

**KEYLOGGER**

**FUNDAMENTALS OF MALWARE ANALYSIS**

**GROUP MEMBERS:**

**MUHAMMAD IBRAHEEM (21K-4765)**

**SYED TARHIM UR RAB (21K-4782)**

**4-DECEMBER-2023**

**INTRODUCTION:**

In the realm of cybersecurity and software development, understanding how certain malicious tools work is crucial for creating effective defense mechanisms. This project explores the creation of a keylogger using C++ and the Windows API. A keylogger is a program that records the keystrokes made on a computer without the user's knowledge.

**OBJECTIVES:**

In this project, the primary objectives revolved around understanding and implementing key aspects of Windows API, file input/output operations, event handling, and stealth mode functionality. Successfully achieving these goals allowed the development of a functional keylogger in C++. The program demonstrates a grasp of Windows API functions to capture keyboard inputs discreetly, utilizing file I/O operations to store the recorded data in a text file. Event handling mechanisms were implemented to efficiently detect and log various keypress events. Furthermore, the inclusion of a stealth mode ensures that the keylogger operates discreetly, remaining inconspicuous to users. Overall, the project provides a practical exploration of fundamental concepts in system-level programming, emphasizing the responsible and ethical use of such knowledge in the realm of cybersecurity education.

Top of Form

**WORKING:**

1. **Initialization:**

The program begins by initializing necessary libraries, including <iostream>, <windows.h>, <winuser.h>, and <fstream>. The main() function is executed, initiating the program.

1. **Stealth Mode:**

The StealthMode() function is called, which allocates a console window using AllocConsole() and then hides it using the ShowWindow function. This ensures that the keylogger operates without a visible console window, enhancing its stealthiness.

1. **Logging Initialization:**

The StartLogging() function is called, representing the main logging loop of the keylogger.

1. **Keylogger Loop:**

Within the StartLogging() function, an infinite loop is established, continuously monitoring keyboard inputs.

1. **Key Detection:**

The loop iterates through virtual key codes (c) ranging from 8 to 222, checking if any key is pressed using GetAsyncKeyState(c) == -32767.

1. **Logging Process:**

Upon detecting a keypress, the program opens a file stream (ofstream) to the specified file path ("C:\Users\Public\Record.txt") in append mode.

Depending on the type of key pressed, the program enters a series of conditions to determine the character to be logged.

1. **Special Key Handling:**

Special keys like space, enter, tab, backspace, and arrow keys are handled separately to ensure meaningful logging.

1. **Shift Key Handling:**

If the shift key is pressed, the program adjusts the character accordingly to capture uppercase letters or symbols.

1. **File Writing:**

The determined character is written to the file, and the file stream is closed.

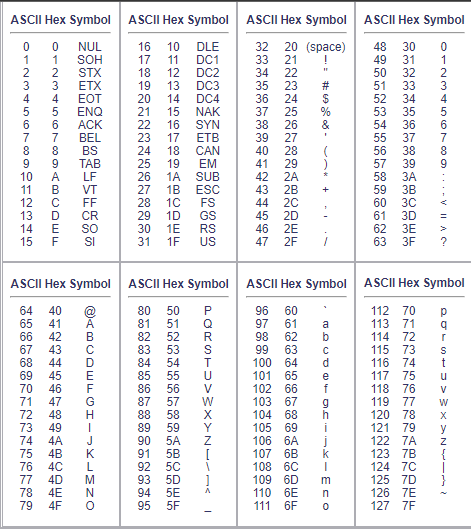
1. **Loop Continuation:**

The loop continues to monitor keypresses indefinitely.

1. **Closing the Program:**

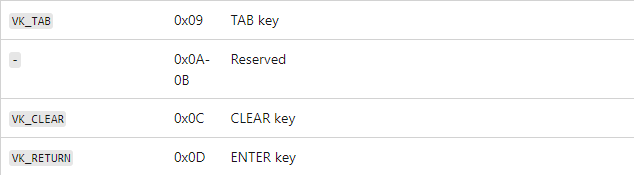
The program can be closed by terminating the execution externally.

While a visual diagram would typically include flowcharts or diagrams for each function and the interactions between them, this textual representation outlines the sequential execution and key components of the keylogger's functionality.

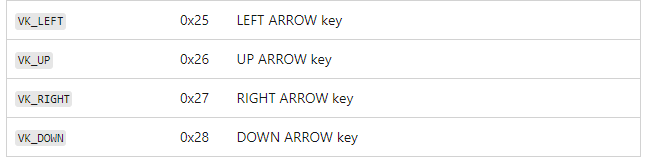


the above table is ascii table which shows the ascii symbols corresponding to ascii values which is a major part of this project.

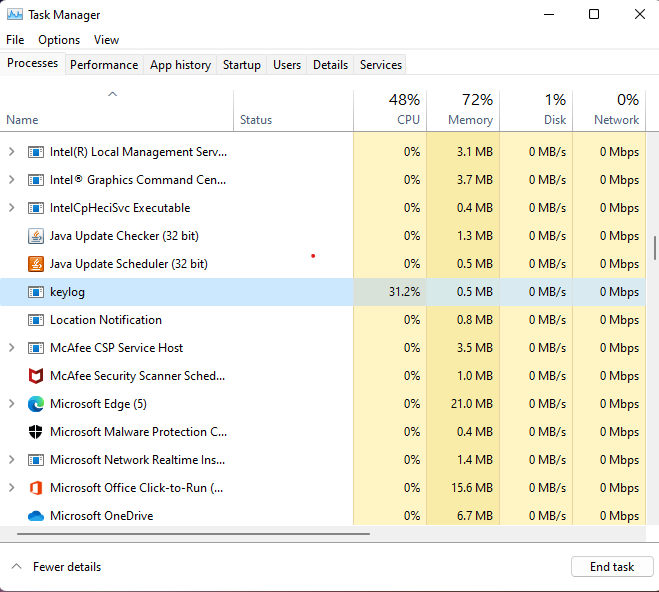
in function startlogging we have also used virtualkeys functions to detect the virtual keys like backspace arrowkeys, delete key, return/enter key. the functions used to detect these keys are attached below:







In order to stop this program you have to go to task manager as the executible file is running at the backend.



**TOOLS USED:**

* DEV C++ IDE
* HEADER FILES (Fstream, Windows.h, Winuser.h)

**CONCLUSION:**

In conclusion, the keylogger project in C++ utilizing the Windows API has provided a practical exploration of fundamental concepts in system-level programming. The project aimed to achieve several objectives, including understanding Windows API functions, implementing file input/output operations, incorporating event handling for keypress detection, and enabling stealth mode functionality.

The inclusion of a stealth mode added an additional layer of complexity to the project, making the keylogger operate in a less conspicuous manner by hiding the console window. This feature demonstrated the importance of considering user experience and ethical considerations when developing applications that may have sensitive implications.