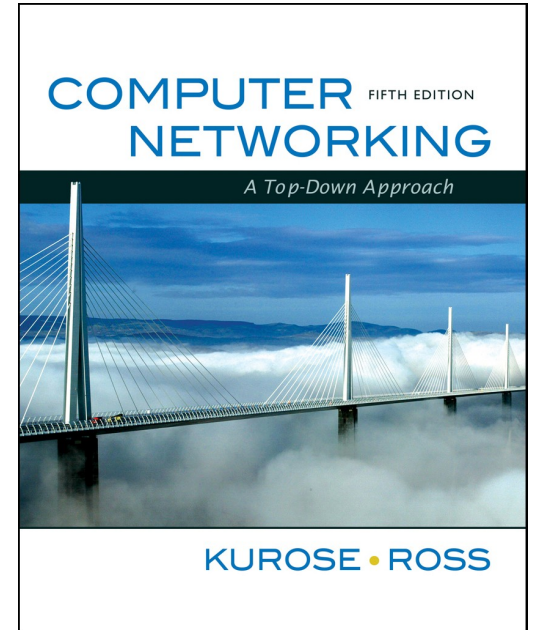


# Capítulo 8

## Seguridad en Redes

### Generalidades y Principios



*Basado en:*  
*Computer Networking: A Top Down Approach ,*  
*5<sup>th</sup> edition.*  
Jim Kurose, Keith Ross  
Addison-Wesley, April 2009.

# Capítulo 8 contenidos

8.1 ¿Qué es la seguridad en la red?

8.2 Principios de criptografía

8.3 Integridad de mensajes

8.4 Dando seguridad a e-mail

8.5 Conexiones TCP seguras: SSL

8.6 Seguridad en capa de Red: IPsec

8.7 Seguridad en redes locales inalámbricas

8.8 Cortafuegos y Sistemas de detección de intrusión (IDS)

# ¿Qué es la seguridad en la Red?

**Confidencialidad:** sólo el Tx y Rx deseado deberían “entender” el contenido del mensaje

- Tx encripta el mensaje
- Rx decripta el mensaje

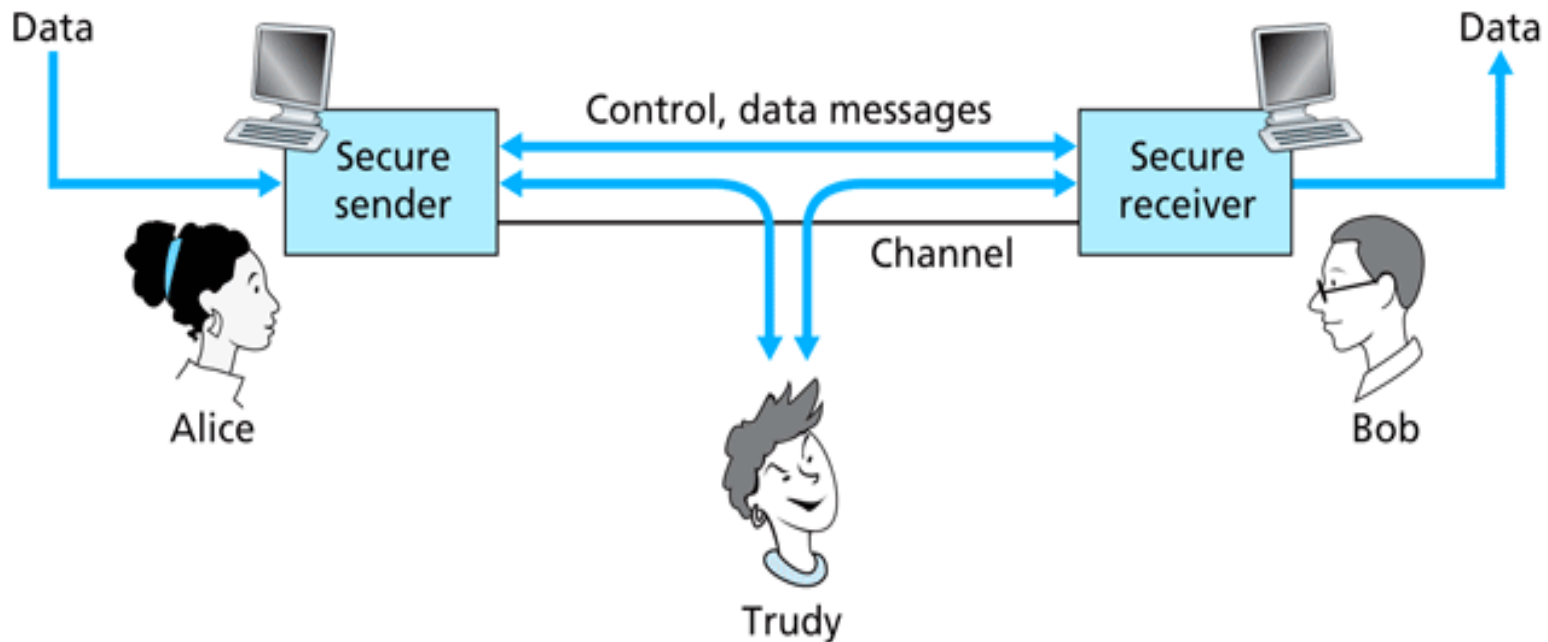
**Autenticación:** Tx y Rx requieren confirmar la identidad del otro.

**Integridad del Mensaje:** Tx y Rx desean asegurar que el mensaje no sea alterado sin ser detectado

**Acceso y disponibilidad:** servicio debe estar accesible y disponible a los usuarios.

# Amigos y enemigos: Alicia, Bob, Trudy

- Bob, Alice desean comunicarse con “seguridad”
- Trudy (intruso) podría interceptar, borrar, agregar mensajes



**Figure 8.1** ♦ Sender, receiver, and intruder (Alice, Bob, and Trudy)

# ¿Qué puede hacer un atacante?

## Mucho

- ▣ *espiar*: interceptar mensajes
- ▣ *Insertar* activamente mensajes en la conexión
- ▣ *Suplantación de Identidad*: puede fingir la dirección fuente del paquete (u otro campo)
- ▣ *secuestro*: capturar la conexión en curso removiendo el Tx o Rx poniéndose a sí mismo en su lugar
- ▣ *Denegación de servicio*: impedir que el servicio pueda ser usado por otros (e.g., sobrecargando el servicio o recursos usados por éste)

# ¿Quién puede ser Bob y Alice?

- ... dos personas!
- Navegador y servidor Web en comercio electrónico
- Cliente/ servidor en bancos en línea
- Servidores DNS
- Routers intercambiando tablas de ruteo
- Otros ejemplos?

# Capítulo 8 contenidos

8.1 ¿Qué es la seguridad en la red?

8.2 Principios de criptografía

8.3 Integridad de mensajes

8.4 Dando seguridad a e-mail

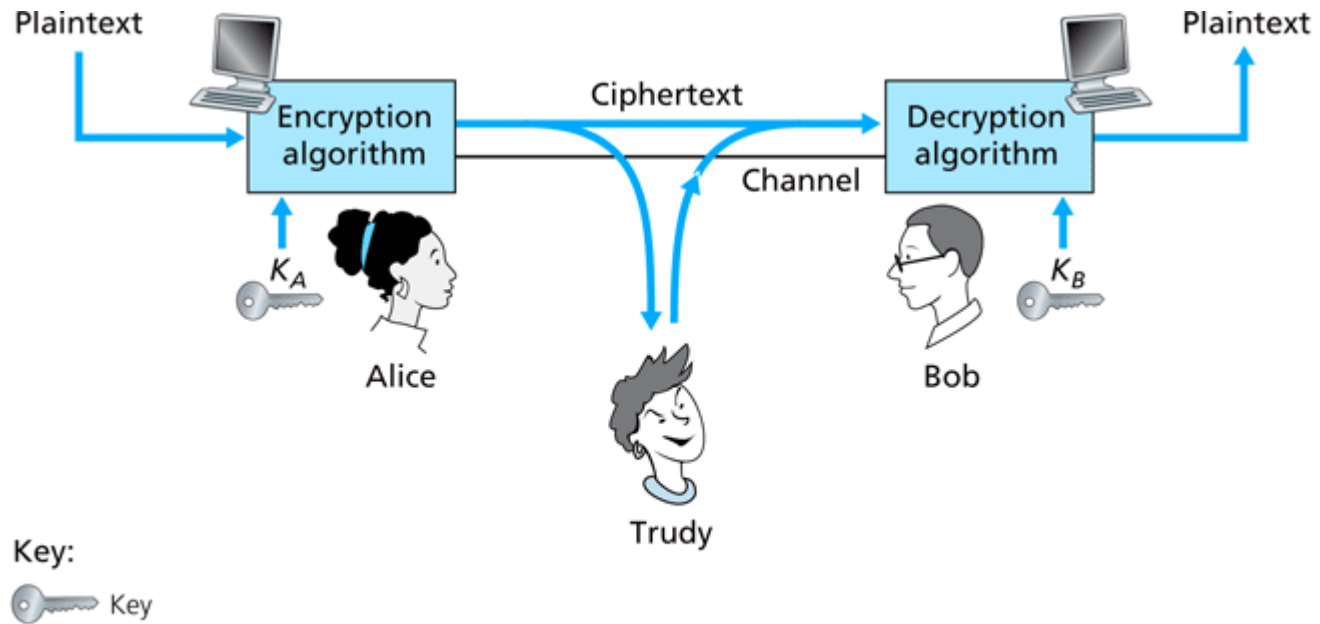
8.5 Conexiones TCP seguras: SSL

8.6 Seguridad en capa de Red: IPsec

8.7 Seguridad en redes locales inalámbricas

8.8 Cortafuegos y Sistemas de detección de intrusión (IDS)

# Términos en Criptografía



**Figure 8.2** ♦ Cryptographic components

$m$  mensaje legible

$K_A(m)$  texto cifrado, encriptado con clave  $K_A$

$m = K_B(K_A(m))$

Cuando  $K_A = K_B$  hablamos de claves simétricas, en otro caso son asimétricas.



# Esquema de encriptación simple

**Cifrado por sustitución:** se sustituye una cosa por otra

- Cifrado mono-alfabético: sustituye una letra por otra

**Texto legible:** abcdefghijklmnopqrstuvwxyz

**Texto cifrado:** mnbvcxzasdfghjklpoiuytrewq

E.g.:      **Texto legible:** bob. i love you. alice  
              **Texto cifrado:** nkn. s gktc wky. mgsbc

Clave: El mapa de 26 letras al otro de 26 letras.

# Encriptación Poli-alfabética

- Usando  $n$  cifrados mono-alfabéticos:  
 $M_1, M_2, \dots, M_n$
- Usando patrón cíclico:
  - e.g.,  $n=4$ ,  $M_1, M_3, M_4, M_3, M_2$ ;  $M_1, M_3, M_4, M_3, M_2$ ;
- Para cada símbolo en texto plano nuevo, usar el patrón mono-alfabético subsecuente en el ciclo.
  - usm: u desde  $M_1$ , s de  $M_3$ , m de  $M_4$
- Clave: los  $n$  cifrados y el patrón cíclico.

# Ruptura de un esquema de encriptación

- **Ataque basado sólo en texto cifrado:** atacante tiene texto cifrado para analizar.
- Dos estrategias:
  - Buscar sobre todas las claves: debe ser capaz de distinguir texto resultante de incoherencias
  - Análisis estadístico
- **Ataque basado en texto legible conocido:** atacante tiene algo de texto legible correspondiente a texto cifrado.
  - eg, en cifrado monoalfabético atacante determina pares
- **Ataque por texto legible seleccionado:** el atacante se las ingenia para conseguir que Tx envíe un texto conocido que él verá en su forma encriptada.

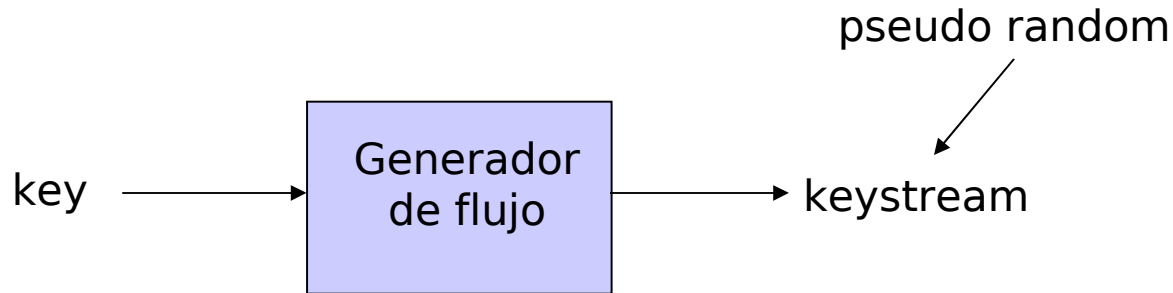
# Tipos de criptografías

- Criptografía normalmente usa un algoritmo conocido por todos y sólo las claves son secretas.
- Criptografía de clave simétrica usa una clave
- Criptografía de clave pública usa dos claves
- Funciones hash
  - No usa claves
  - Nada es secreto, cómo funciona?

# Dos tipos de cifrado simétrico

- ▣ Cifrado de flujo de símbolos básicos
  - ▣ Encriptación de un bit a la vez.
- ▣ Cifrado de bloques
  - ▣ Se divide el mensaje legible en bloques de igual tamaño.
  - ▣ Encriptar cada bloque como una unidad.

# Cifrado de flujo



- Combina cada bit del flujo de clave con el texto legible y obtiene el texto cifrado.
- $m(i) = i^{\text{mo}}$  bit del texto legible
- $ks(i) = i^{\text{mo}}$  bit del flujo keystream
- $c(i) = i^{\text{mo}}$  bit del texto cifrado
- $c(i) = ks(i) \oplus m(i)$  ( $\oplus = \text{or-exclusivo}$ )
- $m(i) = ks(i) \oplus c(i)$

# Ej: RC4 usa cifrado de flujo

- RC4 es un cifrador de flujo popular.
  - Ha sido analizado y considerado bueno.
  - Key puede ser de 1 a 256 bytes
  - Es usado en WEP de 802.11 (Wired Equivalent Privacy)
  - Puede ser usado en SSL (Secure Sockets Layer)

# Cifradores de Bloques

- El mensaje es procesado en bloques de  $k$  bits (e.g., bloques de 64 bit).
- Se usa mapeo 1-a-1 para mapear  $k$  bit del texto a  $k$  bits cifrados.

## Ejemplo con $k=3$ :

<u>input</u>	<u>output</u>	<u>input</u>	<u>output</u>
000	110	100	011
001	111	101	010
010	101	110	000
011	100	111	001

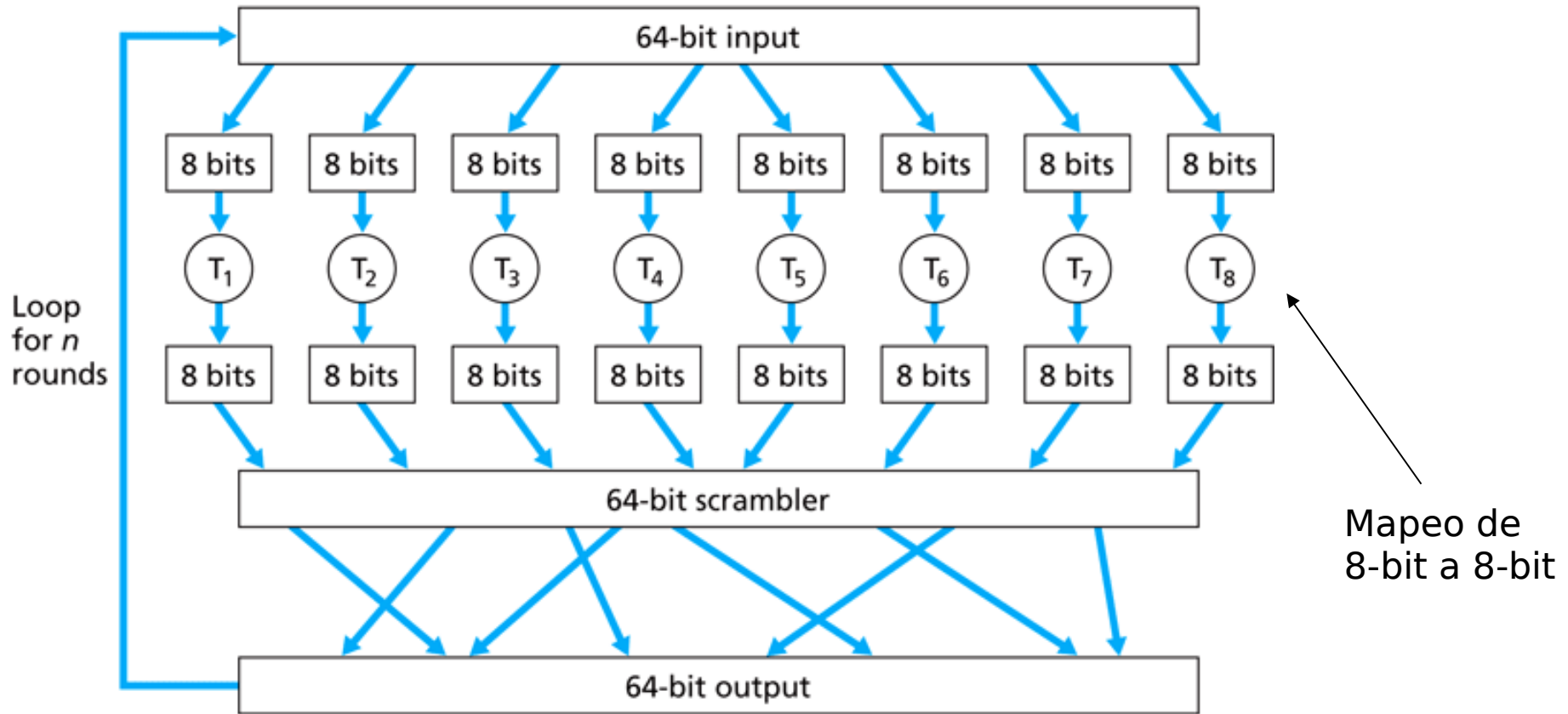
¿Cuál es el texto cifrado para 010110001111 ?



# Cifrado de Bloques

- ¿Cuántos mapeos existen para  $K=3$ ?
  - ¿Cuántos bloques de 3 bits?
  - ¿Cuántos mapeos podemos hacer? (son las permutaciones son posibles)
  - Respuesta: 40.320 ; No muchas
- En general,  $2^k!$  mapeos; muchos para  $k=64$
- Problema:
  - Si lo hacemos por tablas, se requiere tablas de  $2^{64}$  entradas, cada entrada con 64 bits
- Tabla muy grande, en su lugar conviene usar una función que simule una tabla de permutación aleatoria.

# Ej: Función prototipo



**Figure 8.5** ♦ An example of a block cipher

# ¿Por qué iteramos en la función?

- Si iteramos sólo una vez, un cambio de un bit en la entrada sólo altera 8 bits de salida.
- En 2<sup>nd</sup> ciclos, los 8 bits afectados se dispersan afectando a otros.
- ¿Cuántos ciclos?
  - Depende del tamaño del bloque,
  - Se requieren más ciclos (menos eficiente) conforme  $n$  aumenta.

# Cifrado de grandes mensajes

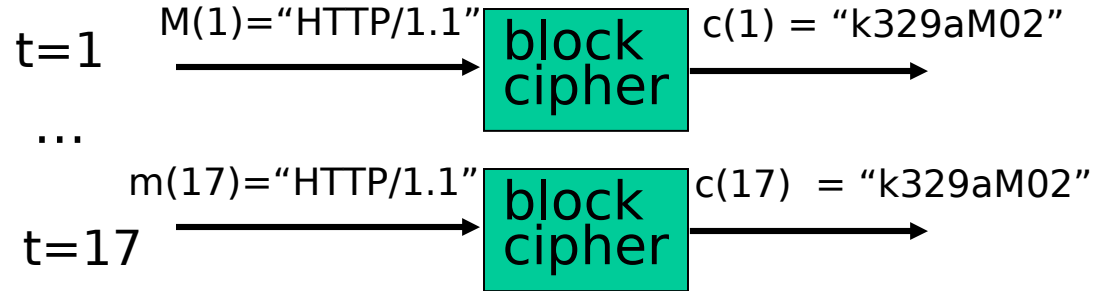
- Otra idea:
  - Generar un número aleatorio de 64 bits  $r(i)$  para cada bloque de texto  $m(i)$
  - Calcular  $c(i) = K_s( m(i) \oplus r(i) )$
  - Transmitir ambos:  $c(i), r(i), i=1,2,\dots$
  - En receptor:  $m(i) = K_s(c(i)) \oplus r(i)$
  - Problema: ineficiente, se debe enviar  $c(i)$  y  $r(i)$

# Cifrado de Bloques en Cadena (CBC)

- ▣ CBC genera sus propios números aleatorios
  - ▣ Hace depender el cifrado del bloque actual del resultado del bloque previo
  - ▣  $c(i) = K_s( m(i) \oplus c(i-1) )$
  - ▣  $m(i) = K_s(c(i)) \oplus c(i-1)$
- ▣ ¿Cómo se encripta el primer bloque?
  - ▣ Usamos un vector de inicialización (IV): bloque aleatorio =  $c(0)$
  - ▣ El Vector de Inicialización (IV) puede no ser secreto.
- ▣ Cambio de IV por cada mensaje (o sesión)
  - ▣ Garantiza que si el mensaje es enviado varias veces el cifrado será diferente.

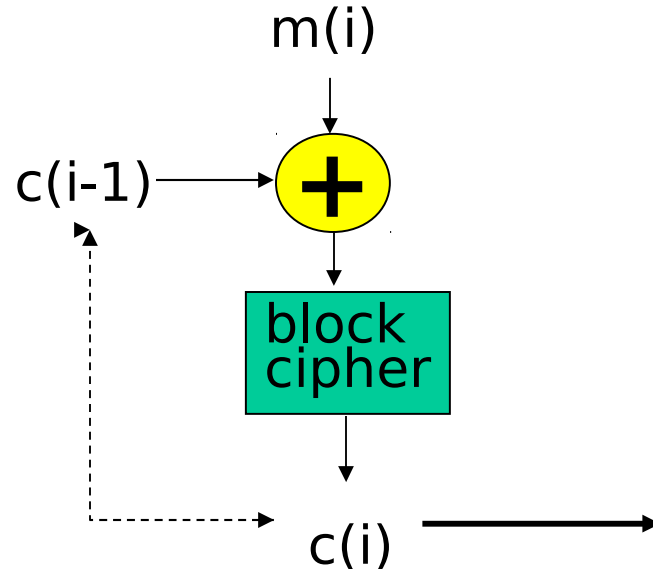
# Ventaja de Cifrado de Bloques en Cadena

- Sólo *Cifrado de Bloques*: Si la entrada se repite obtengo igual salida.



- *Cifrado de Bloques en Cadena*: XOR  $i^{\text{mo}}$  bloque de entrada,  $m(i)$ , con cifrado del texto previo,  $c(i-1)$

- $c(0)$  es transmitido al receptor
- Analizar caso "HTTP/1.1" previo.



# Caso DES: Cifrado de clave simétrica

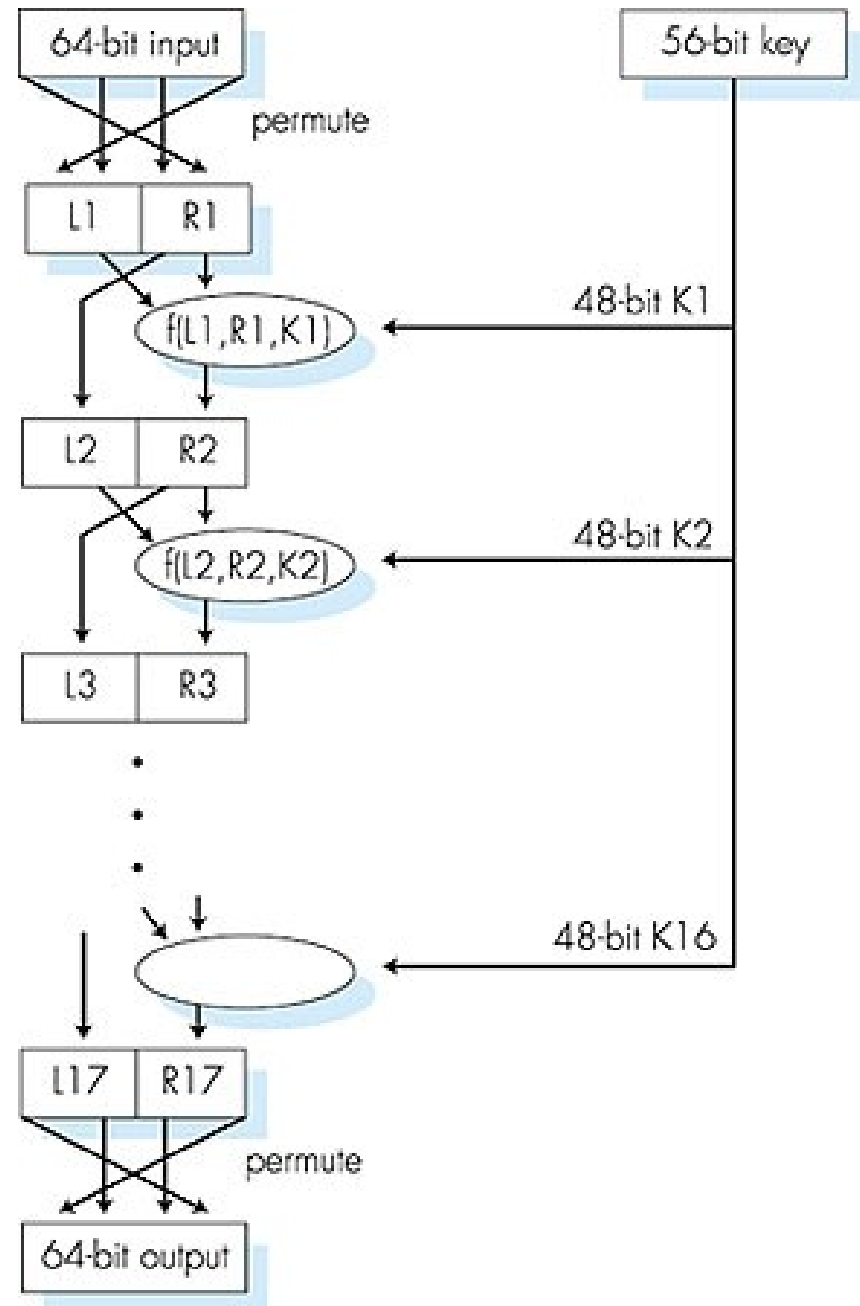
## DES: Data Encryption Standard

- Estándar americano de 1993
- Clave de 56-bit, 64-bit plaintext input
- Usa Cifrado de bloques de 64 bits en cadena
- ¿Cuán seguro es DES?
  - Desafío DES: mensaje encriptado con clave de 56-bit fue descifrada en menos que un día a fuerza bruta.
- DES más seguro::
  - 3DES: encripta 3 veces con diferentes claves (en realidad encripta, descifra, vuelve a encriptar)

# DES

## Operación de DES

- Permutación inicial
- 16 ciclos idénticos pero usando una clave de 48 bits diferente
- Permutación final





# AES: Advanced Encryption Standard

- Nuevo estándar de clave simétrica (Nov. 2001), reemplaza a DES
- Bloques de datos son de 128 bit
- Usa claves de 128, 192, o 256 bit
- Si descriptado a fuerza bruta toma 1 segundo en DES, tomará 149 trillones de años en AES

# Cifrado de Clave Pública: el otro tipo de cifrado

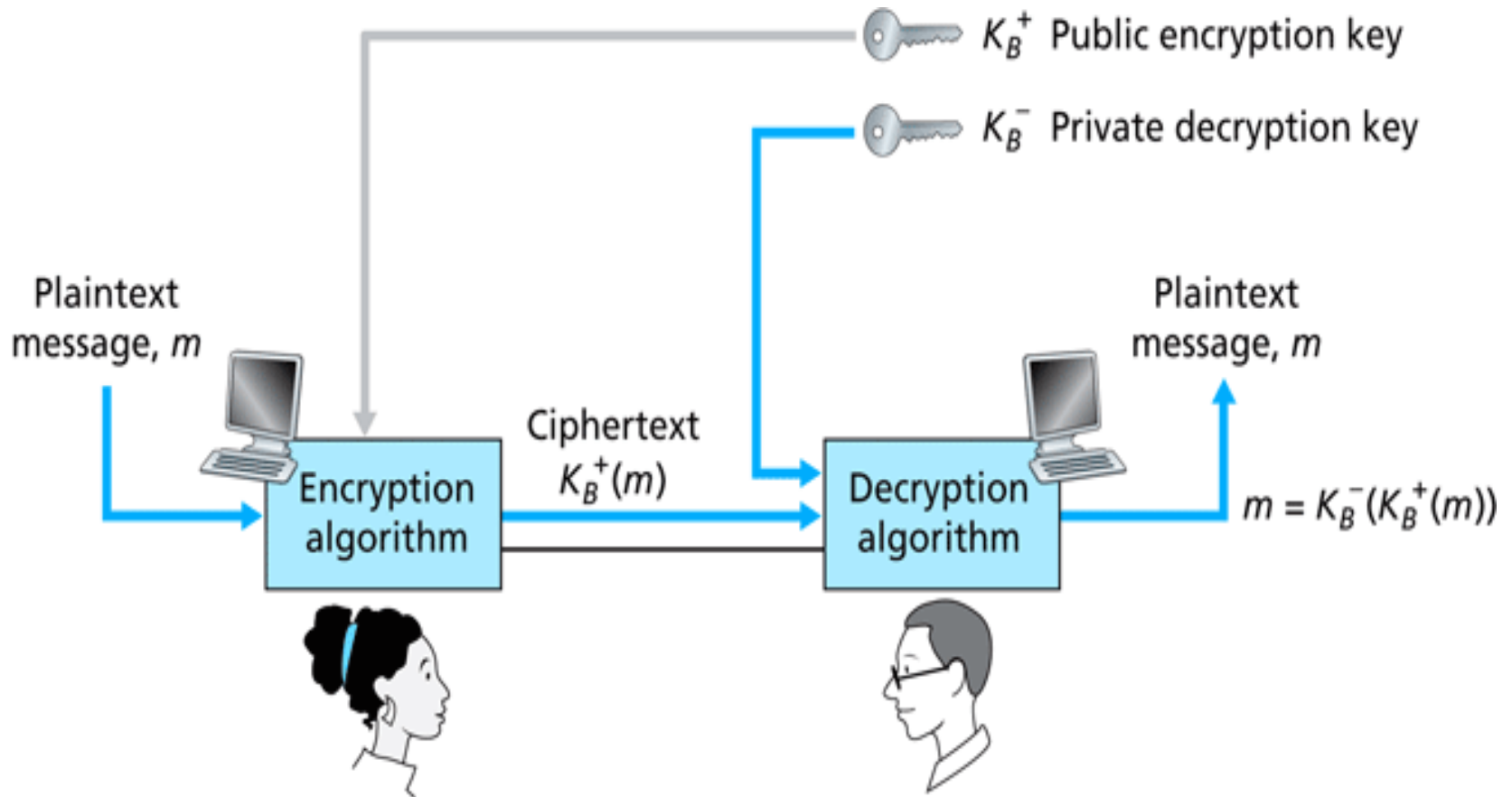
## Cifrado simétrico

- Requiere Tx y Rx conocer secreto compartido
- ¿Cómo ponerse de acuerdo si no nos conocemos?

- Cifrado de clave pública
- Tx y Rx **no** comparten clave secreta
- *Clave pública de cifrado es conocida por todos*
- *Clave privada de descifrado sólo es conocida por Rx.*



# Cifrado de Clave Pública



**Figure 8.6** ♦ Public key cryptography

# Algoritmos de Clave Pública

Requerimientos:

- ① Se requiere  $K_B^+ ( )$  y  $K_B^- ( )$  tal que

$$K_B^- (K_B^+ (m)) = m$$

- ② Conocida la clave pública  $K_B^+$  debería ser imposible obtener la clave privada  $K_B^- ( )$

**RSA:** Algoritmo de **R**ivest, **S**hamir, **A**delson

# Pre-requisito: Aritmética modular

## □ Propiedades:

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

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

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

## □ Así

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

## □ Ejemplo: $x=14$ , $n=10$ , $d=2$

$$x^d = 14^2 = 196 \Rightarrow x^d \bmod 10 = 6$$

$$(x \bmod n)^d \bmod n = 4^2 \bmod 10 = 6$$

# RSA: generalidades

- Un mensaje es un patrón de bits
- Un patrón de bits puede ser unívocamente interpretado por un número entero.
- Cifrar un mensaje es como cifrar un número.
- Ejemplo:
  - $m = 10010001$  . Equivale al decimal 145.
- Para cifrar  $m$ , ciframos el número correspondiente.

# RSA: Debemos crear el par clave pública y privada

1. Elegir 2 números primos grandes  $p$ ,  $q$ . (e.g., 1024 bits cada uno)
2. Obtenga  $n = pq$ ,  $z = (p-1)(q-1)$
3. Elija  $e$  (con  $e < n$ ) tal que no tenga factores comunes con  $z$  ( $e$ ,  $z$  son “primos relativos”).
4. Elija  $d$  tal que  $ed-1$  es exactamente divisible por  $z$ .
5. Así la **clave pública es  $(n,e)$  y la clave privada es  $(n,d)$** . (en otras palabras:  $ed \bmod z = 1$  ).

En linux podemos crear la clave privada con:

```
$ openssl genrsa -out mykey.pem
```

```
$ openssl rsa -in mykey.pem -pubout
```

# RSA: Cifrado, descifrado

0. Dado  $(n,e)$  y  $(n,d)$  obtenidos como antes

1. Para encriptar mensaje  $m (<n)$ , obtener

$$c = m^e \bmod n$$

2. Para descifrar lo recibido,  $c$ , calcular

$$m = c^d \bmod n$$

□ La razón de que esto funcione es:

$$m = \underbrace{(m^e \bmod n)}_c^d \bmod n$$



# ¿Por qué RSA funciona?

- Se debe demostrar que  $c^d \bmod n = m$   
donde  $c = m^e \bmod n$
- Propiedad (no demostrada aquí): para cualquier  $x, y$  tenemos  
 $x^y \bmod n = x^{(y \bmod z)} \bmod n$ ; donde  $n = pq$  y  $z = (p-1)(q-1)$
- Así,  
$$\begin{aligned}c^d \bmod n &= (m^e \bmod n)^d \bmod n \\ &= m^{ed} \bmod n \\ &= m^{(ed \bmod z)} \bmod n \\ &= m^1 \bmod n \\ &= m\end{aligned}$$

# RSA ejemplo:

Bob elige  $p=5$ ,  $q=7$ . entonces  $n=35$ ,  $z=24$ .

$e=5$  (así  $e$ ,  $z$  son primos relativos).

$d=29$  (así  $ed-1 = 144$  es divisible exactamente por  $z$ ).

Cifremos un mensaje de 8-bit

Cifrado: 

<u>mensaje</u>	<u>m</u>	<u>m<sup>e</sup></u>	<u>c = m<sup>e</sup> mod n</u>
00001100	12	24832	17

Descifrado: 

<u>c</u>	<u>c<sup>d</sup></u>	<u>m = c<sup>d</sup> mod n</u>
17	481968572106750915091411825223071697	12

# RSA: Otra propiedad importante

Será útil luego:

$$K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$$

Usa clave  
pública  
primero

Usa clave  
privada  
primero

*El Resultado es el  
mismo*

¿Por qué  $K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$  ?

Se obtiene de la aritmética modular:

$$\begin{aligned}(m^e \bmod n)^d \bmod n &= m^{ed} \bmod n \\ &= m^{de} \bmod n \\ &= (m^d \bmod n)^e \bmod n\end{aligned}$$

# ¿Por qué RSA es seguro?

- Supongamos que conocemos la clave pública de alguien  $(n, e)$ . ¿Será difícil determinar  $d$ ?
- Se requiere encontrar factores de  $n$  sin conocer los factores  $p$  y  $q$
- Un hecho: factorizar grandes números es difícil.

# Claves de sesión

- Exponenciación es computacionalmente intensivo
- DES es al menos 100 veces más rápido que RSA

## Clave de sesión, $K_S$

- Bob y Alice usan RSA para intercambiar una clave simétrica  $K_S$
- Luego con  $K_S$ , usan cifrado de clave simétrica; mucho más rápido.

# Capítulo 8 contenidos

8.1 ¿Qué es la seguridad en la red?

8.2 Principios de criptografía

8.3 Integridad de mensajes

8.4 Dando seguridad a e-mail

8.5 Conexiones TCP seguras: SSL

8.6 Seguridad en capa de Red: IPsec

8.7 Seguridad en redes locales inalámbricas

8.8 Cortafuegos y Sistemas de detección de intrusión (IDS)