## 5.4 Algoritmo RSA

- El primer criptosistema de clave pública propuesto en la literatura científica fue el diseñado, de forma muy elegante, por los investigadores estadounidenses Ronald **R**ivest, Adi **S**hamir y Leonard **A**dleman en 1975.

- Este criptosistema, conocido como RSA, se apoya en el hecho de que la **exponenciación modular** es una **función unidireccional** bajo ciertas condiciones.

- Es un algoritmo aceptado mundialmente como cifrador de clave pública y está **basado en el problema de la factorización de un número con un gran número de cifras en sus factores primos**.

## 5.4 Algoritmo RSA

- La seguridad de RSA radica precisamente en la **dificultad de la factorización de números grandes**:
  - es fácil saber si un número es primo (o probablemente primo), pero es extremadamente difícil obtener la factorización en números primos de un entero elevado, debido no a la dificultad de los algoritmos existentes, sino al consumo de recursos físicos (memoria, necesidades hardware...incluso tiempo de ejecución) de tales algoritmos.
- De entre todos los algoritmos asimétricos, quizá sea el más sencillo de comprender e implementar.
- Sus **claves sirven** indistintamente tanto para **cifrar** como para **autentificar**.

# 5.4 Algoritmo RSA

- Ha estado **bajo patente** de los Laboratorios RSA hasta el 20 de septiembre de 2000, por lo que su uso comercial estuvo restringido hasta esa fecha.
- Sujeto a múltiples controversias, desde su nacimiento **nadie ha conseguido probar o rebatir su seguridad**, pero se le tiene como **uno de los algoritmos asimétricos más seguros**.
- Las claves pública y privada se calculan a partir de un número que se obtiene como producto de dos primos grandes.
- Un atacante se enfrentará, si quiere recuperar un texto claro a partir del criptograma y la clave pública, a **un problema de factorización**.

## 5.4 Algoritmo RSA

**Recordemos que**

Se define la **función de Euler**, $\Phi$, como la función natural de variable natural tal que para un número natural n

$$\Phi(n)=card\{ i \in N \; / \; 1\leq i<n \text{ y } mcd(i,n)=1 \}$$

es decir, $\Phi(n)$ es igual al número de números naturales menores que n, primos con n.

**Ejemplo**

- $\Phi(8) = 4$ (ya que son primos con 8 los números 1, 3, 5 y 7)
- $\Phi(11) = 10$ (ya que son primos con 11 los números 1, 2, 3, 4, 5, 6, 7, 8, 9 y 10)
- Si p es primo entonces $\Phi(p) = p-1$.

## 5.4 Algoritmo RSA

**Proposición**

Si p y q son dos números primos entre sí

$$\Phi(p\,q) = \Phi(p)\,\Phi(q)$$

**Corolario**

Si p y q son dos números primos

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

**Ejemplo**

- $\Phi(55) = \Phi(5 \cdot 11) = 4 \cdot 10 = 40$

## 5.4 Algoritmo RSA

- Para generar un par de claves, en primer lugar, se eligen aleatoriamente dos números primos grandes, p y q (actualmente se recomienda que tengan más de doscientos dígitos) y se calcula el producto **n = pq.**
- A continuación, se escoge un número natural **e**, $0<e<\Phi(n)$, primo con $\Phi(n)$, o sea: **mcd(e, Φ(n))=1**
- Por la elección de e, sabemos que existe un número natural **d** que es el inverso de e mod $\Phi(n)$, esto es: **d e mod Φ(n) =1**
- La **clave pública** del usuario A es el par (n,e) y la **función de cifrado** es

$$c=E_k(m)=m^e \bmod n$$

- La **clave privada** del usuario A es el par (n,d) y la **función de descifrado** es

$$m=D_k(c)=c^d \bmod n$$

- También deben permanecer en secreto los valores de p, q y $\Phi(n)$.

## 5.4 Algoritmo RSA

- **Cuando p y q son muy grandes** (los creadores del sistema sugieren del orden de cien cifras) la función $E_k(x)$ es **unidireccional** ya que para la obtención del algoritmo de descifrado se necesita conocer d, que en definitiva supone el conocimiento de p y q (números primos) dado n=pq.

- La clave privada viene dada por tanto por el par (p,q) ya que d se obtiene como solución de la ecuación

$$\mathbf{d\ e \bmod (p-1)(q-1) = 1}$$

# 5.4 Algoritmo RSA

Consideremos un alfabeto con 28 símbolos a los que asignamos los siguientes números

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | Ñ | O | P | Q | R | S | T | U | V | W | X | Y | Z | _ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |

Supongamos que p=3 y q=11, entonces

$n=33$ y $\Phi(n)=20$.

Si se elige como clave (pública) de cifrado

(n=33,e=3)

Para cifrar el texto en claro

m=FIRMA_DIGITAL

en primer lugar asociamos a cada letra su número

m=07 10 20 14 02 29 05 10 08 10 22 02 13

## 5.4 Algoritmo RSA

Se obtiene entonces

$E_k(07)= 07^3 \bmod 33 = 13$

$E_k(10)= 10^3 \bmod 33 = 10$

$E_k(20)= 20^3 \bmod 33 = 14$

$E_k(14)= 14^3 \bmod 33 = 05$

$E_k(02)= 02^3 \bmod 33 = 08$

$E_k(29)= 29^3 \bmod 33 = 02$

$E_k(05)= 05^3 \bmod 33 = 26$

$E_k(10)= 10^3 \bmod 33 = 10$

$E_k(08)= 08^3 \bmod 33 = 17$

$E_k(10)= 10^3 \bmod 33 = 10$

$E_k(22)= 22^3 \bmod 33 = 22$

$E_k(02)= 02^3 \bmod 33 = 08$

$E_k(13)= 13^3 \bmod 33 = 19$

por lo que el criptograma asociado es

c= 13 10 14 05 08 02 26 10 17 10 22 08 19 = LIMDGAXIOITGQ

# 5.4 Algoritmo RSA

La clave privada de descifrado es

(n=33, d=7)

$d = e^{-1} \bmod \Phi(n)$
$= 3^{-1} \bmod 20$
$= 7$

que se usa para descifrar c como sigue

$D_k(13) = 13^7 \bmod 33 = 07$

$D_k(10) = 10^7 \bmod 33 = 10$

$D_k(14) = 14^7 \bmod 33 = 20$

$D_k(05) = 05^7 \bmod 33 = 14$

$D_k(08) = 08^7 \bmod 33 = 02$

$D_k(02) = 02^7 \bmod 33 = 29$

$D_k(26) = 26^7 \bmod 33 = 05$

$D_k(10) = 10^7 \bmod 33 = 10$

$D_k(17) = 17^7 \bmod 33 = 08$

$D_k(10) = 10^7 \bmod 33 = 10$

$D_k(22) = 22^7 \bmod 33 = 22$

$D_k(08) = 08^7 \bmod 33 = 02$

$D_k(19) = 19^7 \bmod 33 = 13$

## 5.4 Algoritmo RSA

- En la práctica, **el cálculo de las claves se realiza en secreto** en la máquina en la que se va a guardar la clave privada y, una vez generada ésta, **conviene protegerla mediante un algoritmo criptográfico simétrico**.

- En cuanto a las longitudes de claves, el sistema RSA permite longitudes variables, siendo aconsejable actualmente el uso de **claves de no menos de 1024 bits**.

- En 1991 los laboratorios RSA lanzaron varios desafíos de factorización con distintos valores de *n* y, aunque la compañía cerró esta competición en el año 2007, el mayor desafío resuelto hasta hoy ha sido un valor ***n* de 768 bits en diciembre de 2009**.

# 5.4 Algoritmo RSA

- Existen dos posibles técnicas para inutilizar el algoritmo RSA:
  - **Fuerza bruta:** probar todas las claves privadas posibles, actualmente **imposible para el tamaño de claves** que se utilizan.
  - **Factorizar n** como producto de dos números primos ya que así se puede obtener fácilmente $\Phi(n)$ y d. Esta tarea es hoy **computacionalmente imposible en un tiempo razonable** para claves iguales o mayores a 1024 bits.

- RSA **presenta todas las ventajas de los sistemas asimétricos, incluyendo la firma digital**, aunque resulta más útil a la hora de implementar la confidencialidad el uso de sistemas simétricos, por ser más rápidos.
  - Se suele usar también en los sistemas mixtos para cifrar y enviar la clave simétrica que se usará posteriormente en la comunicación cifrada.

## 5.4 Algoritmo RSA

- En el ejemplo anterior se puede observar que el sistema utilizado es en definitiva **un criptosistema de sustitución simple**,
  - susceptible, por tanto, de a**taque mediante técnicas de análisis de frecuencias**;
  - es por ello que se suele utilizar combinado con otro sistema de cifrado que disperse las frecuencias de aparición de los diferentes símbolos del alfabeto.
- Un método que permite enmascarar estas frecuencias consiste en **tomar bloques de varios caracteres** y cifrarlos de una sola vez.

# 5.4 Algoritmo RSA

**Ejemplo**

Consideremos el alfabeto inglés con la siguiente asignación numérica

| | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | _ |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |

Sea p=53 y q=61, entonces n=53·61=3233 y Φ(n)=52·60=3120

Eligiendo e=71 se obtiene d=**791**.

¿ **x** = $71^{-1}$ mod 3120 ? → 1 = **x**·71 mod 3120

3120=43·71 + **67** ⇒ **67** = 3120 - 43·71 mod 3120 = **(-43)·71** mod 3120

71 = 1·**67** + **4** ⇒ **4** = 71 - 1·**67** mod 3120 = 71 – 1·**(-43) ·71** mod 3120 = **44·71** mod 3120

**67** = 16·**4** + 3 ⇒ **3** = **67** - 16·**4** mod 3120 = **(-43)·71** – 16·**44·71** mod 3120 = **(-747)·71** mod 3120

**4** = 1·**3** + 1 ⇒ **1** = **4** - 1·**3** mod 3120 = **44·71**– 1·**(-747)·71** mod 3120 = **791**·71 mod 3120

- Luego **x** = $71^{-1}$ mod 3120 = **791**

# 5.4 Algoritmo RSA

**Ejemplo**

Consideremos el alfabeto inglés con la siguiente asignación numérica

| | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | _ |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |

Sea p=53 y q=61, entonces n=53·61=3233 y $\Phi$(n)=52·60=3120

Eligiendo e=71 se obtiene d=791.

Para cifrar el mensaje

m=RENAISSANCE

haremos en primer lugar la asignación numérica

m=17 04 13 00 08 18 18 00 13 02 04

# 5.4 Algoritmo RSA

Si cifráramos directamente, tendríamos

| m = | RE | NA | IS | SA | NC | E_ |
|---|---|---|---|---|---|---|
| | 1704 | 1300 | 0818 | 1800 | 1302 | 0426 |

$E_k(1704) = 1704^{71} \bmod 3233 =$ **3106**

$E_k(1300) = 1300^{71} \bmod 3233 =$ **0100**

$E_k(0818) = 0818^{71} \bmod 3233 =$ **0931**

$E_k(1800) = 1800^{71} \bmod 3233 =$ **2691**

$E_k(1302) = 1302^{71} \bmod 3233 =$ **1984**

$E_k(0426) = 0426^{71} \bmod 3233 =$ **2927**

**c = 3106 0100 0931 2691 1984 2927**

Que no se puede expresar con el alfabeto.

## 5.4 Algoritmo RSA

- El criptograma c se puede expresar en términos del alfabeto actuando del siguiente modo:
  - El mayor número que podemos obtener al aplicar $E_k(m)$ es 3232, que expresado en base 27 (número de elementos del alfabeto) es

$$3232 = \mathbf{4} \cdot 27^2 + \mathbf{11} \cdot 27 + \mathbf{19} = (\mathbf{4})(\mathbf{11})(\mathbf{19})_{(27)} = \mathbf{ELT}$$

  - Utilizaremos, por tanto tres letras para codificar cada uno de los valores obtenidos para $E_k(m)$ con los elementos del alfabeto.

$$3106 = \mathbf{4} \cdot 27^2 + \mathbf{7} \cdot 27 + \mathbf{1} = (\mathbf{4})(\mathbf{7})(\mathbf{1})_{(27)} = \mathbf{EHB}$$

$$0100 = \mathbf{0} \cdot 27^2 + \mathbf{3} \cdot 27 + \mathbf{19} = (\mathbf{0})(\mathbf{3})(\mathbf{19})_{(27)} = \mathbf{ADT}$$

$$0931 = \mathbf{1} \cdot 27^2 + \mathbf{7} \cdot 27 + \mathbf{13} = (\mathbf{1})(\mathbf{7})(\mathbf{13})_{(27)} = \mathbf{BHN}$$

$$2691 = \mathbf{3} \cdot 27^2 + \mathbf{18} \cdot 27 + \mathbf{18} = (\mathbf{3})(\mathbf{18})(\mathbf{18})_{(27)} = \mathbf{DSS}$$

$$1984 = \mathbf{2} \cdot 27^2 + \mathbf{19} \cdot 27 + \mathbf{13} = (\mathbf{2})(\mathbf{19})(\mathbf{13})_{(27)} = \mathbf{CTN}$$

$$2927 = \mathbf{4} \cdot 27^2 + \mathbf{0} \cdot 27 + \mathbf{11} = (\mathbf{4})(\mathbf{0})(\mathbf{11})_{(27)} = \mathbf{EAL}$$

Así pues:

**c = 3106 0100 0931 2691 1984 2927 = EHBADTBHNDSSCTNEAL**

Tiene 6 caracteres más que m

# 5.4 Algoritmo RSA

- En el ejemplo anterior los números cifrados son, casualmente, todos elementos de $\mathbb{Z}_{3233}$.

| | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | _ |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |

  - Si cifráramos de 2 en 2, el mensaje
    $m = \_\,\_ = (26)\ (26)_{(27)} = 26{\cdot}27 + 26 = 27^2 = 729 < 3233$.
    Al descifrar obtendríamos
    $m = D_k(c) = c^d \bmod 3233 = 729 \bmod 3233 = 729 = 26{\cdot}27 + 26 = (26)\ (26)_{(27)} = \_\,\_$.
    **No hay ambigüedad.**

  - En cambio si agrupamos de 3 en 3, el mensaje
    $m = \_\,\_\,\_ = (26)\ (26)\ (26)_{(27)} = 26{\cdot}27^2 + 26{\cdot}27 + 26 = 27^3 = 19683 > 3233$.
    Al descifrar obtendremos
    $m = D_k(c) = c^d \bmod 3233 = 19683 \bmod 3233 = 285 = 0{\cdot}27^2 + 10{\cdot}27 + 15 = (0)\ (10)\ (15)_{(27)} = \text{AKP}$.
    **Hay ambigüedad.**

- Si se quiere garantizar siempre que los elementos que cifremos sean elementos de $\mathbb{Z}_n$ (y por tanto garantizar que se puede descifrar) deberemos agrupar los caracteres del texto en claro en bloques de tamaño k, siendo

$$a^k \le n < a^{k+1}$$

donde <u>**a** *es el número de elementos del alfabeto utilizado*</u>; y posteriormente codificarlos en base a.

## EJEMPLO DE UTILIZACIÓN DE RSA EN LA PRÁCTICA

- En el ejemplo anterior, n = 3233 y a = 27

  Debemos agrupar m en bloques de tamaño k=2, ya que

$$27^2 \leq 3233 < 27^{(2+1)}$$

| | | | | | | |
|---|---|---|---|---|---|---|
| m = | RE | NA | IS | SA | NC | E_ |
| | 1704 | 1300 | 0818 | 1800 | 1302 | 0426 |

$$\mathbf{RE} = (17)(04)_{(27)} = 17 \cdot 27 + 04 = 463$$
$$\mathbf{NA} = (13)(00)_{(27)} = 13 \cdot 27 + 00 = 351$$
$$\mathbf{IS} = (08)(18)_{(27)} = 08 \cdot 27 + 18 = 234$$
$$\mathbf{SA} = (18)(00)_{(27)} = 18 \cdot 27 + 00 = 486$$
$$\mathbf{NC} = (13)(02)_{(27)} = 13 \cdot 27 + 02 = 353$$
$$\mathbf{E\_} = (04)(26)_{(27)} = 04 \cdot 27 + 26 = 134$$

$$m = 463\ 351\ 234\ 486\ 353\ 134$$

## EJEMPLO DE UTILIZACIÓN DE RSA EN LA PRÁCTICA

m = RENAISSANCE_ = 463 351 234 486 353 134

que cifraremos

$E_k$(**RE**) = $E_k$(463) = $463^{71}$ mod 3233 = **716 = (00)(26)(14)$_{(27)}$= A_O**

$E_k$(**NA**) = $E_k$(351) = $351^{71}$ mod 3233 = **2062 = (02)(22)(10)$_{(27)}$= CWK**

$E_k$(**IS**) = $E_k$(234) = $234^{71}$ mod 3233 = **2483 = (03)(10)(26)$_{(27)}$= DK_**

$E_k$(**SA**) = $E_k$(486) = $486^{71}$ mod 3233 = **1368 = (01)(23)(18)$_{(27)}$= BXS**

$E_k$(**NC**) = $E_k$(353) = $353^{71}$ mod 3233 = **14 = (00)(00)(14)$_{(27)}$= AAO**

$E_k$(**E_**) = $E_k$(134) = $134^{71}$ mod 3233 = **259 = (00)(09)(16)$_{(27)}$= AJQ**

Para obtener el criptogtrama

**c = A_OCWKDK_BXSAAOAJQ**

## EJEMPLO DE UTILIZACIÓN DE RSA EN LA PRÁCTICA

Para descifrar **c = A_OCWKDK_BXSAAOAJQ** debemos agrupar en bloques de 3 caracteres y obtener la expresión en base 10 del trigrama.

$D_k$(**A_O**) = $D_k$(716) = $716^{791}$ mod 3233 = **463 =** (**17**)(**04**)$_{(27)}$= **RE**

$D_k$(**CWK**) = $D_k$(2062) = $2062^{791}$ mod 3233 = **351 =** (**13**)(**00**)$_{(27)}$= **NA**

$D_k$(**DK_**) = $D_k$(2483) = $2483^{791}$ mod 3233 = **234 =** (**08**)(**18**)$_{(27)}$ = **IS**

$D_k$(**BXS**) = $D_k$(1368) = $1368^{791}$ mod 3233 = **486 =** (**18**)(**00**)$_{(27)}$ = **SA**

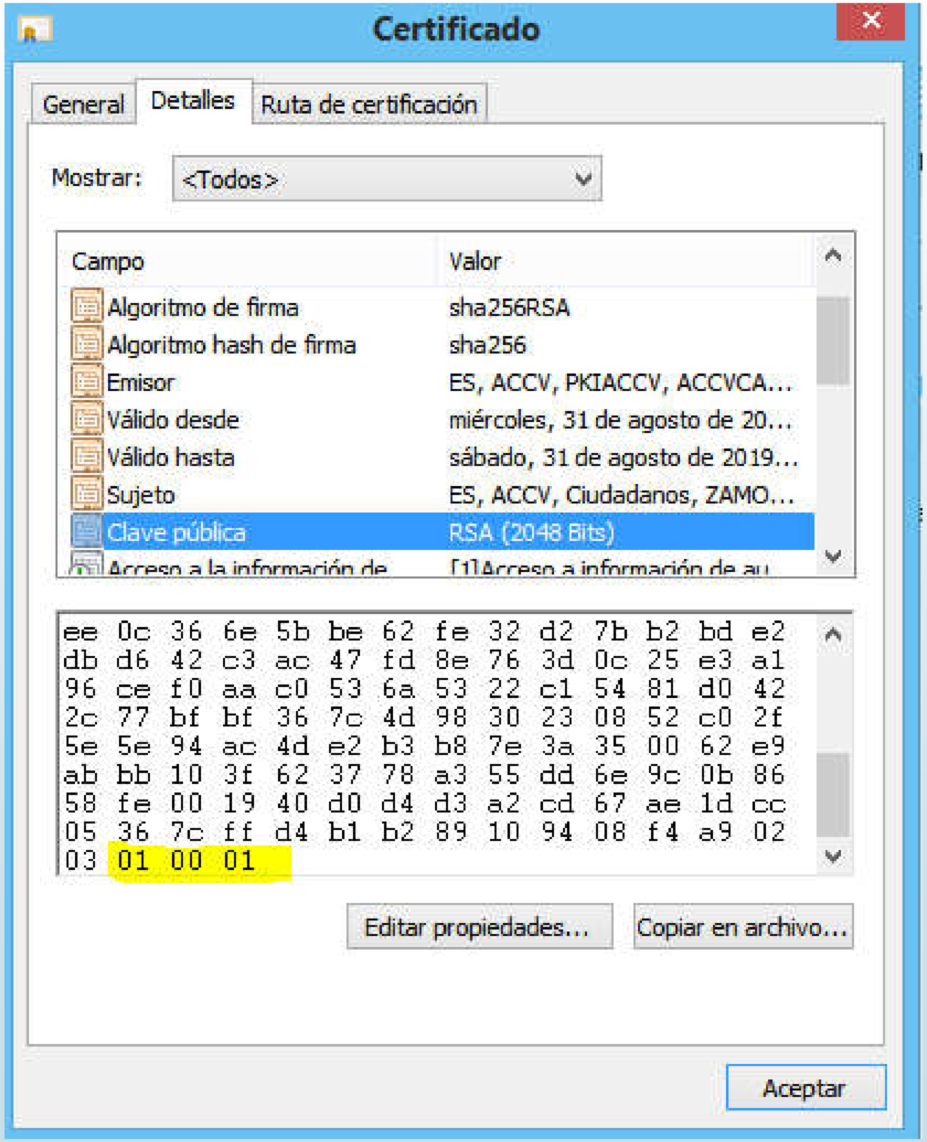
$D_k$(**AAO**) = $D_k$(14) = $14^{791}$ mod 3233 = **353 =** (**13**)(**02**)$_{(27)}$ = **NC**

$D_k$(**AJQ**) = $D_k$(259) = $259^{791}$ mod 3233 = **134 =** (**04**)(**26**)$_{(27)}$ = **E_**

Luego m = **RENAISSANCE_**

# 5.4 Algoritmo RSA

- Certificados



## 5.4 Algoritmo RSA

### EJEMPLO DE RSA REAL

Elegimos dos números primos de 512 bits cada uno:

*p = 99139 39965 46089 30824 88909 93861 03286 96951 12542 27785*
*63290 53802 69243 59622 97966 62570 51166 31784 15643 33030*
*16752 83735 31885 76891 66571 64285 73232 22921 38706 46645*
*4667*

*q = 13136 91819 25708 96843 02581 72409 47022 42864 58333 11752*
*43481 96993 06139 88470 36911 06258 28665 95507 45575 89427*
*96421 73663 31154 90105 78349 59036 89416 42907 63853 18510*
*41021*

Calculamos ahora *n*, que tiene 1024 bits:

*n = 13023 86182 92318 89502 81446 21088 35570 66154 05574 73331*
*40682 54739 98959 56538 57163 11615 41258 54863 46136 57472*
*70014 83693 94664 97674 40264 93450 88904 88932 37197 32536*
*99623 85944 66298 94133 32414 06479 20780 49438 83915 47728*
*71066 51988 01638 29581 61596 27668 24197 08727 22204 06157*
*69033 71955 35028 37798 65973 36760 32443 19911 37588 05059*
*05389 5007*

# 5.4 Algoritmo RSA

## EJEMPLO DE RSA REAL

Calculamos ahora *ϕ (n)*

*ϕ (n) = 13023 86182 92318 89502 81446 21088 35570 66154 05574 73331*
*40682 54739 98959 56538 57163 11615 41258 54863 46136 57472*
*70014 83693 94664 97674 40264 93450 88904 88932 37197 32536*
*99621 55436 08140 90954 33158 91752 02824 75927 58318 51855*
*25756 53877 77904 98939 17269 60590 99043 70901 35345 34755*
*41723 90985 14659 94363 88023 86692 77788 52514 85590 27820*
*73639 9320*

Como exponente de cifrado elegimos $e = 65537 = 2^{16}+1$, mientras que el exponente de descifrado es:

*d = 16516 06202 03467 10050 48918 84868 90218 92489 18279 99581*
*50695 43180 80680 06590 48611 11408 16546 34751 39652 57374*
*55344 81434 97422 92471 72748 50400 58881 01914 48242 51509*
*06748 05656 23580 43535 93387 51598 50264 21324 46463 09835*
*51972 56416 71447 94037 48482 18482 95184 74535 17075 11535*
*81529 12320 91084 24120 45236 48596 01095 63033 24342 09716*
*36483 433*

# 5.4 Algoritmo RSA

## EJEMPLO DE RSA REAL

- Supongamos que el mensaje a transmitir utilizando la clave pública *(n,e)* es:

"Hasta la fecha no se ha demostrado de forma rigurosa la equivalencia entre resolver el problema de la factorización y romper el criptosistema RSA."

- Para transformar el mensaje en un número menor que el módulo RSA y primo con él, se utiliza la base 256 dada por los valores ASCII de los caracteres que componen el mensaje.

- Como la longitud del mensaje no puede ser mayor que el módulo RSA, se analiza si es preciso trocear el mensaje.

## EJEMPLO DE RSA REAL

$m_1$ = 34591 23054 20684 92221 54004 31463 55718 40131 37946 50256
99770 12379 52245 48266 91597 68390 89367 27153 70468 41774
43004 17303 60120 13102 23597 42585 57180 20667 78546 06812
75424 51035 61893 92809 52751 30985 62992 14551 08844 27863
83635 95214 39334 39345 58318 94335 36299 26978 14144 61358
60603 23404 84057 33961 81735 90951 71716 01412 29657 69676 146

$m_2$ = 31029 37695 14888 24008 25180 81526 70973 28927 96416 57111
46496

# 5.4 Algoritmo RSA

## EJEMPLO DE RSA REAL

$c_1$ = 24283 83009 28360 92697 52894 91397 11182 33327 01972 51994
66194 67116 15452 51338 36137 91948 61510 99909 69538 57591
62731 96550 16598 26516 28223 74514 60203 01145 55449 76420
73563 67035 62024 22363 16254 50805 03386 81854 29313 76893
22373 92781 30286 52114 52126 18961 46028 71599 71878 80429
73749 97653 74787 98609 84624 04766 58549 16062 48613 00369
22093 872

$c_2$ = 38682 54957 70121 92903 21010 10135 17239 71957 23710 93292
77161 21301 87357 13461 04331 73889 57425 36031 34884 61661
51664 31071 46625 14562 56910 77963 89701 42435 54123 55176
62692 53475 35346 82015 24635 33695 20098 88439 34168 44397
93387 91590 76644 51408 56154 28962 22028 65447 75223 55380
20368 32538 03134 56525 82206 82408 11962 36481 98184 80787
46237 430

## 5.5 Fundamentos criptográficos de la firma digital

- Una de las aplicaciones inmediatas de los algoritmos asimétricos es el cifrado de la información sin tener que transmitir la clave de descifrado, lo cual permite su uso en canales inseguros.

- Estamos pasando, gracias a Internet y a la aparición de las nuevas tecnologías, de un sistema tradicional de realizar operaciones comerciales a uno nuevo que las efectúa mediante métodos electrónicos.

- Resulta necesario contar con técnicas electrónicas que suplanten la tradicional firma autógrafa y dar así validez a documentos electrónicos.

## 5.5 Fundamentos criptográficos de la firma digital

- La segunda aplicación de los algoritmos asimétricos es la autentificación de mensajes, con ayuda de funciones resumen (Hash), que nos permiten obtener una firma digital a partir de un mensaje.

- La firma digital se justifica desde el momento en que los contratos, las transacciones económicas, las compras, etc. se realizan on-line.

## 5.5 Fundamentos criptográficos de la firma digital

- Dos problemas aquejan a estos documentos electrónicos:

- ***Confidencialidad***: Capacidad de mantener un documento electrónico inaccesible a todos, excepto a una lista determinada de personas.
  - Se resuelve con métodos de cifrado.

- ***Autenticidad***: Capacidad de averiguar de forma segura e irrevocable la procedencia del mensaje.
  - Se resuelve con técnicas como la firma digital.

## 5.5 Fundamentos criptográficos de la firma digital

- Si A desea firmar digitalmente el mensaje m, envía el mensaje cifrado $c=E_{k_B}(m)$ al usuario B y para firmarlo,
  - en primer lugar calcula su rúbrica cifrando el mensaje a enviar con su clave privada, $r= D_{k_A}(m)$, y
  - a continuación determina su firma para el mensaje m, cifrando con la clave pública de B esa rúbrica, $s= E_{k_B}(r)= E_{k_B}[D_{k_A}(m)]$.

- B verifica la firma digital de A determinando,
  - en primer lugar, la rúbrica de A, $D_{k_B}(s) = D_{k_B}[E_{k_B}(r)]$, y
  - a continuación comprobando que $E_{k_A}(r) = E_{k_A}[D_{k_A}(m)]$ coincide con el mensaje m.

## ESQUEMA DE FIRMA DIGITAL CON CIFRADO

**A** $E_{K_A}$ $D_{K_A}$

$m$

$m$; $r = D_{K_A}(m)$; $s = E_{K_B}(r)$

$c=E_{K_B}(m)$ →

$s=E_{K_B}(r)$ →

**B** $E_{K_B}$ $D_{K_B}$

$c$; $m = D_{K_B}(c)$

$c$; $m$; $s$

$c$; $m$; $s$; $D_{K_B}(s)= r$

$c$; $m$; $s$; $r$;

$E_{K_A}(r)= E_{K_A}[D_{K_A}(m)] \overset{?}{=} m$

## FIRMA DIGITAL CON CIFRADO EN RSA

- Consideremos dos usuarios A y B, con claves ($n_A$, $e_A$, $d_A$) y ($n_B$, $e_B$, $d_B$) respectivamente.

- Si A desea cifrar y firmar digitalmente el mensaje m, envía el mensaje cifrado
$c = E_{k_B}(m) = m^{e_B} \bmod n_B$ al usuario B y para firmarlo,
  - en primer lugar calcula su rúbrica cifrando el mensaje a enviar con su clave privada, $r = D_{k_A}(m) = m^{d_A} \bmod n_A$, y
  - a continuación determina su firma para el mensaje m cifrando con la clave pública de B esa rúbrica, $s = E_{k_B}(r) = r^{e_B} \bmod n_B$.

- B verifica la firma digital de A determinando,
  - en primer lugar, la rúbrica de A, $D_{k_B}(s) = s^{d_B} \bmod n_B$, y
  - a continuación comprobando que $E_{k_A}(r) = r^{e_A} \bmod n_A = m^{d_A e_A} \bmod n_A$ coincide con el mensaje m.

# ESQUEMA DE FIRMA DIGITAL CON CIFRADO EN RSA

**A** $n_A, e_A, d_A$

**B** $n_B, e_B, d_B$

$m$

$c = m^{e_B} \bmod n_B$ →

$c$

$m = c^{d_B} \bmod n_B$

$m$

$r = m^{d_A} \bmod n_A$

$s = r^{e_B} \bmod n_B$

$s = r^{e_B} \bmod n_B$ →

$c$

$m$

$s$

$c; m; s;$

$s^{d_B} \bmod n_B = r^{d_B e_B} \bmod n_B = r$

$c; m; s; r;$

$r^{e_A} \bmod n_A = m$ ?

## EJEMPLO DE FIRMA DIGITAL CON CIFRADO EN RSA

**A** $n_A = 5 \cdot 11 = 55$, $e_A = 33$, $d_A = 17$

**B** $n_B = 3 \cdot 17 = 51$, $e_B = 21$, $d_B = 29$

$m = 12$;
$c = m^{e_B} \bmod n_B = 12^{21} \bmod 51 = 3$

$\xrightarrow{c\,=\,3}$

$c = 3$
$m = c^{d_B} \bmod n_B = 3^{29} \bmod 51 = 12$

$m$
$r = m^{d_A} \bmod n_A = 12^{17} \bmod 55 = 12$
$s = r^{e_B} \bmod n_B = 12^{21} \bmod 51 = 3$

$\xrightarrow{s\,=\,3}$

$c = 3$
$m = 12$
$s = 3$

$c = 3$; $m = 12$; $s = 3$;
$r = s^{d_B} \bmod n_B = 3^{29} \bmod 51 = 12$

$c = 3$; $m = 12$; $s = 3$; $r = 12$
$r^{e_A} \bmod n_A = 12^{33} \bmod 55 = 12 = m$

# Otros Algoritmos Asimétricos

- Consultar pág 218  Lucena cuarta edición v 4-0.11.0

- **Algoritmo de ElGamal**
  - **Firmas Digitales de ElGamal**
  - **Cifrado de ElGamal**

- **Algoritmo de Rabin**

- **Criptografía de Curva Elíptica**

## 5.6 Funciones hash

- Los criptosistemas de clave pública, por lo general, son mucho más lentos que los de clave secreta.

- Los esquemas de firma digital, también suelen ser muy lentos y, en ocasiones, la longitud de la firma suele ser similar o mayor que el propio mensaje que se firma.

- La necesidad de firmar los mensajes y el hecho no deseable de que la longitud de la firma sea extensa, hace pensar en la conveniencia de buscar una solución a ese problema.
  - La solución consiste en utilizar unas funciones llamadas hash (picadillo, resumen).

## 5.6 Funciones hash

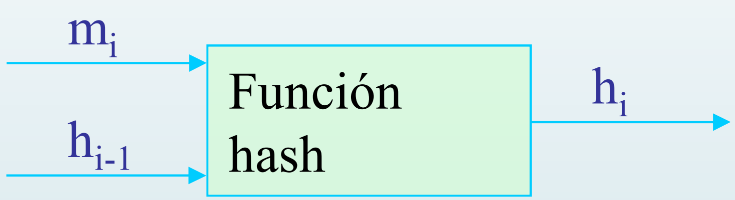
- En lugar de firmar digitalmente el mensaje completo, se firma digitalmente un resumen o hash de dicho mensaje, representado por sólo una centena de bits.

- Las funciones hash también se utilizan para verificar la **integridad** de los mensajes, ya que si se produce un cambio, el resumen que se genera es diferente.

# 5.6 Funciones hash

- En general, las funciones hash se basan en la idea de funciones de compresión, que dan como resultado bloques de longitud n a partir de bloques de longitud m.

- Estas funciones se encadenan de forma iterativa, haciendo que la entrada en el paso i sea función del i-ésimo bloque del mensaje y de la salida del paso i-1




- Frecuentemente, se suele incluir en alguno de los bloques del mensaje m (al principio o al final), información sobre la longitud total del mensaje.

# 5.6 Funciones hash

Para que una función hash se considere segura, debe tener las siguientes características:

- Unidireccionalidad. Conocido un resumen h(m), debe ser computacionalmente imposible encontrar m a partir de dicho resumen.
- Compresión. A partir de un mensaje de cualquier longitud, el resumen h(m) debe tener una longitud fija. Lo normal es que la longitud de h(m) sea menor.
- Facilidad de cálculo. Debe ser fácil calcular h(m) a partir de un mensaje m.
- Difusión. El resumen h(m) debe ser una función compleja de todos los bits del mensaje m. Si se modifica un bit del mensaje m, el hash h(m) debería cambiar aproximadamente la mitad de sus bits.
- Colisión simple. Conocido m, será computacionalmente imposible encontrar otro m' tal que h(m) = h(m'). Se conoce como resistencia débil a las colisiones.
- Colisión fuerte. Será computacionalmente difícil encontrar un par (m, m') de forma que h(m) = h(m'). Se conoce como resistencia fuerte a las colisiones

# 5.6 Funciones hash

f5 39 b6 a0 3d d629 31 a2 47 ea 2f 31 89 a5 fd 2c

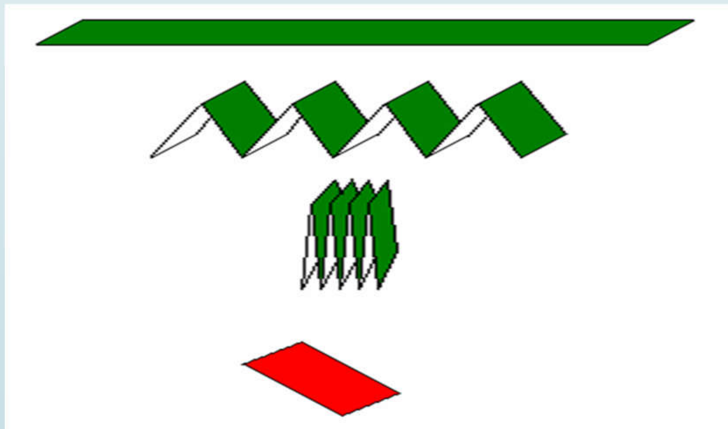
5d c7 92 ab 6f 02

69 2c b4 a7 31 04 f7 2c 84 f8 92 c5 b7 e8 f7 e9 02 a4 8e 2b 7e93 02 83 b3 cf 9f 24 7e f0 0c
5b 93 0f f2 8e 93 72 bc f3 71 c8 8e67 0f b2 7e ba 20 4d f1 47 02 a8 65 92 73 ff 6b 30 99 2c
de 73 b9 a0 82 b3 c9 02 83 59 a0 42 f9 ec 29 aa 2c 62 0e 83 02 c6 21

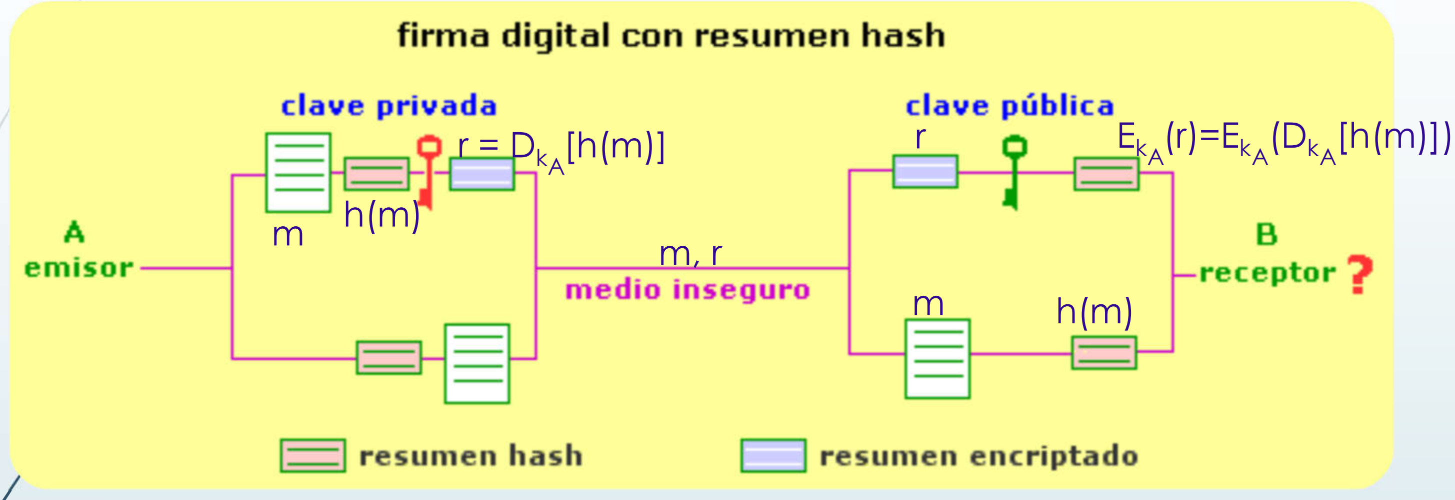
73 a9 03 b7 82 df 94 c5

8c4eff

00 c4 b7 83 d2 1c b5 72 a8



## ESQUEMA DE FIRMA DIGITAL MEDIANTE UNA FUNCIÓN HASH SIN CIFRADO




1. El emisor aplica una función hash conocida al documento, m, con lo que obtiene un resumen hash, h(m), del mismo.
2. Cifra dicho resumen con su clave privada $r = D_{k_A}[h(m)]$.
3. Envía al receptor el documento original en claro y el resumen hash cifrado:

   m, r.

1. El receptor B aplica la función hash al documento m sin cifrar y descifra el resumen cifrado con la clave pública del emisor A, $E_{k_A}(r)$.
2. Si ambos coinciden está seguro de que ha sido A el que le ha enviado el documento. Si no coinciden, está seguro de que no ha sido A o de que el envío ha sido interceptado y modificado.

# 5.6 Funciones hash: Algunas funciones hash

- **MD5**: Ron Rivest 1992. Mejoras al MD4 y MD2 (1990), es más lento pero con mayor nivel de seguridad que estas. Resumen de 128 bits.

- **SHA-1**: Del NIST, National Institute of Standards and Technology, 1994. Similar a MD5 pero con resumen de 160 bits. El NIST ha publicado una revisión del estándar, FIPS 180-2 , en la que se añaden 3 algoritmos de hash adicionales:
  - SHA-256, SHA-384, SHA-512,

  diseñados para que sean compatibles con el estándar de cifrado AES.

http://unaaldia.hispasec.com/2017/02/demostracion-practica-de-colision-en.html

https://shattered.io/

- **SHA-2**: es un conjunto de funciones hash criptográficas (SHA-224, SHA-256, SHA-384, SHA-512) diseñadas por la Agencia de Seguridad Nacional (NSA) y publicadas en 2001 por el Instituto Nacional de Estándares y Tecnología (NIST) como un Estándar Federal de Procesamiento de la Información (FIPS). Incluye un significante número de cambios respecto a su predecesor, SHA-1.

## 5.6 Funciones hash: Algunas funciones hash

- **RIPEMD-160**: Comunidad Europea, RACE, 1992. Resumen de 160 bits.
- N-Hash: Nippon Telephone and Telegraph, 1990. Resumen: 128 bits.
- Tiger: Ross Anderson, Eli Biham, 1996. Resúmenes de hasta 192 bits. Optimizado para máquinas de 64 bits (Alpha).
- Haval: Yuliang Zheng, Josef Pieprzyk y Jennifer Seberry, 1992. Admite 15 configuraciones diferentes. Hasta 256 bits.
- Etc…

# 5.6 Funciones hash: SHA-3

- **SHA-3 (Keccak)**: La competición para seleccionar el algoritmo criptográfico de hash para reemplazar a SHA-1 y a SHA-2, lanzada por el NIST EN 2007, finalizó con la elección oficial por parte del equipo NIST de Keccak como el nuevo algoritmo SHA-3. Tras seis años de proceso, se tomó una decisión y el algoritmo Keccak fue elegido como el nuevo SHA-3. Keccak es obra de Guido Bertoni, Joan Daemen, Michaël Peeter y Gilles Van Assche trabajadores de STMicroelectronics y NXP Semiconductors.

***NIST Cyptographic Hash Algorithm Competition* (SHA-3)**

http://csrc.nist.gov/groups/ST/hash/sha-3/index.html

https://en.wikipedia.org/wiki/NIST_hash_function_competition

gle

Función ganadora del concurso SHA-3: Keccak

# 5.6 Funciones hash

- **Píldoras formativas THOTH - Criptored**

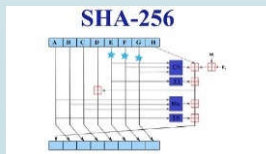
  http://www.criptored.upm.es/thoth/#

43. ¿Qué son y para qué sirven las funciones hash?


44. ¿Cómo funciona el hash MD5?


45. ¿Cómo funciona el hash SHA-1?


46. ¿Qué son SHA-2 y SHA-3?



**EJEMPLO DE FIRMA DIGITAL CON HASH**

En un chat cifrado con RSA, Alberto tiene como clave pública ($n_A$=143, $e_A$=17) y Bea ($n_B$=119, $e_B$=35).

Bea recibe el mensaje cifrado de Alberto c = 32 68 55 25 y firma digital s = 97.

El sistema utiliza el alfabeto

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | Ñ | O | P | Q | R | S | T | U | V | W | X | Y | Z |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 |

La firma digital se realiza sobre la suma de los elementos del texto en claro módulo $n_A$ para Alberto y módulo $n_B$ para Bea (*hash*).

a) ¿Qué mensaje en claro ha recibido Bea?

b) ¿Cómo comprueba Bea que el mensaje de Alberto es auténtico?

# 5.6 Funciones hash

**EJEMPLO DE FIRMA DIGITAL CON HASH**

a) En primer lugar, vamos a obtener la clave de descifrado de Bea.

Como $n_B = 119 = 7 \cdot 17$, se tiene que $\Phi(n_B) = 6 \cdot 16 = 96$.

Por definición, $d_B = e_B^{-1} \bmod 96 = 35^{-1} \bmod 96 = 11$.

$96 = 2 \cdot 35 + 26 \rightarrow 26 = 96 - 2 \cdot 35 \bmod 96 = -2 \cdot 35 \bmod 96;$
$35 = 1 \cdot 26 + 9 \rightarrow 9 = 35 - 1 \cdot 26 \bmod 96 = 35 - (-2) \cdot 35 \bmod 96 = 3 \cdot 35 \bmod 96;$
$26 = 2 \cdot 9 + 8 \rightarrow 8 = 26 - 2 \cdot 9 \bmod 96 = -2 \cdot 35 - 2 \cdot 3 \cdot 35 \bmod 96 = -8 \cdot 35 \bmod 96;$
$9 = 1 \cdot 8 + 1 \rightarrow 1 = 9 - 1 \cdot 8 \bmod 96 = 3 \cdot 35 - (-8) \cdot 35 \bmod 96 = 11 \cdot 35 \bmod 96$
$\rightarrow 35^{-1} \bmod 96 = 11$

La función de descifrado para Bea viene dada por $D_{KB}(c) = c^{11} \bmod 119$

$D_{K_B}(32) = 32^{11} \bmod 119 = 09 \rightarrow H$

$D_{K_B}(68) = 68^{11} \bmod 119 = 17 \rightarrow O$

$D_{K_B}(55) = 55^{11} \bmod 119 = 13 \rightarrow L$

$D_{K_B}(25) = 25^{11} \bmod 119 = 02 \rightarrow A$

Luego el mensaje en claro es m = HOLA

Sea $t = (09+17+13+02) \bmod n_A = 41 \bmod n_A = 41$ (*hash*)

## EJEMPLO DE FIRMA DIGITAL CON HASH

b) Para verificar la autenticidad del mensaje, Bea comprueba la firma digital de t, para ello, obtiene en primer lugar la rúbrica

$$D_{K_B}(s) = 97^{11} \text{ mod } 119 = 6 = r$$

y a continuación $E_{KA}$ (r) para compararlo con el mensaje original, t

$$E_{K_A}(r) = 6^{17} \text{ mod } 143 = 41 = t$$

luego el mensaje es auténtico.

**5.10)** En una red, Alicia desea enviar a Belén un mensaje m y firmarlo digitalmente. Para ello se uti-liza un algoritmo de clave pública en el que las funciones de cifrado y descifrado de Alicia son $\boldsymbol{E}_{k_A}$ **y** $\boldsymbol{D}_{k_A}$ y las de Belén son $\boldsymbol{E}_{k_B}$ **y** $\boldsymbol{D}_{k_B}$. El proceso seguido consiste en que Alicia cifra el mensaje $\mathbf{c} = \boldsymbol{E}_{k_B}(\mathbf{m})$, obtiene la firma digital $\mathbf{s} = \boldsymbol{E}_{k_B}\left[\boldsymbol{D}_{k_A}(\mathbf{m})\right]$ y envía los valores de c y s a Belén. ¿Qué proceso tiene que realizar Belén para descifrar c y comprobar que el mensaje es auténtico?

**5.4)** Consideremos un sistema de cifrado RSA en el que n=55 y e=7.

a) Cifra el número 10.

b) Factoriza n para obtener p y q y de esa manera descifrar el criptograma c=35.