🔐 PROJECT REPORT 🔐

Post-Breach Black Box Logger — Encrypted, Tamper-Evident Keylogger

🔹 Introduction

Today's organizations face sophisticated cyberthreats and insider attacks that often destroy or manipulate logs to conceal their traces.
Effective incident response depends on having a reliable and tamper-evident view into what exactly occurred on a compromised host.

The **Post-Breach Black Box Logger** was designed to aid cybersecurity analysts by:

✅ **Capturing keystrokes alongside process and directory context**
✅ **Encrypting all captured data to maintain confidentiality**
✅ **Applying HMAC-SHA256 to guarantee tamper-evident storage**

This lightweight tool lets incident responders piece together a timeline of attacker activity, without needing elevated privilege or kernel components.

🔹 **Abstract**

This keylogger is a lightweight, forensic-grade tool tailored for incident response and post-breach investigations.
 It performs:

➥ **AES-256-GCM Encryption:** To keep captured keystrokes and context confidential.
 ➥ **HMAC-SHA256:** To assure tamper-evident storage of messages.
 ➥ **Context Capture:** To provide additional information (like process and directory) alongside keystrokes.

The encrypted messages, base64-encoded and serialized in JSON, are convenient for storage, parsing, and eventual decryption by analysts.

🔹 Tools Used

✅ **Python 3.x** — Main scripting and implementation
 ✅ **pynput (Python)** — To capture keystrokes
 ✅ **pywin32, psutil** — To extract process and directory context
 ✅ **Cryptography (AES-GCM, HMAC)** — To encrypt messages and validate integrity
 ✅ **os, json, base64** — To handle file operations and encoding
 ✅ **Environment Variables or Windows Credential Vault** — To safely store the encryption key

---

🔹 Key Features

➥ **AES-256-GCM Encryption:**
 Encrypts each keystroke with a unique 96-bit nonce.

➥ ** HMAC-SHA256 Tamper Detection:**
 Ensuring messages have not been altered in storage.

➥ **Process and Window Title Capture:**

Provides context alongside keystrokes — useful for attack reconstruction.

➥ **Environment-Safe:**

Runs under **current-user privilege**, avoiding suspicion and administrative intervention.

➥ **Base64 Encoded Logs:**

For convenient storage, parsing, and eventual decryption by analysts.

➥ **Controlled Duration:**

Easily exits with a predefined key (such as ESC).

🔹 Implementation Summary (Workflow)

➥ **Initiate:**

Logger starts and attaches to the keyboard stream.

➥ **Capture:**

For each keystroke, it:

- Records the key pressed
- Captures the active window title
- Logs the associated process path

➥ **Encrypt:**

Encrypts the data with AES-256-GCM alongside a unique 96-bit nonce.

➥ **HMAC:**

Generates a HMAC-SHA256 to guarantee tamper-evident storage.

➥ Serialize:

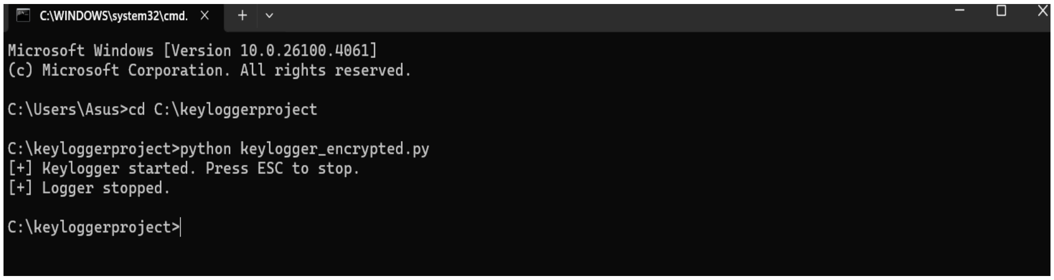
Packages the encrypted blob, HMAC, and context into base64-encoded JSON.

➥ Store:

Writes safely to a local file.

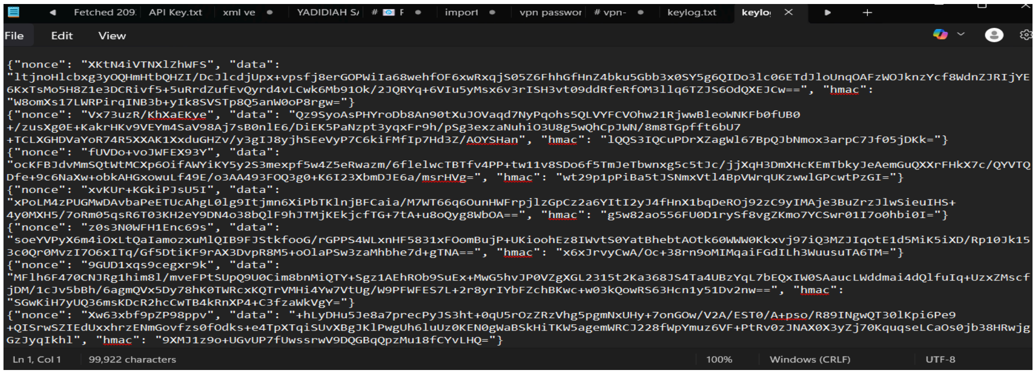
🔹 Photos and Screenshots

✅ **Screenshot 1:** Logger in action (CMD window)



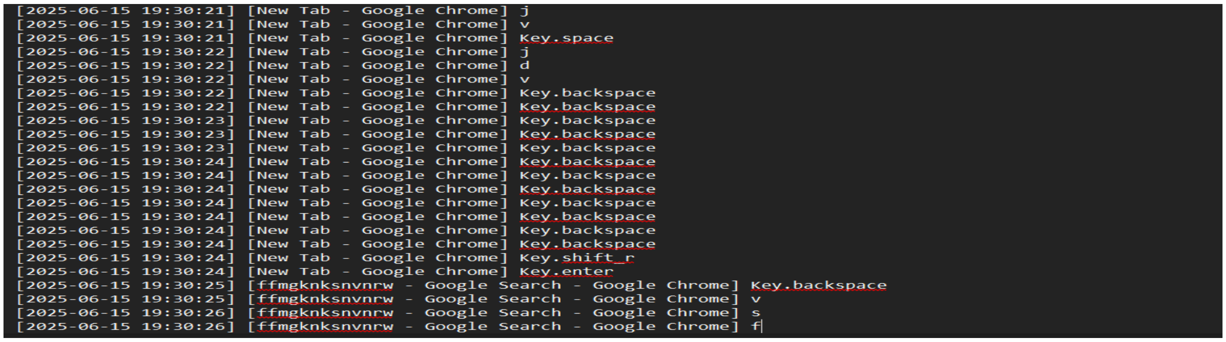
```
Microsoft Windows [Version 10.0.26100.4061]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Asus>cd C:\keyloggerproject

C:\keyloggerproject>python keylogger_encrypted.py
[+] Keylogger started. Press ESC to stop.
[+] Logger stopped.

C:\keyloggerproject>
```

✅ **Screenshot 2:** Encrypted base64-encoded messages in a text file



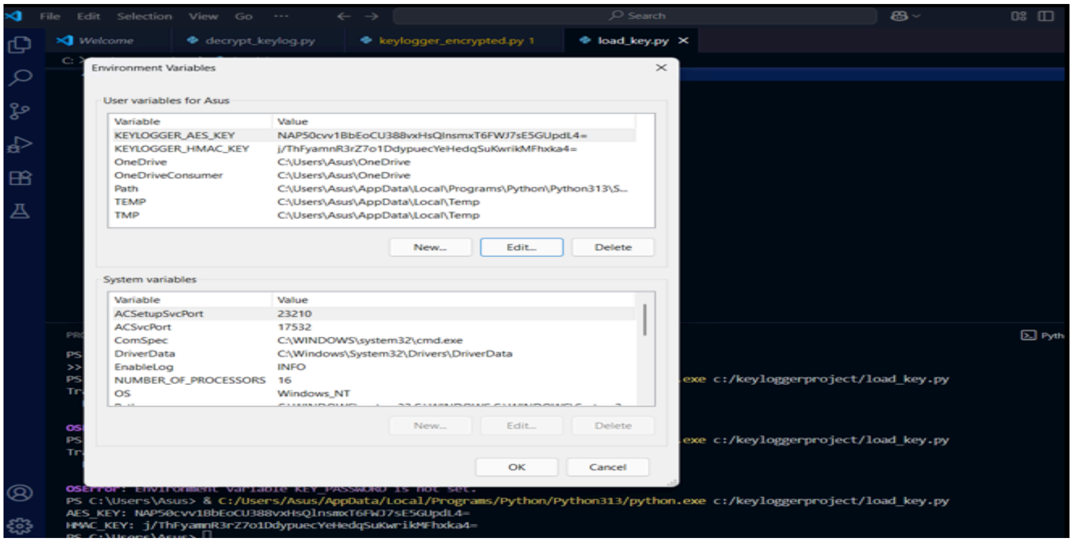
```
{"nonce": "XKtN4iVTNXlZhWFS", "data":
"ltjnoHlcbxg3yOQHmHtbQHZI/DcJlcdjUpx+vpsfj8erGOPWiIa68wehfOF6xwRxqjS05Z6FhhGfHnZ4bku5Gbb3x0SY5g6QIDo3lc06ETdJloUnqOAFzWOJknzYcf8WdnZJRIjYE
6KxTsMo5H8Z1e3DCRivf5+5uRrdZufEvQyrd4vLCwk6Mb91Ok/2JQRYq+6VIu5yMsx6v3rISH3vt09ddRfeRfOM3llq6TZJS6OdQXEJCw==", "hmac":
"W8omXs17LWRPirqINB3b+yIk8SVSTp8Q5anW0oP8rgw="}
{"nonce": "Vx73uzR/KhXaEKye", "data": "Qz9SyoAsPHYroDb8An90tXuJOVaqd7NyPqohs5QLVYFCVOhw21RjwwBleoWNKFb0fUB0
+/zusXg0E+KakrHKv9VEYm4SaV98Aj7sB0nlE6/DiEK5PaNzpt3yqxFr9h/pSg3exzaNuhiO3U8g5wQhCpJWN/8m8TGpfft6bU7
+TCLXGHDVaYoR74R5XXAK1XxduGHZv/y3gIJ8yjhSEeVyP7C6kiFMfIp7Hd3Z/AOYSHan", "hmac": "lQQS3IQCuPDrXZagWl67BpQJbNmox3arpC7Jf05jDKk="}
{"nonce": "fUVDo+voJWFEX93Y", "data":
"ocKFBJdvMmSQtWtMCXp6OifAWYiKY5y2S3mexpf5w4Z5eRwazm/6flelwcTBTfv4PP+tw11v8SDo6f5TmJeTbwnxg5c5tJc/jjXqH3DmXHcKEmTbkyJeAemGuQXXrFHkX7c/QYVTQ
Dfe+9c6NaXw+obkAHGxowuLf49E/o3AA493FOQ3g0+K6I23XbmDJE6a/msrHVg=", "hmac": "wt29p1pPiBa5tJSNmxVtl4BpVWrqUKzwwlGPcwtPzGI="}
{"nonce": "xvKUr+KGkiPJsU5I", "data":
"xPoLM4zPUGMwDAvbaPeETUcAhgL0lg9Itjmn6XiPbTKlnjBFCaia/M7WT66q6OunHWFrpjlzGpCz2a6YItI2yJ4fHnX1bqDeROj92zC9yIMAje3BuZrzJlwSieuIHS+
4y0MXH5/7oRm05qsR6T03KH2eY9DN4o38bQlF9hJTMjKEkjcfTG+7tA+u8oQyg8WbOA==", "hmac": "g5w82ao556FU0D1rySf8vgZKmo7YCSwr01I7o0hbi0I="}
{"nonce": "z0s3N0WFH1Enc69s", "data":
"soeYVPyX6m4iOxLtQaIamozxuMlQIB9FJStkfooG/rGPPS4WLxnHF5831xFOomBujP+UKioohEz8IWvtS0YatBhebtAOtk60WWW0Kkxvj97iQ3MZJIqotE1d5MiK5iXD/Rp10Jk15
3c0Qr0MvzI7O6xITq/Gf5DtiKF9rAX3DvpR8M5+oOlaPSw3zaMhbhe7d+gTNA==", "hmac": "x6xJrvyCwA/Oc+38rn9oMIMqaiFGdILh3WuusuTA6TM="}
{"nonce": "9GUD1xqs9cegxr9k", "data":
"MFlh6F47OCNJRg1him8l/mveFPtSUpQ9U0Cim8bnMiQTY+Sgz1AEhROb9SuEx+MwG5hvJP0VZgXGL2315t2Ka368JS4Ta4UBzYqL7bEQxIW0SAaucLWddmai4dQlfuIq+UzxZMscf
jDM/1cJv5bBh/6agmQVx5Dy78hK0TWRcxKQTrVMHi4Yw7VtUg/W9PFWFES7L+2r8yrIYbFZchBKwc+w03kQowRS63Hcn1y51Dv2nw==", "hmac":
"SGwKiH7yUQ36msKDcR2hcCwTB4kRnXP4+C3fzaWkVgY="}
{"nonce": "Xw63xbf9pZP98ppv", "data": "+hLyDHu5Je8a7precPyJS3ht+0qU5rOzZRzVhg5pgmNxUHy+7onGOw/V2A/EST0/A+pso/R89INgwQT30lKpi6Pe9
+QISrwSZIEdUxxhrzENmGovfzs0fOdks+e4TpXTqiSUvXBgJKlPwgUh6luUz0KEN0gWaBSkHiTKW5agemWRCJ228fWpYmuz6VF+PtRv0zJNAX0X3yZj70KquqseLCaOs0jb38HRwjg
GzJyqIkhl", "hmac": "9XMJ1z9o+UGvUP7fUwssrwV9DQGBqQpzMu18fCYvLHQ="}
```

✅ **Screenshot 3:**Decrypted log showing keystrokes and context



```
[2025-06-15 19:30:21] [New Tab - Google Chrome] j
[2025-06-15 19:30:21] [New Tab - Google Chrome] v
[2025-06-15 19:30:21] [New Tab - Google Chrome] Key.space
[2025-06-15 19:30:22] [New Tab - Google Chrome] j
[2025-06-15 19:30:22] [New Tab - Google Chrome] d
[2025-06-15 19:30:22] [New Tab - Google Chrome] v
[2025-06-15 19:30:22] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:22] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:23] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:23] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:23] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.backspace
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.shift_r
[2025-06-15 19:30:24] [New Tab - Google Chrome] Key.enter
[2025-06-15 19:30:25] [ffmgknksnvnrw - Google Search - Google Chrome] Key.backspace
[2025-06-15 19:30:25] [ffmgknksnvnrw - Google Search - Google Chrome] v
[2025-06-15 19:30:26] [ffmgknksnvnrw - Google Search - Google Chrome] s
[2025-06-15 19:30:26] [ffmgknksnvnrw - Google Search - Google Chrome] f
```

✅ **Screenshot 4:** Secure key storage in environment variables



Environment Variables

User variables for Asus

Variable	Value
KEYLOGGER_AES_KEY	NAP50cvv1BbEoCU388vxHsQlnsmxT6FWJ7sE5GUpdL4=
KEYLOGGER_HMAC_KEY	j/ThFyamnR3rZ7o1DdypuecYeHedqSuKwrikMFhxka4=
OneDrive	C:\Users\Asus\OneDrive
OneDriveConsumer	C:\Users\Asus\OneDrive
Path	C:\Users\Asus\AppData\Local\Programs\Python\Python313\S...
TEMP	C:\Users\Asus\AppData\Local\Temp
TMP	C:\Users\Asus\AppData\Local\Temp

New... Edit... Delete

System variables

Variable	Value
ACSetupSvcPort	23210
ACSvcPort	17532
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
EnableLog	INFO
NUMBER_OF_PROCESSORS	16
OS	Windows_NT

New... Edit... Delete

OK Cancel

```
PS
>>
PS                                                            .exe c:/keyloggerproject/load_key.py
Tr

OS
PS                                                            .exe c:/keyloggerproject/load_key.py
Tr

OSError: Environment variable KEY_PASSWORD is not set.
PS C:\Users\Asus> & C:/Users/Asus/AppData/Local/Programs/Python/Python313/python.exe c:/keyloggerproject/load_key.py
AES_KEY: NAP50cvv1BbEoCU388vxHsQlnsmxT6FWJ7sE5GUpdL4=
HMAC_KEY: j/ThFyamnR3rZ7o1DdypuecYeHedqSuKwrikMFhxka4=
PS C:\Users\Asus> 
```

🔹 Conclusion

The **Post-Breach Black Box Logger** successfully provides a lightweight, tamper-evident, encrypted logging solution tailored for **post-incident investigations**.

➥ It performs:

✅ **Confidential data collection**

✅ **Context enrichment**

✅ **Integrity validation**

➥ All this while avoiding kernel components or elevated privilege — making it an ideal tool for lightweight incident response.

➥ The implementation highlights proficient use of:
✅ **Cryptography fundamentals (AES-GCM, HMAC)**
✅ **Python scripting and API integration**
✅ **Environment-safe techniques and lightweight persistence**

🔹 Future Improvement Ideas

➥ Integrate GUI for forensics team to view/decrypt messages directly
➥ Provide automatic backup to a remote server
➥ Support additional context (network activity, file accesses)

✅ **Final Notes:**
This project underscores my ability to combine coding skills with cybersecurity principles — designing a lightweight, tamper-evident, post-breach logging tool that is **technically sophisticated yet practical for incident response**.
