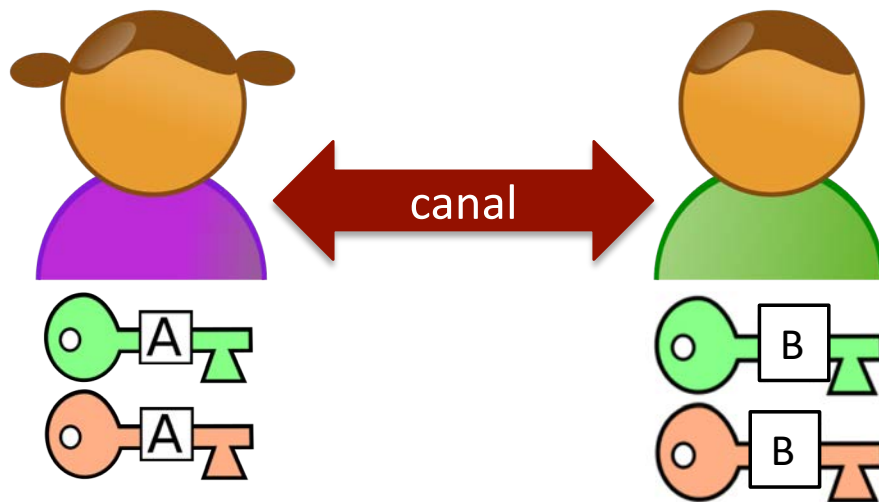


Algoritmos asimétricos (o de clave pública)



Introducción

- El concepto de criptografía de clave pública (o asimétrica) fue inventado en 1976 por *Diffie y Hellman*, e independientemente por *Merkle*, para dar solución a algunos de los problemas de los criptosistemas simétricos



- La asimetría reside en que, **las claves K y K^* son distintas**, al contrario de lo que ocurría en el caso simétrico

Introducción

- Las claves se utilizan por pares, de tal forma que cada usuario U posee dos claves:
 - una **clave pública**, conocida por todos los usuarios
 - una **clave privada**, conocida sólo por U
- Una clave se usa para cifrar, y la otra para descifrar
- Si se cifra un mensaje con una de las claves, esa misma no servirá para descifrar, sino que necesariamente habrá que usar la otra
- Existen tres funcionalidades básicas con cripto. asimétrica



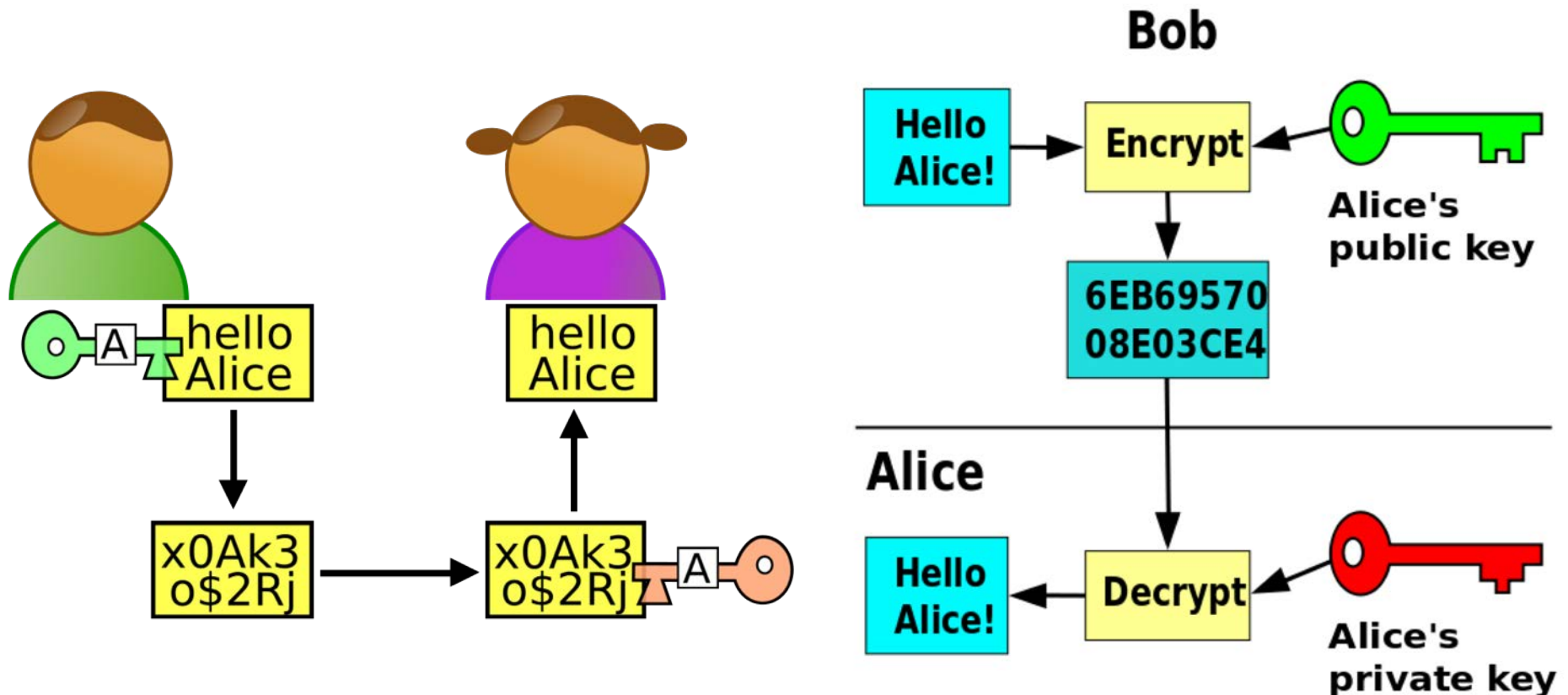
CIFRADO

FIRMA DIGITAL

INTERCAMBIO DE
CLAVES

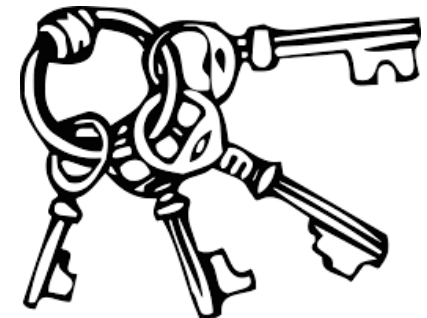
Cifrado/descifrado

- En el caso de que *Bob* necesite del servicio de confidencialidad para su comunicación con *Alice*, el **cifrado/descifrado** se realiza de la siguiente forma:



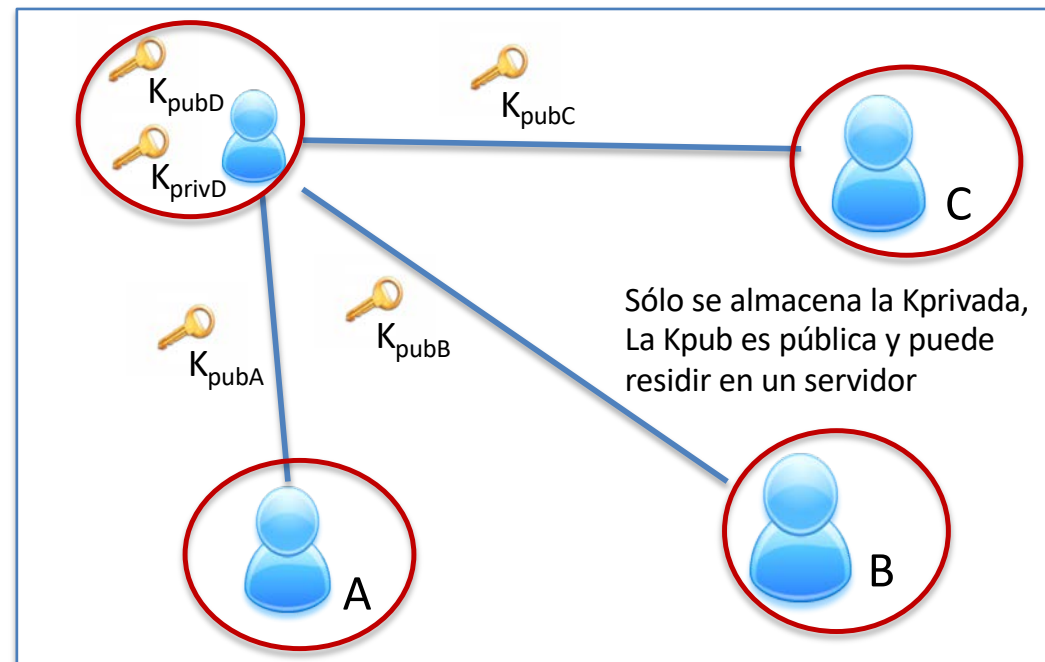
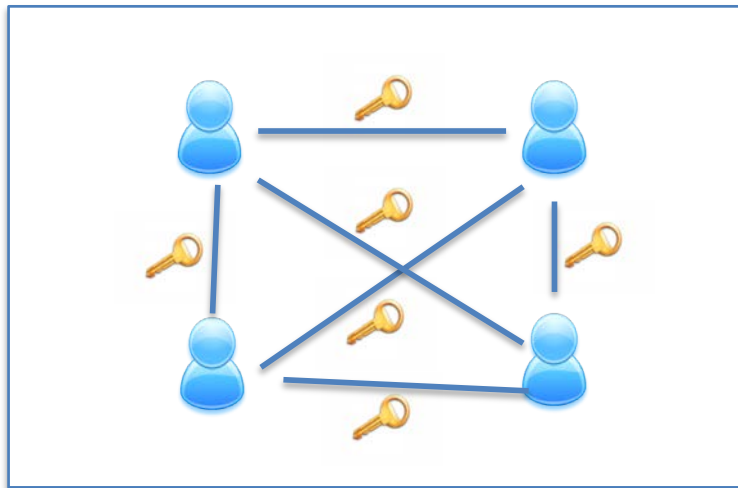
Cifrado/descifrado

- Del esquema anterior se entiende que *Alice* y *Bob* no necesitan acordar a priori ninguna clave (a diferencia de los algoritmos simétricos)
- Pero *Bob* ha de conocer la clave pública de *Alice*
 - y también la clave pública de cada uno de los usuarios con los que desee contactar
- Para ello, la solución más simple es que *Bob* almacene las claves públicas de los otros usuarios en un **key-ring** personal
 - Aunque la solución más común es que las claves públicas estén almacenadas en un **directorio de claves** en Internet



Cifrado/descifrado

- Se deduce fácilmente que, usando un criptosistema de clave pública, y para una comunidad de n usuarios:
 - el número de claves en el sistema será **$2n$**
 - en lugar de $(n * (n-1))/2$ como era el caso simétrico

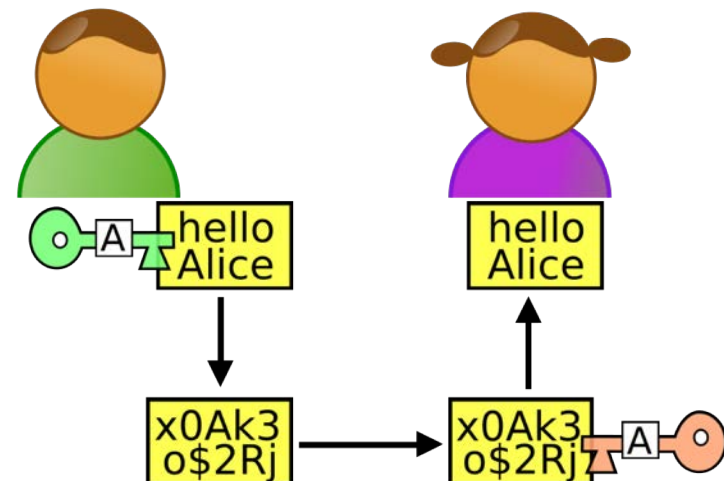


Cifrado/descifrado



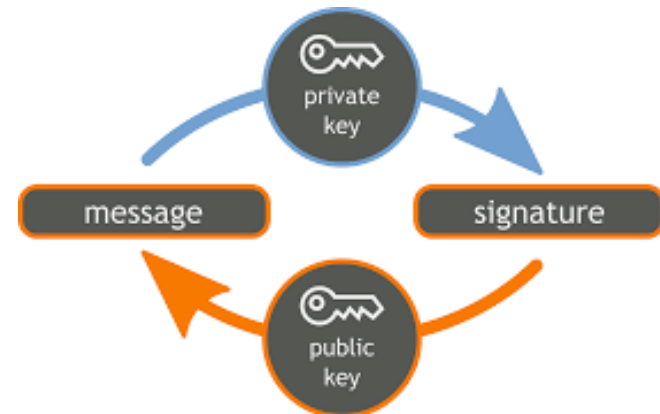
- Otras características relevantes:

- es **computacionalmente imposible deducir la clave privada** del usuario U a partir de su clave pública
- cualquier usuario con la **clave pública de U puede cifrar** un mensaje hacia U , pero no descifrarlo
 - cifrar el mensaje con la clave pública es como poner el correo en un buzón (todo el mundo puede hacerlo)
- sólo U , con la correspondiente **clave privada, puede descifrar** el mensaje
 - descifrar el mensaje con la clave privada es como coger el correo del buzón (sólo el que tiene la llave del buzón puede hacerlo)



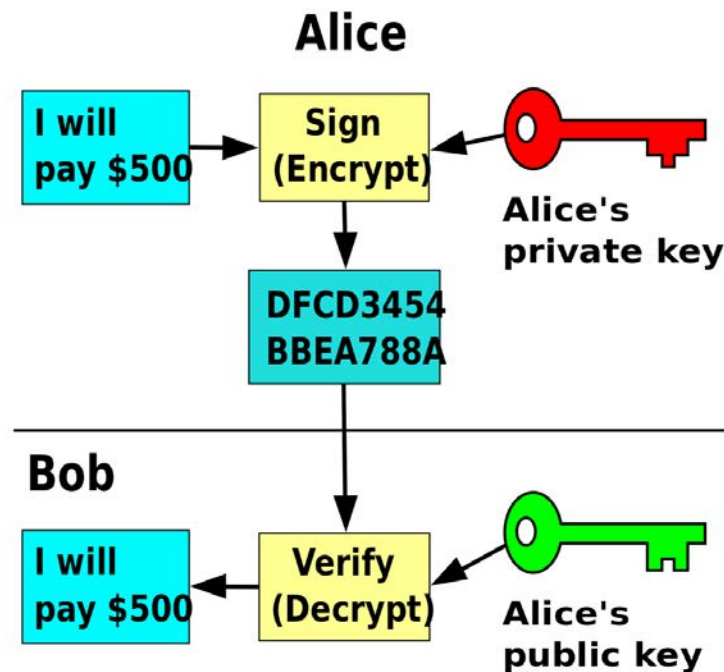
Cifrado/descifrado

- Por lo tanto, hemos visto tres **ventajas inmediatas de la criptografía de clave pública** con respecto a la simétrica
 1. Cuando dos usuarios se comunican confidencialmente **no necesitan acordar una clave a priori**
 2. Por lo anterior, **no resulta problemático que estén físicamente lejanos y no puedan reunirse presencialmente**
 3. El **número de claves** en el sistema **se reduce** sustancialmente
- Como se ve en la figura, existe aún una **ventaja adicional** tanto o más importante que las anteriores
 - Esa ventaja se deriva de la **dualidad de funcionamiento** de algunos (no todos) algoritmos de clave pública



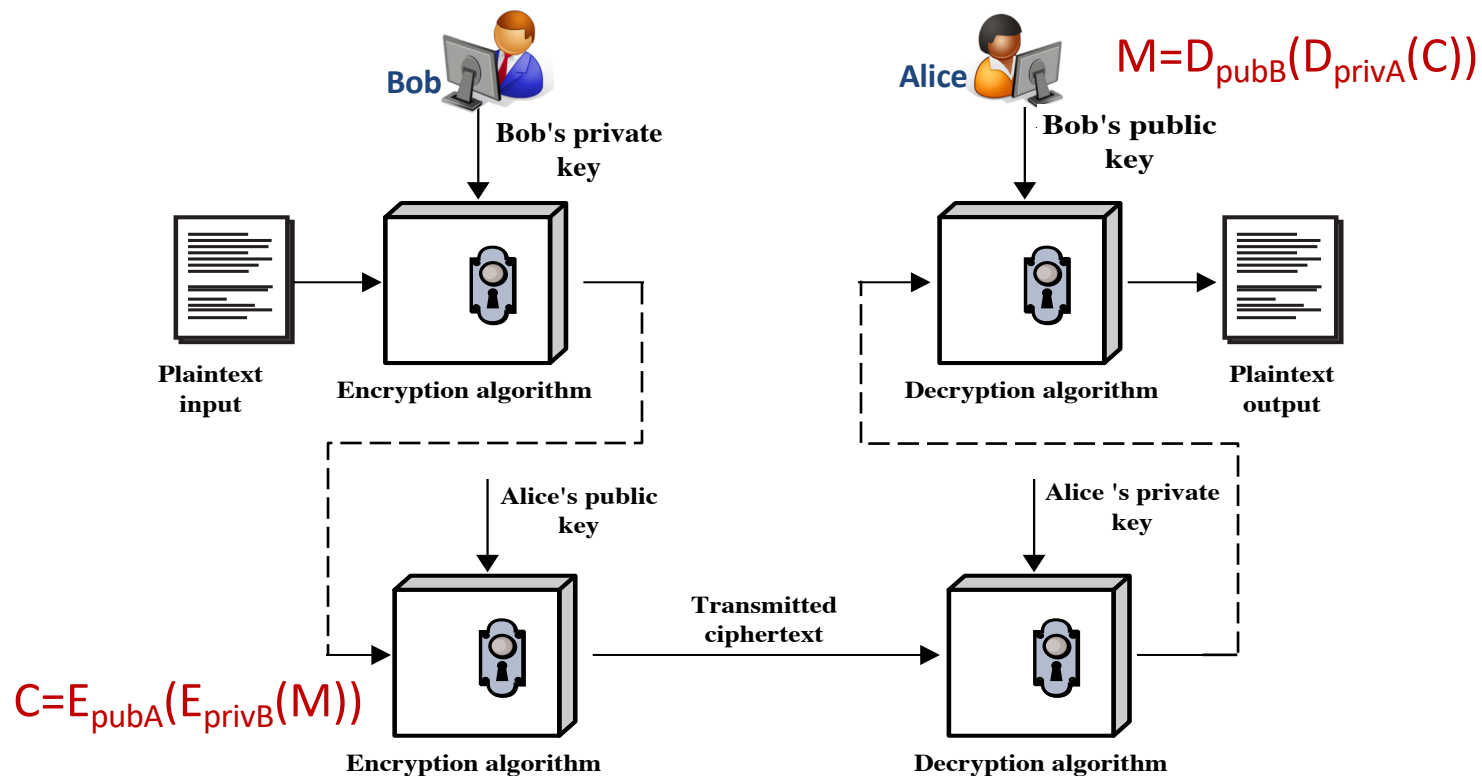
Firma digital

- *Bob* cifra el mensaje usando su propia clave privada, y cualquiera que tenga la clave pública de *Bob* podrá descifrar el criptograma
- Cuando *Alice* descifra el criptograma y obtiene el mensaje original, le queda garantizado que el mensaje viene de *Bob*
 - porque *Bob* es el único que pudo hacer la operación de cifrado (ya que sólo él posee la clave privada que generó el criptograma)



Firma digital

- Adicionalmente, es posible usar en secuencia las operaciones de firma digital y cifrado/descifrado, obteniendo autenticidad y confidencialidad en un mismo envío



Desventajas de criptografía de clave pública

- A pesar de estas ventajas, los algoritmos de clave pública tienen una **desventaja** de peso, el del **gran tamaño de sus claves**
 - Ejemplo de clave pública:

```
98 3f ad 19 36 93 3d 3e fe 76 42 14 fd 35 6f f1
fa ad 22 7a 58 e3 46 d0 5d c6 5a f9 62 2d 8f 31
5e fe b4 30 fe 50 74 ac d6 9d 1d e0 62 c6 49 dd
14 12 7d 71 0b ac 06 c1 3f d7 06 87 e0 90 89 d6
e5 e3 03 b2 f2 27 b1 9f 33 c8 aa 6b 36 4a a3 c4
3f 79 41 9d 89 46 2f 2b 3e 63 d4 38 56 91 aa 1d
b1 0d 42 75 4d f3 87 4e e3 0f 4d cc b4 6c bf 62
13 87 ea d0 9b 8e b6 e2 ff 19 f4 94 09 d5 96 61
```



- Además, los algoritmos de clave pública se basan en **funciones matemáticas complejas** (en lugar de en las convencionales sustituciones y permutaciones de los simétricos)
- Ambos hechos hacen que **el rendimiento** de estos algoritmos sea **sustancialmente menor** que el de los simétricos
 - en general, se **puede afirmar que son unos 1000 veces más lentos**

Desventajas de criptografía de clave pública

```
$ openssl speed rc4
```

To get the most accurate results, try to run this program when this computer is idle.

```
Doing rc4 for 3s on 16 size blocks: 73270739 rc4's in 2.99s
Doing rc4 for 3s on 64 size blocks: 19548456 rc4's in 2.99s
Doing rc4 for 3s on 256 size blocks: 5017905 rc4's in 2.99s
Doing rc4 for 3s on 1024 size blocks: 1274653 rc4's in 2.98s
Doing rc4 for 3s on 8192 size blocks: 159407 rc4's in 2.97s
```

```
$ openssl speed aes
```

To get the most accurate results, try to run this program when this computer is idle.

```
Doing aes-128 cbc for 3s on 16 size blocks: 30108378 aes-128 cbc's in 2.97s
Doing aes-128 cbc for 3s on 64 size blocks: 7712443 aes-128 cbc's in 2.96s
Doing aes-128 cbc for 3s on 256 size blocks: 1953741 aes-128 cbc's in 2.98s
Doing aes-128 cbc for 3s on 1024 size blocks: 490976 aes-128 cbc's in 2.98s
Doing aes-128 cbc for 3s on 8192 size blocks: 61237 aes-128 cbc's in 2.98s
Doing aes-192 cbc for 3s on 16 size blocks: 26695873 aes-192 cbc's in 2.98s
Doing aes-192 cbc for 3s on 64 size blocks: 6930418 aes-192 cbc's in 2.98s
Doing aes-192 cbc for 3s on 256 size blocks: 1729199 aes-192 cbc's in 2.97s
Doing aes-192 cbc for 3s on 1024 size blocks: 444845 aes-192 cbc's in 2.98s
Doing aes-192 cbc for 3s on 8192 size blocks: 52989 aes-192 cbc's in 2.97s
Doing aes-256 cbc for 3s on 16 size blocks: 23329778 aes-256 cbc's in 2.97s
Doing aes-256 cbc for 3s on 64 size blocks: 5958585 aes-256 cbc's in 2.98s
Doing aes-256 cbc for 3s on 256 size blocks: 1565944 aes-256 cbc's in 2.97s
Doing aes-256 cbc for 3s on 1024 size blocks: 377290 aes-256 cbc's in 2.97s
Doing aes-256 cbc for 3s on 8192 size blocks: 47844 aes-256 cbc's in 2.94s
```

```
$ openssl speed rsa
```

To get the most accurate results, try to run this program when this computer is idle.

```
Doing 512 bit private rsa's for 10s: 79651 512 bit private RSA's in 9.98s
Doing 512 bit public rsa's for 10s: 1079143 512 bit public RSA's in 9.95s
Doing 1024 bit private rsa's for 10s: 22746 1024 bit private RSA's in 9.96s
Doing 1024 bit public rsa's for 10s: 460663 1024 bit public RSA's in 9.96s
Doing 2048 bit private rsa's for 10s: 4362 2048 bit private RSA's in 9.96s
Doing 2048 bit public rsa's for 10s: 174994 2048 bit public RSA's in 9.97s
Doing 4096 bit private rsa's for 10s: 729 4096 bit private RSA's in 9.98s
Doing 4096 bit public rsa's for 10s: 50938 4096 bit public RSA's in 9.98s
```

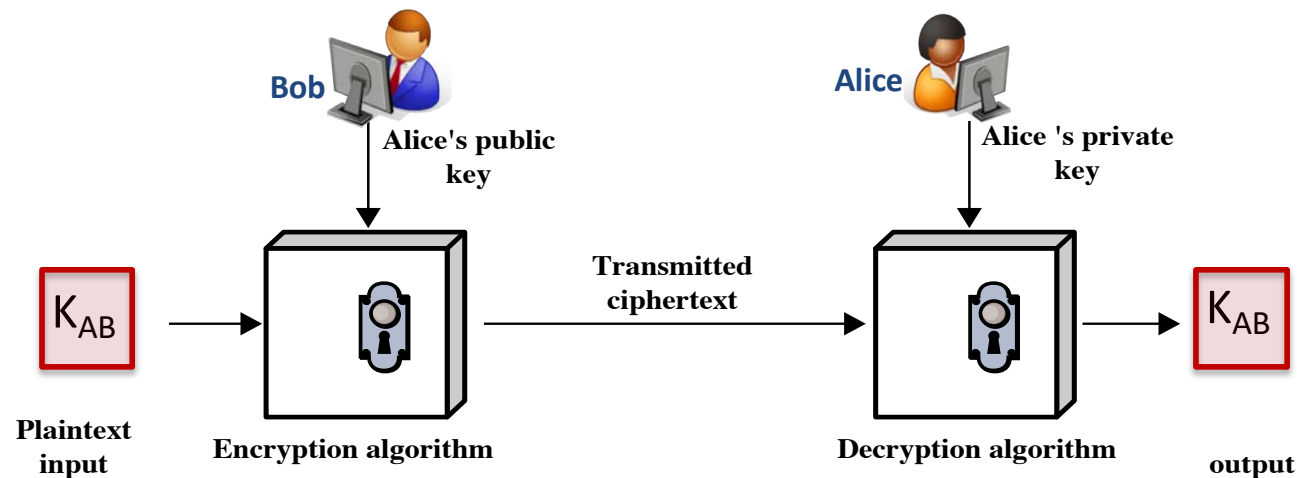
```
$ openssl speed dsa
```

To get the most accurate results, try to run this program when this computer is idle.

```
Doing 512 bit sign dsa's for 10s: 125836 512 bit DSA signs in 9.99s
Doing 512 bit verify dsa's for 10s: 114530 512 bit DSA verify in 9.99s
Doing 1024 bit sign dsa's for 10s: 54566 1024 bit DSA signs in 10.00s
Doing 1024 bit verify dsa's for 10s: 46194 1024 bit DSA verify in 10.00s
Doing 2048 bit sign dsa's for 10s: 18965 2048 bit DSA signs in 10.00s
Doing 2048 bit verify dsa's for 10s: 16315 2048 bit DSA verify in 10.00s
```

Intercambio de claves

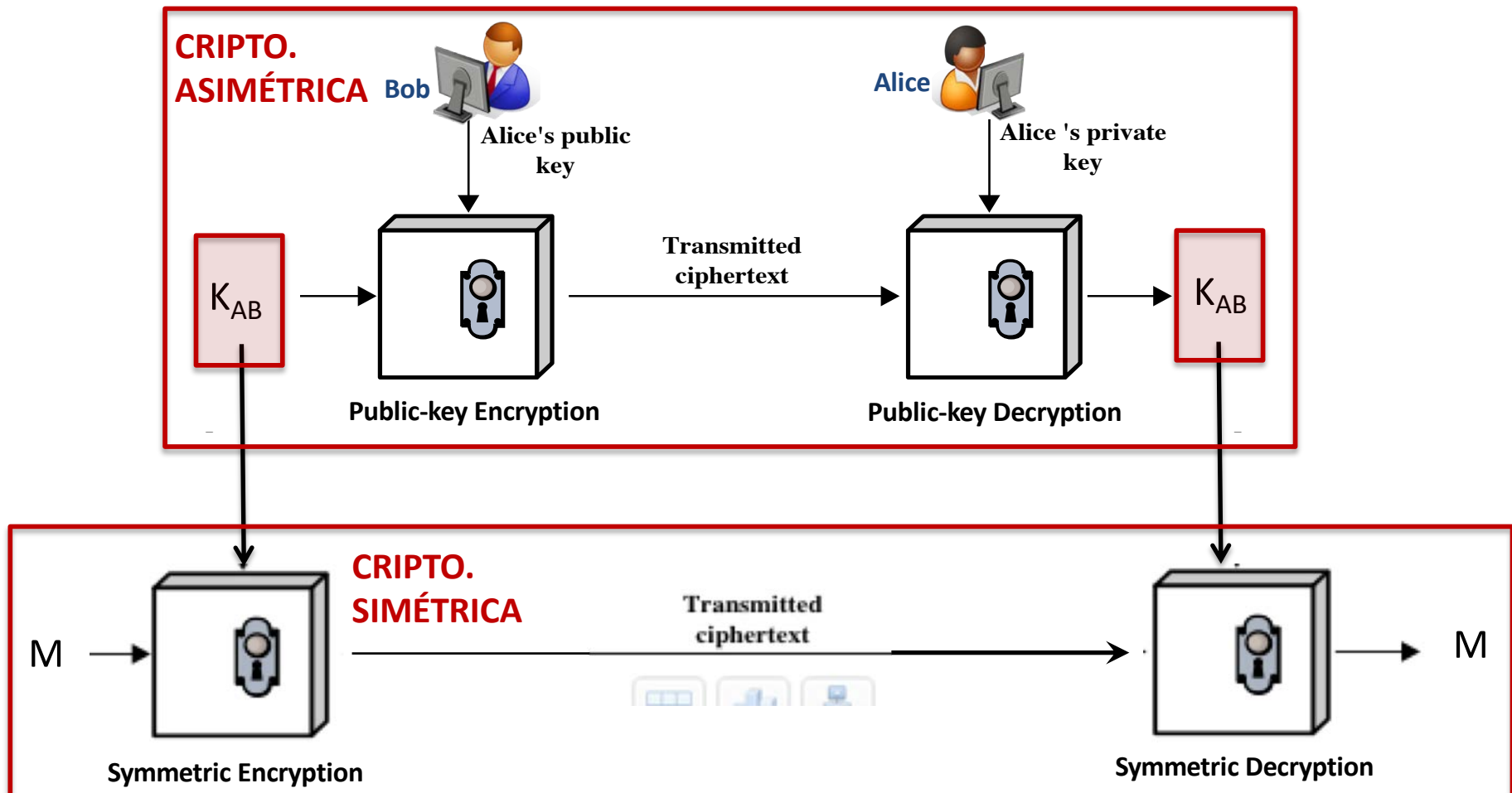
- Debido a su bajo rendimiento, se ha ideado una tercera funcionalidad para estos criptosistemas (además de cifrado/descifrado y firma digital): el **intercambio de claves**
 - *Paso 1:* Bob y Alice usan el algoritmo asimétrico para la transmisión (cifrado/descifrado) de la **clave secreta K_{AB}**
 - *Paso 2:* Ambos usarán K_{AB} para, posteriormente, cifrar sus comunicaciones con un algoritmo simétrico



- El resultado final es un **criptosistema híbrido** (uso de criptosistema de clave pública + uso de criptosistema simétrico)

Intercambio de claves

- Una evolución del planteamiento anterior es el siguiente:
 - se envía simultáneamente la clave de sesión y el mensaje cifrado con esa misma clave



Funcionalidades de la criptografía de clave pública

- Hay que hacer notar que, como muestra la siguiente tabla, no todos los algoritmos de clave pública son capaces de realizar las tres funcionalidades mencionadas:

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

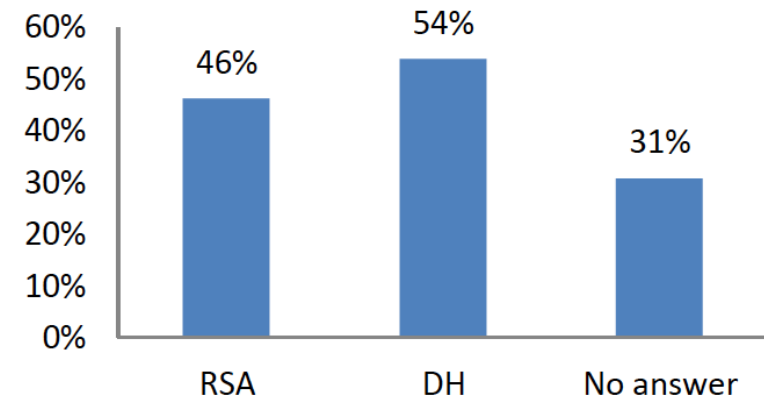
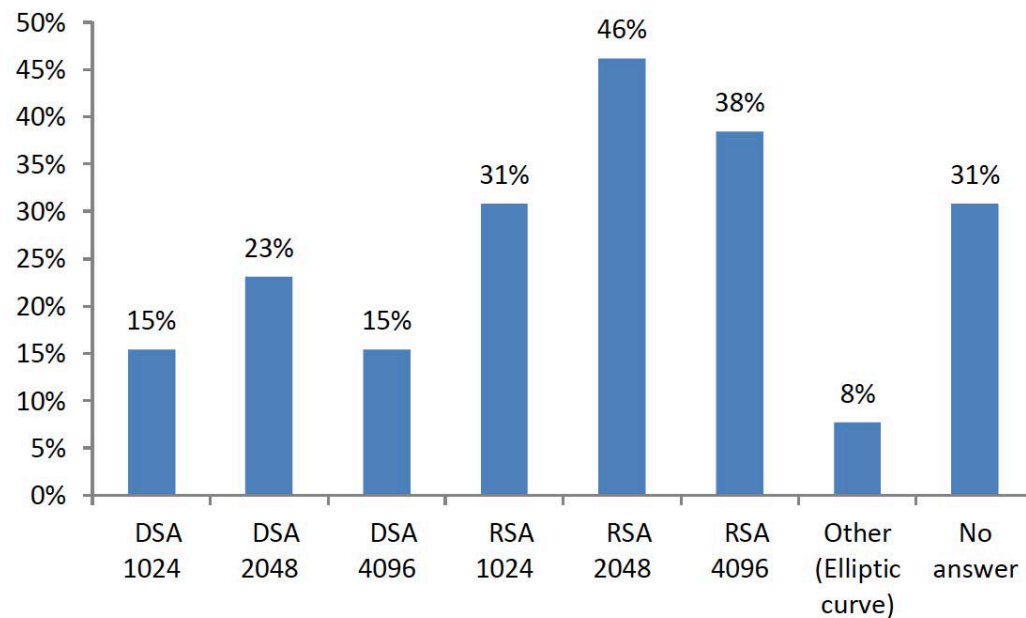
Cifrado/Descifrado

Firma digital

**Intercambio de
claves**

DSS: Digital Signature Standard, e incorpora
DSA (Digital Signature Algorithm)

- De acuerdo a un estudio reciente entre agencias e instituciones gubernamentales de la Unión Europea, el uso de algoritmos de clave pública es el mostrado en las figuras (para firma digital e intercambio de claves, respectivamente)



Algoritmo Diffie-Hellman

- Diseñado en 1976, se denomina normalmente “**algoritmo Diffie-Hellman de intercambio de clave**”
- Permite a dos usuarios cualesquiera **intercambiar**, de forma confidencial, una **clave secreta K (o de sesión)** para, posteriormente, cifrar de forma simétrica los mensajes entre ellos dos
- La efectividad del algoritmo depende de la **dificultad de computar logaritmos discretos**
 - Si α es una *raíz primitiva* del número primo q , entonces los números
$$\alpha \bmod q, \alpha^2 \bmod q, \dots, \alpha^{q-1} \bmod q$$
son distintos entre sí, y sus valores son los enteros 1 a $q-1$, en cualquier orden
 - Para cualquier entero b y una raíz primitiva α del número primo q , se puede encontrar un exponente único i tal que:
$$b \equiv \alpha^i \pmod{q} \quad \text{donde } 0 \leq i \leq (q-1)$$
 i es el **logaritmo discreto de b** , y se representa $dlog_{\alpha,q}(b)$

Global Public Elements

q prime number

α $\alpha < q$ and α a primitive root of q

User A Key Generation

Select private X_A $X_A < q$

Calculate public Y_A $Y_A = \alpha^{X_A} \bmod q$

User B Key Generation

Select private X_B $X_B < q$

Calculate public Y_B $Y_B = \alpha^{X_B} \bmod q$

Calculation of Secret Key by User A

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

Calculation of Secret Key by User B

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

**Algoritmo
Diffie-Hellman
de intercambio
de clave**

User A

Generate
random $X_A < q$;
Calculate
 $Y_A = \alpha^{X_A} \bmod q$

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

User B

Generate
random $X_B < q$;
Calculate
 $Y_B = \alpha^{X_B} \bmod q$;
Calculate
 $K = (Y_A)^{X_B} \bmod q$

Y_A

Y_B

**Secuencia ordenada
de pasos del algoritmo**

- Alice:

- Valores públicos: q, α

- $X_a < q$: clave privada

- $Y_a = \alpha^{X_a} \bmod q$

- $Y_b = \alpha^{X_b} \bmod q$

- $K = (\alpha^{X_b} \bmod q)^{X_a} \bmod q$

- $K = (\alpha^{X_b})^{X_a} \bmod q$

- Bob:

- Valores públicos: q, α

- $X_b < q$: clave privada

- $Y_b = \alpha^{X_b} \bmod q$

- $Y_a = \alpha^{X_a} \bmod q$

- $K = (\alpha^{X_a} \bmod q)^{X_b} \bmod q$

- $K = (\alpha^{X_a})^{X_b} \bmod q$

$$K = \alpha^{X_b X_a} \bmod q = \alpha^{X_a X_b} \bmod q$$

- Ejemplo:

① $q = 353$; $\alpha = 3$ (3 es raíz primitiva de 353)

② A y B seleccionan sus correspondientes claves privadas: $X_A = 97$, $X_B = 233$

③ A calcula su clave pública: $Y_A = 3^{97} \bmod 353 = 40$

④ B calcula su clave pública: $Y_B = 3^{233} \bmod 353 = 248$

⑤ A y B se intercambian sus claves públicas

⑥ A calcula $K = (Y_B)^{X_A} \bmod 353 = 248^{97} \bmod 353 = 160$

⑦ B calcula $K = (Y_A)^{X_B} \bmod 353 = 40^{233} \bmod 353 = 160$

- Ejemplo (continuación):
 - Se asume que el atacante tendrá la siguiente información:
 $q = 353; \alpha = 3; Y_A = 40; Y_B = 248$
 - En este ejemplo simple, sería posible usar un ataque exhaustivo para determinar que $K = 160$
 - El atacante calcula fácilmente $3^{X_a} \bmod 353 = 40$ o bien $3^{X_b} \bmod 353 = 248$
 - Con números más grandes el problema no es resoluble en un tiempo razonable

- http://dkerr.home.mindspring.com/diffie_hellman_calc.html

Example: Diffie - Hellman Define Public Values: $n = 997$
 $g = 2$

Both Alice and Bob each pick a private x and compute a public $X = g^x \bmod n$

Alice

Alice chooses a Private Value $a = 12$

Bob

Bob chooses a Private Value $b = 13$

Compute Values Below - or - Clear Input Values - or - Display Usage Explanation

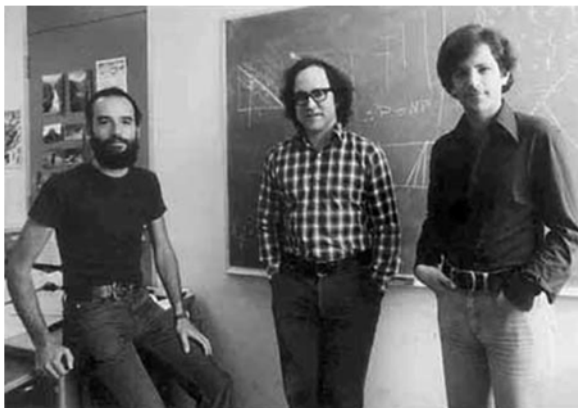
Alice computes Public Value: $A = g^a \bmod n$ Bob computes Public Value: $B = g^b \bmod n$
 (Public) $A = 108$ (Public) $B = 216$

Alice and Bob exchange Public Values
 Alice and Bob each compute Same Master Value M

Alice computes $M = B^a \bmod n = g^{ba} \bmod n$ Bob computes $M = A^b \bmod n = g^{ab} \bmod n$
 Alice computes $M = 480$ Bob computes $M = 480$

Algoritmo RSA

- Es el criptosistema de clave pública más usado
 - Su nombre procede de sus inventores: *Rivest, Shamir y Adleman* del Instituto Tecnológico de Massachusetts (MIT)
- Estos se basaron para su invención, en 1.977, en una idea bastante “simple”:
 - *“es muy fácil multiplicar dos números enteros primos grandes, pero extremadamente difícil hallar la factorización de ese producto”*
 - puede darse a conocer dicho producto sin que nadie descubra esos números de procedencia, a no ser que conozca al menos uno de ellos



Se piensa que RSA será seguro mientras no se conozcan “**formas rápidas**” de descomponer un número grande en producto de primos

- Parámetros necesarios:

- encontrar dos números primos grandes p y q , y calcular

$$n = p * q \quad ; \quad p \neq q$$

- se calcula $\varphi(n)$, de forma que

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



Para extraer los primos,
donde $\varphi(n)$ se define como
el número de enteros positivos
menores o iguales a n y coprimos de n

- se elige aleatoriamente un número grande e tal que

$$\text{MCD}(e, \varphi(n)) = 1; \quad e < \varphi(n)$$

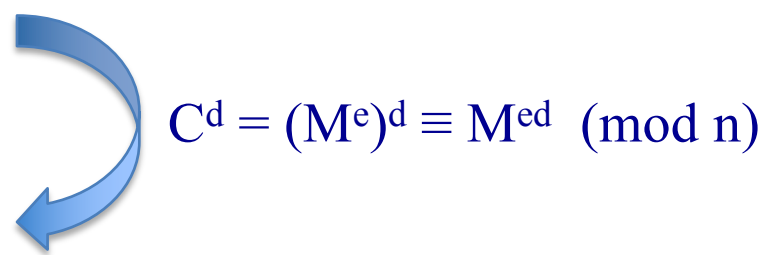
(o sea, e y $\varphi(n)$ son primos relativos o coprimos)

- Se determina el número d que cumple

$$e * d \equiv 1 \pmod{\varphi(n)}$$

(donde d debe ser el multiplicador inverso de $e \bmod \varphi(n)$)

Calcular
las
claves

- n , d y e (y por supuesto, p y q) son la base del sistema
 - n módulo
 - d clave privada
 - e clave pública
 - Las claves tienen el siguiente uso:
 - la **clave privada**: para descifrar o firmar mensajes
 - la **clave pública**: para cifrar o verificar la firma
 - Para *cifrar*: $C = M^e \pmod{n}$
 - Para *descifrar*: $M = C^d \pmod{n}$
- 
- $$C^d = (M^e)^d \equiv M^{ed} \pmod{n}$$

- Ejemplo del cálculo de la clave pública y privada:

① Seleccionar primos: $p = 17$, $q = 11$

① Calcular $n = pq = 17 \times 11 = 187$

② Calcular $\phi(n) = (p-1)(q-1) = 16 \times 10 = 160$

③ Seleccionar e : $\text{mcd}(e, 160) = 1$ y $e < 160$
se selecciona $e = 7$

⑤ Determinar d : $d \cdot e \equiv 1 \pmod{160}$
 $d = 23$ dado que $23 \times 7 = 161 \pmod{160} = 1$

⑥ Clave pública = 7 y módulo = 187

⑦ Clave privada = 23

- El texto original, M (y también el cifrado, C) debe tomarse como un número decimal. De esta forma, si se trabaja con el código ASCII:

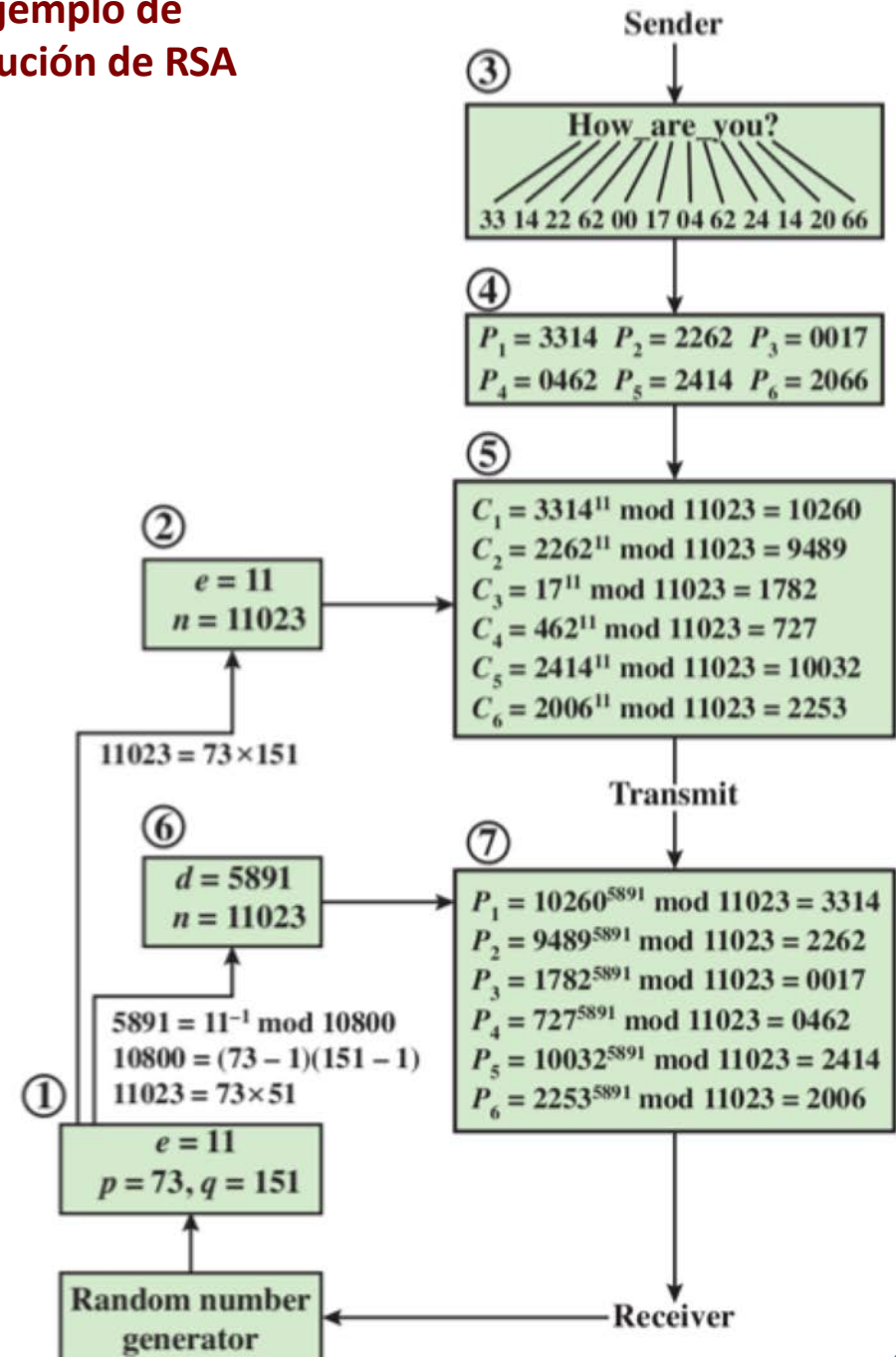
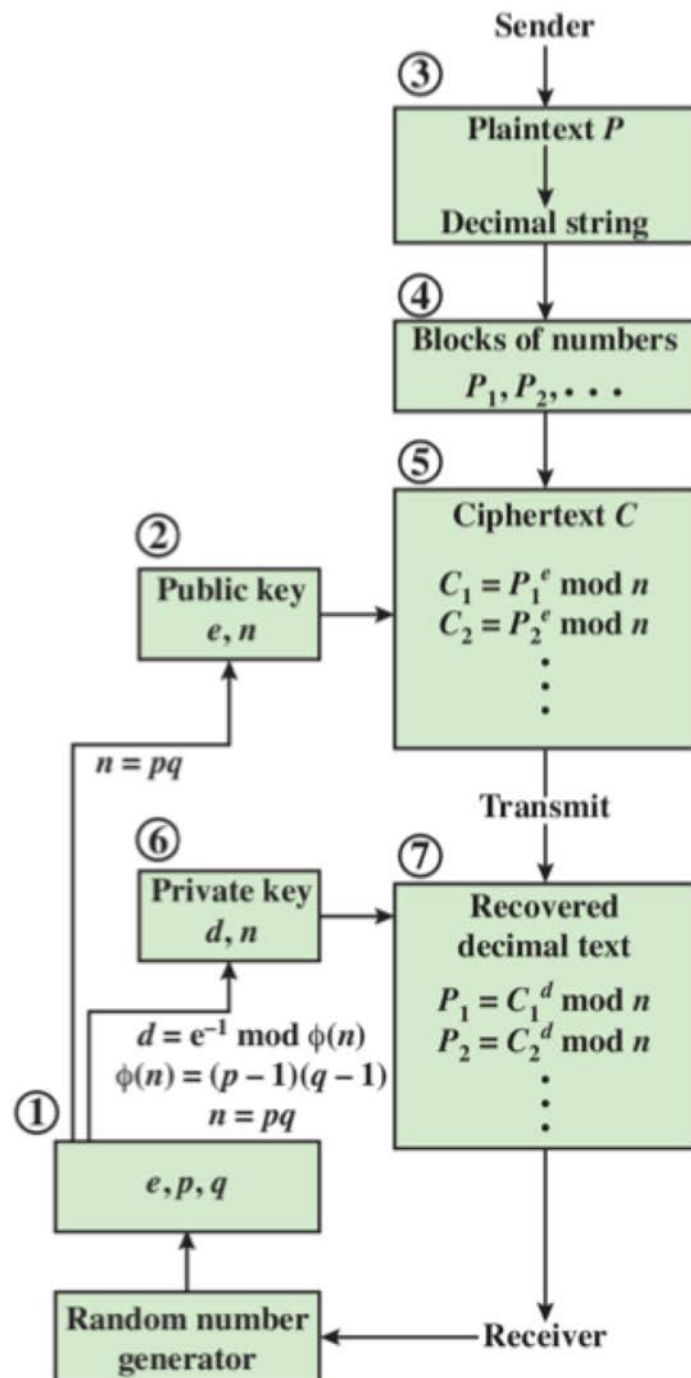
$M = \text{“Hola”}$ equivaldría al valor decimal 72 111 108 097

- Además, como se está trabajando en aritmética modular, todos los valores usados deben ser menores que el módulo n
 - Esto significa que si el valor decimal del mensaje es superior al módulo, $M > n$, entonces M debe ser troceado en otros menores que n
 - Suponiendo $n = 100000$, daría lugar a los bloques

$$m_0 = 72111, m_1 = 10809 \text{ y } m_2 = 7$$

- Los bloques m_i no deben/pueden ser pequeños, pues entonces el criptoanalista puede construirse una tabla donde tenga todos los posibles m_i y sus respectivos c_i
- Para seleccionar p ó q debe testearse si es primo o no con cualquiera de los tests existentes (se recomienda *Miller-Rabin*)
- También hay que tener cuidado al elegir el tamaño
 - **cuanto mayor sea, más seguro, pero también más lento** será el sistema en sus cálculos
- Se han propuesto sistemas en los que se busca un valor e muy pequeño, con lo cual hacer más rápido el proceso de cifrado

Ejemplo de ejecución de RSA



Comparativa: Criptografía Simétrica vs. Asimétrica

Atributo	Clave simétrica	Clave asimétrica
Años en uso	Miles	Menos de 50
Uso principal	Cifrado de grandes volúmenes de datos	Intercambio de claves; firma digital
Estándar actual	DES, Triple DES, AES	RSA, Diffie-Hellman, DSA
Velocidad	Rápida	Lenta
Claves	Compartidas entre emisor y receptor	Privada: sólo conocida por una persona Pública: conocida por todos
Intercambio de claves	Difícil de intercambiar por un canal inseguro	La clave pública se comparte por cualquier canal La privada nunca se comparte
Longitud de claves	56 bits (vulnerable) 256 bits (seguro)	1024 – 2048 (RSA) 172 (curvas elípticas)
Servicios de seguridad	Confidencialidad Integridad Autenticación	Confidencialidad Integridad Autenticación, No repudio