

# Cifrado

Mikel Egaña Aranguren

[mikel-egana-aranguren.github.io](https://mikel-egana-aranguren.github.io)

[mikel.egana@ehu.eus](mailto:mikel.egana@ehu.eus)



# Cifrado

<https://doi.org/10.5281/zenodo.4302267>

<https://github.com/mikel-egana-aranguren/EHU-SGSSI-01>



# Indice

- Introducción
- Esteganografía
- Métodos de encriptación
  - Ataques por fuerza bruta
  - Algoritmos de resumen
  - Contraseñas en Sistemas Operativos
- Encriptación asimétrica

# Introducción

Criptografía: cifrar la información

Mecanismo de seguridad muy antiguo

Asegura

- Confidencialidad (Cifrado)
- Integridad (Algoritmos resumen)
- Autenticidad (Certificados digitales)

# Introducción

Esteganografía: **ocultar** la información

Criptografía: **cifrar** la información

# Introducción

Historia de la Criptografía:

- Hasta 1948, criptografía pre científica
- En 1948, Claude Shannon sienta las bases de la Teoría de la Información y de la criptografía moderna
- En 1976 Diffie & Hellman introducen el concepto de criptografía de clave pública

# Introducción

Criptoanálisis: técnicas para descifrar mensajes encriptados

- Sin conocer la clave
- Obteniendo la clave a partir de uno o varios mensajes encriptados
- El algoritmo es público - [Principio de Kerckhoffs \(1883\)](#)

Criptología: Criptografía + Criptoanálisis

# Introducción

Criptosistema:  $D_K ( E_K ( M ) ) = M$

- M: Conjunto de todos los mensajes sin cifrar
- C: Conjunto de todos los mensajes encriptados (criptogramas)
- K: Conjunto de claves posibles
- E: algoritmo de encriptación
- D: algoritmo de desencriptación

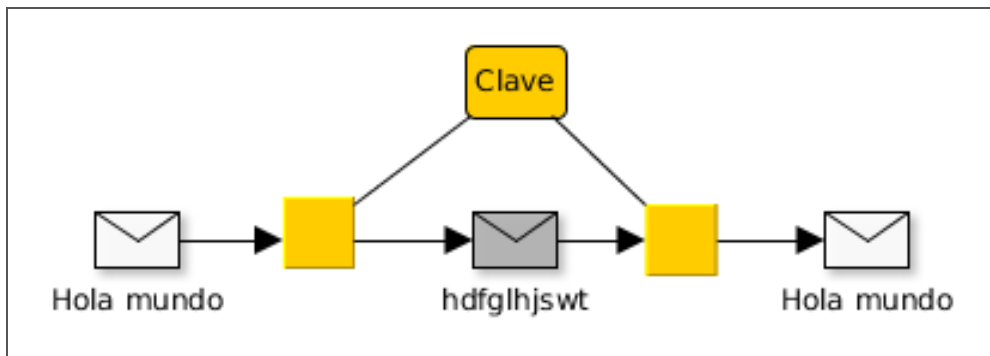


# Introducción

## Criptosistemas

- Simétricos o de clave privada
  - Una clave para encriptar y desencriptar
  - Cifrado en bloque o cifrado en flujo
- Asimétricos o de clave pública
  - Una clave para encriptar y otra para desencriptar
  - Lo que una encripta, la otra lo desencripta

# Criptosistemas de clave privada



# Criptosistemas de clave privada

## Claves débiles

- Pueden presentarse según las características de cada algoritmo
- Claves cuyo comportamiento no es el deseado
  - $E_K(M)=M$
  - $E_K(E_K(M))=M$
  - $D_{K2}(E_{K1}(M))=M$

# Esteganografía

Consiste en ocultar información de forma que no sea “visible” para quien no sepa la clave

Sin saber la clave, puede parecer que no hay información oculta

Es la técnica precursora de la criptografía

# Esteganografía

Histaiaeo (gobernador de Mileto) buscaba aliados para sublevarse contra el rey persa Darío I

Necesitaba enviar mensajes que nadie detectara

- Rapaba el pelo a los mensajeros
- Les grababa el mensaje en la cabeza
- Esperaba a que les creciera el pelo, y los mandaba al destino
- En el destino les volvían a rapar la cabeza y leían el mensaje

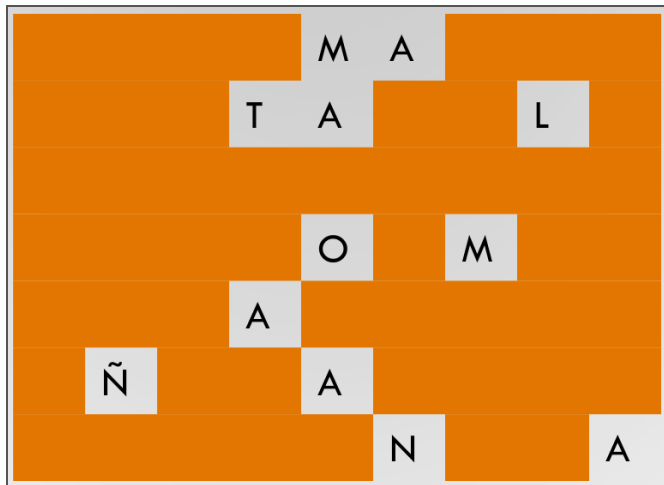
# Esteganografía

Usando una plantilla

La clave es la forma de extraer la información



# Esteganografía



# Esteganografía

Seleccionando unos caracteres determinados

Los asirios tenían amarrados los caballos a anclajes mientras los olmecas sólo ajustaban largos amarres sobre octogonales calesas que se hacían ocultar.

Clave: primera letra de cada palabra no monosílaba

Los **A**sirios **T**enían **A**marrados los **C**aballos a **A**ncclajes **M**ientras los **O**lmecas **S**ólo  
**A**justaban **L**argos **A**marres **S**obre **O**ctogonales **C**alesas que se **H**icían **O**cultar.



# Esteganografía

Ocultación de información en archivos multimedia (normalmente imágenes)

En formato BMP cada pixel en RGB son 3 bytes

LSB (Less Significant Bit): Modificar el último bit de cada byte es inapreciable

# Esteganografía

Por ejemplo, para ocultar texto, insertamos el código ASCII del carácter deseado

A → 65 → 01000001

```
(11011010) (01001001) (01000010)
(00011110) (01011010) (11011110)
(00001110) (01000111) (00000111)
```

# Método de encriptación

## Objetivos

- Convertir el mensaje en ininteligible
- Recuperar la información cifrada
- Implementación lo más sencilla posible

# Método de encriptación

Técnicas básicas en criptografía clásica

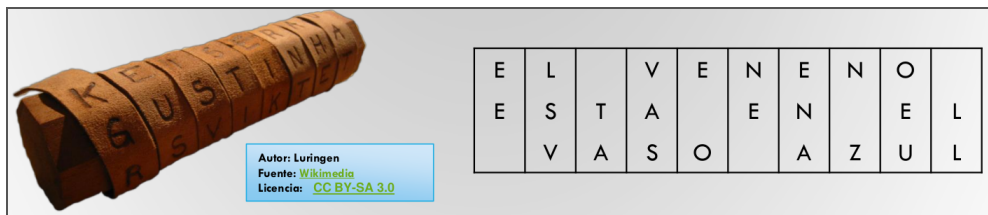
- Transposición (los caracteres originales simplemente cambian de posición)
- Sustitución (los caracteres originales se sustituyen por otros)

# Método de Escitalo de Esparta

Enrollar una tira de papel en un bastón y escribir el mensaje

Desenrollar el papel y enviarlo al destino

# Método de Escitalo de Esparta



EE\_LSV\_TAVASE\_ONE\_ENAN\_ZOEU\_LL

# Método de Escitalo de Esparta

Se necesita un bastón exactamente igual para descifrar el mensaje

Enrollar la tira de papel alrededor del bastón y leer el mensaje

La clave de este sistema es el diámetro del bastón

# Método de Escitalo 2.0

Distribuir el mensaje en columnas


La clave viene determinada por la cantidad y orden de las columnas



# Método de Escitalo 2.0

Clave 32154

1	2	3	4	5
E	L		P	E
R	R	O		D
E		S	A	N
	R	O	Q	U
E		N	O	T
I	E	N	E	
R	A	B	O	.



3	2	1	5	4
	L	E	E	P
O	R	R	D	
S		E	N	A
O	R		U	Q
N		E	T	O
N	E	I		E
B	A	R	.	O

\_OSONNBLR\_R\_EAERE\_EIR\_EDNUT\_.P\_AQOEO

# Método de Escitalo 2.0

## Criptoanálisis

- Basado en combinatoria
- Calcular el tamaño de los bloques
- Combinar los bloques en distinto orden hasta encontrar alguno con sentido

# Método de Atbash (Espejo)

Cifrado monoalfabético

Técnica proveniente del alfabeto hebreo

Consiste en sustituir cada carácter por su "contrario"

# Método de Atbash (Espejo)

a	b	c	d	e	f	g	h	i	j	k	l	m	n	ñ	o	p	q	r	s	t	u	v	w	x	y	z
Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A

Quedamos a las dos → Jfvwzñlh z ozh wlh

# Método César

Cifrado monoalfabético

Empleado por Julio César

Consiste en sumar 3 a la posición de cada letra en el alfabeto

# Método César

a	b	c	d	e	f	g	h	i	j	k	l	m	n	ñ	o	p	q	r	s	t	u	v	w	x	y	z
D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Los galos se resisten → Ñrv jdñrv vh uhvhwph

# Método Afín

Cifrado monoalfabético

Generalización método César

$$E_{(a;b)}(M) = (aM + b) \bmod N$$

N es el número de caracteres del alfabeto

César es una transformación afín con  $E(1,3)$

# Método Diccionario

Cifrado monoalfabético

Generar la tabla de correspondencias de manera "manual"

a	b	c	d	e	f	g	h	i	j	k	l	m	n	ñ	o	p	q	r	s	t	u	v	w	x	y	z
K	V	D	M	J	L	E	A	N	T	F	Q	X	Z	B	P	Y	R	O	G	C	I	Ñ	S	H	W	U

Desordenado

a	b	c	d	e	f	g	h	i	j	k	l	m	n	ñ	o	p	q	r	s	t	u	v	w	x	y	z
M	U	R	C	I	E	L	A	G	O	B	D	F	H	J	K	N	Ñ	P	Q	S	T	V	W	X	Y	Z

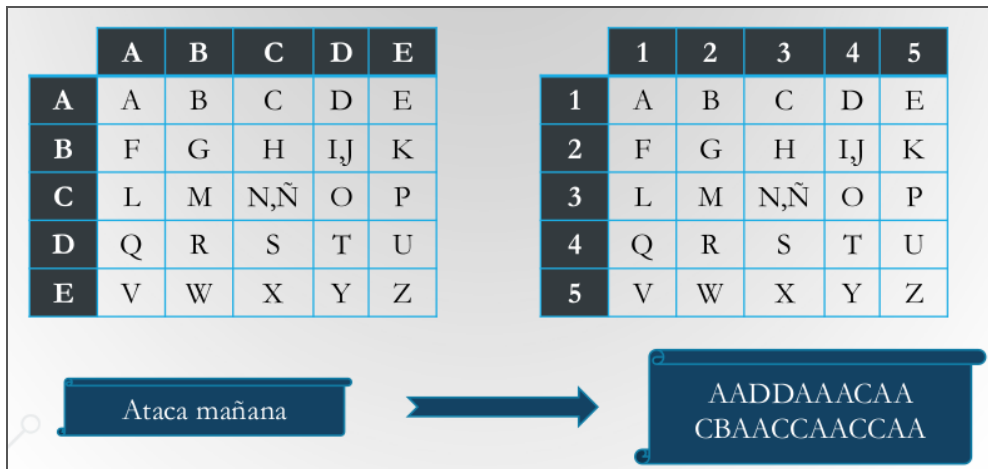
En base a una palabra



# Método Polybius

Cifrado monoalfabético

Pueden ser caracteres o dígitos



# Métodos de Sustitución monoalfabéticos

Criptografía basado en estadística

Método establecido por Al-Kindi en el siglo 9

Un carácter "original" siempre se sustituye por el mismo carácter/es

Se sabe cuáles son los caracteres más frecuentes en cada idioma

Se sabe las palabras de dos/tres/cuatro caracteres (bigramas, trigramas y tetragramas) más frecuentes en cada idioma

# Métodos de Sustitución monoalfabéticos

Se va "probando" y deduciendo

Cuanto más largo es el texto cifrado, mejor

Hay que saber el idioma del texto original

# Métodos de Sustitución monoalfabéticos

Porcentaje de aparición de caracteres en castellano

e - 16,78%	r - 4,94%	y - 1,54%	j - 0,30%
a - 11,96%	u - 4,80%	q - 1,53%	ñ - 0,29%
o - 8,69%	i - 4,15%	b - 0,92%	z - 0,15%
l - 8,37%	t - 3,31%	h - 0,89%	x - 0,06%
s - 7,88%	c - 2,92%	g - 0,73%	k - 0,00%
n - 7,01%	p - 2,776%	f - 0,52%	w - 0,00%
d - 6,87%	m - 2,12%	v - 0,39%	

Ejemplo de descifrado por análisis de frecuencias

# Métodos de Sustitución monoalfabéticos

Técnicas para dificultar el criptoanálisis

- Eliminar los espacios en blanco
- Alterar el texto original manteniendo su significado (Ej. SMS, WhatsApp, ...)
- Usar pictogramas con significado (Libro de códigos)
- Evitar la correspondencia 1-1 usando el mismo carácter en más de una ocasión (Sistemas polialfabéticos)

# El disco de Alberti

Primer sistema polialfabético

Dos discos concéntricos, el interior móvil

Durante el cifrado, se va moviendo, por lo que en el cifrado se usan X alfabetos (correspondencias) distintas

La clave es la posición inicial, cada cuántos caracteres se gira el disco, cuánto se gira y en qué dirección

# El disco de Alberti

The Alberti and Jefferson Code Disks



# La máquina enigma

Es probablemente el elemento criptográfico más conocido de la historia

Originalmente diseñada para uso civil

Modificada para uso militar y usada por los nazis



# La máquina enigma

158,962,555,217,826,360,000 (Enigma Machine) - Numberp...



# La máquina enigma

El matemático polaco Marian Rejewski estableció las bases para descryptar Enigma

- Crearon máquinas electromecánicas llamadas "bombas"
- Los nazis añadieron 2 nuevos rotores y las "bombas" polacas no daban abasto con el nuevo número de posibilidades

# La máquina enigma

El equipo de [Alan Turing](#) partió de esta información para crear su propia "bomba" más eficiente y resistente a cambios de configuración

Flaw in the Enigma Code - Numberphile



# Métodos de sustitución polialfabéticos

## Criptoanálisis

- Usando métodos estadísticos
- Se buscan patrones para deducir tamaños de las claves, orden de los distintos trozos, etc.
- Hace falta más texto encriptado que en los sistemas monoalfabéticos

# Métodos de cifrado de flujo

En vez de cifrar un mensaje, cifran bit a bit

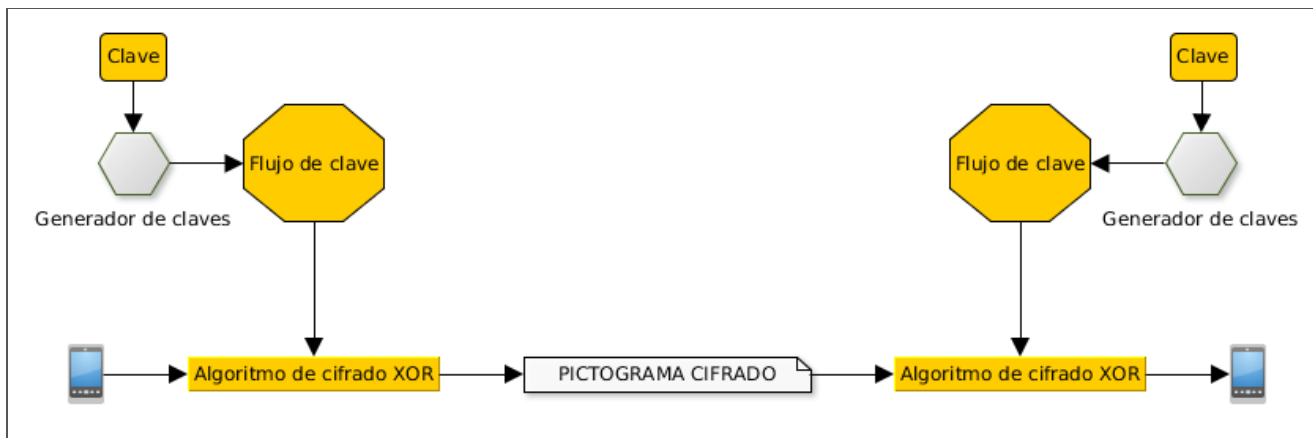
Usado para comunicaciones en tiempo real (No se puede esperar a tener el mensaje completo para cifrarlo y transmitirlo)

# Métodos de cifrado de flujo

A partir de la clave, se usa un generador "pseudoaleatorio" que genera el flujo de clave

La operación XOR entre el bit a cifrar y el flujo de clave, genera el criptograma

# Métodos de cifrado de flujo



# Método de Vernam

Realiza el XOR entre el texto y una clave aleatoria de la misma longitud

El generador es realmente aleatorio

La clave (el flujo de clave) es lo denominado "libreta de un solo uso":

- Sólo se usa una vez
- Hay que enviársela al receptor del mensaje
- Está demostrado matemáticamente que es irrompible



# Otros métodos de cifrado de flujo

Basados en el método de Vernam

Usar claves pseudoaleatorias generadas a partir de una semilla y un algoritmo de generación

Con la semilla y el algoritmo de generación se podría reconstruir la clave pseudoaleatoria (Depende del número de posibles semillas distintas)

# Otros métodos de cifrado de flujo

No son matemáticamente irrompibles

Ejemplos:

- RC4 (ARC4) usado en TLS/SSL , WEP y WPA entre otros (Roto)
- A5/1 (y distintas versiones) usado en comunicaciones GSM (A5/1 y A5/2 rotos)

# Métodos de cifrado por bloques

Partir el mensaje original en bloques de tamaño fijo:

- Si el tamaño es suficientemente pequeño, puede considerarse cifrado de flujo
- Existen algoritmos de rellenado para aquellos casos en los que el tamaño del mensaje no sea múltiplo del tamaño del bloque

# Métodos de cifrado por bloques

Cada bloque de mensaje original genera un bloque de mensaje cifrado

Se pueden añadir iteraciones, permutaciones y operaciones entre los distintos bloques

# Métodos de cifrado por bloques

DES (Data Encryption Standard) - 1975:

- Bloques de 64 bits
- Clave de 56 bits ( longitud real 64 bits )
- 16 rondas
- Se puede romper en menos de 24 horas

# Métodos de cifrado por bloques

Triple DES - 1998: Basado en realizar 3 ejecuciones de DES (Cifrar – Descifrar - Cifrar):

- Usar 2 claves:  $(E_{k_1} (D_{k_2} (E_{k_1})))$
- Usar 3 claves:  $(E_{k_1} (D_{k_2} (E_{k_3})))$
- Usado extensamente en comercio (tarjetas de crédito)

# Métodos de cifrado por bloques

IDEA - 1991:

- Bloques de 64 bits
- Clave 128 bits
- 8 rondas y media
- Está considerado seguro (excepto algunas claves débiles)
- Su uso no está extendido por estar patentado

# Métodos de cifrado por bloques

KASUMI (A5/3) – 2000:

- Bloques de 64 bits
- Clave de 128 bits
- 8 rondas
- Usado en las redes 3G (con alguna variación)
- El original se puede [romper fácilmente](#) (probado en 2010)



# Métodos de cifrado por bloques

AES (Advanced Encryption Standard) - 2001:

- Muy usado en todo tipo de comunicaciones y transacciones
- Bloques de 128 bits
- Claves de 128, 192 ó 256 bits
- 8 rondas (claves de 128) , 12 rondas (claves de 192), 14 rondas (claves de 256)

# Métodos de cifrado por bloques

Redes 4G:

- Usa pares de algoritmos (si uno comprometido, el otro sigue protegiendo)
- Sufijo EEA para algoritmos que aseguren la Confidencialidad
- Sufijo EIA para algoritmos que aseguren la Integridad

# Ataques por fuerza bruta

Siempre encuentra la solución

Consiste en probar todas las claves posibles

Hay que conocer el algoritmo de cifrado y el espacio de claves

No siempre es posible por su coste temporal

# Ataques por fuerza bruta

Espacio de claves:

- 56 bits:  $2^{56}$  posibilidades
- 128 bits:  $2^{128}$  posibilidades
- 256 bits:  $2^{256}$  posibilidades

# Ataques por fuerza bruta

Tiempo que se tardaría con un superordenador:

- 56 bits: 0,04 segundos
- 128 bits: 7.193.522.047 milenios (año arriba, año abajo)
- 256 bits: ...

# Ataques por fuerza bruta

Se puede hacer un ataque por fuerza menos bruta y más "inteligente":

- Usando un diccionario
- Usando datos del dueño de la clave
- ...

# Algoritmos de resumen

Funciones de dispersión (one-way hash)

Generan un criptograma de un determinado tamaño que representa todo el contenido original (Si cambia lo más mínimo, el resumen criptográfico es distinto)

No tienen inversa

No se puede descifrar

# Algoritmos de resumen

Utilidad:

- Certificar la integridad de la información
- Almacenar claves
- Implementar la firma digital



# Algoritmos de resumen

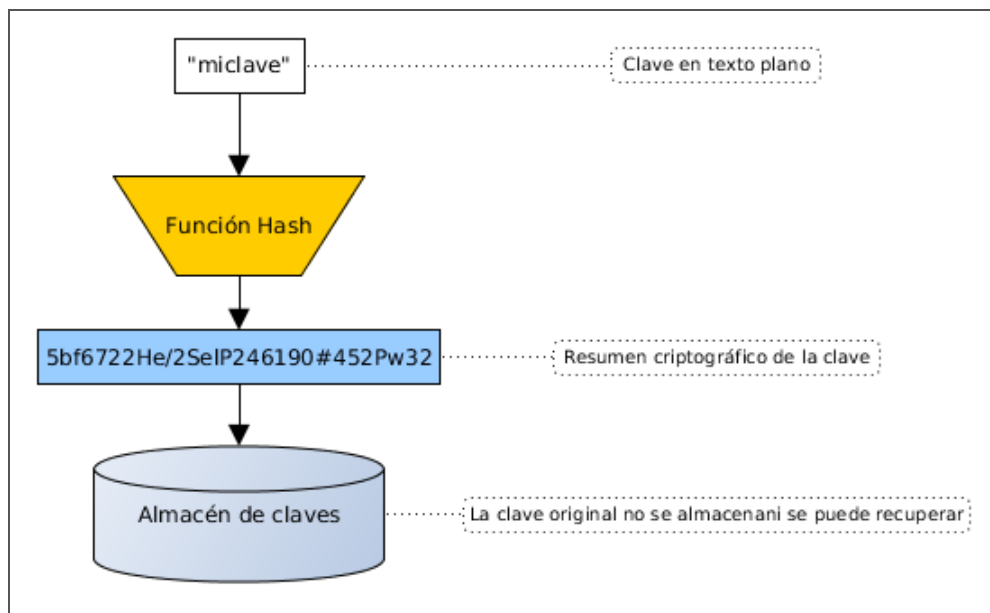
Certificar la integridad de la información

<http://ftp.mozilla.org/pub/mozilla.org/xulrunner/releases/2.0/MD5SUMS>

```
5acef7cc816691f5c8726731ee0d8bdf ./runtimes/xulrunner-2.0.en-US.linux-i686.tar.bz2
cb0dc6ff5304b325098fc8910057884f ./runtimes/xulrunner-2.0.en-US.linux-x86_64.tar.bz2
aa40c07c8669a04170c7501023133acb ./runtimes/xulrunner-2.0.en-US.mac-pkg.dmg
38e5c5ad08927278ed6c333aef836882 ./runtimes/xulrunner-2.0.en-US.win32.zip
1ec6039ee99596551845f27d4bc83436 ./sdk/xulrunner-2.0.en-US.linux-i686.sdk.tar.bz2
101eb57d3f76f77e9c94d3cb25a8d56c ./sdk/xulrunner-2.0.en-US.linux-x86_64.sdk.tar.bz2
cf56e216a05feed16cb290110fd89802 ./sdk/xulrunner-2.0.en-US.mac-i386.sdk.tar.bz2
ac2ddb114107680fe75ee712cddf1ab4 ./sdk/xulrunner-2.0.en-US.mac-x86_64.sdk.tar.bz2
5cfa95a2d46334ce6283a772eff19382 ./sdk/xulrunner-2.0.en-US.win32.sdk.zip
0f4876068fa922498d62abf7d293c9c4 ./source/xulrunner-2.0.bundle
a3b387489ba1738ea504e83cb811c82a ./source/xulrunner-2.0.source.tar.bz2
```

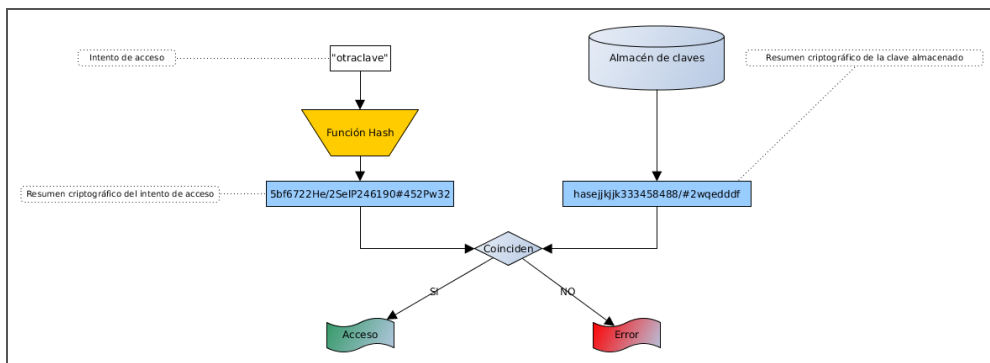
# Algoritmos de resumen

## Almacenar claves



# Algoritmos de resumen

## Identificación



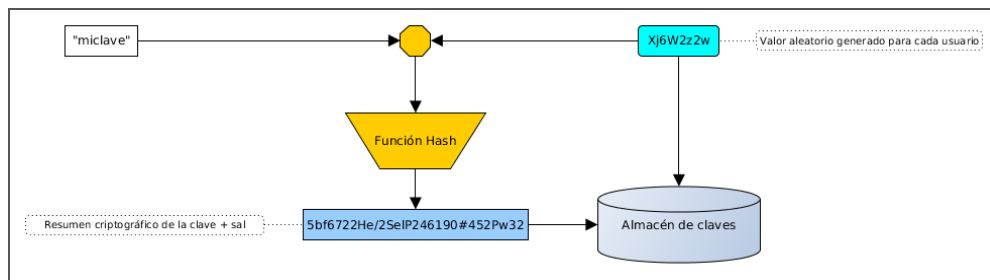
# Algoritmos de resumen

## Problemas

- Todo el mundo con la misma clave tiene el mismo hash
- Se pueden precalcular los hash de todo el espacio de claves

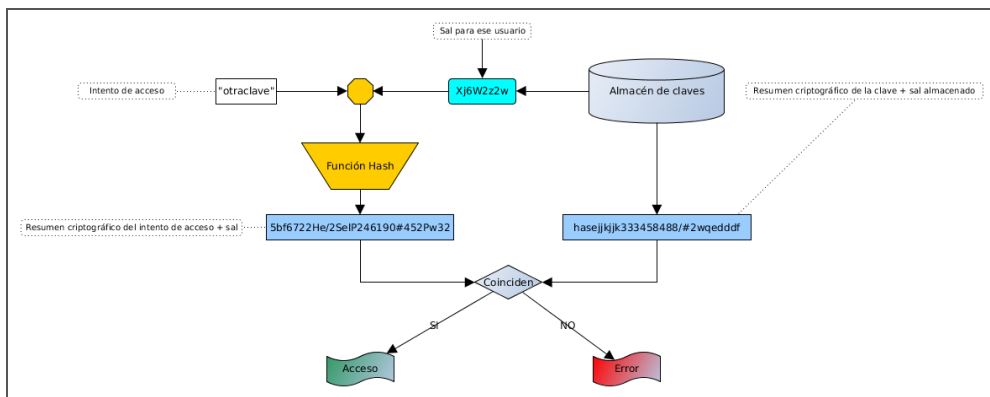
# Algoritmos de resumen

Solución: usar "sal" (Salt), o semilla



# Algoritmos de resumen

## Identificación con sal



# Algoritmos de resumen

Ventajas del uso de sal

- La misma clave tiene una codificación distinta cada vez
- Dificulta los ataques por fuerza bruta

Si roban la BBDD con las claves y la sal, no aporta nada

# Algoritmos de resumen

Funciones de dispersión más usadas

- MD5
- SHA-1



# Algoritmos de resumen

## Problemas

- Colisiones: dos textos distintos que generen el mismo resumen
- Ataques que debilitan el algoritmo

## Soluciones

- Usar otros algoritmos que generan resúmenes criptográficos más largos
- SHA-224, SHA-256, SHA-384, SHA-512, ...

# Contraseñas en S.O.

Linux:

- Ubicación: `etc/shadow`
- Obtención: `sudo cat /etc/shadow`
- Formato: `user:$Algoritmo usado$sal$Resumen Criptográfico:A:B:C:D:E:F:`

# Contraseñas en S.O.

Formato Linux:

- Algoritmo usado: 1: MD5; 2: Blowfish; 3: NT; 5: SHA-256; 6: SHA-512
- Sal: Cadena aleatoria para derivar las claves

# Contraseñas en S.O.

Formato Linux:

- A: número de días sin cambiar la clave (desde 01/01/1970)
- B: número de días hasta poder cambiar la clave
- C: número máximo de días que se puede estar sin cambiar la clave

# Contraseñas en S.O.

Formato Linux:

- D: número de días de antelación con el que hay que avisar al usuario de que tiene que cambiar la clave
- E: número de días desde que caduque la contraseña hasta que se desactive la cuenta
- F: número de días hasta que la cuenta se desactive (desde 01/01/1970)

# Criptografía de clave pública

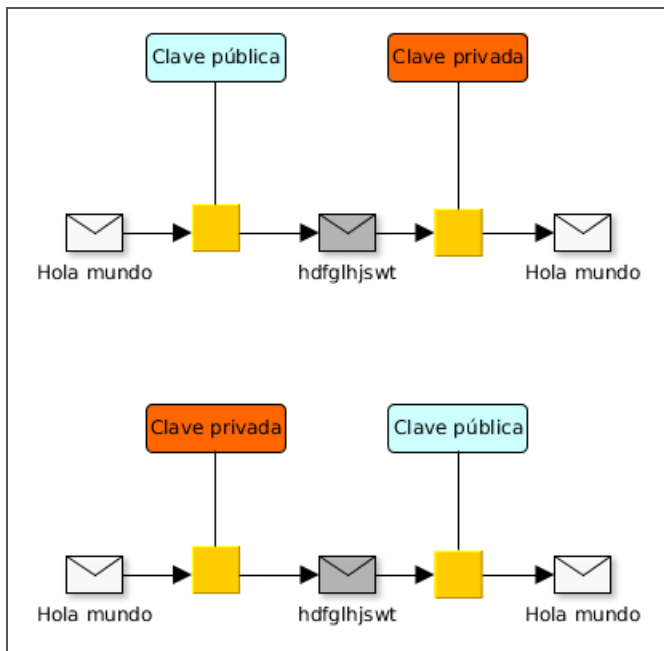
Usa algoritmos de clave asimétrica: la clave que cifra no es la que descifra

Usa dos claves por usuario:

- La clave pública que conoce todo el mundo
- La clave privada que sólo conoce el usuario

Lo que una clave cifra sólo lo puede descifrar la otra

# Criptografía de clave pública



# Criptografía de clave pública

- Iker tiene su clave privada  $I_{\text{privada}}$  y todos tienen la clave pública de Iker,  $I_{\text{pública}}$
- Miren cifra su mensaje  $m$  usando la clave pública de Iker:  $c = e ( m , I_{\text{pública}} )$
- Miren manda el criptograma  $c$  a Iker
- Iker recibe  $c$
- Iker descifra  $c$  usando su clave privada  $I_{\text{privada}}$ :  $m = d ( c , I_{\text{privada}} )$
- Confidencialidad. Sólo Iker puede descifrar el mensaje



# Criptografía de clave pública

Ventajas:

- Sólo el destinatario puede leer el mensaje
- Sólo hay que almacenar una clave
- Cualquiera puede usar la clave pública para enviar un mensaje confidencial a Iker
- No son necesarios canales seguros para comunicar la clave pública

# Criptografía de clave pública

Problemas:

- La clave privada debe mantenerse privada
- Debería ser (prácticamente) imposible sacar la clave privada a partir de la clave pública
- Cifrado y descifrado son más lentos que en los sistemas de clave secreta
- Miren debe estar segura de que está usando la clave pública de Iker
- Debe ser fácil obtener las claves públicas

# Criptografía de clave pública

Cada usuario genera su par (clave pública, clave privada) y publica la clave pública en un servidor de claves: Key Certification Authority o Key Distribution Center (KDC)

# Criptografía de clave pública

Más problemas:

- ¿Cómo sabe Iker si el mensaje es realmente de Miren?
- Cuando Iker conteste ¿Cómo sabe Miren que el mensaje es realmente de Iker?

# Criptografía de clave pública

- Si Iker lo cifra con su clave privada lo puede descifrar cualquiera ( $I_{\text{pública}}$  la conoce todo el mundo)
- Solución:
  - Iker cifra el mensaje con su clave privada:  $C1 = e ( m, I_{\text{privada}} )$
  - Luego lo vuelve a cifrar con la clave pública de Miren:  $C2 = e ( C1 , M_{\text{pública}} )$

# Criptografía de clave pública

- Sólo Miren puede desenscriptarlo con su clave privada:
  - Confidencialidad: Sólo Miren puede descifrar el mensaje:  $C1 = d ( C2 , M_{privada} )$
  - Autenticidad y No Repudio: Sólo Iker ha podido enviar el mensaje:  $m = d ( C1, I_{pública} )$

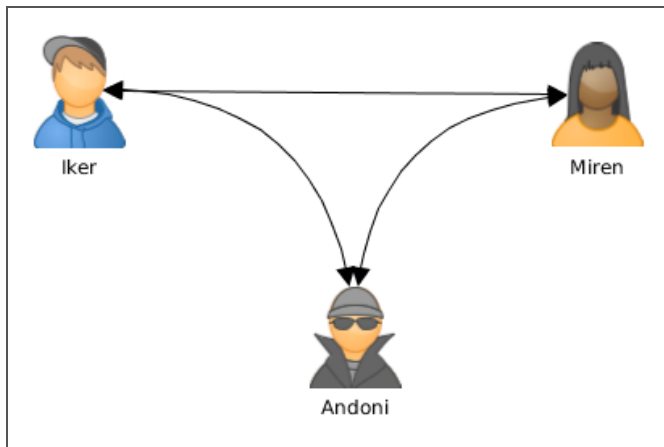
# Criptografía de clave pública

¿Qué ocurre si se interpone alguien en las comunicaciones?

Ataque Man in the middle:

- Un intermediario recibe todos los mensajes sin que las otras partes se enteren
- Se necesita interceptar todas las comunicaciones entre las dos partes

# Criptografía de clave pública

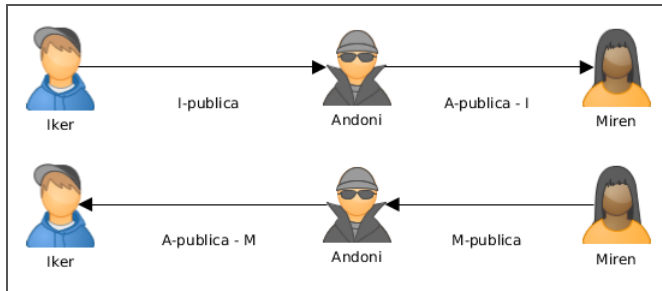




# Criptografía de clave pública

Cuando Iker y Miren quieren comenzar a comunicarse de manera secreta, se intercambian las respectivas claves públicas

Andoni las intercepta y las intercambia por la suya

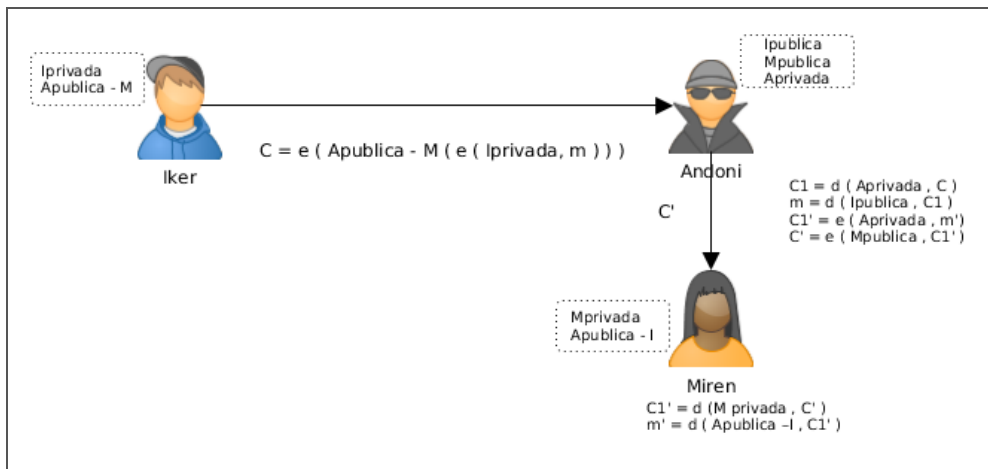


# Criptografía de clave pública

Iker y Miren cifran sus mensajes con la que CREEN la clave pública del otro y con su clave privada

Andoni intercepta los mensajes, los lee, modifica y los encripta con su clave privada

# Criptografía de clave pública



# Criptografía de clave pública

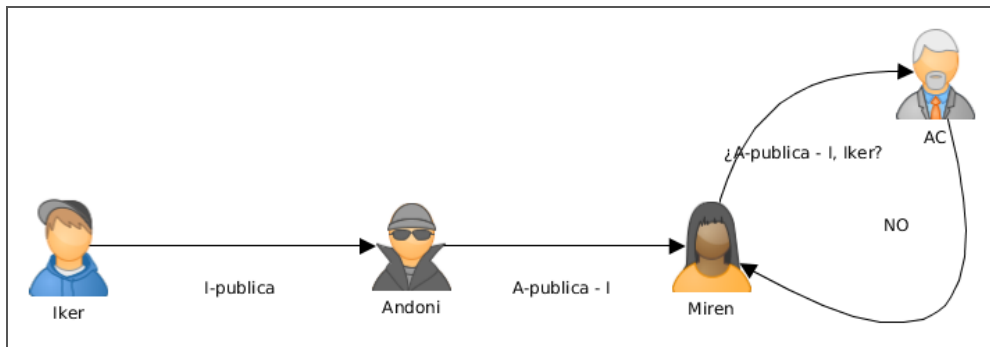
Iker y Miren creen que están comunicándose de manera segura

Andoni está enterándose de todo y modificándolo a su antojo

Formas de evitarlo:

- Paso de claves en canales "seguros"
- Uso de una autoridad que certifique que una clave pública pertenece a quien dice: Autoridad de Certificación (AC)

# Criptografía de clave pública



# Cifrado híbrido

Los sistemas de clave secreta son mucho más rápidos que los de clave pública

Muchas veces se usa una combinación: El sistema de clave pública se usa para compartir una clave secreta  $S$  que sólo se usa una vez

El sistema de clave secreta usa  $S$  para cifrar el mensaje

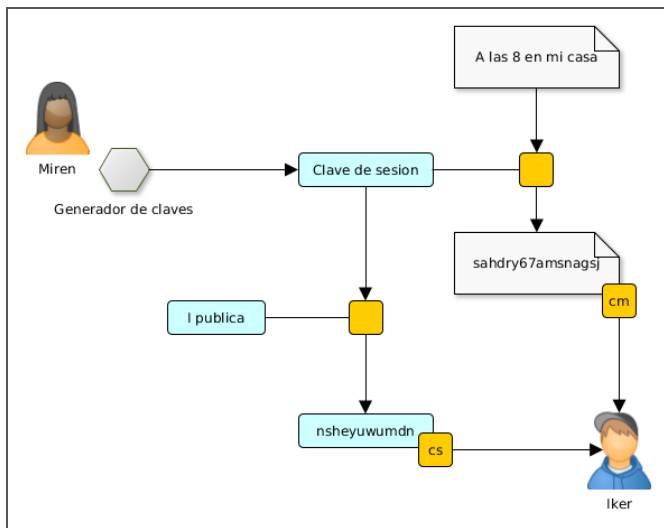
# Cifrado híbrido

Miren genera una clave secreta  $S$  y cifra su mensaje usándola:  $c_m = e_1(m, S)$

Miren cifra  $S$  usando la clave pública de Iker  $c_s = e_2(S, I_{\text{pública}})$

Miren manda  $[c_m, c_s]$  a Iker

# Cifrado híbrido





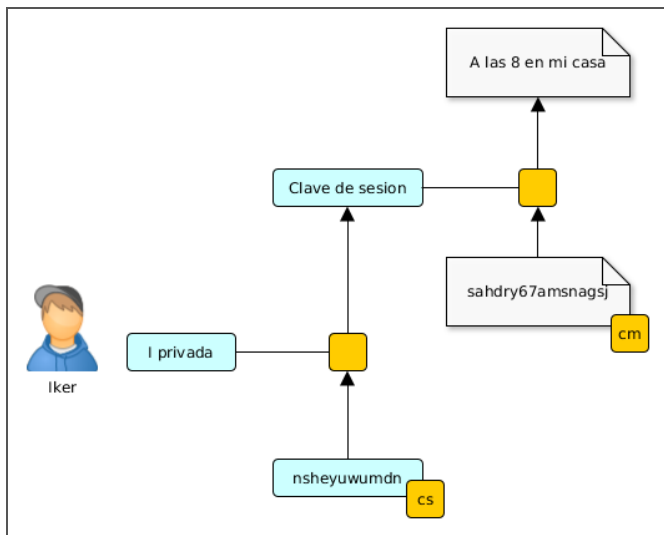
# Cifrado híbrido

Iker recibe [ cm , cs ]

Iker descifra S usando su clave privada  $I_{privada}$ :  $d2 ( cs , I_{privada} ) = S$

Iker descifra m usando S:  $d1 ( cm , S ) = m$

# Cifrado híbrido



# Firma digital

Miren le manda un mensaje a Iker usando un sistema de clave pública

Nadie puede leer el mensaje de Miren a Iker pero cualquiera podría haberlo mandado

¿Cómo sabe Iker que se lo ha mandado Miren o que nadie lo ha modificado?

Solución: Miren firma sus mensajes

# Firma digital

Sólo el usuario legítimo puede firmar su documento

Nadie podrá falsificar una firma

Cualquiera puede verificar una firma digital

# Firma digital

No se puede reutilizar una firma

No se puede modificar una firma

No se puede negar haber firmado un documento

No se puede alterar un documento después de haberlo firmado

Logramos Autenticidad, Integridad y No repudio

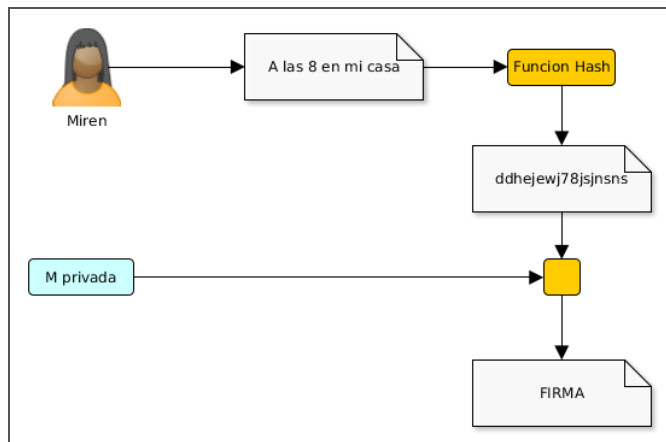
# Firma digital

Miren obtiene un resumen criptográfico del mensaje:  $RC = \text{hash}(m)$

Miren cifra el resumen criptográfico con su clave:  $\text{Firma} = e(RC, M_{\text{privada}})$

Miren envía el mensaje (cifrado o sin cifrar) y su Firma

# Firma digital



# Firma digital

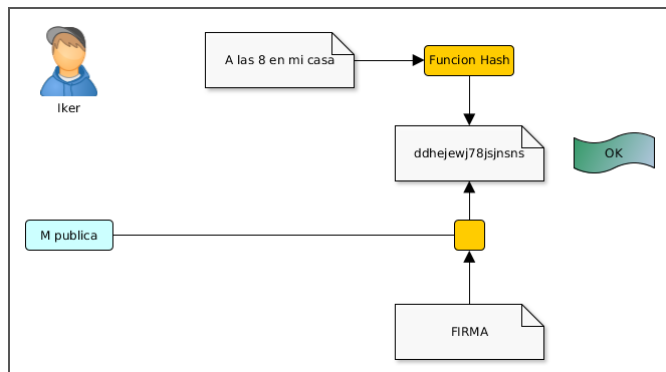
Iker descrypta la Firma usando la clave pública de Miren:  $RC = (Firma, M_{pública})$

Iker obtiene el resumen criptográfico del mensaje:  $RC' = \text{hash}(m)$

Iker compara  $RC'$  con  $RC$  para asegurarse que no ha sido modificado



# Firma digital



# Firma digital

Si además de firmarlo, Miren encripta su mensaje sólo Iker podrá leerlo: Se logra Confidencialidad, Autenticidad, Integridad y No Repudio

Puede hacerlo usando:

- Un sistema de criptografía asimétrica
- Un sistema de criptografía híbrido

# Firma digital

Un sistema de criptografía asimétrica. Enviaría a Iker:

- Criptograma del mensaje cifrado con  $M_{\text{privada}}$  y con  $I_{\text{pública}}$
- Su Firma digital (el resumen criptográfico cifrado con  $M_{\text{privada}}$  )

# Firma digital

Un sistema de criptografía híbrido. Enviaría a Iker:

- Criptograma del mensaje cifrado con la clave de sesión
- Criptograma con la clave de sesión cifrada con  $I_{\text{pública}}$
- Su Firma digital (el resumen criptográfico cifrado con  $M_{\text{privada}}$ )

# Algoritmos de clave pública

Diffie-Hellman - 1976

RSA - 1977

ElGamal - 1984

DSA - 1991

Curvas elípticas - 1985

# Algoritmos de clave pública

Elliptic Curve Cryptography & Diffie-Hellman



# Algoritmos de clave pública

Encryption and HUGE numbers - Numberphile



# Algoritmos de clave pública

DNI electrónico (DNLe 3.0):

- RSA
- SHA-1 / SHA-256
- TripleDES / AES



# Algoritmos de clave pública

PGP:

- RSA / DSA
- IDEA / TripleDES

# Algoritmos de clave pública

SSH:

- RSA / DSA

SSL / TLS:

- RSA / DSA / Diffie-Hellman
- IDEA / DES / TripleDES / AES

# Confianza de firmas

Aunque utilicemos firmas digitales:

- ¿Cómo sabemos que la firma es de quien dice ser?
- ¿Cómo nos asegura una autoridad de certificación que una firma es de quien dice ser?
- ¿No podemos fiarnos de una firma que no esté avalada por una autoridad de certificación?

# Confianza de firmas

- Se usa en PGP, GnuPG y similares
- Un usuario certifica (firmando con su clave privada) que la clave pública de otro usuario es de confianza
- La confianza se propaga según la confianza que demos a los usuarios que firmen las claves

# Niveles de confianza

- Desconocido: no nos fiamos de nada que firme ese usuario (por desconocimiento)
- Ninguno: no nos fiamos de nada que firme ese usuario (porque sabemos que lo hace mal)
- Marginal: nos fiamos de las claves firmadas por dos usuarios con confianza marginal
- Absoluto: nos fiamos de todo lo firmado por ese usuario

# Certificados digitales

- Un certificado digital consiste en que una entidad “de confianza” firme mediante su clave privada, la clave pública de un usuario
- Sirve para certificar que el usuario es quien dice ser
- Depende de la confianza en la entidad que lo certifica

# Certificados digitales

- Siguen el estándar X.509
- Validez != Confianza
  - Validez: cumple los requisitos de una firma (caducidad, etc.)
  - Confianza: nos podemos fiar de esa firma
- Una firma puede ser válida, pero no de confianza
- Una firma de confianza que no sea válida no tiene sentido

# Certificados digitales

Una autoridad de certificación (AC) certifica la validez de una firma

- Prestadores de Servicios de Certificación (PSC): Ley de Firma Electrónica (Ley 59/2003, LFE), Ley de Acceso Electrónico de los Ciudadanos a los Servicios Públicos (Ley 11/2007, LAESCP)
- Los PSCs deben proporcionar un método de consulta de la vigencia de sus certificados: Sólo las Administraciones Públicas tienen la obligación de que sea gratuito



# Certificados digitales

Jerarquía de certificación (RFC 1422)

Internet Policy Registration Authority (IPRA) >> Policy Certification

Authorities (PCA) >> Certification Authorities (CA): Verisign, Thawte, GeoTrust,  
RapidSSL, DigiCertSSL

# Un AC debe

- Mantener una base de datos de nombres distinguidos (ND) y de ACs subordinadas
- Permitir la revocación de certificados:
  - Clave privada del usuario comprometida
  - AC ha emitido un certificado a quien no debía
  - El usuario cambia de AC
  - Violación de la seguridad de la AC
- CRL, Certification Revocation List: ejemplo [GeoTrust](#)

# Un AC debe

- El protocolo OCSP (Online Certificate Status Protocol RFC 2560) permite validar el estado de un certificado digital de manera online
- Es más eficiente que la verificación mediante Listas de Revocación de Certificados (CRL)
- Ventaja: su actualización constante
- Desventaja: necesidad de conexión para la comprobación

# Certificados digitales

Cada AC que proporciona el servicio, mantiene un servidor OCSP

Este servicio responde a las aplicaciones cliente que remitan una petición estandarizada y sepan interpretar la respuesta

# Certificados digitales

Tipos de certificados de clave pública:

- Certificado de autoridad
- Certificado de servidor
- Certificado personal
- Certificado de productor de software

# Certificados digitales

Componentes de un certificado:

- Versión
- Número de serie
- Identificador del algoritmo de firmado
- Nombre del emisor
- Periodo de validez
- Nombre del sujeto

# Certificados digitales

Con un certificado digital conseguimos:

- Confidencialidad al poder encriptar la información
- Integridad al poder realizar hash de la información y poder firmarla
- Autenticidad al venir firmada la información
- No repudio al firmar la información

# Criptografía cuántica

Quantum Cryptography in 6 Minutes

