Curso CTF Competitivo

Presentación: Raúl Martín

Universidad
Rey Juan Carlos

Criptografía básica

Elia Seco y Santiago Tovar

Universidad
Rey Juan Carlos

Universidad Rey Juan Carlos

Índice

1. Criptografía y codificaciones básicas
   - Representación de los datos
   - Codificaciones y cifrados
   - Otros cifrados (XOR, Dcodefr…)
   - Hashes (MD5, SHA1, SHA256)
2. Retos básicos

Criptografía - Representación de los datos

Es esencial entender que **nos podemos encontrar los datos con diferentes formatos**. Sin embargo, **su significado será el mismo**. Las formas más comunes son:

ASCII

Relaciona caracteres con números. A cada carácter le corresponde un valor de la tabla ASCII.

Hexadecimal

Utiliza base 16 como representación de los datos. En caracteres toma como referencia el valor ASCII

ASCII Table

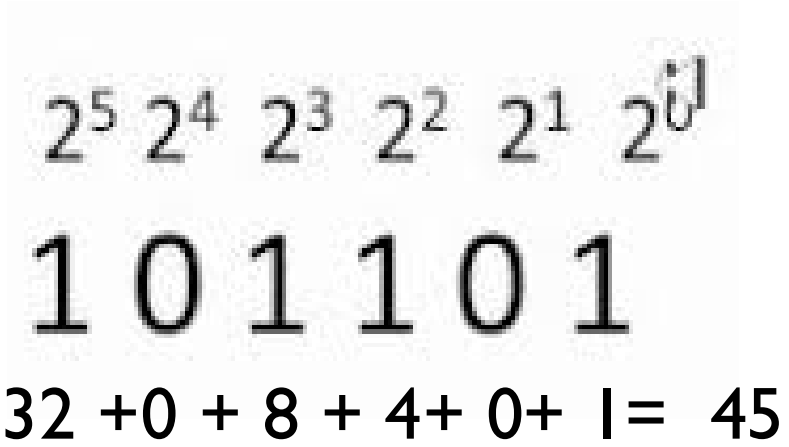
Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

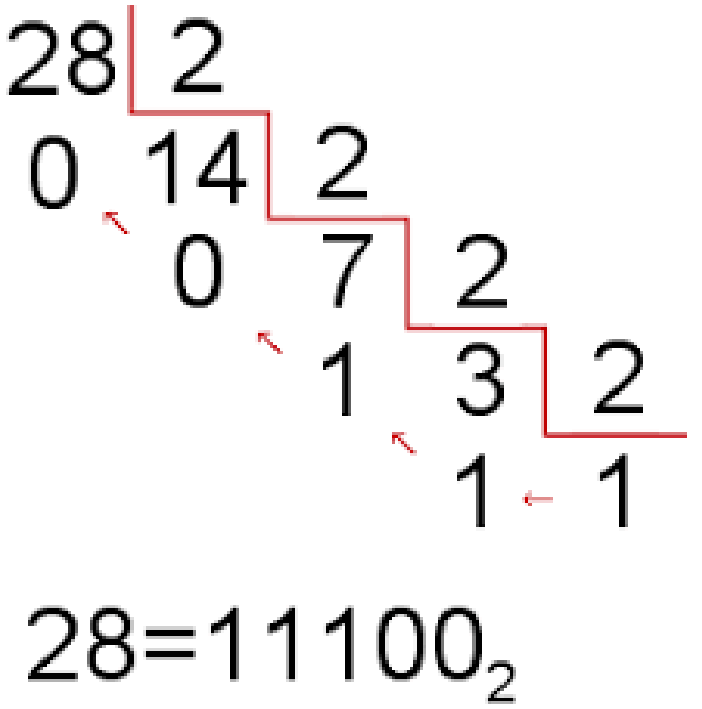
Criptografía - Representación de los datos

Es esencial entender que **nos podemos encontrar los datos con diferentes formatos**. Sin embargo, **su significado será el mismo**. Las formas más comunes son:

Binario

Es la representación más básica. Tan solo utiliza dos valores: 1 y 0.





Palabra

CTF{Bienvenidos}

ASCII

067 084 070 123
066 105 101 110
118 101 110 105
100 111 115 125

Binario

01000011 01010100 01000110
01111011 01000010 01101001
01100101 01101110 01110110
01100101 01101110 01101001
01100100 01101111 01110011
01111101

Hexadecimal

43 54 46 7b 42 69 65 6e
76 65 6e 69 64 6f 73 7d e2
80 8b

CyberChef: https://gchq.github.io/CyberChef/
Dcode.fr: https://www.dcode.fr/

Criptografía - Codificaciones y cifrados

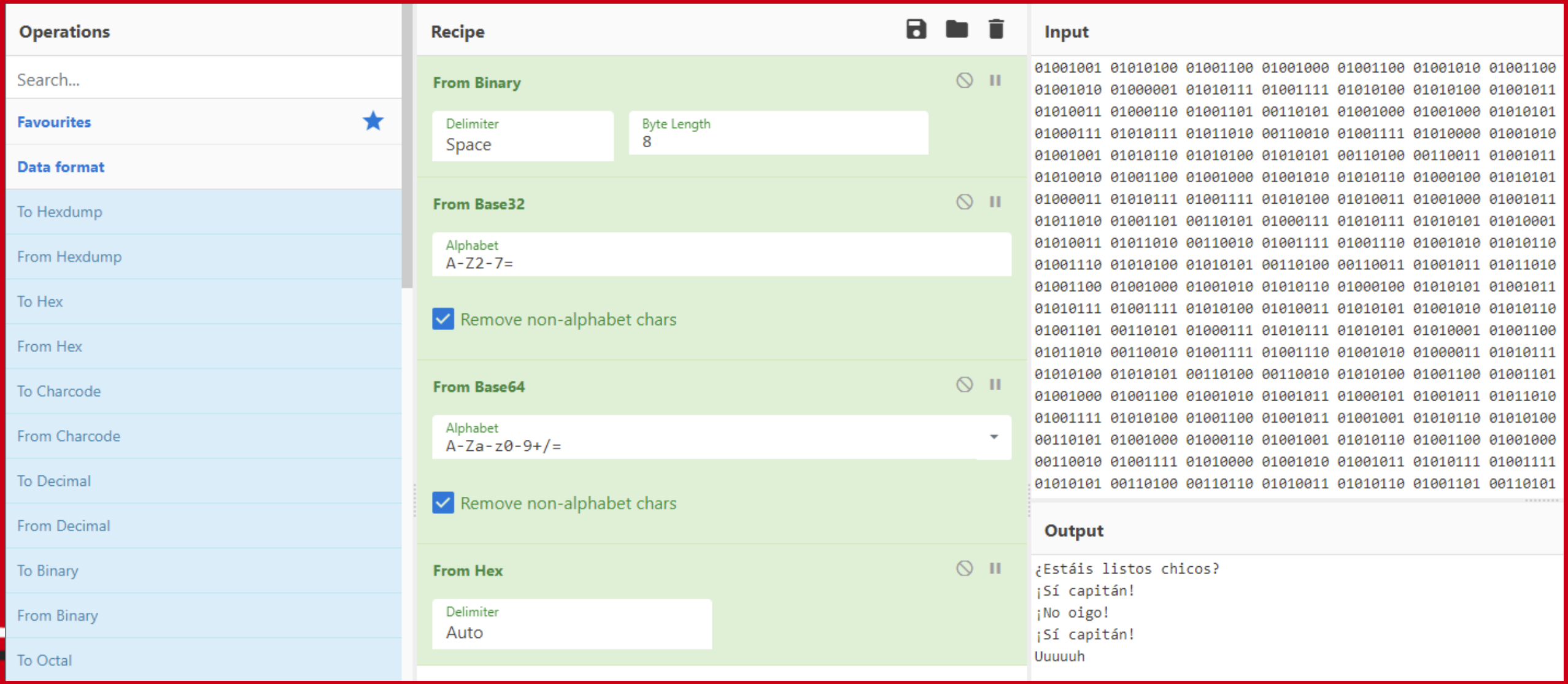
Algunas de las maneras más comunes de ocultar información son mediante **codificaciones y cifrados**. Esto consiste en utilizar una única clave para cifrar y descifrar la información. Por lo tanto, siendo el cifrado **reversible**.

Codificaciones

- Representan la misma información de diferentes maneras.
- Es reversible
- Algunos ejemplos son Base64, Base32 o ASCII

Cifrados

- Ocultan la información mediante claves, normalmente secretas, y un conjunto de operaciones.
- Es reversible
- Algunos ejemplos son ROT-N/César o Vigenère



Universidad Rey Juan Carlos

CyberChef: https://gchq.github.io/CyberChef/

Criptografía - Codificaciones y cifrados

BASE64

Es un sistema de **numeración posicional** que usa 64 caracteres como base.
Sirve para representar cualquier información en binario como texto. **Se suele identificar rápidamente** por su estructura (en general, suelen acabar en ==)

Texto original

CTF{Esto es un texto en Base64. También existen otras como Base32, Base58 o Base85, por ejemplo}

Texto en Base64

Q1RGe0VzdG8gZXMgdW4gdGV4dG8gZW4gYXNlNjQuIFRhbWJp6W4gZXhpc3RlbiBvdHJhcyBjb21vIEJhc2UzMiwgQmFzZTU4IG8gQmFzZTg1LCBwb3IgZWplbXBsb30=

Criptografía - Codificaciones y cifrados

ROT-N

Es un tipo particular de cifrado en el que los caracteres se desplazan N posiciones. Por ello, N será nuestra clave secreta que ayudará a cifrar y descifrar el texto. Además de conocer la clave, deberemos conocer el diccionario que se usa.

Texto original

CTF{El rot solo va a modificar las letras, pero no las llaves}

abcdefghijklmnopqrstuvwxyz

Texto en ROT 13

PGS{Ry ebg fbyb in n zbqvsvpne ynf yrgenf, creb ab ynf yynirf}

nopqrstuvwxyzabcdefghijklm

Cifrado de César

Vigenère

Se basa en una **tabla con dos entradas.** Una será **la clave** y la otra **el texto a cifrar**. Iremos sustituyendo en el texto carácter a carácter con ayuda de la tabla y la clave. La clave será la misma para cifrar y descifrar.

Texto original

CTF{Mi clave de cifrado es Chachipiruli}

Texto en Vigenère

EAF{Op kaimy om epfthld mj Wsieoirpzjtz}

Criptografía - Codificaciones y cifrados

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

CTF{Miclavedecifradoes Chachipiruli}
CHA{chipiruliChachipiruliChachipirl}
EAF{Opkaimyomepfthldmj Wsieoirpzjtz}

Texto original

CTF{Mi clave de cifrado es Chachipiruli}

Texto en Vigenère

EAF{Op kaimy om epfthld mj Wsieoirpzjtz}

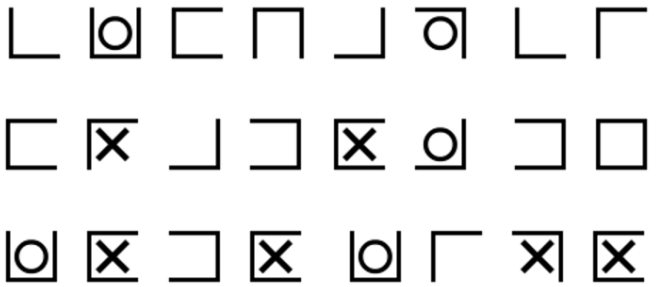
Criptografía - Otras codificaciones

Tic-Tac-Toe

Texto original

CTF{Hay cifrados de todo tipo}

Texto en Tic-Tac-Toe



DCode.fr: https://www.dcode.fr/chiffre-tic-tac-toe

EJ 1: VVJKQ3tNdXkgYmllbiwgdmVvlHF1ZSBzYWJlcyBpZGVudGlmaWNhciB1biBiYXNlNjR9

EJ 2: GXJ{Rs xshsw psw VSX wsr 13}

EJ 3: D1ETr0RtqzIwMKZtoT9mVUEyrUEipIOyp3EuovOwnJMlLJEipIOgLKZtMTHtqJ5uVUMyra0=

EJ 4: -.-. - ..-. / . ... - --- / . .../ -.-. --- -.. .. --. --- / -- --- .-. ... . /

Ejercicios propuestos

Criptografía - Otros cifrados

XOR

Consiste en cifrar siguiendo unas **reglas matemáticas** y una **clave secreta**. Como la **longitud** de la **clave** suele ser **menor al texto**, se repetirá **cíclicamente**. Todos los caracteres se pasarán a binario y se operará con ellos. Reglas:

1. **Conmutativa:** `A xor B = B xor A`
2. **Asociativa:** `(A xor B) xor C = A xor (B xor C)`
3. **Autoinversa:** `(A xor B) xor B = A`

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

XOR

EJEMPLO I

Esta vez nos dan directamente la flag, pero parece que está cifrada:

Flag: 13 3e 2b 24 3d 3d 14 02 66 1f 04 47 34 09 11 0e 32 0d 41 0b 27 4c 02 0b 27 1a 04 47 28 03 41 02 35 4c 0c 12 3f 4c 12 02 21 19 13 08 3b

Formato de la flag: URJC{}

XOR

EJEMPLO II

¡Ayúdanos a descifrar este texto! Conocemos la correspondencia de algunas cadenas:

"cifrado muy utilizado" = 0c 2d 03 3e 00 31 3d 6a 2e 36 2d 66 16 1b 04 1c 0c 0e 08 10 06

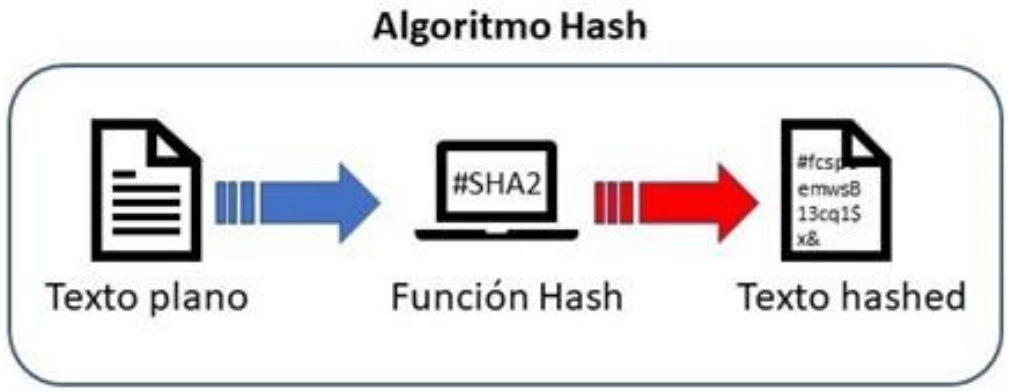
"propiedades importantes" = 3c 13 3a 22 23 26 27 35 22 06 1c 4d 19 08 04 06 06 1d 17 01 30 00 3f

06 38 66 3b 20 3f 50 00 07 49 01 07 56 0c 2d 03 3e 00 31 3d 6a 2e 36 2d 66 16 1b 04 1c 0c 0e 08 10 06 56 0a 2a 45 20 00 75 31 38 2a 33 20 29 04 1d 0c 16 0c 15 63 20 00 13 01 21 45 3c 13 3a 22 23 26 27 35 22 06 1c 4d 19 08 04 06 06 1d 17 01 30 00 3f 6b 16 27 2b 2d 27 3b 66 17 06 08 1e 00 07 49 01 07 56 0c 2d 03 3e 00 31 3d 6a 1b 0c 06 66 1a 4f 0e 1f 0b 1b 0a 11 1a 56 1f 25 17 38 04 75 36 2f 2f 63 20 23 1b 1b 02 50 00 1a 49 17 05 17 1d 2b 45 3c 0e 31 20 ab 30 63 35 36 0f 06 0e 11 17 54 05 15 1a 56 1f 36 0a 3c 08 30 36 2b 27 26 27 66 07 0a 01 50 3d 3b 3b 54 19 17 1d 25 45 3f 00 36 33 38 63 2f 35 66 00 03 0c 06 00 58 49 54 0d 13 1c 27 0c 2a 13 34 20 26 2c 63 2d 66 00 00 03 03 00 13 1c 1d 1b 56 03 25 45 2a 0d 34 35 40 16 11 1e 05 18 37 22 22 45 11 1a 54 0f 17 0c 2d 09 31

Criptografía - Hashes

¿Qué es un hash?

- Es una **función matemática o criptográfica**, resume la información
- Da como **resultado** una cadena de caracteres de longitud fija (***digest***), **independientemente** de la longitud entrada
- Es **irreversible (One Way)**. Una vez aplicada **no se puede obtener el valor inicial.**



Criptografía - Otras codificaciones

¿Qué es un hash?

Your String	Hola Mundo
MD5 Hash	d501194c987486789bb01b50dc1a0adb Copy
SHA1 Hash	48124d6dc3b2e693a207667c32ac672414913994 Copy

Your String	En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo...
MD5 Hash	8ad3504f03861ad37fd575ebef0ebe9f Copy
SHA1 Hash	125bf1068008747a2095af763b940d374174592d Copy

Criptografía - Hashes

Propiedades

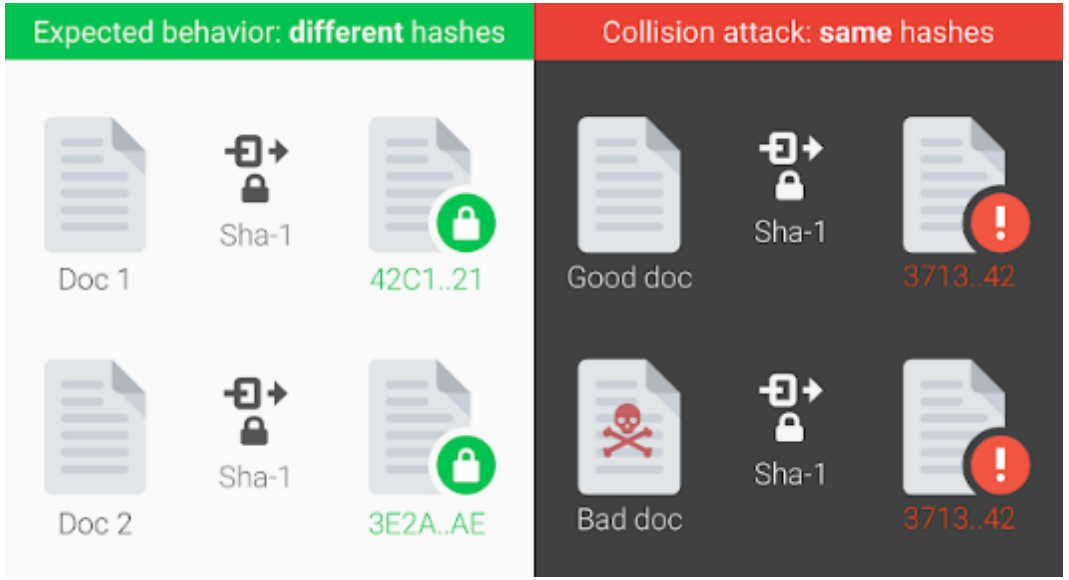
- **Collision Resistance (CR)**
  - Dado un hash H, encontrar dos mensajes m y m' tal que H(m) = H (m')
- **Target-collision resistance (TCR)**
  - Dado un hash H y un mensaje m, encontrar m' tal que H(m) = H (m')
- **Preimage resiatance(PR)**
  - Dado un hash H y una imagen I, encontrar m tal que H(m) = I

Criptografía - Hashes

- Lo que sí puede hacerse es **pre-computar** cadenas típicas, dado que una función hash devolverá el mismo resultado para la misma cadena (es determinista)
- **Conociendo la función utilizada** podemos realizar ataques de **fuerza bruta** sobre los hashes, de forma que, si en nuestro **diccionario** se encuentra la palabra *hasheada*, **sabremos qué esconde el hash**
- Es importante destacar que esto **NO ES LO MISMO QUE REVERTIR EL CÁLCULO**
- **Intentar adivinar un hash** de una palabra de longitud mayor que 8 es **computacionalmente muy costoso**

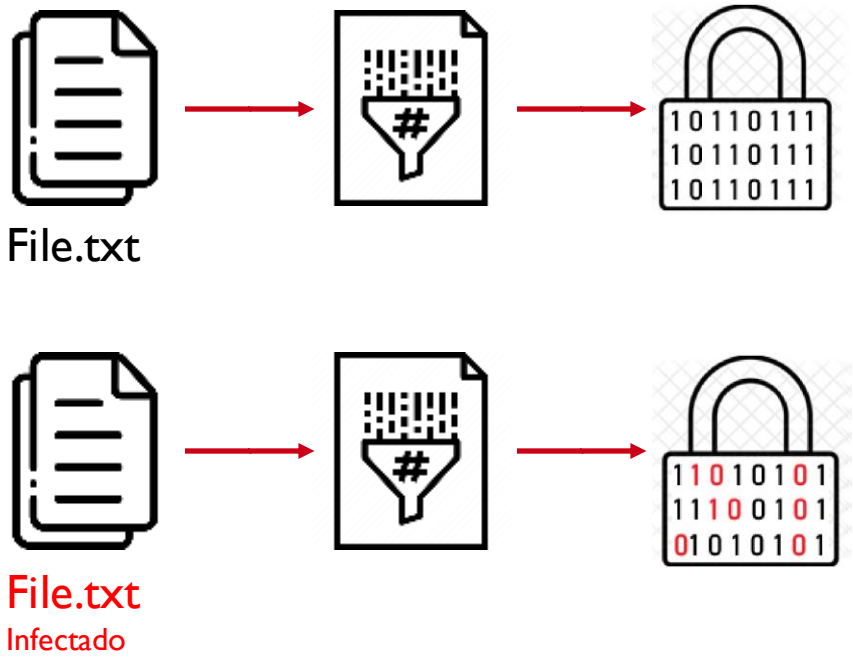
Criptografía - Hashes

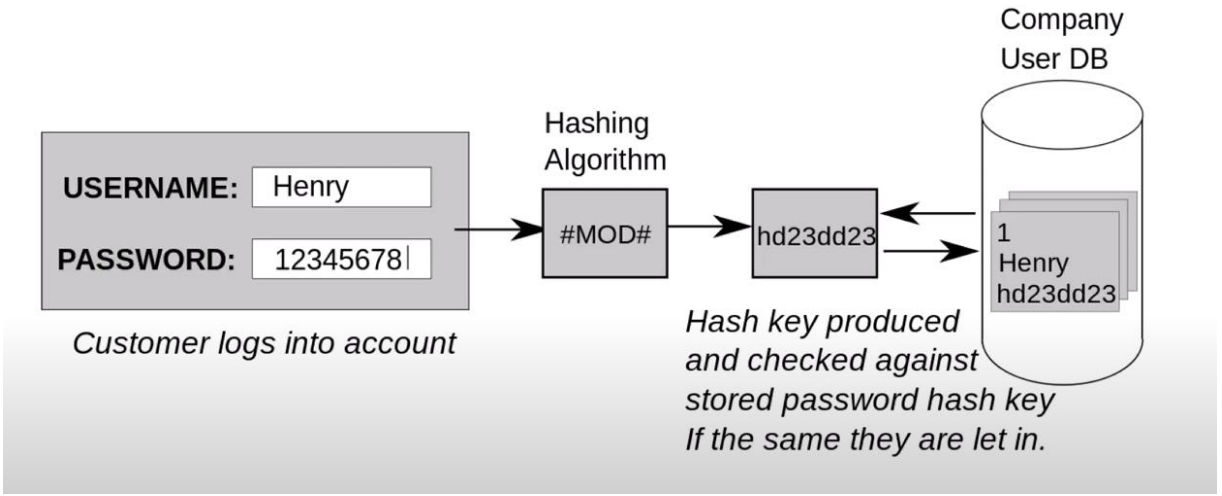
- Existen determinadas funciones hash cuyo uso **no se recomienda**
  - **MD5**
  - **SHA1**
- Aunque la probabilidad es muy baja, podrían existir **colisiones**



¿Para qué sirven los hashes?

Criptografía - Hashes





https://bazaar.abuse.ch/browse/

Criptografía - Hashes

- Cada **fichero** se puede resumir con un **valor hash**
- Existen herramientas que, dada una lista de **hashes,** nos automatizan el proceso de obtener un valor que genere dicho hash.
- **Esto permite obtener la contraseña de ficheros cifrados**

Criptografía - Hashes

Ataque por fuerza bruta

```
┌──(kali㉿kali)-[~/Documents]
└─$ hashcat -a 3 -m 0 hash.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 5.0+debian  Linux, None+Asserts, RELOC, SPIR, LLVM 16.0.6, SLEEF

* Device #1: cpu-sandybridge-11th Gen Intel(R) Core(TM) i7-11700K @ 3.60GHz, 2919/5902 MB (1

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
```

```
Hardware.Mon.#1..: Util: 66%

4d186321c1a7f0f354b297e8914ab240:hola

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 0 (MD5)
```

Ataque por diccionario/wordlist

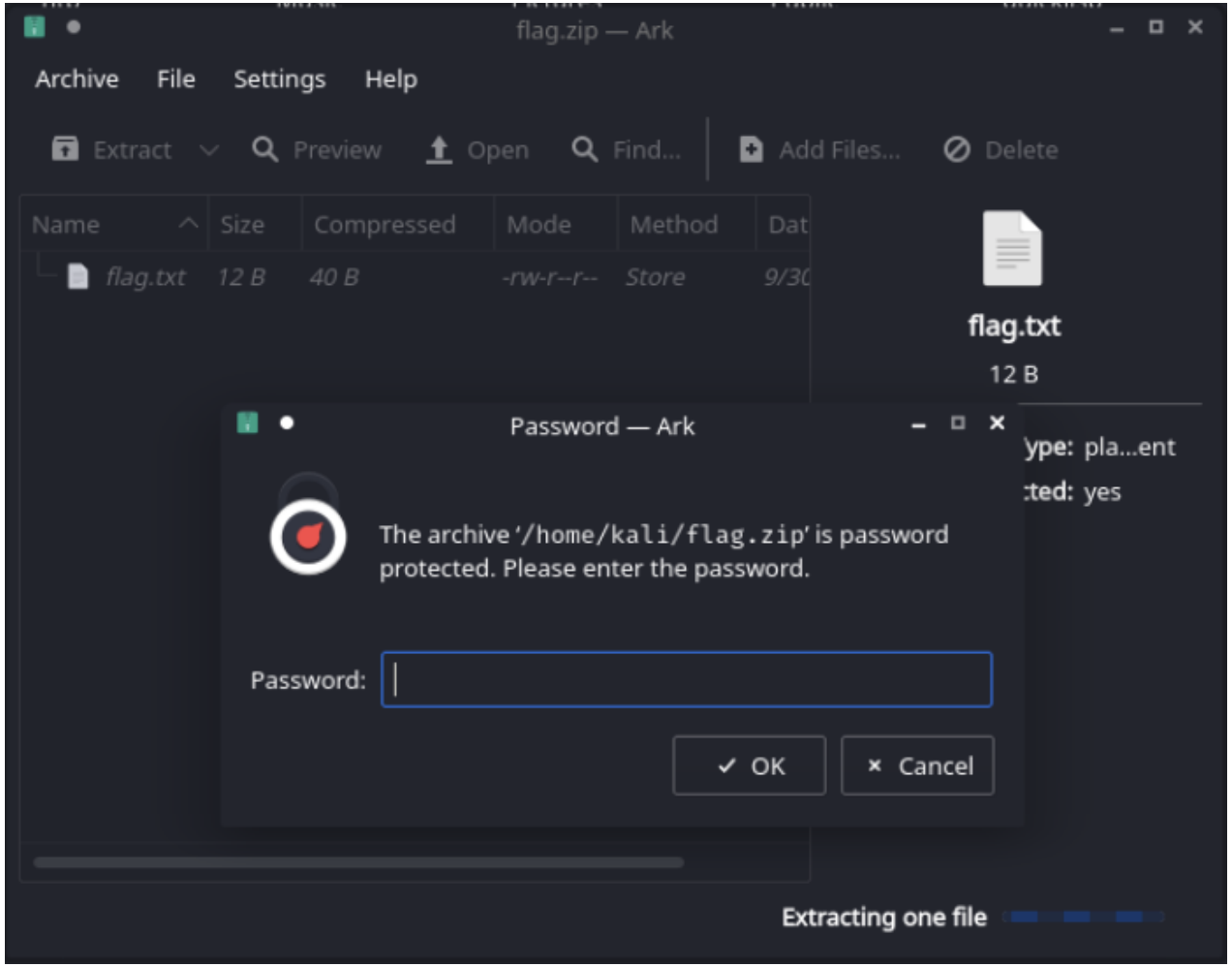
```
┌──(kali㉿kali)-[~/Documents]
└─$ hashcat -a 0 -m 0 hash.txt rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 5.0+debian  Linux, None+Assert

* Device #1: cpu-sandybridge-11th Gen Intel(R) Core(TM) i7
```

```
4d186321c1a7f0f354b297e8914ab240:hola

┌──(kali㉿kali)-[~/Documents]
```

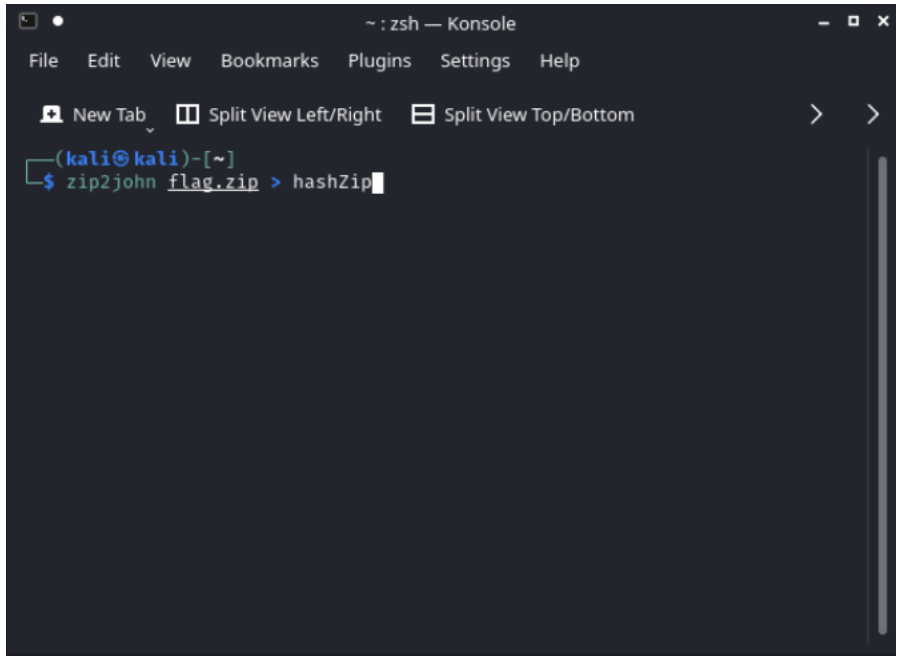
Ataque por fuerza bruta con mascara

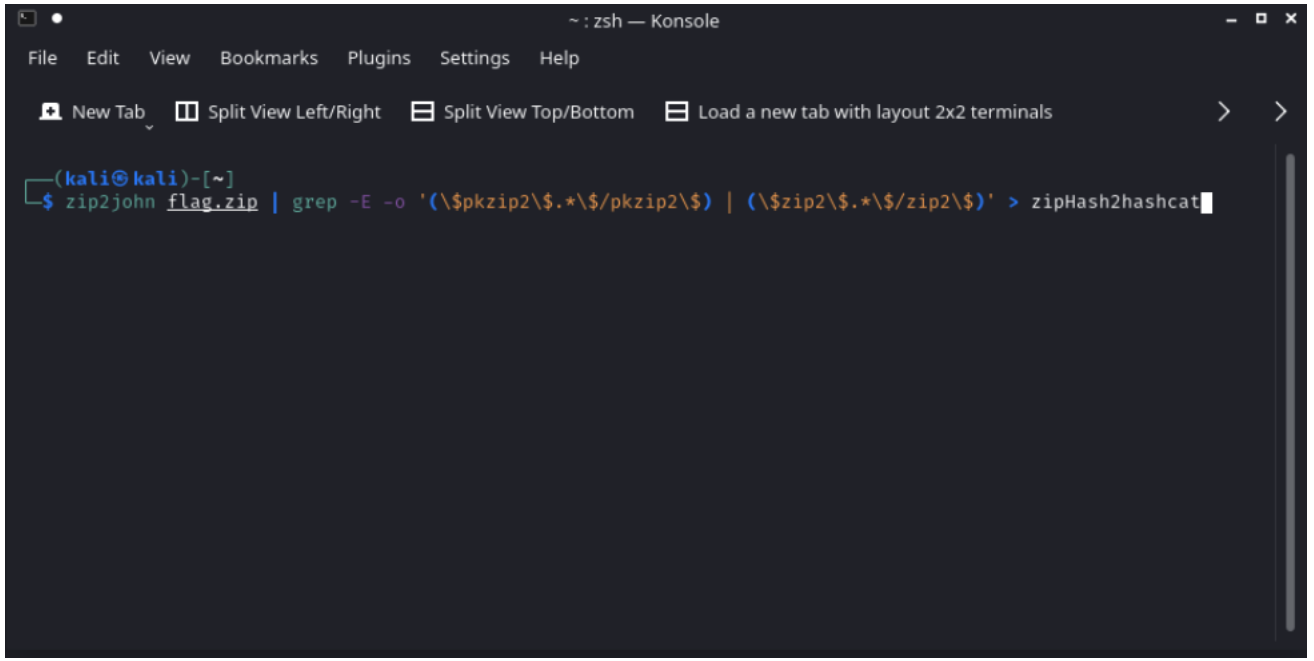
```
┌──(kali㉿kali)-[~/Documents]
└─$ hashcat -a 3 -m 0 hash.txt ?uabc?d?d?s
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 5.0+debian  Linux, None+Asserts, RELOC, SP

* Device #1: cpu-sandybridge-11th Gen Intel(R) Core(TM) i7-11700K @ 3.

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
```

Criptografía – Hashes (Ejemplo)

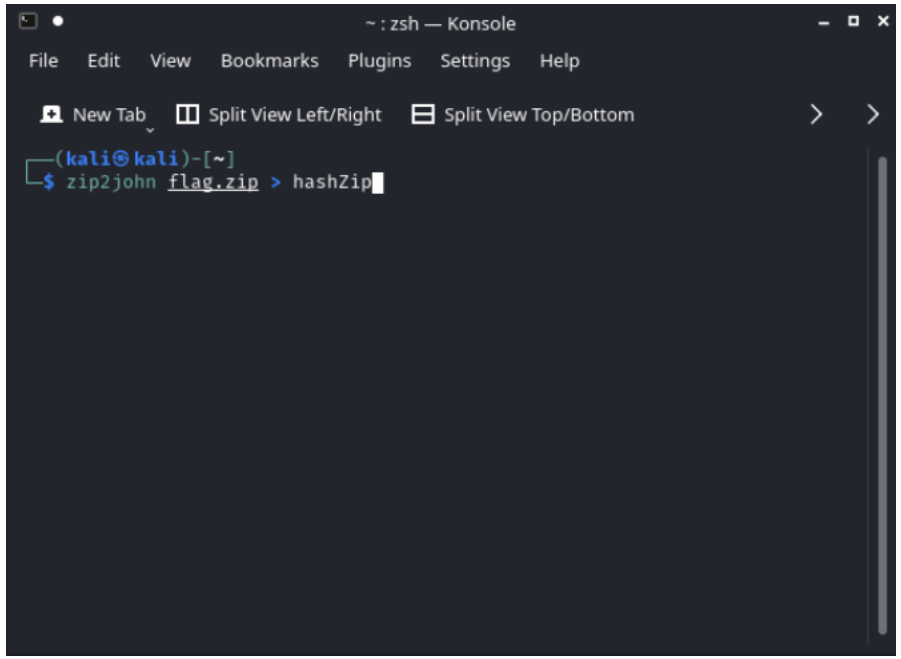


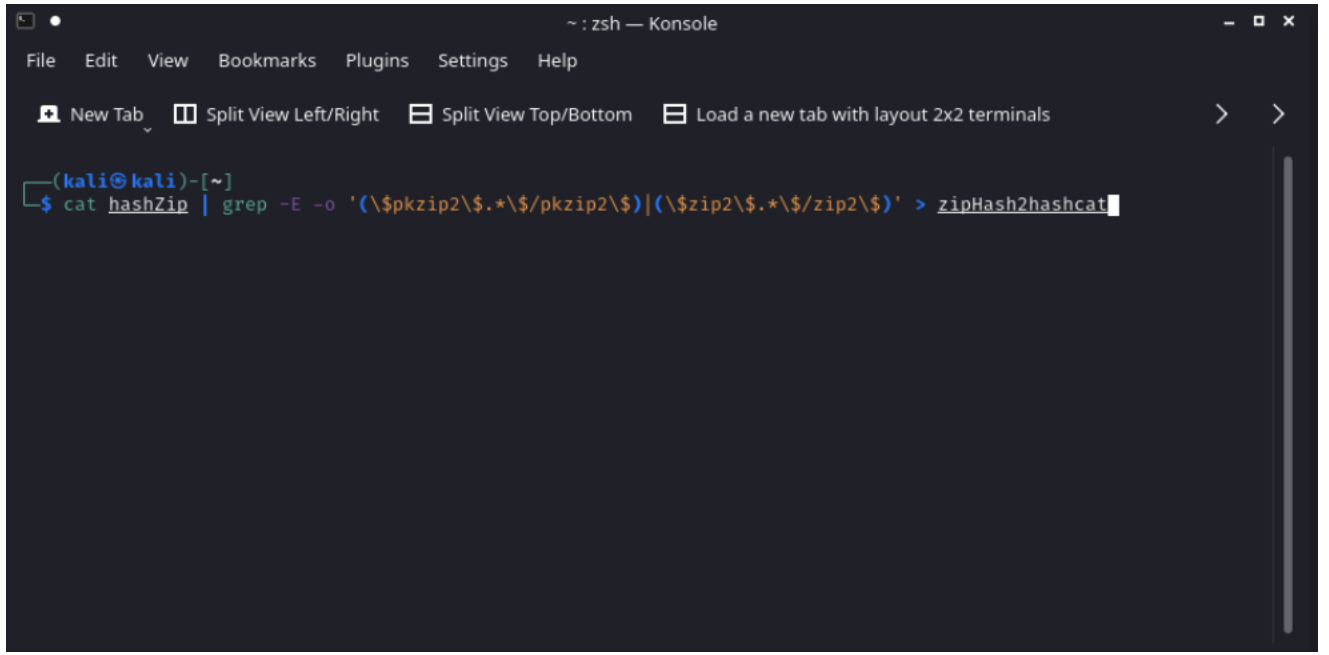
Criptografía – Hashes (Ejemplo)





Criptografía – Hashes (Ejemplo)

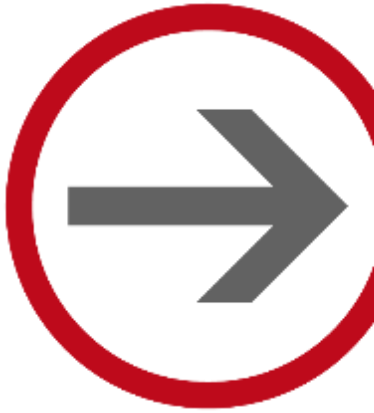




Criptografía – Hashes (Ejemplo)


```
~ : zsh — Konsole
File  Edit  View  Bookmarks  Plugins  Settings  Help
New Tab  Split View Left/Right  Split View Top/Bottom

┌──(kali㉿kali)-[~]
└─$ john hashZip --wordlist=wordlists.txt
Using default input encoding: UTF-8
Loaded 1 password hash (ZIP, WinZip [PBKDF2-SHA1 128/128 SSE2 4x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 1 candidate left, minimum 16 needed for performance.
hola1234         (flag.zip/flag.txt)
1g 0:00:00:00 DONE (2021-09-30 16:55) 100.0g/s 100.0p/s 100.0c/s 100.0C/s hola1234
Use the "--show" option to display all of the cracked passwords reliably
Session completed

┌──(kali㉿kali)-[~]
└─$ john hashZip --show
flag.zip/flag.txt:hola1234:flag.txt:flag.zip:flag.zip

1 password hash cracked, 0 left

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~]
└─$ hashcat -m 13600 zipHash2hashcat ./wordlists.txt
hashcat (v6.1.1) starting ...

Session..........: hashcat
Status...........: Cracked
Hash.Name........: WinZip
Hash.Target......: $zip2$*0*3*0*f819c01513f1f5018f4e73128d711b52*8d6c* ... /zip2$
Time.Started.....: Thu Sep 30 16:59:35 2021 (0 secs)
Time.Estimated...: Thu Sep 30 16:59:35 2021 (0 secs)
Guess.Base.......: File (./wordlists.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:        3 H/s (1.66ms) @ Accel:64 Loops:999 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests
Progress.........: 1/1 (100.00%)
Rejected.........: 0/1 (0.00%)
Restore.Point....: 0/1 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-999
Candidates.#1....: hola1234 → hola1234

Started: Thu Sep 30 16:58:55 2021
Stopped: Thu Sep 30 16:59:37 2021

┌──(kali㉿kali)-[~]
└─$ hashcat -m 13600 zipHash2hashcat --show
$zip2$*0*3*0*f819c01513f1f5018f4e73128d711b52*8d6c*c*327662bd488eec34fe3ad3fa*4b36073395bdba927dda*$/zip2$:hola1234

┌──(kali㉿kali)-[~]
└─$
```



RETOS BÁSICOS

Universidad
Rey Juan Carlos

Para practicar lo aprendido

- Para **practicar lo que hemos visto hasta ahora**, podéis realizar los **primeros 7 retos** de la categoría ***Básica*** de la plataforma **Atenea**

https://atenea.ccn-cert.cni.es/challenges

- Estos retos resumen **lo visto hasta ahora**
- La semana que viene, veremos **criptografía más avanzada**
  - **RSA, AES, etc.**