Deliverable:

- Source Code covering:
  - Windows based key-logger app written in C++
  - A keylogger-server that store the keylogs written in python (with flask)
- A slide deck that explains the whole project.
- Project Summary

Project Goal:

- From the Red team's perspective explore how to build a Windows based keylogger and improve the stealth of the app over iteration of discovery analysis.
- From the Blue team's perspective, analyse how to detect the keylogger.

Short Description:

A keylogger is a spyware that records keyboard strokes made on a device. In the CIA-triad cybersecurity concept, this spyware is attacking from the "Confidentiality" component.

While most of the case for attacks targeting the "confidentiality" component, can be countermeasured with encryption, it won't work for keylogger. As the key strokes are captured directly by our spyware in the target device not upon data transmitting across the device.

Design Consideration:

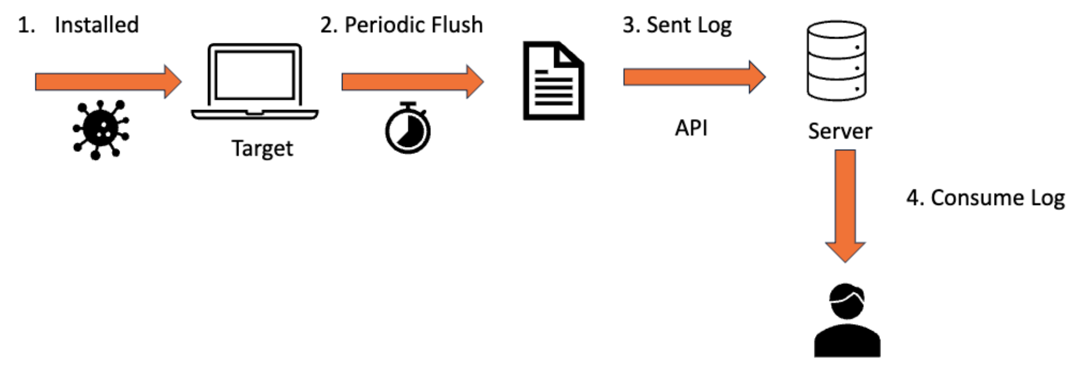
- Programming Language: Python or **CPP**
  A lot of keyloggers found online are built using python, and honestly python is my go-to language, but after some research keylogger with CPP can have more customization in terms of **stealth considering CPP is compiled** while Python is not.
- OS Target: **Windows** or MacOS
  I am an active MacOS, but I choose to target Windows as 70% **of PC users are on Windows**,

so it might be more useful and relevant for me to practise to learn how to attack on the larger market (ethically of course).

- Flush Frequency
  For **performance and stealthy consideration** rather than flushing all key events to log file directly, **I choose to buffer them and flush them into a log file in batch (triggering by time, max_buffer, and "enter" keypressed).** This is to r**educe disk I/O operation** frequency that might slow down the victim's user experience and raise suspicious.
- What to capture:
  While several keylogger references I found online were trying to capture all keyboard events, I would like mine to be a bit different by **only capturing alphanumeric + symbols (which are all the characters used in password)**. I don't want to capture everything, especially as a multi-tasking person. I used shortcuts like Ctrl+Tab, Arrow Keys a lot, those shortcuts don't actually map to any character. I find no point to log it.
- Method of log delivery: APIs (Https) vs Email (SMTP)
  The keylogger I referenced was using email to deliver the logs. I find it less stealthy compared to APIs, first to send it via email, we have to create and pass our email credentials, also **email traffic is less than APIs,** delivering via APIs can better blend our keylogger trace with daily browsing https trace hence outbound email is relatively **easier to be detected by firewall.**
- How the program run:
  Over several analyses against the first MVP, in the final, I design the keylogger to **run in the background** and **add it to the Windows startup registry** so it can autolaunch on new sessions. This is to further reduce attention from the user and scanner.

High Level Flow:

Flowchart



How my keylogger works in high level overview:

1. First, install the spyware in a Windows device. Exactly how to deliver the program binary into the target device is outside of the scope of my project, but in short, a few possible ways such as:
   a. Phishing Link to download the binary
   b. Installing the spyware via USB port plug-in
   c. RCE into the victim's device and install the deliverable, etc

2. Once the binary is installed in the target device, it will run in the background. The program is also configured to be registered to the Windows startup registry so it can auto-launch itself on every login session for the target user.
3. The program running in background will filter only relevant keyboard event stroke, buffer it, and periodically flush it to a local log file.
4. Periodically the content inside the local log file will be sent as a file over API to a server configured in the program.
5. From the spy's / hacker's perspective, I can consume the log by using the API's built in.

Code Structure & High Level walkthrough:

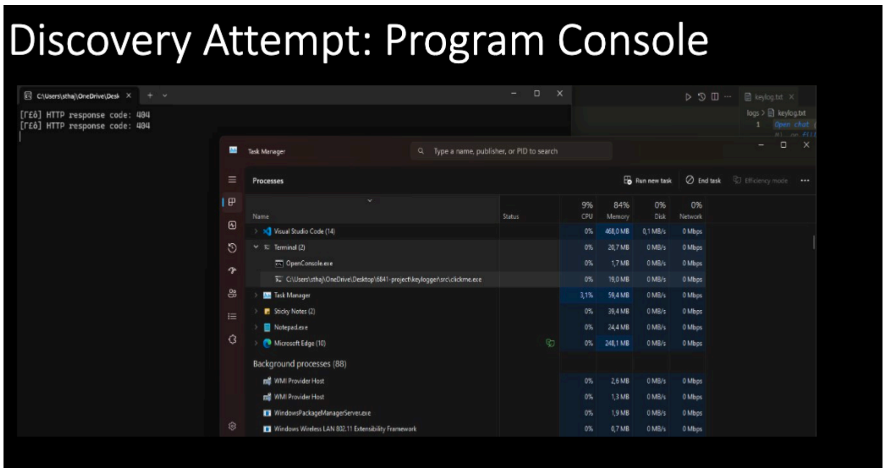
Structure	High level walkthrough
keylogger/ |── src/ | ├── main.cpp | ├── hook.cpp | ├── hook.h | ├── config.h | └── build.bat |── src/keylogger-server/ |── render.yaml	main.cpp: The entry point of the program Mainly do: 1. Install keyboard event hook `installHook` via `SetWindowsHookEx` 2. Add the program to registry `addToStartup` 3. Start a background timer to periodically send log content to the server via APIs `startSendTimer` hook.cpp: Containing logic to what to do when a keystroke even is captured in the hook callback `hookCallback`, mainly: ● Filtering ● Key mapping ● Buffering keystrokes ● Flushing into a log file (the flushing logic can be triggered based on a periodic flush timer and a maximum buffer length reach or an "enter" keyevent stroke) Config.h Containing the configurable config like the flush buffer timer, maximum buffer length, api send timer, etc

	src/keylogger-server: Everything inside this folder is related to the server that accepts the log sent from the victim's device. APIs available ● POST /upload: to upload log file ● GET /log: to list all the available log filename (named by timestamp) ● GET /download-latest-log?n=1 to download latest n log files ● DELETE /logs: to delete all log files

Hiding and Discovery Attempt:

Upon developing the process, I have had multiple perspectives and attempts in hiding and discovering the spyware. Here is the breakdown:

- Program Console



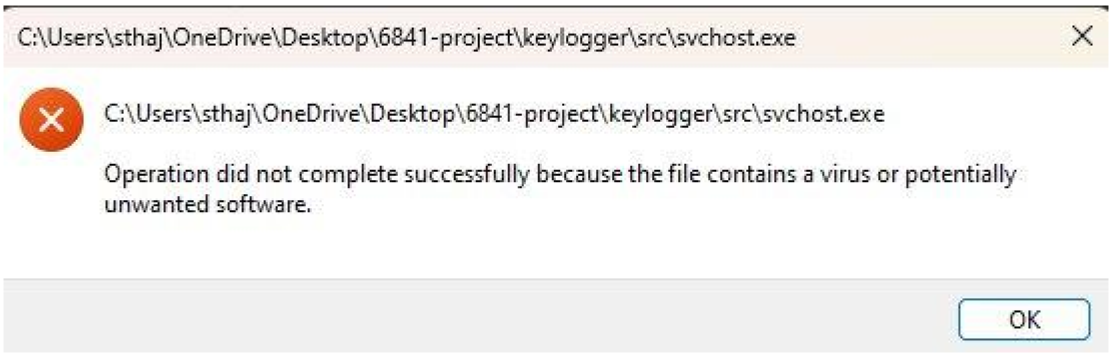
Discovery Attempt: Program Console

```
[Γ£ô] HTTP response code: 404
[Γ£ô] HTTP response code: 404
```

Name	Status	CPU 9%	Memory 84%	Disk 0%	Network 0%
Visual Studio Code (14)		0%	468,0 MB	0,1 MB/s	0 Mbps
Terminal (2)		0%	20,7 MB	0 MB/s	0 Mbps
OpenConsole.exe		0%	1,7 MB	0 MB/s	0 Mbps
C:\Users\sthaj\OneDrive\Desktop\6841-project\keylogger\src\clickme.exe		0%	19,0 MB	0 MB/s	0 Mbps
Task Manager		3,1%	59,4 MB	0 MB/s	0 Mbps
Sticky Notes (2)		0%	39,4 MB	0 MB/s	0 Mbps
Notepad.exe		0%	24,4 MB	0 MB/s	0 Mbps
Microsoft Edge (10)		0%	248,1 MB	0 MB/s	0 Mbps
Background processes (88)					
WMI Provider Host		0%	2,6 MB	0 MB/s	0 Mbps
WMI Provider Host		0%	1,3 MB	0 MB/s	0 Mbps
WindowsPackageManagerServer.exe		0%	1,9 MB	0 MB/s	0 Mbps
Windows Wireless LAN 802.11 Extensibility Framework		0%	0,7 MB	0 MB/s	0 Mbps

  - Discovery:
    On the first MVP, when my spyware is running, there is a giant pop up of the console window showing up. This is due to the program being compiled as --mconsole by default.
  - Hiding:
    The behavior can be changed using the `-mwindows` flag. This flag tells Windows that the program is running with its own GUI, but in this case, we are not building any GUI, so there is no window showing up and the program is running in the background.

- Installation:

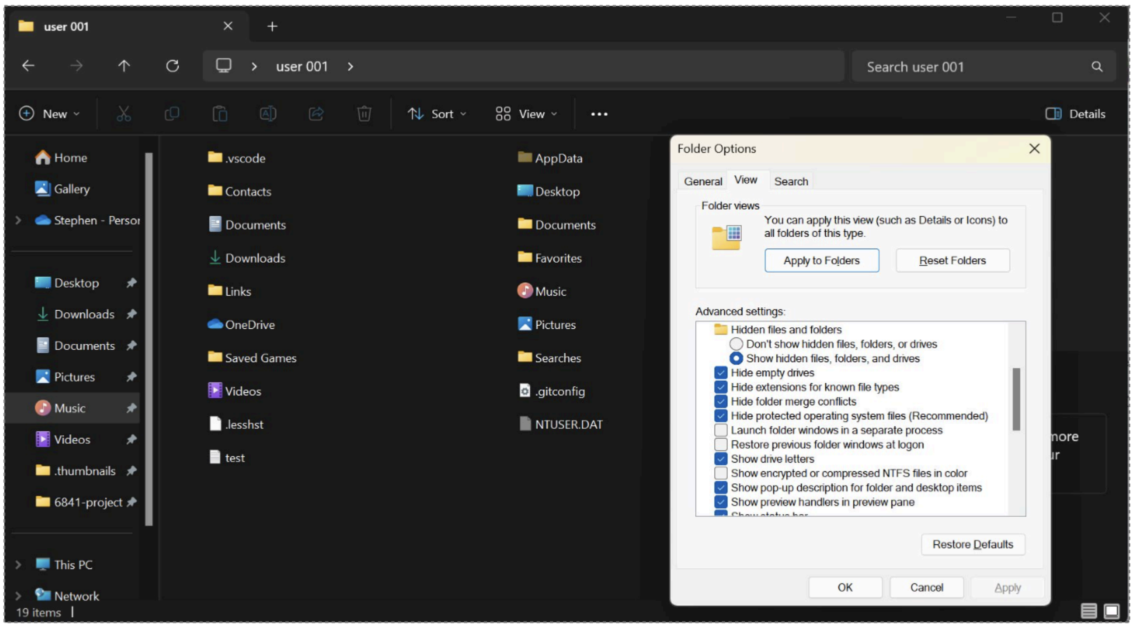

C:\Users\sthaj\OneDrive\Desktop\6841-project\keylogger\src\svchost.exe

C:\Users\sthaj\OneDrive\Desktop\6841-project\keylogger\src\svchost.exe

Operation did not complete successfully because the file contains a virus or potentially unwanted software.

- Discovery:
  Upon installation my logger binary named `svchost.exe` actually got detected as virus and installation is blocked
- Hiding:
  Svchost.exe, also known as the Service Host process, is a crucial system process in Windows that hosts and manages various services. It's a reverse psychology thinking, at first, I was thinking to name my logger binary to some .exe that predefined in Windows so my spyware can hide itself as if it's a legit program, but turns out with the predefined naming of `svchost.exe`. I end up naming my binary as logger.exe which is obvious that it's a logger but somehow not detected by Windows as a potential virus upon installation.

- Log file placement:
  - Discovery:
    Upon building for MVP, my keystroke buffers are flushing to a log file with a relative path with the binary file. Which might caught the user attention
  - Hiding:
    I tried to place the log file using absolute file path, placing the file into a relatively hidden spot inside AppData. The folder is by default hidden in Windows File Explorer UI, unless we specifically set the view to show hidden files.



.vscode
Contacts
Documents
Downloads
Links
OneDrive
Saved Games
Videos
.lesshst
test
AppData
Desktop
Documents
Favorites
Music
Pictures
Searches
.gitconfig
NTUSER.DAT

Folder Options

General | View | Search

Folder views

You can apply this view (such as Details or Icons) to all folders of this type.

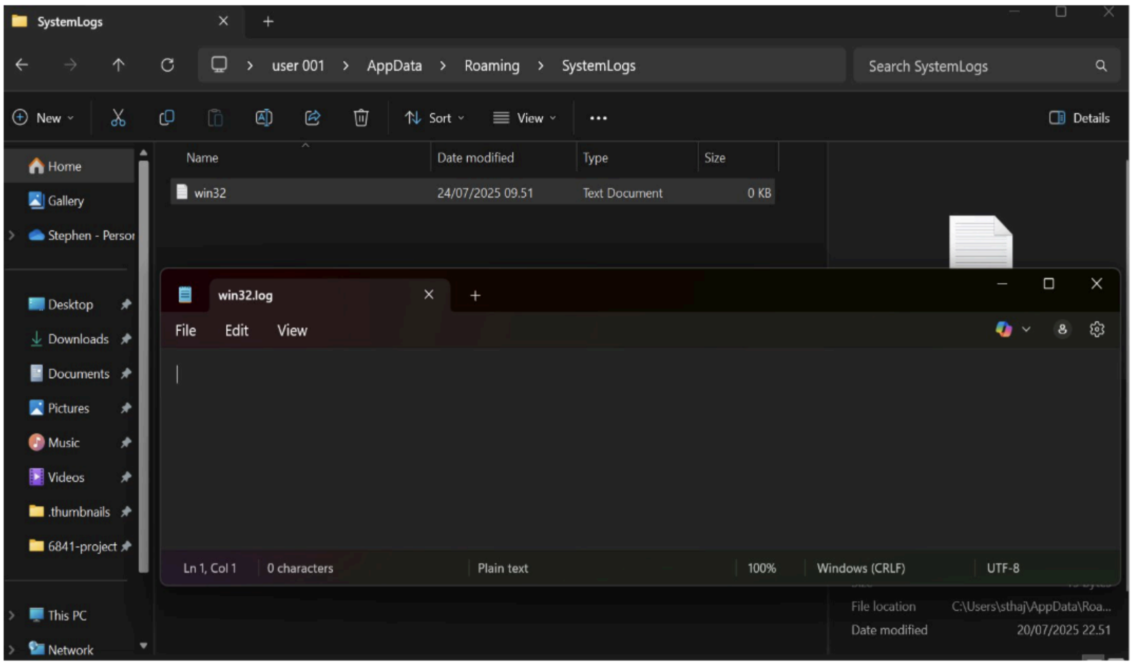
Apply to Folders | Reset Folders

Advanced settings:

- Hidden files and folders
  - ○ Don't show hidden files, folders, or drives
  - ● Show hidden files, folders, and drives
- ☑ Hide empty drives
- ☑ Hide extensions for known file types
- ☑ Hide folder merge conflicts
- ☑ Hide protected operating system files (Recommended)
- ☐ Launch folder windows in a separate process
- ☐ Restore previous folder windows at logon
- ☑ Show drive letters
- ☐ Show encrypted or compressed NTFS files in color
- ☑ Show pop-up description for folder and desktop items
- ☑ Show preview handlers in preview pane

Restore Defaults

OK | Cancel | Apply



user 001 > AppData > Roaming > SystemLogs

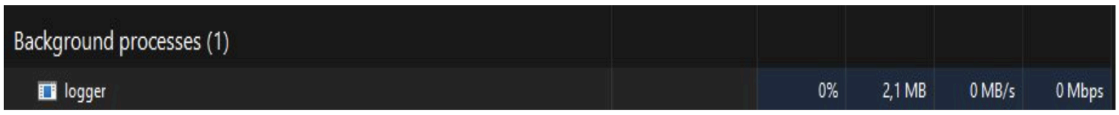
Name	Date modified	Type	Size
win32	24/07/2025 09.51	Text Document	0 KB

Ln 1, Col 1 | 0 characters | Plain text | 100% | Windows (CRLF) | UTF-8

File location C:\Users\sthaj\AppData\Roa...
Date modified 20/07/2025 22.51

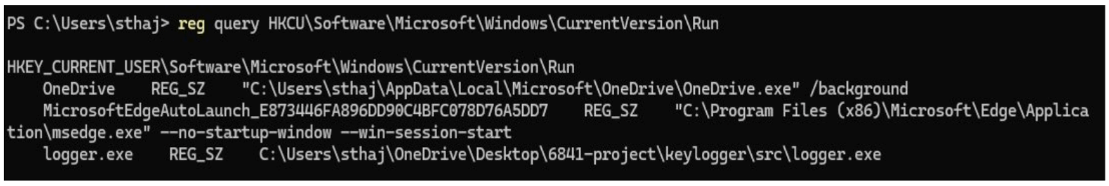
- Program Running Mode

Execute in Background without any obviously noticable window



```
Background processes (1)
    logger                                  0%    2,1 MB    0 MB/s    0 Mbps
```

Execute in Background so program can Autolaunch



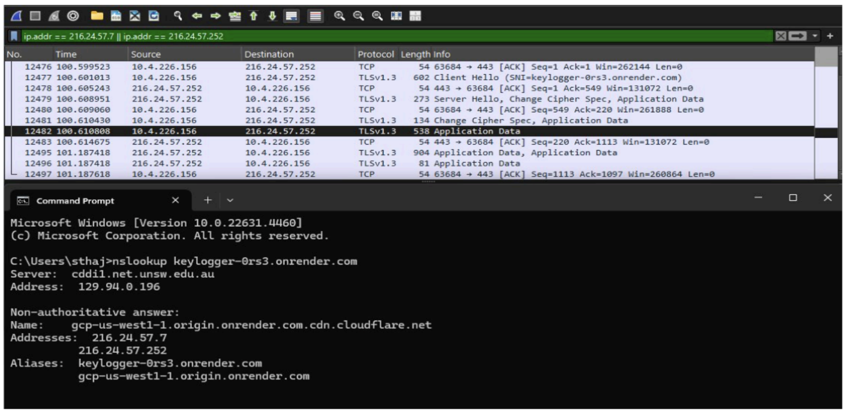
```
PS C:\Users\sthaj> reg query HKCU\Software\Microsoft\Windows\CurrentVersion\Run

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
    OneDrive    REG_SZ    "C:\Users\sthaj\AppData\Local\Microsoft\OneDrive\OneDrive.exe" /background
    MicrosoftEdgeAutoLaunch_E873446FA896DD90C4BFC078D76A5DD7    REG_SZ    "C:\Program Files (x86)\Microsoft\Edge\Applica
tion\msedge.exe" --no-startup-window --win-session-start
    logger.exe    REG_SZ    C:\Users\sthaj\OneDrive\Desktop\6841-project\keylogger\src\logger.exe
```

  - Discovery:
    As mentioned above, at first my MVP spyware had a console pop up upon running.
  - Hiding:
    I made the program run in background and auto add it into windows startup registry, so everytime the user start a Windows session, the program will auto-launched

- Additional Discovery Attempt:

- Windows Defender Scanning

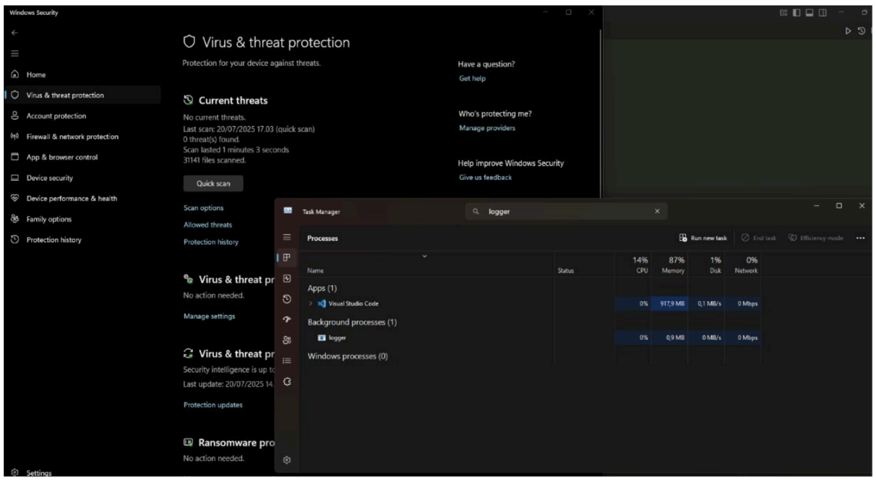


```
ip.addr == 216.24.57.7 || ip.addr == 216.24.57.252
No.   Time       Source          Destination     Protocol Length Info
12476 100.599523 10.4.226.156    216.24.57.252   TCP        54 63684 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
12477 100.601013 10.4.226.156    216.24.57.252   TLSv1.3   602 Client Hello (SNI=keylogger-0rs3.onrender.com)
12478 100.605243 216.24.57.252   10.4.226.156    TCP        54 443 → 63684 [ACK] Seq=1 Ack=549 Win=131072 Len=0
12479 100.608951 216.24.57.252   10.4.226.156    TLSv1.3   273 Server Hello, Change Cipher Spec, Application Data
12480 100.609060 10.4.226.156    216.24.57.252   TCP        54 63684 → 443 [ACK] Seq=549 Ack=220 Win=261888 Len=0
12481 100.610430 10.4.226.156    216.24.57.252   TLSv1.3   134 Change Cipher Spec, Application Data
12482 100.610808 10.4.226.156    216.24.57.252   TLSv1.3   538 Application Data
12483 100.614675 216.24.57.252   10.4.226.156    TCP        54 443 → 63684 [ACK] Seq=220 Ack=1113 Win=131072 Len=0
12495 101.187418 216.24.57.252   10.4.226.156    TLSv1.3   904 Application Data, Application Data
12496 101.187418 216.24.57.252   10.4.226.156    TLSv1.3    81 Application Data
12497 101.187618 10.4.226.156    216.24.57.252   TCP        54 63684 → 443 [ACK] Seq=1113 Ack=1097 Win=260864 Len=0
```

```
Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sthaj>nslookup keylogger-0rs3.onrender.com
Server:  cddi1.net.unsw.edu.au
Address:  129.94.0.196

Non-authoritative answer:
Name:    gcp-us-west1-1.origin.onrender.com.cdn.cloudflare.net
Addresses:  216.24.57.7
          216.24.57.252
Aliases:  keylogger-0rs3.onrender.com
          gcp-us-west1-1.origin.onrender.com
```



Virus & threat protection
Protection for your device against threats.

Current threats
No current threats.
Last scan: 20/07/2025 17.03 (quick scan)
0 threat(s) found.
Scan lasted 1 minutes 3 seconds
31141 files scanned.

Virus & threat pr
No action needed.

Virus & threat pr
Security intelligence is up to
Last update: 20/07/2025 14.

Ransomware pro
No action needed.

Task Manager — logger

Name	Status	14% CPU	87% Memory	1% Disk	0% Network
Apps (1)					
Visual Studio Code		0%	917,9 MB	0,1 MB/s	0 Mbps
Background processes (1)					
logger		0%	0,9 MB	0 MB/s	0 Mbps
Windows processes (0)					

With the spyware actively running in the background, Windows Defender did not report with anything suspicious.

- Wireshark packet tracing
  At first, I was thinking the point where the spyware sends the local log content from the victim's device to the keylogger server can leave some evidence or trace via the network HTTP payload. But after some time searching the trace, I just remembered my spyware was sending the log content via HTTPS, which means the payload is actually encrypted and not viewable by wireshark or any network tracing tools.

How to test:

1. Run the build.bat to compile the binary on the Windows based device
2. Install the binary on the device, search for logger.exe to make sure it is running.
3. Try typing some stuff in parallel and open the log file in `C:\Users\%USERNAME%\AppData\Roaming\SystemLogs\win32.log` to see what keystroke is flush into the file.
4. Wait for sometime, to let the file content be sent over the network to the keylogger server.
5. Download the latest n log file from the server via browser or any other user agent. keylogger-0rs3.onrender.com/download-latest-log?n=1

Reflections:

- While most of the viruses that attack confidentiality can often be defended cryptographically (encryption), however it doesn't work for keylogger spyware as the keystroke is recorded in the victim's device not upon transiting over the network. Though there will be a few signs that your device is infected with keylogger spyware:
  - A slow browser
  - A lag in mouse movements and keystrokes
  - A disappearing cursor
- Upon working on this assignment, I feel like both red-team and blue-team need another knowledge and perspective of thinking to think about the attack vector, knowing how a particular attack works and how stealthy it can be.

References:

- OS Statistics overview
- Keylogger Reference: Ref1, Ref2, Ref3
- Windows API: Hook , Key Capture , Virtual Code
- Keylogger detection sign