BỘ GIÁO DỤC VÀ ĐÀO TẠO

ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC CNTT VÀ TRUYỀN THÔNG VIỆT – HÀN

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN

NĂM 2021

NGHIÊN CỨU VỀ VIRUS MÁY TÍNH VÀ PHÁT TRIỂN PHẦN MỀM PHÁT HIỆN VIRUS MÁY TÍNH ỨNG DỤNG TRÍ TUỆ NHÂN TẠO

< DHVH-SV-2021-14 >

Thuộc lĩnh vực khoa học và công nghệ: Mạng máy tính và truyền thông (An ninh mạng)

Sinh viên chịu trách nhiệm chính thực hiện đề tài: Đỗ Tấn Tĩnh

Ngành học: Công nghệ thông tin

Người hướng dẫn: TS. Trần Thế Sơn

Đà Nẵng , Tháng 5 năm 2021

ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC CNTT VÀ TRUYỀN THÔNG VIỆT – HÀN

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN

NĂM 2021

NGHIÊN CỨU VỀ VIRUS MÁY TÍNH
VÀ PHÁT TRIỂN PHẦN MỀM PHÁT HIỆN VIRUS MÁY TÍNH
ỨNG DỤNG MÔ HÌNH HỌC MÁY

< DHVH-SV-2021-14 >

Xác nhận của đơn vị chủ trì	Sinh viên chịu trách nhiệm chính
(ký, họ tên, đóng dấu)	(ký, họ tên)

Đà Nẵng, tháng 5 năm 2021

DANH SÁCH NHỮNG THÀNH VIÊN THAM GIA NGHIÊN CỨU VÀ ĐƠN VỊ PHỐI HỢP CHÍNH

1. Giảng viên hướng dẫn: TS. Trần Thế Sơn
2. Sinh viên nghiên cứu:
    a. Sinh viên 1:
        - Họ và tên: Đỗ Tấn Tĩnh
        - Lớp: 18IT3
        - Khoa: Khoa học máy tính
        - Số điện thoại: 0389909772
        - Email: dttinh.18it3@vku.udn.vn

    b. Sinh viên 2:
        - Họ và tên: Đỗ Thanh Tùng
        - Lớp: 18IT3
        - Khoa: Khoa học máy tính
        - Số điện thoại: 0382352146
        - Email: dttung.18it3@vku.udn.vn

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG

ĐẠI HỌC ĐÀ NẴNG TRƯỜNG ĐẠI HỌC CNTT VÀ TRUYỀN THÔNG VIỆT - HÀN _____	CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM Độc lập – tự do – hạnh phúc _____ Đà Nẵng, ngày 17 tháng 05 năm 2021

THÔNG TIN KẾT QUẢ NGHIÊN CỨU CỦA ĐỀ TÀI

1. Thông tin chung:

- Tên đề tài: Nghiên cứu về virus máy tính và phát triển phần mềm phát hiện virus máy tính ứng dụng trí tuệ nhân tạo
- Mã đề tài:
- Sinh viên chịu trách nhiệm chính:
- Lớp: 18IT3 Khoa: Khoa học máy tính Năm thứ: 3 Số năm đào tạo: 4,5
- Người hướng dẫn: TS. Trần Thế Sơn
- Thời gian thực hiện: 2 tháng

2. Mục tiêu:

- Hiểu rõ về virus máy tính, các đặc trưng của virus máy tính.
- Phát triển phần mềm cho phép phát hiện sự tồn tại của virus trên máy tính dựa vào các đặc trưng của virus

3. **Tính mới và sáng tạo:**

4. **Kết quả nghiên cứu:**

5. **Sản phẩm**: Phần mềm phát hiện virus máy tính ứng dụng trí tuệ nhân tạo

6. **Về các đóng góp của đề tài cho giáo dục và đào tạo, kinh tế - xã hội, an ninh, quốc phòng:**

7. **Công bố khoa học của sinh viên từ kết quả nghiên cứu của đề tài**

Sinh viên chịu trách nhiệm chính thực hiện đề tài
(ký, họ và tên)

NHẬN XÉT

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Xác nhận của đơn vị

Người hướng dẫn

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CNTT
VÀ TRUYỀN THÔNG VIỆT - HÀN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – tự do – hạnh phúc

Đà Nẵng, ngày 17 tháng 05 năm 2021

THÔNG TIN VỀ SINH VIÊN
CHỊU TRÁCH NHIỆM CHÍNH THỰC HIỆN ĐỀ TÀI

I. SƠ LƯỢC VỀ SINH VIÊN:

Họ và tên: Đỗ Tấn Tĩnh

Sinh ngày: 12 tháng 08 năm 2000

Nơi sinh: xã Tịnh Hà huyện Sơn Tịnh, tỉnh Quảng Ngãi

Lớp: 18IT3 Khóa: 18

ảnh 4x6

Khoa: Khoa học máy tính

Địa chỉ liên hệ: xã Tịnh Hà, huyện Sơn Tịnh, tỉnh Quảng Ngãi

Điện thoại: 0389909772 Email: dttinh.18it3@vku.udn.vn

II. *QUÁ* TRÌNH HỌC TẬP:

* Năm thứ 1:

Ngành học: Công nghệ thông tin Khoa: Khoa học máy tính

Kết quả xếp loại học tập: Giỏi

Sơ lược thành tích:

* Năm thứ 2:

Ngành học: Công nghệ thông tin Khoa: Khoa học máy tính

Kết quả xếp loại học tập: Khá

Sơ lược thành tích:

Xác nhận của đơn vị

Sinh viên chịu trách nhiệm chính thực hiện đề tài
(ký, họ và tên)

MỞ ĐẦU

Với sự ra đời của chiếc máy tính đầu tiên, nghành công nghệ thông tin đã để lại một cột mốc to lớn trong lịch sử phát triển khoa học công nghệ của loài người. Theo thời gian cùng sự cải tiến không ngừng nghỉ, ngày nay mọi người đều có thể dễ dàng sở hữu cho mình một chiếc máy tính gọn nhẹ và cực kì đa dùng trong công việc cũng như đời sống. Dựa vào đó, hệ sinh thái phần mềm ngày càng trở nên đa dạng, phong phú hơn, đáp ứng hầu hết nhu cầu của người dùng hiện nay. Tuy nhiên, bên cạnh những phần mềm tiện ích thì cũng tồn tại những phần mềm được tạo ra với mục đích xấu phục vụ cho một hoặc một nhóm người có khả năng hiểu biết về công nghệ thông tin. Ngày nay, sự phát triển của các phần mềm đó gần như là rầm rộ, sự đơn giản hóa về mặt giao diện ngày càng làm cho ngay cả những người ít hiểu biết về kỹ thuật vẫn có thể sử dụng để phục vụ các ý đồ xấu của bản thân hay tổ chức.

Virus tin học hiện đang là nỗi băn khoăn lo lắng của những người làm công tác tin học, là nỗi sợ của những người sử dụng khi máy tính của mình bị nhiễm virus. Khi máy tính của mình bị nhiễm virus, họ chỉ biết trông chờ vào các phần mềm diệt virus hiện có trên thị trường, trong trường hợp các phần mềm này không phát hiện được hoặc không tiêu diệt được, họ bị lâm vào tình huống rất khó khăn, không biết phải làm thế nào.

Vì lý do đó, có một cách nhìn nhận cơ bản về hệ thống, cơ chế và các nguyên tắc họa động của virus tin học là rất cần thiết. Trên cơ sở dó, có một cách nhìn đúng đắn về virus tin học trong việc phòng chống, kiểm tra, chữa trị cũng như phân tích, nghiên cứu một virus mới xuất hiện.

Ứng với mỗi hệ điều hành đều có những loại virus hoạt động riêng trên nó như ứng với hệ điều hành DOS ta có virus DOS, ứng với hệ điều hành Windows ta có virus Windows. Và sự phát triển của virus tin học là mỗi khi có một phần mềm, một chương trình, một hệ điều hành mới xuất hiện thì virus mới cũng xuất hiện theo và kéo theo là các phần mềm diệt virus. Vì vậy, việc nghiên cứu, nhận dạng và phát hiện virus để từ đó có biện pháp thích hợp để ngăn chặn và phòng trừ virus đạt kết quả cao nhất.

CHƯƠNG 1. MÃ ĐỘC

1.1 Khái niệm

Phần mềm độc hại, hay mã độc là một loại phần mềm hệ thống do các tin tặc hay các kẻ phá hoại tạo ra nhằm gây hại cho các máy tính. Tùy theo cách thức mà tin tặc dùng, sự nguy hại của các loại phần mềm ác ý có khác nhau từ chỗ chỉ hiển thị các cửa sổ có chủ đích cho đến việc tấn công chiếm máy và lây lan sang các máy khác.

Mã độc sẽ bao gồm các loại sau:

- ***Virus*** (vi-rút máy tính):

Các dạng mã độc hoạt động độc lập:

- ***Worm*** (Sâu máy tính)

Các dạng mã độc lây nhiễm qua Internet, USB, mạng LAN...

- ***Trojan***
- ***Spyware***: tự động ghi lại các thông tin của máy tính bị xâm nhập
- ***Adware***: tự động hiện các bảng quảng cáo
- ***Keylogger***: ghi nhận lại toàn bộ thao tác của bàn phím
- ***Backdoor***: mở cửa hậu cho kẻ khác xâm nhập
- ***Rootkit***: Dạng mã độc "tàng hình" trước các chương trình kiểm soát file, tiến trình (process),... tạo đường truy nhập cho kẻ xâm nhập trở lại.

Một số mã độc nguy hiểm:

- ***Wanna Cry***
- ***Worm Sasser***
- ***Baza***

Các tập tin trên hệ điều hành Windows mang đuôi mở rộng sau có nhiều khả năng bị virus tấn công:

- ***.bat***: Microsoft Batch File (Tệp xử lý theo lô nhiều câu lệnh)
- ***.chm***: Compressed HTML Help File (Tệp tài liệu dưới dạng nén HTML)
- ***.cmd***: Command file for Windows NT (Tệp thực thi của Windows NT)
- ***.com***: Command file (program) (Tệp thực thi)
- ***.cpl***: Control Panel extension (Tệp của Control Panel)
- ***.doc***x: Microsoft Word (Tệp của chương trình Microsoft Word)

- ***.exe***: Executable File (Tệp thực thi)
- ***.hlp***: Help file (Tệp nội dung trợ giúp người dùng)
- ***.hta***: HTML Application (Ứng dụng HTML)
- ***.js***: JavaScript File (Tệp JavaScript)
- ***.jse***: JavaScript Encoded Script File (Tệp mã hoá JavaScript)
- ***.lnk***: Shortcut File (Tệp đường dẫn)
- ***.msi***: Microsoft Installer File (Tệp cài đặt)
- ***.pif***: Program Information File (Tệp thông tin chương trình)
- ***.reg***: Registry File (Tệp can thiệp và chỉnh sửa Registry)
- ***.scr***: Screen Saver (Portable Executable File)
- ***.sct***: Windows Script Component
- ***.shb***: Document Shortcut File
- ***.shs***: Shell Scrap Object
- ***.vb***: Visual Basic File
- ***.vbe***: Visual Basic Encoded Script File
- ***.vbs***: Visual Basic File (Tệp được lập trình bởi Visual Basic)
- ***.wsc***: Windows Script Component
- ***.wsf***: Windows Script File
- ***.wsh***: Windows Script Host File
- .**{*}**: Class ID (CLSID) File Extensions

1.2 Tác hại của mã độc

Các chương trình phần mềm độc hại đầu tiên được tạo ra dưới dạng thử nghiệm hoặc cho vui. Ngày nay, phần mềm độc hại thường được sử dụng để đánh cắp thông tin - tài chính, cá nhân hoặc liên quan đến kinh doanh. Phần mềm độc hại có thể được sử dụng cả cho các cuộc tấn công vào các tổ chức (xâm nhập vào mạng cục bộ) và thậm chí trên một quốc gia, cũng như để đánh cắp thông tin cụ thể về một cá nhân (đánh cắp dữ liệu ngân hàng, truy cập chi tiết vào các dịch vụ khác nhau, v.v.).

Hầu hết các loại virus và sâu hiện có được thiết kế để giành quyền kiểm soát thiết bị bị tấn công (máy tính, điện thoại thông minh, v.v.). Sau này, thiết bị được giám sát có thể được sử dụng để gửi thư rác, lưu trữ thông tin bất hợp pháp (ví dụ: nội dung khiêu dâm trẻ em) hoặc để thực hiện các cuộc tấn công của các loại khác.

Một số sản phẩm phần mềm độc hại có thể thuộc một số loại cùng một lúc; những chương trình như vậy thường có đặc điểm của Trojans và sâu, và đôi khi cả virus. Thông thường, một chương trình độc hại được gửi đến người dùng cuối dưới dạng ngựa thành Troia, nhưng sau khi khởi chạy, nó sẽ tự sửa trên thiết bị của người dùng và lây nhiễm các tệp thực thi của các chương trình khác, tức là hoạt động như virus; nó cũng có thể tấn công các thiết bị khác qua mạng, tức là hoạt động như một con sâu.

1.3 Lịch sử của mã độc

1.3.1 Những năm đầu tiên (1980 – 1990)

Vào cuối những năm 1980, mục tiêu của các chương trình độc hại là khu vực khởi động đơn giản (***boot sector***) và lây nhiễm tệp tin hoàn toàn ngoại tuyến thông qua đĩa mềm của người dùng.

Khi việc tiếp nhận và mở rộng mạng máy tính tiếp tục diễn ra trong nửa đầu những năm 1990, việc phân phối phần mềm độc hại trở nên dễ dàng hơn, do đó số lượng mã độc cũng tăng lên. Khi các công nghệ được tiêu chuẩn hóa, một số loại phần mềm độc hại đã phát triển. Các ***macro virus*** (cho phép lây lan phần mềm độc hại qua tệp đính kèm email) khai thác các sản phẩm Microsoft Office đã tăng khả năng phân phối nhờ việc tăng cường sử dụng email.

Vào giữa những năm 1990, các doanh nghiệp ngày càng bị ảnh hưởng nhiều hơn, phần lớn là do macro virus, có nghĩa là việc lan truyền đã trở nên theo hướng mạng.

1.3.2 Phần mềm độc hại trong thời đại Web 2.0

Việc phân phối được đẩy nhanh hơn nữa nhờ sự gia tăng sử dụng internet và việc áp dụng các công nghệ Web 2.0, điều này đã thúc đẩy một môi trường phát triển và lây lan của phần mềm độc hại trở nên thuận lợi hơn.

Vào cuối những năm 1990, virus đã bắt đầu ảnh hưởng đến người dùng gia đình, với việc lan truyền email ngày càng gia tăng.

Dưới đây là mẫu của một số phần mềm độc hại tiêu biểu đã được phát hành trong thời gian này:

Brain: Được phát hành năm 1986, là loại ***stealth virus*** - "virus tàng hình" đầu tiên (có phương tiện che giấu sự tồn tại của nó). Mục đích của nó là lây nhiễm vào các máy tính ***MS-DOS***.

- ***Jerusalem***: Được phát hiện vào năm 1987, là một loại virus ***DOS*** được thực thi vào thứ sáu ngày 13 trong các năm sau đó. Nó xoá, làm quá tải và phá hỏng hoạt động của các chương trình đang thực hiện trong ngày hôm đó.
- ***Morris Worm***: Được phát hành vào năm 1988, là tác phẩm đầu tiên được phát tán qua internet. Cơ bản hoạt động của nó là khiến mạng ngừng hoạt động trong vòng 24 giờ.
- ***Michelangelo***: Được phát hiện vào năm 1991, được thiết kế để lây nhiễm các hệ thống dựa trên ***DOS***. Nó phá hỏng khiến dữ liệu của ổ cứng không thể sử dụng. Nó sẽ hoạt động vào ngày 6 tháng 3 hằng năm.
- ***CIH***: Được phát hành năm 1998, là một loại vi rút ***Microsoft Windows 9x***. Nó có thể ghi đè dữ liệu trên ổ cứng máy tính, biến dữ liệu thành một mớ vô dụng. ***CIH*** cũng có khả năng ghi đè thông tin BIOS, ngăn không cho máy tính khởi động.
- ***Melissa***: Được phát hành năm 1999, là một loại macro virus gửi email hàng loạt đầu tiên. Nó sử dụng sổ địa chỉ Outlook của các máy tính đã bị nhiễm để tự gửi thư cho 50 người cùng lúc.

1.3.3 Phần mềm độc hại trong thế kỉ 21

Sự gia tăng sử dụng các bộ công cụ khai thác (***exploit kits*** - các chương trình được tội phạm mạng sử dụng để khai thác các lỗ hổng hệ thống) đã dẫn đến sự bùng nổ của phần mềm độc hại có khả năng lây nhiễm trực tuyến trong những năm 2000.

Tự động chèn SQL (***Automated SQL injection*** - một kỹ thuật được sử dụng để tấn công các ứng dụng theo hướng dữ liệu) và các hình thức xâm nhập trang web hàng loạt khác đã làm tăng khả năng phân phối trong năm 2007. Kể từ đó, số lượng các cuộc tấn công bằng phần mềm độc hại đã tăng theo cấp số nhân, tăng gấp đôi hoặc hơn mỗi năm.

Vào đầu thiên niên kỷ mới, ***internet*** và ***email worms*** đã trở thành tiêu đề trên toàn cầu. Điển hình trong số đó là:

- ***ILOVEYOU***: đã tấn công hàng chục triệu máy tính chạy Windows vào năm 2000. Nó có thể ghi đè lên các file CSS, HTA, JSE và phá huỷ các file JPEG và file âm thanh. Nó còn có thể lấy thông tin cá nhân của nạn nhân để gửi tới kẻ tấn công. Nguy hiểm nhất là nó có khả phát tán cho mọi

người trong sổ địa chỉ của phần mềm Outlook tạo nên sức lây nhiễm cực kì lớn.

- ***Anna Kournikova***: ra đời năm 2001, đã gây ra sự cố cho các máy chủ email trên khắp thế giới bằng cách phát tán tới các địa chỉ email liên hệ của nạn nhân.
- ***Sircam***: hoạt động vào năm 2001, tự phát tán qua email trên các hệ thống chạy Windows. Các file (thường là ***.docs*** hoặc ***.xls***) sẽ được chọn ngẫu nhiên để lây nhiễm virus. Sau đó ***Sircam*** sẽ gửi những file này tới các địa chỉ trong sổ địa chỉ của máy chủ khiến các tệp của nạn nhận bị phát tán cũng như giúp Sircam có thể lây nhiễm rộng hơn.
- ***CodeRed***: lây lan vào năm 2001 nhắm đến các máy tính chạy phần mềm máy chủ ***Web Internet Information Server (IIS)*** bằng cách lợi dụng lỗ hổng tràn bộ đệm. Nó tấn công các dịch vụ website lưu trữ ở máy chủ này và từ chối dịch vụ (***DoS***) vào những địa chỉ IP nhất định.
- ***Nimda***: xuất hiện vào năm 2001, đã ảnh hưởng đến các máy tính chạy các phiên bản Windows khác nhau. Mục đích thực của nó là làm cho lưu lượng Internet bị chật cứng. Nó có khả năng lây lan bằng nhiều phương pháp trực tuyến, trong đó có cả phát tán bằng email.

1.3.4 Phần mềm độc hại từ năm 2000

Trong suốt ***năm 2002*** và ***2003***, người dùng internet đã bị cản trở bởi các cửa sổ bật lên mất kiểm soát và các lệnh với mục đích xấu được nhúng vào website bằng Javascript. Năm 2003 là năm xuất hiện ***Slammer*** - một loại worm gây ra từ chối dịch vụ (***DoS***) trên một số mày chủ và làm chậm lưu lượng truy cập internet. Nó có khả năng lan truyền với vận tốc kỉ lục, truyền cho khoảng 75 ngàn máy trong 10 phút.

Năm 2004: Thế hệ mới của mã độc hại là Sasser ***worm*** ra đời. Với loại ***worm*** này thì người ta không cần phải mở đính kèm của điện thư mà chỉ cần mở lá thư là đủ cho nó xâm nhập vào máy. ***Sasser worm*** không hoàn toàn hủy hoại máy mà chỉ làm cho máy chủ trở nên chậm hơn và đôi khi nó làm máy tự khởi động trở lại.

Năm 2005: Sự xuất hiện của các Virus lây qua các dịch vụ chatting của các mạng xã hội. Các dịch vụ chatting trực tuyến như Yahoo!, MSN, Twitter,.. bắt đầu được virus lợi dụng như một công cụ phát tán trên mạng.

Năm 2008: Các tin tặt tận dụng thông tin đăng nhập ***FTP*** đánh cắp được để đưa iframe vào hàng chục nghìn trang web. Đặc biệt là các cuộc tấn công lớn thuộc kiểu tấn công tự động chèn SQL (***Automated SQL Injection attacks***).

Năm 2010: Các hệ thống máy tính công nghiệp là mục tiêu của ***Stuxnet worm***. Công cụ độc hại này nhắm mục tiêu vào máy móc trên dây chuyền lắp ráp của nhà máy cụ thể. Nó đã từng gây thiệt hại đến mức nó được cho là đã tấn công và gây ra sự phá hủy hàng trăm máy ly tâm làm giàu ***Uranium*** của Iran.

Năm 2011: Các hệ thông máy tính bị tấn công bởi các ***Trojan horse***. Chúng có thể ẩn mình khỏi hệ điều hành bằng ***rootkits***. Nó âm thầm thực thi các công việc mà kẻ tấn công mong muốn, cụ thể lúc bấy giờ là lưu lại nhật kí bàn phím hoặc dùng như công cụ khai thác bitcoin.

Năm 2013: Bắt đầu kỉ nguyên của ***ransomware***. Cụ thể với:

- ***CryptoLocker*** là một Trojan horse khóa các tệp trên máy tính của người dùng, khiến họ phải trả tiền chuộc cho khóa giải mã.
- ***Gameover ZeuS*** đã sử dụng tính năng ghi bằng phím bấm để lấy cắp thông tin đăng nhập của người dùng từ các trang web giao dịch tài chính.

Năm 2014: Phần mềm độc hại được thiết kế để xâm nhập nhằm mục đích gián điệp, đánh cắp tài liệu và thông tin quan trọng.

Năm 2016: Tiếp tục đánh dấu sự tấn công mạnh mẽ của ***ransomware***. Cụ thể là sự xuất hiện và tấn công của ***Cerber*** và ***Locky***. Có thời điểm, Microsoft phát hiện nhiều PC doanh nghiệp bị nhiễm Cerber hơn bất kỳ dòng ***ransomware*** nào khác.

Năm 2017: Sự tấn công toàn cầu mạnh mẽ của ***ransomware*** mang tên ***WannaCry***. Nó tấn công các máy tính lớn và yêu cầu tiền chuộc hoặc bị mất hoàn toàn dữ liệu. Virus này đã gây ảnh hưởng đến hơn 150 quốc gia bao gồm các tập đoàn lớn, ngân hàng, bệnh viện,…, trong đó các máy tính lớn ở Nga, Trung Quốc, Anh và Mỹ đã phải bó tay. Nó yêu cầu nạn nhân trả tiền chuộc bằng bitcoin.

Với khả năng của các tay tin tặc, virus ngày nay có thể xâm nhập bằng cách bẻ gãy các rào an toàn của hệ điều hành hay chui vào các chỗ hở của các phần mềm, nhất là các chương trình thư điện tử, rồi từ đó lan tỏa khắp nơi theo các nối kết mạng hay qua thư điện tử. Do đó, việc truy tìm ra nguồn gốc phát tán virus sẽ càng khó hơn nhiều. Chính Microsoft, hãng phần mềm tạo ra các phần mềm phổ biến, cũng là 1 nạn nhân. Họ đã phải nghiên cứu, sửa chữa và phát hành rất nhiều các phần mềm nhằm sửa

các khiếm khuyết của phần mềm cũng như phát hành các cập nhật của gói dịch vụ (***service pack***) nhằm giảm hay vô hiệu hóa các tấn công của virus. Nhưng dĩ nhiên với các phần mềm có hàng triệu dòng mã nguồn thì mong ước chúng hoàn hảo theo ý nghĩa của sự an toàn chỉ có trong lý thuyết. Đây cũng là cơ hội cho các nhà sản xuất các loại phần mềm bảo vệ, sửa lỗi phát triển.

Trong tương lai không xa, virus sẽ có thêm các bước biến đổi khác, nó bao gồm mọi điểm mạnh sẵn có (***polymorphic***, ***sasser*** hay tấn công bằng nhiều cách thức, nhiều kiểu) và còn kết hợp với các thủ đoạn khác của phần mềm gián điệp (***spyware***). Đồng thời nó có thể tấn công vào nhiều hệ điều hành khác nhau chứ không nhất thiết nhắm vào 1 hệ điều hành độc nhất như trong trường hợp của Windows hiện giờ. Và có lẽ virus sẽ không hề (thậm chí là không cần) thay đổi phương thức tấn công: lợi dụng điểm yếu của máy tính cũng như chương trình.

1.4 Phân loại

1.4.1 Virus

Trong khoa học máy tính, virus máy tính virus tin học (thường được người sử dụng gọi tắt là virus) là những đoạn mã chương trình được thiết kế để thực hiện tối thiểu là 2 việc:

- Tự xen vào hoạt động hiện hành của máy tính một cách hợp lệ, để thực hiện tự nhân bản và những công việc theo chủ ý của lập trình viên. Sau khi kết thúc thực thi mã virus thì điều khiển được trả cho trình đang thực thi mà máy không bị "treo", trừ trường hợp virus cố ý treo máy.
- Tự sao chép chính nó, tức tự nhân bản, một cách hợp lệ lây nhiễm vào những tập tin (***file***) hay các vùng xác định (***boot***, ***FAT sector***) ở các thiết bị lưu trữ như đĩa cứng, đĩa mềm, thiết bị nhớ flash (phổ biến là USB),... thậm chí cả ***EPROM*** chính của máy.

Trước đây, virus thường được viết bởi một số người am hiểu về lập trình muốn chứng tỏ khả năng của mình nên thường virus có các hành động như: cho 1 chương trình không hoạt động đúng, xóa dữ liệu, làm hỏng ổ cứng,... hoặc gây ra những trò đùa khó chịu.

Những virus mới được viết trong thời gian gần đây không còn thực hiện các trò đùa hay sự phá hoại đối máy tính của nạn nhân bị lây nhiễm nữa, mà đa phần hướng đến việc lấy cắp các thông tin cá nhân nhạy cảm (các mã số thẻ tín dụng) mở cửa sau

cho tin tặc đột nhập chiếm quyền điều khiển hoặc các hành động khác nhằm có lợi cho người phát tán virus.

Chiếm trên 90% số virus đã được phát hiện là nhắm vào hệ thống sử dụng hệ điều hành họ Windows chỉ đơn giản bởi hệ điều hành này được sử dụng nhiều nhất trên thế giới. Do tính thông dụng của Windows nên các tin tặc thường tập trung hướng vào chúng nhiều hơn là các hệ điều hành khác. Cũng có quan điểm cho rằng Windows có tính bảo mật không tốt bằng các hệ điều hành khác (như Linux) nên có nhiều virus hơn, tuy nhiên nếu các hệ điều hành khác cũng thông dụng như Windows hoặc thị phần các hệ điều hành ngang bằng nhau thì cũng lượng virus xuất hiện có lẽ cũng tương đương nhau.

Lịch sử của virus máy tính[1] có thể khái quát qua các sự kiện và mốc thời gian sau đây:

- ***Năm 1949***: John von Neumann (1903-1957) phát triển nền tảng lý thuyết tự nhân bản của 1 chương trình cho máy tính.
- ***Vào cuối thập niên 1960 đầu thập niên 1970***: Xuất hiện trên các máy Univax 1108 1 chương trình gọi là "***Pervading Animal***" tự nó có thể nối với phần sau của các tập tin tự hành, lúc đó chưa có khái niệm về virus.
- ***Năm 1981:*** Các virus đầu tiên xuất hiện trong hệ điều hành của máy tính Apple II.
- ***Năm 1983***: Tại Đại học miền Nam California, tại Hoa Kỳ, Fred Cohen lần đầu đưa ra khái niệm "Virus máy tính" (***computer virus***) như định nghĩa ngày nay.
- ***Năm 1986***: Virus "***the Brain***", virus cho máy tính cá nhân (PC) đầu tiên, được tạo ra tại Pakistan bởi Basit và Amjad. Chương trình này nằm trong phần khởi động (boot sector) của 1 đĩa mềm 360Kb và nó sẽ lây nhiễm tất cả các ổ đĩa mềm. Đây là loại "stealth virus" đầu tiên.
- ***Cũng trong tháng 12 năm 1986***: virus cho DOS được khám phá ra là virus "***VirDem***". Nó có khả năng tự chép mã của mình vào các tệp tự thi hành (executable file) và phá hoại các máy tính ***VAX/VMS***.
- ***Năm 1987***: Virus đầu tiên tấn công vào command.com là virus "***Lehigh***".

[1] Theo Wikipedia về virus máy tính

- ***Năm 1988***: Virus ***Jerusalem*** tấn công đồng loạt các đại học và các công ty trong các quốc gia vào ngày thứ Sáu 13. Đây là loại virus hoạt động theo đồng hồ của máy tính (giống bom nổ chậm cài hàng loạt cho cùng 1 thời điểm).
- ***Tháng 11 năm 1988***: Robert Morris, 22 tuổi, chế ra worm chiếm cứ các máy tính của ***ARPANET***, làm liệt khoảng 6.000 máy. Morris bị phạt tù 3 năm và 10.000 dollar. Mặc dù vậy anh ta khai rằng chế ra virus vì "chán đời" (***boresome***).
- ***Năm 1990***: Chương trình thương mại chống virus đầu tiên ra đời bởi Norton[2].
- ***Năm 1991***: Virus đa hình (***polymorphic virus***) ra đời đầu tiên là virus "***Tequilla***". Loại này biết tự thay đổi hình thức của nó, gây ra sự khó khăn cho các chương trình chống virus.
- ***Năm 1994***: Những người thiếu kinh nghiệm, vì lòng tốt đã chuyển cho nhau 1 điện thư cảnh báo tất cả mọi người không mở tất cả những điện thư có cụm từ "***Good Times***" trong dòng bị chú (***subject line***) của chúng. Đây là một loại virus giả (***hoax virus***) đầu tiên xuất hiện trên các điện thư và lợi dụng vào "tinh thần trách nhiệm" của các người nhận được điện thư này để tạo ra sự luân chuyển.
- ***Năm 1995***: Virus văn bản (***macro virus***) đầu tiên xuất hiện trong các mã macro trong các tệp của Word và lan truyền qua rất nhiều máy. Loại virus này có thể làm hư hệ điều hành. ***Macro virus*** là loại virus viết ra bằng công cụ ***VBA***[3] (***Visual Basic for Applications***) và tùy theo khả năng, có thể lan nhiễm trong các ứng dụng văn phòng của Microsoft như Word, Excel, PowerPoint, OutLook,… Loại macro này, nổi tiếng có virus Baza và virus Laroux, xuất hiện năm 1996, có thể nằm trong cả Word hay Excel. Sau này, virus Melissa, năm 1997, tấn công hơn 1 triệu máy, lan truyền bởi 1 tệp đính kèm kiểu Word bằng cách đọc và gửi đến các địa chỉ của Outlook trong các máy đã bị nhiễm virus. Virus Tristate, năm 1999, có thể nằm trong các tệp Word, Excel và PowerPoint.

[2] Trung tâm điện toán Peter Norton, sau này được tập đoàn Symantec mua lại.
[3] VBA là một phần mềm công cụ, hỗ trợ tạo các ứng dụng sử dụng ngôn ngữ lập trình Visual Basic.

- ***Năm 2000***: ***Virus Love Bug***, còn có tên ***ILOVEYOU***, đánh lừa tính hiếu kì của mọi người. Đây là một loại macro virus. Đặc điểm là nó dùng đuôi tập tin dạng "***ILOVEYOU.txt.exe***", lợi dụng điểm yếu của Outlook thời bấy giờ: theo mặc định sẵn, đuôi dạng ***.exe*** sẽ tự động bị giấu đi. Ngoài ra, virus này còn có 1 đặc tính mới của spyware: nó tìm cách đọc tên và mã nhập của máy chủ và gửi về cho tay hắc đạo. Khi truy cứu ra thì đó là 1 sinh viên người Philippines. Tên này được tha bổng vì lúc đó Philippines chưa có luật trừng trị những người tạo ra virus cho máy tính.
- ***Năm 2002***: Tác giả của virus ***Melissa***, David L. Smith, bị xử 20 tháng tù.
- ***Năm 2003:*** Virus ***Slammer***, một loại worm lan truyền với vận tốc kỉ lục, truyền cho khoảng 75.000 máy tính trong 10 phút.
- ***Năm 2004***: Đánh dấu 1 thế hệ mới của virus là worm ***Sasser***. Với virus này thì người ta không cần phải mở đính kèm của điện thư mà chỉ cần mở lá thư là đủ cho nó xâm nhập vào máy. Cũng may là ***Sasser*** không hoàn toàn hủy hoại máy mà chỉ làm cho máy chủ trở nên chậm hơn và đôi khi nó làm máy tự khởi động trở lại. Tác giả của worm này cũng lập 1 kỉ lục khác: tay tin tặc nổi tiếng trẻ nhất, chỉ mới 18 tuổi, Sven Jaschan, người Đức. Tuy vậy, vì còn nhỏ tuổi, nên vào tháng 7/2005, tòa án Đức chỉ phạt anh này 3 năm tù treo và 30 giờ lao động công ích.
- ***Năm 2017***: Vụ tấn công của WannaCry vào ngày 12/5/2017 đang tiếp tục phát tán. ***WannaCry*** (tạm dịch là "Muốn khóc") còn được gọi là ***WannaDecryptor 2.0***, là 1 phần mềm độc hại mã độc tống tiền tự lan truyền trên các máy tính sử dụng Microsoft Windows. Vào tháng 5/2017, 1 cuộc tấn công không gian mạng quy mô lớn sử dụng nó được đưa ra, tính tới ngày 15/5 (3 ngày sau khi nó được biết đến) gây lây nhiễm trên 230.000 máy tính ở 150 quốc gia, yêu cầu thanh toán tiền chuộc từ 300 - 600 Euro bằng bitcoin với 20 ngôn ngữ (bao gồm tiếng Thái và tiếng Trung Quốc). Hiện thời người ta biết tới 5 tài khoản bitcoin của họ, đến nay chỉ có không hơn 130 người chịu trả tiền, thu nhập tối đa chỉ khoảng 30.000 Euro.

- ***Ngày nay***: Với khả năng của các tay tin tặc, virus có thể xâm nhập bằng cách bẻ gãy các rào an toàn của hệ điều hành hay chui vào các chỗ hở của các phần mềm nhất là các chương trình thư điện tử, rồi từ đó lan tỏa khắp nơi theo các nối kết mạng hay qua thư điện tử. Do đó, việc truy tìm ra nguồn gốc phát tán virus sẽ càng khó hơn nhiều. Chính Microsoft, hãng phần mềm tạo ra các phần mềm phổ biến, cũng là 1 nạn nhân. Họ đã phải nghiên cứu, sửa chữa và phát hành rất nhiều các phần mềm nhằm sửa các khiếm khuyết của phần mềm cũng như phát hành các cập nhật của gói dịch vụ (***service pack***) nhằm giảm hay vô hiệu hóa các tấn công của virus. Nhưng dĩ nhiên với các phần mềm có hàng triệu dòng mã nguồn thì mong ước chúng hoàn hảo theo ý nghĩa của sự an toàn chỉ có trong lý thuyết. Đây cũng là cơ hội cho các nhà sản xuất các loại phần mềm bảo vệ, sửa lỗi phát triển.
- ***Trong tương lai không xa***: Virus sẽ có thêm các bước biến đổi khác, nó bao gồm mọi điểm mạnh sẵn có (polymorphic, sasser hay tấn công bằng nhiều cách thức, nhiều kiểu) và còn kết hợp với các thủ đoạn khác của phần mềm gián điệp (***spyware***). Đồng thời nó có thể tấn công vào nhiều hệ điều hành khác nhau chứ không nhất thiết nhắm vào 1 hệ điều hành độc nhất như trong trường hợp của Windows hiện giờ. Và có lẽ virus sẽ không hề (thậm chí là không cần) thay đổi phương thức tấn công: lợi dụng điểm yếu của máy tính cũng như chương trình.

1.4.2 Sâu máy tính (Worm)

Sâu máy tính là một chương trình máy tính chứa phần mềm độc hại độc lập tự sao chép để lây lan sang các máy tính khác. Nó thường sử dụng mạng máy tính để tự lây lan, dựa vào các lỗi bảo mật trên máy tính mục tiêu để truy cập. Nó sẽ sử dụng máy này làm máy chủ để quét và lây nhiễm cho các máy tính khác. Khi các máy tính bị sâu mới này kiểm soát, sâu sẽ tiếp tục quét và lây nhiễm các máy tính khác sử dụng các máy tính này làm máy chủ và hành vi lây lan này sẽ tiếp tục. Sâu máy tính sử dụng phương pháp đệ quy để tự sao chép mà không cần chương trình chủ và tự phân phối dựa trên quy luật tăng trưởng theo cấp số nhân, sau đó điều khiển và lây nhiễm ngày càng nhiều máy tính trong thời gian ngắn. Sâu hầu như luôn gây ra ít nhất một số tác

hại cho mạng, ngay cả khi chỉ bằng cách tiêu tốn băng thông, trong khi virus hầu như luôn làm hỏng hoặc sửa đổi các tệp trên máy tính được nhắm mục tiêu.

Nhiều loại sâu được thiết kế chỉ để lây lan và không cố gắng thay đổi hệ thống mà chúng đi qua. Tuy nhiên, như sâu "***Morris***" và "***Mydoom***" đã chỉ ra, ngay cả những con sâu "không cần tải" này cũng có thể gây ra gián đoạn lớn bằng cách tăng lưu lượng mạng và các tác động ngoài ý muốn khác.

Đặc điểm:

- ***Độc lập:*** Virus máy tính thường yêu cầu một chương trình chủ . Virus viết mã của chính nó vào chương trình chủ. Khi chương trình chạy, chương trình vi rút đã viết được thực thi trước, gây nhiễm trùng và hư hỏng. Sâu không cần chương trình chủ vì nó là một chương trình độc lập hoặc đoạn mã độc lập. Do đó, nó không bị hạn chế bởi chương trình chủ mà có thể chạy độc lập và chủ động thực hiện các cuộc tấn công.
- ***Các cuộc tấn công khai thác***: Bởi vì sâu không bị giới hạn bởi chương trình chủ, sâu có thể lợi dụng các lỗ hổng hệ điều hành khác nhau để thực hiện các cuộc tấn công chủ động.
- ***Tính phức tạp***: Một số sâu được kết hợp với các tập lệnh trang web và được ẩn trong các trang HTML bằng ***VBScript, ActiveX*** và các công nghệ khác. Khi người dùng truy cập một trang web có chứa vi rút, vi rút sẽ tự động cư trú trong bộ nhớ và chờ được kích hoạt.
- ***Tính lây nhiễm***: Sâu có khả năng lây nhiễm cao hơn các loại virus truyền thống. Chúng không chỉ lây nhiễm vào các máy tính cục bộ mà còn lây nhiễm sang tất cả các máy chủ và máy khách trên mạng dựa trên máy tính cục bộ. Sâu có thể dễ dàng lây lan qua các thư mục chia sẻ, e-mail, các trang web độc hại và các máy chủ có nhiều lỗ hổng trong mạng.

1.4.3 Trojan Horse

Là một loại phần mềm ác tính. Không giống như virus, nó không có chức năng tự sao chép nhưng lại có chức năng hủy hoại tương tự virus. Một trong những thứ giăng bẫy của ***Trojan Horse*** là nó tự nhận là giúp cho máy của thân chủ chống lại các virus nhưng thay vì làm vậy nó quay ra đem virus vào máy.

Đặc điểm:

- Trojan horse là chương trình máy tính thường ẩn mình dưới dạng một chương trình hữu ích và có những chức năng mong muốn, hay ít nhất chúng trông như có các tính năng này. Một cách bí mật, nó lại tiến hành các thao tác khác không mong muốn. Những chức năng mong muốn chỉ là phần bề mặt giả tạo nhằm che giấu cho các thao tác này.
- Trong thực tế, nhiều ***Trojan horse*** chứa đựng các phần mềm gián điệp nhằm cho phép máy tính thân chủ bị điều khiển từ xa qua hệ thống mạng.
- Khác nhau căn bản với virus máy tính là ***Trojan Horse*** về mặt kỹ thuật chỉ là một phần mềm thông thường và không có ý nghĩa tự lan truyền. Các chương trình này chỉ lừa người dùng để tiến hành các thao tác khác mà thân chủ sẽ không tự nguyện cho phép tiến hành. Ngày nay, các ***Trojan horse*** đã được thêm vào đó các chức năng tự phân tán. Điều này đẩy khái niệm ***Trojan horse*** đến gần với khái niệm virus và chúng trở thành khó phân biệt.
- ***Trojan horse*** thường là các tệp khả thi trên Windows và do đó sẽ có các đuôi như là***.exe***, ***.com***, ***.scr***, ***.bat***, hay ***.pif***.

1.4.4 Phần mềm tống tiền (*Ransomware*)

Mã độc tống tiền hay Ransomware bao gồm nhiều lớp phần mềm ác ý với chức năng hạn chế truy cập đến hệ thống máy tính mà nó đã lây nhiễm, và đòi hỏi một khoản tiền cho người đã tạo ra malware đó nhằm mục đích xóa bỏ việc hạn chế truy cập mà nó đã tạo ra trước đó. Một vài dạng của ransomware mã hóa tệp tin, dữ liệu trên ổ đĩa cứng (nhằm tống tiền), trong khi một vài dạng khác thì đơn giản hơn, chúng khóa hệ thống lại và hiển thị một thống báo để thuyết phục người bị hại trả tiền.

Mã độc tống tiền thường lan truyền qua email, như các virus máy tính khác, đính kèm chẳng hạn file.zip mà khi mở file này, máy tính của người dùng sẽ bị kiểm soát. Lúc đó, mã độc quét toàn bộ ổ đĩa của máy tính và mã hoá các file bằng mã hoá khoá công khai (public key cryptography). Hầu hết các tệp tin quan trọng trên máy tính với định dạng ***.doc, .pdf, .xls, .jpg, .zip***… sẽ không thể mở được nữa. Để giải mã bắt buộc phải có khoá bí mật (private key) mà chỉ có kẻ phát tán mới có và nạn nhân sẽ nhận được thông báo trên desktop đòi tiền chuộc nếu muốn giải mã file.

Có hai loại ransomware chính:

- ***Locker Ransomware***: khóa máy tính hoặc thiết bị.
- ***Crypto Ransomware***: ngăn chặn truy cập vào tệp hoặc dữ liệu, thường thông qua mã hóa.

Một số loại ransomware tiêu biểu: RYUK, Sodinokibi, Phobos, Globelmposter, DoppelPaymer, Mamba, Snatch, Dharma, HiddenTear, Estemani, Rapid,…

Tên một số ransomware nguy hiểm nổi bật: CryptoLocker (2013), CryptoWall (2014), CTB-Locker (2014), TorrentLocker (2014), Bitcryptor và CoinVault (2015), TeslaCrypt (2015), Locky (2017), WannaCry (2017), GandCrab (2018)

1.4.5 Phần mềm gián điệp (Spyware)

Là loại phần mềm chuyên thu thập các thông tin từ các máy chủ (thông thường vì mục đích thương mại) qua mạng Internet mà không có sự nhận biết và cho phép của chủ máy. Một cách điển hình, ***spyware*** được cài đặt một cách bí mật như là một bộ phận kèm theo của các phần mềm miễn phí (***freeware***) và phần mềm chia sẻ (***shareware***) mà người ta có thể tải về từ Internet. Một khi đã cài đặt, spyware điều phối các hoạt động của máy chủ trên Internet và lặng lẽ chuyển các dữ liệu thông tin đến một máy khác (thường là của những hãng chuyên bán quảng cáo hoặc của các tin tặc).

Spyware "được" cài đặt một cách vô tội vạ khi mà người chủ máy chỉ muốn cài đặt phần mềm có chức năng hoàn toàn khác.

Ngoài các vấn đề nghiêm trọng về đạo đức và tự do cá nhân bị xâm phạm, spyware còn được sử dụng để đánh cắp tài nguyên của bộ nhớ (***memory resource***), ăn chặn băng thông khi nó gửi thông tin về cho các tin tặc qua Internet. Vì ***spyware*** dùng tài nguyên của bộ nhớ và của hệ thống, các ứng dụng chạy trong nền (***background***) có thể dẫn tới hư máy hay máy không ổn định.

Một số Spyware nổi tiếng:

- ***WildTangent***: phần mềm này được cài đặt thông qua ***American Online Instant Messenger*** (***AIM***). Theo ***AOL*** (***American Online***) thì nó cần dùng để tạo nối kết giữa các thành viên trong các trò chơi trên Internet. Một khi được cài đặt, nó sẽ lấy các thông tin về tên họ, số điện thoại, địa chỉ thư điện tử cũng như là tốc độ của CPU, các tham số của video card và ***DirectX***. Các thông tin này có thể bị chia sẻ cho các nơi khác chiếm dụng.

- ***Xupiter***: chương trình này sẽ tự nảy các bảng quảng cáo. Nó thêm các chỗ đánh dấu (***bookmark***) lên trên menu của chương trình duyệt và, nguy hại hơn, nó thay đổi cài đặt của trang chủ. ***Xupiter*** còn chuyển các thói quen xài Internet (***surfing***) về xupiter.com và do đó làm chậm máy. Ngoài ra nó còn đọc cả địa chỉ IP và các tin tức khác.
- ***DoubleClick***: dùng một tệp nhỏ gọi là cookies theo dõi hoạt động trực tuyến của bạn.
- ***WinWhatWhere***: thông báo cho người khác biết về các tổ hợp phím (keystroke) mà bạn gõ.
- ***Gator*** và ***eWallet***: ăn cắp tên, quốc gia, mã vùng bưu điện và nhiều thứ khác.

1.4.6 Botnet

Botnet thuật ngữ đầy đủ là "***Bots network***" dùng để chỉ một mạng lưới các máy tính bị chi phối bởi ai đó và bị điều khiển bởi một con máy tính khác từ xa. ***Botnet*** là một phần mềm độc hại, đa phần các máy tính đều bị nhiễm bởi một Bot nào đó mà chúng ta không thể nào phát hiện được.

Các máy tính đang bị nhiễm Botnet nôm na đều gọi là các "***Zombie***". Máy tính bị nhiễm sẽ bị chi phối bởi một ***Botmaster*** ở trên và điều khiển mọi hoạt động của máy tính đang dính mã độc làm cản trở hoạt động, gián đoán gây mất nhiều thời gian, giảm năng suất công việc của người dùng.

Cách chúng ta trở thành nạn nhân của nó giống như việc bị lây nhiễm ***malware***, và cách thức chiếm và sử dụng dữ liệu đánh cắp cũng chỉ với mục đích riêng của hacker.

Do đặc thù của ***Botnet*** là một mạng lưới các ***Bot***, nên Hacker dùng Botnet để tấn công DdoS, sau đó điều khiển tất cả các máy tính từ xa, có thể cùng lúc hàng ngàn, hàng nghìn chiếc máy tính cùng truy cập vào một website chỉ định, tạo ra lưu lượng quá tải trong website đó và gây tình trạng nghẽn mạng, treo máy. Botnet có thể vào bằng nhiều đường và ở nhiều dạng thù khác nhưng những mục đích của nó có thể là:

- ***Gửi mail spam:*** cách thức kiếm tiền phổ biến của các ***Spammer***. Hơn nữa các Botnet cũng tạo ra các web gian lận chèn bổ sung quảng cáo chạy trên nền web, người sử dụng tương tác click vào link quảng cáo sẽ tạo ra tiền cho các Hacker.

- ***Tấn công DdoS dùng Botnet***: Bot Herder sẽ lập trình ra một link website bất kỳ nào đó và điều khiển tất cả các máy tính là nạn nhân của Bot truy cập vào website đó, tạo ra tình trạng nghẽn mạng, dẫn đến không truy cập được nữa. Gửi hăm dọa, làm gian lận và tống tiền người dùng.
- Botnet đào tiền ảo từ một máy tính lớn, giúp chúng thu về tiền ảo như Bitcoin và các chi phí khác sẽ bị chịu bởi người dùng.
- Botnet cũng tạo và phát tán các loại virut, ***malware*** đến máy tính bạn và dùng nó tiếp tục lây lan sang các máy tính khác để tạo một mạng lưới ***Botnet*** lớn rộng để thu được nhiều lợi nhuận hơn.

Botnet có 2 loại cơ bản là ***DNS*** và ***IRC***. Mỗi loại có một chức năng riêng và ưu điểm riêng.

- ***DNS Bot***: dễ thực hiện và điều khiển đơn giản không quá cầu kỳ, được dùng để chạy Bot trên nền Web. Nhưng vẫn bị hạn chế trong việc trao đổi thông tin giữa ***Bot master*** và các ***Bot.***
- ***IRC Bot***: Giúp trao đổi thông tin giữa Bot Master và các Bot, điều khiển qua mạng chat IRC. Nhưng lại bị phụ thuộc vào các IRC và người quản trị Server.

1.4.7 Keylogger

Keylogger hay "trình theo dõi thao tác bàn phím" theo cách dịch ra tiếng Việt là một chương trình máy tính ban đầu được viết nhằm mục đích theo dõi và ghi lại mọi thao tác thực hiện trên bàn phím vào một tập tin nhật ký (***log***) để cho người cài đặt nó sử dụng. Vì chức năng mang tính vi phạm vào riêng tư của người khác này nên các trình ***keylogger*** được xếp vào nhóm các phần mềm gián điệp.

Về sau, khi ***keylogger*** phát triển cao hơn nó không những ghi lại thao tác bàn phím mà còn ghi lại cả các hình ảnh hiển thị trên màn hình (***screen***) bằng cách chụp màn hình (***screenshot***) hoặc quay phim (***screen-capture***) thậm chí còn ghi nhận cách con trỏ chuột trên máy tính di chuyển.

1.4.8 Tấn công giả mạo (Phishing)

Tấn công giả mạo (thuật ngữ gốc tiếng Anh: ***phishing***, biến thể từ fishing, nghĩa là câu cá, có thể ảnh hưởng từ chữ phreaking, nghĩa là sử dụng điện thoại người khác không trả phí, ám chỉ việc "nhử" người dùng tiết lộ thông tin mật), trong lĩnh vực bảo

mật máy tính, là một hành vi giả mạo ác ý nhằm lấy được các thông tin nhạy cảm như tên người dùng, mật khẩu và các chi tiết thẻ tín dụng bằng cách giả dạng thành một chủ thể tin cậy trong một giao dịch điện tử.

Thông thường, tin tặc sẽ giả mạo thành ngân hàng, trang web giao dịch trực tuyến, ví điện tử, các công ty thẻ tín dụng để lừa người dùng chia sẻ các thông tin nhạy cảm như: tài khoản & mật khẩu đăng nhập, mật khẩu giao dịch, thẻ tín dụng và các thông tin quý giá khác.

Phương thức tấn công này thường được tin tặc thực hiện thông qua email và tin nhắn. Người dùng khi mở email và click vào đường link giả mạo sẽ được yêu cầu đăng nhập. Nếu "mắc câu", tin tặc sẽ có được thông tin ngay tức khắc.

1.4.9 Rootkit

Rootkit là một bộ công cụ phần mềm do kẻ xâm nhập đưa vào máy tính nhằm mục đích cho phép quay lại xâm nhập máy tính đó và dùng nó cho các mục đích xấu mà không bị phát hiện, bộ công cụ này cho phép truy nhập vào hoạt động của máy tính ở mức căn bản nhất. Các mục đích của kẻ xâm nhập khi sử dụng ***rootkit*** bao gồm:

- Thu thập dữ liệu về máy tính (kể các máy tính khác trong cùng mạng) và những người sử dụng chúng (chẳng hạn mật khẩu và thông tin tài chính)
- Gây lỗi hoặc sai trong hoạt động của máy tính
- Tạo hoặc chuyển tiếp spam

Rootkit tạo đường truy nhập cho kẻ xâm nhập trở lại, việc này được thực hiện bằng cách cài đặt một cửa hậu (***backdoor*** - một phương pháp ẩn cho việc lấy quyền truy nhập máy tính). Cửa hậu này có thể là một daemon truy nhập từ xa, chẳng hạn một phiên bản đã được sửa chữa của telnetd hoặc sshd, được cấu hình để chạy trên một cổng không phải cổng mặc định mà các daemon này thường nghe. (daemon là một loại chương trình chạy ngầm, đợi được kích hoạt bởi một điều kiện hoặc một sự kiện cụ thể, ***daemon*** không chịu sự kiểm soát trực tiếp của người dùng).

Một rootkit được thiết kế tốt sẽ có khả năng che giấu hoặc xóa bỏ bất cứ dấu vết nào của việc nó được đưa vào máy tính, sự tồn tại và hoạt động của nó.

Các loại rootkit:

- ***Kernel mode rootkit***: được thiết kế để thay đổi chức năng của hệ điều hành. Loại rootkit này thường thêm code riêng của nó, đôi khi là cả cấu trúc dữ liệu vào các phần của kernel. Nhiều ***kernel mode rootkit*** khai thác

thực tế là các hệ điều hành cho phép driver thiết bị hoặc những mô-đun có thể load thực thi với cùng mức đặc quyền hệ thống như ***kernel***, vì vậy các ***rootkit*** được đóng gói dưới dạng ***driver*** hoặc mô-đun để tránh bị phần mềm diệt virus phát hiện.

- ***User mode rootkit*** (hoặc gọi là rootkit ứng dụng): thực thi theo cách tương tự như một chương trình người dùng thông thường. ***User mode rootkit*** có thể được khởi tạo như các chương trình thông thường khác trong quá trình khởi động hệ thống, hoặc chúng có thể được đưa vào hệ thống bởi một dropper (chương trình được thiết kế để cài đặt một số loại virus vào hệ thống muốn lây nhiễm). Phương pháp phụ thuộc vào hệ điều hành. Ví dụ, một ***rootkit*** Windows thường tập trung vào điều khiển chức năng cơ bản của các file ***DLL*** trong Windows, nhưng trong một hệ thống ***Unix***, toàn bộ ứng dụng có thể được thay thế hoàn toàn bởi rootkit.
- ***Bootkit*** (hoặc ***bootloader rootkit***): lây nhiễm vào ***Master Boot Record*** của ổ cứng hoặc thiết bị lưu trữ khác được kết nối với hệ thống đích. Bootkit có thể phá hoại quá trình khởi động và duy trì quyền kiểm soát hệ thống sau khi khởi động, do đó, đã được sử dụng thành công để tấn công những hệ thống sử dụng mã hóa toàn bộ ổ đĩa.
- ***Firmware rootkit*** tận dụng phần mềm được nhúng trong firmware hệ thống và tự cài đặt trong image firmware, được sử dụng bởi card mạng, ***BIOS***, router hoặc các thiết bị ngoại vi khác.

Hầu hết các loại nhiễm rootkit có thể tồn tại trong hệ thống suốt một thời gian dài, vì chúng tự cài đặt trên các thiết bị lưu trữ hệ thống vĩnh viễn, nhưng ***memory rootkit*** lại tự load vào bộ nhớ máy tính (***RAM***). Memory rootkit chỉ tồn tại cho đến khi ***RAM*** hệ thống bị xóa, thường là sau khi máy tính được khởi động lại.

1.5 Thống kê tác hại của mã độc

Các số liệu thống kê chính của Kaspersky từ tháng 5/2020 đến tháng 4/2021[4]:

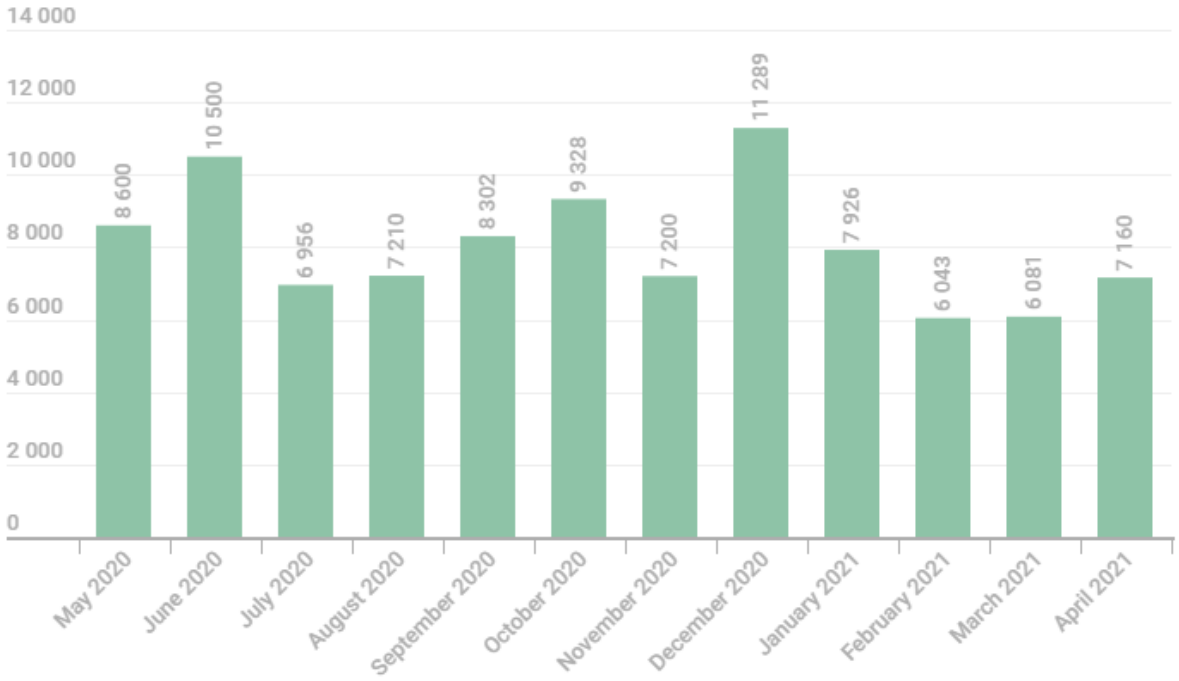
- 70% máy tính người dùng Internet ở Liên minh Châu Âu đã trải qua ít nhất một cuộc tấn công cấp Phần mềm độc hại .
- Tại EU, các giải pháp của Kaspersky đã chặn 115.452.157 cuộc tấn công web.
- 2.676.988 URL duy nhất đã được chương trình Chống vi rút trên web của Kaspersky công nhận là độc hại.
- 377.685 đối tượng độc hại duy nhất đã bị chương trình chống virus web chặn lại.
- Mã độc được lây nhiễm bởi phần mềm độc hại được thiết kế để ăn cắp tiền thông qua truy cập trực tuyến vào tài khoản ngân hàng đã được đăng nhập trên thiết bị của 79.315 người dùng.
- 56.877 người dùng riêng ở EU đã bị tấn công bởi ransomware.
- 132.656 người dùng riêng ở EU đã bị tấn công bởi các miners.
- 40% người dùng các giải pháp của Kaspersky ở EU đã gặp phải ít nhất một cuộc tấn công lừa đảo.
- 86.584.675 nỗ lực lừa đảo đã bị chặn bởi các giải pháp của Kaspersky riêng ở EU.

[4] Các số liệu thống kê dựa vào số lượng người dùng đã cho phép Kaspersky sử dụng dữ liệu theo dõi

1.5.1 Thống kê tấn công tài chính

Số lượng người dùng bị phần mềm độc hại tấn công tài chính

- Trong thời gian báo cáo, các giải pháp của Kaspersky đã chặn các nỗ lực khởi chạy một hoặc nhiều chương trình độc hại được thiết kế để lấy cắp tiền từ tài khoản ngân hàng trên máy tính của 79.315 người dùng.



Hình 1.1 Số lượng người dùng bị tấn công tài chính qua các tháng

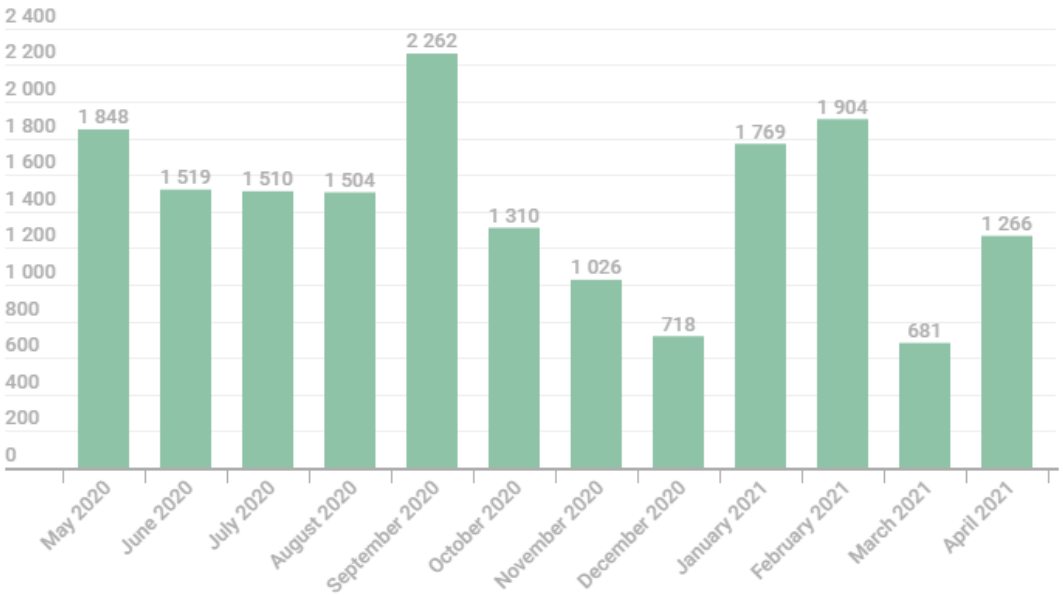
10 họ phần mềm độc hại tấn công tài chính hàng đầu được ghi nhận:

	Name	%*
1	Zbot	24.7
2	Nymaim	11.5
3	Danabot	9.9
4	Emotet	8.9
5	CliptoShuffler	7.7
6	BitStealer	5.6
7	SpyEyes	3.5
8	Gozi	3.4
9	Dridex	3.2
10	Trickster	1.9

Hình 1.2. Top 10 họ phần mềm tấn công tài chính

1.5.2 Các chương trình ransomware

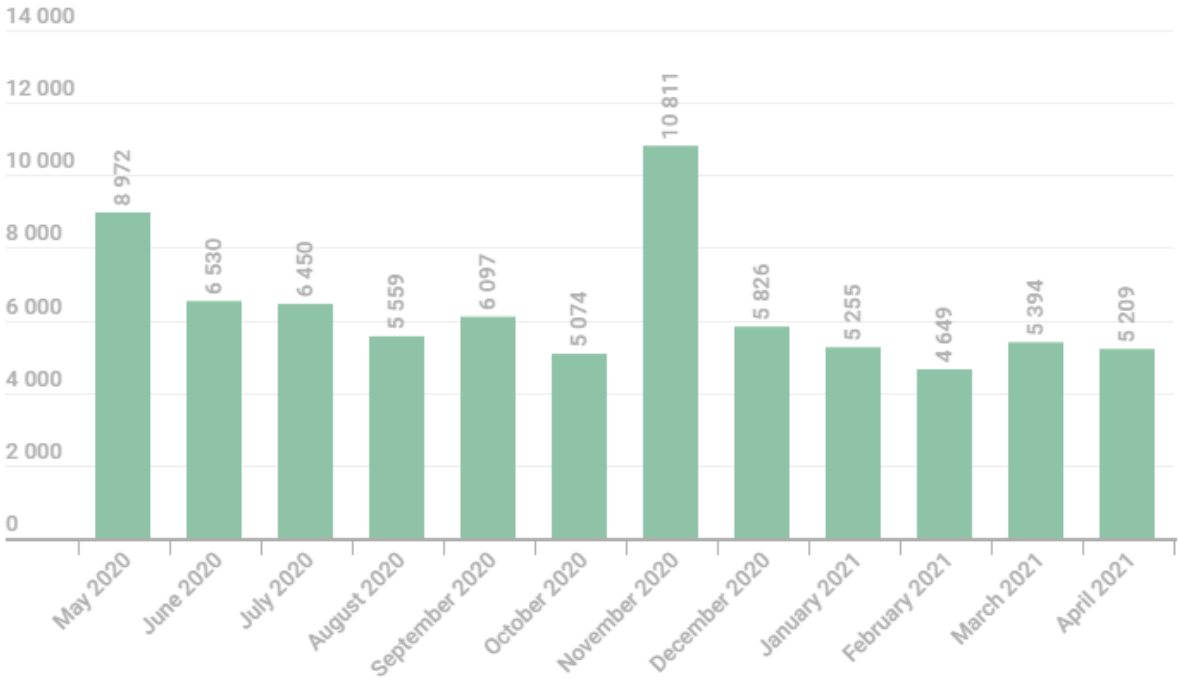
Trong khoảng thời gian thống kê, Kaspersky đã xác định được hơn 17.317 sửa đổi ransomware và phát hiện 25 họ mới. Hầu hết các mối đe dọa thuộc loại này đều được gán kết quả chung chung



Hình 1.3Thống kê về các chương trình ransomware

Số lượng người dùng bị tấn công bởi ransomware Trojan:

- Trong kỳ báo cáo, ransomware Trojan tấn công 56.877 người dùng, trong đó có 12.358 người dùng doanh nghiệp (không bao gồm SMBs) và 2,274 người sử dụng kết hợp với các doanh nghiệp vừa và nhỏ.



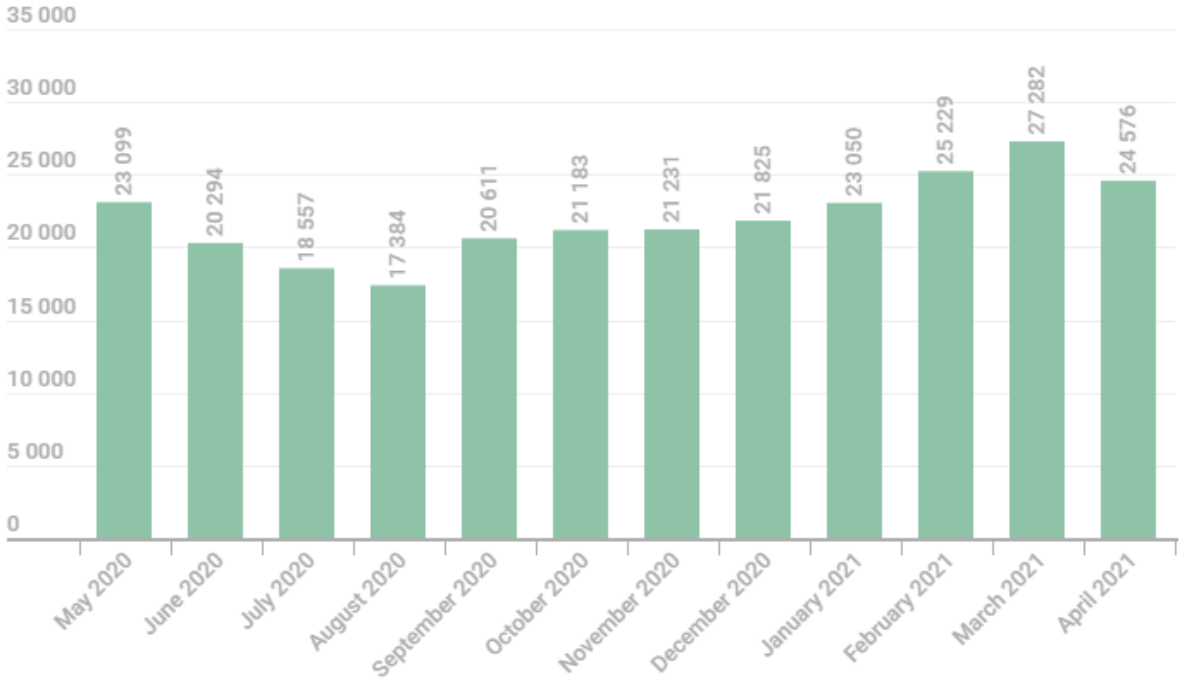
Hình 1.4 Số lượng người dùng bị tấn công ransomware Trojan

	Name	Verdict	%*
1	(generic verdict)	Trojan-Ransom.Win32.Gen	14.40
2	(generic verdict)	Trojan-Ransom.Win32.Agent	12.58
3	(generic verdict)	Trojan-Ransom.Win32.Encoder	10.80
4	(generic verdict)	Trojan-Ransom.Win32.Generic	5.94
5	Stop	Trojan-Ransom.Win32.Stop	3.87
6	WannaCry	Trojan-Ransom.Win32.Wanna	3.20
7	(generic verdict)	Trojan-Ransom.Win32.Crypmod	2.31
8	(generic verdict)	Trojan-Ransom.Win32.Crypren	2.30
9	REvil/Sodinokibi	Trojan-Ransom.Win32.Sodin	1.97
10	(generic verdict)	Trojan-Ransom.Win32.Cryptor	1.85

Hình 1.5 Top 10 họ ransomware Trojan

1.5.3 Miners

Trong khoảng thời gian thống kê, Kaspersky đã phát hiện thấy các nỗ lực cài đặt công cụ khai thác trên máy tính của 132.656 người dùng. Miners chiếm 0,53% tổng số cuộc tấn công và 10,31% trong tất cả các chương trình kiểu Risktool.



Hình 1.6 Số lượng người dùng bị tấn công "Miners"

Trong khoảng thời gian báo cáo, các sản phẩm của Kaspersky phát hiện Trojan.Win32.Miner.gen (họ chung) thường xuyên hơn các sản phẩm khác, chiếm 13,62% trên tổng số người dùng bị tấn công bởi miners, theo sau đó là hai họ khác là Trojan.Win32.Miner.bbb (8,67%) và Trojan.JS.Miner.m (2,84%).

1.5.4 Một số lỗ hổng phổ biến và dễ bị khai thác

Vào năm 2020, hầu hết các lỗ hổng đã được các nhà nghiên cứu phát hiện trước khi những kẻ tấn công có thể khai thác chúng, trong đó Kaspersky đã tìm thấy:

- ***CVE-2020-1380***: một lỗ hổng sử dụng sau khi sử dụng trong thành phần Jscript9 của trình duyệt Internet Explorer của Microsoft do không đủ kiểm tra trong quá trình tạo mã ***JIT*** được tối ưu hóa. Lỗ hổng này rất có thể đã được nhóm APT DarkHotel sử dụng ở giai đoạn đầu của quá trình xâm phạm hệ thống, sau đó trọng tải được phân phối bởi một khai thác bổ sung làm tăng đặc quyền trong hệ thống.

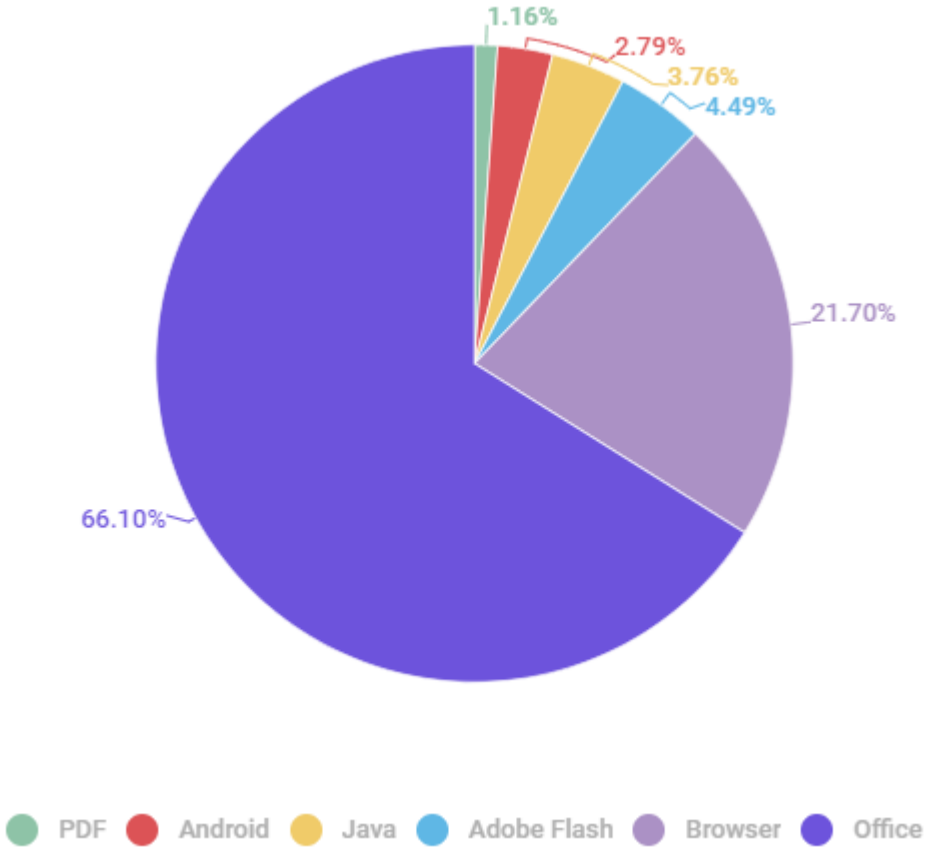
- ***CVE-2020-0986:*** trong thành phần Bộ đệm in / in ***GDI*** của hệ điều hành Windows của Microsoft, cho phép thao tác bộ nhớ quy trình để thực thi mã tùy ý trong ngữ cảnh của quy trình dịch vụ hệ thống. Việc khai thác lỗ hổng này cung cấp cho những kẻ tấn công khả năng vượt qua các Sandboxes , chẳng hạn như trong trình duyệt.

Quý đầu tiên của năm 2021 trở nên phong phú không chỉ ở các lỗ hổng nổi tiếng mà còn ở các lỗ hổng zero-day. Đặc biệt, cả các chuyên gia bảo mật công nghệ thông tin và tội phạm mạng đều tỏ ra rất quan tâm đến các lỗ hổng mới của Microsoft Exchange Server:

- ***CVE-2021-26855***: lỗ hổng bảo mật yêu cầu phía dịch vụ cho phép kẻ tấn công thực hiện yêu cầu máy chủ giả mạo và thực thi mã tùy ý (***RCE***);
- ***CVE-2021-26857:*** giải mã đối tượng không an toàn bằng dịch vụ Unified Messaging, có thể dẫn đến việc thực thi mã tùy ý ở phía máy chủ;
- ***CVE-2021-26858***: cho phép kẻ tấn công ghi dữ liệu vào các tệp máy chủ, điều này cũng có thể dẫn đến việc thực thi mã từ xa;
- ***CVE-2021-27065***: tương tự như CVE-2021-26858 , lỗ hổng này cho phép người dùng Microsoft Exchange được ủy quyền viết mã tùy ý vào các tệp hệ thống.

Các lỗ hổng này đã được tìm thấy trong tự nhiên và đã được sử dụng bởi APT và các nhóm ransomware.

Thêm một tập các lỗ hổng xuất hiện trong ***Infosec*** là một nhóm ba lỗi nghiêm trọng trong Nền tảng ***SolarWinds Orion*** phổ biến - ***CVE-2021-25274*** , ***CVE-2021-25275*** , ***CVE-2021-25276*** . Việc khai thác thành công bất kỳ lỗ hổng nào trong số chúng có thể gây ra sự lây nhiễm cho hệ thống nơi nền tảng được cài đặt (chủ yếu là PC doanh nghiệp và chính phủ).



Hình 1.7 Biểu đồ thống kê

Các cuộc tấn công mạng là phương pháp xâm nhập hệ thống phổ biến nhất và một phần đáng kể trong số chúng được tạo thành từ các cuộc tấn công mạnh mẽ vào các dịch vụ mạng khác nhau: RDP, Microsoft SQL Server, v.v. Ngoài ra, thời gian qua đã chứng minh rằng mọi thứ trong hệ điều hành Windows đều theo chu kỳ và hầu hết các lỗ hổng được phát hiện đều tồn tại trong các dịch vụ tương tự, chẳng hạn như trong trình điều khiển của ***SMB*** (SMBGhost, Giao thức mạng SMBBleed), ***DNS*** (SigRed) và ***ICMPv6*** (BadNeighbor). Hai lỗ hổng nghiêm trọng (***CVE-2020-0609***, ***CVE-2020-0610***) đã được tìm thấy trong dịch vụ ***Remote Desktop Gateway***. Một lỗ hổng thú vị, được đặt tên là Zerologon, cũng đã được phát hiện trong dịch vụ ***NetLogon***. Trong quý 1 năm 2021, các nhà nghiên cứu đã tìm thấy ba lỗ hổng mới trong mã ngăn xếp mạng Windows liên quan đến xử lý giao thức IPv4 / IPv6 là ***CVE-2021-24074*** , ***CVE-2021-24086*** và ***CVE-2021-24094***. Cuối cùng, mặc dù thực tế là các khai thác cho dòng EternalBlue và EternalRomance đã cũ, chúng vẫn được những kẻ tấn công sử dụng.

	Nhận định	% *
1	Monitor.OSX.HistGrabber.b	14,50
2	AdWare.OSX.Bnodlero.at	12.04
3	AdWare.OSX.Bnodlero.ay	11.42
4	AdWare.OSX.Bnodlero.ax	10,56
5	AdWare.OSX.Bnodlero.bg	9.18
6	Trojan-Downloader.OSX.Shlayer.a	8.06
7	AdWare.OSX.Pirrit.j	6.23
số 8	AdWare.OSX.Pirrit.ac	6,05
9	AdWare.OSX.Ketin.h	5.30
10	AdWare.OSX.Bnodlero.t	4,94
11	AdWare.OSX.Bnodlero.av	4,82
12	Trojan-Downloader.OSX.Agent.h	4,48
13	AdWare.OSX.Pirrit.o	4,35
14	AdWare.OSX.Cimpli.k	3,75
15	AdWare.OSX.Pirrit.gen	3,75
16	AdWare.OSX.Pirrit.aa	3.58
17	AdWare.OSX.Ketin.m	3,22
18	AdWare.OSX.Pirrit.q	3,20
19	AdWare.OSX.Ketin.l	3,13
20	AdWare.OSX.Spc.a	2,87

Hình 1.8 Top 20 mối đe dọa đối với macOS

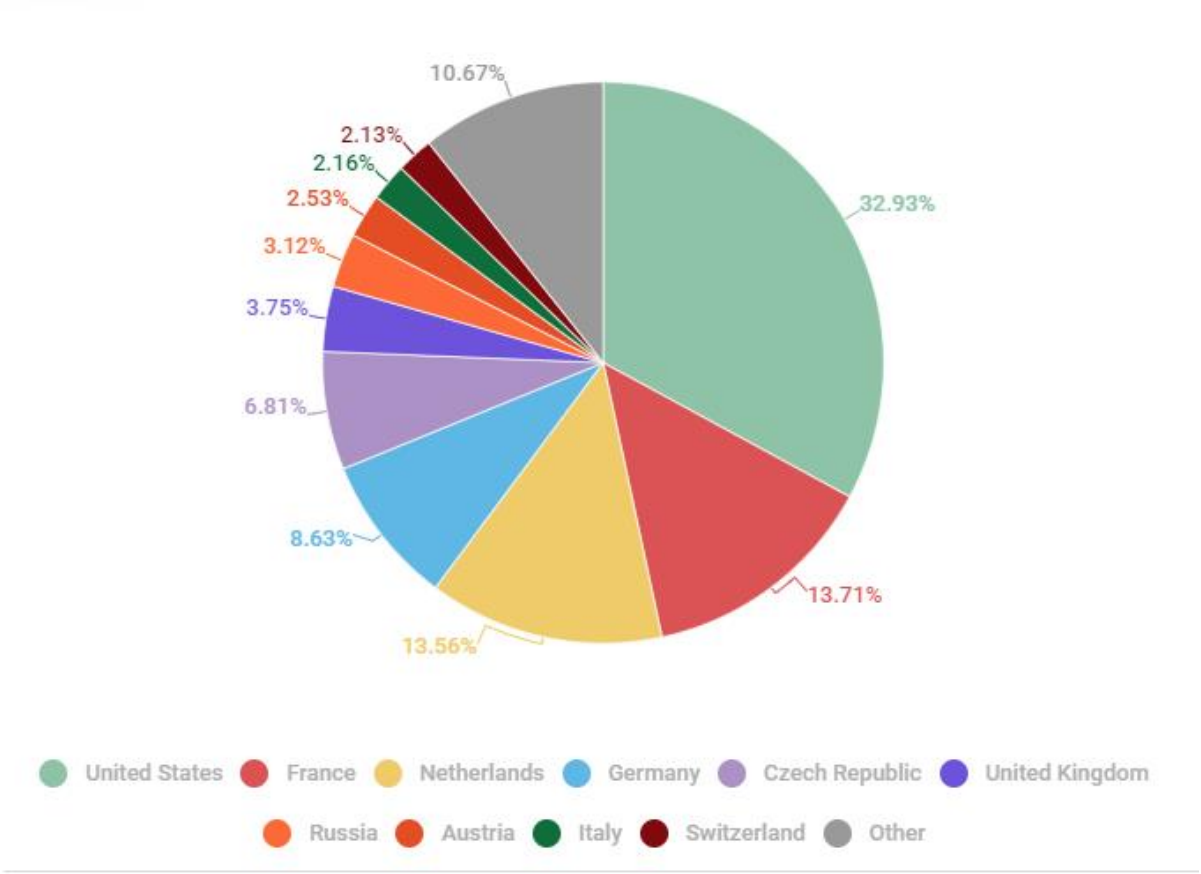
Riêng về các cuộc tấn công IoT, Kaspersky đã thống kê được hơn 80% (81,31%) các cuộc tấn công vào bẫy của Kaspersky được thực hiện bằng giao thức Telnet, và khoảng 18,69% được thực hiện bằng giao thức SSH.

	Nhận định	% *
1	Backdoor.Linux.Mirai.b	42,57
2	Trojan-Downloader.Linux.NyaDrop.b	20,96
3	Backdoor.Linux.Mirai.ba	9,79
4	Backdoor.Linux.Gafgyt.a	5,42
5	Backdoor.Linux.Gafgyt.a	2,74
6	Backdoor.Linux.Gafgyt.bj	1,44
7	Trojan-Downloader.Shell.Agent.p	1,31
số 8	Backdoor.Linux.Agent.bc	1,20
9	Backdoor.Linux.Mirai.cw	1,15
10	Backdoor.Linux.Mirai.cn	0,82

Hình 1.9. Top 10 phần mềm độc hại được tải vào honeypots

1.5.5 Tấn công qua tài nguyên web

Các giải pháp của Kaspersky ở EU đã chặn 115.452.157 cuộc tấn công được phát động từ các nguồn trực tuyến trên toàn cầu. Hơn nữa, 89,33% các nguồn tài nguyên này chỉ nằm ở 10 quốc gia.



Hình 1.10. 10 quốc gia là nguồn tấn công tài nguyên web ở EU

Trung bình, 13,70% máy tính người dùng Internet ở EU đã trải qua ít nhất một cuộc tấn công bằng phần mềm độc hại trong thời gian báo cáo.

Cũng trong khoảng thời gian này, phần mềm chống virus web của Kaspersky đã phát hiện 377.685 đối tượng độc hại (tập lệnh, khai thác, tệp thực thi, v.v.), cũng như 2.676.988 URL độc hại mà chương trình chống virus trên web đã được kích hoạt. Dựa trên dữ liệu thu thập được, Kaspersky đã xác định được 20 chương trình độc hại được sử dụng nhiều nhất trong các cuộc tấn công trực tuyến vào máy tính của người dùng.

	Verdict*	%**
1	Blocked	49.22
2	Trojan.Script.Generic	12.52
3	Hoax.HTML.FraudLoad.m	8.38
4	Trojan.PDF.Badur.gen	2.46
5	Trojan.Script.Agent.dc	2.16
6	Trojan.Multi.Preqw.gen	2.11
7	Trojan-Downloader.Script.Generic	1.99
8	Trojan.Script.Miner.gen	1.56
9	Exploit.MSOffice.CVE-2017-11882.gen	1.02
10	Trojan-PSW.Script.Generic	0.91
11	DangerousObject.Multi.Generic	0.74
12	Trojan.BAT.Miner.gen	0.74
13	Trojan.MSOffice.SAgent.gen	0.60
14	Trojan.Script.SAgent.gen	0.50
15	Trojan-Downloader.MSOffice.SLoad.gen	0.47
16	Trojan-Downloader.Win32.Upatre.pef	0.33
17	Trojan-Downloader.JS.Inor.a	0.30
18	Trojan-Downloader.MSWord.Agent.btl	0.30
19	Hoax.Script.Dating.gen	0.27
20	Trojan-Downloader.JS.SLoad.gen	0.27

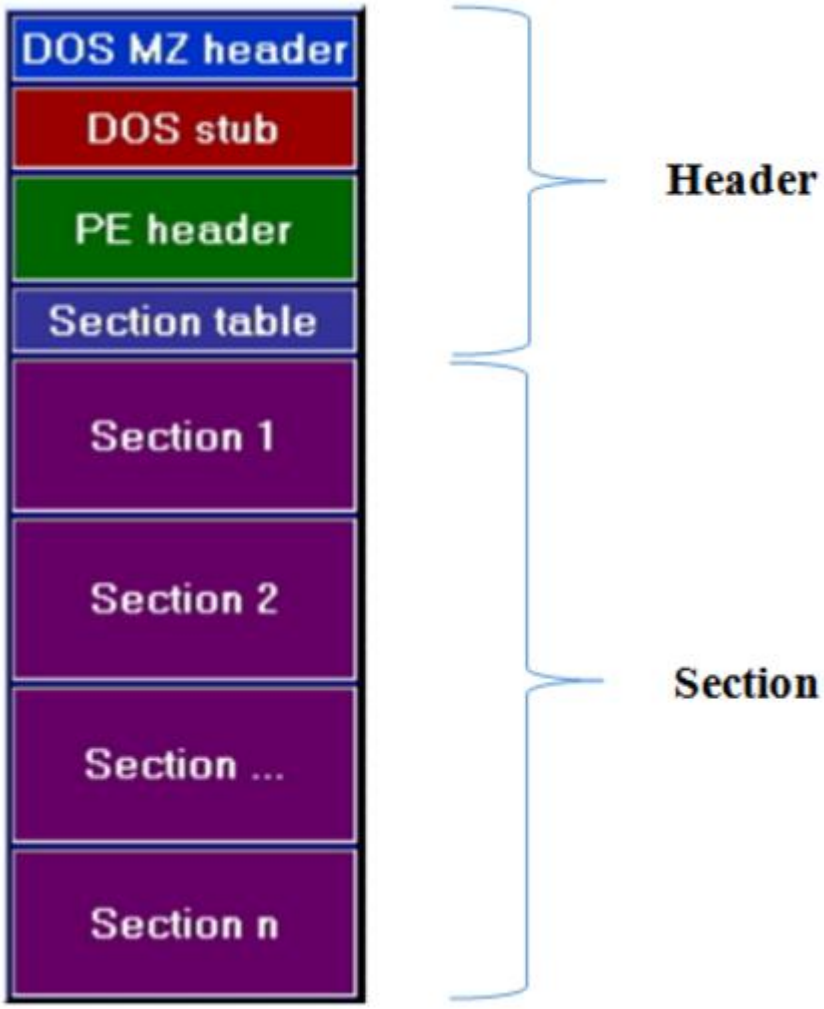
Hình 1.11 Top 20 mã độc được sử dụng để tấn công web

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1 PE File

PE File Format (***Portable Executable File Format***): là định dạng file riêng của Win32. Tất cả các file có thể thực thi được trên ***Win32*** như: *.EXE, *.DLL (32 bit),*.COM,*.NET, *.CPL,… đều là định dạng PE. Ngoại trừ các tập tin VxDs và *.DLL (16 bit). Ngay cả NT's kernel mode driver cũng sử dụng định dạng tệp PE. Do đó nghiên cứu định dạng tập tin PE cung cấp cho bạn cái nhìn sâu sắc về cấu trúc của Windows.

Cấu trúc cơ bản của một PE File:



Hình 2.1. Cấu trúc cơ bản của một PE File

PE file được chia làm 2 phần ***Header*** và ***Section***, trong đó Header dùng để lưu các giá trị định dang file và các offset của các section trong phần Section. Sau đâu chúng ta đi tìm hiểu về phần Header của file.

2.1.1 DOS MZ Header

Tất cả các PE file đếu bắt đầu bằng một ***DOS MZ Header*** đơn giản. Nó chiếm 64 bytes đầu tiên. Vùng này được dùng trong trường hợp chương trình chạy trên nền DOS, hệ điều hành DOS nhận biết đây là một file thực thi hợp lệ và sẽ thực thi nội dung trong phần ***DOS STUB***.

```
struct _IMAGE_DOS_HEADER {
0x00   WORD e_magic;
0x02   WORD e_cblp;
0x04   WORD e_cp;
0x06   WORD e_crlc;
0x08   WORD e_cparhdr;
0x0a   WORD e_minalloc;
0x0c   WORD e_maxalloc;
0x0e   WORD e_ss;
0x10   WORD e_sp;
0x12   WORD e_csum;
0x14   WORD e_ip;
0x16   WORD e_cs;
0x18   WORD e_lfarlc;
0x1a   WORD e_ovno;
0x1c   WORD e_res[4];
0x24   WORD e_oemid;
0x26   WORD e_oeminfo;
0x28   WORD e_res2[10];
0x3c   DWORD e_lfanew;
};
```

Hình 2.2. DOS MZ Header

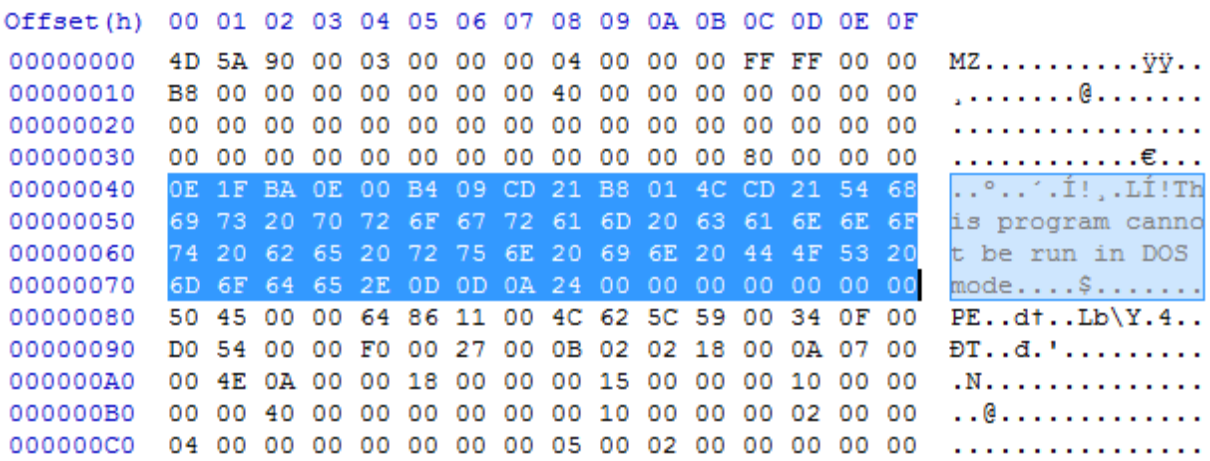
Trong đó chúng ta cần quan tâm tới hai trường:

- ***e_magic***: là chữ ký của PE file, giá trị: 4Dh, 5Ah (Ký tự "MZ", tên của người sáng lập MS-DOS: Mark Zbikowsky). Giá trị này đánh dấu một DOS Header hợp lệ và được phép thực thi tiếp.
- ***e_lfanew***: là một ***DWORD***[5] nằm ở cuối cùng của DOS Header, là trường chứa offset của PE Header so với vị trí đầu file.

[5] là đơn vị dữ liệu tự nhiên

2.1.2 DOS STUB

DOS Stub chỉ là một chương trình DOS EXE nhỏ hiển thị một thông báo lỗi, là phần để tương thích với Windows 16bit.



Hình 2.3. DOS STUB minh họa

Ví dụ như trong hình minh họa trên, thông báo sẽ hiện ra như sau: "This is program cannot be run in DOS mode"

2.1.3 PE Header

PE Header thực chất là cấu trúc IMAGE_NT_HEADERS bao gồm các thông tin cần thiết cho quá trình loader load file lên bộ nhớ. Cấu trúc này gồm 3 phần được định nghĩa trong windows.inc

```
struct _IMAGE_NT_HEADERS {
0x00  DWORD Signature;
0x04  _IMAGE_FILE_HEADER FileHeader;
0x18  _IMAGE_OPTIONAL_HEADER OptionalHeader;
};
```

Hình 2.4. Thành phần của PE Header

a) ***Signature***: Là 1 ***DWORD*** bắt đầu PE Header chứa chữ ký PE: 50h, 45h, 00h, 00.

b) ***FILE_HEADER***: Bao gồm 20 bytes tiếp theo của PE Header, phần này chứa thông tin về sơ đồ bố trí vật lý và các đặc tính của file.

```
struct _IMAGE_FILE_HEADER {
0x00   WORD Machine;
0x02   WORD NumberOfSections;
0x04   DWORD TimeDateStamp;
0x08   DWORD PointerToSymbolTable;
0x0c   DWORD NumberOfSymbols;
0x10   WORD SizeOfOptionalHeader;
0x12   WORD Characteristics;
};
```

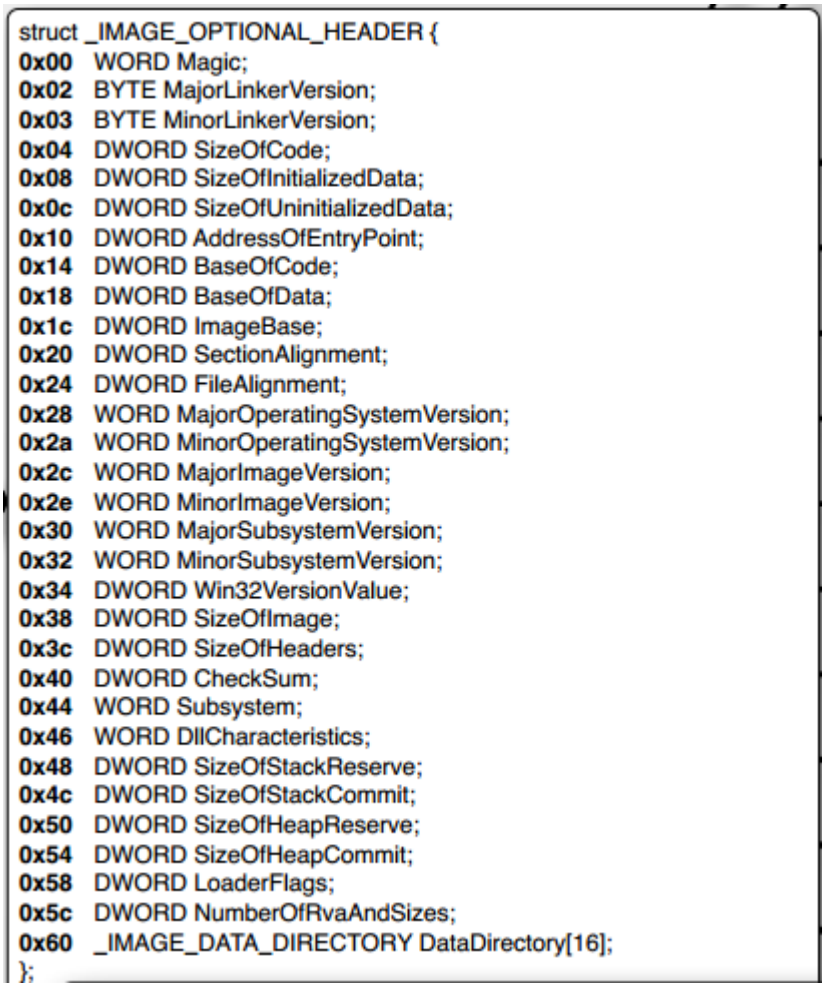
Hình 2.5. File Header

Trong đó:

- ***Machine***: giá trị xác định PE File này được biên dịch cho dòng máy nào
- ***NumberOfSections***: đây là trường chứa số section của file. Nếu muốn thêm/xoá section trong PE file, ta cần thay đổi tương ứng trường này.
- ***Characteristics***: là bit cờ, xác định định dạng PE File.

c) *OPTIONAL_HEADER*:

- Bao gồm 224 bytes tiếp theo sau ***FILE_HEADER***. Cấu trúc này được định nghĩa trong ***windows.inc***, đây là phần chứa thông tin về sơ đồ logic trong PE File.
- Dưới đây là danh sách các trường trong cấu trúc này, đồng thời sẽ đưa ra một số chỉ dẫn về thông tin của một số trường cần quan tâm khi muốn chỉnh sửa file.



Hình 2.6. Các thành phần của Optional File

Trong đó, các thành phần được mô tả như sau:

- ***Magic*** (2 bytes): xác định là file 32 bit (0B 01) hay 64 bit (0B 20)
- ***AddressOfEntryPoint*** (4bytes): chứa địa chỉ ảo tương đối (RVA) của câu lệnh đầu tiên sẽ được thực thi khi chương trình PE Loader sẵn sàng để chạy ***PE File*** (.text). Nếu muốn chương trình bắt đầu từ một địa chỉ khác (để thực thi câu lệnh với mục đích khác) thì cần thay đổi địa chỉ này về địa chỉ tương đối của câu lệnh muốn thực thi.
- ***ImageBase***: địa chỉ nạp được ưu tiên cho PE File.
- ***Section Alignment***: Phần liên kết của các Section trong bộ nhớ. Tức là một section luôn luôn được bắt đầu bằng bội số của sectionAlignment.
- ***File Alignment***: Phần liên kết của các Section trong File. Tương tự như ***SectionAlignment*** nhưng áp dụng với file.
- ***SizeOfImage***: Toàn bộ kích thước của Pe image trong bộ nhớ, là tổng của tất cả các headers và sections được liên kết tới Section Alignment

- ***SizeOfHeaders***: Kích thước của tất cả các headers cộng với section table bằng kích thước file trừ đi tổng kích thước của các section trong file.
- ***Data Directory***: là một mảng gồm 16 phần tử, trong đó mỗi phần liên quan đến một cấu trúc dữ liệu quan trọng trong PE File.

2.1.4 SECTION TABLE

Section Table là thành phần ngày sau PE Header, bao gồm một mảng những cấu trúc ***IMAGE_SECTION_HEADER***, mỗi phần tử chứa thông tin về một section trong PE file.

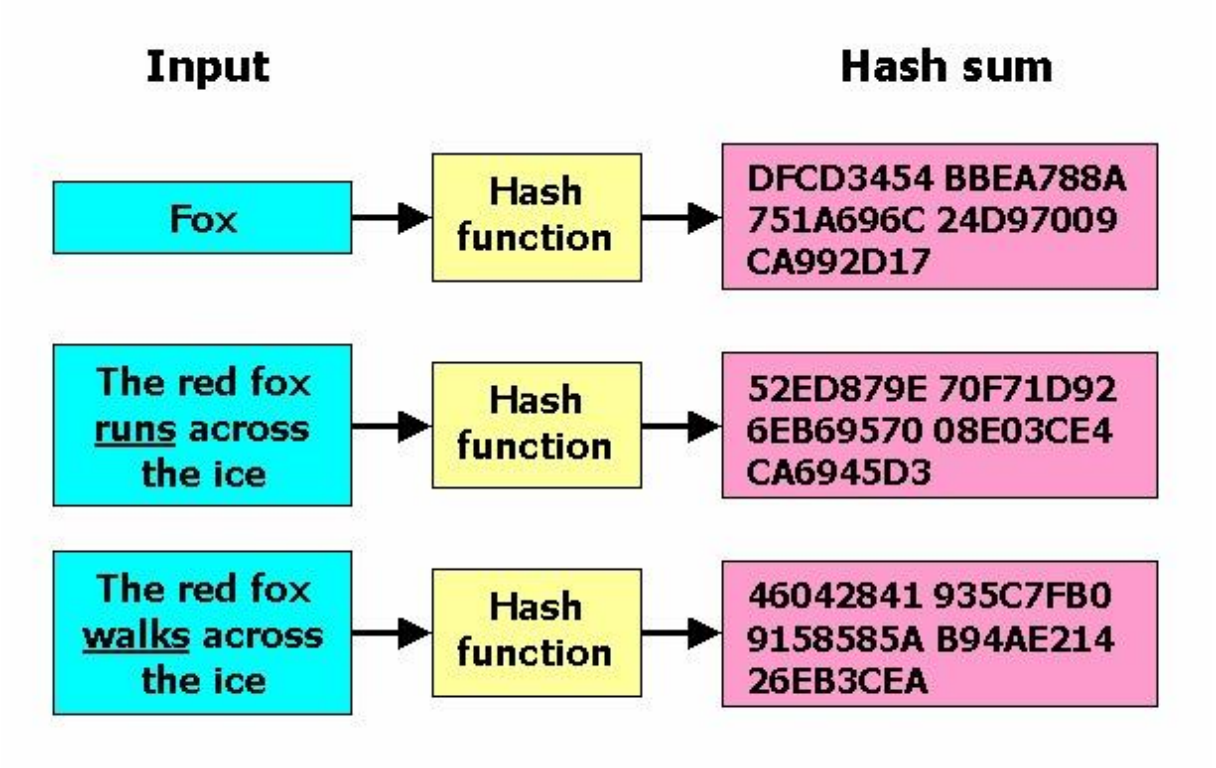
```
typedef struct _IMAGE_SECTION_HEADER {
0x00   BYTE  Name[IMAGE_SIZEOF_SHORT_NAME];
       union {
0x08           DWORD PhysicalAddress;
0x08           DWORD VirtualSize;
       } Misc;
0x0c   DWORD VirtualAddress;
0x10   DWORD SizeOfRawData;
0x14   DWORD PointerToRawData;
0x18   DWORD PointerToRelocations;
0x1c   DWORD PointerToLinenumbers;
0x20   WORD  NumberOfRelocations;
0x22   WORD  NumberOfLinenumbers;
0x24   DWORD Characteristics;
};
```

Hình 2.7. Section Table

Thông tin về một số trường quan trọng:

- ***VirtualSize***: Kích thước thật sự của dữ liệu trên section tính theo byte, giá trị này có thể nhỏ hơn kích thước trên ổ đĩa (SizeOfRawData).
- ***VirtualAddress***: RVA của section, là giá trị để ánh xạ khi section được load lên bộ nhớ.
- ***SizeOfRawData***: Kích thước section data trên ổ đĩa.
- ***PointerToRawData***: là offset từ vị trí đầu file tới section data.
- ***Characteristics***: đặc tính của section: thực thi, dữ liệu khởi tạo …

2.2 Hash



Hình 2.8. Hash

2.2.1 Hash (hàm băm) là gì ?

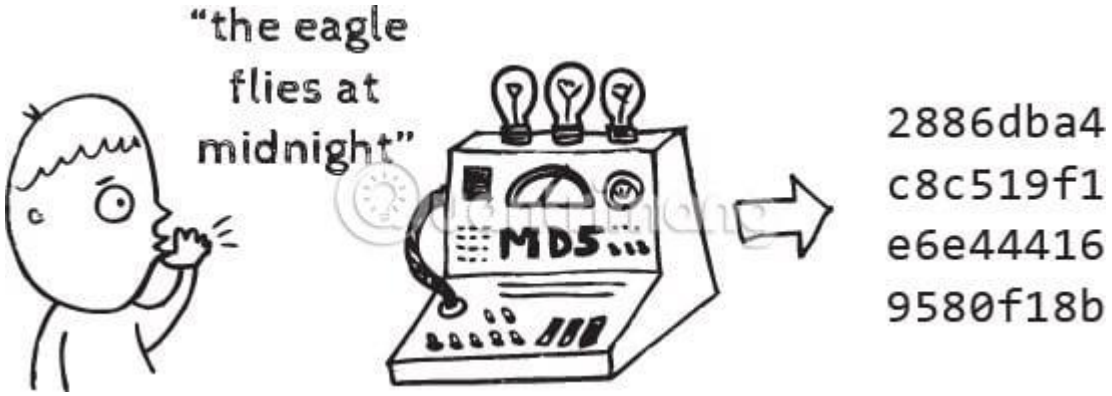
Hashing là quá trình biến đầu vào là dữ liệu có kích thước, độ dài bất kỳ thông quan sử dụng những thuật toán băm hoặc hàm băm để tạo thành đầu ra có độ dài nhất định và duy nhất đại diện cho dữ liệu đầu vào. (***Hash function***).

Giả dụ, bạn tải một video trên Youtube về, sau đó cho nó chạy qua hàm băm có tên ***MD5*** sẽ trả về một chuỗi dài 32 ký tự, hoặc bạn tải một bức ảnh trên mạng về, cho chạy qua hàm ***MD5***, thứ bạn nhận được vẫn là một chuỗi dài 32 ký tự. Thậm chí, nếu như bạn cho chạy từ "Apple" hay một file nào đó qua hàm hash ***MD5*** kia, kết quả sẽ vẫn là một chuỗi có 32 ký tự. Những thuật toán băm khác cũng hoạt động tương tự như vậy, bạn cho bất kỳ thứ gì vào hàm, đầu ra sẽ luôn là một chuỗi có độ dài nhất định.

2.2.2 Hàm băm mật mã:

Như tên gọi của nó vậy, những hàm băm như vậy được sử dụng vào mục đích mã hóa dữ liệu. Những hàm băm mật mã cũng giống như những hàm băm thông thường, nhưng mang trong mình một số đặc điểm khác, quan trọng nhất là không thể đảo

ngược. Điều này có nghĩa là khi bạn có trong tay giá trị sau khi băm, bạn không thể biết giá trị ban đầu là gì.



Hình 2.9. Minh họa hàm băm mật mã

2.2.3 Một số hàm băm phổ biến

a) MD5

MD5 được Ronald Rivest thiết kế vào năm 1991 để thay thế hàm băm ***MD4*** trước đó và được đưa thành tiêu chuẩn vào năm 1992 trong ***RFC 1321***. MD5 tạo ra một bản tóm tắt có kích thước 128 bit (16 byte). Tuy nhiên, đến đầu những năm 2000 thì hàm băm MD5 trở lên không an toàn trước sức mạnh tính toán của các hệ thống tính toán thế hệ mới. Với sức mạnh tính toàn và sự phát triển của công nghệ thám mã thời gian gần đây, chúng ta có thể tính toán các va chạm trong MD5 với độ phức tạp 2^{21}, phép toán chỉ trong vòng vài giây khiến thuật toán không phù hợp với hầu hết các trường hợp sử dụng trong thực tế.

b) SHA-*1*

SHA-1, viết tắt của ***Secure Hash Algorithm***, được phát triển như một phần của dự án Capstone của Chính phủ Hoa Kỳ. Phiên bản đầu tiên, thường được gọi là SHA-0 được xuất bản năm 1993 với tiêu đề ***Secure Hash Standard***, ***FIPS PUB 180***, bởi ***NIST*** (Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ). Nó đã bị ***NSA*** rút lại ngay sau khi xuất bản và được thay thế bởi phiên bản sửa đổi, được xuất bản vào năm 1995 trong ***FIPS PUB 180-1*** và thường được đặt tên là SHA-1. SHA-1 tạo ra bản tóm tắt có kích thước 160 bit (20 byte). Các va chạm chống lại thuật toán SHA-1 đầy đủ có thể được tạo ra bằng cách sử dụng tấn công phá vỡ. Do đó, hàm băm này cho đến nay được coi là không đủ an toàn.

c) SHA-2

SHA-2 là một tập hợp các hàm băm mật mã được thiết kế bởi Cơ quan an ninh quốc gia Hoa Kỳ (NSA), được xuất bản lần đầu tiên vào năm 2001. Chúng được xây dựng bằng cấu trúc ***Merkle–Damgard***, chức năng nén một chiều của nó được xây dựng bằng cấu trúc ***Davies–Meyer*** từ một hệ mật mã khối chuyên dụng.

2.2.4 Ý nghĩa của hàm băm

a) Kiểm tra tính toàn vẹn của tập tin

Như đặc điểm của hàm băm, cùng một giá trị sẽ cho ra cùng một giá trị băm. Vậy nên ta có thể đối chiếu tệp tin ta tải trên mạng về với bản gốc bằng cách so sánh giá trị băm của chúng với nhau. Nếu chúng có chung giá trị băm tức là tệp tin của bạn trùng với bản gốc, nếu không tệp tin của bạn đã bị sửa đổi hoặc bị hỏng. Một số trường hợp, tệp tin của bạn tải về bị can thiệp bởi bên thứ ba trước khi đến thiết bị của bạn và chúng có thể cài mã độc vào tệp tin đó. Việc kiểm tra giá trị băm giúp đảm bảo tệp tin của bạn an toàn.

b) Xác minh mật khẩu

Có một điều rất hay bạn nên biết, trong những thiết kế cơ sở dữ liệu hiện đại, thứ lưu trong đó không phải mật khẩu của bạn dưới dạng văn bản đơn thuần mà là giá trị hash của chúng. Khi bạn nhập mật khẩu, mật khẩu của bạn sẽ được chạy qua hàm hash, sau đó sẽ được so sánh với giá trị băm trong cơ sở dữ liệu để quyết định bạn có được chứng thực để sử dụng dịch vụ không. Điều này làm giảm đáng kể thiệt hại khi cơ sở dữ liệu bị tấn công, khi những gì bị lộ ra ngoài là những giá trị băm chứ không phải mật khẩu của bạn. Để an toàn hơn, hệ thống còn thêm giá trị muối (***salt***) vào mật khẩu gốc của bạn, rồi cho chạy qua hàm băm, sau đó mới lưu vào cơ sở dữ liệu. Vậy nên kể cả khi giá trị băm của mật khẩu bạn bị lộ và bị giải mã, kẻ tấn công vẫn chưa thể có được mật khẩu thực sự của bạn do nó đã được thêm vào giá trị "***salt***".

2.2.5 Virus signature

Một chữ ký virus là một chuỗi các ký tự hoặc số tạo nên chữ ký rằng các chương trình chống virus được thiết kế để phát hiện. Một chữ ký có thể chứa nhiều chữ ký virus, đó là các thuật toán hoặc băm mà nhận ra duy nhất một loại virus cụ thể. Một số lượng lớn các virus có thể chia sẻ một chữ ký duy nhất, cho phép một máy quét virus để phát hiện virus nó chưa bao giờ thấy trước đây.

Generic hoặc tìm ***Heuristic*** là hai loại quét mà phần mềm chống virus sử dụng khi tìm kiếm chữ ký virus. Generic là không hiệu quả như Heuristic quét bởi vì nó bỏ qua việc xác định vị trí các chữ ký virus mới, nhưng Generic lại tốt hơn trong việc tìm virus mới đã được phát triển từ dòng virus hiện có. Phương pháp phát hiện Heuristic bao gồm hơn 250.000 chữ ký virus mới và hiệu quả nhất cho việc định vị các chữ ký virus mới. Chữ ký mới (Signature) được tạo ra mỗi lần một loại virus mới tạo ra để họ có thể phát hiện virus trong quá trình quét khi virus mới được sinh ra mà chương trình không thể phát hiện được. Khi các nhà cung cấp chống virus đã kiểm tra chữ ký mới, các nhà cung cấp sẽ gửi nó ra dưới hình thức một bản cập nhật chữ ký để nó tương quan với phần mềm quét virus của người dùng. Điều này cũng có thể bao gồm thay thế chữ ký, hoặc loại bỏ các chữ ký trước khi họ không còn có thể quét đúng cách cho virus chữ ký sửa đổi. Đó là lý do tại sao các chuyên gia máy tính tư vấn cho người sử dụng để luôn cập nhật máy quét chống virus của họ khi các nhà cung cấp gửi các gói tin.

2.3 Windows API

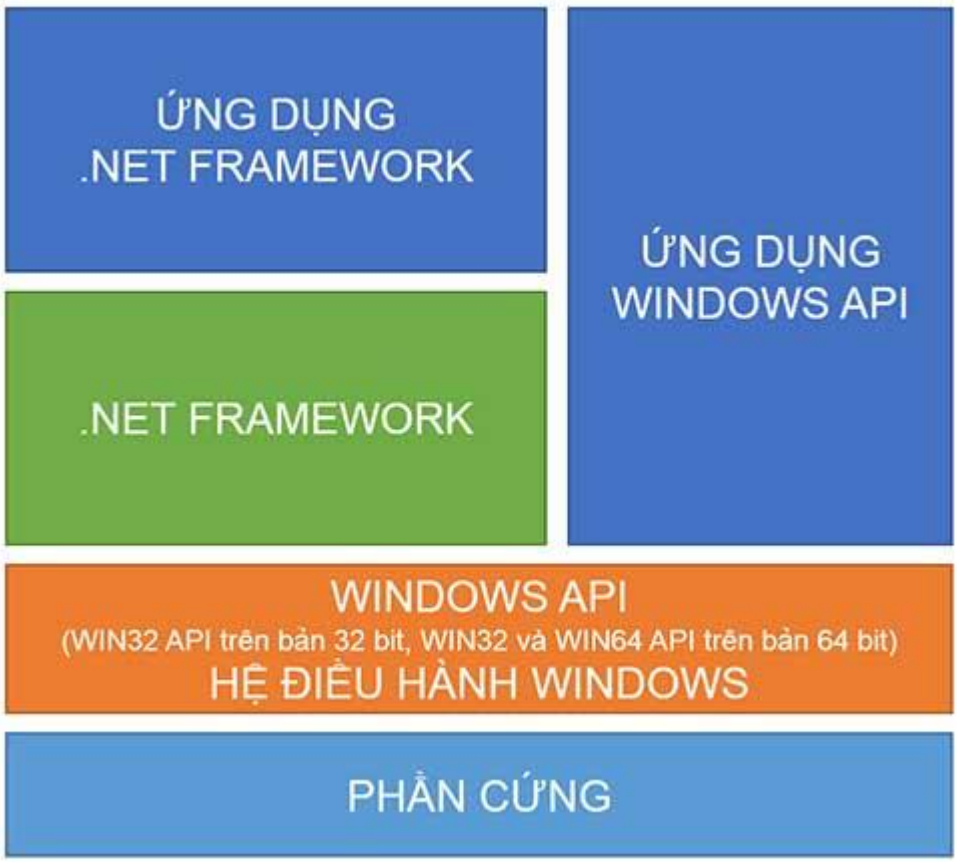
2.3.1 Tổng quan về Windows API

Windows API[6] là các hàm thư viện và các định nghĩa khác (***struct***, ***enum***,…) được Windows cung cấp cho người lập trình, để viết các ứng dụng trên nền Windows.

Hệ điều hành Windows là một phần mềm hệ thống, nó thao tác trực tiếp với phần cứng máy tính. Các phần mềm ứng dụng khác chạy trên Windows không thao tác trực tiếp với phần cứng, mà luôn trung gian qua hệ điều hành.

Windows API là giao diện lập trình nằm ngay trên nền Windows, cung cấp các hàm thao tác trực tiếp với hệ điều hành và phần cứng máy tính. Các ứng dụng Windows sẽ thông qua Windows API để thao tác với máy tính.

[6] WinAPI, hay nhiều nơi vẫn dùng tên cũ là Win32 API



Hình 2.10. Mô hình Windows API

Windows API là cách gọi chung của ***Win16 API***, ***Win32 API*** và gần đây là ***Win64*** API. Ở một số chỗ, người ta vẫn hay gọi Win32 API để chỉ lập trình Windows API. Lý do là trước đây và thậm chí cho đến ngày nay, Windows 32 bit được sử dụng rộng rãi. Tuy nhiên, khi nói đến Windows API hay Win32 API, bạn phải hiểu nó bao gồm cả Win64 API, vì các API của Win64 hầu hết giống hệt Win32 API, chỉ có điểm khác chính ở kích thước con trỏ. Ngoài ra, các ứng dụng viết từ Win32 API vẫn chạy tốt trên các máy 64 bit.

Gần đây, Microsoft giới thiệu thêm ***WinRT*** (***Windows Runtime Library***) cho các ứng dụng metro trên Windows Store. Nhưng nó không thay thế hoàn toàn Windows API nhất là khi bạn viết các ứng dụng Desktop, hay cần thao tác trực tiếp với hệ thống.

Windows API được viết chủ yếu bằng C. Cách hiệu quả nhất để sử dụng Windows API là dùng C hoặc C++. Tuy nhiên, bạn cũng có thể sử dụng các ngôn ngữ khác như C#, .NET, VB.NET, Delphi, thậm chí cả Java.

2.3.2 Mối liên hệ Windows API và .NET Framework

Ngày nay, để viết các ứng dụng trên Windows nhanh chóng và hiệu quả, chúng ta sử dụng ***.NET Framework*** là chủ yếu. ***.NET Framework*** cũng cung cấp các thư viện để viết ứng dụng trên nền Windows, và càng ngày càng làm được nhiều việc thay thế Windows API. Nhưng nó khác Windows API ở các điểm chính:

- ***.NET Framework*** chỉ là ***wrapper*** gọi lại các hàm Windows API, tức là, nếu bạn gọi một phương thức .NET nào đó, nó sẽ gọi lại các hàm Windows API có chức năng tương ứng để thực hiện, chứ không thao tác trực tiếp đến hệ điều hành.
- Thông qua Mono, các ứng dụng viết trên .NET Framework có thể chạy được trên Linux, Mac,… còn Windows API thì không.
- Vẫn có một số thao tác cấp thấp với hệ thống không làm được với .NET Framework
- Lợi điểm của .NET Framework là nó cung cấp các API dễ dùng hơn Windows API. Nhưng nếu vậy thì chúng ta sẽ dùng đến Windows API khi nào?

Như đã nói, .NET Framework chỉ là wrapper, tức nó là trung gian giữa lập trình viên và Windows API. Khi gọi một phương thức .NET Framework, hệ thống sẽ phải gọi lại một hàm Windows API tương ứng. Do đó, hiệu suất ứng dụng sẽ giảm đi phần nào do phải gọi hàm nhiều lần. Nếu quan tâm đến hiệu suất, bạn nên gọi trực tiếp đến Windows API. Ngoài ra, nếu bạn cần viết driver, hay cần thao tác cấp thấp với hệ thống (như theo dõi gõ phím chẳng hạn),… thì gọi trực tiếp Windows API là giải pháp tốt nhất.

2.3.3 Các thành phần của Windows API

Các hàm API có thể được chia thành một số loại chính:

- ***Base Services***: thao tác với các tài nguyên trên Windows như hệ thống tập tin (file systems), các thiết bị, các tiến trình (***processes***), các luồng (***threads***), quản lý bộ nhớ,… Các hàm này nằm trong thư viện ***kernel32.dll***.
- ***Advanced Services***: loại này thường dùng để truy xuất ***Windows registry***, tắt, mở máy tính, ***Windows service***, tài khoản người dùng,… Các hàm này nằm trong thư viện ***dvapi32.dll***.

- ***Graphics Device Interface***: xuất dữ liệu đồ họa ra màn hình, máy in,… Các hàm này nằm trong thư viện ***gdi32.dll.***
- ***User Interface***: cung cấp các thành phần giao diện người dùng như form, nút, textbox, nhận tín hiệu chuột, bàn phím,… Các hàm này nằm trong thư viện ***shell32.dll, user32.dll, comctl32.dll, comdlg32.dll***:
  - ***Common Dialog Box Library***: các hộp thoại cơ bản trong Windows mà hầu như ứng dụng nào cũng sử dụng, như hộp thoại mở file, lưu file,…
  - ***Common Control Library***: nút, status bars, progress bars, toolbars, tabs,…
  - ***Windows Shell***: thu tín hiệu chuột, bàn phím,…
- ***Network Services***: các thao tác liên quan đến mạng
- ***Multimedia***: thao tác với các thiết bị đa phương tiện

2.4 Window registry

2.4.1 Registry là gì?

Registry là một cơ sở dữ liệu dùng để lưu trữ thông số kỹ thuật của Windows. Nó ghi nhận tất cả các thông tin và cài đặt cho những phần mềm bạn cài trên máy, các thiết bị phần cứng, hồ sơ người dùng, cấu hình hệ điều hành và rất rất nhiều thông tin khác.

Ví dụ, khi một phần mềm A được cài đặt, sẽ có các hướng dẫn và tệp tham chiếu đến A được thêm vào registry ở vị trí cụ thể. Nhờ đó, hệ thống/các phần mềm khác có thể tương tác với phần mềm A, tham khảo thêm thông tin như vị trí của các file, tùy chọn nào sẽ sử dụng trong A...

Có thể hiểu nôm na, registry giống như một loại DNA cho hệ điều hành Windows.

2.4.2 Nơi lưu trữ Registry

Trong Win95 & 98, Registry được ghi trong 2 file: user.dat và system.dat, Windows Me là trong file Classes.dat trong thư mục Windows. Trên Windows 7, 8, Windows 10, Registry được lưu trong thư mục "Windows\System32\Config".

Trên ổ cứng, Windows Registry không đơn giản là một file mà là một tập hợp các file riêng lẻ, gọi là hive. Mỗi hive chứa một nhánh Registry. Cụ thể:

- HKEY_LOCAL_MACHINE\ SYSTEM: \system32\config\system
- HKEY_LOCAL_MACHINE\ SAM: \system32\config\sam
- HKEY_LOCAL_MACHINE\ SECURITY: \system32\config\security
- HKEY_LOCAL_MACHINE\ SOFTWARE: \system32\config\software
- HKEY_USERS\ UserProfile: \winnt\profiles\username
- HKEY_USERS.DEFAULT: \system32\config\default

2.4.3 Cấu trúc của Registry

Registry có cấu trúc cây, giống cấu trúc thư mục. Thông thường có sáu nhánh chính. Mỗi nhánh được giao nhiệm vụ lưu giữ những thông tin riêng biệt. Trong các nhánh chính có rất nhiều nhánh con. Những nhánh con này cũng được lưu giữ những thông tin riêng biệt.

a) HKEY_CLASSES_ROOT

Lưu những thông tin dùng chung cho toàn bộ hệ thống. Là một nhánh con của HKEY_LOCAL_MACHINE\ Software. Thông tin lưu trữ ở đây đảm bảo khi bạn mở một file trong Windows Explorer thì chương trình tương ứng với file đó sẽ được mở. Bắt đầu từ Windows 2000, thông tin này được lưu trong cả HKEY_LOCAL_MACHINE và HKEY_CURRENT_USER.

- HKEY_LOCAL_MACHINE\ Software\ Classes chứa các cài đặt mặc định dùng cho tất cả user.
- HKEY_CURRENT_USER\ Software\ Classes chứa các cài đặt ghi đè lên cài đặt mặc định và chỉ áp dụng cho user chịu ảnh hưởng.

HKEY_CLASSES_ROOT cung cấp chế độ xem registry hợp nhất từ cả 2 nguồn trên. Để thay đổi cài đặt cho user bị ảnh hưởng thì các thay đổi phải được thực hiện trong HKEY_CURRENT_USER\Software\Classes thay vì HKEY_CLASSES_ROOT Tương tự, để có thể thay đổi cài đặt mặc định, các thay đổi phải được thực hiện trong HKEY_LOCAL_MACHINE\Software\Classes. Nếu bạn viết tạo key hay chỉnh sửa trong HKEY_CLASSES_ROOT, hệ thống sẽ tự động lưu thông tin key trong thư mục HKEY_LOCAL_MACHINE\Software\Classes. Còn nếu bạn tạo key trong thư mục KEY_CLASSES_ROOT và key đã tồn tại trong HKEY_CURRENT_USER\Software\ Classes, hệ thống sẽ lưu thông tin ở đó thay vì trong HKEY_LOCAL_MACHINE\ Software\ Classes.

b) HKEY_CURRENT_USER

Lưu những thông tin cho người dùng đang đăng nhập. Các thư mục, màu màn hình, cài đặt Control Panel được lưu trữ tại đây. Thông tin này được liên kết với profile của user. Nhánh này đôi khi được viết tắt là HKCU. Nó là nhánh con của HKEY_USERS

c) HKEY_LOCAL_MACHINE

Chứa thông tin cấu hình cụ thể của máy tính (cho bất kỳ user nào). Key này đội khi được viết tắt là HKLM.

d) HKEY_USERS

Lưu những thông tin của tất cả các user, mỗi user là một nhánh với tên là số ID của user đó.

e) HKEY_CURRENT_CONFIG

Lưu thông tin về phần cứng hiện tại đang dùng.

f) HKEY_DYN_DATA

Đây cũng là một phần của nhánh HKEY_LOCAL_MACHINE.

Lưu ý: Key của registry trong bản 64bit từ Windows XP trở lên sẽ chia thành key 32bit và key 64bit

2.4.4 Các kiểu dữ liệu dùng trong Registry

- REG_BINARY: Kiểu nhị phân
- REG_DWORD: Kiểu Double Word
- REG_EXPAND_SZ: Kiểu chuỗi mở rộng đặc biệt. VD: "%SystemRoot%"
- REG_MULTI_SZ: Kiểu chuỗi đặc biệt
- REG_SZ: Kiểu chuỗi chuẩn

2.4.5 Một số lưu ý

Khi sử dụng Registry Editor bạn phải tiến hành sao lưu Registry. Mọi thay đổi có thể làm máy của bạn không khởi động, treo máy,... Sao lưu bằng cách chạy Registry Editor: File - Export... và lưu vào chỗ an toàn.

Nếu chưa có kiến thức về Windows Registry xin bạn đọc một chút các dòng hướng dẫn bên dưới.

Dòng sau các từ: User Key, Sytem Key, hoặc Key cho biết đường dẫn đến nhánh cần sửa chữa hoặc tạo mới nếu nó không tồn tại.

Lưu ý: User Key: là để thay đổi đó có tác dụng với người đang Logon. System Key: là để thay đổi đó có tác dụng với tất cả người dùng.

Dòng Name là tên của khoá cần tạo, nó là nhánh con bên cửa sổ bên phải của Registry.

Dòng Type là kiểu dữ liệu của khoá mới tạo, dòng Value là giá trị của khoá.

CHƯƠNG 3. CÁC KỸ THUẬT PHÁT HIỆN MÃ ĐỘC

Các kỹ thuật phát hiện mã độc là một quá trình tìm kiếm và thẩm định xem một tệp tin, chương trình phần mềm có thể đã bị lây nhiễm mã độc hay bên trong có chứa các đoạn mã được xem là mã độc hay không. Thêm vào đó các hành vi của chúng cũng được phân tích và xem xét là nhóm các hành vi thông thường hay các hành vi thuộc về mã độc, dựa vào các kết quả đó để có thể chứng minh và phát hiện sự tồn tại của mã độc trên các hệ thống.

3.1 Các kỹ thuật phát hiện dựa trên phân tích tĩnh

Kỹ thuật phát hiện mã độc dựa trên phương pháp phân tích tĩnh có đặc điểm là phát hiện mã độc mà không cần phải chạy hay thực thi bất kỳ đoạn mã nào của nó gồm có 3 phương pháp chính là kỹ thuật dò quét, chẩn đoán dựa trên kinh nghiệm và kiểm tra tính toàn vẹn.

3.1.1 Kỹ thuật tìm các chuỗi

Một tệp mã độc cổ điển sẽ có thể được tìm thấy bằng nhiều cách khác nhau.

Cách đơn giản nhất là chuyển đổi tệp tin đó thành các mã hash như ***MD5*** hoặc ***SHA256*** và so sánh với cơ sở dữ liệu đã được tạo ra từ trước. Từ đó có thể xác định được tệp đó có phải là tệp mã độc hay không mà không cần phải thực thi hoặc phân tích mã nguồn của nó.

Tuy nhiên nếu gặp phải các biến thể của các mã độc này thì cách trên không còn tác dụng nữa. Vì chỉ cần thay đổi đôi chút về mã nguồn là có thể thay đổi được mã hash của tệp đó. Từ đó tệp có thể vượt qua được quá trình quét của cách trên.

Cách thứ hai sẽ giúp ta cải thiện hơn về khả năng quét của ứng dụng. Đó là quét dựa trên các dấu hiệu (***signatures***) của tệp tin mã độc. Các dấu hiệu này đôi khi còn được gọi là các chuỗi (***scan strings***). Để có thể phân tích trích xuất được các dấu hiệu đặc trưng của mã độc này ta cần phải phân tích mã nguồn của chúng. Từ đó có thể sưu tầm được các đặc trưng, đặc biệt là các chuỗi, các thư viện động, các khoá registry hoặc là các đường dẫn, địa chỉ website mà mã độc sử dụng. Tập hợp tất cả các yếu tố sưu tầm được ở trên ta sẽ có được dấu hiệu đặc trưng của tệp mã độc ấy.

3.1.2 Kỹ thuật Static Heuristics

Kỹ thuật này được áp dụng để nhân lên khả năng phát hiện mã độc trong các phần mềm chống virus, chẩn đoán dựa trên kinh nghiệm trong phương pháp phân tích tĩnh có thể tìm thấy các mã độc đã biết hoặc chưa biết bằng cách tìm kiếm một mẩu mã mà có những đặc điểm chung giống như là một mã độc thay vì scanning các dấu hiệu đặc biệt của mã độc. Một số phương pháp phức tạp trong các phân tích dữ liệu có thể sử dụng trí tuệ nhân tạo như là mạng neural, hệ chuyên gia, hay các kỹ thuật khai phá dữ liệu.

3.1.3 Kỹ thuật kiểm tra sự toàn vẹn (Integrity Checkers)

Kỹ thuật kiểm tra tính toàn vẹn nhằm khai thác các hành vi này để tìm ra mã độc bằng cách xem các thay đổi trái phép vào các tập tin. Một kiểm tra toàn vẹn được khởi đầu bằng việc tính và lưu dữ một checksum cho mỗi tập tin trong hệ thống được xem xét. Sau đó một checksum của tập tin cần kiểm tra sẽ được tính lại và so sánh với giá trị checksum gốc của nó. Nếu checksum khác nhau có nghĩa là đã có một sự thay đổi diễn ra

3.2 Các kỹ thuật phát hiện dựa trên phân tích động

Kỹ thuật phát hiện mã độc dựa trên phân tích động là kỹ thuật quyết định một tập tin có bị lây nhiễm hay không thông qua việc thực thi các mã chương trình và quan sát các hành vi của nó.

3.2.1 Kỹ thuật Behavior Monitors/Blockers

Một behavior blocker là một kỹ thuật chống mã độc giám sát các hành vi thực thi của một chương trình trong thời gian thực, theo dõi các hành động, các khối lệnh khả nghi của nó.

3.2.2 Kỹ thuật Emulation

Giám sát các hành vi kỹ thuật Behavior blocking cho phép mã chương trình chạy trên máy thực. Ngược lại kỹ thuật phát hiện mã độc sử dụng mô phỏng giả lập (emulation) cho phép các mã được chạy và phân tích trong một môi trường mô phỏng giả lập. Bao gồm 2 kỹ thuật chính là Dynamic heuristic và Generic decryption.

3.3 ỨNG DỤNG MÔ HÌNH HỌC MÁY VÀO PHÁT HIỆN MÃ ĐỘC

Việc phát hiện mã độc theo phương pháp phân tích tĩnh thực sự mang lại kết quả không cao, có thể dễ dàng bị vượt qua. Vì thế để có thể phát hiện được mã độc chính xác, hiệu quả hơn, đặc biệt là có thể phát hiện được các biến thể của các mã độc, hệ thống của chúng ta cần phải áp dụng các phương pháp phân tích động.

Chúng ta biết rằng giai đoạn quan trọng nhất của kỹ thuật phát hiện mã độc là xác định rõ việc biểu diễn các tập tin mã độc. Việc biến đổi một tập hợp dữ liệu đầu vào lớn thành tập các đặc trưng được gọi là trích rút đặc trưng

3.4 Một số phương pháp trích rút đặc trưng phổ biến

3.4.1 Đặc trưng byte n-gram

Đặc trưng Byte n-gram là chuỗi n byte được trích xuất từ các mã độc, chúng được sử dụng như dấu hiệu để nhận dạng phần mềm độc hại. Mặc dù kiểu đặc trưng này không cung cấp thông tin có ý nghĩa nhưng nó mang lại độ chính xác cao trong việc phát hiện những mẫu mã độc mới.

3.4.2 Đặc trưng opcode n-gram

Các nghiên cứu trước đây chỉ ra rằng trích rút đặc trưng opcode là hiệu quả hơn cho sự phân loại. Chúng biểu thị những thống kê đa dạng giữa các phần mềm độc hại và hợp pháp. Một vài opcode hiếm khi là sự dự đoán tốt cho các hành vi độc hại. Đầu tiên trong tập dữ liệu tất cả các tập tin thực thi được dịch ngược và opcode được trích rút. Một opcode là một một lệnh trong hợp ngữ, nó miêu tả thao tác được thực thi. Nó là dạng ngắn của mã hoạt động. Một lệnh gồm có một opcode và nhiều toán hạng. Một số thao tác có các toán hạng mà nhờ đó opcode có thể hoạt động, tùy thuộc vào kiến trúc CPU, thanh ghi, giá trị lưu trữ trên bộ nhớ và ngăn xếp...

3.4.3 Đặc trưng PE

Các đặc trưng này được trích rút từ một phần nào đó của tập tin EXE. Các đặc trưng PE được trích rút bằng phân tích tĩnh sử dụng thông tin cấu trúc của PE. Những đặc trưng đầy ý nghĩa này chỉ ra rằng tập tin này đã được sửa đổi hoặc bị nhiễm độc để thực hiện những hoạt động nguy hiểm.

3.4.4 Đặc trưng chuỗi

Những đặc trưng này dựa trên văn bản trơn đã được mã hóa trong tập tin thực thi như windows, getversion, getstartupinfo, getmodulefilename, messagebox, library, vv.

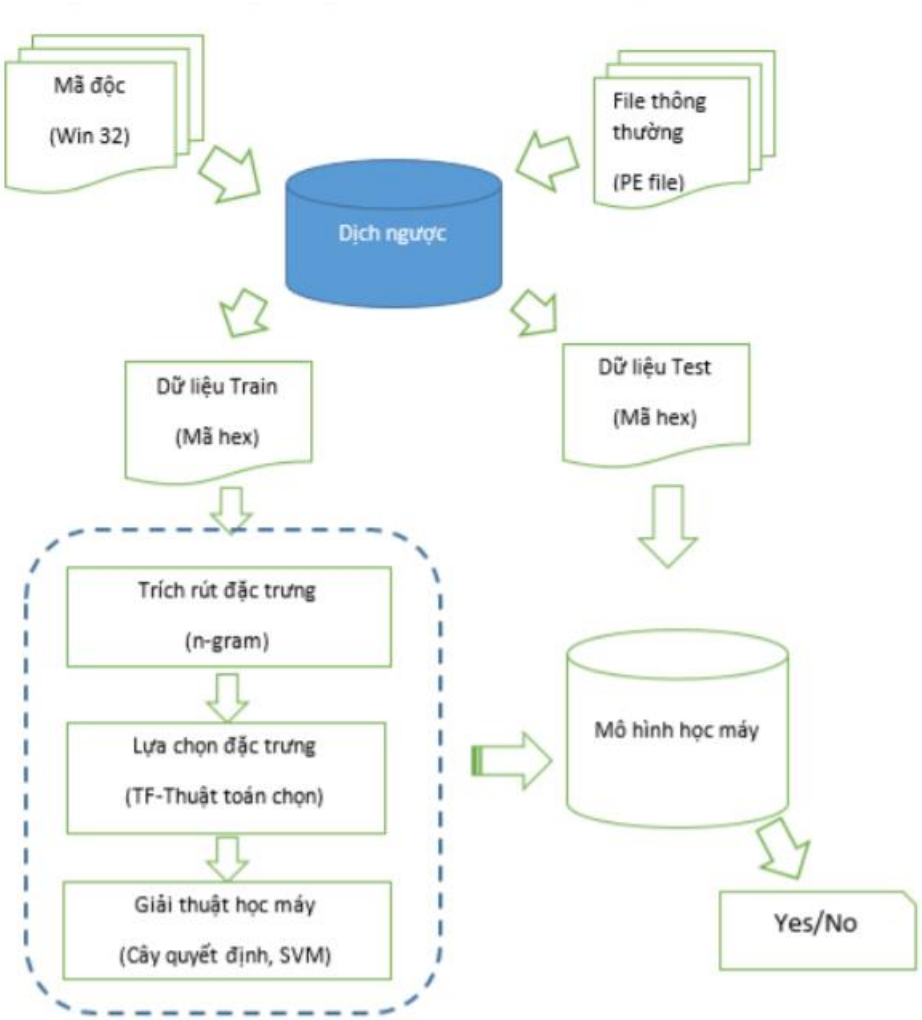
Những chuỗi này là những ký tự có thể in liên tiếp được mã hóa trong các tập tin PE cũng như phi PE.

3.4.5 Đặc trưng dựa trên chức năng

Đặc trưng này được trích rút ra theo các hành vi tại thời điểm chạy của tập tin chương trình. Các chức năng này thuộc về một tập tin thực thi và sử dụng chúng để tạo ra các thuộc tính khác nhau đại diện cho tập tin này. Các lời gọi hàm phân tích động bao gồm lời gọi hệ thống, lời gọi đến các API,… Bất kỳ phần mềm độc hại triêu gọi một vài lời gọi hệ thống mức nhân để giao tiếp với hệ điều hành, nó là một dấu hiệu của mã độc.

3.5 Tổng quan phương pháp thực hiện của hệ thống

Tổng quan về phương pháp được mô tả qua 6 bước sau:



Hình 3.1. Sơ đồ hoạt động của hệ thống

Bước 1: Thu thập dữ liệu các file mã độc và các file PE thông thường.

Bước 2: Các file này sẽ được dịch ngược về mã hex thông qua một tiện ích của chương trình.

Bước 3: Sau khi dịch ngược thì các dữ liệu huấn luyện sẽ được trích rút ra các đặc trưng là các mã hex dựa vào phương pháp n-gram. Trong khuôn khổ nghiên cứu này kích thước sử dụng là 2-gram.

Bước 4: Từ các đặc trưng là các mã n-gram byte thực hiện tính tần số 17 xuất hiện (Term Frequency) của các mã n-gram này trên mỗi tập dữ liệu. Sau đó áp dụng thuật toán được tôi đề xuất (được trình bày ở phần sau) để chọn ra được một bộ đặc trưng tốt nhất.

Bước 5: Từ bộ đặc trưng có được ta đưa chúng vào xây dựng các mô hình học máy ở đây tôi sử dụng hai giải thuật để thực nghiệm và so sánh là cây quyết định và ***svm***.

Bước 6: Sau khi xây dựng xong mô hình thì đưa các dữ liệu test vào để đánh giá kết quả.

3.6 Tiền xử lý dữ liệu

3.6.1 Sử dụng các kỹ thuật phân tích mã độc

Ở giai đoạn tiền xử lý này các file thông thường và các file mã độc là các file dạng PE sẽ được dịch ngược về các mã hex. Có rất nhiều công cụ cho phép thực hiện điều này có thể kể ra như: IDA, OLLYDBG… Tuy nhiên để đồng bộ và phát triển các hệ thống tự động sau này, chúng em đã xây dựng một tiện ích trong chương trình dựa trên cấu trúc của một file PE cho phép dịch ngược các file này về mã hex. Chương trình được viết bằng ngôn ngữ Python và sử dụng thư viện Pefile. Các file dữ liệu mẫu sau khi được dịch ngược sẽ được xử lý để lấy các mã hex quan trọng chủ yếu chúng nằm ở các phần PE header và Section nơi chứa các mã chương trình (***Executable Code Section***), các dữ liệu (Data Section), các tài nguyên (***Resources Section***), các thư viện (Import Data ,Export Data) …

Các file thông thường và mã độc sau khi được dịch ngược sẽ lấy nội dung các mã hex của từng file này và lưu lại thành các file text tương đương phục vụ cho quá trình trích chọn đặc trưng và xây dựng mô hình dự đoán.

3.6.2 Phương pháp n-gram

Trong nghiên cứu này tôi sử dụng chuỗi các byte là các đặc trưng đầu vào trong đó ở giai đoạn tiền xử lý các file dữ liệu mẫu được trích xuất dựa vào việc tính tần số xuất hiện của các n-gram byte. N-gram là một dãy các byte liên tiếp có độ dài N được mô tả như sau:

Với một dãy các mã hex sau khi dịch ngược giả sử là “AB C0 EF 12” thì dãy các n-gram byte thu được là:

1-gram	2-gram	3-gram	4-gram
AB	AB C0	AB C0 EF	AB C0 EF 12
C0	C0 EF	C0 EF 12	
EF	EF 12		
12			

Bảng 3-1 Bảng rút trích đặc trưng mẫu

Có thể nhận thấy rằng với độ dài n càng cao thì kích thước đặc trưng càng lớn. Đối với mã hex có 16 giá trị khác nhau như vậy không gian đặc trưng của 1-gram sẽ là 162=256 với 2-gram là 164=65536. Trong nội dung luận văn này tôi chủ yếu tập trung vào phương pháp chọn đặc trưng vì vậy các kết quả được thử nghiệm trên dãy 2-gram. Như vậy ở giai đoạn tiền xử lý các file dữ liệu mẫu được dịch ngược sang các mã hex và sau đó tiếp tục được trích rút ra các đặc trưng dựa vào phương pháp n-gram.

3.6.3 Tính tần số xuất hiện (Term Frequency)

Như chúng ta đã phân tích ở các phần trên, dữ liệu mẫu sau khi thực hiện tiền xử lý sẽ được lưu dưới dạng các file text và được trích rút các dãy n-gram, một trong những phương pháp được biết đến trong các bài toán khai phá và phân lớp dữ liệu text đó là phương pháp tính tần số xuất hiện (TF). Dựa vào phương pháp như vậy ở bước này chúng tôi thực hiện tính toán tần số xuất hiện của mỗi n-gram byte khác nhau trên từng file dữ liệu mẫu. Các kết quả này được lưu như mỗi véc tơ đặc trưng và trước khi được đưa vào mô hình học sẽ được xử lý để trích ra một bộ đặc trưng tốt nhất.

Công thức để tính các giá trị TF được cho như sau:

$$tf(t,d) = \frac{f(t,d)}{max\{f(w,d): w \in d}$$

Tần số xuất hiện của một mã n-gram byte trong một tập mẫu (file dữ liệu mẫu đã được đưa về dạng text) được tính bằng thương của số lần xuất hiện n-gram byte đó trong tập mẫu và số lần xuất hiện nhiều nhất của một n-gram byte bất kỳ trong tập mẫu đó (giá trị sẽ thuộc khoảng (0, 1))

- f(t,d) - số lần xuất hiện của một n-gram byte t trong tập mẫu d.
- max{f(w,d):w∈d} - số lần xuất hiện nhiều nhất của một n-gram byte bất kỳ trong tập mẫu.

3.6.4 Xây dựng mô hình dự đoán dựa trên các thuật toán phân lớp

Sau khi chọn được tập đặc trưng dựa vào phương pháp đã trình bày phía trên, bước tiếp theo ở giai đoạn này là tiến hành đưa các giá trị tần số xuất hiện của các đặc trưng n-gram byte đã được chọn trên các tập mẫu ban đầu vào các thuật toán trong học máy để xây dựng mô hình dự đoán., các dữ liệu mẫu được gán nhãn thành 2 lớp mã độc và file bình thường. Giai đoạn này sử dụng các véc tơ đặc trưng như là dữ liệu huấn luyện đầu vào cho các thuật toán phân lớp dữ liệu có thể áp dụng như: k láng giềng (KNN), Naive Bayes (NB), cây quyết định (DT), Rừng ngẫu nhiên (RF), máy véc tơ hỗ trợ (SVM)… Trong phương pháp học máy giám sát khó có thể nói là thuật toán phân lớp nào tốt nhất vì mỗi thuật toán có những ưu, nhược điểm riêng phù hợp cho mỗi loại dữ liệu khác nhau. Riêng đối với các bài toán phân lớp dữ liệu văn bản thì cây quyết định và SVM là 2 thuật toán được cho là có độ chính xác cao và được sử dụng khá rộng rãi hiện nay. Trong đó thuật toán máy véc tơ hỗ trợ (SVM) được đánh giá là rất phù hợp và hiệu quả trong các bài toán phân lớp dữ liệu văn bản, trong khi đó cây quyết định (DT) là thuật toán dễ hiểu, dựa vào việc sinh luật khá gần gũi tư duy của con người .Trong nghiên cứu này tôi lựa chọn 2 thuật toán tiêu biểu là cây quyết định và máy véc tơ hỗ trợ để xây dựng mô hình đự đoán phát hiện mã độc đồng thời các kết quả của 2 phương pháp sẽ được so sánh và đánh giá thông qua quá trình thực nghiệm.

CHƯƠNG 4. XÂY DỰNG ỨNG DỤNG PHÁT HIỆN MÃ ĐỘC

4.1 Công nghệ sử dụng

4.1.1 Python

Python là một ngôn ngữ lập trình bậc cao cho các mục đích lập trình đa năng, do Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Python được thiết kế với ưu điểm mạnh là dễ đọc, dễ học và dễ nhớ. Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu. Vào tháng 7 năm 2018, Van Rossum đã từ chức Leader trong cộng đồng ngôn ngữ Python sau 30 năm lãnh đạo.

Python hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động; do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk, và Tcl. Python được phát triển trong một dự án mã mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.

Ban đầu, Python được phát triển để chạy trên nền Unix. Nhưng rồi theo thời gian, Python dần mở rộng sang mọi hệ điều hành từ MS-DOS đến Mac OS, OS/2, Windows, Linux và các hệ điều hành khác thuộc họ Unix. Mặc dù sự phát triển của Python có sự đóng góp của rất nhiều cá nhân, nhưng Guido van Rossum hiện nay vẫn là tác giả chủ yếu của Python. Ông giữ vai trò chủ chốt trong việc quyết định hướng phát triển của Python.

4.1.2 PyQt5

a) Giới thiệu chung

Qt là một Application framework đa nền tảng viết trên ngôn ngữ C++ , được dùng để phát triển các ứng dụng trên desktop, hệ thống nhúng và mobile. Hỗ trợ cho các platform bao gồm: Linux, OS X, Windows, VxWorks, QNX, Android, iOS, BlackBerry, Sailfish OS và một số platform khác. PyQt là Python interface của Qt, kết hợp của ngôn ngữ lập trình Python và thư viện Qt, là một thư viện bao gồm các thành phần giao diện điều khiển (widgets , graphical control elements).

PyQt API bao gồm các module bao gồm số lượng lớn với các classes và functions hỗ trợ cho việc thiết kế ra các giao diện giao tiếp với người dùng của các phần mềm chức năng. Hỗ trợ với Python 2.x và 3.x.

PyQt được phát triển bởi Riverbank Computing Limited.

Các class của PyQt5 được chia thành các module, bao gồm:

+ ***QtCore*** : là module bao gồm phần lõi không thuộc chức năng GUI, ví dụ dùng để làm việc với thời gian, file và thư mục, các loại dữ liệu, streams, URLs, mime type, threads hoặc processes.
+ ***QtGui*** : bao gồm các class dùng cho việc lập trình giao diện (windowing system integration), event handling, 2D graphics, basic imaging, fonts và text.
+ ***QtWidgets*** : bao gồm các class cho widget
+ ***QtMultimedia*** : thư viện cho việc sử dụng âm thanh, hình ảnh, camera,…
+ ***QtBluetooth*** : bao gồm các class giúp tìm kiếm và kết nối với các thiết bị có giao tiếp với phần mềm.
+ ***QtNetwork*** : bao gồm các class dùng cho việc lập trình mạng, hỗ trợ lập trình TCP/IP và UDP client , server hỗ trợ việc lập trình mạng.
+ ***QtPositioning*** : bao gồm các class giúp việc hỗ trợ xác định vị.
+ ***Enginio*** : module giúp các client truy cập các Cloud Services của Qt.
+ ***QtWebSockets*** : cung cấp các công cụ cho WebSocket protocol.
+ ***QtWebKit*** : cung cấp các class dùng cho làm việc với các trình duyệt Web, dựa trên thư viện WebKit2.
+ ***QtWebKitWidgets*** : các widget cho WebKit.
+ ***QtXml*** : các class dùng cho làm việc với XML file.
+ ***QtSvg*** : dùng cho hiển thị các thành phần của SVG file.
+ ***QtSql*** : cung cấp các class dùng cho việc làm việc với dữ liệu.
+ ***QtTest*** : cung cấp các công cụ cho phép test các đơn vị của ứng dụng với PyQt5.

b) Một số công cụ hỗ trợ thiết kế với PyQt5

Qt Designer : Qt sử dụng IDE tên ***Qt Creator*** với một tool thiết kế giao diện người dùng Qt Designer. Qt Designer có thể làm việc một mình độc lập với Qt Creator .

Qt Designer sử dụng XML .ui file để lưu thiết kế và không sinh thêm bất kỳ mã nguồn nào của nó. User Interface Compiler (UIC) đọc định dạng file XML (***.ui***) và xuất ra header file mã nguồn C++ tương ứng. Qt có một class ***QUiLoader*** cho phép một ứng dụng tải một file .ui và tạo một giao diện động tương ứng.

UIC Python module: PyQt5 không chứa class ***QUiLoader*** nhưng thay vào đó là module Python UIC. Cũng giống như ***QUiLoader*** , module này cũng tải định dạng file ***.ui*** và tạo giao diện động tương ứng. Giống như UIC (User Interface Compiler) module python này cũng sinh ra mã nguồn python tạo nên giao diện tương ứng.

4.1.3 JSON

JSON (***JavaScript Object Noattion***) là 1 định dạng hoán vị dữ liệu nhanh. Chúng dễ dàng cho chúng ta đọc và viết. Dễ dàng cho thiết bị phân tích và phát sinh. Chúng là cơ sở dựa trên tập hợp của ngôn ngữ lập trình JavaScript, tiêu chuẩn ***ECMA-262*** phiên bản 3 - tháng 12 năm 1999. JSON là 1 định dạng kiểu text mà hoàn toàn độc lập với các ngôn ngữ hoàn chỉnh, thuộc họ hàng với các ngôn ngữ họ hàng C, gồm có C, C++, C#, Java, JavaScript, Perl, Python, và nhiều ngôn ngữ khác. Những đặc tính đó đã tạo nên JSON – một ngôn ngữ hoán vị dữ liệu lý tưởng.

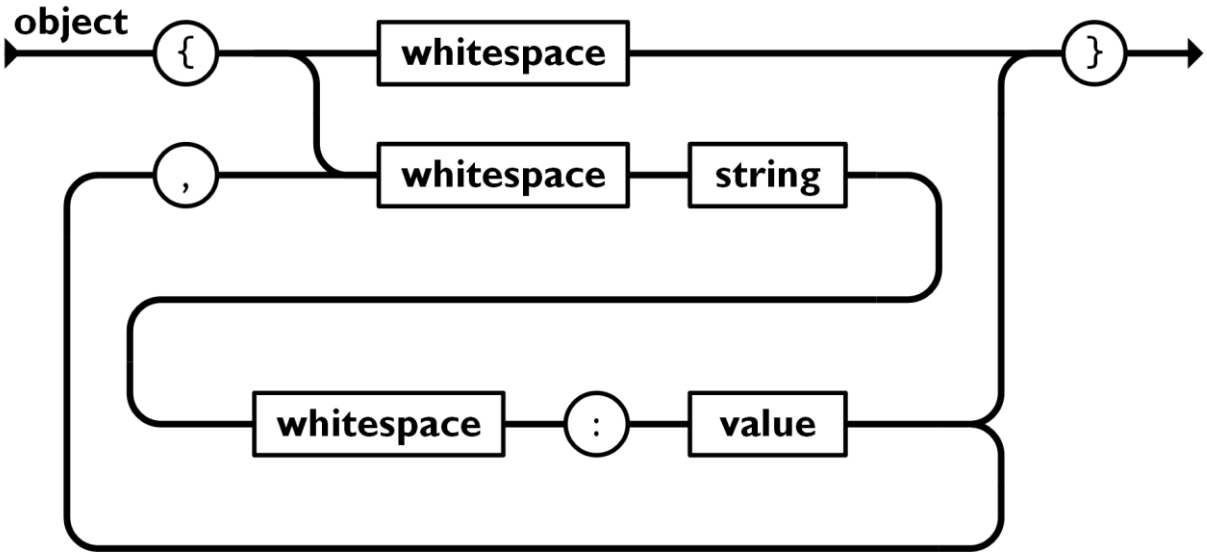
JSON được xây dựng trên 2 cấu trúc:

+ Là tập hợp của các cặp tên và giá trị name-value. Trong những ngôn ngữ khác nhau, đây được nhận thấy như là 1 đối tượng (***object***), sự ghi (***record***), cấu trúc (***struct***), từ điển (***dictionary***), bảng băm (***hash table***), danh sách khoá (***keyed list***), hay mảng liên hợp.
+ Là 1 tập hợp các giá trị đã được sắp xếp.

Đây là một cấu trúc dữ liệu phổ dụng. Hầu như tất cả các ngôn ngữ lập trình hiện đại đều hỗ trợ chúng trong 1 hình thức nào đó. Chúng tạo nên ý nghĩa của 1 định dạng hoán vị dữ liệu với các ngôn ngữ lập trình cũng đã được cơ sở hoá trên cấu trúc này.

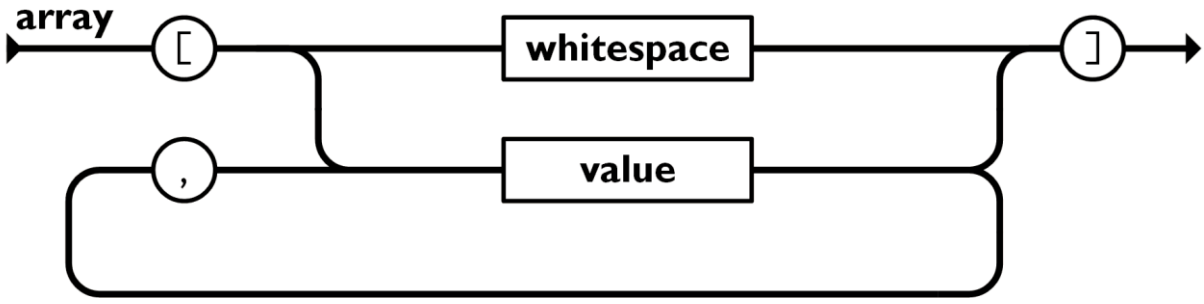
Trong JSON, chúng có những thứ trên các định dạng:

+ 1 đối tượng là 1 hổn độn của các cặp tên và giá trị. 1 đối tượng bắt đầu bởi dấu ngoặc đơn trái "{" và kết thúc với dấu ngoặc đơn phải "}". Từng tên được theo sao bởi dấu 2 chấm ":" và các cặp tên hoặc giá trị được tách ra bởi dấu phẩy ,.



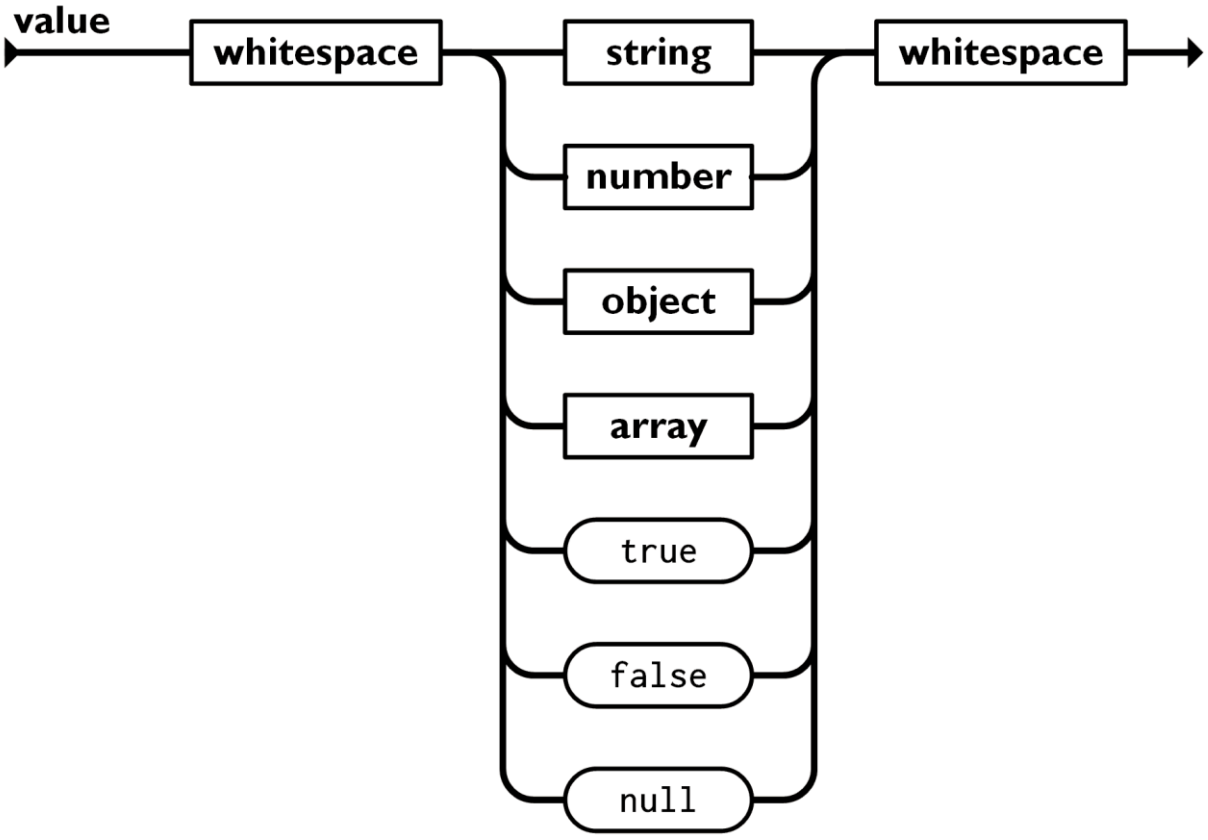
Hình 4.1. Định dạng JSON kiểu Object

+ 1 mảng là 1 tập hợp các giá trị đã được sắp xếp. 1 mảng bắt đầu bởi dấu mở ngoặc vuông [và kết thúc với dấu ngoặc vuông phải]. Các giá trị được cách nhau bởi dấu phẩy “,”.



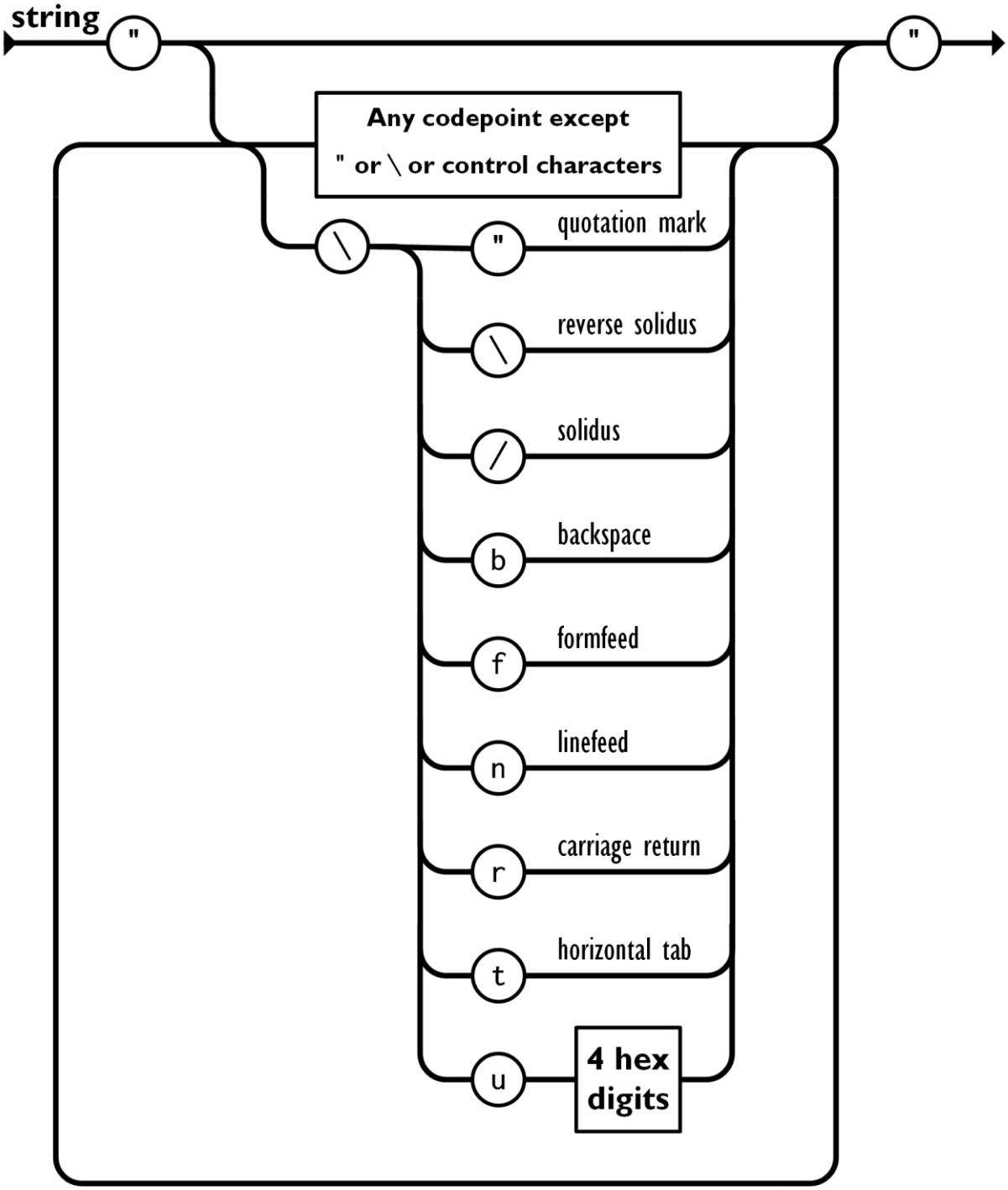
Hình 4.2. Định dạng JSON kiểu mảng

+ 1 giá trị có thể là 1 chuỗi string trong những trích dẫn kép hay là 1 số, hay “***true***” hay “***false***” hay “***null***”, hay là 1 đối tượng hay là 1 mảng. Những cấu trúc này có thể đã được lồng vào nhau.



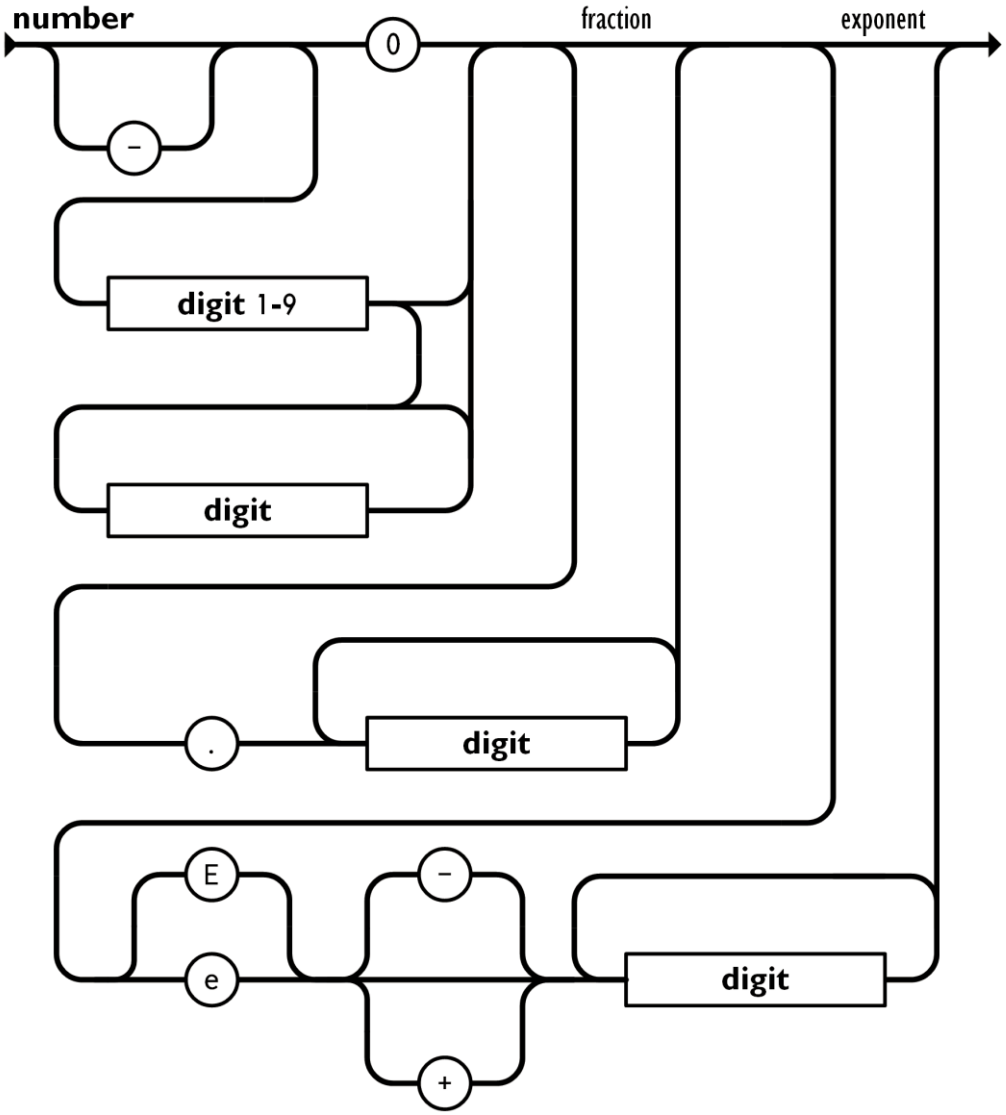
Hình 4.3. Định dạng JSON chứa một giá trị

+ 1 chuỗi string là 1 tập hợp của zero hay ngay cả mẫu tự Unicode, được bao bọc trong các dấu trích dẫn kép ("), dùng để thoát ra dấu chéo ngược. 1 ký tự đã được hiển thị như là 1 chuỗi ký tự đơn độc. 1 chuỗi string rất giống như là chuỗi string C hay là Java.



Hình 4.4. Định dạng JSON chứa một chuỗi String

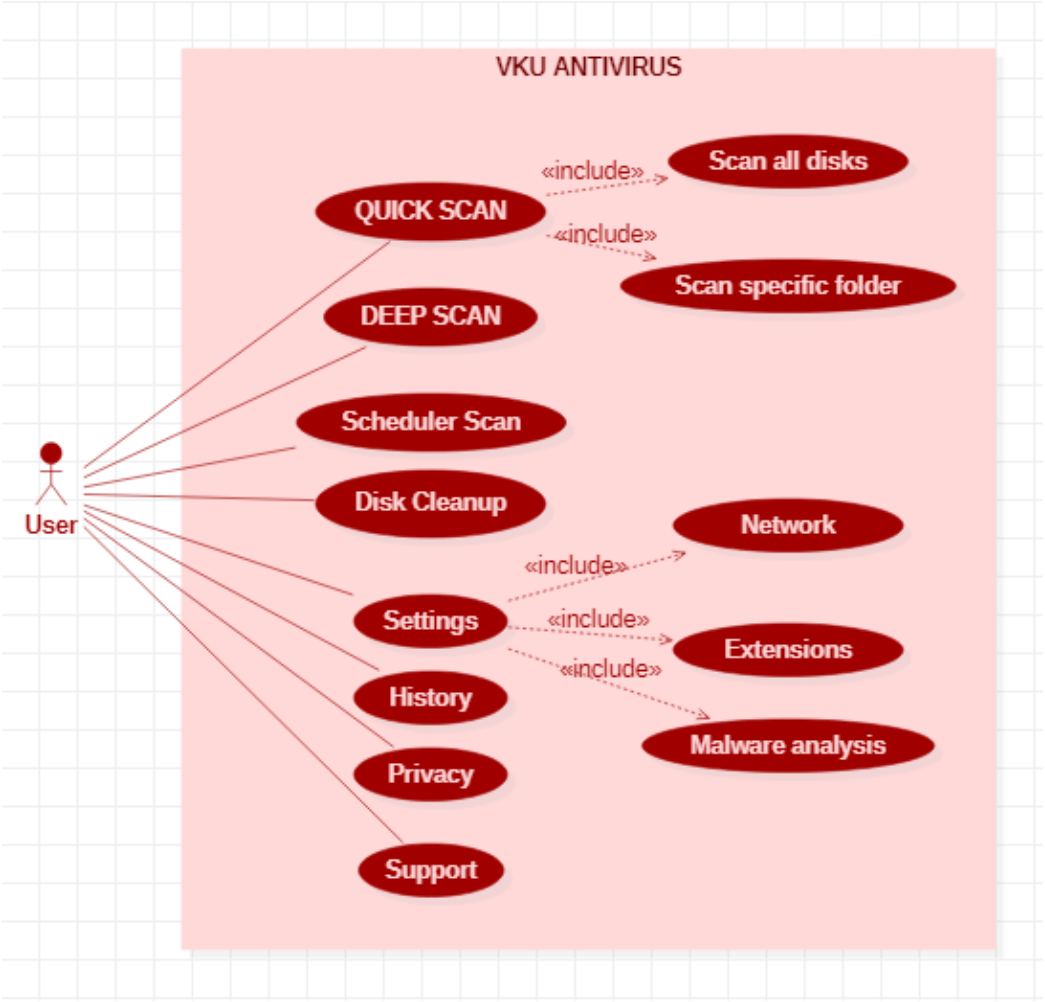
+ 1 số rất giống 1 số C và Java, trừ định dạng bát phân và hex là không thể dùng.



Hình 4.5. Định dạng JSON kiểu số

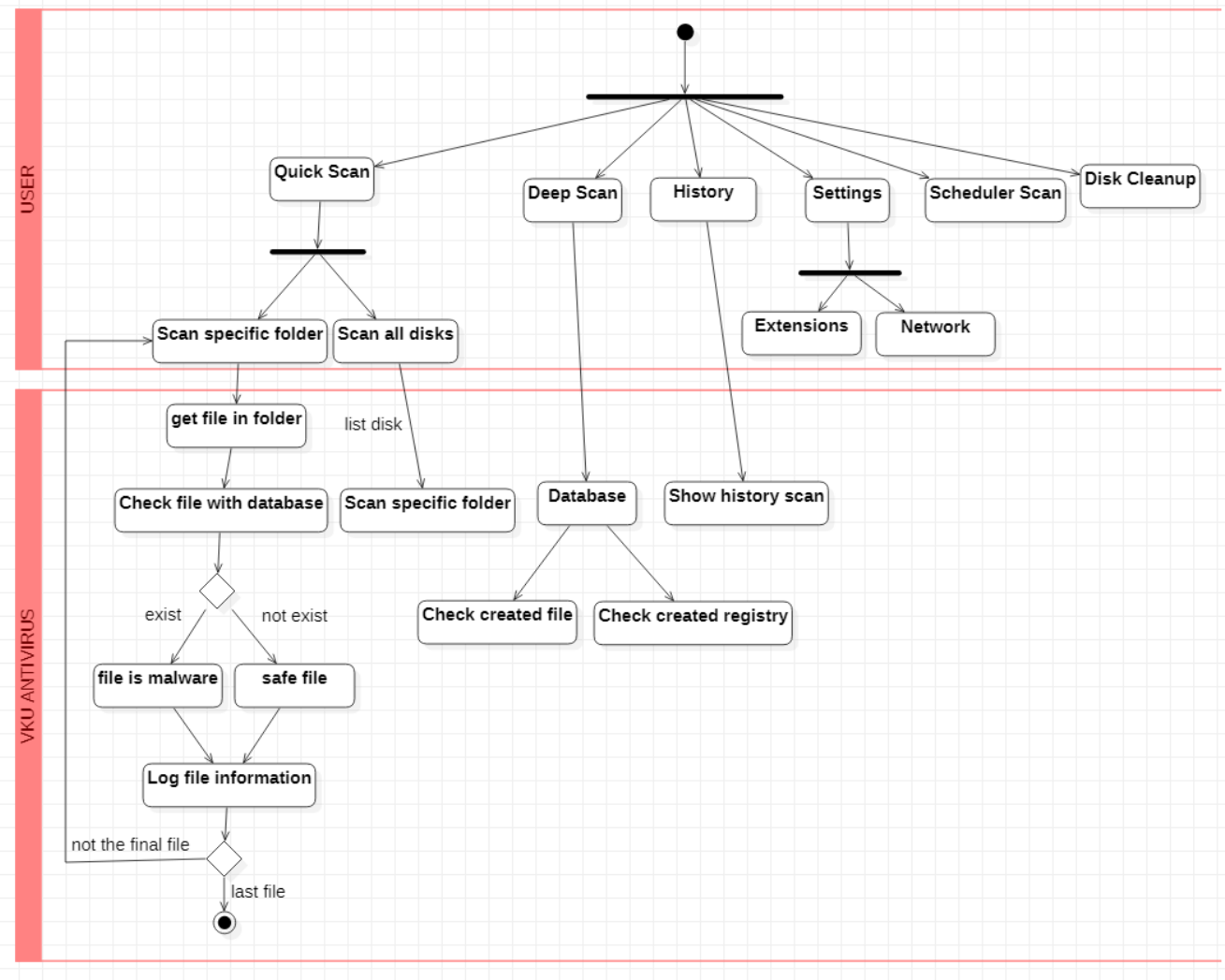
4.2 Phân tích và thiết kế hệ thống

4.2.1 Biểu đồ use case



Hình 4.6. Biểu đồ use case của hệ thống

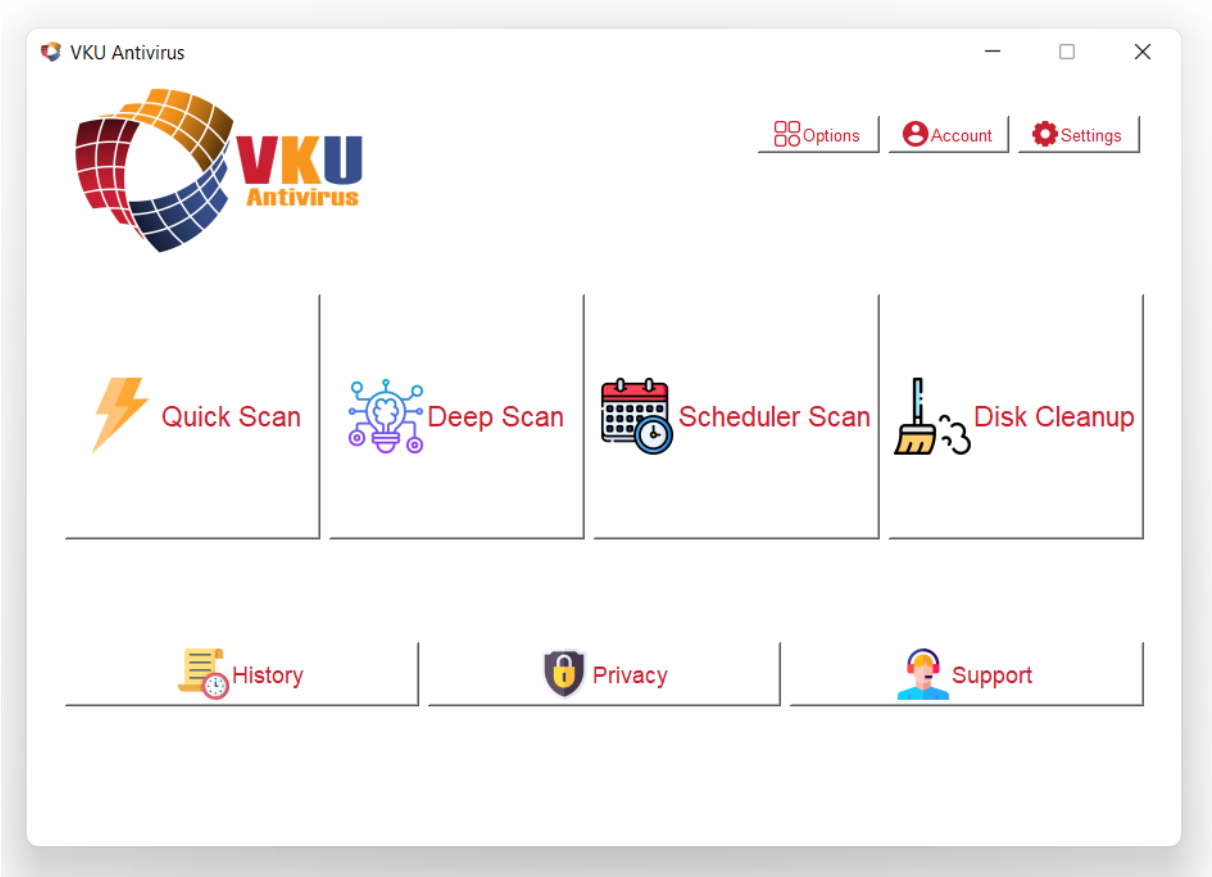
4.2.2 Biểu đồ hoạt động



Hình 4.7. Biểu đồ hoạt động của hệ thống

4.3 Chức năng ứng dụng

4.3.1 Màn hình chính



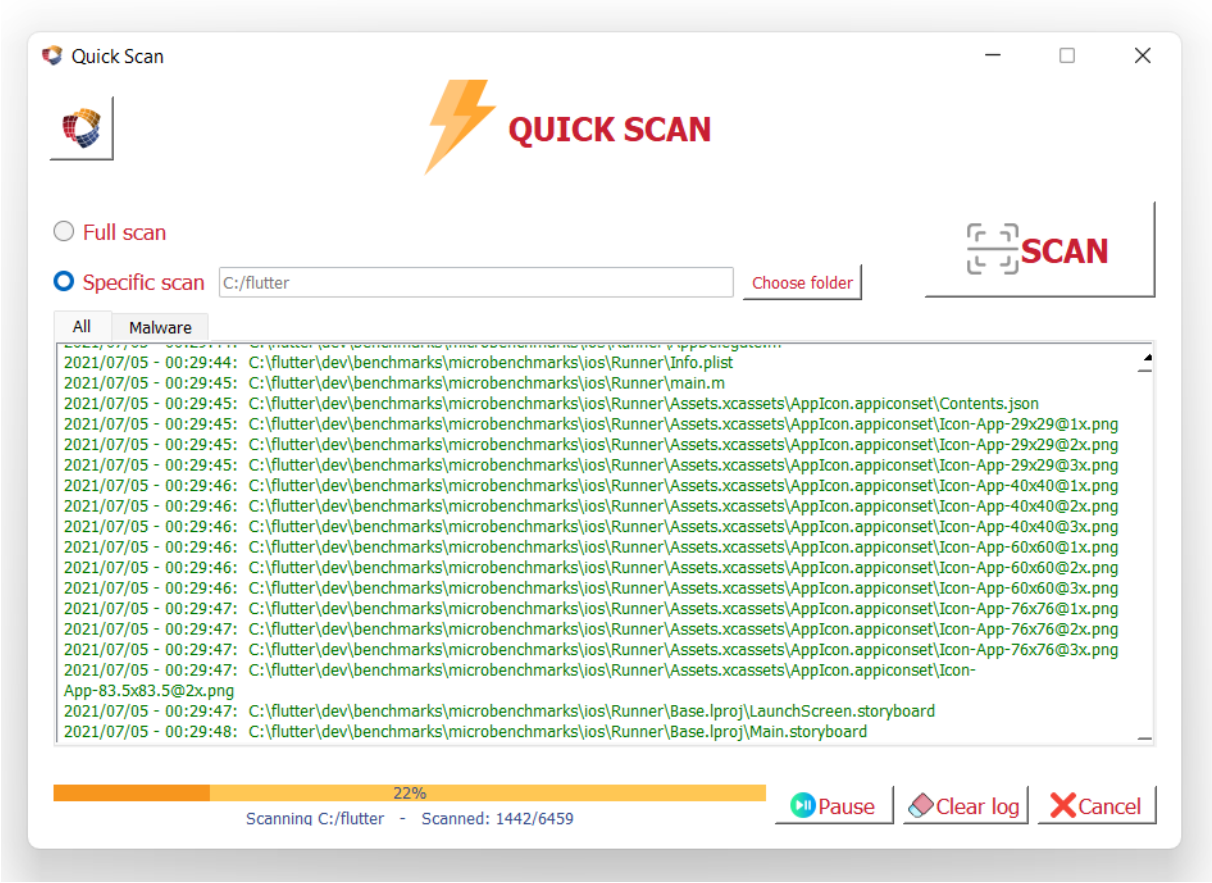
Hình 4.8. Giao diện màn hình chính

a) Mô tả chức năng

Màn hình chính hiển thị tên ứng dụng và các nút dẫn đến các chức năng cụ thể của ứng dụng.

Bao gồm: Quick scan. Deep scan, Scheduler scan, Disk cleanup và các chức năng hỗ trợ người dùng như History, Privacy và Support.

4.3.2 Quick scan



Hình 4.9. Quick scan

a) Mô tả chức năng

Ở màn hình quick scan, người dùng có thể chọn trong hai chế độ quét.

- ***Full scan***: cho phép ứng dụng quét toàn bộ các ổ đĩa của thiết bị.
- ***Specific scan***: cho phép ứng dụng quét một thư mục cụ thể nào đó.

Trong quá trình quét, ứng dụng sẽ hiển thị các tệp đã quét, nếu tệp tin an toàn tệp sẽ được đánh dấu an toàn bằng màu xanh lục, nếu tệp tin không an toàn thì sẽ được đánh dấu là mã độc bằng chữ màu đỏ và được hiển thị thêm bên tab riêng (***tab Malware***). Thanh ***progress bar*** hiển thị tỉ số tệp đã quét được so với tổng số lượng file trong thư mục. Người dùng có thể tạm dừng việc quét bằng nút ***Pause***, xoá các dòng nhật kí tệp đã quét trên màn hình bằng nút ***Resume*** và có thể huỷ quá trình quét bằng nút ***Cancel***.

b) *Đặc tả*

User case	Quick scan
Actor	Người sử dụng
Mô tả	Người sử dụng muốn quét nhanh thiết bị hoặc một thư mục cụ thể nào đó
Mục đích	Tìm các tệp mã độc trong thiết bị
Yêu cầu đặc biệt	Không có
Điều kiện tiên quyết	Người dùng chọn thư mục cần quét
Điều kiện phát sinh	Không có
Điều kiện hậu quyết	Không có
Dòng sự kiện chính	Tìm các tệp là mã độc trong thư mục
Dòng sự kiện phụ Rẽ nhánh	Home: Trờ về màn hình chính Pause: Tạm dừng phân tích Resume: Tiếp tục phân tích Cancel: Huỷ phân tích Clear log: làm sạch nhật kí
Dòng sự kiện ngoại lệ	Không có.
Khác	Không có.

c) *Nguyên lý hoạt động*

Ở màn hình "Quick scan" có 2 lựa chọn: "Full scan" và "Specific scan".

Specific scan

- Người dùng chọn một thư mục để quét. Sau khi người dùng click vào nút "***Scan***" thì chương trình sẽ tạo ra 2 luồng hoạt động. Luồng tính toán sẽ tính toán tổng số lượng tệp trong thư mục đã chọn. Luồng phân tích sẽ chạy thuật toán quét mã độc.
- Nếu luồng tính toán chưa hoàn thành thì thanh "***progress bar***" sẽ chạy dưới dạng không dữ liệu. Sau khi có kết quả trả về từ luồng tính toán, thanh "***progress bar***" sẽ hiển thị thông tin thư mục quét và tỉ số giữa tệp đã quét trên tổng số tệp.
- Ở luồng phân tích, sẽ chạy tuần tự tất cả các tệp tin trong thư mục đã chọn để tiền hành thuật toán quét mã độc. Với mỗi tệp tin, chương trình sẽ tiến hành "hashing" - chuyển đổi tệp thành mã dạng MD5 và so sánh với cơ sở dữ liệu của chương trình. Nếu không trùng với bất kì thông tin mã độc nào của chương trình thì tệp tin đó sẽ tiếp tục được chuyển đổi thành mã SHA256 để so sánh với dữ liệu. Nếu tiếp tục không trùng khớp với bất kì thông tin nào ở cơ sở dữ liệu, chương trình sẽ tiến hành phân tích mã nguồn của tệp tin để trích xuất các đặc trưng của tệp như các thư viện động mà tệp dùng để thực thi, các chuỗi

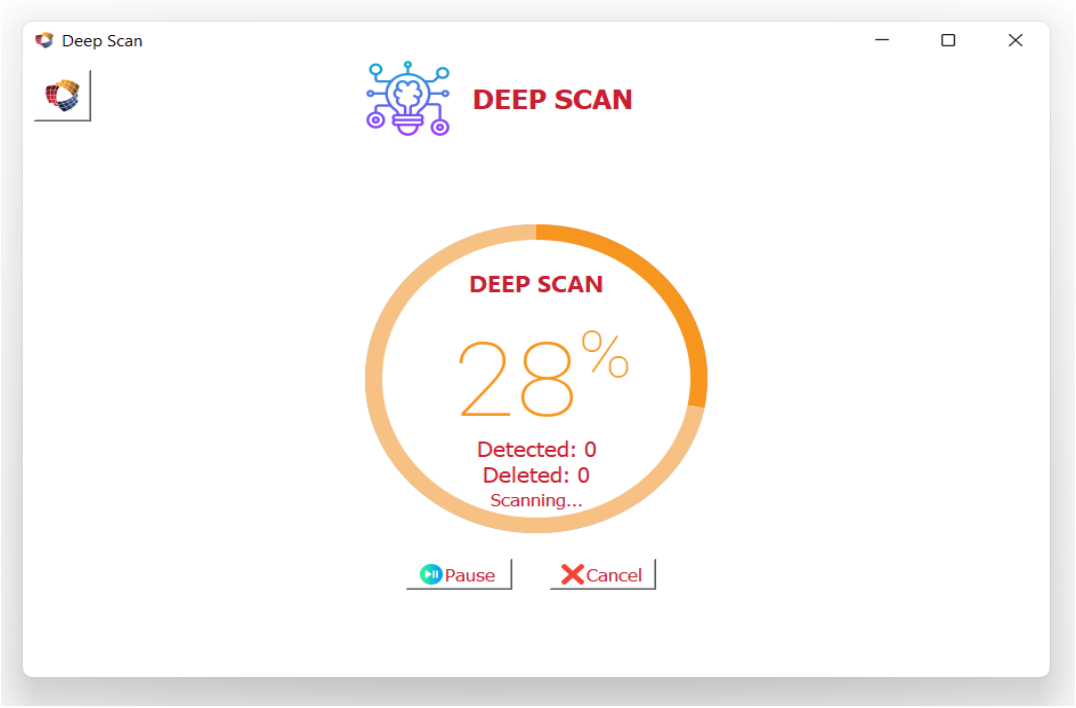
kí tự, các địa chỉ website có trong tệp,… Từ đó so sánh với cơ sở dữ liệu. Sau tất cả quá trình trên nếu vẫn không trùng khớp với dữ liệu ở cơ sở dữ liệu thì tệp đó được đánh nhãn là tệp an toàn. Nếu một trong các lần so sánh trên có kết quả trùng khớp với thông tin mã độc của chương trình thì tệp đó sẽ được gán nhãn là mã độc và được hệ thống tiến hành cô lập tệp tin đó. Sau đó sẽ được lưu vào lịch sử.

- Tab "***All***" sẽ hiển thị lần lượt tất cả các tệp tin mà chương trình đã quét. Nếu là tệp tin an toàn thì sẽ được in với chữ màu xanh. Nếu tệp tin có nhãn là mã độc thì sẽ được in với chữ màu đỏ.
- Tab "***Malware***" sẽ hiển thị tất cả các tệp tin nhãn là mã độc mà hệ thống đã tìm thấy.
- Trong quá trình phân tích thì người dùng có thể tạm dừng bằng nút "***Pause***" và tiếp tục quá trình phân tích trở lại bằng nút "***Resume***". Người dùng có thể xoá tất cả các dòng nhật kí ở tab "***All***" bằng nút "***Clear Log***". Ngoài ra để huỷ quá trình phân tích bằng nút "Cancel".
- Người dùng quay về màn hình chính bằng nút "***Home***".

Full scan

- Sau khi người dùng click vào nút "***Scan***", hệ thống sẽ liệt kê ra tất các ổ đĩa trên thiết bị. Sau đó lần lượt tiến hành phân tích các tệp trên từng ổ đĩa. Quá trình phân tích sẽ tương tự với chức năng "specific scan".

4.3.3 Deep scan



Hình 4.10. Deep scan

a) Mô tả chức năng

Ở màn hình Deep scan, ứng dụng sẽ dựa vào dữ liệu có sẵn để tìm xem có sự tồn tại của các tệp mã độc trên thiết bị và cả các registry mà mã độc đã tạo trong hệ thống.

Ứng dụng sẽ chủ động xoá tệp và các registry đã phát hiện được và lưu thông tin vào dữ liệu lịch sử.

Người dùng có thể tạm dừng hoặc huỷ quá trình quét của ứng dụng.

b) *Đặc tả*

User case	Deep scan
Actor	Người sử dụng
Mô tả	Người sử dụng muốn phân tích xem thiết bị đã thực thi mã độc nào chưa
Mục đích	Tìm các tệp là tệp và registry do mã độc thực thi sinh ra.
Yêu cầu đặc biệt	Không có
Điều kiện tiên quyết	Không có
Điều kiện phát sinh	Không có
Điều kiện hậu quyết	Không có
Dòng sự kiện chính	Tìm các tệp là tệp và registry do mã độc thực thi sinh ra.
Dòng sự kiện phụ Rẽ nhánh	Home: Trờ về màn hình chính Pause: Tạm dừng phân tích Resume: Tiếp tục phân tích Cancel: Huỷ phân tích
Dòng sự kiện ngoại lệ	Không có.
Khác	Không có.

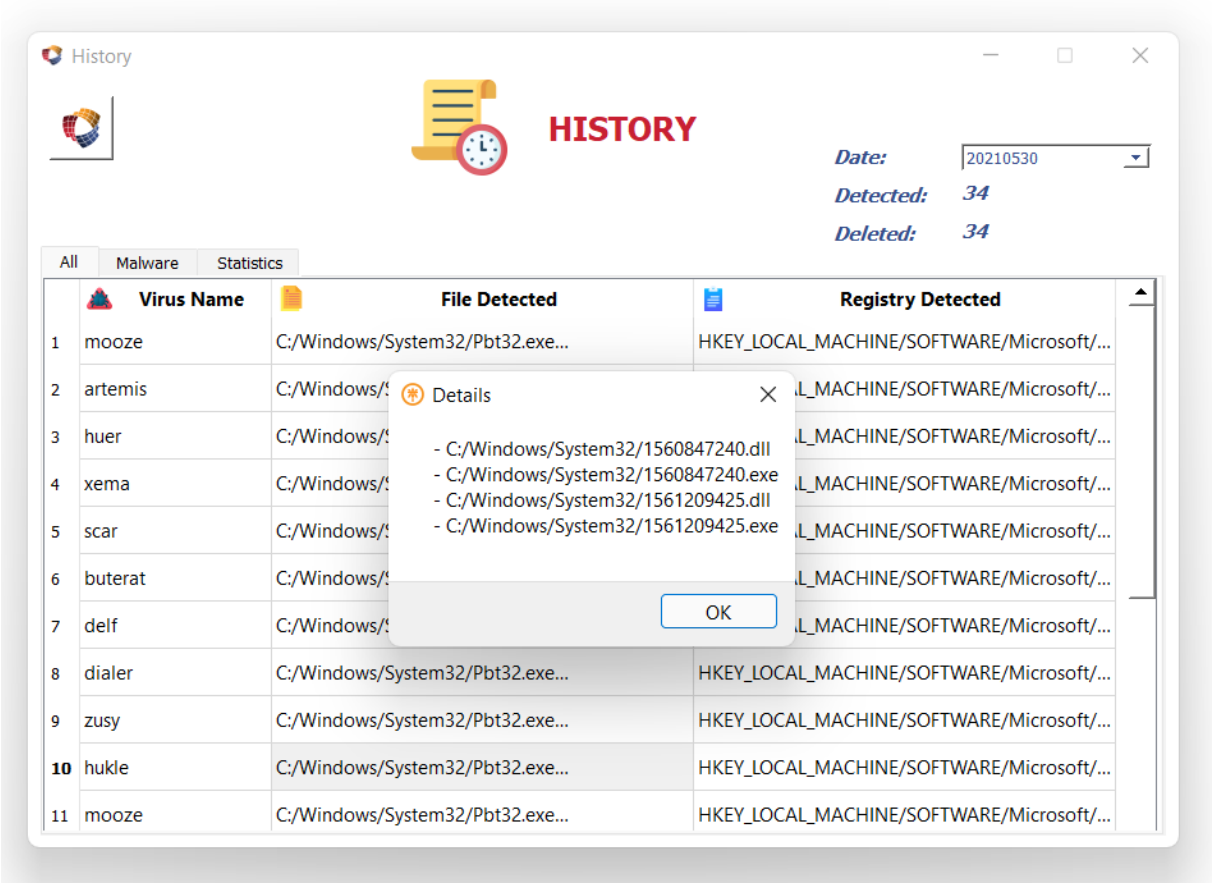
Bảng 4-1 Đặc tả chức năng quét sâu của ứng dụng

c) *Nguyên lý hoạt động*

Khi người dùng chọn chức năng Deep scan thì chương trình sẽ chạy thuật toán tìm các tệp tin và các ***window registry*** mà mã độc đã tạo ra trong quá trình chúng thực thi.

Cơ sở dữ liệu của hệ thống sẽ chứa thông tin tệp tin và registry sẽ được tạo bởi mã độc khi được thực thi. Dựa vào đó, chương trình phân tích tìm xem có sự tồn tại tệp tin và các registry ấy trong hệ thống hay không. Nếu có, hệ thống sẽ tiến hành xoá tệp tin vả registry đó. Sau đó lưu thông tin tệp tin và các ***registry*** đó vào ***history***.

4.3.4 History



Hình 4.11. Lịch sử quét

a) Mô tả chức năng

Ở màn hình History, ứng dụng sẽ hiển thị thông tin tất cả các tệp và các registry đã quét được trong lần quét gần nhất.

Người dùng có thể chọn những ngày trước đó để xem.

b) *Đặc tả*

User case	History
Actor	Người sử dụng
Mô tả	Người sử dụng muốn xem thông tin lịch sử mã độc đã phân tích được.
Mục đích	Người sử dụng muốn xem thông tin lịch sử mã độc đã phân tích được.
Yêu cầu đặc biệt	Không có
Điều kiện tiên quyết	Không có
Điều kiện phát sinh	Không có
Điều kiện hậu quyết	Không có
Dòng sự kiện chính	Người sử dụng muốn xem thông tin lịch sử mã độc đã phân tích được.
Dòng sự kiện phụ Rẽ nhánh	Home: Trờ về màn hình chính Chọn ngày để xem Xem chi tiết các tệp và registry của mã độc
Dòng sự kiện ngoại lệ	Không có.
Khác	Không có.

Bảng 4-2 Đặc tả chức năng xem lịch sử đã quét của ứng dụng

c) *Nguyên lý hoạt động*

Hiển thị thông tin các tệp tin mã độc đã tìm thấy theo ngày.

4.4 Đánh giá ứng dụng

4.4.1 Khả năng phát hiện mã độc


```
{
  "name": "Trojan.Win32.Wootbot.m!c",
  "type": "Win32 EXE",
  "size": "262002",
  "md5": "86b6c59aa48a69e16d3313d982791398",
  "sha256": "570eab9ce89db496a22196746cdcc68d5924abed65655bd4042b14306eadd98f",
  "file_created": [
    "C:\\WINDOWS\\system32\\msvc32.exe"
  ],
  "hkey_created": [
    "\\REGISTRY\\MACHINE\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\\MySLScan",
    "\\REGISTRY\\MACHINE\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\RunOnce\\MySLScan",
    "\\REGISTRY\\USER\\S-1-5-21-1482476501-1645522239-1417001333-500\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\MySLScan",
    "\\REGISTRY\\USER\\S-1-5-21-1482476501-1645522239-1417001333-500\\Software\\Microsoft\\Windows\\CurrentVersion\\RunOnce\\MySLScan",
    "HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ZoneMap\\UNCAsIntranet",
    "\\REGISTRY\\MACHINE\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\RunServices\\MySLScan",
    "\\REGISTRY\\USER\\.DEFAULT\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\MySLScan",
    "\\REGISTRY\\USER\\.DEFAULT\\Software\\Microsoft\\Windows\\CurrentVersion\\RunOnce\\MySLScan"
  ],
  "signature": [
    "1337.m0rtus.info",
    "ADVAPI32.dll ",
    "GDI32.dll ",
    "MSVCRT.dll ",
    "NTDLL.DLL ",
    "KERNEL32.dll ",
    "USER32.dll ",
    "comctl32.dll ",
    "imm32.dll ",
    "kernel32.dll ",
    "WS2_32.dll ",
    "PSAPI.DLL ",
    "DnsMulticastQueryTimeouts ",
    "KERNEL32.DLL ",
    "advapi32.dll ",
    "rpcrt4.dll ",
    "WS2HELP.dll ",
    "hnetcfg.dll ",
    "C:\\WINDOWS\\System32\\wshtcpip.dll"
  ]
},
```


Hình 4.12 Thông tin mẫu của một mã độc

Với các thông tin như tên mã độc, loại mã độc, tệp và các khoá registry được sinh ra khi chúng được thực thi cũng các mã hash dạng MD5 và SHA256,.. sẽ được sưu tầm trên các kho thông tin lớn và uy tín như virustotal.com và virusshare.com.

Với các thông tin về chữ ký (signatures) của mã độc sẽ được chúng em tự phân tích và lưu lại.

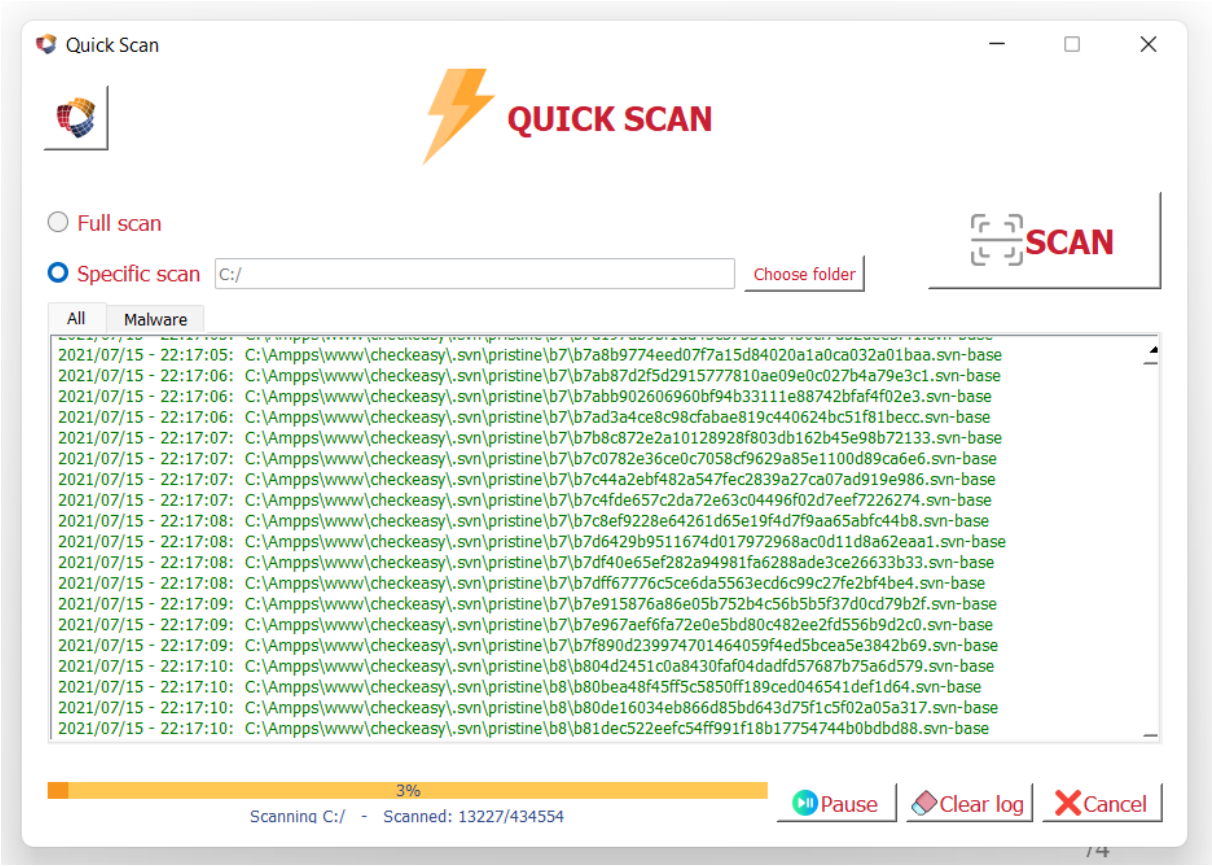
Ứng dụng hiện tại thực hiện phát hiện mã độc dựa vào cơ sở dữ liệu thu thập được như trên. Vì thế phạm vi mã độc mà ứng dụng có thể phát hiện được tỉ lệ thuận với độ lớn của cơ sở dữ liệu.

Với sự hỗ trợ cực kì mạnh mẽ từ công nghệ và các kho dữ liệu trên thế giới, việc thu thập thông tin của một tệp mã độc đã trở nên dễ dàng hơn nhiều. Tuy nhiên, với những mã độc mới được sinh ra, chưa có dữ liệu nhiều thì việc thu thập thông tin cũng như phát hiện cũng trở nên khó khăn hơn nếu như hệ thống vẫn chưa được nâng cấp khả năng phân tích.

Số lần thử	Số lượng tệp mã độc	Số lượng tệp không có dữ liệu	Phát hiện được	Không phát hiện được	Tỉ lệ chính xác (%)
3	7	3	4	3	100
4	10	3	7	3	100
4	30	7	23	7	100

Bảng 4-3 Khả năng phát hiện mã độc của ứng dụng

4.4.2 Hiệu năng



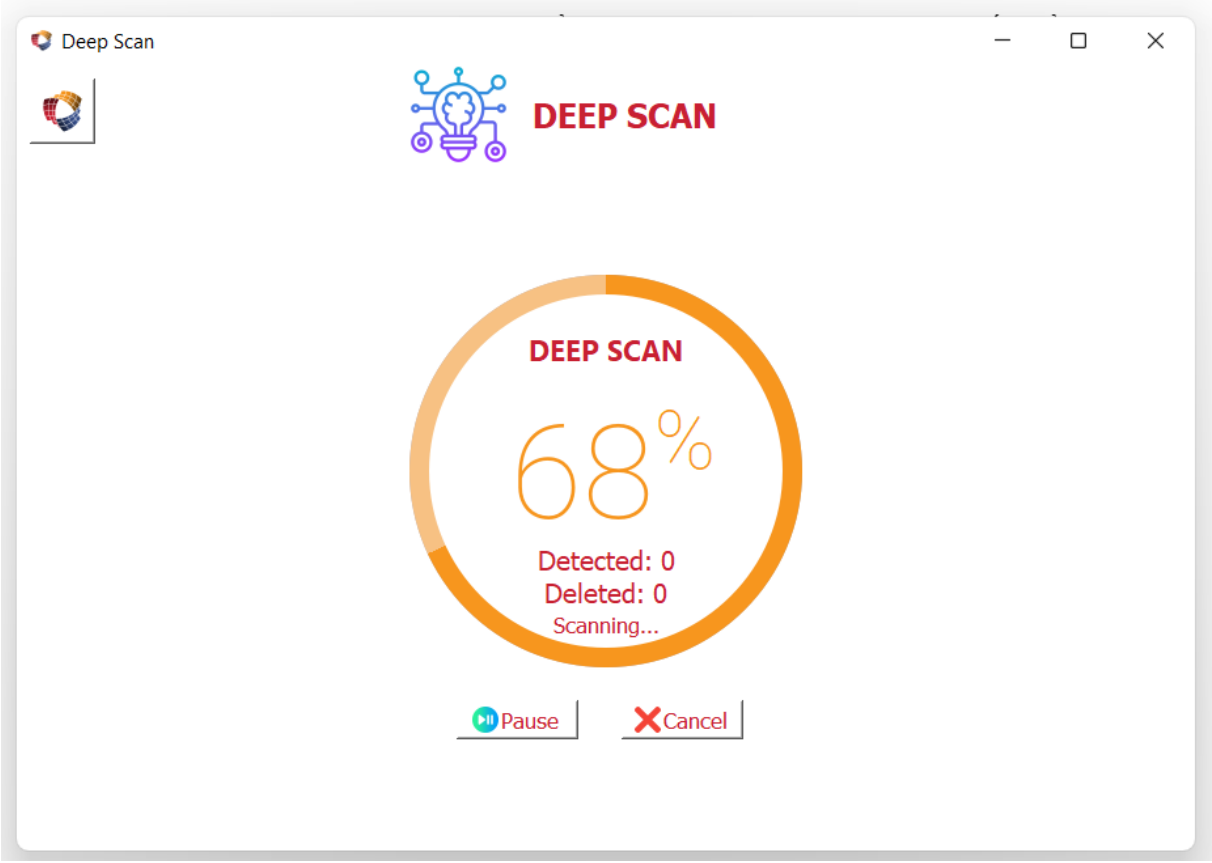
Hình 4.13 Phân tích ổ đĩa C

Hiện tại ứng dụng còn chưa thể tích hợp được các thuật toán tìm kiếm để nâng cao hiệu năng tìm kiếm và phát hiện mã độc. Do đó thời gian có thể quét được xong một thư mục hoặc một ổ đĩa của ứng dụng còn phụ thuộc khá nhiều vào sức mạnh xử lý của chip xử lý. Tốc độ hiện tại có thể quét được là từ 4 – 6 tệp trong thời gian 1 giây. So với tốc độ của các ứng dụng thông dụng thì tốc độ của ứng dụng chậm hơn khá nhiều.

Ví dụ như hình, quét cả ổ đĩa C:/ với tổng cộng 434554 tệp, hệ thống phân tích với tốc độ khoảng từ 4 – 6 tệp trên 1 giây. Cụ thể phân tích 13227 tệp trong thời gian khoảng 50 phút.

Số lần thử	Số lượng tệp (tệp)	Thời gian (phút)
5	12302	48
5	15523	61
5	30131	130

Bảng 4-4 Hiệu năng của ứng dụng



Hình 4.14 Deep scan

Về chức năng "deep scan", ứng dụng vẫn phụ thuộc vào cơ sở dữ liệu để tìm sự tồn tại của các tệp mã độc. Hiệu năng của chức năng sẽ phụ thuộc vào độ lớn và độ chính xác của tập dữ liệu.

4.4.3 Xử lý sau khi phát hiện được

Như đã trình bày ở chương 4, chức năng "quick scan" sẽ phát hiện mã độc dựa trên các mã hash và các chữ ký đặc trưng của mã độc. Và những tệp mã độc này là các tệp dưới dạng chưa được thực thi. Khi phát hiện, chương trình sẽ chủ động cô lập tệp

tin đó bằng các nén lại để tệp không thể thực thi được nữa và di chuyển vào một thư mục quản lý riêng. Để có thể thực hiện cô lập, chúng ta sử dụng sự hỗ trợ của "***UPX".***

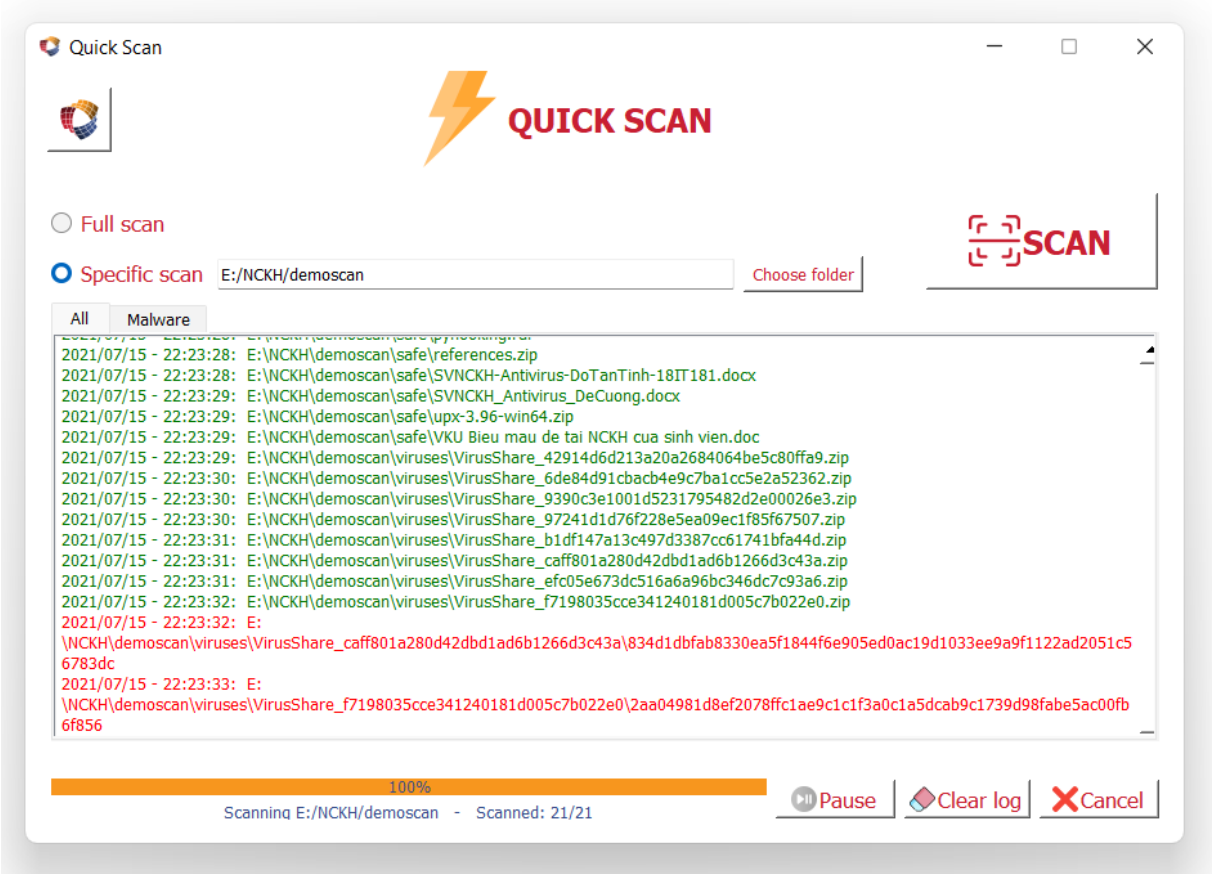
Đối với chức năng "deep scan", hệ thống có thể phát hiện được các tệp tin và các khoá registry được sinh ra khi mã độc được thực thi. Vì các hành vi của mã độc chủ yếu gây hại cho thiết bị và dữ liệu của chúng ta nên hệ thống sẽ chủ động xoá các tệp và các khoá registry này khỏi thiết bị.

4.4.4 In nhật kí

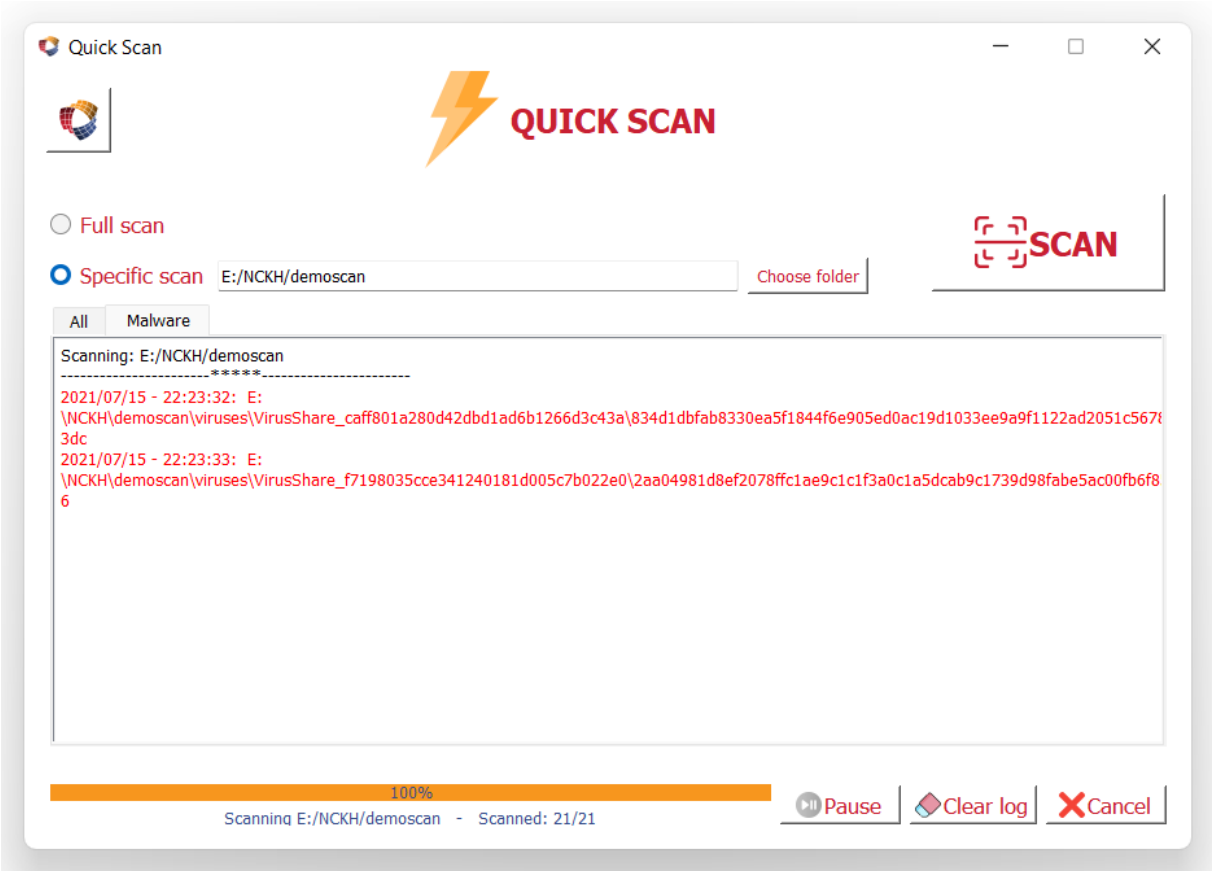
Để có thể tránh được các thời gian chết của ứng dụng như thời gian tính toán tổng số lượng tệp tin của thư mục hoặc thời gian tiến hành các thuật toán phân tích thì ứng dụng được thiết kế dựa trên kĩ thuật đa luồng. Luồng chính sẽ dùng để chuyển giao giữa các chức năng, điều khiển ở giao diện người dùng. Trong khi đó, luồng tính toán và luồng phân tích sẽ chạy song song với nhau thực hiện nhiệm vụ riêng. Luông tính toán tính toán tổng số lượng tệp để có thể trả về cho luồng chính hiển thị lên ứng dụng. Luồng phân tích sẽ trả về đường dẫn và kết quả của tệp vừa quét xong. Dựa vào kết quả trả về có phải mã độc hay không mà luồng chính sẽ thực hiện in nhật kí.

Nếu là tệp an toàn sẽ in đường dẫn với màu xanh lục ở tab "***All***".

Nếu là tệp mã độc sẽ được in đường dẫn với màu đỏ ở cả tab "***All***" và tab "***Malware***".

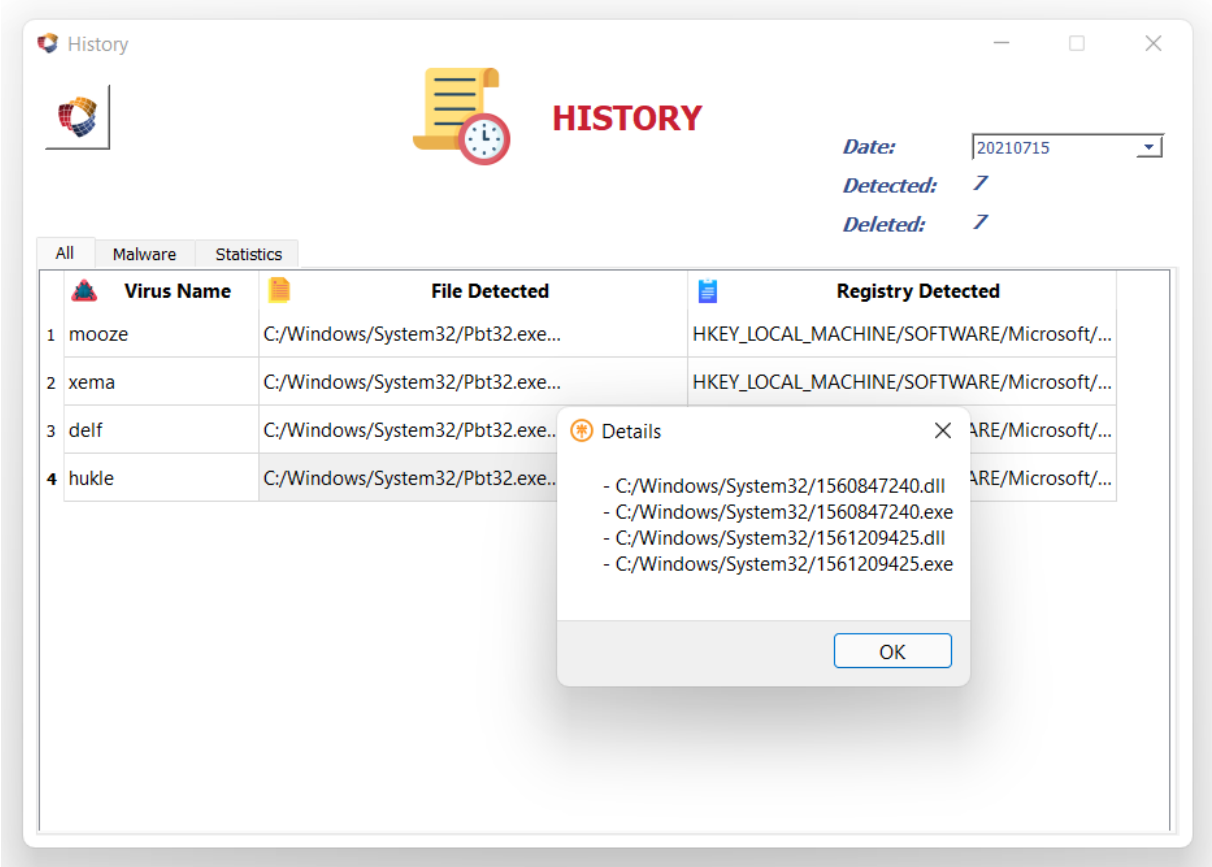


Hình 4.15 Tab All



Hình 4.16 Tab Malware

4.4.5 Lịch sử



Hình 4.17 Lịch sử phân tích

Sau khi phát hiện, cô lập, xoá các tệp và các khoá registry thì sẽ được lưu lại thông tin như tên của mã độc, các tệp và registry đã xoá.

Người dùng có thể xem được tổng số lượng tệp đã tìm thấy và xoá. Cũng như có thể chọn xem những ngày trước.

KẾT LUẬN

Virus tin học hiện đang là nỗi băn khoăn lo lắng của những người làm công tác tin học, là nỗi sợ của những người sử dụng khi máy tính của mình bị nhiễm virus. Cùng với sự phát triển của tin học thì virus cũng đang ngày một phát triển theo hướng phức tạp hơn, tinh vi hơn, dễ dàng tiếp cận mục tiêu của những phần tử xấu.

Với việc ứng dụng trí tuệ nhân tạo vào phần mềm, việc đào tạo và dự đoán được nâng lên rất nhiều nhờ vào những thuật toán của chúng. Nó có thể phân tích tất cả dữ liệu thô trong tệp và khai thác chúng một cách hiệu quả hơn. Đây là một bước đệm nhảy vọt trong khoa học máy tính nói chung và an ninh mạng nói riêng. Nó giúp nâng cao mức độ bảo vệ hơn, với tỉ lệ phát hiện phần mềm độc hại không xác định cao hơn, phát hiện, ngăn chặn phần mềm độc hại một cách nhanh chóng trước khi chúng được thực thi và có thể bảo vệ người dùng theo thời gian thực.

Quan việc nghiên cứu chúng thì chúng ta đã có một cách nhìn nhận cơ bản về hệ thống, cơ chế và các nguyên tắc họa động của virus tin học. Trên cơ sở đó, có một cách nhìn đúng đắn về virus tin học trong việc phòng chống, kiểm tra, tiêu diệt cũng như phân tích, nghiên cứu một virus mới xuất hiện mà ít gặp trở ngại hơn.

TÀI LIỆU THAM KHẢO

1. https://virusshare.com/
2. https://www.virustotal.com/
3. https://securitybox.vn/2161/ma-doc-la-gi-7-loai-ma-doc-pho-bien/
4. https://quantrimang.com/tim-hieu-ve-windows-registry-phan-i-2077
5. http://www.ngthanhbinh.com/gioi-thieu-lap-trinh-windows-api/
6. https://www.kaspersky.com/blog/signature-virus-disinfection/13233/
7. https://www.json.org/json-vi.html
8. https://mlab.vn/index.php?_route_=17161-bai-7-lap-trinh-giao-dien-voi-pyqt5-cho-raspberrypi-phan-1.html
9. https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh).
10. https://securelist.com/kaspersky-security-bulletin-2020-2021-eu-statistics/102335/
11. https://us-cert.cisa.gov/ncas/tips/ST18-271
12. https://docs.microsoft.com/en-us/windows/win32/apiindex/windows-api-list
13. https://en.wikipedia.org/wiki/Hash_function
14. https://en.wikipedia.org/wiki/Portable_Executable
15. https://www.lifewire.com/brief-history-of-malware-153616

PHỤ LỤC

Ý kiến của hội đồng:

Tác giả lưu ý cân nhắc các vấn đề sau đây để hoàn thiện đề cương và triển khai thực hiện đề tài:

- Cần hoàn thiện đề cương theo mẫu, lưu ý thay SICT bằng VKU

ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC CNTT VÀ TRUYỀN THÔNG VIỆT - HÀN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – tự do – hạnh phúc

Đà Nẵng, ngày 15 tháng 04 năm 2021

ĐỀ CƯƠNG NGHIÊN CỨU KHOA HỌC SINH VIÊN

(Năm học 2021)

1. Thông tin chung

+ Tên đề tài:

Nghiên cứu về virus máy tính và phát triển phần mềm phát hiện virus máy tính ứng dụng trí tuệ nhân tạo.

+ Chuyên ngành (*theo nội dung đề tài*):

Mạng máy tính và truyền thông (An ninh mạng)

+ Sinh viên chịu trách nhiệm chính:

- Họ và tên: Đỗ Tấn Tĩnh
- Mã số sinh viên: 18IT181
- Lớp: 18IT3 - Khoa: Khoa học máy tính
- Địa chỉ thường trú: K09/60 Hà Văn Trí, phường Khuê Trung, quận Cẩm Lệ, TP Đà Nẵng.
- Địa chỉ liên lạc: K09/60 Hà Văn Trí, phường Khuê Trung, quận Cẩm Lệ, TP Đà Nẵng.
- Số điện thoại: 0389909772
- Email: dttinh.18it3@vku.udn.vn

+ Các thành viên tham gia:

- Họ và tên: Đỗ Thanh Tùng
- Lớp: 18IT3 - Khoa: Khoa học máy tính
- Điện thoại: 0382352146
- Mã số sinh viên: 18IT186
- Email: dttung.18it3@vku.udn.vn

2. Tổng quan tình hình nghiên cứu ở trong nước và nước ngoài

Những năm gần đây đã chứng kiến sự phát triển nhanh chóng của các phần mềm độc hại cả về số lượng và chủng loại. Trong năm 1992 số lượng mã độc đã tăng từ 1000 lên 2300, năm 2002 có đến 60000 loại mã độc và các biến thể của chúng được phát hiện. Ngày này số lượng này đã tăng lên trên hơn 847 triệu mẫu mã độc tính đến cuối năm 2018. Trước sự phát triển nhanh chóng và sự tàn phá nặng nề của các phần mềm độc hại, trên thế giới đã có rất nhiều nghiên cứu về các phương pháp phát hiện và loại bỏ các phần mềm độc hại, nhưng nhìn chung các nghiên cứu này đều xoay quanh hai phương pháp chính phát hiện dựa trên sự bất thường hay dị thường và dựa trên dấu hiệu.

Vấn đề quan trọng nhất để phát hiện đúng mã độc là cần phải có phương pháp trích rút đặc trưng thực sự hiệu quả, trong đó có một phương pháp phát hiện mã độc sử dụng dấu hiệu đặc trưng byte n-grams được cho là có hiệu quả cao.

Mục tiêu của phân tích tĩnh có thể là mã nhị phân hoặc mã nguồn (theo Christodorescu and Jha - 2003). Trước tiên, một tệp PE cần được giải nén / giải nén nếu nó được nén bằng công cụ nén nhị phân của bên thứ ba (ví dụ: UPX và ASPack Shell) hoặc được nhúng trong một trình đóng gói tự chế. Để dịch ngược các tệp thực thi của cửa sổ, có thể sử dụng các công cụ trình tháo gỡ và kết xuất bộ nhớ. Các công cụ tháo rời (ví dụ: IDA Pro) hiển thị mã phần mềm độc hại dưới dạng hướng dẫn lắp ráp Intel ×86. Các công cụ kết xuất bộ nhớ (ví dụ: OllyDump và LordPE) được sử dụng để lấy các mã được bảo vệ nằm trong bộ nhớ chính và kết xuất chúng vào một tệp. Kết xuất bộ nhớ khá hữu ích để phân tích các tệp thực thi được đóng gói khó tháo rời. Sau khi tệp thi hành được giải nén được đóng gói và giải mã, các mẫu phát hiện được sử dụng trong phân tích tĩnh có thể được trích xuất, chẳng hạn như lệnh gọi API Windows, byte n-gram, chuỗi, mã opcodes (mã hoạt động) và đồ thị luồng điều khiển.

Ngoài ra còn có nhiều tính năng tĩnh khác để đại diện cho các mẫu tệp, chẳng hạn như thuộc tính tệp, thông tin tài nguyên tệp và bảng xuất. Phân tích tĩnh có thể khám phá / điều tra tất cả các đường dẫn thực thi có thể có trong các mẫu phần mềm độc hại. Do đó, nó có lợi thế là toàn diện trong việc phát hiện logic độc hại. Nói cách khác, phân tích tĩnh không có vấn đề bao trùm mà phân tích động gặp phải. Một ưu điểm khác của phân tích tĩnh là bộ phân tích không thể bị phần mềm độc hại đang nghiên cứu tấn công hay vô hiệu hóa. Một nhược điểm của phân tích tĩnh là, khi xử lý các tình huống nhất định, tính bất khả thi của nó do không xác định được (ví dụ: chuyển điều khiển gián tiếp thông qua con trỏ hàm). Vì vậy, sự cân bằng giữa độ chính xác và hiệu quả phải được thực hiện bất cứ khi nào có liên quan đến phân tích điểm đến. Những hạn chế khác của phân tích tĩnh bao gồm thiếu hỗ trợ cho mã đóng gói thời gian chạy và hạn chế liên quan đến quá trình xáo trộn phức tạp. Moser và cộng sự vào năm 2007 đã thảo luận về những hạn chế của phân tích tĩnh. Họ gợi ý rằng phân tích động có thể là một bổ sung cần thiết và hữu ích cho phân tích tĩnh.

Các kỹ thuật phân tích động (ví dụ: gỡ lỗi và lập hồ sơ) quan sát việc thực thi (trên bộ xử lý thực hoặc ảo) của các tệp PE để rút ra các tính năng. Có thể áp dụng nhiều kỹ thuật khác nhau, chẳng hạn như các điểm mở rộng tự động khởi động, phân tích tham số chức năng, giám sát cuộc gọi chức năng, theo dõi luồng thông tin và dấu vết hướng dẫn có thể được áp dụng để thực hiện phân tích động. Đã có đáng kể các nghiên cứu về phân tích động của phần mềm độc hại, khác nhau về môi trường thực thi đối với phần mềm độc hại và mức độ chi tiết của phân tích.

3. Tính cấp thiết của đề tài

Ngày nay thời đại công nghệ thông tin, công việc thực hiện với máy tính cá nhân ngày càng nhiều, đặc biệt là việc trao đổi tệp tin với nhau ngày càng dễ dàng và phổ biến. Tuy nhiên sự thuận tiện ấy cũng tiềm ẩn nhiều nguy bị cơ tấn công từ các tổ chức bất chính. Theo số liệu thống kê của các tổ chức an ninh mạng như Kaspersky, Symantec, số lượng tấn công có chủ đích (APT) thông qua mã độc trong những năm gần đây ngày một gia tăng cả về số lượng lẫn mức độ nghiêm trọng. Vì vậy, việc phát hiện các mã độc để hạn chế tối đa tác hại của chúng lên hệ thống máy tính là rất cần thiết nhằm đảm bảo an toàn, an ninh mạng. Đặc biệt trong xu thế chuyển đổi số đang diễn ra mạnh mẽ ở nước ta trên nhiều lĩnh vực và cấp độ khác nhau, việc phát hiện và phòng ngừa các cuộc tấn công bằng mã độc có một vai trò hết sức quan trọng.

Vì lẽ đó, đề tài nhằm mục đích nghiên cứu các loại mã độc, các kỹ thuật phát hiện (phân loại) mã độc dựa vào các đặc trưng của mã độc, trên cơ sở đó huấn luyện các mô hình trí tuệ nhân tạo để phát hiện mã độc để xây dựng một phần mềm có khả năng phát hiện và diệt mã độc khi xâm nhập vào máy tính

4. Mục tiêu của đề tài

Hiểu rõ về virus máy tính, các đặc trưng của virus máy tính.

Phát triển phần mềm cho phép phát hiện sự tồn tại của virus trên máy tính dựa vào các đặc trưng của virus.

5. Phương pháp nghiên cứu

Về mặt lý thuyết: Nghiên cứu tổng quan về các loại mã độc, cách thức lây lan phá hoại của chúng. Nghiên cứu các phương pháp phát hiện mã độc phổ biến hiện nay.

Về mặt thực nghiệm: dựa trên cơ sở lý thuyết đề xuất phương pháp trích rút đặc trưng cho bài toán phát hiện mã độc, đồng thời tiến hành xây dựng một phần mềm có khả năng phát hiện mã độc.

6. Đối tượng và phạm vi nghiên cứu

a) Đối tượng nghiên cứu:
- Nghiên cứu các phương pháp phân tích mã độc
- Nghiên cứu các phương pháp phát hiện và loại bỏ mã độc.
- Nghiên cứu các phương pháp trích rút đặc trưng mã độc.
- Nghiên cứu các phương pháp tạo cơ sở dữ liệu mẫu mã độc.
- Nghiên cứu phần mềm mã nguồn mở BKAV.
- Tìm hiểu và nghiên cứu các mô hình học máy (trí tuệ nhân tạo) cho phép phát hiện mã độc hiệu quả.

b) Phạm vi nghiên cứu:
- Các vấn đề liên quan đến phân tích mã độc.
- Các vấn đề liên quan đến mã độc.
- Các vấn đề về trích rút đặc trưng mã độc.
- Các tiêu chí đánh giá hiệu quả, chất lượng của các phương pháp phát hiện và trích rút đặc trưng.
- Nghiên cứu phát triển phần mềm cho phép phát hiện các mã độc dựa trên các đặc tính của chúng ứng dụng trí tuệ nhân tạo.

7. Nội dung nghiên cứu

- Tìm hiểu lập trình Win32 API.
- Tìm hiểu về các loại mã độc, cách thức lây lan phá hoại của chúng.
- Tìm hiểu các phương pháp phát hiện mã độc.
- Tìm hiểu các phương pháp trích rút đặc trưng mã độc
- Phân tích và thiết kế hệ thống (biểu đồ UC, SD, Class, ERD, …)
- Xây dựng CSDL về đặc trưng về các mã độc hiện có.
- Triển khai xây dựng phần mềm (giao diện, chức năng, ….)
- Kiểm thử - Đánh giá hoạt động
- Hiệu chỉnh và hoàn thiện chương trình.

8. Kết quả nghiên cứu

Nắm được cơ chế hoạt động của mã độc và các mô hình phát hiện mã độc hiện đang được áp dụng. Từ đó xây dựng phần mềm phát hiện và tiêu diệt mã độc khi xâm nhập vào máy tính.

9. Sản phẩm

Phần mềm diệt virus ứng dụng trí tuệ nhân tạo.

10. Về các đóng góp của đề tài đến giáo dục và đào tạo, kinh tế xã hội và an ninh quốc phòng

Phát hiện và ngăn chặn nạn ăn cắp thông tin, cài mã độc làm ảnh hưởng hoạt động của hệ thống, ngăn chặn nguy cơ mất mát dữ liệu.

Người hướng dẫn *(Ký và ghi rõ họ tên)*	**Sinh viên chịu trách nhiệm chính** *(Ký và ghi rõ họ tên)*
TS. Trần Thế Sơn	Đỗ Tấn Tĩnh