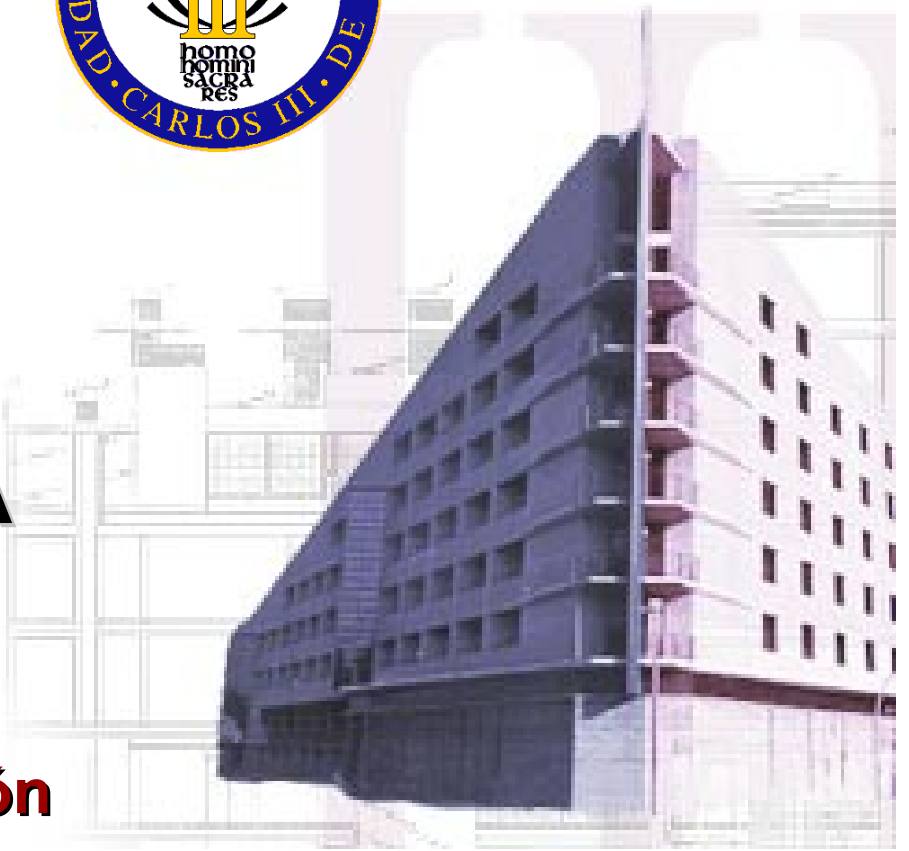


TEMA 4: CIFRADO CON CLAVE PÚBLICA

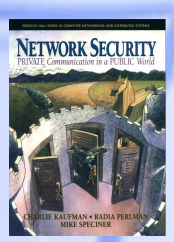
Técnicas de codificación

Mario Muñoz Organero

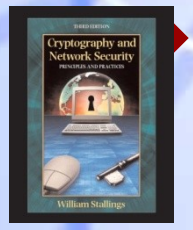
Curso 2009-2010

Departamento de INGENIERÍA TELEMÁTICA
www.it.uc3m.es

Bibliografía básica

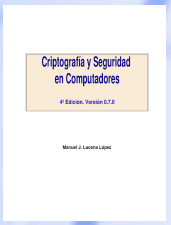


- **CHARLIE KAUFMAN, RADIA PERLMAN, MIKE SPECINER: Network Security: Private Communication in a Public World. Prentice Hall; Primera edición (Marzo, 1995). ISBN: 0130614661** ***[L/S 004.056 KAU]***
  - ***Capítulo 2: 2.5, Capítulo 5: 5.2, Capítulo 5: 5.3 y Capítulo 6***

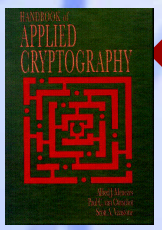


- **[2] STALLINGS, WILLIAMS: Cryptography and network security. Principles and practice. Cuarta Edición. Prentice Hall 2006. ISBN:0131873164 http://williamstallings.com/** ***[L/S 004.7 TAN]***
  - ***Capítulo 4, 8 y 9***

Bibliografía complementaria



- [3] LUCENA, J. MANUEL: Criptografía y seguridad en computadores. Cuarta edición.
  http://wwwdi.ujaen.es/~mlucena/bin/cysec4.zip
  - Capítulos 5, 6 y 12



- [4] MENESES, ALFRED: Handbook of applied cryptography. CRC Press. 1996.
  http://www.cacr.math.uwaterloo.ca/hac/
  - Capítulos 2 y 8

Bibliografía específica para EC

- Implementing Elliptic Curve Cryptography por Michael Rosing
- Guide to Elliptic Curve Cryptography por Darrel Hankerson
- Elliptic Curves in Cryptography por I. Blake

Índice del tema

- Introducción a la criptografía en clave asimétrica.
- Base matemática previa.
- Algoritmos basados en exponenciación:
  - RSA
  - ElGamal
- Curvas elípticas.
- Criptografía basada en curvas elípticas.
  - ElGamal
  - Diffie-Hellman

Introducción

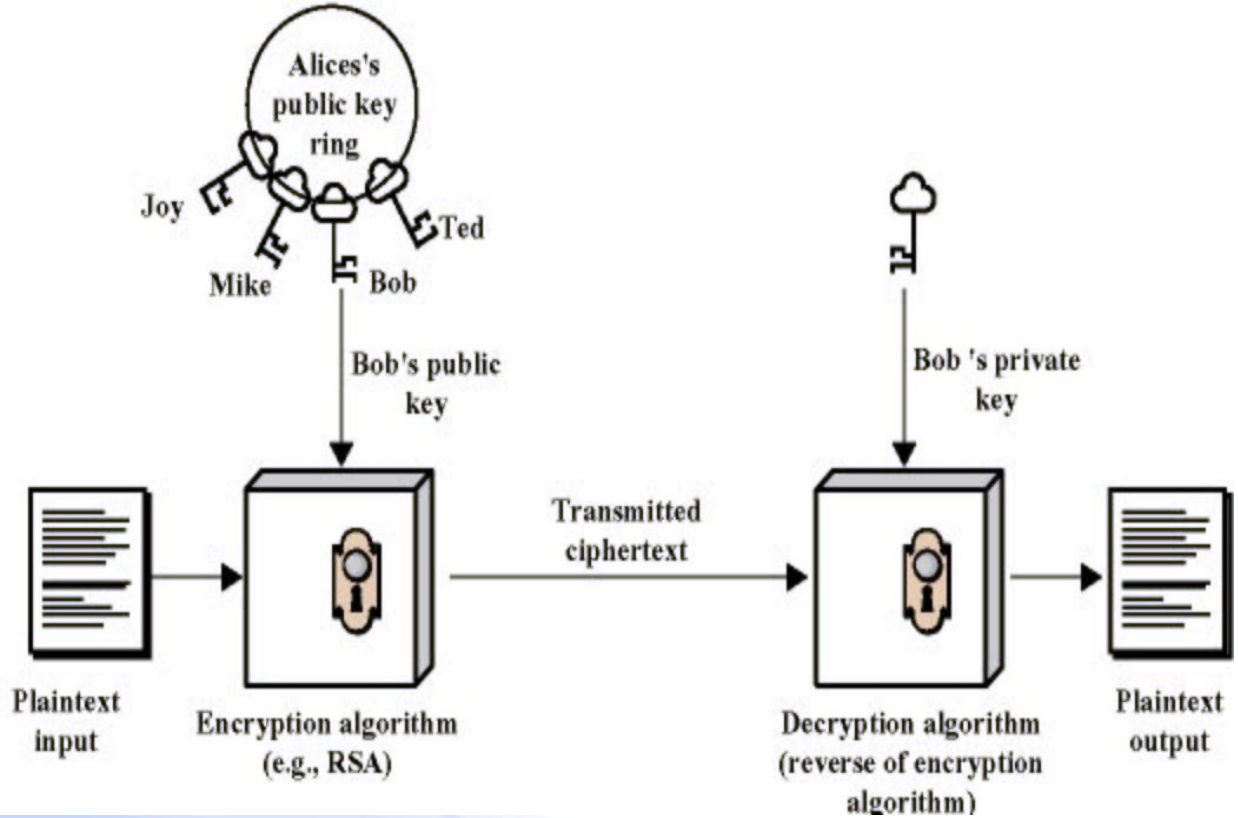
- Los algoritmos de clave pública fueron introducidos por Whitfield Diffie y Martin Hellman a mediados de los años 70 (1976) y su novedad fundamental con respecto a la criptografía simétrica es que las claves no son únicas, sino que forman pares.
- La longitud de las claves suele ser mayor que las usadas en algoritmos simétricos (e.g. 1024 bits).
- Los algoritmos asimétricos poseen dos claves diferentes en lugar de una, k_{pr} y K_{PU} , denominadas clave privada y clave publica (una de ellas se emplea para decodificar, mientras que la otra se usa para codificar).

Introducción

- Para que los criptosistemas basados en clave pública sean seguros ha de cumplirse que a partir de una de las claves resulte extremadamente difícil calcular la otra.
- Dos de las aplicaciones principales de estos algoritmos son:
  - Cifrar (nos permitirá la confidencialidad).
  - Autenticar (será la base para las firmas digitales).

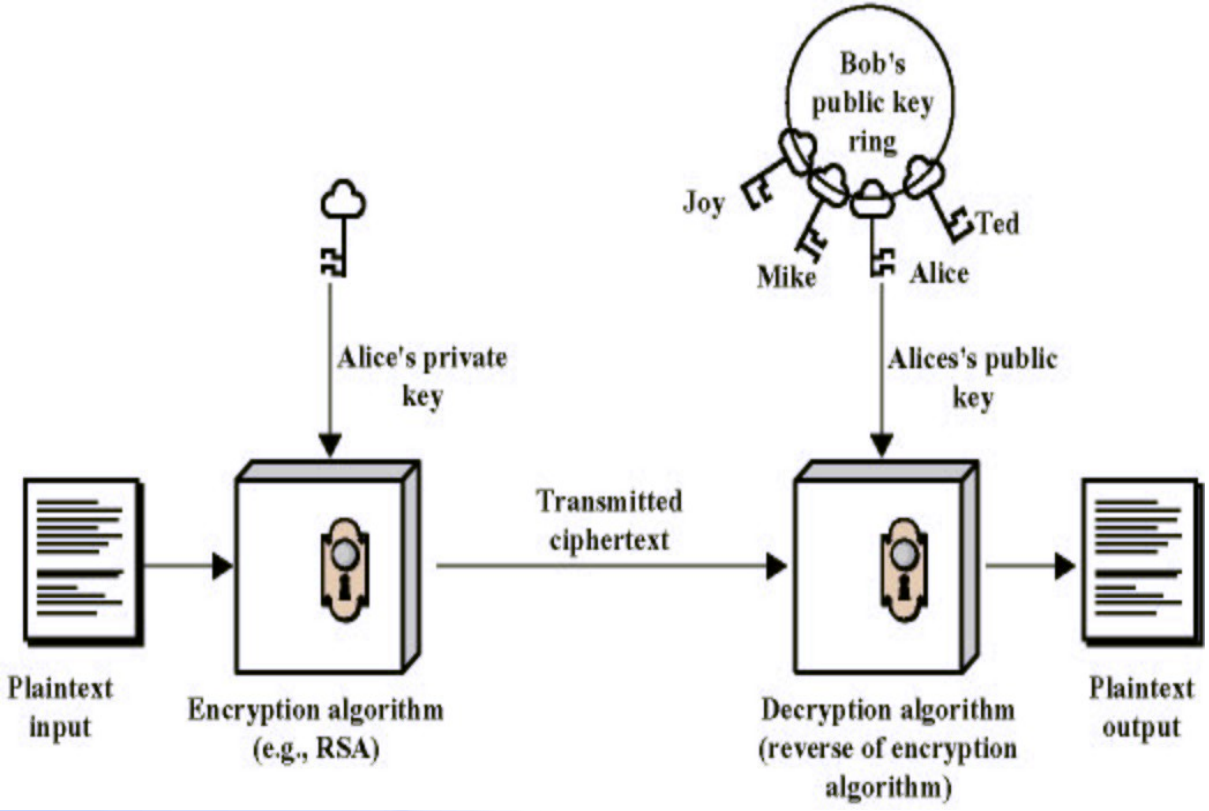
Cifrado con algoritmos asimétricos

- La siguiente figura recoge el proceso de cifrado:



Autenticación con algoritmos asimétricos

- La siguiente figura recoge el proceso de autenticación:



Clasificación de algoritmos asimétricos

- Existen dos familias principales de algoritmos asimétricos de aplicación en criptografía en función de las funciones matemáticas usadas:
  - Basados en exponenciación (primera parte del tema).
  - Basados en curvas elípticas (segunda parte del tema).

Base matemática previa para algoritmos en exponenciación.

Departamento de INGENIERIA TELEMATICA
www.it.uc3m.es

No perdamos el norte

- **Las bases:**
  - ¿Qué queremos hacer? Pues alguna operación criptográfica como cifrar o firmar.
  - ¿Sobre qué lo queremos hacer? Sobre un mensaje o documento.
- **Pues bien en criptografía asimétrica basada en exponenciación tendremos:**
  - El mensaje no será más que una concatenación de números (truncando el mensaje en bloques de n bits).
  - La criptografía se basará en aplicar operaciones (basadas en exponenciación) a cada uno de estos números

Necesito un conjunto finito de textos cifrados → Concepto de congruencia

- Es la base matemática (matemáticas discretas) en la que se sustentan las operaciones de cifrado asimétrico basado en exponenciación.
- Concepto de congruencia:
  - Sean dos números enteros a y b: a es congruente con b en el módulo o cuerpo n en Z_n si y sólo si existe algún entero k que divide de forma exacta la diferencia (a - b)

$$a - b = k * n$$
$$a \equiv_n b$$
$$a \equiv b \bmod n$$

Conjunto completo de restos CCR

Para cualquier entero positivo n, el conjunto completo de restos será CCR = {0, 1, 2, ... n-1}, es decir:

$$\forall\, a \in Z \quad \exists\, ! \, r_i \in CCR \,/\, a \equiv r_i \bmod n$$

$$CCR\,(11) = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

$$CCR\,(6) = \{0, 1, 2, 3, 4, 5\} = \{12, 7, 20, 9, 16, 35\}$$

El segundo conjunto es equivalente: $12 \rightarrow 0$, $7 \rightarrow 1$...

Cálculo del mcd

En criptografía muchas veces nos interesará encontrar el máximo común denominador mcd entre dos números *a* y *b*.

Por ejemplo: para la existencia de inversos en un cuerpo n, la base a y el módulo n deberán ser primos entre sí. $\Rightarrow$ mcd (a, n) = 1

Algoritmo de Euclides:

- **a) Si x divide a *a* y *b*** $\Rightarrow$ $a = x * a'$ y $b = x * b'$
- **b) Por lo tanto:** $a - k * b = x * a' - k * x * b'$

  $a - k * b = x (a' - k * b')$
- **c) Entonces se concluye que x divide a $(a - k * b)$**

Algoritmo de Euclides

- **Si x divide a "a" y "b" ⇒**
  - ***x dividirá a "a mod b" (a - k * b)***
  - ***x dividirá a "b mod a" (b - k * a)***
- **Algoritmo:**
  - **Si a > b → r_0 = a y r_1 = b**
  - **calcular r_2 = a mod b, ie: $r_2 = r_0$ mod r_1**
  - **Iterar $r_n = r_{n-2}$ mod r_{n-1}**
  - **El último resto antes de llegar a 0 será el mcd (a, b)**

Divisibilidad con algoritmo de Euclides

mcd (148, 40)

$148 = 3 * 40 + 28$

$40 = 1 * 28 + 12$

$28 = 2 * 12 + 4$

$12 = 3 * 4 + 0$

mcd (148, 40) = 4

$148 = 2^2 * 37$

$40 = 2^3 * 5$

Factor común

$2^2 = 4$

No hay factor común

$385 = 5 * 7 * 11$

$78 = 2 * 3 * 13$

Será importante en criptografía

mcd (385, 78)

$385 = 4 * 78 + 73$

$78 = 1 * 73 + 5$

$73 = 14 * 5 + 3$

$5 = 1 * 3 + 2$

$3 = 1 * 2 + 1$

$2 = 2 * 1 + 0$

mcd (385, 78) = 1

Inversos multiplicativos mod n

- En criptografía deberá estar permitido invertir una operación para recuperar un cifrado $\Rightarrow$ descifrar.
- Si bien la cifra es una función, en lenguaje coloquial la operación de cifrado sería una "multiplicación" y la operación de descifrado una "división".
- La analogía anterior sólo será válida en el cuerpo de los enteros Z_n con inverso.
- Luego, si en una operación de cifra la función es el valor *a* **dentro** de un cuerpo n, deberemos encontrar el inverso a^{-1} **mod n** para descifrar; en otras palabras ...

Inversos multiplicativos mod n

- Si $a * x \bmod n = 1$
  - *x será el inverso multiplicativo (a^{-1}) de a en el módulo n*
- No siempre existen los inversos. En realidad lo raro es que existan.
- Veamos en la siguiente transparencia cuándo existen inversos.

Existencia del inverso por primalidad

$\exists$ inverso a^{-1} en mod n *ssi* mcd (a, n) = 1

Si mcd (a,n) = 1, el resultado de a∗i mod n (para i todos los restos de n) serán valores distintos dentro del cuerpo n.

mcd (a, n) = 1 $\Rightarrow$ $\exists$ x ! 0<x<n / a ∗ x mod n = 1

Sea: a = 4 y n = 9. Valores de i = {1, 2, 3, 4, 5, 6, 7, 8}

4∗1 mod 9 = 4 4∗2 mod 9 = 8 4∗3 mod 9 = 3

4∗4 mod 9 = 7 4∗5 mod 9 = 2 4∗6 mod 9 = 6

4∗7 mod 9 = 1 4∗8 mod 9 = 5

Inexistencia de inverso (no primalidad)

¿Y si no hay primalidad entre a y n?

Si mcd (a, n) ≠ 1

No existe ningún x que $0 < x < n \ / \ a * x \bmod n = 1$

Sea: a = 3 y n = 6 Valores de i = {1, 2, 3, 4, 5}

$3*1 \bmod 6 = 3$ $3*2 \bmod 6 = 0$ $3*3 \bmod 6 = 3$

$3*4 \bmod 6 = 0$ $3*5 \bmod 6 = 3$

No existe el inverso para ningún resto del cuerpo.

Si n=10

- ¿Cuántos números tendrán inverso multiplicativo?

(A∗B) mod 10

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

10=2*5

=(2-1)*(5-1)

Conjunto reducido de restos CRR (1)

- El conjunto reducido de restos, conocido como CRR de n, es el subconjunto {0, 1, ... n_i, ... n-1} de restos primos con el grupo n.
- Si n es primo, todos los restos serán primos con él.
- Como el cero no es una solución, entonces:

$$CRR = \{1, ..., n_i, ... n-1\} \ / \ mcd\ (n_i, n) = 1$$

Ejemplo: CRR mod 8 = {1, 3, 5, 7}

CRR mod 5 = {1, 2, 3, 4}

Conjunto reducido de restos CRR (2)

- ¿Qué utilidad tiene esto en criptografía?

- El conocimiento del CRR permitirá aplicar un algoritmo para el cálculo del inverso multiplicativo de un número x dentro de un cuerpo o grupo n a través de la función $\phi(n)$, denominada Función de Euler o Indicador de Euler.

Función de Euler ϕ(n)

- **Función ϕ(n) de Euler**
- **Entregará el número de elementos del CRR.**
- **Podremos representar cualquier número n de estas cuatro formas:**
  - a) n es un número primo.
  - b) n se representa como $n = p^k$ con p primo y k entero.
  - c) n es el producto $n = p*q$ con p y q primos.
  - d) n es un número cualquiera (genérico).

$$n = p_1^{e1} * p_2^{e2} * \ldots * p_t^{et} = \prod_{i=1}^{t} p_i^{ei}$$

Función ϕ(n) de Euler (n = p)

Caso 1: n es un número primo

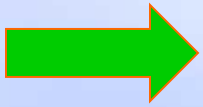
Si n es primo, ϕ(n) será igual a CCR menos el 0.

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

Se usará en sistemas ElGamal y DSS

Si n es primo, entonces CRR = CCR - 1 ya que todos los restos de n, excepto el cero, serán primos entre sí.

Ejemplo



CRR(7) = {1,2,3,4,5,6} seis elementos
∴ϕ(7) = n - 1 = 7-1 = 6
ϕ(11) = 11-1 = 10; ϕ(23) = 23-1 = 22

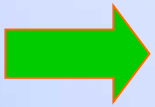
Función $\phi(n)$ de Euler ($n = p^k$)

Caso 2: $n = p^k$ (con p primo y k un entero)

$$\phi(n) = \phi(p^k) = p^k - p^{k-1} \qquad \phi(p^k) = p^{k-1}(p-1)$$

De los p^k elementos del CCR, restaremos todos los múltiplos $1*p$, $2*p$, $3*p$, ...$(p^{k-1}-1)*p$ y el cero.

Ejemplo



CRR(16) = {1,3,5,7,9,11,13,15} ocho elementos

$\therefore \phi(16) = \phi(2^4) = 2^{4-1}(2-1) = 2^3*1 = 8$

$\phi(125) = \phi(5^3) = 5^{3-1}*(5-1) = 5^2*4 = 25*4 = 100$

Función ϕ(n) de Euler (n = p∗q) (1)

Caso 3: n = p∗q (con p y q primos)

$$\phi(n) = \phi(p*q) = \phi(p)*\phi(q) = (p-1)(q-1)$$

De los p∗q elementos del CCR, restaremos todos los múltiplos de p = 1∗p, 2∗p, ... (q - 1)∗p, todos los múltiplos de q = 1∗q, 2∗q, ... (p - 1)∗q y el cero.

$$\phi(p*q) = p*q - [(q-1) + (p-1) + 1] = \underbrace{p*q - q - p + 1}_{(p-1)(q-1)}$$

Función ϕ(n) de Euler (n = p*q) (2)

Ejemplo

CRR(15) = {1,2,4,7,8,11,13,14} ocho elementos

∴ ϕ(15) = ϕ(3*5) = (3-1)(5-1) = 2*4 = 8

ϕ(143) = ϕ(11*13) = (11-1)(13-1) = 10*12 = 120

Será una de las funciones más utilizadas ya que es la base del sistema RSA, durante muchos años un estándar de hecho.

Función ϕ(n) de Euler (n = genérico)

Caso 4: $n = p_1^{e1} * p_2^{e2} * \ldots * p_t^{et}$ (p_i son primos)

$$\phi(n) = \prod_{i=1}^{t} p_i^{ei-1} (p_i - 1)$$

(demostración no inmediata)

Ejemplo

CRR(20) = {1, 3, 7, 9, 11, 13, 17, 19} ocho elementos

$\therefore \phi(20) = \phi(2^2 * 5) = 2^{2-1}(2-1) * 5^{1-1}(5-1) = 2^1 * 1 * 1 * 4 = 8$

$\phi(360) = \phi(2^3 * 3^2 * 5) = 2^{3-1}(2-1) * 3^{2-1}(3-1) * 5^{1-1}(5-1) = 96$

Teorema de Euler

Dice que si mcd (a,n) = 1 $\Rightarrow$ $a^{\phi(n)} \bmod n = 1$

Ahora igualamos $a*x \bmod n = 1$ y $a^{\phi(n)} \bmod n = 1$

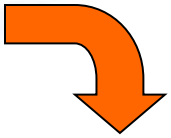
$\therefore$ $a^{\phi(n)} * a^{-1} \bmod n = x \bmod n$

$\therefore$ $x = a^{\phi(n)-1} \bmod n$

El valor x será el inverso de a en el cuerpo n

Nota: Observe que se ha ***dividido*** por **a** en el cálculo anterior. Esto se puede hacer porque mcd (a, n) = 1 y por lo tanto hay un único valor inverso en el cuerpo n que lo permite.

Cálculo de inversos con Teorema Euler

Ejemplo 

¿Cuál es el inverso de 4 en módulo 9? $\Rightarrow$ inv (4, 9)

Pregunta: ¿Existe a $*$ x mod n = 4 $*$ x mod 9 = 1?

Como mcd (4, 9) = 1 $\Rightarrow$ Sí ... aunque 4 y 9 no son primos.

$\phi(9) = 6 \;\therefore\; x = 4^{6-1} \bmod 9 = 7 \Rightarrow 7*4 = 28 \bmod 9 = 1$

Resulta obvio que: inv (4, 9) = 7 e inv (7, 9) = 4

Teorema de Euler para n = p*q

- Si el factor a es primo relativo con n y n es el producto de 2 primos, seguirá cumpliéndose el Teorema de Euler también en dichos primos.
- Por ejemplo:
  - Si $n = p*q \Rightarrow \phi(n) = (p-1)(q-1)$
  - $\forall$ a / mcd {a, (p,q)} = 1
  - se cumple que:
    - ✓ $a^{\phi(n)} \bmod p = 1$
    - ✓ $a^{\phi(n)} \bmod q = 1$

Ejemplo Teorema de Euler para n = p*q

- Sea n = p*q = 7*11 = 77
  - $\phi(n) = (p - 1)(q - 1) = (7 - 1)(11 - 1) = 6*10 = 60$
- Si k = 1, 2, 3, ...
  - Para a = k*7 $a^{\phi(n)} \bmod n = k*7^{60} \bmod 77 = 56$
  - Para a = k*11 $a^{\phi(n)} \bmod n = k*11^{60} \bmod 77 = 22$
  - Para $\forall$ a $\neq$ k*7,11 $a^{\phi(n)} \bmod n = a^{60} \bmod 77 = 1$
- Y se cumple que:
  - Para $\forall$ a $\neq$ k*7,11 $a^{\phi(n)} \bmod p = a^{60} \bmod 7 = 1$
  - $a^{\phi(n)} \bmod q = a^{60} \bmod 11 = 1$
- En caso contrario: $a^{\phi(n)} \bmod p = 0$
- $a^{\phi(n)} \bmod q = 0$

Reducibilidad

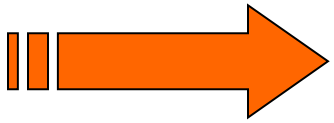
- Si queremos calcular $81^{30} \bmod 91$
  - ¿Cómo lo haremos?
- Directamente con calculadora de Windows:
  - 1,797010299914431210413179829509 6e+57 mod 91
- Aplicando reducibilidad:
  - $81^{30} \bmod 91 = (81^5 \bmod 91)^6 \bmod 91 = 9^6 \bmod 91 = 1$
- ¿Curioso?
  - $81^6 \bmod 91 = 1$

¿Qué hacemos si no se conoce ϕ(n)?

- Calcular a^i mod n cuando los valores de *i* y *a* son grandes, se hace tedioso pues hay que utilizar la propiedad de la reducibilidad repetidas veces.
- Si no conocemos ϕ(n) o no queremos usar el teorema de Euler/Fermat, siempre podremos encontrar el inverso de *a* en el cuerpo *n* usando el

Algoritmo Extendido de Euclides

Es el método más rápido y práctico



Algoritmo extendido de Euclides

- Se basa en el de Euclides pero toma en cuenta los cocientes además de los restos en cada iteración (i.e. Interesa tanto *a mod n* como *n/a*).
- El algoritmo realiza una serie de iteraciones tipo:

$$g_i = nu_i + av_i$$

- El último g_i será el mcd *(a,n)*. Si *a* y *n* son primos entre si se tendrá:

$$1 = nu_i + av_i$$

Cálculo de inversos con el AEE

Encontrar el inv (9, 25) por el método de restos de Euclides.

a) 25 = 2*9 + 7

b) 9 = 1*7 + 2

c) 7 = 3*2 + 1

d) 2 = 2*1 + 0

$7 = 25 - 2*9$

$2 = 9 - 1*7$

$1 = 7 - 3*2$

restos

$7 = 25 - 2*9$

$2 = 9 - 1*(25 - 2*9) = 3*9 - 1*25$

$1 = (25 - 2*9) - 3*(3*9 - 1*25)$

$1 = 4*25 - 11*9 \mod 25$

Tabla de Restos

	2	1	3	2	
25	9	7	2	1	0

El inv (9,25) = -11

-11 + 25 = 14

inv (9, 25) = 14

Algoritmo para el cálculo de inversos

Para encontrar x = inv (A, B)

Hacer $(g_0, g_1, u_0, u_1, v_0, v_1, i) = (B, A, 1, 0, 0, 1, 1)$

Mientras $g_i \neq 0$ **hacer**

- **Hacer** y_{i+1} = parte entera (g_{i-1}/g_i)
- **Hacer** $g_{i+1} = g_{i-1} - y_{i+1} * g_i$
- **Hacer** $u_{i+1} = u_{i-1} - y_{i+1} * u_i$
- **Hacer** $v_{i+1} = v_{i-1} - y_{i+1} * v_i$
- **Hacer** i = i+1

Si $(v_{i-1} < 0)$ **Hacer**

- **Hacer** $v_{i-1} = v_{i-1} + B$

x = inv (A, B)

x = inv (9, 25)

i	y_i	g_i	u_i	v_i
0	-	25	1	0
1	-	9	0	1
2	2	7	1	-2
3	1	2	-1	3
4	3	1	4	-11
5	2	0	-9	25

x = inv (9, 25) = -11+25 = 14

Ejemplo

Algoritmos basados en exponenciación

Departamento de INGENIERIA TELEMATICA
www.it.uc3m.es

Algoritmos basados en exponenciación

- Se basan en la peculiaridad que el cálculo de exponenciales es bastante más rápido que resolver logaritmos discretos o factorizar grandes números.
- Algunos de los algoritmos más extendidos son:
  - RSA
  - ElGamal (y su versión generalizada)
  - Rabin
  - McEliece
  - Merkle-Hellman
  - Chor-Rivest
  - Goldwasser-Micali
  - Blum-Goldwasser

RSA

- El algoritmo de RSA fue inventado en 1978 por Ron Rivest, Adi Shamir, y Leonard Adleman.
- Es uno de los más sencillos de comprender.
- Los pares de claves son duales por lo que sirve tanto para codificar como para autentificar.
- Se tiene por uno de los algoritmos asimétricos más seguros (aunque no se ha demostrado que no existan mecanismos para romperlo).
- Se basa en la dificultad para factorizar grandes números.
- Las claves pública y privada se obtienen a partir de un número que se calcula como producto de dos grandes números primos.

RSA

- **Las matemáticas de RSA se pueden expresar como:**
  1. **Encontrar dos números primos grandes *p* y *q* (e.g., 1024-bits).**
  2. **Calcular el producto *n=pq***
  3. **Elejir *e* tal que *e* es menor que *pq* , y tal que *e* y *(p-1)(q-1)* sean primos relativos *(*no tienen ningún factor primo común). *e* no tiene que ser primo pero si debe ser impar. *(p-1)(q-1)* no puede ser primo porque es un número par. *e* tendrá inversa módulo *(p-1)(q-1).***

RSA

4. Calcular *d* para que sea la inversa de *e* módulo (*p-1)(q-1) (i.e. de = 1 mod (p-1)(q-1))*.
5. La clave privada será (*d,n*).
6. La codificación se lleva a cabo según la expresión $c=m^e \pmod n$
7. La decodificación se hace como $m=c^d \pmod n$

RSA

- Así pues al decodificar obtenemos el mensaje:

$$c^d = (m^e)^d = m^{ed} = m^{k(p-1)(q-1)+1} = (m^k)^{(p-1)(q-1)} m$$

- Consideraciones:
  - p y q han de tener un número grande de bits
  - El atacante, si quiere recuperar la clave privada a partir de la pública debe conocer los factores p y q de n (complejo en coste computacional).

Ejemplo RSA

$n = 91 = 7*13$; $\phi(n) = \phi(7*13) = (7-1)(13-1) = 72$ M = 48

Elegimos e = 5 pues mcd (5,72) = 1 ∴ d = inv(5,72) = 29

CIFRADO:

$C = M^e \bmod n = 48^5 \bmod 91 = 5245.803.968 \bmod 91 = 55$

DESCIFRADO:

$M = C^d \bmod n = 55^{29} \bmod 91 = 48$ ... 55^{29} ya es "*número grande*"

55^{29} es un número con 51 dígitos...

55^{29} = 295473131755644748809642476009391248226165771484375

¿Cómo podemos acelerar esta operación?

1ª opción: *usar reducibilidad*

¿Algo más óptimo? Exponenciación rápida

Un método de exponenciación rápida

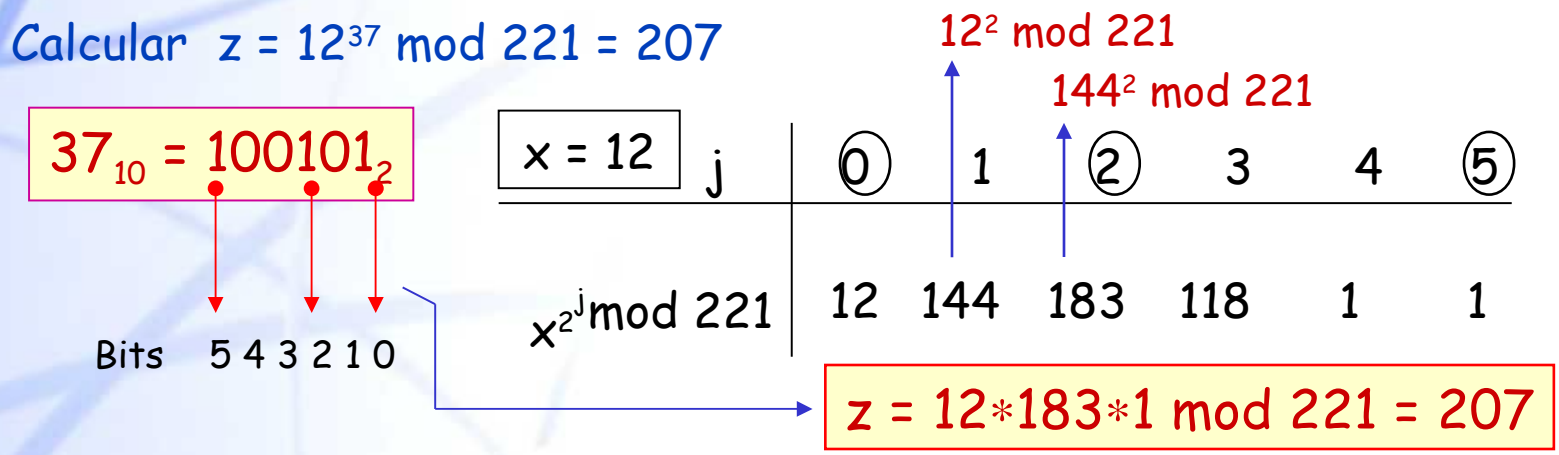
- En x^y mod n se representa el exponente *y* en binario.
- Se calculan los productos x^{2^j} con j = 0 hasta n-1, siendo n el número de bits que representan el valor *y* en binario.
- Sólo se toman en cuenta los productos en los que en la posición j del valor *y* en binario aparece un 1.

Ejemplo

Calcular $z = 12^{37} \bmod 221 = 207$

12^{37} es un número de 40 dígitos

Un método de exponenciación rápida



En vez de 36 multiplicaciones y sus reducciones módulo 221 en cada paso ... 72 operaciones...

Hemos realizado cinco multiplicaciones (para j = 0 el valor es x) con sus reducciones módulo 221, más dos al final y su correspondiente reducción. Un ahorro superior al 80% ☺.

La otra historia del algoritmo RSA

- Rivest, Shamir y Adleman son los autores de RSA pero un algoritmo de cifra asimétrico basado en la dificultad de factorizar números grandes como función unidireccional fue descubierto mucho antes...
- En el año 1969 el Government Communications Headquarters (GCHQ) en Gran Bretaña comienza a trabajar en la idea de poder distribuir claves a través de una cifra no simétrica. En 1973, el matemático Clifford Cocks llegará a la misma conclusión que los creadores de RSA.
- Desgraciadamente este trabajo fue considerado como alto secreto por el gobierno británico por lo que su contenido no se hace público ni se patenta como invento, algo que sí hacen Diffie y Hellman en 1976 con su intercambio de claves y en 1978 otro tanto los creadores del algoritmo RSA.

Elección de los números primos

- Los valores primos deben elegirse apropiadamente:
  - **a) p y q deben diferir en unos pocos dígitos.**
  - **b) p y q no deben ser primos muy cercanos.**
  - **c) Longitud mínima de p y q: 250 bits.**
  - **d) Valores de (p-1) y (q-1) del Indicador de Euler deben tener factores primos grandes.**
  - **e) El mcd entre p-1 y q-1 debe ser pequeño.**

Números primos seguros

Primos seguros: se elige r un primo grande de modo que:

$p = 2*r + 1$ y $q = 2*p + 1$ también sean primos

EJEMPLO: Si r es el primo de 4 dígitos 1.019:

$p = 2*1.019 + 1 = 2.039$ Es primo 

$q = 2*2.039 + 1 = 4.079$ Es primo 

$p-1 = 2.038$; $q-1 = 4.078$

$p-1 = 2*1.019$; $q-1 = 2*2.039$ mcd (p, q) = 2

El módulo será $n = p*q = 8.317.081$

Ataque a la clave por factorización de n

- ¿Qué fortaleza tendrá este algoritmo ante ataques?
- El intruso que desee conocer la clave secreta d a partir de los valores n y e se enfrentará al Problema de la Factorización de Números Grandes (PFNG), puesto que la solución para conocer esa clave privada es conocer primero el valor del Indicador de Euler $\phi(n) = (p-1)(q-1)$ para así poder encontrar $d = inv [e, \phi(n)]$, pero para ello deberá saber los valores de los primos p y q.

Tiempo necesario para afrontar el PFNG

Para un procesador de $2x10^8$ instrucciones por segundo (años noventa).
Fuente: Criptografía Digital, José Pastor. Prensas Univ. de Zaragoza, 1998.

Nº de bits (n)	Nº de dígitos	Días	Años
60	18	$1,7 \times 10^{-8}$	-
120	36	$1,5 \times 10^{-5}$	-
256	77	1,0	-
363	109	$9,0 \times 10^{2}$	2,5
442	133	$9,4 \times 10^{4}$	$2,5 \times 10^{2}$
665	200	$3,8 \times 10^{8}$	$1,0 \times 10^{6}$

Desafío RSA640 (193 dígitos) roto en noviembre de 2005 en la Universidad de Bonn. Lo que en 1998 se valoraba en un millón de años, hoy se ha roto en un tiempo equivalente a 30 años con un PC a 2,2 GHz. Y se resolverán nuevos desafíos de números mayores. Por lo tanto, ... deberemos ser siempre muy cautos.

Claves privadas parejas en RSA

- Una clave privada pareja CPP d_P, permite descifrar el criptograma C resultado de una cifra con la clave pública e sin que d_P sea el inverso de la clave pública e. En el sistema RSA habrá como mínimo una clave d_P pareja de la clave privada d.
- Esto se debe a que las claves inversas e y d lo serán en $\phi(n)$ y en cambio la cifra se hace en el cuerpo n.
- Ejemplo:
- Si p = 13; q = 19; n = 247, $\phi(n)$ = 216 y elegimos e = 41, entonces
- d = inv (41, 216) = 137, que es único. Si ciframos con la clave pública el número N = 87 obtenemos $C = 87^{41} \bmod 247 = 159$.
- Luego sabemos que $N = C^d \bmod n = 159^{137} \bmod 247 = 87$
- Pero también lo desciframos con d_P = 29, 65, 101, 173, 209 y 245.

Número de claves privadas parejas

- Si $\gamma = mcm[(p-1),(q-1)]$ y sea $d\gamma = e^{-1} \bmod \gamma = inv(e, \gamma)$
- La clave pública e tendrá λ claves parejas di de la forma:
- $di = d\gamma + i\,\gamma \qquad 1 < di < n$
- $i = 0, 1, \ldots \lambda \qquad \lambda = \lfloor (n - d\gamma)/\gamma \rfloor$
- En el ejemplo anterior tenemos que:
- $\gamma = mcm[(p-1),(q-1)] = mcm(12, 18) = 36$
- Luego: $d\gamma = inv(41, 36) = 29$, así $di = d\gamma + i\,\gamma = 29 + i*36$
- Es decir di = 29, 65, 101, 137, 173, 209, 245. Observe que en aparece (137) la clave privada d y comprobamos que:
- $\lambda = \lfloor (n - d\gamma)/\gamma \rfloor = \lfloor (247 - 29)/36) \rfloor = 6{,}05 = 6$

Números no cifrables en RSA

- Si N^e mod n = N se dice que N es un número no cifrable, NNC. Aunque la clave e sea válida, el número N se enviará en claro ☹.
- En RSA habrá como mínimo 9 números no cifrables.
- En el caso más crítico, todos los números del cuerpo n pueden ser no cifrables.
- Para conocer estos valores no cifrables, habrá que hacer un ataque de cifrado por fuerza bruta en p y q, es decir deberemos comprobar que X^e mod p = X y X^e mod q = X con 1 < X < n-1 ☹.
- Ejemplo:
  - Sea el cuerpo n = 35 (p = 5, q = 7), con $\phi(n) = 24$ y e = 11.
  - Dentro de los números posibles {0, 34} serán no cifrables: {6, 14, 15, 20, 21, 29, 34} además de los obvios {0, 1}. El valor n-1 (en este caso 34) será también siempre no cifrable.

La paradoja del cumpleaños

- Podemos idear un ataque a la clave privada basado en este problema.
- Pregunta: ¿Cuántas personas tiene que haber en un aula para que la probabilidad de que al menos 2 de ellas cumplan años el mismo día sea superior a 0,5?
- Solución: Se escribe en la pizarra los 365 días del año y las personas entran al aula de uno en uno, borrando el día de su cumpleaños de la pizarra. Basta que entren sólo 23 personas al aula para tener una probabilidad superior a 0,5. Este es un valor muy bajo, en principio inimaginable y de allí el nombre de paradoja, aunque matemáticamente no lo sea.
- Explicación: El primero en entrar tendrá una probabilidad de que su número no esté borrado igual a n/n = 1, el segundo de (n-1)/n, etc. De esta manera, la probabilidad de no coincidencia será $p_{NC} = n!/(n-k)!n^k$. Para k = 23 se tiene $p_{NC} = 0,493$ y así la probabilidad de coincidencia será igual a $p_C = (1- pNC) = 0,507$, que es mayor que 0,5.

Ataque a la clave por paradoja cumpleaños

- **Algoritmo propuesto por Merkle y Hellman en 1981:**
  - El atacante elige dos números aleatorios distintos i, j dentro del cuerpo de cifra n. Lo interesante es que elige, además, un mensaje o número N cualquiera.
  - Para i = i+1 y para j = j+1 calcula N^i mod n y N^j mod n.
  - Cuando encuentra una coincidencia de igual resultado de cifra para una pareja (i, j), será capaz de encontrar d.
  - Un ejemplo para resolver en siguientes diapositivas:
    - sea p = 7; q = 13, n = 91, e = 11, d = 59. El atacante sólo conoce n = 91 y e = 11. Partirá con el número N = 20 y elegirá los valores i = 10 y j = 50.

Ejemplo de ataque paradoja cumpleaños

i	C_i	j	C_i
i = 10	$C_{10} = 20^{10} \bmod 91 = 43$	j = 50	$C_{50} = 20^{50} \bmod 91 = 36$
i = 11	$C_{11} = 20^{11} \bmod 91 = 41$	j = 51	$C_{51} = 20^{51} \bmod 91 = 83$
i = 12	$C_{12} = 20^{12} \bmod 91 = 1$	j = 52	$C_{52} = 20^{52} \bmod 91 = 22$
i = 13	$C_{13} = 20^{13} \bmod 91 = 20$	j = 53	$C_{53} = 20^{53} \bmod 91 = 76$
i = 14	$C_{14} = 20^{14} \bmod 91 = 36$	j = 54	$C_{54} = 20^{54} \bmod 91 = 64$
i = 15	$C_{15} = 20^{15} \bmod 91 = 83$	j = 55	$C_{55} = 20^{55} \bmod 91 = 6$
i = 16	$C_{16} = 20^{16} \bmod 91 = 22$	j = 56	$C_{56} = 20^{56} \bmod 91 = 29$
i = 17	$C_{17} = 20^{17} \bmod 91 = 76$	j = 57	$C_{57} = 20^{57} \bmod 91 = 34$

Hay una colisión en el paso quinto al coincidir el valor C = 36 en contador i que ya había aparecido en contador j. Observe los valores repetidos.

Con los valores de i, j y el desplazamiento observado en uno de ellos cuando se detecta la colisión (i = 14), se establece un conjunto de ecuaciones y, si el ataque prospera, obtenemos la clave privada, una clave privada pareja, o bien un valor de clave privada particular que sólo sirve para descifrar el número elegido (aquí el 20) y no un número genérico. En este caso se hablará de un falso positivo.

Resultado del ataque paradoja cumpleaños

- La primera coincidencia se encuentra para i = 14; j = 50. Así, el atacante conociendo la clave pública e = 11, calcula:
- w = (14-50) / mcd (11, |14-50|) = -36 / mcd (11, 36) = - 36.
- Entonces deberán existir valores s, t de forma que se cumpla lo siguiente:
- $w*s + e*t = 1 \Rightarrow -36*s + 11*t = 1$
- Las posibles soluciones a la ecuación son: $w*s \bmod e = 1$; $e*t \bmod w = 1$
- $-36*s = 1 \bmod 11 \Rightarrow s = inv(-36, 11) = inv(8, 11) = 7$
- $11*t = 1 \bmod 36 \Rightarrow t = inv(11, 36) = 23$
- El valor t = 23 será una clave privada pareja de d = 59. Compruebe que se verifica $w*s + e*t = 1$ y que las claves parejas son 11, 23, 35, 47, 71 y 83.

ElGamal

- Fue diseñado para producir firmas digitales pero se extendió también para codificar mensajes.
- Se basa en el problema de la complejidad de resolver logaritmos discretos.
- Elegimos un número primo *p* y dos números aleatorios *g* y *x* menores que *p*.
- Se calcula:

$$y = g^x \pmod{p}$$

- La clave pública es (*g*,*y*,*p*) y la privada es *x*.

ElGamal

- Veamos la codificación con este algoritmo:

$$\begin{aligned} a &= g^k \pmod{p} \\ b &= y^k m \pmod{p} \end{aligned}$$

- El texto codificado será pues a y b (doble longitud que el mensaje original).
- Para decodificar se hará:

$$m = b/a^x \pmod{p}$$

Ejemplo de cifrado con ElGamal

Adelaida (A) enviará a Benito (B) el mensaje M = 10 cifrado dentro del cuerpo p = 13 que usa Benito.

CIFRADO

Claves públicas de Benito: p = 13, g = 6, $(g^x) \bmod p = 2$

Adelaida A elige por ejemplo k = 4 y calcula:

$(g^k) \bmod p = 6^4 \bmod 13 = 9$

$(g^x)^k \bmod p = 2^4 \bmod 13 = 3$

$M*(g^x)^k \bmod p = 10*3 \bmod 13 = 4$

Envía a B $(g^k) \bmod p$, $M*(g^x)^k \bmod p = [9, 4]$

Ejemplo de descifrado con ElGamal

DESCIFRADO

La clave privada de Benito es x = 5

Benito recibe: $[(g^k) \bmod p, M*(g^x)^k \bmod p] = [9, 4]$

Benito calcula:

$(g^k)^x \bmod p = 9^5 \bmod 13 = 3$

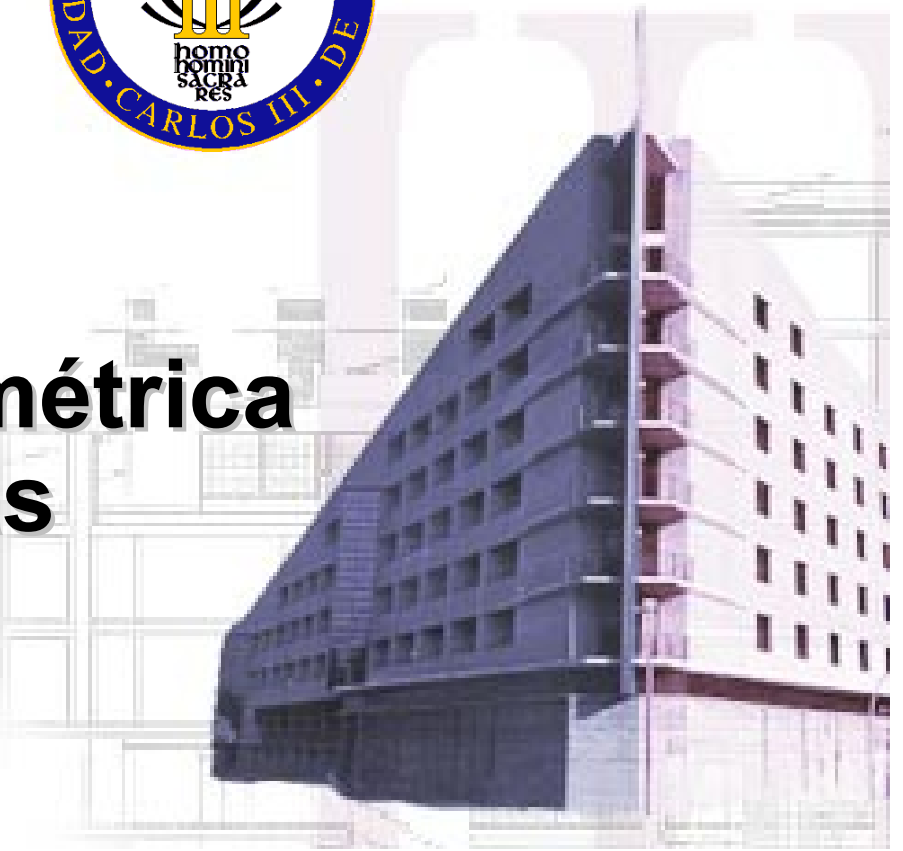
$[M*(g^x)^k] * inv[(g^k)^x, p] = 4 * inv\ (3, 13) = 4 * 9$

$M = 4 * 9 \bmod 13 = 10$ (se recupera el mensaje)

Resumen de los sistemas de clave pública basados en exponenciación

- Emisor y receptor generan un par de claves, pública y privada, relacionadas por una "función con trampa".
- Emisor y receptor de un mensaje usan claves diferentes para las operaciones de cifrado, descifrado.
- La seguridad del sistema va asociada a la resolución de un problema matemático de difícil solución en el tiempo.
- Nos permiten firmar digitalmente.
- Son sistemas de cifra muy lentos.
- Se usarán para cifrar datos cortos como claves de sesión o para firmar resúmenes de mensajes.

Criptografía asimétrica basada en curvas elípticas.



Departamento de INGENIERIA TELEMATICA
www.it.uc3m.es

ECC

- La criptografía basada en curvas elípticas (ECC) fue introducida por Victor Miller y Neal Koblitz en 1985.
- Su principal ventaja con la criptografía basada en exponenciaciones es que requiere claves de menor tamaño para obtener una seguridad similar (coste computacional para romper el sistema)
  - Claves menores → mayor velocidad de cálculo y menores requisitos de almacenamiento
  - Ideal para dispositivos limitados

Tamaños de claves

Equivalencias: tamaños claves para obtener misma seguridad

PLD y RSA (bits)	PLDE (bits)	Ratio tamaño claves	AES (bits)
1024	163	1:6	
3072	256	1:12	128
7680	384	1:20	192
15360	512	1:30	256

Cuadro 1: NIST guidelines for public key sizes for AES

Curvas elípticas

- La ecuación que nos define las curvas es:
  - y^2 [+ x·y] = x^3 + a·x + b
- Con a y b constantes.
- Para su aplicación a criptografía será necesario que:
  - x, y, a, b pertenezcan a un cuerpo finito.
- Se usan los cuerpos finitos o cuerpos de Galois (GF)
- No veremos los GF pero un caso particular de ellos son los generados por la operaciones modulo n si n es primo (GF(n))

Curvas elípticas

Si E es una curva elíptica E sobre $\mathbb{K}$, denotaremos por $E(\mathbb{K})$ el conjunto de puntos de $\mathbb{K}^2$ que satisfacen la ecuación de la curva junto con el punto del infinito $\mathcal{O}$, es decir,

$$E(\mathbb{K}) = \{(x, y) \in \mathbb{K}^2 \mid y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}$$

- Si $\mathbb{K} = \mathbb{R}$, la gráfica de una curva elíptica puede ser:

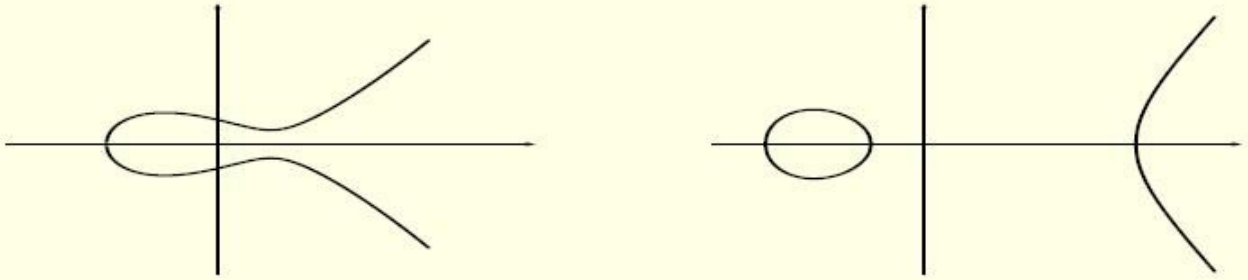


Figura 1: Curvas $y^2 = x^3 - 3x + 3$ e $y^2 = x^3 - 13x - 12$ sobre $\mathbb{R}$

- Si $\mathbb{K}$ es un cuerpo finito, obviamente $E(\mathbb{K})$ tiene un número finito de puntos. Por ejemplo, el conjunto de puntos de la curva $E: y^2 = x^3 + x + 1$ sobre $\mathbb{F}_7$ es

$$E(\mathbb{F}_7) = \{(0, 1), (0, 6), (2, 2), (2, 5), \mathcal{O}\}$$

Procedimiento para sumar usando curvas elípticas

En $E(\mathbb{K})$ se puede definir una operación $+$ mediante el método de la cuerda y la tangente:

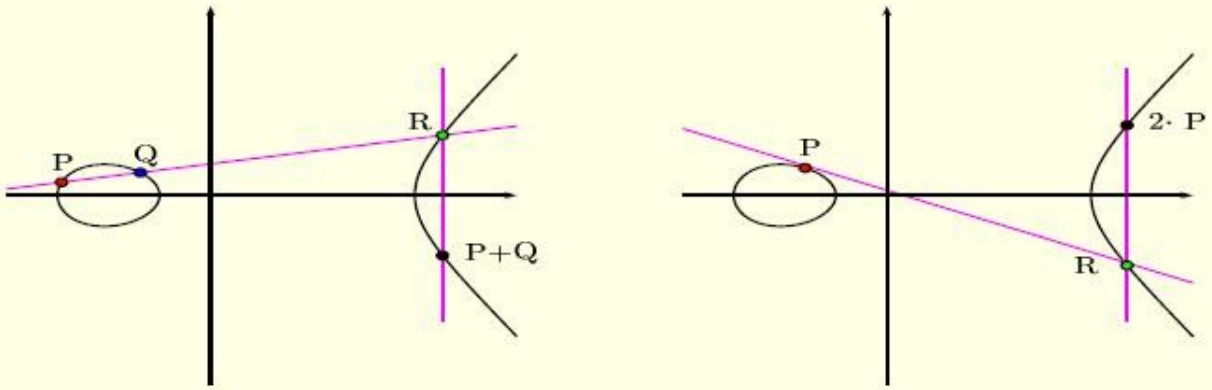


Figura 2: Suma y doblado de puntos en una curva elíptica

- $(E(\mathbb{K}), +)$ es un grupo abeliano con neutro el punto $\mathcal{O}$
  - El opuesto o simétrico de un punto $P = (x, y)$ de $E(\mathbb{K})$ es el punto $-P = (x, -y)$

Número de puntos en una CE

- **Teorema de Hasse:**
  - Para una curva elíptica definida sobre GF(q), el número de puntos sobre la curva será:

$$q + 1 - t$$

  - Donde $|t| \leq 2\sqrt{q}$
  - ***t*** **se llama traza de Frobenius**
  - **Se incluye el O**

Operaciones sobre curvas elípticas

- Se definen la operación de suma de puntos y producto de un punto por un entero.
- La suma de puntos se explica de forma gráfica en la siguiente transparencia (en criptografía usaremos las mismas expresiones que para números reales pero aplicadas a cuerpos finitos).
- Definimos el producto de un punto de la curva *P* por un entero *k* iterando como:
  - $2P = P+P$
  - $3P = P+P+P$ ...

Orden de un punto y orden de la curva

- El orden de un punto P de la curva es el entero n tal que:
  - nP=O
  - (n+1)P=P

Seguridad de las Curvas elípticas

- La seguridad vendrá dada porque existe una forma de calcular eficientemente:
  - Q=kP
- si conocemos k y conocemos P (técnica parecida al método de exponenciación rápida visto en clase) y sin embargo, si conocemos P y Q no hay un algoritmo eficiente de conocer k. De esta manera estamos protegiendo k al enviarla al receptor.

Complejidad implícita en ECC

- Por lo conocido hasta la fecha, para calcular *k* a partir de *k*F* si el orden de *F* tiene n bits es aproximadamente $2^{n/2}$.
- Por ejemplo, si el orden de es un primo de 240 bits se puede esperar que se necesiten del orden de 2^{120} operaciones. Si se tiene una máquina que es capaz de hacer un millón de operaciones por segundo se tardaría 2^{100} segundos, o bien 2^{75} años, o aproximadamente 10^{23} años ...

ElGamal sobre ECC

- Alicia y Bartolo eligen un cuerpo finito GF, una curva elíptica E y un punto $P \in E$
- Alicia elige un valor *a* aleatorio y hace público el valor del punto $aP \in E$
- Para enviar un mensaje m a Alicia, Bartolo calcula M (punto de la curva E cuya componente x es m), elige un *k* aleatorio y envía la pareja de puntos *k*P y M+*k*(*a*P) a Alicia.
- Para leer el mensaje, Alicia multiplica el primer punto por *a* y se lo resta al segundo punto:
  - $M+k(aP) - a(kP) = M$

Comparación ElGamal sobre ECC y basado en exponenciación

- **ECC:**
  - $M+k(aP) - a(kP)$
- **Exponenciación:**
  - $M(g^a)^k / (g^k)^a$
- **Generalización para convertir un método de exponenciación a ECC:**
  - Las exponenciaciones → productos
    - La base se convierte en el punto de la curva
    - El exponente se convierte en el escalar
  - Los productos → sumas
  - Divisiones → restas

Criptografía EC : Ejemplo D-H

- Se publica una determinada curva elíptica y de un punto de esa curva *P*.
- Cada usuario creará la clave privada eligiendo u entero aleatorio *k*.
- La clave pública consiste en multiplicar *P* por *k*.
- Por ejemplo: Alicia tendría:
  - $A_p = k_1$.
  - $A_P = k_1 * P$
- Y Carlos:
  - $C_p = k_2$.
  - $C_P = k_2 * P$

Criptografía EC : Ejemplo D-H

- Se puede obtener un secreto común usando:
  - $A_p * C_P = C_p * A_P = k_1 * k_2 * P$.
- La seguridad se basa en que resulta muy complejo calcular k_1 y k_2 a partir de las claves públicas.