Competición:

Selección nacional ECSC2023

Solucionario reto 8

Financiado por la Unión Europea
NextGenerationEU

GOBIERNO DE ESPAÑA
VICEPRESIDENCIA PRIMERA DEL GOBIERNO
MINISTERIO DE ASUNTOS ECONÓMICOS Y TRANSFORMACIÓN DIGITAL
SECRETARÍA DE ESTADO DE DIGITALIZACIÓN E INTELIGENCIA ARTIFICIAL

Plan de Recuperación, Transformación y Resiliencia

INSTITUTO NACIONAL DE CIBERSEGURIDAD

ÍNDICE

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.

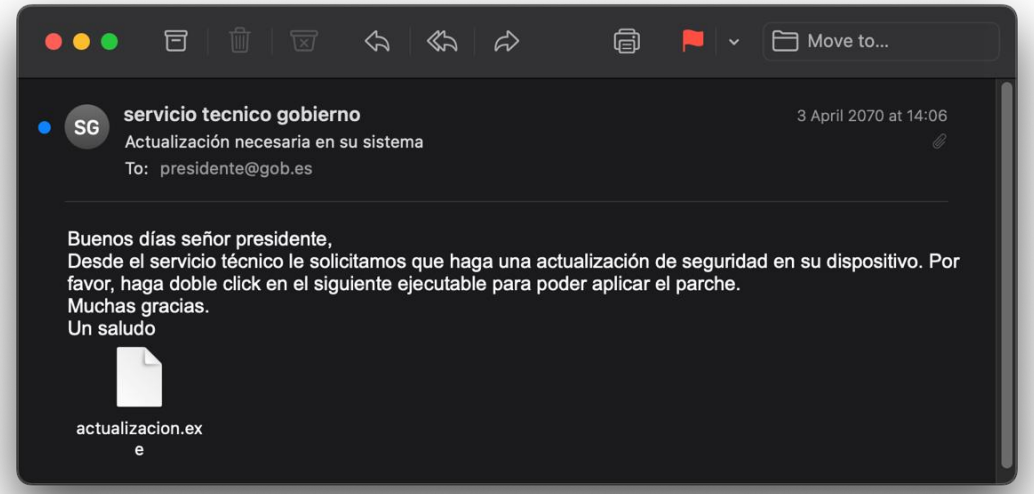


Ilustración 1 - Correo electrónico

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.

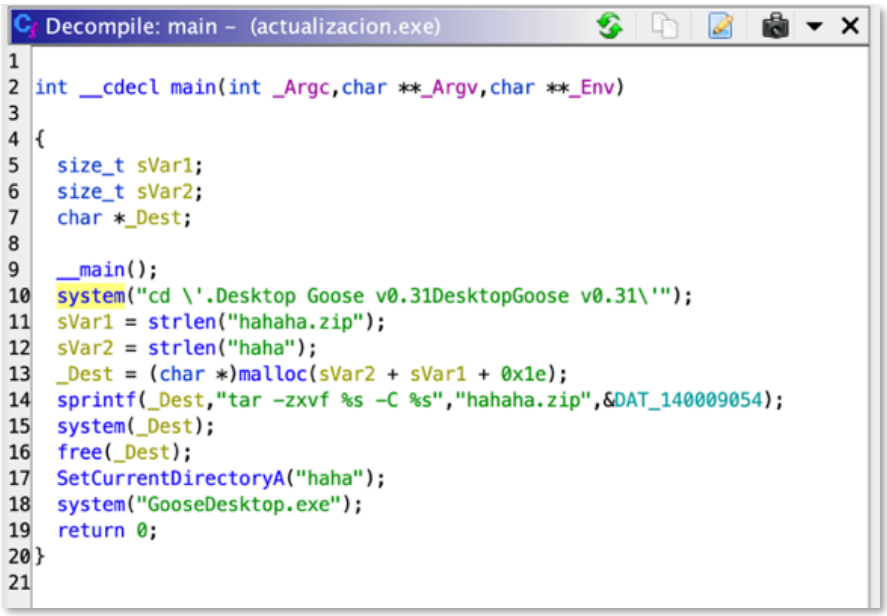


Ilustración 2 - Función main descompilada en Ghidra

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

Ilustración 3 - Función main en Ghidra

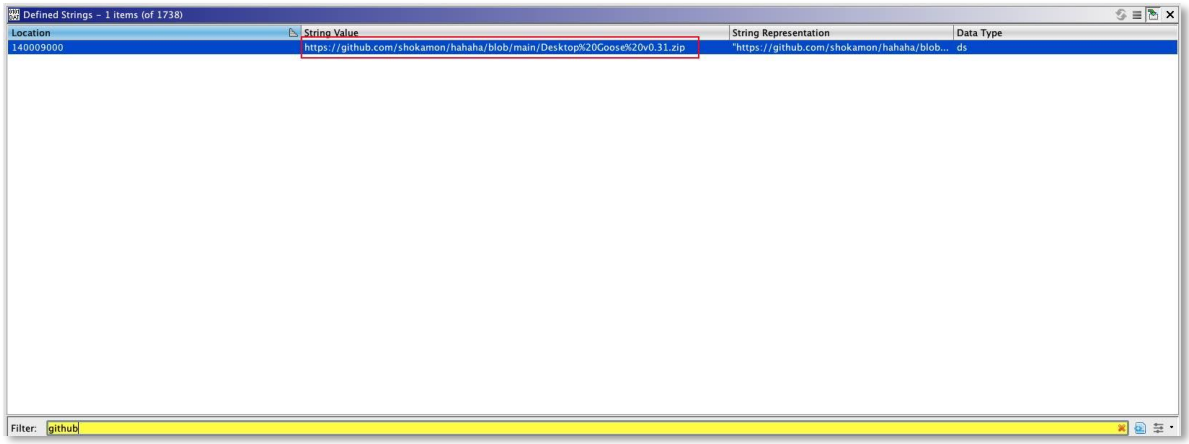


Ilustración 4 - Declaración de strings en Ghidra

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

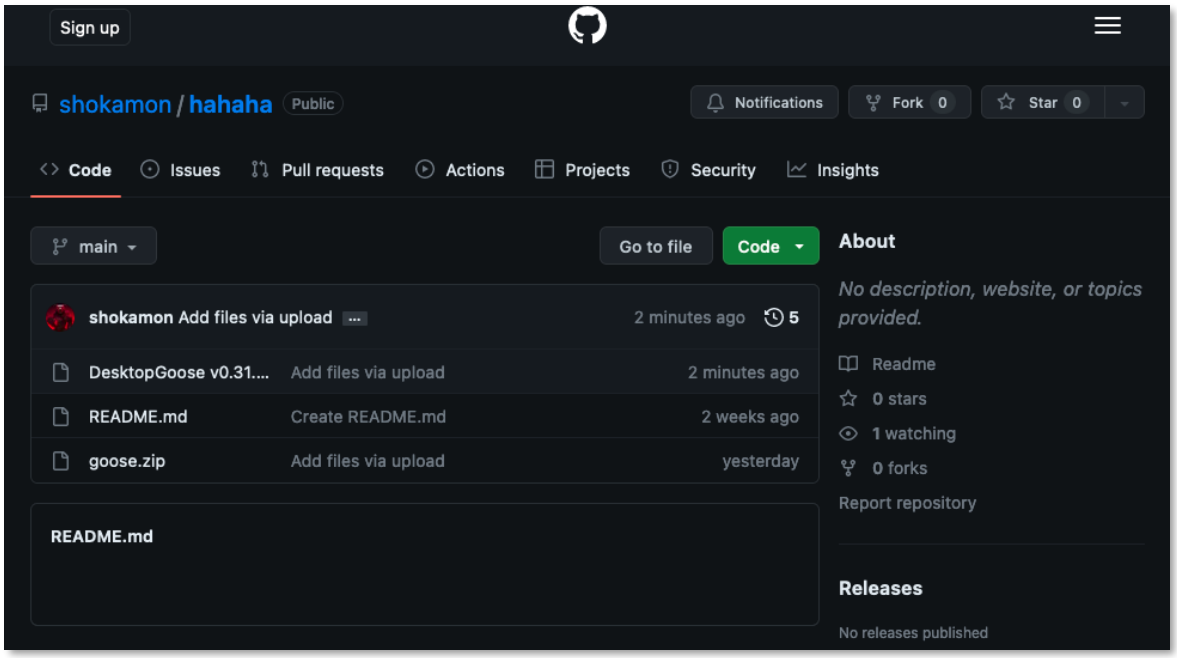


Ilustración 5 - Página del repositorio en GitHub

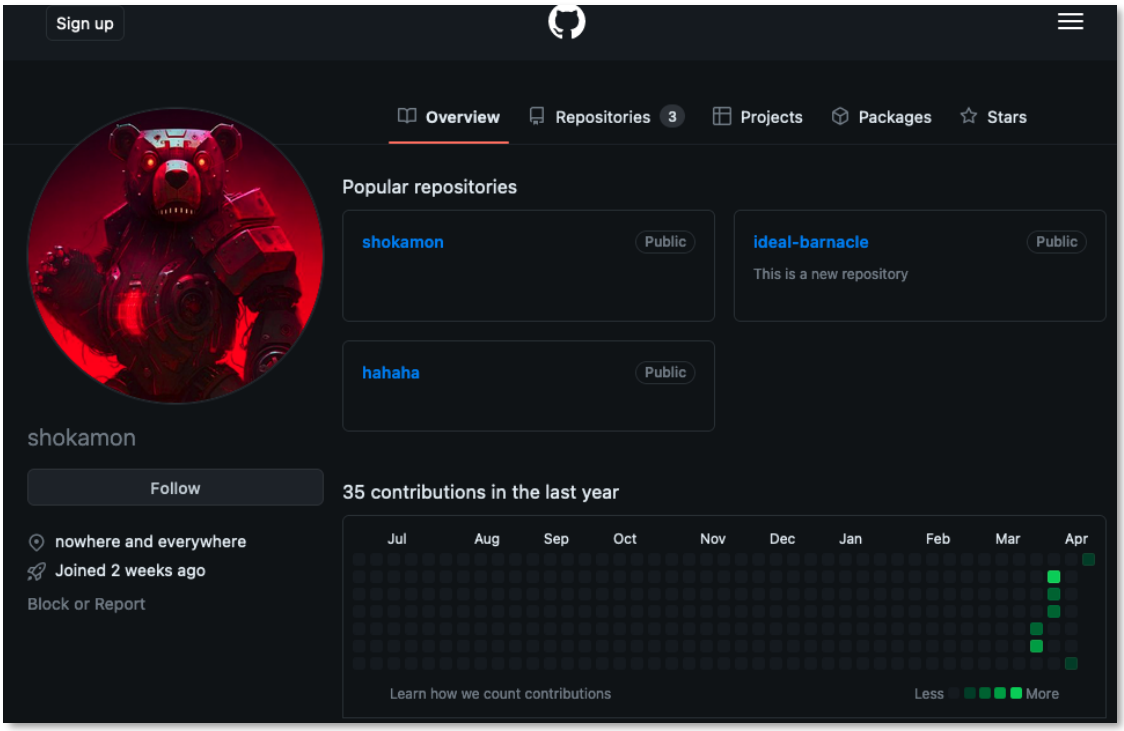


Ilustración 6 - Páginad de usuario en GitHub

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.

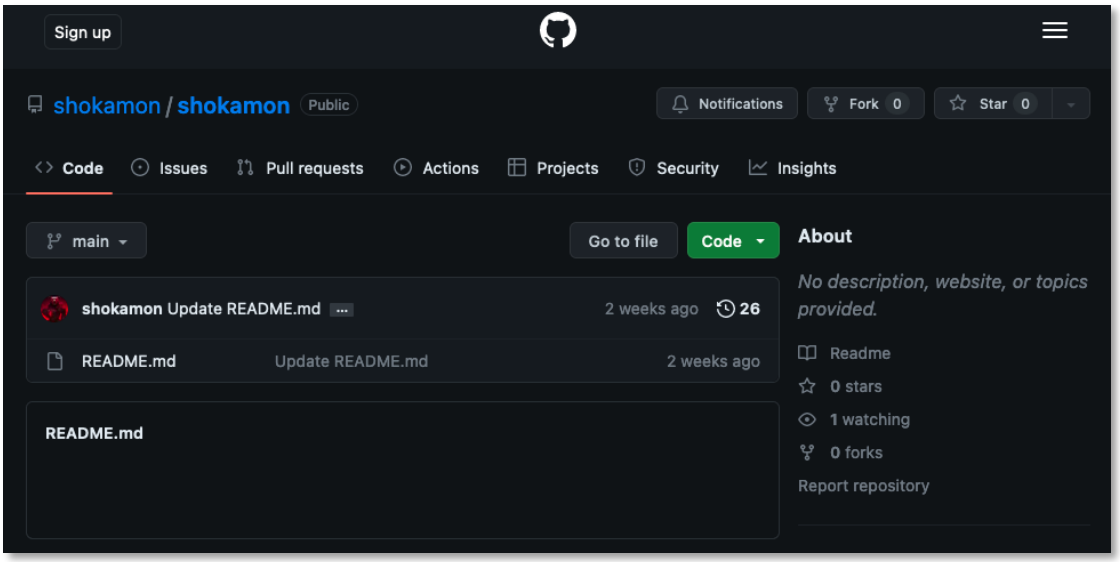


Ilustración 7 - README.md del usuario

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.

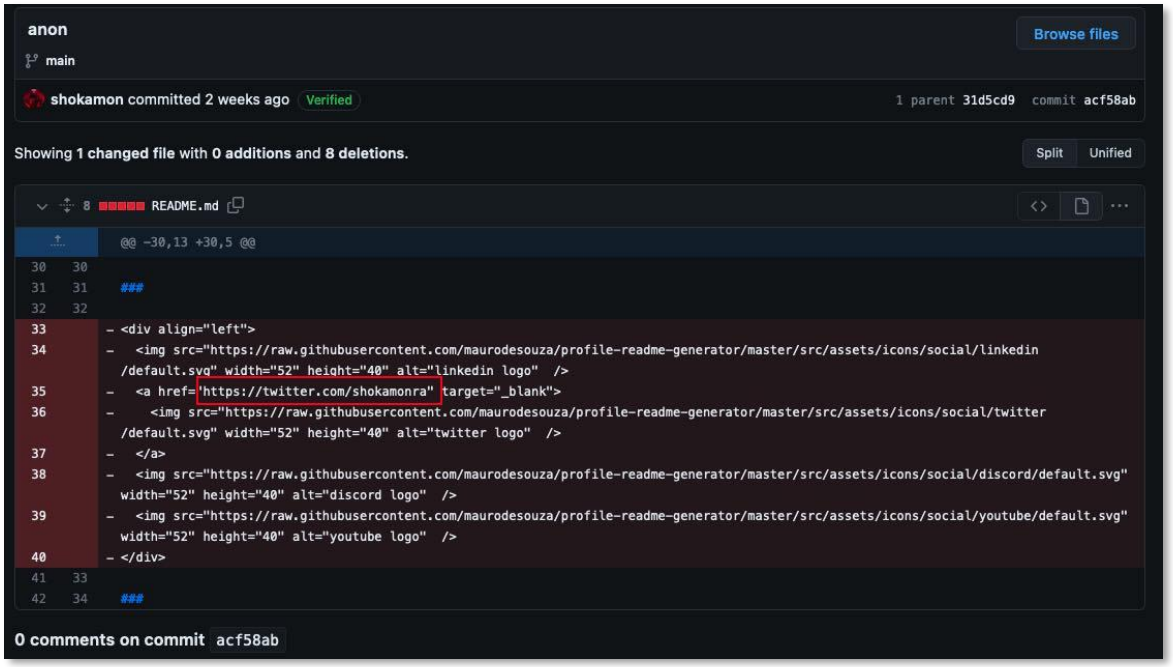


Ilustración 8 - Commit en el que aparece un enlace a Twitter

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

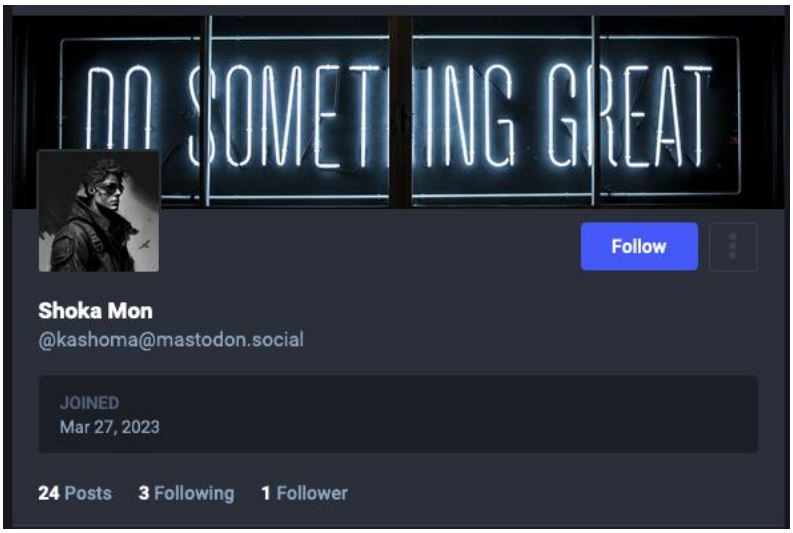


Ilustración 8 - Perfil de Mastodon

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

Ilustración 10 - Interacción con otra cuenta

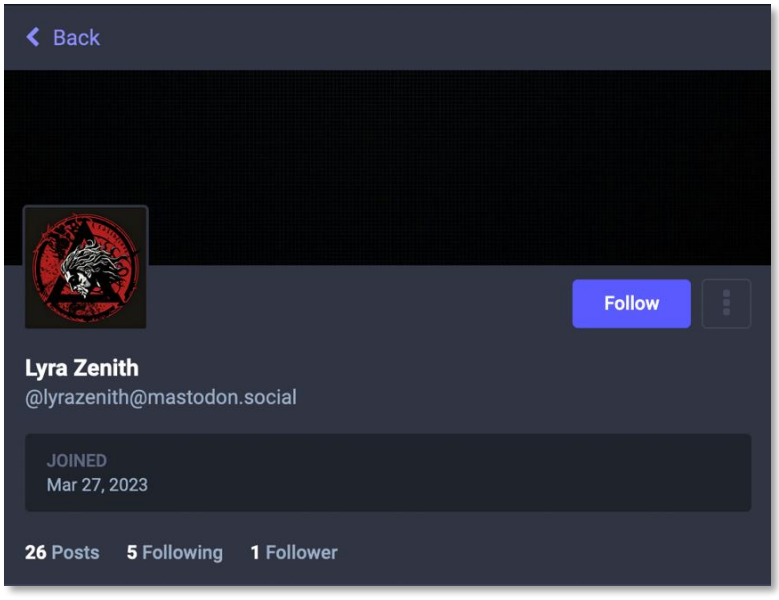


Ilustración 9 - Cuenta 2

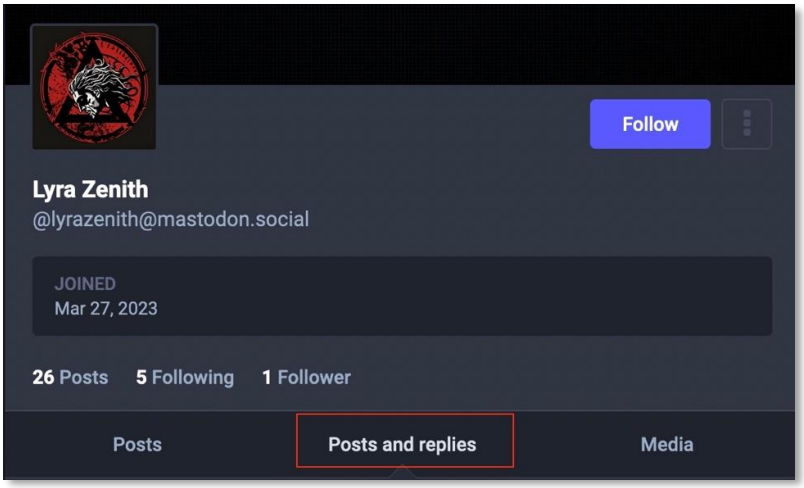


Ilustración 11 - Publicaciones y respuestas

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.

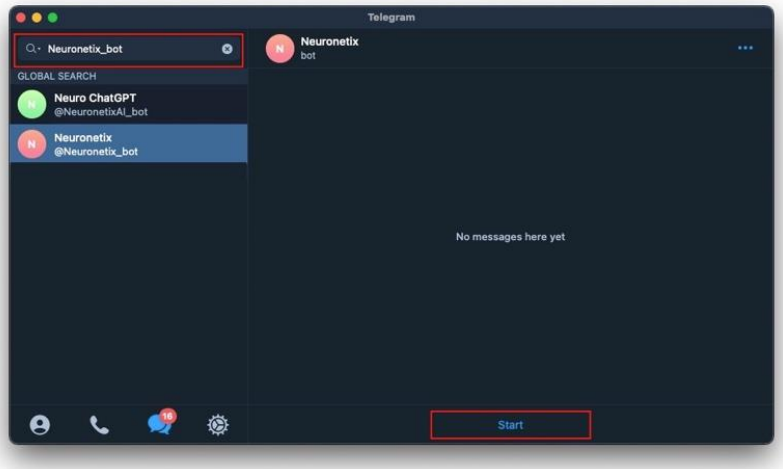


Ilustración 13 - Bot de Telegram

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.

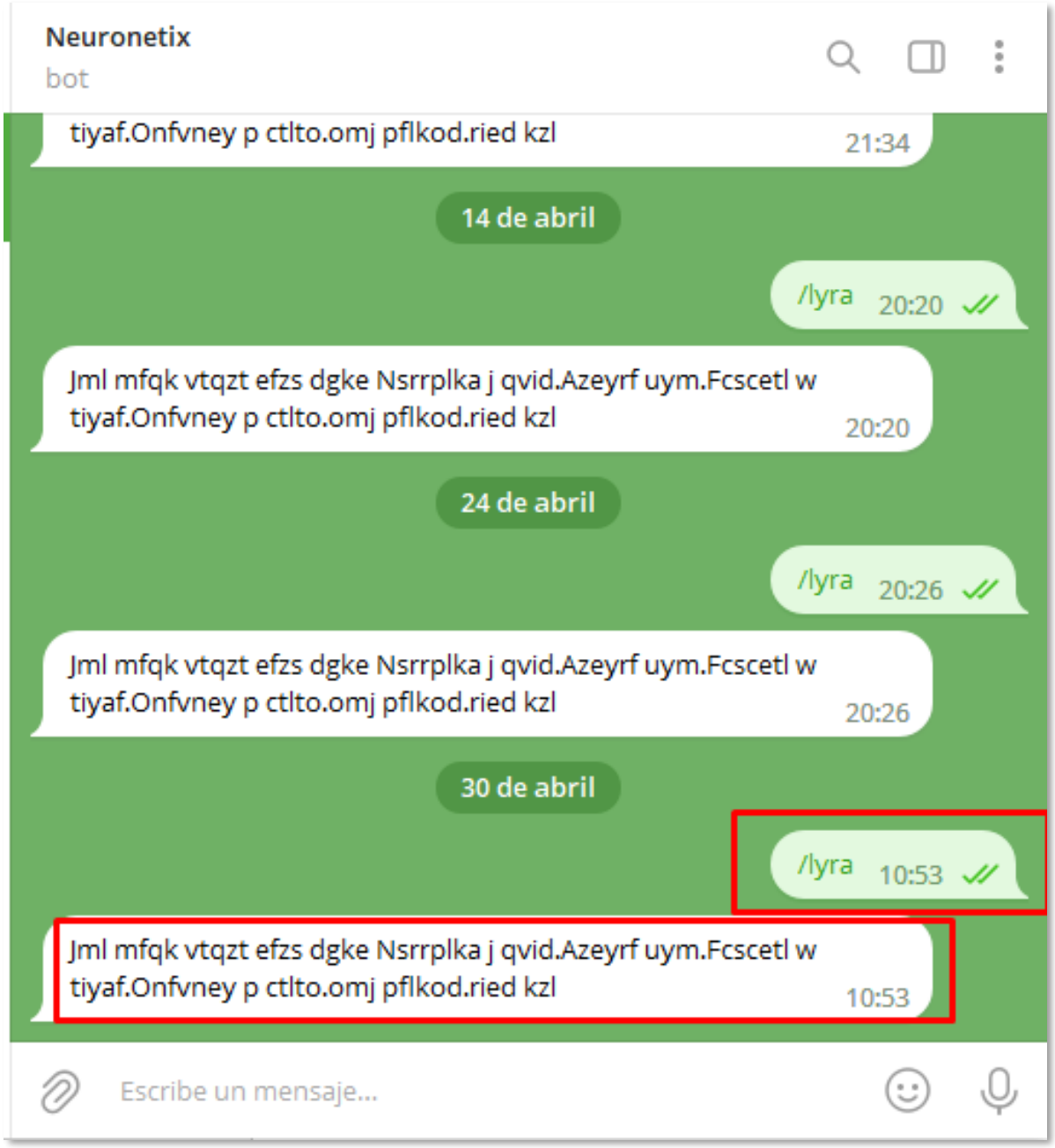


Ilustración 14 - Obtención de mensaje encriptado en Telegram

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.

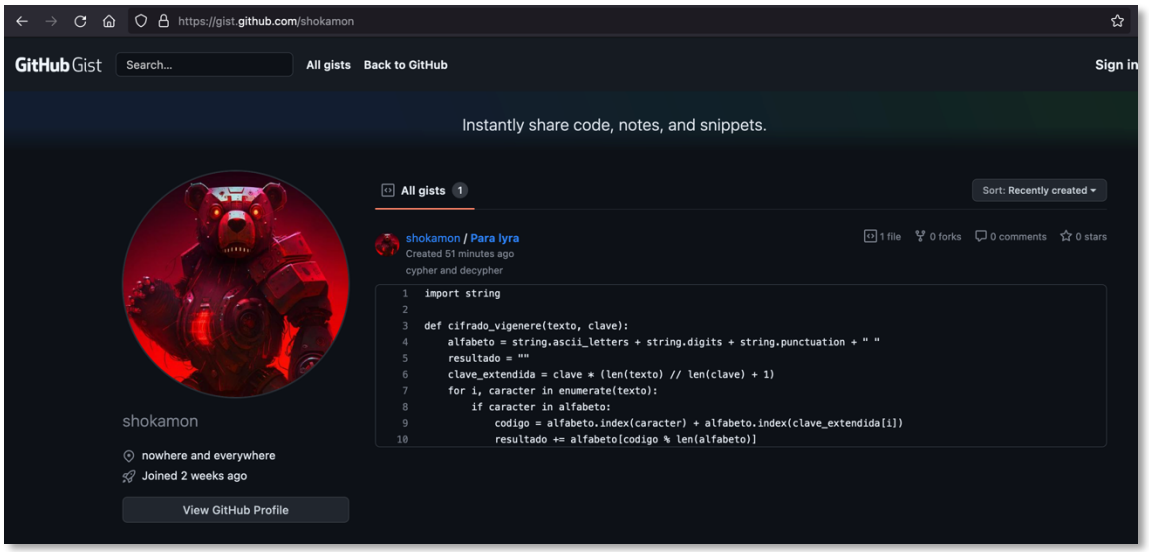


Ilustración 15 - Gist de GitHub

Para lyra — Raw

```python
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_original = input("Ingrese el texto que desea encriptar: ")
clave = input("Ingrese la clave de encriptación: ")

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

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

Ilustración 16 - Cifrador publicado por Shoka

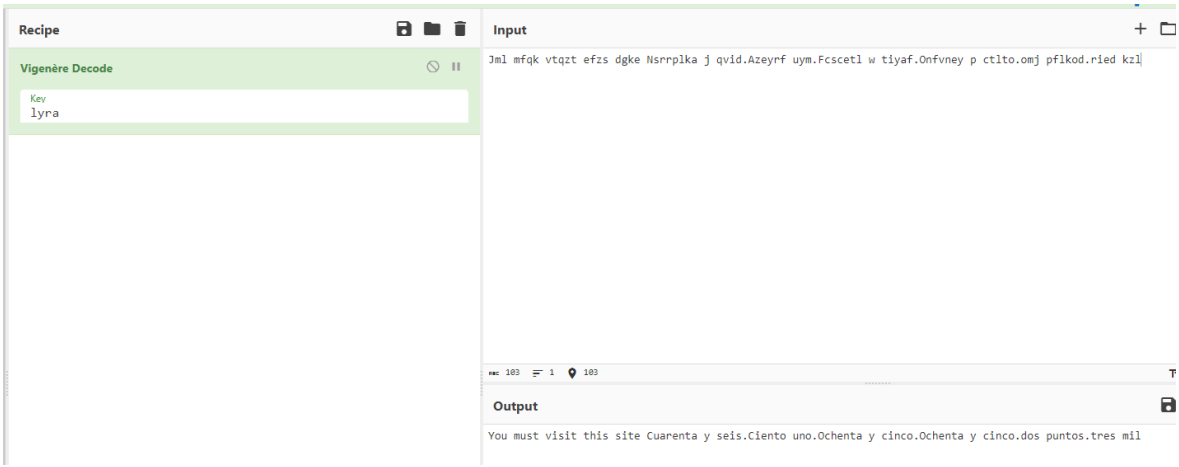
Probando el programa nos damos cuenta de que solicita un texto, una contraseña y devuelve el texto encriptado.

```
❯ python cypher.py
Ingrese el texto que desea encriptar: prueba
Ingrese la clave de encriptación: prueba
Texto encriptado: EIOica
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.




```
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.

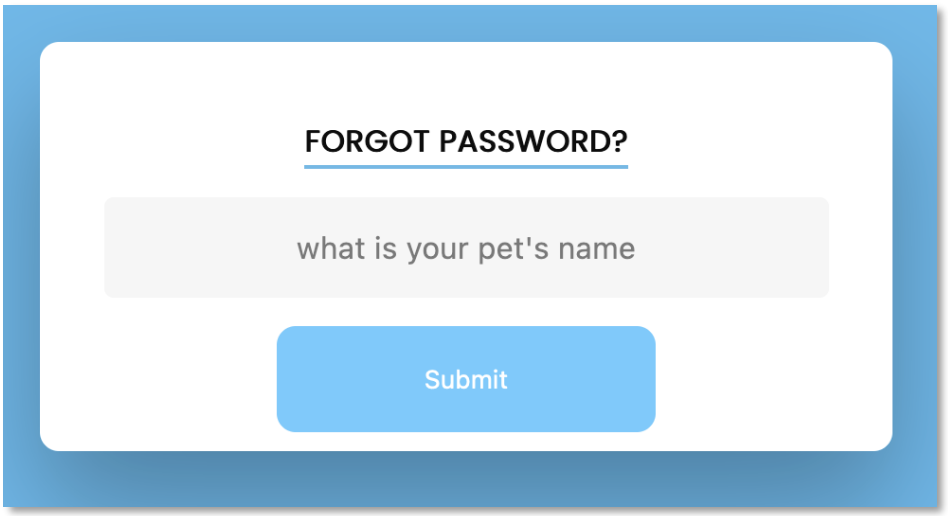


Ilustración 18 - Inicio de sesión del enlace recibido

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".

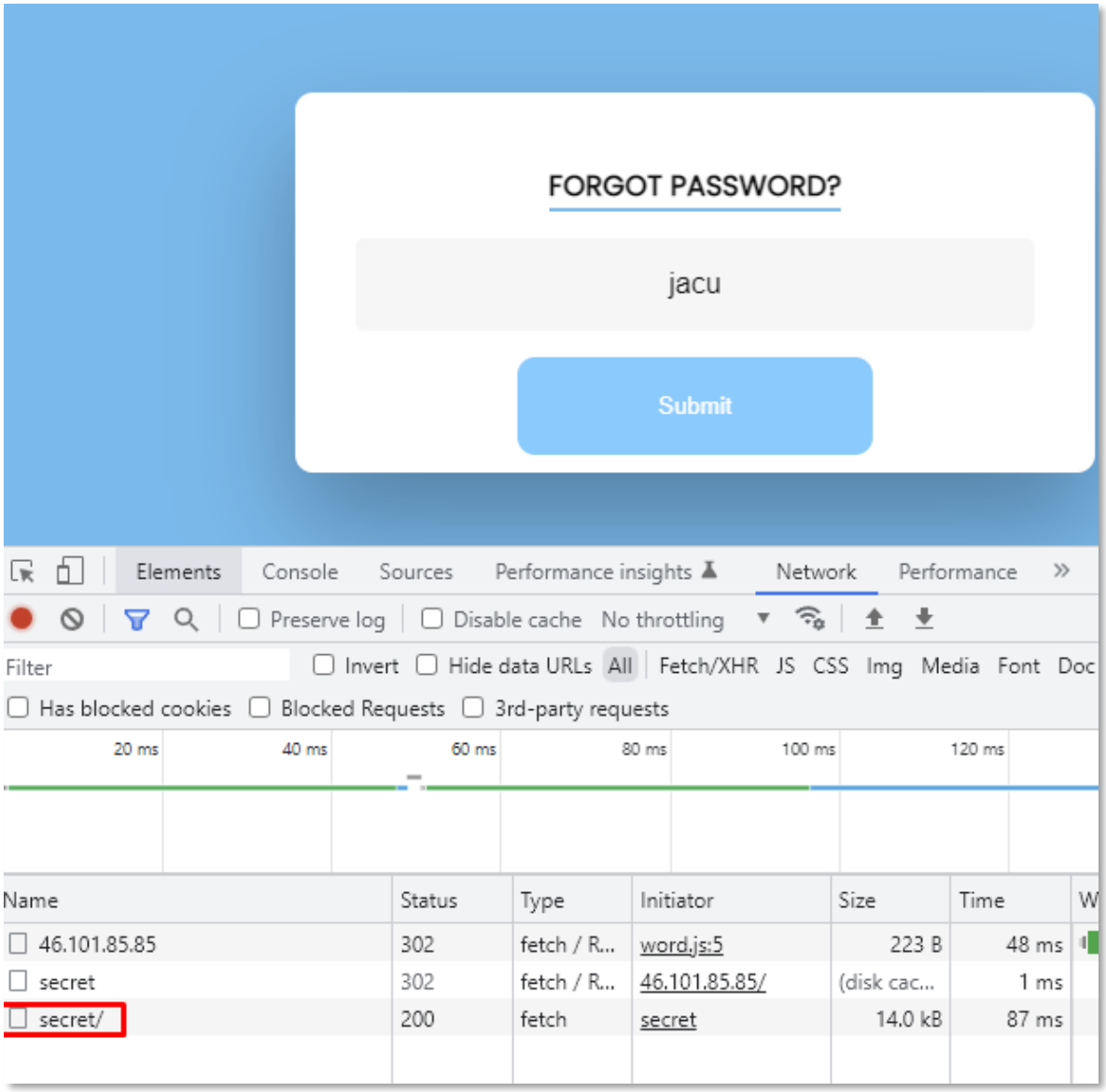


Ilustración 21: Vista de las herramientas de desarrollador con la petición creada.

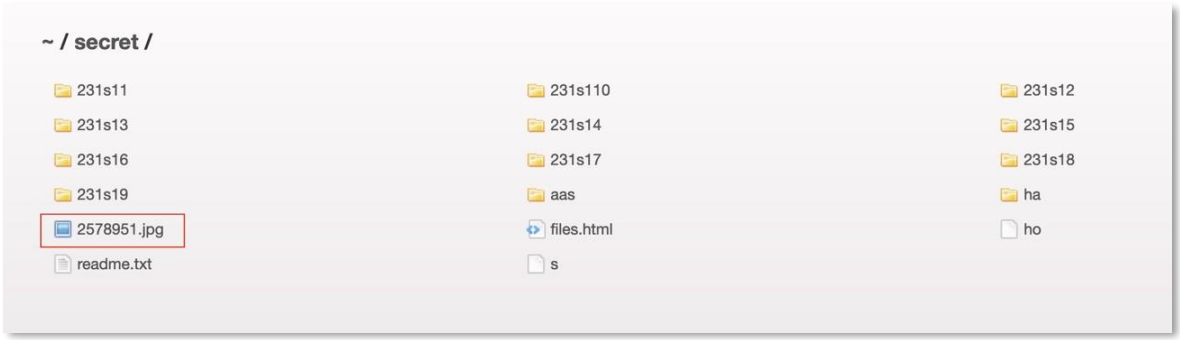


Ilustración 22: Vista de los archivos.

Ilustración 23: Imagen del servidor.

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

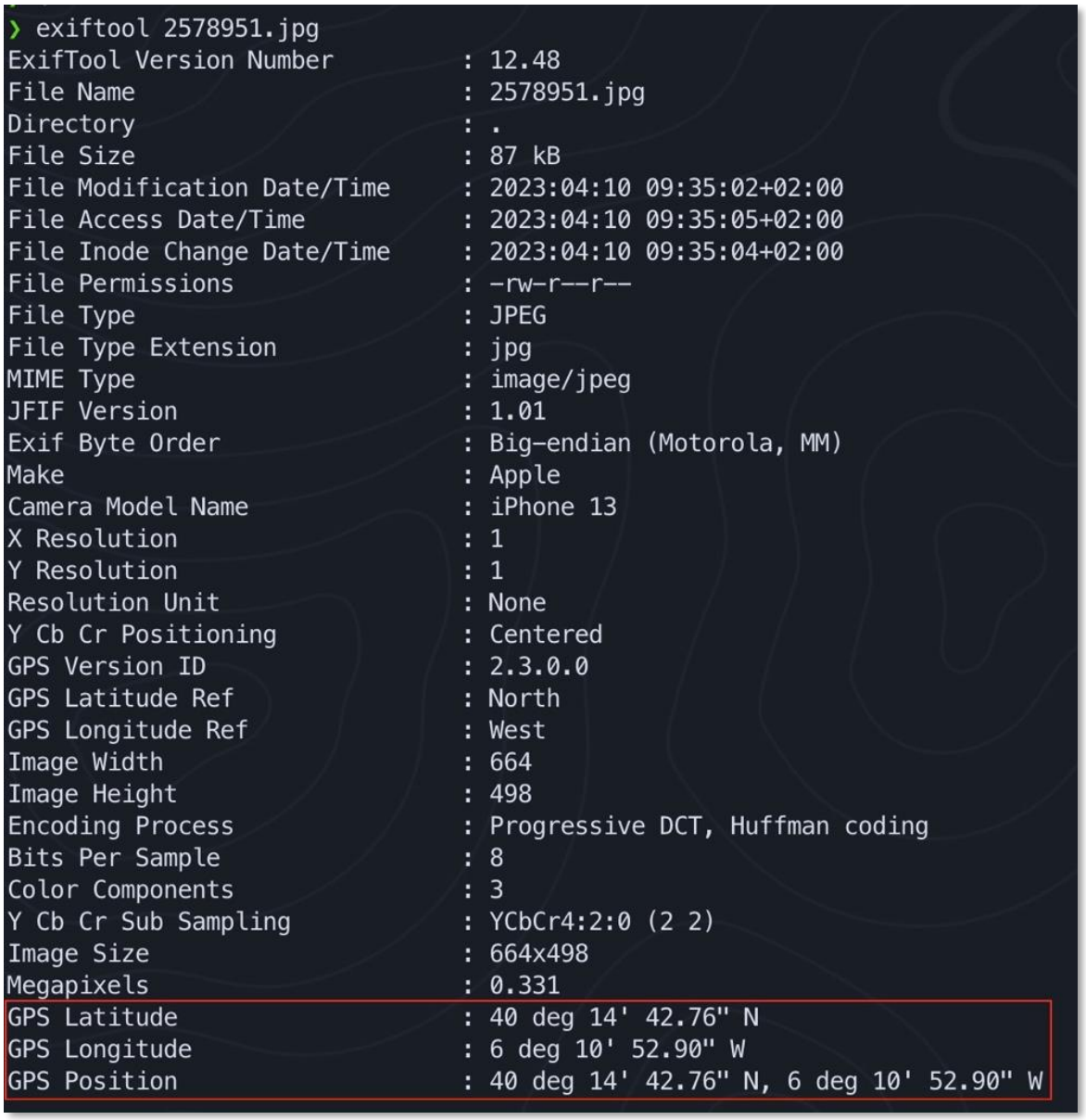


Ilustración 24: Coordenadas GPS en metadatos.

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

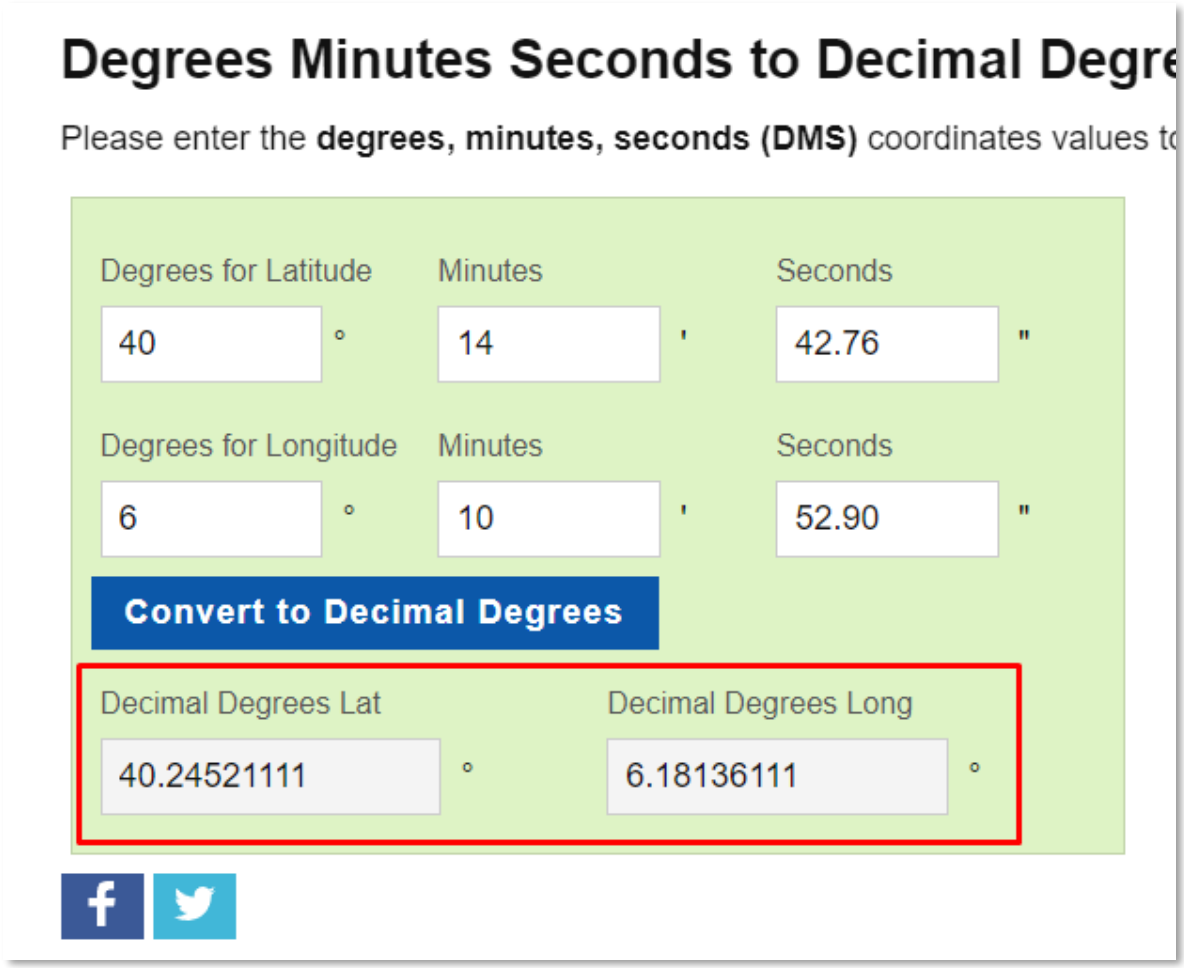


Ilustración 26: Convertidor de coordenadas a dotación decimal de la web gps-coordinates.org

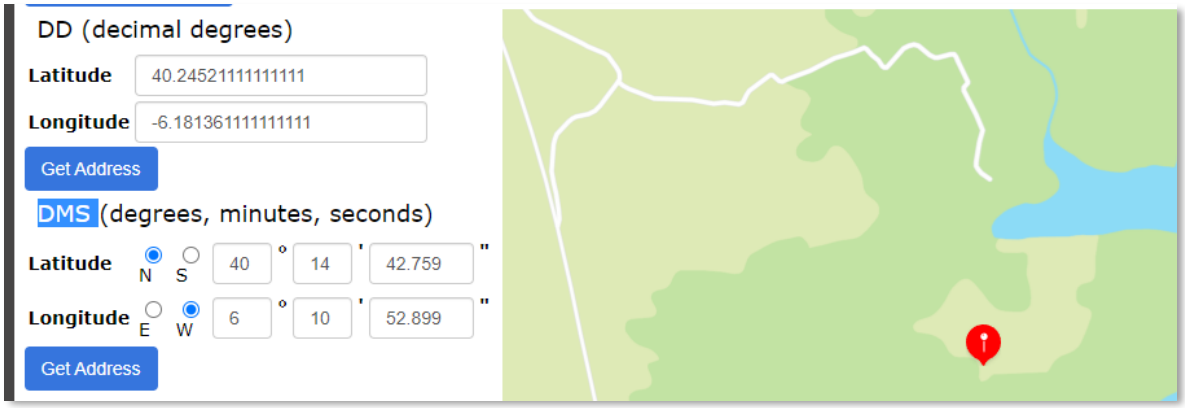


Ilustración 27: Muestra de la localización.

Flag: flag{40.24521, -6.18136}