

# Practica 1 - Let's Go Cryptohacking

## Equipo Caifanes

### Integrantes

- Díaz Tinoco Gisel Maite (317020326)
- Vázquez González Melissa (317209468)
- Hernández Rojas Saúl Alejandro (315304433)
- Palma López Rossana (312047890)

### Introducción

La práctica se centró en la introducción a CryptoHack, una plataforma de entrenamiento en criptografía. Durante esta práctica, se abordaron varios conceptos fundamentales, incluyendo la codificación y decodificación de datos en diferentes formatos como ASCII, hexadecimal y Base64. Además, se exploró en detalle la operación XOR, que es fundamental en la criptografía simétrica.

### Desarrollo

Dentro del equipo, cada quien realizó su curso por separado y a continuación presentamos las capturas de algunos de los ejercicios que se realizaron dentro de él con breves explicaciones sobre cómo se hicieron dentro del código anexado.

### Ejercicio 6:

CRYPTOHACK

5

110

COURSES

CHALLENGES

SCOREBOARD

BLOG

CHAT

CAREERS

FAQ

MELIVG

LOGOUT

★ Bytes and Big Integers

10 pts • 27685 Solves

Cryptosystems like RSA works on numbers, but messages are made up of characters. How should we convert our messages into numbers so that mathematical operations can be applied?

The most common way is to take the ordinal bytes of the message, convert them into hexadecimal, and concatenate. This can be interpreted as a base-16/hexadecimal number, and also represented in base-10/decimal.

To illustrate:

```

message: HELLO
ascii bytes: [72, 69, 76, 76, 79]
hex bytes: [0x48, 0x45, 0x4c, 0x4c, 0x4f]
base-16: 0x48454c4c4f
base-10: 310400273487

```

Python's PyCryptodome library implements this with the methods `bytes_to_long()` and `long_to_bytes()`. You will first have to install PyCryptodome and import it with `from Crypto.Util.number import *`. For more details check the [FAQ](#).

Convert the following integer back into a message:

```
11515195063862318899931685488813747395775516287289682636499965282714637259206269
```

You have solved this challenge!

```
from Crypto.Util.number import *
intMessage = 11515195063862318899931685488813747395775516287289682636499965282714637259206269
print(long_to_bytes(intMessage))
```

**FLAG :** `crypto{3nc0d1n6_4l1_7h3_w4y_d0wn}`

## Ejercicio 7 (Rossana Palma López):

CRYPTOHACK

3

★ 35

COURSES

CHALLENGES

SCOREBOARD

BLOG

CHAT

CAREERS

FAQ

ROSSANAPL

LOGOUT

← Prev

INTRODUCTION TO CRYPTOHACK

Next →

☆ XOR Starter

10 pts · 23491 Solves

XOR is a bitwise operator which returns 0 if the bits are the same, and 1 otherwise. In textbooks the XOR operator is denoted by  $\oplus$ , but in most challenges and programming languages you will see the caret `^` used instead.

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

For longer binary numbers we XOR bit by bit:  $0110 \wedge 1010 = 1100$ . We can XOR integers by first converting the integer from decimal to binary. We can XOR strings by first converting each character to the integer representing the Unicode character.

Given the string `label`, XOR each character with the integer `13`. Convert these integers back to a string and submit the flag as `crypto(new_string)`.

💡

The Python `pwntools` library has a convenient `xor()` function that can XOR together data of different types and lengths. But first, you may want to implement your own function to solve this.

Enter flag here: crypto(FLAGS)

crypto{aloha}

SUBMIT

← Prev

Next →

CRYPTOHACK

Light Mode

FAQ

Blog

COURSES

Introduction to CryptoHack

Modular Arithmetic

Symmetric Cryptography

Public-Key Cryptography

Elliptic Curves

CATEGORIES

General

Mathematics

Symmetric Ciphers

RSA

Diffie-Hellman

Elliptic Curves

Hash Functions

Crypto on the Web

Miscellaneous

Post-Quantum

CTF Archive

CREDITS

Illustrations from Ouch.pics

Drawn by Thierry Fousse and Tatyana Krasutskaya

Privacy Policy

© 2024 CryptoHack

Flag resultante Ejercicio 7

CRYPTOHACK

⚡ 4

★ 45

COURSES

CHALLENGES

SCOREBOARD

BLOG

CHAT

CAREERS

FAQ

ROSSANAPL

LOGOUT

← Prev

INTRODUCTION TO CRYPTOHACK

Next →

★ XOR Starter

10 pts · 23497 Solves

LEVEL UP

You are now level 4

Output

0

1

1

0

1

1

0

XOR is a bitwise operator which is used to compare two bits. If the bits are the same, the result is 0, otherwise, the result is 1. In textbooks the XOR operator is denoted by  $\oplus$ , but in most challenges and programming languages the `^` operator is used instead.

For longer binary numbers we XOR bit by bit:  $0110 \oplus 1010 = 1100$ . We can XOR integers by first converting the integer from decimal to binary. We can XOR strings by first converting each character to the integer representing the Unicode character.

Given the string `label`, XOR each character with the integer `13`. Convert these integers back to a string and submit the flag as `crypto{new_string}`.

The Python `pwntools` library has a convenient `xor()` function that can XOR together data of different types and lengths. But first, you may want to implement your own function to solve this.

You have solved this challenge!

← Prev

Next →

CRYPTOHACK

Light Mode

FAQ

Blog

COURSES

Introduction to CryptoHack

Modular Arithmetic

Symmetric Cryptography

Public-Key Cryptography

Elliptic Curves

CATEGORIES

General

Mathematics

Symmetric Ciphers

RSA

Diffie-Hellman

Elliptic Curves

Hash Functions

Crypto on the Web

Miscellaneous

Post-Quantum

CTF Archive

CREDITS

Illustrations from Ouch.pics

Drawn by Thierry Fousse and Tatyana Krasutskaya

Privacy Policy

© 2024 CryptoHack

Captura Ejercicio 7 exitoso

FLAG: `crypto{aloha}`

Recurso empleado

```

str = "label"
#Define la funcion xor usando el operador ^
def XOR(A, B):
    return '{0:b}'.format(int(A,2) ^ int(B,2))
#Guardamos en un arreglo los codigos ascii de 'label'
ascii_array = [ord(c) for c in str]
#Cada codigo en ascii_array es transformado a su forma binaria
binary_array = [bin(c)[2:] for c in ascii_array]
#Creamos un nuevo arreglo donde se opera la representación binaria
#con la representación binaria de 13
new_binary_array = [XOR(a,bin(13)[2:]) for a in binary_array]
#Obtenemos la representación decimal de cada elemento en new_binary_array
result_ascii = [int(x, 2) for x in new_binary_array]
#Guardamos el nuevo string resultante de xor 'label' con 13
result_string = "".join(chr(i) for i in result_ascii)
#Impresiones de verificación
print(binary_array)
print(new_binary_array)
print(result_ascii)
#Impresión del nuevo string
print(result_string)

```

## Ejercicio 8:

CRYPTOHACK

4

★ 45

COURSES

CHALLENGES

SCOREBOARD

BLOG

CHAT

CAREERS

FAQ

DIGITAL

LOGOUT

Associative:  $A \oplus (B \oplus C) = (A \oplus B) \oplus C$

Identity:  $A \oplus 0 = A$

Self-Inverse:  $A \oplus A = 0$

Let's break this down. Commutative means that the order of the XOR operations is not important. Associative means that a chain of operations can be carried out without order (we do not need to worry about brackets). The identity is 0, so XOR with 0 "does nothing", and lastly something XOR'd with itself returns zero.

Let's put this into practice! Below is a series of outputs where three random keys have been XOR'd together and with the flag. Use the above properties to undo the encryption in the final line to obtain the flag.

KEY1 = a6c8b6733c9b22de7bc0253266a3867df55acde8635e19c73313

KEY2 ^ KEY1 = 37dcb292030faa90d07eec17e3b1c6d8daf94c35d4c9191a5e1e

KEY2 ^ KEY3 = c1545756687e7573db23aa1c3452a098b71a7fbf0fddde5fc1

FLAG ^ KEY1 ^ KEY2 ^ KEY3 = 04ee9855208a2cd59091d04767ae47963170d1660df7f56f5faf

Before you XOR these objects, be sure to decode from hex to bytes.

Enter flag here: crypto{FLAG}

crypto{x0r\_i5\_ass0c1at1v3}

SUBMIT

FLAG: crypto{x0r\_i5\_ass0c1at1v3}

Recurso empleado:

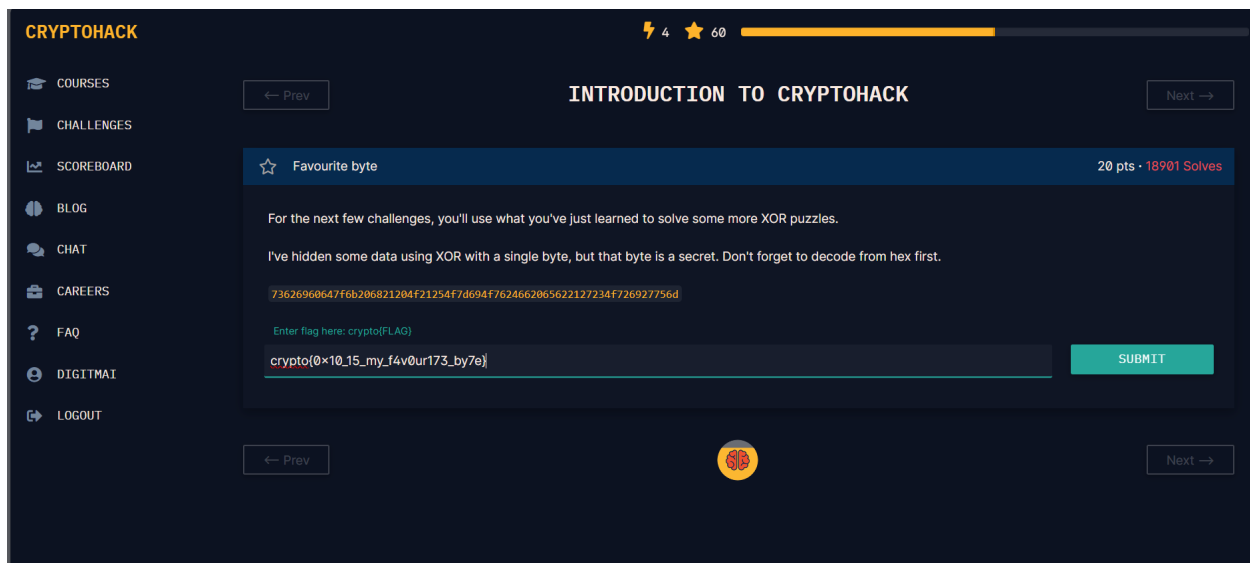
```
from pwn import xor
```

```
k1 = bytes.fromhex("a6c8b6733c9b22de7bc0253266a3867df55acde8635e19c73313")
k21 = bytes.fromhex("37dcb292030faa90d07eec17e3b1c6d8daf94c35d4c9191a5e1e")
k23 = bytes.fromhex("c1545756687e7573db23aa1c3452a098b71a7fbf0fddde5fc1")
all = bytes.fromhex("04ee9855208a2cd59091d04767ae47963170d1660df7f56f5faf")
# Por la propiedad de inverso obtenemos k2 a partir de k21 que
k2 = xor(k21, k1)
k3 = xor(k23, k2) # Ya teniendo el valor de k2, de igual manera
# Teniendo el valor de cada key independiente, podemos obtener el
# Notamos que no debe de preocuparnos el orden en el que se lleva
# Ya que XOR es asociativa
flag = xor(xor(all, k2), xor(k3, k1))
print(flag.decode())
```

## Ejercicio 9:

Practica 1 - Let's Go Cryptohacking

6

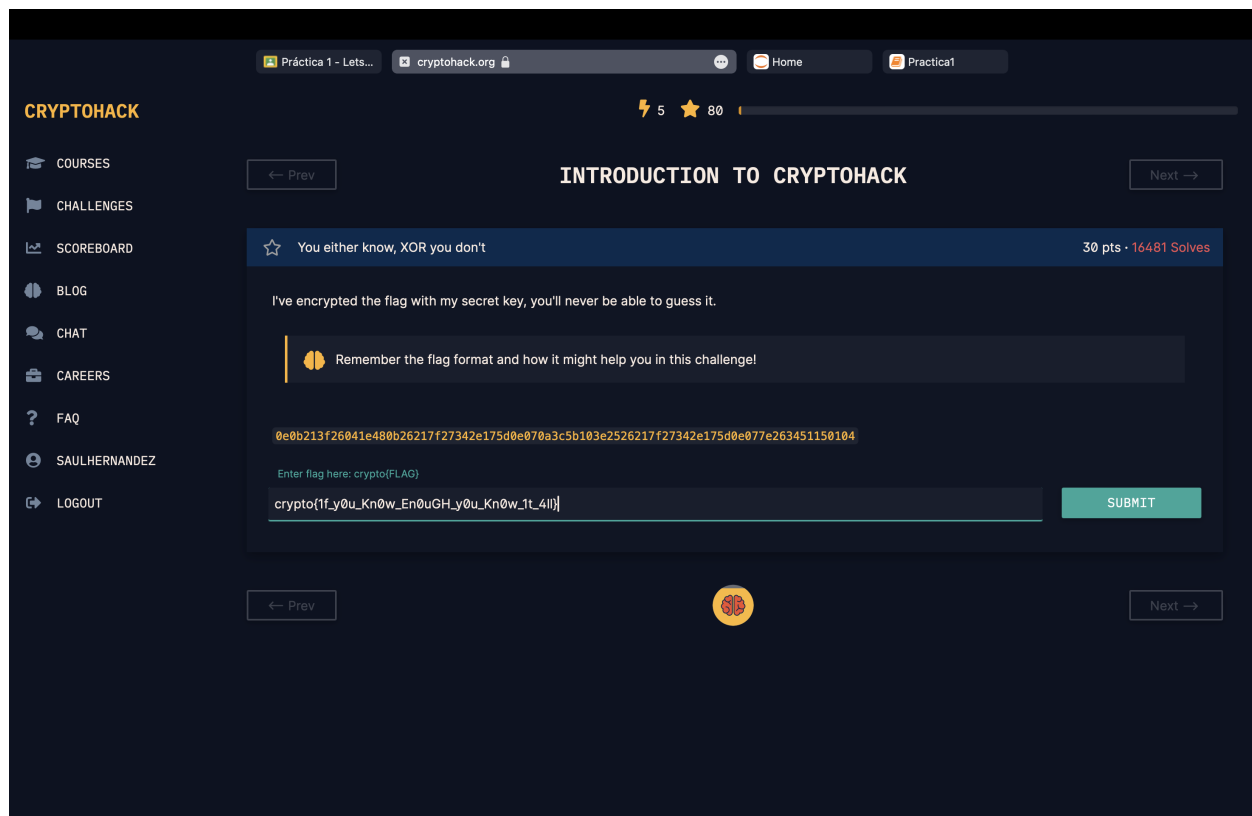


FLAG: crypto{0x10\_15\_my\_f4v0ur173\_by7e}

Recurso empleado:

```
dat = bytearray.fromhex("73626960647f6b206821204f21254f7d694f7624662065622127234f726927756d")
# Como no sabemos que byte se usó, intentamos todos por fuerza bruta
for num in range(256):
    par = [chr(p^num) for p in dat]
    flag = "".join(par)
    # Como sabemos el formato de las banderas, sabemos cuando empieza
    if flag.startswith("crypto"):
        print(flag)
        break
```

## Ejercicio 10



crypto{1f\_y0u\_Kn0w\_En0uGH\_y0u\_Kn0w\_1t\_4ll}

Recurso empleado:

```
from pwn import *
# Se tomaron los bytes de el hex proporcionado
input = bytes.fromhex("0e0b213f26041e480b26217f27342e175d0e070a3c5b103e2526217f27342e175d0e077e263451150104")
#primero, se hizo un xor entre los bytes y la estructura generada
print(xor(input, "flag{}"))
# Con esto obtuvimos como resultado una cadena que contenía la clave
#la cual tomamos como llave para finalmente obtener la bandera
print (xor (input, "myXORkey") )
```

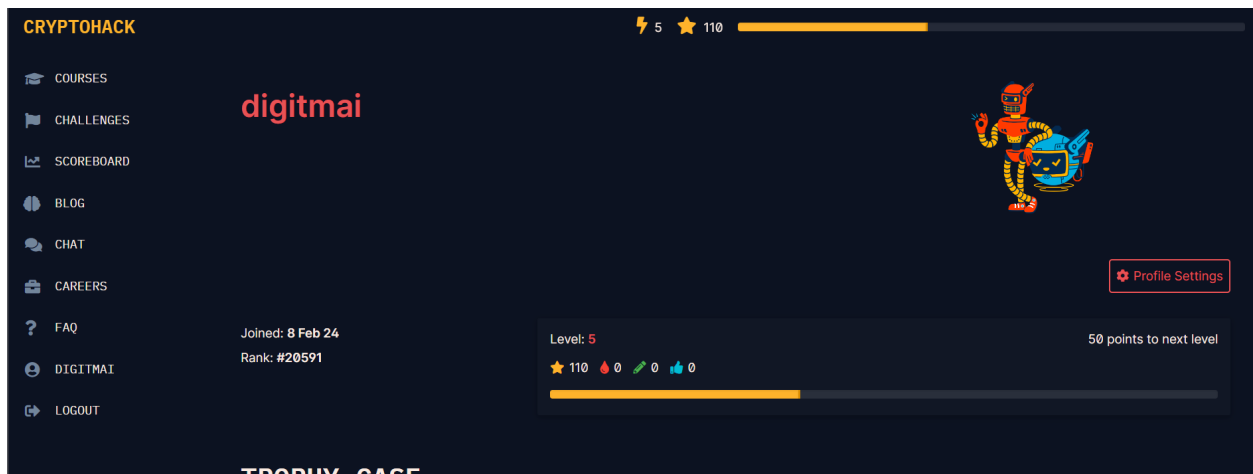
## Curso Completado:

Nombre: Gisel Maite Díaz Tinoco

No. de cuenta: 317020326

URL: [CryptoHack – digitmai](#)

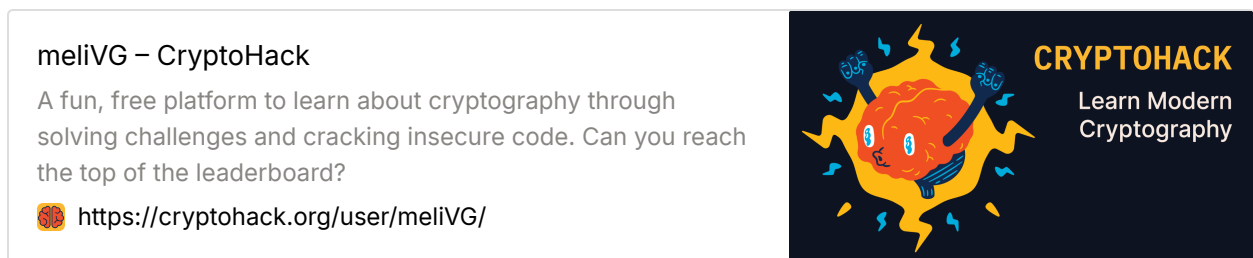




Nombre: Melissa Vázquez González

No. de cuenta: 317209468

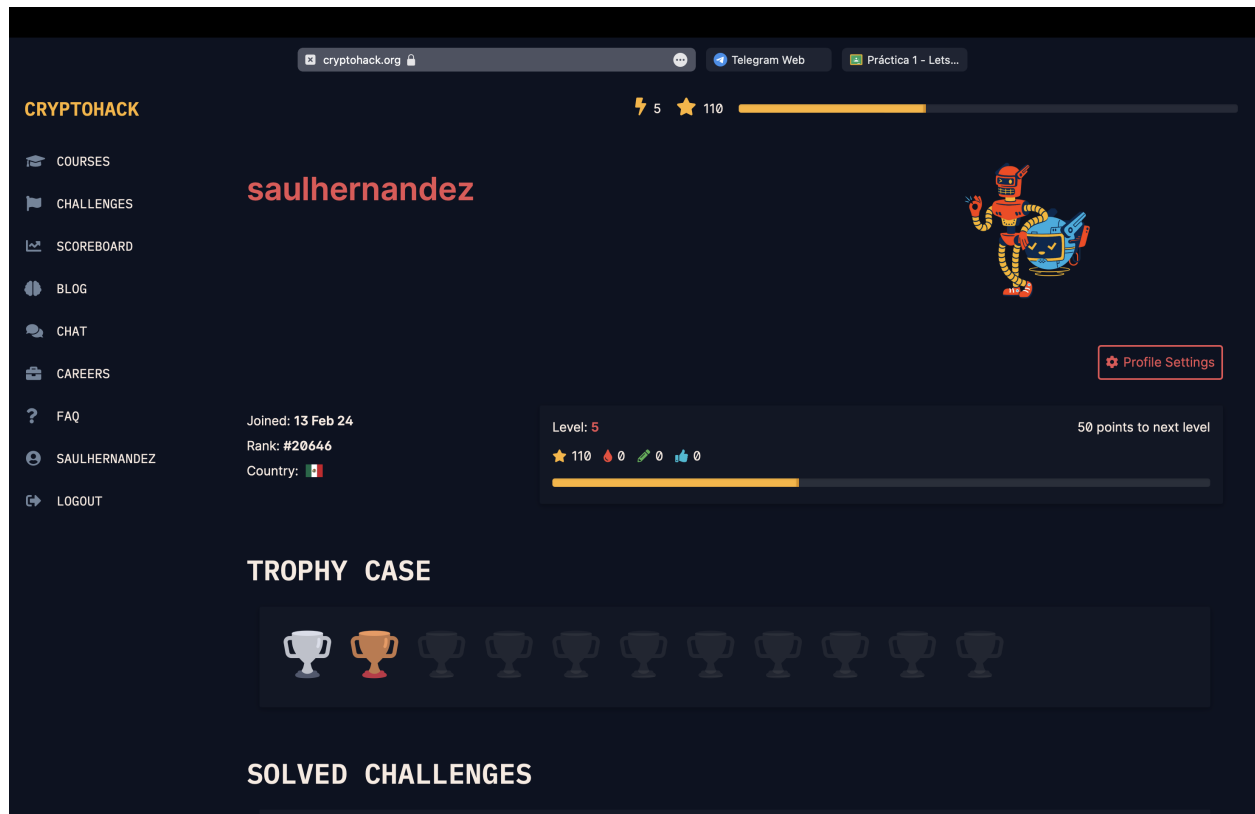
URL:



Nombre: Hernández Rojas Saúl Alejandro

No. de cuenta: 315304433

URL: <https://cryptohack.org/user/saulhernandez/>



Nombre: Rossana Palma López

Nº Cuenta: 312047890

URL: <https://cryptohack.org/user/RossanaPL/>

CRYPTOHACK

5

110

COURSES

CHALLENGES

SCOREBOARD

BLOG

CHAT

CAREERS

FAQ

ROSSANA PL

LOGOUT

RossanaPL

Profile Settings

Joined: 10 Feb 24

Rank: #20663

Level: 5

50 points to next level

110

0

0

0

TROPHY CASE

SOLVED CHALLENGES

13 Feb 24	General	You either know, XOR you don't	★ 30
13 Feb 24	General	Favourite byte	★ 20
13 Feb 24	General	XOR Properties	★ 15
12 Feb 24	General	XOR Starter	★ 10
12 Feb 24	General	Bytes and Big Integers	★ 10
11 Feb 24	General	Base64	★ 10
11 Feb 24	General	Hex	★ 5
11 Feb 24	General	ASCII	★ 5
10 Feb 24	Introduction	Great Snakes	★ 3
10 Feb 24	Introduction	Finding Flags	★ 2

CRYPTOHACK

Light Mode

FAQ

Blog

COURSES

Introduction to CryptoHack

Modular Arithmetic

Symmetric Cryptography

Public-Key Cryptography

Elliptic Curves

CATEGORIES

General

Mathematics

Symmetric Ciphers

RSA

Diffie-Hellman

Elliptic Curves

Hash Functions

Crypto on the Web

Miscellaneous

Post-Quantum

CTF Archive

CREDITS

Illustrations from Ouch.pics

Drawn by Thierry Fousse and Tatyana Krasutskaya

Privacy Policy

© 2024 CryptoHack

Practica 1 - Let's Go Cryptohacking

11

## Investigación

- **Malware:** Son las aplicaciones o código que se pueden presentar en distintas formas (un correo electrónico de phishing, un archivo o unidad USB infectada, etc.) y se usan de manera maliciosa para dañar, alterar el dispositivo y muchas veces, obtener acceso a tu información personal.
- **Spam:** Se le dice al tipo de correo que es enviado a usuarios de manera anónima y masiva. El contenido de los correos spam puede variar desde publicidad, engaños, estafas, spam con malware.

## Conclusiones

Creemos que el curso fue una manera interactiva de acercarnos a estos conceptos básicos que nos van a servir más adelante, además, de repasar o volver a empaparnos en la sintaxis de Python, que como también sabemos, vamos a tener que seguir usando a lo largo del curso.

En cuanto a las dificultades, fue difícil lograr instalar las herramientas necesarias de manera correcta en las computadoras de todas las personas en el equipo. También fue un poco complicado manejar todos los tipos de datos involucrados al momento de operarlos ya que en Python no existen funciones por defecto para operar hexadecimales con binarios.

## Punto Extra: 🤖

### Preguntas Ep: 000 - *Operation Aurora*

#### 1. ¿Qué sucedió el 14 de Diciembre de 2009 en Google?

El 14 de diciembre, a un empleado de Google le llegó un mensaje con un link. Cuando le dio click, se le empezó a descargar malware a su computadora. Así, los atacantes se metieron a la red de Google.

**2. ¿En qué consiste el proyecto Aurora?**

Fue la respuesta que tuvo Google hacia el ataque.

**3. ¿Qué relevancia tiene el nombre "Aurora"?**

Es el nombre del barco de guerra que disparó un misil y dio inicio a la revolución rusa. Así como este barco marcó las décadas siguientes a su disparo, la operación Aurora hizo lo mismo en el mundo de la ciberseguridad.

**4. Describe en tus propias palabras ¿qué es el análisis forense?**

Es analizar el hardware afectado por un malware, específicamente el disco duro.

**5. ¿Qué es un Galimatías?**

Es un algo que no tiene mucho sentido y es difícil de entender.

**6. ¿Cómo entró en acción el equipo de Google para erradicar el ataque?**

Se basó en juntar a expertos en ciberseguridad de todo el mundo en el campus, analizar el hardware de las computadoras infectadas, y finalmente reiniciar la información de acceso de todos las las personas empleadas en la empresa. Logrando así eliminar al atacante de su red.

**7. ¿De dónde provenía el ataque?**

De China, se tomó como un ataque del gobierno Chino hacia Estado Unidos.

**Preguntas Ep: 002 - Threat Analysis Group**

**1. ¿Internet está lleno de personas que intentan hacer cosas malas? ¿Cuál es el deber por parte del equipo de Google?**

Toni Giswani, manager de ingeniería en seguridad, dijo que sí. Su trabajo es entender quienes son los atacantes y cómo operan para prevenir ataques.

**2. Menciona algunas amenazas monitoreadas por Google**

vigilan amenazas al gobierno, amenazas financieras como ransomware y amenazas de desinformación, donde grupos pretenden esparcir datos falsos.

**3. ¿Qué significa TAG? ¿Qué tipo de contenido bloquea?**

Significa Threat Analysis Group. Bloquea ransomware, mensajes de phishing.

**4. ¿Qué es WannaCry? Explica brevemente lo que hace y de donde proviene**

Es un ataque de ransomware, los atacantes secuestran la computadora de la víctima y piden un rescate para liberarla. Esta vez los responsables fueron los norcoreanos.

**5. Con tus propias palabras, ¿qué es el phishing?**

Es un ataque en donde se tiente a al usuario con contenido llamativo que lo convence de entrar a un enlace malicioso que podría robarle su información.

## **Referencias:**

1. ¿Qué es el malware? Definición y tipos | Seguridad de Microsoft
2. ¿Qué es Spam? (uach.mx).
3. ¿Qué es el Spam? | Enciclopedia de Kaspersky.