RETOS – Academia Hacker

# Reto 1

### Ubicación

<https://www.youtube.com/watch?v=4tcNbOtbPhM>

### Descripción

Desde la creación del instituto, algunos alumnos se han encargado de la gestión del periodido local, "El noticiero", donde se intercambian mensajes y escriben artículos didácticos. En uno de los artículos alguien anónim@ ha lanzado un reto. Asegura haber escondido un mensaje en el libro más universal de la literatura castellana. En su mensaje advierte que para empezar, nos proporciona algunas coordenadas para que las mentes mas brillantes del instituto puedan resolverlo.

### Pregunta

¿Cuál es el mensaje secreto que ha enviado el alumno anónimo y ha escondido este alumn@?

### Datos proporcionados

Libro: "El quijote de la mancha"

10:8:2

23:11:1

30:8:2

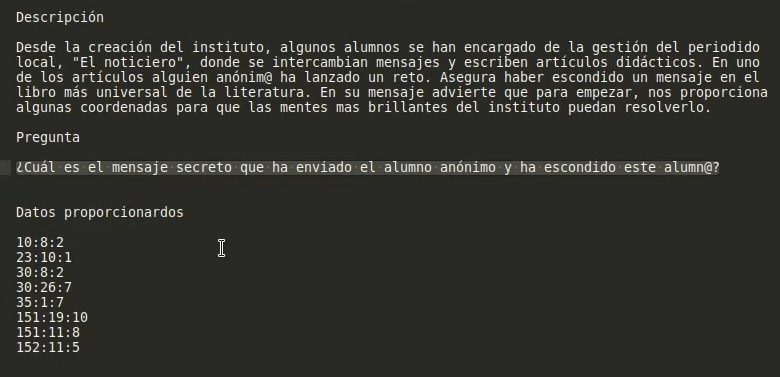
30:26:7

35:1:7

151:19:10

151:11:8

152:11:5



## Solución

Nos posicionamos en: **página:línea:posición**

flag{querer\_saber\_noticia\_sobre\_conocer\_misterio\_quien\_hizo}

# RETO 2

### Ubicación

<https://www.youtube.com/watch?v=r6Ym82jR3UQ>

### Descripción:

Después de muchos años recopilando documentación, fotos y artículos, se ha decidido realizar una limpieza y llevar la documentación antigua a una nueva sala que le ha cedido el instituto.

En esta sala se pueden archivar de una forma sencilla por fecha todos los documentos del periodico.

Durante este proceso moviendo documentos, una hoja extraña se cae entre los documentos que lleváis.

Contiene un texto ilegible junto a la referencia del emperador Julio César. ¿Podrás sacar en claro que quería decir la nota?

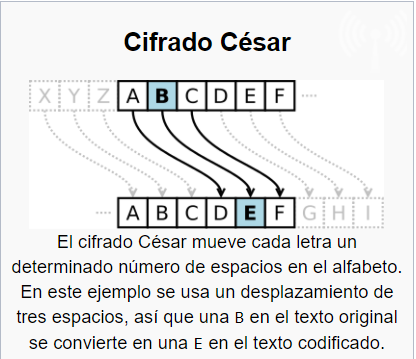
### Datos proporcionados:

*Yn fvthvragr vasbeznpvba rf pbasvqrapvny. Gr nlhqnen n cebfrthve ra yn vairfgvtnpvba. Ab gr pbasvrf, ab gbqnf ynf vasbeznpvbarf qr ynf dhr qvfcbarzbf fba gna snpvyrf pbzb rfgn ebgnpvba qr pnenpgrerf. Sveznqb: ha nzvtb.*

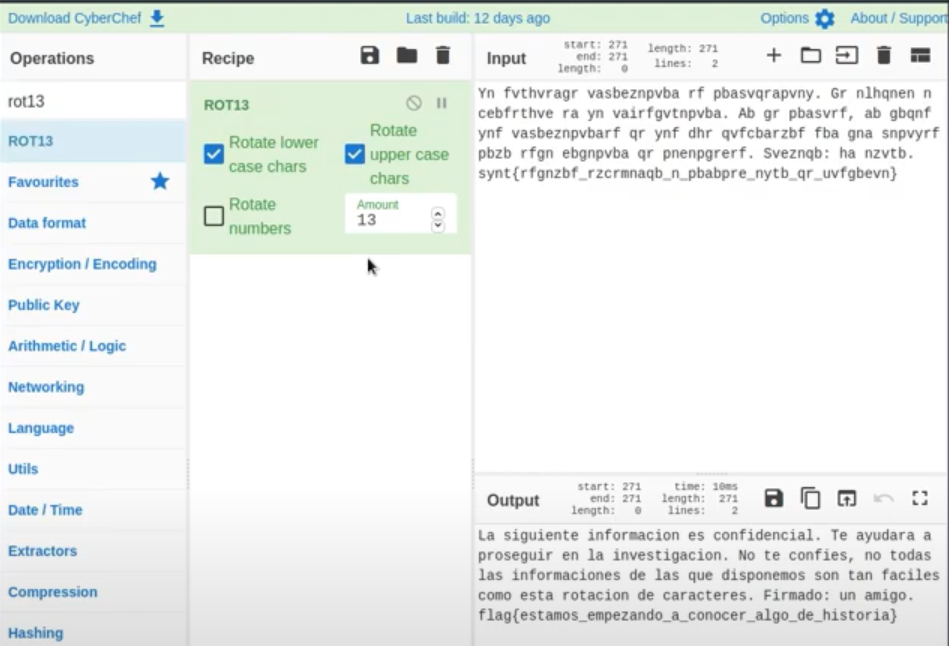
*synt{rfgnzbf\_rzcrmnaqb\_n\_pbabpre\_nytb\_qr\_uvfgbevn}*

## SOLUCIÓN

Este texto es un texto de cifrado por sustitución de la familia de los cifrados César.



Utilizamos la web <https://gchq.github.io/CyberChef/>



La siguiente informacion es confidencial. Te ayudara a proseguir en la investigacion. No te confies, no todas las informaciones de las que disponemos son tan faciles como esta rotacion de caracteres. Firmado: un amigo

flag{estamos\_empezando\_a\_conocer\_algo\_de\_historia}

# Reto 3

### Ubicación

<https://www.youtube.com/watch?v=pdt9zsTsPcM&feature=youtu.be>

### Descripción

Al darle la vuelta a la hoja, veis que tambien tiene otro texto escrito en esa parte, aunque tampoco lo entendeis. Se adjunta también un fichero con extensión pyc. ¿Nos servirá de ayuda?

### Pregunta

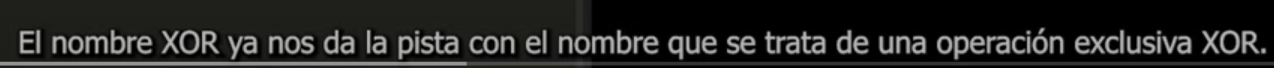
¿Podrás proporionarnos la otra información que está en la otra cara de la hoja?

### Datos proporcionados (generado con xor.pyc)

Ax8VCB4HEAAGDDMEBRERPhENAh8DLQQcEQERMAgaBjERAhwEGgADHA==

## Solución

1. Desde la VM, después de instalar uncompyle6

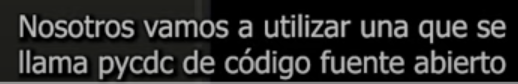






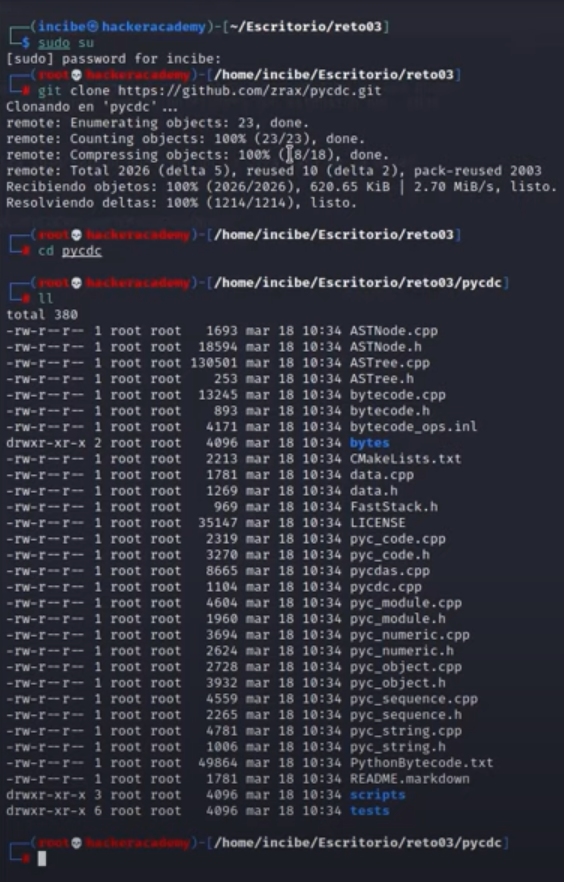
SI intentamos abrir el fichero nos dice que no porque el fichero o bien es binario o utiliza una codificación que el programa no reconoce.

Los ficheros “p” y “c” son creados por Python cuando el intérprete importa un módulo o ejecuta un fichero, es decir, parece ser que el fichero “xor.pyc” viene de ejecutar el fichero original xor.py que contiene el código fuente.

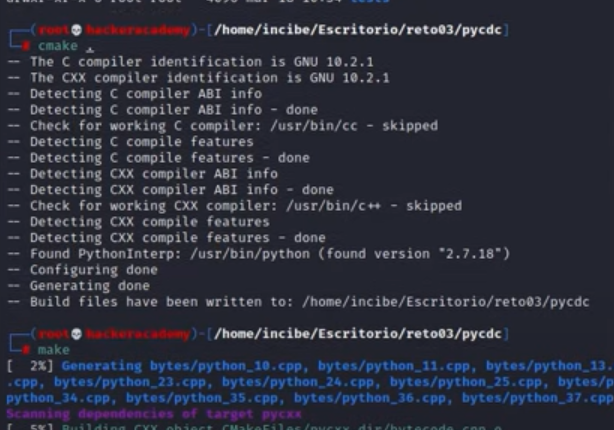
Podemos utilizar varias herramientas: uncompyle2, uncompyle6, decompile3 para las diferentes versiones de Python. Utilizaremos:

**Pycdc** de código fuente abierto

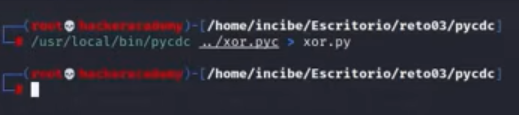
Y vamos a compilar toda la herramienta con **cmake**



Entrando en el directorio se puede ver que los ficheros de c++ están pendientes de compilación. También existe un cmakelist.txt con lo cual proceden a compilar con cmake – make – make install.



Una vez todo esté compilado hacemos make install y se puede utilizar el binario pycdc

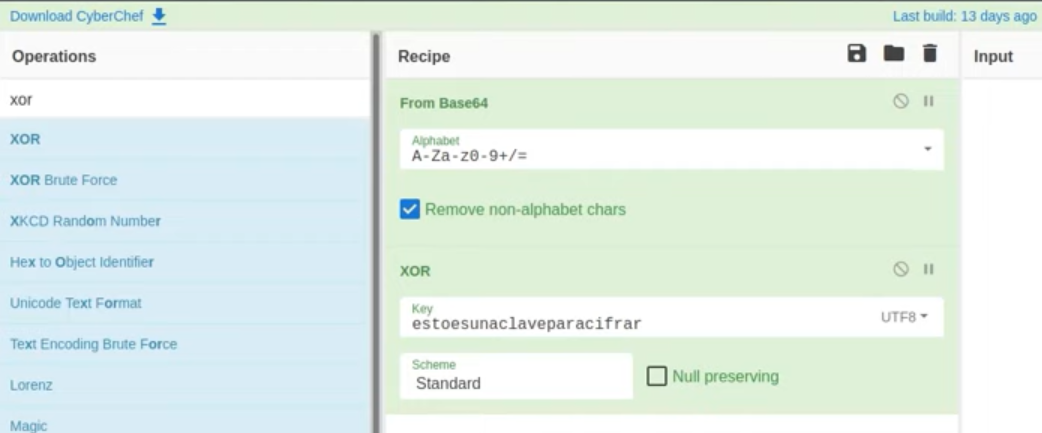




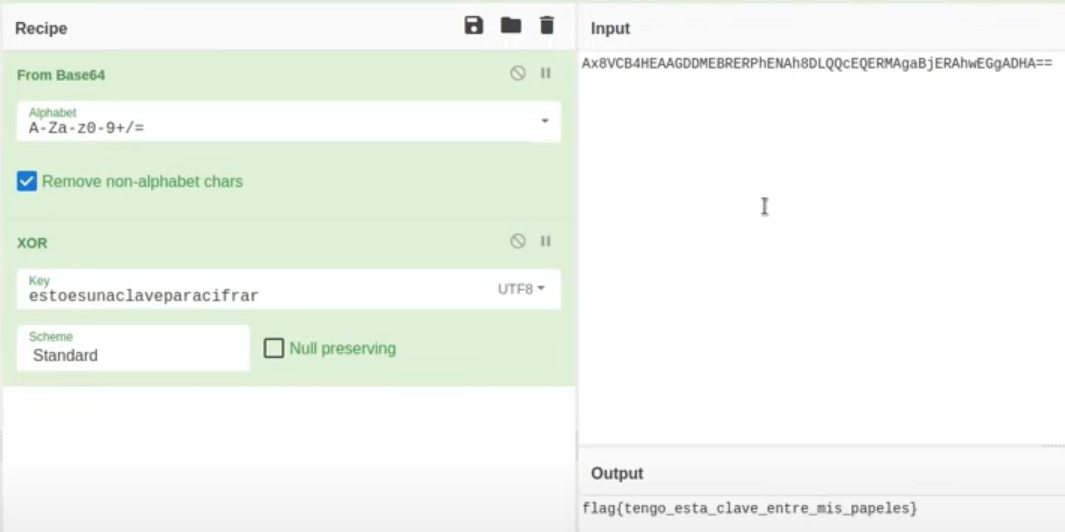


La función xor\_crypt\_string tiene 4 argumentos: data, key, encode y decode

Utilizando CyberChef y añadiendo las dos operaciones que realiza la función se rellenan los datos en la entrada para que nos dé una salida y la clave de cifrado XOR.



Y agregamos el base64 de los datos proporcionados.



Me parece que hice un proceso más simple y llegué al resultado.

## Lo que hice yo ----

Nota: Se puede utilizar lo que me pasó Gabriel: Usar pyc decompiler online <https://www.toolnb.com/tools-lang-en/pyc.html>

$ uncompyle6 xor.pyc

# uncompyle6 version 3.7.4

# Python bytecode 2.7 (62211)

# Decompiled from: Python 2.7.18 (default, Jul 14 2021, 08:11:37)

# [GCC 10.2.1 20210110]

# Embedded file name: xor.py

# Compiled at: 2021-02-01 16:35:44

import binascii, itertools, base64, sys

def xor\_crypt\_string(data, key='estoesunaclaveparacifrar', encode=False, decode=False):

from itertools import izip, cycle

import base64

if decode:

data = base64.decodestring(data)

xored = ('').join(chr(ord(x) ^ ord(y)) for x, y in izip(data, cycle(key)))

if encode:

return base64.encodestring(xored).strip()

return xored

secret\_data = sys.argv[1]

print 'Cifrado'

print xor\_crypt\_string(secret\_data, encode=True)

print 'Descifrado'

print xor\_crypt\_string(xor\_crypt\_string(secret\_data, encode=True), decode=True)

# okay decompiling xor.pyc

1. Si hago esto obtengo algo que no veo como me ayuda

$ python xor.pyc estoesunaclaveparacifrar

Cifrado

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

Descifrado

estoesunaclaveparacifrar

1. Sin embargo,
   1. utilizando la herramienta CyberChef
   2. Con la opción Encryption / Encoding
   3. XOR
   4. ponemos el dato que nos dan encriptado, esto es: Ax8VCB4HEAAGDDMEBRERPhENAh8DLQQcEQERMAgaBjERAhwEGgADHA==
   5. usamos la key que hemos visto en el código: estoesunaclaveparacifrar
   6. le decimos que utilice BASE64
   7. nos brinda como resultado: flag{tengo\_esta\_clave\_en
   8. al repetir la misma key: estoesunaclaveparacifrarestoesunaclavepa
   9. obtengo el resultado final

flag{tengo\_esta\_clave\_entre\_mis\_papeles}

# Reto 4

### Ubicación

<https://www.youtube.com/watch?v=rB816oeXXLI>

### Descripción

Con la información que habéis obtenido, intentáis utilizar las credenciales para poder acceder al ordenador del periodico y al entrar os descargáis un archivo.

El fichero original ha cifrado alguna información secreta que debes obtener para que la investigación no se atasque. Esto es lo que sabemos que ha devuelto la salida del fichero que has obtenido al entrar en la web del periódico.

### Datos proporcionados:

Hemos recibido este mensaje junto a unas instrucciones para su manejo. Parece un completo enigma.

59556843656d517a545764684d3035355a4735425a317074526e526a626d646e57544a61633249794e476461534842785a55644e5a316b7a566a42615630316e596c684f4d5751795a32646156315a6f5a4668765a325674536e4e684d6e4e6e576d3543626d4e595157645a57454a74576a4a6e5a32517962486c684d30316e596c6857613246755a32645a5748426f597a4e725a316c59516d356957456c6e59556477626c6c745557646c57484278595663775a324e496144466b6257396e5a56646f64574a745a326469563342785930646a5a32466e5054303d

Modelo: M3

Reflector: ????

ROT1.: 1

ROT2.: 2

ROT3.: 3

POS1.: A

POS2.: B

POS3.: C

ANILLO1: A

ANILLO2: B

ANILLO3: C

PLUGBOARD: bq cr di ej kw mt os px uz gh

BQ CR DI EJ KW MT OS PX UZ GH

## Lo que hice: SOLUCION

1) De Hex a ASCII

2) Base64 decode

3) Base64 decode

4) Enigma decode con los datos indicados

DESCIFRADO:

PONGA ATENC IONSI FRACA SAMOS ENEST AMISI ONNOS VEREM OSABO CADOS APERD ERLOT ODOAN OTEES TABAN DERAE NIGMA MTRES UKWBV I

Obtener solo la parte final, y pasar a minusculas.

pongaatencionsifracasamosenestamisionnosveremosabocadosaperderlotodoanoteestabanderaenigmamtresukwbvi

Encerrar en flag{}

flag{enigmamtresukwbvi}

## Solución

Vemos que hay una porción del texto que parece estar codificado en hexadecimal. (0.27)

# RETO 10

## Links

|  |
| --- |
| 1. <http://web.archive.org/web/20120624074941/http://home.comcast.net/~acabion/refscore.html> |
| 1. <http://practicalcryptography.com/cryptanalysis/text-characterisation/identifying-unknown-ciphers/> |
| 1. <https://bionsgadgets.appspot.com/ww_forms/aristo_pat_web_worker3.html> |

# Cifrado hexadecimal

<https://securityhacklabs.net/articulo/rompiendo-la-criptografia-sin-herramientas>

# Cifrado base 64

Supongamos que en la descripción de un texto les coloquen: (64 esab).  
Cómo pueden notar dice base 64 al revés, pero es fácil identificar cuando es base 64 ya que se suele ver lo que es mayúsculas y el signo ‘=’.

**Ejemplo**

Un mensaje en base 64 se vería de esta forma: RXN0byBlcyB1biB0ZXN0IHBhcmEgY29tcHJvYmFyIGVsIGJhc2U2NA==

Es identificable porque se compone de mayúsculas, minúsculas, números y termina en =.

Prueba con:

<https://gchq.github.io/CyberChef/#recipe=To_Base64('A-Za-z0-9%2B/%3D')From_Base64('A-Za-z0-9%2B/%3D',true)&input=RXN0byBlcyB1biB0ZXN0IHBhcmEgY29tcHJvYmFyIGVsIGJhc2U2NA>

# Binarios

Ya sabemos que los binarios se componen de 0 y 1 por lo que identificar cuando un texto es en binario es sencillo.

**Ejemplo**

01100101 01110011 01110100 01101111 00100000 01100101 01110011 00100000 01110101 01101110 00100000 01101101 01100101 01101110 01110011 01100001 01101010 01100101 00100000 01100101 01101110 00100000 01100010 01101001 01101110 01100001 01110010 01101001 01101111 

Prueba con:

<https://gchq.github.io/CyberChef/#recipe=From_Binary('Space',8)&input=MDExMDAxMDEgMDExMTAwMTEgMDExMTAxMDAgMDExMDExMTEgMDAxMDAwMDAgMDExMDAxMDEgMDExMTAwMTEgMDAxMDAwMDAgMDExMTAxMDEgMDExMDExMTAgMDAxMDAwMDAgMDExMDExMDEgMDExMDAxMDEgMDExMDExMTAgMDExMTAwMTEgMDExMDAwMDEgMDExMDEwMTAgMDExMDAxMDEgMDAxMDAwMDAgMDExMDAxMDEgMDExMDExMTAgMDAxMDAwMDAgMDExMDAwMTAgMDExMDEwMDEgMDExMDExMTAgMDExMDAwMDEgMDExMTAwMTAgMDExMDEwMDEgMDExMDExMTEg>

01110100

128-64-32-16-8-4-2-1

0+64+32+16+0+4 = 116 – t

# Links

<https://www.seguridadwireless.net/php/conversor-universal-wireless.php>