



# Curso CTF Competitivo

---

Presentación: Raúl Martín



Universidad  
Rey Juan Carlos

# Índice

1. Horario de clase
2. Material básico necesario
3. Módulos del curso
4. Plataformas que vamos a utilizar

# Horario y aula de clase



- 22 Septiembre – 15 Diciembre
- Todos los viernes de **17:00 a 19:00**
- **Presencial:** Se anunciará el aula por correo cada semana y si no la web estará actualizada.
- Información: <https://urjc-ctf.github.io/web/>
- Recomendado asistir a todas las sesiones. Solo se realizarán presencial y sin grabaciones, si te pierdes alguna tienes la grabación del año anterior.

# Material básico necesario

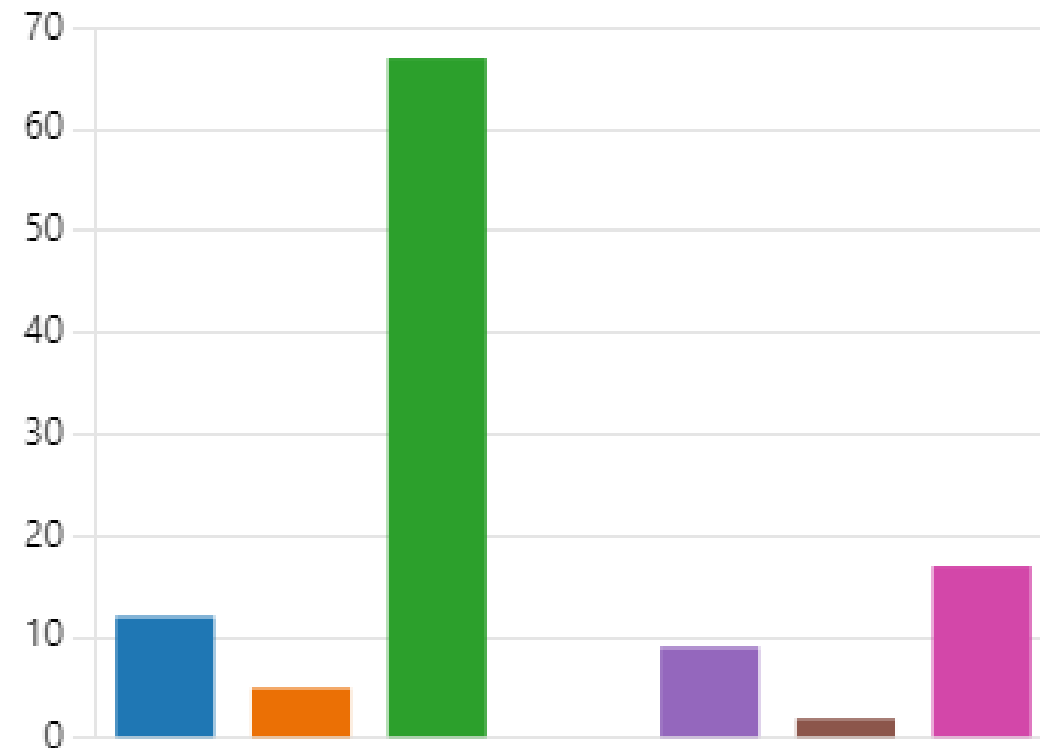


## ¿Qué es necesario para participar?

- Un ordenador con conexión a internet (obviamente)
- VirtualBox, para usar la imagen que os vamos a proporcionar
- Ganas de aprender
- Recomendado:
  - Conocimientos básicos en algún lenguaje de programación
  - Conocimientos básicos de Linux y/o terminal
  - Soltura utilizando los buscadores (Google, Bing, DuckDuckGo...)

# ¿Quiénes sois vosotros?

● Software	12
● Informática	5
● Ciberseguridad	67
● Matemáticas	0
● Computadores	9
● Videojuegos	2
● Otras	17

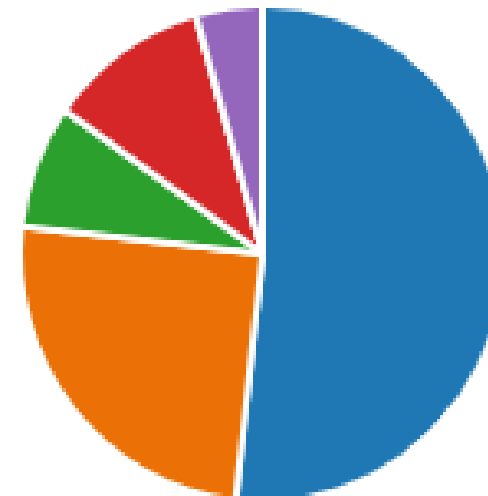


# ¿Quiénes sois vosotros?

Más detalles

💡 Información

● 1º	58
● 2º	28
● 3º	9
● 4º	12
● Otras	5

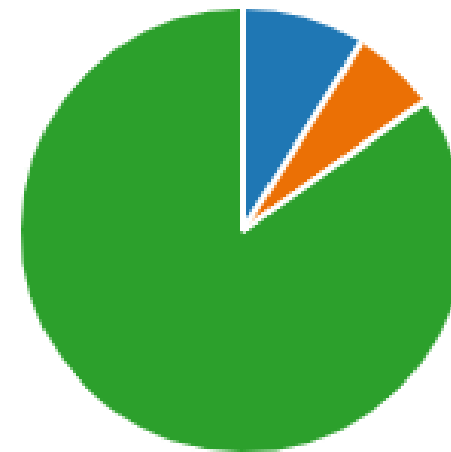


# ¿Quiénes sois vosotros?

## 5. ¿Asististe a la edición del año pasado?

Más detalles

● Sí, conseguí los créditos RAC	10
● Sí, pero no conseguí los crédito...	7
● No	95



# Módulos del curso

**Módulo I:** Introducción a retos básicos, criptografía

**Módulo II:** OSINT, Forense y Esteganografía

**Módulo III:** Ataques a servidores y explotación web

**Módulo IV:** Reversing y explotación de binarios



# Personas que han cursado el curso

**1. Nuevo contenido**

**2. Nuevos retos (facilidad en algunas asignaturas de la carrera el realizarlos)**

# <https://ctf-curso.numa.host/>

**Credenciales en correo de bienvenida**

---

# Plataformas recomendadas

[Pico CTF: https://picoctf.org/](https://picoctf.org/)

[OverTheWire: https://overthewire.org/wargames/](https://overthewire.org/wargames/)

[TryHackMe: https://tryhackme.com/](https://tryhackme.com/)

[HackTheBox: https://www.hackthebox.eu/](https://www.hackthebox.eu/)

[Atenea: https://atenea.ccn-cert.cni.es/home](https://atenea.ccn-cert.cni.es/home)

# Créditos RAC

Asistencia mínima de **10 clases y registrado** en el control de asistencia de la aplicación de la URJC.

Créditos: 1,25 ECTS

# ¿Cómo interactuar?

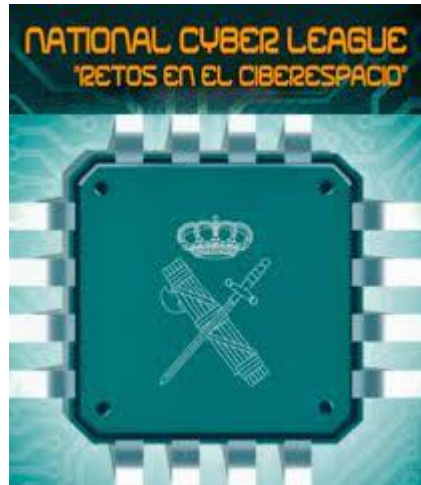
Levantamos manos.

Somos un equipo numeroso y para los retos se pueden preguntar dudas y se irá a cada ordenador a solucionarlas.

El propósito es aprender, no solo competir.

Y responder al correo en copia a todos, en otro caso os revienta

# Experiencia de los Docentes



INSTITUTO NACIONAL DE CIBERSEGURIDAD





# I. Introducción a los CTF y criptografía básica

---

Adrián Zamora y Pablo López

# Índice

1. ¿Qué es un CTF?
2. ¿Qué tipos de retos se encuentran en los CTF?
3. Conceptos básicos: encuentra la bandera
4. Criptografía y codificaciones básicas
  - Representación de los datos
  - Codificaciones y cifrados
  - Otros cifrados (XOR, Dcodefr...)
  - Hashes (MD5, SHA1, SHA256)
5. Retos básicos



# ¿Qué es un CTF?

---



# ¿Qué es un CTF?

## Capture the flag experience



## CTF = Capture The Flag (Captura la Bandera)

- Competición de **hacking**, en la que ponemos a prueba nuestras **habilidades resolviendo retos de ciberseguridad** en un **tiempo limitado**, con el objetivo de sumar puntos.
- Por **equipos o individual**



# Tipos de CTF

Existen **3 tipos principales** de competiciones CTF:

1. **Jeopardy**: retos de **distintas categorías** a resolver en un **tiempo limitado**.
2. **Ataque – Defensa**: **2 equipos, 2 redes y servicios vulnerables** en cada red. Ambos equipos deben **atacar a los servicios del contrincante** a la vez que **defienden los suyos**.
3. **Boot2root**: **máquinas creadas con fallos** de seguridad que se deben **explotar** para convertirse en **superusuario (root)**.
4. **Mezcla**

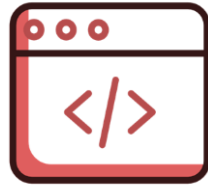


# Categorías



## Forense

Investigaciones sobre incidentes informáticos



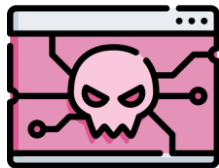
## Web

Búsqueda y explotación de vulnerabilidades en aplicaciones web



## Criptografía

Descifrado de mensajes ilegibles a simple vista



## Reversing

Análisis del código de programas y ejecutables



## OSINT

Recolección de datos a través de fuentes públicas de información



## Esteganografía

Técnica que oculta mensajes o archivos dentro de otros



# CRIPTOGRAFÍA BÁSICA

---

# 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

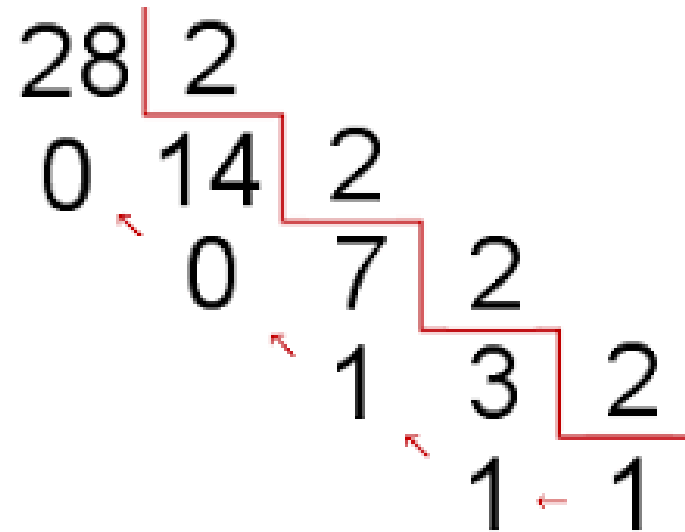
DEC	HEX	Simbolo
96	60h	`
97	61h	a
98	62h	b
99	63h	c
100	64h	d
101	65h	e
102	66h	f
103	67h	g
104	68h	h
105	69h	i
106	6Ah	j
107	6Bh	k
108	6Ch	l
109	6Dh	m
110	6Eh	n
111	6Fh	o
112	70h	p
113	71h	q
114	72h	r
115	73h	s
116	74h	t
117	75h	u
118	76h	v
119	77h	w
120	78h	x
121	79h	y
122	7Ah	z
---	---	-

# 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.



$$28 = 11100_2$$



## 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
0111101
```

## 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
  - ASCII

## Cifrados

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

## 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

```
QIRGe0VzdG8gZXMgdW4  
gdGV4dG8gZW4gYXNINj  
QulFRhbWJp6W4gZXhpc3  
RlbiBvdHJhcyBjb2IvIEJhc2U  
zMiwgQmFzZTU4IG8gQm  
FzZTgILCBwb3lgZWplbXB  
sb30=
```

## 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 yrgefn, creb ab  
ynf yynirf}

nopqrstuvwxyzabcdefghijklm

Cifrado de  
César



# Criptografía - Codificaciones y cifrados

## 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 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
ENTRADA CLAVE	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{Mi clave de cifrado es Chachipiruli}  
 CHA{chhipiruliChachipiruliChachipiruli}  
 EAF{Opkaimyom epfthldmjWsieoirpzjtz}

## Texto original

CTF{Mi clave de cifrado es  
Chachipiruli}

## Texto en Vigenère

EAF{Op kaimy om epfthld mj  
 Wsieoirpzjtz}

# 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:**  $A \text{ xor } B = B \text{ xor } A$
- 2. Asociativa:**  $(A \text{ xor } B) \text{ xor } C = A \text{ xor } (B \text{ xor } C)$
- 3. Autoinversa:**  $(A \text{ xor } B) \text{ xor } B = A$

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

## Tic-Tac-Toe

### Texto original

CTF{Hay cifrados de todo  
tipo}

### Texto en Tic-Tac-Toe

L	o	C	n	J	o	L	r
C	x	J	C	x	o	C	C
o	x	C	x	o	r	x	x



[DCode.fr: https://www.dcode.fr/chiffre-tic-tac-toe](https://www.dcode.fr/chiffre-tic-tac-toe)





### Operations

Search...

**Favourites** ★

**Data format**

- To Hexdump
- From Hexdump
- To Hex
- From Hex
- To Charcode
- From Charcode
- To Decimal
- From Decimal
- To Binary
- From Binary
- To Octal

### Recipe

**From Binary** ⊘ ||

Delimiter:  Byte Length:

**From Base32** ⊘ ||

Alphabet:   
 Remove non-alphabet chars

**From Base64** ⊘ ||

Alphabet:   
 Remove non-alphabet chars

**From Hex** ⊘ ||

Delimiter:

### Input

```
01001001 01010100 01001100 01001000 01001100 01001010 01001100
01001010 01000001 01010111 01001111 01010100 01010100 01001011
01010011 01000110 01001101 00110101 01001000 01001000 01010101
01000111 01010111 01011010 00110010 01001111 01010000 01001010
01001001 01010110 01010100 01010101 00110100 00110011 01001011
01010010 01001100 01001000 01001010 01010110 01000100 01010101
01000011 01010111 01001111 01010100 01010011 01001000 01001011
01011010 01001101 00110101 01000111 01010111 01010101 01010001
01010011 01011010 00110010 01001111 01001110 01001010 01010110
01001110 01010100 01010101 00110100 00110011 01001011 01011010
01001100 01001000 01001010 01010110 01000100 01010101 01001011
01010111 01001111 01010100 01010011 01010101 01001010 01010110
01001101 00110101 01000111 01010111 01010101 01010001 01001100
01011010 00110010 01001111 01001110 01001010 01000011 01010111
01010100 01010101 00110100 00110010 01010100 01001100 01001101
01001000 01001100 01001010 01001011 01000101 01001011 01011010
01001111 01010100 01001100 01001011 01001001 01010110 01010100
00110101 01001000 01000110 01001001 01010110 01001100 01001000
00110010 01001111 01010000 01001010 01001011 01010111 01001111
01010101 00110100 00110110 01010011 01010110 01001101 00110101
```

### Output

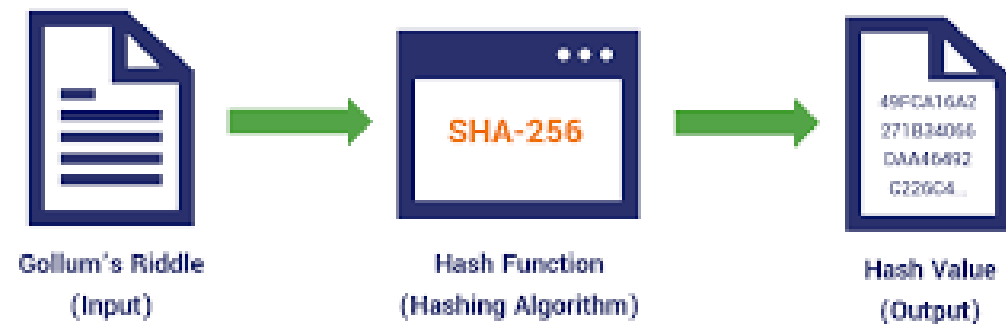
```
¿Estáis listos chicos?
¡Sí capitán!
¡No oigo!
¡Sí capitán!
Uuuuuh
```



## ¿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**. Una vez aplicada **no se puede obtener el valor inicial**.

### How Hashing Works

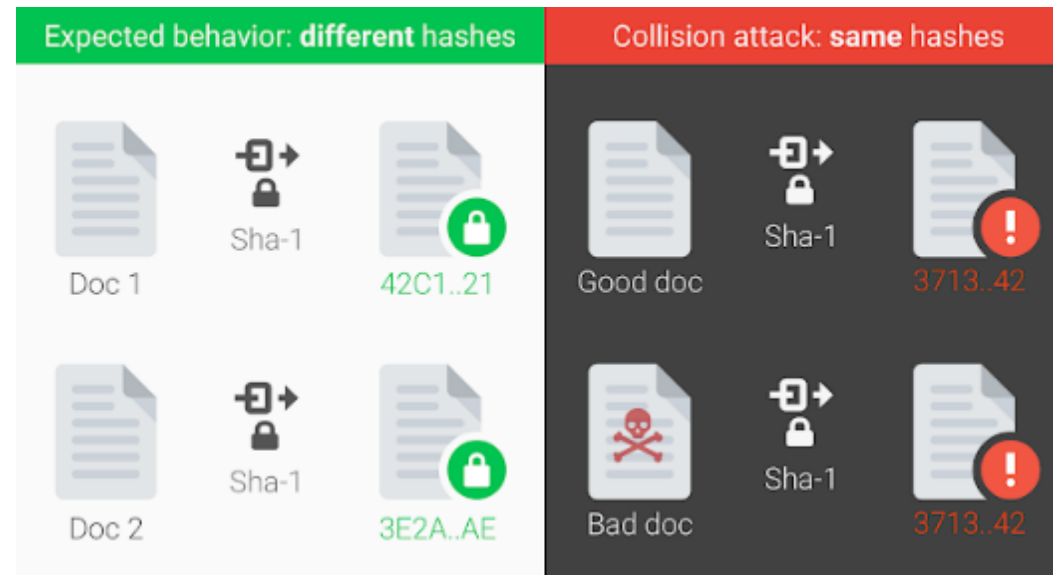


## Propiedades

- **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 resistance(PR)**
  - Dado un hash  $H$  y una imagen  $I$ , encontrar  $m$  tal que  $H(m) = I$

- 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**

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

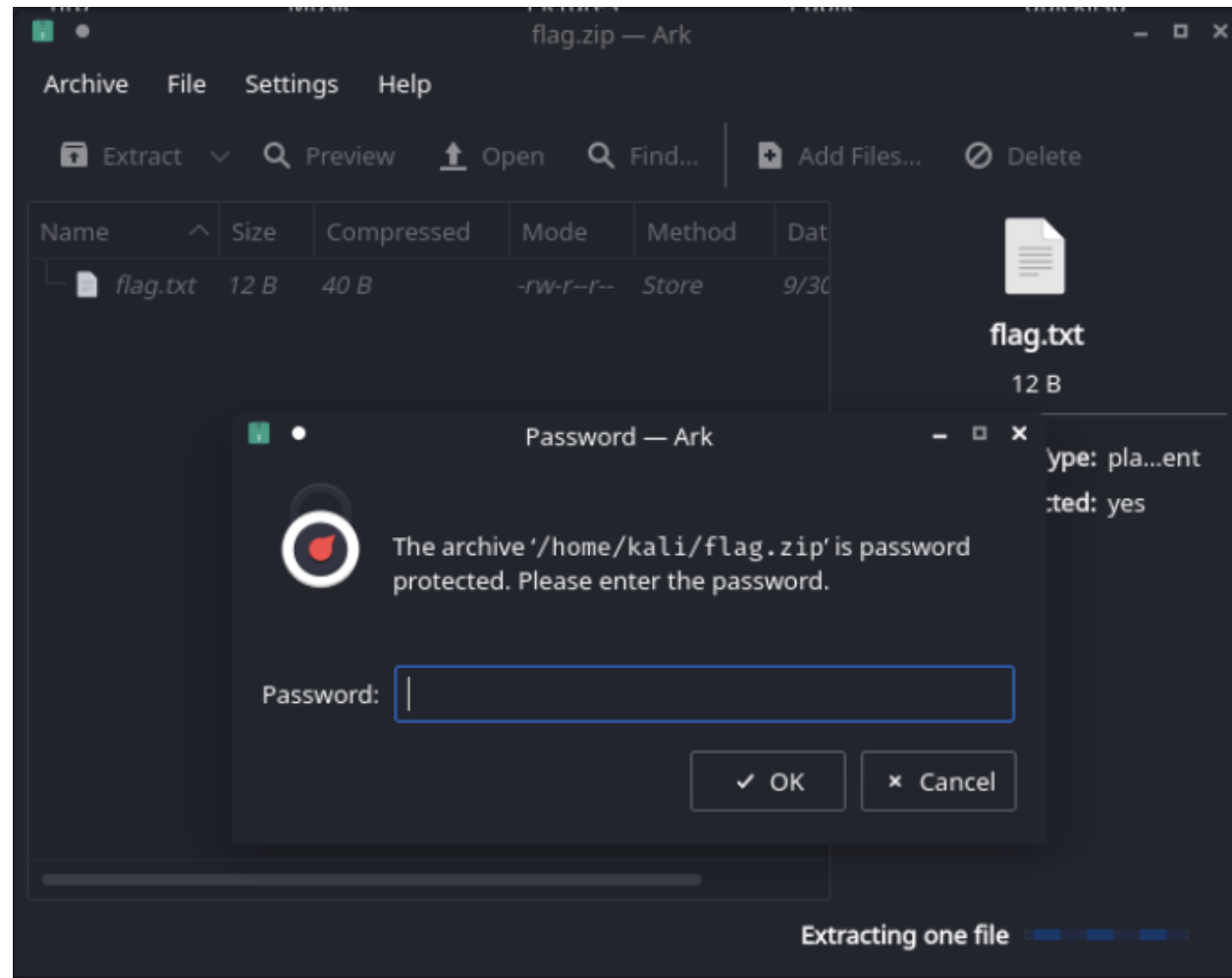


# Criptografía - Hashes

- 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 (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)-[~]
└─$ zip2john flag.zip > hashZip
```

```
~: zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals
(kali@kali)-[~]
└─$ zip2john flag.zip | grep -E -o '(\$pkzip2\$.*\$/pkzip2\$) | (\$zip2\$.*\$/zip2\$)' > zipHash2hashcat
```

# Criptografía – Hashes (Ejemplo)



```
~: zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom
(kali@kali)-[~]
└─$ zip2john flag.zip > hashZip
```

```
~: zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals
(kali@kali)-[~]
└─$ cat hashZip | grep -E -o '(\$pkzip2\$.+$/pkzip2\$)|(\$zip2\$.+$/zip2\$)' > zipHash2hashcat
```

# 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)-[~]
└─$
```

# XOR

## XOR

- Cifrado que opera XOR en binario
- Utiliza una **clave secreta**.
- Como la **longitud** de la **clave** suele ser **menor al texto**, se repetirá **cíclicamente**.

### Propiedades



**1. Conmutativa:**  $A \text{ xor } B = B \text{ xor } A$

**1. Asociativa:**  $(A \text{ xor } B) \text{ xor } C = A \text{ xor } (B \text{ xor } C)$

**1. Autoinversa:**  $(A \text{ xor } B) \text{ xor } B = A$

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

## Reto 7 Atenea

XOR (4pts)  

**Dificultad:** ★☆☆☆☆

En criptografía, el cifrado XOR es, como su nombre indica, un algoritmo de cifrado basado en el operador binario XOR:

$A \text{ xor } 0 = A$   
 $A \text{ xor } A = 0$   
 $(B \text{ xor } A) \text{ xor } A = B$

Una cadena de texto puede ser cifrada aplicando el operador de bit XOR sobre cada uno de los caracteres utilizando una clave. Para descifrar la salida, sólo hay que volver a aplicar el operador XOR con la misma clave.

Usa la clave **encryptXOR** para descifrar el siguiente mensaje:

**UGFzc3dvcnQ6IHhvFzYMAcEfiBiAglA==**

Recuerda que la respuesta hay que ponerla en el formato correcto: flag{md5}

Referencias:  
[https://es.wikipedia.org/wiki/Cifrado\\_XOR](https://es.wikipedia.org/wiki/Cifrado_XOR)  
<http://xor.pw>  
<https://conv.darkbyte.ru>



<http://xor.pw/>  
<https://gchq.github.io/CyberChef/>



# XOR

## DADO UN PAR TEXTO EN CLARO/TEXTO CIFRADO

Aplicando la propiedad autoinversa del XOR podremos descifrar la clave

$$(A \text{ xor } B) \text{ xor } B = A$$

**Texto en claro:** A

**Texto cifrado:** (A xor K)



**Clave:** Texto xor Cifrado

**Clave:** A xor (A xor K) = K

# 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



# XOR

## ¿Cómo lo resolvemos?

Conocemos el formato de la flag...

CIFRADO	FLAG
13	U
3e	R
2b	J
24	C
3d	{
3d	?
14	?
02	?
66	?
...	...

$$(A \text{ xor } B) \text{ xor } B = A$$

**Tenemos:**

Cifrado = Texto[i] XOR Clave[i]

Flag = Texto[i]

**Entonces:**

Cifrado[i] XOR Texto[i] = Clave[i]

# XOR

## ¿Cómo lo resolvemos?

CIFRADO		FLAG		CLAVE
13		U		k[0]
3e		R		k[1]
2b		J		k[2]
24		C		k[3]
3d	XOR	{	=	k[4]
3d		?		?
14		?		?
02		?		?
66		?		?
...		...		...

**(A xor B) xor B = A**

**Tenemos:**

Cifrado = Texto[i] XOR Clave[i]

Flag = Texto[i]

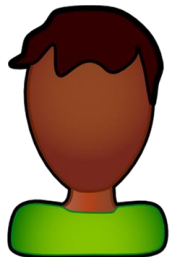
**Entonces:**

Cifrado[i] XOR Texto[i] = Clave[i]

## EJEMPLO III: Alice y Bob quieren intercambiar mensajes

**¡Te mando la contraseña secreta cifrada con mi clave!**

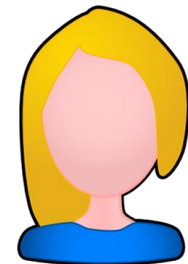
16 3e 2b 35 1e 06 11 0a 1e 0c 0e 00 43 63 08 0e 05 45 25 00 00 17 16 54 0d 50 63 0f 0d 17 13 36 45  
0b 13 06 11 41 40 36 09 41 05 00 73 06 02 1c 06 11 0d 54 3e



Bob

**Te la mando de vuelta cifrada con mi clave**

5a 5f 78 50 79 73 7f 6e 7f 47 6b 79 0f 02 5b 6b 62 30 4b 64 61 5c 73 2d 41 31 30 6a 6a 62 7d 52 24 40  
76 7f 5d 20 13 53 6e 34 6b 64 12 4d 67 65 4a 70 5e 31 59



Alice

**Genial, voy a volver a cifrar otra vez con mi clave**

19 33 19 26 1c 20 1a 0d 0d 22 1f 18 3e 41 37 0a 14 55 18 01 02 2e 16 59 20 00 73 06 0b 14 18 01 41  
23 04 1a 29 41 22 10 02 55 1d 01 41 28 04 17 2f 04 3f 00 1a



# XOR

¿Qué está pasando?



Bob



K1

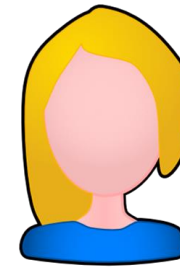
$$M1 = \text{Texto XOR } K1$$



$$M2 = M1 \text{ XOR } K2$$



$$M3 = M2 \text{ XOR } K1$$



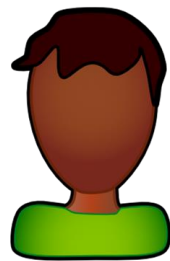
Alice



K2

# XOR

¿Qué está pasando?



Bob



K1

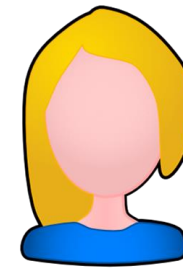
$$M1 = \text{Texto XOR } K1$$



$$M2 = (\text{Texto XOR } K1) \text{ XOR } K2$$



$$M3 = ((\text{Texto XOR } K1) \text{ XOR } K2) \text{ XOR } K1$$



Alice



K2

# XOR

¿Qué está pasando?



Bob



K1

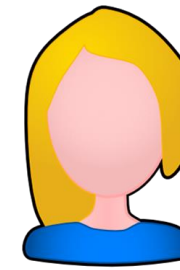
$$M1 = \text{Texto XOR } K1$$



$$M2 = (\text{Texto XOR } K1) \text{ XOR } K2$$



$$M3 = ((\text{Texto XOR } K1) \text{ XOR } K2) \text{ XOR } K1$$



Alice



K2

# XOR

## ¿Cómo descifrarlo?

Si hacemos  $M2 \text{ XOR } M3$  conseguiremos  $K1$ , y con la clave ya podremos descifrar el primer mensaje ( $M1$ )



Bob

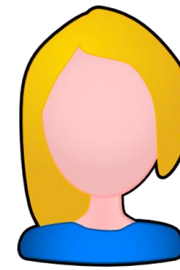


K1

$$M1 = \text{Texto XOR } K1$$

$$M2 = (\text{Texto XOR } K1) \text{ XOR } K2$$

$$M3 = \text{Texto XOR } K2$$



Alice



K2

# CRIPTOGRAFÍA SIMÉTRICA

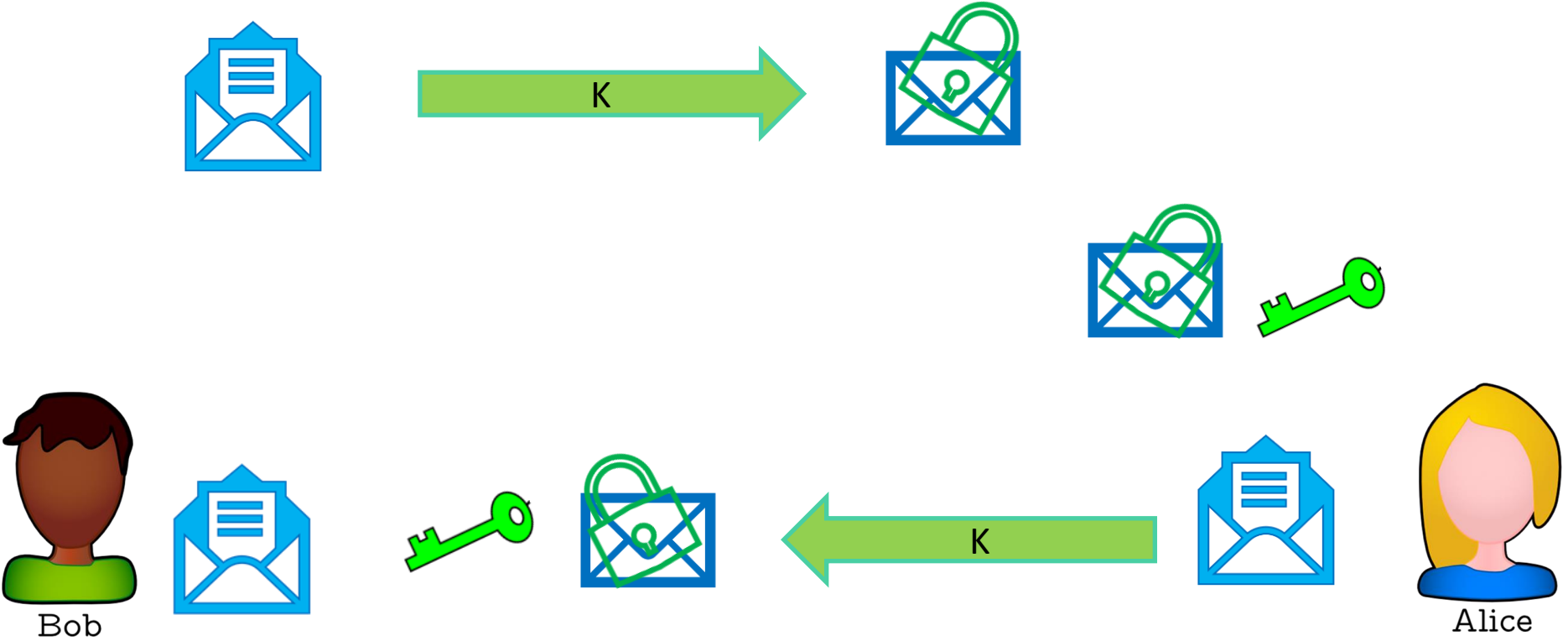
## ¿Qué es la criptografía simétrica?

Es un tipo de cifrado que utiliza la misma clave para cifrar que para descifra  
AES, RC4...





# CRIPTOGRAFÍA SIMÉTRICA



¡Bob ha podido leer el mensaje de Alice!

¡Alice ha podido leer el mensaje de Bob!

# CRIPTOGRAFÍA ASIMÉTRICA

**Bob quiere mandar un mensaje seguro a Alice pero no han acordado ninguna clave secreta previamente**

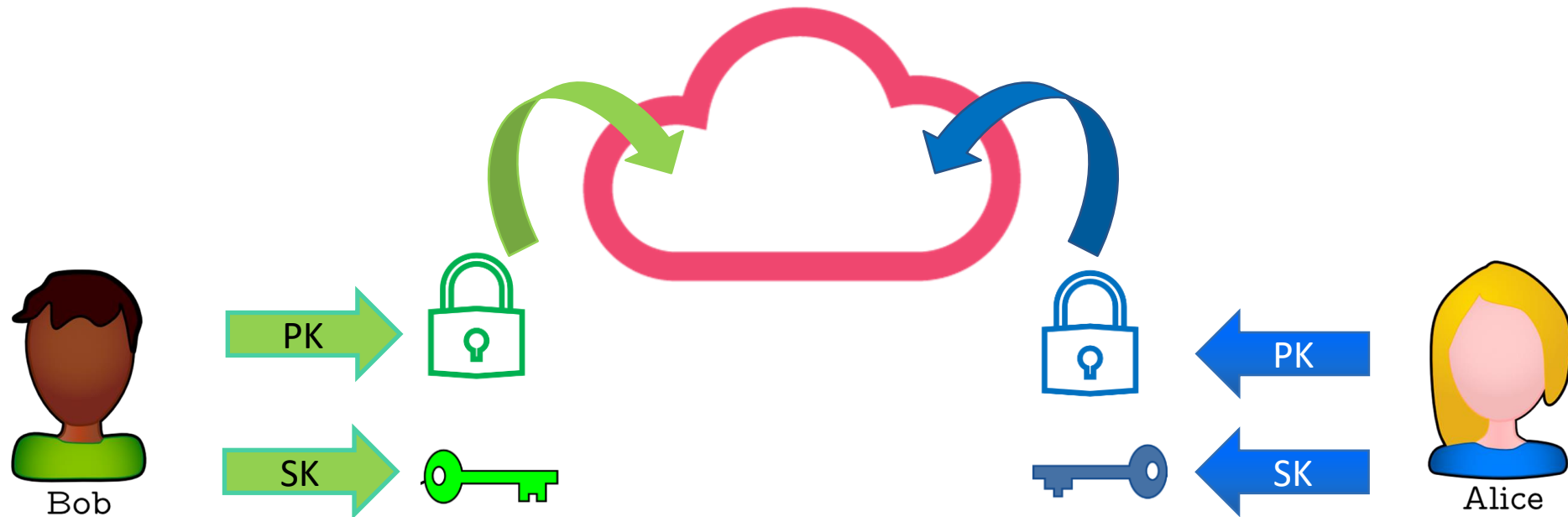
**¿Cómo lo hacen?**

# CRIPTOGRAFÍA ASIMÉTRICA

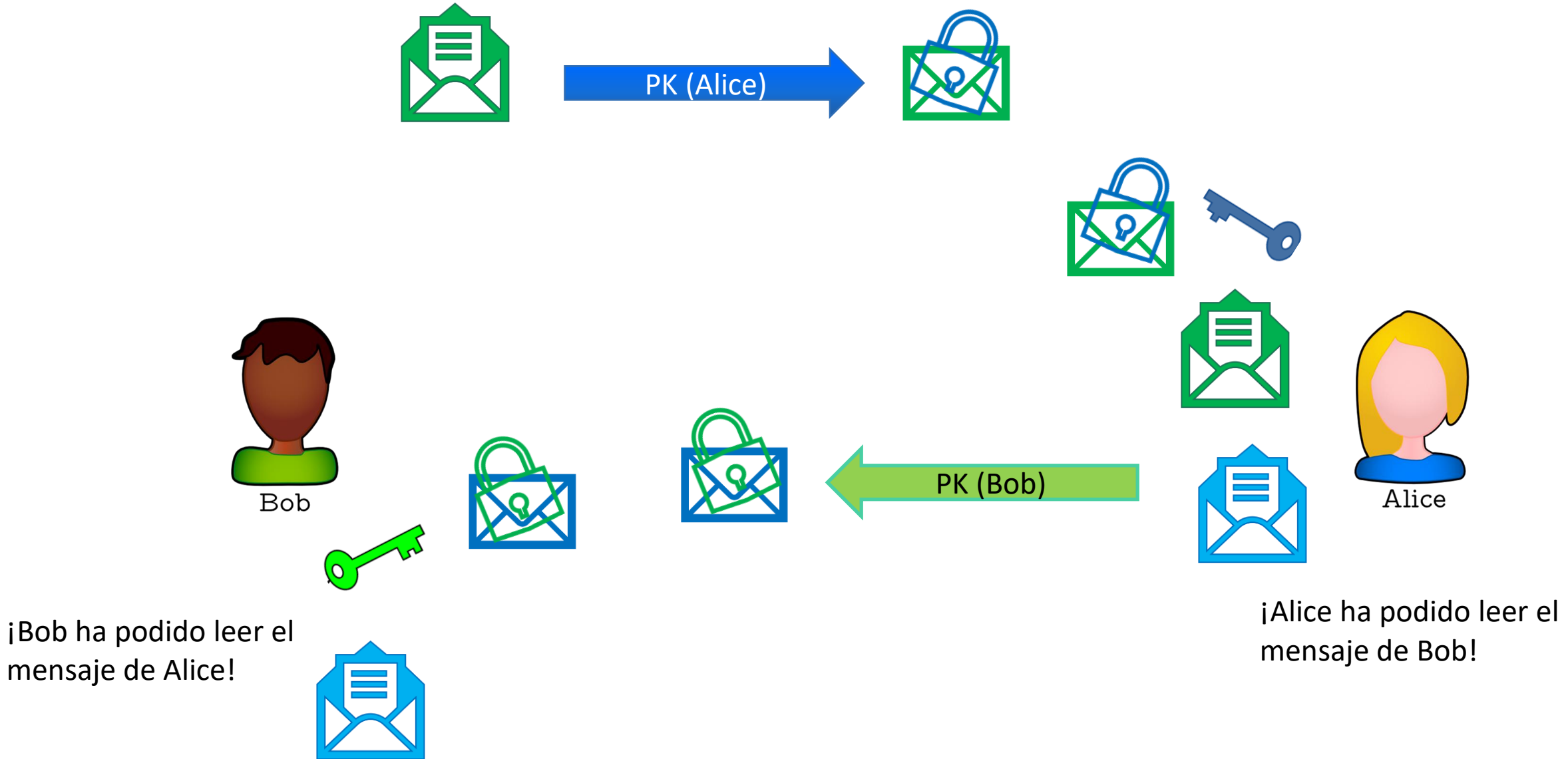
## ¿Qué es la criptografía asimétrica?

Es un tipo de cifrado que utiliza una clave pública para cifrar y otra privada para descifrar.

RSA o el Gamal



# CRIPTOGRAFÍA ASIMÉTRICA



¡Bob ha podido leer el mensaje de Alice!

¡Alice ha podido leer el mensaje de Bob!



# RETOS BÁSICOS

---

# 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.**



# I. Introducción a los CTF y retos básicos

---

Adrián Zamora y Pablo López