Universidad Autónoma de Nuevo León
Facultad de Ciencias Físico Matemáticas

PIA

ETAPA 2:

Documentación y Presentación

Equipo 03:

- Eduardo Vanoye Chi.
- Yair González Reyes.
- Alfredo Mendoza Hernández.
- Yolanda del Carmen García Carranza.
- Oliver Mauricio Alvarado García.

Start

Resumen

Este proyecto tiene como objetivo analizar indicadores de amenazas cibernéticas usando datos de la plataforma AlienVault OTX. El sistema permite:

- Autenticación de usuarios mediante correo y contraseña.
- Obtención de indicadores maliciosos desde la API de OTX, o desde archivos .txt locales en formato JSON.
- Almacenamiento y gestión de datos utilizando listas y diccionarios, evitando duplicados.
- Análisis estadístico de los indicadores por tipo y por año de creación, incluyendo cálculo de media, mediana, moda y desviación estándar.
- Visualización de datos mediante gráficas de barras y gráficas circulares usando matplotlib.
- Exportación de resultados estadísticos a archivos Excel con pandas.

El proyecto es útil para detectar tendencias en amenazas de ciberseguridad y visualizar de forma clara la evolución de los indicadores maliciosos con base en los datos obtenidos.

Algoritmo

D.V:

LIMITE_DE_RESULTADOS: Entero.

INDICADOR[]:Lista de tipo caracter.

LIMITE: Entero.

INDICADORES: Entero.

URL: Cadena de caracteres.

RESPONSE: Cadenas de caracteres.

DATOS: Cadena de caracteres.

PIA_M: Variable de tipo caracter.

PIA: Variable de tipo caracter.

OS: Variable de tipo caracter.

VERIFICAR_DOMINIO: Variable de tipo caracter.

VERIFICAR_LINK_GOOGLE: Variable de tipo caracter.

IMPORTAR_DATOS: Verificar variable de tipo caracter.

VERIFICAR_MALWARE: Variable de tipo caracter.

VERIFICAR_DATOS_IOS: Variable de tipo caracter.

1.- Inicio.

2.- Importar (API_KEY)

3.- Leer (LÍMITE_DE_RESULTADOS)

4.- Intentar,

    5.- Obtener (PULSO)

    6.- INDICADORES [:LIMITE]

    7.- Leer (INDICADOR)

    8.- Para INDICADORES desde 0 hasta INDICADOR

        9.- Mostrar "Tipo", "Dominio", "Indicador"

        10.- sino,

            11.- Mostrar "No hay indicadores disponibles para este pulso."

        12.- Fin del condicional.

        13.- INDICADORES +=1

14.- Fin del repetir.

15.- Intentar,

    16.- Mostrar "Hubo un error al obtener indicadores"

    17.- Leer (URL)

    18.- Intentar,

17.- Leer (URL)

18.- Intentar,

19.- Leer (RESPONSE)

20.-. DATOS = RESPONSE.JASON()

21.- Excepto,

22.- Mostrar ("Error HTTP: " ERRH)

23.-Mostrar ("Error de conexión: " ERRC)

24.- Mostrar ("Tiempo de espera agotado: " ERRT)

25.- Mostrar ("Error desconocido en la petición: " ERR)

26.- Mostrar ("Error al decodificar JSON: " ERRJ")

27.- Fin del algoritmo.

28.- Importar PIA_M como PIA.

29.- Importar OS

30.- Definir main()

31.- Intentar,

32.- Leer (MENU)

33.- Si MENU =

33.- 1: entonces,

34.- Llamar (VERIFICAR_DOMINIO)

35.- 2: entonces,

36.- Llamar (VERIFICAR_LINK_GOOGLE)

37.- 3: entonces,

38.- Llamar (IMPORTAR_DATOS)

39.- 4: entonces,

40.- Llamar (VERIFICAR_MALWARE)

41.- 5: entonces,

42.- Llamar (VERIFICAR_DATOS_IOS)

43.- 6: entonces,

44.- Mostrar ("Gracias por usar el sistema")

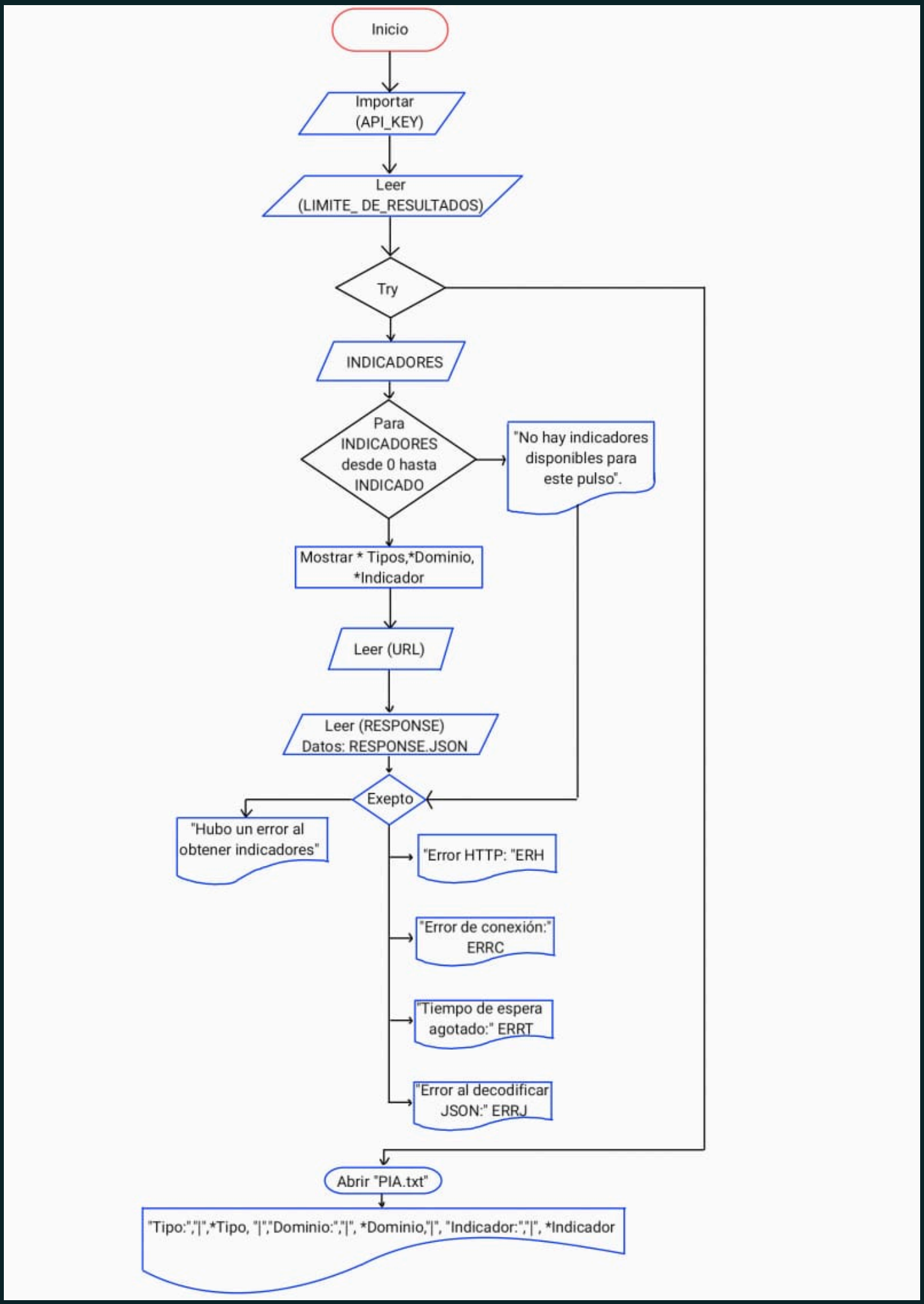
45.- Fin del si múltiple.

46.- Excepto,

47.- Mostrar (Se debe ingresar un número").

48.-Fin.

Diagrama


Inicio
Importar (API_KEY)
Leer (LIMITE_ DE_RESULTADOS)
Try
INDICADORES
Para INDICADORES desde 0 hasta INDICADO
"No hay indicadores disponibles para este pulso".
Mostrar * Tipos,*Dominio, *Indicador
Leer (URL)
Leer (RESPONSE) Datos: RESPONSE.JSON
Exepto
"Hubo un error al obtener indicadores"
"Error HTTP: "ERH
"Error de conexión:" ERRC
"Tiempo de espera agotado:" ERRT
"Error al decodificar JSON:" ERRJ
Abrir "PIA.txt"
"Tipo:","|",*Tipo, "|","Dominio:","|", *Dominio,"|", "Indicador:","|", *Indicador

Cuadro comparativo

	Open Threat Exchange	***API NASA***	***Rest Countries***	***PokéApi***
Función principal	*Consultar y proporcionar datos sobre inteligencia de amenazas cibernéticas*	*Proveer datos científicos, astronómicos y sobre la exploración espacial*	*Proporcionar datos y estadísticas sobre los países del mundo*	*Mostrar datos de los personajes del mundo de Pokémon*
Datos que ofrece	*Información para la ciber seguridad, como: ip maliciosas, dominios, indicadores, etc.*	*Información e imágenes sobre: asteroides, astronomía, fotos del rover de marte, etc.*	*Información de los países, como: idiomas hablados, población, localización, nombre, bandera, etc.*	*Información de los Pokémon, como: habilidades, tipos, movimientos, objetos, generaciones, etc.*
Proveedor	*AlienVault*	*NASA*	*Desarrollada por terceros en base al proyecto de Fayder Flórez*	*Paul Hallett y otros contribuidores*
Método de autenticación	*Requiere el uso de una API Key*	*Requiere el uso de una API Key*	*No requiere un método de autenticación*	*No requiere un método de autenticación*
Formato de respuesta	*JSON*	*JSON*	*JSON*	*JSON*
Costo	*API gratuita con número limitado de usos. Y opción de adquirir planes empresariales*	*API gratuita con número limitado de usos*	*API gratuita*	*API gratuita*

Guion

Carmen: Sean bienvenidos, a esta nuevo episodio de "Código Rojo" El día de hoy hablaremos un poco sobre un tema que ha tomado mucha importancia en la actualidad: la ciberseguridad.

Eduardo: Para comenzar, pues como muchos de nuestros oyentes saben, somos estudiantes de la Universidad Autónoma de Nuevo León , se nos presentó una oportunidad para tratar con esta temática de la ciberseguridad ¿Recuerdan cómo comenzó?

Yair: Sí, no se me olvida que estábamos trabajando en un proyecto en línea, cuando en mitad de la llamada, a nuestro compañero Oliver le llegó un mensaje de uno de nuestros profesores. Ese mensaje tenía una URL y en el chat nuestro "profesor" le especificaba que debía llenar sus datos para el trámite de una beca. Pero nos pareció raro que ese maestro le enviara ese tipo de avisos, pues no era nuestro tutor ni nada por el estilo.

Alfredo: Sí, después nos enteramos que a ese maestro le habían hackeado la cuenta. Era un URL peligroso.

Oliver: Yo considero que fui inteligente por no haberlo abierto en el momento, sin pensar.

Carmen: Exacto, entonces de ahí nos pareció conveniente codificar la idea de protegernos digitalmente, de tal manera de que nos fuera de utilidad en el futuro.

Eduardo: Sí, ahora procederemos a dar una explicación de cómo funciona la lógica detrás de este código.

Carmen: Primeramente, en nuestro algoritmo explicamos cómo es que se recopila la información de la API para ser utilizada a lo largo del código, obviamente, haciendo los arreglos pertinentes a nuestra llave.

Después del acceso, se recopilan los datos, especificando algún límite que nos permita abarcar cierta cantidad de estos mismos, así como el formato que se le dará al código.

Finalmente, se llamarán las funciones que el usuario seleccione en el menú, para que por ejemplo, se verifique el malware, el dominio o se de fin al programa.

Eduardo: Así es, ahora hablemos un poco más de lo que respecta al código.

Yair:
El primer paso es verificar que exista el archivo key.txt. Este archivo contiene una clave de acceso (API Key), que necesitamos para conectarnos a la plataforma AlienVault OTX.
#Mostrar codigo de lectura de txt de Api Key

Con esto, leemos la clave y creamos un objeto que nos permitirá hacer consultas a la API. También aseguramos que la clave esté guardada de forma segura fuera del código.

Oliver:
Una vez conectados, el programa consulta datos desde AlienVault OTX. Por ejemplo, podemos pedir información sobre:

- Direcciones IP maliciosas.
- Dominios peligrosos.
- URLs sospechosas.
-

#Mostrar funcion de como se consigue un indicador

El resultado se guarda para su análisis. Así obtenemos los indicadores que luego procesaremos estadísticamente.

Alfredo:
Después de recolectar los datos, el programa realiza análisis estadístico. Calculamos:

- Media
- Mediana
- Moda
- Desviación estándar
- Varianza

Esto lo hacemos usando módulos como statistics y numpy.

Luego, usamos matplotlib para generar gráficas. Por ejemplo:

- Gráficas de barras: muestran cuántas amenazas hay por tipo.
- Gráficas de líneas: muestran cambios a lo largo del tiempo.
- Gráficas de pastel: muestran la distribución por país.

#Mostrar Alguna funcion que genere la grafica

Eduardo::
En resumen, nuestro programa:

1. Lee una clave desde un archivo seguro.
2. Se conecta a una API de amenazas.
3. Extrae y guarda indicadores peligrosos.
4. Analiza los datos con estadísticas.
5. Y genera gráficas automáticas para visualizar la información.

#Basicamente hacer una pequeña recopilación de todo el programa

Conclusión:

Este proyecto final nos permitió aplicar los conocimientos fundamentales de programación aprendidos durante el curso. A través del uso de Python, desarrollamos un programa funcional que se conecta a una API externa (AlienVault OTX) para obtener datos reales sobre amenazas cibernéticas.
Pusimos en práctica conceptos clave como:

- Lectura y escritura de archivos (key.txt para manejar claves de acceso).
- Uso de estructuras de datos como listas y diccionarios para almacenar y procesar la información.
- Uso de condicionales y funciones para organizar el código.
- Manejo de librerías externas para trabajar con APIs, estadísticas y visualización de datos (como OTXv2, statistics y matplotlib).

El resultado fue un programa capaz de recopilar datos actualizados, analizarlos estadísticamente, y representarlos en gráficas que muestran el comportamiento de las amenazas por año y por tipo. Esto no solo refuerza la lógica de programación, sino que también demuestra cómo podemos aplicar la programación básica a un caso práctico con utilidad en el mundo real.

Así, este proyecto integró todos los temas esenciales de la materia, fomentó el trabajo organizado, la solución de problemas reales y nos ayudó a comprender mejor cómo la programación puede aplicarse más allá del aula.