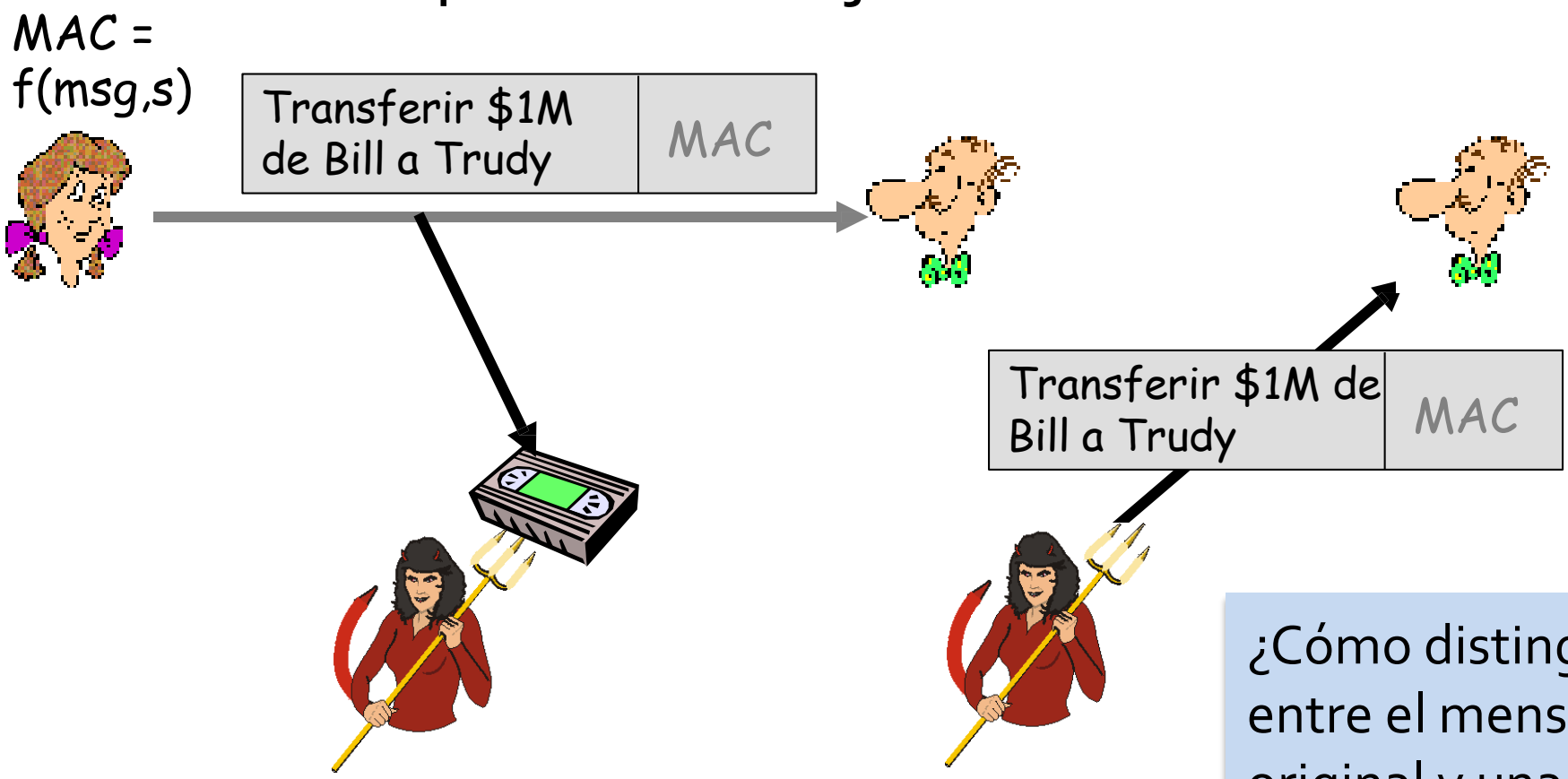
Índice

4. Autenticación del punto terminal

- ¿Cómo es posible saber que el otro extremo es quién afirma ser?
  - Ataques por reproducción
  - Ataques por interposición
- Reproducción:
  - Enviar copia de mensajes válidos anteriores
    - Solución: **números distintivos**
- Interposición
  - Suplantación de personalidad
    - Solución: **certificados digitales**

Autenticación del punto terminal

- Ataque por reproducción
  - Enviar copia de mensajes válidos anteriores

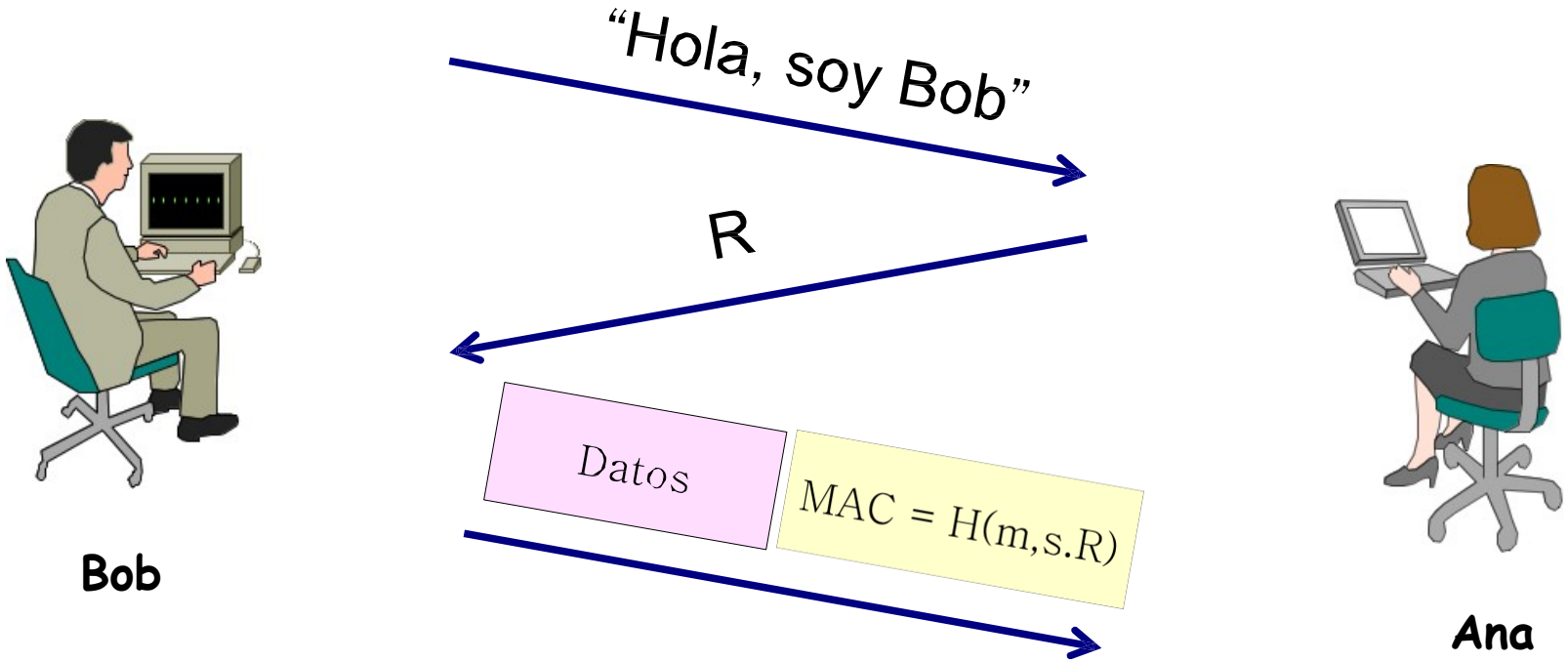


¿Cómo distinguir entre el mensaje original y una copia?

- Solución:
  - Números distintivos

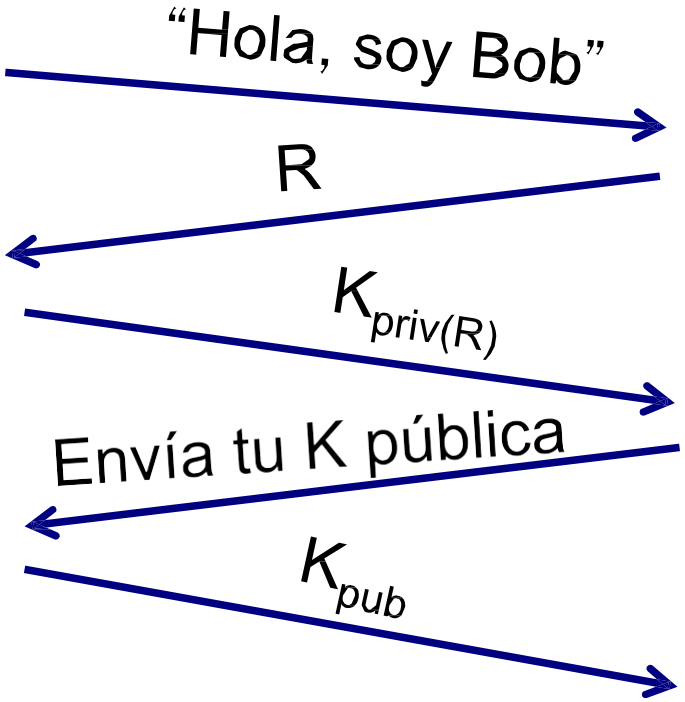
Números distintivos

Autenticación por MAC con números distintivos



- R (número distintivo) se genera aleatoriamente para cada conexión.
- El MAC se genera como *Hash* del mensaje **m**, la clave secreta **s** y el nº distintivo **R.**

Autenticación por clave pública



Bob

Ana

Ana calcula

$K_{pubB}(K_{privB}(R)) = R$

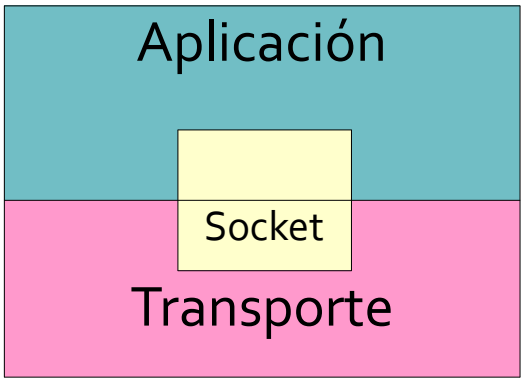
Y sabe que solamente Bob puede tener la clave privada, que ha encriptado R

$K_{pubB}(K_{privB}(R)) = R$

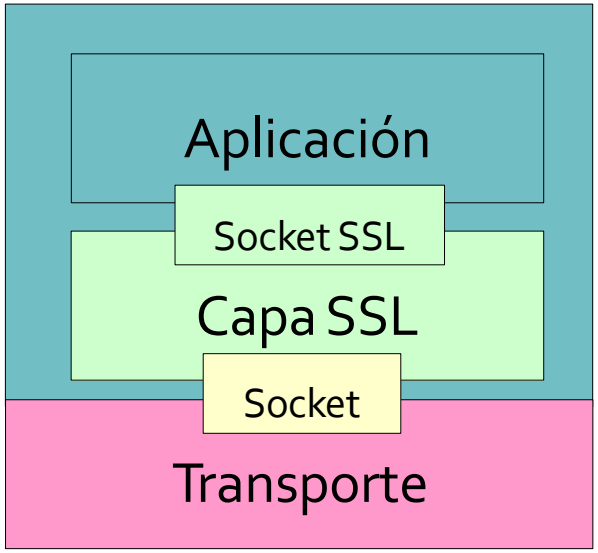
Índice

- ***Secure Sockets Layer* (SSL)**
  - Protocolo de seguridad ampliamente utilizado
    - Soportado por la mayoría de los navegadores, servidores web y lectores de correo
    - Empleado en la mayoría de los sitios de comercio electrónico
  - Diseñado originalmente por Netscape (1993)
  - Versión actual 3.0
  - Muy parecido a TLS (*Transport Layer Security*), RFC 5246, estandarizada por IETF *(Internet Engineering Task Force)*

SSL y TCP/IP



API de TCP



TCP mejorado con SSL

- Agrega confidencialidad, integridad y autenticación (de servidor y cliente) a TCP
  - Con clases y librerías para trabajar en C y en Java de forma similar al API de los *sockets*
  - Se definen puertos estándar diferentes de los habituales
    - Ejemplos: HTTP: 443 en lugar del 80, IMAP: 993 en vez del 143

SSL simplificado - Etapas

- Fase de acuerdo
  - Autenticación de Cliente (opcional) y Servidor mediante certificados
  - Acuerdo acerca de los algoritmos de clave simétrica, de clave pública y código MAC
  - Deducción de las claves: a partir de un secreto compartido (clave maestra) generan un conjunto de claves de sesión
- Transferencia de datos
  - Divididos en registros
- Cierre de la conexión
  - De forma segura

SSL – Fase de acuerdo

- Fase de acuerdo
  - Establecimiento de la conexión TCP
  - El cliente envía su número distintivo (R) y sus cifrados permitidos.
  - El servidor envía su número distintivo (R') y selecciona de entre ellos:
    - Un cifrado simétrico (para cifrar el tráfico) (**confidencialidad**)
    - Uno de clave pública (para cifrar la clave común del cifrado simétrico)
    - Un algoritmo MAC (Message Authentication Code) (**integridad**)

SSL – Fase de acuerdo

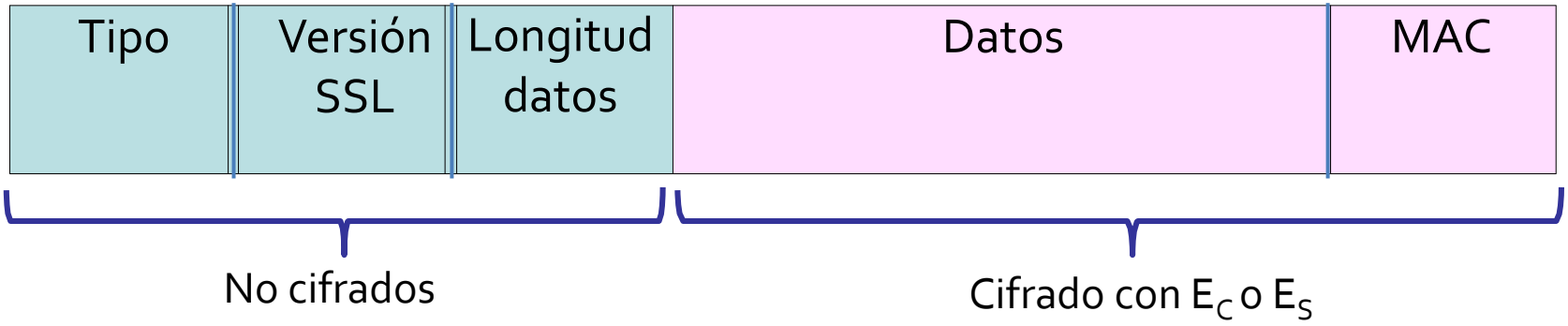
- El servidor envía su certificado (autenticación)
- El cliente verifica el certificado y extrae la clave pública del servidor
- El cliente genera una clave pre-maestra (PMS, Pre-Master Secret) y la envía cifrada (clave pública del servidor) al servidor.

SSL – Fase de acuerdo

- **Deducción de claves:**
  - A partir de la PMS y de los nº distintivos (R y R') se obtiene la clave maestra (MS) en cada extremo de forma independiente
  - La MS genera cuatro claves distintas:
    - Dos claves de cifrado E_c y E_s
    - Dos claves de MAC: M_c y M_s
    - En caso de CBC (Cipher Block Chaining), dos IV
- El cliente envía un MAC de todos los mensajes de acuerdo.
- El servidor envía un MAC de todos los mensajes de acuerdo.

SSL – Fase de Transferencia

- **Registros SSL**
  - SSL divide el flujo de datos en registros y añade un MAC por registro (comprobación de integridad)



  - El MAC se calcula sobre los 4 campos anteriores, la clave MAC ***M*** (M_c o M_s) y un número de secuencia que se incrementa en cada registro
  - El campo ***tipo*** *(fase de acuerdo, datos de aplicación o cierre)*
    - Permite cerrar la conexión de forma segura.
    - Aunque va sin cifrar, está cubierto por el MAC.