

IC

Criptografía

Índice de temas

- Encoding
- Criptografía
- Hashing
- Ofuscación / Esteganografía (para ir mirando)

Codificación

- El encoding de caracteres es usado para representar un repertorio de caracteres en algún tipo de sistema de codificación.
- En ocasiones nos referimos al encoding con la conversión de la representación de los caracteres de un sistema de codificación a otro.

Algunos ejemplos de sistemas de codificación

- ASCII
- Representación numérica en distintas otras bases
- Unicode
- EBCDIC
- Morse
- Braille
- Lenguaje de señas
- Base64

ASCII

American Standard Code for Information Interchange es un metodo de codificación de 7-bit character usado en Windows, UNIX, and Macintosh machines.

representación numerica

```
- base 2 o binario
  01000011 01101001 01100010 01100101 01110010 01110011 01100101 01100111

- base 10 o decimal      67 105  98 101 114 115 101 103
- base 16 o hexadecimal  0x43 0x69 0x62 0x65 0x72 0x73 0x65 0x67
```

Binario	Dec	Hex	Representación	Binario	Dec	Hex	Representación	Binario	Dec	Hex	Representación
0010 0000	32	20	espacio ()	0100 0000	64	40	@	0110 0000	96	60	`
0010 0001	33	21	!	0100 0001	65	41	A	0110 0001	97	61	a
0010 0010	34	22	"	0100 0010	66	42	B	0110 0010	98	62	b
0010 0011	35	23	#	0100 0011	67	43	C	0110 0011	99	63	c
0010 0100	36	24	\$	0100 0100	68	44	D	0110 0100	100	64	d
0010 0101	37	25	%	0100 0101	69	45	E	0110 0101	101	65	e
0010 0110	38	26	&	0100 0110	70	46	F	0110 0110	102	66	f
0010 0111	39	27	'	0100 0111	71	47	G	0110 0111	103	67	g
0010 1000	40	28	(0100 1000	72	48	H	0110 1000	104	68	h
0010 1001	41	29)	0100 1001	73	49	I	0110 1001	105	69	i
0010 1010	42	2A	*	0100 1010	74	4A	J	0110 1010	106	6A	j
0010 1011	43	2B	+	0100 1011	75	4B	K	0110 1011	107	6B	k
0010 1100	44	2C	,	0100 1100	76	4C	L	0110 1100	108	6C	l
0010 1101	45	2D	-	0100 1101	77	4D	M	0110 1101	109	6D	m
0010 1110	46	2E	.	0100 1110	78	4E	N	0110 1110	110	6E	n
0010 1111	47	2F	/	0100 1111	79	4F	O	0110 1111	111	6F	o
0011 0000	48	30	0	0101 0000	80	50	P	0111 0000	112	70	p
0011 0001	49	31	1	0101 0001	81	51	Q	0111 0001	113	71	q
0011 0010	50	32	2	0101 0010	82	52	R	0111 0010	114	72	r
0011 0011	51	33	3	0101 0011	83	53	S	0111 0011	115	73	s
0011 0100	52	34	4	0101 0100	84	54	T	0111 0100	116	74	t
0011 0101	53	35	5	0101 0101	85	55	U	0111 0101	117	75	u

Unicode

- Unicode es un estándar de codificación de caracteres diseñado para facilitar el tratamiento informático, transmisión y visualización de textos de numerosos idiomas.
- El término Unicode proviene de los tres objetivos perseguidos: universalidad, uniformidad y unicidad.
- La versión 12.1 de Unicode contiene un repertorio de 137994 caracteres
- Unicode puede ser implementado por diferentes codificaciones de caracteres UTF (Unicode Transformation Format): UTF-8 UTF-16, UTF-32.

character	encoding	bits
A	UTF-8	01000001
A	UTF-16	00000000 01000001
A	UTF-32	00000000 00000000 00000000 01000001
あ	UTF-8	11100011 10000001 10000010
あ	UTF-16	00110000 01000010
あ	UTF-32	00000000 00000000 00110000 01000010

EBCDIC

Extended **B**inary **C**oded **D**ecimal
Interchange **C**ode es un esquema de
codificación de 8 bits desarrollado
por IBM en 1963 para mainframes
IBM.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX		PT			GE				FF	CR		
1	DLE	SBA	EUA	1C		NL				EM			DUP	SF	FM	ITB
2							ETB	ESC						ENQ		
3			SYN					EOT					RA	NAK		
4	SP										¢	.	<	(+	
5	&										!	\$	*)	;	¬
6	-	/										,	%	_	>	?
7											:	#	@	'	=	"
8		a	B	c	d	e	f	g	h	i						
9		j	K	l	m	n	o	p	q	r						
A		~	S	t	u	v	w	x	y	z						
B																
C	{	A	B	C	D	E	F	G	H	I						
D	}	J	K	L	M	N	O	P	Q	R						
E	\		S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

Base64

```
nico@nico-Latitude-5421: ~/ic/clases_ic/eje...
$ xxd number_checker_v1.exe | head
00000000: 4d5a 9000 0300 0000 0400 0000 ffff 0000 MZ.....
00000010: b800 0000 0000 0000 4000 0000 0000 0000 .....@.....
00000020: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00000030: 0000 0000 0000 0000 0000 0000 e000 0000 .....
00000040: 0e1f ba0e 00b4 09cd 21b8 014c cd21 5468 .....!..L.!Th
00000050: 6973 2070 726f 6772 616d 2063 616e 6e6f is program canno
00000060: 7420 6265 2072 756e 2069 6e20 444f 5320 t be run in DOS
00000070: 6d6f 6465 2e0d 0d0a 2400 0000 0000 0000 mode....$.
00000080: a338 2339 e759 4d6a e759 4d6a e759 4d6a .8#9.YMj.YMj.YMj
00000090: ee21 de6a e459 4d6a e759 4c6a ad59 4d6a .!.j.YMj.YLj.YMj
$
```

ASCII	M		Z																				
HEXA	4d		5a		90				0														
binario	0	1	0	0	1	1	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
b64 value	19		21		42		16		0		...												
b64 char	T		V		q		Q		A		...												

```
nico@nico-Latitude-5421: ~/Working/Portege/clases_ic...
$ cat number_checker_v1.exe | base64 | head -2
TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAA4AAAAA4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFtIGNhbm5vdCBiZSBydW4gaW4gRE9TIG1v
$
```

Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

Morse

-. .-. . . - -
C I B E R S E G

INTERNATIONAL MORSE CODE

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to five dots.

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 - - - - - . . .

1 . - - - - - - - -

2 . . - - - - - - -

3 . . . - - - - - -

4 - - - - -

5

6 - - -

7 - - - - -

8 - - - - -

9 - - - - -

0 - - - - - - - - - -

Braile

letters

⠠ 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	

BRAILLE ALPHABET

english version


numbers

⠠ 0	⠠ 1	⠠ 2	⠠ 3	⠠ 4
⠠ 5	⠠ 6	⠠ 7	⠠ 8	⠠ 9

punctuation

⠠ +	⠠ -	⠠ *	⠠ :	⠠ /
⠠ =	⠠ .	⠠ (⠠)	⠠ @
⠠ ?	⠠ !	⠠ ,	⠠ #	



Sistema de codificación	Ciberseg
Binario Ascii	01000011 01101001 01100010 01100101 01110010 01110011 01100101 01100111
Decimal Ascii	67 105 98 101 114 115 101 103
Hexadecimal Ascii	43 69 62 65 72 73 65 67
Valor Decimal	4857521860947174759
Morse	-.-. .- -... .- .-... .-.-
Braile	

¿Estamos protegiendo la información?

Ejemplo Vida real

App Cuidar - Encodear no es encriptar

sources > com > globant > pasaportesanitario > utils > token > TokenUtils.java

```
1  package com.globant.pasaportesanitario.utils.token;
2
3  import dev.turingcomplete.kotlinonetimepassword.HmacAlgorithm;
4  import dev.turingcomplete.kotlinonetimepassword.TimeBasedOneTimePasswordConfig;
5  import dev.turingcomplete.kotlinonetimepassword.TimeBasedOneTimePasswordGenerator;
6  import java.util.Date;
7  import java.util.concurrent.TimeUnit;
8  import org.apache.commons.codec.binary.Base32;
9
10 public class TokenUtils {
11     public static TokenInfo getTokenInfo() {
12         TimeBasedOneTimePasswordConfig timeBasedOneTimePasswordConfig = new
13             TimeBasedOneTimePasswordConfig(25, TimeUnit.MINUTES, 8, HmacAlgorithm.SHA1);
14         return new TokenInfo(Integer.parseInt(new TimeBasedOneTimePasswordGenerator(new Base32().
15             decode("JRSWSYI="), timeBasedOneTimePasswordConfig).generate(new Date(System.currentTimeMillis
16                 ()))) & 4095);
17     }
18 }
```

Criptografía

¿Qué es?

La criptografía (del griego “ocultar” y “escribir”), literalmente “escritura oculta”, es el arte o ciencia de cifrar y descifrar información utilizando técnicas que hagan posible el intercambio de mensajes de manera segura de forma tal que sólo puedan ser leídos por las personas a quienes van dirigidos.

Criptografía Clásica

- Muy antigua
- Sin computadoras
- Se ocultaba el algoritmo

Algunos ejemplos

Scytale / Escítala

ROT

Columnar Transposition

Máquina Enigma (2da guerra mundial)

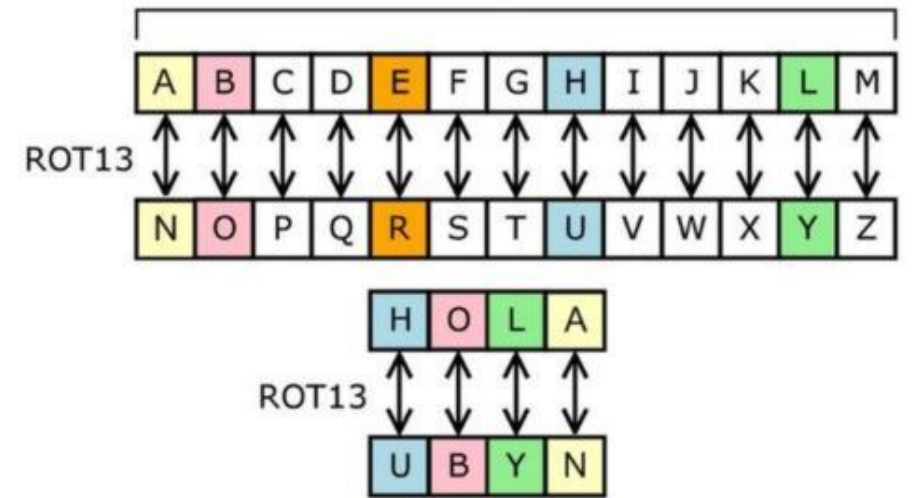
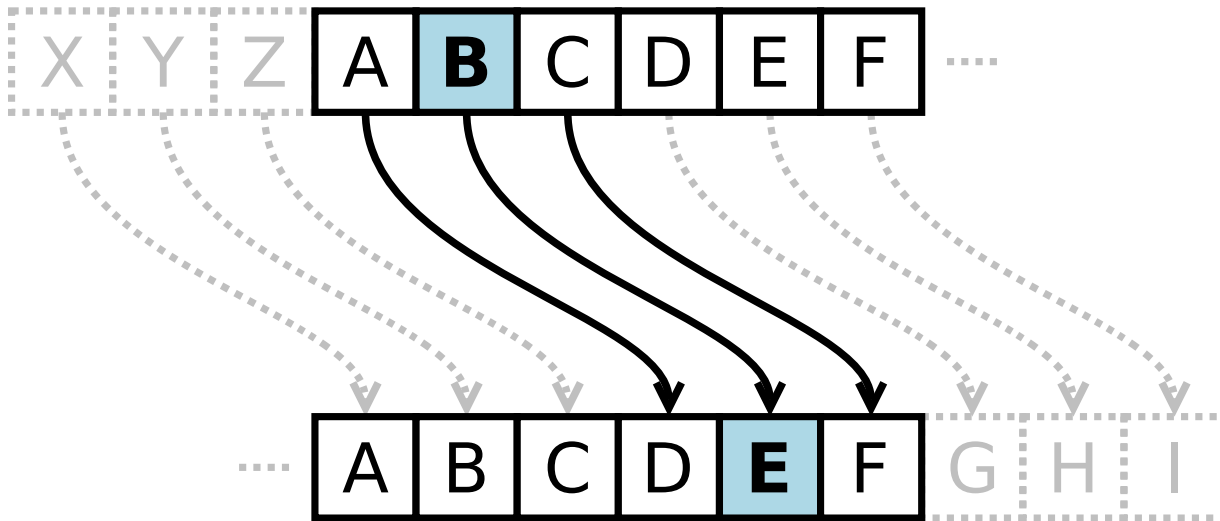
Cripto Clásica - Escítala

En siglo V a.c. los lacedemonios, un antiguo pueblo griego, usaban el método de la escítala para cifrar sus mensajes. El sistema consistía en una cinta que se enrollaba en un bastón sobre el cual se escribía el mensaje en forma longitudinal



ROT 3 - Cifrado Caesar

El cifrado César recibe su nombre en honor a Julio César, que, según Suetonio, lo usó con un desplazamiento de tres espacios para proteger sus mensajes importantes de contenido milita



Cifrado de transposición

En criptografía, un cifrado por transposición es un tipo de cifrado en el que unidades de texto plano se cambian de posición siguiendo un esquema bien definido; las 'unidades de texto' pueden ser de una sola letra, pares de letras, tríos de letras, mezclas de lo anterior.

Ejemplo de transposición columnar

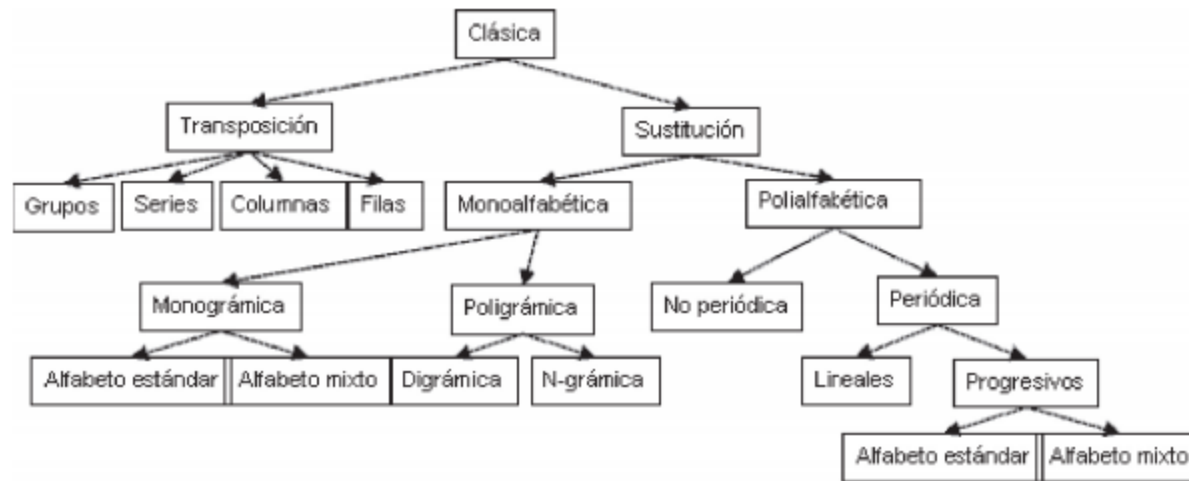
Mensaje cifrado usando la clave CAT

C	A	T
T	H	E
S	K	Y
I	S	B
L	U	E

The sky is blue -> HKSUTSILEYBE

Criptografía clásica

Antes de la computadora era importante ocultar el algoritmo para no comprometer el mensaje.



- ... todo cambió cuando aparecieron las computadoras



Criptografía Moderna

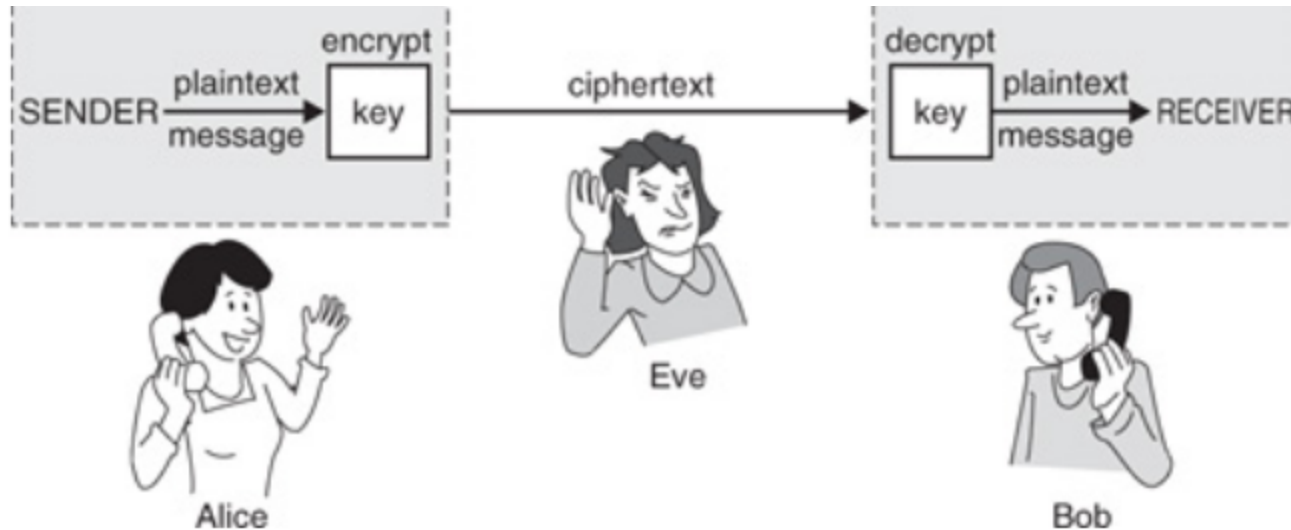
La criptografía es la ciencia encargada de diseñar funciones o dispositivos capaces de transformar mensajes legibles en mensajes cifrados de tal manera que esta transformación (cifrar) y su transformación inversa (descifrar) sólo pueden ser factibles con el conocimiento de una o más **claves** o **llaves**.

Cifrar

Conversión de un mensaje legible (**texto plano** o **plaintext**) a un dato sin sentido aparente (**mensaje cifrado** o **ciphertext**).

Esta conversión requiere del uso de una **clave** o **key**.

El emisor (creador del mensaje cifrado) y los destinatarios deben compartir tanto la técnica de descifrado como la clave a utilizar.



Sistemas de criptografía

Existen dos tipos básicos de criptosistemas modernos:

- **Sistemas de cifrado simétrico** (también conocidos como sistemas de clave secreta o clave privada)
- **Sistemas de cifrado asimétrico** (también conocidos como sistemas de clave pública)

Un concepto muy importante utilizado en criptografía moderna es el de las funciones de HASH.

Funciones de hash

- Son funciones que transforman una entrada y retornan una string de salida de longitud fija, conocido como “message digest”.
- Propiedades de las funciones de hash:
 - Determinístico.
El mismo mensaje siempre da el mismo hash
 - No son reversibles.
No es posible obtener el mensaje original a partir del hash
 - Un pequeño cambio en un mensaje cambia completamente el valor del hash o message digest
 - Es rápido calcular el hash para un mensaje dado

Algunas funciones de hash

```
MD5      (128 bits, RFC 1321)
SHA-1    (160 bits, NIST FIPS 180-2)
SHA-2    (SHA-224, SHA-256, SHA-384, SHA-512)
SHA-3    (SHA-224, SHA-256, SHA-384, SHA-512)
```

MD5

```
$ echo "Ciberseguridad" | md5sum
0e6e40411263311de0fa956a73cb2130 -

$ echo "CiberSeguridad" | md5sum
d5c4752e4a718ee4ad8fc8ba81a833e2 -





$ ls -lh Parrot-security-5.0.1_amd64.iso
-rw-rw-r-- 1 nico nico 4,6G jul 14 14:01 Parrot-security-5.0.1_amd64.iso

$ md5sum Parrot-security-5.0.1_amd64.iso
74ca72645896f83a65acd35ade46b0e0 Parrot-security-5.0.1_amd64.iso
```

Usos de funciones de hash

Las funciones de HASH además de ser muy importantes en la criptografía moderna, otros usos por las cuales son muy requeridas son:

- Implementación de Firma digital (veremos mas adelante)
- Para implementar políticas de almacenamiento de contraseñas que:
 - Hasheen las contraseñas usando SALT.
 - Que compliquen los ataques de fuerza bruta

				
Password	p4s5w3rdz	p4s5w3rdz	p4s5w3rdz	p4s5w3rdz
Salt	-	-	et52ed	ye5sf8
Hash	f4c31aa	f4c31aa	1vn49sa	z32i6t0

Problemas con hashes:

- Las debilidades en una función de hash están asociadas con la posibilidad de manipular las colisiones.
- La rapidez en la generación lo hace vulnerable a ataques de fuerza bruta

¿Cuál usar hoy para almacenar contraseñas?

Los recomendados actualmente por su robustez y dificultad son: Bcrypt, scrypt, PBKDF2 o Argon2.

https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html

Colisiones en función de HASH



Colisión de MD5

```
$md5sum ship.jpg  
253dd04e87492e4fc3471de5e776bc3d  ship.jpg  
  
$ md5sum plane.jpg  
253dd04e87492e4fc3471de5e776bc3d  plane.jpg
```

<https://natmchugh.blogspot.com/2015/02/create-your-own-md5-collisions.html>