UNIVERSITY OF PLYMOUTH & HKU SPACE

BSc (Hons) Computer and Information Security

PRCO304HK Computing Project

Develop an Anti-Keylogger Program (AKP) for Detection and Deletion of Keyloggers in Computer System

UK Student Ref.: 10706836

KHU SPACE Student ID: 10072329

UK Supervisor: Dr. Hai-Van Dang

HK Supervisor: Dr. Beta Yip

Acknowledgements

Abstract

This report describes the development of a program to detect and delete the Keyloggers within the computer system. IT security professionals would undertake when a cyber-security incident occurs in the aspect of Keyloggers as well as provide additional security for organizations through specific monitoring and response.

The report begins with introduction into the current state of cyber security issue related to Keyloggers and then progresses onto the aims, goals, objectives and deliverables of the project. Legal, social, ethical and professional issues that may arise because of this project are then highlighted before moving onto the main body of the report.

The main body of the report divided into several phases, followed by some evaluations which described my Anti-Keyloggers Program (AKP). This includes background aims and objectives development, project approach and methods, architecture and design of program, as well as the development of the Keyloggers detections and deletions that uses several technologies including program language C++, C# and SQLite3 database which Issued and challenged that arose throughout development are also discussed to highlight how they were overcome.

The report then progresses into the critical evaluation of the project and a full post-evaluation which highlights the overall state of the final product, a technology spectrum and personal review as well as future work that will further enhance the functionality of the Keyloggers detection and deletion.

Further information included in the form of appendices can be found at the end of this report which constitutes other materials generated over the course of the project such as highlight reports, project schedule and the project initiation document.

Content:

1. Statement of Word Count

Word Count: 7947 words (Citations and References were excluded)

2. Code Submission URL

GitHub Repository:

https://github.com/wintersyau/PRCO304_2122_YAUCHAKMAN.git

Plymouth OneDrive File Space:

https://liveplymouthac-
my.sharepoint.com/:f:/r/personal/chak_yau_students_plymouth_ac_uk/Documents/PRCO30
4%20Computing%20Project?csf=1&web=1&e=gc8PXh

Trello:

https://trello.com/invite/b/RuN4y8Yw/932ef72b556c26b6fd4455517dee2250/fyp-anti-
keylogger-program

Video presentation of the product

https://youtu.be/V6oYhfO6tnA

3. Introduction

As we know, Keylogger is a program that enable to monitor and records every activities of the computer user by their typing on specific keyboard of a computer or a mobile device (Bhardwaj & Goundar, 2020).   The software tracks or logs on the keys without the knowledge of the user.   As the result, it may lead to a great threat to user to leak of the important data while they typing on the keypads.   Most of the computer users are laymen users who will not realize the present of the Keylogger and its functions, also they will not have such behavior to remove the important Keylogger when they using of the computer (Kaspersky., 29 MAR 2007). The threat is not only to retrieve the data, but also intercept passwords and other confidential information entered through the keyboards.   The hackers could steal the PIN codes, bank account numbers, passwords to emails and social networking account credentials. The final results to the loss of the users' property and money from close relatives (Kaspersky., 29 JUN 2011).

Although there have some paid or free anti-free Keylogger program, but we don't know the program is safety or not if there is a back door of the program which will leading to lost the data security.   Design of the program to detect and delete the Keyloggers found in the computer system would completely understand what the Keylogger done and how to avoid it in stealing the data (Bhardwaj & Goundar, 2020).

4. Background

In the recent decades, there were many criminal cases related to the use of the Keyloggers.　In the year of 2016, there were a large-scale cyber-attack by using the "Hawkeye Keyloggers", until the 2018, the updated version of "HawkEye Keyloggers" was sold to the criminals for invading the target computer systems (Bisson, 2022).　The Keyloggers functioned by injecting the malware into the programs like MSBuild.exe、RegAsm.exe、VBC.exe in the computer OS, and execute the payload code as it wishes.　Such cyber-attacks brought large amount of money and property lost as the results. Business transactions and personal activities such as E-commerce, online banking, email chatting, and system database were all affected by the Keylogging activities (Bhardwaj & Goundar, 2020).　In the personal activity aspect, keylogging activities would result the leakage of the personal information like names, age, gender, height, weight, and personal account's profile etc., such information retrieved by criminals would sold to other companies for product promotions, invasion of the account and made illegal transactions. Therefore, Anti-Keyloggers became an important issue in preventing those cyber-attacks.

In the market, there were many Anti-Keyloggers products like NORTON 360, Bitdefender, OPSWAT and SpyShelter (Kaspersky., 29 JUN 2011).　Those products provided the functions of Keylogger detection and deletions as well as alert to user for those incoming software (Bhardwaj & Goundar, 2020).　However, we do not know if there were backdoor of those program resulted to the information leakage, and also the concerns of the license of the Anti-Keylogger with copyright of the authors.　In addition, some people adapted the used of the Anti-Virus programs without Copyrights that may even leading to a cyber-security hazards. Therefore, the safest way to prevent the Anti-keylogger is to make our own one.　It can prevent the backdoor of the program and no need to have copyright concern.　Also, develop an Anti-Keyloggers program did raise the public concerns on cyber security control (Kaspersky., 29 MAR 2007).

5. Aim and Objectives

The aim of this report is through developing an Anti- Keylogger Program to understand its underlying mechanisms, and to raise the concerns by people to alert them in dealing with the keylogging activities, and to perform a good cyber security control to prevent the attack by hackers.

In this project, I would try to write up the software program by using the computer languages for Keylogger detection and deletion. The target OS of my project would be Microsoft Windows 10 & 11 64-bit system. The reason of choosing Windows as my target OS because it is a very common OS which mostly used by computer users and me. The second reason is Windows OS may easily attack by hackers by using Keyloggers when compare with other OS.

However, further research has provided influence resulting in altered and newly derived objectives:

5.1 To write up a Keylogger program to understand the mechanism of keylogger in the computer system.

5.2 To write up an Anti-Keyloggers program to detect and delete the Keyloggers in the computer system.

5.3 To analysis the Keyloggers characteristics and stored the data in the database

5.4 To delete the Keyloggers detected by using the program

5.5 To test the program and validate the program

In addition, the core deliverables would be listed as follows:

5.6 To have a client application
- The application should be broken down into a collective set of modules that execute specific tasks: detection and deletion of Keyloggers functions
- The application should aid the prevention data exfiltration
- The application must use a strong encryption module
- The application should have real-time monitoring modules
- The application should have front-end modules

5.7 To have a database
- The database should have standard security controls and it must host a database in a secure manner.
- The database should have a database management tool
- In this project, SQLite3 was selected for the database development because of its easier to use and well known by users.

6. Project Approach and Methods

In order to understand the functions of keylogger and develop the program focusing on it, it is needed to understand the background and the usage of the Keylogger in operation system.

6.1 How keylogger works in window OS

How does spyware keylogging work?   It is the basic question when programmer try to develop the program.   The easiest way for keylogging work is the Ring3 keyboard Hook. The Ring3 is the privilege level of the processor, it can describe the permissions of the system. The Ring3 and some privilege escalation performed by some software in Ring3 is called system. It would be done by changing the machine's codes to implement the keyboard Hook.   A more advanced way is the kernal-level ioapic/idt hook.   It is real-time feedback of the keyboard stroke or information through Hook.   The common types of Hook include Inline, IAT or SEH and the common way of Hook is the most basic HookEX, and the more difficult API Hook or HookSSDT.   What I want to focus on the Keylogger program is the API Hook.

6.2 API Hook

The API Hook is provided by Microsoft under window, it was belonging to low-level operation function for window's operations.   There are four syntax in SetWindowsHookExA with different parameters.   By using the parameters of these syntax in order to monitor the low-level keyboard inputs in order to write up the keylogger programme.

6.3 Program Language selection

As we all know that in the language C#, it is able to deletes a file by saying "File.Delete", therefore, under Window OS, it corresponds to DeleteFileA, the Kernel32.dll which provided export function that we can use DeleteFileA to delete the desired file directly by using this code.   But how about the function if the Window is not up to date? Is it unable to run the software if the OS is out - dated?   Luckily, the programming languages provide a layer of encapsulation on top of this as long as File.Delete is normal no matter what platform or version of the system it is on just delete the file.   That's the reason to use File.Delete instead of using the system function directly and the program language C# is selected to write the program.

6.4 Project Approach

In order to write up a program that prevents the deletion of files on the OS, I would like to Hook DeleteFileA to make a program that prevents the deletion of files on my computer.   I would like to install the Hook and write a function with the same parameters and return value as DeleteFileA in my DLL (Dynamic-link library).   Then I would like to inject to the target program that I want to Hook (DLL injection technique).   As there are many API hooking

methods, I prefer to use Write Process Memory function that directly writes the DLL to the target program memory to decompile the other party's DLL and insert it. That means I would like to use a piece of own codes and replace it that no matter what method to use to inject, the final goal is the same, i.e. replacing the original with a fake one well-understood term.

Then, when the program deletes the file, no matter what language it is written in, it will call DeleteFileA when being deletes it. Under normal circumstances, DeleteFileA is normally deleted and returned. Thus, if I replace DeleteFileA with mine, it will naturally execute mine. I can judge whether it is deleting through the path of FileName passed by and what's in our important folder if it doesn't delete what I don't want him to delete. I am calling the original DeleteFileA to execute it back so that it will not report an error, the purpose is to keep the system from crashing. But what it feels is that it deleted the file successfully but in fact, it did not execute DeleteFileA and was skipped by me.

6.5 Project Methods

In order to injecting the DLL's in system, the most important to remind is that the DLL code is executed by the system itself, but not my program. That means the DLL injected is equivalent to the internal of others. So we just need to hook it off. SetWindowsPos will do, forbid it to create threads Hook off CreateThreadA. In my program, there are 3 methods to implement the objectives:

Method 1

The method 1 is the kernel-level Hook is directly hooked under the hardware layer and the driver layer. The disadvantage is several times better than method 2 and method 3, but if others are also kernel-level programs and they are loaded before the program, they can still pass the hook kernel. Mount the driver entry Learn the driver I want to load and analyze what are doing.

Method 2

Method 2 is the more commonly used method. It is done by DLL Inject secretly into the specified program to read its memory or use the control UI Automation (non-hook to read information). The disadvantages for some high-level and high-level processes, memory injection may be failed. Only passive injection can solve the problem and the cost is very high.

Method 3

Method 3 or use the simplest method SetWindowsEx to install an Exhook hook on WH_KeyBoard. The disadvantage is it is very, very easy to roll over.

In short conclude, for the method 1, it must be the most brilliant and the least easy to find, and the efficiency is also the best. Method 2 is also possible, but the automatic test control is based on the face, and some text boxes are special and cannot get the value. Method 3 is very simple, very effective and hassle-free, but the biggest problem of 3 is that SetWindowsEx can set multiple hooks at the same time. This means there may be hooks on yours... if method 1 and 2 are not easy to find then method 3 is the easiest to find because the intention is too obvious, and the system function SetWindowsEx is the easiest to roll over. That is to say, as long as we get the behavior of WH_KeyBoard, we will immediately terminate its main program and notify the user, then we have to make sure our termination is 100% high intensity. It will terminate the process with PspTerminateProcessById at the kernel level instead of the KillProcess of Ring3 (The user layer). The kernel-level process protection mechanism ObRegisterCallbacks would be used when receiving any operation request or access request, i.e.,

  OperationInformation->Parameters->CreateHandleInformation.DesiredAccess=0;   which abusing our permissions in the kernel and resolve no one to access the program, then ending the program which needs the address of the process to end, but of course it can't even access us. It won't end us, but it will also be killed by the end of the kernel-level process. This is the same as Ring3 and Ring0 class relationship.

6.6 Methods for hook detection
Method A

The principle is to scan the DLL file called by the process and then compare the original DLL file. The behavior of hook is to rewrite the function, so when I change the system function, it will definitely be different from my own system function, so this is just to prevent DLL injection. In order to know which programs were violated. It could be done by time judgement. A queue would be made first and join all processes, and then keep traversing the process queue and keep comparing the changes of their system functions. Every time a new process is found, it will be added to the queue. Therefore, one of the process system functions of the original process has changed, and the changed function is related to keyboard input. Thus, it must be one of the newly added processes (there may be more than one in a short period of time and this situation is rare and mostly does not happen as long as the scan frequency is fast) which modified the program that was changed.

Then at this time, we will first judge whether these new processes already have digital signatures, add 10 to the score, and then judge the process which DLLs are loaded. After determine what language the process is written in by calling the DLLs deduct points for some special languages such as easy Language, then we judge whether the software has a

connection with the Internet, and if so, the score continues to decrease.   If there are multiple or sort from low to high at the end, remind to record the user's operation to the database. The rating value showing to the user what software might be monitoring your keyboard for the end.   If it is only one, that means 100% sure, end directly and enter the record to the database.

Method B

It will be done firstly by putting 64-bit SSDT Hook off this SetWindowsEx and add a springboard.   Then DebugWinEx if hook action is WH_KeyBoard in debugwinex like ring0 layer.   That means, the software layer to write sends a message and then waits in a loop until a return is received.   The software pops up a prompt box, the process name is calling the SetWindowsEx function, and the action of the keyboard hook is executed that blocking return a message to the driver after the operation is completed.   The Kernel-level process protection Kernel-level process termination as the sharpest shield and spear of the program plus the mixed detection of the two modes.   The security software to ensure privacy is developed using C# Wpf And C/C++ mixed development both at the bottom and on the interface must be different.

Then the keyboard monitoring of the driver layer is actually not enough for most anti-software main defenses.   The reason why they can protect them is because they run before kernel-level virus software so they are hooked into loading drivers in advance.   The analyze the behavior of the mounted driver through whitelisting and memory disassembly as well as behavior analysis and Trojan library signatures.   But once when we put its driver in the past. It is not work, so the driver-level response is relatively low. Most of them may rely on passive scanning after file addition, memory scanning in process creation, and passive scanning of mounted drivers to judge whether the process is risky, but we certainly can't analyze the security of the software and find something wrong and kill it.   This is the developmental status and the introduction of the functions to be implemented in the future and the explanation of the ideas.

7. Project Management

The successful delivery of projects mainly relies on project management to ensure deadlines and objectives are constantly being monitored and upheld.   The initial approach taken was the creation of Trello.   The reason to use Trello as scrum master board because of its well organized with time frame and the events.   And it also enables the function of backlog, the work-in-progress column, the validate and complete column which enable for clear understanding of the project progress and what were not done yet.   The Trello timeline figure of my project as showed in the Figure 1.

Figure 1. Trello Timeline Backlog
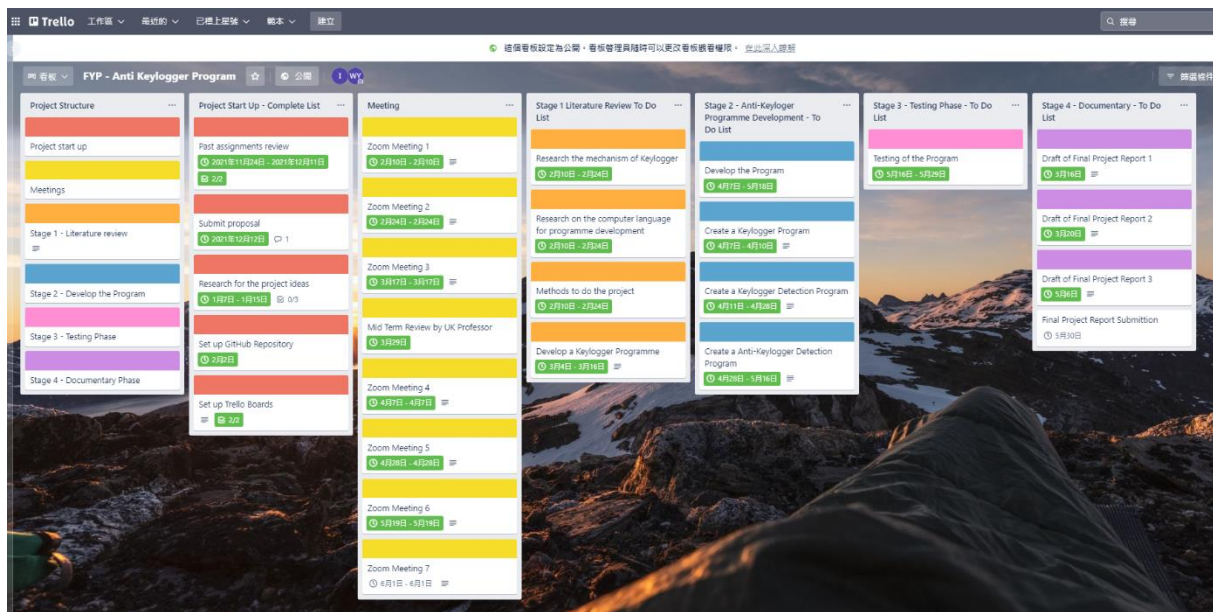
The use of Trello has been an important part to complete this project as I was able to create a variety of tasks and thoughts in each of the stages and define the start and completion time frame.   Once a task was achieved, this was then moved to the completed section which provided an accurate representation of the project's progression.   The complemented of the project progress by Trello provided a range of statistics that offered a sense of satisfaction and on-going encouragement during the project development.   The usage of a project planner is critical to ensuring deadlines are being met, with constant monitoring, tracking and updating of the plan.

The periodic progress review meetings and e-communications with the project supervisors were also an integral part of the project management and successful delivery as improvements to specific aspects of the project were raised and subsequently implemented and resolved.   Moreover, report highlights were also used to provide additional information to the supervisor and continuously planning of the next stage of work for the following week would be agreed and defined within the document.

Since there was a 5th wave of COVID-19 pandemic in HK during the period of project development (from Feb 2022 to May 2022 which exactly the period I conducted the project), the risk mentioned in the project set up became truth and it had a great impact to my working progress.   Luckily, my colleagues gave me lots of support of my part of duties and works, together with the great support by my HKU SPACE professor, of his treasure advices to my project which resulted to the achievement of my project.

8. Legal, Social, Ethical and Professional Issues

Legal, social, ethical and professional issues were constantly being considered throughout the development of this project as Keylogger concept as well as the Anti-Keylogger program heavily relies on collection, storage, and analysis of the information (the Key logs).

8.1 Legal

One of the main legal aspects that must be considered is the General Data Protection Regulation.   In UK, the Data Protection laws mainly stated in the Data Protection Act 2018, which is a well structural and organized law enforced in UK.   The General Data Protection Regulation (GDPR) mentioned in the Data Protection Act 2018 applies to both data controller and data processors who process the data, and the data processor responsible for managing and directly handling the data specified by the controller.   It applies to any piece of information that directly or indirectly relates to an identifiable person which could even be a reference to an identifier.   The reason why this is particularly important and is directly related to this project is because the Anti-Keyloggers Program (AKP) would retrieve and assess one's keystrokes and provide feedback and react to the next steps which is to delete the Keyloggers detected.

In this project, the main legal considerations would be ensured the data process within the program will not be leak to other third party.   It would be achieved by create own keylogger program and Anti-keylogger program that prevents the backdoor from other open source.

In addition, another legal concern is the Copyrights of Keylogger program developer when I take some of the Keylogger products for reference during my product development. (The Copyright, Designs and Patents Act, mentioned by UK Government, 1988).   In order to prevent the copy rights violation, I would like to write up my own program and the program development is only used for the study purpose.

8.2 Social

The main social concern of the project is mainly the risk of computer data misuse resulting to the risk of cyber security crimes incidents. The database of the Anti Keylogger Program may have the risk of data leakage when there are third party to assess the database without permission or acknowledgement. In my own opinion, the most social benefit by developing the Anti-keylogger program is to raise people concern on Keylogging activities, especially there are increasing online transactions and online banking which leading to the great risk.

8.3 Ethical

The main ethical concerns in this project was the utilization of the retrieved information by using Keylogger, i.e. the Keyloggers information. It is a serious ethical issue in relation to this project because there is risk of the programmer to sell the stored data in the database may lead to another cyber security hazard that criminals will make use of the information to steal the money from bank accounts or other important and personal data if the passwords were being sold by the programmer. This would not only be unethical, but also be the problem of legal aspect for data processing regarding to the Anti Keylogger Program which could include hefty fines as well as significant reputational damage.
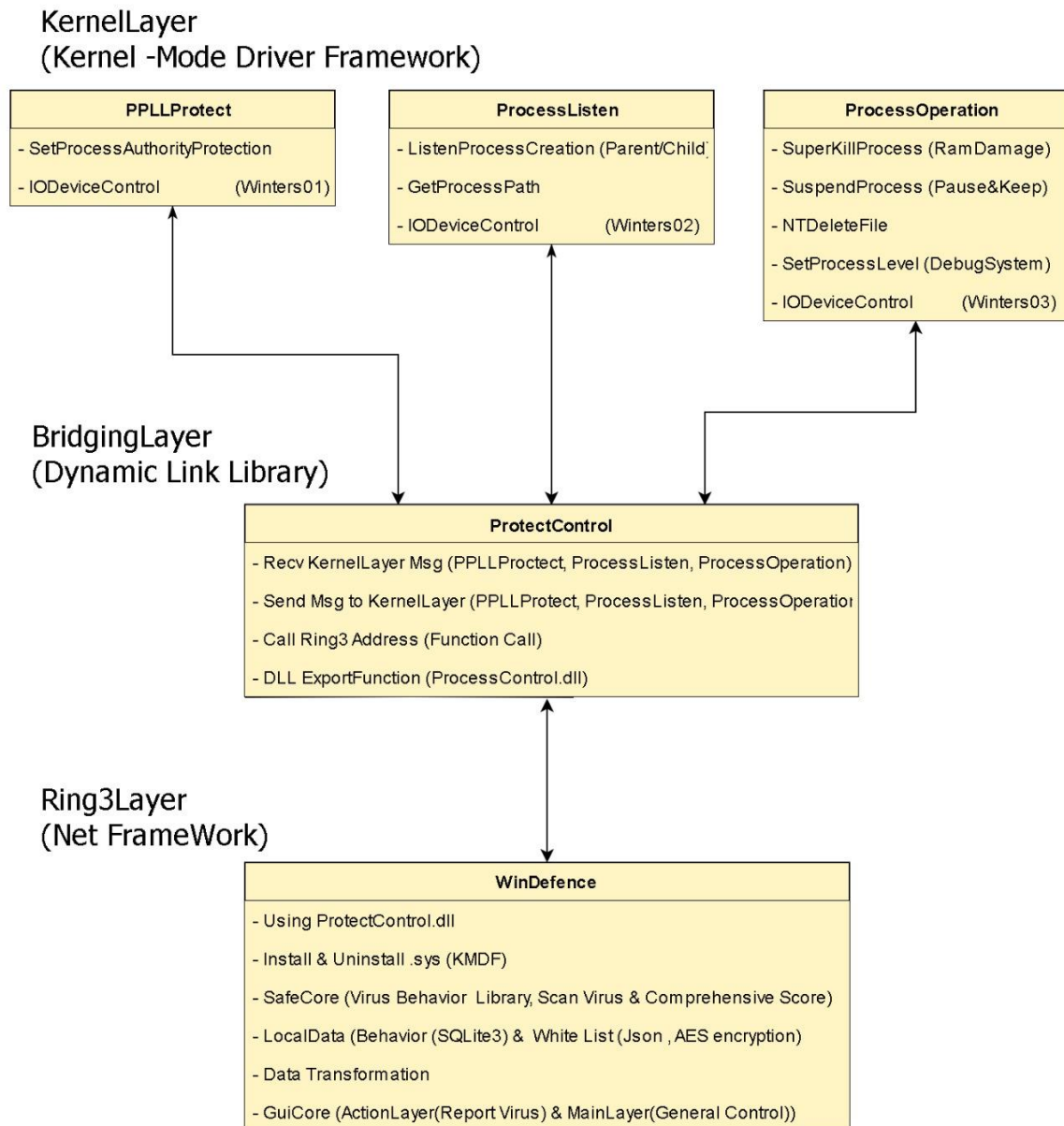
8.4 Professional

The professional aspect of this project is the own development of the program with own database. The limited using of the Open source information also prevents the backdoor of the source's program. In this project, I assume that all data processors were honest and user of the applications would use the program properly to prevent failed running of the program.

9. Architecture & Design

9.1 The UML Diagram of the Anti-Keyloggers Program (AKP):

Anti-Keylogger Program (AKP)

KernelLayer
(Kernel -Mode Driver Framework)

PPLLProtect
- SetProcessAuthorityProtection
- IODeviceControl   (Winters01)

ProcessListen
- ListenProcessCreation (Parent/Child)
- GetProcessPath
- IODeviceControl   (Winters02)

ProcessOperation
- SuperKillProcess (RamDamage)
- SuspendProcess (Pause&Keep)
- NTDeleteFile
- SetProcessLevel (DebugSystem)
- IODeviceControl   (Winters03)

BridgingLayer
(Dynamic Link Library)

ProtectControl
- Recv KernelLayer Msg (PPLLProctect, ProcessListen, ProcessOperation)
- Send Msg to KernelLayer (PPLLProtect, ProcessListen, ProcessOperation)
- Call Ring3 Address (Function Call)
- DLL ExportFunction (ProcessControl.dll)

Ring3Layer
(Net FrameWork)

WinDefence
- Using ProtectControl.dll
- Install & Uninstall .sys (KMDF)
- SafeCore (Virus Behavior Library, Scan Virus & Comprehensive Score)
- LocalData (Behavior (SQLite3) & White List (Json , AES encryption)
- Data Transformation
- GuiCore (ActionLayer(Report Virus) & MainLayer(General Control))

9.2 A real-time detection mechanism

ProcessListen.sys process monitoring is able to get the newly created process, and the target process executes the process and get the path where the process is located in the kernel. It would pass it to the ProtectControl.dll which eventually takes the data and sent to the Ring3 layer of the main program (WinDefense.exe). The data is encapsulated and passed to SafeCore to export the objective function and evaluate the risk score.

9.3 Comprehensive Scoring Mechanism

According to the Common Vulnerabilities and Exposures (CVE) (2022) and National Vulnerabilities Database (2022), most of the Keyloggers behaviors were studied in the database academically, I would like to adapt the exposures details for scoring calculations of the program which contained higher score would have defined as "Malware". Language runs on the Win platform its underlying layer is WinApi, any operation could not be separated from WinApi. By analysis of WinApi, we could get the main behavior of a program concretely. This procedure is added by the degree of risk of possible behavior of being invaded.

Whatever the program is a system process or not, it has a digital signature (with a defined signature score of -5, no signature score +5) which reduced of the score for this program, i.e.,

"-5 → SAFE; 0 → UNKNOWN; > 30 → NOTEWORTHY; >= 45 → JUDGED TO BE A VIRUS"

The higher the score, the dangerous it means and the higher of the risk of the program, and the program with a score greater than 45 will be blocked by the program. The behavior libraries were stored in an encrypted Sqlite3. By reading the target program by specifying path, it was able to have comprehensive analysis of the program behavior after export the function.

Why it was set to formulate the threat score to 45? It was because the program aimed to focus on removing the Keyloggers with reference of the Keylogger's behavior libraries, and the Keylogger needs to receive user keyboard input continuously, the two behaviors of SetWindowsHookEx and CallNextHookEx are very obvious that their combined total threatened score is 36:

SetWindowsHookEx → installs the Global Hook, threatened score was 30
CallNextHookEx → the next hook after the first hook received the event, threatened score was 6
CreateRemoteThread → far thread Dll Injection, threatened score was 11

The above were the examples of three behavior libraries, in the program, since Keyloggers were types of back-end software which need to send keyboard messages to the server in real time, thus it needed Socket to achieve its functions:

Socket, threatened score was 8
Background Program, threatened score was 7 (any process which was not created by the user)

As a result, the real Keylogger certainly consist of these characteristics, i.e., SetWindowsHookEx, CallNextHookEx, WinSock and Socket, the total threatened score would be:

30 + 6 + 7 + 8 = 51, that was the hazard score for unsigned files without Socket;
30 + 6 + 7 = 43 + 5 = 48, that was unsigned signature which meet the interception criteria;
30 + 6 + 7 = 43 – 5 = 38, that was signed and scored in warning but not intercepted.

For the programs which were networked and consisted of global hooks would be blocked whether they were signed or not: (TotalScore) + (-/+)5 +8

9.4 Active process scanning
When the user clicks "Scan" in the client's application, it will iterate all the current process, if there was a progress it would score it and summarized on the scoring page.

9.5 Main interface presentation
The program would summarize the current detection messages and wrote up the score value of the currently established process into the "ListView".   After the threats was discovered, it would prompt to find the threat and prompted that the client was protected by the top-up message.   After the user checked with the virus and clicks the "Agree" button, the risk prompt will be changed to "safe" again.

9.6 Reason to choose hooks at the kernel layer
In order to remove malicious programs and protect of the system with sufficient permissions, it was not enough by acting on the Ring3 layer.   The most important is the interception process. In the Ring0 layer, the execution of the PsSetCreateProcessNotifyRoutine allowed to create a monitoring method to the creation of all processes within the Win System. It is efficient and will be executed before the malware code executes. It also would deliver messages to our main program to alert the user. Working in the kernel layer not only enable to remove the main virus program, but also to ensure that the virus program was suspended in time (kernel level) before it was opened by the user before the code is executed.

9.7 Problems Encountered
After the PPLL implements the process protection, the uninstall driver did not stop the protection leading to the antivirus software cannot exit properly.   The solution is really effective that in the main thread of the program to make one error which forced the program to crash.   The reason to make an error is that SuperKill ZeroMemory needed to be attached to the process in order to empty the process's memory, but the PPLL layer took into account

the need to prevent OD disassembly.   So, denying access and causing the kernel-level process to end directly turned into a kernel-level "time bomb" in the kernel which created an error turned out to be a blue screen on the computer.

10. Project Development

For the product development, it was divided into 2 layers with one connecting bridge between those 2 layers.   The 2 layers were Kernel Layer which working on the Kernel of the OS, while the Ring 3 Layer working on the Net Framework.   In order to achieve the detection and deletion of the Keyloggers, the project was divided into 4 phases:

Phase 1 – Kernel Layer (Kernel Mode Driver Framework)

For the phase 1 of the product development, the Anti-Keyloggers Program (AKP) included several projects in the repository.   The descriptions of the projects are as follow:

10.1 PPLLProtect.sys

This project aimed to protect the process of the program and to prevent to be re-write and end.   The project aimed to provide process protection against memory being ended and maliciously modified by other software.

It adapted ObRegisterCallbacks and PsCreateSystemThread to protect the particular program running (Microsoft, 2022).

By using the PsProcessType to command on call back types of ObRegisterCallbacks, i.e.,
OpOperAtion.ObjectType = PsProcessType (To monitor for the target types of process)
OpOperAtion.Operations = OB_OPERATION_HANDLE_CREATE |
OB_OPERATION_HANDLE_DUPLICATE (To monitor handle develop and updates)
OpOperAtion.PreOperation = (POB_PRE_OPERATION_CALLBAC K) & OneProcessAction
(To monitor on the bound functions)
ObRegisterCallbacks, CallBackReg & ProcessListenThread
(To monitor on the registered bound functions)
CurrentPID = PsGetProcessId (HANDLE) & pOperationInformation ->Object (PEPROCESS)
(To retrieve the reference document of process's ID of operating process)

When the program received any process of the "DUPLICATE" or "CREATE" functions, it will run into the "OneProcessAction" method for monitoring and implement the "ObRegisterCallbacks" process.

10.1.1 SetProcessAuthorityProtection

The sub-project SetProcessAuthorityProtection aimed to work on the Windows NT layer to pass after setting Protection Permissions (PPLL) on a specified process.

10.1.1.1 OneProcessAction Implementation

Using of the function code "char szProcName[25] = { 0 }"to preserve the process name, and using "strcpy(szProcName,GetProcessImageNameByProcessID((ULONG)CurrentPID))" to copy the progress name copied from the ID to "szProcName".

```
/// <summary>
///  Listen NTOpenProcess Message
/// </summary>
/// <param name="RegistrationContext"></param>
/// <param name="pOperationInformation"></param>
/// <returns></returns>

OB_PREOP_CALLBACK_STATUS OneProcessAction(PVOID RegistrationContext, POB_PRE_OPERATION_INFORMATION pOperationInformation)
{
    HANDLE CurrentPID = PsGetProcessId((PEPROCESS)pOperationInformation->Object);

    char szProcName[25] = { 0 };

    UNREFERENCED_PARAMETER(RegistrationContext);

    strcpy(szProcName, GetProcessImageNameByProcessID((ULONG)CurrentPID));
```

10.1.1.2 GetProcessImageNameByProcessID Implementation

Using the function code "char* GetProcessImageNameByProcessID (ULONG ulProcessID)" to implemented the getting process name by process ID, and carry out the protection:

```
main.cpp  → ×  DeFine.cs
PPLLProtect              ▼   (Global Scope)              ▼
   209      }
   210
   211     /// <summary>
   212     /// PID To ProcessName
   213     /// </summary>
   214     /// <param name="ulProcessID"></param>
   215     /// <returns></returns>
   216     char* GetProcessImageNameByProcessID(ULONG ulProcessID)
   217     {
   218         NTSTATUS  Status;
   219         PEPROCESS  EProcess = NULL;
   220
   221
   222         Status = PsLookupProcessByProcessId((HANDLE)ulProcessID, &EProcess);    //EPROCESS
   223
   224         //Testing the effectiveness of PsLookupProcess by ProcessId Call State
   225         if (!NT_SUCCESS(Status))
   226         {
   227             return FALSE;
   228         }
   229         ObDereferenceObject(EProcess);
   230         //Return ProcessName
   231         return (char*)PsGetProcessImageFileName(EProcess);
   232     }
   233
```

10.1.1.3 PsLookupProcessByProcessID

It was used to get details information about the process. The system processes were created to prevent the program from being attacked or modified by spyware. PsCreateSystemThread would create a system process to assign the lightweight protection

A level of protection was assigned:

pSignatureProtect → Protection.Type = 2;

pSignatureProtect → Protection.Audit = 0

pSignatureProtect → Protection.Signer = 6;

10.1.2   IODeviceControl

An IO communication layer and DLLs would communicate by using the IODevice to receive return information. It would also pass parameter content to reception for execution which resulted to the return Real-time message return by ProcessCommand.

```
364  void ProcessCommand(PVOID Buffer)
365  {
366      int i = 0;
367
368      char* GetAdd = (char*)Buffer;
369      char GetFristChar = GetAdd[0];
370      char Cache[25];
371
372      while (*GetAdd)
373      {
374          GetAdd++;
375          Cache[i] = *GetAdd;
376          i++;
377      }
378
379      ULONG Target = Kernel_Atoi(Cache);
380
381      if (GetFristChar == 'P')
382      {
383          DbgPrint("PPL ProtectProcess:%i", Target);//Set PPLLProtect
384          SetPPLProtect(PidToHandleA(Target));
385      }
386      else
387          if (GetFristChar == 'U')
388          {
389              DbgPrint("PPL UNProtectProcess:%i", Target);
390              UNPPLProtect(PidToHandleA(Target));//Cancel PPLLProtect
391          }
392
393  }
```

10.2 Process Listen

It was a real-time response module which monitor for messages created by the process and the "parent" process which was created by the source.

10.2.1 Listen Process Creation (Parent & Child)

It was a Windows NT layer that was monitored by the PsSetCreateProcessNotifyRoutine (It was a native function of a kernel function provided by Windows to monitor new process creation).

10.2.2 Get Process Path

The Windows NT layer would obtain the process path through the process ID. The process path of all processes, including all system processes were able to get by this path.
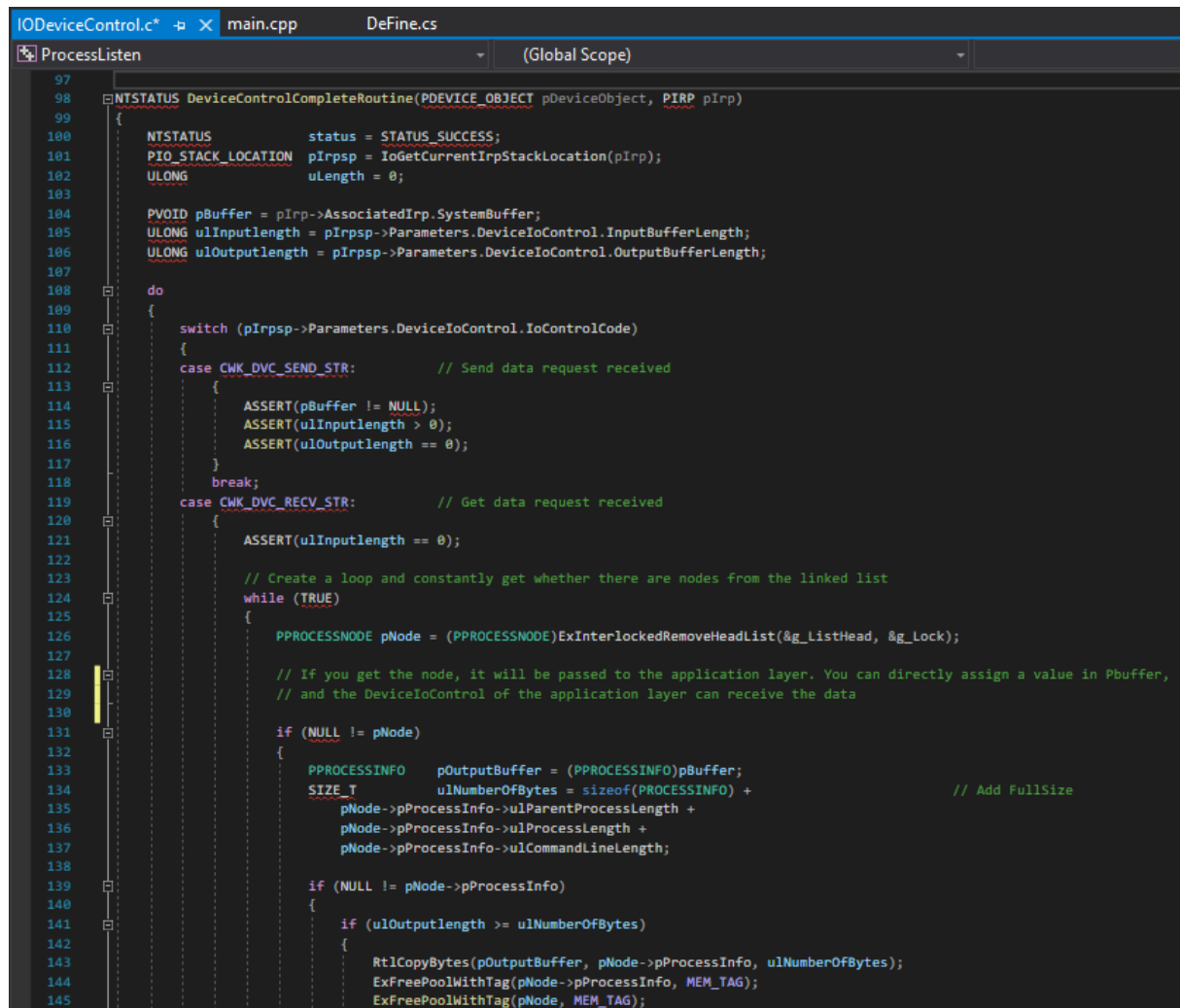
10.2.3 IO Device Control

The IO communication layer and DLLs would communicate by using the IODevice to receive returned information. It would also allow to pass the parameter content and receive the execution results and returned to the real-time message.

It was implemented by caller end the dead loop timing send control code, CWK_DVC_RECV_STR. After sending, the kernel determined whether the linked list had the items. If the list had the item, it would write a pointer to the structure within the linked list to output buffer.

```c
NTSTATUS DeviceControlCompleteRoutine(PDEVICE_OBJECT pDeviceObject, PIRP pIrp)
{
    NTSTATUS             status = STATUS_SUCCESS;
    PIO_STACK_LOCATION   pIrpsp = IoGetCurrentIrpStackLocation(pIrp);
    ULONG                uLength = 0;

    PVOID pBuffer = pIrp->AssociatedIrp.SystemBuffer;
    ULONG ulInputlength = pIrpsp->Parameters.DeviceIoControl.InputBufferLength;
    ULONG ulOutputlength = pIrpsp->Parameters.DeviceIoControl.OutputBufferLength;

    do
    {
        switch (pIrpsp->Parameters.DeviceIoControl.IoControlCode)
        {
        case CWK_DVC_SEND_STR:          // Send data request received
            {
                ASSERT(pBuffer != NULL);
                ASSERT(ulInputlength > 0);
                ASSERT(ulOutputlength == 0);
            }
            break;
        case CWK_DVC_RECV_STR:          // Get data request received
            {
                ASSERT(ulInputlength == 0);

                // Create a loop and constantly get whether there are nodes from the linked list
                while (TRUE)
                {
                    PPROCESSNODE pNode = (PPROCESSNODE)ExInterlockedRemoveHeadList(&g_ListHead, &g_Lock);

                    // If you get the node, it will be passed to the application layer. You can directly assign a value in Pbuffer,
                    // and the DeviceIoControl of the application layer can receive the data

                    if (NULL != pNode)
                    {
                        PPROCESSINFO    pOutputBuffer = (PPROCESSINFO)pBuffer;
                        SIZE_T          ulNumberOfBytes = sizeof(PROCESSINFO) +          // Add FullSize
                            pNode->pProcessInfo->ulParentProcessLength +
                            pNode->pProcessInfo->ulProcessLength +
                            pNode->pProcessInfo->ulCommandLineLength;

                        if (NULL != pNode->pProcessInfo)
                        {
                            if (ulOutputlength >= ulNumberOfBytes)
                            {
                                RtlCopyBytes(pOutputBuffer, pNode->pProcessInfo, ulNumberOfBytes);
                                ExFreePoolWithTag(pNode->pProcessInfo, MEM_TAG);
                                ExFreePoolWithTag(pNode, MEM_TAG);
```
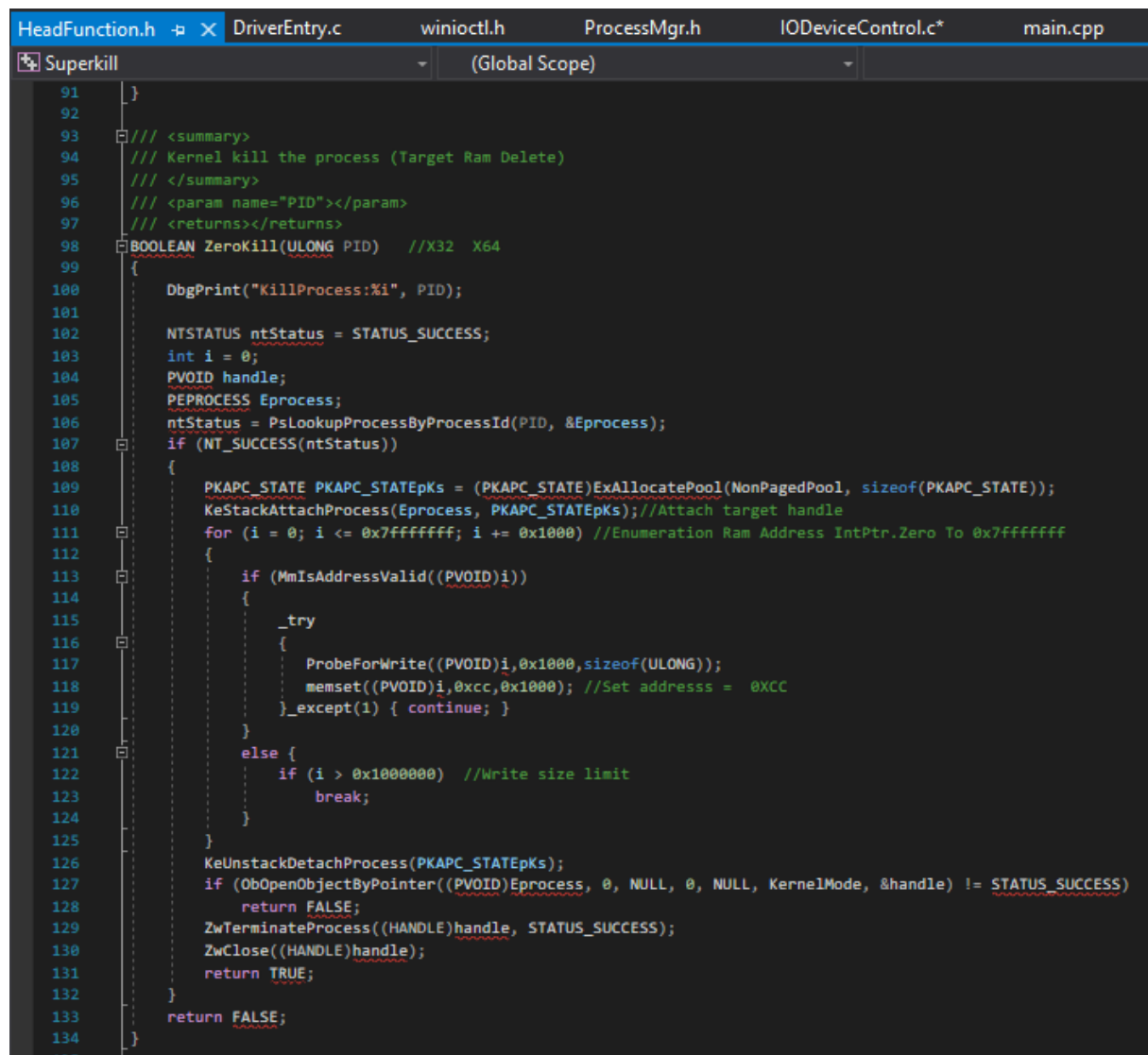
10.3   Process Operation

The project aimed to remove the detected malware program. It provided the process end in the kernel layer and restored the process.

10.3.1   Super Kill Process (Ram Damage)

The project aimed to work in the Windows NT layer, by attaching the NT layer to the process, modified the process memory to clear the content to be 0 in order to achieve the end of the process.
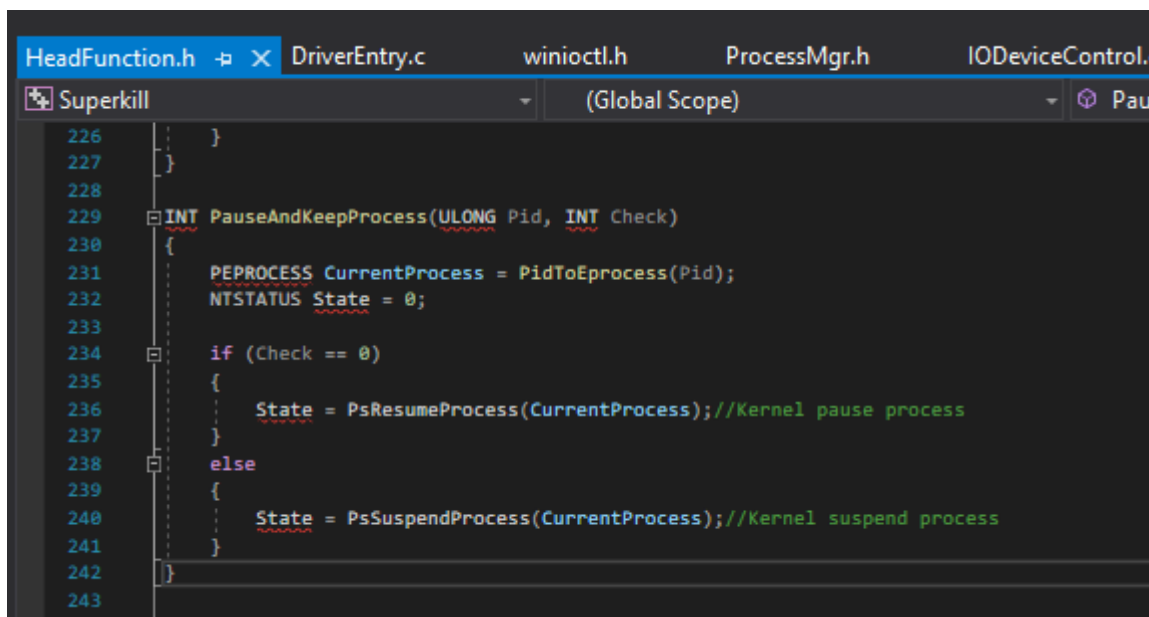
```
HeadFunction.h  ⊟ ✕   DriverEntry.c      winioctl.h      ProcessMgr.h      IODeviceControl.c*      main.cpp
⊞ Superkill                              ▾   (Global Scope)                        ▾

    91    }
    92
    93    ⊟/// <summary>
    94    /// Kernel kill the process (Target Ram Delete)
    95    /// </summary>
    96    /// <param name="PID"></param>
    97    /// <returns></returns>
    98    ⊟BOOLEAN ZeroKill(ULONG PID)    //X32  X64
    99    {
   100        DbgPrint("KillProcess:%i", PID);
   101
   102        NTSTATUS ntStatus = STATUS_SUCCESS;
   103        int i = 0;
   104        PVOID handle;
   105        PEPROCESS Eprocess;
   106        ntStatus = PsLookupProcessByProcessId(PID, &Eprocess);
   107    ⊟    if (NT_SUCCESS(ntStatus))
   108        {
   109            PKAPC_STATE PKAPC_STATEpKs = (PKAPC_STATE)ExAllocatePool(NonPagedPool, sizeof(PKAPC_STATE));
   110            KeStackAttachProcess(Eprocess, PKAPC_STATEpKs);//Attach target handle
   111    ⊟        for (i = 0; i <= 0x7fffffff; i += 0x1000) //Enumeration Ram Address IntPtr.Zero To 0x7fffffff
   112            {
   113    ⊟            if (MmIsAddressValid((PVOID)i))
   114                {
   115                    _try
   116    ⊟                {
   117                        ProbeForWrite((PVOID)i,0x1000,sizeof(ULONG));
   118                        memset((PVOID)i,0xcc,0x1000); //Set addresss =  0XCC
   119                    }_except(1) { continue; }
   120                }
   121    ⊟            else {
   122                    if (i > 0x1000000)  //Write size limit
   123                        break;
   124                }
   125            }
   126            KeUnstackDetachProcess(PKAPC_STATEpKs);
   127            if (ObOpenObjectByPointer((PVOID)Eprocess, 0, NULL, 0, NULL, KernelMode, &handle) != STATUS_SUCCESS)
   128                return FALSE;
   129            ZwTerminateProcess((HANDLE)handle, STATUS_SUCCESS);
   130            ZwClose((HANDLE)handle);
   131            return TRUE;
   132        }
   133        return FALSE;
   134    }
   135
```
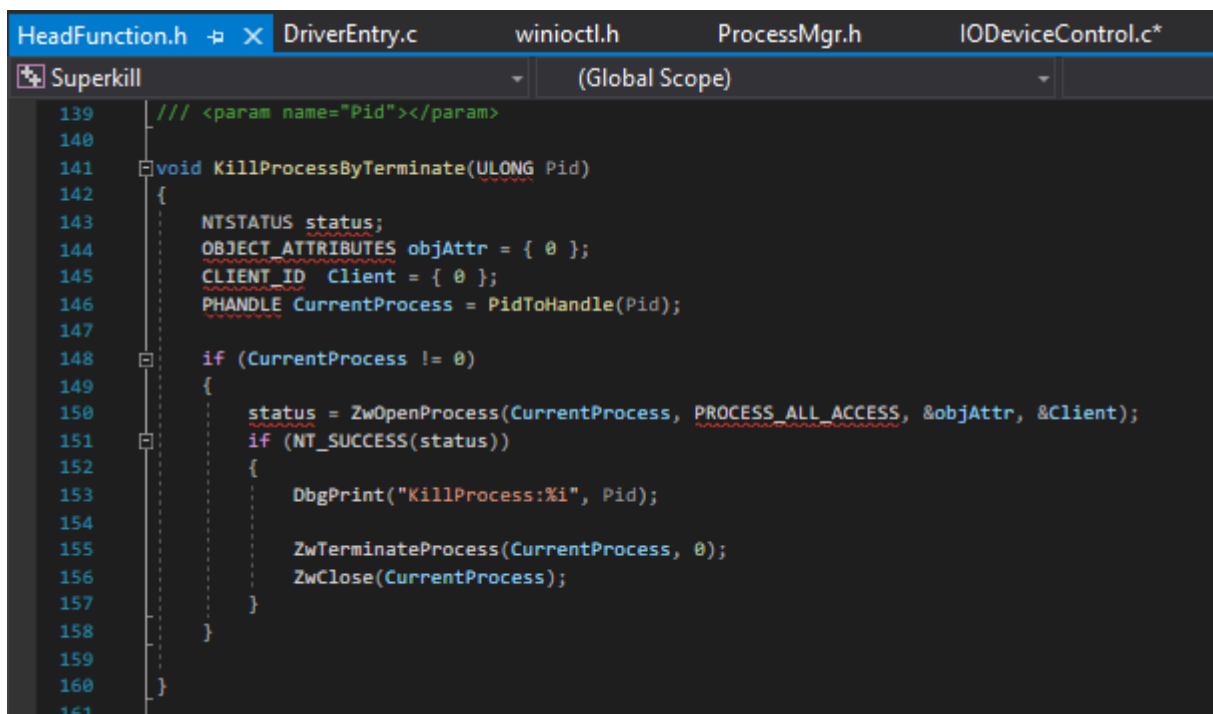
10.3.2 Suspend Process (Pause and Keep)

The sub-project aimed to allow the program to suspend and resume the process in the system, including the system processes.

```
HeadFunction.h  ⊣  ✕   DriverEntry.c        winioctl.h        ProcessMgr.h        IODeviceControl.
Superkill                                (Global Scope)                    ▾  ◉  Pau
226            }
227        }
228
229    ☐INT PauseAndKeepProcess(ULONG Pid, INT Check)
230     {
231          PEPROCESS CurrentProcess = PidToEprocess(Pid);
232          NTSTATUS State = 0;
233
234    ☐     if (Check == 0)
235          {
236              State = PsResumeProcess(CurrentProcess);//Kernel pause process
237          }
238    ☐     else
239          {
240              State = PsSuspendProcess(CurrentProcess);//Kernel suspend process
241          }
242     }
243
```

10.3.3 NT Delete File
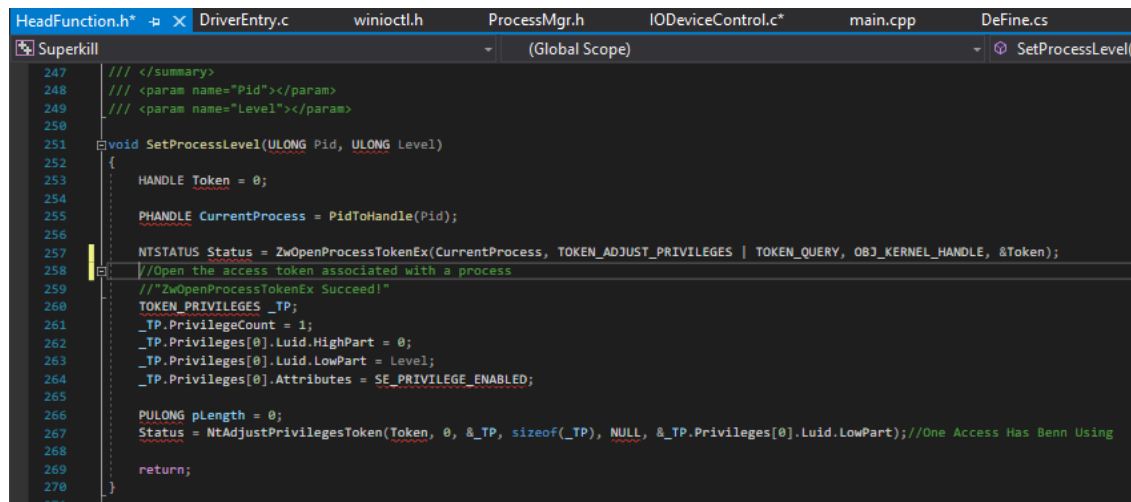
The Windows NT layer deleted files, including those system files, by path. It was important for the system ability to delete the malware files if necessary.

```
HeadFunction.h  ⊣  ✕   DriverEntry.c        winioctl.h        ProcessMgr.h        IODeviceControl.c*
Superkill                                (Global Scope)                    ▾
139       /// <param name="Pid"></param>
140
141    ☐void KillProcessByTerminate(ULONG Pid)
142     {
143          NTSTATUS status;
144          OBJECT_ATTRIBUTES objAttr = { 0 };
145          CLIENT_ID  Client = { 0 };
146          PHANDLE CurrentProcess = PidToHandle(Pid);
147
148    ☐     if (CurrentProcess != 0)
149          {
150              status = ZwOpenProcess(CurrentProcess, PROCESS_ALL_ACCESS, &objAttr, &Client);
151    ☐         if (NT_SUCCESS(status))
152              {
153                  DbgPrint("KillProcess:%i", Pid);
154
155                  ZwTerminateProcess(CurrentProcess, 0);
156                  ZwClose(CurrentProcess);
157              }
158          }
159
160     }
161
```

10.3.4 Set Process Level (Debug System)

It was used to set any process permission in the Windows NT layer. SeDebug permissions of executions of any Open Process, including the system security processes and service processes with specific write-related access rights.
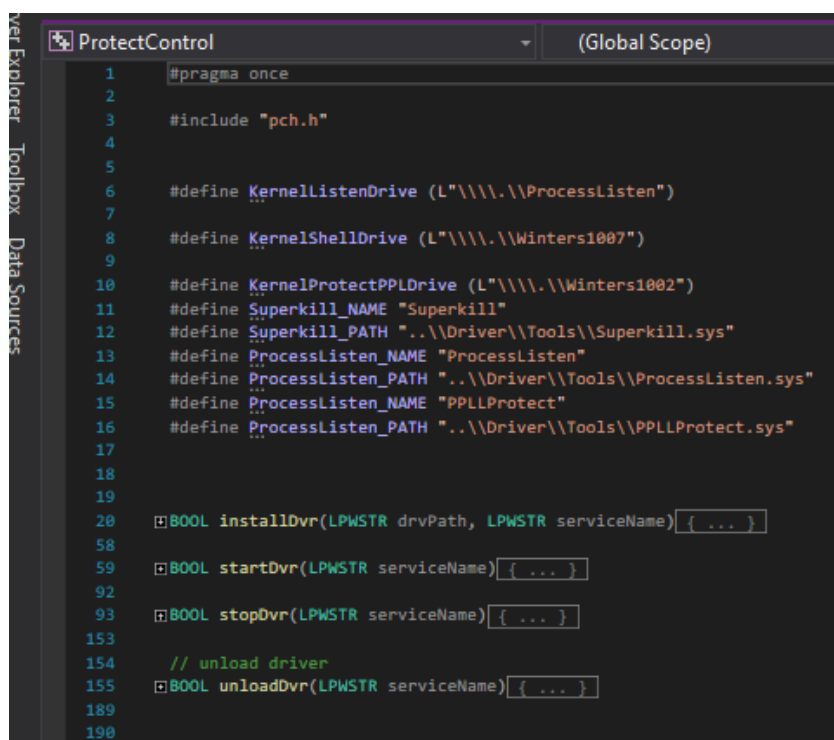
```c
/// </summary>
/// <param name="Pid"></param>
/// <param name="Level"></param>

void SetProcessLevel(ULONG Pid, ULONG Level)
{
    HANDLE Token = 0;

    PHANDLE CurrentProcess = PidToHandle(Pid);

    NTSTATUS Status = ZwOpenProcessTokenEx(CurrentProcess, TOKEN_ADJUST_PRIVILEGES | TOKEN_QUERY, OBJ_KERNEL_HANDLE, &Token);
    //Open the access token associated with a process
    //"ZwOpenProcessTokenEx Succeed!"
    TOKEN_PRIVILEGES _TP;
    _TP.PrivilegeCount = 1;
    _TP.Privileges[0].Luid.HighPart = 0;
    _TP.Privileges[0].Luid.LowPart = Level;
    _TP.Privileges[0].Attributes = SE_PRIVILEGE_ENABLED;

    PULONG pLength = 0;
    Status = NtAdjustPrivilegesToken(Token, 0, &_TP, sizeof(_TP), NULL, &_TP.Privileges[0].Luid.LowPart);//One Access Has Benn Using

    return;
}
```

Phase 2 – Bridging Layer (Dynamic Link Library)

10.4 Protect Control

The protect control project aimed to enable the communications between the Kernel layer and the Ring 3 layer.   It was the bridge layer (DLL Layer) which provided communication to all kernel layers, including PPLLProtect.sys, ProcessListen.sys, SuperKill.sys, it also offered the installation of the driver, execution of the driver, end the executing driver, and uninstall the driver's functionality if necessary.

```c
#pragma once

#include "pch.h"


#define KernelListenDrive (L"\\\\.\\ProcessListen")

#define KernelShellDrive (L"\\\\.\\Winters1007")

#define KernelProtectPPLDrive (L"\\\\.\\Winters1002")
#define Superkill_NAME "Superkill"
#define Superkill_PATH "..\\Driver\\Tools\\Superkill.sys"
#define ProcessListen_NAME "ProcessListen"
#define ProcessListen_PATH "..\\Driver\\Tools\\ProcessListen.sys"
#define ProcessListen_NAME "PPLLProtect"
#define ProcessListen_PATH "..\\Driver\\Tools\\PPLLProtect.sys"



BOOL installDvr(LPWSTR drvPath, LPWSTR serviceName) { ... }

BOOL startDvr(LPWSTR serviceName) { ... }

BOOL stopDvr(LPWSTR serviceName) { ... }

// unload driver
BOOL unloadDvr(LPWSTR serviceName) { ... }
```

Phase 3 – Ring3 Layer (Keyloggers Deletion Layer)

10.5 Win Defense

This project aimed to remove the Keylogger. It called the bridge layer's Dll ProtectControl to manage kernel layer data in real time basis, create timely responses to process, antiviral engines and get messages to the Kernel layer for active defense. The project working on the projects as follow:

- Protect Control.dll
- Install & uninstall.sys
- SafeCore (Including virus behavior libraries, scanning virus and scoring system)
- LocalData (SQLite3 database and behavioral white list)
- Data transformation
- GuiCore (including ActionLayer response for the virus reporting and MainLayer response for general control)
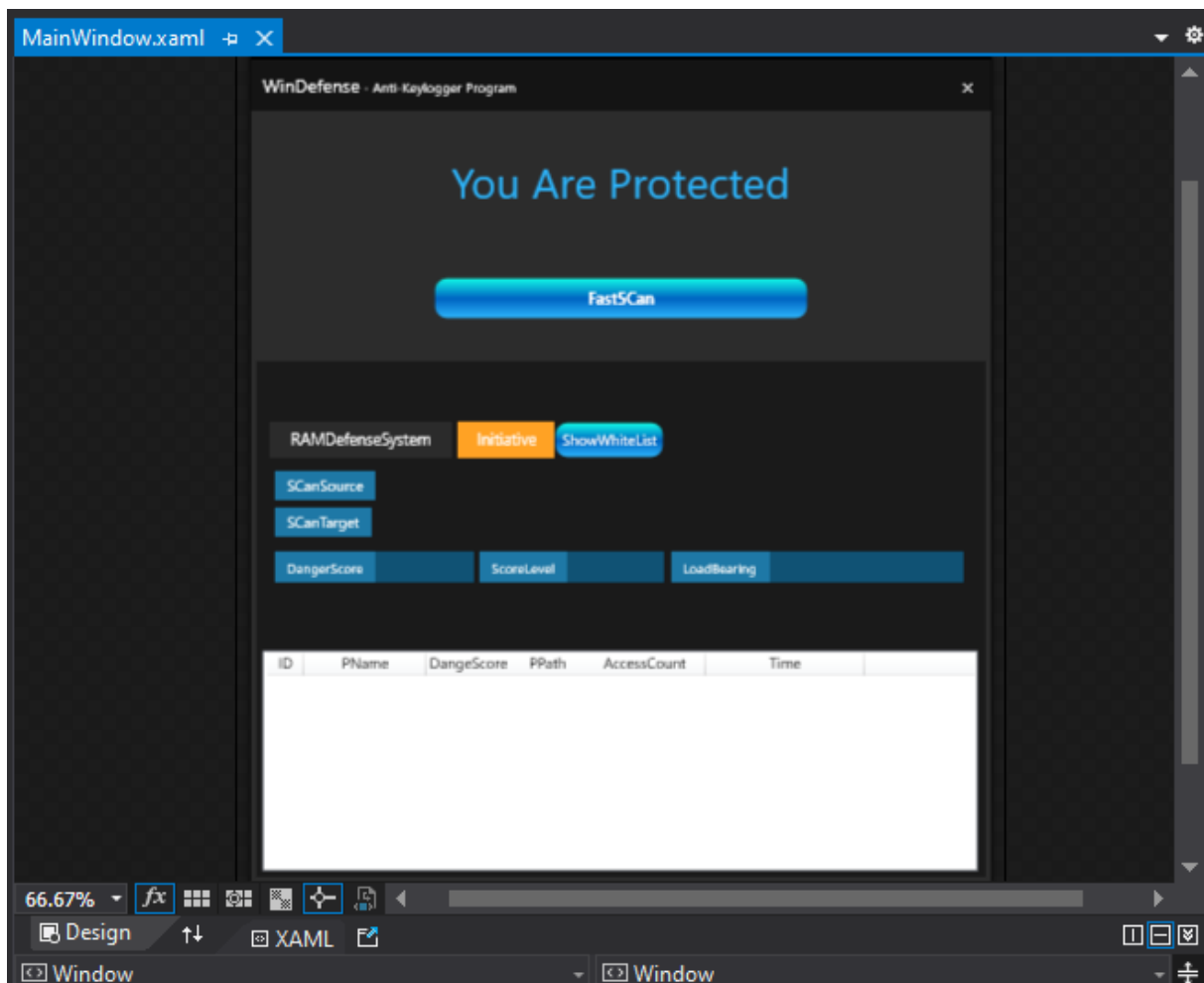
Phase 4 – Product Testing

10.6 BestXSoftware – Free Keylogger Testing

The reason to use this software for program testing because of its free of charge basis. Another reason to use this software because of the features of the free Keyloggers which seemed interesting.   It was found that when the Keylogger used for criminal purpose, it would able to stole all information, including account passwords, browsing history, and other kernel information could be retrieved.   Another feature of the Keylogger was its hermit itself that it could not be found in normal file folder after installation and it could not be uninstalled by user as it ambushed itself in OS already.

10.6.1 Testing Environment

At the beginning, I set up the virtual machine (VM), by using the VMware Workstation 16 Pro, which was the most updated version for product development and testing.   The Window 10 professional 64-bit version 20H2 VM was set up and other device specifications were included Processor: 2.11GHz (2processors) and Installed RAM 8GB.
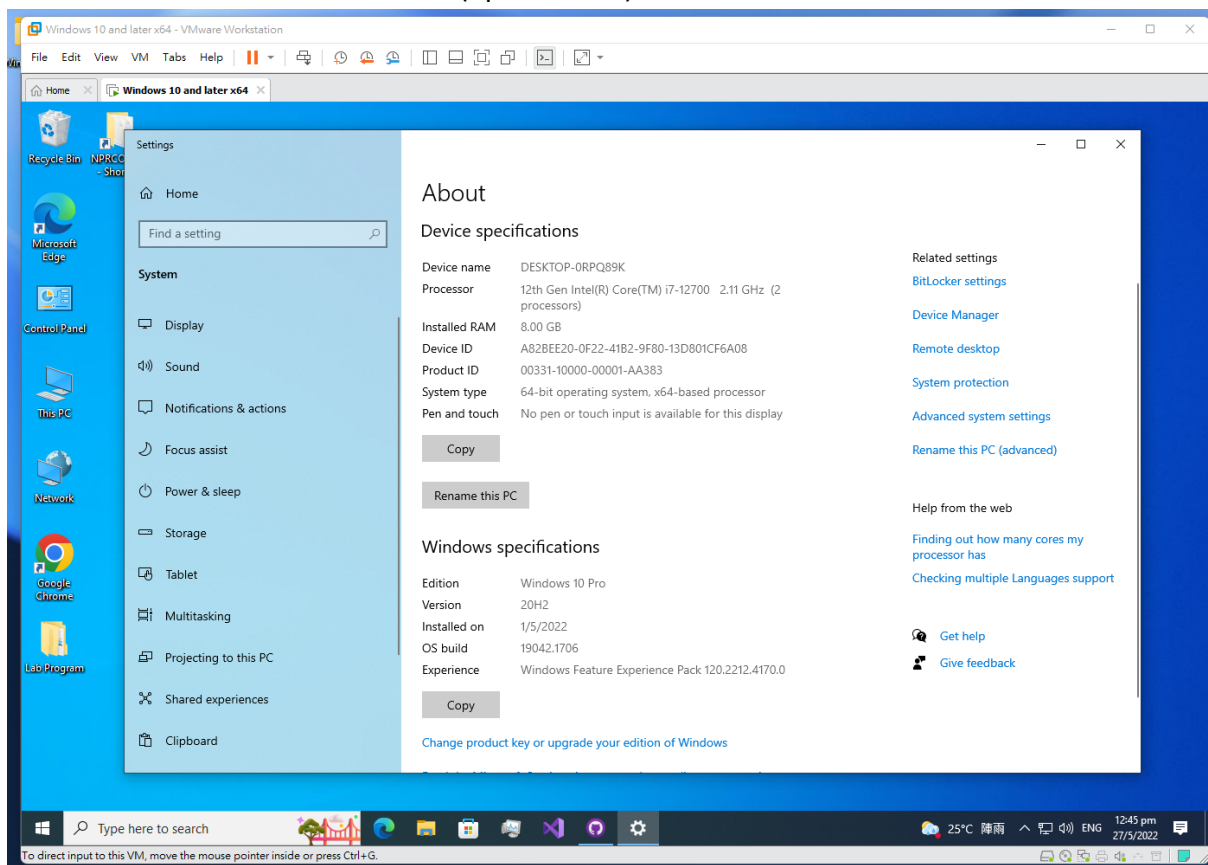
10.6.2 Testing on the downloading stage

After installation of the Anti-Keylogger Program in the VM, while there was a Keylogger downloading in the VM, the program would intercept the downloading process and pop up screen would be showed on the interface.
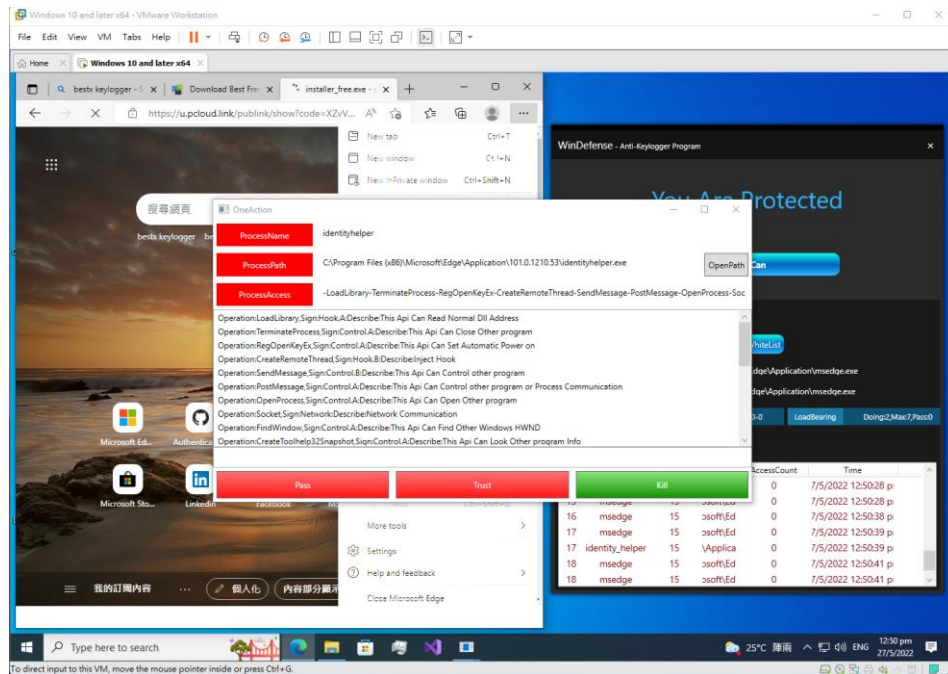
10.6.3 Testing after downloading before installation

In order to test the downloaded Keylogger detection, the Anti-Keylogger Program was firstly closed.   After installation of the Keylogger program in the VM, and after executed the Anti-Keylogger Program, the AKP would scan the Keylogger in the OS and intercept it.   The pop up screen would also be showed on the interface to alert the user.
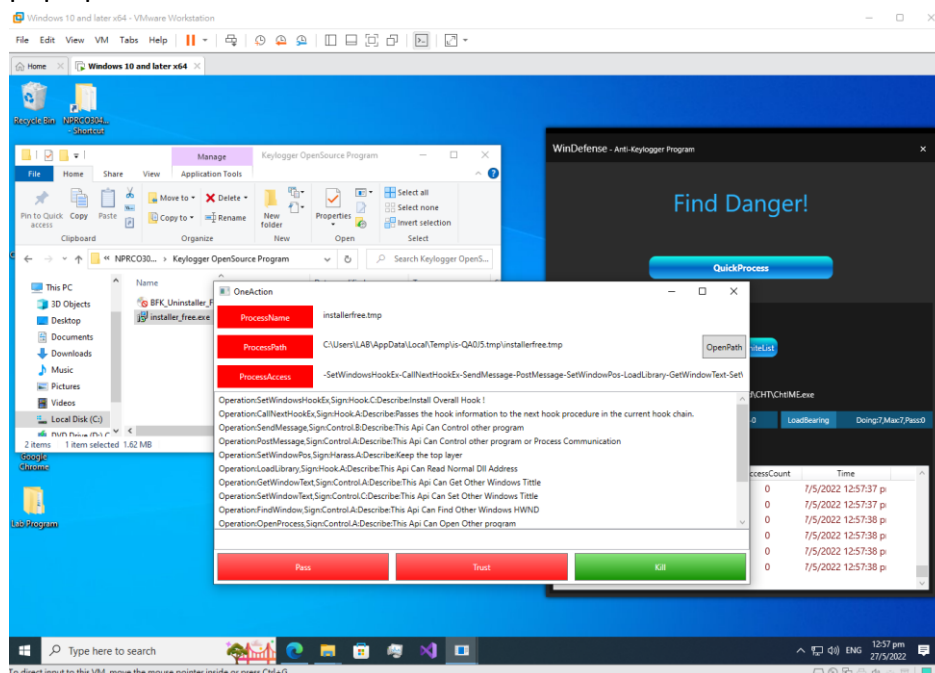
10.6.4 Testing after installation

In order to test the program after installation of the Keylogger, the Keylogger was first installed in the OS of the VM, then installed the Anti-Keylogger Program. The immediate action of the Anti-Keylogger Program would scan the kernel of the OS and detected the Keyloggers. After scanning the Keyloggers, pop up box would ask for the next action by the user. It could be "Pass", "Trust", or "Kill" action provided. If the user choosing "Kill" option, a double confirmed pop up box would also be provided for confirmation. When the Keylogger was "killed", it would be deleted and appeared in the recycle bin of the OS.
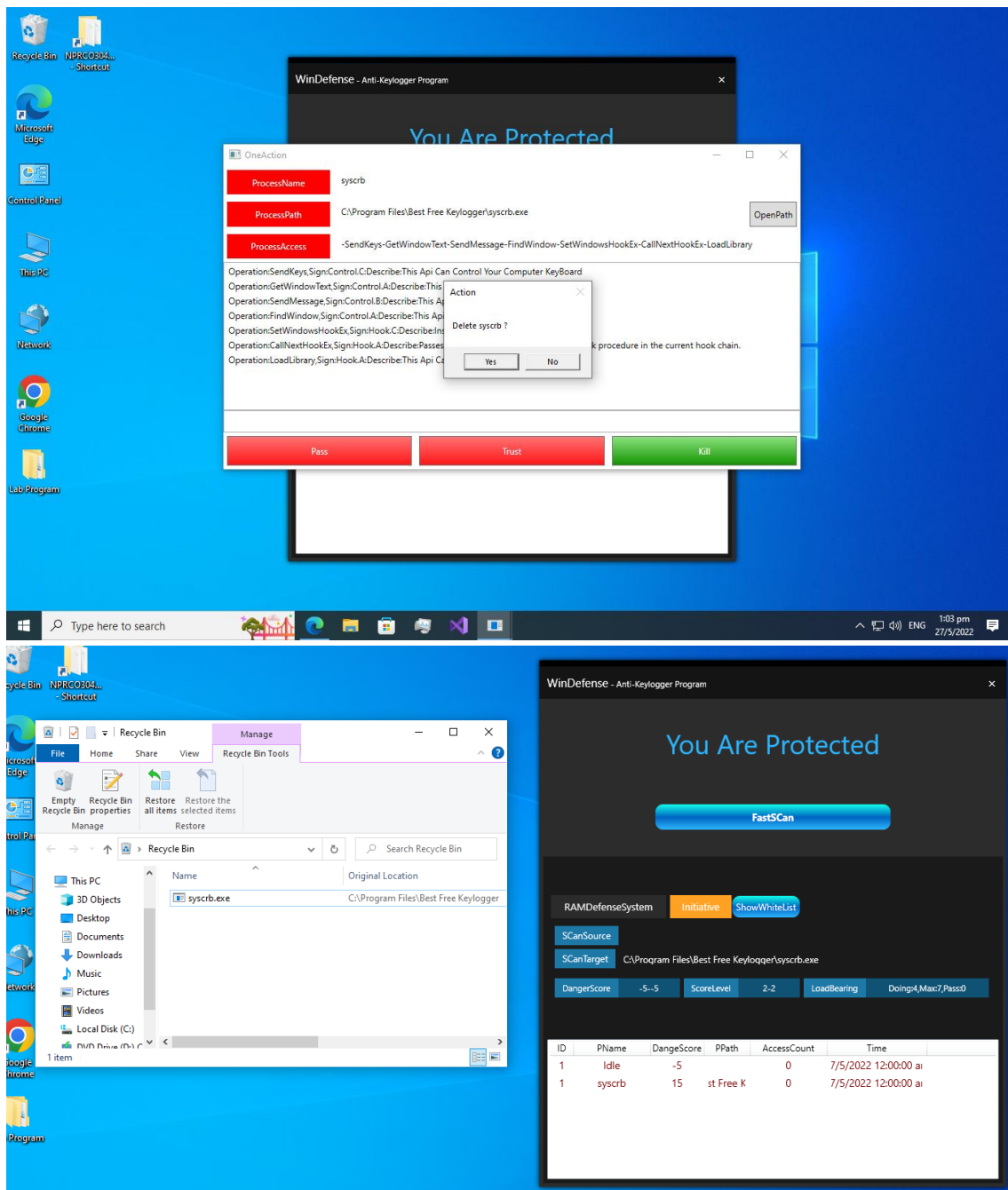
11. Security

11.1 Decompile of the program

The Anti-Keylogger Program was protected by two functions, ObRegisterCallbacks and PsCreateSystemThread, in order to prevent others to compile of the program.

```
/// <summary>
///   Listen NTOpenProcess Message
/// </summary>
/// <param name="RegistrationContext"></param>
/// <param name="pOperationInformation"></param>
/// <returns></returns>
OB_PREOP_CALLBACK_STATUS OneProcessAction(PVOID RegistrationContext, POB_PRE_OPERATION_INFORMATION pOperationInformation)
{
    HANDLE CurrentPID = PsGetProcessId((PEPROCESS)pOperationInformation->Object);

    char szProcName[25] = { 0 };

    UNREFERENCED_PARAMETER(RegistrationContext);

    strcpy(szProcName, GetProcessImageNameByProcessID((ULONG)CurrentPID));
```

The self-protect mechanism of the program prevents other program to assess, read and suspend the cache of the program.
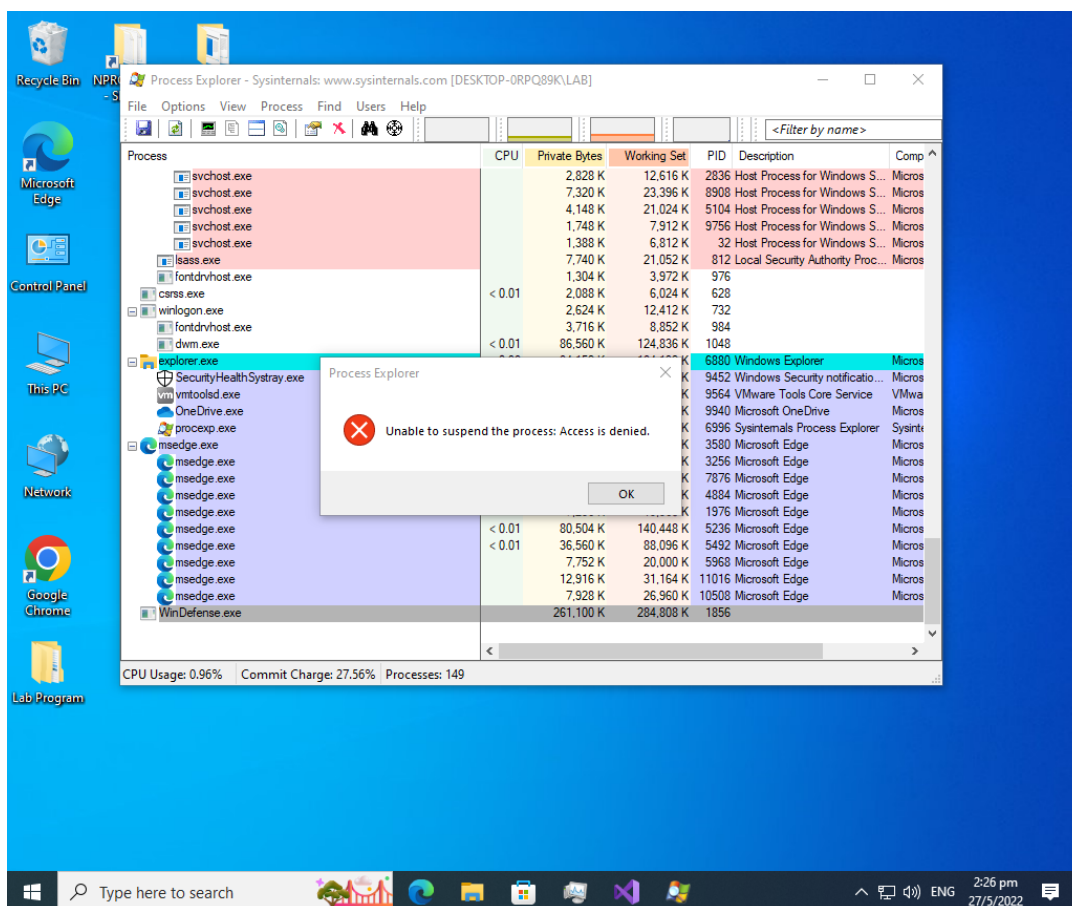
11.2   Database Security

The program used SQLite3 as database for storing the data and Keyloggers behavior. The used of this database because it was very common and easy to use.   It was because only the Keyloggers behavior were needed to be saved in the database, the database was read in the cache of the OS.   If the password of the database was incorrect, it would not be run in the program, therefore, the security of the program was ensured.

12. Project End Report

Project Objective Overview

The project progressed smoothly as there was a careful considerations and planning at the beginning. The stages of development, the approaches and methods, the architecture and the framework were considered in the early stage of the project development.   Both the Keylogger and Anti-Keylogger Program were successfully developed and the project was said to be completed.   It allowed to enhance the understanding of the mechanism of Keyloggers and the way to prevent the invasion by them.

12.1 Project Objectives Review

For the project objectives, the Keylogger program was successfully developed and the mechanism of keylogger in the computer system was understood. The Anti-Keyloggers Program was grossly completed. The functions of detection and deletion of the Keyloggers in the Windows 10 64-bit OS were fulfilled.   The Keyloggers behaviors were learnt and stored in the database. Testing for the functions of the program were done and debugging were grossly completed at the end of the submission date.

For the core deliverables, the client application was created. The user friendly interface was developed and the tasks of the detection and deletion of the Keyloggers were clearly showed on the interface. The prevention of the data exfiltration was ensured, the real-time monitoring modules and front-end modules were included in the program.   In short concluded, most of the objectives of the projects were achieved.

12.2 Review of Project Change

At the beginning of the project development, it was planned that the project would set into a program with scanning of the Windows 10 OS in real-time basis with fast scanning of the new executing process. However, it was found that if there was a malwares installed before the Anti-Keyloggers Program running, the Keyloggers behaviors could not be detected and deleted.

Therefore, the program development was changed into 2 layers (Kernel and Ring 3 layers). It would be more feasible to perform the scanning of the existing process in the OS and preventing the incoming process with malware.

13. Project Evaluations

13.1 Project Objectives Evaluation

The project aimed was understand the Keyloggers behaviors and mechanisms by developing the Anti- Keylogger Program, and to raise the public concerns for the alertness in dealing with the keylogging activities. The aimed for better understanding was fulfilled, while the public concerns would be increased if there are chances to present the program in different platforms in future.

13.2 Development Process Evaluation

The overall structure of the development stages was quite smooth that all working items of each stage were completed. The good used of the Trello for SCRUM master did help for monitoring of the project progress, in addition of meetings with HK supervisor periodically to seek advices also aid for the project development.

Despite the development being monitored, the schedule was very tight which resulted in the hurry for the program testing and the report documentary.

13.3 Technology Evaluation

The complexity of this project was partly attributed to wide range of technologies that were used, one of them was the understanding and writing on the kernel driver. Therefore, lots of time was spent on the studying and the code writing on the both kernel and Ring3 layer. Another problem was the application of the Extended Validation (EV) signature. In order to get a code signing certificate, it required to obtain Extended Validation (EV) certificate which was very expensive and required a company's registration. It became difficult as I don't have any registration of the company with company identity for applying the EV certificate. In order to solve the problems, I finally disable the driver signature verification in the VM's OS in order to complete the program and perform the program testing.

13.4 Limitations of the projects

Advantages and Disadvantages of the Anti-Keyloggers Program (AKP)

The program focuses on the memory protection of the Win System. When any code run, it must be built on the process, and the main protection area would be on the process. It was one of the advantages of the program. Another advantage of the program is that the real-time process creation effectively prevents users from being infected with viruses. However, if the user downloads the virus and doesn't run it, the memory protection is ineffective. The program protects the user's computer from being recorded by Keyloggers forever, that means "No running, no danger". If the users close the software or redo the system or share the virus program with other people, the disadvantage gets bigger and bigger that the program cannot protect the Win System anymore.

Since the program conducted the behavioral analysis alone, it had advantages in high recognition rate but it had quite a high false positive rate.   Once there was a normal software it really needs to install a global keyboard hook and it would also be blocked by the program. On the other hand, the program was lacked of an effective Trojan library and whitelist, it was only included the behavior libraries that the accuracy of the detection would be decreased. Also, there was too few protection layers for real-time interception of process creation which was not enough and it may require a real-time interception of newly created files.

In a short conclude, the comprehensive protection is the memory security and hard drive security.   However, since the works on software engineering is really too large in a given period of time, it was only implemented the protection of the device protected by the program from Keyloggers threats.

13.5   Future Development

When the project aims and objectives were met, and a functional deliverable has been developed, there were areas could be significantly improved.   Starting with studying other malware behaviors cloud updating of the database for evolutions of the Keyloggers or other malware.   In addition, the EV certificate issue would be another concern in future.   It would be better if there is a proper valid EV certificate in order to make the program more complete.

The application used for Keyloggers detection and deletions could also be improved by providing real-time advices and explanations of the detected suspected process which raise the public concerns on the Microsoft OS kernel process behaviors.

13.6   Personal Reflection

In order to have a better understanding of the Keyloggers, some time was spent on the own Keylogger development and it made be the reason for the time insufficiency of the whole project development. During the project development, the 5th wave outbreaks of COVID-19 pandemic gave a great impact to the progress of the works, because most of my family members were infected and my working stress also gave a very great impact on my study. Hopefully, my working colleagues and my supervisors help me to share some of my duties and my supervisor of HKU SPACE, Dr. Beta Yip, gave lots of useful advices in order to enhance my project development.   I already tried my best to overcome all the risks and difficulties during the time of project development and I enjoyed the time during study in HKU SPACE and project development as a mature student.   If I have more time, I would like to engage myself more in the aspect of cyber security and try to explore more opportunities in future.

14. Conclusions

In order to have better understanding of the Keyloggers behavior and raise the public's concerns on the Keylogging attack, an Anti-Keyloggers Program (AKP) was developed. By conducting the project development of the Anti-Keyloggers Program, good understanding on mechanisms of Keyloggers were studied and most of the aims and objectives were achieved. A Keylogger and Anti-Keylogger Program were developed with several testing was carried out for validation. Post project evaluations and self-reflection were mentioned in the report, with future development of the improvement was suggested.

15.References

Atlassian., 2021. *Trello.* Available from: https://trello.com/.

Bestxsoftware., (2022). Best Free Keyloggers. Available from: https://bestxsoftware.com/

BHARDWAJ, A. and GOUNDAR, S., 2020. Keyloggers: silent cyber security weapons. *Network Security*, February 2020, vol. 2020, no. 2, pp. 14-19. Available from: https://www.sciencedirect.com/science/article/pii/S1353485820300210

BROWN, L. and STALLINGS, W., 2018. *Computer security: principles and practice.* Pearson Education.

BISSON, DAVID. (2022). HawkEye Keylogger Acts as First-Stage Loader for Cryptocurrency Miner. Available from: https://securityintelligence.com/news/hawkeye-keylogger-acts-as-first-stage-loader-for-cryptocurrency-miner/

BURNETT, E., PUGH, L. and EYRES, K., 5 Mar, 2021. *Data Protection and Cybersecurity Laws in the United Kingdom.* Available from: https://cms.law/en/int/expert-guides/cms-expert-guide-to-data-protection-and-cyber-security-laws/united-kingdom.

CASE, A., MAGGIO, R.D., FIROZ-UL-AMIN, M., JALALZAI, M.M., ALI-GOMBE, A., SUN, M. and RICHARD, G.G., 2020. Hooktracer: Automatic Detection and Analysis of Keystroke Loggers Using Memory Forensics. *Computers & Security*, September 2020, vol. 96, pp. 101872. Available from: https://www.sciencedirect.com/science/article/pii/S0167404820301450

CRANOR, L., 2005. *Security and usability : designing secure systems that people can use.* L.F. CRANOR and S. GARFINKEL eds., 1st edition ed. Beijing; Sebastopol, California: O'Reilly.

Creative Commons Attribution., 2019. *USB History Viewing.* Available from: https://forensicswiki.xyz/wiki/index.php?title=USB_History_Viewing.

European Commission., 11 Aug, 2021. *The EU Cybersecurity Act.* Available from: https://digital-strategy.ec.europa.eu/en/policies/cybersecurity-act.

European Court of Auditors., 2019. *Challenges to effective EU cybersecurity policy. The briefing paper.* 2019, Available from: https://www.eca.europa.eu/Lists/ECADocuments/BRP_CYBERSECURITY/BRP_CYBERSECURITY_EN.pdf.

FAKHAR, I., 2019. *Computer Forensic: Operating Forensic.* Available from: https://resources.infosecinstitute.com/topic/computer-forensics-operating-system-forensics/.

GITHUB, I., 2022. *SQL Map.* Available from: https://github.com/sqlmapproject/sqlmap.

GlobalSign, GMO. (2021). Secure Your Code. Available from: https://www.globalsign.com/en-hk/company/contact

Kaspersky., 29 MAR 2007. *Keyloggers: How they work and how to detect them (Part 1).* Available from: https://securelist.com/keyloggers-how-they-work-and-how-to-detect-them-part-1/36138/.

Kaspersky., 29 JUN 2011. *Keyloggers: Implementing keyloggers in Windows. Part Two.* Available from: https://securelist.com/keyloggers-implementing-keyloggers-in-windows-part-two/36358/.

MARGOR, J. and Department for Digital, Culture, Media & Sport., 2019. *EU Regulation on ENISA and Cyber Security Certification.* Available from: https://www.gov.uk/government/publications/eu-regulation-on-enisa-and-cyber-security-certification.

Microsoft., (2022). Get a Code Signing Certificate. Available from: https://docs.microsoft.com/en-us/windows-hardware/drivers/dashboard/get-a-code-signing-certificate

Microsoft., (2022). *PsCreateSystemThread function (wdm.h).* Available from: https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-pscreatesystemthread.

Microsoft., (2022). Sysinternals Suite. Available from: https://docs.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite

Microsoft., 2022. *PsGetProcessId function (ntddk.h).* Available from: https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-psgetprocessid.

Microsoft., 2022. *zwDeleteFile* 函数.. Available from: https://docs.microsoft.com/zh-cn/windows-hardware/drivers/ddi/ntifs/nf-ntifs-zwdeletefile?redirectedfrom=MSDN.

National Institute of Standards and Technology, US Department of Commerce. (2022). *National Vulnerabilities Database.* Available from: https://nvd.nist.gov/vuln-metrics/cvss#

Python Software Foundation., 2021. *SQLite 数据库 DB-API 2.0 接口模块.*Available from: https://docs.python.org/zh-cn/3/library/sqlite3.html.

SEN, A. and MADRIA, S., 2020. Application design phase risk assessment framework using cloud security domains. *Journal of Information Security and Applications*, December 2020, vol. 55, pp. 102617. Available from: https://www.sciencedirect.com/science/article/pii/S2214212620307821

Software Freedom Conservancy., 2021. *Git Basics - Working with Remotes.* Available from: https://git-scm.com/book/en/v2/Git-Basics-Working-with-Remotes.

SREENIVAS, R.S. and ANITHA, R., 2011. Detecting keyloggers based on traffic analysis with periodic behaviour. *Network Security*, July 2011, vol. 2011, no. 7, pp. 14-19. Available from: https://www.sciencedirect.com/science/article/pii/S1353485811700769

TALABIS, M. and MARTIN, J., 2013. Chapter 3 - Information Security Risk Assessment: Data Collection. In: M. TALABIS and J. MARTIN eds., *Information Security Risk Assessment Toolkit* Boston: Syngress,2013, pp. 63-104. Available from: https://www.sciencedirect.com/science/article/pii/B9781597497350000038

The Council for Curriculum, Examinations and Assessment., 2022. *Ethical, Legal and environmental impact - Computer Misuse Act (1990)* Available from: https://www.bbc.co.uk/bitesize/guides/z8m36yc/revision/5.

The MITRE Corporation. (2022). *Common Vulnerabilities and Exposures*. Available from: https://cve.mitre.org/

UK Government., 2022. *Data Protection.* Available from: https://www.gov.uk/data-protection.

VATS, R., 2021. *Exciting Cyber Security Project Ideas & Topics For Freshers & Experienced.* Available from: https://www.upgrad.com/blog/cyber-security-project-ideas-topics/.

VIRTUE, T. and RAINEY, J., 2015. Chapter 5 - Information Governance and Risk Management. In: T. VIRTUE and J. RAINEY eds., *HCISPP Study Guide* Boston: Syngress,2015, pp. 91-130. Available from: https://www.sciencedirect.com/science/article/pii/B9780128020432000057

WHEELER, E., 2011. Chapter 11 - Threat and Vulnerability Management. In: E. WHEELER ed., *Security Risk Management* Boston: Syngress,2011, pp. 215-237. Available from: https://www.sciencedirect.com/science/article/pii/B9781597496155000116

Zoho Corp., 2021. *How to Track Group Policy Changes.*
. Available from: https://www.manageengine.com/products/active-directory-audit/how-to/how-to-track-group-policy-changes.html.

大碗宽面., 2020. *原创Hook 技术：Ring3 层下的 Inline Hook 详解.* Available from: https://bbs.pediy.com/thread-259285.htm.

梅森, �., 2021.*什么是渗透测试?*Available from: https://blog.csdn.net/seagal890/article/details/85037669.

16. Appendix

Appendix I – Project Proposal

Project Proposal : Detection and Deletion of Keyloggers in Computer System
A.   Project Vision   Keylogger, which is a surveillance software installed on a system to record the keystroke made on that system.   My project is planned to design a computer program to detect the keylogger software when the computer startup.   The second step is deletion the keyloggers after the detection. Although there have some paid or free anti-free keylogger program, but we don't know the program is safety or not.   Design the program will make me to completely to understanding what the keylogger done and how to avoid it to steal my data.
B.   Keywords   Anti-keylogger/ Keyloggers/ Keystroking/ Capture/ Malicious/ Spyware/   Data leakage/ Remove Keyloggers/

A.   Risk Plan
 	
 -   Changing requirements and priorities.
      Consult to the supervisor when facing the critical problems.
 -   Poor documentation.
      Reference to the previous project.
 -   Failure to deliver on time.
      Apply VL from company when the progress is obviously slow.

Appendix II – Report Highlights

PRCO304HK – Report Highlights 1
Name: Yau Chak Man, Winters
Date: 10th February, 2022
Review of work undertaken: ➢ Set up the GitHub repository for project management ➢ Create the Trello Boards account and update the project set up file ➢ Past assignment review for better management
Plan of work for the next week: ➢ Research the mechanism of the Keylogger ➢ Develop the Keylogger program for more understanding of the mechanism of keylogging activities ➢ Research for the program language used in writing the program
Brief notes from supervisory meeting(s) since last Highlight: ➢ More research on the Big Keylogging events

PRCO304HK – Report Highlights 2
Name: Yau Chak Man, Winters
Date:24th February, 2022
Review of work undertaken: ➢ Research on Keylogging activities done ➢ Decided to work on API hook when writing the program ➢ Updated the Trello Board
Plan of work for the next week: ➢ Write up the Keylogger program for better understanding on the mechanisms of Keylogging activities
Brief notes from supervisory meeting(s) since last Highlight: ➢ Research for outside works on Keyloggers may save more time in better understanding of the Keylogging activites

PRCO304HK – Report Highlights 3
Name: Yau Chak Man, Winters
Date: 17th March, 2022
Review of work undertaken: ➢ Draft of the Project Objectives and Approach of the Program development ➢ Develop the Keylogger Program for evaluation and understanding ➢ Update the Trello Board with the Draft and the Keylogger Program
Plan of work for the next week: ➢ Draft of the Project Aims, Objectives, Approach and Methods in writing the Anti-Keylogger Program ➢ Updated the Project Highlights for record ➢ Write up the Keylogging Behavior Database for program development ➢ Research on CVE on Keylogging for better learning
Brief notes from supervisory meeting(s) since last Highlight: ➢ Advised to use one more Keylogger for testing on the program, e.g., bestx software ➢ Keep main lines of objectives terse ➢ Research on more CVE on Keylogging

PRCO304HK – Report Highlights 4 (Progress report to UK supervisor)
Name: Yau Chak Man, Winters
Date: 29th March, 2022
Review of work undertaken: ➢ Project aim and objective introduction ➢ Method and Approach of the product development explanation ➢ Progress of the product development
Plan of work for the next week: ➢ Completed Project Aims, Objectives, Approach and Methods in writing the Anti-Keylogger Program ➢ Updated the Project Highlights for record ➢ Write up the Keylogging Behavior Database for program development ➢ Research on CVE on Keylogging for better learning
Brief notes from supervisory meeting(s) since last Highlight: ➢ Besides of the aims and the objective which raising public concern on keyloggers attack, how to persuade others to use self-developed anti-keylogger program

PRCO304HK – Report Highlights 5
Name: Yau Chak Man, Winters
Date: 7th April, 2022
Review of work undertaken: - Completed Project Aims, Objectives, Approach and Methods in writing the Anti-Keylogger Program   - Updated the Project Highlights for record   - Write up the Keylogging Behavior Database for program development   - Research on CVE on Keylogging for better learning
Plan of work for the next week: - Git the Keylogging Behavior Database to Git repository   - Keylogging program phase 1 – Detection of the Keyloggers
Brief notes from supervisory meeting(s) since last Highlight: - Seek for help if any difficulties   - Try to use different keyloggers for testing and auditing of the program   - Advised to code the references of the libraries used for counting

PRCO304HK – Report Highlights 6
Name: Yau Chak Man, Winters
Date: 28th April, 2022
Review of work undertaken: - Git the Keylogging Behavior Database to Git repository   - Keylogging program phase 1 – Detection of the Keyloggers   - Updated the Project Highlights for record
Plan of work for the next week: - Debug the program phase 1 – Detection of the Keyloggers   - Testing of the program and git the updated program to the repository   - Construction of the program phase 2 – deletion of the Keyloggers   - Starting on writing the parts of the final report and Git to the repository
Brief notes from supervisory meeting(s) since last Highlight: - Advised to start documentary of the program development, to prevent insufficient time during the last few weeks   - Advised to use different Keyloggers for testing the program   - In order to validate the counting of the program, it is advised to code the libraries used for counting.

PRCO304HK – Report Highlights 7	44

Name: Yau Chak Man, Winters

Date: 19th May, 2022

Review of work undertaken:

➢ Debug the program phase 1 – Detection of the Keyloggers

➢ Testing of the program and git the updated program to the repository

➢ Construction of the program phase 2 – deletion of the Keyloggers

➢ Writing the parts of the final report and Git to the repository

➢ Updated Trello

Plan of work for the next week:

➢ Comprehensive testing of the program

➢ Set up VM for Demonstration

➢ Video preparation for presentation

➢ Presentation PowerPoint preparation

➢ Finalized the Final Project report

Brief notes from supervisory meeting(s) since last Highlight:

➢ Any permanent whitelist programs which skip for scanning during every time startup of the AKP

➢ Kernal driver signature information should be mentioned in the report

➢ Testing keylogger selection

Appendix III – Codes and Libraries Reference

processhacker Github:

https://github.com/processhacker/processhacker/blob/master/phnt/include/ntexapi.h?msclkid=b6006000ced911ec9cc40ae535c790bf

gentilkiwi Github:

https://github.com/gentilkiwi/mimikatz/blob/master/mimidrv/ioctl.h

Mattiwatti Github:

https://github.com/Mattiwatti/PPLKiller/tree/master/PPLKiller

Get Process File Name:

PsGetProcessImageFileName - 获取进程的主模块路径_Vinc 的博客-CSDN 博客

Win64 驱动内核编程-11.回调监控进线程句柄操作:

https://blog.csdn.net/u013761036/article/details/61465245

PsCreateSystemThread function (wdm.h)

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-pscreatesystemthread

ExInterlockedRemoveHeadList function (wdm.h)

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-exinterlockedremoveheadlist

ExAllocatePoolWithTag function (wdm.h)

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-exallocatepoolwithtag

KeWaitForSingleObject function (wdm.h)

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-kewaitforsingleobject

PsLookupProcessByProcessID analysis

https://www.cnblogs.com/aliflycoris/p/5271417.html

驱动层得到进程的完整路径

https://www.cnblogs.com/hi-hooge/p/5833486.html

PsSetCreateProcessNotifyRoutine function (ntddk.h)

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-pssetcreateprocessnotifyroutine

ZwOpenProcess function (ntddk.h)

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-zwopenprocess

ZwTerminateProcess function (ntddk.h)

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-zwterminateprocess

ZwClose function (wdm.h)

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-zwclose

64 位内核开发第十一讲,内核下的进程操作

https://www.cnblogs.com/iBinary/p/11704838.htm

驱动中的进程提权

https://blog.csdn.net/rrrfff/article/details/6163004?locationNum=15&fps=1

zwDeleteFile 函数 (ntifs.h)

https://docs.microsoft.com/zh-cn/windows-hardware/drivers/ddi/ntifs/nf-ntifs-zwdeletefile?redirectedfrom=MSDN

c++加载驱动文件

https://blog.csdn.net/weixin_30478619/article/details/98027844

C#委托实现 C++ Dll 中的回调函数

https://www.cnblogs.com/HappyEDay/p/7742890.html

wpf 下使用 NotifyIcon

https://www.cnblogs.com/lunawzh/p/6135065.html

crc32 根据字符串获取校验值

https://blog.csdn.net/weixin_34348111/article/details/94072691

C#获取文件的 MD5 码

https://blog.csdn.net/weixin_30268921/article/details/98335442

SQLite 数据库的 SQLiteHelper 帮助类

https://blog.csdn.net/u012408847/article/details/80497185

Microsoft Kernel-Mode Driver Framework (KMDF)

https://docs.microsoft.com/en-us/windows-hardware/drivers/wdf/framework-library-versioning#kmdf

CloseHandle

https://docs.microsoft.com/en-us/windows/win32/api/handleapi/nf-handleapi-closehandle

CloseServiceHandle function (winsvc.h)

https://docs.microsoft.com/en-us/windows/win32/api/winsvc/nf-winsvc-closeservicehandl

ZwOpenProcess

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-zwopenprocess

ZwTerminateProcess

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-zwterminateprocess

ZwOpenProcessTokenEx

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntifs/nf-ntifs-zwopenprocesstokenex

GetModuleHandleA

https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-getmodulehandlea

CreateToolhelp32Snapshot

https://docs.microsoft.com/en-us/previous-versions/ms939171(v=msdn.10)

Module32Next

https://docs.microsoft.com/en-us/previous-versions/aa915353(v=msdn.10)

Module32First

https://docs.microsoft.com/en-us/previous-versions/aa915324(v=msdn.10)

lstrlenA

https://docs.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-lstrlena

Process32First

https://docs.microsoft.com/en-us/windows/win32/api/tlhelp32/nf-tlhelp32-process32first

Process32Next

https://docs.microsoft.com/en-us/windows/win32/api/tlhelp32/nf-tlhelp32-process32next

LoadLibraryA

https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-loadlibrarya

GetProcAddress

https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-getprocaddress

OpenSCManager

https://docs.microsoft.com/en-us/windows/win32/api/winsvc/nf-winsvc-openscmanagera

SetWindowsHookEx

https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-setwindowshookexa

CallNextHookEx

https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-callnexthookex

UnhookWindowsHookEx

https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-unhookwindowshookex