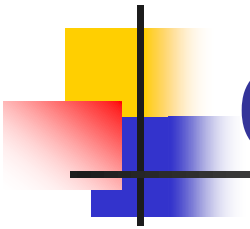
Bloque I
Criptografía

Cifrado en bloque

Seguridad en Redes de Comunicaciones
María Dolores Cano Baños



Contenidos

Introducción

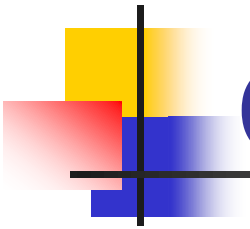
- Criptografía (cryptos=oculto + grafía=escritura): arte de escribir con clave secreta o de un modo enigmático ⇨ conjunto de técnicas que tratan sobre la protección de la información
  - Criptoanálisis
  - Criptología
- Los sistemas criptográficos se clasifican en función de:
  - Tipo de operación: sustitución, transposición.
  - El número de claves: simétrico, asimétrico.
  - Modo de procesar el texto plano: bloque, flujo.

Introducción

- Notación:
  - Texto plano X
  - Texto cifrado Y
  - Algoritmo de cifrado E

  $Y = E_K(X)$
  - Clave K
  - Algoritmo de descifrado D ⇨ $X = D_K(Y)$

- Ej: cifrado del César
- Ej: técnica de transposición

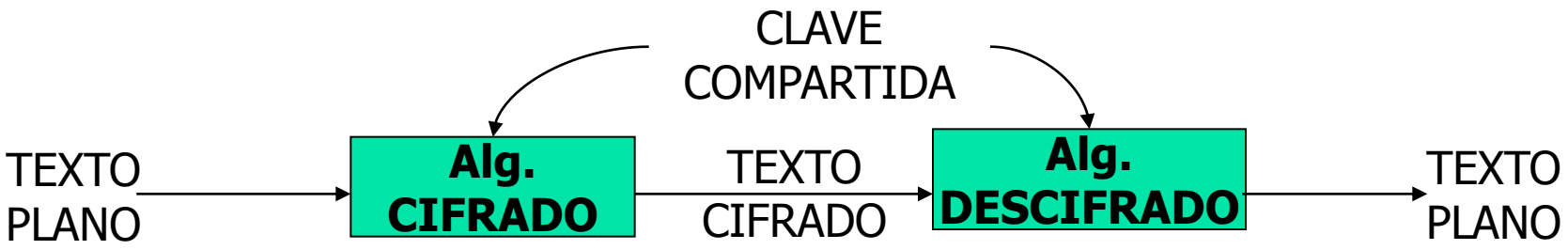


Contenidos

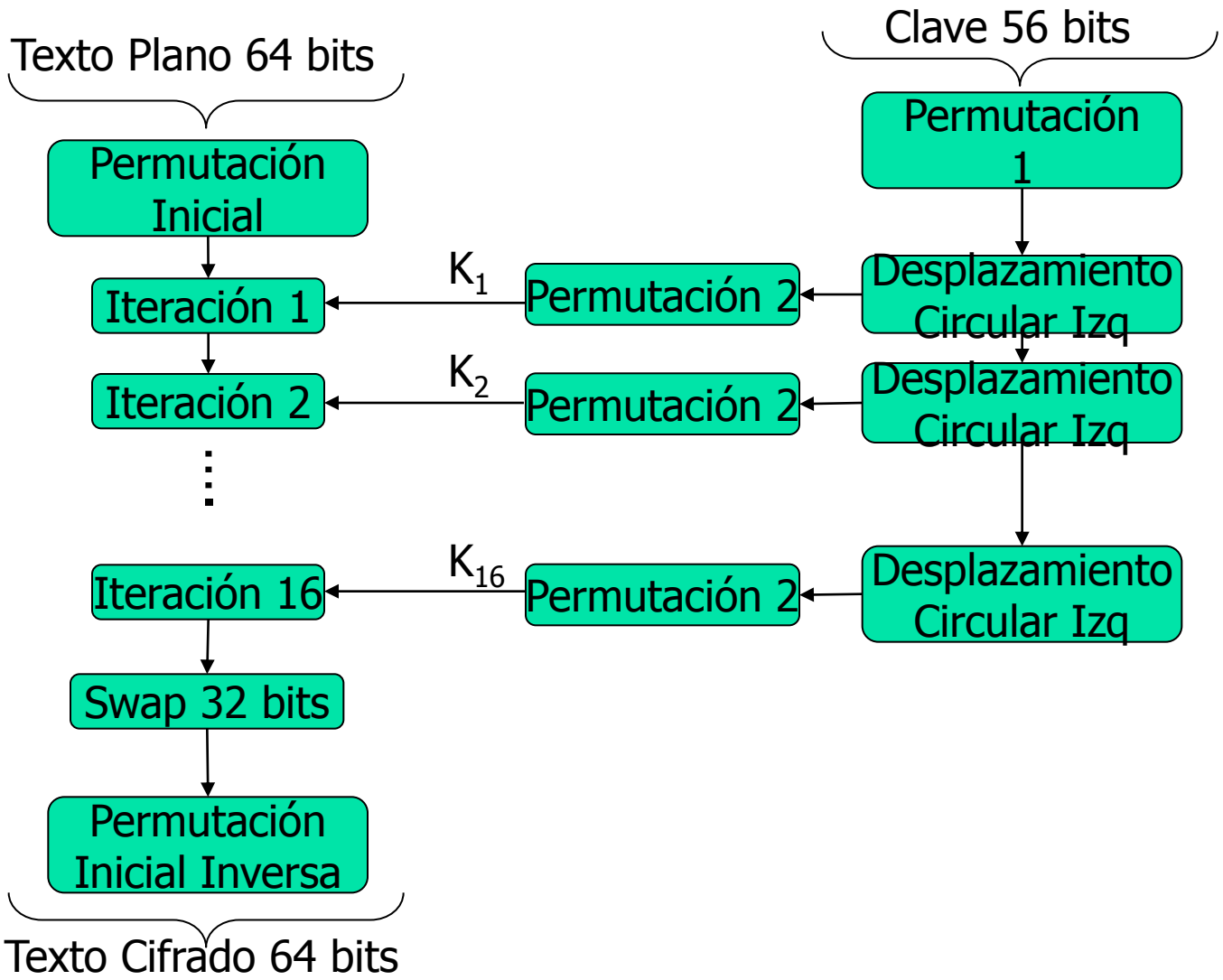
- 2.1 Introducción ✓
- 2.2 Cifrado en bloque
  - 2.2.1 Algoritmos simétricos
    - 2.2.1.1 DES
      - Triple DES
    - 2.2.1.2 AES
  - 2.2.2 Algoritmos asimétricos
    - 2.2.2.1 RSA
    - 2.2.2.2 Diffie-Hellman
    - 2.2.2.3 Curvas elípticas

Algoritmos simétricos

- Clave: valor independiente del texto plano
- Seguridad de criptografía clásica:
  - El algoritmo de cifrado
  - El secreto de la clave
- Ej: DES, triple DES, AES, …



DES: Esquema de cifrado



Tablas de Permutación DES

TABLE 2.5 Permutation Tables for DES

(a) Initial Permutation (IP)

Output bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
From input bit	58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
Output bit	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
From input bit	62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
Output bit	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
From input bit	57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
Output bit	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
From input bit	61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

(b) Inverse Initial Permutation (IP^{-1})

Output bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
From input bit	40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
Output bit	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
From input bit	38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
Output bit	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
From input bit	36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
Output bit	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
From input bit	34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

Tablas de Permutación DES

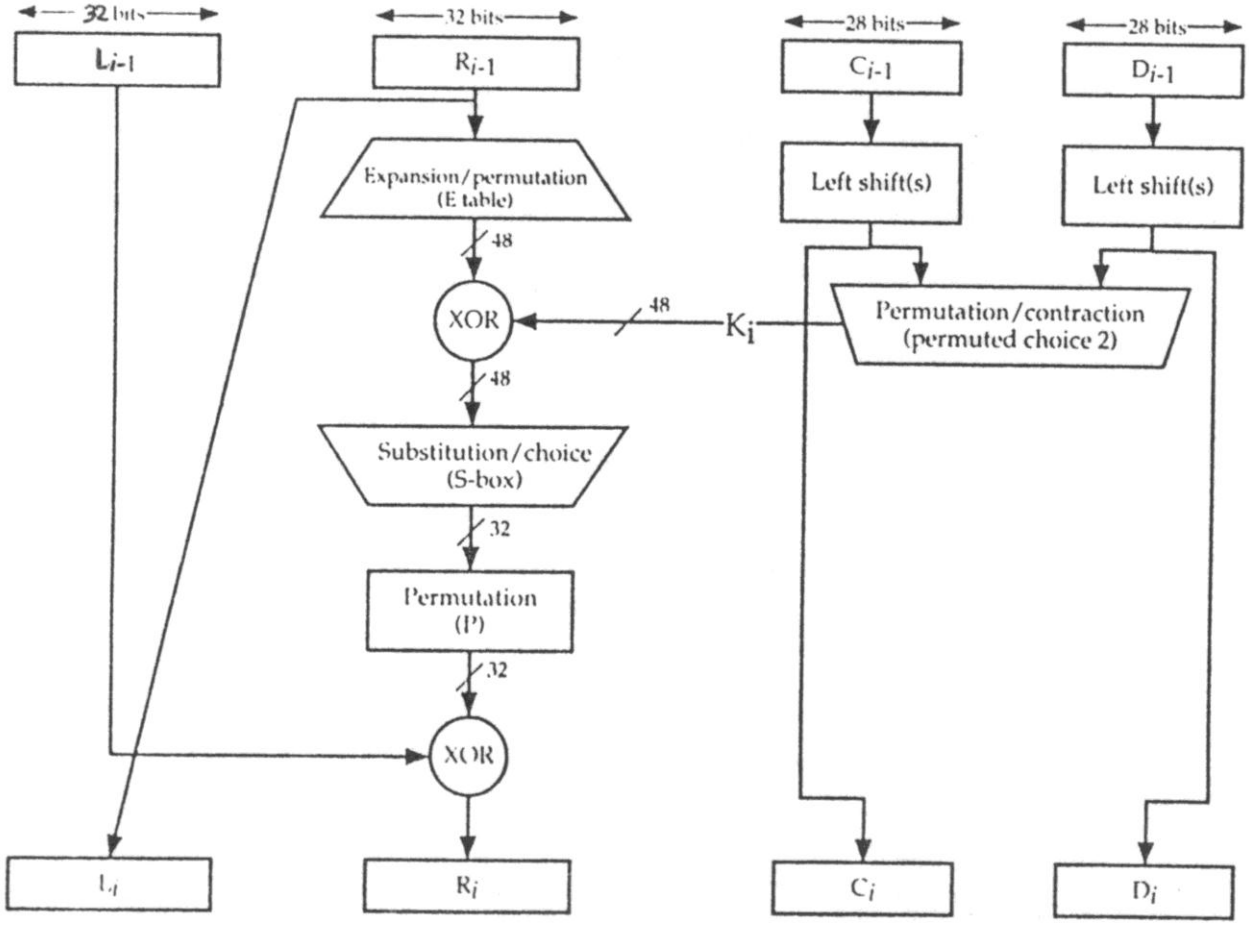
(c) Expansion Permutation (E)

Output bit	1	2	3	4	5	6	7	8	9	10	11	12
From input bit	32	1	2	3	4	5	4	5	6	7	8	9
Output bit	13	14	15	16	17	18	19	20	21	22	23	24
From input bit	8	9	10	11	12	13	12	13	14	15	16	17
Output bit	25	26	27	28	29	30	31	32	33	34	35	36
From input bit	16	17	18	19	20	21	20	21	22	23	24	25
Output bit	37	38	39	40	41	42	43	44	45	46	47	48
From input bit	24	25	26	27	28	29	28	29	30	31	32	1

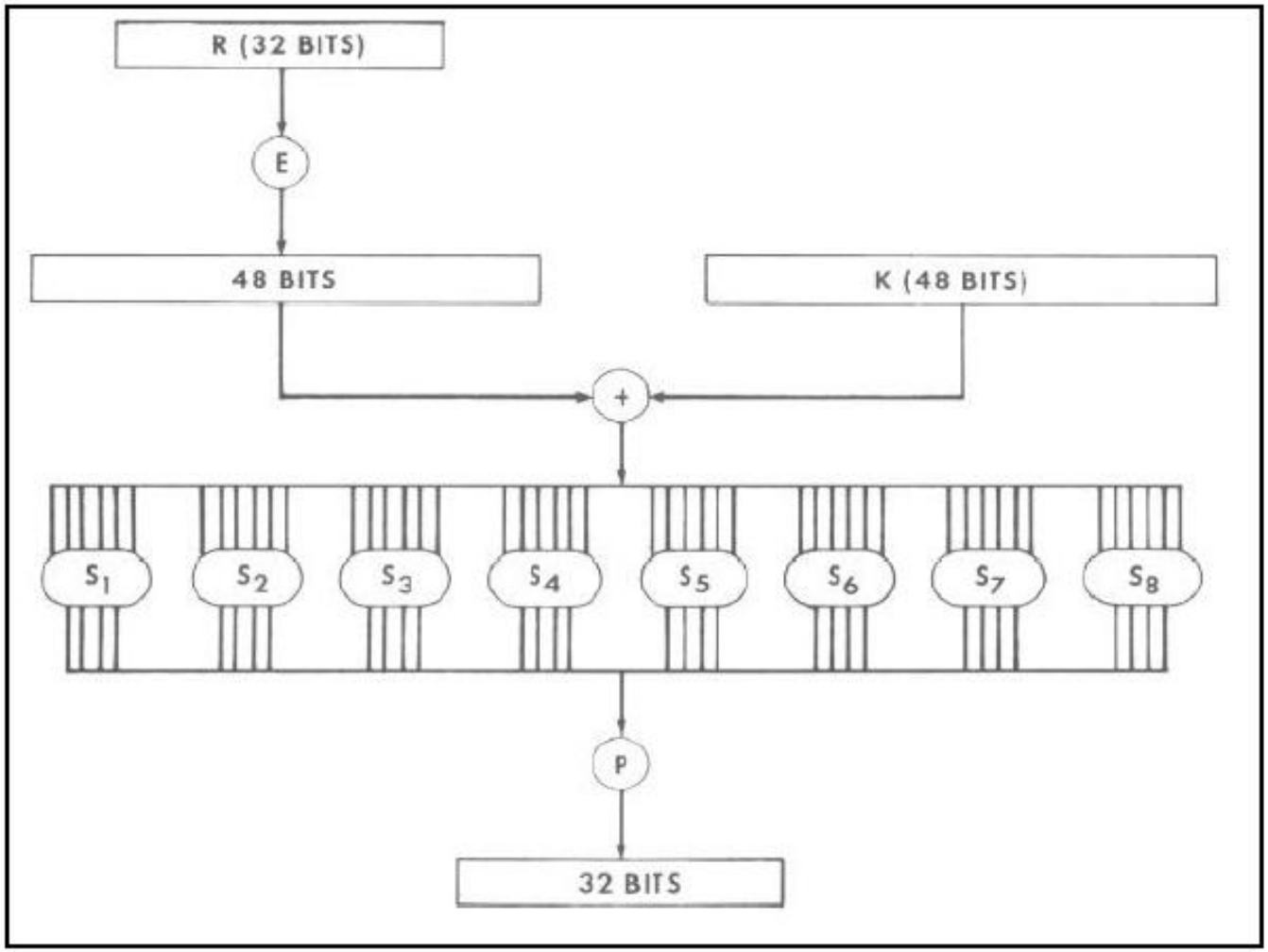
(d) Permutation Function (P)

Output bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
From input bit	16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
Output bit	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
From input bit	2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

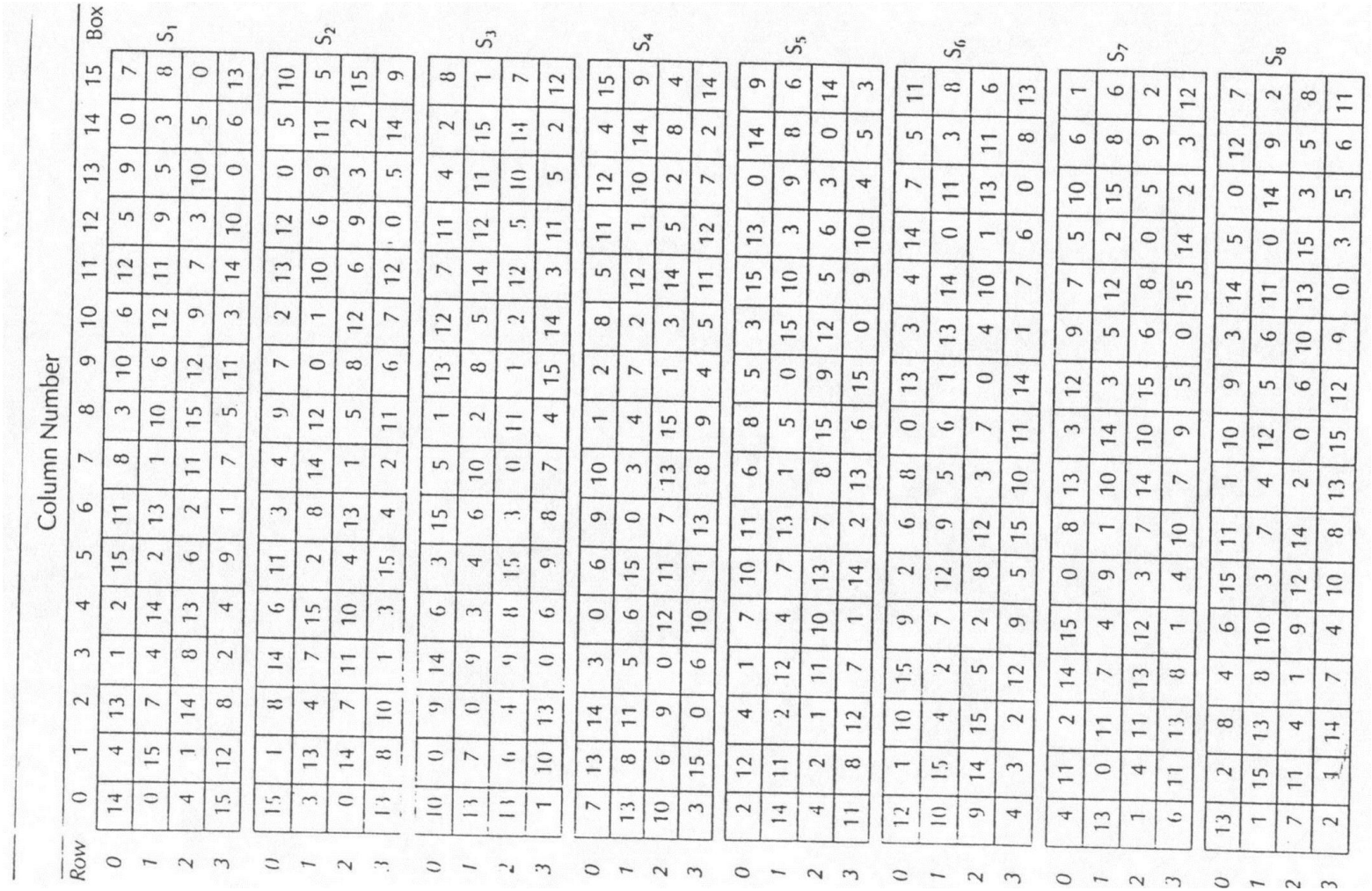
Una única iteración en DES



Cálculo de f(R, K)



Definición de cajas-S en DES



Row	Column Number: 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Box
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S_1
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	S_2
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	S_3
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	S_4
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	S_5
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	S_6
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	S_7
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	S_8
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

Tablas para cálculo de claves en DES

(a) Permuted Choice One (PC-1)

Output bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14
From input bit	57	49	41	33	25	17	9	1	58	50	42	34	26	18
Output bit	15	16	17	18	19	20	21	22	23	24	25	26	27	28
From input bit	10	2	59	51	43	35	27	19	11	3	60	52	44	36
Output bit	29	30	31	32	33	34	35	36	37	38	39	40	41	42
From input bit	63	55	47	39	31	23	15	7	62	54	46	38	30	22
Output bit	43	44	45	46	47	48	49	50	51	52	53	54	55	56
From input bit	14	6	61	53	45	37	29	21	13	5	28	20	12	4

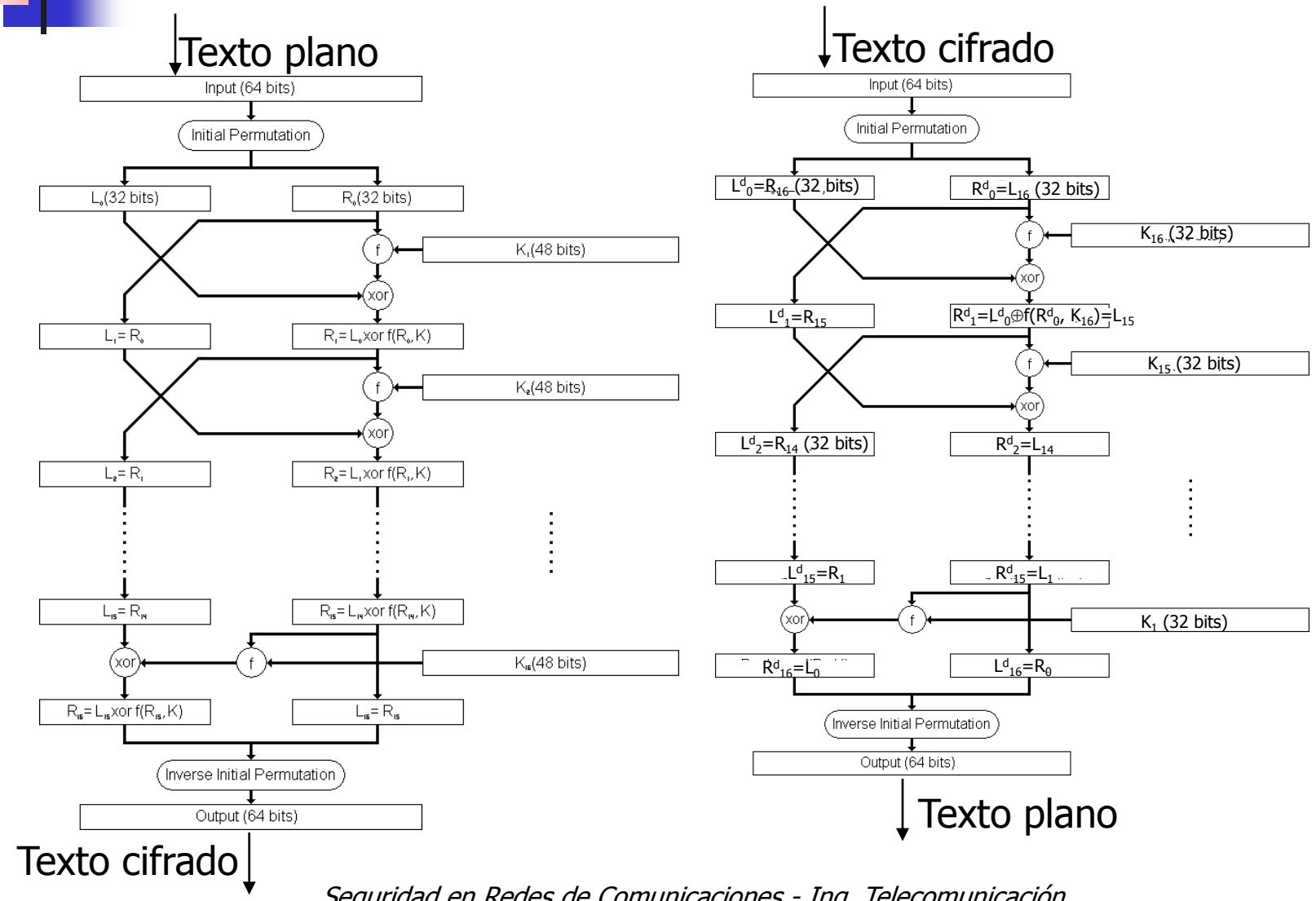
(b) Permuted Choice Two (PC-2)

Output bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
From input bit	14	17	11	24	1	5	3	28	15	6	21	10	23	19	12	4
Output bit	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
From input bit	26	8	16	7	27	20	13	2	41	52	31	37	47	55	30	40
Output bit	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
From input bit	51	45	33	48	44	49	39	56	34	53	46	42	50	36	29	32

(c) Schedule of Left Shifts

Iteration number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

DES cifrado y descifrado



Modos de funcionamiento DES

- Modo ECB (Electronic CodeBook)
  - Cada bloque de 64 bits se cifra independientemente
  - Misma clave
- Modo CBC (Cipher Block Chaining)
  - La entrada al algoritmo es una XOR del texto plano y el texto cifrado anterior
  - Misma clave
- Modo CFB (Cipher FeedBack)
  - Texto cifrado anterior entrada del algoritmo
  - La salida XOR con texto plano
  - Misma clave
- Modo OFB (Output FeedBack)
  - Entrada al algoritmo es la anterior salida DES
  - Misma clave

Modos de funcionamiento DES

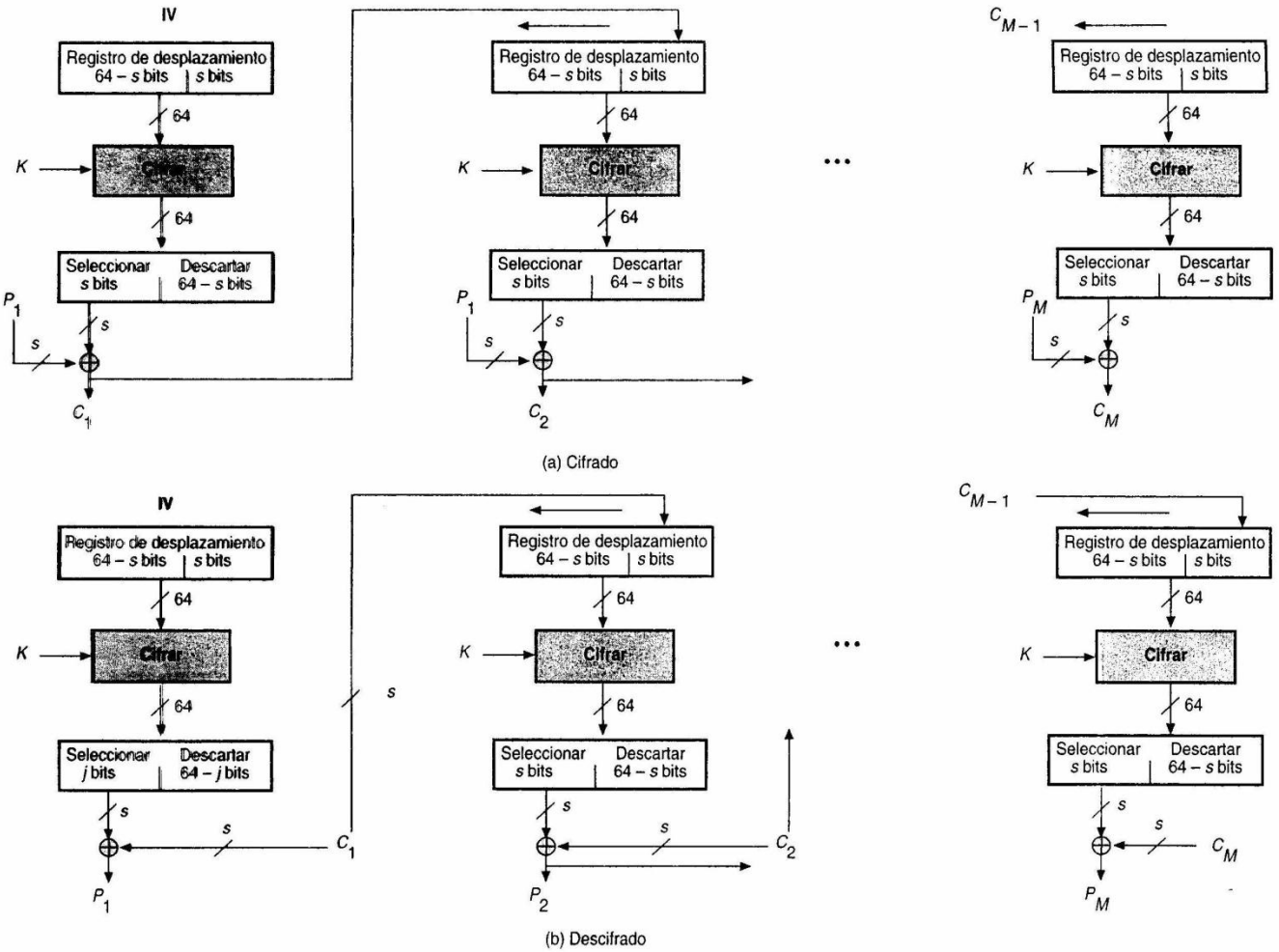
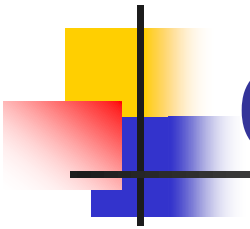


Figura 2.8 Modo CFB de *s* bits

Triple DES

- $Y = E_{K1} [D_{K2} [E_{K1} (X)]]$
- $X = D_{K1} [E_{K2} [D_{K1} (Y)]]$

- Clave: 128 bits
- Bloque: 64 bits



Contenidos

2.1 Introducción ✓

2.2 Cifrado en bloque

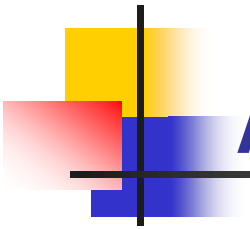
- 2.2.1 Algoritmos simétricos
  - 2.2.1.1 DES ✓
    - Triple DES ✓
  - 2.2.1.2 AES
- 2.2.2 Algoritmos asimétricos
  - 2.2.2.1 RSA
  - 2.2.2.2 Diffie-Hellman
  - 2.2.2.3 Curvas elípticas

AES

- Un byte un elemento de campo finito en GF(2^8)
  - $b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0 \Rightarrow b_7x^7+ b_6x^6+ b_5x^5+ b_4x^4+ b_3x^3+ b_2x^2 +b_1x^1+ b_0x^0$
- Operación suma entre dos polinomios en GF se define como una XOR de los coeficientes con mismo grado de ambos polinomios.
- Operación de multiplicación en GF(2^8) se corresponde con la multiplicación de polinomios con módulo un polinomio irreducible de grado 8
  - AES: $m(x)=x^8+x^4+x^3+x+1$

AES

- Para cualquier polinomio distinto de cero b(x) de grado menor que 8, la inversa multiplicativa de b(x), $b^{-1}(x)$, se puede hallar:
  - Algoritmo de Euclides b(x)a(x)+m(x)c(x)=1
  - Se cumple que a(x)b(x) mod m(x) = 1
  - $b^{-1}(x)$=a(x) mod m(x)



AES

- Octubre 2000, algoritmo de Rijndael
- Algoritmo de cifrado por bloques: 128, 192 o 256 bits
- Clave: 128, 192 o 256 bits
- Operaciones a nivel de byte

- AES aplica un número determinado de rondas a un valor intermedio denominado *estado*
  - Estado ≡ Matriz rectangular ($4xN_b$)
- La clave se almacena en una matriz
  - Clave ≡ Matriz rectangular ($4xN_k$)

AES

- El bloque de texto plano se pasa a una matriz del mismo tamaño que la matriz de estado ($4xN_b$)

ej: $N_b=4$ $\quad X=\begin{pmatrix} b0 & b4 & b8 & b12 \\ b1 & b5 & b9 & b13 \\ b2 & b6 & b10 & b14 \\ b3 & b7 & b11 & b15 \end{pmatrix}$

- **Algoritmo AES**: Siendo X el bloque que queremos cifrar y S la matriz de estado:
  - Calcular K_0, K_1, K_2, …K_n
  - $S = X \oplus K_0$
  - Para i=1 hasta n
    - Aplicar ronda i-ésima del algoritmo con la subclave K_i

AES: Cálculo de subclaves

- Las subclaves se obtienen aplicando dos funciones: expansión y selección
  - La función de expansión permite obtener una secuencia de $(n+1)*4*N_b$ bytes
  - La selección toma consecutivamente bloques del mismo tamaño que la matriz de estado y los asigna a cada K_i

- Sea K(i) un vector de bytes de tamaño $4*N_k$, y sea W(i) un vector de $N_b*(n+1)$ registros de 4 bytes, la función de expansión tiene dos versiones según el valor de N_k:

AES: Cálculo de subclaves

a. Si $N_k \leq 6$
   1. Para i=0 hasta N_{k-1}
   2. W(i)=[K(4*i), K(4*i+1), K(4*i+2), K(4*i+3)]
   3. Para i=N_k hasta $N_b(n+1)$
   4. tmp=W(i-1)
   5. Si i mod N_k =0
   6. tmp=Sub(Rot(tmp))$\oplus$R((i/N_k)-1)
   7. W(i) = W(i-N_k) $\oplus$ tmp

AES: Cálculo de subclaves

a. Si $N_k > 6$
   1. Para i=0 hasta N_{k-1}
   2. W(i)=[K(4*i), K(4*i+1), K(4*i+2), K(4*i+3)]
   3. Para i=N_k hasta $N_b(n+1)$
   4. tmp=W(i-1)
   5. Si i mod N_k = 0
   6. tmp=Sub(Rot(tmp))$\oplus$R((i/N_k)-1)
   7. Si i mod N_k = 4
   8. tmp=Sub(tmp)
   9. W(i) = W(i-N_k) $\oplus$ tmp

AES: Cálculo de subclaves

Función Sub()

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

AES: Cálculo de subclaves

- Función Rot: desplazamiento circular a izquierda de una posición
- Función $R_c((i/N_k)-1)$: Cada $R((i/N_k)-1)$ es el elemento de $GF(2^8)$ correspondiente al valor $x^{((i/Nk)-1)}$
  - R(0)=01000000
  - R(1)=02000000
  - R(2)=04000000
  - R(3)=08000000
  - R(4)=10000000
  - R(5)=20000000
  - R(6)=40000000
  - R(7)=80000000
  - R(8)=1B000000
  - R(9)=36000000
  - R(10)=6C000000
  - R(11)=D8000000
  - R(12)=AB000000
  - R(13)=4D000000
  - R(14)=9A000000

AES: Rondas

- El número de rondas depende del tamaño del bloque y del tamaño de la clave

Valor de n	$N_b = 4$	$N_b = 6$	$N_b = 8$
$N_k=4$	10	12	14
$N_k=6$	12	12	14
$N_k=8$	14	14	14

- Siendo S la matriz de estado y K_i la subclave correspondiente a la ronda i-ésima, cada ronda:
  - 1. S=Sub(S)
  - 2. S=Desplazar_fila(S)
  - 3. S=Mezclar_columnas (S)
  - 4. $S=K_i \oplus S$

NOTA: La última ronda es igual a la primera pero eliminando el paso 3

AES: Rondas

- Función Sub(): sustitución no lineal que se aplica a cada byte de la matriz de estado S mediante una caja-S de 8x8 bits invertible. (Ver diapositiva 25)

- Función Desplazar_fila(): desplazamiento cíclico a izquierda de las filas de la matriz de estado

N_b	f_1	f_2	f_3
4	1	2	3
6	1	2	3
8	1	3	4

  - La fila 0 queda inalterada

AES: Rondas

- Función Mezclar_columnas(): multiplicación de matrices donde cada columna es un vector de estado que se considera un polinomio cuyos coeficientes pertenecen a GF(2^8).

Matriz Multiplicación

$$S=\begin{pmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{pmatrix} \qquad M=\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix}$$

$s_0 = s_0 * 02 \oplus s_1 * 03 \oplus s_2 * 01 \oplus s_3 * 01$

$s_1 = s_0 * 01 \oplus s_1 * 02 \oplus s_2 * 03 \oplus s_3 * 01$

$s_2 = s_0 * 01 \oplus s_1 * 01 \oplus s_2 * 02 \oplus s_3 * 03$

....

Estas multiplicaciones son en GF(2^8) ⇨ por simplicidad usad tablas E y L

AES: Rondas

L Table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		00	19	01	32	02	1A	C6	4B	C7	1B	68	33	EE	DF	03
1	64	04	E0	0E	34	8D	81	EF	4C	71	08	C8	F8	69	1C	C1
2	7D	C2	1D	B5	F9	B9	27	6A	4D	E4	A6	72	9A	C9	09	78
3	65	2F	8A	05	21	0F	E1	24	12	F0	82	45	35	93	DA	8E
4	96	8F	DB	BD	36	D0	CE	94	13	5C	D2	F1	40	46	83	38
5	66	DD	FD	30	BF	06	8B	62	B3	25	E2	98	22	88	91	10
6	7E	6E	48	C3	A3	B6	1E	42	3A	6B	28	54	FA	85	3D	BA
7	2B	79	0A	15	9B	9F	5E	CA	4E	D4	AC	E5	F3	73	A7	57
8	AF	58	A8	50	F4	EA	D6	74	4F	AE	E9	D5	E7	E6	AD	E8
9	2C	D7	75	7A	EB	16	0B	F5	59	CB	5F	B0	9C	A9	51	A0
A	7F	0C	F6	6F	17	C4	49	EC	D8	43	1F	2D	A4	76	7B	B7
B	CC	BB	3E	5A	FB	60	B1	86	3B	52	A1	6C	AA	55	29	9D
C	97	B2	87	90	61	BE	DC	FC	BC	95	CF	CD	37	3F	5B	D1
D	53	39	84	3C	41	A2	6D	47	14	2A	9E	5D	56	F2	D3	AB
E	44	11	92	D9	23	20	2E	89	B4	7C	B8	26	77	99	E3	A5
F	67	4A	ED	DE	C5	31	FE	18	0D	63	8C	80	C0	F7	70	07

E Table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	01	03	05	0F	11	33	55	FF	1A	2E	72	96	A1	F8	13	35
1	5F	E1	38	48	D8	73	95	A4	F7	02	06	0A	1E	22	66	AA
2	E5	34	5C	E4	37	59	EB	26	6A	BE	D9	70	90	AB	E6	31
3	53	F5	04	0C	14	3C	44	CC	4F	D1	68	B8	D3	6E	B2	CD
4	4C	D4	67	A9	E0	3B	4D	D7	62	A6	F1	08	18	28	78	88
5	83	9E	B9	D0	6B	BD	DC	7F	81	98	B3	CE	49	DB	76	9A
6	B5	C4	57	F9	10	30	50	F0	0B	1D	27	69	BB	D6	61	A3
7	FE	19	2B	7D	87	92	AD	EC	2F	71	93	AE	E9	20	60	A0
8	FB	16	3A	4E	D2	6D	B7	C2	5D	E7	32	56	FA	15	3F	41
9	C3	5E	E2	3D	47	C9	40	C0	5B	ED	2C	74	9C	BF	DA	75
A	9F	BA	D5	64	AC	EF	2A	7E	82	9D	BC	DF	7A	8E	89	80
B	9B	B6	C1	58	E8	23	65	AF	EA	25	6F	B1	C8	43	C5	54
C	FC	1F	21	63	A5	F4	07	09	1B	2D	77	99	B0	CB	46	CA
D	45	CF	4A	DE	79	8B	86	91	A8	E3	3E	42	C6	51	F3	0E
E	12	36	5A	EE	29	7B	8D	8C	8F	8A	85	94	A7	F2	0D	17
F	39	4B	DD	7C	84	97	A2	FD	1C	24	6C	B4	C7	52	F6	01

AES

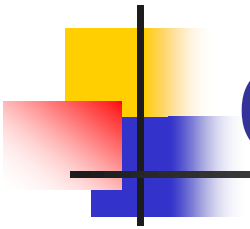
- Descifrado:
  - Aplicar las inversas de cada una de las funciones en el orden contrario y utilizar las mismas K_i pero en orden inverso.
  - Inversa de Sub() en siguiente diapositiva.
  - Inversa de Desplazar_fila(): desplazamiento circular a derecha.
  - Inversa de Mezclar_columna(): la matriz de multiplicación es

$$M = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix}$$

AES

Función Sub() inversa

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D



Contenidos

2.1 Introducción ✓

2.2 Cifrado en bloque

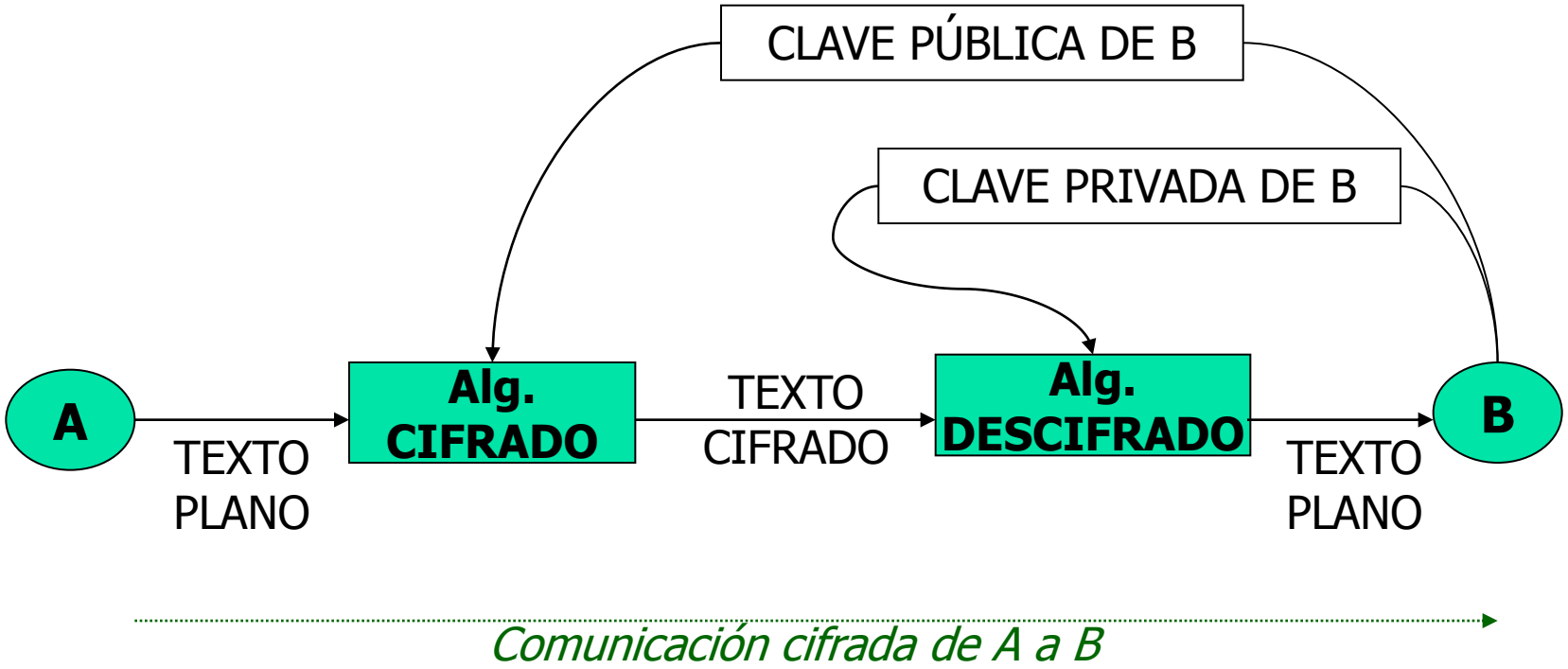
- 2.2.1 Algoritmos simétricos
  - 2.2.1.1 DES ✓
    - Triple DES ✓
  - 2.2.1.2 AES ✓
- 2.2.2 Algoritmos asimétricos
  - 2.2.2.1 RSA
  - 2.2.2.2 Diffie-Hellman
  - 2.2.2.3 Curvas elípticas

Algoritmos asimétricos

- Algoritmos asimétricos
  - Basados en funciones matemáticas
  - Dos claves
  - Claves de longitud considerable
  - Más lentos
  - Se usan básicamente para cifrar la clave de sesión simétrica de cada mensaje o transacción particular
  - Ej: RSA
- Problemas a solucionar en criptografía simétrica
  - Distribución de claves
  - Firma digital

Algoritmos asimétricos

- En criptografía asimétrica: una clave para cifrar y otra para descifrar (caso particular RSA)



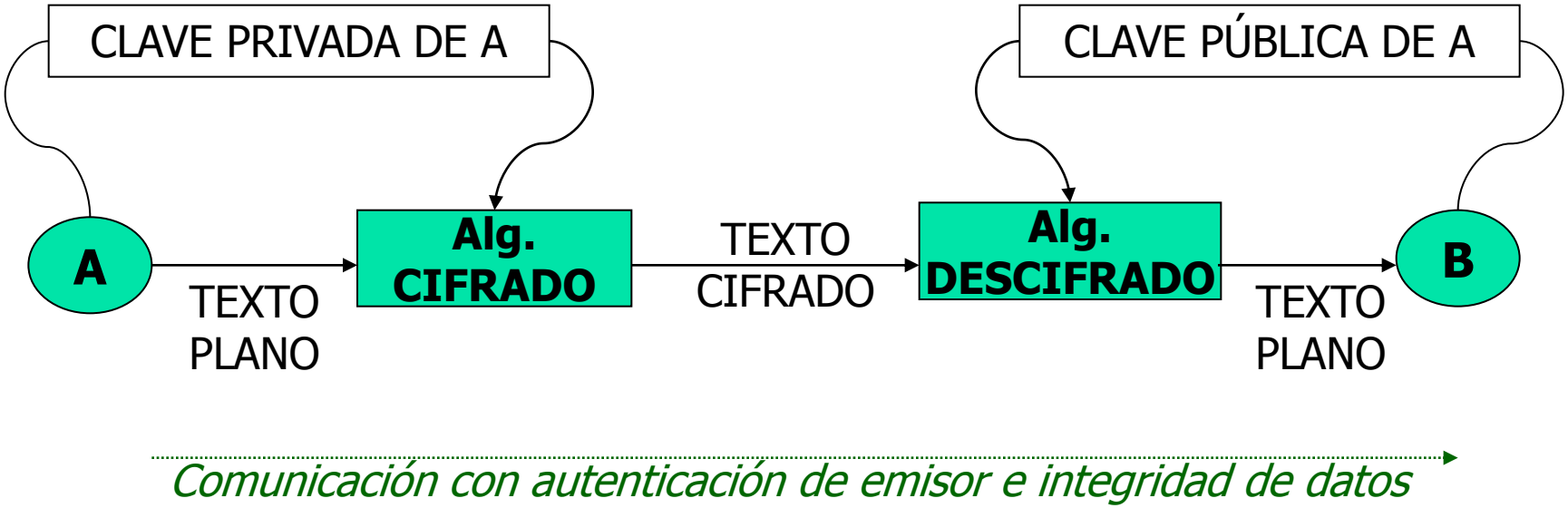
Algoritmos asimétricos

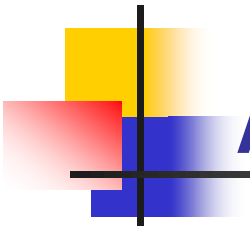
- NOTACIÓN:
  - Clave pública de A KP_A
  - Clave privada de A Kp_A

  - Según el ejemplo el usuario A haría $Y=E_{KPB}(X)$
  - Según el ejemplo el usuario B haría $X=D_{KpB}(Y)$

Algoritmos asimetricos

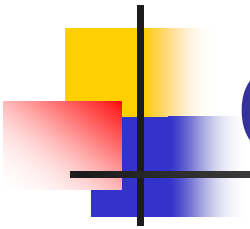
- Uso de algoritmos asimétricos para autenticación





Algoritmos asimétricos

- Aplicaciones
  - Cifrado/descifrado
  - Firma Digital
  - Intercambio de claves



Contenidos

2.1 Introducción ✓

2.2 Cifrado en bloque

- 2.2.1 Algoritmos simétricos
  - 2.2.1.1 DES ✓
    - Triple DES ✓
  - 2.2.1.2 AES ✓
- 2.2.2 Algoritmos asimétricos ✓
  - 2.2.2.1 RSA
  - 2.2.2.2 Diffie-Hellman
  - 2.2.2.3 Curvas elípticas

Algoritmo RSA

- RSA = Rivest, Shamir, Adleman en 1977
- Claves indistintamente para cifrar, descifrar y/o autenticar

$$Y = X^e \bmod n$$

$$X = Y^d \bmod n$$

- Claves:

  Clave pública KP={e,n}

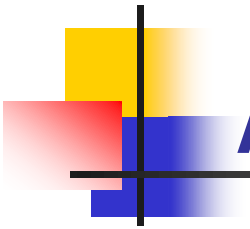
  Clave privada Kp={d,n}

Algoritmo RSA

- Requisitos:
  - Posible encontrar valores e, d y n tales que X^{ed} sea igual a X mod n
  - Relativamente fácil calcular X^e e Y^d para todos los valores de $X < n$
  - Imposible determinar d aunque e y n sean conocidos

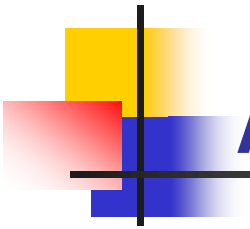
Algoritmo RSA

- Ingredientes:
  - p y q dos números primos
  - n = p*q
  - d / m.c.d.(Φ(n),d)=1 donde 1< d < Φ(n)
  - e / e = d^{-1} mod Φ(n)



Algoritmo RSA

- Ejemplo sencillo

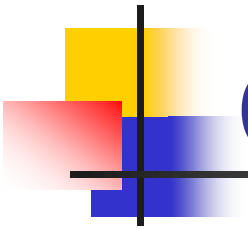


Algoritmo RSA

- Ejemplo de cifrado

Algoritmo RSA

- En la práctica hay que escoger valores de p y q con aproximadamente 154 dígitos (se recomienda que n~1024 bits ⇨308 dígitos)
- Para obtener la clave privada a partir de la pública , el atacante debe conocer p y q ⇨ computacionalmente intratable ⇨ problema de factorización
- Ataques a RSA
  - Fuerza bruta
  - Si $X=X^e \bmod n$ ⇨ $\theta_n = [1+mcd(e-1, p-1)*[1+mcd(e-1, q-1)]$
  - Ataque de intermediario ⇨ anillos de confianza



Contenidos

2.1 Introducción ✓

2.2 Cifrado en bloque

- 2.2.1 Algoritmos simétricos
  - 2.2.1.1 DES ✓
    - Triple DES ✓
  - 2.2.1.2 AES ✓
- 2.2.2 Algoritmos asimétricos ✓
  - 2.2.2.1 RSA ✓
  - 2.2.2.2 Diffie-Hellman
  - 2.2.2.3 Curvas elípticas

Algoritmo Diffie-Hellman

- 1976, primera publicación sobre cifrado de clave pública ⇨ técnica de intercambio de clave
  - No son necesarias claves públicas en sentido estricto
- Efectividad radica en la dificultad de calcular logaritmos discretos:
  - Raíz primitiva de un número primo p es un valor α cuyas potencias generan distintos números enteros entre 1 y p-1.
  - Para cualquier entero b y una raíz primitiva α de un número primo p, es posible encontrar un único exponente i tal que $b= \alpha^i \bmod p$

Algoritmo Diffie-Hellman

- Dos valores públicos:
  - un número primo q
  - un entero α que es raíz primitiva de q
- Si A y B quieren intercambiar una clave:

1) A selecciona nº aleatorio x_A ($x_A < q$) y calcula

$$y_A = \alpha^{x_A} \bmod q$$

Algoritmo Diffie-Hellman

2) De la misma forma B selecciona nº aleatorio x_B (x_B<q) y calcula

$$y_B = \alpha^{x_B} \bmod q$$

3) Cada uno guarda x (x_A y x_B) en secreto y publican y (y_A e y_B) al otro comunicante

4) Usuario A calcula su clave secreta como

$K_A = (y_B)^{x_B} \bmod q$

$$K_A = (y_B)^{x_A} \bmod q$$

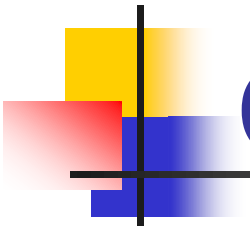
5) Usuario B calcula su clave secreta como

Se puede demostrar que K_A y K_B son el mismo valor.

Algoritmo Diffie-Hellman

- Ambos comunicantes intercambian una clave secreta de modo seguro
  - Si se capturan los mensajes transmitidos (q, α, y_A, y_B) habría que calcular logaritmo discreto
  - Ejemplo

$$x_B = ln\, d_{\alpha,q}(y_B)$$



Contenidos

2.1 Introducción ✓

2.2 Cifrado en bloque

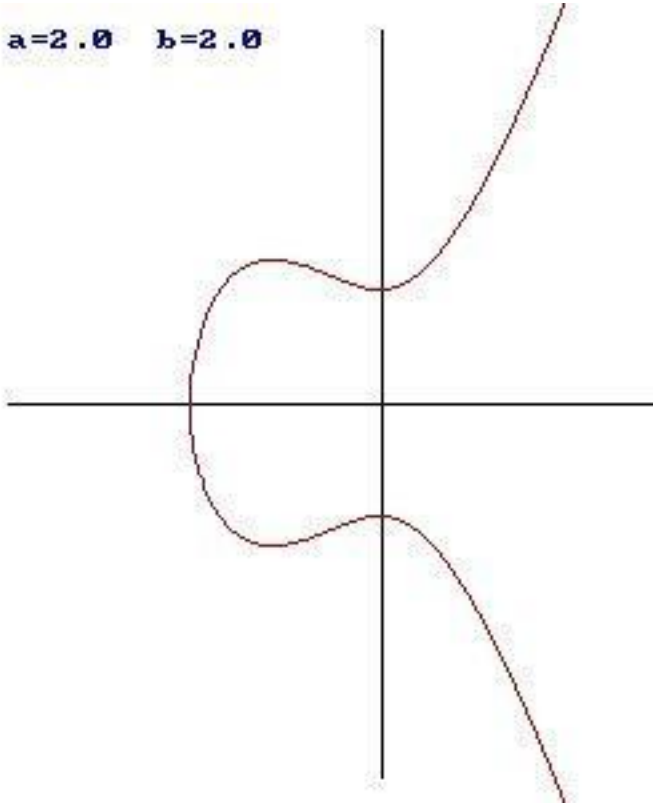
- 2.2.1 Algoritmos simétricos
  - 2.2.1.1 DES ✓
    - Triple DES ✓
  - 2.2.1.2 AES ✓
- 2.2.2 Algoritmos asimétricos ✓
  - 2.2.2.1 RSA ✓
  - 2.2.2.2 Diffie-Hellman ✓
  - 2.2.2.3 Curvas elípticas

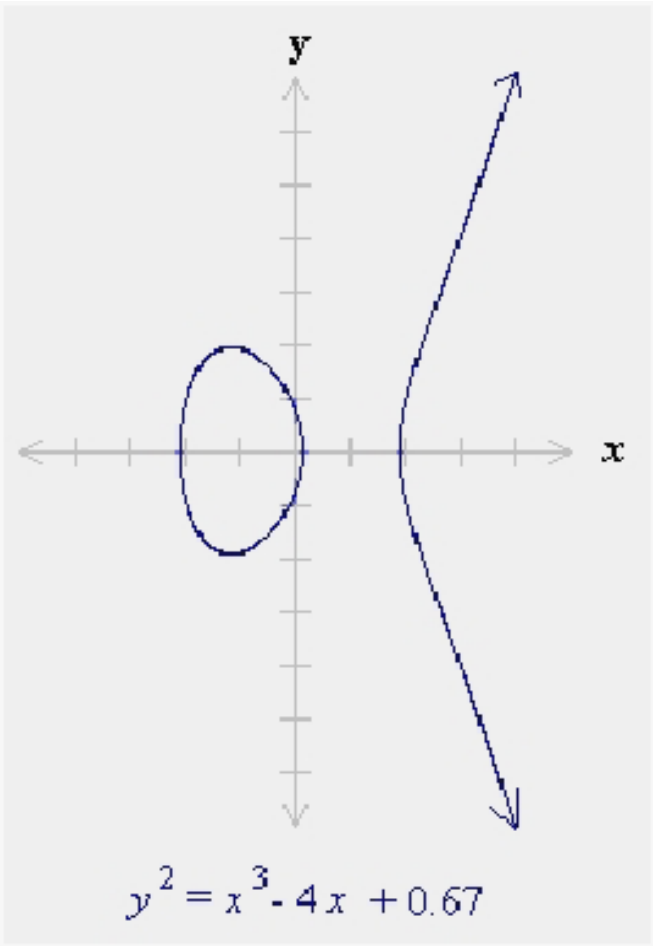
Curvas elípticas

- RSA emplea claves cada vez más largas $\Rightarrow \uparrow t_{procesado}$
- Nuevo sistema competitivo: el cifrado de curva elíptica (Elliptic Curve Cryptography, ECC):
  - RFC 3278, IPSec, etc...
  - Ventaja de ECC frente a RSA: ofrece igual seguridad para longitudes de clave mucho menores
  - Criptoanálisis reciente: www.certicom.com

Curvas elípticas

- Las curvas elípticas no son elipses:





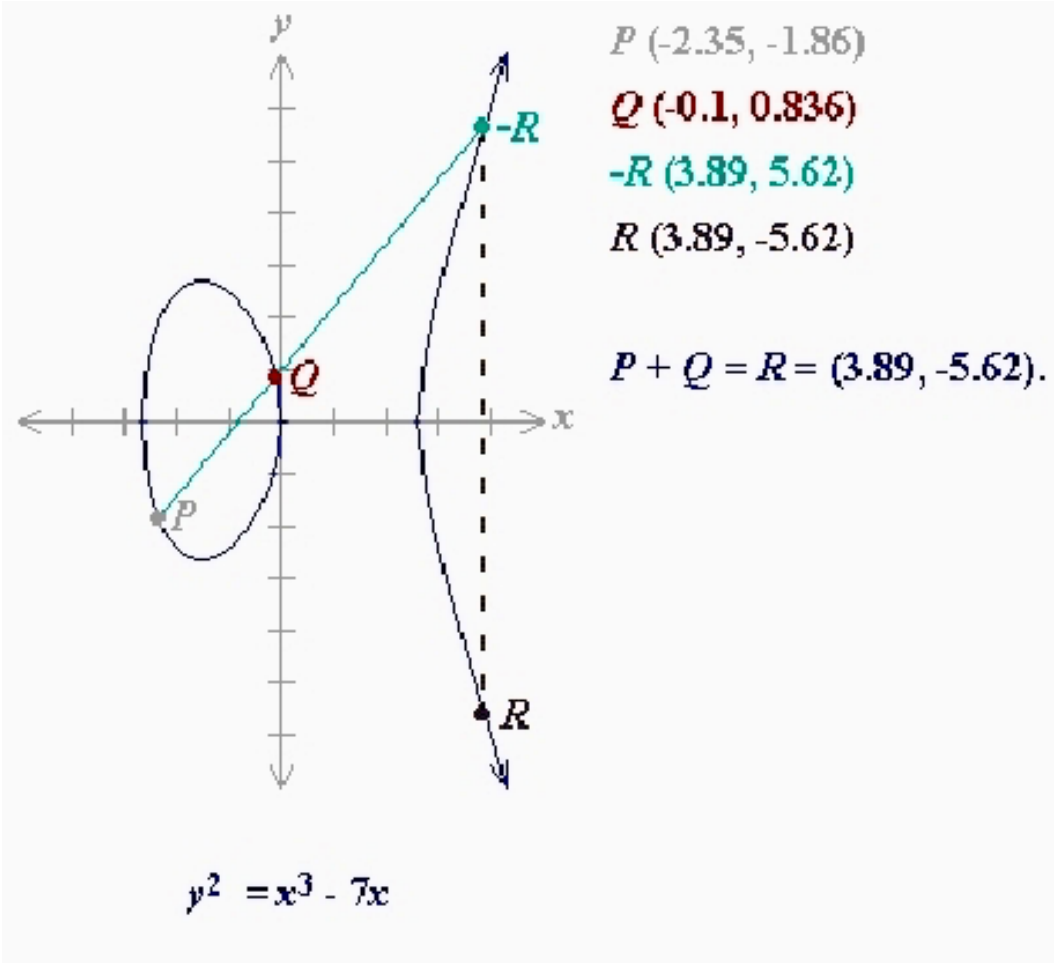
Curvas elípticas

- Ecuación de una curva elíptica:

$$y^2 + axy + by = x^3 + cx^2 dx + e$$

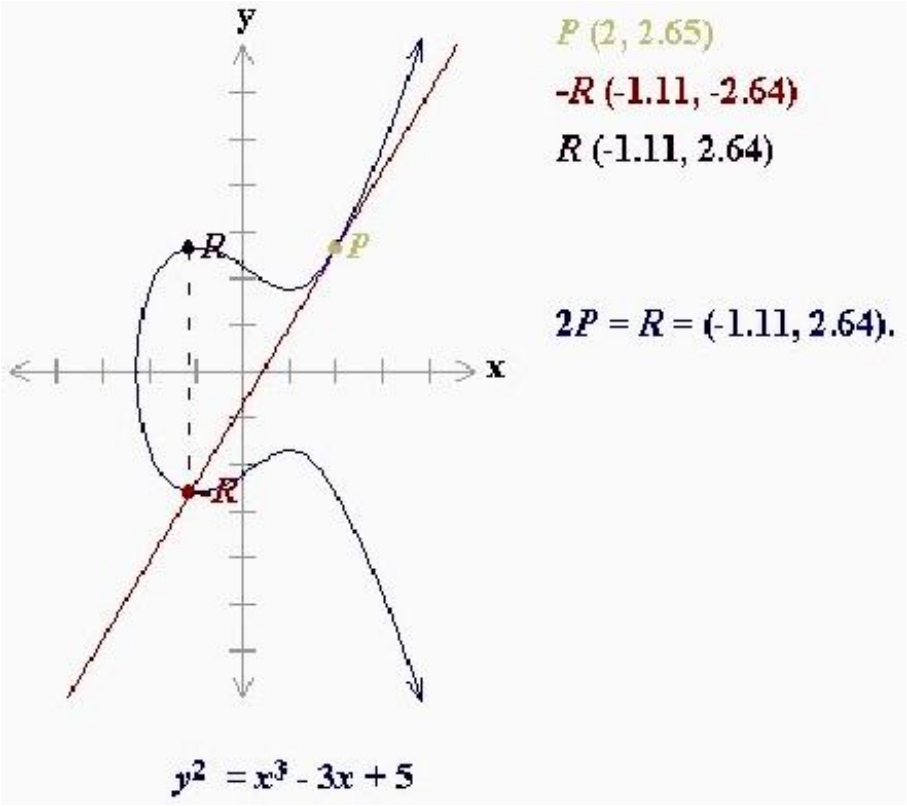
- Operación suma:"si tres puntos de una curva elíptica caen en una línea recta, entonces su suma es *0*"
  - *0* ≡ punto en el infinito
- Reglas de suma de una curva elíptica:
  - 1) Para sumar dos puntos P y Q con diferente coordenada x hay que dibujar una recta entre ellos, encontrar el tercer punto intersección R $\Rightarrow$ Q+P+R=*0* $\Rightarrow$ Q+P=-R

Curvas elípticas

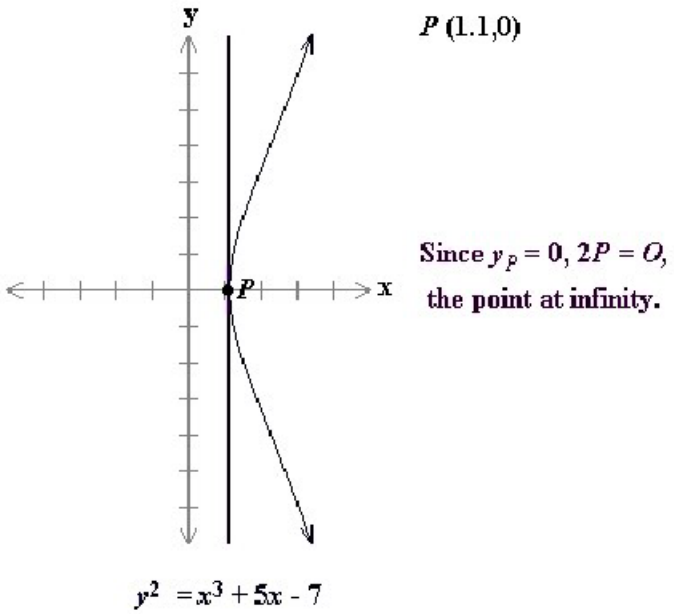


Curvas elípticas

- 2) Para doblar un punto P, dibuje la tangente y encuentre el otro punto de intersección R. Dibuje la vertical paralela al eje que pasa por R $\Rightarrow$ P+P=2P=-R

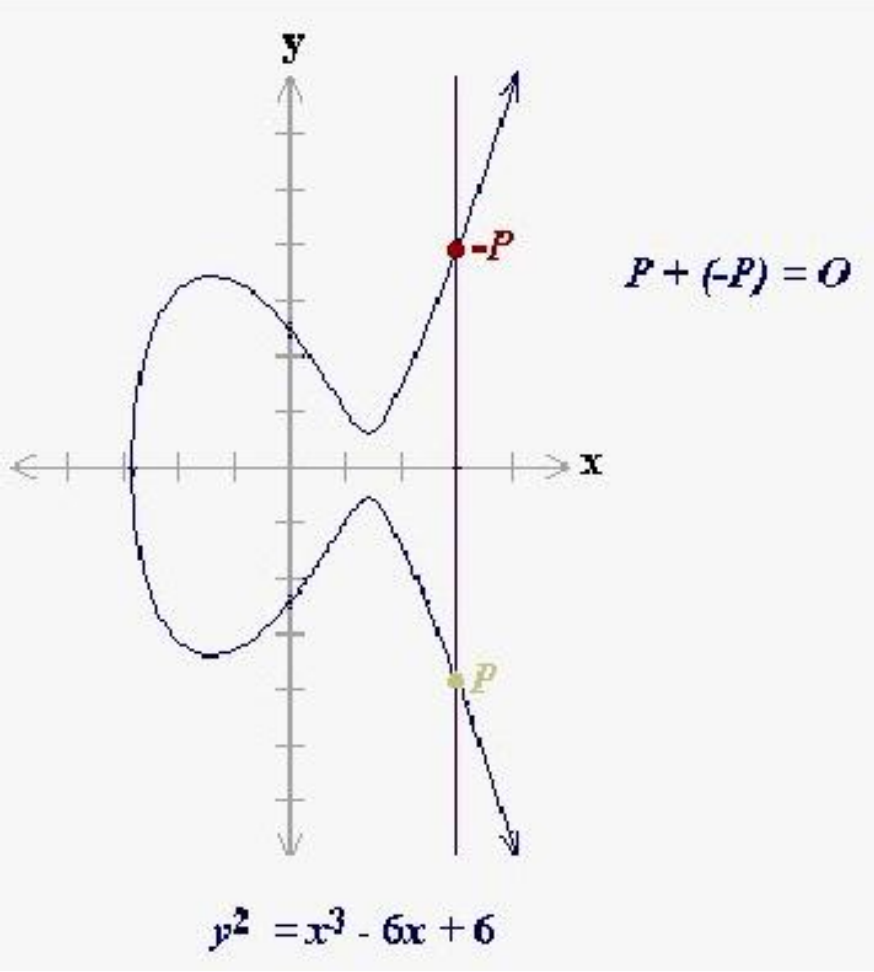


Curvas elípticas



Curvas elípticas

- 3) Dada una línea vertical que corta a la curva elíptica en dos puntos con la misma coordenada x, también corta a la curva en el punto en el infinito $\Rightarrow P_1+P_2=O$ $\Rightarrow P_1=-P_2$



Curvas elípticas

- 4) *0* es la identidad aditiva $\Rightarrow$ *0* = -*0* $\Rightarrow$ P + *0* = P

- En cualquier caso se cumple la propiedad asociativa y la conmutativa

Curvas elípticas

- Curvas elípticas sobre campos finitos: grupo elíptico en módulo p (p es un número primo)
- Escogemos dos números enteros enteros no negativos (a y b) donde a,b < p tales que :

$$4a^3 + 27b^2 (mod\ p) \neq 0$$

- $E_p(a,b)$ **grupo elíptico** mod p cuyos elementos (x,y) son pares enteros no negativos menores que p que satisfacen:

$$y^2 \equiv x^3 + ax + b\ (mod\ p)$$

junto con el punto en el infinito

Curvas elípticas

- Crear lista de puntos que pertenecen a $E_p(a,b)$:
  - Para cada x / $0 \leq x < p$ calcular $x^3+ax+b \pmod p$
  - Para cada resultado anterior determinar si tiene raíz cuadrada mod p.
    - Si no tiene entonces no existen puntos en $E_p(a,b)$ con este valor de x.
    - Si sí tiene entonces habrá dos valores de y que satisfagan la raíz.

Curvas elípticas

- Reglas de suma en $E_p(a,b)$: para todos los puntos $P,Q \in E_p(a,b)$:

1. $P+\mathcal{O}=P$
2. Si $P=(x,y) \Rightarrow P+(x,-y)=\mathcal{O}$ $\quad$ $(x,-y)=-P$
3. Si $P=(x_1,y_1)$ y $Q=(x_2,y_2)$ con $P \neq -Q \Rightarrow P+Q=(x_3,y_3)$

$$x_3 = \lambda^2 - x_1 - x_2 \ (mod\ p) \qquad y_3 = \lambda(x_1 - x_2) - y_1 \ (\text{mod}\ p)$$

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} \ (\text{mod}\ p)\ si\ P \neq Q \\ \dfrac{3x_1^2 + a}{2y_1} \ (\text{mod}\ p)\ si\ P = Q \end{cases}$$

Curvas elípticas

- Necesitamos encontrar un problema difícil de resolver...
  - $Q=k \cdot P$ donde $Q, P \in E_p(a,b)$ y $k<p$
- **Intercambio de clave Diffie-Hellman con ECC**
  - Escogemos número primo p (~180 bits) y parámetros a y b
  - Seleccionamos punto generador $G(x_1,y_1)$ en $E_p(a,b)$ tal que $n \cdot G = O$ para un n muy grande (n número primo)
  - A selecciona nº entero $n_A<n$ y calcula $P_A=n_A \cdot G$ (n_A secreto)
  - B selecciona nº entero $n_B<n$ y calcula $P_B=n_B \cdot G$ (n_B secreto)
  - A y B intercambian P_A, P_B
  - A calcula su clave secreta como $K=n_A \cdot P_B$
  - B calcula su clave secreta como $K=n_B \cdot P_A$

Curvas elípticas

- **Cifrado/descifrado con ECC**
  - Varias posibilidades: (ElGamal, DSA, etc.) con ECC
  - Codificar mensaje en puntos $P_m(x,y)$
  - Punto generador G y grupo elíptico $E_p(a,b)$
  - Cada usuario escoge un valor secreto n_x y genera una clave pública $P_x=n_x \cdot G$
- Cifrar (mensaje de A a B):
  - A escoge número entero positivo aleatorio K ⇨ mensaje cifrado $C_m=\{K \cdot G, P_m+K \cdot P_B\}=\{P_1, P_2\}$
- Descifrar:
  - B multiplica primer punto del par (P_1) con su valor secreto n_B y le sustrae el resultado al segundo punto del par (P_2)

Curvas elípticas

- Ataques con el método Pollard's Rho
- Comparación de niveles de seguridad:

Nivel de Seguridad	Esquema Simétrico (tamaño de clave)	Esquema basado en ECC (tamaño de n)	DSA/RSA (tamaño del modulo)
56	56	112	512
80	80	160	1024
112	112	224	2048
128	128	256	3072
192	192	384	7680
256	256	512	15360

TAMAÑOS DE CLAVE COMPARABLES

	Parámetros del sistema	Clave Pública	Clave Privada
RSA	n/a	1088	2048
DSA	2208	1024	160
ECC	481	161	160

TAMAÑOS DE LOS PARÁMETROS DEL SISTEMA Y PAR DE CLAVES (bits)

	Tamaño mensaje encriptado
RSA	1024
ElGamal	2048
ECC	321

TAMAÑO DE MENSAJES CIFRADOS

	Tamaño de firma
RSA	1024
DSA	320
ECC	320

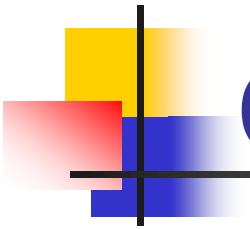
TAMAÑO DE FIRMA (bits)

ECC	
Tamaño clave	MIPS-Años
150	$3.8\ 10^{10}$
205	$7.1\ 10^{18}$
234	$1.6\ 10^{28}$

RSA	
Tamaño clave	MIPS-Años
512	$3\ 10^{4}$
1024	$3\ 10^{11}$
1536	$3\ 10^{16}$
2048	$3\ 10^{20}$

Contenidos

- 2.1 Introducción ✓
- 2.2 Cifrado en bloque
  - 2.2.1 Algoritmos simétricos
    - 2.2.1.1 DES ✓
      - Triple DES ✓
    - 2.2.1.2 AES ✓
  - 2.2.2 Algoritmos asimétricos ✓
    - 2.2.2.1 RSA ✓
    - 2.2.2.2 Diffie-Hellman ✓
    - 2.2.2.3 Curvas elípticas ✓
- 2.3 Cifrado en Flujo
  - 2.3.1 RC4
  - 2.3.2 A5



Cifrado en flujo

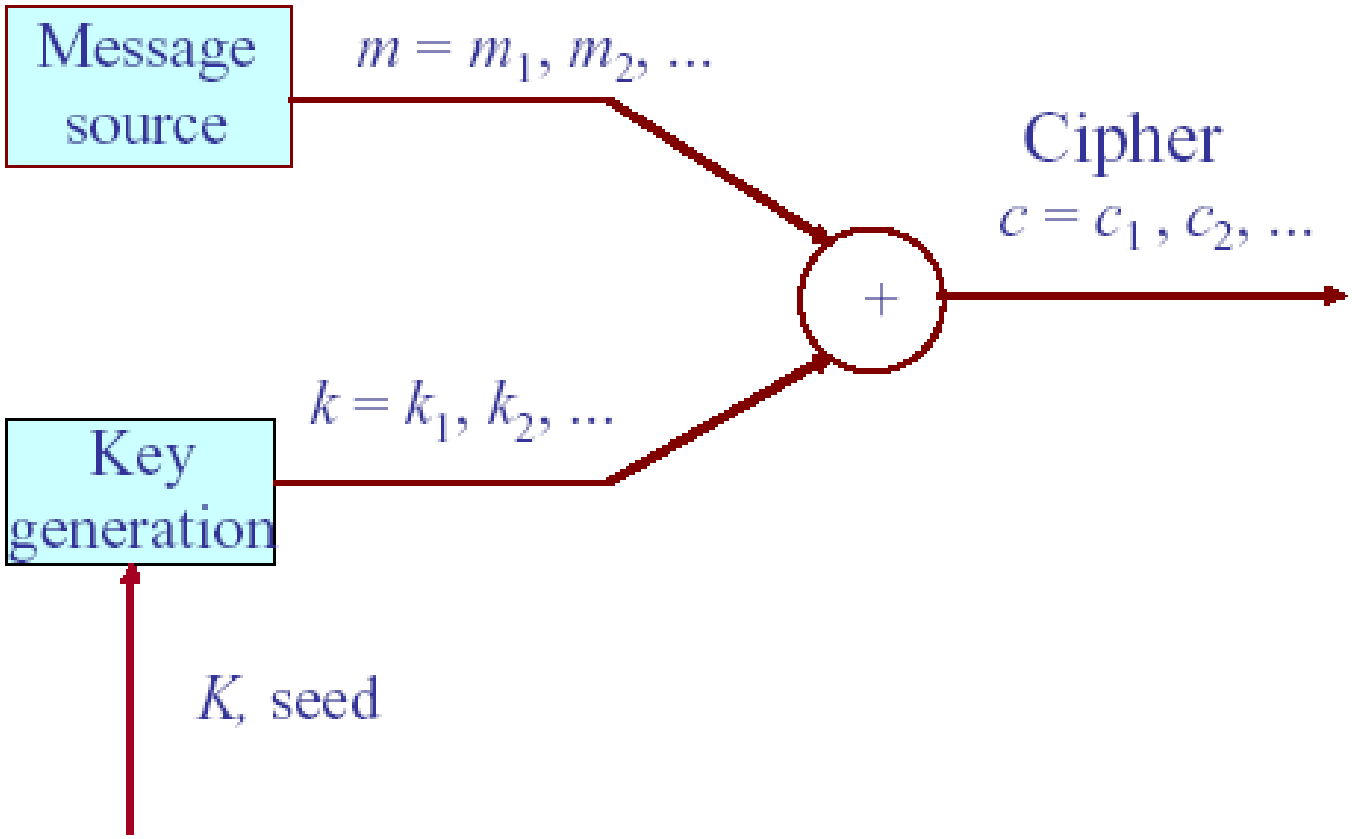
- El mensaje a cifrar NO se divide en bloques
- El cifrado en flujo cifra en tiempo real
- 1917 Mauborgne y Vernam inventaron primer criptosistema de cifrado en flujo:
  - Combinar carácter a carácter texto plano con una secuencia aleatoria de igual longitud utilizando una función simple y reversible (ej. XOR)
  - Enviada una única vez
  - Problema: la clave es tan larga como el propio mensaje y ¿cómo enviar la clave?

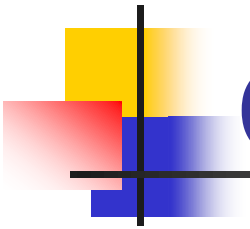
Cifrado en flujo

Generador pseudoaleatorio ⇨ secuencias criptográficamente aleatorias
+
Semilla del generador pseudoaleatorio = clave K

Cifrar: Secuencia pseudoaleatoria XOR con el texto plano
Descifrar: A partir de la semilla reconstruir secuencia pseudoaleatoria y hacer XOR con mensaje cifrado

Cifrado en flujo





Cifrado en flujo

- Veremos criptosistemas de clave privada
  - Especificación de un generador pseudoaleatorio
  - Combinación mediante la función XOR
  - Operaciones byte a byte

- Tipos de generadores:
  - Síncronos
  - Asíncronos

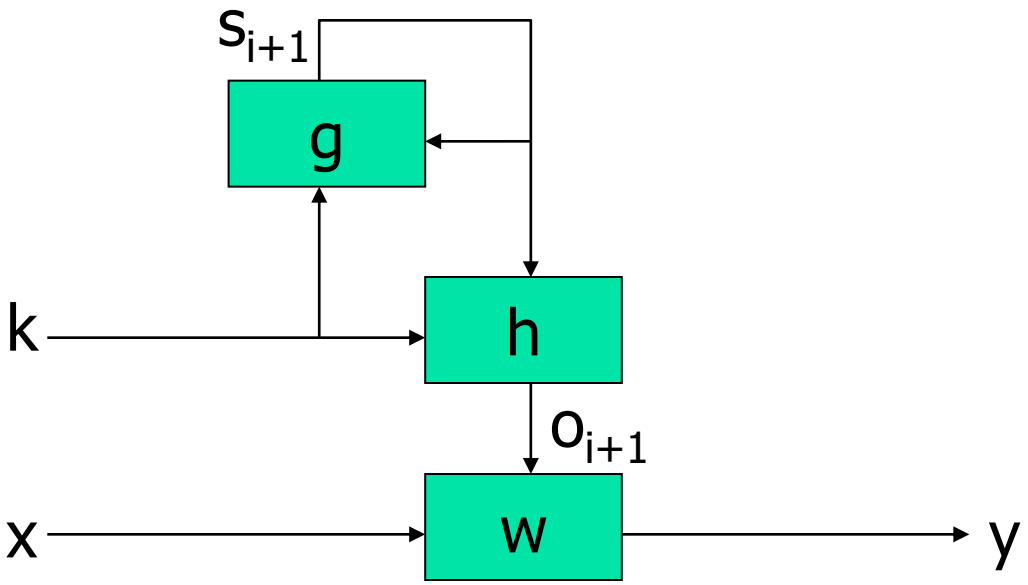
Cifrado en flujo

- **Generador síncrono**: La secuencia se calcula de modo independiente tanto del texto plano como del cifrado.

$s_{i+1} = g(s_i, k)$

$o_i = h(s_i, k)$

$y_i = w(x_i, o_i)$



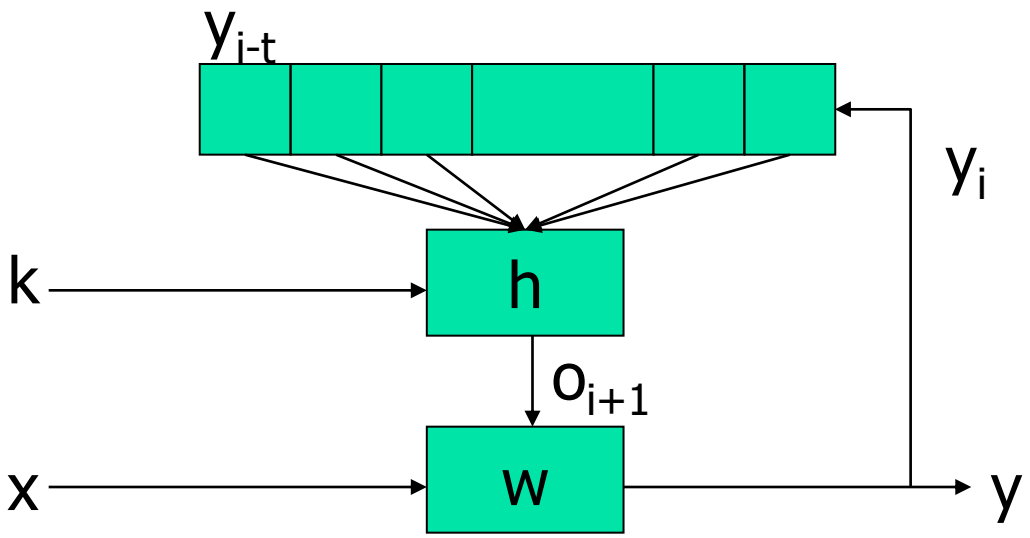
  - Emisor y receptor deben estar sincronizados (técnicas de verificación y de restablecimiento de sincronía)

Cifrado en flujo

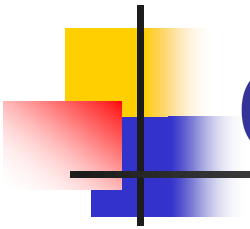
- **Generador asíncrono**: La secuencia generada es función de una semilla más una cantidad fija de los bits anteriores de la propia secuencia.

$o_i = h\ (K, y_{i-t}, y_{i-t+1}, y_{i-t+2}, \ldots, y_{i-1})$

$y_i = w\ (o_i, x_i)$



- Resistentes a pérdida o inserción de bits
- Sensibles a alteración de mensaje cifrado (técnicas de verificación)



Cifrado en flujo

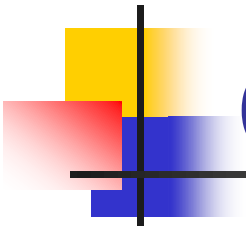
- Generadores de secuencias para cifrado en flujo basados en registros de desplazamiento retroalimentados (Feedback Shift Register):
  - Lineales
  - No lineales

Cifrado en flujo

- **Registros de Desplazamiento Retroalimentados Lineales** (Linear Feedback Shift Register, LFSR)
  - Conjunto de L estados $\{s_0, s_1, ..., s_{L-1}\}$, donde cada estado almacena 1 bit
  - Reloj controla variación de estados
  - Cada unidad de tiempo:
    - s_0 es la salida del registro
    - contenido de s_i se desplaza a s_{i-1} $(1 \leq i \leq L-1)$
    - contenido de s_{L-1}calculado como la suma en módulo 2 de los valores de un subconjunto prefijado del registro.

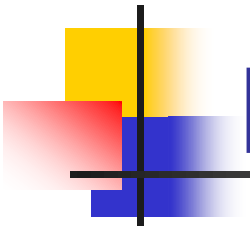
Cifrado en flujo

- **Registros de Desplazamiento Retroalimentados No Lineales** (Non Linear Feedback Shift Register, NLFSR)
  - Conjunto de L estados $\{s_0, s_1, \ldots, s_{L-1}\}$, donde cada estado almacena 1 bit
  - Reloj controla variación de estados
  - Cada unidad de tiempo:
    - s_0 es la salida del registro
    - contenido de s_i se desplaza a s_{i-1} $(1 \leq i \leq L-1)$
    - contenido de s_{L-1} calculado como una función boolena $f(s_{j-1}, s_{j-2}, \ldots, s_{j-L})$
- En general se usan *n* generadores lineales y una función *f* no lineal para combinar sus salidas: $f(R_1, R_2, \ldots, R_n)$



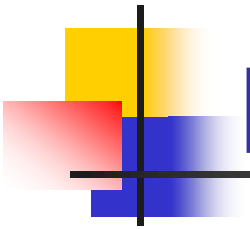
Contenidos

- 2.1 Introducción ✓
- 2.2 Cifrado en bloque
  - 2.2.1 Algoritmos simétricos
    - 2.2.1.1 DES ✓
      - Triple DES ✓
    - 2.2.1.2 AES ✓
  - 2.2.2 Algoritmos asimétricos
    - 2.2.2.1 RSA ✓
    - 2.2.2.2 Diffie-Hellman ✓
    - 2.2.2.3 Curvas elípticas ✓
- 2.3 Cifrado en Flujo
  - 2.3.1 RC4
  - 2.3.2 A5



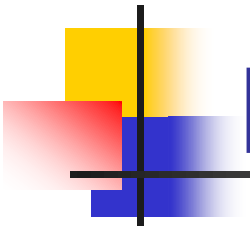
RC4

- Algoritmo de cifrado en flujo de clave privada diseñado por Ron Rivest (1987)
- Algoritmo propietario
- Implementación software
- Incluido en protocolos y estándares como WEP (Wired Equivalent Privacy) o SSL (Secure Socket Layer)
- Clave de hasta 256 bits (típicamente 40-256 bits)



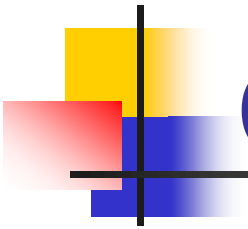
RC4

- Cifrado byte a byte
- Operaciones de cifrado/descifrado: emplea 256 bytes de memoria (s[0] a s[255]) y 3 variables i, j, k.
- Estado inicial:
  - 1) s[i]=i $\forall$i 0 ≤i ≤ 255
  - 2) j=0
  - 3) Para i=0 hasta 255 hacer:
    - j=(j+s[i]+key[i mod key_length]) mod 256
    - Intercambiar s[i] y s[j]



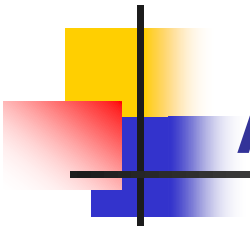
RC4

- Cifrar/descifrar:
  - 1) i=0; j=0
  - 2) para cada byte a cifrar
    - i=(i+1) mod 256
    - j=(j+s[i]) mod 256
    - Intercambiar s[i] y s[j]
    - k=(s[i]+s[j]) mod 256
    - XOR de s[k] con siguiente byte de entrada
- Desechar los primeros bytes de salida del generador y no usarlos para cifrar
- Otras aplicaciones: Lotus Notes, cifrado de claves en windows, MS Access, Adobe Acrobat, Oracle Secure Server, etc.



Contenidos

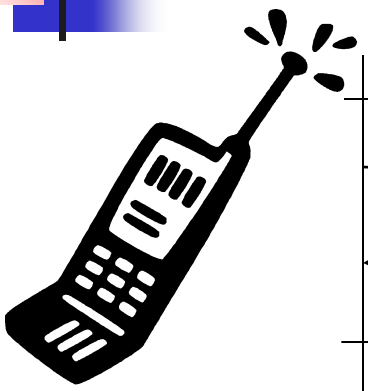
- 2.1 Introducción ✓
- 2.2 Cifrado en bloque
  - 2.2.1 Algoritmos simétricos
    - 2.2.1.1 DES ✓
      - Triple DES ✓
    - 2.2.1.2 AES ✓
  - 2.2.2 Algoritmos asimétricos ✓
    - 2.2.2.1 RSA ✓
    - 2.2.2.2 Diffie-Hellman ✓
    - 2.2.2.3 Curvas elípticas ✓
- 2.3 Cifrado en Flujo
  - 2.3.1 RC4 ✓
  - 2.3.2 A5



A5

- GSM desarrollado por Instituto Europeo de Estándares de Telecomunicaciones ⇨ protocolos criptográficos para confidencialidad y autenticación
- A3 algoritmo de autenticación
- A5 algoritmo de cifrado de voz
- A8 algoritmo generador de claves
- COMP128 algoritmo para ejecutar A3 y A8

A5



A8; K_i de la SIM

¿datos aleatorios?

D_1=datos aleatorios

A3 (K_i+D_1) -> resultado$_1$

resultado$_1$

A3 (K_i de la E.B.+D_1) -> resultado$_2$

¿resultado$_1$= resultado$_2$? SI

comunicación libre

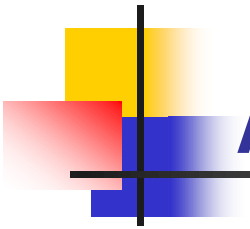
A8; K_i de la SIM

¿datos aleatorios?

D_2=datos aleatorios

A8(K_i+D_2) =K_c

A8(K_i+D_2) =K_c

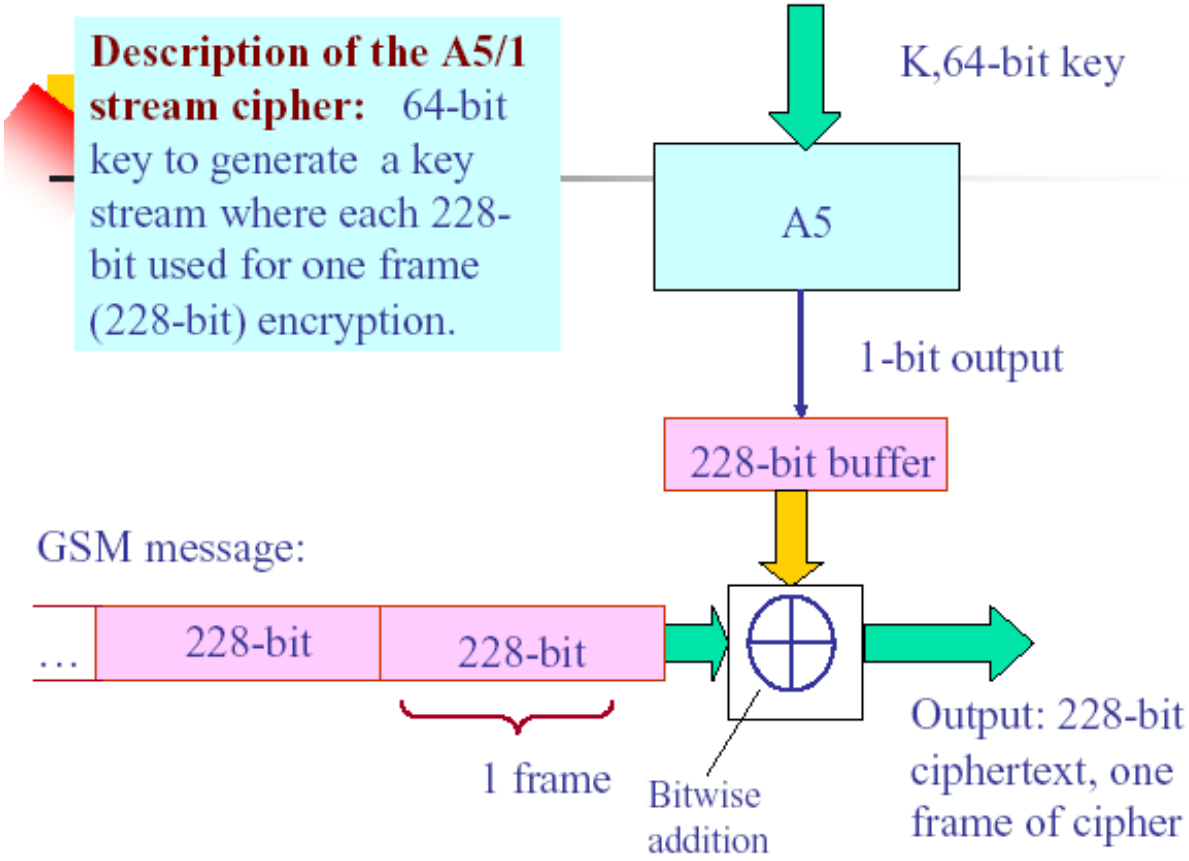


A5

- Proporciona privacidad a las comunicaciones GSM en el interfaz radio (GSM 1 trama cada 4,6 ms; 1 trama 228 bits)
- Dos versiones A5/1 y A5/2
- Ambas son una combinación de tres registros de desplazamiento retroalimentados lineales con señales de reloj irregulares y un combinador no lineal

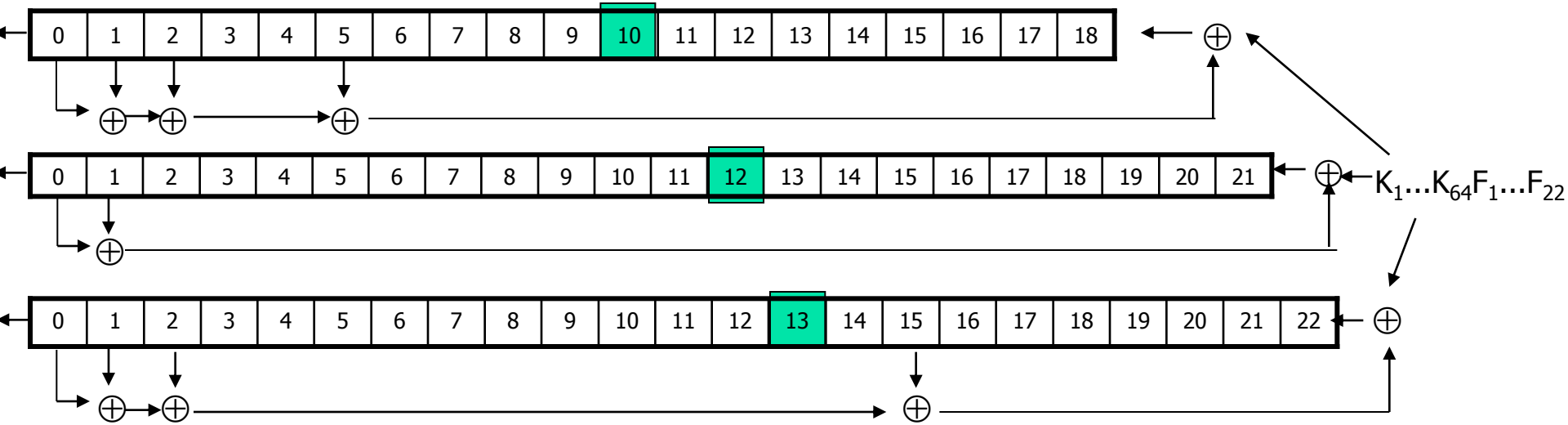
A5

- **A5/1** usa clave secreta de 64 bits y genera secuencia de bits (cada 228 bits se cifra una trama)



A5

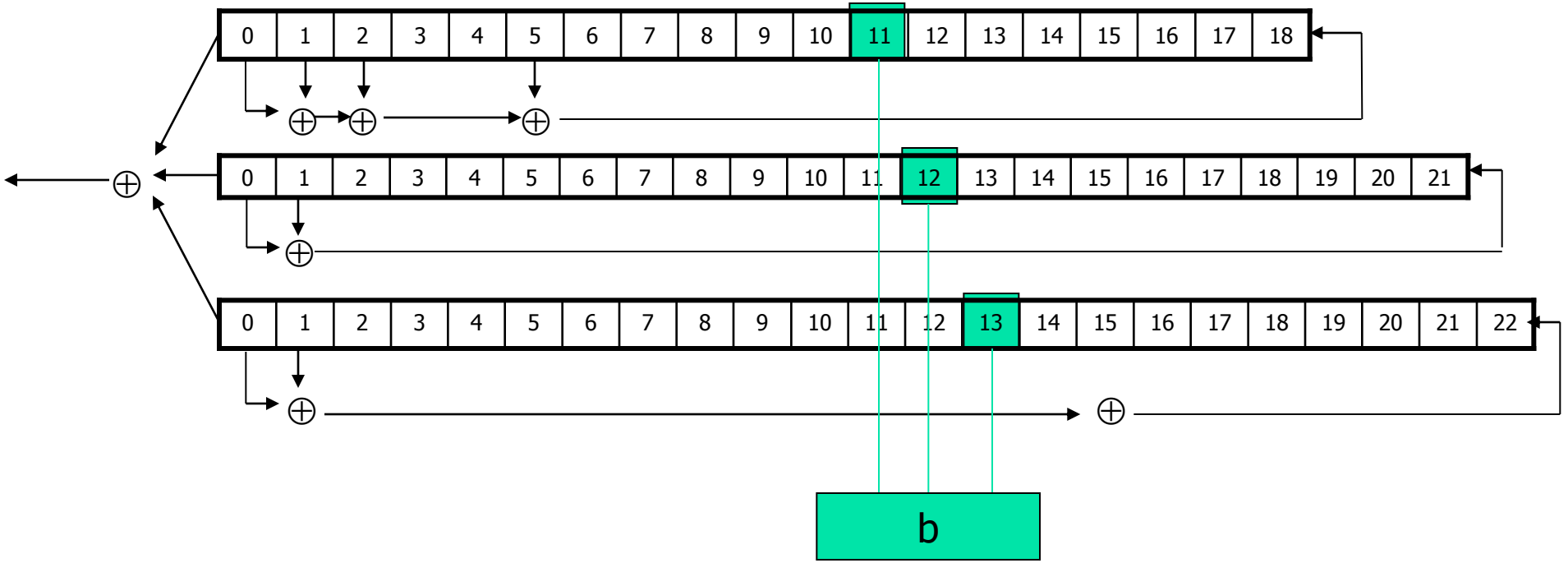
- 3 LFSR de longitudes 19, 22 y 23 bits ($x^{19}+x^5+x^2+x+1$; $x^{22}+x+1$; $x^{23}+x^{15}+x^2+x+1$)
- Inicialización cada trama: desde t=1 hasta t=64 se usa K, desde t=65 hasta t=89 se usa el bit (t-64) del número de trama F

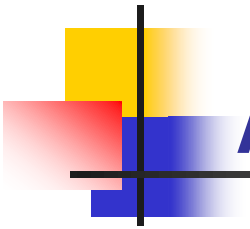


PROCESO DE INICIALIZACIÓN

A5

- Cada LFSR tiene un *tap* de reloj
- Se calcula el valor mayoría (b) de los tres *taps* cada unidad de tiempo
- LFSR recibe señal de reloj sólo si su *tap* coincide con valor b





A5

- Ataques:
  - Por fuerza bruta
  - Goldberg,Wagner, Briceno => “de los 64 bits de la clave diez de ellos son siempre cero”
  - Briceno => ingeniería inversa en Diciembre 1999
  - Biryukov, Shamir => “Real time criptoanalysis of A5/1 on a PC”, 1PC con 128 Mb RAM, 2-4 discos duros de 73 Gb cada uno, escáner digital.
- 3G
  - Generación de claves: MILENAGE
  - Confidencialidad e integridad en interfaz radio: KASUMI