Project Result

- 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.

What I did

Analysis & 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.

Developing keylogger in C++:

Key features of keylogger:

- Able to capture keylogs on target devices (Windows OS)
  - Filtering: alphanumeric + symbols characters (to reduce noise key events)
  - Periodic flush and delivery: Log captured are flush to a file periodically in batch (by storing keyevents into buffer) and periodically sent over network to keylogger-server via API.
- Keylogger can auto launch on each windows login session (process register in windows startup registry) and run in background.
- Deliver the keylogs over API to a hosted server which cover API endpoint to
  - Upload logs (used by the keylogger app to upload the local log file from victim)
  - Log Management (list, download, and delete)
- Keylogger can be installed on Windows devices without any security warning & running on Windows devices without detected by Windows Defender Scan

Manual Testing & Tool Scanning:

- Testing on an Asus 64 bit x64-based processor. Native Windows11 Home V 23H2 (native windows and not VM)
- Analyze Windows Task Manager's process, Windows Defender scan result, Wireshark log trace

Challenges:

- Setup UTM VM on MacOS doesn't work. After several consideration, finally switch to Native Windows
- Unfamiliarity with Windows API & OS
- C++ language fluency on
- Aside from building the keylogger C++ app, I also have to build a simple server to validate the keylogger MVP.

Detail Project Timeline Carried Out (~38h in total).
Note: Time estimation is rough estimation referencing Github commit timestamp

Week	Activity
4	• Setup (~5h) I am a main MacOs user, since I am targeting keyloggers on Windows OS, at first I try to set up a VM on UTM. However after some installation failure and thought about potential different behavior running on VM, I decided to develop and deliver it on native Windows device. • Explore and study reference (~2hours) Referencing some existing keylogger github repo and youtube video.
5	• Ideation (~2h) Pick several features that I want my keylogger to have: • Start coding basic keylogger functionality (~4h) ○ MVP done in week 5, successfully capturing key events into a file
6	• Feature Improvement + Testing (~10h) Add feature to deliver the logs to a localhost server utilizing ngrox public dns service
7	• Deployment + Continuing Feature Improvement + Testing (~6h) Rolling out ngrox and hosting the keylogger server on remote server & continuing testing. • Evaluate & enhance keylogger from both Blue team perspectives. Discovery Attempt (~2h): ○ Windows Task Manager Analyze ○ Conducting Windows Defender Virus Scan ○ Wireshark trace quick analyze • Red Team's Perspective Improvement: Hiding + Stealth Improve (~3h): ○ Less obvious log file placement in target device ○ Remove installation prevention prompt from Windows ○ Let the program run on background ○ Auto launch application (put it into start registry)
8	• Write report, slide (~5h)

Reference:

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