KEYLOGGER

Presented By: KRUTHIKA ROZHINI.K - M.I.E.T ENGINEERING COLLEGE - CSE

OUTLINE

- Problem Statement
- Proposed System/Solution
- System Development Approach
- Algorithm & Deployment
- Result
- Conclusion
- Future Scope
- References

User Problem Statement

In today's digital age, where cybersecurity threats loom large, one of the significant concerns is the proliferation of keyloggers, stealthy software tools designed to monitor and record keystrokes on a user's computer without their knowledge. Keyloggers pose a severe threat to individuals and organizations as they can capture sensitive information such as passwords, credit card details, and other personal data, leading to identity theft, financial loss, and privacy breaches.

PROPOSED SYSTEM/SOLUTION

The proposed system offers a customizable keylogging solution implemented in Python using the **pynput** library. The system consists of the following components:

•**Data Collection:** Captures keyboard input events in real-time, recording pressed, held, and released keys.
•**Data Preprocessing:** Cleans and preprocesses the collected keystroke data to ensure accuracy and consistency.
•**Deployment:** Offers a user-friendly interface for initiating and terminating the keylogging process, ensuring ease of use for users.
•**Result:** Logs keystrokes in both text and JSON formats, providing users with flexibility in accessing and analyzing the recorded data.

System approach

- **Data Collection:**
  - Utilize the **pynput** library to capture keyboard input events in real-time.
  - Record pressed, held, and released keys, along with timestamps.
- **Data Preprocessing:**
  - Clean and preprocess the captured keystroke data to handle any inconsistencies or anomalies.
  - Transform the data into a structured format suitable for analysis.
- **Algorithm Implementation:**
  - Develop event-driven functions to capture and log keystrokes.
  - Implement logic to distinguish between different key events (pressed, held, released).

System approach[contd]

•**Deployment:**

- •Create a user-friendly interface using Tkinter or similar libraries to initiate and terminate the keylogging process.
- •Ensure the application's compatibility and usability across different operating systems.

•**Evaluation:**

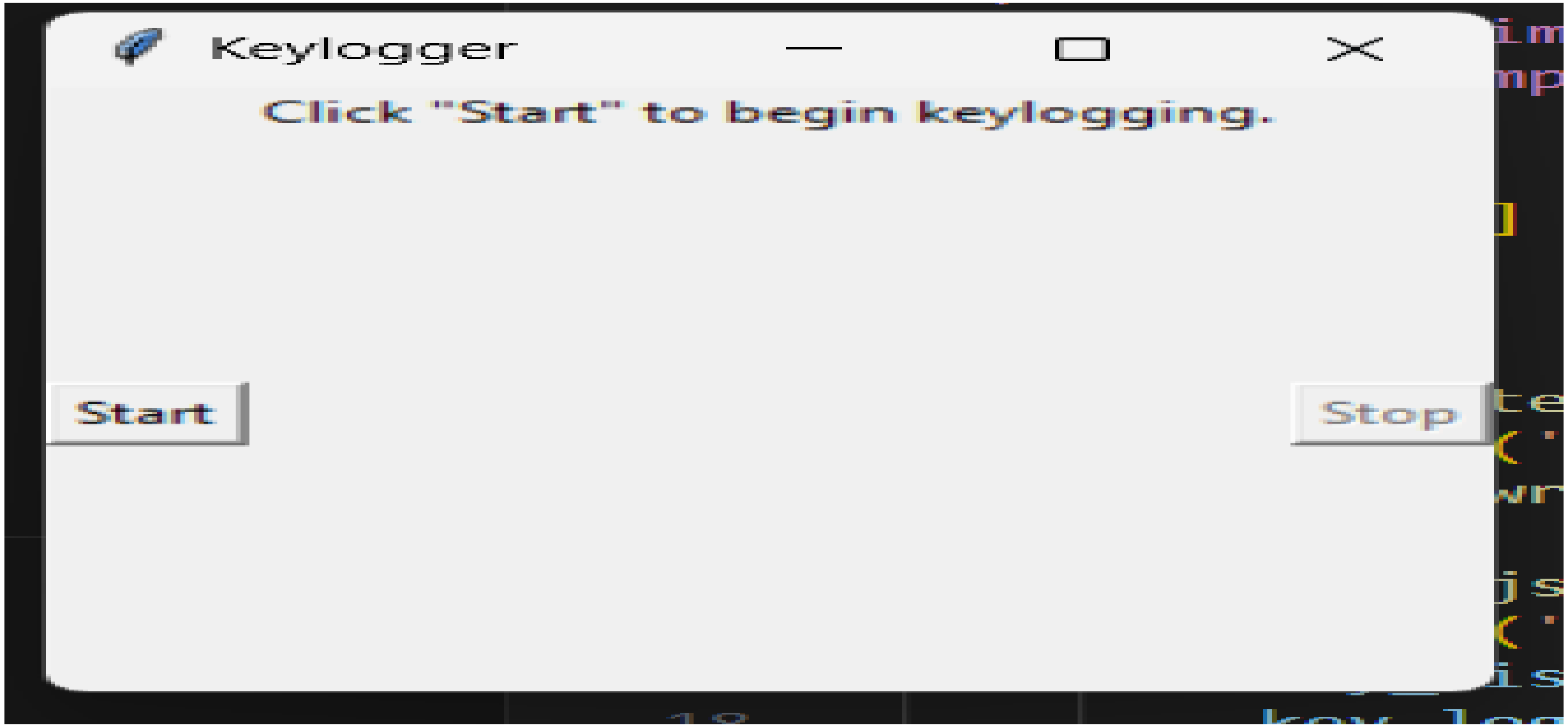
- •Assess the performance of the keylogging system based on metrics such as accuracy, efficiency, and resource usage.
- •Gather user feedback to identify areas for improvement and optimization.

ALGORITHM & DEPLOYMENT

•**Keylogger Algorithm**: The system employs event-driven programming to capture keyboard input events using the **pynput** library. It distinguishes between pressed, held, and released keys, generating corresponding log entries.

•**Deployment**: The keylogger can be deployed on any system with Python installed. It operates in the background, logging keystrokes discreetly while the user continues with their regular activities.

Result



Click "Start" to begin keylogging.

CONCLUSION

In conclusion, the developed keylogger provides users with a flexible and efficient solution for logging keystrokes. By leveraging Python and the **pynput** library, the system offers real-time monitoring capabilities and customizable logging options. The graphical user interface enhances usability, making it accessible to users with varying technical backgrounds.

Future Scope

- Future enhancements to the keylogger could include:
- Adding encryption functionality to secure keystroke logs.
- Implementing remote logging capabilities for monitoring keystrokes across multiple devices.
- Integrating machine learning algorithms for anomaly detection and pattern recognition in keystroke logs.

References

- Python Documentation: https://www.python.org/doc/
- Tkinter Documentation: https://docs.python.org/3/library/tkinter.html
- pynput Documentation: https://pypi.org/project/pynput/
- JSON Documentation: https://docs.python.org/3/library/json.html

THANK YOU