useit

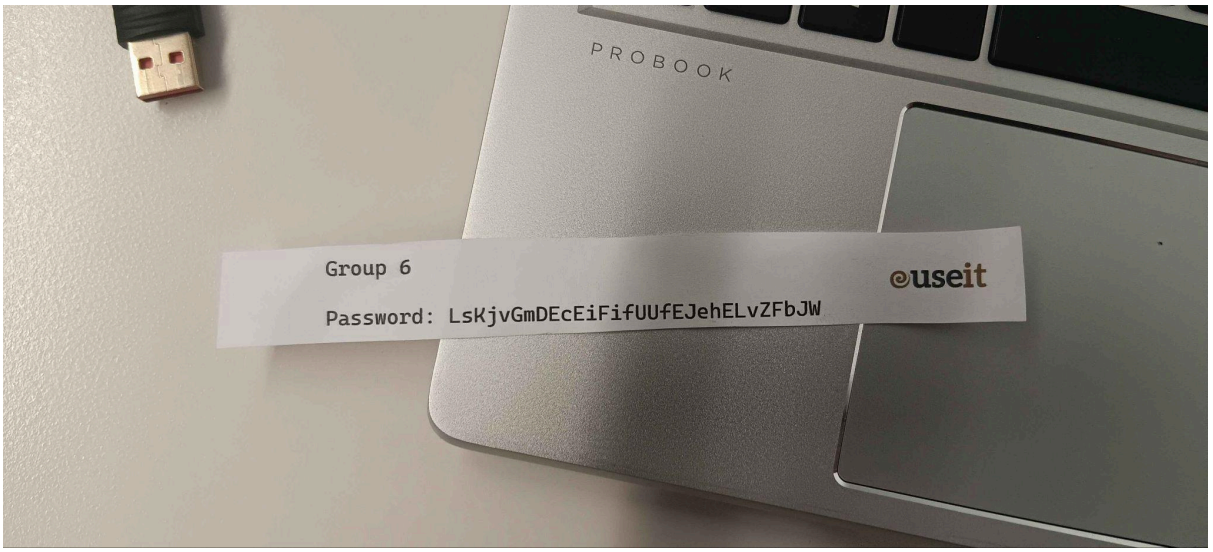
DREAM TEAM

Valentín Martín
David Tomasini
Marc Gàndara
Pol Jaimejuan

HACKATHON 2024: CAPTURE THE FLAG (UseIT)

STEP 0: Introducció

Comencem el repte ingressant a *l'URL*: https://hackaton2024.useitapps.com/



STEP 1: Welcome to the police test.

Iniciem a l'URL: https://hackaton2024.useitapps.com/exam

A l'avançar a la següent pantalla, ens trobem amb la següent imatge, i un missatge indicant que el següent repte és "fàcil" 😉.

Òbviament, el primer pas va ser utilitzar eines d'esteganografia (*la práctica de ocultar información dentro de otro mensaje u objeto físico para evitar su detección*, Kaspersky).

I a l'utilitzar `$ exiftool kennedy.webp ...`

```
password=j9G2FtnXLti6vA6KTwsHL3ttzrFju6NYx8:code_delivered
```

Les metadades contenen la pista. Finalment, cal substituir el fragment *code_delivered* pel codi que ens van donar al començar la CTF i escriure-ho dins d'un element HTML <*input*> que es trobava amagat mitjançant CSS.

Al prémer ENTER, la web ens retorna una resposta en format JSON:


```
{"access_key":"AKIAYS2NWBZUXXXXXXXX","error":0,"msg":"ok","path":"b613e323-540
b-4eb8-a79d-c381e37bd7b6","secret_key":"DJrllrLUSgW7gFKN+XjnV5X3gJtaIV7rXXXXXX
XX"}
```

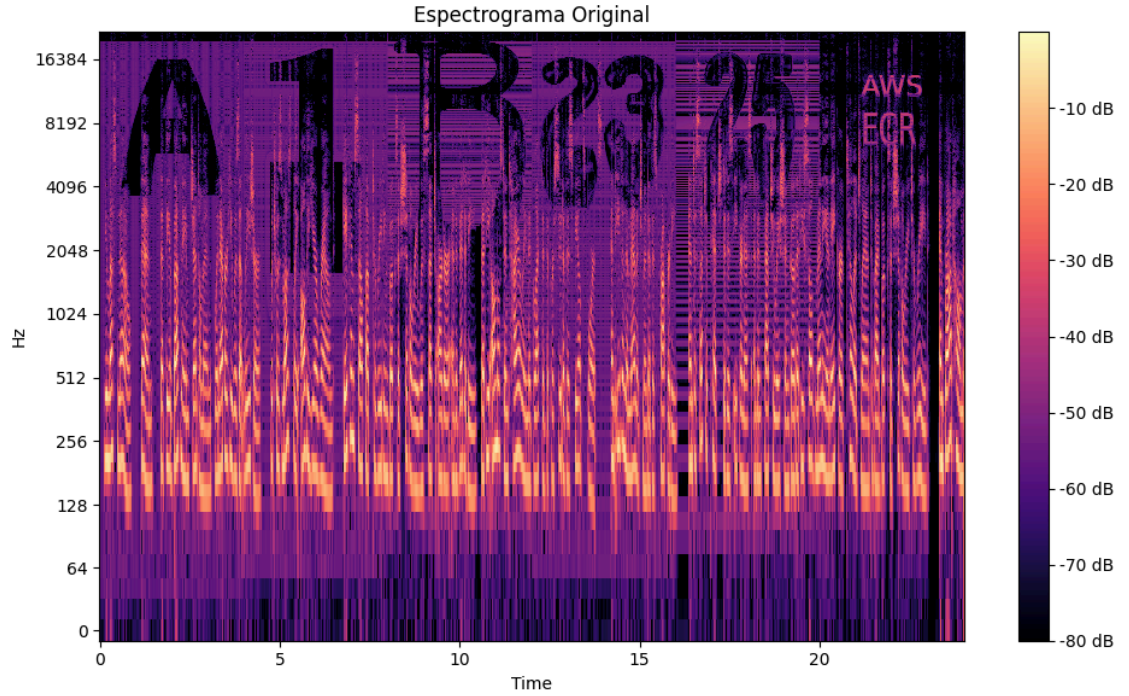

Bingo. Obtenim el següent url:
https://hackaton2024.useitapps.com/b613e323-540b-4eb8-a79d-c381e37bd7b6
Emplenem la URL amb la *path* i avancem al següent nivell.

STEP 2: Welcome to the next step...
A l'entrar a la 2a pantalla, hem obtingut inspeccionant el *HTML* de la pantalla un URL "*audio/audio.wav*". On després d'inspeccionar les metadades i altres dades del fitxer, hem analitzat l'espectrograma del audio mitjançant un *script*.

Per a utilitzar bones pràctiques a l'hora d'implementar el script hem comentat el codi, alhora dels errors (excepcions), hem fet un control d'errors on en comptes d'avortar el programa o llençar l'excepció el que fa es mostrar un missatge d'error informatiu.

```
"""
   Mostra l'espectrograma logarítmic d'un fitxer d'àudio donat.

   Paràmetres:
   -----------
   audio_path : str
       Ruta al fitxer d'àudio que es vol processar.
   """
def mostrar_espectrograma(audio_path):
   try:
       # Carrega l'àudio des del fitxer
       # sr=None manté la freqüència de mostreig original
       y, sr = librosa.load(audio_path, sr=None)

       # Genera l'espectrograma utilitzant la transformada de Fourier
       S = librosa.stft(y)  # Càlcul de la STFT (Short-Time Fourier
Transform)
       S_magnitude = np.abs(S)  # Magnitud de l'espectre complex

       # Configura la figura per a la visualització
       plt.figure(figsize=(10, 6))
       librosa.display.specshow(librosa.amplitude_to_db(S_magnitude,
ref=np.max),
                                sr=sr, y_axis='log', x_axis='time')
       plt.title('Espectrograma (escala logarítmica)')
       plt.colorbar(format='%+2.0f dB')  # Afegim una barra de colors
       plt.tight_layout()  # Ajustem marges per evitar que els elements es
sobreposin
       plt.show()

   except FileNotFoundError:
       print(f"Error: No s'ha trobat el fitxer '{audio_path}'. Comprova la
ruta especificada.")
   except Exception as e:
       print(f"Ha ocorregut un error: {e}")

# Ruta al fitxer d'àudio
audio_path = "audio1.wav"

# Crida a la funció per mostrar l'espectrograma
mostrar_espectrograma(audio_path)
```

Espectrograma resultat:



L'espectrograma analitzat mostra la representació temps-freqüència d'una gravació d'àudio. La visualització permet observar les diferents components freqüencials del senyal, que van des de 0 fins a 16384 Hz, amb una escala d'intensitat en decibels (dB) representada per colors, on el groc indica major intensitat i el violeta/negre menor intensitat.

El senyal sembla correspondre a una seqüència de números pronunciats, amb les característiques típiques de la veu humana concentrades principalment en les freqüències baixes i mitjanes.

La marca "*AWS ECR*" visible al espectrograma suggereix que aquesta gravació forma part d'un sistema que utilitza *Amazon Elastic Container Registry* (ECR), un servei d'AWS que permet emmagatzemar, gestionar i desplegar imatges de contenidors Docker. Això podria indicar que l'àudio està sent processat o emmagatzemat com a part d'una aplicació containeritzada en el núvol d'Amazon.

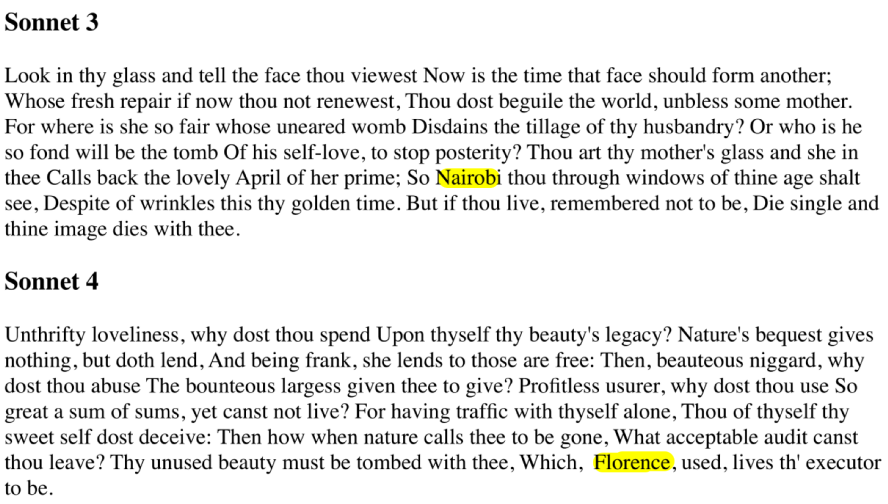
En accedir al contenidor Docker allotjat en aquest registre, hem executat la comanda 'ls' que ens ha revelat l'existència d'uns fitxers *GPG* amb credencials.

Utilitzant aquestes credencials, obtenim el codi i avancem al següent repte:
https://hackaton2024.useitapps.com/761c4303-6381-40dc-b2e9-413ae52b1664

STEP 3: Finally here...

A l'analitzar el codi font *HTML* de la pàgina, vam observar que al prémer una de les paraules destacades es podia obtenir un fitxer anomenat *sonnets.zip*.

Aquest és una agrupació de 10 imatges amb 50 sonets de Shakespeare, als quals se'ls ha afegit de forma artificial el nom d'una ciutat.



Sonnet 3

Look in thy glass and tell the face thou viewest Now is the time that face should form another;
Whose fresh repair if now thou not renewest, Thou dost beguile the world, unbless some mother.
For where is she so fair whose uneared womb Disdains the tillage of thy husbandry? Or who is he
so fond will be the tomb Of his self-love, to stop posterity? Thou art thy mother's glass and she in
thee Calls back the lovely April of her prime; So Nairobi thou through windows of thine age shalt
see, Despite of wrinkles this thy golden time. But if thou live, remembered not to be, Die single and
thine image dies with thee.

Sonnet 4

Unthrifty loveliness, why dost thou spend Upon thyself thy beauty's legacy? Nature's bequest gives
nothing, but doth lend, And being frank, she lends to those are free: Then, beauteous niggard, why
dost thou abuse The bounteous largess given thee to give? Profitless usurer, why dost thou use So
great a sum of sums, yet canst not live? For having traffic with thyself alone, Thou of thyself thy
sweet self dost deceive: Then how when nature calls thee to be gone, What acceptable audit canst
thou leave? Thy unused beauty must be tombed with thee, Which, Florence, used, lives th' executor
to be.

Després d'agafar la primera lletra de cadascun i aconseguint una paraula de 50 caràcters, obtenim el codi del següent pas.

STEP 4: Close close close.
Arribem al pas 4... Definitivament, el repte al que més temps vam dedicar. Després de llegir el text ens sorgeixen més preguntes que respostes: un quadrat? criptografia Caesar? AWS Kinesis? Ford Ranger? Com es relaciona tot això? 😭

Després d'investigar les comandes relacionades amb AWS Kinesis, arribem a les següents operacions del CLI:

```
$ aws kinesis list-streams
```

Descobrim que existeix un stream de dades *info-stream*...

```
$ aws kinesis describe-stream --stream-name info-stream
```

El stream conté una *shard*...

```
$ aws kinesis get-shard-iterator --stream-name info-stream --shard-id <id>
```

Obtenim un *token* de iteració...

```
$ aws kinesis get-records --shard-iterator $SHARD_ITERATOR
```

Bam 💥. Obtenim un JSON que sembla contenir dades codificades en base64.

```
{
    "SequenceNumber": "49657618050361580897192306754760520869691641204404387842",
    "ApproximateArrivalTimestamp": "2024-11-22T15:20:05.254000+01:00",
    "Data":
"eyJ4IjogLTE2LjMxMTczMjQ3NjE2NjU3LCAieSI6IC0xNjguNTQ4NDQ2MjQ2NDU1MTQsICJ0eXBlIjogInkiL
CAiZ2xvYmFsIjogInIifQ==",
    "PartitionKey": "lck"
}
```

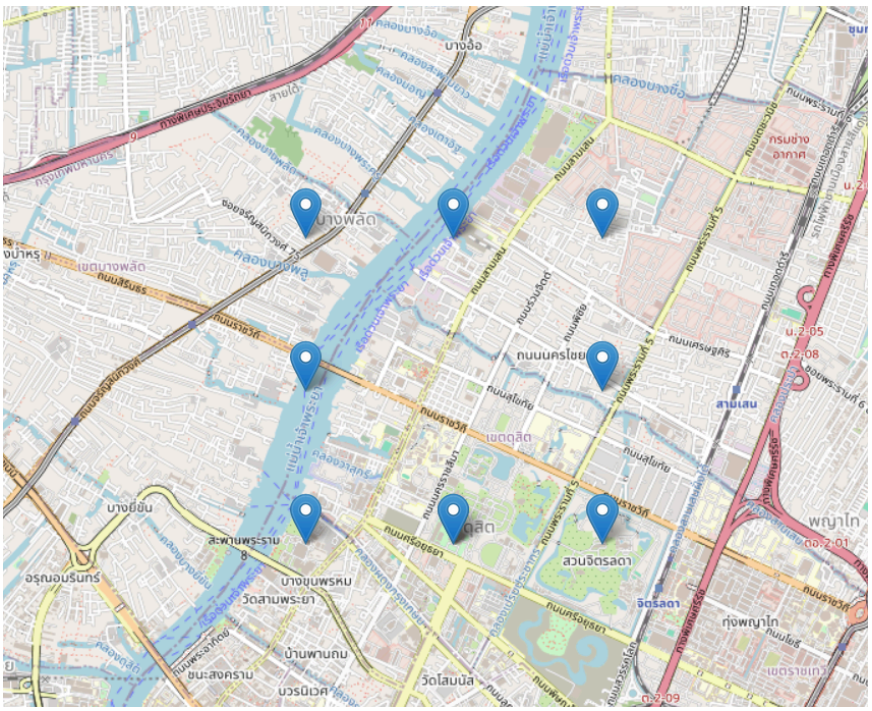

Al decodificar tots els *records*, obtenim...


```
{"x": -26.29192002113991, "y": 4.15972960124392, "type": "d", "global": "x"}
{"x": 62.68342165774419, "y": -104.36418122472904, "type": "v", "global": "b"}
{"x": -48.32459665980522, "y": -4.98544776644951, "type": "v", "global": "b"}
...
```


What. Coordenades?

Després de representar-les en un mapa, podem visualitzar un patró:

Fent zoom en un punt amb alta densitat d'ubicacions...



Aquí està el nostre quadrat.

Després de calcular el seu centre i visualitzar-lo amb *Google Streetview*, podem obtenir la següent imatge:

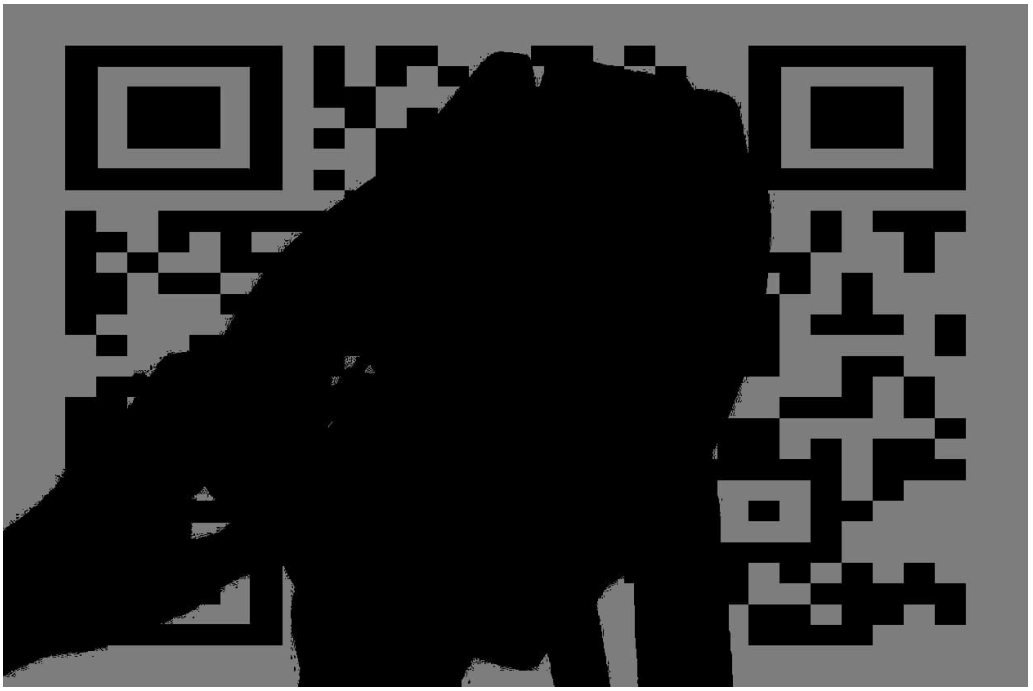
Després de molts intents i desesperació, vam descobrir que la informació “amagada” en aquesta imatge és la marca-model del cotxe gris i el número de casa. És a dir: Honda-Civic i 235.

Apliquem criptografía Caesar al string “Honda-Civic” amb 235 iteracions i obtenim el codi per al següent nivell, on l’URL trobat és: https://hackaton2024.useitapps.com/Ipoeb-Djwjd

STEP 5: You are really close...

Ens anem apropant. Trobem una imatge que mostra un estenògraf.

Al augmentar el contrast i reduir la brillantor, es pot observar un secret...



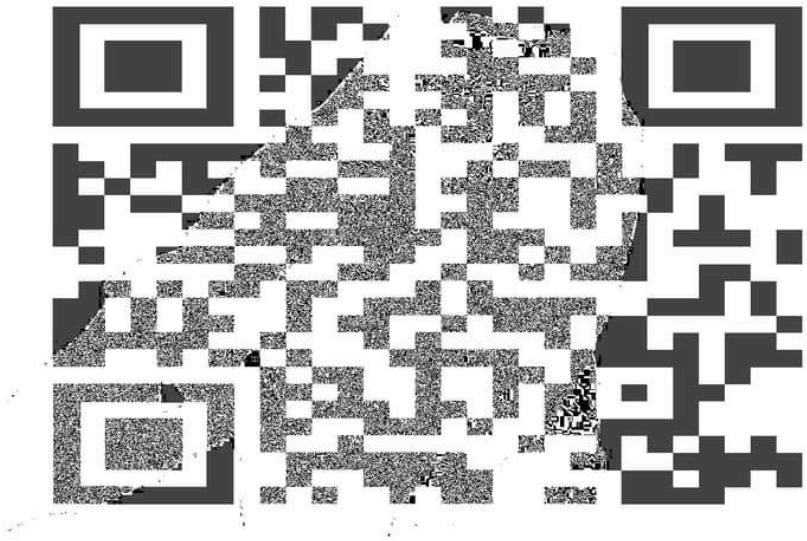
Wow.

Després de l'ajuda d'uns mentors, vam pensar en realitzar una substracció de la imatge original amb la modificada amb la finalitat de revelar el codi QR complet.

El *script* de Python utilitzat va ser:

```
from PIL import Image
import numpy as np

img1 = Image.open('original.png')
grayscale_img = img1.convert('L')
grayscale_img.save('grayscale.png')

img2 = Image.open('stenographer-.webp')
img2 = img2.convert('L')

grayscale_array = np.array(grayscale_img)
reto_array = np.array(img2)

if grayscale_array.shape != reto_array.shape:
    reto_array = np.array(img2.resize(grayscale_img.size))

diff_array = np.absolute(reto_array - grayscale_array)
diff_image = Image.fromarray(diff_array)
diff_image.save('difference.png')
```

després d'una petita inversió de colors el resultat és:



Ja casi ho tenim.

Al escanejar el codi, revelem un *string* misteriós:

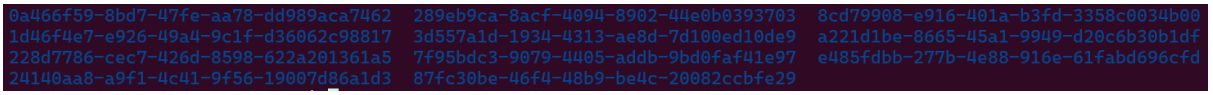
arn:aws:s3:::accompliancenames

Naturalment, el següent pas va ser explorar el bucket de AWS S3 corresponent...

```
$ aws s3 ls s3://accompliancenames
```

I ho copiem a l'ordinador local...

```
$ aws s3 cp s3://accompliancenames/directories/ ./local/ --recursive
```



Al examinar aquestes carpetes, veiem que totes contenen fitxers *.zip* buits, exceptuant d'un...

Al descomprimir-lo, veiem un llistat de carpetes amb noms de persones. Així mateix, aquestes contenen fitxers de text amb identificadors.

Tots els noms segueixen el mateix patró, excepte un: Ibra Hernández. Qui és? 😨
Trobem el fitxer *step.txt* que conté el codi al següent pas.
L'URL obtingut és:
https://hackaton2024.useitapps.com/f081ced9-2c7b-4505-973a-630979eb8100

STEP 6: Hahahaha

Després d'analitzar el *HTML* localitzem un fitxer anomenat *data.csv*, que conté noms de persones, nombre de fills, nombre de ciutats visitades en l'últim any...

Entre tots aquests individus, observem que hi ha una fila que té tots els valors inicialitzats a 0 i té un nom peculiar... la clau per al següent repte.

gg ez 😎

L'URL obtingut és:
https://hackaton2024.useitapps.com/8a4398cf-896b-4d72-999c-4db47ad73400

2136	8a4398cf-896b-4d72-999c-4db47ad73400	0	0	8a4398cf-896b-4d72-999c-4db47ad73400	0

STEP 7: End...
Aquest pas no s'ha pogut completar tot i que s'ha intentat amb múltiples perspectives sense cap resultat suficient, s'ha intentat:

- Persones amb 2 fills i que estan en el CSV i en el directori (vam intentar filtrar però cada vegada que no trobàvem cap URL desgeneralitzavem)
- Persones amb noms incorrectes
- Persones amb una carpeta de mida diferent
- Persones amb una quantitat de documents diferent de 10
- El valor que es troba en el pas 3 i totes les combinacions de xifrat cèsar en els documents de text de cada usuari
- El valor trobat en el pas anterior i totes les combinacions de xifrat cèsar en els documents de text de cada usuari

Per desgràcia, la notícia de què existia una part 2 de la part final va ser devastador per la nostra moral, i després de la decepció del 4t pas que vam estar encallats per més de 6 hores perquè no vam posar l'H i l'C de Honda-Civic en majúscula per encriptar-lo amb 235 vam considerar el repte per acabat per nosaltres.