



**UNIVERSIDAD NACIONAL DE LA MATANZA**  
Departamento de Ingeniería e Investigaciones Tecnológicas

## **Seguridad y Calidad en Aplicaciones Web**



### **Unidad N° 3: Criptografía**

Referente de Cátedra: Walter R. Ureta

Plantel Docente: Emiliano Zarate, Pablo Pomar,  
Walter R. Ureta



## **Historia de la criptografía**



## Historia de la criptografía

- 1900 AC En el antiguo Egipto se usaron símbolos que no eran los normales.
- 1500 AC Los fenicios diseñaron un alfabeto.
- 1000 AC Se usaron otros símbolos distintos a los normales en la antigua Mesopotamia.
- 600 AC En Palestina se cifran textos usando un algoritmo simple de sustitución monoalfabética Atbash.
- 500 AC Los espartanos cifran mensajes utilizando Scytale (escítala).
- 400 AC El Kamasutra describe un algoritmo de cifrado por sustitución monoalfabética.
- 100-44 AC Julio César inventa un código para cifrar sus mensajes (el Código AC César). Éste es el algoritmo de sustitución monoalfabética más conocido.
- 500-1400 DC La "edad oscura de la criptografía" empieza en Europa, se considera como magia negra: Durante este florece en Persia.



## **Historia de la criptografía**

- 1400 DC En Italia se produce un boom de la criptografía debido un alto desarrollo de la vida diplomática.
- 1795 DC Thomas Jefferson diseña el primer dispositivo de cifrado cilíndrico, conocido como la "rueda de Jefferson".
- 1917 DC El americano Gilbert S. Vernam, empleado de AT&T, desarrolla la cinta aleatoria de un sólo uso, el único sistema criptográfico seguro.
- 1918 DC Arthur Scherbius y Richard Ritter inventan la primera Enigma. Al mismo tiempo, la máquina de rotores es inventada y patentada por Alexander Koch (Países Bajos) y Arvid Damm (Suecia).
- 1940-1945 DC - Alan Turing rompe Enigma con la idea de la Bomba de Turing que concibió basándose en el trabajo de Marian Rejewski.
- 1948-1949 DC Claude Shannon establece la bases matemáticas de la teoría de la información y publica "Communication Theory of Secrecy Systems", en donde expone un algoritmo de cifrado teóricamente irrompible que debe satisfacer los requisitos de la cinta aleatoria de un sólo uso.



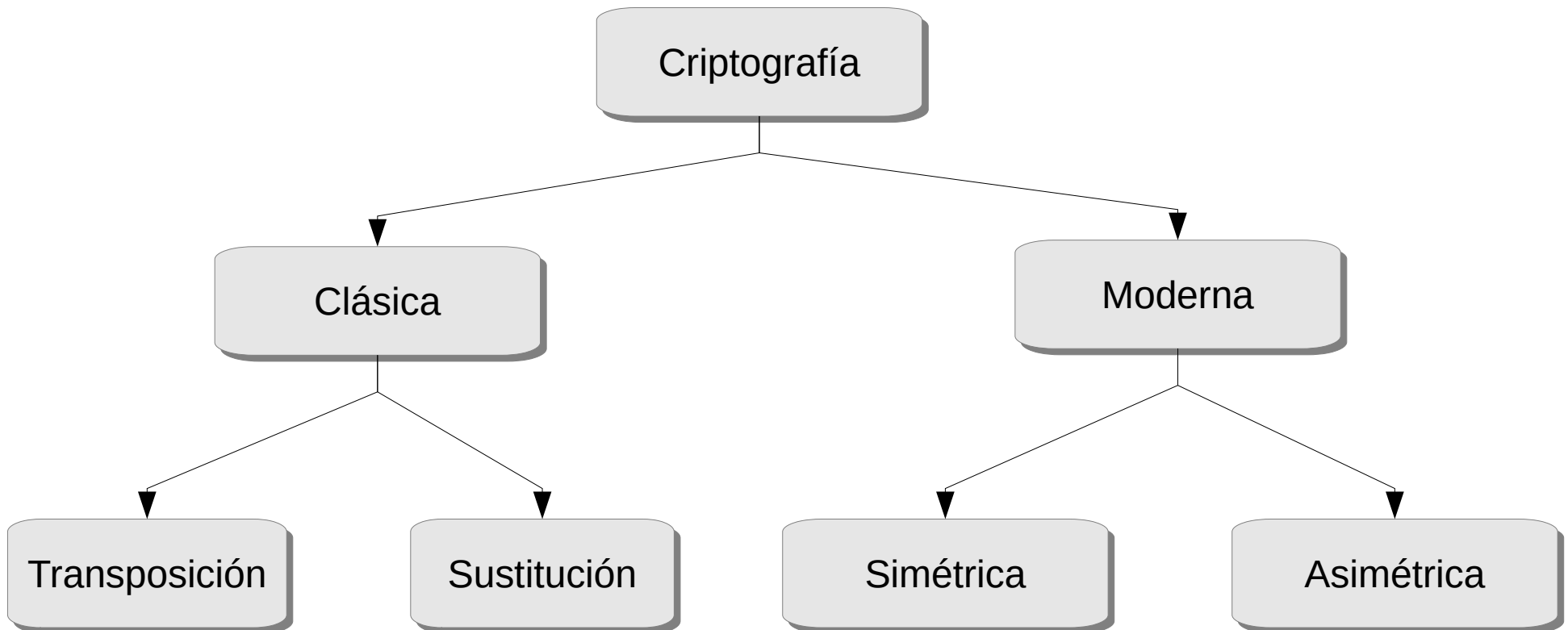
## Elementos teóricos de la criptografía

$$D_k (E_k (m)) = m$$

- **m** representa el conjunto de todos los mensajes sin cifrar (lo que se denomina texto claro, o plaintext) que pueden ser enviados.
- **C** representa el conjunto de todos los posibles mensajes cifrados, o criptogramas.
- **k** representa el conjunto de claves que se pueden emplear en el criptosistema.
- **E** es el conjunto de transformaciones de cifrado o familia de funciones que se aplica a cada elemento de **M** para obtener un elemento de **C**. Existe una transformación diferente **E<sub>k</sub>** para cada valor posible de la clave **k**.
- **D** es el conjunto de transformaciones de descifrado, análogo a **E**.

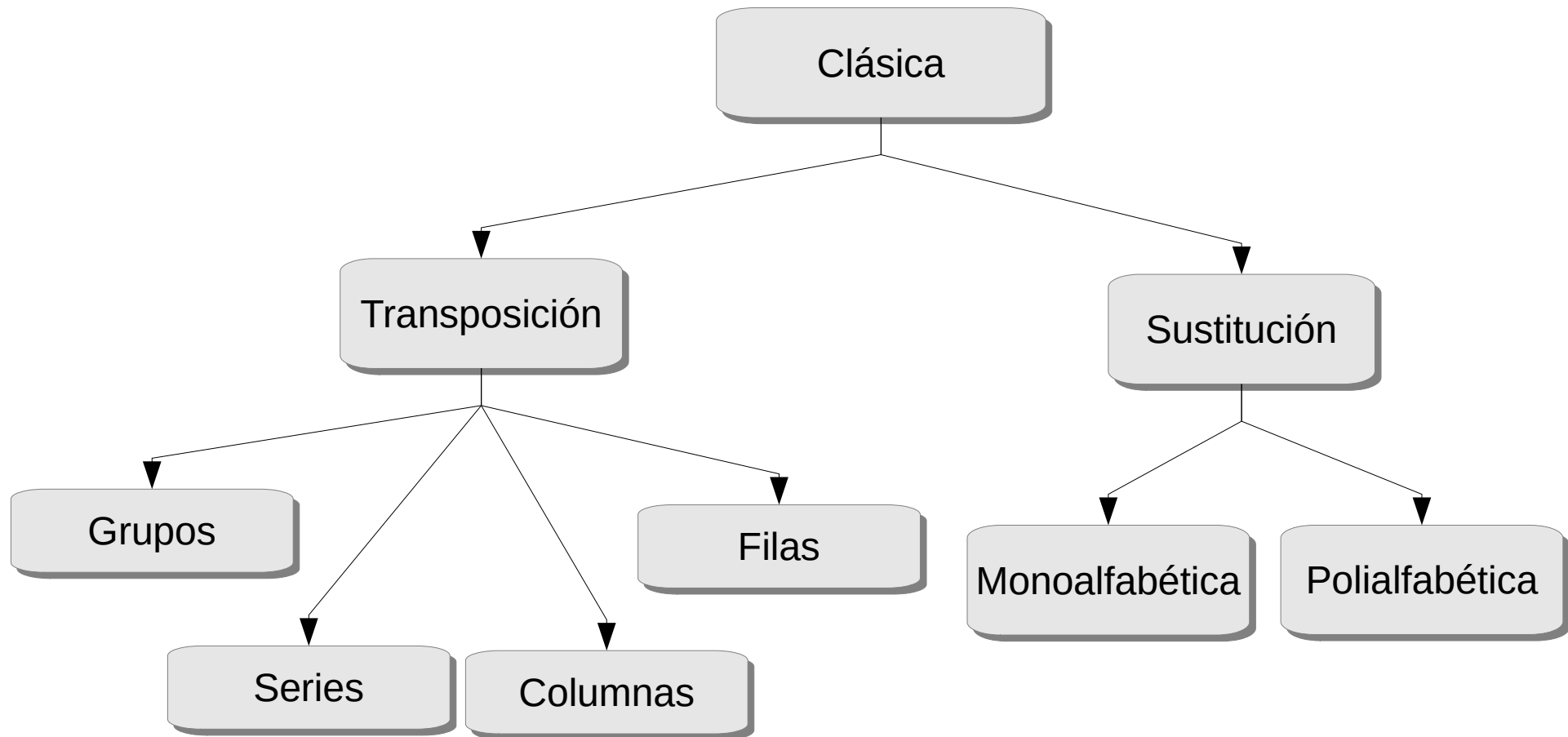


## Clasificación





## Criptografía clásica Sub-Clasificación

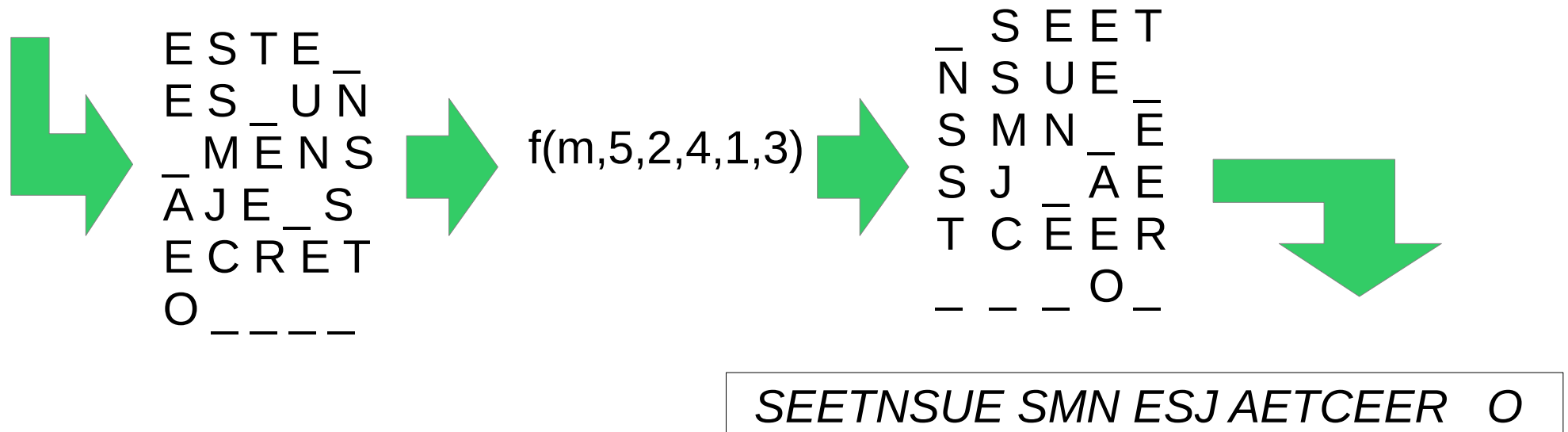




## Criptografía clásica

Los **cifradores por transposición** utilizan la técnica de permutación de forma que los caracteres del texto se reordenan mediante un algoritmo específico.

*ESTE ES UN MENSAJE SECRETO*

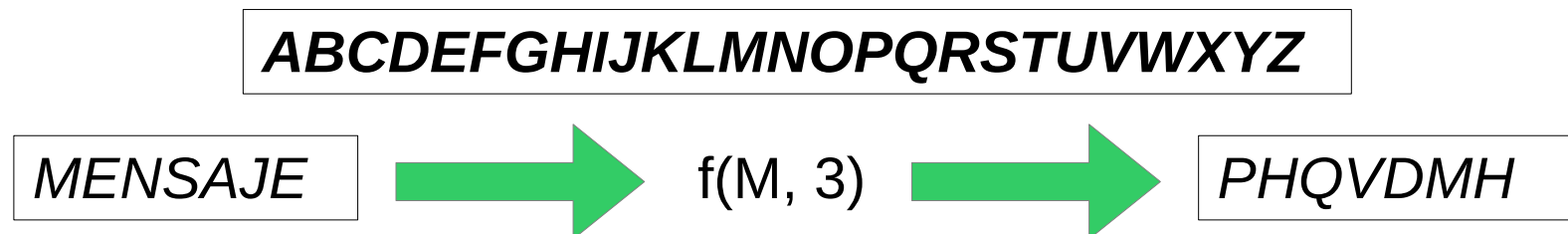






## Criptografía clásica

Los **cifradores por sustitución** utilizan la técnica de modificación de cada carácter del texto en claro por otro correspondiente al alfabeto de cifrado. Si el alfabeto de cifrado es el mismo que el del mensaje o bien el único, hablamos entonces de cifradores monoalfabéticos; es decir, existe un único alfabeto en la operación de transformación del mensaje en criptograma. Por el contrario, si en dicha operación intervienen más de un alfabeto, se dice que el cifrador es polialfabético. Por ejemplo, el cifrado del Cesar (**monoalfabético**)





## Criptografía clásica

Los **cifradores por sustitución polialfabética** utilizan diferentes caracteres para el reemplazo de un mismo carácter de origen.

**Ejemplo:** Cifrado de Vigenere, se basa en una matriz cuyos filas y columnas son alfabetos en orden.

El texto a cifrar es:

**TEXTO DE PRUEBA**

Se ha cifrado con la clave:


**ABCD**

Clave expandida:


**ABCDABCDABCDABC**

El texto cifrado es:

**TFZWO EGSRV GEA**

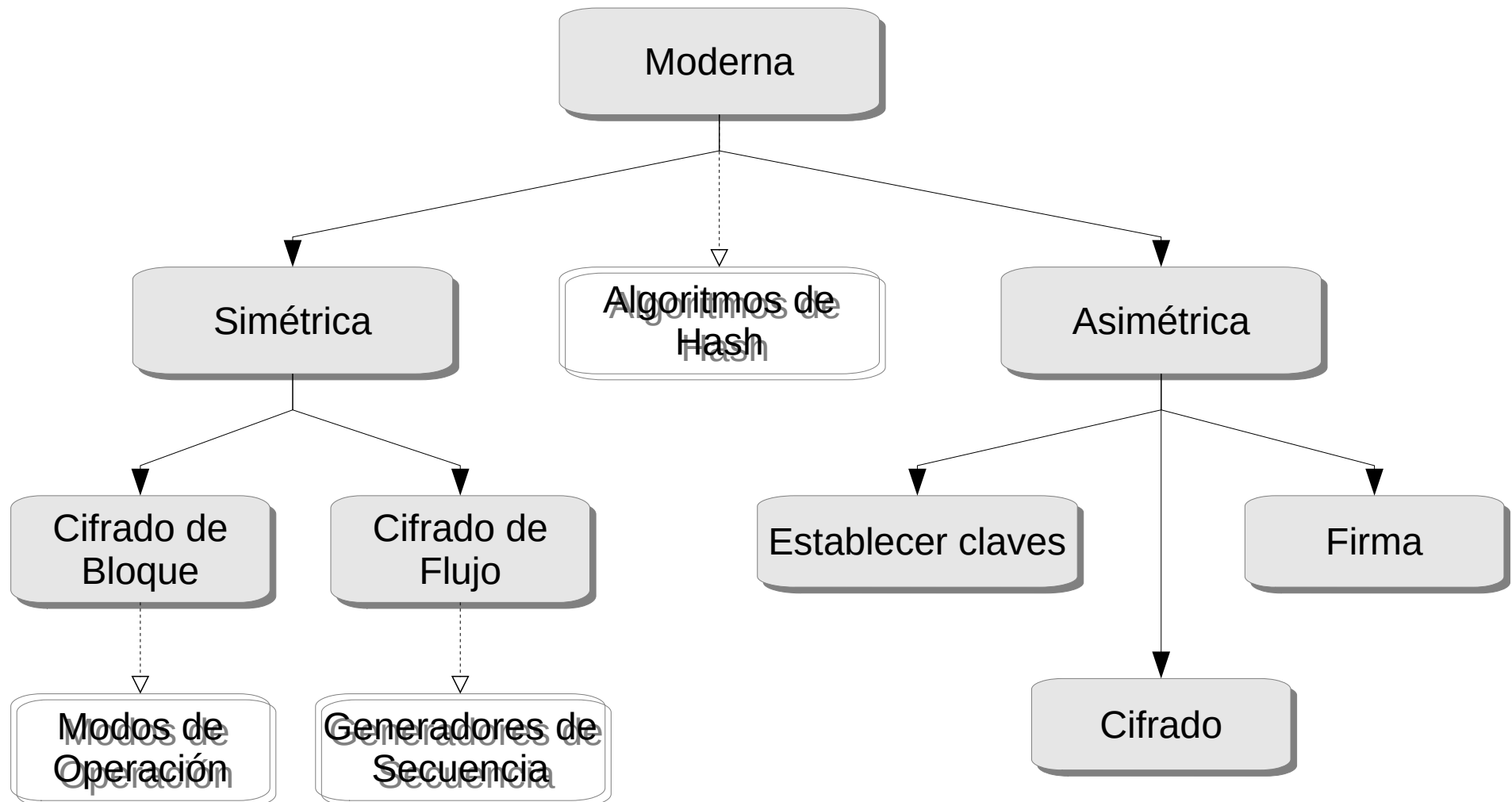


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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





## Criptografía moderna Sub-Clasificación





## **Algoritmos simétricos**

Un sistema de cifrado simétrico es un tipo de cifrado que usa una misma clave para cifrar y para descifrar. Las dos partes que se comunican mediante el cifrado simétrico deben estar de acuerdo en la clave a usar de antemano. Una vez de acuerdo, el remitente cifra un mensaje usando la clave, lo envía al destinatario, y éste lo descifra usando la misma clave.



## **Algoritmos simétricos**

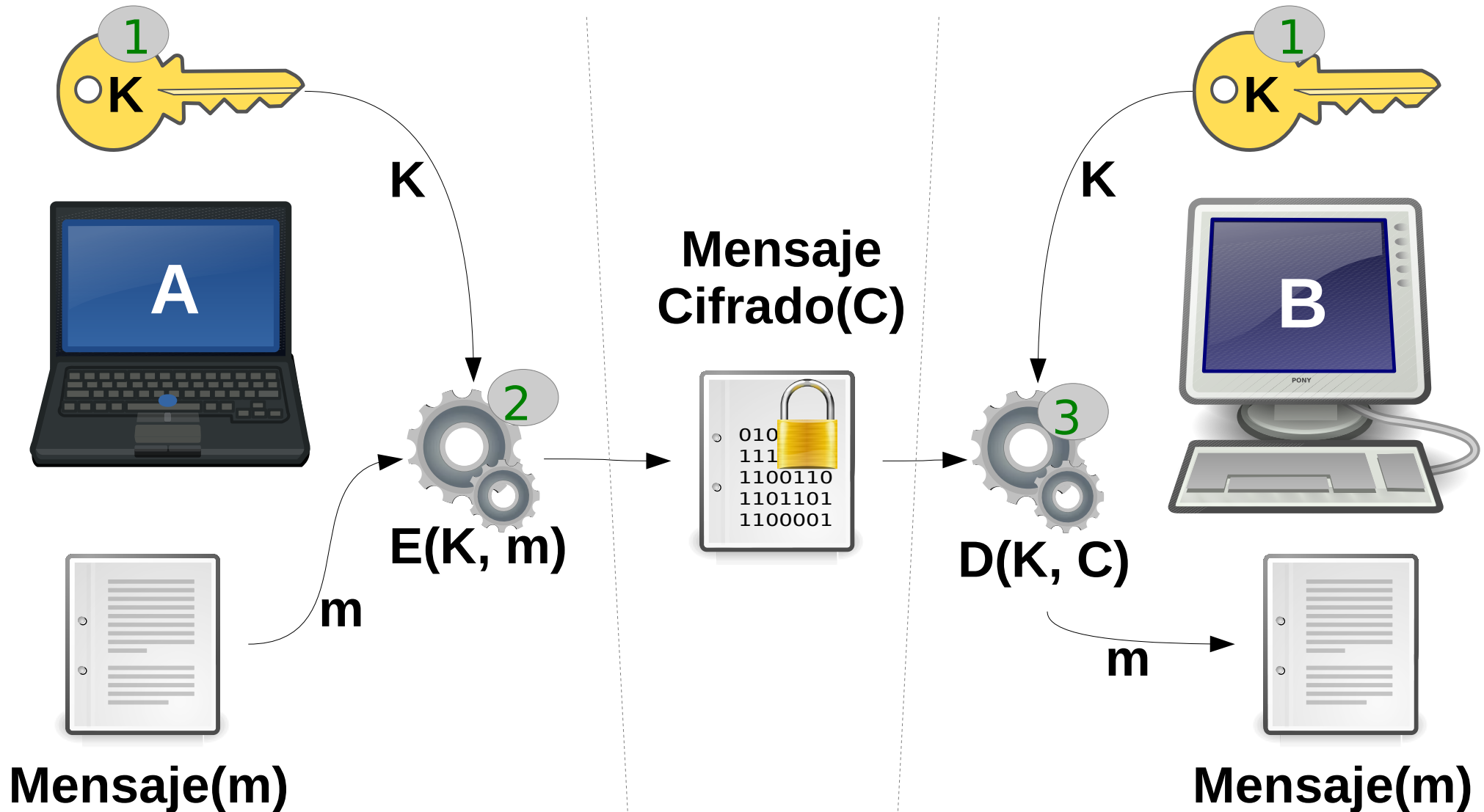
### **Ventajas**

- Sencillez de implementación
- Robustez
- Velocidad de cifrado
- Longitud del mensaje

### **Desventajas**

- La clave debe ser compartida previamente con seguridad
- La comunicación entre múltiples actores requiere numerosas claves

## Algoritmos simétricos - Cifrado

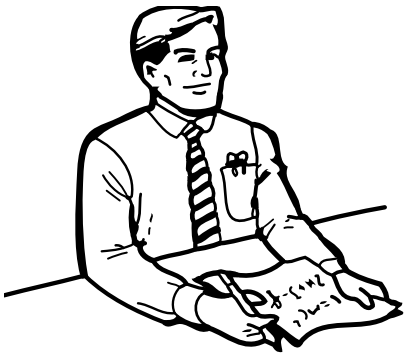




## Algoritmos simétricos de Bloque

### Ejemplos

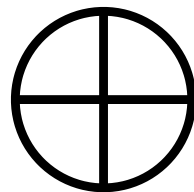
- DES-LUCIFER (1976, Data Encryption Standard)
- 3DES (1998, Triple Data Encryption Standard, NIST)
- **AES-Rijndael (2001, Advanced Encryption Standard, NIST)**
- **Serpent (1998)**
- **Twofish**
- **RC6 (1998, Rivest Cipher 6)**
- **MARS (1998, IBM)**
- GOST (1994, Magma URSS)
- RC5 (1994, Rivest Cipher 5)
- IDEA (1991, International Data Encryption Algorithm)
- Blowfish



## Referencia matemática - XOR

### Propiedades

- Es conmutativa: Es decir que  $A \text{ xor } B = B \text{ xor } A$
- Asociativa:  $(A \text{ xor } B) \text{ xor } C = A \text{ xor } (B \text{ xor } C)$
- Autoinversa:  $(A \text{ xor } B) \text{ xor } B = A$



<u>A</u>	<u>B</u>	<u>XOR</u>
0	0	0
0	1	1
1	0	1
1	1	0



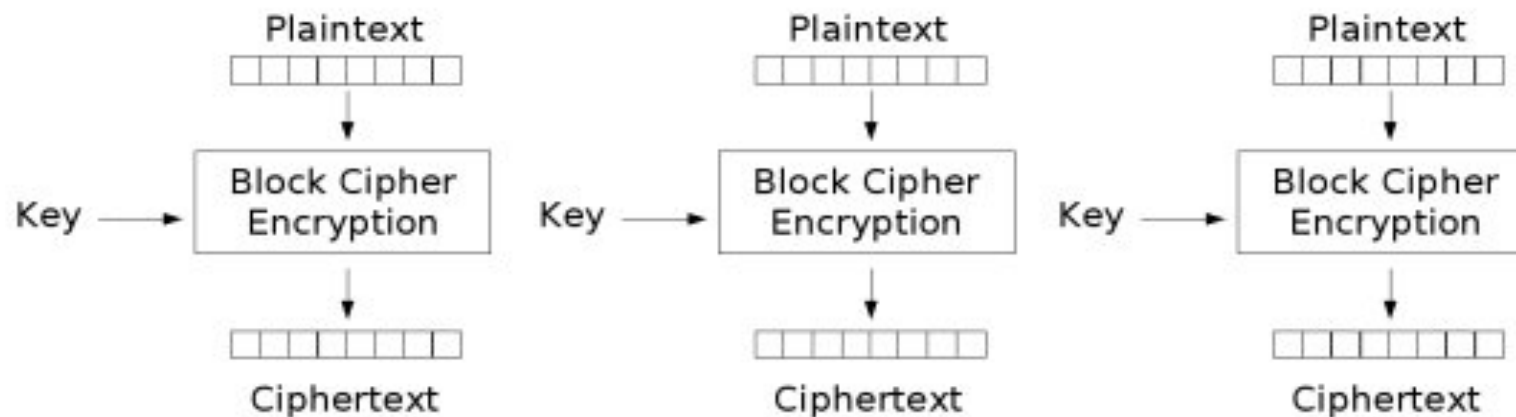
An XOR might keep your kid sister from reading your files, but it won't stop a cryptanalyst for more than a few minutes. -[Bruce Schneier](#)-





## Modos de cifrado de bloques

**ECB:** Electronic codebook, en este método el mensaje se fracciona en partes y cada una es cifrada de manera independiente.



Electronic Codebook (ECB) mode encryption



## Padding o Esquema de Relleno

Los algoritmos simétricos de bloque requieren que el mensaje sea fragmentado en partes de una longitud fija; esto plantea el problema de que el mensaje en su totalidad o su ultimo bloque podría ser de longitud menor a la requerida, en este caso se recurre a los métodos de padding para resolver el problema.

Algunos de ellos son:

- Bit padding (RFC1321, ISO/IEC 9797-1)
- ISO/IEC 7816-4
- **PKCS#7 (RFC2315)**
- ISO 10126
- ANSI X.923



## Padding o Esquema de Relleno

**Bit padding:** operando a nivel de bits adiciona **1** y posteriormente **N** cantidad de **0** hasta completar el tamaño requerido.

01011100 10110011 0110 <b>1000</b> 00000000	4 bytes
---	---------

**ISO/IEC 7816-4:** es idéntico a **bit padding** pero operando a nivel de bytes, agregando el valor 80 y posterior mente **N** cantidad de 0 hasta completar.

FF FF FF 80 00 00 00 00	8 bytes
-------------------------	---------

**PKCS#7:** operando a nivel de bytes **N** cantidad de bytes idénticos cuyo valor es la cantidad de bytes agregados.

FF FF 06 06 06 06 06 06	8 bytes
-------------------------	---------



## Padding o Esquema de Relleno

**ISO 10126:** operando a nivel de bytes agrega **N** cantidad de bytes aleatorios hasta el ante ultimo, luego ingresa el ultimo byte que contendrá la cantidad de bytes agregados.

FF FF FF <b>C2 A7 2E 14 05</b>
--------------------------------

8 bytes
---------

**ANSI X.923:** operando a nivel de bytes adiciona **N** cantidad de **00** hasta el ante ultimo byte, luego ingresa el ultimo que contendrá la cantidad de bytes agregados.

FF FF FF FF FF <b>00 00 03</b>
--------------------------------

8 bytes
---------

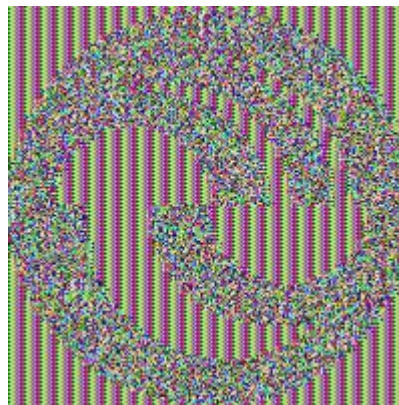


## Ataques por marca de agua

A continuación se observa el resultado de cifrar con AES128 una imagen utilizando diferentes modos de cifrado de bloque.



Original



ECB



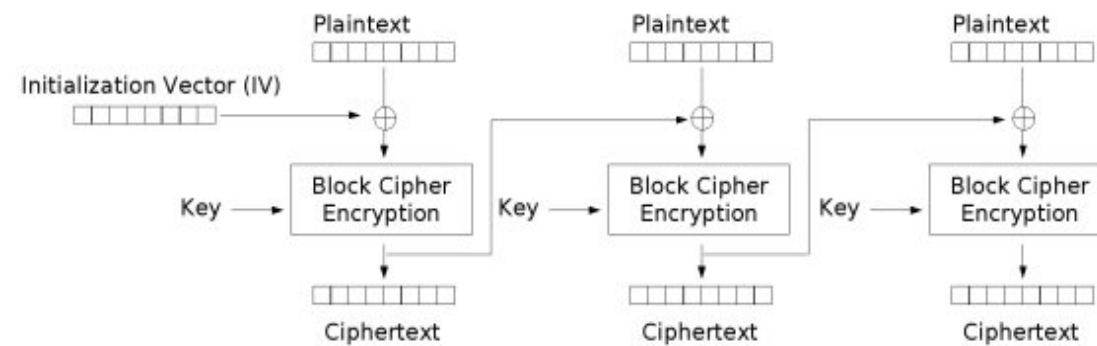
CBC

```
head -n 4 unlam.ppm > header.txt
tail -n +5 unlam.ppm > body.bin
openssl enc -aes-128-ecb -nosalt -pass pass:"scaw" -in body.bin -out body.ecb.bin
cat header.txt body.ecb.bin > unlam.ecb.ppm
head -n 4 unlam.ppm > header.txt
tail -n +5 unlam.ppm > body.bin
openssl enc -aes-128-cbc -nosalt -pass pass:"scaw" -in body.bin -out body.cbc.bin
cat header.txt body.cbc.bin > unlam.cbc.ppm
```

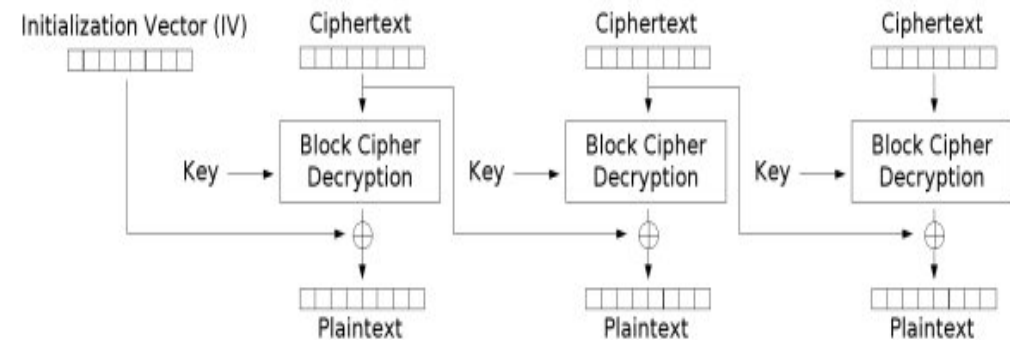


## Modos de cifrado de bloques

**CBC:** Cipher block chaining, en este método el mensaje se fracciona en partes y se realiza un XOR con el bloque previo antes de cifrar cada parte.



Cipher Block Chaining (CBC) mode encryption

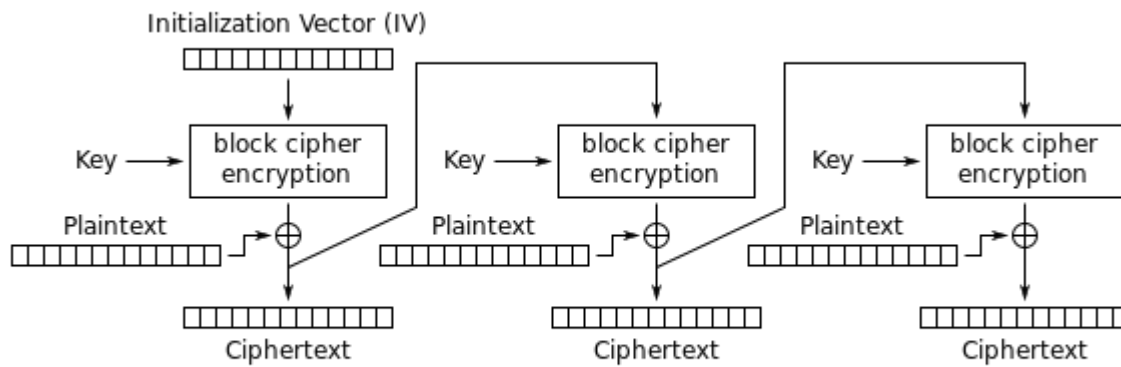


Cipher Block Chaining (CBC) mode decryption

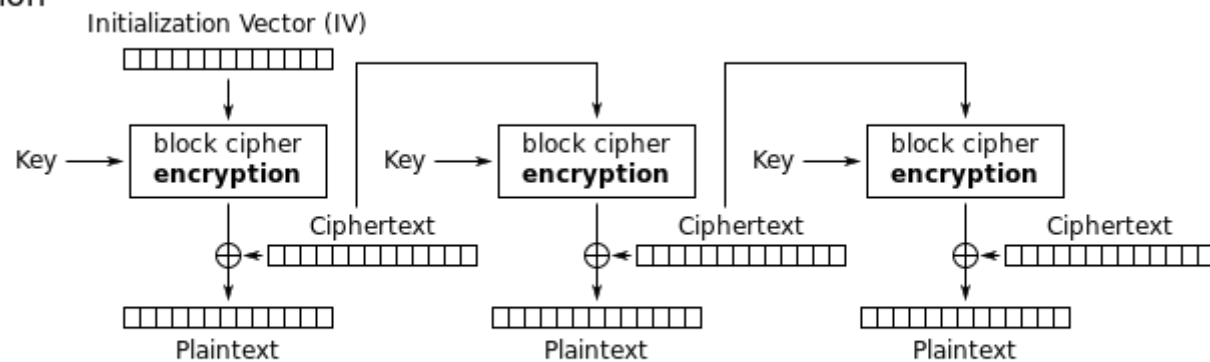


## Modos de cifrado de bloques

**CFB:** Cipher Feedback, en este método el mensaje se fracciona en partes, se cifra un vector de inicialización y al resultado se le realiza un XOR con el bloque del mensaje. Los bloques posteriores utilizan como entrada el texto cifrado para reemplazar al vector de inicialización.



Cipher Feedback (CFB) mode encryption



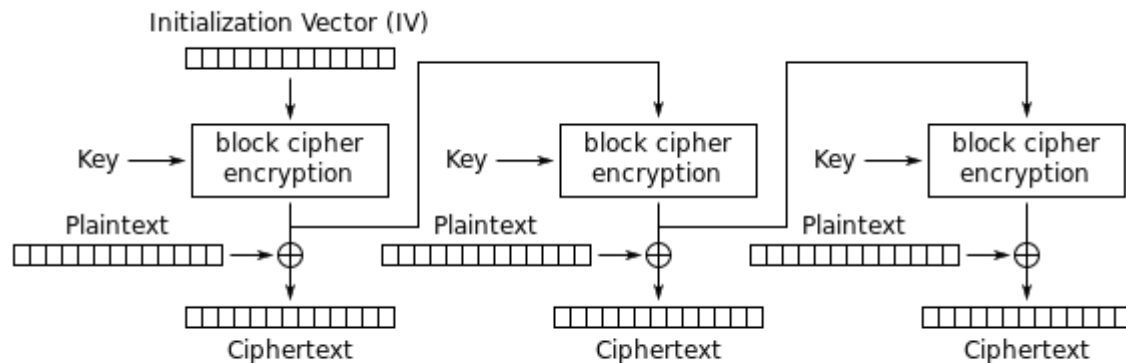
Cipher Feedback (CFB) mode decryption



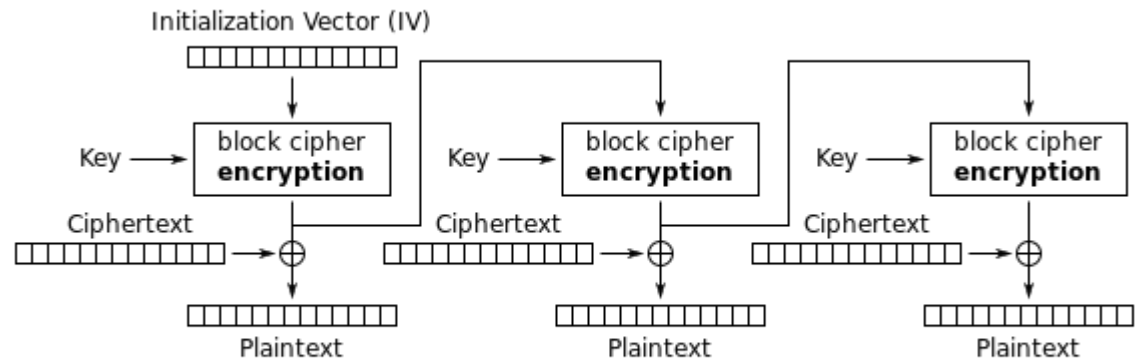


## Modos de cifrado de bloques

**OFB:** Output Feedback, este método opera de manera similar a CFB con la diferencia que el bloque a ser utilizado como entrada del siguiente proceso es tomado de la salida del algoritmo justo antes de realizar el XOR.



Output Feedback (OFB) mode encryption



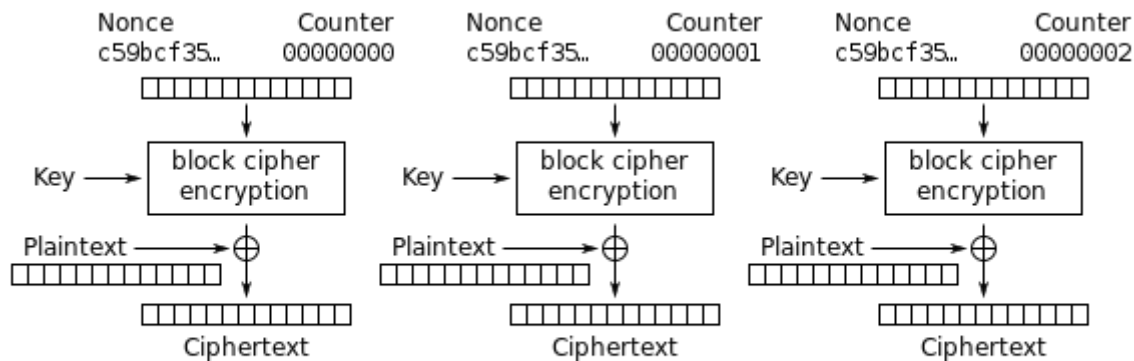
Output Feedback (OFB) mode decryption



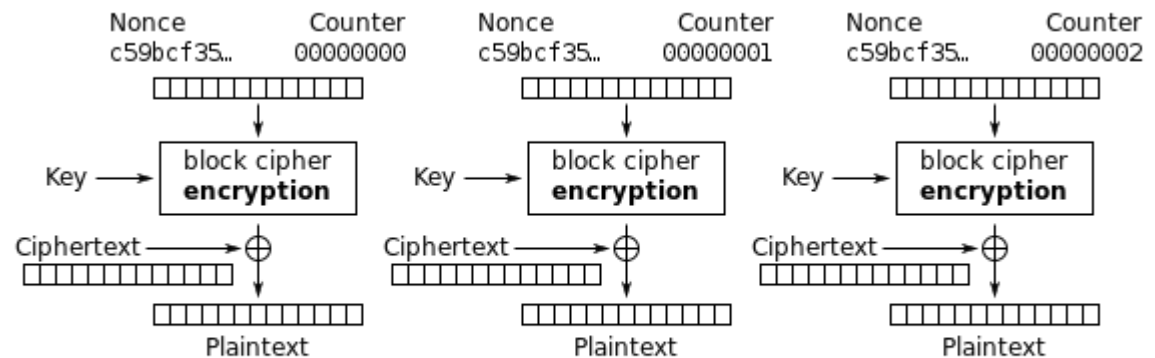


## Modos de cifrado de bloques

**CTR:** Modo de Counter o contador, en este modo de operación (*al igual que en OFB*) se utiliza un “nonce” equivalente al IV anterior, que es alterado por un contador incrementado en cada bloque de datos, para obtener un valor que luego sera operado con el bloque de datos usando XOR .



Counter (CTR) mode encryption



Counter (CTR) mode decryption



## **Otros modos de cifrado de bloques**

Existen diversos modos y algunos de ellos incorporan autenticación a la confidencialidad.

- PCBC
- CCM
- CWC
- EAX
- GCM (Galois Counter Mode)
- PCFB
- XCBC



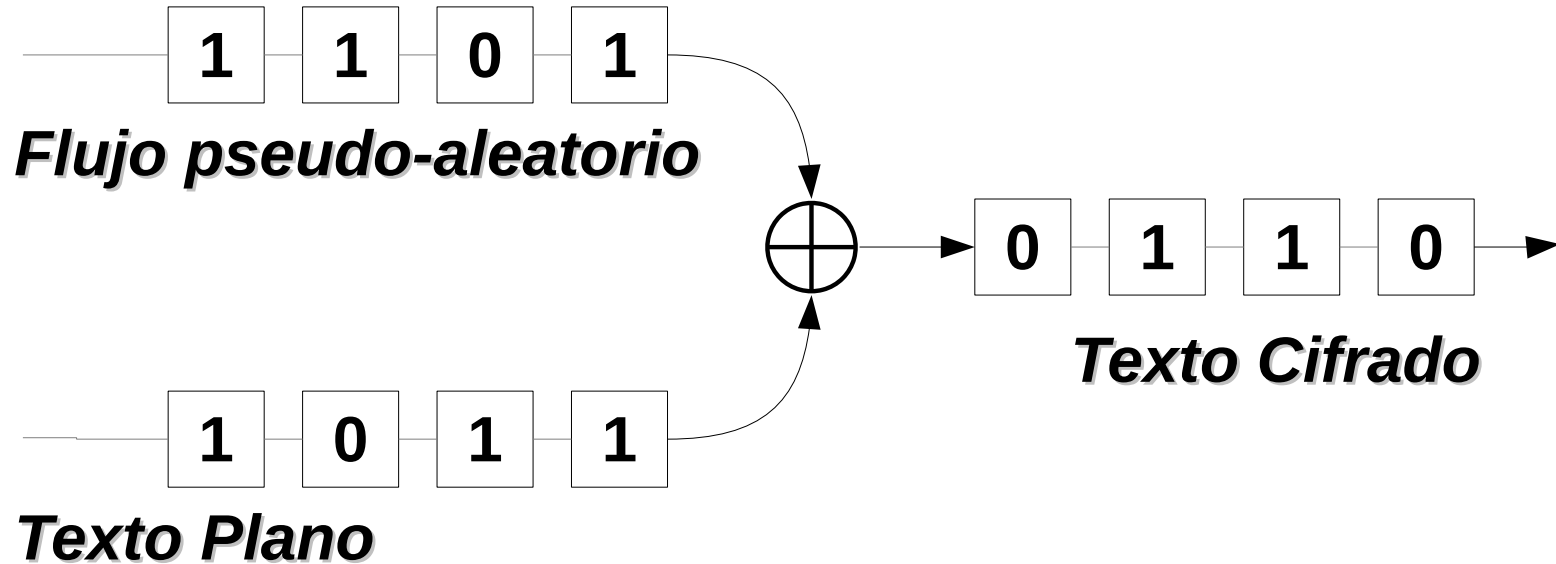
## **Cifrado de flujo**

En 1917, J. Mauborgne y G. Vernam inventaron un criptosistema perfecto según el criterio de Shannon. Dicho sistema consistía en emplear una secuencia aleatoria de igual longitud que el mensaje, que se usaría una única vez (One Time Pad), combinándola mediante alguna función simple y reversible como el or exclusivo(XOR) con el texto en claro carácter a carácter. Este método presenta el grave inconveniente de que la clave es tan larga como el propio mensaje, y si disponemos de un canal seguro para enviar la clave, ¿por que no emplearlo para transmitir el mensaje directamente?



## Cifrado de flujo

Se utiliza una función generadora de bits pseudo-aleatorios a fin de obtener un flujo de bits que pueda ser procesado con los bits del mensaje mediante una operación básica (XOR).





## Cifrado de flujo

**Secuencias criptográficamente aleatorias:** Para que una secuencia pseudoaleatoria sea criptográficamente aleatoria, ha de cumplir la propiedad de ser impredecible. Esto quiere decir que debe ser computacionalmente intratable el problema de averiguar el siguiente número de la secuencia, teniendo total conocimiento acerca de todos los números anteriores y del algoritmo de generación empleado.

Una **función generadora de bits pseudo aleatoria** es la que permite obtener secuencias criptográficamente aleatorias.



## **Cifrado de flujo**

Estos son algunos de los actuales algoritmos de cifrado de flujo

- RC4
- Salsa20\*
- ChaCha
- Trivium\*
- A5/1, A5/2
- Chameleon
- FISH
- Helix
- Grain\*
- ISAAC
- MUGI
- Panama
- Phelix
- Pike
- SEAL
- SOBER/SOBER-128
- WAKE
- Rabbit\*

\*(eSTREAM portfolio, de EU eCRYPT)



## Funciones de HASH

Se define como una función o método no reversible para generar un valor que represente de manera casi unívoca a un dato.

### Principales usos

- Soporte para criptografía asimétrica
- Tablas de Hash
- Verificación de integridad



## Funciones de HASH

### Propiedades

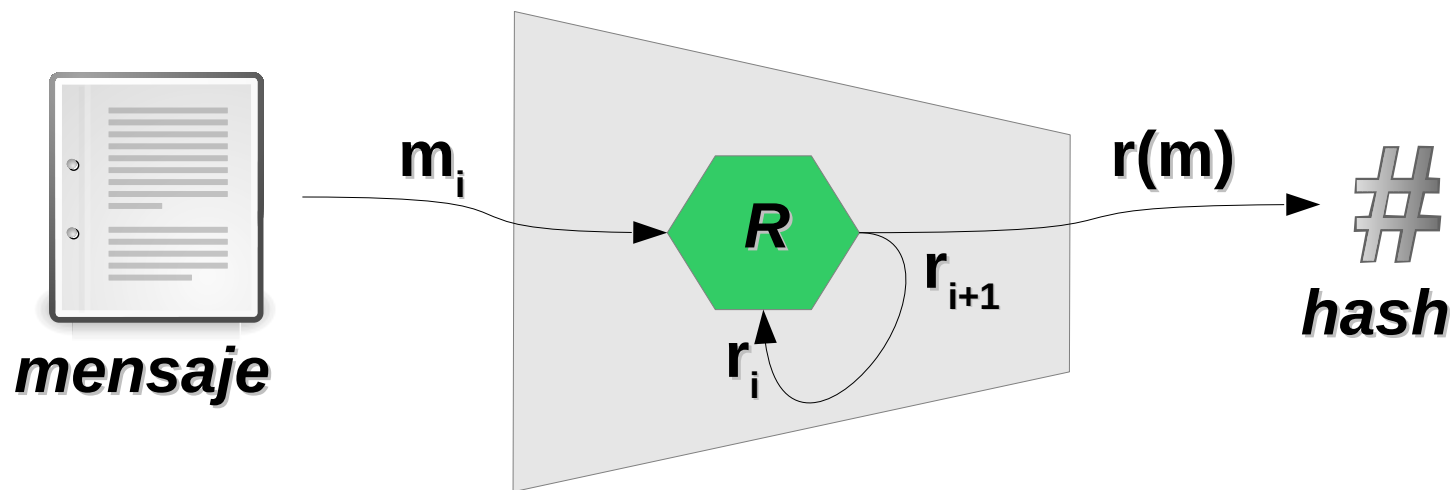
- $r(m)$  es de longitud fija, independientemente de la longitud de  $m$ .
- Dado  $m$ , es fácil calcular  $r(m)$ .
- Dado  $r(m)$ , es computacionalmente intratable recuperar  $m$ .
- Dado  $m$ , es computacionalmente intratable obtener un  $m'$  tal que  $r(m) = r(m')$ .





## Funciones de HASH - MDC

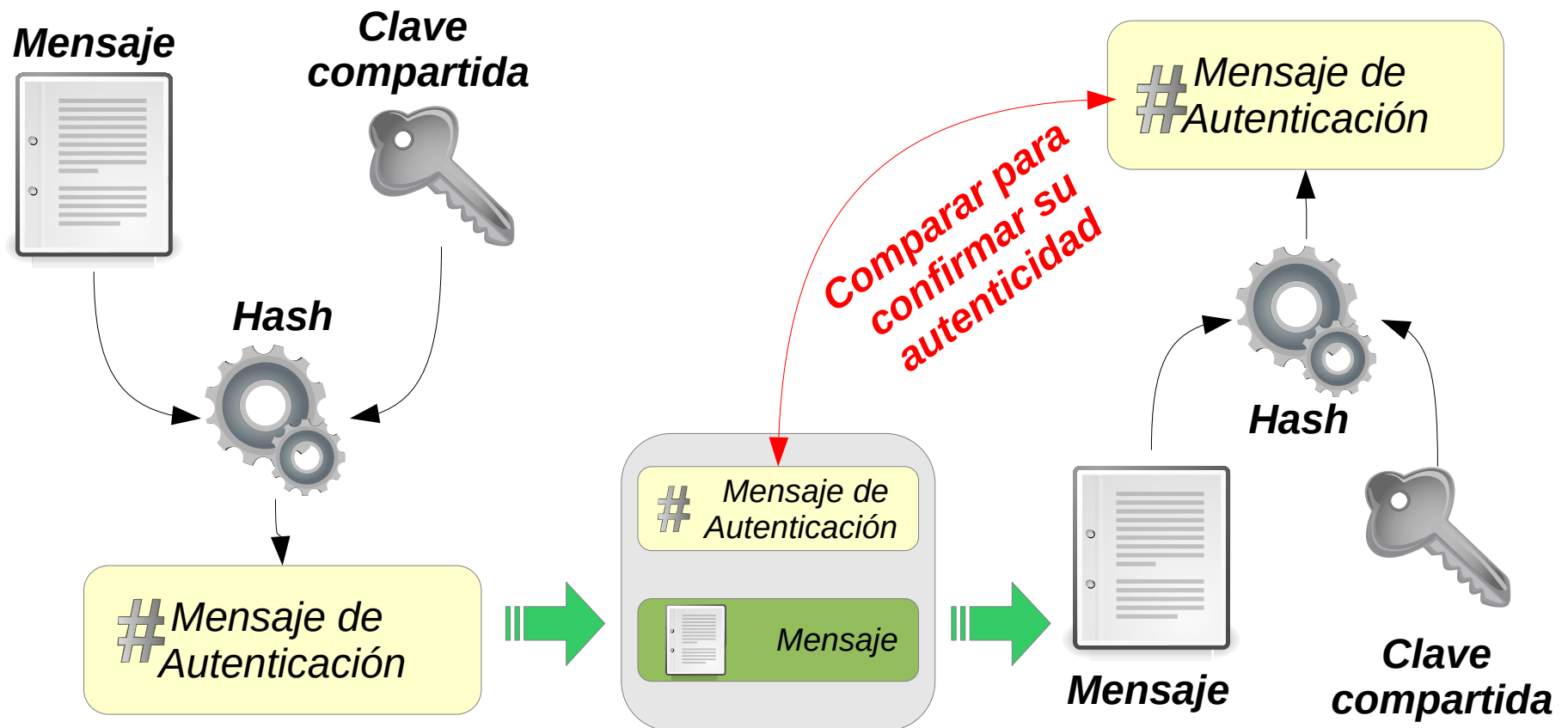
Estas funciones dan como resultado bloques de longitud fija **a** a partir de bloques de longitud fija **b**, con **a** < **b**. Estas funciones se encadenan de forma iterativa, haciendo que la entrada en el paso **i** sea la función del **i**-simo bloque del mensaje (**m<sub>i</sub>**) y de la salida del paso previo, **i-1**. Se considera una buena practica incluir en el mensaje **m** la longitud y características del mensaje.





## Funciones de HASH - MAC

Message Authentication Code: Adiciona criptografía al proceso de hash para aumentar la seguridad del mismo.





## Funciones de HASH - MAC

### Tipos de implementaciones

- **Basados en cifrados por bloques:** Consisten en cifrar el mensaje empleando un algoritmo por bloques en modo de operación CBC. El valor del MAC será entonces el resultado de cifrar el último bloque del mensaje.
- **HMAC:** Se basan en el uso de cualquier función MDC existente, aplicada sobre una versión del mensaje a la que se ha añadido un conjunto de bits, calculados a partir de la clave que se quiere emplear.
- **Basados en generadores de secuencia:** Empleando un generador de secuencia pseudoaleatorio el mensaje se parte en dos subcadenas — correspondientes al mensaje combinado con la secuencia y a la propia secuencia—, cada una de las cuales alimenta un Registro de Desplazamiento Retroalimentado. El valor del MAC se obtiene a partir de los estados finales de ambos registros.



## Funciones HASH

**MD4** (*Message Digest, Mensajes Digitales*). Fue Inventado por Ron Rivest de la Incorporación de Seguridad RSA (RSA Security, Inc.). Produce un valor hash de 128-bits. Se realiza una manipulación de bits para obtener el valor hash, obteniéndolo de forma rápida, provocando que sea más riesgoso en un ataque. Se considera un estándar de Internet(RFC-1320) [[STA98](#)].

**MD5** Extensión a MD4. Produce como salida de 128-bits. La obtención del valor hash es lento pero considerado más seguro. Está especificado como un estándar de Internet(RFC-1321).

**SHA-1** (*Secure Hash Algorithm, Algoritmo Hash Seguro*). Diseñado por NIST (National Institute of Standards and Technology), produce un valor hash de 160-bits. También está considerado como un estándar (FIPS PUB 180-1).

**SHA-2** (*Secure Hash Algorithm, Algoritmo Hash Seguro*). Diseñado por la NSA (National Security Agency) y publicados por el NIST en el 2001 (FIPS PUB 180-2), es un conjunto de algoritmos comprendidos por **SHA-224, SHA-256, SHA-384 y SHA-512**.



## Funciones de HASH

**SHA-3** (*Secure Hash Algorithm, Algoritmo Hash Seguro*). Llamado a concurso abierto organizado por el NIST (National Institute of Standards and Technology), adjudicado a **Keccak** durante el 2012. Sus finalistas fueron:

**Keccak**: Por Guido Bertoni, Joan Daemen, Michaël Peeters and Gilles Van Assche. operación en 224,256,384 o 512 bits.

**Blake**: Por Jean-Philippe Aumasson, Luca Henzen, Willi Meier, Raphael C.-W. Phan, operación en 224,256,384 o 512 bits.

**Grøstl**: Por Praveen Gauravaram, Lars Knudsen, Krystian Matusiewicz, Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. Opera en 256 y 512 bits, utiliza las S-Box de AES.

**JH**: Por Hongjun Wu, operación en 224,256,384 o 512 bits.

**Skein**: Por Bruce Schneier, Niels Ferguson, operación en 256 o 512 bits.

**RIPEMD-160**. Diseñada por Hans Dobbertin, Antoon Bosselaers y Bart Preneel para el proyecto RIPE (Race Integrity Primitives Evaluation, Carrera de Evaluación de Primitivas de Integridad 1988-1992). Genera una salida de 160 bits



## Funciones de Derivación de Claves

Conocidas como **KDF (Key Derivation Function)** son funciones no reversibles que tienen el objetivo de generar una o mas claves en base a un valor maestro o clave inicial secretos, mas un conjunto de parámetros que configuran el comportamiento de la función afectando el resultado.

Normalmente se basan en funciones pseudo-aleatorias, funciones de hash con múltiples iteraciones y procesos de inclusión de 'Salt'. Se han originado para evitar a ataques de diccionario y tablas de arcoiris.

- **PBKDF2** - (2000) Password-Based Key Derivation Function 2 – RFC2898 y (PKCS#5, NIST SP 800-132)
- **bcrypt** – (1999) Basado en el algoritmo de Blowfish
- **scrypt** – (2012) Basado en PBKDF2\_HMAC\_SHA256



## Referencia Matemática - Modular



En matemática, la aritmética modular es un sistema aritmético para clases de equivalencia de números enteros llamadas clases de congruencia.

Relación de congruencia

La aritmética modular puede ser construida matemáticamente mediante la relación de congruencia entre enteros, que es compatible con las operaciones en el anillo de enteros: **suma, resta, y multiplicación**. **a** y **b** se encuentran en la misma "clase de congruencia" módulo **n**, si ambos dejan el mismo resto si los dividimos por **n**, o, equivalentemente, si **a – b** es un múltiplo de **n**.

Esta relación se puede expresar cómodamente utilizando la notación de Gauss:

$$a \equiv b \pmod{n}$$

Así se tiene por ejemplo

$$63 \equiv 83 \pmod{10}$$

ya que ambos, **63** y **83** dejan el mismo **resto (3)** al dividir por **10**, o, equivalentemente, **63 – 83** es un múltiplo de **10**.





## Algoritmos asimétricos – Primos Relativos



Sean  $a, b \in \mathbb{Z}$ , se dice que **son primos relativos (o coprimos) “a” y “b”** si **no tienen ningún factor primo en común**, es decir, si no tienen otro divisor común más que 1 ó -1, o cumplen que **el  $\text{mcd}(a, b) = 1$** .

El algoritmo de Euclides extendido permite, además de encontrar un máximo común divisor de dos números enteros  $a$  y  $b$ , expresarlo como la mínima combinación lineal de esos números, es decir, encontrar números enteros  $s$  y  $t$  tales que  **$\text{mcd}(a, b) = as + bt$** .





## Algoritmos asimétricos

Introducido por Whitfield Diffie y Martin Hellman a mediados de los años 70.

El sistema de cifrado de clave pública usa un par de claves para el envío de mensajes. Las dos claves pertenecen a la misma persona a la que se ha enviado el mensaje. Una clave es *pública* y se puede entregar a cualquier persona. La otra clave es *privada* y el propietario debe guardarla para que nadie tenga acceso a ella. El remitente usa la clave pública del destinatario para cifrar el mensaje, y una vez cifrado, sólo la clave privada del destinatario podrá descifrar este mensaje.



## Algoritmos asimétricos - Diffie-Hellman

Este algoritmo nos permite compartir un mensaje cifrado entre dos actores que no han tenido contacto previo; por esta razón suele utilizarse para acordar una clave de cifrado a través de un canal inseguro y sin autenticación.

Sean **A** y **B** los interlocutores en cuestión. En primer lugar, se calcula un número primo **p** y un generador **z** de  $\mathbb{Z}^*$ , con  $2 \leq z \leq p - 2$ . Esta información es pública y **p** conocida por ambos. El algoritmo queda como sigue:

1. **A** escoge un número aleatorio **x**, comprendido entre **1** y **p - 2** y envía a **B** el valor

$$i = z^x \pmod{p}$$

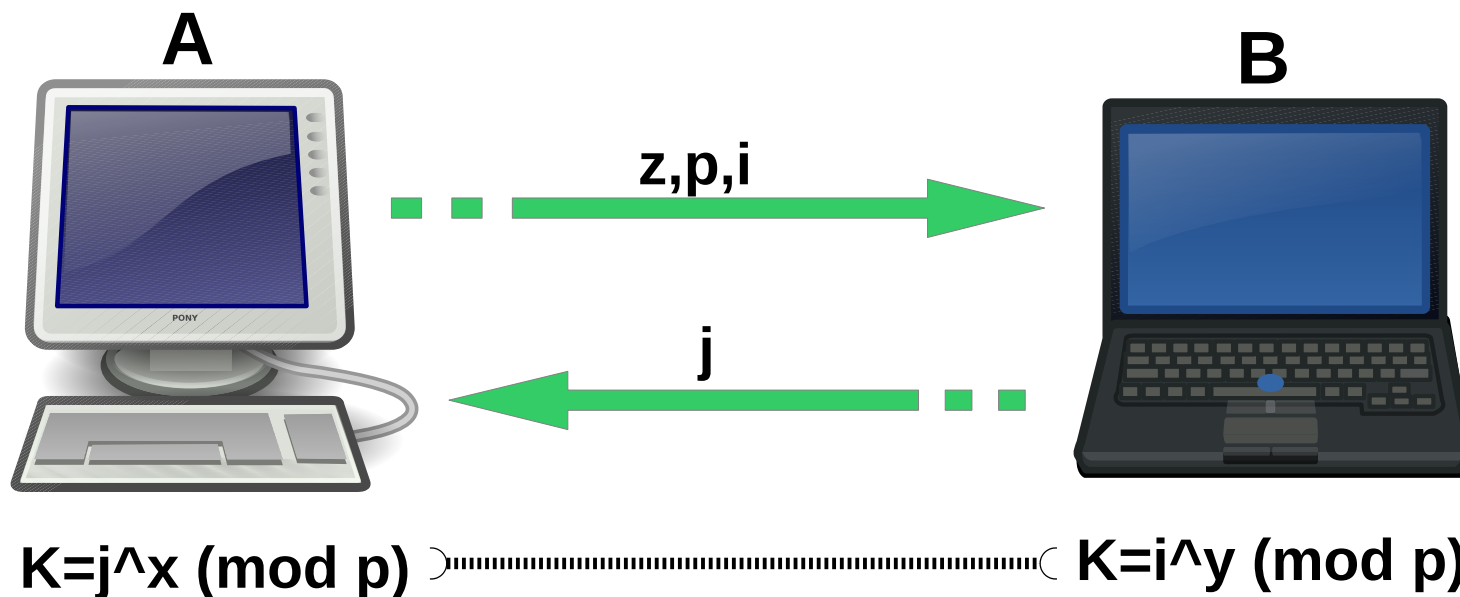
2. **B** escoge un número aleatorio **y**, análogamente al paso anterior, y envía a **A** el valor

$$j = z^y \pmod{p}$$



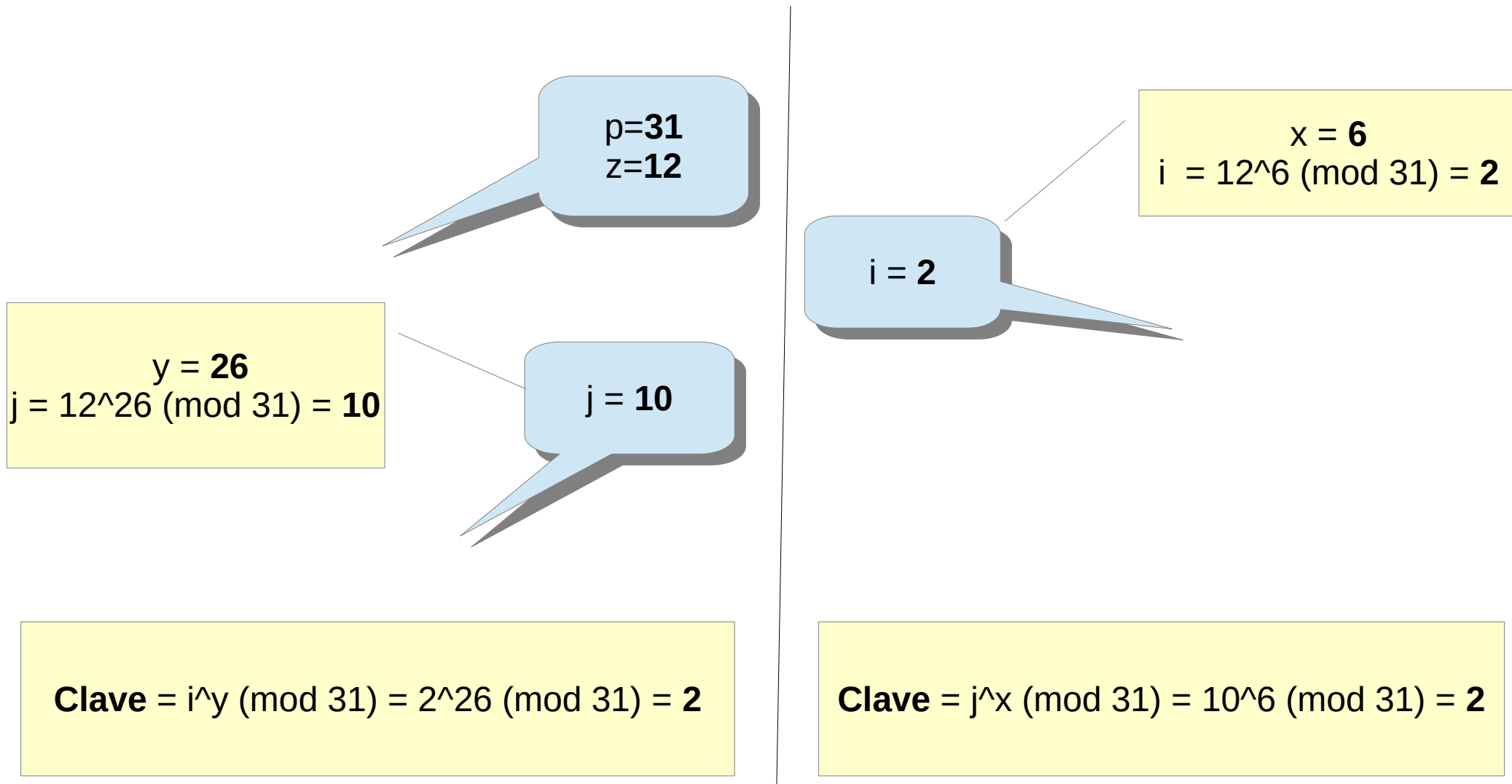
## Algoritmos asimétricos - Diffie-Hellman

3. **B** recoge **i** y calcula  $K = i^y \pmod{p} = (z^x \pmod{p})^y \pmod{p}$ .
4. **A** recoge **j** y calcula  $K = j^x \pmod{p} = (z^y \pmod{p})^x \pmod{p}$ .
5. Conclusión  $K = z^{xy} \pmod{p}$





## Algoritmos asimétricos - Diffie-Hellman





## Algoritmos asimétricos – Diffie-Hellman – Múltiples actores

Las partes (Alicia, Brenda y Carlos) disponen de los valores compartidos para los parámetros de algoritmo  $p$  y  $g$ . Cada actor definirá su valor privado al que llamaremos con el nombre  $a, b, c$  respectivamente.

- 1) Alicia calcula  $g^a \bmod p$  y lo envía a Brenda.
- 2) Brenda calcula  $(g^a)^b \bmod p = g^{ab} \bmod p$  y lo envía a Carlos.
- 3) Carlos calcula  $(g^{ab})^c \bmod p = g^{abc} \bmod p$  y la usa como su clave secreta.
- 4) Brenda calcula  $g^b \bmod p$  y lo envía a Carlos.
- 5) Carlos calcula  $(g^b)^c \bmod p = g^{bc} \bmod p$  y lo envía a Alicia.
- 6) Alicia calcula  $(g^{bc})^a \bmod p = g^{bca} \bmod p = g^{abc} \bmod p$  y lo usa como su clave secreta.
- 7) Carlos calcula  $g^c \bmod p$  y lo envía a Alicia.
- 8) Alicia calcula  $(g^c)^a \bmod p = g^{ca} \bmod p$  y lo envía a Brenda.
- 9) Brenda calcula  $(g^{ca})^b \bmod p = g^{cab} \bmod p = g^{abc} \bmod p$  y lo usa como su clave secreta.



## **Algoritmos asimétricos - Cifrado**

### **Ventajas**

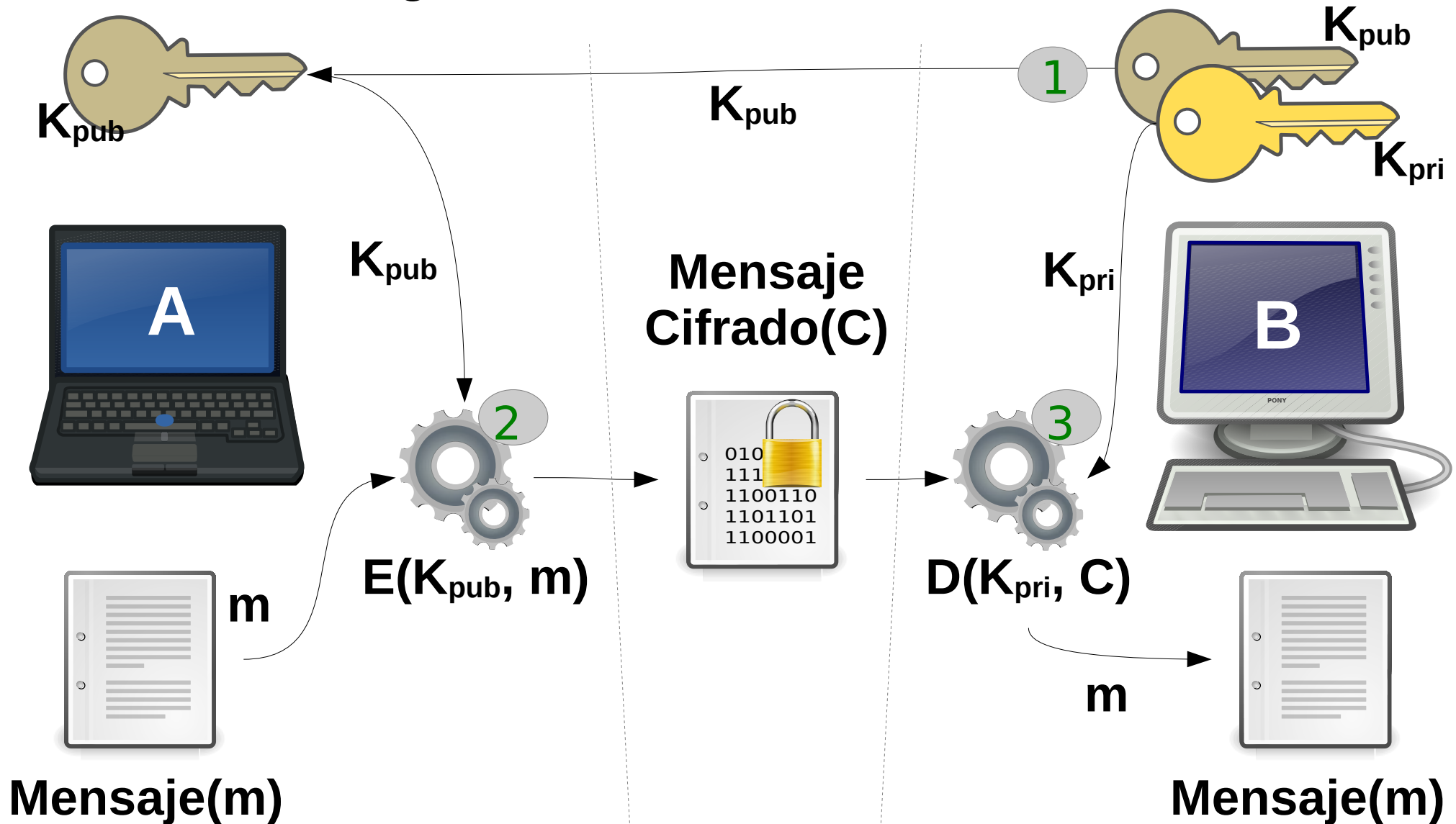
- No requiere confidencialidad en la distribución de clave
- La misma clave puede ser utilizada por múltiples actores en la comunicación
- Permite autenticar mensajes

### **Desventajas**

- Velocidad de cifrado/descifrado
- Longitud de mensaje limitado
- Tamaño del mensaje cifrado es mayor
- Se requieren claves de gran extensión



## Algoritmos asimétricos - Cifrado





## Algoritmos asimétricos

### Ejemplos

- Diffie-Hellman
- RSA
- ElGamal
- DSA
- Criptografía de curva elíptica (ECDH, ECDSA...)





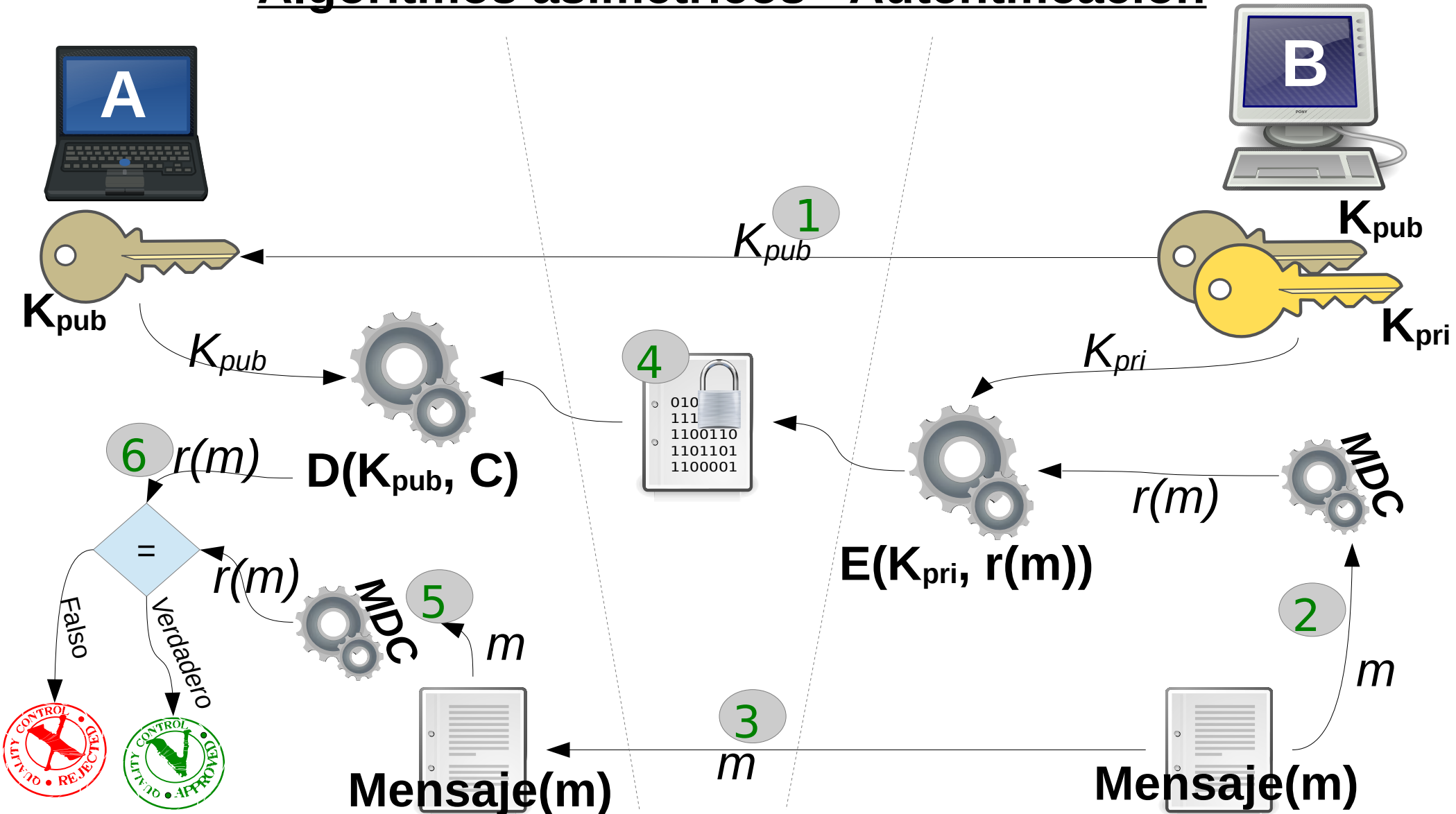
## **Algoritmos asimétricos - Autenticación**

### Autenticación de mensaje (Firma)

Algunos algoritmos asimétricos permiten que se pueda autenticar un mensaje para garantizar su integridad. en este caso la clave que se emplea para cifrar es la clave privada, justo al revés que para la simple codificación de mensajes.



## Algoritmos asimétricos - Autenticación





## **Algoritmos asimétricos - RSA**

- Su nombre deriva de Ronald Rivest, Adi Shamir y Leonard Adleman
- Estuvo bajo patente de los Laboratorios RSA hasta el 20 de septiembre de 2000
- Se basa en la dificultad para factorizar grandes números



## Algoritmos asimétricos - RSA

Se eligen aleatoriamente dos números primos grandes, **p** y **q**. Después se calcula el producto **n = pq**. Escogeremos ahora un número **e** primo relativo con **(p - 1)(q - 1)**. **(e, n)** será la clave pública. Nótese que **e** debe tener **inversa modulo (p - 1)(q - 1)**, por lo que existirá un número **d** tal que

$$\begin{aligned}de &\equiv 1 \pmod{(p - 1)(q - 1)} \\de &\equiv 1 \pmod{\text{mcm}(p - 1, q - 1)}\end{aligned}$$

es decir, que **d** es la inversa de **e** modulo **(p-1)(q -1)**. **(d, n)** será la clave privada. Esta inversa puede calcularse fácilmente empleando el Algoritmo Extendido de Euclides.

Nótese que si desconocemos los factores de **n**, este cálculo resulta prácticamente imposible.



## Algoritmos asimétricos – RSA

Elección de números primos

**p=3**

**q=11**

**n=3 \* 11 = 33**

**phi(n) = (3-1)\*(11-1) = 2\*10 = 20**

Elección del componente publico

**e = 7** (Primo relativo entre 1 y phi(n))

Buscar de 1 a phi(n) el componente privado, para el valor 3 el calculo es:

**e \* d=1 mod(phi(n))**

**7 \* d = 1 mod (20)**

**7 \* 3 = 1 mod (20)**

**21 mod 20 = 1**

**d = 3**

**Clave Pública: (e, n) o (7, 33)**

**Clave Privada: (d, n) o (3, 33)**



## **Algoritmos asimétricos - RSA**

La operación de cifrado se lleva a cabo según la expresión:

$$c = m^e \pmod{n}$$

mientras que el descifrado se hará de la siguiente forma:

$$m = c^d \pmod{n}$$



## **Algoritmos asimétricos - RSA**

Ejemplos de operación sobre el mensaje con valor **2**

$$\begin{aligned}c &= m^e \pmod{n} \\c &= 2^7 \pmod{33} \\c &= 128 \pmod{33} = 29\end{aligned}$$

Descifrado para el valor **29**

$$\begin{aligned}m &= c^d \pmod{n} \\m &= 29^3 \pmod{33} \\m &= 24389 \pmod{33} = 2\end{aligned}$$



## Algoritmos asimétricos - RSA

### Firmas Digitales de RSA

Para firmar un mensaje **h** con la clave privada se procede de la siguiente manera:

$$s = h^d \pmod{n}$$

La firma generada se verifica obteniendo el valor de origen con la clave pública mediante el siguiente método:

$$h = s^e \pmod{n}$$





## Algoritmos asimétricos - ElGamal

Se basa en el problema de los logaritmos discretos

Se conoce como logaritmo discreto de  $x$  en base  $a$  módulo  $n$  a resolver la ecuación  $x = a^y \bmod n$  donde  $x, n$  y  $a$  son constantes e  $y$  es la incógnita. A partir de ahora notamos esta situación como:

$$y = \log_{\text{disc}_a}(x)$$

El hecho de aplicar aritmética modular hace el problema de hallar  $y$  irresoluble en un tiempo razonable



## Algoritmos asimétricos - ElGamal

Para generar un par de claves, se escoge un número primo **n** y dos números aleatorios **p** y **x** menores que **n**. Se calcula entonces

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

La clave pública es **(p, y, n)**, mientras que la clave privada es **x**.



## Algoritmos asimétricos - ElGamal

Numero primo seleccionado al azar

**$n = 17$**

Números complementarios menores que  $n$ , elegidos al azar

**$p=3$**

**$x=6$**

Calculo del valor para el componente publico

$y = p^x \bmod(n)$

$y = 3^6 \bmod(17)$

**$y=15$**

**Clave Pública:  $(p, y, n)$  o  $(3, 15, 17)$**

**Clave Privada:  $(p, x, n)$  o  $(3, 6, 17)$**



## Algoritmos asimétricos - ElGamal

### Cifrado de ElGamal

Para cifrar el mensaje **m** se escoge primero un número aleatorio **k** primo relativo con **(n - 1)**, que también será mantenido en secreto. Calculamos entonces las siguientes expresiones

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

El par **(a, b)** es el texto cifrado, de doble longitud que el texto original. Para decodificar se calcula

$$m = b \cdot a^{-x} \pmod{n}$$



## Algoritmos asimétricos - ElGamal

### Ejemplo de cifrado de ElGamal

Para cifrar el mensaje **m** con el valor **9** y un **k** aleatorio con valor **5** se procede de la siguiente manera:

$$\begin{aligned}a &= p^k \bmod(n) \\a &= 3^5 \bmod(17) = 5 \\b &= y^k m \bmod(n) \\b &= 15^5 * 9 \bmod(17) = 1\end{aligned}$$

El par **(5, 1)** es el texto cifrado, que podrá descifrarse de la siguiente forma

$$\begin{aligned}m &= b \cdot a^{-x} \bmod(n) \\m &= 1 * 5^{-6} \bmod(17) = 9\end{aligned}$$



## Algoritmos asimétricos - ElGamal

### Firmas de ElGamal

Para firmar un mensaje **m** basta con escoger un número **k** aleatorio, que sea primo relativo con **n - 1**, y calcular

$$\begin{aligned}a &= p^k \pmod{n} \\ b &= (m - xa)k^{-1} \pmod{(n - 1)}\end{aligned}$$

La firma la constituye el par **(a, b)**. En cuanto al valor **k**, debe mantenerse en secreto y ser diferente cada vez. La firma se verifica comprobando que

$$y^a a^b = p^m \pmod{n}$$



## **Algoritmos asimétricos - DSA**

El algoritmo DSA (Digital Signature Algorithm) es una parte el estándar de firma digital DSS (Digital Signature Standard), definido en el FIPS-186. Este algoritmo, propuesto por el NIST, data de 1991, es una variante del método asimétrico de ElGamal.



## Algoritmos asimétricos - DSA

### Creación del par clave publica-clave privada

1. Seleccionar un número primo  $q$  tal que  $2^{159} < q < 2^{160}$ .
2. Escoger  $t$  tal que  $0 \leq t \leq 8$ , y seleccionar un número primo  $p$  tal que  $2^{511+64t} < p < 2^{512+64t}$ , y que además  $q$  sea divisor de  $(p - 1)$ .
3. Seleccionar un elemento  $g \in \mathbb{Z}_p^*$  y calcular  $\alpha = g^{(p-1)/q} \bmod p$ .
4. Si  $\alpha = 1$  volver al paso 3.
5. Seleccionar un número entero aleatorio  $a$ , tal que  $1 \leq a \leq q - 1$
6. Calcular  $y = \alpha^a \bmod p$ .
7. La clave pública es  $(p, q, \alpha, y)$ . La clave privada es  $a$ .





## Algoritmos asimétricos - DSA

### Generación de la firma

Siendo **h** la salida de una función MDC sobre el mensaje **m**, la generación de una firma se hace mediante el siguiente algoritmo:

1. Seleccionar un número aleatorio **k** tal que  $0 < k < q$ .
2. Calcular  $r = (\alpha^k \bmod p) \bmod q$ .
3. Calcular  $k^{-1} \bmod q$ .
4. Calculate  $s = k^{-1} (h + ar) \bmod q$ .
5. La firma del mensaje **m** es el par **(r, s)**.



## Algoritmos asimétricos - DSA

### Verificación de la firma

El destinatario efectuar las siguientes operaciones, suponiendo que conoce la clave pública  $(p, q, \alpha, y)$ , para verificar la autenticidad de la firma:

1. Verificar que  $0 < r < q$  y  $0 < s < q$ . En caso contrario, rechazar la firma.
2. Calcular el valor de  $h$  a partir de  $m$ .
3. Calcular  $\omega = s^{-1} \bmod q$ .
4. Calcular  $u1 = \omega \cdot h \bmod q$  y  $u2 = \omega \cdot r \bmod q$ .
5. Calcular  $v = (\alpha^{u1} y^{u2} \bmod p) \bmod q$ .
6. Aceptar la firma si y solo si  $v = r$ .



## Algoritmos públicos y privados

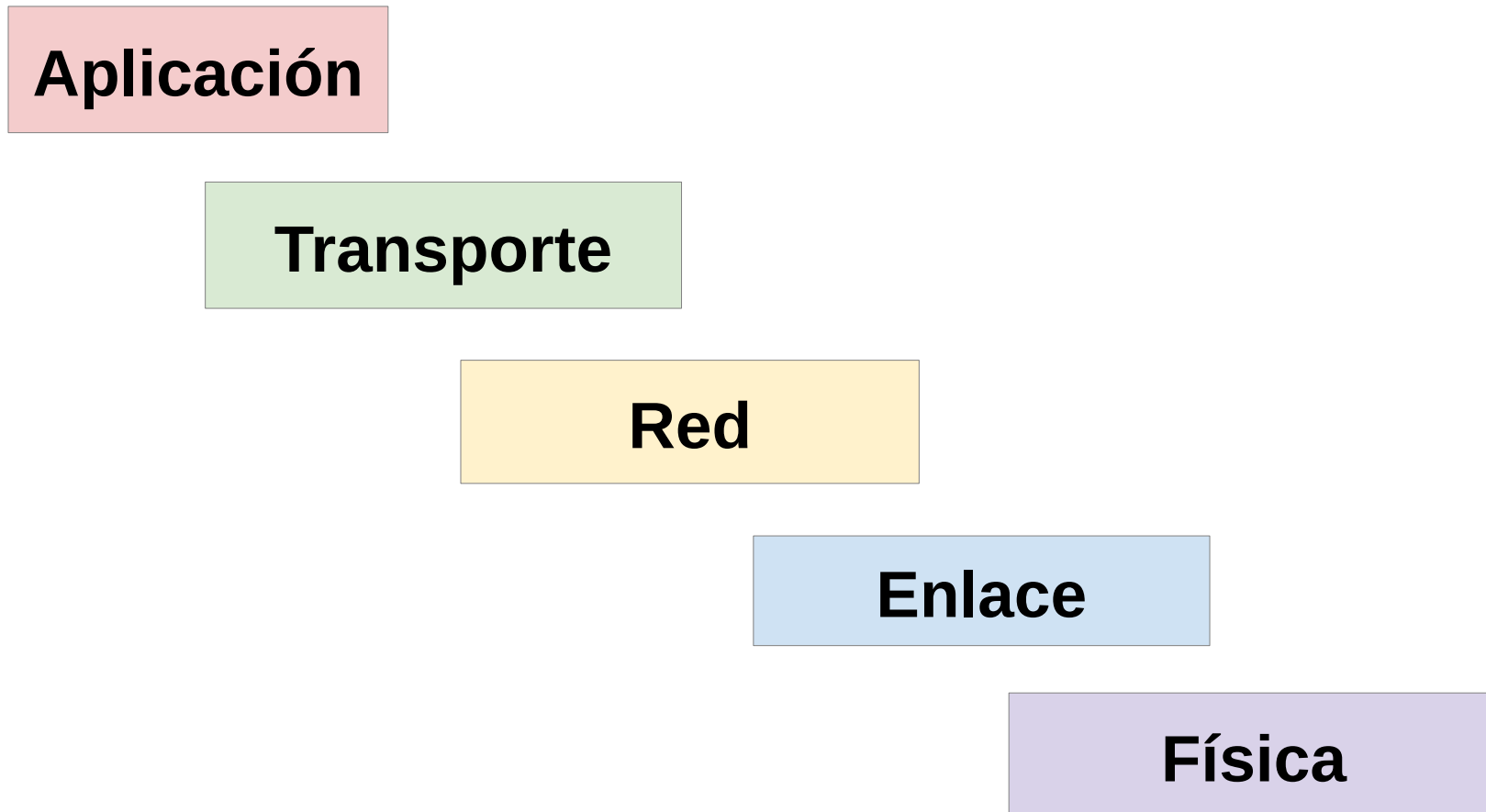
Los algoritmos públicos son aquellos cuya definición y funcionamiento se ponen a disposición pública, permitiendo que cualquier persona o entidad acceda al mismo para su evaluación o investigación.

En contraparte los privados son aquellos cuyo funcionamiento interno es desconocido; en el ámbito de la criptografía estos últimos son considerados menos confiables.

*Nota: Las patentes pueden condicionar el uso de ambos tipos de algoritmos*



## **Seguridad por capas**





## Http vs Https

**http:** Hyper Text Transfer Protocol, protocolo para transmisión de información en plano, sin cifrado. Su puerto por defecto es el número 80.

**https:** Hyper Text Transfer Protocol **Secure**, protocolo para transmisión información cifrada mediante SSL o TLS. Su puerto por defecto es el numero 443.



## SSL (Secure Sockets Layer)

Es un protocolo que proporciona privacidad e integridad entre dos aplicaciones. El sistema SSL es independiente del protocolo utilizado; esto significa que puede asegurar transacciones realizadas en la Web a través del protocolo HTTP y también conexiones a través de los protocolos FTP, POP e IMAP. SSL actúa como una capa adicional que permite garantizarla seguridad de los datos y que se ubica entre la capa de la aplicación y la capa de transporte (por ejemplo, el protocolo TCP)

Originado en Netscape su Version 3.0 data de **1996**, definida en la **RFC-6101** por la Internet Engineering Task Force (IETF).



## **SSL (Secure Sockets Layer)**

Los datos que circulan en un sentido y otro entre el cliente y el servidor se cifran mediante un algoritmo simétrico como DES o RC4. Un algoritmo de clave pública -generalmente RSA- se utiliza para el intercambio de las claves de cifrado y para las firmas digitales. El algoritmo utiliza la clave pública en el certificado digital del servidor. Con el certificado digital del servidor, el cliente también puede verificar la identidad del servidor. Las versiones 1 y 2 del protocolo SSL sólo proporcionan autenticación de servidor. La versión 3 agrega la autenticación del cliente, utilizando los certificados digitales de cliente y de servidor.



## **SSL (Secure Sockets Layer)**

### **Fases**

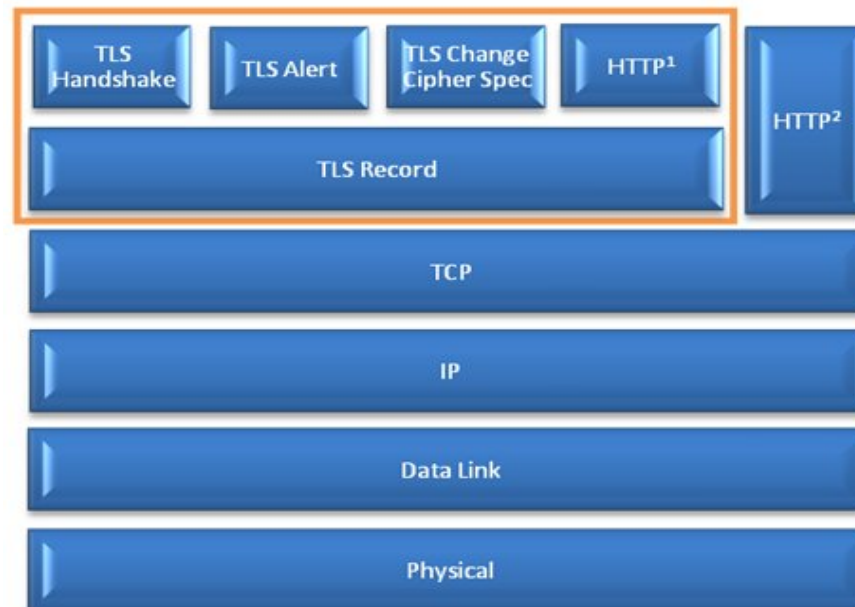
1. Establecimiento de la conexión y negociación de los algoritmos criptográficos que van a usarse en la comunicación, a partir del conjunto de algoritmos soportados por cada uno de los interlocutores.
2. Intercambio de claves, empleando algún mecanismo de clave pública y autenticación de los interlocutores a partir de sus certificados digitales.
3. Cifrado simétrico de tráfico.





## TLS (Transport Layer Security)

**TLS (Transport Layer Security)** es una evolución del protocolo SSL (**Secure Sockets Layer**), es un protocolo mediante el cual se establece una conexión segura por medio de un canal cifrado entre el cliente y servidor.





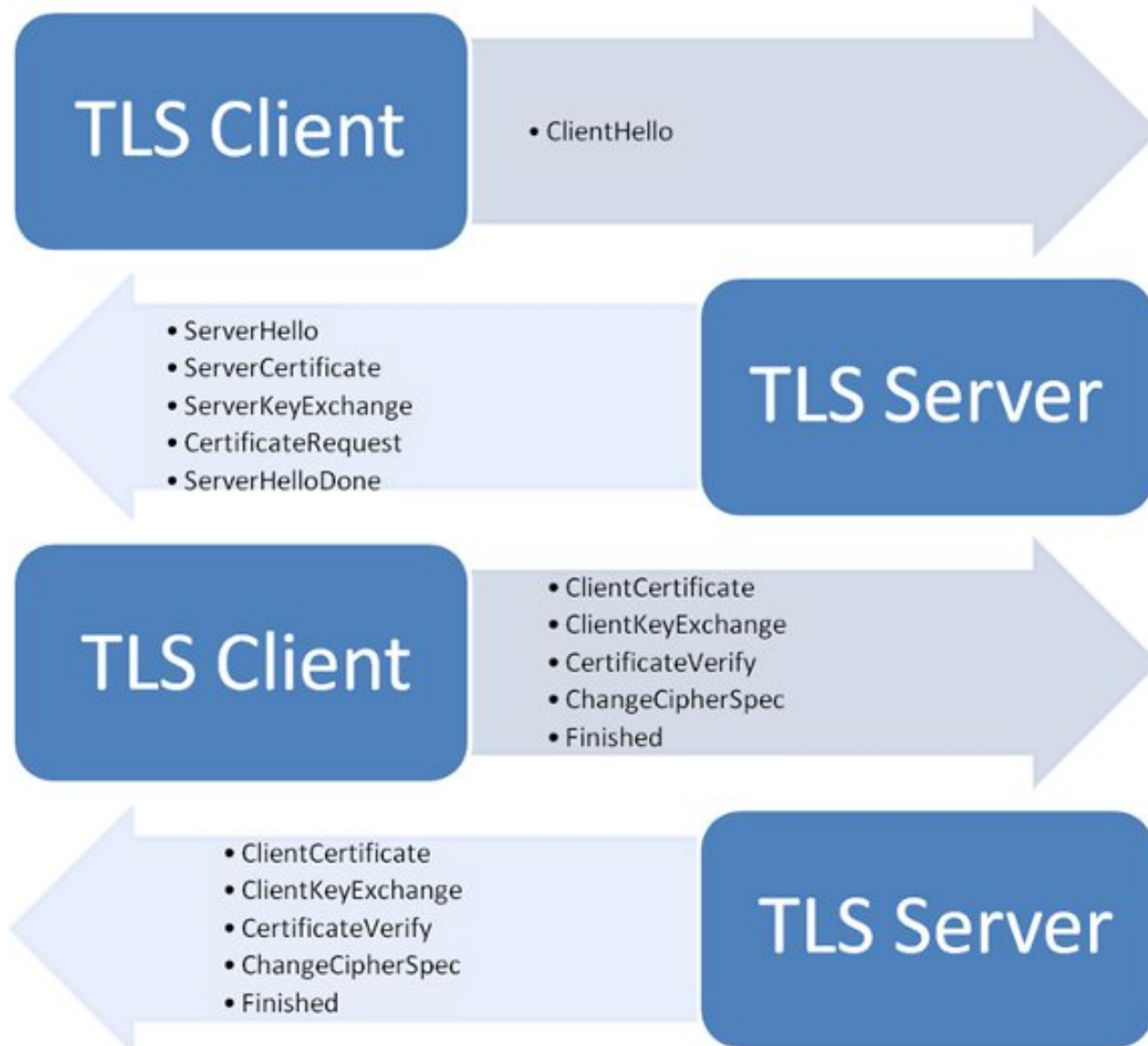
## **TLS (Transport Layer Security)**

### Características adicionales

- Incompatible con SSL v3.0
- Uso de funciones MAC en lugar de funciones MDC únicamente
- Numeración secuencial de todos los campos que componen la comunicación, e incorporación de esta información al cálculo de los MAC.
- Protección frente a ataques que intentan forzar el empleo de versiones antiguas —menos seguras— del protocolo o cifrados más débiles.
- El mensaje que finaliza la fase de establecimiento de la conexión incorpora una signatura (hash) de todos los datos intercambiados por ambos interlocutores.



## TLS (Transport Layer Security)





## **TLS (Transport Layer Security)**

### Algoritmos utilizados

- Cifrado Asimetrico: RSA, Diffie-Hellman(DHE), Curva Eliptica (ECDHE), DSA (Digital Signature Algorithm).
- Cifrado Simetrico: RC2, RC4, IDEA (International Data Encryption Algorithm), DES (Data Encryption Standard), Triple DES o AES (Advanced Encryption Standard)
- Funciones de Hash: MD5 o de la familia SHA .



## **TLS (Transport Layer Security)**

### **RFC 2246 (1999) - The TLS Protocol Version 1.0**

Primer versión del protocolo

### **RFC 4346 (2006) - The TLS Protocol Version 1.1**

Se destacan mejoras en el manejo del padding y protecciones a ataques por CBC usando IV explícitos

### **RFC 5246 (2008) / RFC 6176 (2011) - The TLS Protocol Version 1.2**

Incorporación de SHA-256 para función de PRF(pseudo-aleatoria) y cierre de mensajes, SHA-1 para firma digital, entre otros.

### **RFC 8446 (2018) - The TLS Protocol Version 1.3**

Remoción de cifrados obsoletos, optimización del protocolo (handshaking con Zero Round-Trip Time, etc)



## **Firma Electrónica**

Se entiende por firma electrónica al conjunto de datos electrónicos integrados, ligados o asociados de manera lógica a otros datos electrónicos, utilizado por el signatario como su medio de identificación, que carezca de alguno de los requisitos legales para ser considerada firma digital. En caso de ser desconocida la firma electrónica corresponde a quien la invoca acreditar su validez.



## **Firma Digital**

Se entiende por firma digital al resultado de aplicar a un documento digital un procedimiento matemático que requiere información de exclusivo conocimiento del firmante, encontrándose ésta bajo su absoluto control. La firma digital debe ser susceptible de verificación por terceras partes, tal que dicha verificación simultáneamente permita identificar al firmante y detectar cualquier alteración del documento digital posterior a su firma.



## **Firma digital - Propiedades**

- Va ligada indisolublemente al mensaje. Una firma digital válida para un documento no puede ser válida para otro distinto.
- Solo puede ser generada por su legítimo titular. Al igual que cada persona tiene una forma diferente de escribir, y que la escritura de dos personas diferentes puede ser distinguida mediante análisis caligráficos, una firma digital sólo puede ser construida por la persona o personas a quienes legalmente corresponde.
- Es públicamente verificable. Cualquiera puede comprobar su autenticidad en cualquier momento, de forma sencilla.



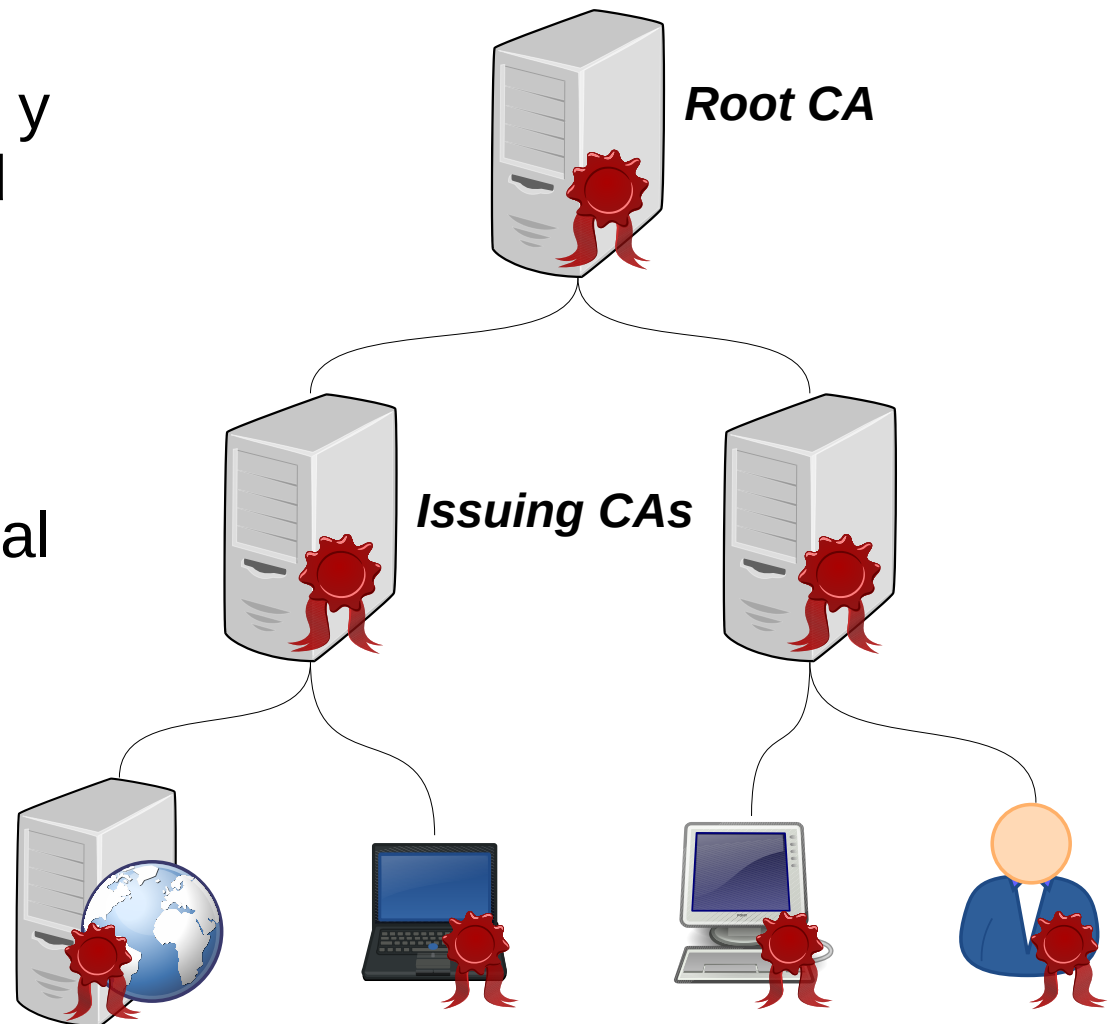


## **Modelos de Infraestructura de Seguridad**



## PKI – Infraestructura de Clave Publica

Es una combinación de hardware, software, políticas y procedimientos de seguridad que define un entorno de confianza **centralizado** y provee garantías para operaciones criptográficas como el cifrado, la firma digital o el no repudio de transacciones electrónicas





## **Certificados Digitales**

Se entiende por certificado digital al documento digital firmado digitalmente por un certificador, que vincula los datos de verificación de firma a su titular

Es esencialmente una clave pública, un identificador e información accesorio; firmados digitalmente por una autoridad de certificación, y su utilidad es demostrar que una clave pública pertenece a un usuario concreto.



## **Certificados Digitales**

El estándar X.509 solo define la sintaxis de los certificados, por lo que no esta atado a ningún algoritmo en particular, y contempla los siguientes campos:

- Versión.
- Numero de serie.
- Identificador del algoritmo empleado para la firma digital.
- Nombre del certificador.
- Periodo de validez.
- Nombre del sujeto.
- Clave pública del sujeto.
- Identificador único del certificador.
- Identificador unico del sujeto.
- Extensiones.
- Firma digital de todo lo anterior generada por el certificador.



## **Certificados Digitales de revocación**

Cuando una clave pública pierde su validez —por destrucción o robo de la clave privada correspondiente, por ejemplo—, es necesario anularla. Para ello se emplean los denominados certificados de revocación que no son más que un mensaje que identifica a la clave pública que se desea anular, firmada por la clave privada correspondiente.



## **PKI – Infraestructura de Clave Publica**

### Componentes:

- **Autoridad de Certificación** (CA, Certificate Authority): Emite y revoca certificados, vinculando las claves publicas con la identidad del propietario.
- **Autoridad de Registro** (RA, Registration Authority): Verifica la relación de los certificados y la identidad de sus titulares.
- **Autoridad de Validación** (VA, Validation Authority): Es la encargada de comprobar la validez de los certificados digitales.
- **Autoridad de Sellado de Tiempo** (TSA, TimeStamp Authority): Es la encargada de firmar documentos con la finalidad de probar que existían antes de un determinado instante de tiempo.
- **Los repositorios**: Almacenan información relativa a la PKI, como certificados y listas de revocacion (CRL, Certificate Revocation List).
- **Los usuarios y entidades finales** que poseen un par de claves (pública y privada) y un certificado asociado a su clave pública.



## Certificate Transparency



CT Logs: Certificate Transparency logs  
SCT: Signed Certificate Timestamp

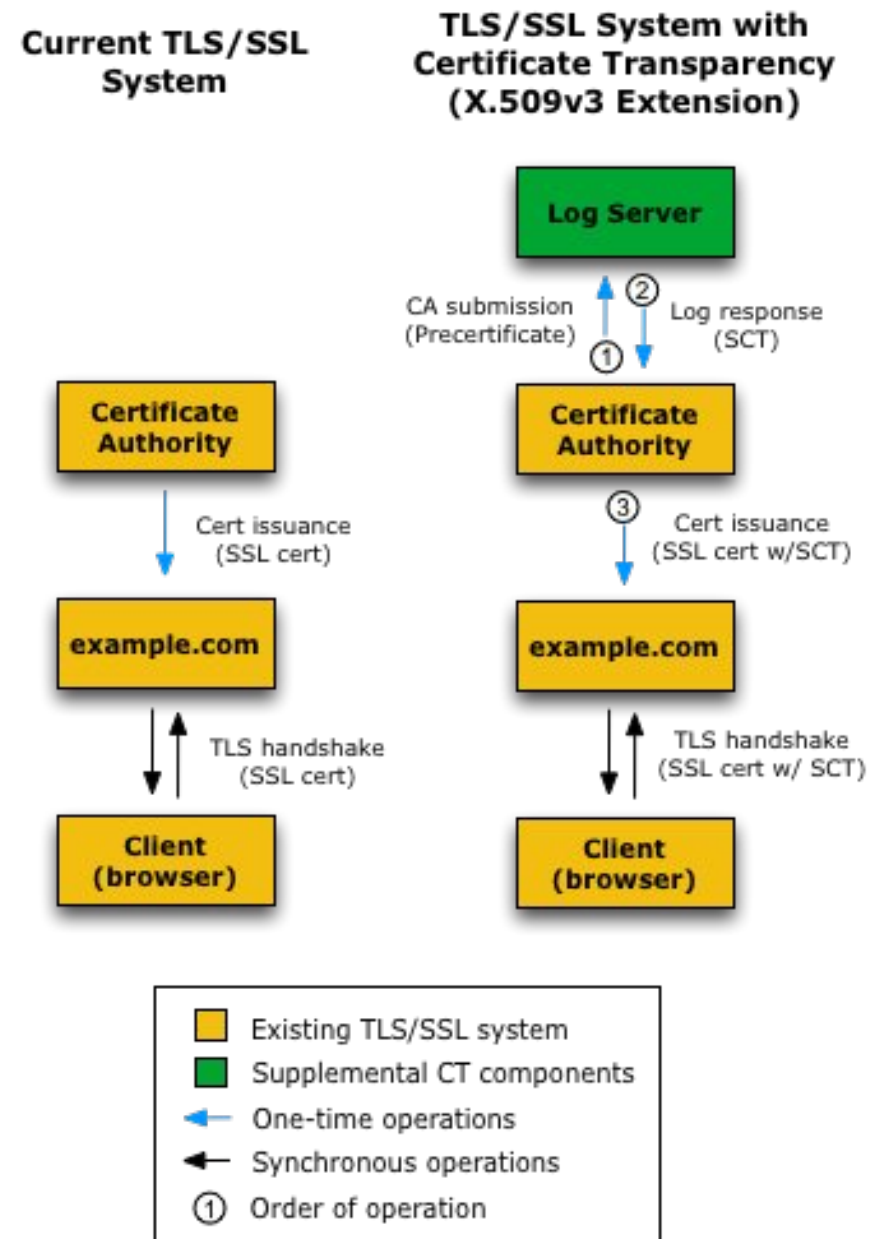


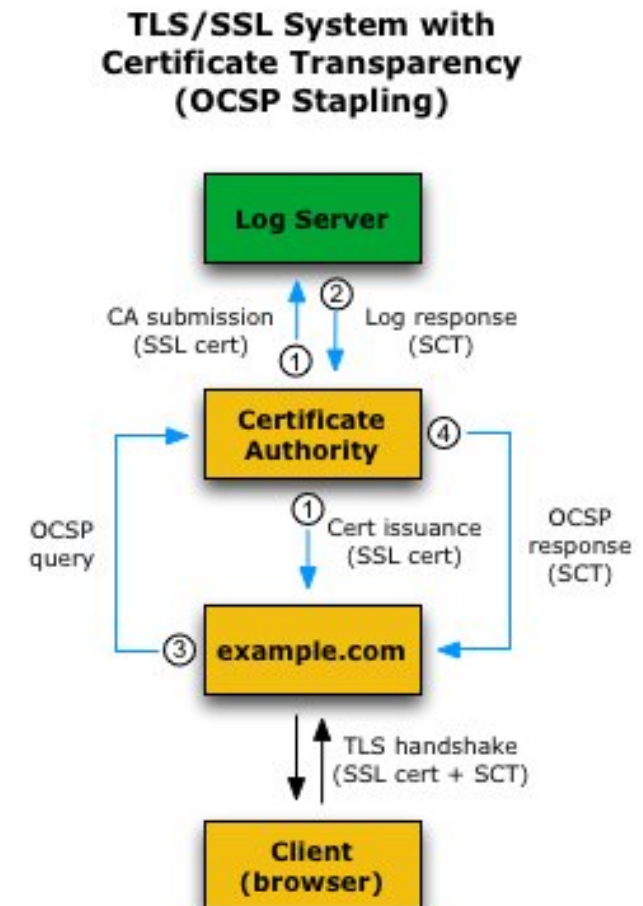
Figure 1



## Online Certificate Status Protocol

OCSP: es un método para determinar el estado de vigencia de un certificado digital X.509 usando otros medios que no sean el uso de CRL (Listas de Revocación de Certificados).

<https://tools.ietf.org/html/rfc6960>

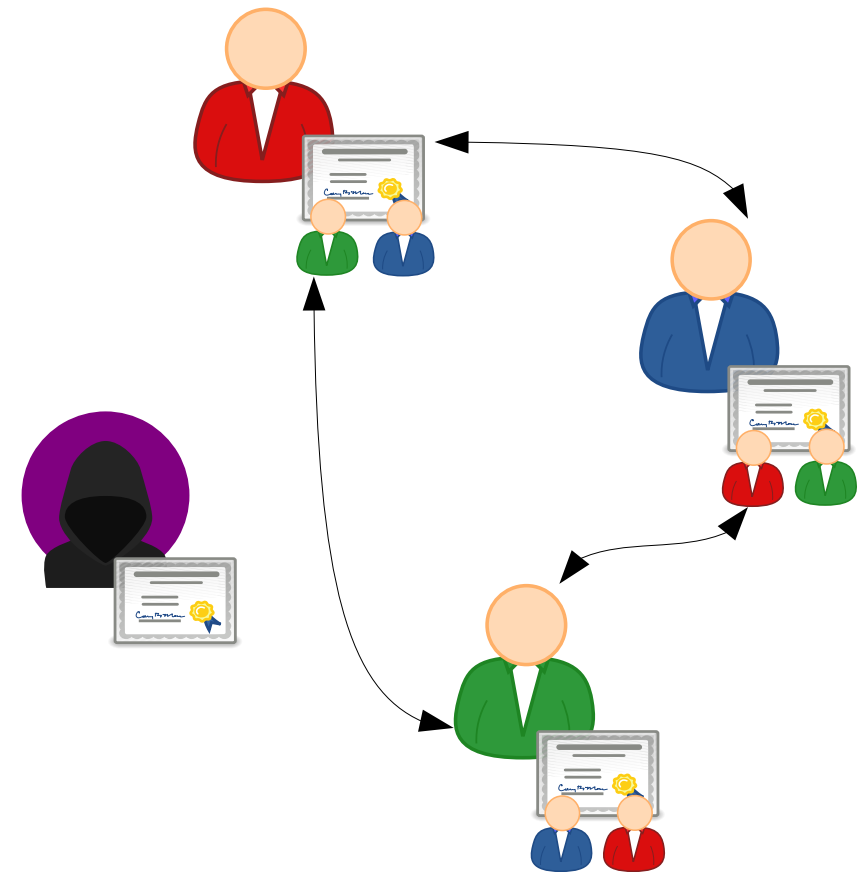






## Anillo o Circulo de confianza

Es un modelo de confianza **distribuido** que provee garantías para operaciones criptográficas como el cifrado, la firma o el no repudio de transacciones electrónicas basado en la cantidad de firmas de actores de confianza que posea una clave publica.





## Anillo o Circulo de confianza

El concepto original es un aporte de Phil Zimmermann, documentado en el manual de PGP Version 2.0 (1992)

*Con el paso del tiempo, usted acumulará claves de otras personas que podría querer designar como introductores de confianza. Todos los demás elegirán sus propios introductores de confianza. Y todos gradualmente acumularán y distribuirán junto con su clave una colección de firmas certificadas por otras personas, en la expectativa de que quien quiera que la reciba confiará por lo menos en una o dos de las firmas. Esto llevará a la aparición (espontánea) de un anillo de confianza descentralizado y resistente a los fallos para todas las claves públicas.*



## **Situación en La República Argentina**



## **Ley de Firma Digital - República Argentina**

### **Ley 25.506**

**Consideraciones generales. Certificados digitales.  
Certificador licenciado. Titular de un certificado digital.  
Organización institucional. Autoridad de aplicación.  
Sistema de auditoría. Comisión Asesora para la  
Infraestructura de Firma Digital. Responsabilidad.  
Sanciones. Disposiciones Complementarias.**

**Sancionada: Noviembre 14 de 2001.**

**Promulgada de Hecho: Diciembre 11 de 2001.**



El proyecto de Firma Digital tiene por objetivo lograr la implementación de esta herramienta tecnológica en los sistemas administrativos y de gestión de los distintos organismos que conforman la Administración Pública, con el fin de que el accionar de éstos resulte más eficiente.

Con este propósito, el equipo de Firma Digital de la ONTI lleva adelante las siguientes tareas: generar un marco tecnológico, legal y procedimental adecuado que conforme la **Infraestructura de Firma Digital Nacional (IFDN)**, con el fin de poder utilizar esta tecnología en forma segura; capacitar/instruir a los distintos actores que conforman la IFDN; proveer de certificados a los organismos del sector público en forma gratuita.



**ACAP**

Autoridad Certificante de la Administración Pública.  
<https://pki.jgm.gov.ar/app>



**AC Raíz**

Autoridad Certificante Raíz de la República Argentina.  
<https://www.acraiz.gob.ar/>



**AC Mail**

Autoridad Certificante para correo electrónico.  
<http://ca.sgp.gov.ar/eMail/>



La **Infraestructura de Firma Digital de la República Argentina (IFDRA)** está conformada por un conjunto de componentes que interactúan entre sí, permitiendo la emisión de certificados digitales para verificar firmas digitales en condiciones seguras, tanto desde el punto de vista técnico como legal.

La **Secretaría de Gabinete y Coordinación Administrativa actúa como Ente Licenciante** otorgando, denegando o revocando las licencias de los certificadores licenciados y supervisando su accionar.

En este sentido, los certificadores licenciados son entidades públicas o privadas que se encuentran habilitados por el Ente Licenciante para emitir certificados digitales, en el marco de la Ley **25.506 de Firma Digital**.



## **Certificadores Licenciados**

- **AFIP** – Administración Federal de Ingresos Públicos. -  
<http://www.afip.gov.ar/firmaDigital/>
- **ANSES** – Administración Nacional de la Seguridad Social.  
- <http://www.anses.gob.ar/firmadigital/index.html>
- **ONTI** – Oficina Nacional de Tecnologías de Información. -  
<https://pki.jgm.gov.ar/app/>
- **ENCODE S.A.** -  
<http://www.encode.com.ar/pages/firma.html>

