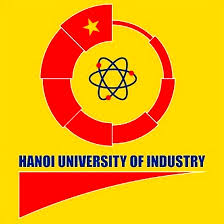
Trường Đại Học Công Nghiệp Hà Nội

Khoa công nghệ thông tin

****

Bài tập lớn môn học: OS

**Đề tài:** Nghiên cứu tìm hiểu về Hệ thống bảo vệ trong HĐH Windows.

Giáo viên: Ths Nguyễn Tuấn Tú

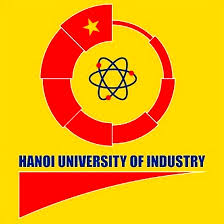
Nhóm số: 11

Lớp: IT6025.6(006)K15

Hà Nội, 2022

Trường Đại Học Công Nghiệp Hà Nội

Khoa công nghệ thông tin

****

Bài tập lớn môn học: OS

**Đề tài:** Nghiên cứu tìm hiểu về Hệ thống bảo vệ trong HĐH Windows.

Giáo viên: Ths Nguyễn Tuấn Tú

Sinh viên thực hiện:

1. Phạm Sỹ Bình

2. Nguyễn Hữu Đại

3. Trương Ngọc Đăng

4. Nguyễn Thị Diệu Linh

5. Bùi Quang Minh

Lớp: IT6025.6(006) K15

Hà Nội, 2022

Mục lục

[Lời nói đầu 2](#_Toc101388204)

[CHƯƠNG 1: BẢO VỆ HỆ THỐNG 3](#_Toc101388205)

[1.1. Mục tiêu của bảo vệ hệ thống 3](#_Toc101388206)

[1.2. Nguyên tắc bảo vệ (Principles of Protection) 3](#_Toc101388207)

[1.3. Miền bảo vệ (Domain of Protection) 4](#_Toc101388208)

[1.3.1. Khái niệm miền bảo vệ. 4](#_Toc101388209)

[1.3.2. Cấu trúc miền bảo vệ 4](#_Toc101388210)

[1.4. Ma trận quyền truy nhập 6](#_Toc101388211)

[1.4.1. Khái niệm về ma trận quyền truy nhập 6](#_Toc101388212)

[1.4.2 Các phương pháp cài đặt ma trận quyền truy cập 10](#_Toc101388213)

[1.5. Kiểm soát quyền truy nhập (Access Control) 12](#_Toc101388214)

[1.6. Thu hồi quyền truy nhập (Revocation of Access Rights) 13](#_Toc101388215)

[CHƯƠNG 2 : AN NINH 15](#_Toc101388216)

[2.1. Các vấn đề về bảo vệ hệ thống 15](#_Toc101388217)

[2.2. Các cơ chế an toàn hệ thống 15](#_Toc101388218)

[2.2.1 Kiểm định danh tính 15](#_Toc101388219)

[2.2.2 Ngăn chặn nguyên nhân từ phía các chương trình 16](#_Toc101388220)

[2.3 Ngăn chặn nguyên nhân từ phía hệ thống 19](#_Toc101388221)

[2.4 Giám sát các nguyên nhân 19](#_Toc101388222)

[CHƯƠNG 3 : VIRUS MÁY TÍNH 20](#_Toc101388223)

[3.1. Khái niệm về virus 20](#_Toc101388224)

[3.3. Cơ chế hoạt động của virus 21](#_Toc101388225)

[4.4. Phòng tránh virus 23](#_Toc101388226)

[4.4.1 Các chương trình phòng tránh và phát hiện virus 23](#_Toc101388227)

[4.4.2 Một số biện pháp phòng chống virus 25](#_Toc101388228)

# Lời nói đầu

Từ trước tới nay, vấn đề bảo mật trên hệ điều hành Windows luôn nhận được nhiều sự quam tâm của cộng đồng IT, bởi Windows là hệ điều hành phổ biến nhất trên toàn thế giới. Hệ thống bảo vệ của Windows trên từng phiên bản cụ thể luôn nhận được nhiều lời khen nhưng cũng không ít lời phàn nàn.

*Như chúng ta đã biết, bất cứ một chương trình nào khi được thiết kế để chạy trên máy tính thì không chỉ được chú tâm đến nội dung, hình thức...của chương trình đó mà có một thành phần cấu thành nên chương trình đó và phải có thành phần này thì chương trình mới hoạt động ổn định và hiệu quả. Đó chính là hệ thống bảo vệ, cũng như bất kể các chương trình máy tính nào đó thì chính hệ điều hành mà chúng ta sử dụng để vận hành các chương trình đó cũng cần có hệ thống bảo vệ và nó là một thành phần quyết định là hệ điều hành đó có tồn tại được và phát triển được hay không?*

Vào tháng 1 năm 2002, hệ thống bảo mật của Microsoft đã có những sáng tạo nổi bật. Tuy nhiên, thành quả đạt được về bảo mật của Microsoft chưa thể bao phủ toàn bộ các sản phẩm của hãng. Mặc dù, gặp phải một số rắc rối sau khi mới phát hành nhưng Windows Server 2003 vẫn được các chuyên gia công nghệ thông tin đánh giá là an toàn hơn nhiều so với tất cả các phiên bản hệ điều hành trước đó của hãng.

Trong đề tài này, chúng ta sẽ nghiên cứu và tìm hiểu một cách chi tiết nhất về hệ thống bảo vệ trong Windows, những hệ thống bảo vệ từ sơ sài cho đến tinh vi tương ứng với những hệ điều hành đã phát hành của hãng Windows như: WinNT, Windows 98, Windows 2000, Windows 2003…thì cứ mỗi lần phát triển hệ điều hành Windows cho ra một phiên bản mới thì tất yếu là phiên bản phải có những chức năng được cải tiến, trong đó chắc chắn có sự cải tiến về bảo mật.

An toàn và bảo vệ hệ thống là chức năng không thể thiếu của các hệ điều hành. Trong đề tài này, chúng ta sẽ làm quen với các khái niệm về tổ chức hệ thống cũng như các cơ chế bảo vệ hỗ trợ việc triển khai chiến lược.

# CHƯƠNG 1: BẢO VỆ HỆ THỐNG

## 1.1. Mục tiêu của bảo vệ hệ thống

Một hệ điều hành đa nhiệm có thể thực hiện đồng thời nhiều tiến trình tại cùng một thời điểm. Khi đó chắc chắn sẽ có hai hay nhiều tiến trình hoạt động song hành trong hệ thống, ngẫu nhiên có thể phát sinh lỗi của một tiến trình và lỗi của tiến trình đó có thể gây ảnh hưởng đến các tiến trình khác đang hoạt động đồng thời trong hệ thống. Vì vậy, để bảo vệ hệ thống khỏi sự lây lan lỗi của một tiến trình này đến các tiến trình khác thì hệ thống phải có chức năng ngăn chặn, không cho lan truyền trên hệ thống làm ảnh hưởng đến các tiến trình khác. Đặc biệt, qua việc phát hiện các lỗi tiềm ẩn trong các thành phần của hệ thống, có thể tăng cường độ tin cậy của hệ thống.

Hệ thống đảm bảo các bộ phận của tiến trình sử dụng tài nguyên theo một cách thức hợp lệ được quy định cho nó trong việc khai thác tài nguyên này.

Vai trò của bộ phận bảo vệ trong hệ thống là cung cấp một cơ chế để áp dụng các chiến lược quản trị việc sử dụng tài nguyên. Cần phân biệt rõ giữa khái niệm cơ chế và chiến lược của bộ phận bảo vệ trong hệ thống:

* *Cơ chế*: xác định làm thế nào để thực hiện việc bảo vệ, có thể có các cơ chế phần mềm hoặc cơ chế phần cứng.
* *Chiến lược*: quyết định việc bảo vệ được áp dụng như thế nào, những đối tượng nào trong hệ thống cần được bảo vệ, và các thao tác thích hợp trên các đối tượng này.

Để hệ thống có độ tương thích cao, cần phân tách các cơ chế và chiến lược được sử dụng trong hệ thống. Các chiến lược sử dụng tài nguyên là khác nhau tùy theo ứng dụng và thường để thay đổi. Thông thường các chiến lược được lập trình viên vận dụng vào ứng dụng của mình để chống lỗi truy xuất bất hợp lệ đến các tài nguyên trong khi đó hệ thống cung cấp các cơ chế giúp người sử dụng có thể thực hiện được chiến lược bảo vệ của mình.

## 1.2. Nguyên tắc bảo vệ (Principles of Protection)

Các nguyên tắc bắt buộc đặc quyền tối thiểu (The principle of least privilege dictates) là một loạt các quy tắc cho rằng các chương trình, người sử dụng, và các hệ thống chỉ đủ quyền để thực hiện các nhiệm vụ của các chương trình, người sử dụng và các hệ thống đó. Các nguyên tắc trên được đặt ra và nó đảm bảo sự thiệt hại do các lỗi của 1 hay nhiều tiến trình phát sinh là ít nhất và hầu như không xảy ra nếu các tiến trình thực hiện đúng các quy tắc đó.

Thông thường, mỗi một tiến trình được cấp những quyền đã được quy định cho tiến trình đó thì những gì tiến trình có thể thực hiện được chỉ nằm trong phạm vi quyền của tiến trình đó.

## 1.3. Miền bảo vệ (Domain of Protection)

### 1.3.1. Khái niệm miền bảo vệ.

Một hệ thống máy tính bao gồm tập hợp các chủ thể (subject’s) và tập hợp các khách thể (object’s). Chủ thể bao gồm các tiến trình và người sử dụng còn khách thể có thể coi là các tài nguyên của máy tính (như bộ nhớ, ổ đĩa, dữ liệu…).

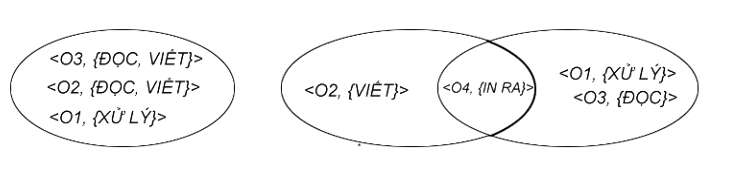
Để có thể kiểm soát được tình trạng sử dụng tài nguyên trong hệ thống, hệ điều hành chỉ cho phép các chủ thế truy cập tới các khách thể mà nó có quyền sử dụng và vào những thời điểm cần thiết (nguyên lý need – to - know) nhằm hạn chế các lỗi xảy ra do tranh chấp tài nguyên.

Mỗi chủ thể trong hệ thống sẽ hoạt động trong một miền bảo vệ (protection domain) nào đó. Một miền bảo vệ sẽ xác định các khách thể mà chủ thể trong miền đó được phép truy nhập và thực hiện thao tác.

Ví dụ: < File F, {read, write}>

### 1.3.2. Cấu trúc miền bảo vệ

Các khả năng thao tác mà chủ thể có thể thực hiện trên khách thể được gọi là quyền truy cập (Access Right), mỗi miền định nghĩa một tập hợp các đối tượng và các hoạt động mà có thể thể hiện trên từng đối tượng. Mỗi quyền truy nhập được định nghĩa bởi một bộ hai thành phần <đối tượng, {quyền thao tác}> (<object, {access right}>). Như vậy, ta có thể hình dung miền bảo vệ là một tập hợp các quyền truy nhập, xác định các thao tác mà chủ thể có thể thực hiện trên các khách thể. Các miền bảo vệ khác nhau có thể giao nhau một số quyền truy cập.



*Hình 1.1: Hệ thống với 3 miền bảo vệ*

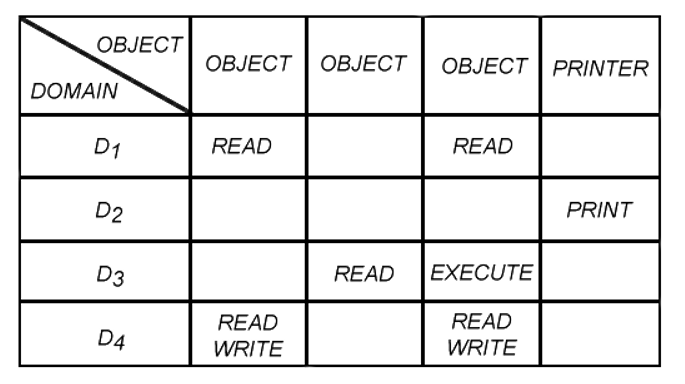
* Sự liên kết giữa một quá trình và một miền có thể là **tĩnh** hoặc **động**:
* **Liên kết tĩnh**: trong suốt thời gian tồn tại của tiến trình trong hệ thống, tiến trình chỉ hoạt động trong một miền bảo vệ. Trong trường hợp tiến trình trải qua các giai đoạn xử lý khác nhau, ở mỗi giai đoạn nó có thể thao tác trên những tập tài nguyên khác nhau. Như vậy trong liên kết tĩnh, miền bảo vệ phải xác định ngay từ đầu các quyền truy nhập cho các tiến trình trong tất cả các giai đoạn xử lý. Điều này khiến cho tiến trình sẽ được dư thừa quyền trong một giai đoạn xử lý nào đó và vi phạm nguyên lý need – to – know. Để đảm bảo được nguyên lý này cần phải có khả năng cập nhật nội dung miền bảo vệ qua các giai đoạn xử lý khác nhau để đảm bảo các quyền tối thiểu của tiến trình trong miền bảo vệ tại một thời điểm.
* **Liên kết động**: cơ chế này cho phép tiến trình chuyển đổi từ miền bảo vệ này sang miền bảo vệ khác trong suốt thời gian tồn tại trong hệ thống của nó. Để tuân thủ nguyên lý need – to – know, thay vì phải sửa đổi nội dung miền bảo vệ hệ thống có thể tạo ra các miền bảo vệ mới với nội dung thay đổi tùy theo từng giai đoạn xử lý của tiến trình và chuyển tiến trình sang hoạt động tại các miền bảo vệ phù hợp với từng thời điểm.
* Một miền bảo vệ có thể được xây dựng cho:
* **Một người sử dụng**: trong trường hợp này, tập các đối tượng được phép truy xuất phụ thuộc vào định danh của người sử dụng, miền bảo vệ được chuyển khi thay đổi người sử dụng.
* **Một tiến trình**: trong trường hợp này, tập các đối tượng được phép truy xuất phụ thuộc vào định danh của tiến trình, miền bảo vệ được chuyển khi quyền điều khiển được chuyển sang tiến trình khác.
* **Một thủ tục**: trong trường hợp này, tập các đối tượng được phép truy xuất là các biến cục bộ được định nghĩa bên trong thủ tục, miền bảo vệ được chuyển khi thủ tục được gọi.

## 1.4. Ma trận quyền truy nhập

### 1.4.1. Khái niệm về ma trận quyền truy nhập

Để biểu diễn miền bảo vệ, các hệ điều hành sẽ cài đặt các ma trận quyền truy cập trong đó các hàng của ma trận biểu diễn các miền bảo vệ, các cột biểu diễn khách thể. Phần tử (i,j) của ma trận xác định quyền truy nhập của chủ thể miền bảo vệ Di, có thể thao tác vớ khách thể Oj.

Một cạch trừu tượng, có thể biểu diễn mô hình bảo vệ trên đây như một ma trận quyền truy xuất (assces matrix). Các dòng của ma trận biểu diễn các miền bảo vệ và các cột tương ứng với các đối tượng trong hệ thống. Phần tử acess[i,j] của ma trận xác định các quyền truy xuất mà một tiến trình hoạt động trong miền bảo vệ Di có thể thao tác trên đối tượng Oj.



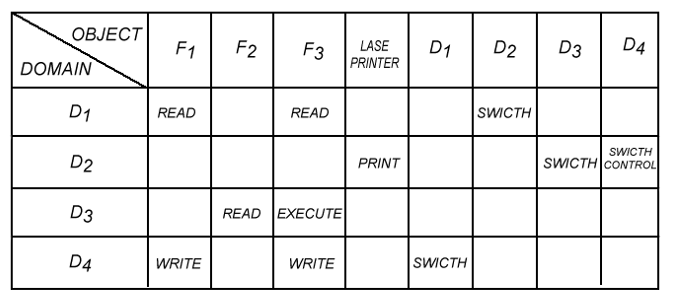
*Hình 1.2: Ma trận truy nhập*

Cơ chế bảo vệ được cung cấp khi ma trận quyền truy xuất được cài đặt (với đầy đủ các thuộc tính ngữ nghĩa mô tả trên lý thuyết), lúc này người sử dụng có thể áp dụng các chiến lược bảo vệ bằng cách đặc tả nội dung các phần tử tương ứng trong ma trận, xác định các quyền truy xuất với từng miền bảo vệ, và cuối cùng hệ điều hành sẽ quyết định cho phép tiến trình hoạt động trong miền bảo vệ thích hợp.

Ma trận quyền truy xuất cũng cung cấp một cơ chế thích hợp để định nghĩa và thực hiện một sự kiểm soát nghiêm ngặt cho cả phương thức liên kết tĩnh và động với các miền bảo vệ.

Có thể kiểm soát việc chuyền đổi giữa các miền bảo vệ nếu quan niệm miền bảo vệ cũng là một đối tượng trong hệ thống, và bổ sung các cột mô tả cho nó trong ma trận quyền truy xuất.

Khi đó tiến trình được phép chuyển từ miền bảo vệ Di sang miền bảo vệ Dj nếu phần tử assces(i,j) chứa đựng quyền ở chuyển Oj (switch).

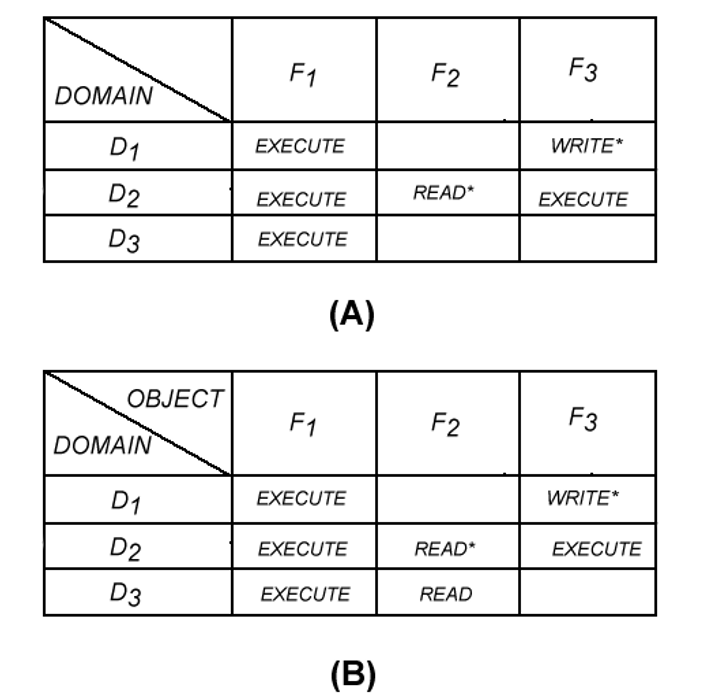


*Hình 1.3: Chỉnh sửa ma trận truy nhập*

Có thể kiểm soát việc sửa đổi nội dung ma trận (thay đổi các quyền truy xuất trong một miền bảo vệ) nếu quan niệm bản thân ma cũng là một đối tượng.

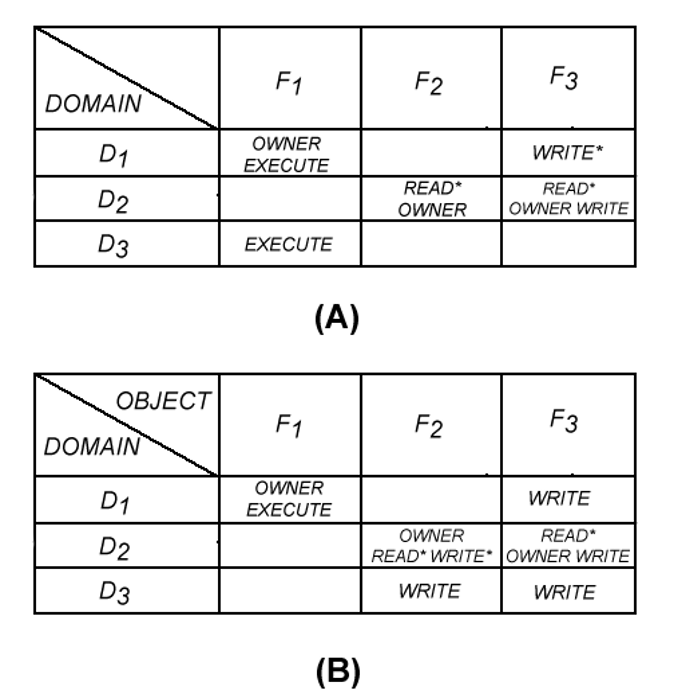
Các thao tác sửa đổi nội dung ma trận được phép thực hiện bao gồm: sao chép quyền (copy), chuyển quyền (transfer), quyền sở hữu (owner), và quyền kiểm soát (control).

* **Copy:** Nếu quyền truy xuất R trong assces[i,j] được đánh dấu là R\* thì có thể sao chép nó sang một phần tử assces[k,j] khác (mở rộng quyền truy xuất R trên cùng đối tượng Oj nhưng trong miền bảo vệ Dk).

****

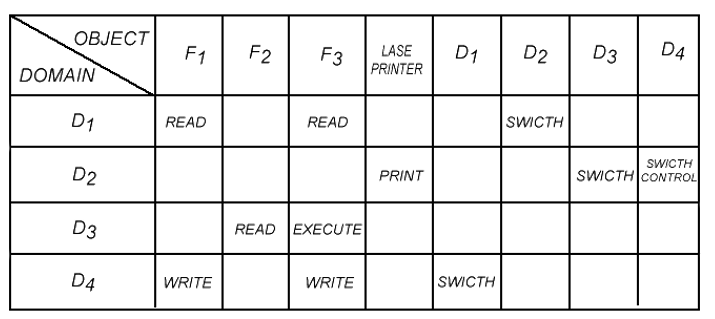
*Hình 1.4: Ma trận truy nhập với quyên copy*

* **Transfer:** Nếu một quyền truy xuất R trong assces[i,j] được đánh dấu là R+ thì có thể chuyển nó sang một phần tử assces[k,j] khác (chuyển quyền truy xuất R+ trên đối tượng Oj sang miền bảo vệ Dk).
* **Owner:** Nếu assces[i,j] chứa quyền truy xuất owner thì tiến trình hoạt động trong miền bảo vệ Di có thể thêm hoặc xóa các quyền truy xuất trong bất kỳ phần tử nào trên cột j (có quyền thêm hay bớt các quyền truy xuất trên đối tượng Oj trong những miền bảo vệ khác).



*Hình 1.5: Ma trận truy nhập với quyền Owner*

* **Control:** Nếu assces[i,j] chứa quyền truy xuất control thì tiến trình hoạt động trong miền bảo vệ Di có thể xóa bất kì quyền truy xuất nào trong các phần từ trên dòng j (có quyền bỏ bớt các quyền truy xuất trong miền bảo vệ Dj).



*Hình 1.6: Chỉnh sửa ma trận truy nhập*

### 1.4.2 Các phương pháp cài đặt ma trận quyền truy cập

***Bảng toàn cục (Global Table):*** Phương pháp này đơn giản nhất, để cài đặt ma trận quyền truy cập, hệ thống sử dụng 1 bảng toàn cục bao gồm các bộ ba thành phần<miền bảo vệ, khách thể, quyền truy cập> (<domain, object, rights>). Mỗi khi thực hiện quyền thao tác M trên khách thể Oj trong miền bảo vệ Di, cần tìm trong bảng toàn cục một bộ ba <Di, Oj, Rk> mà M thuộc Rk (truy cập các quyền truy nhập). Nếu tìm thấy thao tác M được phép ghi thành công, ngược lại sẽ xảy ra lỗi. Tuy nhiên phương pháp bảng toàn cục (Global table) có kích thước rất lớn nên không thể giữ trong bộ nhớ.

***Danh sách quyền truy nhập (Access Control List for Objects - ACL)***: Có thể cài đặt mỗi cột trong ma trận quyền truy xuất như một danh sách quyền truy xuất đối với một đối tượng. Mỗi đối tượng trong hệ thống sẽ có một danh sách bao gồm các phần tử là các bộ hai thứ tự <miền bảo vệ, các quyền truy xuất>, danh sách này sẽ xác định các quyền truy xuất được qui định trong từng miền bảo vệ có thể tác động trên đối tượng. Mỗi khi thực hiện thao tác M trên đối tượng Oj trong miền bảo vệ Di, cần tìm trong danh sách quyền truy xuất của đối tượng Oj một bộ hai < Di, Rk > mà M ở Rk. Nếu tìm thấy, thao tác M được phép thi hành, nếu không, xảy ra lỗi truy xuất.

*Ví dụ:* Một miền bảo vệ trong hệ thống UNIX được xác định tương ứng với một người sử dụng (uid) trong một nhóm (gid) nào đó. Giả sử có 4 người dùng: A, B, C, D thuộc các nhóm tương ứng là system, staff, student, student. Khi đó các tập tin trong hệ thống có thể có các ACL như sau:

File0: A, \*, RWX)

File1: (A, system, RWX)

File2: (A, \*, RW-), (B, staff, R--), (D ,\*, RW-)

File3: (\*, student, R--)

File4: (C, \*, ---), (\*, student, R--)

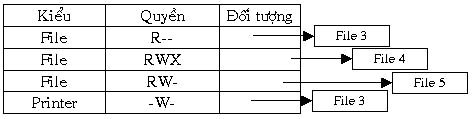
Thực tế, hệ thống tập tin trong UNIX được bảo vệ bằng cách mỗi tập tin được gán tương ứng 9 bit bảo vệ, từng 3 bit sẽ mô tả quyền truy xuất R(đọc), W(ghi) hay X(xử lý) của các tiến trình trên tập tin này theo thứ tự: tiến trình sở hữu các tiến trình cùng nhóm với tiến trình sở hữu, các tiến trình khác. Đây là một dạng ACL nhưng được nén thành 9 bit.

***Danh sách khả năng (Capability List for Domain):*** Mỗi dòng trong ma trận quyền truy xuất tương ứng với một miền bảo vệ sẽ được tổ chức thành một danh sách tiềm năng (capabilities list):

Một danh sách tiềm năng của một miền bảo vệ là một danh sách các đối tượng và các thao tác được quyền thực hiện trên đối tượng khi tiến trình hoạt động trong miền bảo vệ này.

Một phần tử của C-List được gọi là một tiềm năng (capability) là một hình thức biễu diển được định nghĩa một cách có cấu trúc cho một đối tượng trong hệ thống và các quyền truy xuất hợp lệ trên đối tượng này.

*Ví dụ:*



Tiến trình chỉ có thể thực hiện thao tác M trên đối tượng Oj trong miền bảo vệ Di, nếu trong C\_List của Di có chứa tiềm năng tương ứng của Oj.

Danh sách tiềm năng được gán tương ứng với từng miền bảo vệ, thực chất nó cũng là một đối tượng được bảo vệ bởi hệ thống, và tiến trình của người sử dụng chỉ có thể truy xuất đến nó một cách gián tiếp để tránh làm sai lạc C\_List.

Hệ điều hành cung cấp các thủ tục cho phép tạo lập, hủy bỏ và sửa đổi các tiềm năng của một đối tượng, và chỉ các tiến trình đóng vai trò server (thường là tiến trình hệ điều hành) mới có thể sửa đổi nội dung C\_List.

***Cơ chế khóa-chìa (A Lock-Key Mechanism):***

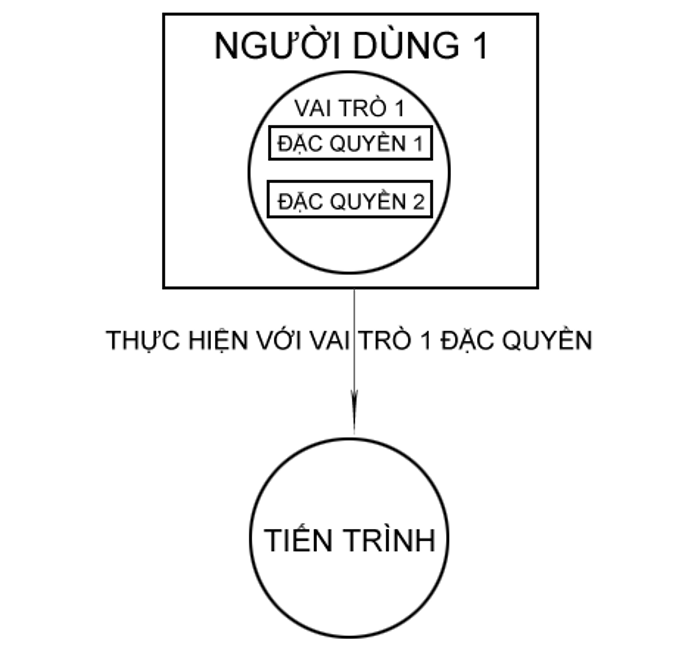
Phương pháp này thực chật là sự kết hợp giữa danh sách quyền truy nhập và danh sách khả năng. Mỗi khách thể sở hữu một danh sách các mã nhị phân gọi là chìa (key). Một chủ thể hoạt động trong miền bảo vệ sở hữu một chìa tương ứng với một khóa trong danh sách của khách thể.

Cũng như phương pháp danh sách khả năng, phương pháp khóa và chìa được quản lý bởi hệ điều hành, người sử dụng không thể truy nhập trực tiếp để thấy được nội dung của nó.

## 1.5. Kiểm soát quyền truy nhập (Access Control)

Access Control là chức năng của hệ thống được thi hành để cho phép chủ thể (người dùng, tiến trình, thiết bị) được truy cập đến 1 mức nào đó (quyền truy cập) tới tài nguyên của hệ thống và chia sẻ quyền truy cập này cho chủ thể khác.

*Ví dụ:* Những tính năng kiểm soát truy nhập được tích hợp sẵn trong Solaris 10 thể hiện qua các đặc quyền trong quy định về quyền cho user 1.

**

*Hình 1.7: Vai trò kiểm soát truy cập trong Solari 10*

## 1.6. Thu hồi quyền truy nhập (Revocation of Access Rights)

Trong nội dung bảo vệ hệ thống, đôi khi việc thu hồi một số quyền thao tác trên các khách thể của các chủ thể cũng được xem là một biện pháp bảo vệ. Khi thu hồi quyền truy nhập cần chú ý tới một số vấn đề sau:

* Thu hồi tức khắc hay trì hoãn và nếu trì hoãn thì tới bao giờ?
* Nếu loại bỏ một quyền truy cập của chủ thể tới một khách thể thì loại bỏ tất cả hay chỉ áp dụng với một số chủ đề.
* Thu hồi một số quyền hay toàn bộ quyền trên một khách thể?
* Thu hồi tạm thời hay vĩnh viễn một quyền truy cập?

Đối với các hệ thống sử dụng danh sách quyền truy nhập. Việc thực hiện quyền thu hồi truy nhập có thể thực hiện một cách dễ dàng bằng cách tìm và hủy trong ACL. Như vậy việc thu hồi sẽ có hiệu lực tức thời và có thể áp dụng cho tất cả các chủ thể hoặc một nhóm các chủ thể: thu hồi một cách vĩnh viễn hay tạm thời đều được.

Tuy nhiên, trong các hệ thống sử dụng danh sách khả năng, vấn đề thu hồi sẽ gặp nhiều khó khăn vì các khả năng được phát tán trên khách các miền bảo vệ trong hệ thống, do đó cần phải tìm ra đúng trước khi loại bỏ. Để giải quyết vấn đề này có thể tiến hành theo các phương pháp:

***Tái yêu cầu***: loại bỏ các khả năng ra khỏi miền bảo vệ sau mỗi chu kì. Nếu miền bảo vệ vẫn còn khả năng nào thì nó sẽ tái yêu cầu khả năng đó.

***Sử dụng con trỏ ngược***: Với mỗi khách thể sẽ tồn tại các con trỏ, trỏ đến các khả năng tương ứng của khách thể. Khi cần thu hồi quyền truy nhập nào trên khách thể hệ thống sẽ dựa vào các con trỏ để tìm kiếm các khả năng tương ứng.

***Sử dụng con trỏ gián tiếp***: trong phương pháp này con trỏ không trỏ trực tiếp tới các khả năng của khách thể mà trỏ tới một bảng toàn cục được quản lý bởi hệ điều hành. Khi cần thu hồi quyền truy nhập chỉ cần xóa phần tử tương ứng trong bảng này.

Trong các hệ thống sử dụng cơ chế khóa và chìa, khi cần thu hồi quyền chỉ cần thay đổi khóa và bắt buộc chủ thể thay đổi chìa khóa mới.

# CHƯƠNG 2: AN NINH

*Bảo vệ hệ thống là một cơ chế kiểm soát việc sử dụng tài nguyên của các chủ thể (tiến trình và người sử dụng). Để đối phó với các tình huống lỗi có thể phát sinh trong hệ thống. Trong khi đó khái niệm an toàn hệ thống muốn đề cập tới mức độ tin cậy mà hệ thống cần duy trì khi phải đối phó không những với các vấn đề nội bộ mà cả với những tác động đến từ môi trường bên ngoài*

## 2.1. Các vấn đề về bảo vệ hệ thống

Hệ thống được coi là an toàn nếu các tài nguyên được sử dụng đúng quy định trong mọi hoàn cảnh. Điều này khó có thể đạt được trong thực tế. Thông thường, cơ chế an toàn hệ thống bị vi phạm vì các nguyên nhân vô tình hoặc cố ý. Việc ngăn chặn các hành vi cố ý là rất khó khăn vì hầu như không thể đạt hiệu quả hoàn toàn.

Bảo đảm an toàn hệ thống ở cấp cao như chống lại các nguyên nhân hỏa hoạn, thiên tai, mất điện... cần được thực hiện ở mức độ vật lý (trang bị các thiết bị đảm bảo an toàn cho hệ thống) và nhân sự (chọn các nhân viên tin cậy làm việc trong hệ thống). Nếu an toàn môi trường được đảm bảo thì an toàn của hệ thống sẽ được duy trì tốt nhờ các cơ chế của hệ điêu hành.

Cần chú ý nếu bảo vệ hệ thống có thể đạt độ tin cậy 100% thì các cơ chế an toàn hệ thống được cung cấp chỉ nhằm ngăn chặn bớt các tình huống bất lợi hơn là đạt đến độ an toàn hệ tuyệt đối.

## 2.2. Các cơ chế an toàn hệ thống

### 2.2.1 Kiểm định danh tính

Để đảm bảo an toàn, hệ điều hành cần phải giải quyết tốt vấn đề kiểm định danh tính (authentication). Hoạt động của hệ thống bảo vệ phụ thuộc vào khả năng xác định các tiến trình đang xử lý. Khả năng này, đến lượt nó lại phụ thuộc vào việc xác định người dùng đang sử dụng hệ thống để có thể kiểm tra người dùng này được phép thao tác trên những tài nguyên nào.

Cách tiếp cận phổ biến nhất để giải quyết vấn đề là sử dụng mật khẩu (pass- word) để kiểm định danh tính người sử dụng. Mỗi khi người dùng muốn sử dụng một tài nguyên, hệ thống sẽ so sánh mật khẩu của họ nhập vào với mật khẩu được lưu trữ, nếu đúng họ mới được phép sử dung tài nguyên. Mật khẩu có thể được áp dụng để bảo vệ cho từng đối tượng trong hệ thống, thậm chí cùng một đối tượng sẽ có các mật khẩu khác nhau tương ứng với các quyền truy nhập khác nhau.

Cơ chế mật khẩu rất đơn giản và dễ sử dụng, do đó được các hệ điêu hành sử dụng rộng rãi, tuy nhiên điểm yếu nghiêm trọng của nó là khả năng bảo mật mật khẩu rất khó đạt được sự hoàn hảo. Những tác nhân tiêu cực có thể tìm ra mật khẩu của người khác nhờ nhiều tác nhân khác nhau.

### 2.2.2 Ngăn chặn nguyên nhân từ phía các chương trình

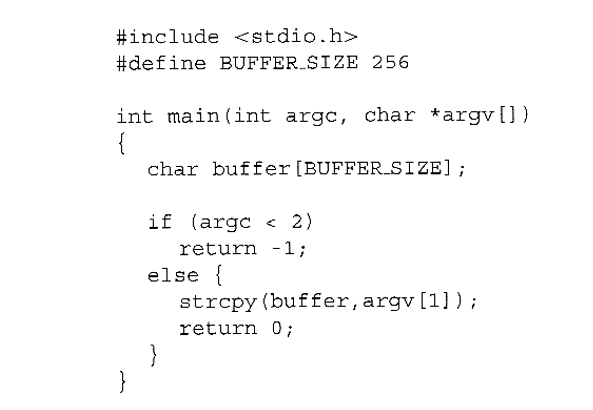
Trong môi trường hoạt động mà một chương trình được tạo lập bởi một người lại được người khác sử dụng rất có thể xảy ra các tình huống sử dụng sai chức năng, từ đó dẫn tới những hậu quả không lường trước được. Bốn trường hợp điển hình gây mất an toàn hệ thống có thể đè xuất là:

* ***Ngựa thành troy:*** khi người sử dụng A- kích hoạt một chương trình (do người sử dụng B viết) dưới danh nghĩa của mình (trong miền bảo vệ được gán tương ứng cho người sử dụng A), chương trình này có thể trở thành “chú ngựa troy” vì khi các đoạn lệnh trong chương trình có thể thao tác với các tài nguyên người sử dụng A có quyền nhưng người sử dụng B lại bị cấm chương trình kiểu này đã lợi dụng hoàn cảnh để gây ra các tác hại đáng tiếc.
* ***Cánh cửa nhỏ (trap-door):*** mối đe doạ đặc biệt nguy hiểm và khó chống đỡ do vô tình hoặc cố ý của các lập trình viên khi xây dựng chương trình. Các lập trình viên có thể để lại một “cánh cửa nhỏ” để đối phó rất phức tạp vì chúng ta cần phải tiến hành phân tích chương trình nguồn để tìm ra chỗ sơ hở.
* ***Logic Bomb***: là mã không được thiết kế để gây ra sự tàn phá mọi lúc, nhưng chỉ khi nhiều tình huống xảy ra, chẳng hạn như khi một ngày hoặc một thời gian cụ thể đạt được hay một sự kiện khác đáng chú ý.

*Ví dụ:* Dead-Man Swicth là đoạn mã được thiết kế để kiểm tra xem một người nào đó (ví dụ như tác giả) đăng nhập mỗi ngày, và nếu họ không đăng nhập lại trong một thời gian dài thì sau đó quả Logic Bomb này sẽ mở một lỗ hổng bảo mật hoặc gây ra các vấn đề khác có hại cho hệ thống.

* ***Stack và Buffer Overflow:*** đây là một phương pháp cổ điển dùng để tấn công, khai thác lỗi trong code của hệ thống, nó có thể làm cho bộ nhớ đệm bị tràn.

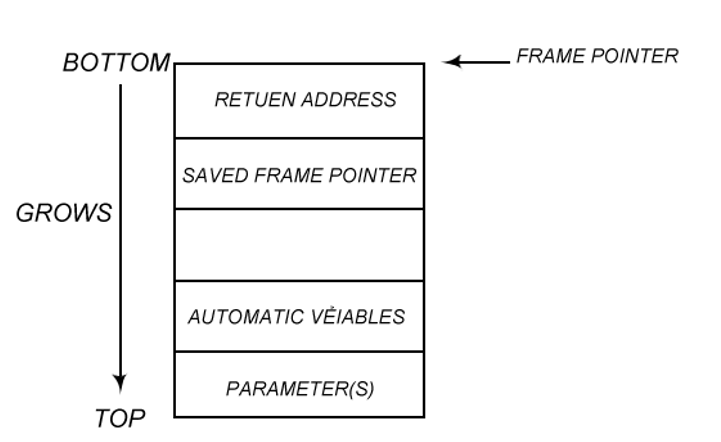
*Một ví dụ* sau cho chúng ta thấy cách thức tấn công của loại đoạn mã này: Lệnh strcpy sẽ làm tràn bộ nhớ đệm, ghi đè lên các khu vực lân cận của bộ nhớ.



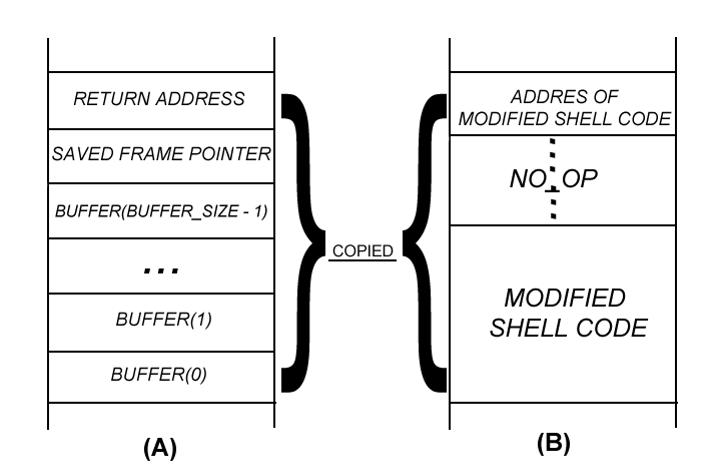
*Hình 2.1. C program with buffer-overflow condition*

Vậy, làm thế nào để gây nên tình trạng tràn bộ nhớ đệm?

Chúng ta sẽ tìm hiểu cấu trúc của ngăn xếp trong bộ nhớ:

*Hình 2.2. Bố trí cho một ngăn xếp điển hình*

Sau đây là cách để khắc phục:



*Hình 2.3. Giả thuyết ngăn xếp cho hình 7, (A) trước (B) sau*

## 2.3 Ngăn chặn nguyên nhân từ phía hệ thống

Hầu hết các tính trình hoạt động trong hệ thống đều có thể tạo ra các tiến trình con. Trong các cơ chế hoạt động này, các tài nguyên hệ thống rất dễ bị sử dụng sai mục đích gây mất an toàn cho hệ thống. Hai mối đe doạ phổ biến theo phương pháp này là:

***+ Các chương trình sâu (worm):*** một chương trình sâu là chương trình lợi dụng cơ chế phát sinh các bản sao trong hệ thống để đánh bại chính hệ thống sau đó chiếm dụng tài nguyên, làm ngừng trễ hoạt động của các tiến trình khác và toàn bộ hệ thống.

***+ Các chương trình virus:*** virus là một chương trình phá hoại khá nguy hiểm đối với các hệ thống thông tin. Khác với các chương trình sâu là những chương trình hoàn chỉnh, virus là những đoạn mã có khả năng lây nhiễm vào các chương trình hệ thống và từ đó tàn phá hệ thống.

## 2.4 Giám sát các nguyên nhân

Nhìn chung, việc đảm bảo an toàn hệ thống là rất phức tạp vì nó liên quan tới yếu tố con người. Hệ điều hành chỉ có thể áp dụng một số biện pháp để giảm bớt thiệt hại như lập nhật kí sự kiện để ghi nhận các tình huống xảy ra trong hệ thống. Ví dụ theo dõi:

- Người sử dụng cố gắng nhập mật khẩu nhiều lần.

- Các tiến trình với định dạng nghi ngờ không được uỷ quyền.

- Các tiến trình lạ trong trong các thư mục hệ thống.

- Các chương trình kéo giài thời gian xử lý một cách đáng ngờ.

- Các tệp tin và các thư mục bị khoá không hợp lý.

- Kích thước các chương trình hệ thống bị thay đổi…

Việc kiểm tra thường kỳ và ghi nhận những thông tin này giúp hệ thống phát hiện kịp thời các nguy cơ, cho phép phân tích, dự đoán và tìm phương pháp đối phó.

# CHƯƠNG 3: VIRUS MÁY TÍNH

## 3.1. Khái niệm về virus

Virus máy tính là một chương trình có khả năng gián tiếp tự kích hoạt, tự lan truyền trong môi trường của hệ thống tính toán và làm thay đổi môi trường hệ thống hoặc cách thực hiện chương trình, điều này có thể dẫn đến đến việc chương trình hoặc dữ liệu bị hỏng, không khôi phục được, thậm chí có thể bị xoá. Như vậy, virus là chương trình thông minh, mang yếu tố thích nghi, lan truyền xa và do đó khả năng phá hoại của nó rất lớn.

Khái niệm “gián tiếp kích hoạt” ở đây có nghĩa là trừ người viết virus và lần đầu tiên đưa hệ thống phải tiếp nạp chương trình virus và thực hiện nó, còn những người sử dụng nó chỉ nạp chương trình của mình. Nếu đó là chương trình nhiễm virus.

Thì virus sẽ chiếm quyền điều khiển trước tiến hành lây lan và sửa đổi sau đó mới trả quyền điều khiển cho chương trình được gọi.

Một số biểu hiện của máy tính bị nhiễm virus:

* Hệ thống hoạt động không ổn định.
* Các chương trình ứng dụng không thể hoạt động được hoặc hoạt động sai chức năng.
* Dữ liệu bị sai lệch.
* Kích thước các file tăng.
* Xuất hiện các file trên đĩa.

**3.2. Phân loại virus**

Dựa vào cơ chế lây lan của virus, người ta có thể phân thành một số loại như sau:

**Boot virus (B - virus):** là những virus chỉ lây lan vào các boot sector hoặc master boot record của các ổ đĩa.

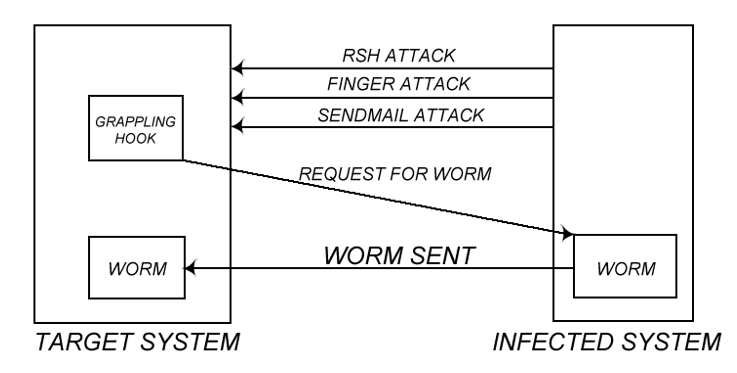
**File (F – virus):** là những virus lây lan vào các file chương trình của người sử dụng (các file com hoặc exe).

**Virus lưỡng tính (B/F virus):** là những virus có thể lây lan vào các file chương trình.

**Maco virus:** là những virus viết bằng các lệnh maco. Chúng thường lây nhiễm vào các file văn bản hay các bảng tính.

**Troyjan virus (Troyjan hors):** là những virus nằm tiềm ẩn trong hệ thống máy tính dưới dạng các chương trình ứng dụng nhưng thực tế khi chương trình này được kích hoạt, các lệnh phá hoại sẽ hoạt động.

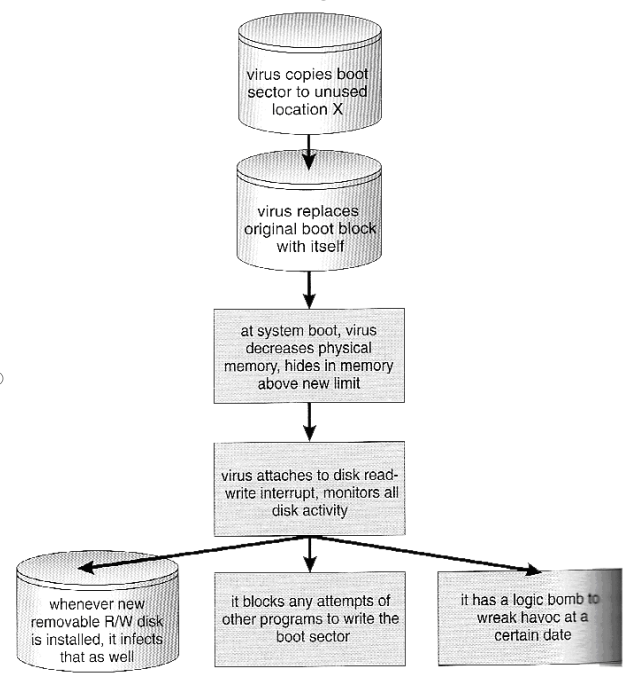
**Worm (sâu):** trên thực tế không được coi là virus vì nó không gây tác hại cho phần mềm hay phần cứng. Sâu di chuyển hệ thống mạng từ máy này sang máy khác. Nhiệm vụ chính của nó là thu nhập các thông tin cá nhân của người sử dụng (mật khẩu tài khoản, các thông tin quan trọng, các tài liệu mật...) để chuyển về một địa chỉ nhất định cho người điều khiển.



*Hình 3.1. The mories internet worm*

## 3.3. Cơ chế hoạt động của virus

Sau đây là sơ đồ mô tả cơ chế hoạt động của một loại virus khởi động điển hình:



*Hình 3.2: A Boot-Sector Computer Virus*

Về cơ chế hoạt động của virus, chúng ta có thể hình dung quá trình như sau:

Khi đọc một đĩa hoặc thi hành một chương trình bị nhiễm virus, nó sẽ tạo ra một bản sao đoạn mã của mình và nằm thường trú trong bộ nhớ máy tính. Khi đọc một đĩa hoặc một chương trình, đoạn mã virus nằm trong bộ nhớ sẽ kiểm tra đĩa/file dó đã tồn tại đoạn mã chưa? Nếu chưa thì tạo một bản sao khác lây nhiễm vào đĩa/file.

*Ví dụ:* Về cơ chế chiếm quyền điều khiển của B- virus: khi máy tính bắt đầu khởi động, mọi thanh ghi cpu sẽ bị xoá. Các thanh ghi đoạn được gắn các giá trị 0FFDh:0, tại đĩa chỉ này, một lệnh JMP FAR chuyển quyền điều khiển đến một đoạn chương trình dịch sẵn trong ROM BIOS. Đoạn chương trình này sẽ thực hiện quá trình POST.

Quá trình POST sẽ lần lượt kiểm tra các thanh ghi, bộ nhớ, khởi tạo các chíp điều khiển DMA, bộ điều khiển ngắt đĩa… nếu quá trình này hoàn thành tốt đẹp thì công việc tiếp theo sẽ là do tìm các card mở rộng (card màn hình, âm thanh…) và trao quyền điều khiển cho chúng tự khởi tạo. Tuy vậy cần phải chú ý rằng toàn bộ các chương trình này nằm trong ROOM (bộ nhớ chỉ đọc) của thiết bị không nên sửa đổi hoặc chèn thêm mã vào được.

Sau khi mọi việc khởi tạo hoàn thành, lúc này một chương trình trong ROOM BIOS đọc boot sector từ đĩa vật lý đầu tiên vào bộ nhớ trong (RAM). Nếu quá trình đọc thành công thì quyền điều khiển sẽ được trao cho đoạn mã nằm trong boot sector bằng một lệnh JMP FAR 0:07C00 mà không cần biết đoạn mã này làm gì, như vậy tại thời điểm này, bất kể trong boot sector chứa đoạn mã nào thì quyền điều khiển vẫn được trao cho nó, đây là một sơ hở đầu tiên của máy tính mắc phải. Điều này cũng dễ hiểu, máy tính không thể kiểm tra được đoạn mã trong boot record vì nó ứng với hệ điều hành cụ thể. Ngay cả khi dùng một hệ điều hành nhưng với version khác nhau thì nội dung đoạn mã cũng khác nhau.

Lợi dụng khe hở đầu tiên này, B- virus sẽ tấn công vào các boot record, nó thay thế boot record chuẩn bằng một đoạn mã của mình. Như vậy, quyền điều khiển sẽ được trao cho virus trước khi boot record chuẩn nhận được quyền điều khiển.

## 4.4. Phòng tránh virus

### 4.4.1 Các chương trình phòng tránh và phát hiện virus

Coi thường sự tồn tại của virus là con đường trực tiếp dẫn đến sự mất an toàn cho hệ thống nhưng mặt khác, giao phó toàn bộ sự an toàn cửa hệ thống cho một loạt chương trình phòng chống virus hiện có thì cũng không ổn, vì một chương trình chống virus làm cho người sử dụng cảm thấy mất an toàn hơn như trong một số trường hợp, các chương trình đó không đưa ra bất kì một sự bảo vệ nào.

Điều mà người sử dụng thường làm nhất là trang bị cho mình một phần mềm chống virus nhưng họ cần phải chú ý rằng: virus có trước phần mềm chống virus có sau. Như vậy, nếu một virus mới xuất hiện thì nó sẽ miễn dịch với tất cảc các phần mềm chống virus hiện có.

Các phần mềm chống virus hiện nay chia thành hai loại: các chương trình phòng ngừa và các chương trình phát hiện.

***Các chương trìmh phòng ngừa:*** Đây là các chương trình thường trú trong trong bộ nhớ của máy tính. Chúng hoạt động dựa vào sự giám sát thường xuyên các ngắt để phát hiện và ngăn chặn các yêu cầu được điều khiển bằng phần mềm như: nạp chương trình, ghi thông tin vào đĩa… ví dụ như một chương trình được nạp và bí mật yêu cầu hệ điều hành cho phép ghi đè lên các boot sector… Khi đó các chương trình phòng ngừa phải báo động ngay và lập tức nhắc nhở người sử dụng phải tự quyết định việc có ngăn chặn hay không.

Về mặt nguyên tắc thì đây là một giải pháp tốt nhưng trên thực tế, không thể cảnh báo một cách chính xác và có hiệu quả cho người sử dụng. Mặt khác, vì thường trú trong bộ nhớ nên nó chiếm mất một phần không gian bộ nhớ, đồng thời thông báo về các ý đồ nạp chương trình; đọc/ghi đĩa là những hoạt động thường xuyên xảy ra trong hệ thống. Dẫn đến người sử dụng không thể phân biệt hết được hoạt động nào là của chương trình, hoạt động nào là của virus.

Tuy nhiên ở một chừng mực nào đó, các chương trình phòng ngừa vẫn nên được sử dụng nhằm mục đích cung cấp “một lá chắn” cơ bản để ngăn chặn một số loại virus đơn giản.

***Các chương trình phát hiện:*** Trong khi các chương trình phòng chống, ngăn chặn, giám sát các sự kiện xảy ra khi chương trình hoạt động thì các chương trình phát hiện lại tiến hành kiểm tra mã chương trình trước khi nó được thực hiện. Người sử dụng khi được nhắc nhở về các mối nguy hiểm khi có mã lạ trong chương trình của mình sẽ phải tự quyết định xem có nên dùng chương trình hay loại bỏ nó. Nếu so sánh một cách trực quan, chúng ta sẽ thấy các chương trình phát hiện có giao diện khá thân thiện, gần gũi với người sử dụng hơn các chương trình phòng chống.

Các chương trình phát hiện được nạp, thực hiện và thoát ra giống như các chương trình ứng dụng thông thường. Chúng không thường trú trong bộ nhớ, không ngăn chặn và ngắt các chương trình của người sử dụng nhưng nhược điểm của nó là phải tuân thủ một số nguyên tắc nhất định trong quá trình kiểm tra và phát hiện virus.

### 4.4.2 Một số biện pháp phòng chống virus

Chúng ta thấy, virus là một sản phẩm trí tuệ do con người tạo ra nên không thể có các biện pháp phòng chống tuyệt đối. Nguyên tắc chung của phòng chống virus là tạo một hàng rào ngăn chặn bất cứ một chương trình lạ nào muốn xâm nhập và phá hoại hệ thống. Việc sử dụng càng nhiều mức ngăn chặn thì càng đảm bảo mức an toàn cho hệ thống người sử dụng hơn. Một số biện pháp phòng tránh có thể đề cập như sau:

***Hạn chế trao đổi dữ liệu:*** Con đường chính để virus phát tán được là do người sử dụng thường dùng các phương tiện trao đổi thông tin giữa các máy tính như đĩa mềm, Flash disk, CD-ROM... hoặc trao đổi thông tin thông qua hệ thống mạng. Nhưng nếu không trao đổi thông tin thì chúng ta sẽ không giải quyết được yêu cầu công việc. Do đó, chúng ta cần phải xác định rõ mục đích của việc trao đổi thông tin (có thực sự cần thiết hay không?). Nếu thật sự cần thiết thì phải kiểm tra độ an toàn của thông tin trước khi đưa vào sử dụng.

***Hạn chế sử dụng các phần mềm phá khóa hoặc không rõ nguồn gốc:*** Những phần mềm kiểu là môi trường mà các lập trình viên tạo virus gửi “sản phẩm” của mình vào để phát tán. Khi chúng thực hiện các phần mềm kiểu này, chúng ta sẽ được đảm bảo tính an toàn từ phía các công ty sản xuất phần mềm chính thống.

***Sử dụng thường xuyên các phần mềm phòng ngừa vào phát hiện virus:*** Phát hiện virus càng sớm bao nhiêu thì các thiệt hại do virus gây ra càng hạn chế. Khi phát hiện dược các file bị nhiễm virus, chúng ta cần xóa bỏ hoặc sử dụng chương trình diệt virus để xóa các đoạn mã của chúng.

***Thay đổi các thuộc tính của file chương trình:*** Virus khi lây nhiễm vào các file thường phải biết được cấu trúc của chúng. Với từng kiểu file khác nhau, chúng sẽ có các cơ chế lây nhiễm khác nhau. Nhưng trong một số hệ điều hành, chúng ta có thể đổi các file \*.COM thành các file \*.EXE mà không gây ra bất cứ hiệu ứng nào. Như vậy, chúng ta có thể đổi phần mở rộng từ COM thành EXE và ngược lại để đánh lừa các chương trình virus.

***Cài đặt lại các chương trình ứng dụng:*** Nếu cảm thấy có nguy hiểm hoặc theo định kỳ, chúng ta nên cài đặt lại các chương trình ứng dụng từ đĩa cài đặt gốc. Nếu có chương trình ứng dụng nào bị nhiễm virus trước khi cài đặt lại thì sau đó chúng sẽ được loại bỏ virus.

***Cài đặt lại hệ thống:*** Là một mức cao hơn của cài đặt lại các chương trình ứng dụng vì lúc đó cả hệ điều hành cũng được thiết lập lại.

***Tạo lại khuôn dạng cho đĩa từ:*** Một số loại virus cất dấu đoạn mã của mình trong boot sector của đĩa từ và các sector này thường được đánh dấu là “hỏng” hoặc “không được sử dụng” đối với các hệ điều hành. Do đó, cần phải tạo lại khuôn dạng cho đĩa từ để làm sạch vùng này.

***Thường xuyên sao lưu dự phòng dữ liệu:*** Tốt hơn cả là chúng ta cần phải thường xuyên sao lưu lại dữ liệu để đảm bảo tính an toàn và có thể phục hồi lại khi hệ thống gặp sự cố. Các hệ điều hành đều cung cấp công cụ này với tên chương trình backup.