



Criptologia moderna

Roberto Gómez

rogomez@campus.cem.itesm.mx

<http://webdia.cem.itesm.mx/dia/ac/rogomez>



Contenido



- Definiciones y componentes
- Criptosistemas simétricos
 - criptosistemas en flujo
 - criptosistemas en bloque
 - criptosistema DES
- Criptosistemas asimétricos
 - diffi hellman
 - la aritmética modular
 - RSA
 - huellas digitales y MD5
- Implementaciones
 - PGP
 - otras implementaciones



Definición y componentes



- *Criptología*.- Ciencia que estudia los aspectos y contenidos de información en condiciones de secrecía.
- Del griego: *criptos* oculto y *logos* tratado
- La Criptología se divide en:
 - *Criptografía*.
 - *Criptoanálisis*.



Criptografía



- Es el *arte* de construir códigos secretos.
- Es el conjunto de técnicas o procedimientos que alteran los símbolos de información sin alterar el contenido, convirtiendo a la información modificada en un conjunto de símbolos sin contenido para las partes que no disponen de las técnicas.



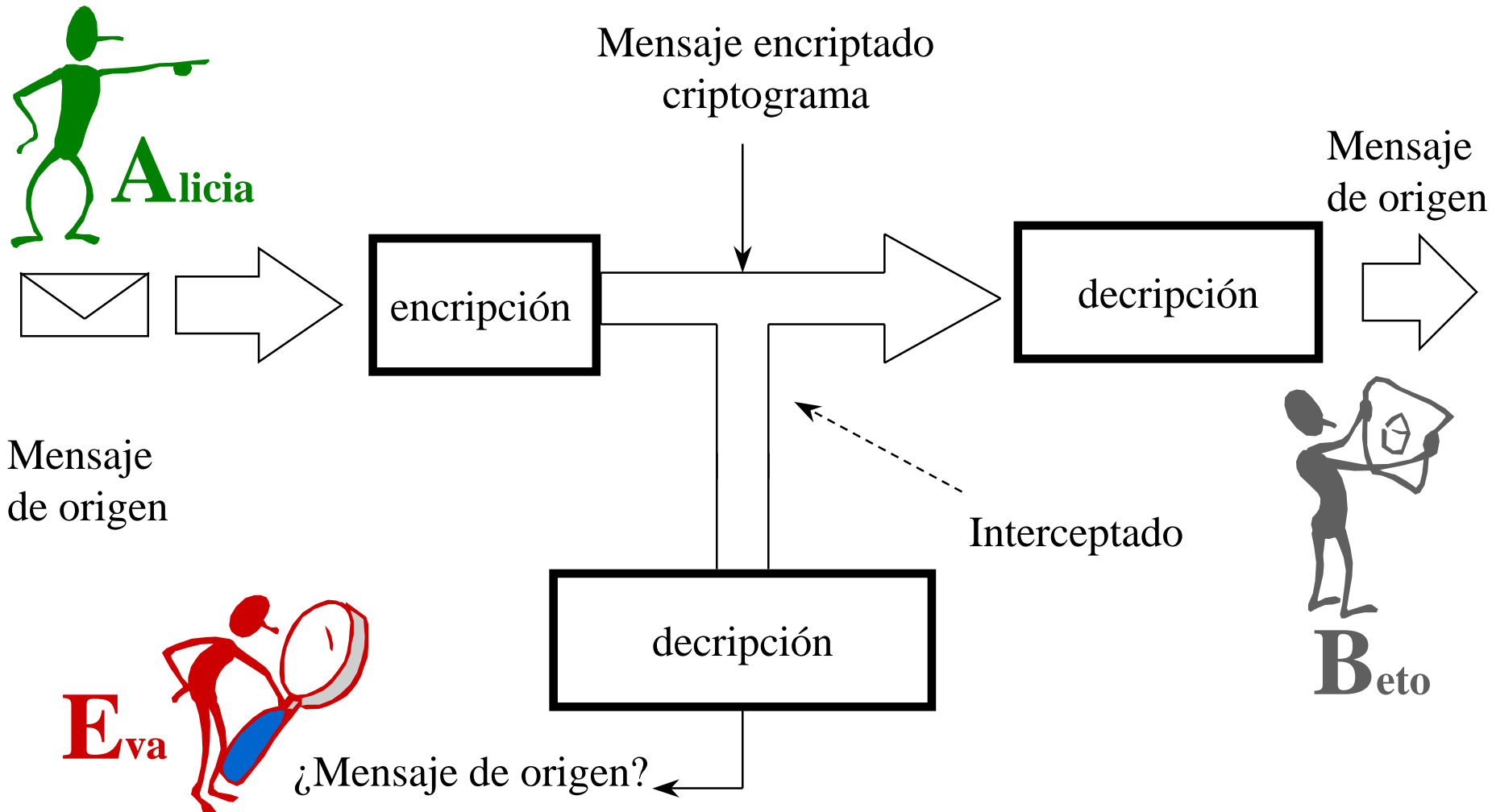
Criptoanálisis



- Metodologías y técnicas que permiten recuperar la información que ha sido previamente tratada por un procedimiento criptográfico, sin conocer “*a priori*” la técnica utilizada para la criptografía.



Proceso encriptación/decriptación





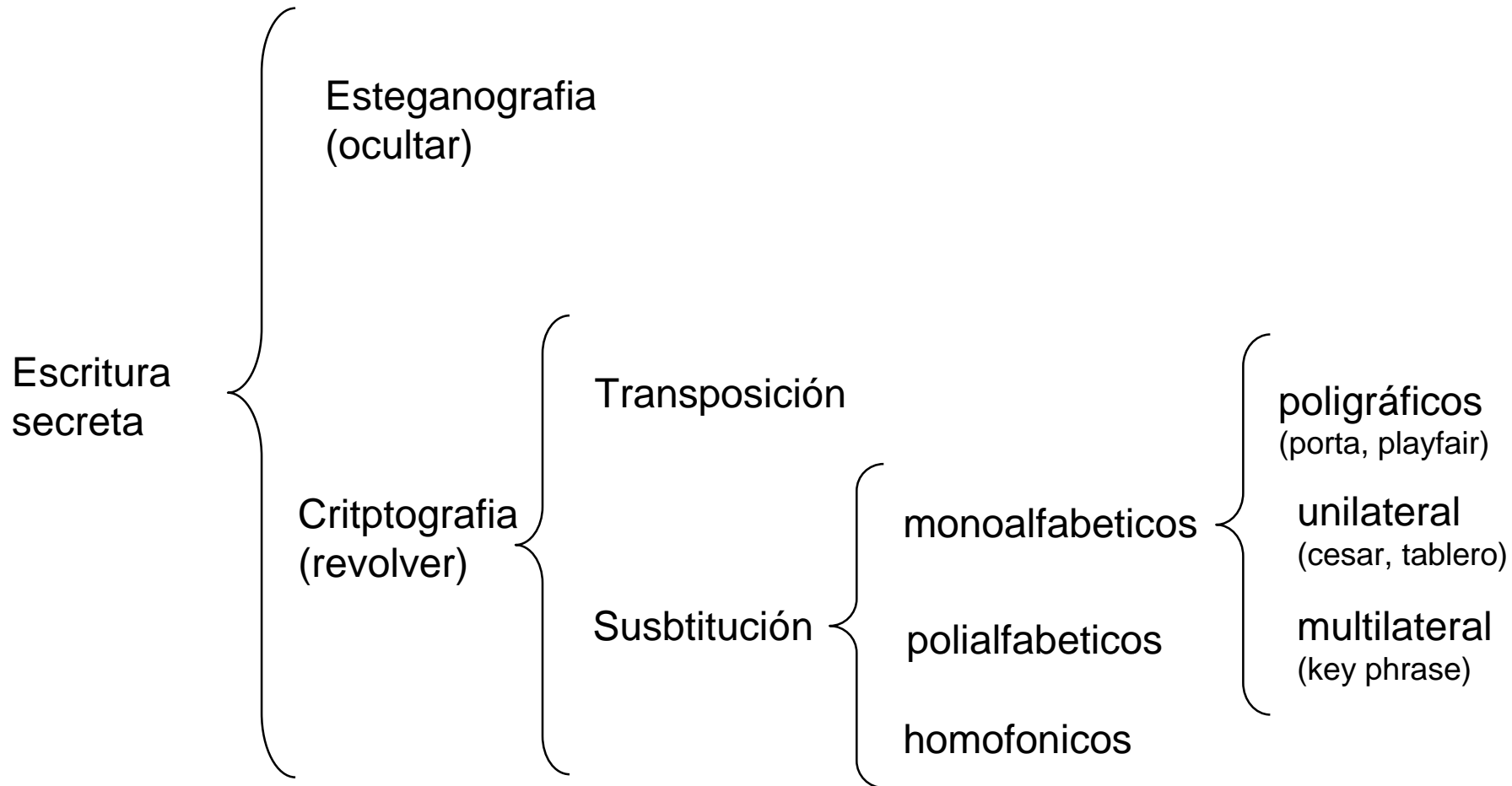
Algunos criptosistemas clásicos



- Cesar
- Pigpen
- Redefence
- Nihilist
- Grilla
- El criptosistema de Bacon
- El Polybius square
- Checker board
- Skytale
- Atbash
- Los nomenclators
- Porta
- Playfair
- Grandpre
- Beale
- Criptosistema ADFGVX



Una clasificación metodos secrecía





Máquinas criptograficas



- Los discos de encriptamiento
- El cilindro de Jefferson
 - el dispositivo M-94
- La máquina enigma
- La máquina de Lorenz
- La Bomba
- La máquina Coloussus

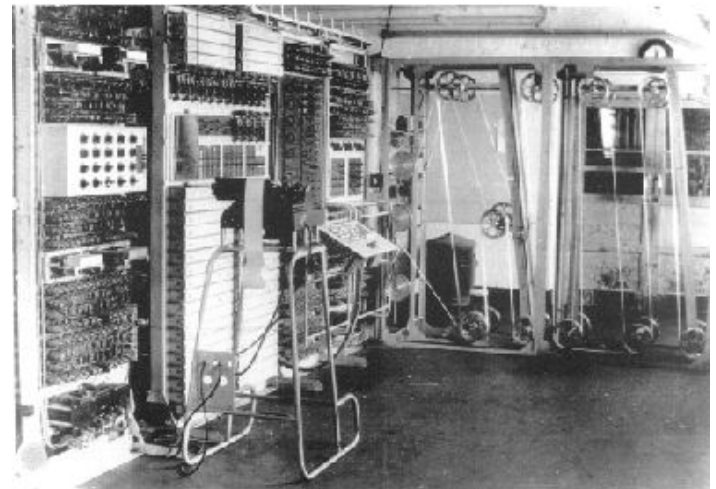
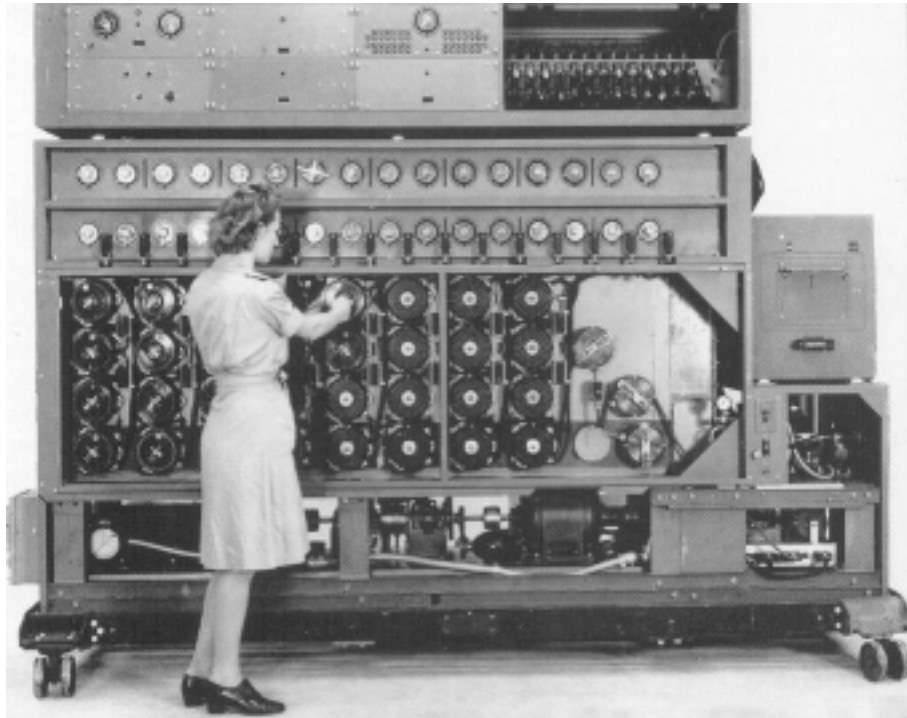


Imágenes máquinas criptográficas





Otras imagenes





Encriptando con una computadora



- La computadora “*maneja*” números en lugar de letras
 - solo números binarios (digitos binarios = bits)

$a = 1100001$

$! = 0100001$

$\& = 0100110$

- La encripción se realiza bajo mismo principio de substitución y transposición
 - elementos del mensaje son substituidos por otros elementos, o sus posiciones son intercambiadas o ambas



Encriptación por computadora



- Convertir mensaje a ASCII

Texto claro:

HELLO = 1001000 1000101 1001100 1001100 1001111

- Transposición: intercambiar las letras en un orden predeterminado

Texto claro:

HELLO = 10010001000101100110010011001001111

Criptograma:

LHOEL = 10011001001000100111110001011001100

- La transposición puede darse a nivel de bits

Letra original: 1001000

Letra encriptada: 0010010



Utilizando una llave



- Es posible utilizar una llave para transformar los bits.
- Por ejemplo supongamos el uso de la llave DAVID.

DAVID = 1000100 1000001 1010110 1001001 1000100

- Para encriptar/decriptar sumamos la llave al mensaje original, (suma binaria: xor)

Texto claro: HELLO

Texto ASCII: 10010001000101100110010011001001111

Llave: 10001001000001101011010010011000100

Criptograma: 00011000000100001101000001010001011



Métodos Criptográficos



- Métodos Simétricos
 - llave encriptado coincide con la de descifrado
 - la llave tiene que permanecer secreta
 - emisor y receptor se han puesto de acuerdo previamente o existe un centro de distribución de llaves
- Métodos asimétrico
 - llave encriptado es diferente a la de descriptado
 - llave encriptado es conocida por el público, mientras que la de decriptado solo por el usuario



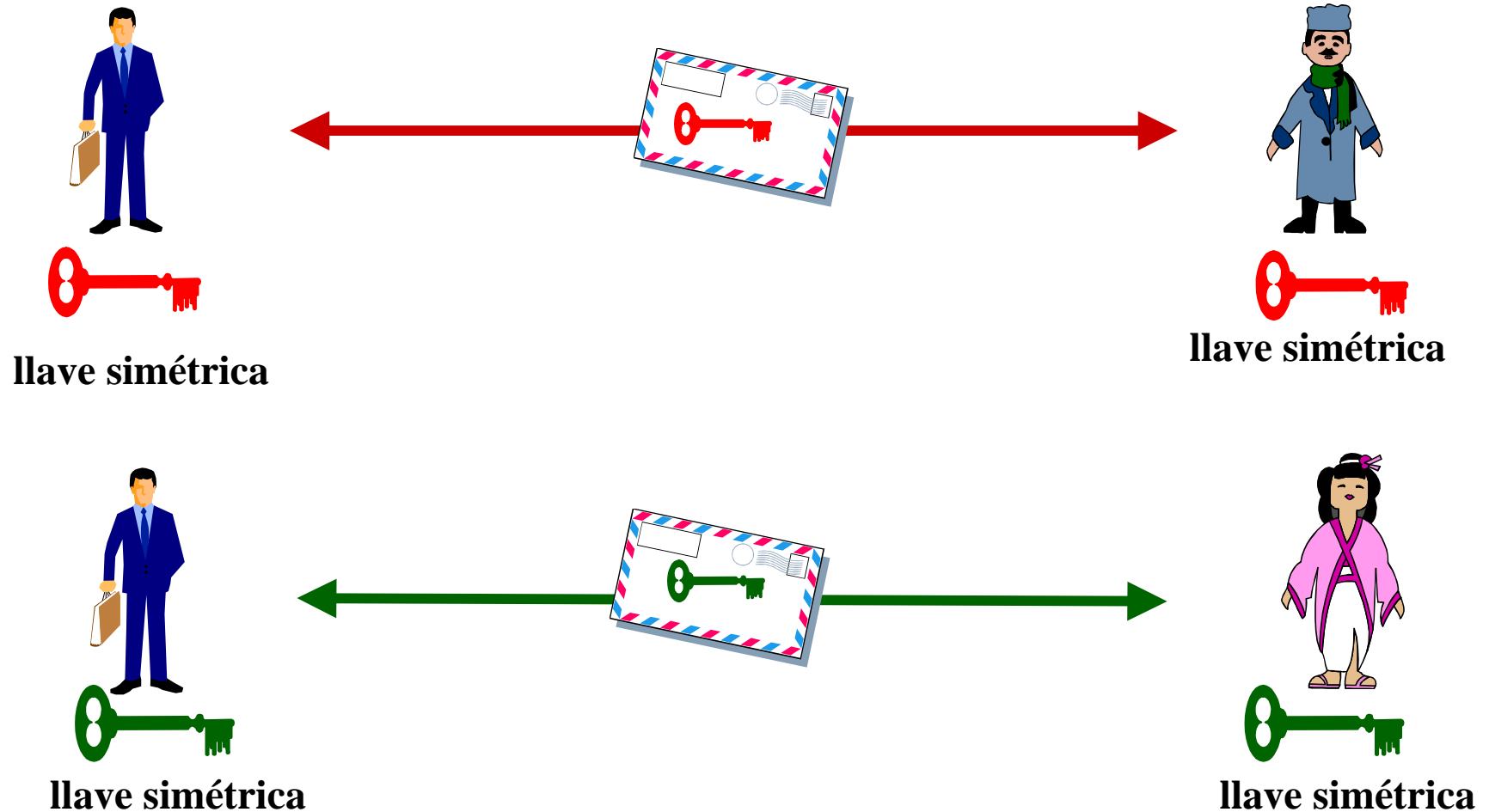
Sinónimos métodos



- Los métodos simétricos son propios de la criptografía clásica o criptografía de llave secreta
- Los métodos asimétricos corresponden a la criptografía de la llave pública, introducida por Diffie y Hellman en 1976



Esquema general encriptación llave secreta





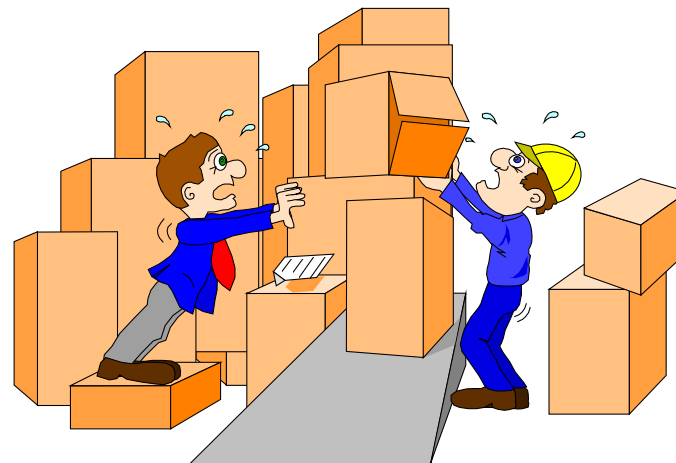
Clasificación métodos encriptación simetricos



- Encriptación en flujo



- Encriptación en bloques





Encriptado en flujo



- En inglés: stream ciphers.
- Usa la llave como semilla de un generador de números pseudo-aleatorio.
- El resultado del generador es una secuencia de bits.
- La secuencia se suma módulo 2 con el texto claro (emisión) o con el criptograma (recepción)



Características encriptación flujo



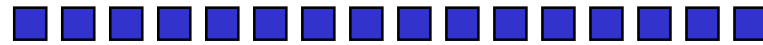
- Generar una secuencia larga e imprevisible de dígitos binarios a partir de una llave corta elegida de forma aleatoria
- Es sencillo, rápido y seguro
- Es más seguro conforme más se aproxima la secuencia binaria generada a una autentica secuencia aleatoria



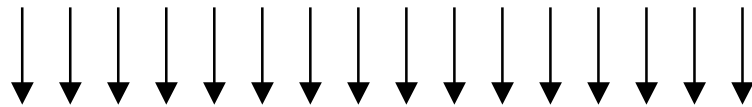
Encriptación de criptosistemas de flujo



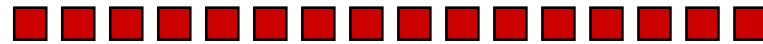
Texto Claro:



GNPA(semilla):



Criptograma:



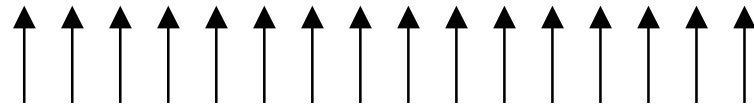
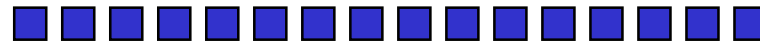
GNPA: Generador Números Pseudo-Aleatorios



Descripción de criptosistemas de flujo



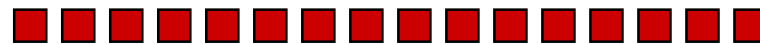
Texto Claro:



GNPA(semilla):



Criptograma:



GNPA: Generador Números Pseudo-Aleatorios



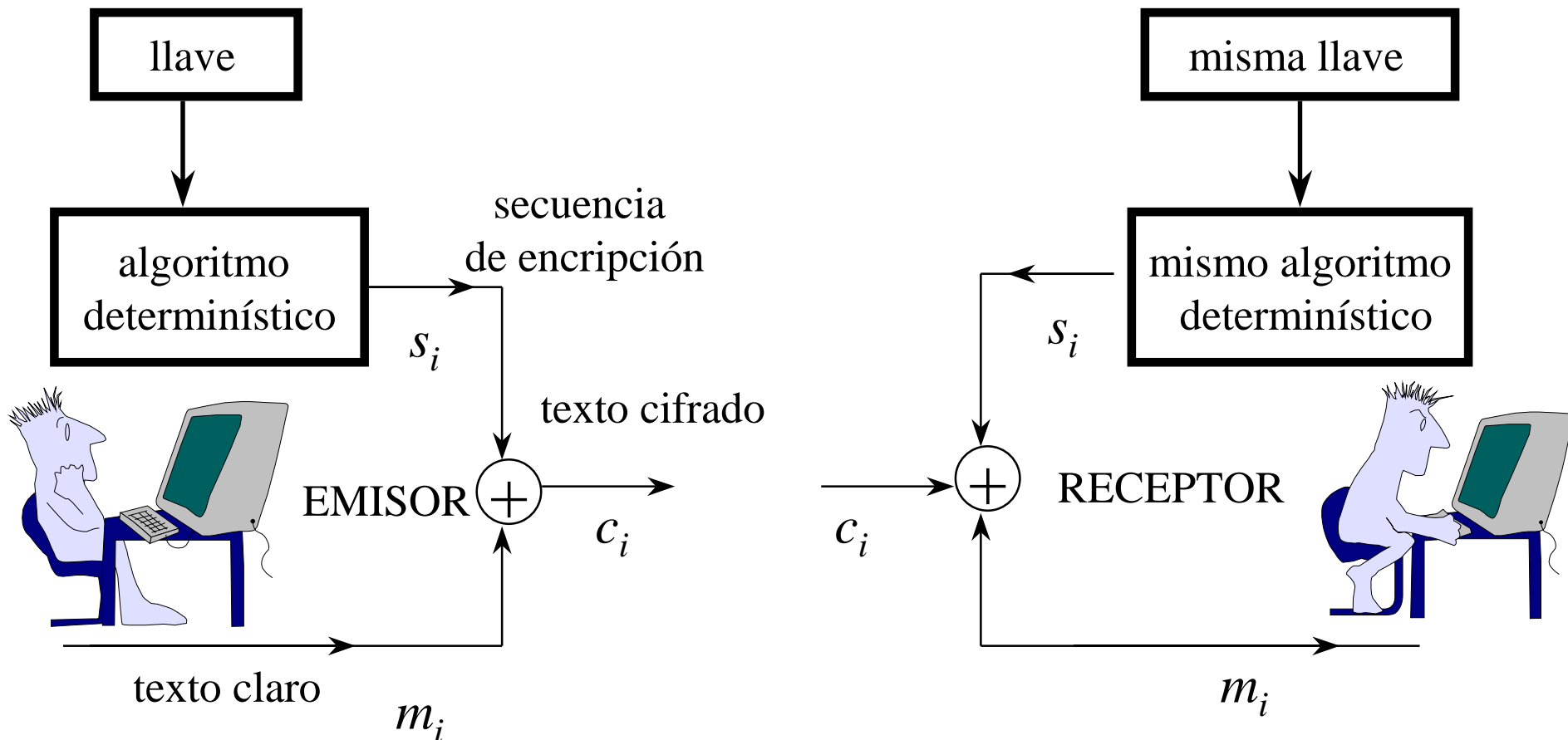
Elementos de encriptación en flujo



- Emisor A
- Una llave corta
- Un algoritmo determinístico
- Receptor B



Esquema cifrado en flujo





Generadores pseudoaleatorios



- Son algoritmos determinísticos que a partir de una llave corta (128 bits), conocida por emisor y receptor, generan simultáneamente una determinada secuencia de la longitud deseada.
- Estas secuencias nunca podrán ser auténticas secuencias aleatorias
- Son secuencias periódicas que deben ser lo más semejantes a una secuencia aleatoria



Requerimientos secuencias cifrantes



- Período
- Distribución unos y ceros
- Imprevisibilidad
- Facilidad implementación



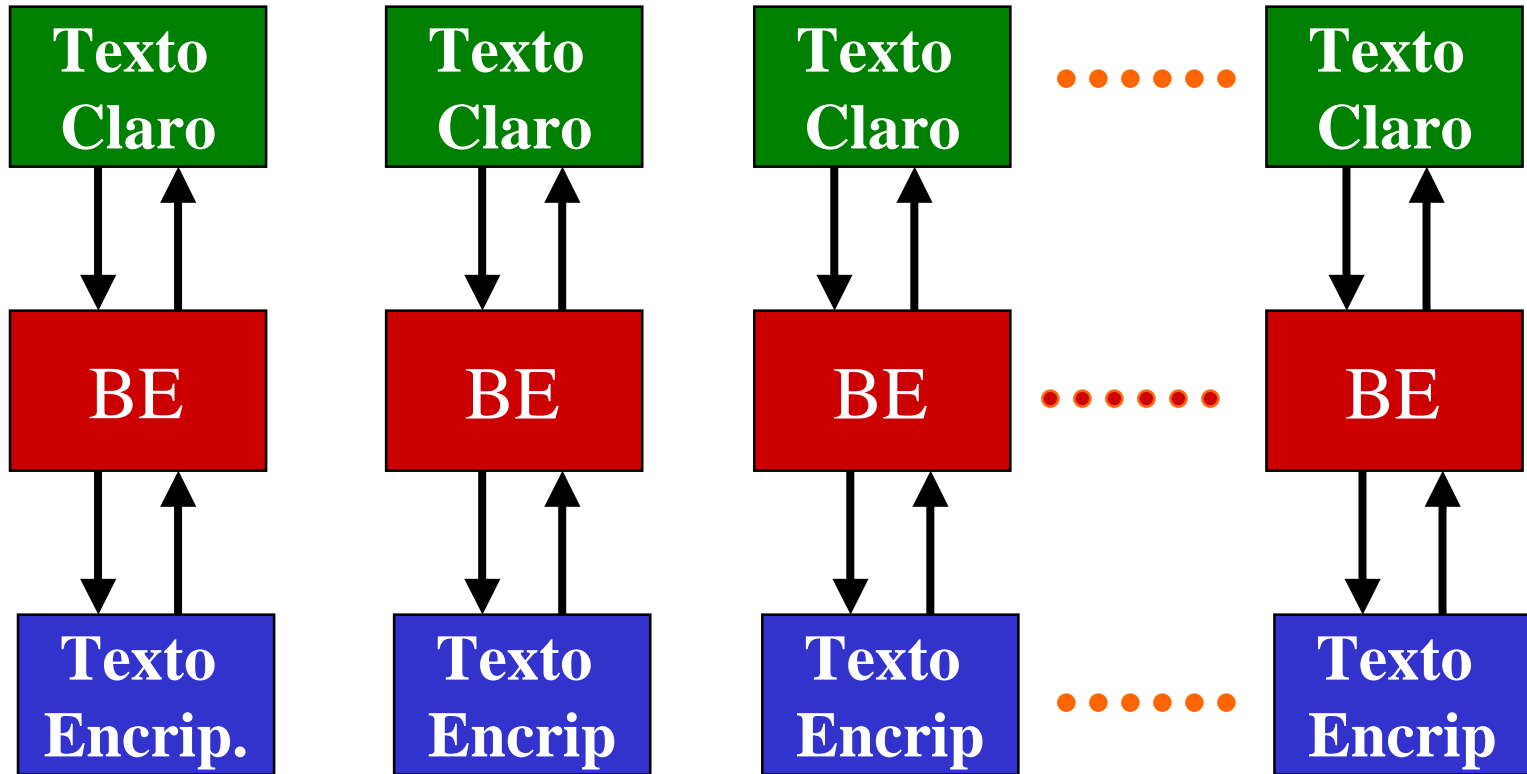
Métodos de encriptación en bloque



- Se encripta el mensaje original agrupando los símbolos en grupos (bloques) de dos o más elementos
- Modos operación de encriptación en bloque:
 - ECB: Electronic Code Book
 - CBC: Cipher Block Chaining



Esquema ECB de encriptación en bloque



ECB: Electronic Code Book



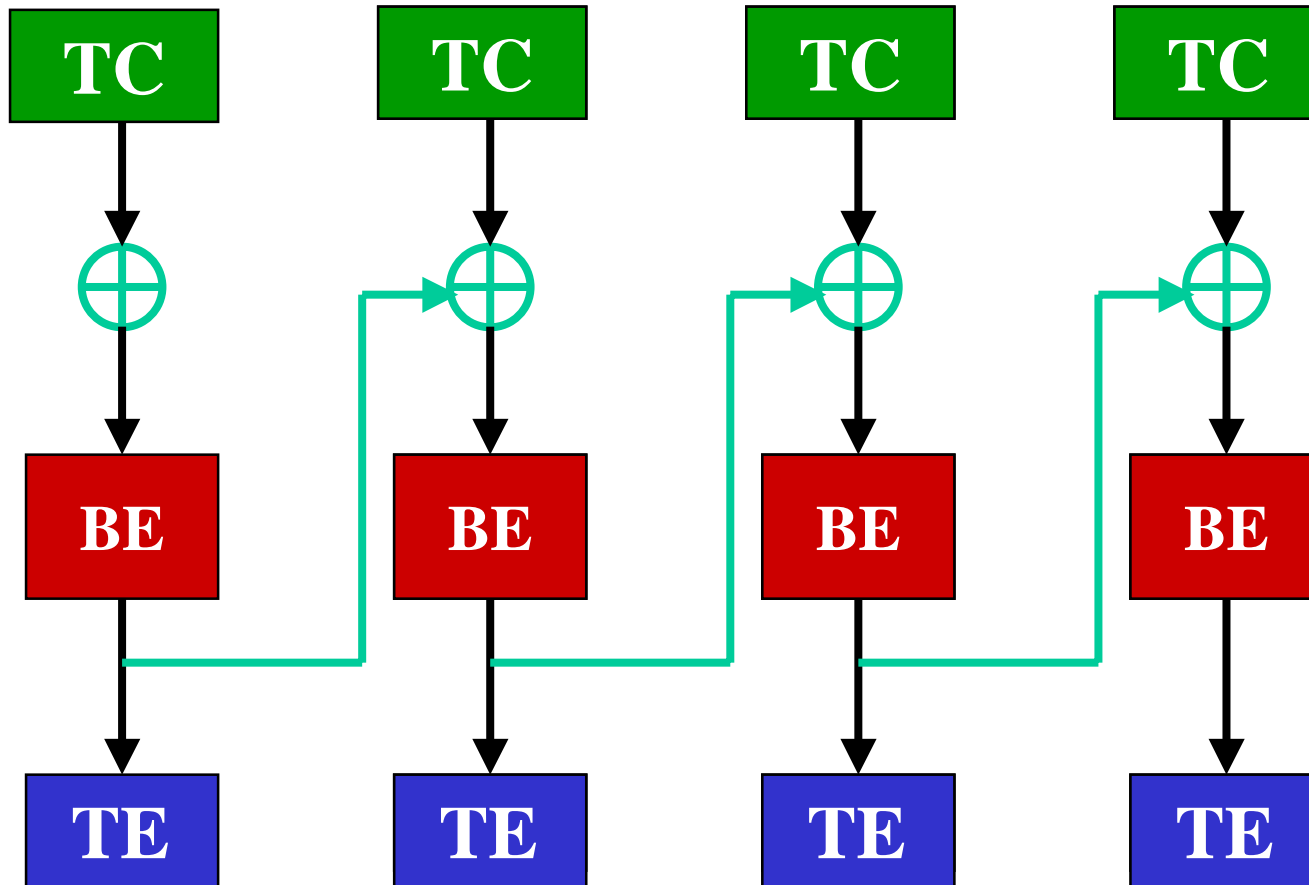
Problemas de ECB



- Bloques identicos me dan salidas identicas
- Se pueden encontrar patrones en los datos por parte de un observador externo
- Solución:
 - “barajear” los datos antes de que entren al bloque de encripción (BC)



Cipher Block Chaining (CBC) Encipción



TC: Texto Claro

TE: Texto Encriptado



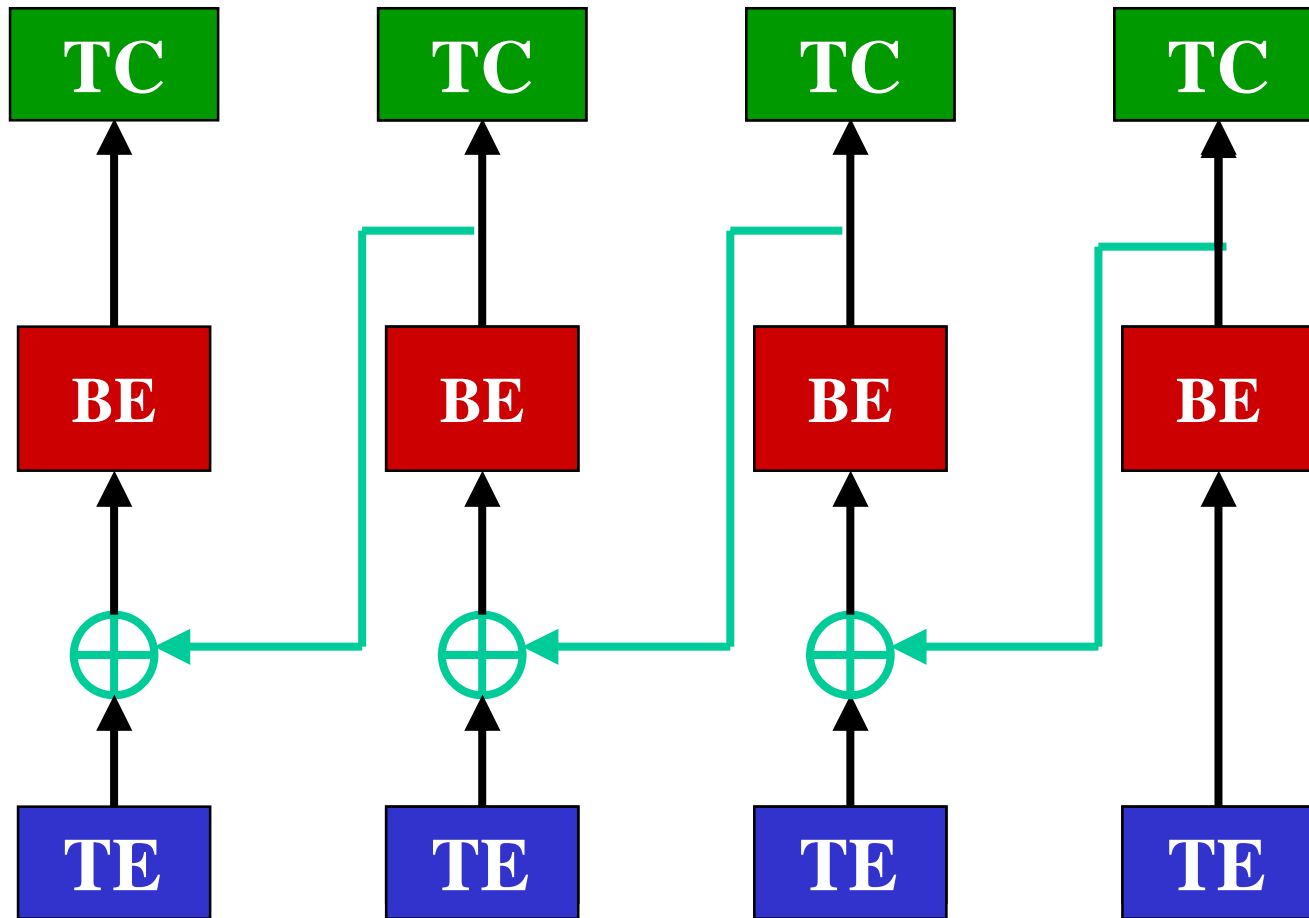
Comentarios sobre CBC



- Solución problema de bloques iguales me da salidas iguales
- Como decriptar:
 - se invierten las flechas
- Desventaja:
 - difícil si se tiene que encriptar/decriptar toda la información
 - no es posible decriptar solo una parte



Cipher Block Chaining (CBC) Decripción

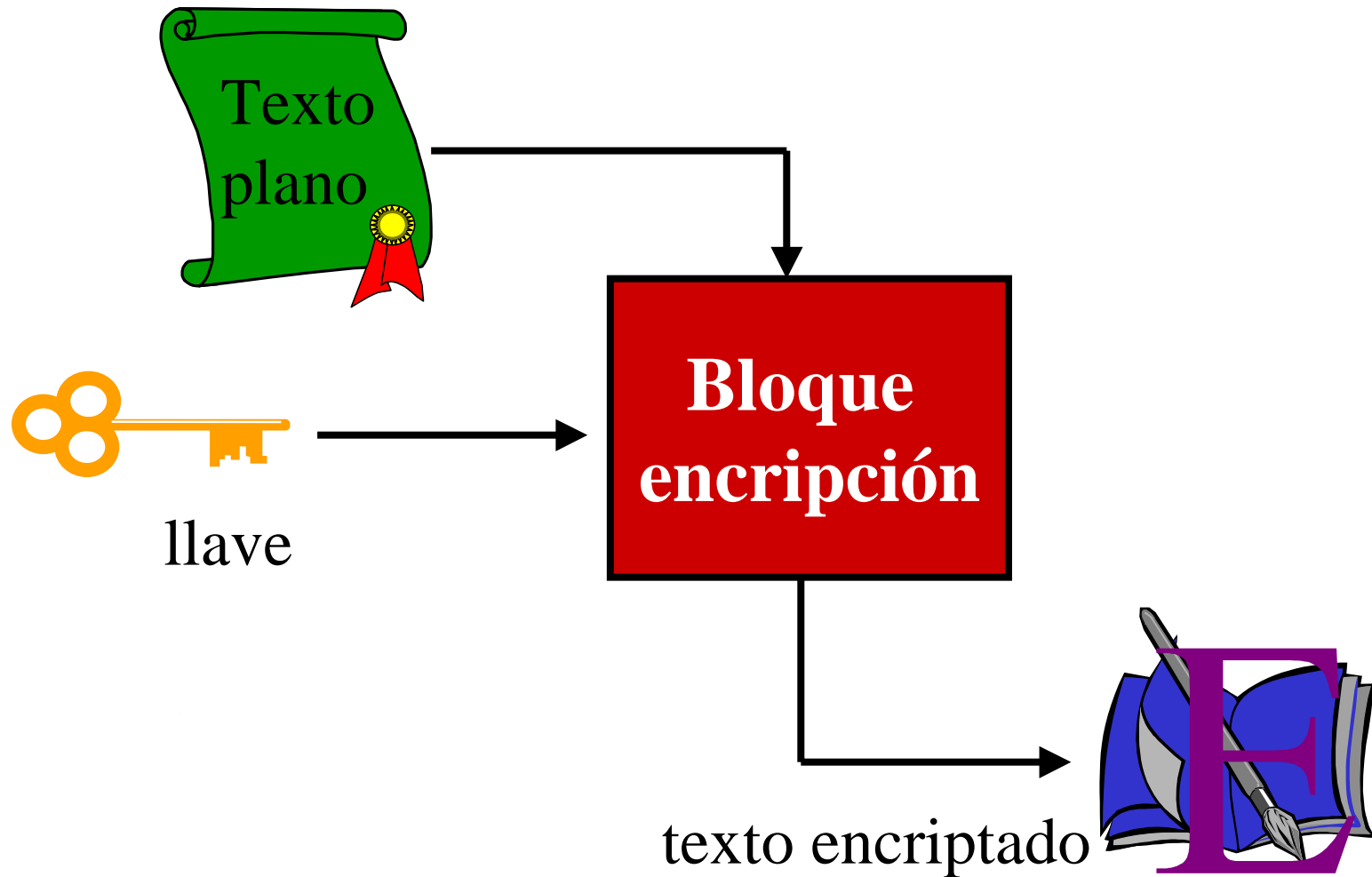


TC: Texto Claro

TE: Texto Encriptado



¿Cómo construir un block cipher?





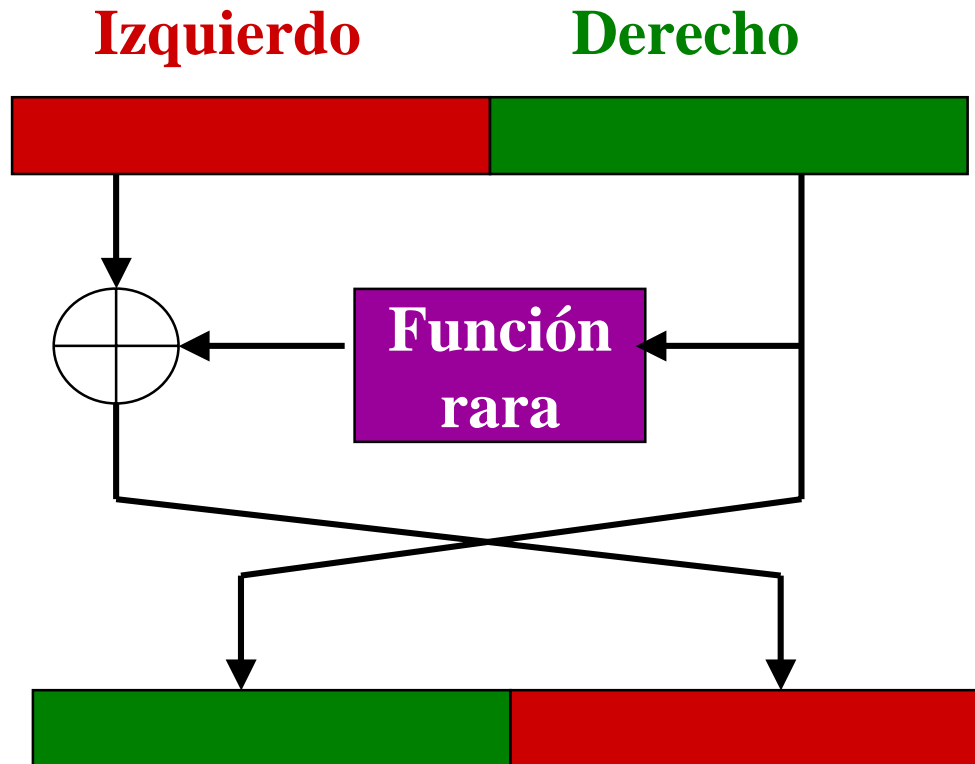
Los criptosistemas de Feistel



- Criptosistemas en los que el bloque de datos se divide en dos mitades y en cada vuelta de encriptación se trabaja alternadamente, con una de las mitades
- Ejemplos:
 - LUCIFER
 - DES
 - LOKI
 - FEAL

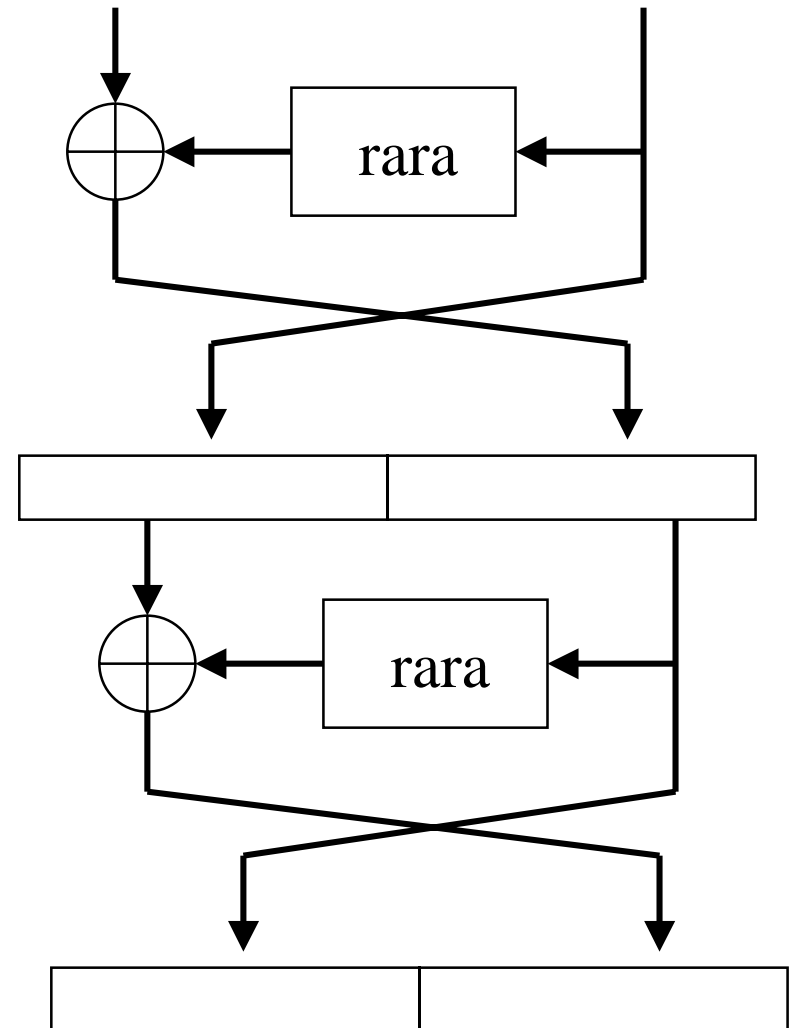
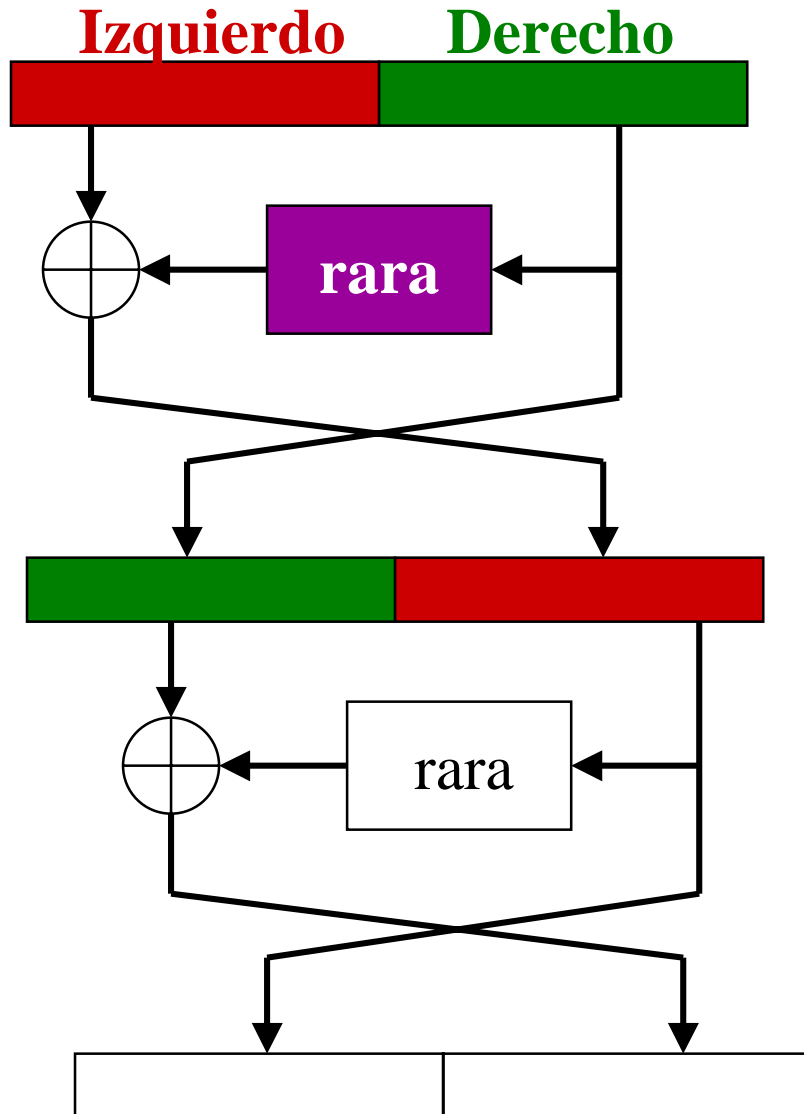


Barajeando los datos de entrada





Repitiendo





Comentarios



- Tipicamente los ciphers de Feistel son iterados unas 16 veces
- Otra opción es que la función fea de cambie en cada iteración:
 - usar sub-llaves diferentes en cada turno
- Cada iteración debil puede construir un Fiestel más fuerte



Tiempo requerido para búsqueda de llaves



Tamaño llave	Número de llaves posibles	Tiempo requido con 1 encrip/us	Tiempo requido con 10^6 encrip/us
32	$2^{32} = 4.3 \times 10^9$	2^{32} us=35.8 minutos	2.15 milisegundos
56	$2^{56} = 7.2 \times 10^{16}$	2^{56} us=1,142 años	10.01 horas
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} us= 3.4×10^{24} años	5.4×10^{18} años



DES: ejemplo de encriptación simétrica



- Data Encryption Standard
- Nació en 1974 en IBM
- Propuesto a raíz de una petición de la NIST (National Institute of Standards and Technology, USA) en 1972 y 1974.
- Inspirado de sistema LUCIFER de IBM.
- Aprobado y modificado por la NSA (National Security Agency, USA)
- NSA impuso la longitud de la llave



Características de DES



- Algoritmo cifrado en bloque y simétrico
- Longitud bloque: 64 bits
- Longitud llave: 56 bits, por lo que existen $2^{56} = 7.2 \times 10^{16}$ llaves diferentes
- Norma exige que DES se implemente mediante un circuito integrado
- En 1981 ANSI adopto el DES con el nombre de Data Encryption Algorithm
 - no exige chip y puede ser programado



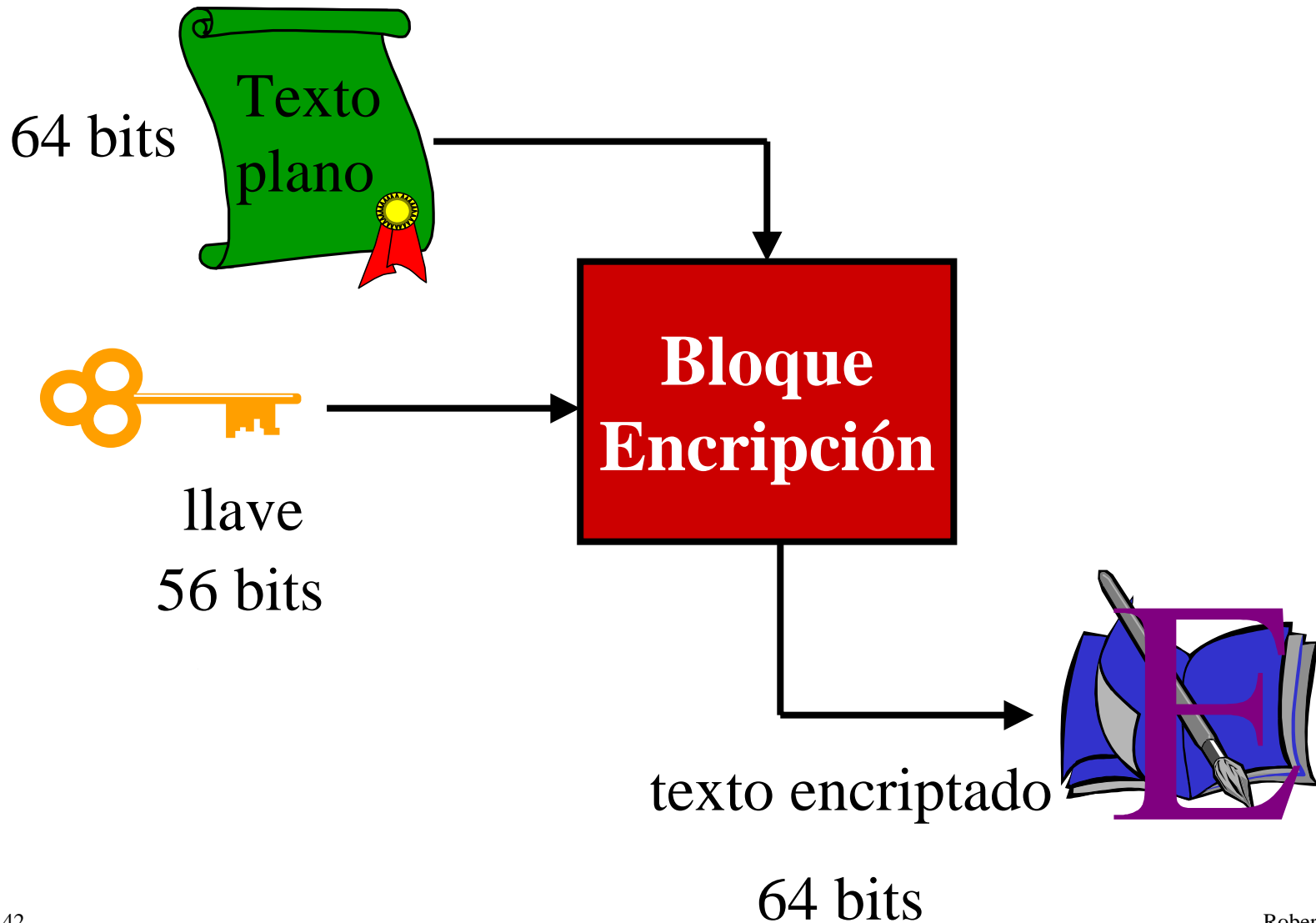
Normas y descripciones



- Como FIPS(Fed. Info. Processing Stands.)
 - Publicadas por el NTIS (Nat. Tech. Infor. Service)
 - FIPS 46-1, FIPS PUB 74, FIPS PUB 81,
 - FIPS PUB 113
- Las normas ISO
 - ISO 8372 (equivalente a ANSI X3.92-1981)
 - ISO 9797, ISO 9798 e ISO 10118



Esquema general DES





Operación de DES



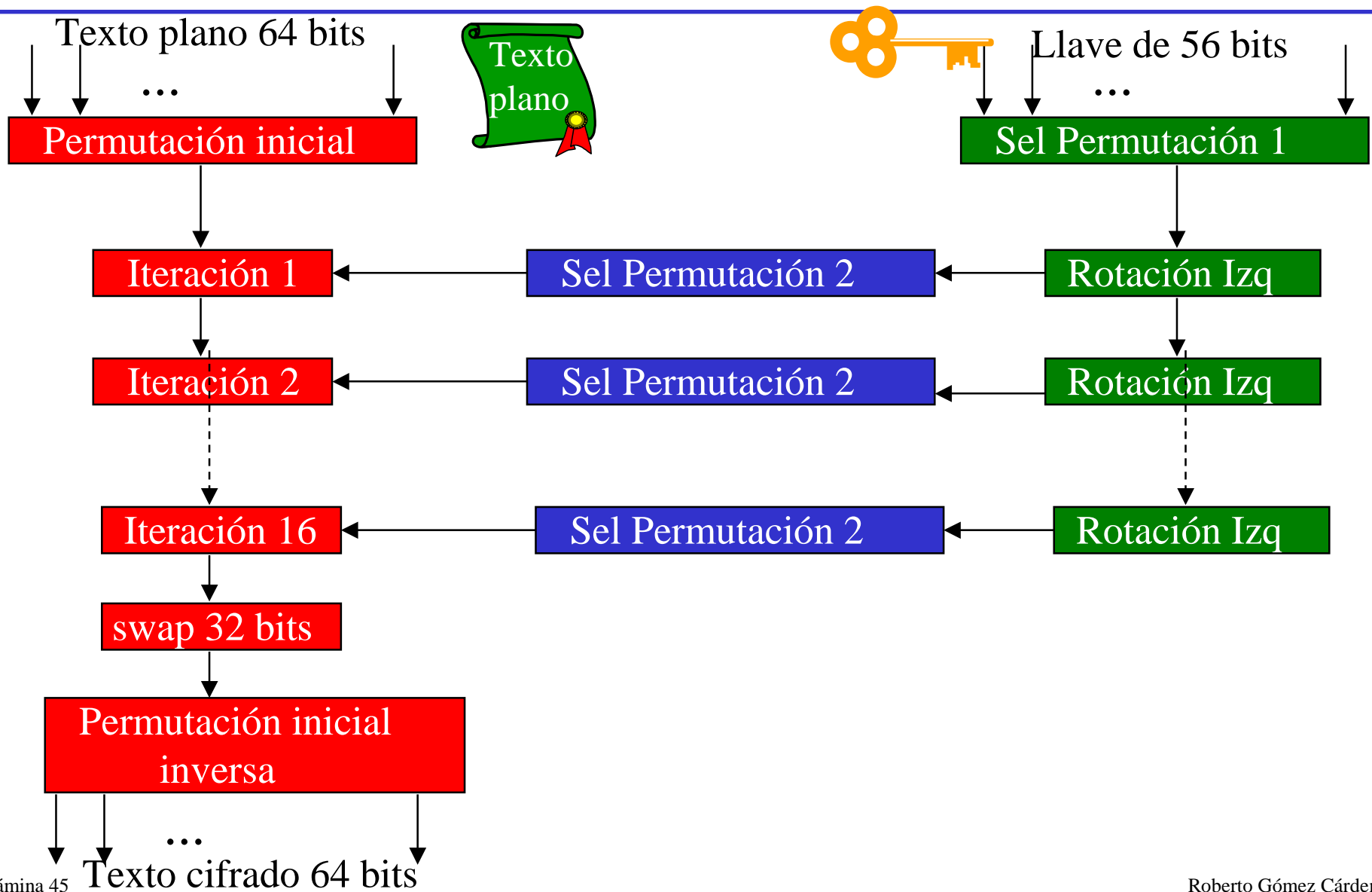
- Trabaja alternativamente sobre las dos mitades del bloque a cifrar
- Se hace una permutación inicial fija
- Se divide el bloque en dos mitades: derecha e izquierda
- Se realiza 16 veces la operación modular :
 - sumar módulo 2 la parte izquierda con una transformación $g(k_1)$ de la parte derecha, gobernada por una llave k_1



- Se intercambian las partes derecha e izquierda
- En la vuelta 16 se omite el intercambio, pero se remata el algoritmo con una permutación final que es la inversa de la final

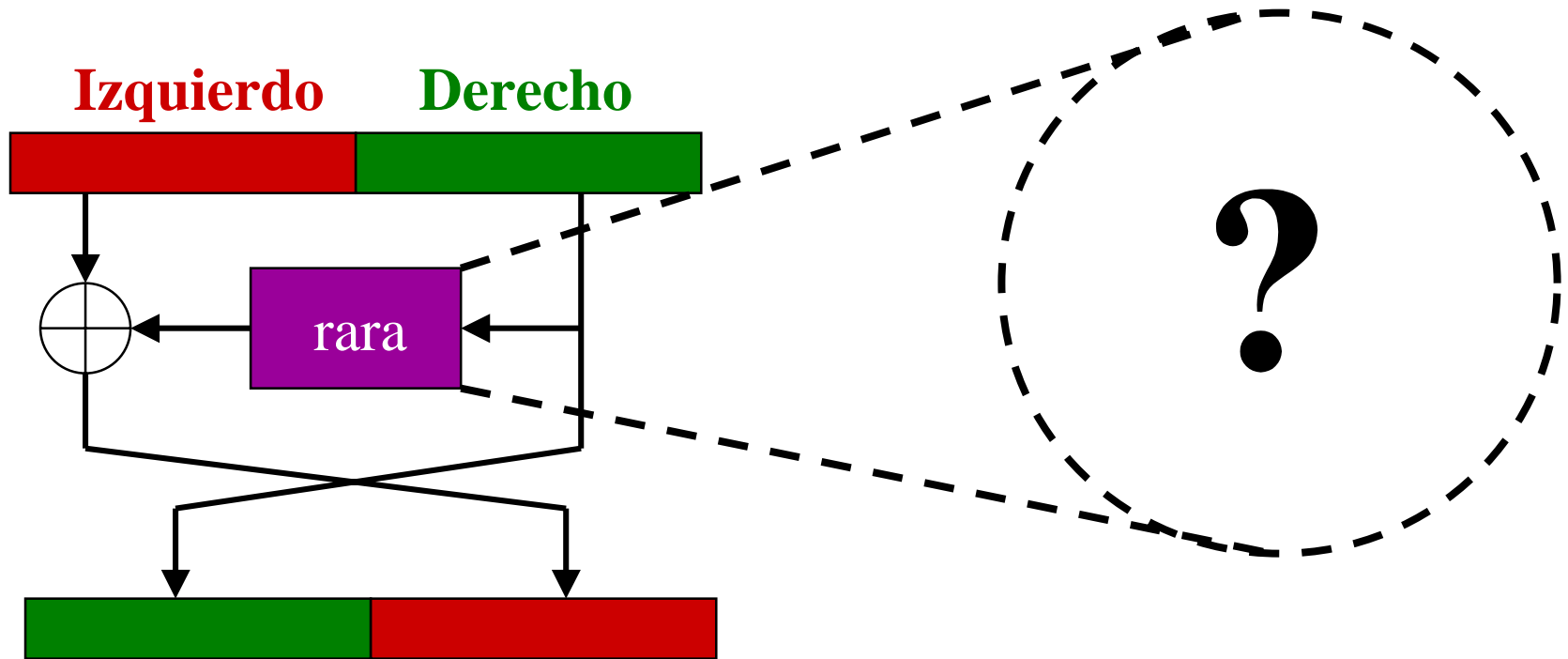


Descripción general de DES



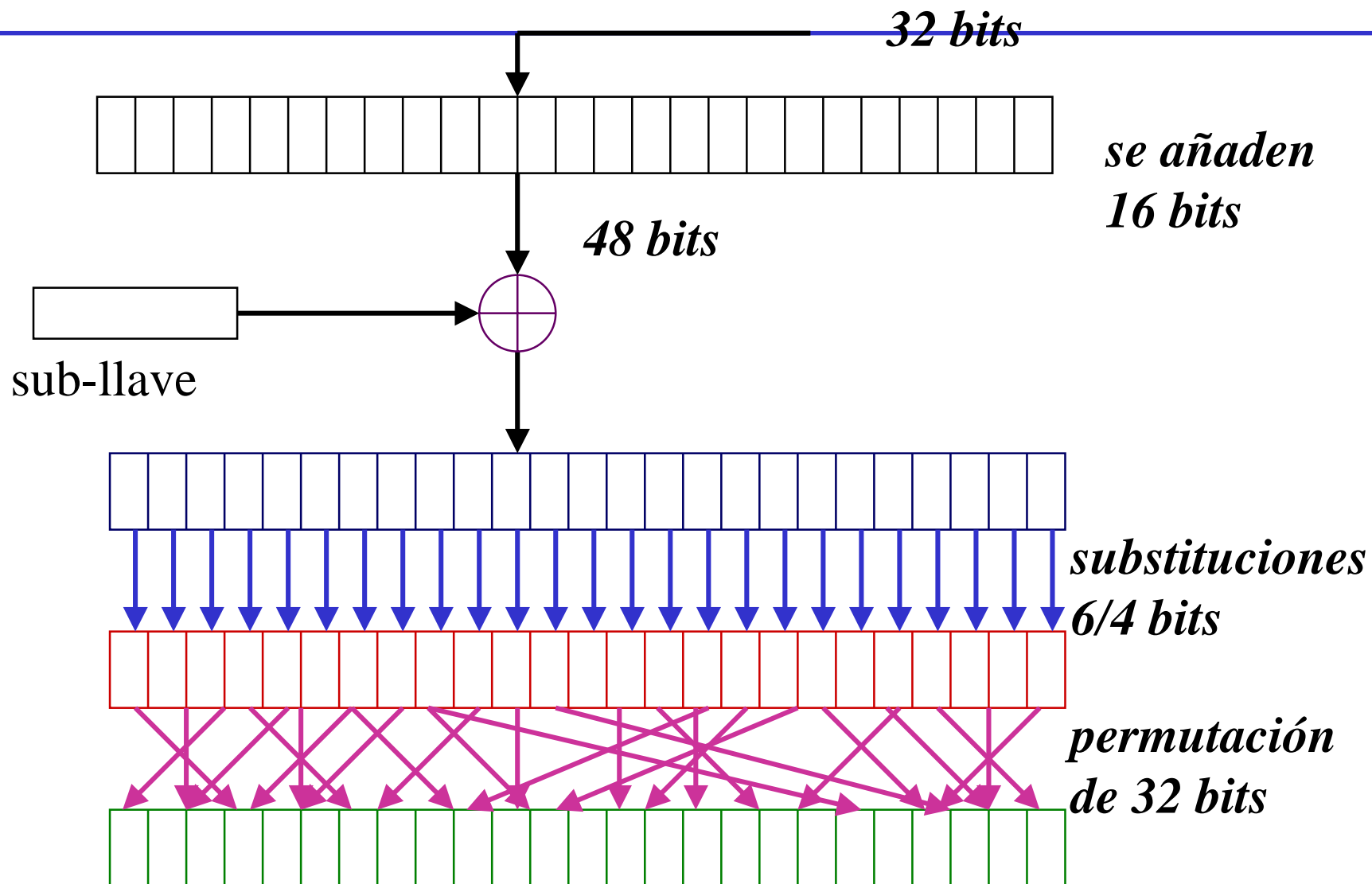


La función rara



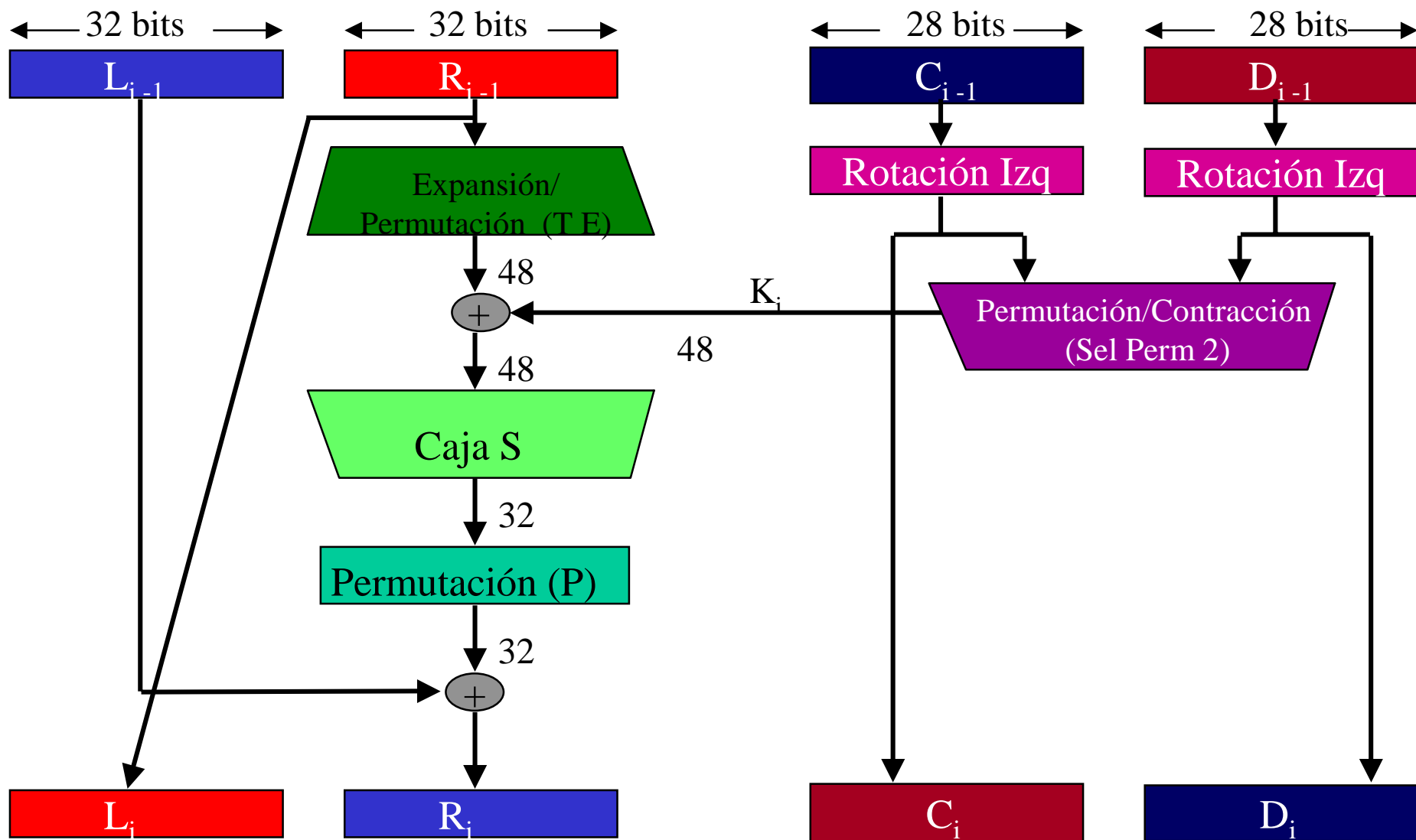


Esquema función rara





Esquema de una iteración en DES





Iteración i de DES

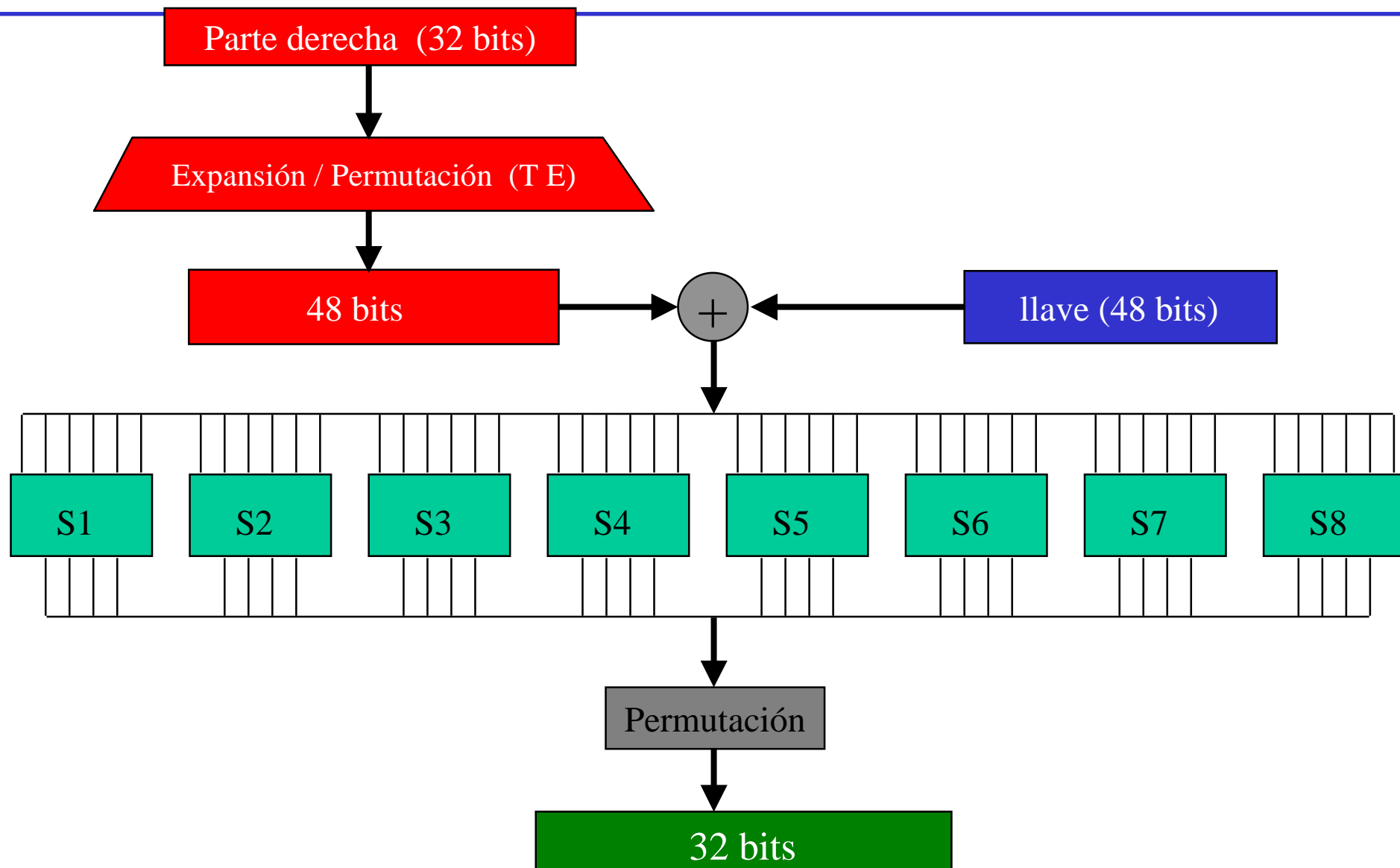


$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$



Cálculo de $f(R_{i-1}, K_i)$

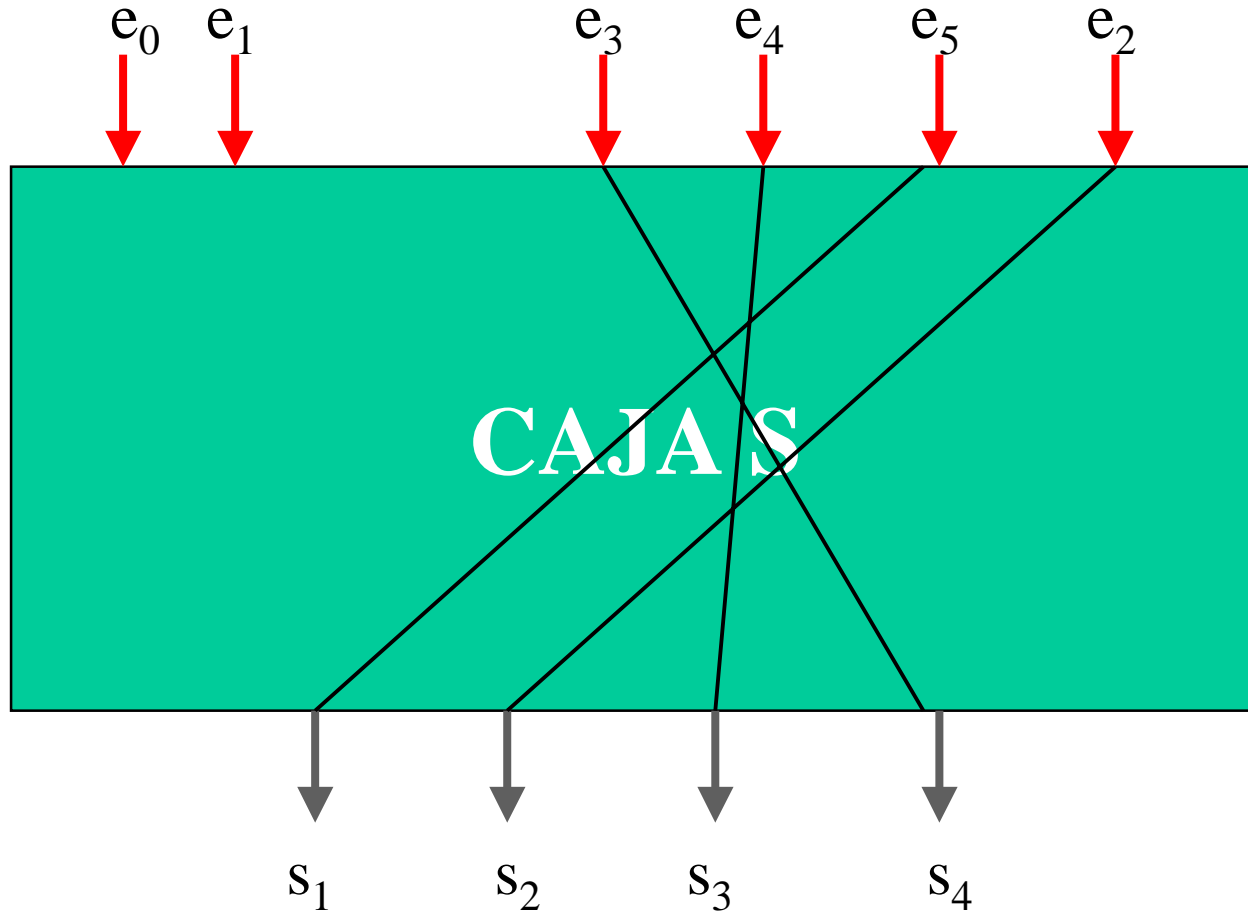




Detalle de la caja S



dos primeros bits son usados para
definir las transformaciones y
después se eliminan





Ejemplos cajas S



S_0

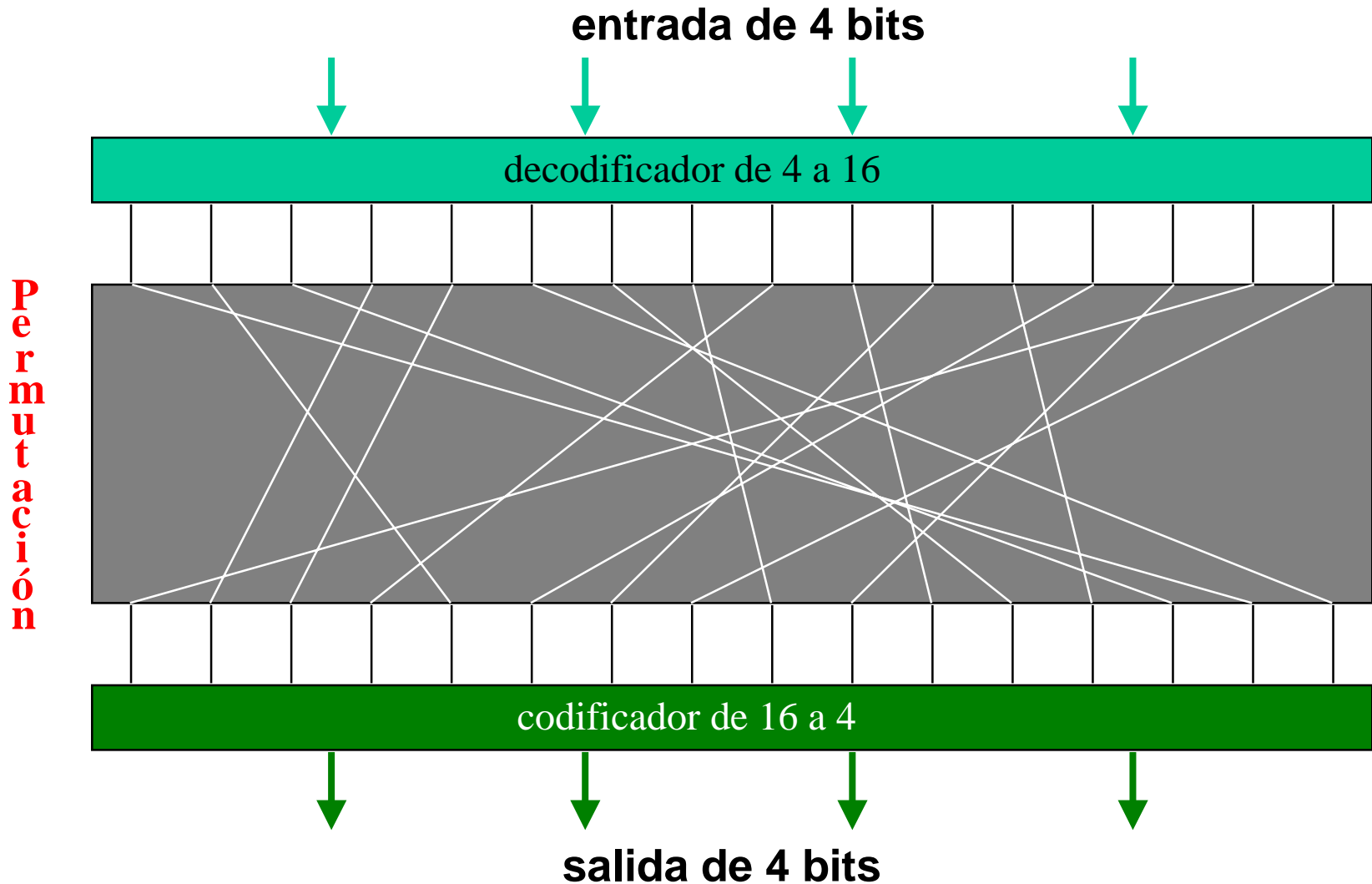
Bit	Bits 2, 3, 4, and 5 form:														
1 6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14 15
0 0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0 7
0 1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3 8
1 0	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5 0
1 1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6 13

S_1

Bit	Bits 8, 9, 10, and 11 form:														
7 12	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14 15
0 0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5 10
0 1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11 5
1 0	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2 15
1 1	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14 9



La permutación final





Consideraciones función $f(R_{i-1}, K_i)$



- Transformación $f(R_{i-1}, K_i)$ es un conjunto de operaciones
 - fabricar vector 48 bits a partir de los 32 iniciales mediante una expansión lineal

Izquierda	32	1	2	3	4	5	4	5	6	7	8	9
Centro izda.	8	9	10	11	12	13	12	13	14	15	16	17
Centro dcha.	16	17	18	19	20	21	20	21	22	23	24	25
Derecha	24	25	26	27	28	29	28	29	30	31	32	1

tabla de expansión

bits originales

bits originales



- se combina la clave local de 48 bits con el vector anterior por suma módulo 2 bit a bit, obteniéndose otro vector de 48 bits, que se divide en ocho grupos de seis bits
- cada uno de los grupos de seis bits entra en una de las ocho funciones denominadas como cajas S
- en cada caja entran seis bits pero salen solo cuatro
- *finalmente* se pasa la información por una caja P, que es una permutación lineal fija

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

tabla de permutación



Expansión de llaves



- DES maneja llaves 64 bits
- Primera operación: reducción a 56 bits, eliminando un bit de cada ocho (8,16,32,40,48,56 y 64)
- Se reordenan los bits restantes de acuerdo a la tabla siguiente:

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



- Después se generan las 16 subllaves necesarias en las 16 vueltas del algoritmo
- Cada subllave esta compuesta por 48 bits
- Durante el descifrado se toman en orden inverso al del cifrado
- Para regenerar las subllaves:
 - se divide la clave de 56 bits en dos mitades de 28
 - las mitades se rotan (permutan circularmente) a la izquierda uno o dos bits dependiendo de la vuelta:

No. vuelta	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
No. bits	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1



- Después rotaciones se vuelven a unir mitades, (teniendo 16 grupos de 56 bits)
- Se seleccionan 48 bits de cada grupo para formar finalmente las 16 subllaves, (permutación con compresión)
- Los bits elegidos son iguales para todas las subllaves y se rigen por la permutación:

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



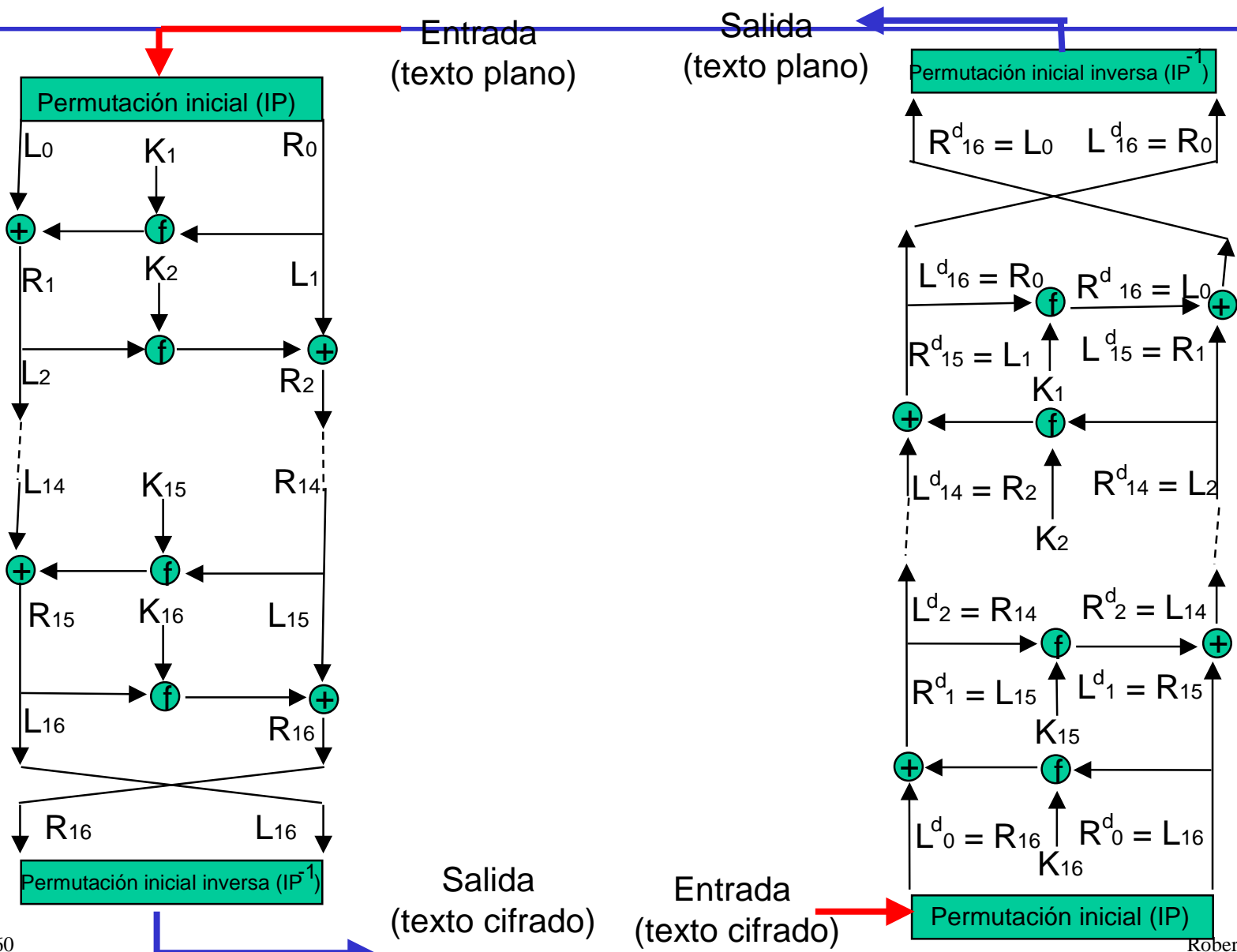
El proceso de decriptado



- El proceso de decriptado es en esencia el mismo que el de encriptado.
- La regla es la siguiente:
 - usar el texto cifrado como entrada a DES,
 - usar la llave K_i en orden inverso.



Encriptación/decipción en DES





Observaciones sobre el decriptado



- El diagrama indica que en cada paso:
 - el valor intermedio en el proceso de decriptado es igual al valor intermedio correspondiente en el proceso de encriptado, con las dos mitades invertidas.
- Dicho de otra manera:
 - sea $L_i \parallel R_i$ la salida de la i -ésima iteración del proceso de encriptado.
 - entonces, la $(16 - i)$ -ésima entrada al proceso de decriptado es $R_i \parallel L_i$



El efecto avalancha



- Una propiedad deseable de cualquier algoritmo de encriptado es que un pequeño cambio en el texto original (un bit) o en la llave produzca un cambio significativo en el texto encriptado.
- DES exhibe un efecto avalancha bastante fuerte.



Efecto de avalancha en DES



a) Cambio en texto plano		b) Cambio en llave	
Iteración	Número de bits que difieren	Iteración	Número de bits que difieren
0	1	0	0
1	6	1	2
2	21	2	14
3	35	3	28
4	39	4	32
5	34	5	30
6	32	6	32
7	31	7	35
8	29	8	34
9	42	9	40
10	44	10	38
11	32	11	31
12	30	12	33
13	30	13	28
14	26	14	26
15	29	15	34
16	34	16	35



Efecto de avalancha en DES



a) Cambio en texto plano

b) Cambio en llave

Iteración	Número de bits que difieren	Iteración	Número de bits que difieren
0	1	0	0
1	6	1	2
2	21	2	14
3	35	3	28
4	39	4	32
5	34	5	30
6	32	6	32
7	31	7	35
8	29	8	34
9	42	9	40
10	44	10	38
11	32	11	31
12	30	12	33
13	30	13	28
14	26	14	26
15	29	15	34
16	34	16	35



The DES Key Search Project



- Un máquina construida por Cryptography Research, Advanced Wireless Technologies y EFF, ha demostrado una búsqueda rápida de llaves para DES.
- El DES Key Search Project diseñó hardware y software para buscar 90 billones de llaves por segundo, determinando la llave y ganando \$10,000 en el concurso RSA DES después de una búsqueda de 56 horas.
- Referencia:
 - <http://www.cryptography.com/des/despictures/index.htm>



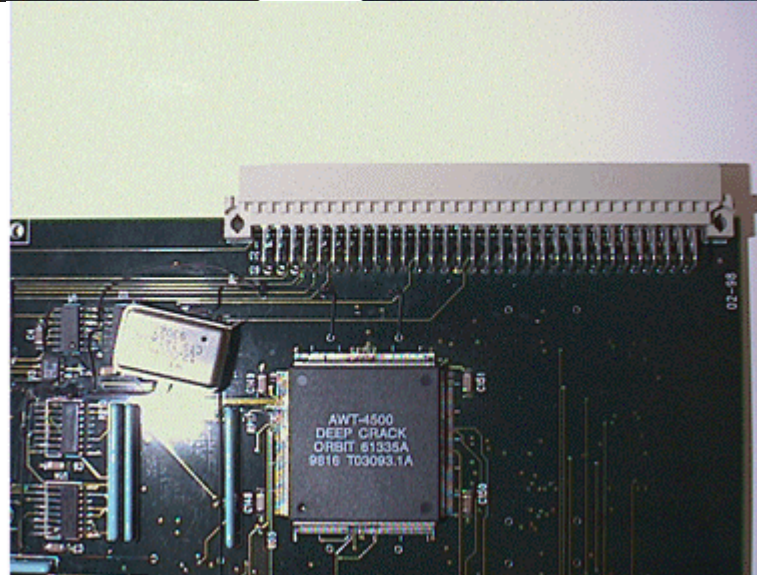
Detalles de la máquina



- Basda en un proceso de búsqueda y chequeo que puede encontrar llaves aun cuando se conozaca poco acerca del texto plano.
- Cada chip procesa dos criptogramas separados y contiene un vector de 256 bits especificando cuales bytes pueden aparecer en el texto plano.
- La máquina se encuentra “hospedada” en cabinas recicladas SUN-2 y consiste de 27 tarjetas que almacenan 1800 chips
 - cada chip contiene 24 unidades de búsqueda, los que independientemente buscan a través de un rango de llaver, filtrando aquellas que no paasn el criterio para los dos criptogramas



Imágenes de la máquina





Mejoras a DES



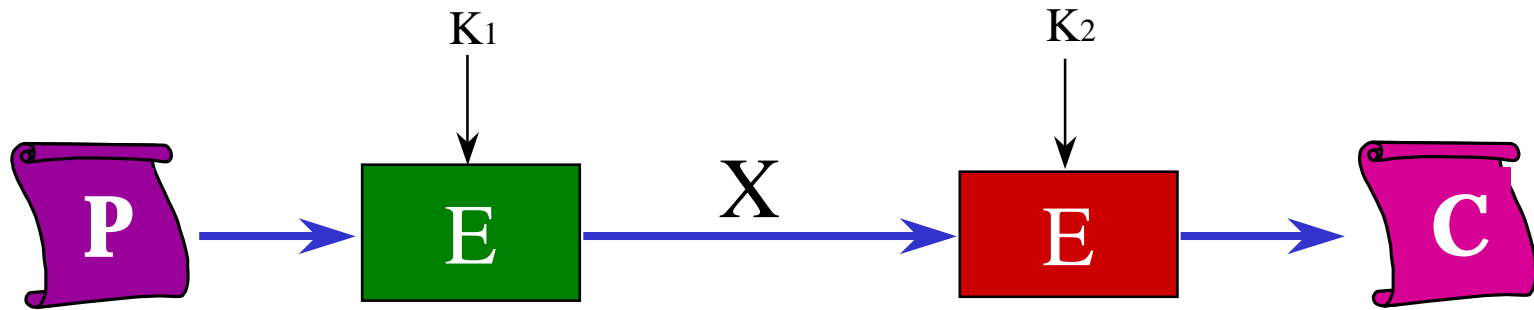
- Debido a las vulnerabilidades que presenta DES contra ataques de fuerza bruta, se han buscado alternativas.
- Una de estas es realizar un múltiple encriptado con DES usando más de una llave.



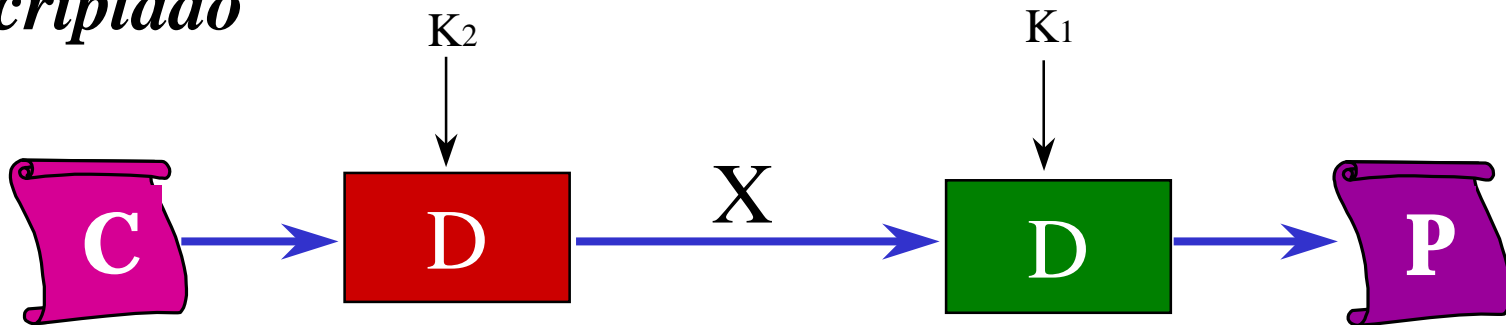
Doble DES



Encriptado



Decriptado

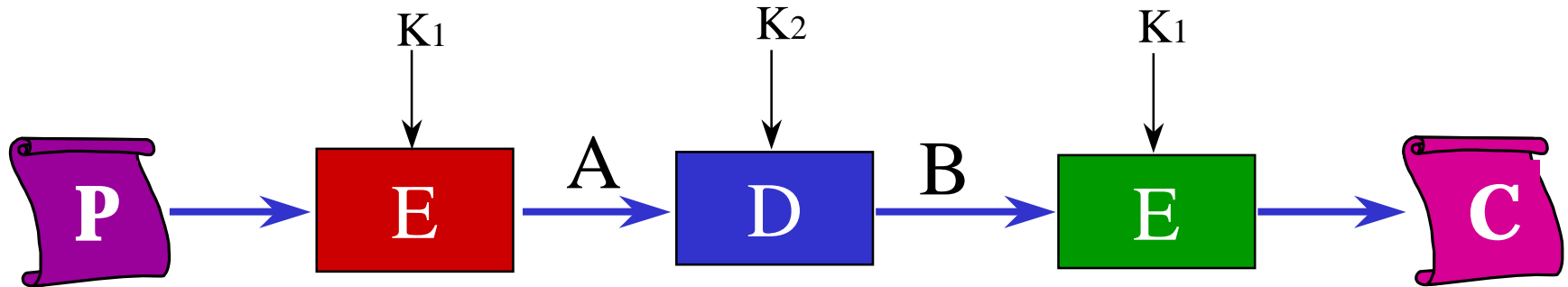




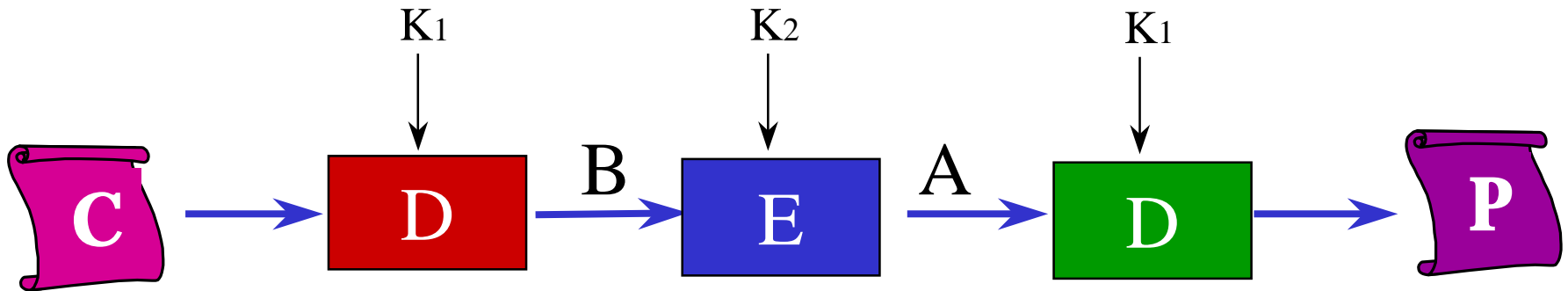
Triple DES



Encriptado



Decriptado





Substituto DES: AES



- En 1997 la NIST anuncia el sustituto de DES: AES (Advanced Encryption Standard)
- Referencia: <http://csrc.nist.gov/encryption/aes/>
- Candidatos (al 20-abril- 2000):
 - MARS (IBM)
 - RC6 (Laboratorios RSA)
 - **Rijndael (J. Daemen y V. Rijmen) !!!! (2.10.2000)**
 - Serpent (R. Anderson, E.Biham, L.Knudsen)
 - Twofish (B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson)



Otros algoritmos llave simétrica



- Twofish
- Blowfish
- IDEA
- RC2, RC4 y RC5
- NewDES
- Feal
- SKIPJACK
- MMB
- GOST
- CRAB 342
- CAST
- SAFER
- 3-WAY
- FEAL
- REDOC
- LOKI
- MADRYGA
- Lucifer
- Khufu and Khafre
- CA-1.1



Desventajas llave secreta



- Distribución de llaves
 - usuarios tienen que seleccionar llave en secreto antes de empezar a comunicarse
- Manejo de llaves
 - red de n usuarios, cada pareja debe tener su llave secreta particular, i.e. $n(n-1)/2$ llaves
- Sin firma digital
 - no hay posibilidad , en general, de firmar digitalmente los mensajes



Criptosistema Diffie Hellman

Criptosistema intercambio llaves



Diffie-Hellman



- Primer algoritmo de llave pública (1976)
 - Williamson del CESG¹ UK, publica un esquema idéntico unos meses antes en documento clasificado
 - asegura que descubrió dicho algoritmo varios años antes
- Varios productos comerciales utilizan esta técnica de intercambio de llaves.
- Propósito del algoritmo
 - permitir que dos usuarios intercambien una llave de forma segura
 - algoritmo limitado al intercambio de llaves
- Basado en la dificultad para calcular logaritmos discretos



Algoritmo de Diffie-Hellman



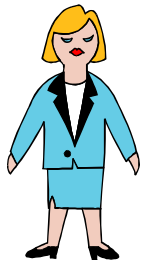
1. Los dos usuarios A y B seleccionan públicamente un grupo multiplicativo finito, G , de orden n y un elemento de G
2. A genera un número aleatorio X_a , calcula Y_a en G y transmite este elemento a B
3. B genera un número aleatorio X_b , calcula Y_b en G y transmite este elemento a A
4. A recibe Y_b y calcula $(Y_b)^{X_a}$ en G
5. B recibe Y_a y calcula $(Y_a)^{X_b}$ en G



Esquema Diffie Hellman



Elementos globales públicos: q (numero primo) y α ($\alpha < q$)



A



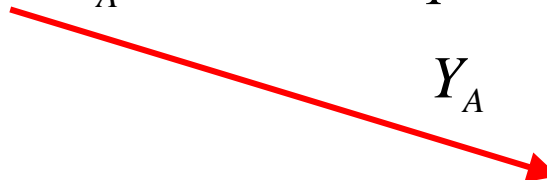
La llave de A y B es K



B

Selecciona val. priv: X_A ($X_A < q$)

Calcula valor pub: $Y_A = \alpha^{X_A} \bmod q$

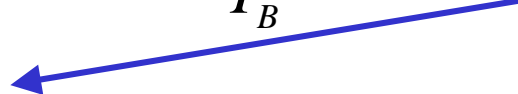


Y_A

Selecciona val. priv: X_B ($X_B < q$)

Calcula valor pub: $Y_B = \alpha^{X_B} \bmod q$

Y_B



Generando llave secreta A

$$K = (Y_B)^{X_A} \bmod q$$

Generando llave secreta B

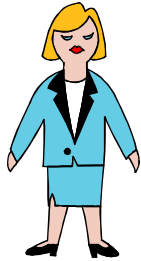
$$K = (Y_A)^{X_B} \bmod q$$



Ejemplo Diffie Hellman



Elementos globales públicos: $q = 53$ $\alpha = 2$ ($2 < 53$)



A



La llave de A y B es 21



B

Selecciona val. priv: $X_A = 29$ ($29 < 53$)

Calcula valor pub: $Y_A = 2^{29} \bmod 53$
 $= 45 \bmod 53$

Y_A (45)

Selecciona val. priv: $X_B = 19$ ($19 < 53$)

Calcula valor pub: $Y_B = 2^{19} \bmod 53$
 $= 12 \bmod 53$

Y_B (12)

Generando llave secreta A

$$K = 12^{29} \bmod 53 = 21 \bmod 53$$

Generando llave secreta B

$$K = 45^{19} \bmod 53 = 21 \bmod 53$$



Continuación ejemplo



- La clave privada o la información secreta que comparten ahora A y B es 21
- Un escucha, S, conoce del protocolo anterior:
 - Z_{53}^* , 2, 45 y 12
 - no puede conocer que la información secreta compartida por A y B es 21



Criptosistemas de llave pública



Características y ejemplos





Background



- Concepto de llave pública fue inventado por Whitfield Diffie y Martin Hellman e independientemente por Ralph Merkle.
- Contribución fue que las llaves pueden presentarse en pares.
- Concepto presentado en 1976 por Diffie y Hellman.
- Desde 1976 varios algoritmos han sido propuestos, muchos de estos son considerados seguros, pero son impracticos.
- Algunos solo son buenos para distribución de llaves.



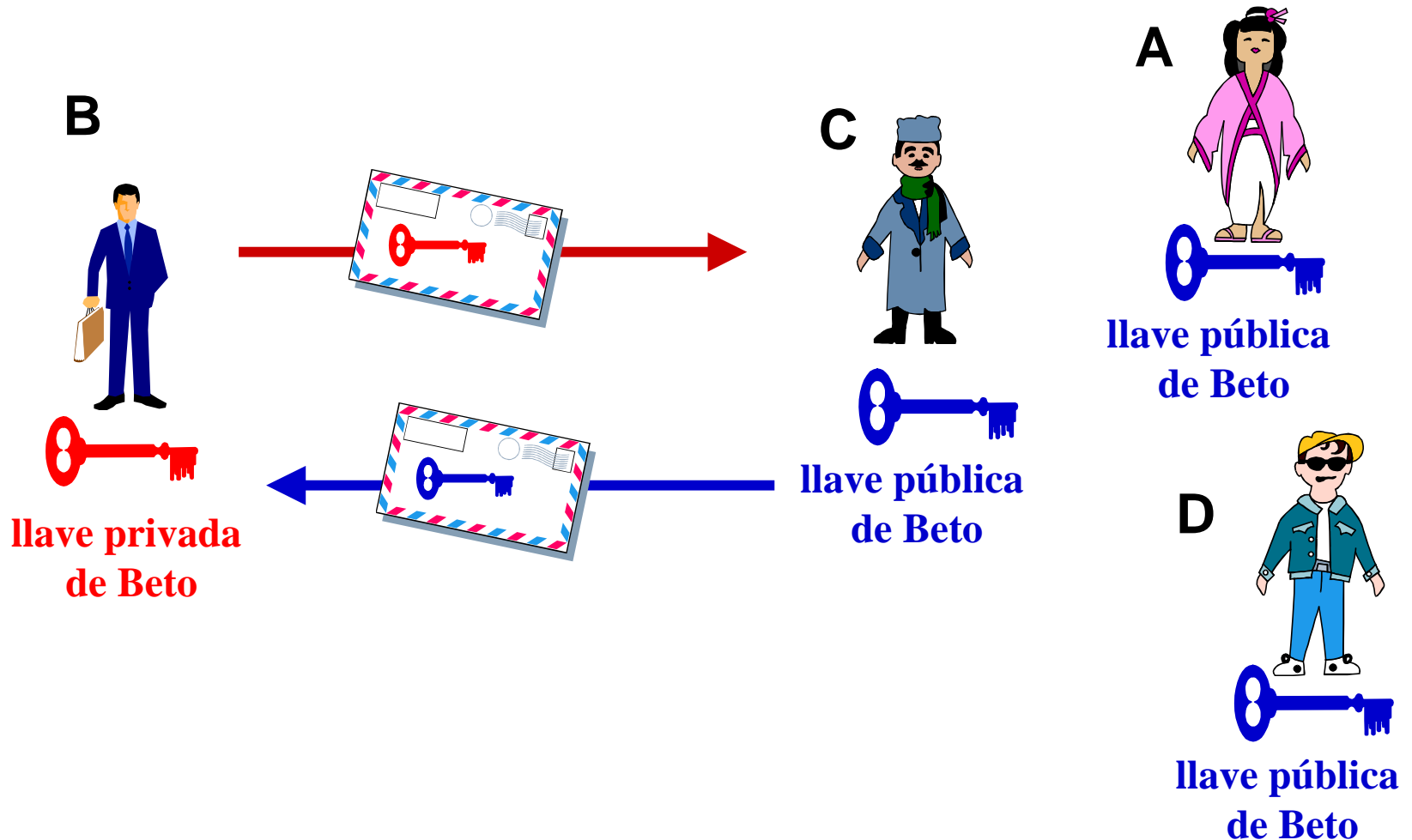
Background



- Otros solo son buenos para encriptación.
- Algunos más solo son buenos para firmas digitales.
- Solo tres algoritmos son buenos para encriptación y firmas digitales:
 - RSA,
 - ElGamal
 - Rabin.
- Los tres algoritmos son más lentos que los algoritmos simétricos.



Criptograma llave pública (asimétrico)





Función Unidireccional



- Una función One-Way Function (OWF):
 - $f: M \rightarrow C$
 - es fácil calcular $f(m) = C$
 - es difícil calcular $f^{-1}(c) = m$
- Función unidireccional puerta trasera (Trapdoor One-way Function TOF) si puede ser invertida fácilmente cuando se conoce alguna información adicional extra
- Dicha información se conoce como *puerta trasera*



Uso de TOFs en la práctica



- No se ha demostrado existencia TOFs
- Hay dos funciones candidatas a serlo
 - producto de números enteros, cuya inversa es la factorización del número obtenido
 - la exponenciación discreta, cuya inversa es el logaritmo discreto
- Las dos funciones son fáciles de calcular, mientras que sus inversas no lo son



- En el caso de las funciones anteriores
 - dado un número n , es difícil determinar su descomposición en factores primos
 - dados a y b es difícil calcular x de modo que $a^x = b$
- La primera se utiliza en el criptosistema RSA, mientras que la segunda es la base del criptosistema de ElGamal



Aritmética Modular



- Utiliza enteros no negativos
- Realiza operaciones aritméticas ordinarias (suma, multiplicación).
- Reemplaza su resultado con el residuo cuando se divide entre n .
- El resultado es modulo n o *mod* n .



Ejemplo suma modular



- $5 + 5 = 10 \bmod 10 = 0$
- $3 + 9 = 12 \bmod 10 = 2$
- $2 + 2 = 4 \bmod 10 = 4$
- $9 + 9 = 18 \bmod 10 = 8$



Tabla suma modular



+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	0
2	2	3	4	5	6	7	8	9	0	1
3	3	4	5	6	7	8	9	0	1	2
4	4	5	6	7	8	9	0	1	2	3
5	5	6	7	8	9	0	1	2	3	4
6	6	7	8	9	0	1	2	3	4	5
7	7	8	9	0	1	2	3	4	5	6
8	8	9	0	1	2	3	4	5	6	7
9	9	0	1	2	3	4	5	6	7	8



Encriptación usando suma modular



- Suma modulo 10 puede usarse como esquema de encriptación de dígitos.
- Encriptación:
$$\text{dígito} + \langle \text{constante} \rangle \bmod 10$$
- Se mapea cada dígito decimal a uno diferente de tal forma que es reversible.
- La constante es la llave secreta
- Decripción:
$$\text{dígito} - \langle \text{constante} \rangle \bmod 10$$

si el resultado es menor a cero \Rightarrow sumar 10



Ejemplo encriptación suma modular

- Llave secreta: 5
- Encriptación:
$$7 + 5 = 12 \bmod 10 = 2$$
$$8 + 5 = 13 \bmod 10 = 3$$
$$3 + 5 = 8 \bmod 10 = 8$$
- Decripción:
$$2 - 5 = -3 + 10 = 7$$
$$3 - 5 = -2 + 10 = 8$$
$$8 - 5 = 3$$



Encriptación con inversa aditiva de x



- Aritmética regular:
 - substraer x puede hacerse sumando $-x$
- Inversa aditiva de x
 - número que se le tiene que sumar a x para obtener 0
- Por ejemplo:
 - inversa aditiva de 4 es 6
 - aritmética mod 10: $4 + 6 = 10 \bmod 10 = 0$
- Si la llave secreta es 4:
 - para encriptar se añade 4 mod 10
 - para decriptar se añade 6 mod 10



Ejemplo encriptación inversa aditiva



- Llave secreta: 4
- Encriptación:
 - $7 + 4 \bmod 10 = 11 \bmod 10 = 1$
 - $8 + 4 \bmod 10 = 12 \bmod 10 = 2$
 - $3 + 4 \bmod 10 = 7 \bmod 10 = 7$
- Decipción:
 - $1 + 6 \bmod 10 = 7 \bmod 10 = 7$
 - $2 + 6 \bmod 10 = 8 \bmod 10 = 8$
 - $7 + 6 \bmod 10 = 13 \bmod 10 = 3$



Llave encriptación:

4



Llave decriptación:

6

¿Es posible decriptar si solo se conoce la llave de encriptación?



Encriptación con multiplicación modular



- Multiplicación modular: mismo principio que la suma:
 - $7 * 4 \bmod 10 = 8$
 - $3 * 9 \bmod 10 = 7$
 - $2 * 2 \bmod 10 = 4$
 - $9 * 9 \bmod 10 = 1$
- Diferencia:
 - no es posible aplicar el mismo principio de encriptación que en la suma



Tabla multiplicación modular



*	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	0	2	4	6	8
3	0	3	6	9	2	5	8	1	4	7
4	0	4	8	2	6	0	4	8	2	6
5	0	5	0	5	0	5	0	5	0	5
6	0	6	2	8	4	0	6	2	8	4
7	0	7	4	1	8	5	2	9	6	3
8	0	8	6	4	2	0	8	6	4	2
9	0	9	8	7	6	5	4	3	2	1



¿Cómo decriptar?



- Inverso multiplicativo
 - aritmética normal: inverso de x es: $x^{-1} = 1/x$
 - número por el cual se debe multiplicar x para obtener el valor de 1: número fraccionario
 - en aritmética modular solo hay enteros
- Entonces:
 - los números $\{1,3,7,9\}$ tienen inversos multiplicativos, por lo que son los que se van a usar como llaves



¿Cuales números pueden usarse como llave?



*	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	0	2	4	6	8
3	0	3	6	9	2	5	8	1	4	7
4	0	4	8	2	6	0	4	8	2	6
5	0	5	0	5	0	5	0	5	0	5
6	0	6	2	8	4	0	6	2	8	4
7	0	7	4	1	8	5	2	9	6	3
8	0	8	6	4	2	0	8	6	4	2
9	0	9	8	7	6	5	4	3	2	1

- Se debe escoger con cuidado el multiplicador
- La llave puede ser 1,3,7 o 9 ya que realizan substitución uno a uno de los dígitos
- Problema: ¿Cómo decriptar?



Características multiplicación modular



Encriptando con 5

- $1 * 5 \bmod 10 = 5$
- $2 * 5 \bmod 10 = 0$
- $3 * 5 \bmod 10 = 5$
- $4 * 5 \bmod 10 = 0$
- $5 * 5 \bmod 10 = 5$
- $6 * 5 \bmod 10 = 0$
- $7 * 5 \bmod 10 = 5$
- $8 * 5 \bmod 10 = 0$
- $9 * 5 \bmod 10 = 5$

Encriptando con 8

- $1 * 8 \bmod 10 = 8$
- $2 * 8 \bmod 10 = 6$
- $3 * 8 \bmod 10 = 4$
- $4 * 8 \bmod 10 = 2$
- $5 * 8 \bmod 10 = 0$
- $6 * 8 \bmod 10 = 8$
- $7 * 8 \bmod 10 = 6$
- $8 * 8 \bmod 10 = 4$
- $9 * 8 \bmod 10 = 2$



Ejemplos inversos multiplicativos



- Ejemplo 1:
 - 7 es el inverso multiplicativo de 3
 - $3 \times 7 \bmod 10 = 21 \bmod 10 = 1$
 - Entonces: encriptación con 3 y decriptación con 7

Encriptación

$$7 * 3 \bmod 10 = 1$$

$$8 * 3 \bmod 10 = 4$$

$$3 * 3 \bmod 10 = 9$$

Decriptación

$$1 * 7 \bmod 10 = 7$$

$$4 * 7 \bmod 10 = 8$$

$$9 * 7 \bmod 10 = 3$$



Otro ejemplo



- Ejemplo 2:
 - 9 es su propio inverso multiplicativo
 $9 \times 9 \bmod 10 = 81 \bmod 10 = 1$
 - Entonces: encriptación con 9 y decriptación con 9

Encriptación

$$7 * 9 \bmod 10 = 3$$

$$8 * 9 \bmod 10 = 2$$

$$3 * 9 \bmod 10 = 7$$

Decriptación

$$3 * 9 \bmod 10 = 7$$

$$2 * 9 \bmod 10 = 8$$

$$7 * 9 \bmod 10 = 3$$



Primera observación



- No es tan simple encontrar un inverso multiplicativo mod n , especialmente si n es muy grande,
- Si $n = 100$ digitos
 - no es lógico realizar una búsqueda de fuerza bruta para encontrar un inverso multiplicativo
- Algoritmo ecludiano
 - permite encontrar inversos mod n , dado x y n encuentra y tal que:
$$x * y \bmod n = 1 \text{ (si existe)}$$



Segunda observación



- ¿Por qué los números $\{1,3,7,9\}$ son los únicos que tienen inversos multiplicativos?
 - respuesta: son relativamente primos a 10.
- Relativamente primos a 10:
 - significa que no comparte ningún factor común aparte de 1
 - el entero más largo que divide 9 y 10 es 1
 - el entero más largo que divide 7 y 10 es 1
 - el entero más largo que divide 3 y 10 es 1
 - el entero más largo que divide 1 y 10 es 1



- En contraste 6 es primo en 10 ya que:
 - 2 divide a 6 y 10
 - 2 divide a 2 y 10
 - 2 divide a 4 y 10
 - 5 divide a 5 y 10
 - 2 divide a 8 y 10



En general



- Cuando se trabaja con aritmética mod n , todos los números relativos primos a n tienen multiplicativos inversos y los otros números no.
- Una multiplicación mod n por un número x es un criptograma ya que:
 - se puede multiplicar por x para encriptar
 - se puede multiplicar por x^{-1} para decriptar



En general



- No es un buen criptograma en el sentido de seguridad.
- Criptograma: se puede modificar la información a través de un algoritmo y revertir el proceso para obtener la información original.



La función totient



- ¿Cuántos números a n pueden ser relativamente primos a n ?

- Respuesta: función totient $\Phi(n)$
- to = total tient = quotient (cociente)

- Si n es primo:

$$\Phi(n) = n - 1$$

existen $n-1$ números relativamente primos a n

- Si n es un producto de dos números primos (p y q)

$$\Phi(n) = \Phi(pq) = \Phi(p) \times \Phi(q)$$

$$\Phi(n) = (p-1)(q-1)$$

existen $(p-1)(q-1)$ números relativamente primos a n :



Criptosistema RSA



- Primera realización del modelo de Diffie-Hellman
- Realizado por Rivest, Shamir y Adleman en 1977 y publicado por primera vez en 1978
 - se dice que un método casi idéntico fue creado por Clifford Cocks en 1973
- Basado en una TOF en que funciona con números primos
- Podría considerarse un criptosistema de bloque
 - texto claro y criptograma son enteros entre 0 y $n-1$ para algún valor de n



Protocolo RSA (cálculo llaves)



1. Usuario U elige dos números primos p y q :

calcular: $n = p \times q$

calcular: $\Phi(n) = (p-1)(q-1)$

2. U selecciona un entero positivo e , tal que:

$$1 < e < \Phi(n)$$

sea relativamente primo con $\Phi(n)$ es decir

$$\text{mcd}(\Phi(n), e) = 1$$



3. Mediante el algoritmo de Euclides extendido calcular el inverso de e en $\mathbb{Z}_{\Phi(n)}$ (i.e. multiplicativo inverso), sea d dicho inverso, entonces:

$$d = e^{-1} \bmod \Phi(n)$$

$$e \bullet d \equiv 1 \pmod{\Phi(n)} \text{ con } 1 \leq d < \Phi(n)$$

4. Calcular las llaves:

Llave pública: (e, n)

Llave privada: (d, n)

Deben permanecer secretos: p, q y $\Phi(n)$



Encriptación/decriptación mensajes



- Tomando en cuenta que las llaves son:
 Llave pública: (e, n)
 Llave privada: (d, n)
- Si se desea encriptar un mensaje M de Z_n^* (i.e. $M < n$)
$$C = M^e \bmod n$$
- Para decriptar el criptograma C es necesario:

$$M = C^d \bmod n$$



Ejemplo RSA: contexto



- Se considera una codificación del alfabeto que transforma las letras de la A a la Z en los números del 0 al 25
- Se desea enviar un mensaje a un usuario B
- Usuario elabora su llave pública y privada:
 - elige dos números primos: $p_b = 281$ y $q_b = 167$
 - calcula $n_b = 281 * 167 = 46927$



Ejemplo RSA: calculando llaves



- Orden grupo: $\Phi(46927) = 280 \times 166 = 46480$
- B elige número $e_b=39423$ y comprueba que:
$$\text{mcd}(39423, 46480) = 1$$
- B determina el inverso de 39423 módulo 46480, el cual es $d_b=26767$
- Por lo que la llave pública de B es
$$(n_b, e_b) = (46927, 39423)$$
- Mantiene en secreto el resto de los valores



Consideraciones envío mensaje



- En primer lugar se debe determinar la longitud del mensaje
- Se va a codificar las letras del alfabeto en base 26
- La longitud del mensaje no puede exceder el valor de $n = 46927$
- Dado que $26^3 = 1756 < n < 456976 = 26^4$
- Por lo que el mensaje debe tener un máximo de tres letras



¿Y si el mensaje es muy grande?



- Si se desea enviar un mensaje más largo, habrá que romper el mensaje original m en grupos de tres letras
- En la práctica la longitud es mucho mayor dado que n es un número con mucho más dígitos



Datos usuario A



- llave pública usuario A:

$$(n_a, e_a) = (155011, 2347)$$

- llave privada usuario A:

$$d_a = 151267 \text{ con } p_a = 409 \text{ y } q_a = 379 \text{ y}$$

$$\Phi(n) = 154224$$



Enviando el mensaje



- Para enviar mensaje m , se tiene que codificar, expresarlo en base 26

$$\begin{aligned}\text{YES} &= Y \cdot 26^2 + E \cdot 26 + S \\ &= (24 \cdot 26^2) + (4 \cdot 26) + 18 = 16346 = m\end{aligned}$$

- Se encripta m con la llave pública de B

$$\begin{aligned}c &= m^{e_b} \bmod n_b \\ &= 16346^{39423} \bmod 46927 = 21166\end{aligned}$$



Decodificando el mensaje



- Se decodifica el mensaje encriptado

$$\begin{aligned}c &= 21166 \\&= (1 \cdot 26^3) + (5 \cdot 26^2) + (8 \cdot 26) + 2 \\&= \text{BIFC}\end{aligned}$$

- Por lo tanto el mensaje a enviar a B es
BFIC



Recepción mensaje



- Para recuperar mensaje B debe codificar los datos recibidos en base 26 y realizar las operaciones anteriores

$$\begin{aligned}\text{BIFC} &= (1 \cdot 26^3) + (5 \cdot 26^2) + (8 \cdot 26) + 2 \\ &= 21166 \\ &= c\end{aligned}$$



Decriptando el mensaje



- Se recupera m calculando

$$\begin{aligned} m &= c^{d_b} \bmod n_b \\ &= 21166^{26767} \bmod 46927 = 16346 \end{aligned}$$

- Se codifica m y se obtiene el texto original

$$\begin{aligned} m &= 16346 \\ &= (24 \cdot 26^2) + (4 \cdot 26) + 18 = \text{YES} \end{aligned}$$



El problema de factorización



- En 1997 se lanzó un reto matematico
- Artículo *A New Kind of Cipher that Would Take Millions of Years to break*
- Columna *Mathematical Games* en *Scientific American*
- Criptosistema encriptado con llave pública

114,381,625,757,888,867,669,235,779,926,146,612,010,218,296,721,
242,362,562,561,842,935,706,935,245,733,897,830,597,123,563,958,
705,058,989,075,147,599,290,026,879,543,541

- Se estima que la factorización tomo aproximadamente 4000 a 6000 MIPS años de computo sobre un periodo de seis a ocho meses.



La solución



- El 26 de abril de 1994, un equipo de 600 voluntarios anunciaron los factores de N
- El factor q
3,490,529,510,847,650,949,147,849,619,903,898,133,417,764,638,
493,387,843,990,820,577
- El factor p
32,769,132,993,266,709,549,961,988,190,834,461,413,177,642,967,
992,942,539,798,288,533
- El mensaje era:
200805001301070903002315180419000118050019172105011309190800
151919090618010705

"THE MAGIC WORDS ARE SQUEAMISH OSSIFRAGE"



¿Y hoy en día?



RSA Laboratories | Challenges | Factoring Challenge - Netscape

File Edit View Go Communicator Help

PRODUCTS SERVICES PARTNERS CAREERS RSA ONLINE: MEMBERS ONLY

NEWS COMPANY EVENTS RSA Worldwide GO

BUY CONTACT DOWNLOAD SUPPORT SEARCH GO

[RSA Security Home](#) > [RSA Laboratories](#) > [Challenges](#) > Factoring

The New RSA Factoring Challenge

RSA Laboratories continues its sponsorship of the RSA Factoring Challenge to encourage research into computational number theory and the practical difficulty of factoring large integers. The information received during this challenge is a valuable resource to the cryptographic community and can be helpful for users of the RSA public-key cryptosystem in choosing suitable key lengths for an appropriate level of security.

The RSA Challenge numbers are the kind we believe to be the hardest to factor; these numbers should be particularly challenging. These are the kind of numbers used in devising secure RSA cryptosystems.

A cash prize is awarded to the first person to factor each challenge number. The prize amount is listed on the page with the challenge number. Prizes range from \$10,000 (US) for the 576-bit challenge to \$200,000 for 2048 bits. The prize money will be paid once RSA Laboratories has verified the correctness of the factorization.

[The RSA Challenge Numbers](#)

[Factoring Challenge FAQ](#)

[Submitting a Factorization](#)

More About

- Factoring Challenge
- [The RSA Challenge Numbers](#)
- [Factoring Challenge FAQ](#)
- [Submit a Factorization](#)

Document: Done

Start RSA Laboratori... Microsoft PowerPoi...

9:19 PM



Ejemplo de una llave pública



Oliver Roberts - My PGP Public Key - Netscape

File Edit View Go Communicator Help

Bookmarks Location: <http://www.nanunanu.org/~oliver/pgpkey.html> What's Related

My PGP Public Key

If you want to add my PGP public key to your PGP keyring, then it may be easier if you download my key as a separate [ASCII text file](#).

PRIVACY Now! PGP

Type	Bits/KeyID	Date	User ID
pub	1024/8C0C5D61	1996/07/20	Oliver Roberts <oliver@futura.co.uk> Oliver Roberts <oliver.roberts@iname.com> Oliver Roberts <oliver@nanunanu.org>

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.3i

```
mQCNazHxRgUAAAEAMiIn19TTDRUFVB2vhYUm4hgzi0f29b/s/YA6L8gSEkSEJWs
SkZZewdQkcjAfzS3Fk1P2jwFkKD0Q7pCd2GBnN/TnGS1fupVB3ydkspodPLhU4iD
y8lao2hfEy9bHScI5lKu9DhGXUZGUrzm3NEoYoYtGiAEEZk9N37cgqaMDFlhAAUR
tCVPbG12ZXIgm9iZXJ0cyA8b2xpdMVyQGZldGF1cmEuY28udWs+iQCVAwUQTZL
BH7cgqaMDFlhAQF2ogP/Vrsiumv704zfgk3+ruvzXwNwU1ecmAJd0kPrR/5V0YR
8tHmDb/eWhv8t8TUeu6b833SWAsZnT3mbLYAyyioJV10wekRx1VNXQIQpaXUbf3+
sjNw+QKLCFJbqlpUEPKpwYCo9Kwx10XpoUTFb0eJ4Pm0kLo/yYaczq6M1WaaGa0
KU9saXZ1ciBSb2JlcnRzIDxvbG12ZXIucm9iZXJ0c0BpbmFtZS5jb20+iQCVAwUQ
NXSGMH7cgqaMDFlhAQFCAQA2g2017KmOD78BAqyqAoXh/v0lrrshJqW30BknCY
2XsHFaCgw6NjpEgOn0h40NQx69K4jinrzH/v7emyRs9BXsaDhELN6BWduwIGwcnF
TwIp2HomjvhTcx5f5mKR66qN9GLAhQWQjWZ86xQALMw5RoBD4cjAr7VsJF0ir8XHA
0j+0JE9saXZ1ciBSb2JlcnRzIDxvbG12ZXJAbmFudW5hbnUub3JnPokA1QMFEFV0
iHB+3IKmjAxdiYQEBMgOEALxjx9PpRjwTrFyIKX7bnTLK7KBua0KOYyPx15dRn/Qf
5xlaLxGlsjRxTxSvB1UKESiEiVbaGkPXBEIPuXVTG15BoktA+sX3/vthYfyfR00a
9XGOCMxiJh7QUMDMb6s4awA8tEkuP4iS96GsaTBW1I3q6z4k5Yb1hoezXt4+ezy
=EWwi
```

-----END PGP PUBLIC KEY BLOCK-----

Buy from **HiSOFT**

[Browse 2.2](#)
[DOpus Magellan II](#)
[MakeCD](#)

W3C HTML 4.01

Document: Done

Start | Ex... | Mi... | Dr... | 0... | Ne... | 9:51 PM



Otros algoritmos de llave pública



- El Gammal
- Pohling Hellman
- Curvas elípticas
- Rabin
- McEliece
- Criptosistemas de llave pública de automatas finitos.



Algoritmos de intercambio de llaves



- Protocolo de estación-estación
- Protocolo de tres pasos de Shamir
- COMSET
- Encrypted Key Exchange
- Fortified Key Negotiation
- Conference Key Distribution and Secret Broadcasting



Sistemas Híbridos



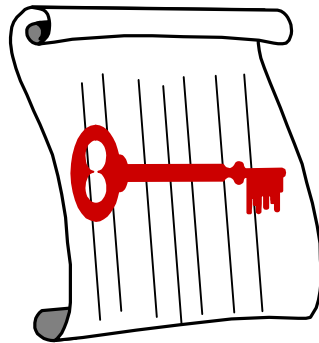
- Un algoritmo simétrico con una llave de sesión aleatoria es usada para encriptar un mensaje.
- Un algoritmo de llave pública es usado para encriptar la llave de sesión aleatoria.



Encriptación sistema híbrido



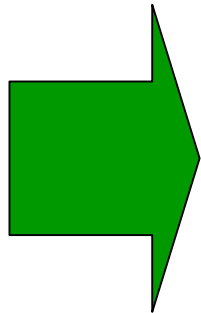
2. Generar una llave simétrica aleatoria



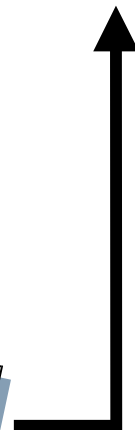
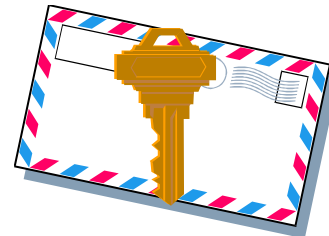
3. Encriptar mensaje con llave simétrica



5. Poner mensaje y llave encriptados en un solo mensaje y enviarlo

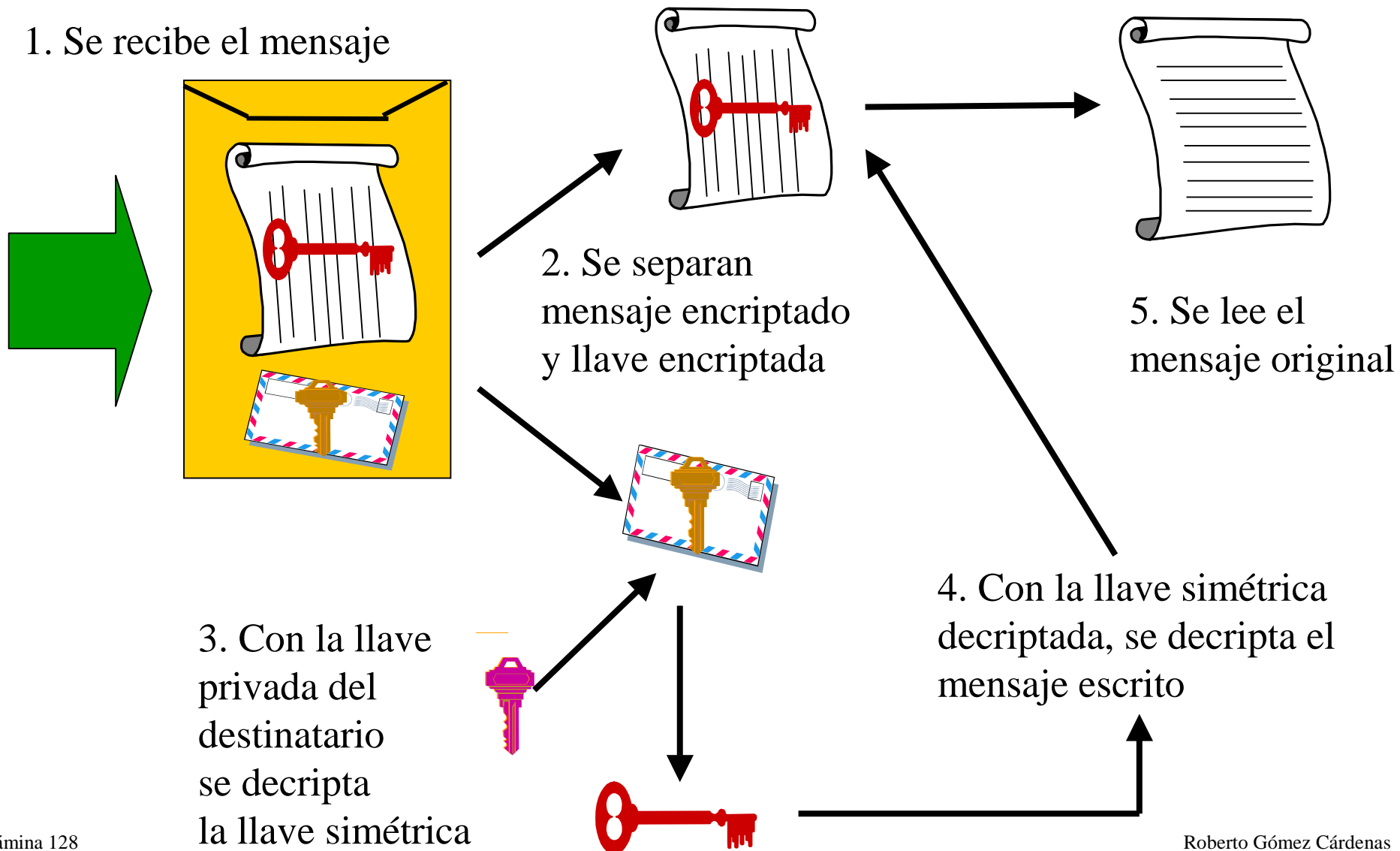


4. Tomar llave pública destinatario y encriptar llave simétrica





Decripción sistema hibrido

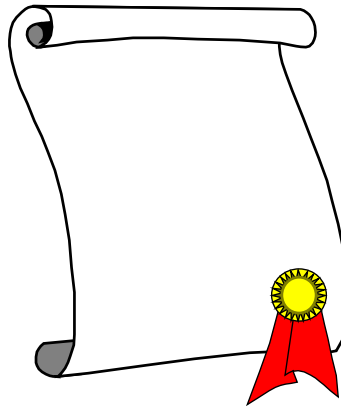
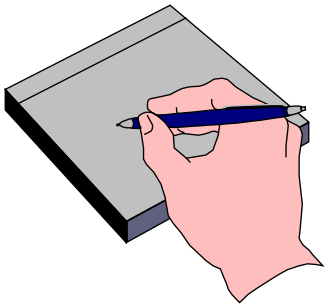




Firmas, huellas y MACs



características y usos





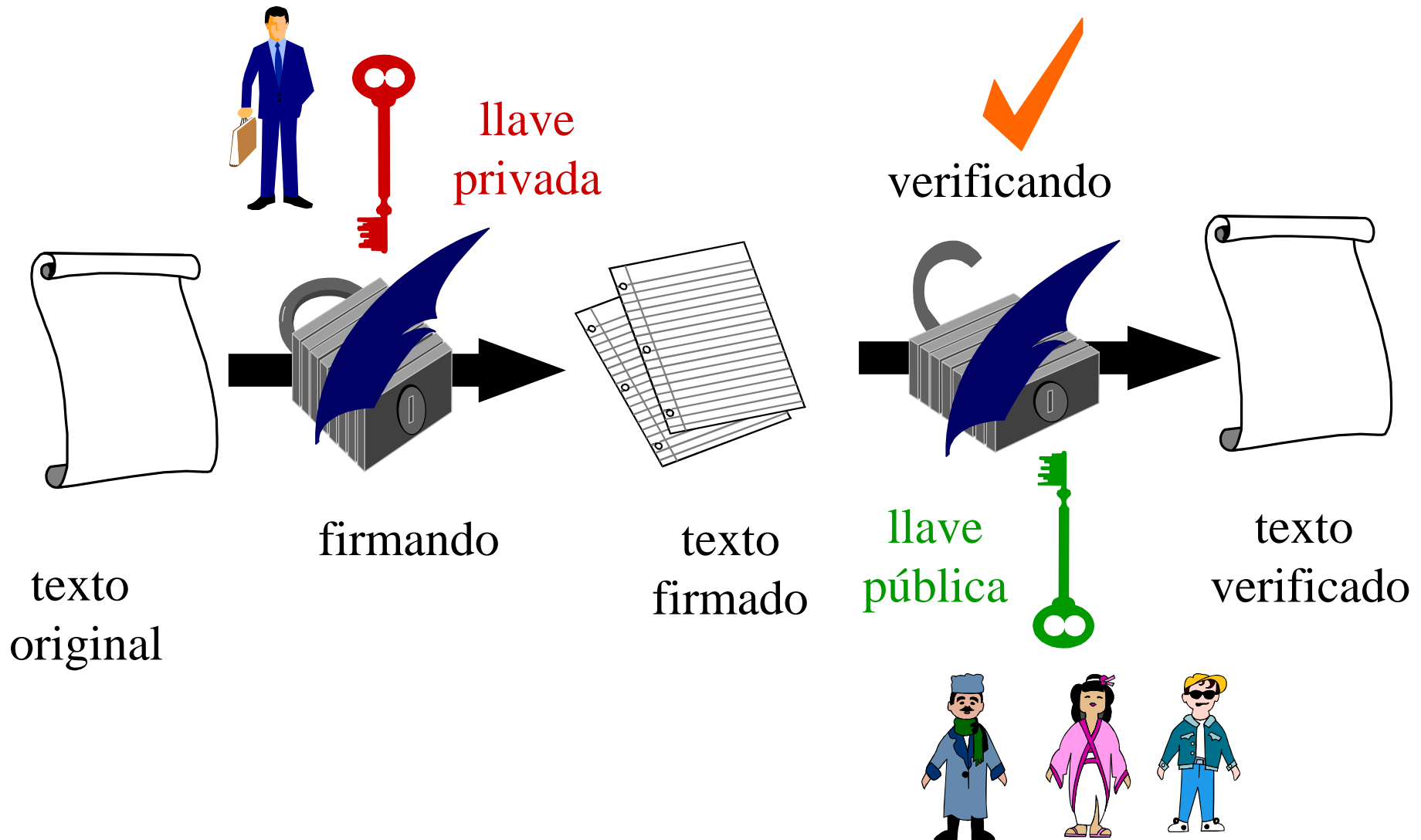
La firma digital



- Permiten al receptor verificar:
 - la autenticidad del origen de la información
 - que la información esta intacta (integridad)
 - no-repudiación: que el emisor argumente que no envió la información
- Tiene mismo propósito firma escrita
- Ventaja: no puede ser falsificada tan fácilmente como la escrita



Un esquema de firma digital





Las funciones hash



- Sistema anterior es lento y produce gran cantidad de información
- Mejoramiento: añadir una one-way hash function
 - función toma una variable de tamaño variable (cientos o miles de bits) y una salida de tamaño fijo (p.e. 160 bits)
- Función asegura que, si la información es cambiada (aún en sólo un bit) un valor completamente diferente es producido



Ejemplo función hash



- Un ejemplo simple es tomar una entrada y regresar un byte que consista de XOR de todos los bytes de entrada
- Función de un solo sentido: fácil de calcular un valor de hash de la entrada, pero difícil generar una entrada que corresponda a una salida



Ejemplo función simple



	Bit 1	Bit 2	...	Bit n
Bloque 1	b_{11}	b_{11}		b_{n1}
Bloque 2	b_{12}	b_{22}		b_{n2}
	\vdots	\vdots	\vdots	\vdots
Bloque m	b_{1m}	b_{2m}		b_{nm}
Código hash	C_1	C_2		C_n



Ejemplo código función hash



```
main(int argc, char *argv[])
{
    unsigned long hash[4] = {0, 0, 0, 0}, data[4];
    FILE *fp;    int i;

    if ((fp = fopen(argv[1], "rb")) != NULL) {
        while ( fread(data, 4, 4, fp) != NULL)
            for (i=0; i<4; i++)
                hash[i] ^= data[i];
        fclose(fp);
        for (i=0; i<4; i++)
            printf("%08lx",hash[i]);
        printf("\n");
    }
}
```



Salida del ejemplo código función hash



```
rogomez@armagnac:68>gcc hash1.c -o hash1
rogomez@armagnac:69>more toto
ULTRA SECRETO
```

Siendo las 19:49 hrs del día 19 de noviembre de 1999
pretendo anunciar que se terminó el presente texto
para pruebas de programas hash.

Atte;

RGC

```
rogomez@armagnac:70>hash1 toto
0f7621300e2b431d6457510e09780853
rogomez@armagnac:71>
```




rogomez@armagnac:71>more toto
ULTRA SECRETO

Siendo las 19:49 hrs del día 19 de noviembre de 1999
pretendo anunciar que se terminó el presente texto
para pruebas de programas hash.

Atte

RGC

rogomez@armagnac:72>hash1 toto
57632579652b431d6457510e09780853
rogomez@armagnac:73>



La función hash MD5



- **MD5** toma como entrada un mensaje de longitud arbitraria y regresa como salida una “*huella digital*” de 128 bits del mensaje (llamado message-digest o compendio del mensaje).
- Se estima que es imposible obtener dos mensajes que produzcan la misma huella digital.
- También es imposible producir un mensaje que arroje una huella predefinida



Descripción del algoritmo



- El mensaje de entrada puede tener cualquier longitud, no necesariamente debe ser múltiplo de 8.
- Los pasos que sigue el algoritmo son:
 - **Paso 1.** Agregado de bits de relleno (*Padding*).
 - **Paso 2.** Agregado de la longitud.
 - **Paso 3.** Inicialización del buffer del MD
 - **Paso 4.** Procesamiento del mensaje en bloques de 16 palabras.
 - **Paso 5.** Compendio del mensaje.



Paso 1: bits de relleno



- El mensaje es extendido de tal forma de que sea casi múltiplo de 512 bits de longitud.
- Casi porque se reservarán 64 bits.
- Estos 64 bits serán cubiertos con el tamaño del mensaje (expresado en 64 bits).



Paso 2: longitud mensaje



- El tamaño del mensaje es agregado al final del mensaje resultante del paso 1 (64 bits).
- En el caso de que la longitud del mensaje requiera más de 64 bits, sólo los 64 bits menos significativos se tomarían en cuenta.
- Como resultado de este paso se tiene un mensaje cuya longitud es múltiplo de 512.



Paso 3: Inicialización buffer



- Un buffer de 4 palabras es inicializado de la forma siguiente:

palabra A: 01 23 45 67

palabra B: 89 ab cd ef

palabra C: fe dc ba 98

palabra D: 76 54 32 10

- Al final el buffer sea la fima digital.



Paso 4: procesamiento mensaje



- Se definen 4 funciones de procesamiento:

$$F(X,Y,Z) = XY \vee \text{not}(X) Z$$

$$G(X,Y,Z) = XZ \vee Y \text{not}(Z)$$

$$H(X,Y,Z) = X \text{ xor } Y \text{ xor } Z$$

$$I(X,Y,Z) = Y \text{ xor } (X \vee \text{not}(Z))$$

- Donde X, Y, Z son palabras de 32 bits.



Diagrama general hash

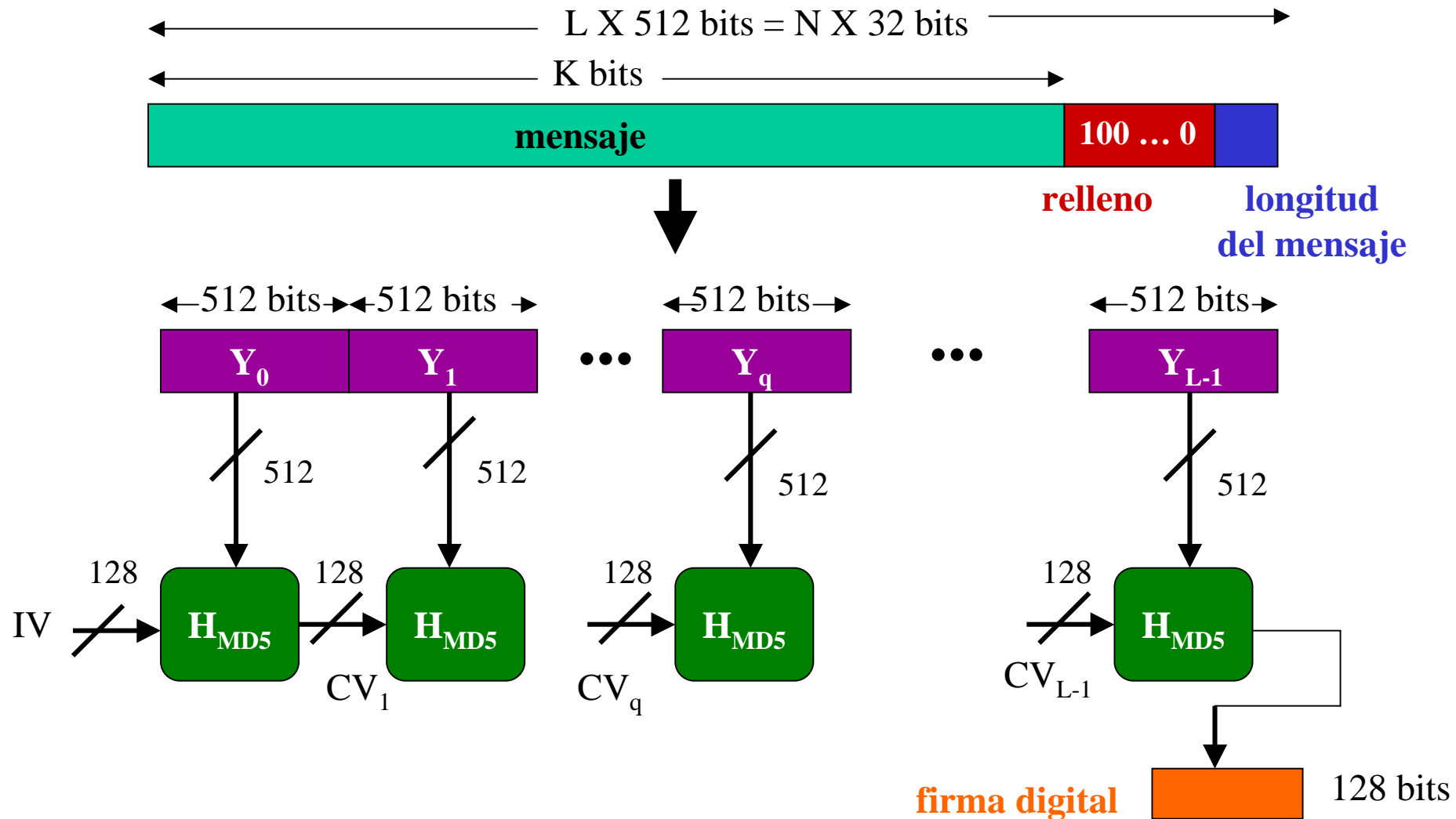
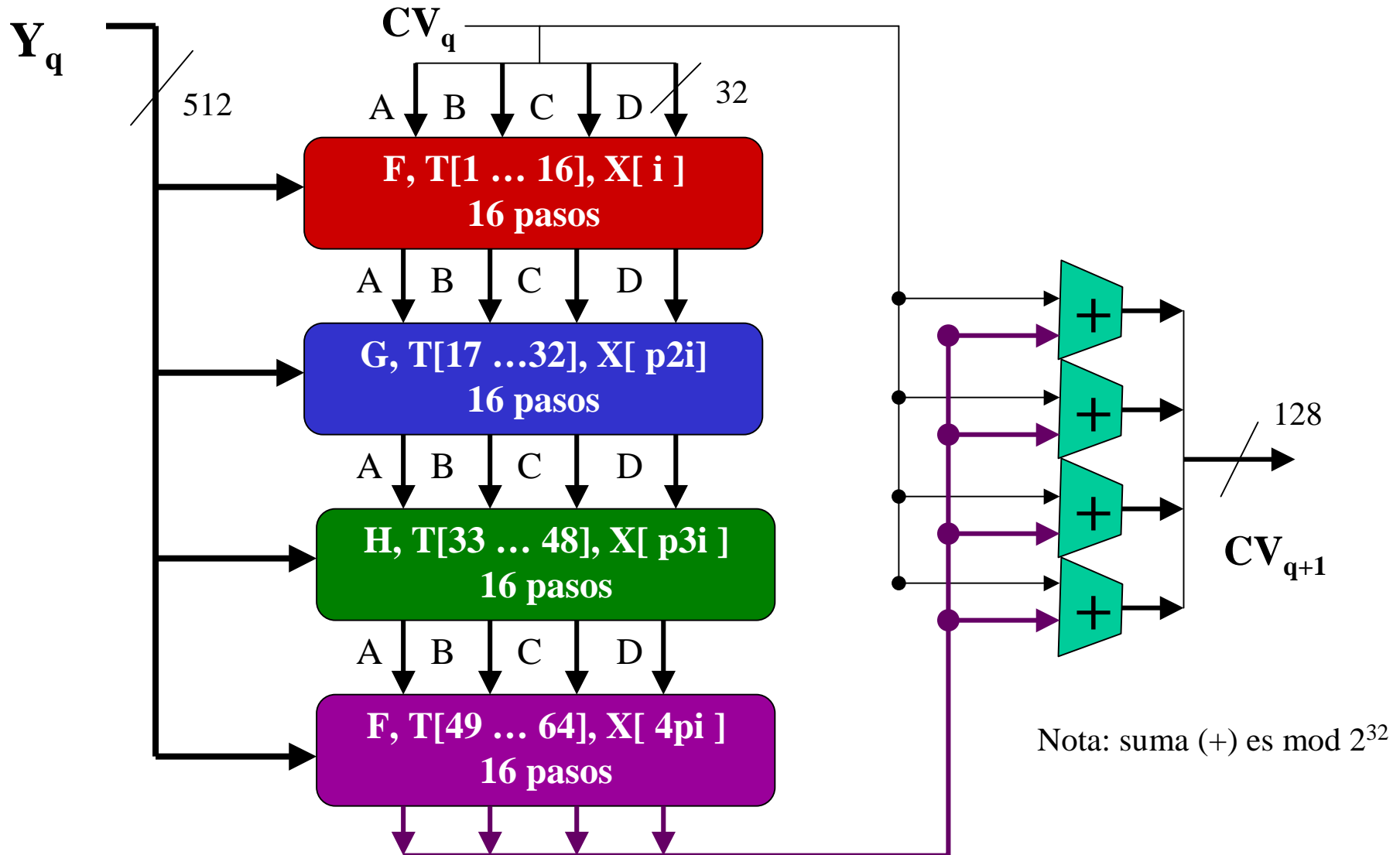




Diagrama procesamiento bloque 512 bits





Salida de MD5



rogomez@armagnac:464>more toto
ULTRA SECRETO

Siendo las 19:49 hrs del dia 19 de noviembre de 1999
pretendo anunciar que se termino el presente texto
para pruebas de programas hash.

Atte;

RGC

rogomez@armagnac:465>md5 toto

MD5 (toto) = 0c60ce6e67d01607e8232bec1336cbf3

rogomez@armagnac:466>



rogomez@armagnac:467>more toto
ULTRA SECRETO

Siendo las 19:49 hrs del dia 19 de noviembre de 1999
pretendo anunciar que se termino el presente texto
para pruebas de programas hash.

Atte

RGC

rogomez@armagnac:468>hash1 toto
MD5 (toto) = 30a6851f7b8088f45814b9e5b47774da
rogomez@armagnac:469>



Otras funciones hash de un solo sentido



- SHA-1
- Algoritmo MD2
- Algoritmo MD4
- RIPE MD-160
- HMAC
- N-Hash
- Havalk



La huella digital



- La salida producida por una función hash aplicada a un documento, es conocida con el nombre de huella digital de dicho documento
- Cualquier cambio en el documento produce una huella diferente
- Huella digital también es conocida como compendio de mensaje (cuando el documento es un mensaje)

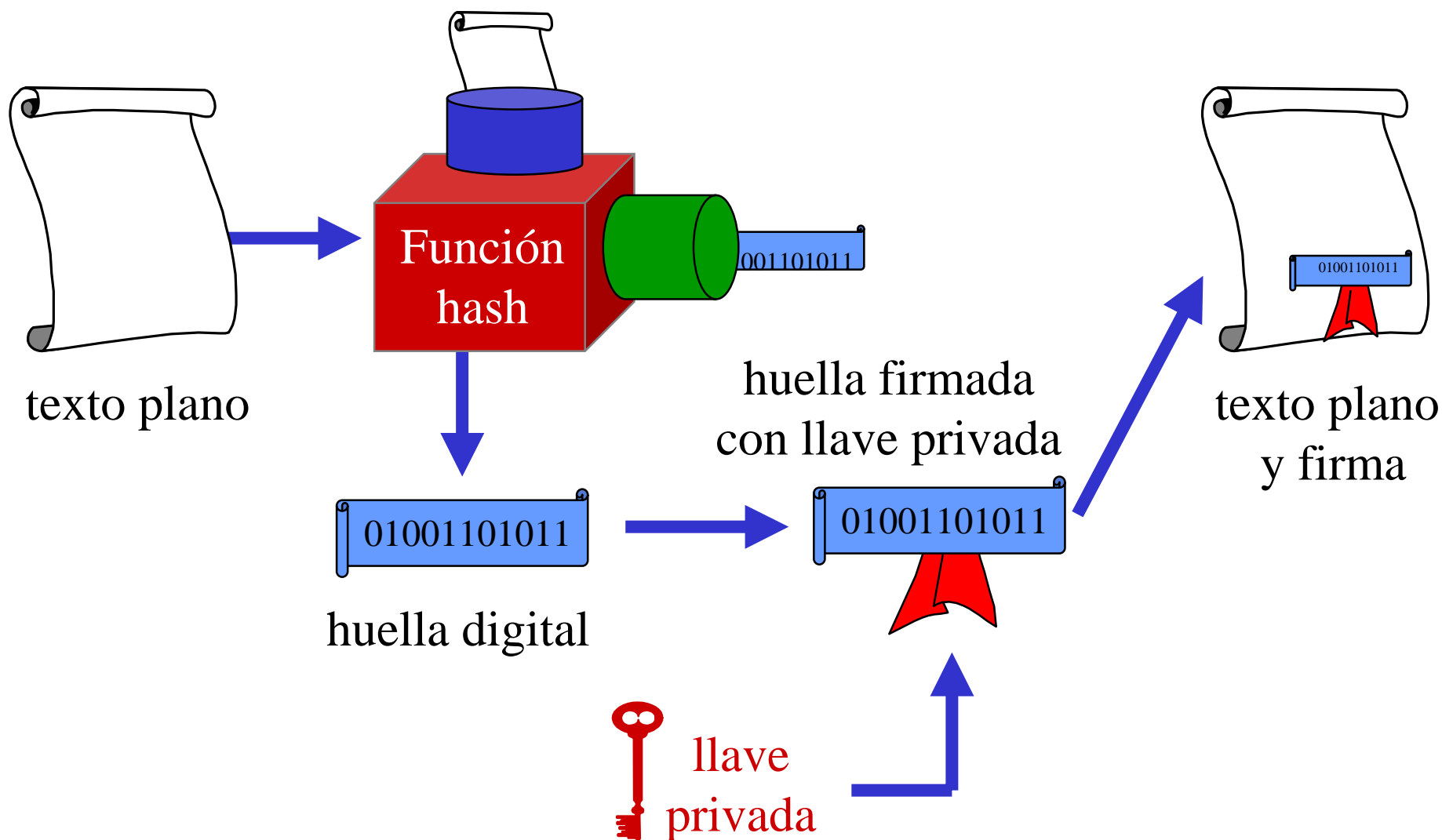


Firmas y huellas digitales

- Es posible usar la huella y la llave privada para producir una firma
- Se transmite el documento y la firma juntos
- Cuando el mensaje es recibido, el receptor utiliza la función hash para recalcular la huella y verificar la firma
- Es posible encriptar el documento si así se desea

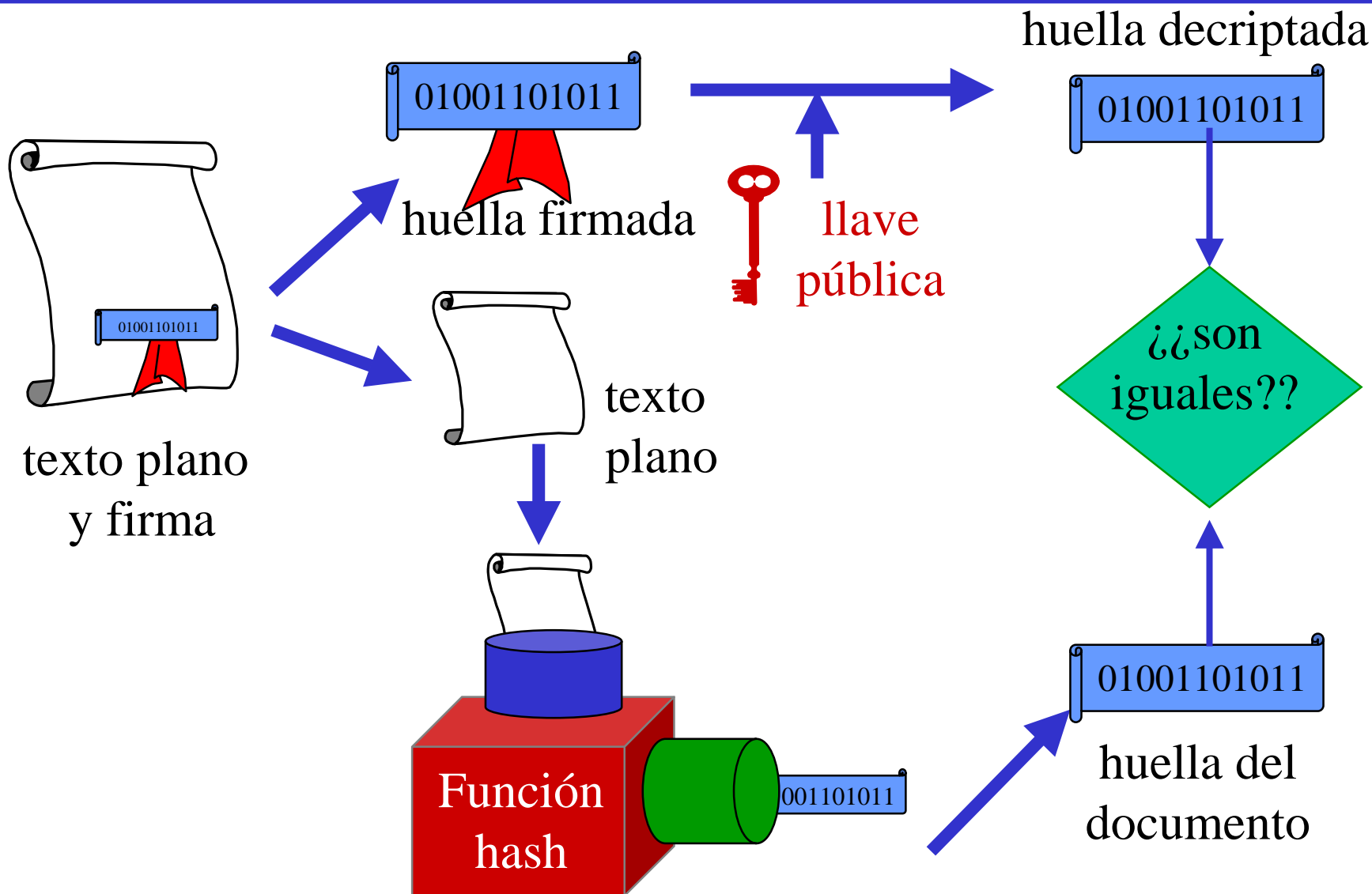


Firma digital segura (envío)





Firma digital segura (recepción)





Seguridad de la firma



- Seguridad depende de lo seguro de la función hash
- No existe ninguna forma de tomar la firma de alguien de un documento y ponerla en otro
- No es posible alterar un mensaje firmado
- El más simple cambio en el documento firmado se verá en la verificación



Otro uso de la huella digital



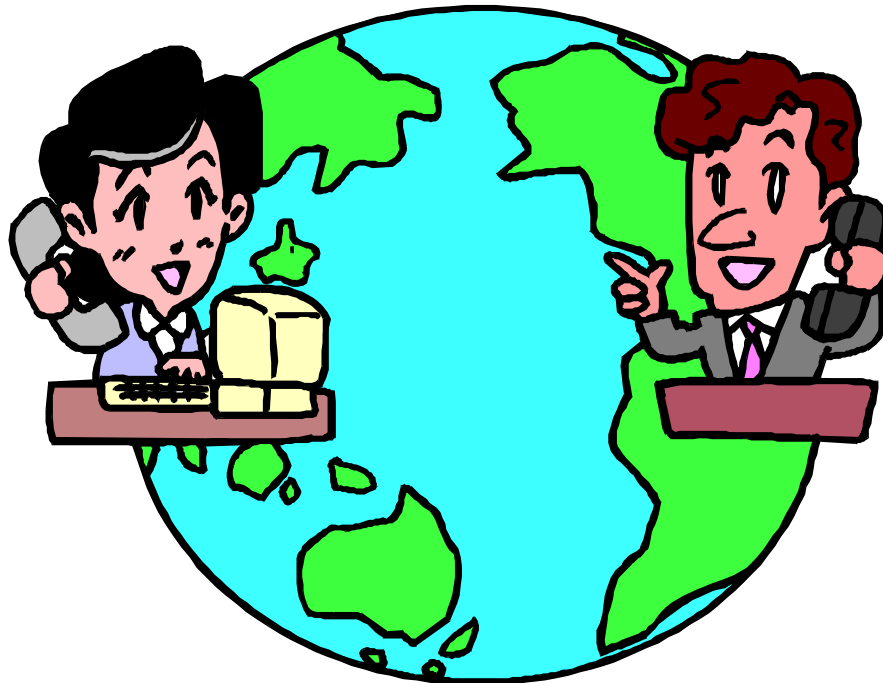
- Verificar la integridad de un documento
- Se obtiene la huella del documento y se almacena
- Tiempo despues se vuelve a calcular la huella y se calcuala con lo almacenado
 - si coincide: no hubo cambios
 - si no coincide: la información fue alterada
- Ejemplo
 - tripwire



Criptología y transmisión de datos



Protocolos de transmisión de datos seguros en Internet





Protocolos existentes



- SSL
- PCT
- TLS
- S-HHTTP
- Ipsec e IPv6
- SSH
- PGP
- S/MIME
- iKP
- SET
- CyberCash/CyberCoin
- DNSEC
- Kerberos
- S/Key



SSL, PCT y TCL



- Protocolos criptográfico de propósito general para asegurar canales de comunicación bidireccionales
- Se utilizan comúnmente junto con el protocolo TCP/IP
- Sistema encriptación usado por navegadores Netscape e Internet Explorer



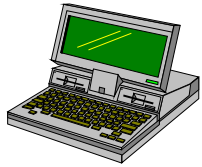
SSL, PCT y TLS



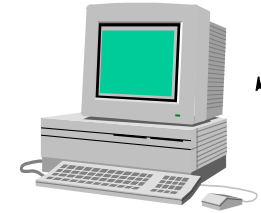
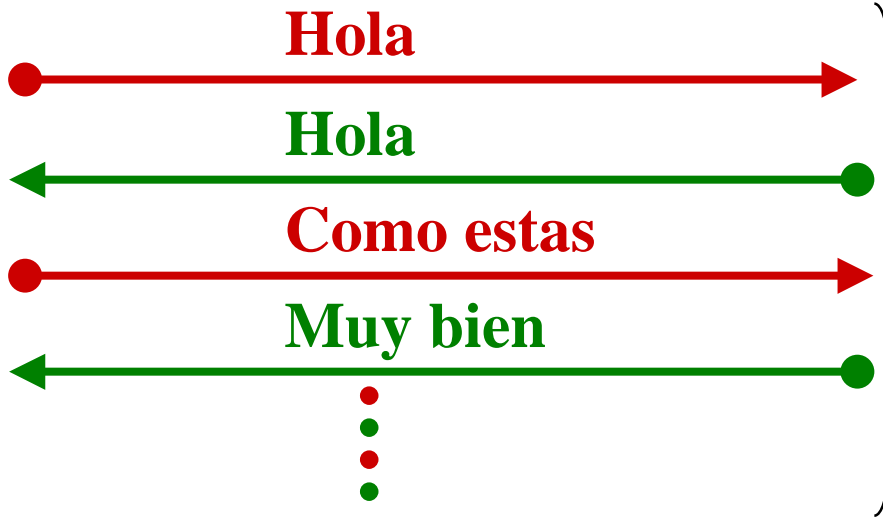
- 1994: SSL V 2.0 (Netscape)
microsoft descubre un problema en SSL
- 1995: PCT V 1.0
- 1996: SSL V 3.0
- 1997: PCT V 4.
se decide terminar con la pelea: Microsoft y
Netscape deciden sacar un protocolo en común
- 1999: TLS V 1.0



¿Cómo funciona?



Cliente

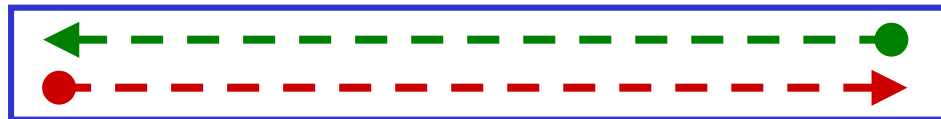


Servidor

No hay autenticación
ni privacidad, ni
encriptación



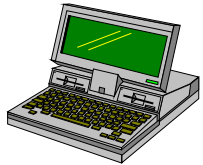
Hablemos en forma
segura



Comunicación encriptada con la llave enviada por el cliente



Otro posible escenario



Cliente

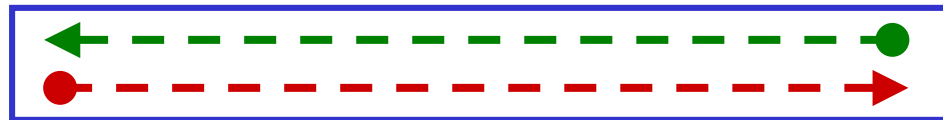


Servidor

**Hablemos de forma segura, aquí están
los protocolos y criptogramas que manejo**

**Escogo este protocolo y criptograma. Aquí
esta mi llave pública, un certificado digital y
un número random**

**Usando tu llave pública encripte una
llave simétrica aleatoria**



*Comunicación encriptada con la llave enviada por el cliente
y un hash para autenticación de mensajes*



Ejemplo protocolo seguro (1er. paso)



Amazon.com: buying info: Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edit - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Stop

Bookmarks Location: amazon.com/exec/obidos/ASIN/0471117099/o/qid=971146521/st=2-2/102-5972864-6451304 What's Related

Members WebMail Connections BizJournal SmartUpdate Mktplace RealPlayer

amazon.com. YOUR ACCOUNT HELP

WELCOME DIRECTORY BOOKS

SEARCH BROWSE SUBJECTS BESTSELLERS NEW & FUTURE RELEASES BARGAIN BOOKS AWARDS SPANISH LANGUAGE

TODAY'S FEATURED STORES BOOKS ELECTRONICS DVD SOFTWARE CAMERA & PHOTO

SEARCH

Books GO!

BOOK INFORMATION

Explore this book

[buying info](#)

[table of contents](#)

[Amazon.com articles](#)

[editorial reviews](#)

[customer reviews](#)

See more by this

Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition

by [Bruce Schneier](#)

List Price: \$54.95

Our Price: **\$43.96**

You Save: **\$10.99 (20%)**

Availability: Usually ships within 24 hours.

[See larger photo](#)

Paperback - 784 pages 2 edition (October 18, 1995)

John Wiley & Sons; ISBN: 0471117099 ; Dimensions (in inches): 1.87 x 9.20 x 7.54

Other Editions: [Hardcover](#)

READY TO BUY?

Add to Shopping Cart (you can always remove it later)

Shopping with us is 100% safe. Guaranteed.

Add to Wish List

(We'll set one up for you)

[View my Wish List](#)

Start Exploring... Microsoft... Amazon...

9:56 PM

Ejemplo protocolo seguro (2do.paso)



Amazon.com Checkout: Sign In - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Stop

Bookmarks Location: <https://www.amazon.com/exec/obidos/checkout-sign-in/103-4769853-4624626> What's Related

Instant Message WebMail Contact People Yellow Pages Download Find Sites Channels

amazon.com

WELCOME ADDRESS ITEMS WRAP SHIP PAY CONFIRM

***** Please fix the areas indicated below. *****

***** You didn't provide an e-mail address. We'll need it to communicate with you about the status of your orders. And, when you visit us again, you'll use it to access your account. *****

Ordering online is easy.
We'll walk you through the process, step by step.

Enter your e-mail address:

☐ I am a new customer.
(You'll create a password later.)

☐ I am a returning customer,
and my password is:

[Forgot your password?](#)

[Sign in using our secure server](#)

Amazon.com Safe Shopping Guarantee

We guarantee that every transaction you make at Amazon.com will be 100% safe. This means you pay nothing if unauthorized charges are made to your credit card as a result of shopping at Amazon.com.

[Learn More](#)

Document: Done



¿Y que hago con todo esto?

Implementaciones criptográficas



Implementando lo anterior



- Programar las rutinas de encriptación/decriptación uno mismo
- Usar librerías/bibliotecas con rutinas de encriptación decriptación
- Utilizar estándares aplicaciones disponibles en internet.



Liberías/rutinas criptográficas



- Crypto++
- Cryptix
- Cryptlib Encryption Toolkit
- OpenSSL
- JCSI - Java Crypto and Security Implementation
- JGSS
- The Delphi Cryptography Page
- Encrypt-COM
- API Java Card
- PowerCrypt
- Elliptic



Crypto C++



- Página
 - <http://www.eskimo.com/~weidai/cryptlib.html>
- Aspectos importantes
 - librería gratuita de clases C++.
 - Compilable sin cambios en Visual C++ 6.0 SP3 y gcc (y con reservas en otros compiladores Windows, UNIX y Mac).
 - Permite implementar la mayoría de algoritmos criptográficos, incluidos los cinco candidatos AES.



Cryptix



- Página
 - <http://www.cryptix.org/>
- Aspectos importantes:
 - Proyecto internacional de voluntarios destinado a proporcionar librerías gratuitas en Java que permitan implementar los principales algoritmos criptográficos.



Cryptlib Encryption Toolkit



- Página
 - <http://www.cs.auckland.ac.nz/~pgut001/cryptlib/>
- Aspectos importantes:
 - Conjunto de herramientas dirigidas a proporcionar seguridad criptográfica a usuarios poco experimentados "en tan sólo media hora".
 - Gratuito para usos no comerciales, existen ciertos terminos para su uso comercial



Implementación funciones criptologicas



- S/MIME
- PEM
- PGP



S/MIME



- MIME: Multipurpose Internet Mail Extensions
 - estándar para enviar mensajes con archivos binarios anexos (attach) a través de Internet
- S/MIME extiende el estándar MIME para proporcionar correo electrónico firmado
- Proviene de RSA Data Security (1996)



S/MIME (cont)



- No fue implementado como un programa sencillo, sino como una biblioteca diseñada para agregarse a los paquetes de correo
- Ofrece:
 - confidencialidad (usuario elige algo encriptación)
 - integridad a través de una función hash
 - autenticación con certificados
 - no repudiación con mensajes firmados



PEM: Privacy-Enhanced Mail



- Estandar de Internet que proporciona intercambio seguro de correo electrónico (RFC 1421).
- Emplea una serie de técnicas criptográficas que proporcionan confidencialidad, autenticación del emisor e integridad del mensaje.
- Autenticación emisor permite a un usuario verificar que el mensaje que recibió es verdaderamente de la persona que dice que lo envió.
- Integridad permite asegurarse que el mensaje no fue modificado durante su transporte.



¿Donde se puede obtener PEM?



- Existen dos implementaciones de PEM.

1

- Riordan's Internet Privacy Enhanced Mail (RIPEM)
- escrito por Mark Riordan
- disponible de ripem.msu.edu, directorio /pub/crypt y leer el archivo GETTING ACCESS

2

- Originalmente llamada TIS/PEM escrita por Trusted Information Systems
- substituida por TIS/MOSS (versión 7.1)
- un programa que implementa PEM dentro de MIME
- disponible es ftp.tis.com en directorio /pub/MOSS, leer el archivo README



Notas



- Verificar las leyes locales de los países donde se van a utilizar/programar las funciones criptográficas.
- Verificar los permisos y las licencias
- Algunos sistemas ya están preconfigurados de acuerdo al país donde se instalen (ejemplo Netscape)



Pretty Good Privacy





¿Qué es PGP?



- Encriptación de archivos
- Encriptación de correo electrónico
- Manejo de llaves
- Borrado seguro (secure wipe)
- No es
 - esteganografía



Características de PGP



- Software acceso libre (<http://www.pgpi.org>).
- Desarrollado por Phil Zimmermann en 1994.
- Protección de e-mail y de archivos de datos.
- Comunicación segura a través de canales inseguros.
- Administración de llaves.
- Firmas digitales.
- Compresión de datos.



Versiones de PGP



- PGP Freeware v6.5.8 está disponible para Windows 95/98/NT/2000! y el Macintosh.
- PGP Freeware v6.5.8 está disponible para MacOS 7.6.1+
- PGP Command Line Freeware v6.5.8 está disponible para AIX/HP UX/Linux/Solaris!
- PGP Certificate Server Freeware v2.5.8 está disponible para Windows NT/2000 y Solaris



Servicios de Seguridad con PGP



- Privacidad.
 - *Sólo aquellos que deben recibir un mensaje pueden leerlo.*
- Autenticación.
 - *El origen de un mensaje es comprobable.*
- Borrado seguro
 - *Un archivo es borrado escribiendo n veces sobre el sector*



Funciones de PGP



- Criptosistemas
 - 1) Convencional (Llave secreta)
 - 2) Llave Pública
- Firmas Digitales
- Compendios de Mensajes (huellas digitales)
- Administración de llaves
- VPN: Virtual Private Networks



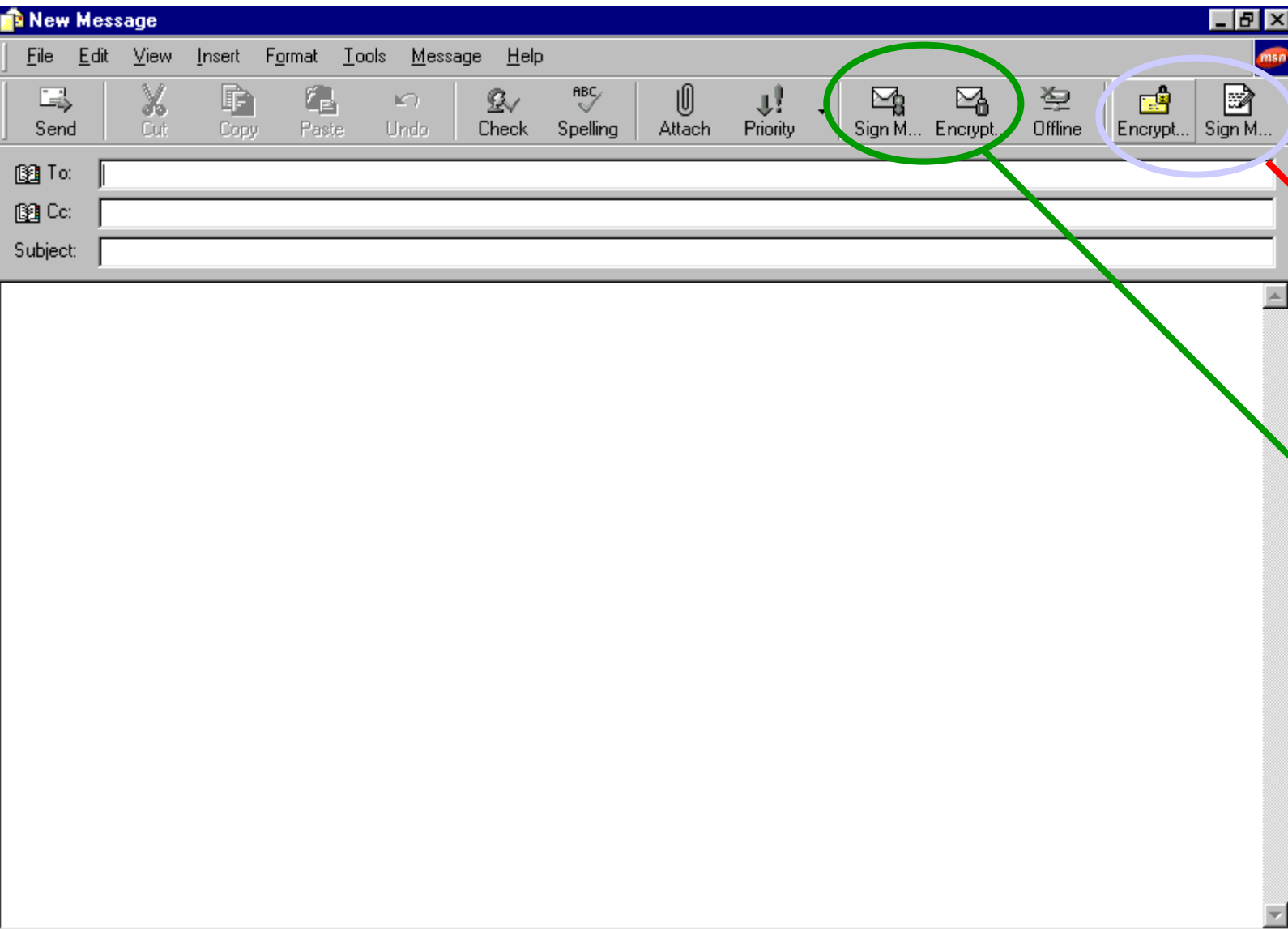
Algoritmos usados por PGP



- Especificados en RFC 2440.
- En orden de preferencia son:
 - ElGamal y RSA para intercambio de llaves
 - triple DES, IDEA y CAST5 para encriptación completa de mensajes.
 - DSA y RSA son usados para firmas digitales
 - SHA-1 y MD5 son usados para obtener huellas digitales
 - El programa shareware ZIP es usado para comprimir mensajes para su transmisión y almacenamiento.
- Compatibilidad de correo es lograda con el uso de conversión Radix-64.



Integración con Outlook



PGP

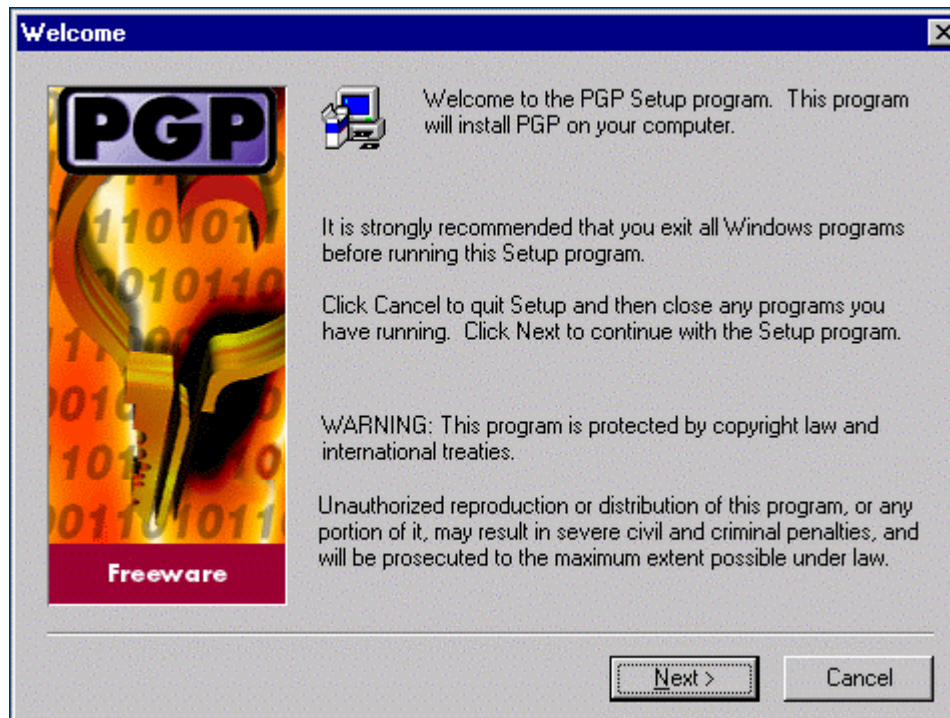
S/MIME



Instalando PGP (MS Outlook)



- Extraer archivo ZIP
- Cerrar programas
- Ejecutar Setup.exe





Manejo de llaves



- Generar llaves
- Importar llaves
- Exportar llaves
- Firmar una llave
- Ajustando el nivel de seguridad
- Revocando llaves
- Particionando llaves



Generando llaves



- Introducir nombre y correo
- Seleccionar tipo DH o RSA
- Seleccionar tamaño llave
 - más grande es mejor
 - más grande es más lento
- Seleccionar opción de expiración
 - un periodo determinado
 - indeterminado

The screenshot shows the 'Key Generation Wizard' window. On the left is a blue silhouette of a person holding a key, with the 'PGP' logo below it. The main text asks for a name and email address. The 'Full name' field contains 'Prueba' and the 'Email address' field contains 'Prueba@prueba.com'. At the bottom are four buttons: '< Atrás', 'Siguiendo >', 'Cancelar', and 'Ayuda'.

Key Generation Wizard

What name and email address should be associated with this key pair?

By listing your name and email address here, you let your correspondents know that the key they are using belongs to you.

Full name:
Prueba

Email address:
Prueba@prueba.com

< Atrás Siguiendo > Cancelar Ayuda



Generando llaves ...



- Dar una frase
 - escoger una buena frase
 - confirmar frase
- Enviar la llave al servidor (opcional)

En unix:

```
toto@kiko:1>pgp -kg
```

The screenshot shows the 'Key Generation Wizard' window. On the left is a blue silhouette of a person holding a key, with the 'PGP' logo below it. The main text area contains instructions: 'Your private key will be protected by a passphrase. It is important that you do not write this passphrase down.' and 'Your passphrase should be at least 8 characters long and should contain non-alphabetic characters.' Below this is a 'Passphrase:' label, a checked 'Hide Typing' checkbox, and a text input field. Underneath the input field is a 'Passphrase Quality' indicator consisting of a row of blue bars. At the bottom is a 'Confirmation:' label and another text input field. The window has a standard Windows-style title bar and a footer with four buttons: '< Atrás', 'Siguiete >', 'Cancelar', and 'Ayuda'.



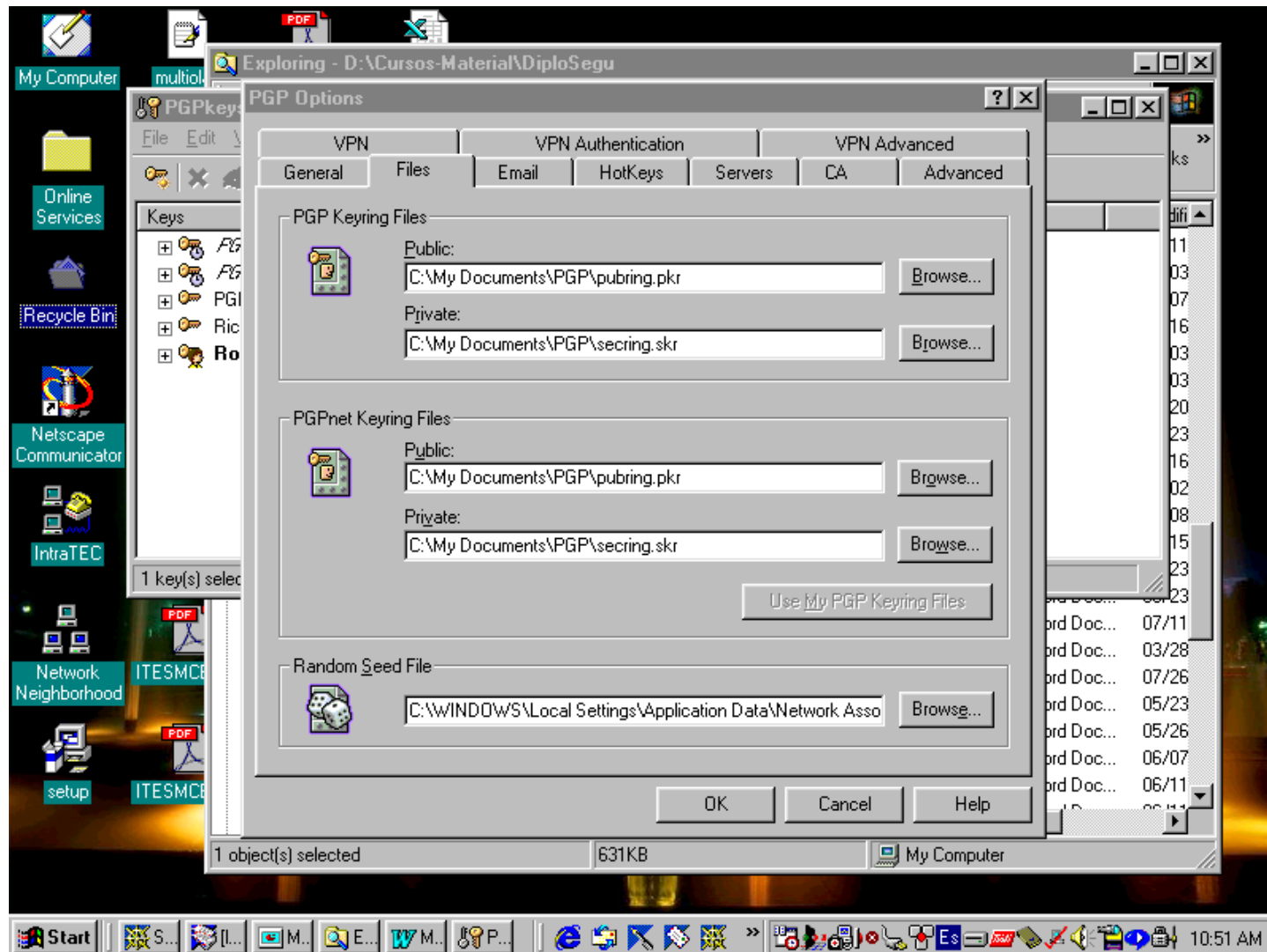
Importando llaveros existentes



- Cuando se pregunte el uso de llaves existentes responder [yes]
- Seleccionar los archivos .pkr y .skr a importar
- Despues de la instalación:
 - menu Options de Edit de PGPKKeys
 - asignar la llave secreta al archivo .skr
 - asignar la llave publica al archivo .pkr



Definiendo lugar llaves

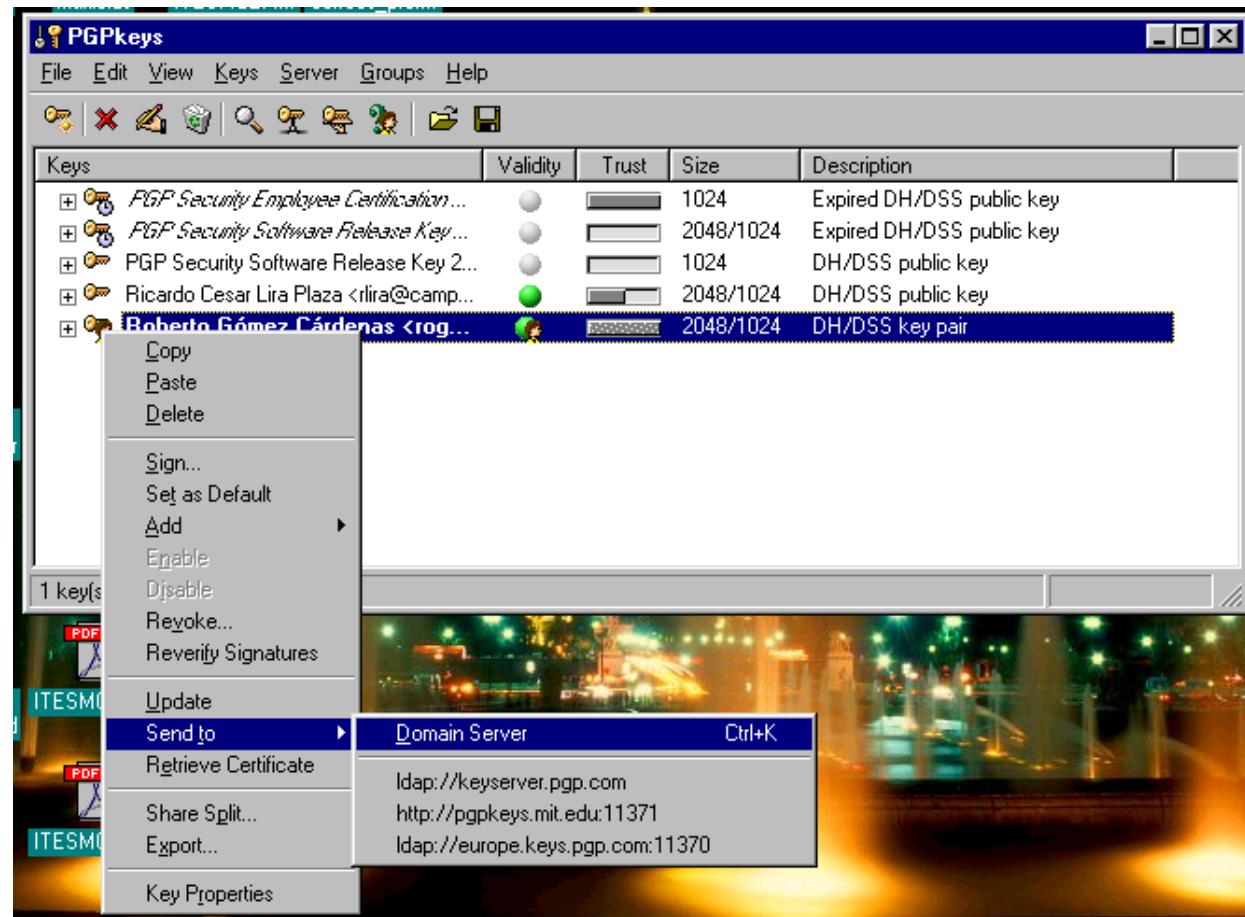




Exportando llaves públicas a servidores llaves



- Abrir PGPkeys.
- Seleccionar la llave a enviar y presionar botón derecho.
- Seleccionar enviar.
- Seleccionar servidor.





Exportar llaves públicas a archivos o correo



- Abrir PGPkeys
- Seleccionar la llave
- Del menu de Edit seleccionar Copy (o dar ctrl-c)
- Abrir el archivo o el correo
- Seleccionar Paste
- El bloque de texto que representa la llave es pegada al objetivo
- Otra opción es seleccionar opcion Export de Keys de PGPkeys



Ejemplo llave



-----BEGIN PGP PUBLIC KEY BLOCK-----

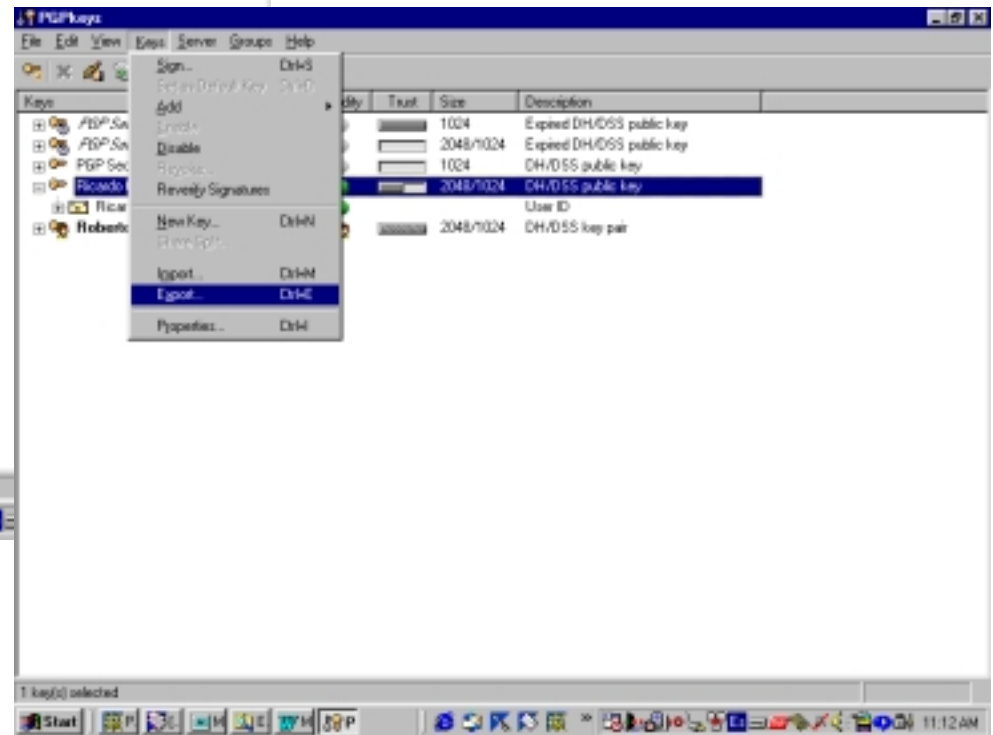
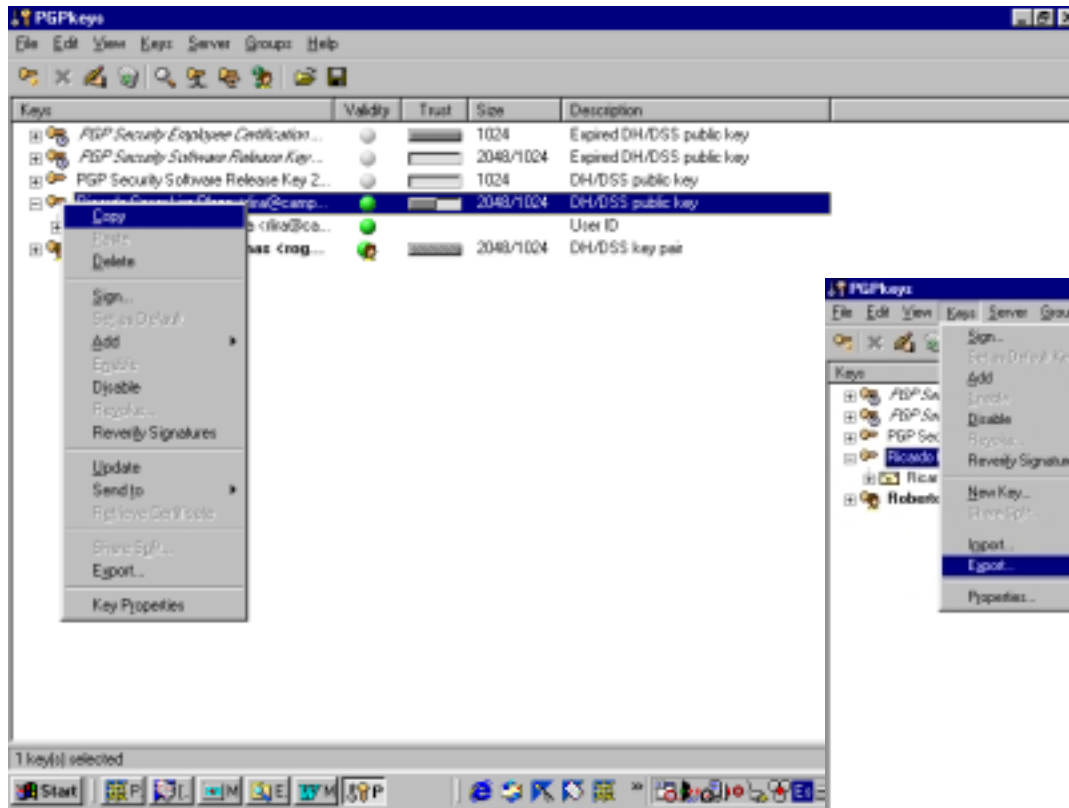
Version: PGPfreeware 7.0.3 for non-commercial use <<http://www.pgp.com>>

```
mQGIBDsVdARBADaMEJC5APkTg7N8mSL1uBvwugX3qMEw12dPsUfAAbayhK9obV9
rapBewT+8d3Z6Pkacc4zeoaHMidaVwjkaHH9EQ4mHrzVaj2JS3bQu4YIDWfGB7f
ycKGpotoMIwhgVWrqtQbM5y6v/CZpowZ/LDgldeaACYwDvdzdE2dOVjRmwCg/8oY
2ShKzhwrwNeaGzvHWjMPqGsEAMmTPqPmG2+ZWojI4NywP0yfUodrilJNzkou8Lg0
TxkWz4B24km2q+JCxDC22Za6/7FI6FjdKFxsvQCOH3h2H9KrohyG8UOIea/xiFeO
GKQOZgJpQkpS6z74JD0uBzUVhf8W0BzVFOfygJEWcNBJBwwUm+PQD6nWsslg22uE
Wkg+BACLIOgIgMQHEyVbS7nQtUK+2++kkuafMIHkhD1ir66qT4Z2YFmx8dYwHhuH
5WW7Q6XIZ4fYZwNxWGYF5HOnKP1mWg7OKSBTjWqnooimGI+6o+nivPdncZP0eS6U
aTv38iH3omjVuH7q4xU021d10axmqpKYP0kWt1NoCwJXcrOeW7Q0Um9iZXJ0byBH
821leiBD4XJkZW5hcyA8cm9nb21lekBjYW1wdXMuY2VtLm0ZXNtLm14PokAWAQQ
EQIAGAUCOxK90AgLAWkIBwIBCgIZAQUBAwAAAAAKCRBXbZv9ILL0Jd+LAJ0Tr2vl
4fhZ5uC9iFwOQfSONq8lwwCdHNFcmkRCfyT73uRbAj6RPj1GvEW5Ag0EOxK90BAI
APZCV7clfwgXcqK61qlC8wXo+VMROU+28W65Szzg2gGnVqMU6Y9AVfPQB8bLQ6mU
rfdMZIZJ+AyDvWXpF9Sh01D49Vlf3HZSTz09jdvOmeFXklnN/biudE/F/Ha8g8VH
MGHOfMlm/xX5u/2RXscBqtNbno2gpXI61Brwv0YAWCv19Ij9WE5J280gtJ3kkQc2
azNsOA1FHQ98iLMcfFstjvbySPAQ/CiWxiNjrtVjLhdONM0/XwXV0OjHRhs3jMh
LLUq/zzhsSIAGBGNfISnCNLWhsQDGcgHKXrKlQzZlp+r0ApQmwJG0wg9ZqRdQZ+c
fL2JSylZJrql7DVeKyCzsAAgIIAKci2FNty+7XFOoaMJ7CNYS556Kx0nHfIYWP
b+qw46TXBTtNnDJ0RiT/G2vePo3nL6FgaHQ/SJsKoFZvbpSeM1hTgAR97VE5y0j6
iJC1u9tM9B48ccAHIhpQLiyv49TAxk/Tp8buWjornM5+FH5J6ZCb5mikVRdtdQ8
iWQPjAuWTxnHUNEGwYah1GJK6C+yZZT90EDprvb8MqYkkBfgiI5Fd2Lmh6Nsppqz
bXAJw8u36HUJkdCZ6lKWUy4EMP09X1TNFBHYd3IcE+34F1kKvTbz1syoSqyxYSf4
5x3H6uMZmDbRYKSc+rYwerAhVu4fkAXIwGisUewS/ZsvTaSPLpGJAEwEGBECAAwF
AjsSvdAFGwwAAAAACgkQV22b/ZSy6CWUCACgxIbD3+x/vMCNXPXBLwp/4XvKvWgA
oJxAsML1N7AgOkFBTZk5IXiUN65t
=VYg2
```

-----END PGP PUBLIC KEY BLOCK-----



Opciones para exportar llaves públicas





Importando llaves públicas de servidores



- Abrir PGPkeys
- Del menu de Server seleccionar Search
- Seleccionar el servidor
- Introducir opciones busqueda
- Dar click a Search
- Con el botón derecho seleccionar la llave adecuada y seleccionar importar al llavero local



Ejemplo importación



PGPkeys

File Edit View Keys Server Groups Help

PGPkeys Search Window

Search for keys on: where

User ID: contains

Key ID: is

Keys	Validity	Trust	Size	Description
PGP Security Employee Certification...			1024	Expired DH/DSS public key
PGP Security Software Release Key...			2048/1024	Expired DH/DSS public key
PGP Security Software Release Key 2...			1024	DH/DSS public key
Ricardo Cesar Lira Plaza <rlira@camp...			2048/1024	DH/DSS public key
Ricardo Cesar Lira Plaza <rlira@ca...				User ID

Keys	Validity	Trust	Size	Description
Anderson Roberto Grella <argrella@u...			2048/1024	DH/DSS public key
Audrey Roberto Beloto Baldin <audrey...			2048/1024	DH/DSS public key
Bisanti Roberto <rbisanti@tiscali.net...			2048/1024	DH/DSS public key
Bombo Roberto <_bombo@hamm.net...			2048/1024	DH/DSS public key
bucalo roberto <bucalo@cad.it>			1024/1024	DH/DSS public key
Carlos Roberto <carlos@hotmail.com>			3072/1024	DH/DSS public key
Carlos Roberto Carraro <carlos_r@ser...			1536/1024	DH/DSS public key
Carlos Roberto Grieco de Moraes <gm...			2048/1024	DH/DSS public key
Charles Roberto Canato <chdr@usa...			2048/1024	Revoked DH/DSS public key
Charles Roberto Pilger <cpilger@hotm...			2048/1024	DH/DSS public key
Claudio Roberto Carvalho <carvalho...			1024/1024	DH/DSS public key
Clemente Roberto Garcia <clmense...			2048/1024	Expired DH/DSS public key

1 key(s) se... Search exceeded server limits. First 364 keys are shown.

Start



Respaldando llaves



- Exportar llave a un archivo o a una ubicación segura (CD, diskette, zip, etc.)
- O
 - respaldar archivos del llavero
 - copiar pubring.pkr
 - copiar secring.pkr
- Tener cuidado en la selección de la ubicación.
- No hay backdoor en PGP, si se pierden las llaves
NO se puede recuperar la información encriptada.



¿¿Y en Unix???



- Agregar una llave al llavero

```
pgp -ka archivo_llave [llavero]
```

- Suprimir una llave del llavero

```
pgp -kr usuario_id [llavero]
```

- Copiar una llave del llavero

```
pgp -kx usuario_id archivo_llave  
[llavero]
```

```
pgp -kax usuario_id archivo_llave  
[llavero]
```



¿¿Y en Unix...???



- Ver el contenido del llavero

```
pgp -kv[v] [usuario_id] [llavero]  
pgp keyfile
```



Validando llaves



- Abrir PGPkeys
- Seleccionar la llave a verificar
- Presionar botón derecho
- Seleccionar Key Properties
- Contactar al dueño de la llave
 - asegurarse de verificar su identidad
- Verificar la huella (fingerprint)
 - a través de las palabras
 - a través del número hexadecimal



Valiando llaves con palabras



The screenshot shows the PGPkeys application window. The left pane displays a list of keys with columns for Keys, Validity, Trust, and Size. A context menu is open over the selected key, showing options like Copy, Paste, Delete, Sign..., Set as Default, Add, Enable, Disable, Revoke..., Reverify Signatures, Update, Send to, Retrieve Certificate, Share Split..., and Export... The right pane shows the 'Key Properties' dialog for the selected key, displaying details such as ID, Type, Size, Created, Expires, Cipher, and a list of words for the fingerprint.

Keys	Validity	Trust	Size
PGP Security Employee Certification ...			1024
PGP Security Software Release Key ...			2048/1024
PGP Security Software Release Key 2...			1024
Ricardo Cesar Lira Plaza <rlira@camp...			2048/1024
Ricardo Cesar Lira Plaza <rlira@ca...			2048/1024
Ricardo Cesar Lira Plaza <rlira@ca...			2048/1024

Key Properties for Ricardo Cesar Lira Plaza <rlira@campus.cem.itesm.mx>

General | Subkeys

ID: 0x3B35DE54
Type: DH/DSS
Size: 2048/1024
Created: 06/14/2001
Expires: Never
Cipher: CAST
☒ Enabled

Fingerprint:

reindeer	candidate	spearhead	Pandora
cowbell	publisher	indoors	component
buzzard	voyager	accrue	telephone
aardvark	belowground	stapler	misnomer
clockwork	conformist	tactics	equation

☐ Hexadecimal

Trust Model:

Invalid ☐ Valid ☐ Untrusted ☐ Trusted

Close Help



Firmando llaves



- Primero hay que verificar la llave
- Abrir PGPkeys
- Boton derecho de la llave seleccionada
- Seleccionar Sign
- Introducir la frase secreta
- Seleccionar ok



Ajustando el nivel de confianza



- Primero hay que verificar y firmar la llave.
- Decidir el nivel de confianza
 - como se adquirio y se verifico la llave
- Boton derecho en la llave y seleccionar la opción Key Properties
- Ajustar el indicador al nivel de confianza deseado.



Revocando llaves



- Abrir PGPkeys
- Seleccionar la llave a revocar
- Estar seguros de que se desea revocar dicha llave!!!
- Boton derecho en esa llave y seleccionar la opción Revoke
- Introducir la frase secreta de la llave
- Confirmar la acción
 - aparecerá una X en el icono de la llave
 - al descripción mostrará revocado



Particionando llaves



- Aplicando el concepto de secretos compartidos.
- Boton derecho de la llave que se desea particionar
- Selecciona Split
- En la caja de dialogo de split introducir los nombres de las personas que van a compartir las llaves y sus frases secretas
- Seleccionar el número requerido para reconstruir la llave
- Seleccionar la ubicación de los partes de la llave
- Distribuir las partes a sus propietarios



Encriptación/Decripción



- Encriptación
- Decripción
- Firmas
- Verificación de firmas
- Combinaciones



Encriptando y firmando correos (consejos generales)



- Asegurarse de que se tiene la llave del destinatario
- Realizar operaciones de validación y firma
- Si se firma, asegurarse de que el destinatario cuenta con una copia de su llave de firma
 - ellos deben realizar verificación y firma
- Seleccionar entre enviar via Attachment o de forma automatica via Outlook



Encriptando y firmando correos (usando MS Outlook)



- Crear un nuevo mensaje
- Seleccionar opciones Sign y Verify del mnei de PGP
 - usar el botón de la barra
- Introducir el mensaje (y añadir attachments)
- Introducir dirección del destinatario y enviar
- Seleccionar las llaves a usar para encriptar
 - las que pertenecen al recipiente
 - PGPkeys seleccionara la llave apropiada si puede
- Si se firma el archivo
 - seleccionar la llave para firmar e introducir la frase



Encriptando y formando correos (usando attach)



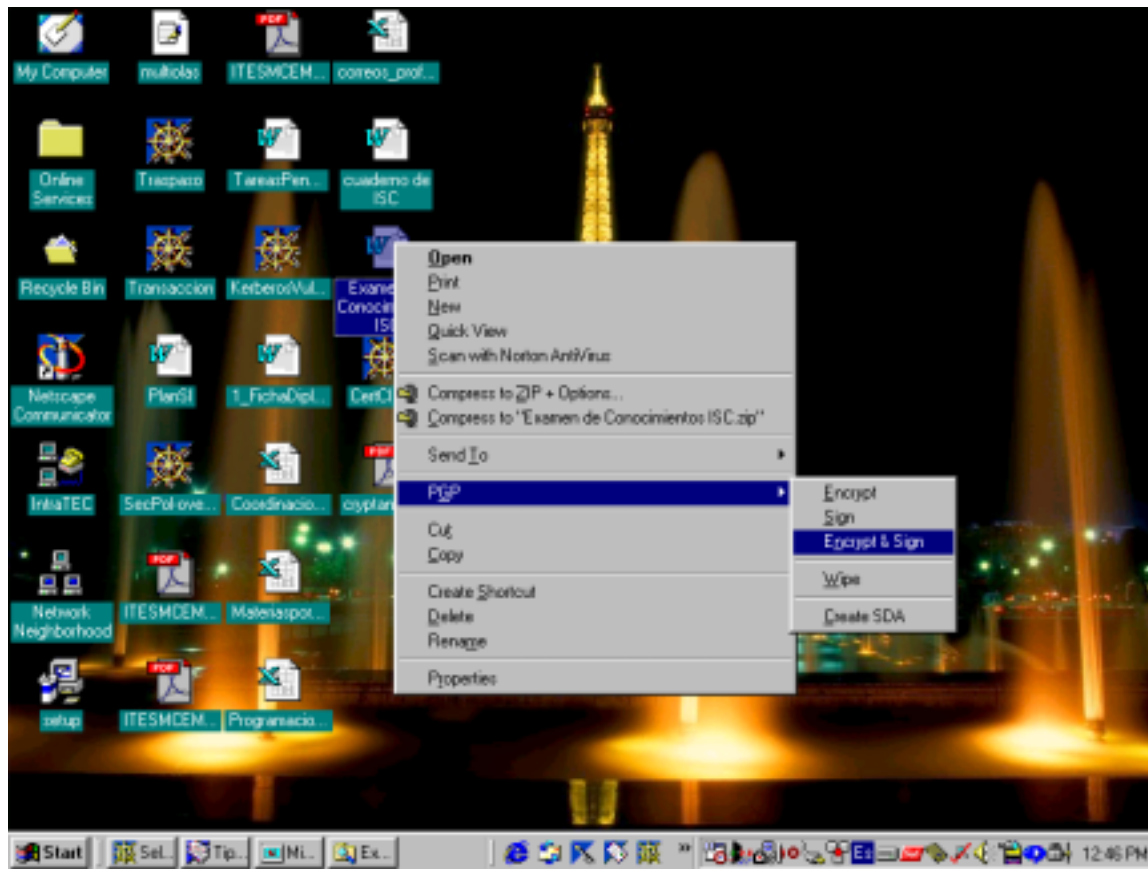
- Escribir mensaje en un archivo
 - asegurarse que el destinatario pueda leer dicho archivo (i.e. cuente con el programa)
- Del boton derecho del archivo seleccionar encrypt o encrypt and sign
- Seleccionar la(s) llave(s) a encriptar
- Seleccionar la llave de firma
- Introducir la frase de la llave con la que se va a firmar



Ejemplo encriptación/firma



- Attach el mensaje encriptado a su correo electrónico y enviarlo como normalmente se hace
 - solo el contenido del archivo en attach es seguro





Decriptando y verificando e-mail



- Determinar el metodo de envio
 - si el mensaje aparece como un archivo en attach con una extensión .asc
 - utilizar el método de attach del archivo



Decripción/verificación automática



- Abrir el mensaje en su correo electrónico
- Seleccionar Decrypt Verify del menu PGP
 - usar el button bar
- Introducir la frase secreta de la llave
- El mensaje decriptado debe aparecer en el cuadro de dialogo del correo



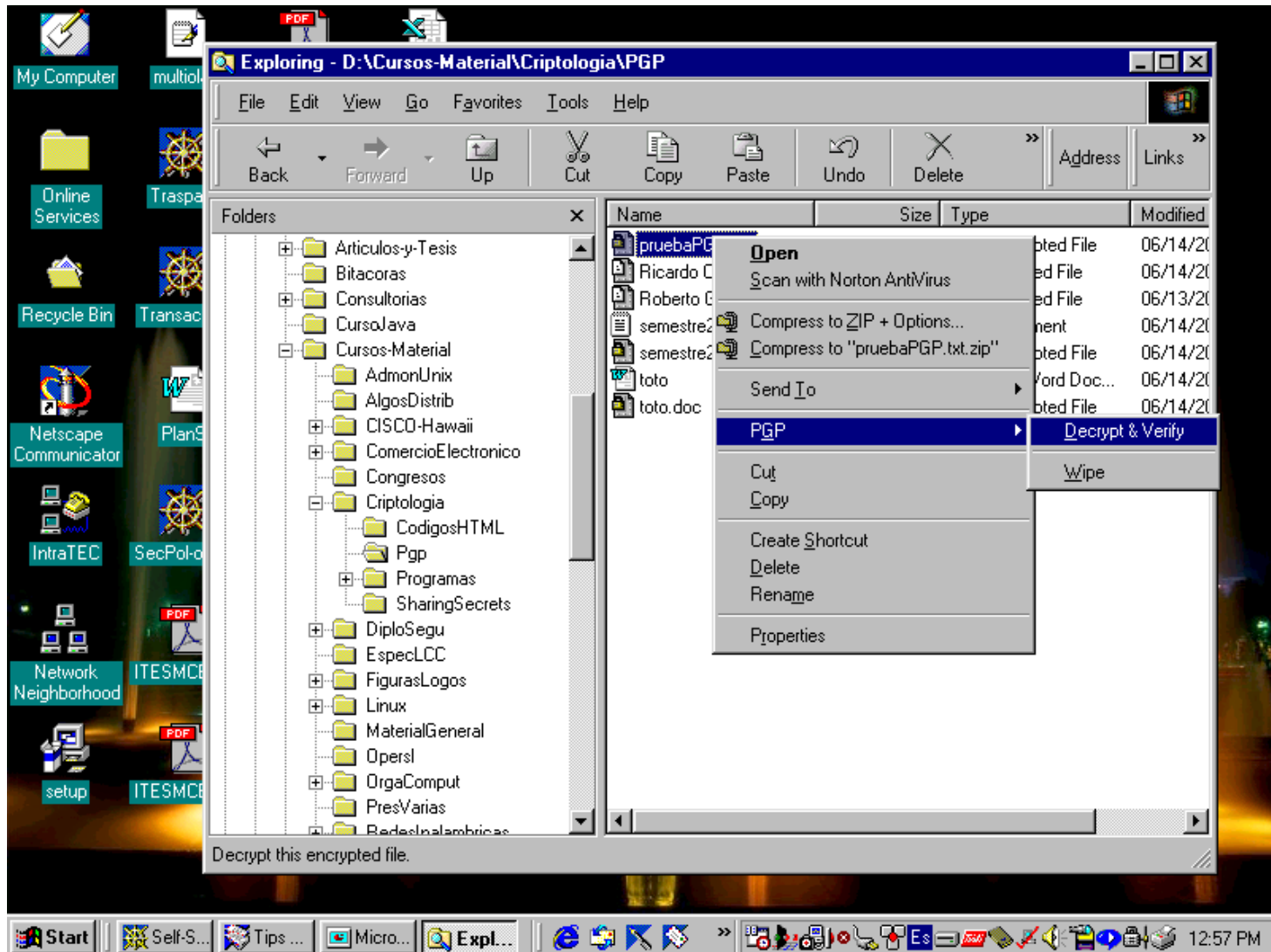
Decripción y verificación en attach



- Abrir el mensaje con su correo
- Copiar el archivo en attach al disco durto
- Botón derecho en el archivo
 - escoger Decrypt verify del menu de PGP
- Seleccionar la ubicación del archivo decriptado.
- Borrar el archivo (wipe) si así se desea



Ejemplo descripción





¿Y en Unix?



- Encriptado (convencional)

```
pgp -c archivo
```

- Decriptado (convencional)

```
pgp archivo [-o arch_salida]
```

- Encriptado (llave pública)

```
pgp -e archivo receptor_id
```

- Decriptado (llave pública)

```
pgp archivo [-o arch_salida]
```



¿Y en Unix...?



- Firma de un documento
`pgp -s documento [-u tu_id]`
- Comprobación de la firma
`pgp archivo [-o arch_salida]`
- Firma y encriptado de un documento
`pgp -se documento receptor_id [-u tu_id]`
- Comprobación de la firma y decriptado
`pgp archivo [-o arch_salida]`



Reuniendo llaves



- Decriptar o firmar usando una llave particionada.
- Aparece cuadro dialogo de Key Share Collection
- Click sobre los archivos compartidos
- Seleccionar los archivos .shf a ser usados
- Introducir la frase secreta
- Repetir seleccionando los archivos e introduciendo el resto de las frases.



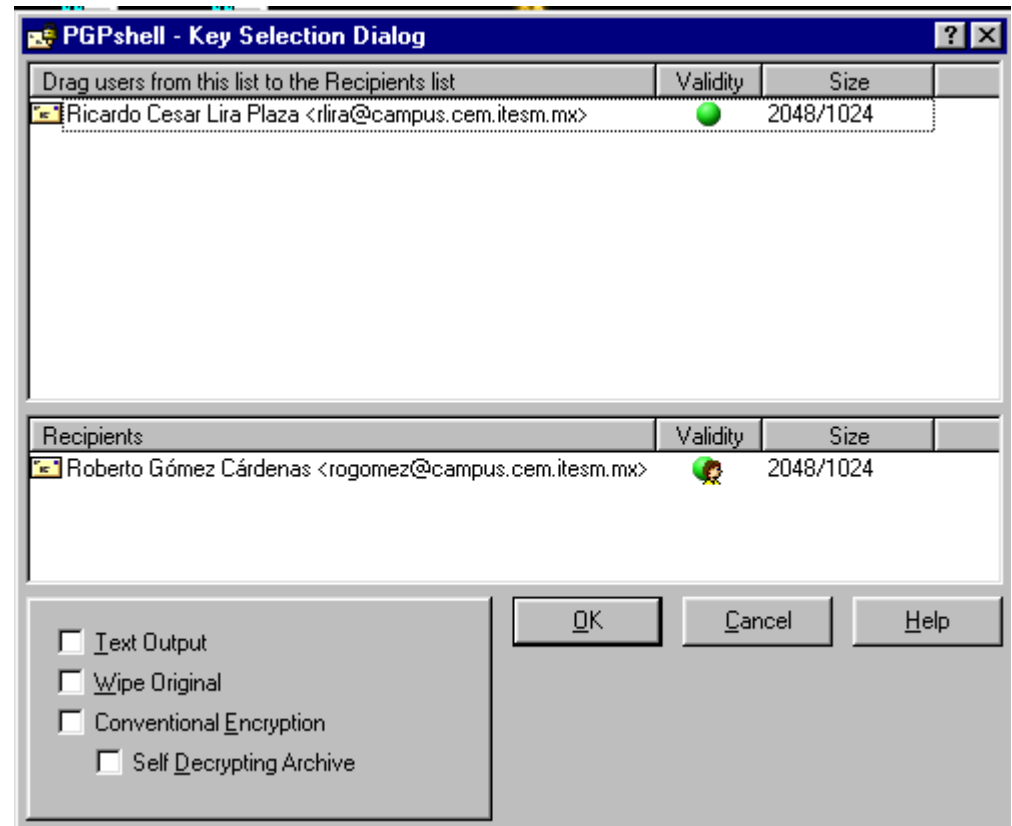
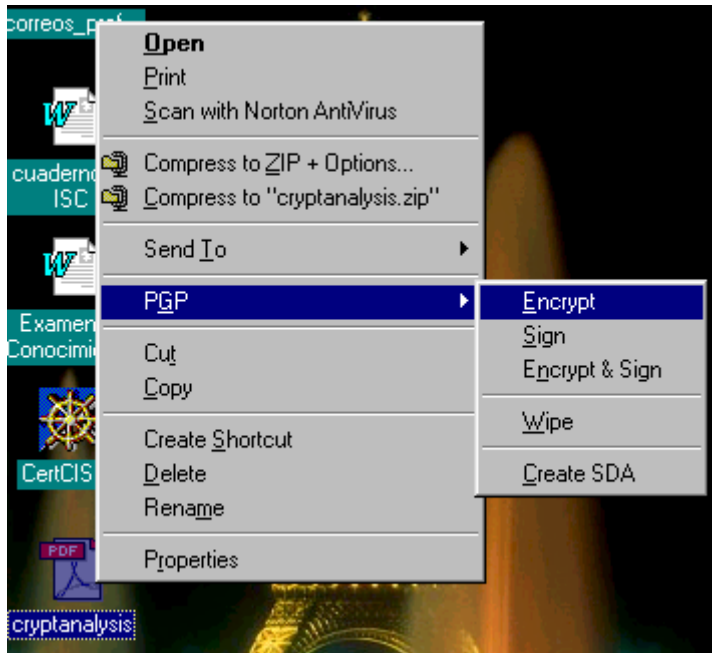
Uso de criptologia convencional



- Destinatario no necesita conocer PGP
- Seleccionar archivo a encriptar
- Boton derecho sobre el archivo y seleccionar Encrypt del menú de PGP
- Cuando aparesca la llave seleccionar opciones:
 - Use conventional encryption
 - Self Decrypting Archive
- Introducir la frase secreta usada y encriptar el archivo.
 - necesario que los dos conozcan la frase



Ejemplo encriptación convencional





Uso de borrado seguro



- Asegurarse de que el archivo no se necesita.
- Boton derecho del archivo a borrar
- Seleccionar Wipe del Menu de PGP
- Click OK
- Es posible ajustar el número de pasadas del General Tab del menú de opciones de la ventana de dialogo.
 - elegir Options del menu Edit de PGPkeys
 - más es mejor y más lento



Desventajas PGP



- Fuera de Estados Unidos, debe usarse la versión internacional.



- En Estados Unidos, no puede usarse la versión internacional.





Integrando PGP al correo electrónico



- PGP proporciona plug-ins para integrar PGP a los programas de correo más comunes:
 - Microsoft Outlook 97/98/2000,
 - Microsoft Outlook Express 4.x/5.x, Qualcomm Eudora 4.x
 - Claris EMailer 2.x.
- Para usuarios de Emacs en sistemas Unix, existe un Mailcrypt disponible en:
 - <http://cag-www.lcs.mit.edu/mailcrypt/>
- El MIT pone a la disposición de todo el mundo un servidor de llaves públicas PGP.



Integrando PGP al correo electrónico



- Para usar el correo MH de Unix, exmh es una interfaz de sistema de X Windows para el programa de correo MH que proporciona soporte PGP.
- Offline AutoPGP es un paquete de encriptación de correo electrónico para ser usado con PGP y lectores de correo fuera de línea en máquinas DOS.



Bajo la lupa



- Se usa la llave actual (pública/privada) seleccionada para encriptar el mensaje.
- No realmente:
 - Llave pública es muy lenta para encriptar el mensaje.
 - Las llaves DH o RSA son usadas para “negociar” una llave de sesión.
 - Llaves de sesión son usadas para encriptar el mensaje.



VPNs y PGP



- VPN: Virtual Private Network
 - es un canal de comunicación seguro definido sobre un medio inseguro de comunicación (generalmente Internet)
- PGPnet
 - posible definir una VPN entre dos organismos
- Posible crear un VPN a nivel
 - host
 - subred
 - gateway
- Posible definir un intercambio de llaves en condiciones seguras.



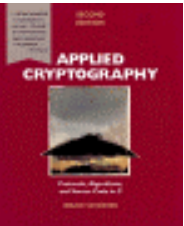
VPNs y PGP



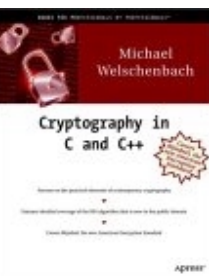
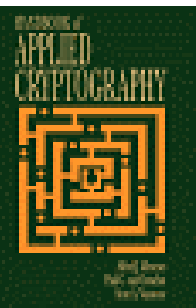
- Posibilidad de bloquear comunicaciones, activar bitacoras (logs), basado en el concepto de SA (Security Association)
 - acuerdo que contiene los terminos para establecer una comunicación segura entre dos máquinas
 - se crea la primera vez que una máquina se conecta con otra
 - describe como una máquina se va a comunicar con otra: tipo de encriptación, duración de la asociación y metodo de autenticación



Referencias/bibliografía

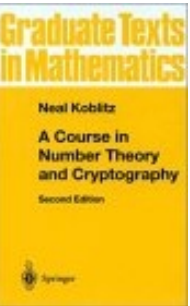


- *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Bruce Schneier, John Wiley & Son, 2da edición, 1995,
- *Handbook of Applied Cryptography (CRC Press Series on Discrete Mathematics and Its Applications)*, Alfred J. Menezes, Paul C. Van Oorschot (Editor), Scott A. Vanstone (Editor), CRC Press, 1996,
- *Cryptography in C and C++*, Michael Welschenbach, Bk&Cd-Rom edition, Apress, 2001

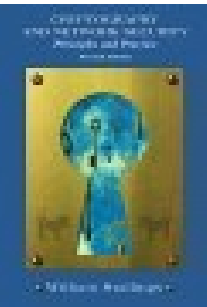




Más libros ...



- *A Course in Number Theory and Cryptography* (*Graduate Texts in Mathematics, No 114*), Neal I. Koblitz, Springer Verlag, 2nd edition, 1994



- *Cryptography & Network Security: Principles & Practice*, William Stallings, Prentice Hall, 2nd edition, 1998,



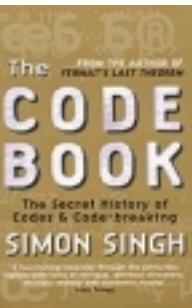
- *Network Security: Private Communication in a Public World*, Charlie Kaufman, Radia Perlman, Mike Speciner, Prentice Hall, 1995



Para leer en casa



- *The Codebreakers, The Comprehensive History of Secret Communication from Ancient Times to the Internet*, David Kahn, Scribner, 1996, (Revised edition)



- *The code book, The Science of Secrecy from Ancient Egypt to Quantum Cryptography*, Simon Singh, Anchor Books, 2000



- *Secrets and Lies : Digital Security in a Networked World*, Bruce Schneier, John Wiley & Sons, 2000