

Competición: Selección nacional ECSC2023

Solucionario reto 8

ÍNDICE

1. CARACTERÍSTICAS DE LA COMPETICIÓN	3
2. INFORMACIÓN DE RETO	4
3. SOLUCIÓN RETO 08	5

1. CARACTERÍSTICAS DE LA COMPETICIÓN

La competición "**Selección nacional para los ECSC2023**" tuvo lugar el 10 de junio de 2023 de forma online.

2. INFORMACIÓN DE RETO

Información general

- **Identificador:** Reto 08
- **Categoría:** Miscelánea
- **Puntuación:** 300
- **Dificultad:** Alta
- **Tipo:** Descargable

Conocimientos y habilidades

- **MITRE:** Discovery / Collection / Account Discovery
- **NICE:** Knowledge of Application Security Risks, Knowledge of hacking methodologies
- **ENISA:** Information and Knowledge Management, Information Security Management

3. SOLUCIÓN RETO 08

Enunciado

El equipo de seguridad del presidente del gobierno ha detectado un posible email malicioso, por suerte este dio error al ejecutarse, deberás investigarlo y sacar toda la información posible acerca de él.

Se nos piden unas coordenadas con 5 decimales.

Formato: flag{latitud, longitud}

Flag

flag{40.24521, -6.18136}

Pistas

1. Investiga los commits de los repositorios.
2. Cualquier información es valiosa. Vuelve hacia atrás.
3. Mira que hay detrás de las webs

Solución

En el archivo data.eml, encontramos un correo electrónico. Dentro del mismo, hay un archivo ejecutable .exe. Observamos que no se puede ejecutar, de manera que procedemos a abrirlo en el programa Ghidra.

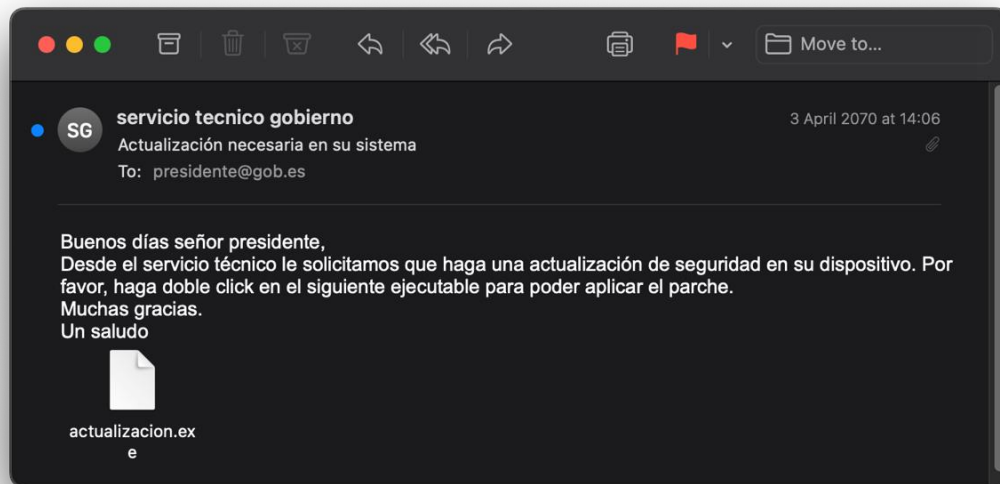


Ilustración 1 - Correo electrónico

Dentro de Ghidra, procedemos a realizar ingeniería inversa al binario para descubrir su funcionamiento y poder extraer la máxima información posible.

Debemos buscar la función “main” para empezar a entender el programa, podemos observar que se realizan varias llamadas al sistema.

```
Decompile: main - (actualizacion.exe)
1
2 int __cdecl main(int _Argc, char **_Argv, char **_Env)
3
4 {
5     size_t sVar1;
6     size_t sVar2;
7     char *_Dest;
8
9     __main();
10    system("cd \'.Desktop Goose v0.31DesktopGoose v0.31\");
11    sVar1 = strlen("hahaha.zip");
12    sVar2 = strlen("haha");
13    _Dest = (char *)malloc(sVar2 + sVar1 + 0x1e);
14    sprintf(_Dest, "tar -zxvf %s -C %s", "hahaha.zip", &DAT_140009054);
15    system(_Dest);
16    free(_Dest);
17    SetCurrentDirectoryA("haha");
18    system("GooseDesktop.exe");
19    return 0;
20 }
21
```

Ilustración 2 - Función main descompilada en Ghidra

Procedemos a fijarnos en los datos declarados en la función. Encontramos un enlace hacia GitHub.

```
*****
*                               *
*                               *
*****
int __cdecl main(int _Argc, char **_Argv, char **_Env)
    assume GS_OFFSET = 0xffff00000000
    EAX:4      <RETURN>
    ECX:4      _Argc
    EDI:4      _Argv
    R8:8       _Env
    Stack[-0x18]:1 local_18
    Stack[-0x20]:8 local_20
    Stack[-0x28]:8 local_28
    Stack[-0x30]:8 local_30
    Stack[-0x38]:8 local_38
    main
    XREF[3]:  __tmainCRTStartup:1400013bc(c),
    14000a1d8(*), 14000a1e0(*)
140001655 55      PUSH    RBP
140001656 53      PUSH    RBX
140001657 48 83 ec 48  SUB     RSP, 0x48
14000165b 48 8d 6c      LEA     RBP, local_18, [RSP + 0x40]
140001660 e8 7b 01      CALL    __main
    undefined __main(void)
140001665 48 8d 05      LEA     RAX, [s_https://github.com/shokamon/haha_140009... = "https://github.com/shokamon/h...
    94 79 00 00
14000166c 48 89 45 f8    MOV     qword ptr [RBP + local_20], RAX=>s_https://gith... = "https://github.com/shokamon/h...
140001670 48 8d 05      LEA     RAX, [s_hahaha.zip_140009049] = "hahaha.zip"
```

Ilustración 3 - Función main en Ghidra

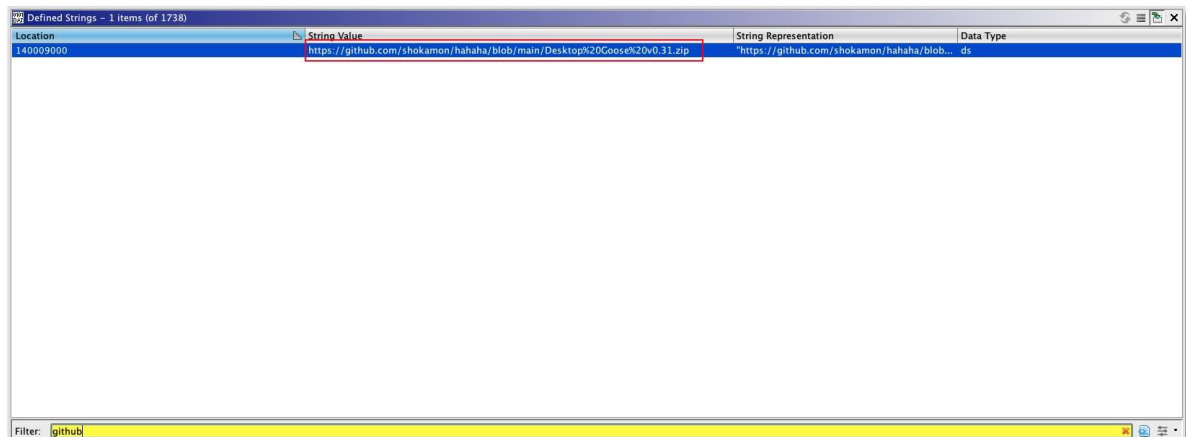


Ilustración 4 - Declaración de strings en Ghidra

Tras buscar las strings definidas, podemos ver el enlace completo, un enlace de descarga a un archivo .zip de un repositorio de GitHub.

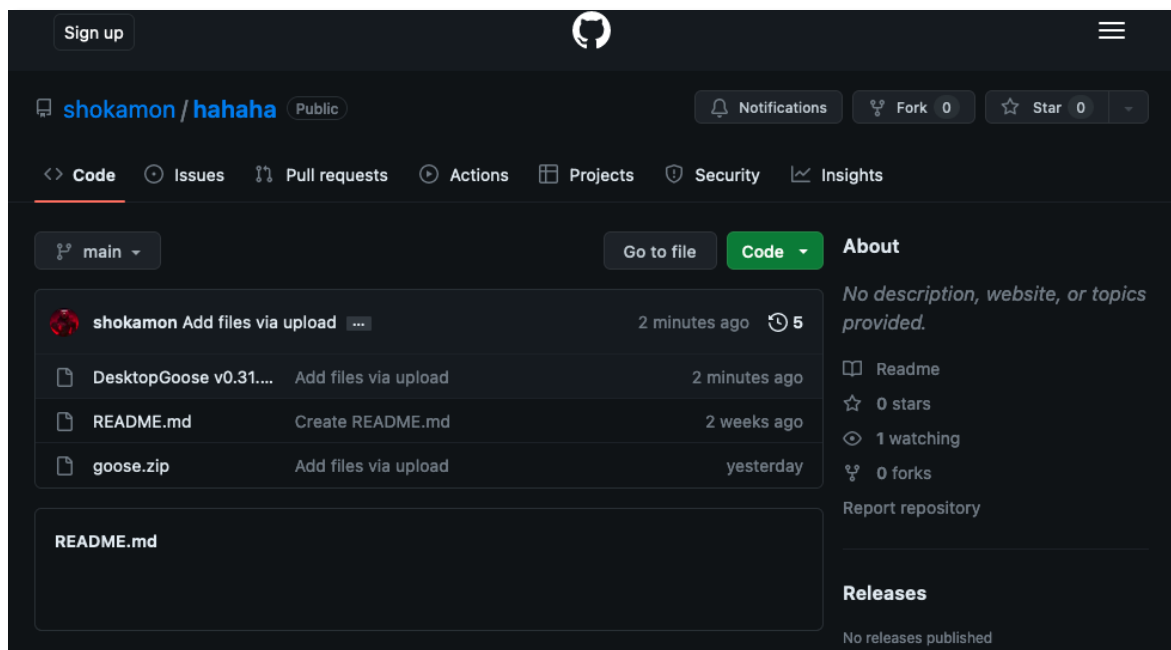


Ilustración 5 - Página del repositorio en GitHub

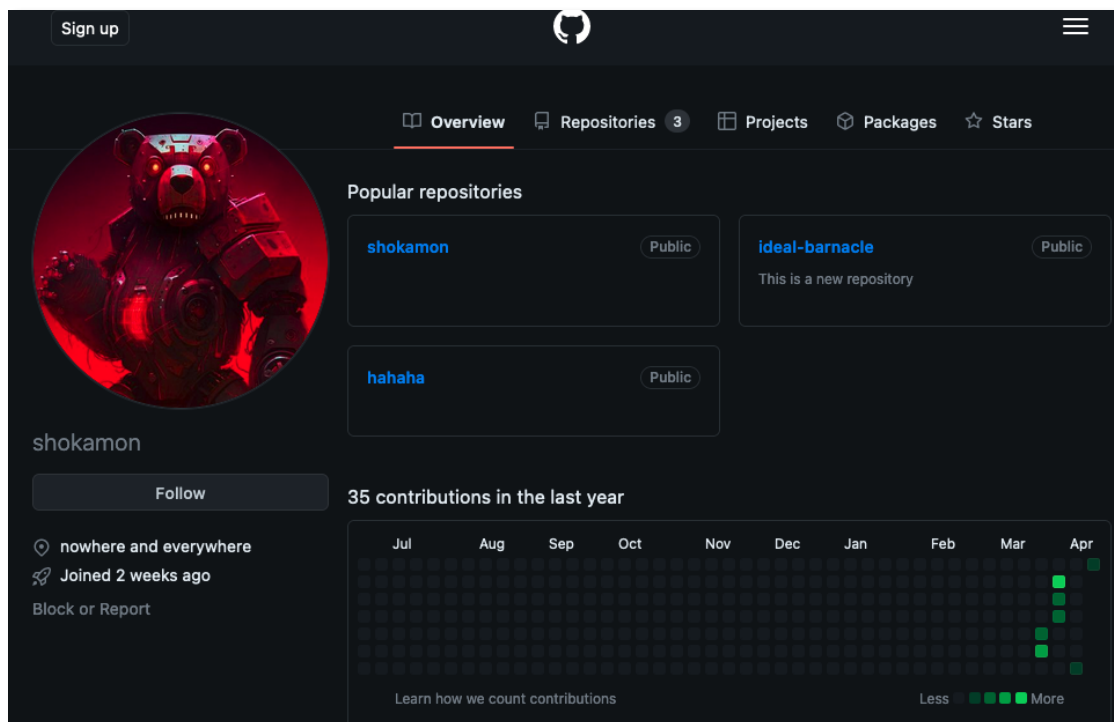


Ilustración 6 - Páginad de usuario en GitHub

Investigamos el perfil de github y encontramos 3 repositorios. En GitHub, el README es un archivo de texto breve que se encuentra en la raíz del repositorio y proporciona información sobre el proyecto.

El repositorio que tiene el mismo nombre que el usuario, actua como “página principal del perfil”, normalmente encontraremos en ellos información sobre el usuario, RRSS, etc.

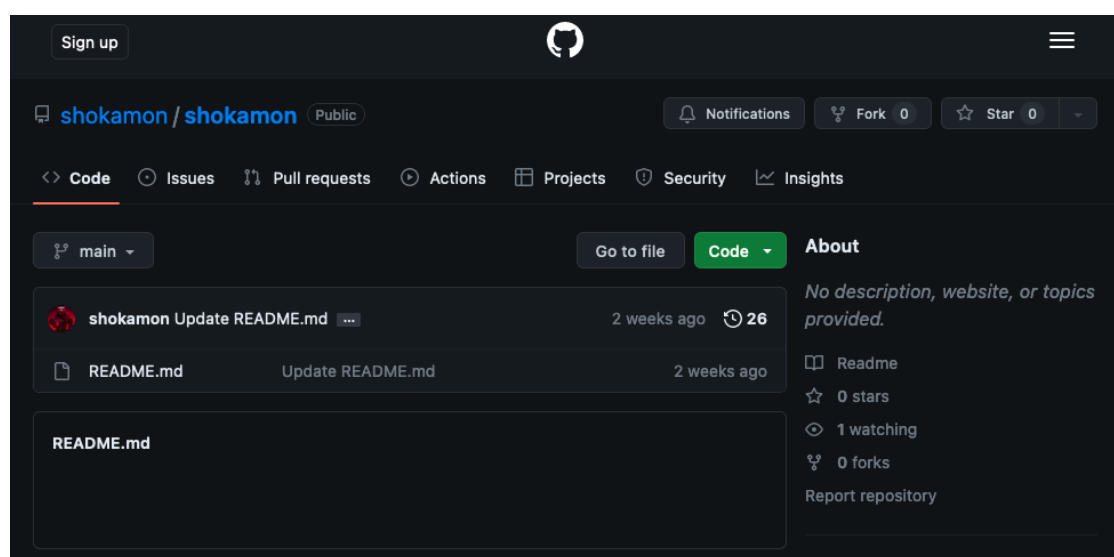


Ilustración 7 - README.md del usuario

Procedemos a observar los distintos cambios del repositorio, en el commit `acf58ab107cf11f678cafa29d428415de05b39f3` (anon), encontraremos que el usuario eliminó diversas cosas del `readme.md`, entre ellas un link a una cuenta de Twitter.

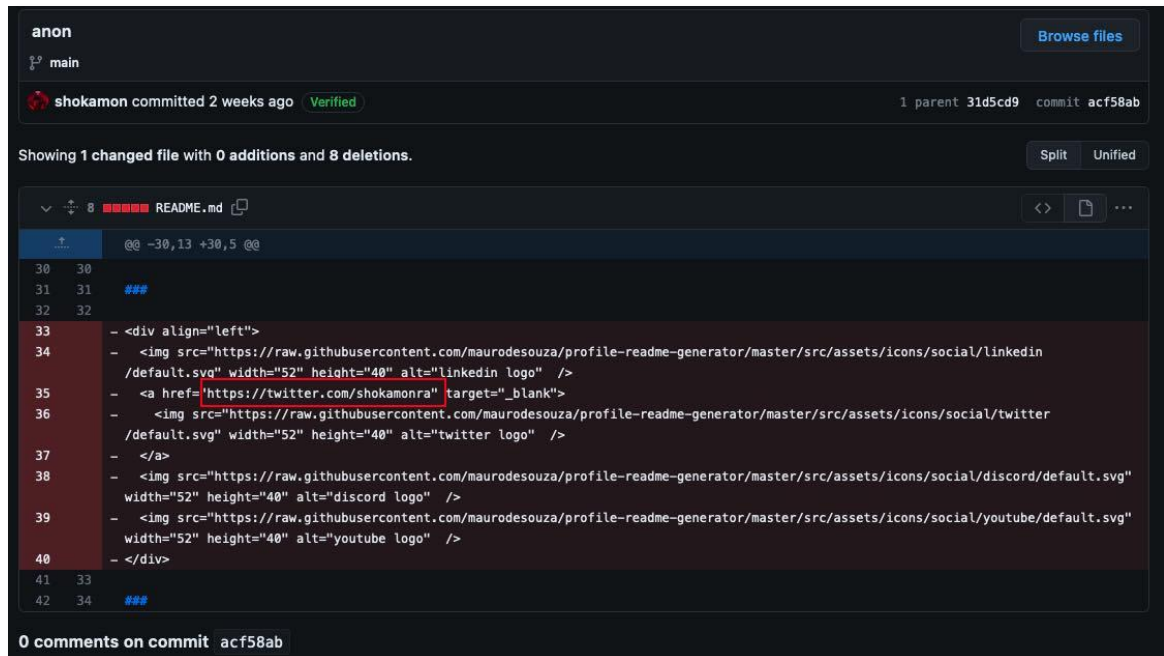


Ilustración 8 - Commit en el que aparece un enlace a Twitter



Ilustración 9 - Perfil de Twitter



Ilustración 7 - Tweet con enlace a Mastodon

Buscando en tweets antiguos, encontramos un enlace a la red social Mastodon.
<https://mastodon.social/@kashoma>

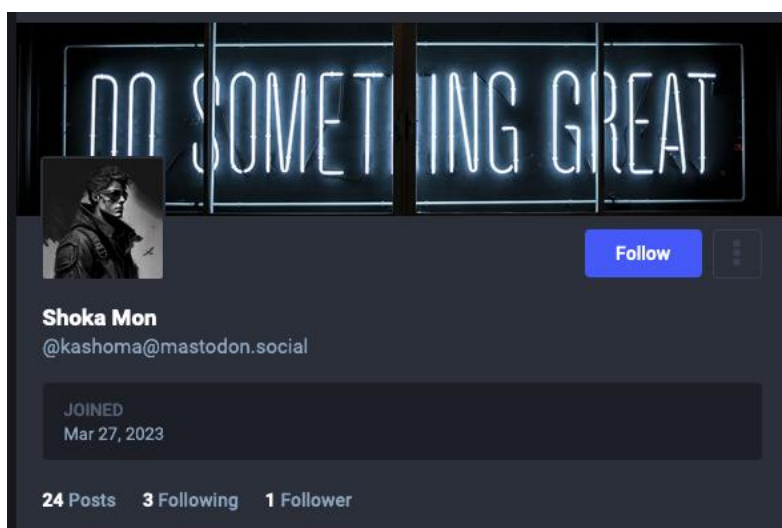


Ilustración 8 - Perfil de Mastodon

Analizando todas las publicaciones y las respuestas a ellas, encontramos que el usuario @lyrazenith, interactúa en diversas ocasiones con @kashoma.



Ilustración 10 - Interacción con otra cuenta

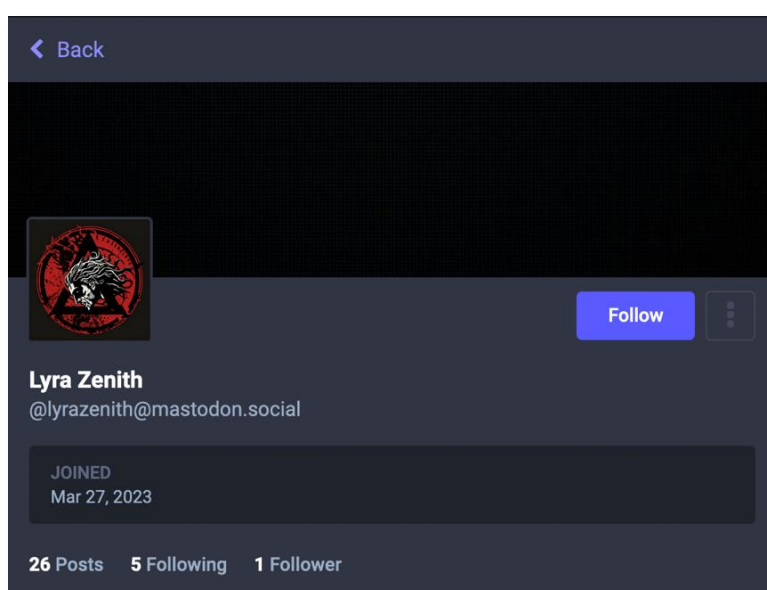


Ilustración 9 - Cuenta 2

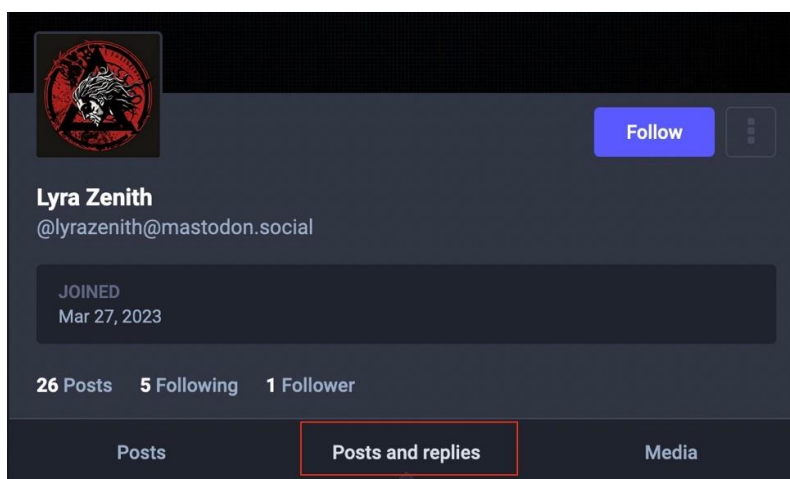


Ilustración 11 - Publicaciones y respuestas

Nos desplazamos a la sección de “Posts and replies” para analizar todas las interacciones de esta cuenta con otras.

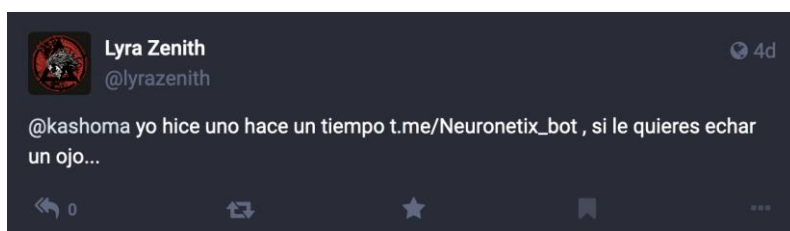


Ilustración 12 - Enlace a Bot de Telegram

En esta respuesta a @kashoma encontramos un enlace a un bot de telegram, t.me/Neuronetix_bot.

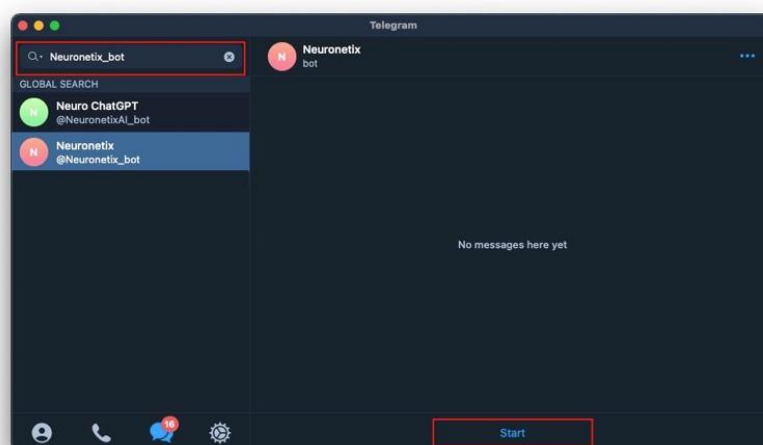


Ilustración 13 - Bot de Telegram

Tras iniciar el Bot, debemos intentar sacar la máxima información posible. Para ello ejecutamos el comando /help, que suele estar presente en la mayoría de Bots, nos pregunta que cual es nuestro nombre.

Ejecutamos el comando Lyra, ya que es la creadora del Bot y nos devuelve una string cifrada.

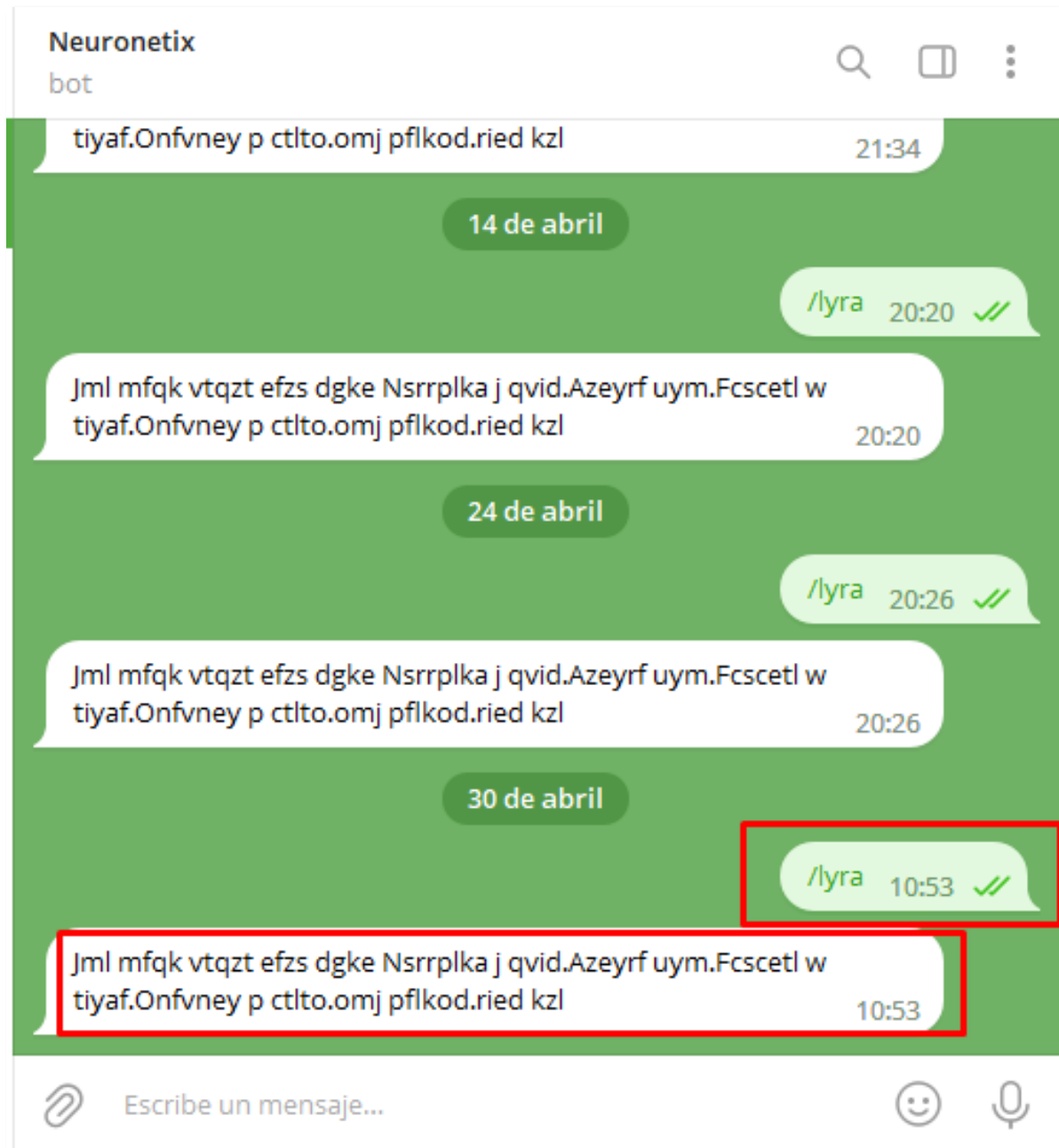


Ilustración 14 - Obtención de mensaje encriptado en Telegram

Para conseguir descifrarlo, debemos volver hacia atrás y analizar otra vez el perfil de GitHub. Además de todos los repositorios publicados en el perfil, encontramos los Gist, son pequeñas publicaciones de código. En este caso encontramos un programa escrito en Python que cifra strings.

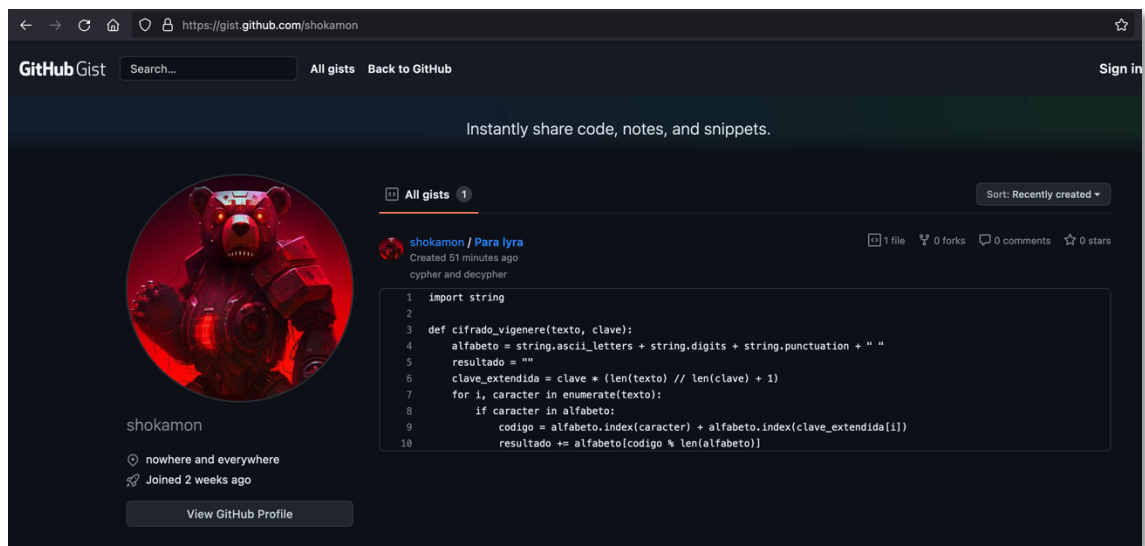


Ilustración 15 - Gist de GitHub

```
Para Lyra Raw
1 import string
2
3 def cifrado_vigenere(texto, clave):
4     alfabeto = string.ascii_letters + string.digits + string.punctuation + " "
5     resultado = ""
6     clave_extendida = clave * (len(texto) // len(clave) + 1)
7     for i, caracter in enumerate(texto):
8         if caracter in alfabeto:
9             codigo = alfabeto.index(caracter) + alfabeto.index(clave_extendida[i])
10            resultado += alfabeto[codigo % len(alfabeto)]
11        else:
12            resultado += caracter
13    return resultado
14
15
16 def descifrado_vigenere(texto, clave):
17     alfabeto = string.ascii_letters + string.digits + string.punctuation + " "
18     resultado = ""
19     clave_extendida = clave * (len(texto) // len(clave) + 1)
20     for i, caracter in enumerate(texto):
21         if caracter in alfabeto:
22             codigo = alfabeto.index(caracter) - alfabeto.index(clave_extendida[i])
23             resultado += alfabeto[codigo % len(alfabeto)]
24         else:
25             resultado += caracter
26    return resultado
27
28
29 texto_original = input("Ingrese el texto que desea encriptar: ")
30 clave = input("Ingrese la clave de encriptación: ")
31
32 texto_encriptado = cifrado_vigenere(texto_original, clave)
33 print("Texto encriptado:", texto_encriptado)
34
35 texto_desencriptado = descifrado_vigenere(texto_encriptado, clave)
36 print("Texto desencriptado:", texto_desencriptado)
37
```

Ilustración 16 - Cifrador publicado por Shoka

Probando el programa nos damos cuenta de que solicita un texto, una contraseña y devuelve el texto encriptado.

```
> python cypher.py
Ingrese el texto que desea encriptar: prueba
Ingrese la clave de encriptación: prueba
Texto encriptado: EI0ica
Texto desencriptado: prueba
```

Ilustración 17 - Muestra del funcionamiento del programa


```
import string

def cifrado_vigenere(texto, clave):
    alfabeto = string.ascii_letters + string.digits + string.punctuation + " "
    resultado = ""
    clave_extendida = clave * (len(texto) // len(clave) + 1)
    for i, caracter in enumerate(texto):
        if caracter in alfabeto:
            codigo = alfabeto.index(caracter) +
alfabeto.index(clave_extendida[i])
            resultado += alfabeto[codigo % len(alfabeto)]
        else:
            resultado += caracter
    return resultado

def descifrado_vigenere(texto, clave):
    alfabeto = string.ascii_letters + string.digits + string.punctuation + " "
    resultado = ""
    clave_extendida = clave * (len(texto) // len(clave) + 1)
    for i, caracter in enumerate(texto):
        if caracter in alfabeto:
            codigo = alfabeto.index(caracter) -
alfabeto.index(clave_extendida[i])
            resultado += alfabeto[codigo % len(alfabeto)]
        else:
            resultado += caracter
    return resultado

texto_encriptado = input("Ingrese el texto que desea desencriptar: ")
clave = input("Ingrese la clave de encriptación: ")

texto_desencriptado = descifrado_vigenere(texto_encriptado, clave)
print("Texto desencriptado:", texto_desencriptado)

texto_encriptado = cifrado_vigenere(texto_desencriptado, clave)
print("Texto encriptado:", texto_encriptado)
```

La cadena encontrada en el bot de Telegram, es un cifrado “Vigenère” y su clave es “lyra”, al descifrarlo obtenemos el siguiente mensaje.

Recipe

Vigenère Decode

Key
lyra

Input
JmL mFqk vtqzt efzs dgke Nsrpika j qvid.Azeyrf uym.Fcscetl w tiyaf.Onfvney p ctltio.omj pfikod.rled kzl

Output
You must visit this site Cuarenta y seis.Ciento uno.Ochenta y cinco.Ochenta y cinco.dos puntos.tres mil

You must visit this site **Cuarenta y seis.Ciento uno.Ochenta y cinco.Ochenta y cinco.dos puntos.tres mil**

Lo que es lo mismo, esta URL:

<http://46.101.85.85:3000/>

Siguiendo este enlace, nos lleva a un formulario de inicio de sesión con una sección de forgot password.

FORGOT PASSWORD?

what is your pet's name

Submit

Ilustración 18 - Inicio de sesión del enlace recibido

Nos pide el nombre de nuestra mascota para recuperar la contraseña.

Ilustración 19 - Página de contraseña olvidada

Volviendo a los perfiles de mastodon, encontramos un post en el cual Lyra, publica el nombre de su gato.



Ilustración 20 - Nombre de la mascota de Lyra

Tras introducir este nombre la página no realiza ningún cambio, analizando el monitor de red de las herramientas para desarrolladores del navegador, observamos una ruta hacia “secret”.

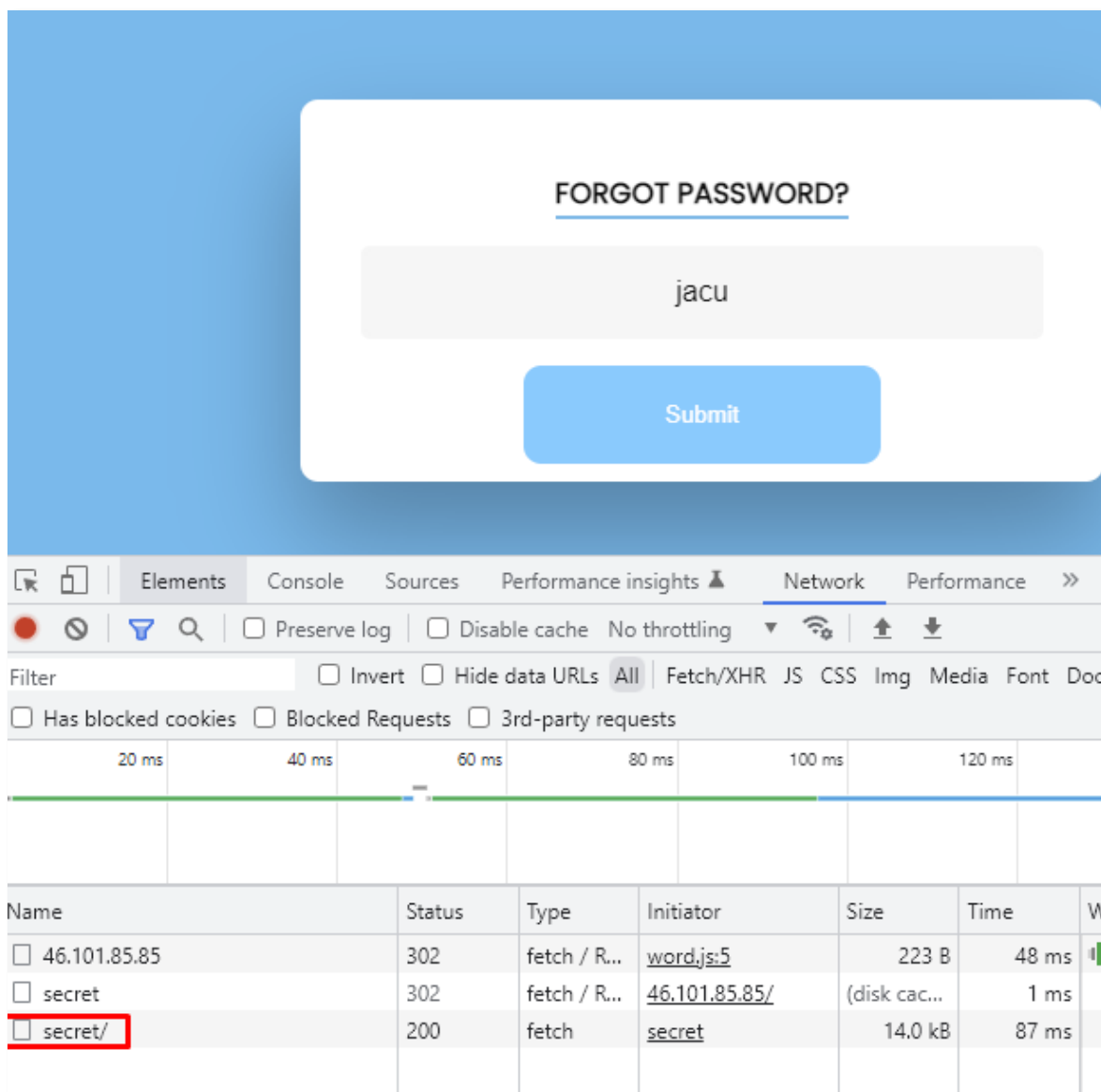


Ilustración 21: Vista de las herramientas de desarrollador con la petición creada.

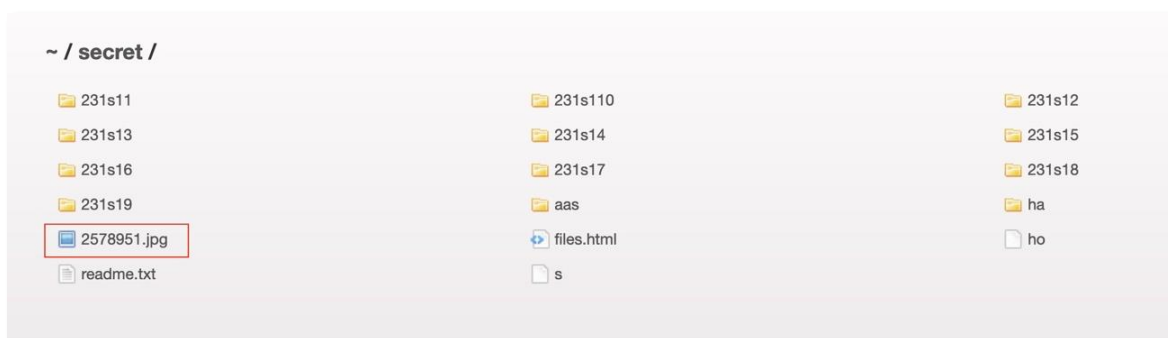


Ilustración 22: Vista de los archivos.



Ilustración 23: Imagen del servidor.

Para analizar la imagen utilizaremos la herramienta Exiftool para extraer los metadatos.

```
> exiftool 2578951.jpg
ExifTool Version Number      : 12.48
File Name                    : 2578951.jpg
Directory                    : .
File Size                    : 87 kB
File Modification Date/Time   : 2023:04:10 09:35:02+02:00
File Access Date/Time        : 2023:04:10 09:35:05+02:00
File Inode Change Date/Time   : 2023:04:10 09:35:04+02:00
File Permissions              : -rw-r--r--
File Type                    : JPEG
File Type Extension          : jpg
MIME Type                    : image/jpeg
JFIF Version                  : 1.01
Exif Byte Order               : Big-endian (Motorola, MM)
Make                          : Apple
Camera Model Name             : iPhone 13
X Resolution                  : 1
Y Resolution                  : 1
Resolution Unit               : None
Y Cb Cr Positioning           : Centered
GPS Version ID                : 2.3.0.0
GPS Latitude Ref              : North
GPS Longitude Ref             : West
Image Width                   : 664
Image Height                  : 498
Encoding Process              : Progressive DCT, Huffman coding
Bits Per Sample               : 8
Color Components              : 3
Y Cb Cr Sub Sampling          : YCbCr4:2:0 (2 2)
Image Size                    : 664x498
Megapixels                    : 0.331
GPS Latitude                   : 40 deg 14' 42.76" N
GPS Longitude                  : 6 deg 10' 52.90" W
GPS Position                   : 40 deg 14' 42.76" N, 6 deg 10' 52.90" W
```

Ilustración 24: Coordenadas GPS en metadatos.

Obtenemos las coordenadas en las que se tomo esta imagen. Debemos convertirlo a dotación “Decimal Degrees” en lugar de DMS (Degrees, minutes, seconds)

Ilustración 25 - Metadatos de la imagen

Degrees Minutes Seconds to Decimal Degrees

Please enter the **degrees, minutes, seconds (DMS)** coordinates values to

Degrees for Latitude	Minutes	Seconds
40 °	14 '	42.76 "
Degrees for Longitude	Minutes	Seconds
6 °	10 '	52.90 "
Convert to Decimal Degrees		
Decimal Degrees Lat 40.24521111 °	Decimal Degrees Long 6.18136111 °	

[f](#) [t](#)

Ilustración 26: Convertidor de coordenadas a dotación decimal de la web gps-coordinates.org

DD (decimal degrees)

Latitude 40.24521111111111

Longitude -6.181361111111111

Get Address

DMS (degrees, minutes, seconds)

Latitude ☒ N ☐ S 40 ° 14 ' 42.759 "

Longitude ☐ E ☒ W 6 ° 10 ' 52.899 "

Get Address

Ilustración 27: Muestra de la localización.

Flag: flag{40.24521, -6.18136}