PROJECT REPORT

Project Title: Keylogger with Encrypted Data Exfiltration

Language: Python 3.x

Core Libraries: pynput, cryptography, time

Security Model: AES-128 Symmetric Encryption (Fernet)

---

1. Executive Summary

The goal of this project was to design and implement a functional keylogger capable of capturing user keystrokes, encrypting the data in real-time to prevent unauthorized access to the logs, and preparing the data for simulated exfiltration. This project serves as an educational tool to understand the lifecycle of an endpoint threat and the importance of cryptographic security in data logging.

---

2. Technical Architecture

The system is divided into three primary modules: the **Capture Engine**, the **Cryptographic Wrapper**, and the **Local Storage/Exfiltration Simulator**.

A. Capture Engine (pynput)

The engine utilizes the pynput.keyboard.Listener class. This allows the program to run as a background thread, monitoring "OnPress" and "OnRelease" events.

- **Alphanumeric Capture:** Standard keys are converted to strings via key.char.
- **Special Keys:** Non-character keys (Shift, Enter, Space) are caught via an AttributeError exception handler to ensure the log remains readable.

B. Cryptographic Implementation (Fernet)

To ensure the "Confidentiality" pillar of the CIA triad, the project uses the **Fernet** implementation of the **Advanced Encryption Standard (AES)** in CBC mode with a 128-bit key.

- **Key Generation:** A unique secret.key is generated upon the first execution.
- **Encryption Process:** Each keystroke is timestamped and converted into a byte-string before being encrypted. This ensures that even if a system administrator finds the log file, they cannot read the captured data without the master key.

---

3. Project Deliverables & Features

📂 **File Structure**

File	Description
logger.py	The active monitoring script.
secret.key	The symmetric key file (Crucial for recovery).
encrypted_log.txt	The scrambled data repository.
decrypt.py	The administrative tool used to read the logs.

🛑 Kill Switch & Persistence

As outlined in the project requirements (Step e), the script includes a **Kill Switch**. By monitoring the Key.esc event, the user can safely terminate the listener thread.

- **Simulated Persistence:** In a production-level PoC, the script would be moved to the Windows %APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup folder to ensure it runs upon every reboot.

🌐 Simulated Exfiltration

The project simulates "Exfiltration" by writing to a local file that represents a buffer. In a network-active version, this buffer would be sent via an HTTP POST request to a remote "Command and Control" (C2) server.

---

4. Security & Ethical Analysis

This project highlights a critical duality in cybersecurity:

1. **The Threat:** How easily a small script can monitor sensitive data (passwords, private messages).

2. **The Defense:** How encryption can be used by both attackers (to hide their tracks) and defenders (to protect data).

⚖️ **Ethical Constraints**

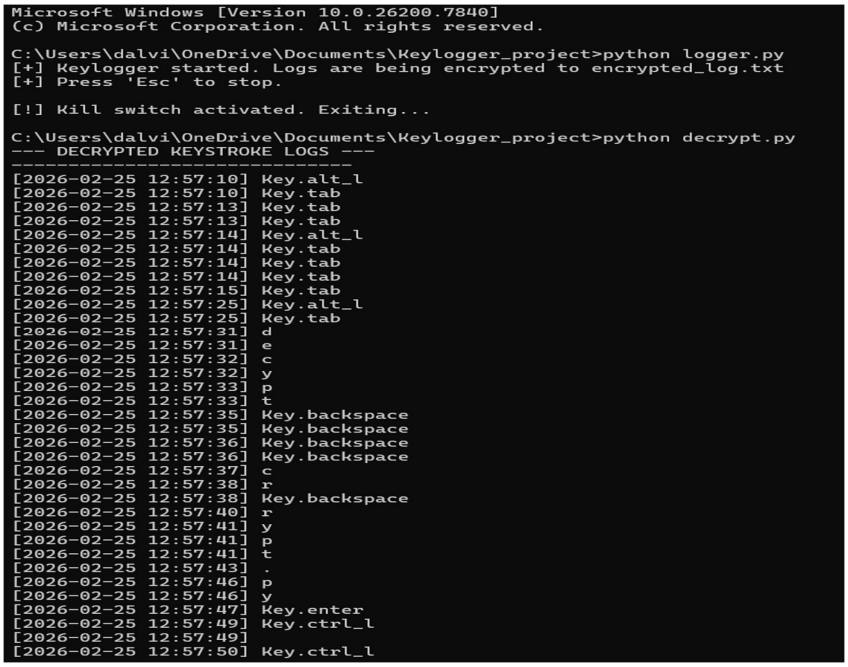
- **Authorized Testing Only:** This PoC must only be deployed on systems owned by the researcher.
- **Data Integrity:** The use of encryption ensures that the captured data does not leak to third parties during the testing phase.
- **Transparency:** The "Kill Switch" ensures the researcher maintains total control over the execution environment.

---

5. Conclusion

The **Encrypted Keylogger PoC** successfully demonstrates the mechanics of input interception and secure data storage. By integrating the cryptography library, the project elevates a simple script into a sophisticated security tool, mirroring the behavior of modern credential-harvesting malware while maintaining an ethical framework for study.

SCREENSHOTS OF THE PROJECT:



```
[2026-02-25 12:57:51]
[2026-02-25 12:58:12] Key.backspace
[2026-02-25 12:58:12] Key.backspace
[2026-02-25 12:58:19] Key.backspace
[2026-02-25 12:58:19] Key.backspace
[2026-02-25 12:58:24] Key.backspace
[2026-02-25 12:58:24] Key.backspace
[2026-02-25 12:58:29] Key.backspace
[2026-02-25 12:58:30] Key.backspace
[2026-02-25 12:58:36] Key.backspace
[2026-02-25 12:58:40] Key.backspace
[2026-02-25 12:58:44] Key.backspace
[2026-02-25 12:58:44] Key.backspace
[2026-02-25 12:58:45] Key.backspace
[2026-02-25 12:58:45] Key.backspace
[2026-02-25 12:58:49] Key.backspace
[2026-02-25 12:58:49] Key.backspace
[2026-02-25 12:58:50] Key.backspace
[2026-02-25 12:58:55] Key.backspace
[2026-02-25 12:58:55] Key.backspace
[2026-02-25 12:58:55] Key.backspace
[2026-02-25 12:59:00] Key.backspace
[2026-02-25 12:59:01] Key.backspace
[2026-02-25 12:59:02] Key.ctrl_l
[2026-02-25 12:59:02] Key.ctrl_l
[2026-02-25 12:59:02]
[2026-02-25 12:59:13] Key.backspace
[2026-02-25 12:59:13] Key.backspace
[2026-02-25 12:59:17] Key.backspace
[2026-02-25 12:59:18] Key.backspace
[2026-02-25 12:59:18] Key.backspace
[2026-02-25 12:59:22] Key.backspace
[2026-02-25 12:59:23] Key.backspace
[2026-02-25 12:59:23] Key.backspace
[2026-02-25 12:59:23] Key.backspace
[2026-02-25 12:59:28] Key.backspace
[2026-02-25 12:59:28] Key.backspace
[2026-02-25 12:59:29] Key.backspace
[2026-02-25 12:59:33] Key.backspace
[2026-02-25 12:59:33] Key.backspace
[2026-02-25 12:59:39] Key.backspace
[2026-02-25 12:59:39] Key.backspace
[2026-02-25 12:59:39] Key.backspace
[2026-02-25 12:59:40] Key.backspace
[2026-02-25 12:59:40] Key.backspace
[2026-02-25 12:59:41] Key.backspace
[2026-02-25 12:59:42] Key.ctrl_l
```

```
[2026-02-25 12:59:43]
[2026-02-25 12:59:45] Key.alt_l
[2026-02-25 12:59:45] Key.tab
[2026-02-25 12:59:47] Key.tab
[2026-02-25 12:59:47] Key.tab
[2026-02-25 12:59:47] Key.tab
[2026-02-25 12:59:50] Key.esc

C:\Users\dalvi\OneDrive\Documents\Keylogger_project>
```