

# HideToSee

## Método 1:

### Steganographic Decoder

This form decodes the payload that was hidden in a JPEG image or a WAV or AU audio file using the [encoder form](#). When you submit, you will be asked to save the resulting payload file to disk. This form may also help you guess at what the payload is and its file type...

Select a JPEG, WAV, or AU file to decode:

No file chosen

Password (may be blank):

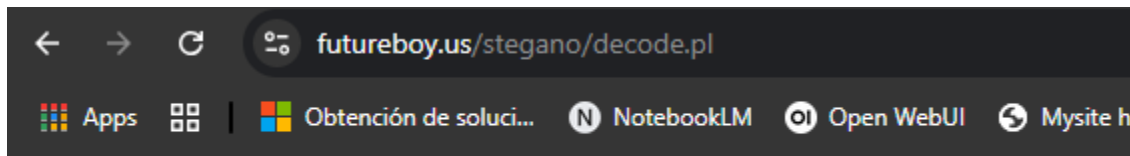
- ☒ View raw output as MIME-type   
☐ Guess the payload  
☐ Prompt to save (you must guess the file type yourself.)

To use this form, you must first [encode a file](#).

These pages use the [steghide](#) program to perform steganography, and the files generated are fully compatible with steghide.

Please send comments or questions to [Alan Eliassen](#).

[Back to Alan's Home Server](#)



krx1XGU{zgyzhs\_xizxp\_1u84w779}

Recipe

Atbash Cipher

Input

krx1XGU{zgyzhs\_xizxp\_1u84w779}

Output

picoCTF{atbash\_crack\_1f84d779}

Medium

Cryptography

picoCTF 2023

AUTHOR: SUNDAY JACOB NWANYIM

Hints 

## Description

1

How about some hide and seek heh?

Look at this image [here](#).

11,368 users solved



40%

Liked



picoCTF{atbash\_crack\_1f84d779}|

Submit  
Flag

Método 2:

Utilizamos:

1. steghide extract -sf atbash.jpg
2. binwalk -e atbash.jpg
3. exiftool atbash.jpg
4. strings atbash.jpg | les

Sacamos el resultado de:

krxIXGU{zgyzhs\_xizxp\_1u84w779}

Lo cual manualmente seguimos los siguientes pasos:

- k → p

- $r \rightarrow i$
- $x \rightarrow c$
- $l \rightarrow o$
- $X \rightarrow C$
- $G \rightarrow T$
- $U \rightarrow F$
- $\{ \rightarrow \{$  (se mantiene igual)
- $z \rightarrow a$
- $g \rightarrow t$
- $y \rightarrow b$
- $z \rightarrow a$
- $h \rightarrow s$
- $s \rightarrow h$
- $\_ \rightarrow \_$  (se mantiene igual)
- $x \rightarrow c$
- $i \rightarrow r$
- $z \rightarrow a$
- $x \rightarrow c$
- $p \rightarrow k$
- $\_ \rightarrow \_$
- $1 \rightarrow 1$  (números no cambian)
- $u \rightarrow f$
- $8 \rightarrow 8$
- $4 \rightarrow 4$
- $w \rightarrow d$
- $7 \rightarrow 7$
- $7 \rightarrow 7$
- $9 \rightarrow 9$
- $\} \rightarrow \}$  (se mantiene igual)

El mensaje unido es:

picoCTF{atbash\_crack\_1f84d779}

rsa Oracle

Método 1:

**Obtener el texto cifrado de la contraseña:** El archivo password.enc contiene la contraseña cifrada.

**Seleccionar un mensaje conocido y cifrarlo:** Elegimos el número 2 como mensaje conocido y lo ciframos utilizando el oráculo RSA para obtener  $c_a$ .

**Manipular el texto cifrado:** Calculamos el producto de  $c_a$  y el texto cifrado de la contraseña ( $c$ ), obteniendo  $c_b = c_a * c$ .

**Descifrar el texto manipulado:** Enviamos  $c_b$  al oráculo para su descifrado, obteniendo  $m_b$ .

**Recuperar la contraseña:** Dividimos  $m_b$  entre el mensaje original (2) para obtener la contraseña en texto plano.

Código de Python:

```
C: > Users > Matias > Downloads > import sys.py > ...
3  context.log_level = 'critical'
4  p = remote("titan.picoctf.net", 62026)
5
6  p.recvuntil(b"decrypt.")
7
8  with open("password.enc") as file:
9      c = int(file.read())
10
11  p.sendline(b"E")
12  p.recvuntil(b"keysize: ")
13  p.sendline(b"\x02")
14  p.recvuntil(b"mod n ")
15
16  c_a = int(p.recvline())
17
18  p.sendline(b"D")
19  p.recvuntil(b"decrypt: ")
20  p.sendline(str(c_a * c).encode())
21  p.recvuntil(b"mod n: ")
22
23  password = int(p.recvline(), 16) // 2
24  password = password.to_bytes((password.bit_length() + 7) // 8, "big").decode("utf-8").strip()
25
26  print("Password:", password)
27
```

Descifrar:

openssl enc -aes-256-cbc -d -in secret.enc -pass pass:<contraseña>

Medium

Cryptography

picoCTF 2024

browser\_webshell\_solvable

AUTHOR: GEOFFREY NJOGU

## Description

Can you abuse the oracle?

An attacker was able to intercept communications between a bank and a fintech company. They managed to get the [message](#) (ciphertext) and the [password](#) that was used to encrypt the message.

Additional details will be available after launching your challenge instance.

This challenge launches an instance on demand.

Its current status is:

NOT\_RUNNING

Launch  
Instance

Hints 

1 2 3 4

1,952 users solved



95%  
Liked



picoCTF{su((3ss\_(r@ck1ng\_r3@\_da099d93}

Submit

Método 2:

Modulo Común

Obtenemos los valores de los exponentes  $e_1$ ,  $e_2$ , y los textos cifrados  $C_1$ ,  $C_2$ .

Aplicamos la identidad de Bezout para obtener los coeficientes  $x$  y  $y$  tales que:  
 $x \cdot e_1 + y \cdot e_2 = 1$

Usamos los coeficientes  $x$  y  $y$  para calcular el mensaje original  $M$  utilizando exponentes negativos modularmente (inversos modulares).

Código de Python:

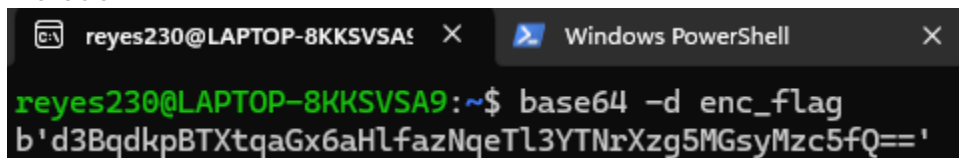
```

1  from Crypto.Util.number import inverse, long_to_bytes
2  import gmpy2
3
4  # Valores dados en el problema
5  N = 0x... # Colocar el valor del módulo
6  e1 = 0x... # Primer exponente
7  e2 = 0x... # Segundo exponente
8  C1 = 0x... # Primer texto cifrado
9  C2 = 0x... # Segundo texto cifrado
10
11 # Calcular coeficientes de la identidad de Bezout
12 g, x, y = gmpy2.gcdext(e1, e2)
13
14 # Asegurar que x es positivo
15 if x < 0:
16     x = -x
17     C1 = inverse(C1, N)
18 if y < 0:
19     y = -y
20     C2 = inverse(C2, N)
21
22 # Recuperar el mensaje original
23 M = (pow(C1, x, N) * pow(C2, y, N)) % N
24 print("Mensaje original:", long_to_bytes(M).decode())
25

```

interendec

Método 1:



```

reyes230@LAPTOP-8KKSUSA9: ~$ base64 -d enc_flag
b'd3BqdkpBTXtqaGx6aHlfazNqeTl3YTNrXzg5MGsyMzc5fQ=='

```

## Decode from Base64 format

Simply enter your data then push the decode button.



d3BqdkpBTXtqaGx6aHlfazNqeTl3YTNRXzg5MGsyMzc5fQ==

 For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8  Source character set.

☐ Decode each line separately (useful for when you have multiple entries).

☒ Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

 **DECODE**  Decodes your data into the area below.

wpjvJAM{jh1zhhy\_k3jy9wa3k\_890k2379}



### Search for a tool

★ SEARCH A TOOL ON dCODE BY KEYWORDS:

e.g. type 'sudoku'

★ BROWSE THE FULL dCODE TOOLS' LIST

### Results

Brute-Force mode: the 25 shifts (for the alphabet ABCDEFGHIJKLMNOPQRSTUVWXYZ) are tested and sorted from most probable to least probable.

↑↓	↑↓
→7 (←19)	picoCTF{caesar_d3cr9pt3d_890d2379}
→18 (←8)	exrdRIU{rpthpg_s3rg9ei3s_890s2379}

## CAESAR CIPHER

Cryptography > Substitution Cipher > Caesar Cipher

### CAESAR CIPHER DECODER

★ CAESAR SHIFTED CIPHERTEXT 

wpjvJAM{jh1zhhy\_k3jy9wa3k\_890k2379}

Test all possible shifts (26-letter alphabet A-Z)

► DECRYPT (BRUTEFORCE)

interencdec 



Easy

Cryptography

picoCTF 2024

base64

browser\_webshell\_solvable

caesar

AUTHOR: NGIRIMANA SCHADRACK

Hints 

## Description

1

Can you get the real meaning from this file.

Download the file [here](#).

33,977 users solved



87%

Liked



 picoCTF{caesar\_d3cr9pt3d\_890d2379}

Submit  
Flag

Método 2:

El cifrado Cesar lo puse en Python

```
1 def caesar_decrypt(ciphertext, shift):
2     decrypted = ""
3     for char in ciphertext:
4         if char.isalpha():
5             offset = 65 if char.isupper() else 97
6             decrypted += chr((ord(char) - offset - shift) % 26 + offset)
7         else:
8             decrypted += char
9     return decrypted
10
11 cipher = ""
12 for s in range(26):
13     print(f"Shift {s}: {caesar_decrypt(cipher, s)}")
14
```



