SALFORD & CO.

KEYLOGGER

ARIS JOSE PAULA
BATISTA-1116810

SALFORD & CO.

MOTIVACION

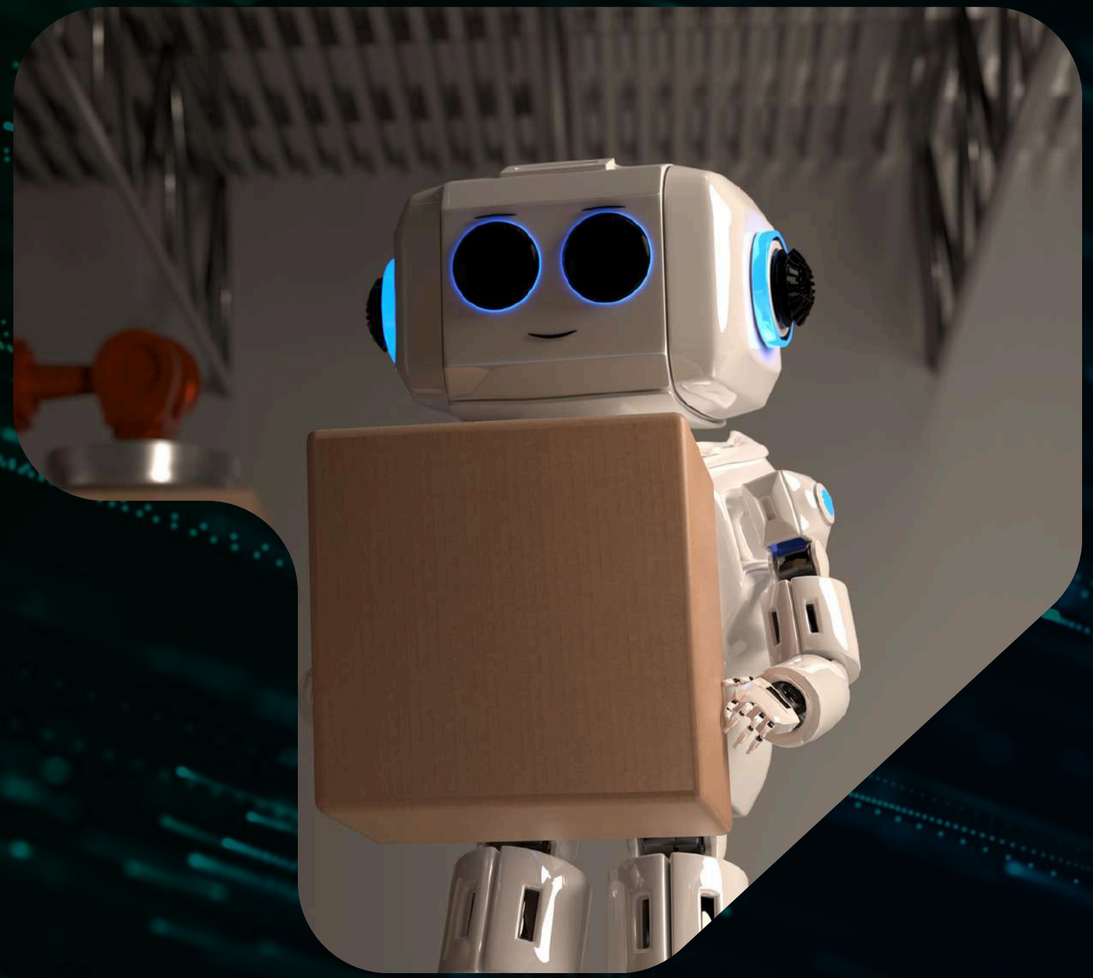
El desarrollo de un keylogger como proyecto académico responde a la necesidad de comprender de manera profunda el funcionamiento interno de los sistemas operativos y las técnicas que pueden emplearse tanto para proteger como para vulnerar la seguridad informática.

Si bien los keyloggers son comúnmente asociados a actividades maliciosas, desde un enfoque ético y educativo constituyen una excelente herramienta

SALFORD & CO.

OBJETIVO GENERAL

Este proyecto consiste en una aplicación de consola en C# que actúa como keylogger: registra las teclas presionadas por el usuario en el sistema operativo Windows, las almacena en un archivo de texto, y cuando este alcanza un tamaño determinado, envía su contenido a un chat de Telegram mediante la API de bots.



SALFORD & CO.

PROPÓSITO

Este proyecto fue desarrollado con fines educativos para demostrar:
- El uso de hooks globales de teclado en Windows mediante la API user32.dll.
- La captura y almacenamiento de pulsaciones de teclas.
- El manejo de archivos en C#.
- La ejecución de hilos.
- La comunicación con servicios externos (API de Telegram).

SALFORD & CO.

TECNOLOGÍAS UTILIZADAS

- C#: Lenguaje de programación principal.
- Windows API (user32.dll, kernel32.dll): Instalación de hooks de teclado y gestión de módulos.
- WebClient: Consumo de API HTTP.
- Threads: Envío asincrónico de logs a Telegram.
- Telegram Bot API: Recepción de logs en un chat privado.

FUNCIONAMIENTO INTERNO

1. Instalación del Hook de Teclado
Se instala un hook de bajo nivel global (WH_KEYBOARD_LL) que intercepta todas las teclas presionadas en el sistema.
2. Captura de Pulsaciones
Cada vez que se detecta una tecla:
- Se verifica si es una tecla especial (espacio, punto o coma).
- Se convierte a texto.
- Se agrega al buffer.
- Cuando el buffer supera la cantidad definida, se guarda en C:\ProgramData\mylog.txt.
3. Gestión de Logs
Si el archivo de log supera 300 bytes:
- Se copia a mylog_archive.txt.
- Se elimina el log original.
- Se crea un nuevo hilo que envía el contenido por Telegram.
4. Envío de Log a Telegram
Se usa la API de Telegram para enviar el log como un mensaje de texto.

SALFORD & CO.

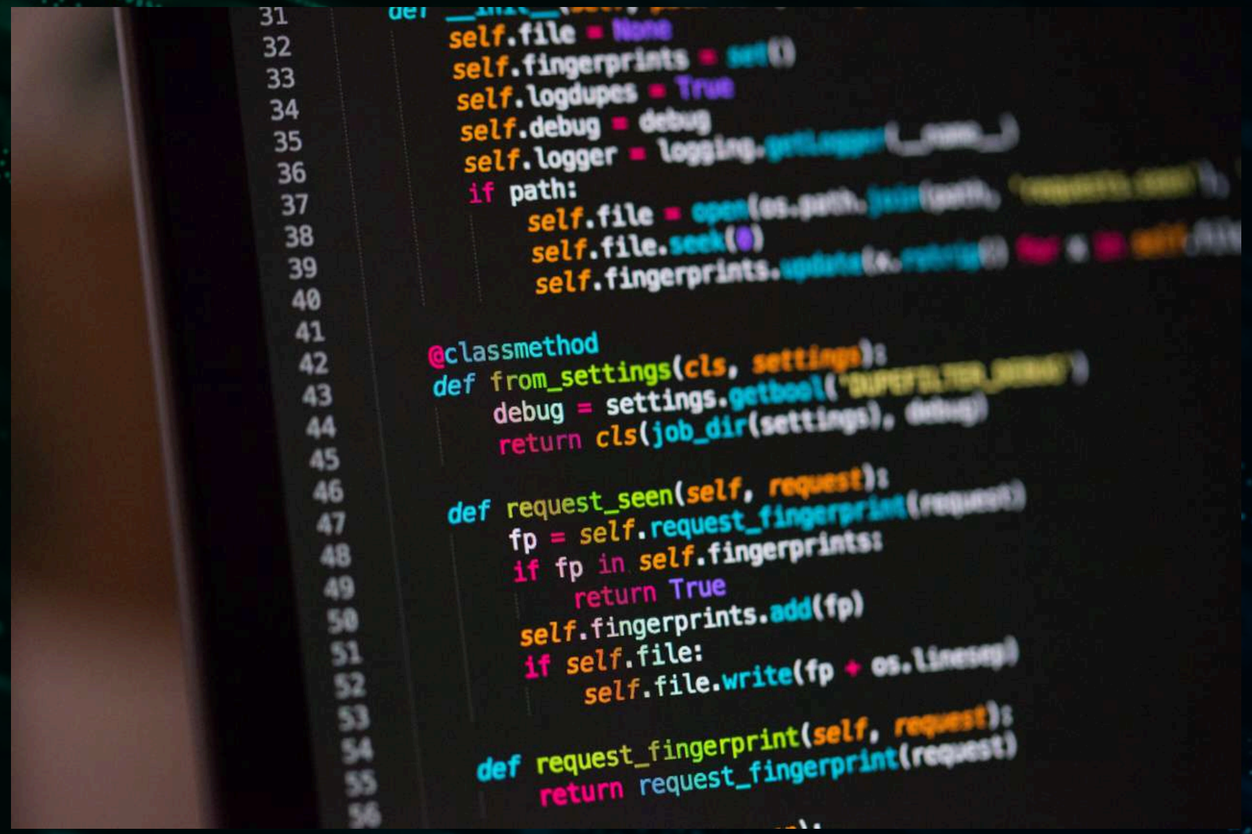
CONFIGURACIÓN

Parámetro	Descripción
BOT_TOKEN	Token único de acceso al bot de Telegram que permite enviar mensajes.
CHAT_ID	ID del chat de destino en Telegram donde se enviarán los logs.
MAX_LOG_LENGTH_BEFORE_SENDING	Tamaño máximo (en bytes) que debe alcanzar el archivo de log antes de enviarlo.
LOG_FILE_NAME	Ruta completa y nombre del archivo principal donde se guardan las teclas capturadas.
ARCHIVE_FILE_NAME	Ruta y nombre del archivo temporal donde se copia el log antes de enviarlo a Telegram.

SALFORD & CO.

¿CÓMO FUNCIONA UN BOT DE TELEGRAM?

Un bot de Telegram es como una cuenta de usuario automatizada que puede recibir y enviar mensajes, pero controlada por código en lugar de una persona. Telegram expone una API HTTP que permite a cualquier aplicación enviar y recibir mensajes a través de URLs.



IMPORTACIÓN DE LIBRERÍAS

Estas librerías permiten:

- System: Funciones básicas.
- Runtime.InteropServices: Para usar funciones de la API de Windows (DLL).
- Diagnostics: Obtener información de procesos y módulos.
- Windows.Forms: Control de aplicación y captura de teclas.
- IO: Leer y escribir archivos.
- Net: Enviar peticiones HTTP.
- Threading: Crear hilos secundarios para tareas.

```
using System;
using System.Runtime.InteropServices;
using System.Diagnostics;
using System.Windows.Forms;
using System.IO;
using System.Net;
using System.Threading;
```

CONFIGURACIÓN DE VARIABLES

Estas constantes definen:

- Token del bot de Telegram.
- ID del chat donde se enviarán los logs.
- Ruta del archivo de log.
- Ruta de la copia del log antes de enviar.
- Tamaño máximo del log antes de enviarlo.
- Cantidad de teclas que deben almacenarse antes de escribir en el archivo (0 = siempre se escribe).

```
private const string BOT_TOKEN = "...";
private const string CHAT_ID = "...";
private const string LOG_FILE_NAME = "...";
private const string ARCHIVE_FILE_NAME = "...";
private const int MAX_LOG_LENGTH_BEFORE_SENDING = 300;
private const int MAX_KEYSTROKES_BEFORE_WRITING_TO_LOG = 0;
```

VARIABLES PARA EL HOOK

- WH_KEYBOARD_LL: Tipo de hook global de teclado.
- WM_KEYDOWN: Mensaje que recibe cuando se presiona una tecla.
- hook: Referencia al hook.
- llkProcedure: Callback asociado al hook.
- buffer: Almacena las teclas presionadas temporalmente.

```
private static int WH_KEYBOARD_LL = 13;
private static int WM_KEYDOWN = 0x0100;
private static IntPtr hook = IntPtr.Zero;
private static LowLevelKeyboardProc llkProcedure = HookCallback;
private static string buffer = "";
```

MAIN

- SetHook(): Instala el hook de teclado.
- Application.Run(): Mantiene la aplicación ejecutándose escuchando teclas.
- UnhookWindowsHookEx(): Desinstala el hook cuando se cierra.

```
hook = SetHook(llkProcedure);
Application.Run();
UnhookWindowsHookEx(hook);
```

HOOKCALLBACK()

SEsta función se ejecuta cada vez que se presiona una tecla:

1. Guarda el contenido del buffer en archivo cuando excede la cantidad de teclas configuradas.
2. Si el archivo de log supera el tamaño límite:
   - Lo copia a otro archivo.
   - Lo borra.
   - Envía el contenido por Telegram en un hilo aparte.
3. Detecta teclas especiales (., ,, espacio) y el resto las guarda con su nombre.
4. Llama al siguiente hook en la cadena.

```
private static IntPtr HookCallback(int nCode, IntPtr wParam, IntPtr lParam)
```

PUBLIC STATIC VOID SENDTELEGRAM(STRING MESSAGE)

Envía el contenido del log por Telegram:

1. Obtiene nombre de la máquina.
2. Obtiene IP pública.
3. Construye un mensaje con log + nombre + IP.
4. Guarda una copia previa en telegram_message_preview.txt.
5. Envía mensaje vía API de Telegram usando WebClient.

```
public static void sendTelegram(string message)
```

GETPUBLICIP()

```
public static string GetPublicIP()
```

Devuelve la IP pública consultando a https://api.ipify.org. Si no puede, devuelve "IP no disponible".

SETHOOK()

Instala el hook de teclado global:

- Obtiene el módulo actual.
- Llama a SetWindowsHookEx() con los parámetros correctos.

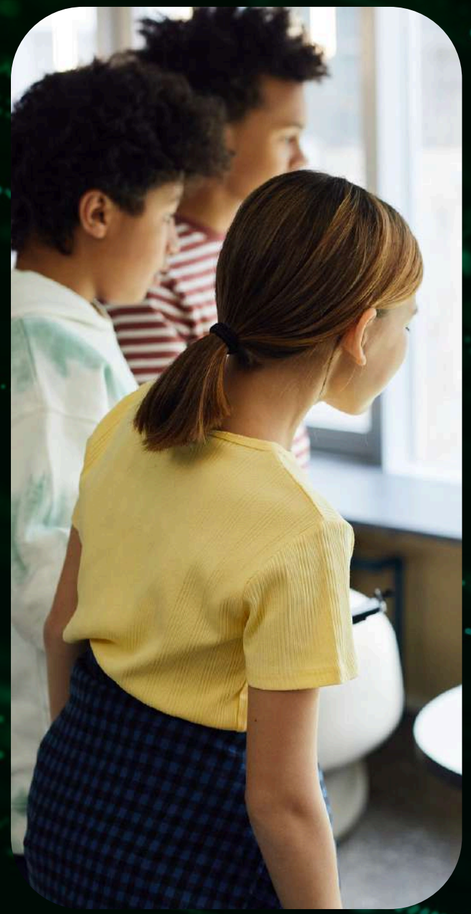
```
private static IntPtr SetHook(LowLevelKeyboardProc proc)
```

FUNCIONES DE LA API DE WINDOWS

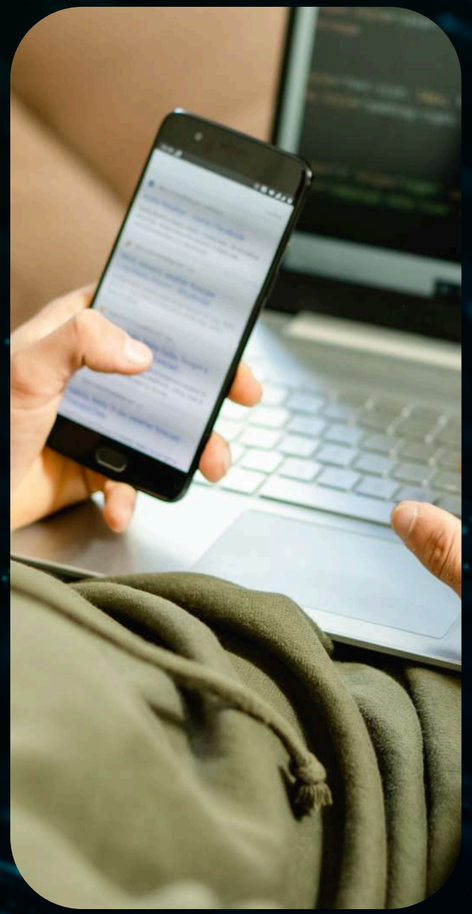
Permiten:

- CallNextHookEx(): Pasar el evento al siguiente hook.
- SetWindowsHookEx(): Instalar hook.
- UnhookWindowsHookEx(): Desinstalar hook.
- GetModuleHandle(): Obtener handle de un módulo.

```
[DllImport("user32.dll")]
...
[DllImport("kernel32.dll")]
...
```



DEMO



SALFORD & CO.

GRACIAS