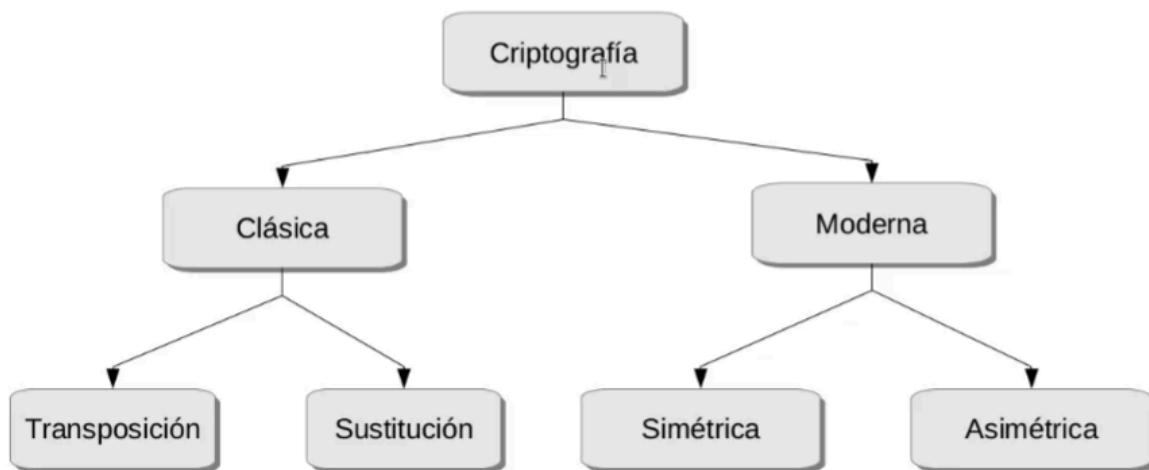
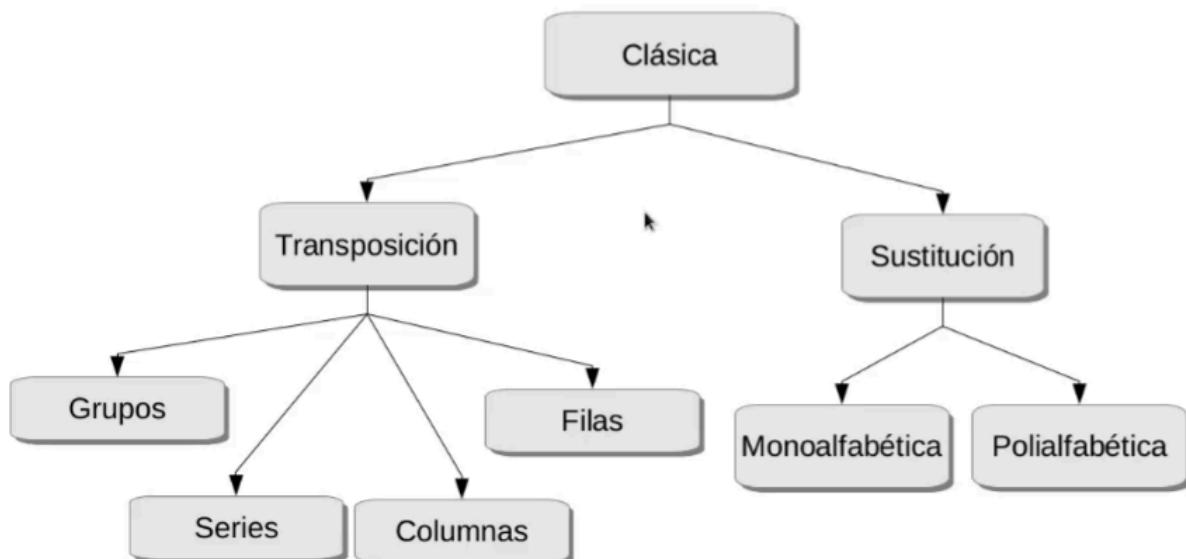


criptografía

Clasificación



Criptografía clásica Sub-Clasificación



Elementos teoricos de la criptografia

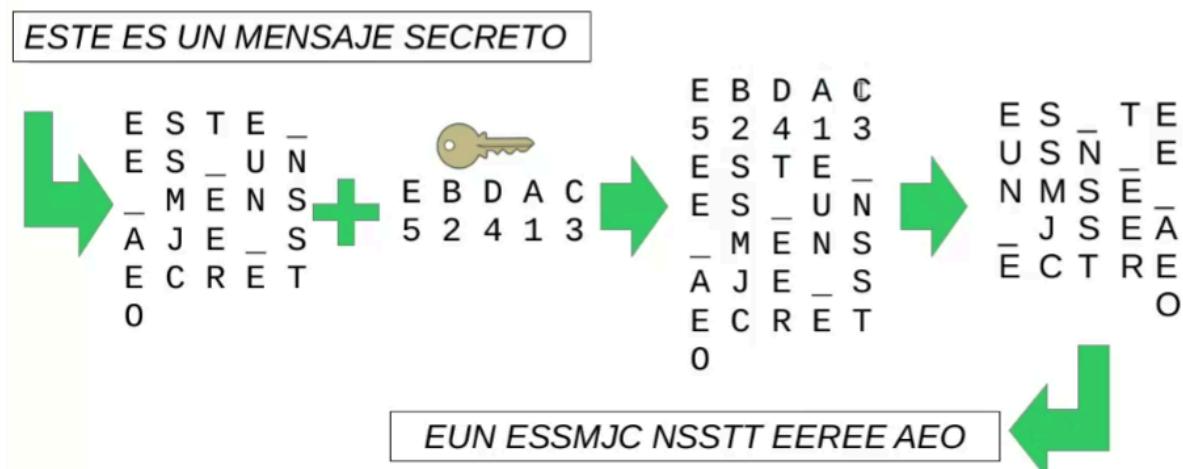
$$D(k, E(k, m)) = m$$

- 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 para cada valor posible de la clave k .
- D es el conjunto de transformaciones de descifrado, análogo a E .

Criptografía clásica

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

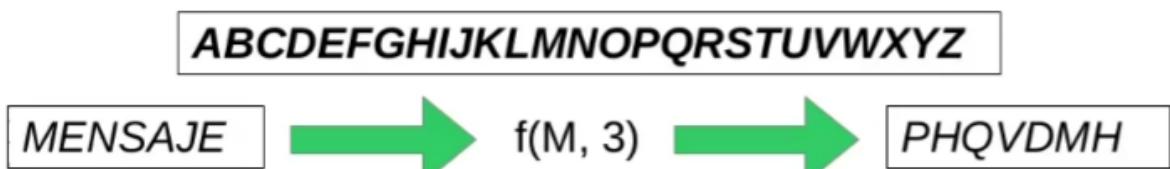
Este es un mensaje secreto



1 columna por cada carácter de la contraseña ,si la contraseña es de 5 caracteres ,5 columnas

Cifradores por sustitución: 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)



NOTA:en el parcial puede aparecer de un cifrado simple analfabetica y van a dar un práctico para practicar y no tener sorpresas

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

Ejemplo: Cifrado de Vigenère, 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: **ABCDABCDABCDAB**

El texto cifrado es: **TFZWO EGSIV 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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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

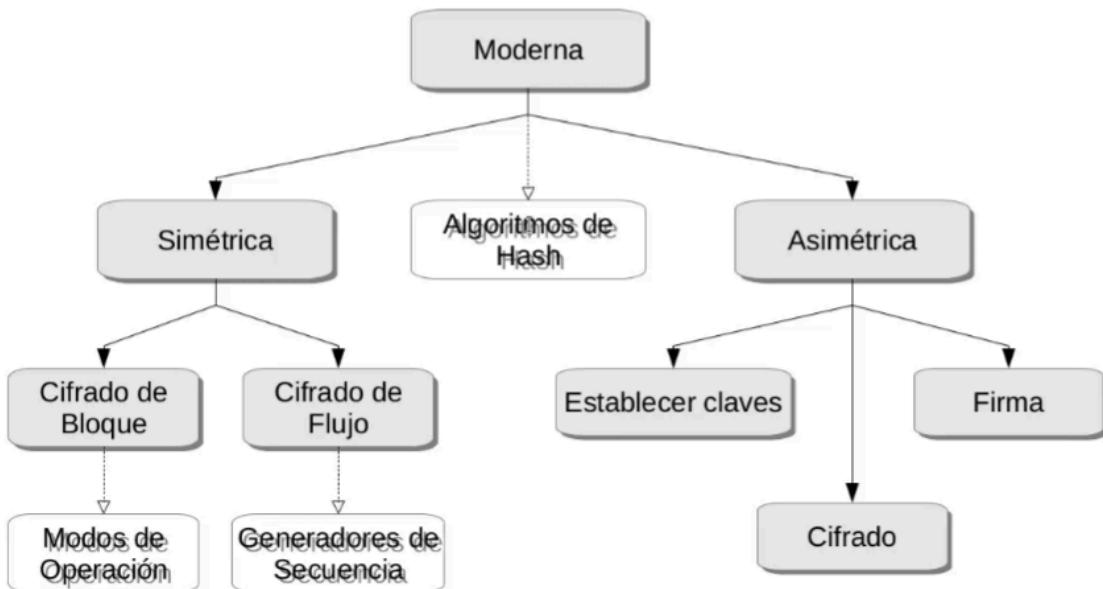
Referencia – Esquemas de codificación

Binario: Representa la información a nivel de bits con 0 y .

Hexadecimal: Representa la información con 16 caracteres con representación gráfica (0123456789ABCDEF), asignando cuatro bits a cada carácter.

Base64: Representa la información con 64 caracteres representables (ABCDEFGHIJKLMNPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/-), indicando con un carácter especial (=) los octetos faltantes del último bloque.

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.

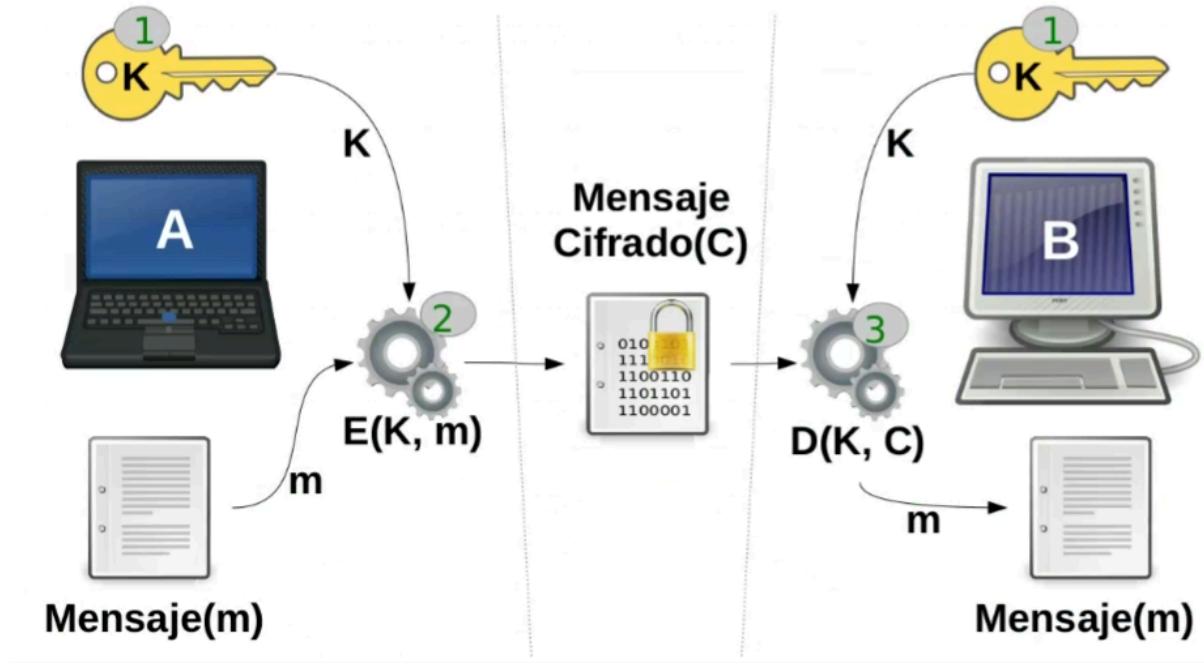
Ventajas

- 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) (se encuentra en la pc ,equipamiento militar,etc)**
- Serpent (1998)
- Twofish
- RC6 (1998, Rivest Cipher 6)
- MARS (1998, IBM)
- GOST (1994, Magma URSS)
- CAMELLIA (2000, NTT y Mitsubishi Electric)
- IDEA (1991, International Data Encryption Algorithm)
- Blowfish (1993, diseñado por Bruce Schneier)
- RC5 (1994, Rivest Cipher 5)

Referencia matemática - XOR

Propiedades

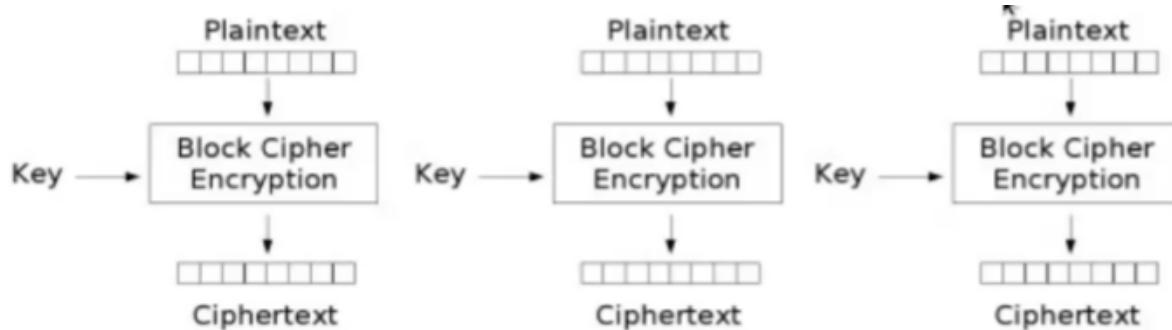
- Es conmutativa: Es decir que $A \text{ xor } B = B \text{ xor } A$
- Asociativa: $(A \text{ xor } B) \text{ xor } C = A \text{ xor } (B \text{ xor } C)$
- Autoinversa: $(A \text{ xor } B) \text{ xor } B = A$



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

Modos de cifrado de bloques

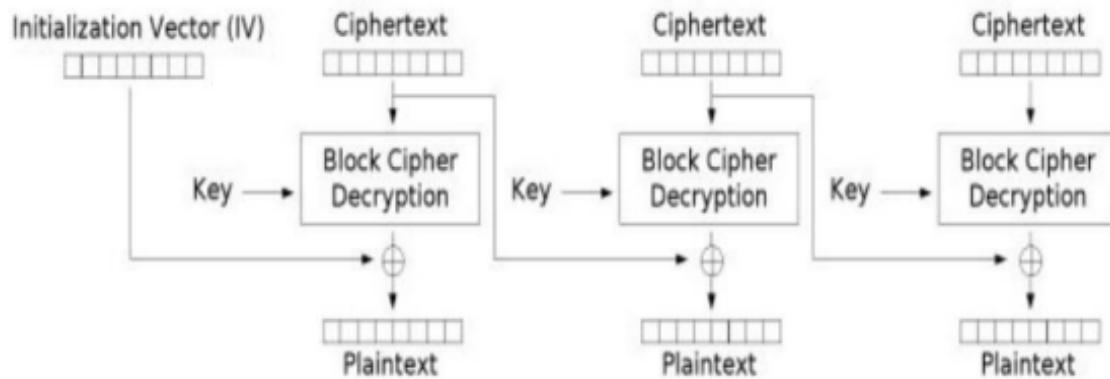
ECB: Electronic codebook, en este método el mensaje se fracciona en partes y cada una es cifrada de manera independiente.



Electronic Codebook (ECB) mode encryption

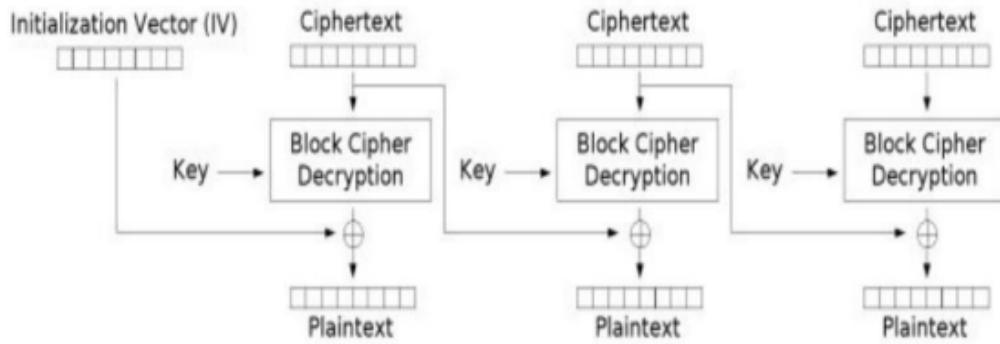
Un ejemplo de ECB es el ataque por marca de agua.

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.

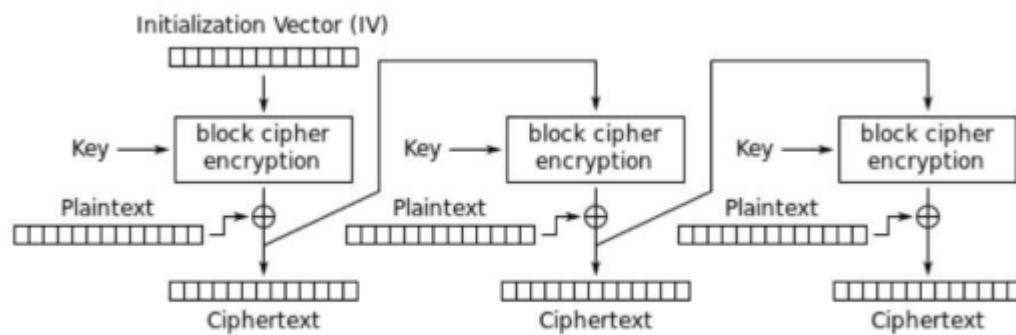


Cipher Block Chaining (CBC) mode decryption

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.

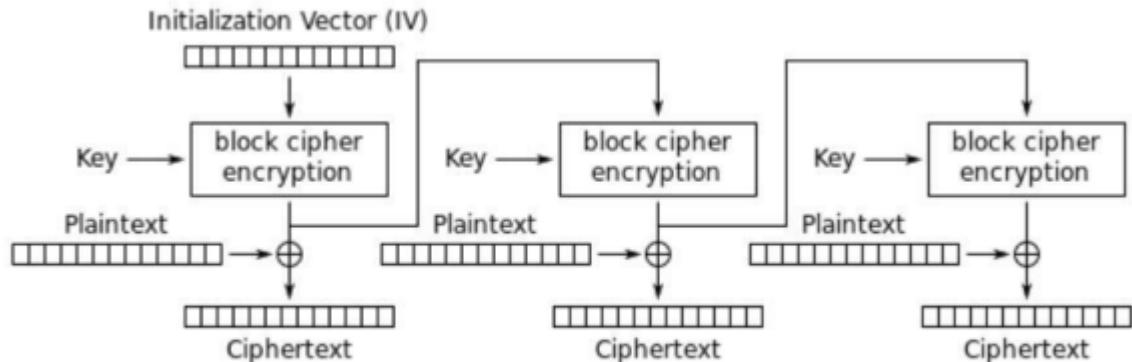


Cipher Block Chaining (CBC) mode decryption



Cipher Feedback (CFB) mode encryption

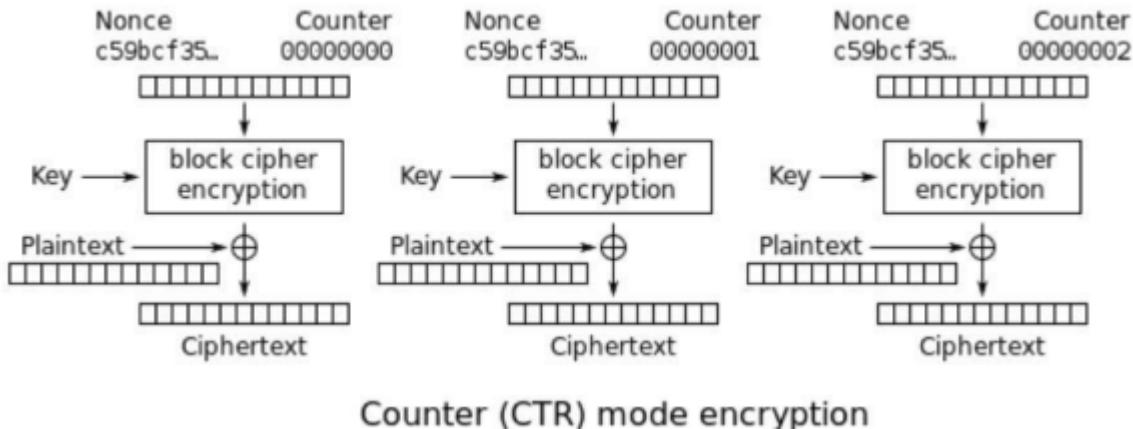
OFB: Output Feedback, este método opera de manera similar a CFB con la diferencia que el bloque a ser utilizado como entrada del siguiente proceso es tomado de la salida del algoritmo justo antes de realizar el XOR.



Output Feedback (OFB) mode encryption

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.



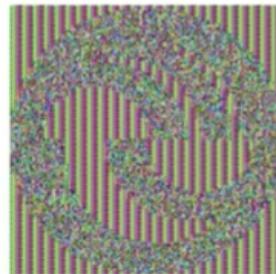
Otros modos de cifrado de bloques: Existen diversos modos y algunos de ellos incorporan autenticación a la confidencialidad.

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

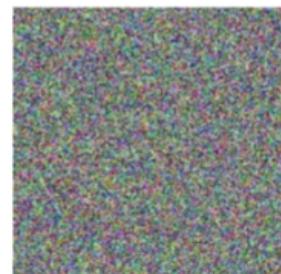
Ataques por marca de agua: A continuación se observa el resultado de cifrar con AES128 una imagen utilizando diferentes modos de cifrado de bloque.



Original



ECB



CBC

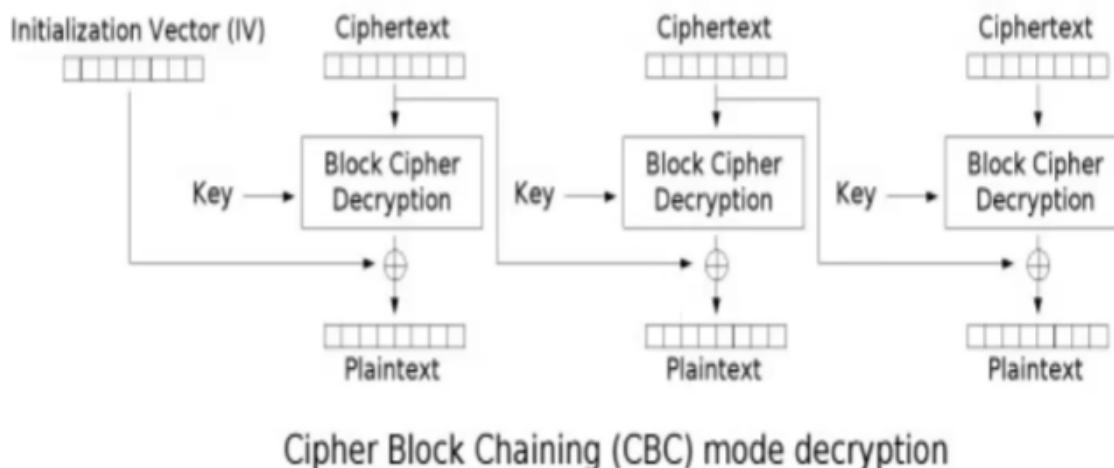
head -n 4 unlam.ppm > header.txt

```

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.ecb.bin
cat header.txt body.ecb.bin > unlam.ecb.ppm

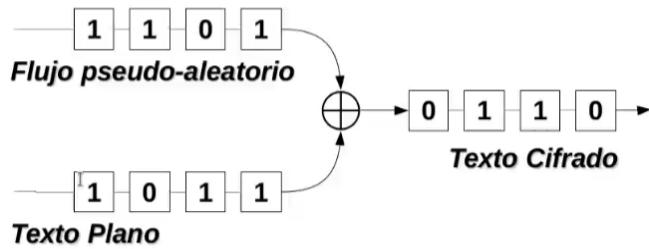
```

vector inicialización es un elemento que se usa una vez por cifrado tiene el mismo tamaño que el valor del bloque y que no es secreto ,secreto es la contraseña.



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).



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.

Estos son algunos de los actuales algoritmos de cifrado de flujo:

- RC4
- Salsa20*
- ChaCha20
- Trivium*
- A5/1, A5/2
- Chameleon
- FISH
- Helix
- Grain*
- ISAAC
- MUGI
- Panamá
- Phelix
- Pike
- SEAL
- SOBER/SOBER-128
- WAKE
- Rabbit*

Funciones de HASH

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

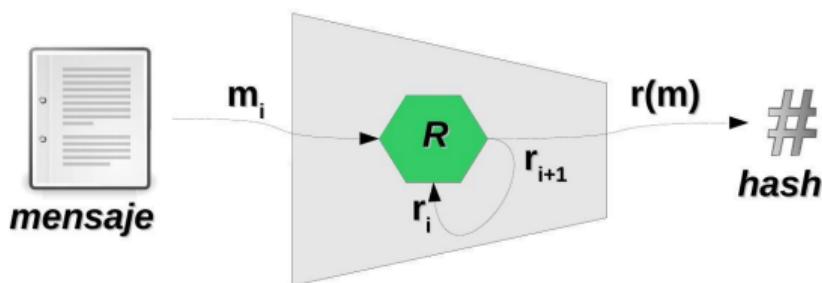
Principales usos

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

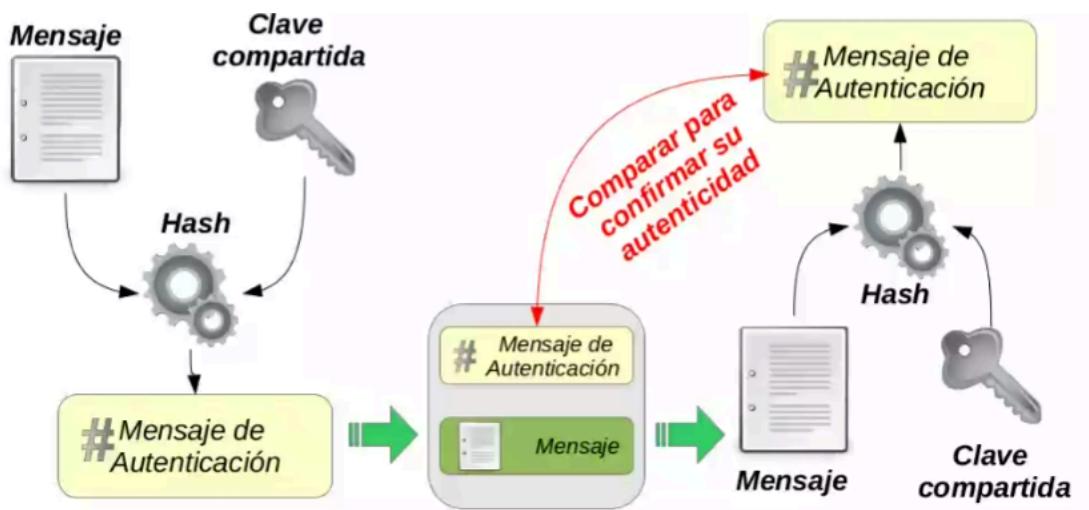
Propiedades

- $r(m)$ 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 -ó bloque del mensaje (m_i) y de la salida del paso previo, $i-1$. Se considera una buena práctica incluir en el mensaje m la longitud y características del mensaje.



Funciones de HASH - MAC: Message Authentication Code: Añade criptografía al proceso de hash para aumentar la seguridad del mismo. El MAC es un código de hash.



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 subcadena — 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.

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, SHA384 y SHA-512.

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 y 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.
- ❖ **Groestl**: Por Praveen Gauravaram, Lars Knudsen, Krystian Matusiewicz, Florian Mendel, Christian Rechberger, Martin Schläffer, y 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 más claves en base a un valor maestro o clave inicial secretos, más 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 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
- **HKDF** - (2010) HMAC-based Extract-and-Expand Key Derivation Function - RFC5869
- **Argon2** - (2015) Por la Universidad de Luxemburgo - RFC9106

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.

hasta aca verifique termine la primera ptt de criptos

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.

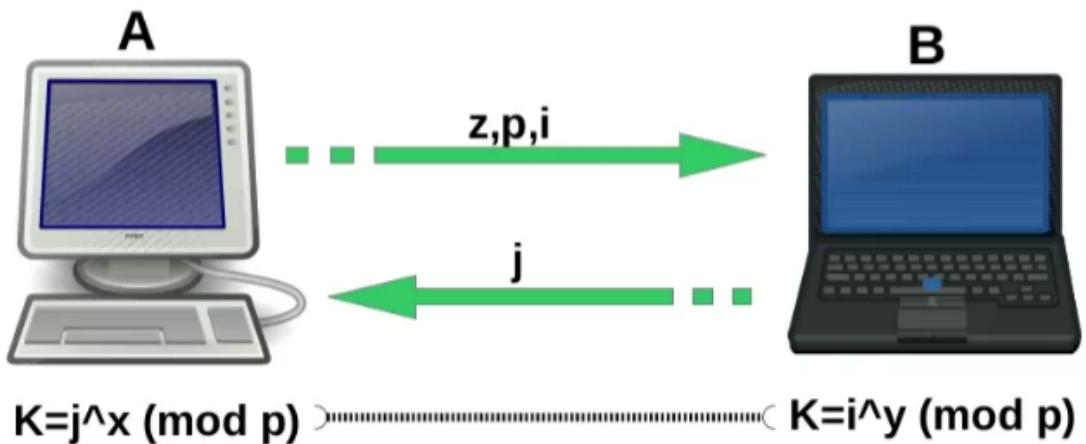
El sistema de cifrado de clave pública usa un par de claves para el envío de mensajes. Las dos claves pertenecen a la misma persona a la que se ha enviado el mensaje. Una clave es *pública* y se puede entregar a cualquier persona. La otra clave es *privada* y el propietario debe guardarla para que nadie 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 usarse 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
 $(i = z^x \backslash (\text{mod } p))$
 2. B escoge un número aleatorio y , análogamente al paso anterior, y envía a **A** el valor
 $(j = z^y \backslash (\text{mod } p))$
- Z no sabe Y
3. **B** recoge i y calcula $K = j^i \pmod{p} = (z^x \pmod{p})^i \pmod{p}$
 4. **A** recoge j y calcula $K = j^i \pmod{p} = (z^y \pmod{p})^i \pmod{p}$
 5. Conclusión $K = z^{xy} \pmod{p}$

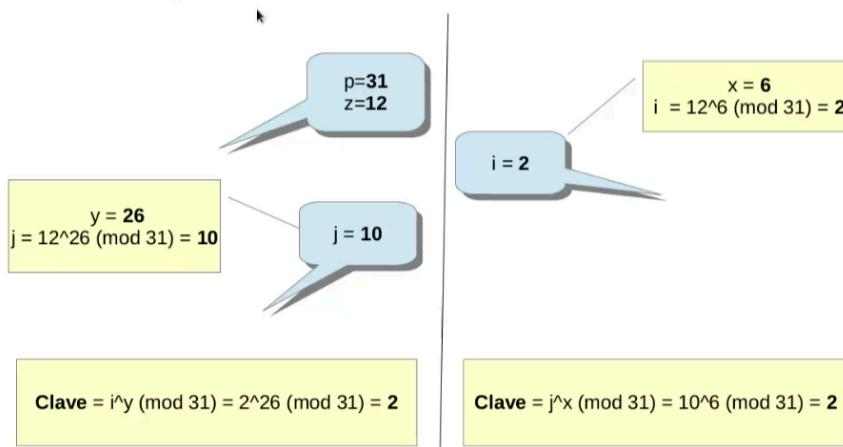


$K = j^{j^i} \pmod{p}$

$K = i^{i^j} \pmod{p}$

K de la pc B solo lo conoce a y b

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

Curvas Elípticas

Una curva elíptica es un tipo especial de curva algebraica definida por una ecuación de la forma:

$$[y^2 = x^3 + ax + b]$$

donde **a** y **b** son constantes y los números **x** e **y** son variables que pertenecen a un campo definido.

Ejemplos de uso criptográfico:

- P-256 (secp256r1)
- P-384 (secp384r1)
- Curve25519
- secp256k1

Algoritmos asimétricos

Ejemplos

- Diffie-Hellman
- RSA
- ElGamal
- DSA
- ECC, Criptografía de curva elíptica (ECDH, ECDSA...)
- CRYSTALS-Kyber (ML-KEM)
- CRYSTALS-Dilithium (ML-DSA)
- Sphincs+ (SLH-DSA)
- FALCON (FN-DSA)

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

El hardware limita la longitud

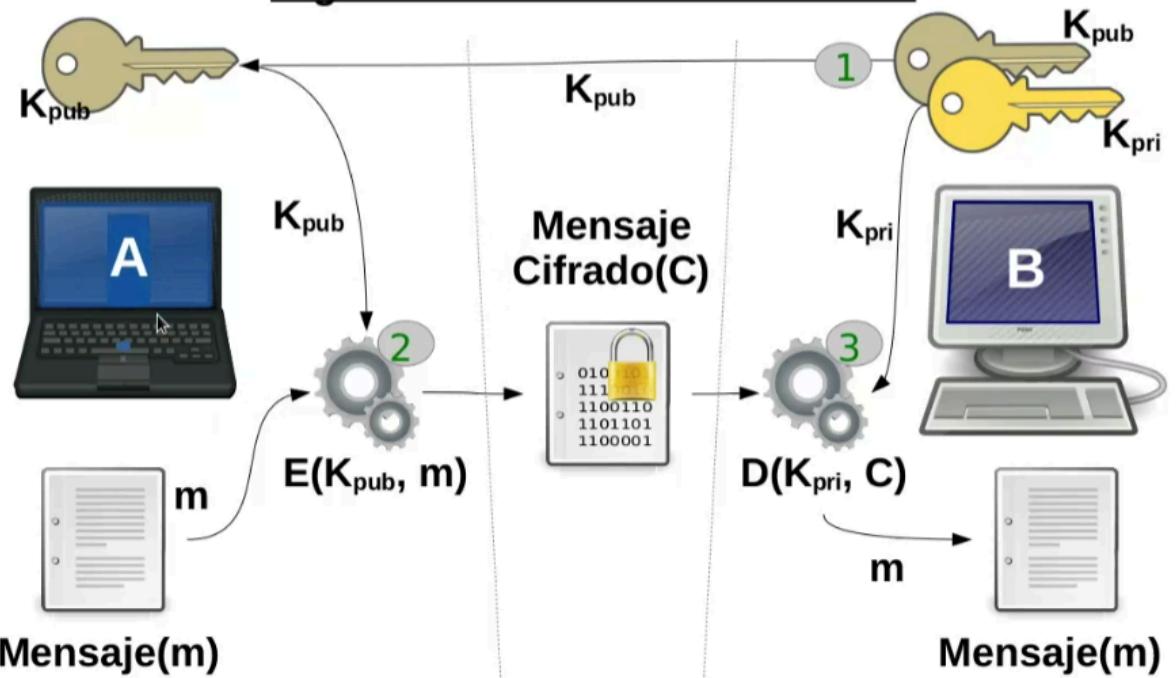
Este tipo de algoritmos suelen disponer de las siguientes operaciones:

1. Generación de claves
2. Cifrado(clave publica)
3. Descifrado (clave privada)
4. Firma (Esto se parece a un MAC) te deja tomar un mensaje y tomar una firma sobre ese mensaje,(firma con la privada)
5. Verificación de firma (clave pública)

Todo lo que sea de cifrado empieza por la clave del receptor

NOTA : solo interesa si las 2 cosas son correctas y hay una falla no sirve ,no vale la pena revisar y buscar la falla,matemáticamente es imposible

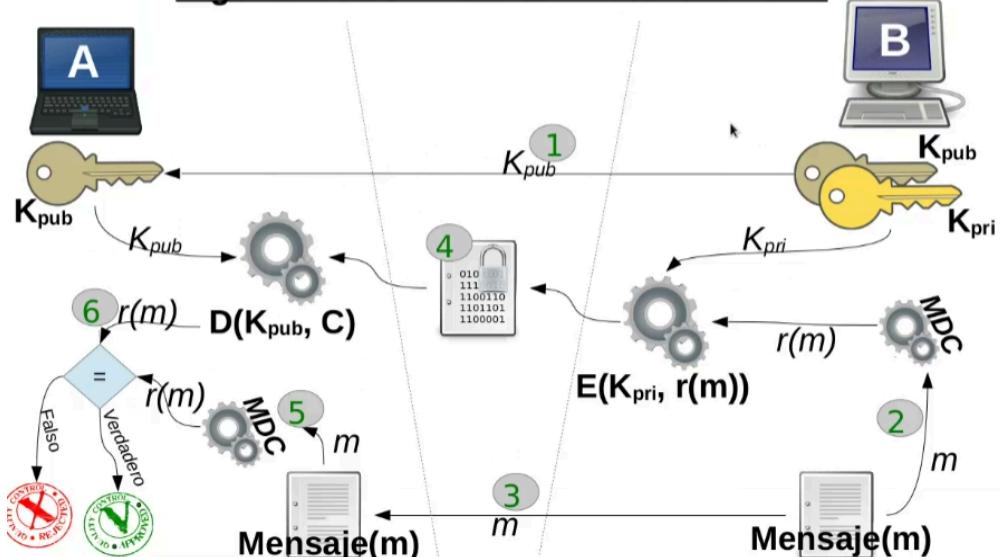
Algoritmos asimétricos - Cifrado



Si pierdo las clave se pierde en mensaje

La clave publica del receptor es la que uso para filtrar información

Algoritmos asimétricos - Autentificación



Pasos para hacer una vista

En B distribuye la clave K pública ,A obtiene la clave de B y de ahí se va a D después se filtra con un MDC y el mensaje se traslada se procesa el MDC de la clave privada de B y se genera la firma E

Algoritmos asimétricos - RSA

- Su nombre deriva de Ronald Rivest, Adi Shamir y Leonard Adleman
 - Fue publicado en 1977
 - 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**. Escogemos 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 existiría un número **d** tal que

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

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.

Aquí tienes el texto extraído de la vigésima segunda imagen:

Algoritmos asimétricos – RSA

Elección de números primos

$$p=3$$

$$q=11$$

$$n=3 * 11 = 33$$

$$\phi(n) = (3-1)(11-1) = 210 = 20$$

Elección del componente público
 $e = 7$ (Primo relativo entre 1 y $\phi(n)$)

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

$$e * d = 1 \text{ mod } (\phi(n))$$

$$7 * d = 1 \text{ mod } (20)$$

$$7 * 3 = 1 \text{ mod } (20)$$

$$21 \text{ mod } 20 = 1$$

$$\mathbf{d = 3}$$

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:

$$[c = m^e \text{ mod } n]$$

Mientras que el descifrado se hará de la siguiente forma:

$$[m = c^d \text{ mod } n]$$

Algoritmos asimétricos - RSA

Ejemplos de operación sobre el mensaje con valor 2

$$[c = m^e \text{ mod } n]$$

$$[c = 2^7 \text{ mod } 33]$$

$$[c = 128 \text{ mod } 33 = 29]$$

Descifrado para el valor 29

$$[m = c^d \text{ mod } n]$$

$$[m = 29^3 \text{ mod } 33]$$

$$[m = 24389 \text{ mod } 33 = 2]$$

Este ejemplo muestra cómo se cifra y descifra un mensaje con valor 2 utilizando RSA.

Firmas Digitales de RSA

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

$$[s = h^d \mod n]$$

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

$$[h = s^e \mod n]$$

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

$$[s = h^d \mod n]$$

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

$$[h = s^e \mod n]$$

Con esto termina los 5 métodos asimétricos

ELGAMAL YA ES OTRO TEMA

Algoritmos asimétricos - El Gamal

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 ($x=a^y \mod n$) donde (x, n) y (a) son constantes, y es la incógnita. A partir de ahora notamos esta situación como:

$$[y = \log_a x \mod n]$$

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

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

$$[y = p^x \mod n]$$

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

Número primo seleccionado al azar
(n = 17)

Números complementarios menores que (n), elegidos al azar
(p = 3)
(x = 6)

Cálculo del valor para el componente público

$$\begin{aligned}[y &= p^x \text{ mod } n] \\ [y &= 3^6 \text{ mod } 17] \\ [y &= 15]\end{aligned}$$

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

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

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:

$$[a = p^k \text{ mod } n] [b = y^k m \text{ mod } n]$$

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

$$[m = b \cdot a^{-x} \text{ mod } n]$$

Firmas de ElGamal: Para firmar un mensaje (m) basta con escoger un número (k) aleatorio, que sea primo relativo con (n - 1), y calcular:

$$[a = p^k \text{ mod } n] [b = (m - x a)^{k^{-1}} \text{ mod } n - 1]$$

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:

$$[y^a \cdot a^b = p^m \cdot (\text{mod} \cdot n)]$$

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.

Creación del par clave pública-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 $g \in Z^*$ calcular $a = g^{(p-1)/q} \pmod{p}$.
4. Si $a = 1$ volver al paso 3.
5. Seleccionar un número entero aleatorio a , tal que $1 \leq a \leq q-1$.
6. Calcular $y = a^q \pmod{p}$.
7. La clave pública es (p, q, a, y) . La clave privada es a .

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 $r = (a^k \pmod{p}) \pmod{q}$.
3. Calcular $k^{-1} \pmod{q}$.
4. Calculate $s = k^{-1} (h + ar) \pmod{q}$.
5. La firma del mensaje m es el par (r, s) .

Verificación de la firma

El destinatario efectuar las siguientes operaciones, suponiendo que conoce la clave pública (p, q, a, 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 $w = s^{-1} \pmod{q}$.
4. Calcular $u_1 = w \cdot h \pmod{q}$ y $u_2 = w \cdot r \pmod{q}$.
5. Calcular $v = (q_1 y u_2^2 \pmod{p}) \pmod{q}$.
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

algoritmos post cuántico

no solo corre en un pac cuántica sino que no ti

ALGORITMO CUÁNTICO Y POST CUANTICO son completamente distintos

EM MIEL PASA ESTA PTT

16 JUNIO FERIADO

7 JULIO SEGUNDO PARCIAL

14 DE JULIO RECUPERATORIO DEL PARCIAL

NOTA HAY EJERCICIO DE CRIPTO PARA PRACTICAR CON SU RESPUESTA

DIFERENCIA ENTRE

EL DEL FLUJO COMO OPERA DE BIT A A BIT ,EN LOS DECIFRADORES DE BLOQUE POR QUE FRACCIONA EL BLOQUE A BLOQUE EN LOS 2 PUEDOS MANEJAR GRANDES CANTIDADES DE DATOS SOLO QUE UNO LO HACE POR PEDAZOS Y OTRO POR BIT

a256 es simetrico de bloque

Mac: ¿QUE ES un codigo de mac ? es un algoritmo que proporciona un valor pequeño de salida para un mensaje pero ademas del mensaje te da una clave para que coincida con el codigo .

EL MDC: Es un equivalente del codigo de has

Ventaja del codigo asimetrico venian a solucionar los problemas simetricos ,solucionaron agregando la clave publica ,el algoritmo de difit gelman sirve para crear un valor para poder hablar sin la necesidad de que te escuche un tercero, no se puede elegir el numero es un calculo

Cada valor asimetrico tiene 5 operaciones :

1)Generar (tipica pregunta de examen)

Cifras con la clave publica de receptor y descifra con la clave privada y puede ser descifrada con la clave publica de B

El del firma es inverso ,para verificar si el

lo que importa el calculo matematico ,ver como identificar los algoritmos , los pasajes de claves,los pasajes de swa (algo asi) y las 5 operaciones de gelman.

Clase 2/ 6

Algoritmos Post-Cuánticos

La criptografía post-cuántica (PQC, Post-Quantum Cryptography), referencia a algoritmos criptográficos resistentes a ataques efectuados mediante computación cuántica.

- **FIPS 203 – CRYSTALS-Kyber**, ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism)
 - **FIPS 204 – CRYSTALS-Dilithium**, ML-DSA (Module-Lattice-Based Digital Signature Algorithm)
 - **FIPS 205 – Sphincs**, SLH-DSA (Stateless Hash-Based Digital Signature Algorithm)
 - *FIPS 206 - FALCON**, FN-DSA (FFT, fast-Fourier transform)
-

Algoritmos Post-Cuánticos

La criptografía post-cuántica (PQC, Post-Quantum Cryptography), referencia a algoritmos criptográficos resistentes a ataques efectuados mediante computación cuántica.

- **FIPS 203 – CRYSTALS-Kyber**, ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism)
- **FIPS 204 – CRYSTALS-Dilithium**, ML-DSA (Module-Lattice-Based Digital Signature Algorithm)
- **FIPS 205 – Sphincs**, SLH-DSA (Stateless Hash-Based Digital Signature Algorithm)
- *FIPS 206 - FALCON**, FN-DSA (FFT, fast-Fourier transform)

Por supuesto, aquí tienes el texto extraído de la tercera imagen:

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 de información cifrada mediante SSL o TLS. Su puerto por defecto es el número 443.
-

SSL (Secure Socks 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 garantizar la 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 Versión 3.0 data de **1996**, definida en la **RFC-6101** por la Internet Engineering Task Force (IETF).

Por supuesto, aquí tienes el texto extraído de la última imagen:

SSL (Secure Socks 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 solo proporcionan autenticación de servidor. La versión 3 agrega la autenticación del cliente, utilizando los certificados digitales de cliente y de servidor.

Aquí tienes el texto extraído de la última imagen:

SSL (Secure Socks Layer)

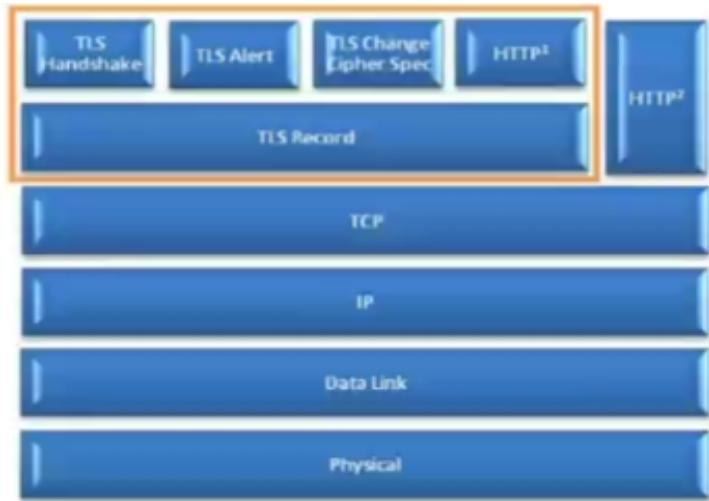
Fases

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.

Aquí tienes el texto extraído de la imagen:

TLS (Transport Layer Security)

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

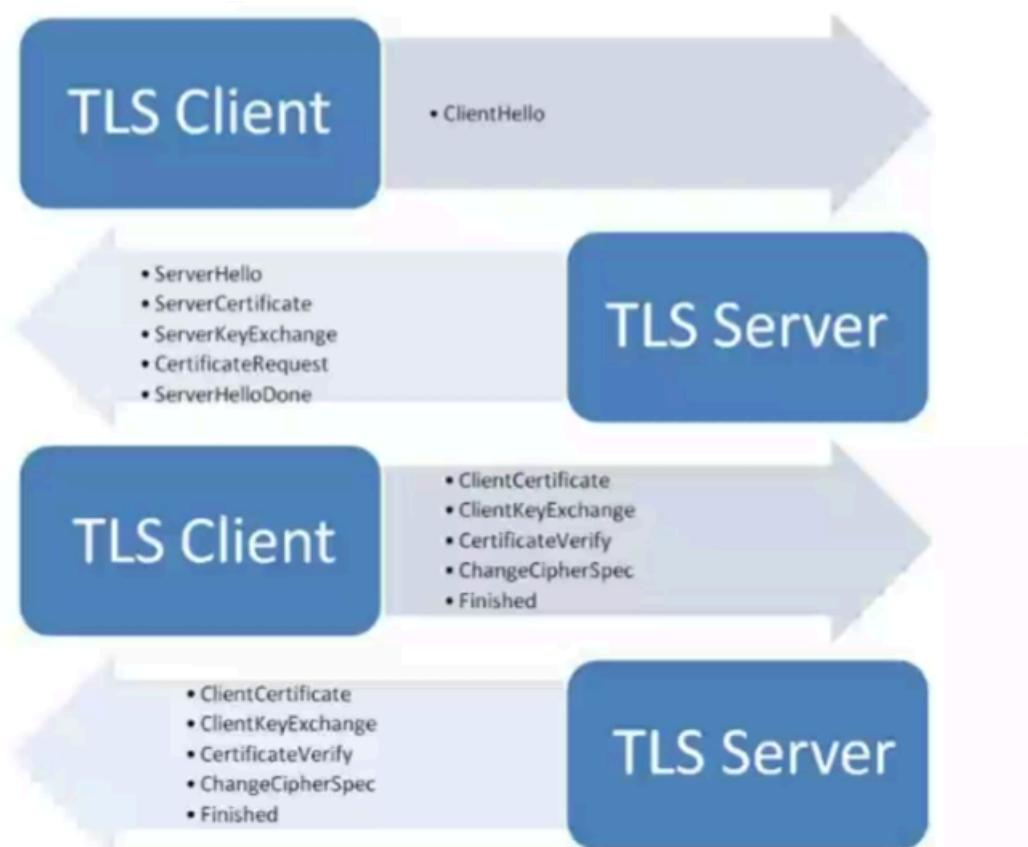


Aquí tienes el texto extraído de la última imagen:

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 firma (hash) de todos los datos intercambiados por ambos interlocutores.



TLS (Transport Layer Security)

Algoritmos utilizados

- Cifrado Asimétrico: RSA, Diffie-Hellman (DHE), Curva Elíptica (ECDHE), DSA (Digital Signature Algorithm).
- Cifrado Simétrico: 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.

Aquí tienes el texto extraído de la última imagen:

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)

Aquí tienes el texto extraído de la última imagen:

TIENE QUE VER CON EL ASIMETRICO

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.

Los algoritmos son lo mismo las diferencias son las leyes que esta detras

Aquí tienes el texto extraído de la última imagen:

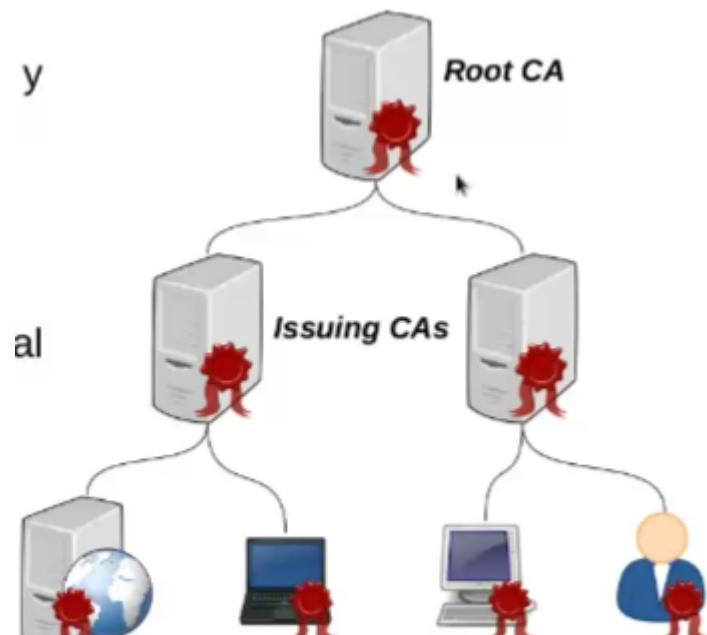
Firma digital - Propiedades

- 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. Cualquier puede comprobar su autenticidad en cualquier momento, de forma sencilla.
-

PKI – Infraestructura de Clave Pública

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 rechazo 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 accesorios; 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 está atado a ningún algoritmo en particular, y contempla los siguientes campos:

- Versión.
- Número 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 único 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.

Las listas de CRL suelen aparecer en internet

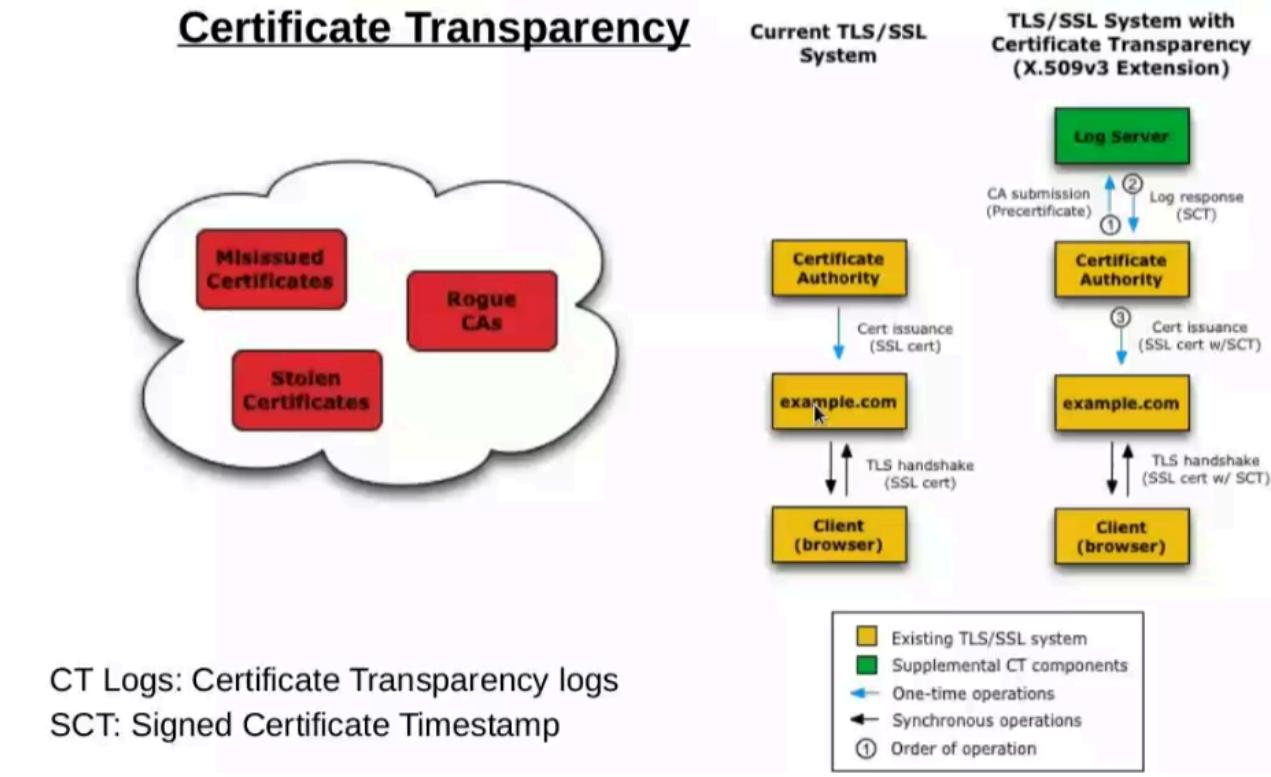
Aquí tienes el texto extraído de la última imagen:

PKI – Infraestructura de Clave Pública

Componentes:

- **Autoridad de Certificación** (CA, Certificate Authority): Emite y revoca certificados, vinculando las claves públicas 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 revocación (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



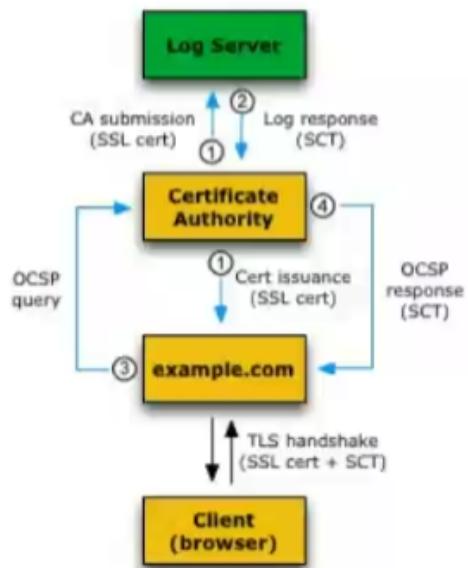
Este trabajo de verificación lo hacen los navegadores

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).

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

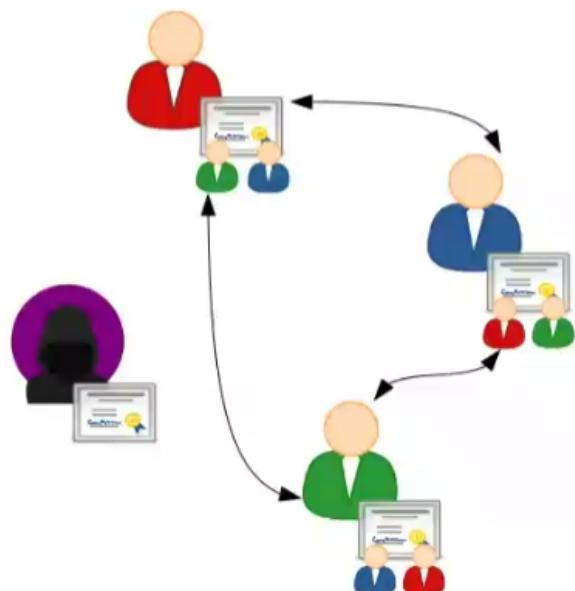
TLS/SSL System with Certificate Transparency (OCSP Stapling)



Aquí tienes el texto extraído de la última imagen:

Anillo o Círculo de confianza

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



Anillo o Círculo 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.

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.

La firma electronica es mas segura pero requiere mas permisos,requerimientos para implementarla y la firma digital tiene el mismo valor que una firma comun por escribania (chequear)

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 procedural 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.

FIN DE LA UNIDAD 3

Clase que viene vamos a generar algunos certificados y mostrarlos en texto plano

pública para cifrar y privada para descifrar. Ejemplo del profe

```
dagger@deepblue: ~/SCAW2025/ciphers $ java DesCipher hola 12345678
Mensaje cifrado: GXR8jbie4qE=
Mensaje descifrado: hola
dagger@deepblue: ~/SCAW2025/ciphers $ java DesCipher hola 123456
Exception in thread "main" java.security.InvalidKeyException: Wrong key size
    at java.base/com.sun.crypto.provider.DESCrypt.init(DESCrypt.java:536)
    at java.base/com.sun.crypto.provider.ElectronicCodeBook.init(ElectronicCodeBook.java:97)
    at java.base/com.sun.crypto.provider.CipherCore.init(CipherCore.java:482)
    at java.base/com.sun.crypto.provider.CipherCore.init(CipherCore.java:400)
    at java.base/com.sun.crypto.provider.DESCipher.engineInit(DESCipher.java:187)
    at java.base/javax.crypto.Cipher.implInit(Cipher.java:867)
    at java.base/javax.crypto.Cipher.chooseProvider(Cipher.java:929)
    at java.base/javax.crypto.Cipher.init(Cipher.java:1299)
    at java.base/javax.crypto.Cipher.init(Cipher.java:1236)
    at DesCipher.encrypt(DesCipher.java:40)
    at DesCipher.main(DesCipher.java:22)
dagger@deepblue: ~/SCAW2025/ciphers $ java DesCipher hola 12345678
Mensaje cifrado: GXR8jbie4qE=
Mensaje descifrado: hola
dagger@deepblue: ~/SCAW2025/ciphers $ vi AesGcmCipher.java
dagger@deepblue: ~/SCAW2025/ciphers $ vi SHA3Hashing.java
dagger@deepblue: ~/SCAW2025/ciphers $ java SHA3Hashing hola
Mensaje: hola
SHA-3 Hash (Hex): 8af13d9244618eee876d0431f3449aa4ff95274ca3e7e5c6541979499f5b85de
dagger@deepblue: ~/SCAW2025/ciphers $ vi ChaCha20Cipher.java
dagger@deepblue: ~/SCAW2025/ciphers $ java ChaCha20Cipher hola 1234567890123456
La clave debe tener 256 bits
dagger@deepblue: ~/SCAW2025/ciphers $ java ChaCha20Cipher hola 12345678901234567890123456789012
Utilizando clave provista por el usuario...
Mensaje cifrado: ef21Vda4YLyeTpTomHmf+A==
Mensaje descifrado: hola
dagger@deepblue: ~/SCAW2025/ciphers $ vi R
dagger@deepblue: ~/SCAW2025/ciphers $ vi RSACipher.java
dagger@deepblue: ~/SCAW2025/ciphers $ java RSACipher hola
Mensaje cifrado: P/qAAD083dm03CLtdkg9pIk7B81U0vUObRCvShIVyuqJEQvbUngvo5ZLe2LYszVP2/1bjI2905KpsEv8tFlQ5T0JAhN5/33dQboh3AXD
mJM5Q+56n9x7wCLRh+H/MoPLUpiCsPeVqiIBbDt0kJVWk7urRXBVNA4jT/tfcabxpIeqYP+E FouzKHoyw7Gaf6kyGL0hQadRKN0s2oGJCpeh3na2XcW0gWla
51cRU8sJIQHVJ9YD6e0vumfqWRgrlUdevnSH6/Zjaat2d67/vGrwb0yab62UznPKyVJzhdqrguv4qRaqlQMFAwMDG30CTS++CdmnRV0iAzhH8Te++JQ==
```

