



# HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



## BÀI GIẢNG MÔN HỌC AN TOÀN HỆ ĐIỀU HÀNH CHƯƠNG 3 – AN TOÀN CÁC DỊCH VỤ CƠ BẢN CỦA HĐH

Giảng viên:

PGS.TS. Hoàng Xuân Dậu

E-mail:

dauhx@ptit.edu.vn

Khoa:

An toàn thông tin

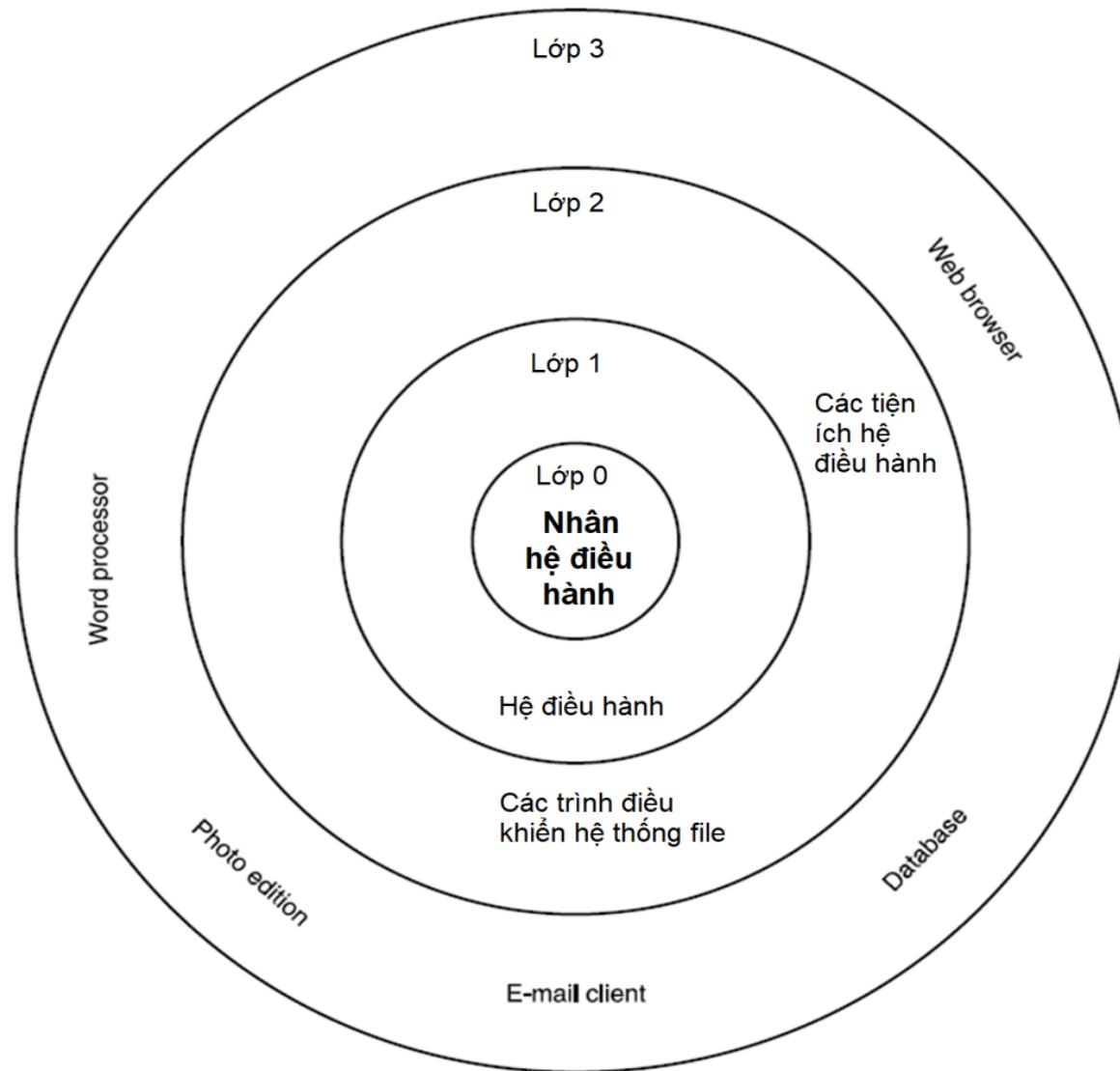
## NỘI DUNG CHƯƠNG 3

1. Quản lý tiến trình
2. Quản lý bộ nhớ
3. Nền tảng tính toán tin cậy
4. Bảo vệ hệ thống file
5. Phân tích các vấn đề an toàn của các hệ điều hành Windows và Unix/Linux

### 3.1 Quản lý tiến trình

- ❖ Kiến trúc tập lệnh x86 hỗ trợ việc quản lý thực thi các chương trình bằng cách triển khai cơ chế bảo vệ theo lớp đặc quyền dựa trên kiểm soát truy cập đến:
  - Các câu lệnh của chương trình;
  - Các dữ liệu của chương trình.
- ❖ Cơ chế bảo vệ theo lớp đặc quyền được thực hiện nhằm hạn chế các thao tác mà chương trình người dùng có thể thực hiện dựa trên sự phối hợp giữa:
  - Phần cứng (CPU) và
  - Hệ điều hành.

## Quản lý tiến trình - Bảo vệ không gian thực thi theo lớp



## Quản lý tiến trình

- ❖ Các tài nguyên được bảo vệ đối với các thao tác của tiến trình người dùng gồm:
  - Bộ nhớ
  - Các cổng vào/ra
  - Khả năng chạy một số câu lệnh đặc biệt.
- ❖ Tại bất kỳ thời điểm nào, bộ xử lý x86 hoạt động tại một lớp đặc quyền nhất định mà lớp này quyết định các thao tác mà đoạn mã có thể thực thi được hay không.

## Quản lý tiến trình – Các câu lệnh đặc quyền

- ❖ Có khoảng 15 câu lệnh được bảo vệ dành riêng cho lớp có đặc quyền cao nhất (lớp 0);
  - Là các lệnh dành riêng cho nhân hay phần hệ thống của HĐH;
  - Các tiến trình chạy ở các lớp đặc quyền thấp hơn không thể thực hiện.
  - Nếu các tiến trình người dùng có thể thực hiện các câu lệnh này thì có thể phá vỡ cơ chế bảo vệ hay gây ra sự hỗn loạn.
  - Bất kỳ nỗ lực nào nhằm chạy các câu lệnh này bên ngoài lớp 0 sẽ gây ra lỗi như khi chương trình cố truy cập ô nhớ không hợp lệ.

## Quản lý tiến trình – Các câu lệnh đặc quyền

- ❖ Bộ xử lý x86 theo dõi mức đặc quyền (lớp bảo vệ) thông qua các trường:
  - RPL (Requested Privilege Level) trên thanh ghi đoạn dữ liệu:
    - Giá trị của trường này không thể được gán trực tiếp bởi các câu lệnh nạp dữ liệu mà
    - Chỉ bởi các câu lệnh thay đổi luồng thực hiện như câu lệnh *call*.
  - CPL (Current Privilege Level) trên thanh ghi đoạn lệnh:
    - Giá trị này được duy trì bởi chính CPU và nó luôn bằng với mức bảo vệ hiện thời của CPU;
    - Nói cách khác, giá trị CPL cho biết mức độ bảo vệ của đoạn mã được thực hiện.

## Thẻ chọn đoạn dữ liệu và lệnh

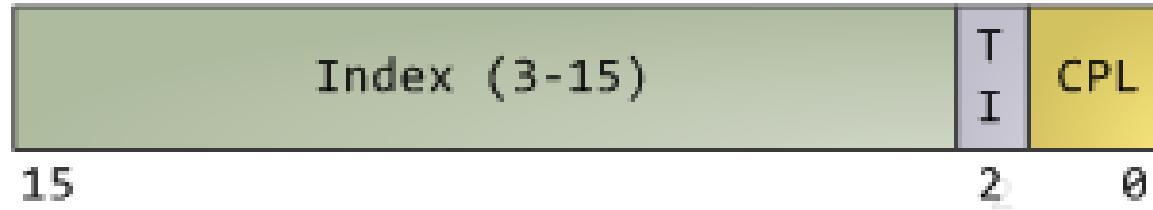
Data segment selector

16 bits



Code segment selector

16 bits



## Thẻ chọn đoạn dữ liệu và lệnh

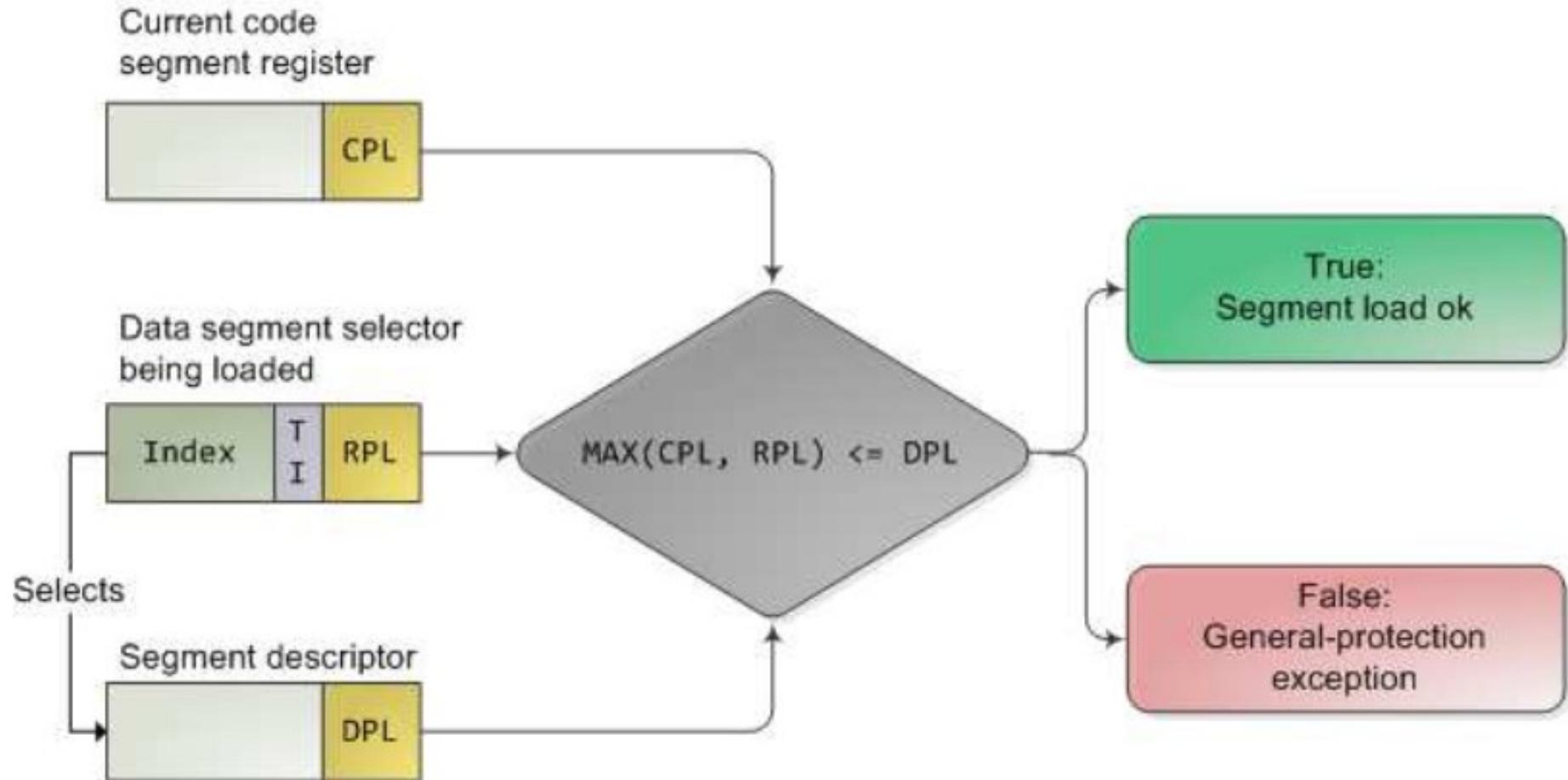
### ❖ Các thẻ:

- Data segment selector: thẻ chọn đoạn dữ liệu
- Code segment selector: thẻ chọn đoạn lệnh.

### ❖ Các mức độ đặc quyền của bộ xử lý không liên quan tới vai trò người dùng trong hệ điều hành, dù là quản trị hay người dùng thông thường.

- Toàn bộ các đoạn mã (lệnh) của người dùng được chạy ở lớp 3 (mức ít đặc quyền nhất) và
- Toàn bộ đoạn mã của phần nhân hoạt động ở lớp 0 (mức đặc quyền cao nhất).

## Bảo vệ đoạn nhớ trong kiến trúc x86



## Bảo vệ đoạn nhớ trong kiến trúc x86

- ❖ Do mức đặc quyền (level) cao có nghĩa là có ít đặc quyền, hàm MAX() sẽ chọn thành phần có đặc quyền thấp hơn trong CPL và RPL;
- ❖ Sau đó so sánh với DPL (Descriptor Privilege Level):
  - Nếu DPL lớn hơn hoặc bằng --> phê duyệt yêu cầu truy cập
  - Nếu DPL nhỏ hơn --> từ chối yêu cầu truy cập.

## Bảo vệ đoạn nhớ trong kiến trúc x86

- ❖ Ý tưởng của việc sử dụng RPL là cho phép mã nhân HĐH nạp 1 đoạn nhớ sử dụng đặc quyền thấp hơn;
  - VD, sử dụng RPL=3 để đảm bảo rằng một thao tác cho trước sử dụng các đoạn truy cập vào chế độ người dùng;

## Quản lý chương trình người dùng

### ❖ Chương trình chạy trong chế độ người dùng:

- Bị hạn chế truy cập tới bộ nhớ, các cổng vào/ra;
- Hầu như không thể tác động tới các tài nguyên bên ngoài trừ phi thực hiện lời gọi đến phần nhân.

### ❖ Chương trình người dùng không thể trực tiếp:

- mở file
- gửi các gói dữ liệu hay
- cấp phát bộ nhớ.

## Quản lý chương trình người dùng

- ❖ Chương trình người dùng bị đóng trong “hộp kín” (sandbox) do đoạn mã của phần nhân đặt ra;
- ❖ Về mặt thiết kế không thể xảy ra chuyện rò rỉ bộ nhớ của chương trình hay truy cập trực tiếp đến các cấu trúc dữ liệu quan trọng của hệ thống;
- ❖ Khi chương trình người dùng kết thúc thì sandbox này cũng bị xóa bỏ.

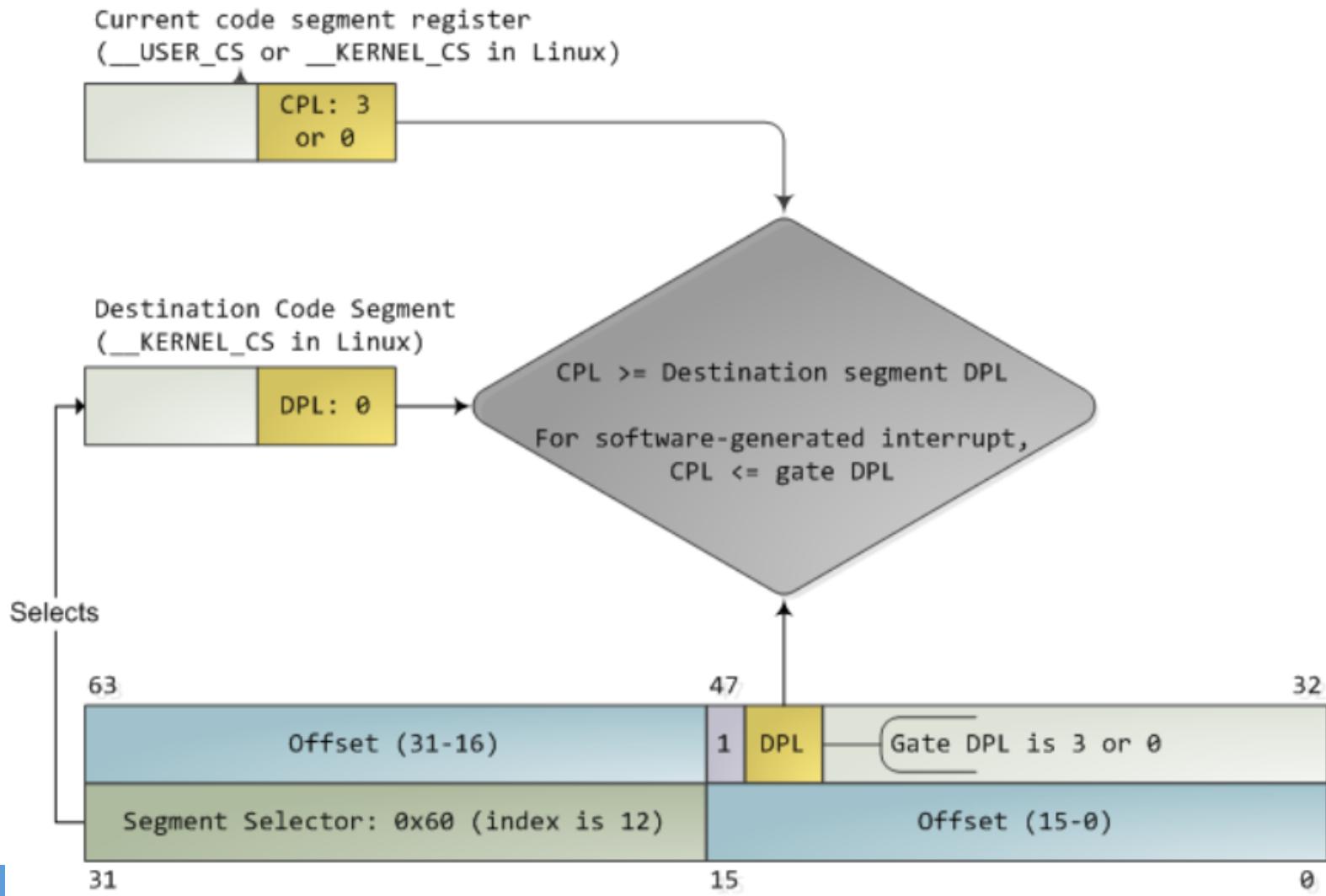
## Kiểm soát thực hiện lệnh có đặc quyền cao

- ❖ Lệnh jmp trong bộ xử lý x86 được sử dụng để chuyển luồng thực hiện từ đoạn mã này sang đoạn mã khác;
- ❖ Tuy nhiên, các đoạn mã với đặc quyền thấp bị hạn chế sử dụng lệnh jmp để chuyển luồng thực hiện tới các đoạn mã của nhân HĐH (có đặc quyền cao);
- ❖ Việc kiểm soát được thực hiện thông qua thẻ mô tả cổng (gate descriptor):
  - Cổng gọi hàm (call-gate)
  - Cổng ngắt (interrupt-gate)
  - Cổng bẫy (trap-gate) và
  - Cổng nhiệm vụ (task-gate).

## Các thẻ mô tả cổng

- ❖ Cổng gọi hàm được sử dụng với các lời gọi hàm thông thường như *call* hay *jmp*;
- ❖ Cổng nhiệm vụ thường được sử dụng khi bị lỗi kép do nhân hay phần cứng;
- ❖ Cổng ngắt và cổng bẫy được dùng để xử lý các ngắt phần cứng, như bộ định thời, ổ cứng và các ngoại lệ, như lỗi trang nhớ, chia 0;
  - Các cổng này được lưu trữ trong bảng mô tả ngắt IDT (Interrupt Descriptor Table) và
  - Cung cấp thông tin về mức đặc quyền của cổng thông qua trường DPL (Descriptor Privilege Level).

## Kiểm tra đặc quyền đoạn lệnh qua cổng ngắt



## Kiểm tra đặc quyền đoạn lệnh qua cỗng ngắn

- ❖ Một ngắt không thể chuyển điều khiển từ lớp có đặc quyền cao sang lớp có đặc quyền thấp;
  - Đặc quyền khi ngắt thường được giữ nguyên hoặc hạ xuống.
  - Trong cả 2 trường hợp CPL = DPL của đoạn mã đích.
    - Nếu CPL thay đổi dẫn đến thay đổi ngăn xếp.
- ❖ Nếu việc chuyển đổi đoạn mã được thực hiện thông qua lệnh INT thì cần đảm bảo thêm:
  - Mức đặc quyền DPL phải giống hoặc thấp hơn của CPL.
  - Việc này ngăn cản chương trình người dùng kích hoạt ngẫu nhiên các đoạn mã ngắt.
  - Nếu việc kiểm tra không thành công thì lỗi bảo vệ sẽ được sinh ra.

## Kiểm tra đặc quyền vào/ra

- ❖ Với các câu lệnh sử dụng các thiết bị vào/ra cũng được kiểm tra theo cách tương tự:
  - Mỗi câu lệnh IN/OUT được liên kết với cờ trạng thái mô tả mức đặc quyền IOPL (I/O privilege level) cho biết lớp bảo vệ tương ứng.
  - Câu lệnh vào/ra sẽ chỉ được thực hiện khi mức đặc quyền của đoạn mã nhỏ hơn hoặc bằng với mức đặc quyền của lệnh vào/ra.

### 3.2 Quản lý bộ nhớ

- ❖ Bảo vệ bộ nhớ sử dụng đặc quyền
- ❖ Ngăn chặn thực thi dữ liệu

## Bảo vệ bộ nhớ sử dụng đặc quyền

- ❖ Hệ điều hành cần phải bảo vệ bộ nhớ của các tiến trình:
  - Không gian nhớ của các tiến trình cần được cách ly với nhau sao cho mỗi tiến trình chỉ có thể truy cập được không gian nhớ của riêng mình;
  - Các không gian nhớ của các tiến trình phải được cách biệt với nhau và với phần nhân của hệ điều hành.

==> Ngăn chặn lỗi hay các tiến trình xấu gây ảnh hưởng tới các tiến trình khác và/hoặc bản thân hệ điều hành.
- ❖ Việc một tiến trình cố truy cập bộ nhớ không phải của mình sẽ gây ra lỗi phần cứng, như là lỗi xâm phạm ô nhớ được bảo vệ, và làm cho tiến trình vi phạm phải chấm dứt hoạt động.

## Bảo vệ bộ nhớ sử dụng đặc quyền

- ❖ Hệ điều hành sử dụng bộ nhớ ảo với không gian địa chỉ thống nhất để quản lý các dạng bộ nhớ vật lý.
  - HĐH cấp phát bộ nhớ ảo thông qua địa chỉ bộ nhớ lô-gíc thay vì địa chỉ bộ nhớ vật lý mà bộ xử lý kết nối tới thông qua buýt hệ thống;
  - Như vậy, cần phải thực hiện việc chuyển đổi (ánh xạ) từ địa chỉ lô-gíc của chương trình thành địa chỉ vật lý thực sự trước khi thao tác đọc/ghi bộ nhớ được diễn ra.

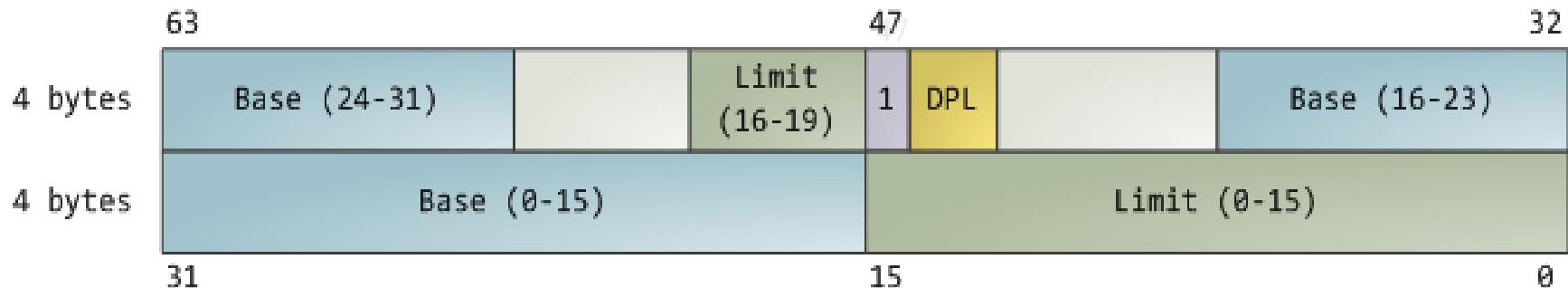
## Quản lý bộ nhớ phân đoạn

- ❖ Kiến trúc x86 sử dụng phương pháp phân đoạn để quản lý không gian nhớ chương trình nhờ vào việc tách biệt các chức năng của không gian nhớ (đoạn mã, đoạn dữ liệu, đoạn ngắn xếp);
- ❖ Các đoạn chức năng được truy cập dựa vào các thanh ghi đoạn như thanh ghi đoạn lệnh/mã CS hay đoạn dữ liệu DS. Các thông tin quản lý của các thanh ghi đoạn được lưu trong thẻ mô tả thanh ghi đoạn (segment descriptor).

## Thẻ mô tả đoạn nhớ

### ❖ Các thông tin trong thẻ mô tả đoạn:

- Base cho biết địa chỉ bắt đầu của đoạn;
- Limit cho biết độ lớn của đoạn;
- DPL là mức đặc quyền của đoạn (0, 1, 2, 3).
  - 0 -> đặc quyền cao nhất
  - 3 -> đặc quyền thấp nhất.



## Thẻ mô tả đoạn nhớ

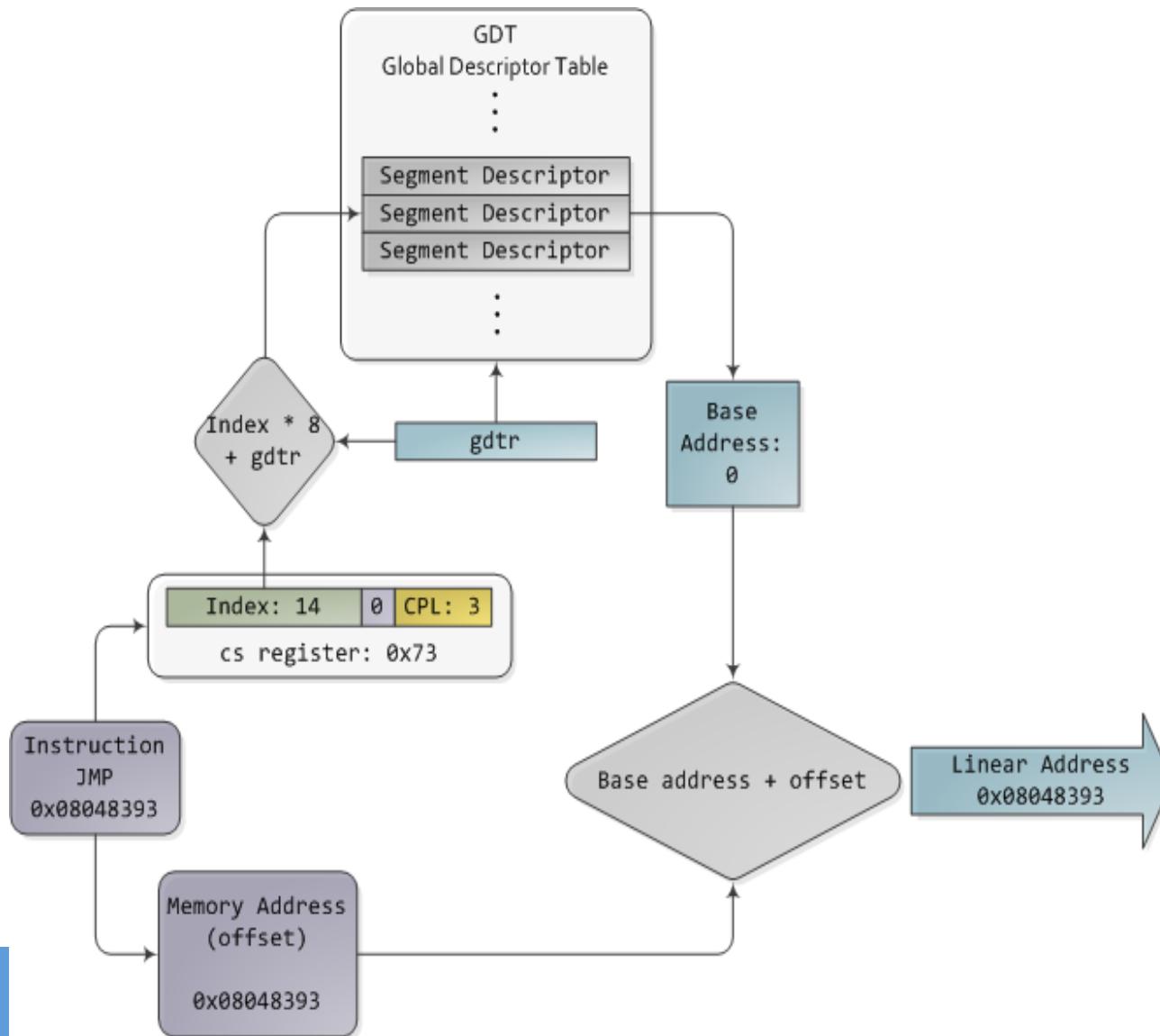
- ❖ Các thẻ mô tả đoạn được lưu trong hai bảng:
  - Bảng mô tả toàn cục GDT (Global Descriptor Table) và
  - Bảng mô tả cục bộ LDT (Local Descriptor Table).
- ❖ Thông thường cần 1 bảng GDT để truy cập trực tiếp vào không gian nhớ phẳng;
  - Vị trí đầu tiên của bảng GDT được lưu trong thanh ghi *gdtr*.

## Thẻ chọn đoạn nhớ

- ❖ Để chọn đoạn, chương trình cần sử dụng thẻ chọn đoạn (segment selector):
  - Bít TI giúp phân biệt GDT ( $TI=0$ ) và LDT ( $TI=1$ );
  - RPL mô tả mức đặc quyền cần thiết khi truy cập vào đoạn tương ứng.



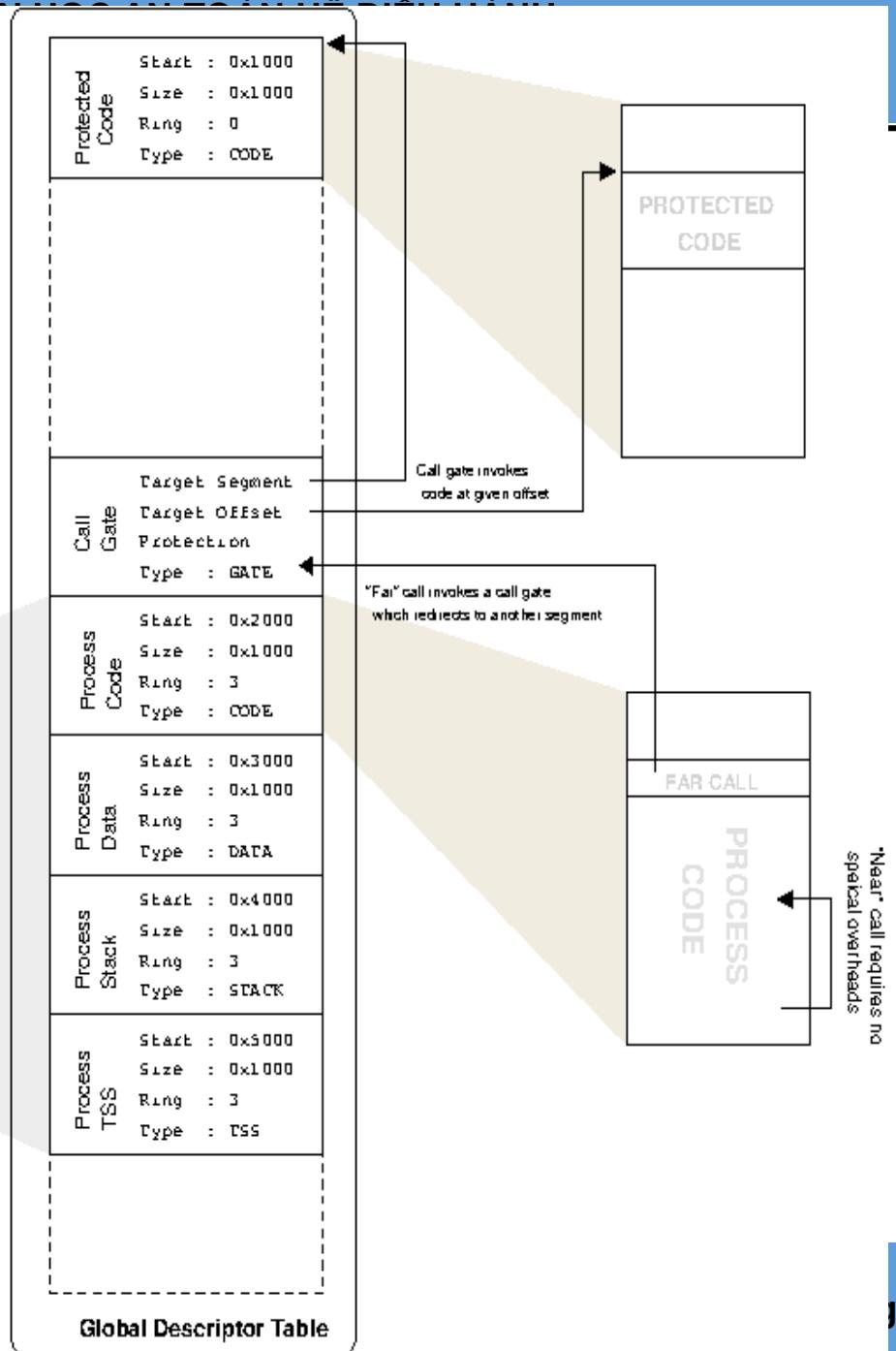
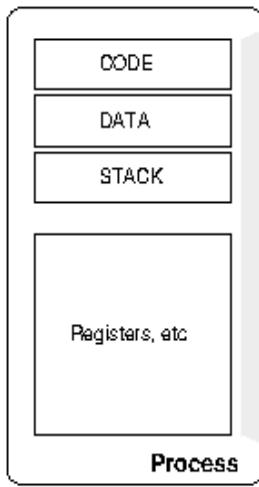
## Truy cập bộ nhớ qua GDT



## Truy cập bộ nhớ qua GDT

- ❖ Nếu tiến trình người dùng có khả năng sửa đổi thông tin trong bảng GDT thì tự bản thân tiến trình này có khả năng thay đổi đặc quyền của mình;
  - Như vậy, hệ điều hành cần phải lưu bảng GDT trong phần bộ nhớ dành cho nhân và không thể truy cập trực tiếp từ tiến trình người dùng.
  - Nói cách khác, các thông tin này cần được lưu vào không gian nhớ được bảo vệ.

# Cấu trúc không gian nhớ của tiến trình và thông tin quản lý bộ nhớ



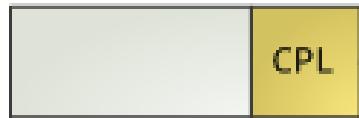
## Kiểm tra đặc quyền truy cập bộ nhớ

❖ Khi đoạn dữ liệu được nạp việc kiểm tra được diễn ra theo các bước như sau:

- Mức độ bảo vệ của đoạn bộ nhớ được yêu cầu truy cập DPL được so sánh với CPL và RPL.
- Nếu  $DPL \geq \text{MAX}(CPL, RPL)$  thì việc truy cập là hợp lệ.
- Nếu không kia đó sẽ phát sinh lỗi truy cập bộ nhớ được bảo vệ.
- Việc sử dụng RPL cho phép đoạn mã nhân nạp đoạn bộ nhớ dùng mức đặc quyền thấp hơn.
- Riêng với đoạn ngăn xếp, cả ba mức CPL, RPL và DPL phải giống nhau hoàn toàn.

## Kiểm tra đặc quyền truy cập bộ nhớ

Thanh ghi đoạn mã  
lệnh hiện thời

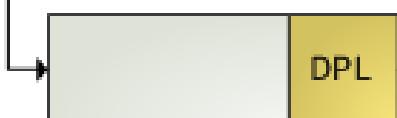


Thẻ chọn đoạn dữ  
liệu được nạp



Chọn

Thẻ mô tả đoạn



$$\text{MAX(CPL, RPL)} \leq \text{DPL}$$

True:  
Nạp đoạn

False:  
Lỗi truy nhập bộ nhớ

## Bảo vệ bộ nhớ trong các HĐH hiện đại

- ❖ Các hệ điều hành hiện đại sử dụng không gian địa chỉ phẳng còn tiến trình người dùng truy cập toàn bộ không gian địa chỉ tuyến tính (ảo);
- ❖ Cơ chế bảo vệ bộ nhớ được thực hiện tại bộ phận quản lý trang khi thực thi việc chuyển đổi địa chỉ phẳng thành địa chỉ vật lý:
  - Mỗi trang nhớ được mô tả bởi khoản mục bảng trang PTE (page table entry) chứa hai trường phục vụ cho việc bảo vệ là:
    - cờ giám sát (supervisor) và
    - cờ đọc/ghi;

## Bảo vệ bộ nhớ trong các HĐH hiện đại

- Cờ giám sát cho phép bảo vệ không gian nhớ của nhân.
  - Khi cờ này bật lên, trang nhớ tương ứng sẽ không truy cập được từ mức người dùng (mức đặc quyền 3).
- Cờ đọc/ghi không thực sự quan trọng với việc thực thi đặc quyền.
  - Khi chương trình được nạp vào bộ nhớ, trang chứa mã được bật cờ chỉ đọc.
    - Như vậy các thao tác ghi lên trang nhớ này sẽ bị báo lỗi và từ chối thực hiện.

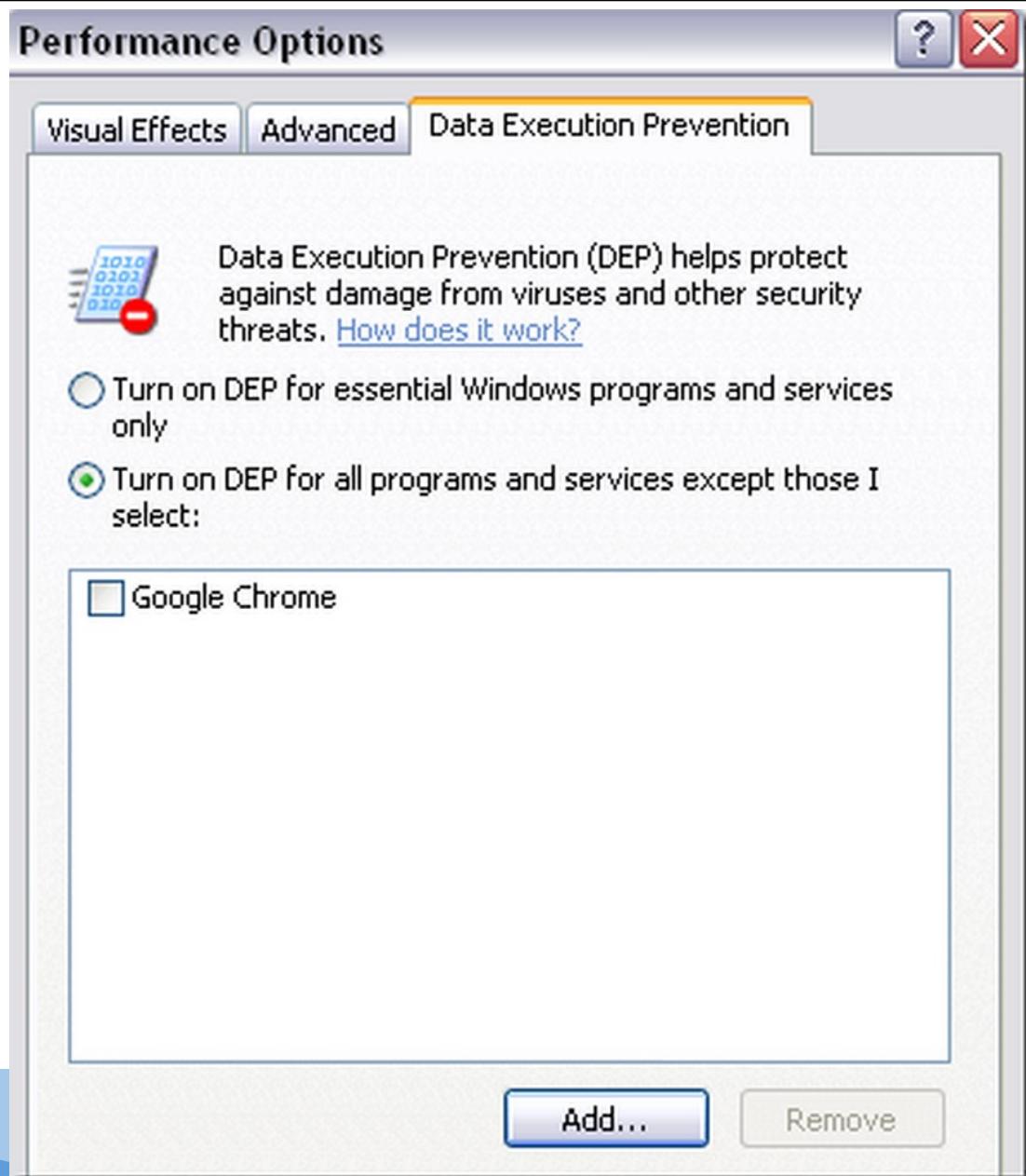
## Ngăn chặn thực thi dữ liệu

- ❖ Ngăn chặn thực thi dữ liệu - DEP (Data Execution Prevention) là kỹ thuật bảo vệ bộ nhớ ở mức hệ thống được tích hợp vào hệ điều hành.
  - Về cơ bản, DEP cho phép hệ thống đánh dấu một hay nhiều trang bộ nhớ là không thực thi được.
- ❖ Tấn công khai thác tràn bộ đệm là một trong những kỹ thuật khai thác lỗ hổng trong việc tổ chức và quản lý không gian nhớ của chương trình;
  - DEP có thể giúp ngăn chặn hiệu quả dạng tấn công này.

# BÀI GIẢNG MÔN HỌC AN TOÀN HỆ ĐIỀU HÀNH

## CHƯƠNG 3 – AN TOÀN CÁC DỊCH VỤ CƠ BẢN CỦA HĐH

Bật cơ chế  
DEP trong  
HĐH  
MS Windows



## Ngăn chặn thực thi dữ liệu

- ❖ Kiến trúc x86 hỗ trợ cơ chế này thông qua bit cấm thực thi (execute-disable hay no-execute) đặt tại trang nhớ.
- ❖ Khi chương trình cố thực thi đoạn mã từ trang nhớ có bít cấm thực thi được bật sẽ gây ra lỗi trang.
- ❖ Ở mức hệ điều hành nếu lỗi này không được xử lý thì chương trình đang hoạt động sẽ bị chấm dứt việc thực thi.
- ❖ Cơ chế cấm thực thi có thể được kích hoạt bằng bít cấm thực thi tại bất kỳ mức nào trong cấu trúc trang nhớ.

## Ngăn chặn thực thi dữ liệu

- ❖ DEP có thể được thực hiện ở mức hệ điều hành như trong Windows XP nhằm ngăn chặn mã độc lợi dụng cơ chế xử lý lỗi (exception) trong Windows.
- ❖ DEP từ phần mềm có thể hoạt động trên bất kỳ bộ xử lý nào, bất kể bộ xử lý đó có hỗ trợ DEP từ phần cứng hay không.
  - Tuy nhiên, DEP bằng phần mềm chỉ giới hạn bảo vệ các file nhị phân của hệ thống.

### 3.3 Nền tảng tính toán tin cậy

- ❖ Khái quát về tính toán tin cậy
- ❖ Mô đun nền tảng tính toán tin cậy
- ❖ Các tác động của tính toán tin cậy
- ❖ Các chỉ trích/vấn đề đối với tính toán tin cậy

## Đặt vấn đề

❖ Máy tính ngày nay sử dụng kiến trúc mở cho phép người dùng toàn quyền lựa chọn phần mềm cũng như khả năng đọc, xóa hay sửa dữ liệu lưu trữ trên máy tính của mình. Điều này dẫn đến các vấn đề tiêu biểu sau:

- Không an toàn cho người dùng vì kiến trúc mở có nguy cơ bị lây nhiễm vi-rút, sâu, hay người dung vô tình cài đặt phần mềm độc hại;
- Không an toàn cho mạng mà máy tính kết nối vào do máy tính này có thể chứa phần mềm độc hại có thể đe dọa máy tính khác trong mạng;
- Không an toàn cho người sản xuất phần mềm và nội dung do kiến trúc mở cho phép các chương trình, file âm nhạc ... có thể bị sao chép không giới hạn và không bị giảm chất lượng.

## Khái niệm tính toán tin cậy

- ❖ Hệ thống tin cậy (Trusted System) là một hệ thống dựa vào một mức độ cụ thể để thực thi một chính sách bảo mật cụ thể;
  - Điều này tương đương với việc nói rằng một hệ thống tin cậy là một hệ thống mà khi có lỗi sẽ phá vỡ chính sách bảo mật (nếu tồn tại một chính sách mà hệ thống tin cậy được tin cậy để thực thi).
- ❖ Tính toán tin cậy (Trusted Computing) là khái niệm được phát triển và thúc đẩy bởi Trusted Computing Group:
  - Tính toán tin cậy được đưa ra từ khái niệm Các hệ thống tin cậy.
  - Với tính toán tin cậy, máy tính sẽ hoạt động nhất quán theo những cách mong đợi và những hành vi đó sẽ được thực thi bởi phần cứng và phần mềm của máy tính.

## Khái niệm tính toán tin cậy

- ❖ Khái niệm tính toán tin cậy đã được giới thiệu khá lâu trong lĩnh vực an toàn máy tính và có ảnh hưởng tới việc thiết kế các thế hệ máy tính phổ thông như PC, thiết bị di động.
- ❖ Tính toán tin cậy đòi hỏi thiết kế lại kiến trúc hệ thống sao cho các thành phần riêng lẻ được định nghĩa một cách tường minh các đặc tính của mình.
  - Điều này cho phép người thiết kế có thể xác định chính xác hành vi của hệ thống.

## Khái niệm tính toán tin cậy

- ❖ Tính toán tin cậy bao gồm sáu khái niệm công nghệ chính, trong đó tất cả đều được yêu cầu cho một hệ thống tin cậy hoàn toàn:
  - Khóa chứng thực (Endorsement key)
  - Đầu vào và đầu ra an toàn (Secure input and output)
  - Màn che bộ nhớ / thực thi được bảo vệ (Memory curtaining / protected execution)
  - Lưu trữ có niêm phong (Sealed storage)
  - Chứng thực từ xa (Remote attestation)
  - Bên thứ ba tin cậy (TTP - Trusted Third Party)

## Khái niệm tính toán tin cậy

- ❖ Tính toán tin cậy mô tả các sửa đổi cần thiết về phần cứng và phần mềm để cung cấp nền tảng ổn định để hệ thống máy tính có thể hoạt động trên đó.
- ❖ Hệ thống này có đặc tính:
  - Độ đảm bảo cao về trạng thái (cấu hình, tình trạng hoạt động của phần mềm ...) của hệ thống máy tính cục bộ.
    - Do vậy có thể xác định khả năng chấp nhận các tác động không mong muốn.
  - Mức độ đảm bảo tương đối về trạng thái của hệ thống ở xa.
    - Điều này thể hiện độ tin cậy của việc tương tác trong hệ thống phân tán.

## Các thách thức triển khai tính toán tin cậy

- ❖ Các máy tính và phần mềm được cung cấp từ nhiều nhà sản xuất và phân phối khác nhau dẫn đến khó khăn trong việc xác định mức độ ổn định và tin cậy của hệ thống máy tính cũng như là phần mềm.
- ❖ Hầu hết hệ điều hành được thiết kế dựa trên ý tưởng có 1 người quản trị hệ thống chuyên nghiệp trong khi người dùng cuối thường thiếu kỹ năng và hiểu biết.
- ❖ Nhiều rủi ro xuất hiện do sự bất cẩn của người dùng cuối.

## Các thách thức triển khai tính toán tin cậy

- ❖ Các vấn đề tương tự cũng gặp phải với các hệ thống mạng:
  - Người dùng không thể tiếp xúc trực tiếp với các hệ thống máy chủ cung cấp dịch vụ;
  - Như vậy, người dùng phải chấp nhận giả định hệ thống máy chủ là tin cậy và được quản trị bởi đội ngũ phù hợp.

## Tính toán tin cậy: Nền tảng di động

- ❖ Nền tảng di động là ví dụ tiêu biểu với tính toán tin cậy vì có yêu cầu tin cậy chặt chẽ hơn máy tính cá nhân PC:
  - Các thiết bị vô tuyến cần tuân thủ chặt chẽ các qui định quan trọng như không cho phép phần mềm thay đổi tùy ý các tham số hoạt động.
  - Thời gian hoạt động cũng có thể được tính tiền và phải được gán một cách chính xác tới thuê bao.
- ❖ Nhờ các yêu cầu khắt khe hơn và thu hút được sự quan tâm hơn từ nhà sản xuất cũng như người dùng, nền tảng tính toán tin cậy trên di động có nhiều tiến bộ hơn so với PC truyền thống.

## Mô đun nền tảng tính toán tin cậy

- ❖ Giới thiệu TPM
- ❖ Ứng dụng của TPM
- ❖ Các dịch vụ an toàn
- ❖ Khởi tạo TPM

## Giới thiệu TPM

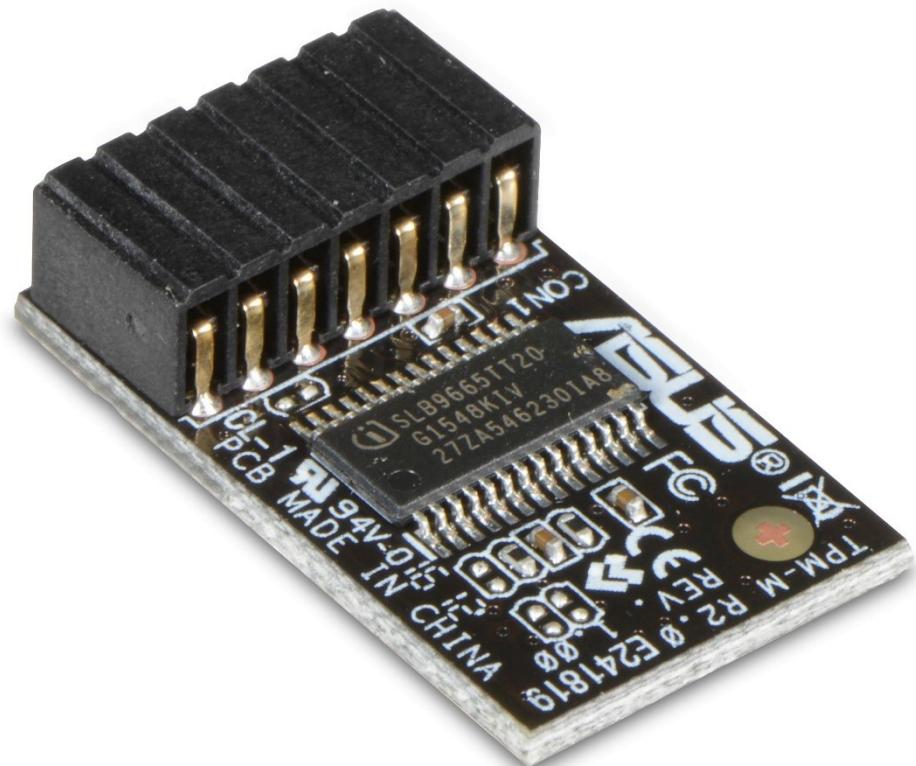
- ❖ Mô-đun nền tảng tin cậy (TPM - Trusted Platform Module) là tiêu chuẩn quốc tế dành cho bộ xử lý mật mã an toàn, một bộ vi điều khiển chuyên dụng được thiết kế để bảo mật phần cứng thông qua các khóa mật mã tích hợp;
- ❖ TPM được hình thành bởi Trusted Computing Group (TCG) và được tiêu chuẩn hóa bởi Tổ chức Tiêu chuẩn hóa Quốc tế (ISO) và Ủy ban Kỹ thuật Điện Quốc tế (IEC) vào năm 2009 với tên gọi ISO / IEC 11889.



## Một số mô đun TPM tiêu biểu



VK VUONGKHANG.COM



## BIOS máy tính hỗ trợ TPM

Aptio Setup Utility - Copyright (C) 2019 American Megatrends, Inc.

Advanced

TPM20 Device Found	▲ Enables or Disables BIOS support for security device. O.S. will not show Security Device. TCG EFI protocol and INT1A interface will not be available.
Security TPM Device Support	[Enable]
Active PCR banks	SHA256
Available PCR banks	SHA-1,SHA256
SHA-1 PCR Bank	[Disabled]
SHA256 PCR Bank	[Enabled]
Pending operation	[None]
Platform Hierarchy	[Enabled]
Storage Hierarchy	[Enabled]
Endorsement	[Enabled]
Hierarchy	
TPM2.0 UEFI Spec Version	[TCG_2]

▼: Select Screen  
▲: Select Item  
Enter: Select  
+/-: Change Opt.  
F1 : General Help  
F7 : Discard Changes  
F9 : Optimized Defaults  
▼ F10: Save & Exit  
ESC: Exit

## Giới thiệu TPM

- ❖ Mô-đun TPM cung cấp cơ sở để đảm bảo an toàn cho các chương trình máy tính trong quá trình hoạt động cũng như tương tác với nhau thông qua việc sử dụng các cơ chế định danh và mã hóa.
- ❖ Các chức năng của TPM được xây dựng bằng phần mềm, song các chức năng an ninh cần được bảo vệ chặt chẽ được thực hiện thông qua thiết bị phần cứng.

## Giới thiệu TPM

- ❖ Về cơ bản, tính toán tin cậy cần thiết bị lưu trữ được bảo vệ mà thiết bị này chỉ có thể truy cập qua các giao tiếp đặc biệt và không sử dụng các thức đọc ghi thông thường như với thanh ghi hay bộ nhớ.
- ❖ Cách này tương tự như việc truy cập thông tin trên thẻ tín dụng. Việc này cho phép TPM mã hóa các thông tin bằng mã khóa chỉ tồn tại bên trong TPM và giúp bảo vệ dữ liệu được lưu trữ.

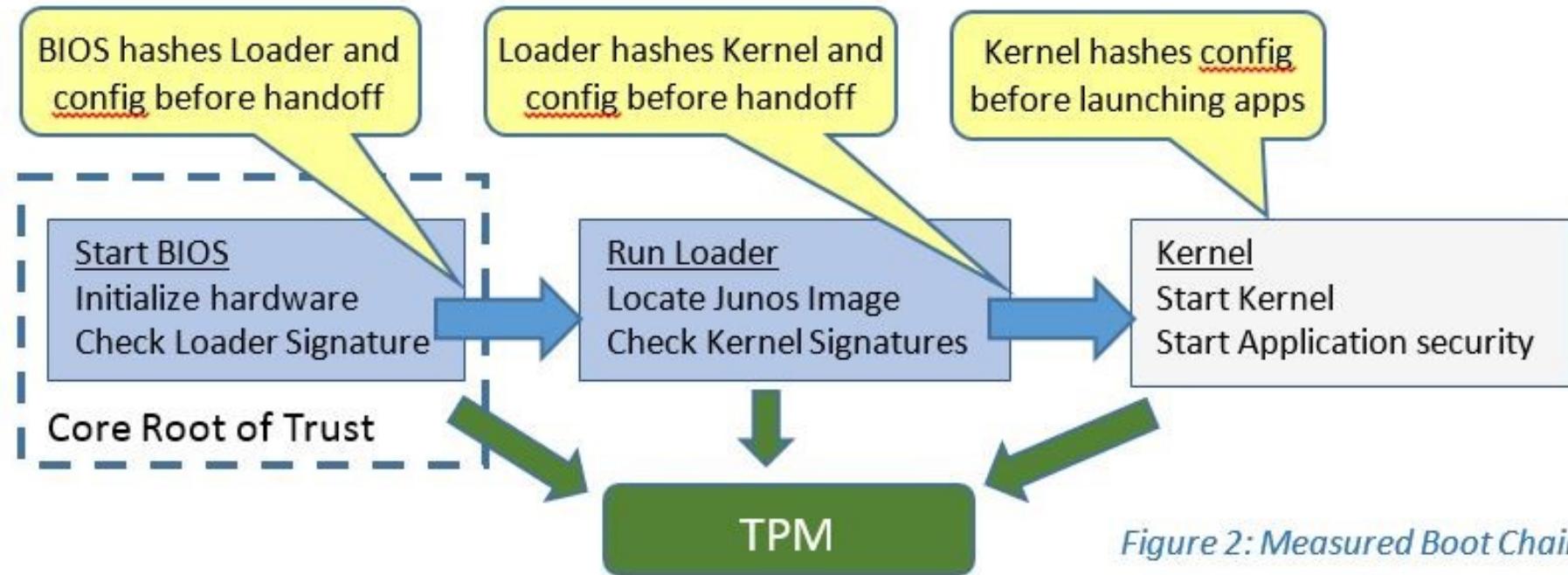
## Ứng dụng của TPM

- ❖ Đảm bảo toàn vẹn nền tảng
- ❖ Mã hoá ổ đĩa
- ❖ Bảo vệ mật khẩu
- ❖ Một số ứng dụng khác:
  - Quản lý bản quyền số
  - Bảo vệ và thực thi bản quyền phần mềm
  - Ngăn chặn lừa đảo trong game trực tuyến.

## Đảm bảo toàn vẹn nền tảng

- ❖ Mục đích chính của việc sử dụng TPM là đảm bảo tính toàn vẹn nền tảng:
  - Nền tảng là bất kỳ một thiết bị tính toán chạy HĐH bất kỳ
  - Nền tảng hoạt động đúng như nó được thiết kế.
- ❖ Cụ thể, TPM đảm bảo:
  - Hệ thống khởi động từ sự kết hợp tin cậy giữa phần cứng và phần mềm và
  - Tiếp tục cho đến khi HĐH khởi động xong và các ứng dụng đang hoạt động;
- ❖ Ví dụ:
  - TPM được sử dụng để đảm bảo tính toàn vẹn của bản quyền Microsoft Office 365 và bộ phần mềm Outlook Exchange.

## Đảm bảo toàn vẹn nền tảng

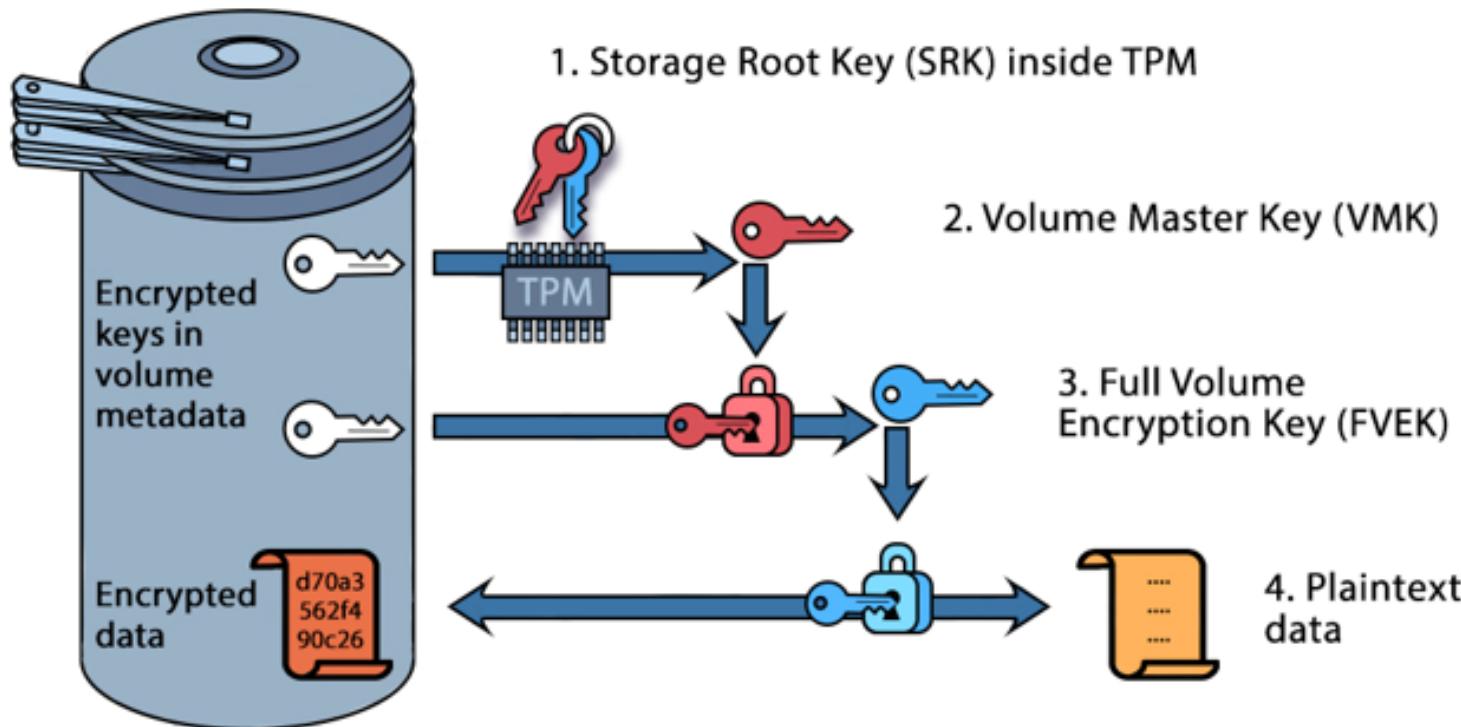


## Mã hóa ổ đĩa

- ❖ TPM có thể được sử dụng trong các công cụ mã hóa toàn bộ đĩa, như dm-crypt và BitLocker để:
  - Bảo vệ các khóa mã hóa các thiết bị lưu trữ của máy tính;
  - Cung cấp tính năng xác thực tính toàn vẹn cho một đường dẫn khởi động tin cậy bao gồm phần firmware và cung khởi động.

## Mã hóa đĩa

# BitLocker Keys



## Bảo vệ mật khẩu

- ❖ Các HĐH thường yêu cầu thực hiện việc xác thực (liên quan đến mật khẩu hoặc phương tiện khác) để bảo vệ các khoá, dữ liệu hoặc hệ thống;
  - Nếu thành phần này được thực hiện bằng phần mềm, nó có thể bị tấn công phá mật khẩu dựa trên từ điển.
- ❖ TPM là mô đun phần cứng chuyên dụng, được tích hợp sẵn cơ chế phòng chống hiệu quả các dạng tấn công dựa trên từ điển, nhất là các dạng tấn công tự động.

## Quản lý bản quyền số

- ❖ Tính toán tin cậy cho phép các công ty tạo ra một hệ thống quản lý quyền kỹ thuật số (DRM) mà sẽ rất khó bị phá vỡ,
  - Một ví dụ là tải xuống một tệp nhạc: Bộ nhớ với màn che có thể được sử dụng để ngăn người dùng mở tệp bằng máy tính trái phép.
  - Chứng thực từ xa có thể được sử dụng để chỉ cho phép phát bởi các trình phát nhạc thực thi các quy tắc của công ty thu âm.
  - Nhạc sẽ được phát từ bộ nhớ có màn che, điều này sẽ ngăn người dùng tạo bản sao không hạn chế của tệp khi đang phát và I / O an toàn sẽ ngăn chặn việc thu những gì đang được gửi đến hệ thống âm thanh.
  - Việc phá vỡ một hệ thống như vậy sẽ yêu cầu thao tác phần cứng của máy tính, thu tín hiệu tương tự bằng thiết bị ghi âm hoặc micro.

## Ngăn chặn lừa đảo trong game trực tuyến

- ❖ Tính toán tin cậy có thể được sử dụng để chống gian lận trong các trò chơi trực tuyến.
  - Một số người chơi sửa đổi bản sao trò chơi của họ để đạt được lợi thế không công bằng trong trò chơi;
  - Chứng thực từ xa, I/O an toàn và khóa bộ nhớ có thể được sử dụng để xác định rằng tất cả người chơi được kết nối với máy chủ đang chạy bản sao chưa sửa đổi của phần mềm.

## Các dịch vụ an toàn của TPM

- ❖ TPM khi được nhúng vào bên trong hệ thống PC nhằm cung cấp và đảm bảo các cơ sở tin cậy sau:
  - Cơ sở tin cậy cho việc đo kiểm (Root of Trust for Measurement- RTM): triển khai một cách tin cậy các thuật toán băm chịu trách nhiệm cho các biện pháp bảo vệ đầu tiên với nền tảng tính toán như tại thời điểm khởi động hay xác lập trạng thái tin cậy của hệ thống.
  - Cơ sở tin cậy cho lưu trữ (Root of Trust for storage - RTS) : triển khai tin cậy vị trí được bảo vệ cho việc lưu trữ một hay nhiều khóa bí mật và một khóa lưu trữ gốc SRK (storage root key);
  - Cơ sở tin cậy cho việc báo cáo (Root of Trust for reporting-RTR): triển khai tin cậy vị trí được bảo vệ để lưu khóa bí mật đại diện cho định danh duy nhất của hạ tầng, còn gọi là khóa chứng thực EK (endorsement key).

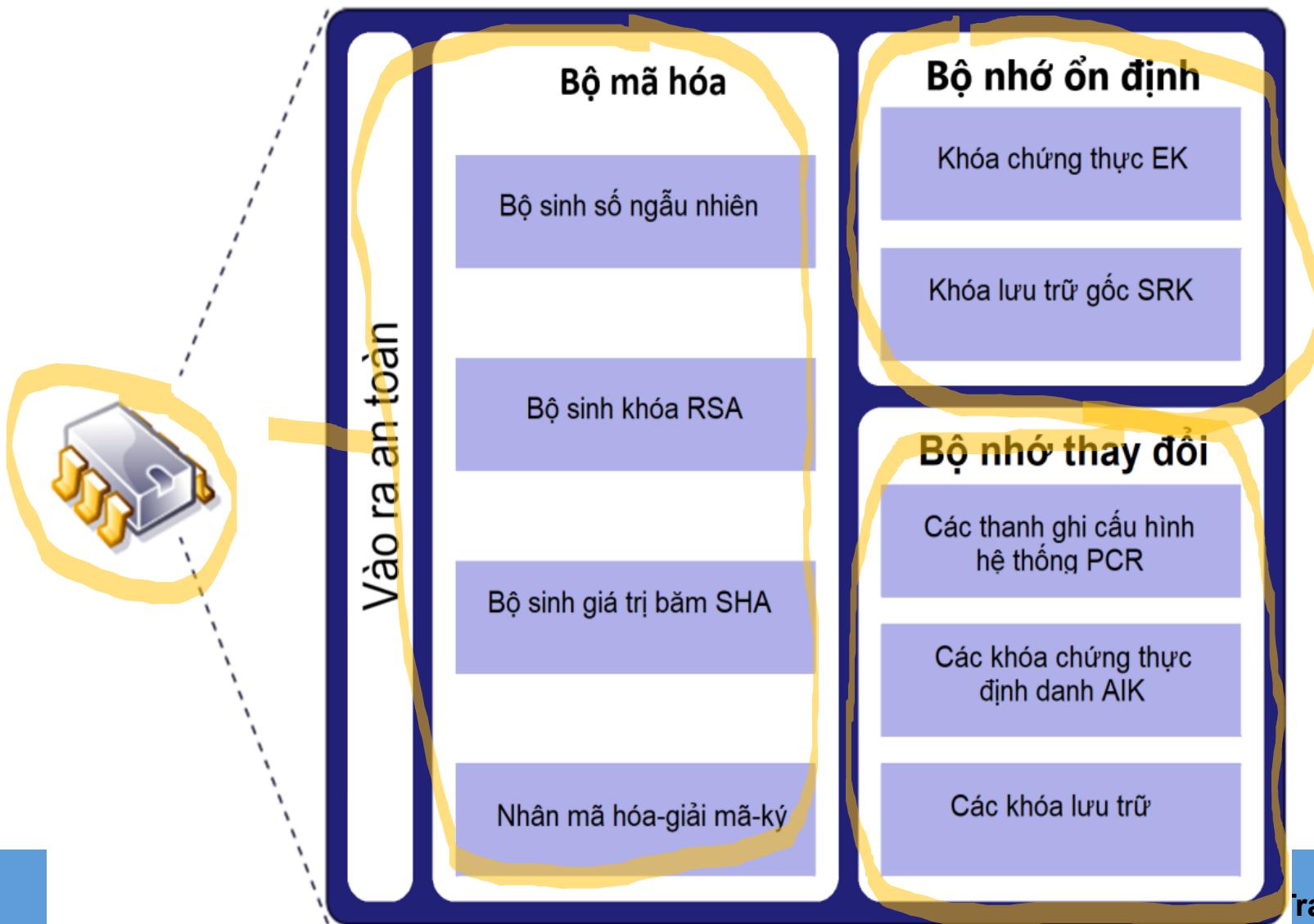
## Các dịch vụ an toàn của TPM

- ❖ Các khóa SRK và EK sử dụng cách mã hóa dị bộ (cơ chế mã hóa dùng khóa công khai).
- ❖ TPM có nhiệm vụ bảo vệ khóa bí mật trong cặp khóa trên.
- ❖ Phần khóa công khai EK được đăng ký với nơi chứng thực.
- ❖ Khóa EK tồn tại không đổi trong suốt thời gian hoạt động của hạ tầng tính toán và được sử dụng cho một số hạn chế các thủ tục.
  - Điều này là vì EK là duy nhất cho mỗi thiết bị TPM và có thể được dùng để nhận dạng thiết bị.

## Các dịch vụ an toàn của TPM

- ❖ Trong một số trường hợp, người dùng có thể tự quản lý khóa EK.
- ❖ Tuy nhiên khi đó người dùng không thể cung cấp chứng thực cho khóa nên việc ứng dụng bị hạn chế.
- ❖ SRK được thiết lập khi triển khai cho người dùng và có thể được khởi tạo lại khi thay đổi người dùng thông qua cơ chế xác lập quyền sở hữu với thiết bị TPM.

## Các bộ phận chức năng của TPM



## Các bộ phận chức năng của TPM

- ❖ Các khóa chứng thực định danh AIK (Attestation Identity Key) được tạo ra nhằm xác định định danh của hệ thống mà không phải cung cấp EK như là khóa dùng để ký.
- ❖ Việc này nhằm đảm bảo tính riêng tư của người sử dụng trong tình huống các nhà cung cấp dịch vụ yêu cầu bằng chứng về định danh người dùng.

## Các bộ phận chức năng của TPM

- ❖ Các thanh ghi cấu hình hệ thống PCR (Platform Configuration Register) được dùng để lưu lại các giá trị xác định tính toàn vẹn của hệ thống.
- ❖ Các giá trị lưu trong các thanh ghi này có thể tính toán độ toàn vẹn của bất cứ đoạn mã nào từ BIOS tới các ứng dụng chủ yếu là trước khi đoạn mã được thực hiện.
- ❖ TPM có tối thiểu 16 thanh ghi kiểu này, trong đó 8 thanh ghi đầu dành riêng cho TPM.

## Các bộ phận chức năng của TPM

- ❖ Bộ phận RTM (Roots of Trust for Measurement) và RTR (Roots of Trust for Reporting) là các bộ phận cơ bản để thiết lập tin cậy:
  - Nhờ vào thủ tục “niêm phong” (seal), bộ phận RTS (Roots of Trust for Storage) xác lập sự tin cậy với hệ thống cục bộ.
  - Đồng thời RTS cũng giúp bảo vệ tính bí mật cũng như cung cấp lưu trữ được bảo mật cho các ứng dụng hay người dùng không tin cậy trao đổi thông tin lẫn nhau.
- ❖ Bộ phận mã hóa là phần thực thi các hàm cần thiết dùng cho việc cung cấp các dịch vụ an toàn của mô-đun TPM như tạo số ngẫu nhiên, tính toán giá trị băm, ký số,...

## Các tác động của tính toán tin cậy

- ❖ Cách tiếp cận của tính toán tin cậy làm thay đổi mạnh mẽ thiết kế của hệ thống máy tính truyền thống và ứng dụng phân tán.
- ❖ Điều mới với tính toán tin cậy là việc đảm bảo chắc chắn phần mềm chạy cục bộ hay ở xa dựa trên cơ chế mã hóa sử dụng cách thức xác thực đảm bảo.
- ❖ Tính toán tin cậy ngăn chặn các vụ tấn công dựa trên phần mềm nhờ vào các thao tác thiết yếu cho hoạt động phải có sự chứng thực từ phần cứng TPM.

## Các chỉ trích/vấn đề đối với tính toán tin cậy

- ❖ Tính riêng tư: Không bảo vệ định danh người dùng với một số giao dịch;
- ❖ Kiểm soát của bên bán hàng: Bên bán hàng có thể sử dụng TPM khiến cho việc lựa chọn và thay đổi sản phẩm khó khăn hơn với người dùng cuối;
- ❖ Chứng thực: Việc chứng thực sử dụng chữ ký khó khăn do số lượng lớn phần mềm, nhất là khi các phần mềm này có thể được cập nhật định kỳ, vì vậy việc chứng thực cần thực hiện trên cơ sở hành vi của chương trình.

## Các chỉ trích/vấn đề đối với tính toán tin cậy

- ❖ Không hỗ trợ mã khóa đối xứng. Thiết kế của TPM dù dùng rất nhiều phần của bộ xử lý mã hóa song không cung cấp chức năng mã hóa đối xứng nào. Điều này ảnh hưởng tới vấn đề như thực thi luật pháp;
- ❖ Thực thi luật pháp: việc mã hóa mạnh tác động cả hai bên người dùng hợp lệ và người bẻ khóa.
  - Được dùng một cách đúng đắn, TPM giúp bảo vệ dữ liệu bí mật của người dùng. TPM mô tả một cách rõ ràng không có cửa hậu trong thiết bị hợp chuẩn.
  - Như vậy, không có cách thức hợp lệ nào khác ngoài thông tin của chủ sở hữu để truy cập vào các thông tin được bảo vệ.
  - Ngược lại, khi bị lạm dụng sẽ rất khó khăn cho các cơ quan thực thi luật pháp để khai thác các thông tin bí mật này.

### 3.4 Bảo vệ hệ thống file

- ❖ Giới thiệu
- ❖ Khởi động được bảo vệ
- ❖ Lưu trữ an toàn
  - Cơ chế lưu trữ an toàn TPM
  - Mã hoá ổ đĩa Bitlocker

## Giới thiệu

- ❖ Hệ thống file (File system) là công cụ của hệ điều hành cho quản lý, tổ chức lưu trữ các file dữ liệu.
- ❖ Hầu hết các HĐH đều hỗ trợ các cơ chế quản lý, phân quyền truy cập hệ thống file (NTFS, EXT3, EXT4,...).
- ❖ Hỗ trợ đảm bảo an toàn hệ thống file từ phần cứng (như TPM) giúp tăng cường an toàn:
  - Bảo vệ quá trình khởi động
  - Bảo vệ dữ liệu lưu trên đĩa (mã hoá ổ đĩa,...).

## Khởi động được bảo vệ

- ❖ Trong kiến trúc x86, BIOS (Basic Input Output System) chứa các phần mềm quan trọng được chạy trước tiên và cung cấp các dịch vụ cơ bản cho hệ điều hành.
- ❖ Phần BIOS có tất cả các đặc quyền truy cập tới toàn bộ phần cứng máy tính.
- ❖ Trong một số trường hợp BIOS có thể lập trình lại các thiết bị như thay đổi vị trí nhớ để nạp các đoạn mã của hệ điều hành hay cách thức thực hiện các giao dịch truy cập bộ nhớ trực tiếp DMA.

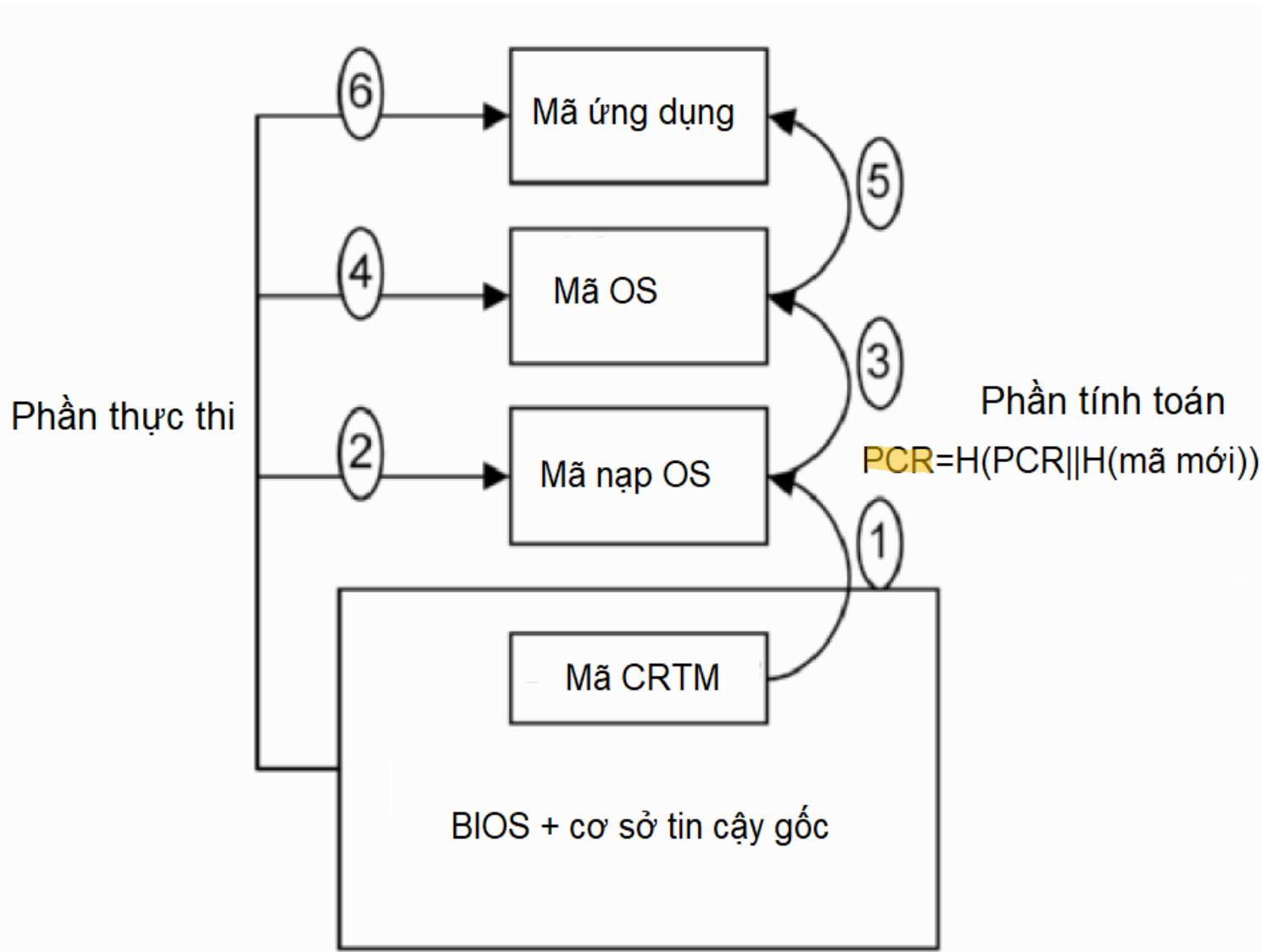
## Khởi động được bảo vệ

- ❖ Ngoài ra, BIOS cung cấp các đoạn mã cho chế độ quản lý hệ thống SMM (System Management Mode);
  - Các đoạn mã này được sử dụng trong toàn bộ thời gian hoạt động của hệ thống.
- ❖ Có thể thấy BIOS đóng vai trò quan trọng trong việc xác lập sự tin cậy trong suốt quá trình hoạt động của hệ điều hành.
- ❖ Tuy vậy, BIOS có thể chứa mã xấu do bên cung cấp xây dựng một cách có chủ đích như cửa hậu (backdoor) hoặc do người dùng sửa đổi BIOS mà không có cơ chế bảo vệ chống lại việc lập trình lại BIOS.

## Khởi động được bảo vệ dựa trên TPM

- ❖ TPM có thể được sử dụng để bảo vệ các phần mềm trong quá trình khởi động máy tính:
  - RTM (Root of Trust for Measurement) và RTR (Root of Trust for Reporting) là các thành phần căn bản để tạo dựng được sự tin cậy thông qua:
    - Quá trình khởi động được bảo vệ (measured boot) và;
    - Xác lập khởi động được xác thực (authenticated boot).

## Quá trình khởi động được bảo vệ



## Quá trình khởi động được bảo vệ

- ❖ Tại thời điểm khởi động, phần BIOS được kích hoạt và chạy trước.
- ❖ Sau đó, tại bước 2 BIOS tiến hành nạp đoạn mã nạp OS giúp khởi động hệ điều hành.
- ❖ Bước 4 là thực thi mã hệ điều hành. Cuối cùng, bước 6 là thực thi ứng dụng.
- ❖ Các bước 1, 3, và 5 tiến hành *tính toán cơ sở tin cậy cho các đoạn mã* và theo công thức  $PCR = H(PCR|H(\text{mã mới}))$  với H là hàm băm của RTM và PCR là các thanh ghi bên trong của TPM.

## Tính toán cơ sở tin cậy cho quá trình khởi động

1. RTM lưu lại chỉ số định danh của hệ thống vào vị trí an toàn là các thanh ghi PCR.
  - Chỉ số này đóng vai trò là giá trị tin cậy gốc CRTM (Core Root of Trust for Measurement) cho việc tính toán mức độ toàn vẹn của quá trình khởi động.
  - Đây có thể chỉ là biện pháp bảo vệ mã của hạ tầng tính toán hay đơn giản chỉ là định danh.
2. Trước khi khởi tạo các phần mã tiếp theo trong chuỗi khởi động, RTM tính toán mã băm của bộ phận đó và lưu lại vào vị trí an toàn. Sau đó chuyển quyền điều khiển cho phần mã đó.
3. Lặp lại bước 2 cho từng liên kết trong chuỗi khởi động.

## Bảo vệ tính toàn vẹn khởi động bằng TPM

- ❖ Tính toán mã băm của các đoạn mã khởi động và lưu vào TPM;
- ❖ Khi hệ thống khởi động:
  - Tính lại mã băm của đoạn mã trong chuỗi khởi động;
  - So sánh mã băm này với mã băm gốc của đoạn mã đã lưu trong TPM:
    - Nếu 2 mã băm trùng khớp thì cho phép nạp và thực hiện đoạn mã;
    - Nếu 2 mã băm khác biệt thì ngừng quá trình khởi động.

==> Đảm bảo tính toàn vẹn cho chuỗi khởi động.

## Bảo vệ tính toàn vẹn khởi động bằng TPM

- ❖ Các thông tin được coi là lưu trữ an toàn khi sử dụng các thanh ghi cấu hình của hạ tầng PCR (*Platform Configuration Registers*) bên trong TPM là nhờ các yếu tố sau:
  - Các ô nhớ này được bảo vệ bằng cách có thể đọc nhưng không thể ghi tùy ý;
  - Dữ liệu được ghi vào theo dạng tổ hợp với giá trị băm của dữ liệu hiện thời và giá trị trước đó.
- ❖ TPM cung cấp bản sao có xác nhận trạng thái PCR đảm bảo đối tác (chương trình) có thể kiểm tra trạng thái của hạ tầng tính toán.
- ❖ Điều quan trọng là việc xác nhận diễn ra bên trong TPM như vậy chống lại việc xâm nhập và giả mạo dữ liệu.

## Cơ sở tin cậy tĩnh và động

- ❖ Trong cơ sở tin cậy tĩnh, các giá trị băm được tính toán và lưu trữ cố định vào TPM cho tất cả các thành phần/chương trình trong chuỗi.
- ❖ Do chuỗi chương trình tham gia vào quá trình khởi động thường rất lớn nên việc tính toán mã băm cho toàn bộ chuỗi khởi động là không đơn giản, nhất là trong trường hợp các đoạn mã và chương trình từ hệ điều hành liên tục được cập nhật và chỉnh sửa.  
==> giải pháp linh hoạt bằng các sử dụng *cơ sở tin cậy động* (dynamic root of trust).

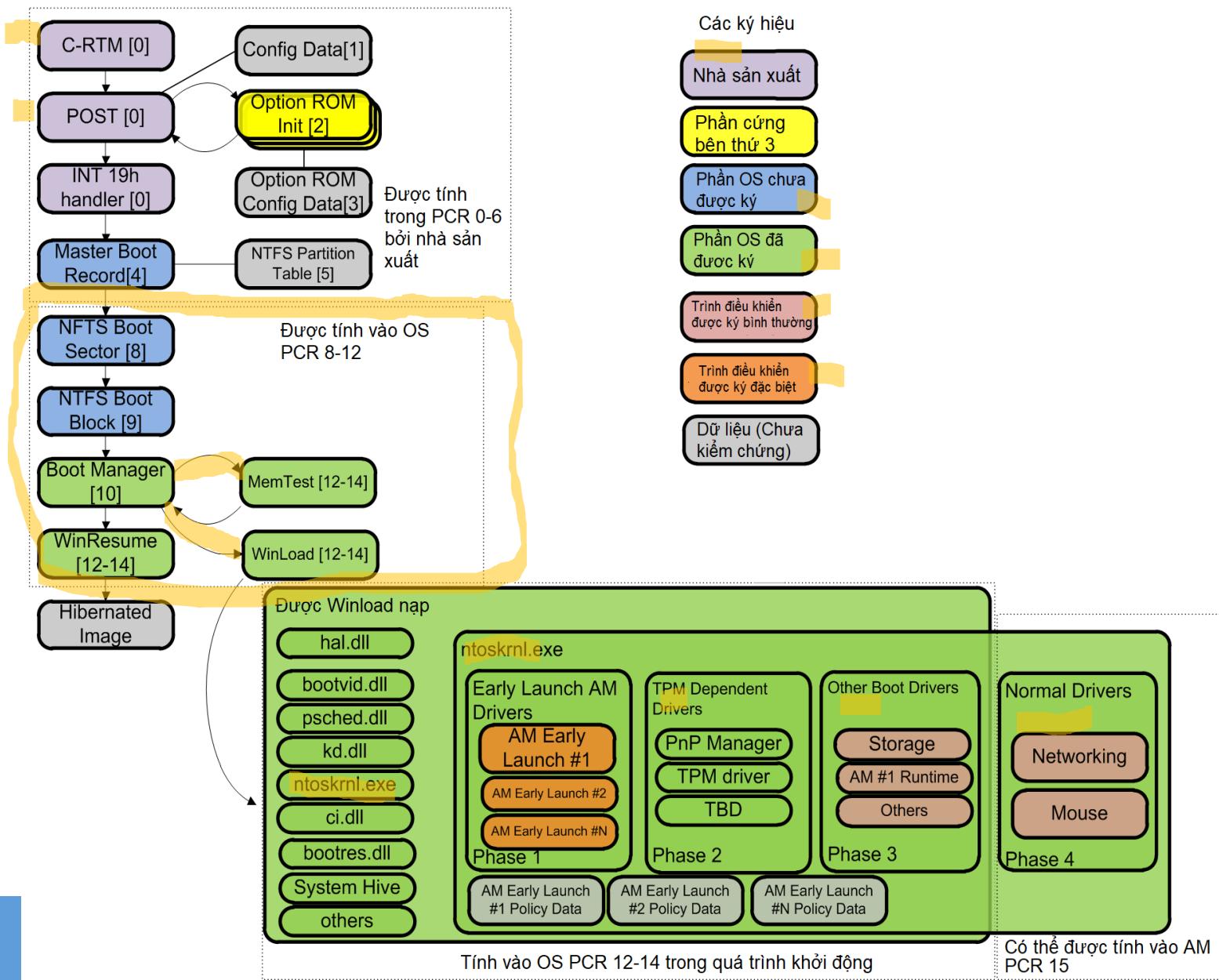
## Cơ sở tin cậy tĩnh và động

- ❖ Cơ sở tin cậy động: chuỗi tính toán sử dụng đoạn mã cố định từ nguồn tin cậy được phép nạp và chạy:
  - Chương trình được chọn như vậy giúp giảm độ dài của chuỗi khởi động.
- ❖ Quá trình khởi động được bảo vệ và kết hợp với phần mềm chống mã độc trong Windows để nâng cao tính toàn vẹn của hệ thống sử dụng BIOS truyền thống.

# BÀI GIẢNG MÔN HỌC AN TOÀN HỆ ĐIỀU HÀNH

## CHƯƠNG 3 – AN TOÀN CÁC DỊCH VỤ CƠ BẢN CỦA HĐH

Khởi động  
được bảo  
vệ trong  
Windows.



## Quá trình khởi động được bảo vệ trong Windows

- ❖ Quá trình khởi động được bảo vệ trong Windows gồm các giai đoạn (số hiệu các thanh ghi an toàn PCR trong []):
  - Cơ sở tin cậy gốc CRTM và bộ phận firmware khởi động chuỗi tính toán cơ sở tin cậy;
  - Thành phần kiểm tra khi bật máy POST;
  - Nạp bootstrap qua ngắt 19H;
  - Nạp Master Boot Record
  - Nạp NTFS boot
  - Nạp Boot Manager
  - WinLoad: Nạp các thành phần của Windows theo 4 pha (Phase 1, 2, 3, 4).
    - AM (Anti-Malware)
    - TPM driver...

## Quá trình khởi động được bảo vệ trong Windows

- ❖ Các thành phần được WinLoad nạp trong các phase của quá trình nạp OS gồm các trình điều khiển:
  - Trình phòng chống mã độc nạp sớm (Early Launch AM - AntiMalware);
  - Trình điều khiển TPM;
  - Trình điều khiển khởi động khác.
- ❖ AM cần được nạp và chạy ở chế độ đặc quyền.

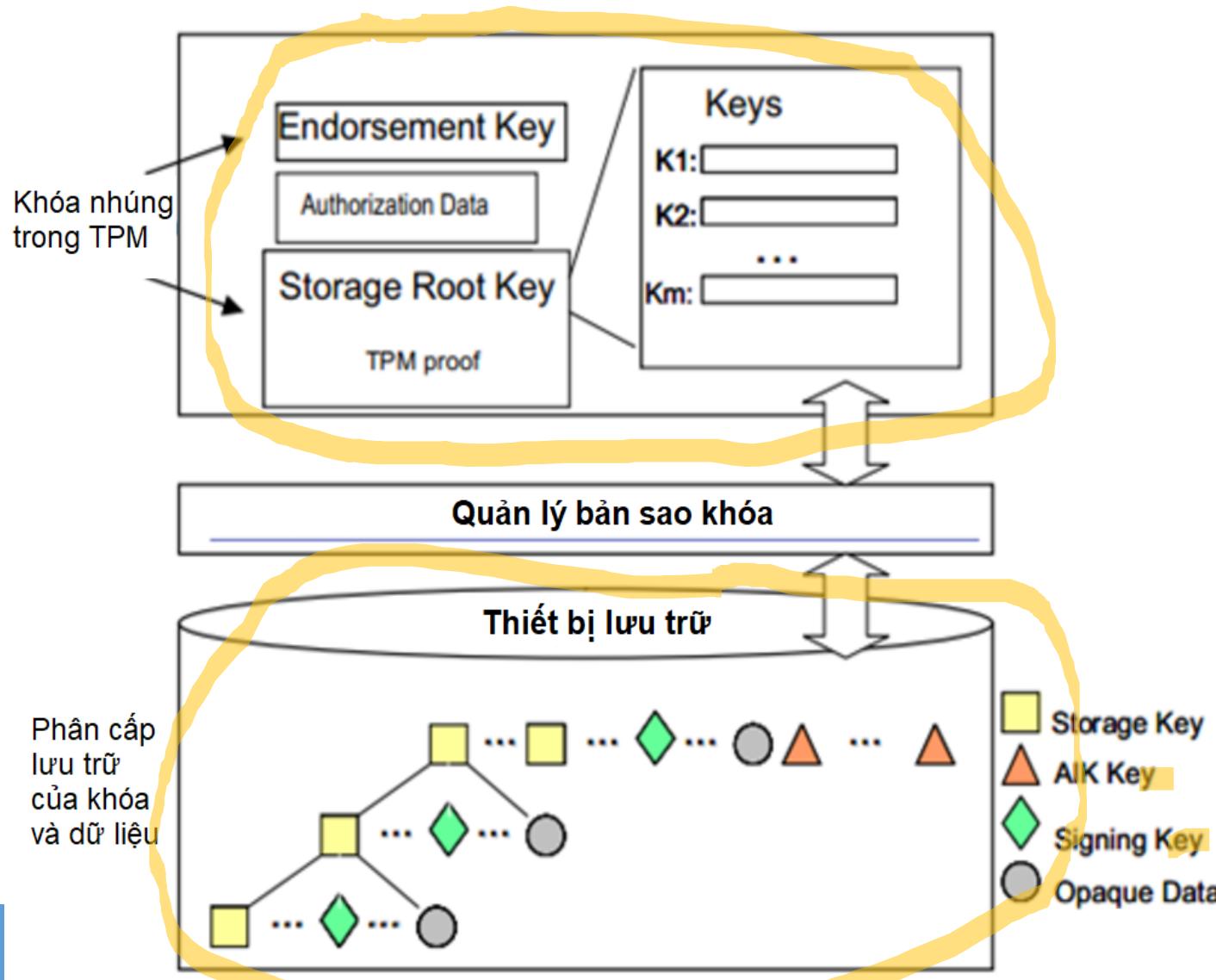
## Lưu trữ an toàn - Cơ chế lưu trữ an toàn TPM

- ❖ Việc lưu trữ an toàn hay tin cậy thường sử dụng kỹ thuật mã hóa một phần hoặc toàn bộ thiết bị lưu trữ nhằm đảm bảo tính bí mật của dữ liệu lưu trên thiết bị.
- ❖ Một cách tiếp cận phổ biến là mã hóa dựa vào dịch vụ hệ thống file của hệ điều hành.
  - Khi đó TPM cung cấp khóa mã cho thiết bị được mã hóa và đóng dấu (seal) sao cho khóa mã chỉ được cung cấp khi có cấu hình hợp lệ.

## Lưu trữ an toàn - Cơ chế lưu trữ an toàn TPM

- ❖ Mã hóa thiết bị lưu trữ đảm bảo khi các thiết bị bị mất trộm, thông tin lưu trữ không thể giải mã được do cấu hình (ngữ cảnh) chính xác để thực hiện việc này không tồn tại trong hệ thống của kẻ tấn công.
- ❖ Máy tính vẫn khởi động song kẻ tấn công cần các thông tin đăng nhập hợp lệ mới có thể truy cập vào dữ liệu của thiết bị.
  - Bitlocker là một tính năng tiêu biểu do Microsoft phát triển và cung cấp cho việc bảo vệ dữ liệu của người dùng.

## Quá trình mã hóa dựa trên TPM



## Quá trình mã hóa dựa trên TPM

- ❖ Khóa lưu trữ gốc SRK (Storage Root Key) được tạo ra bởi TPM và luôn được lưu trong thiết bị này.
- ❖ Khóa SRK chỉ có thể truy cập được khi người dùng cung cấp dữ liệu bí mật, hay dữ liệu cấp phép SRK.
- ❖ Dữ liệu bí mật này tương tự như dữ liệu cấp phép chủ sở hữu và được nạp vào TPM cùng lúc trong quá trình xác lập quyền sở hữu.
- ❖ Cùng với dữ liệu cấp phép sở hữu, dữ liệu cấp phép SRK được mã hóa bởi khóa chứng thực EK (Endorsement Key) trước khi được gửi tới TPM.
  - Khóa SRK tạo nên gốc cho việc phân cấp các khóa.

## Quản lý bản sao khóa

- ❖ Cơ chế phân cấp khóa SRK cho phép dữ liệu hay các khóa được mã hóa sao cho chúng chỉ có thể được giải mã thông qua TPM.
  - Các dữ liệu được mã hóa sử dụng khóa lưu trữ (SK - Storage Key) cụ thể.
  - Khóa SK được mã hóa và lưu bên ngoài TPM.
  - Để truy cập dữ liệu, bản mã khóa SK được nạp vào trong TPM thông qua bộ quản lý bản sao khóa (Key cache manager) và giải mã bằng khóa lưu trữ gốc SRK.
    - Khóa SK có được sau giải mã được sử dụng để mã hóa/giải mã dữ liệu.
  - Do khóa SRK luôn lưu bên trong TPM và TPM được gắn cứng vào máy tính nên dữ liệu mã hóa chỉ có thể được giải mã từ máy tính đó.

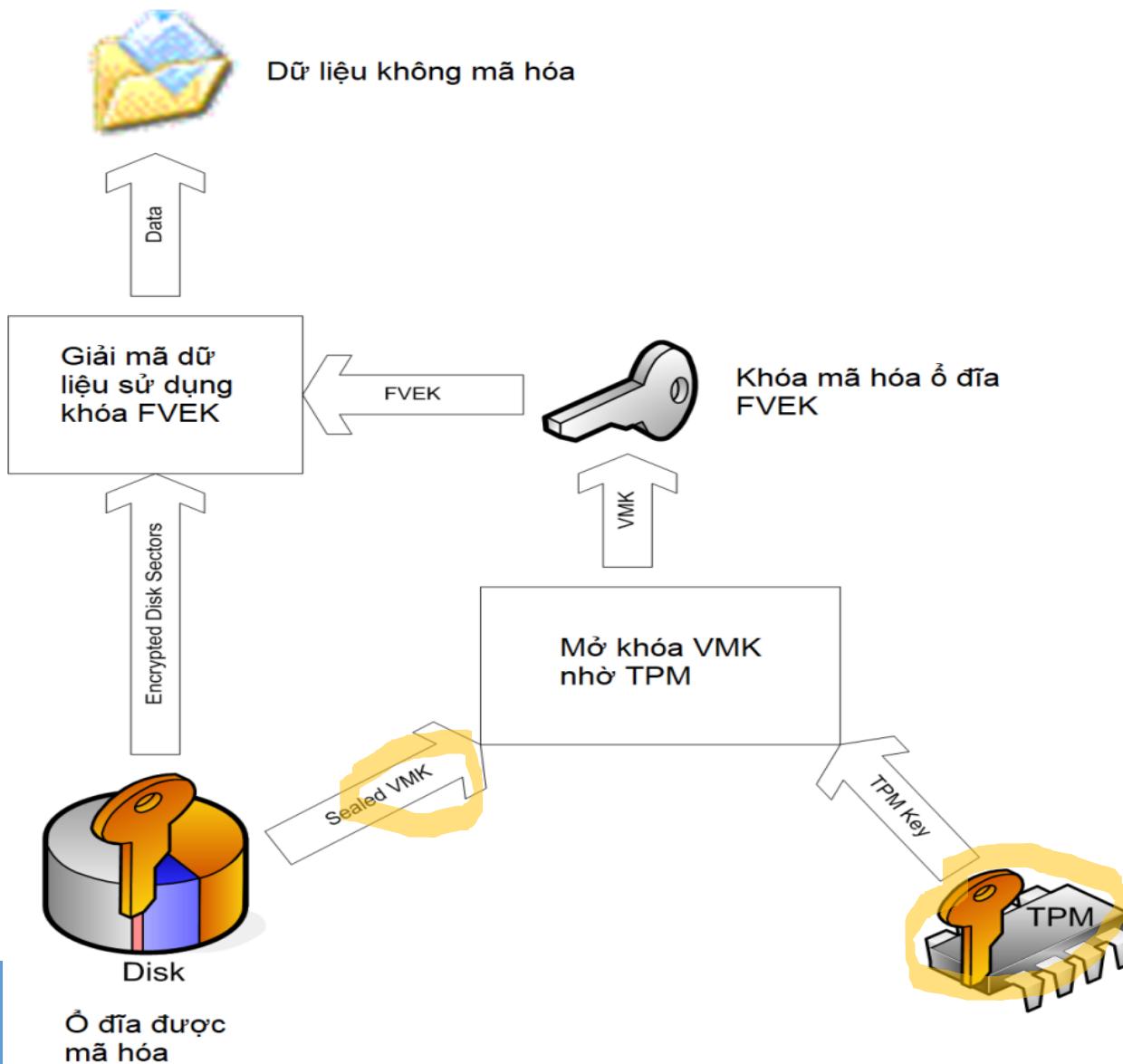
## Cơ chế mã hóa dựa trên TPM

- ❖ TPM cung cấp hai cơ chế khác cho việc lưu trữ an toàn là gắn (binding) và đóng dấu (sealing):
  - Thao tác gắn mã hóa dữ liệu sử dụng khóa được quản lý như cách đã mô tả.
  - Quá trình đóng dấu mở rộng thêm bằng cách chỉ cho phép giải mã được tiến hành khi máy tính ở trạng thái xác định.
    - Trạng thái này được thể hiện thông qua các dữ liệu lưu trong các thanh ghi PCR.
    - Như vậy, khi dữ liệu được đóng dấu, không chỉ máy tính phải cùng cấu hình mà máy tính còn phải ở trạng thái xác định trước trước khi dữ liệu có thể được giải mật.

## Mã hóa ổ đĩa Bitlocker

- ❖ Bitlocker là dịch vụ mã hóa ổ cứng kết hợp với TPM để tăng khả năng bảo vệ do Microsoft cung cấp.
- ❖ Điều này đảm bảo không chỉ đảm bảo an toàn dữ liệu người dùng mà còn chống lại việc xâm nhập máy tính khi hệ thống không hoạt động.
- ❖ Bitlocker có khả năng kiểm chứng tính toàn vẹn của file cho khởi động và hệ thống trước khi giải mã vùng lưu trữ được bảo vệ.

## Quá trình giải mã ổ đĩa BitLocker



## Mã hóa ổ đĩa Bitlocker

- ❖ Khi Bitlocker được kích hoạt, toàn bộ nội dung của ổ đĩa dành cho hệ điều hành được mã hóa bằng khóa mã toàn bộ ổ đĩa FVEK (full-volume encryption key) và khóa này lại được bảo vệ bằng khóa mã chủ VMK (volume master key).
- ❖ Việc bảo vệ khóa chính VMK sẽ giúp bảo vệ gián tiếp tính an toàn của dữ liệu trên ổ đĩa. Với sự hỗ trợ từ TPM, khóa VMK được đóng dấu và bảo vệ bằng phần cứng bên trong TPM.

## Mã hóa ổ đĩa Bitlocker

- ❖ Việc truy cập tới các dữ liệu bên trong ổ đĩa chỉ được phép khi TPM kiểm chứng thành công tính toàn vẹn của các phần tham gia vào quá trình khởi động máy tính.
  - Cách kiểm chứng mặc định của TPM bảo vệ khóa VMK chống lại các thay đổi trong đoạn mã MBR, sector khởi động NTFS, và phần quản lý khởi động và các thành phần quan trọng khác như đã thể hiện trong quá trình khởi động bảo vệ.
- ❖ Tất cả các khóa mã sử dụng trong quá trình khởi tạo của Bitlocker không truy cập được từ phía người dùng cũng như không thể được thay đổi, sao chép hay rút lại.

## Mã hóa ổ đĩa Bitlocker

- ❖ Như vậy, TPM cung cấp cơ chế cho phép lưu trữ các khóa sử dụng cho BitLocker một cách an toàn chống lại việc sao chép và xâm nhập.
- ❖ Hệ thống file mã hóa có thể được dùng cùng với BitLocker để bảo vệ hệ điều hành khi chạy.
- ❖ BitLocker không bảo vệ được các file dữ liệu khi người dùng có quyền truy cập vào hệ thống;
  - Việc bảo vệ các file khỏi các chương trình và người dùng trong hệ thống chỉ có thể được thực hiện bằng phần mềm mã hóa độc lập.

### 3.5 Phân tích các vấn đề an toàn của các hệ điều hành Windows và Unix/Linux

- ❖ Microsoft Windows
- ❖ Unix và Linux
- ❖ Mô-đun an toàn cho Linux
- ❖ Nhân SCOMP

## Microsoft Windows

- ❖ Giới thiệu
- ❖ Các cơ chế đảm bảo an toàn
- ❖ Cơ chế xác thực
- ❖ Đánh giá
- ❖ Các vấn đề / lỗ hổng bảo mật tiêu biểu

## Microsoft Windows - Giới thiệu

- ❖ HĐH Windows không hỗ trợ an toàn, phân quyền (dựa trên MS-DOS):
  - Windows 3.x
  - Windows 95, 98, Me.
- ❖ HĐH Windows hỗ trợ an toàn, phân quyền (dựa trên kiến trúc mới hoàn toàn):
  - Windows NT workstation, server
  - Windows 2000
  - Windows XP, Vista
  - Windows 7, Windows 8, Windows 10, Windows 11
  - Windows server 2000, 2003, 2008, 2012, 2016, 2019.

## Microsoft Windows - Giới thiệu

- ❖ MS-Windows ban đầu được thiết kế hướng tới người dùng PC riêng lẻ, không kết nối với mạng và vấn đề an toàn không được đặt ra với hệ thống như vậy.
- ❖ Người dùng tự quản trị hệ thống của mình, việc cài đặt và chạy phần mềm theo nhu cầu của người dùng.
- ❖ Sự ra đời của dịch vụ Web khiến cho việc kết nối các máy tính Windows trở thành dịch vụ cơ bản và các dịch vụ mạng người dùng sử dụng như thư điện tử, duyệt Web, tải phần mềm dễ dàng làm lộ ra các điểm yếu mà Windows không được thiết kế để đối phó lại.

## Microsoft Windows - Giới thiệu

- ❖ Windows sử dụng mô hình mở, mềm dẻo, cho phép người dùng tự quản trị khiến cho hệ điều hành này trở thành mục tiêu dễ dàng của kẻ tấn công.
- ❖ Trong một thời gian dài, Microsoft khá chậm chạp trong việc xử lý các mối đe dọa này.
- ❖ Gần đây, Microsoft đã tập trung xử lý và có một số thành công trong việc giảm thiểu lỗ hổng nhờ vào các qui định xây dựng mã chương trình tốt hơn, các công cụ phân tích mã và cấu hình hệ thống an toàn hơn.

## MS Windows - Các cơ chế đảm bảo an toàn

- ❖ Cơ chế bảo vệ của Windows NT cung cấp mô hình kiểm soát truy cập theo kiểu tùy chọn để quản lý các trạng thái bảo vệ, dán nhãn các đối tượng và dịch chuyển miền bảo vệ.
- ❖ Hệ thống bảo vệ của Windows có khả năng mở rộng và dễ biểu diễn các quyền cũng như là người dùng của hệ thống. Thực tế cho thấy, các cải thiện về tính mở rộng và biểu diễn có ảnh hưởng tiêu cực đến tính an toàn của hệ thống.

## MS Windows - Các cơ chế đảm bảo an toàn

### ❖ Các chủ thể của Windows cũng tương tự như Unix:

- Mỗi tiến trình được gán một thẻ (token) mô tả định danh của tiến trình.
- Định danh bao gồm định danh an ninh SID (Security Identifier Descriptor) của người dùng, SID nhóm, bí danh SID (Alias) để hoạt động bằng định danh SID khác, danh sách các quyền.
- Định danh Windows liên kết với người dùng, tuy nhiên thẻ của tiến trình cho người dùng đó có thể gắn với bất kỳ tổ hợp quyền nào.

## MS Windows - Các cơ chế đảm bảo an toàn

- ❖ Các đối tượng cần kiểm soát trong Windows gồm:
  - File
  - Các đối tượng do chương trình người dùng định nghĩa và thêm vào thư mục động (*active directory*).
- ❖ Từ góc độ kiểm soát truy cập, các kiểu đối tượng được xác định thông qua các tập thao tác của chúng.
- ❖ Mô hình Windows hỗ trợ cách nhìn tổng quát lên các thao tác mà đối tượng có thể sở hữu.

## MS Windows - Các cơ chế đảm bảo an toàn

- ❖ Windows mô tả tới hơn 30 thao tác với mỗi kiểu đối tượng bao gồm một số thao tác riêng biệt cho kiểu dữ liệu.
- ❖ Thậm chí cho các đối tượng file, Windows định nghĩa nhiều hơn các thao tác như truy cập thuộc tính file, đồng bộ các thao tác.
- ❖ Ngoài ra, các ứng dụng có thể thêm các kiểu đối tượng mới và tự định nghĩa các thao tác lên đối tượng đó.

## MS Windows - Các cơ chế đảm bảo an toàn

- ❖ Windows sử dụng danh sách kiểm soát truy cập ACL để đảm bảo an toàn.
  - ACL bao gồm tập các mục kiểm soát truy cập ACE (Access control entry).
  - Mỗi mục mô tả các thao tác mà một SID có thể thực hiện trên đối tượng đó.
  - Các hoạt động trong ACE được diễn giải dựa trên kiểu đối tượng.
  - Trong Windows, các ACE có thể cho phép cũng như từ chối bất kỳ truy cập nào.

## Danh sách kiểm soát truy cập và thẻ của các tiến trình



## MS Windows - Cơ chế xác thực

- ❖ Các yêu cầu xác thực được xử lý bởi Bộ tham chiếu an toàn (Security Reference Monitor – SRM).
- ❖ SRM là phần mềm chạy trong nhân và nhận các tham số đầu vào thẻ tiến trình, SID của đối tượng và tập thao tác, trả về kết quả của yêu cầu truy cập (chấp nhận/tù chối) trên cơ sở ACL mà nó tìm thấy.
- ❖ Do sử dụng quyền tù chối, cách thức SRM xử lý các truy vấn xác thực phức tạp hơn so với Unix.

## MS Windows - Cơ chế xác thực

❖ Sự khác biệt chủ yếu là thứ tự của các mục ACE:

- SRM tìm kiếm trong danh mục ACE cho đến khi tìm thấy ACE chấp thuận truy cập hay ACE từ chối.
- Nếu ACE chấp thuận thao tác cần thiết thì yêu cầu được cho phép.
- Tuy nhiên nếu tìm thấy ACE từ chối mà chứa các thao tác được yêu cầu thì toàn bộ yêu cầu bị từ chối.
- Như trong ví dụ trên, tiến trình P1 sẽ chỉ được phép thực thi đọc mà không được ghi vì yêu cầu nằm trong mục bị từ chối.

## MS Windows - Cơ chế xác thực

- ❖ Bộ quản lý đối tượng đảm bảo việc đúng trung gian (ngăn chặn) cho các yêu cầu truy cập.
  - Các bộ quản lý đối tượng chạy trong nhân, nhưng các bộ phận này là các thực thể độc lập với nhau.
  - Điều này mang lại lợi thế cho việc mô-đun hóa, nhưng lại làm nảy sinh thách thức cho hệ thống trong việc ngăn chặn các yêu cầu.
  - Điều cần thiết là các bộ quản lý đối tượng cần ngăn chặn mọi thao tác và xác định một cách chính xác quyền cho những thao tác này.
  - Không có quá trình nào đảm bảo điều này trong Windows.

## MS Windows - Cơ chế xác thực

- ❖ Trong Windows, nền tảng tính toán tin cậy bao gồm toàn bộ các dịch vụ hệ thống và các tiến trình hoạt động sử dụng định danh người dùng tin cậy như *Administrator*.
- ❖ Windows cung cấp cơ chế tương tự như *setuid* của Unix/Linux để gọi các dịch vụ Windows để chạy với đặc quyền định trước, tối thiểu đủ để hỗ trợ tất cả người dùng.
- ❖ Như vậy, lỗ hổng trong các dịch vụ như vậy có thể dẫn đến hệ thống bị chọc thủng.
- ❖ Hơn thế, việc dễ dàng cài đặt phần mềm và độ phức tạp của mô hình kiểm soát truy cập tùy chọn Windows thường khiến cho người dùng sử dụng tài khoản có đặc quyền, *Administrator*.

## MS Windows - Cơ chế xác thực

- ❖ Với các phiên bản sau này, *mô hình của Windows được mở rộng để ngăn cản các chương trình tải về từ Internet được tự động cập nhật vào hệ thống bất kể định danh người dùng là gì.*
- ❖ Mặc dù điều này đảm bảo mức độ toàn vẹn nhất định song không hoàn toàn bảo vệ tính toàn vẹn của hệ thống.
- ❖ Cải tiến này không ngăn cản việc gọi, các yêu cầu có mục đích xấu hay giả mạo các đoạn mã có độ an toàn cao nhúng trong file có độ toàn vẹn thấp.

## MS Windows - Đánh giá

### ❖ Mô hình kiểm soát truy cập của Windows:

- Ưu điểm:
  - Mềm dẻo
  - Có khả năng biểu diễn tốt.
- Nhược điểm:
  - Khó quản trị
  - Nhìn chung kém an toàn hơn so với Unix/Linux.

## MS Windows - Đánh giá

- ❖ Việc ngăn chặn các truy cập trái phép được các bộ quản lý đối tượng thực hiện, tuy nhiên:
  - Không có mã nguồn (mã đóng) nên không thể biết nó được thực hiện ở đâu?
  - Windows cho phép mở rộng các bộ quản lý đối tượng --> tạo thêm các bộ phận không an toàn.

## MS Windows - Đánh giá

❖ Vấn đề phát sinh do Windows sử dụng cơ chế kiểm soát truy cập tùy chọn (DAC):

- Các tiến trình không tin cậy của người dùng có thể sửa đổi quyền truy cập đến dữ liệu của người dùng một cách tùy ý;
- Như vậy, việc đặt mục tiêu an toàn với dữ liệu người dùng là không khả thi;
- Do người dùng thường sử dụng tài khoản *Administrator* để thuận tiện cho việc quản trị, nên bất kỳ khía cạnh nào của hệ thống bảo vệ cũng có thể bị sửa đổi.

## MS Windows - Đánh giá

### ❖ Các biện pháp bảo vệ nhôm: Có nhiều hạn chế:

- Nhôm của Windows có thể bị sửa đổi thông qua các mô-đun của nhôm:
  - Quá trình đóng dấu đoạn mã có thể dùng để chứng thực mô-đun nhôm. Như vậy, người đóng dấu không nhất thiết phải là người viết mã.
  - Người quản trị phải chịu trách nhiệm về tính tin cậy của người đóng dấu.
- Thủ tục an toàn phụ thuộc vào quyết định chủ quan của người dùng thường dễ bị lỗi do người dùng thường thiếu hiểu biết về những vấn đề như vậy.
- Mặt khác, nhôm Windows không xác định các cách bảo vệ lời gọi hệ thống.

## MS Windows - Đánh giá

❖ Nền tảng tính toán tin cậy của Windows: không thực sự chặt chẽ:

- Gần như bất kỳ chương trình nào cũng có thể nằm trong nền tảng tin cậy của hệ thống và
- Bất kỳ tiến trình được kích hoạt từ các chương trình này đều có thể sửa đổi các chương trình tin cậy khác và điều này làm mất hiệu lực của nền tảng tin cậy.

## MS Windows - Đánh giá

### ❖ Về giao diện lập trình ứng dụng (API):

- Windows cung cấp giao diện lập trình cho phép một tiến trình xâm nhập các tiến trình khác, như các hàm CreateRemoteThread, OpenProcess hay WriteProcessMemory;
- Các hàm thư viện này cho phép khởi tạo các luồng trong tiến trình khác hay chèn mã vào tiến trình trước khi khởi tạo luồng;
- Mặc dù các hàm này cần có đặc quyền để được sử dụng và phục vụ mục đích phát hiện lỗi của tiến trình;
- Tuy nhiên, cần phải đảm bảo các đặc quyền này không bị lợi dụng để bảo đảm cho các cơ chế chống xâm nhập của cơ sở tính toán tin cậy.

## MS Windows - Đánh giá chung

- ❖ Các cơ sở của tính đúng đắn trong Windows là không chính xác và Windows có cơ sở tính toán tin cậy không bị giới hạn về kích cỡ cũng như hệ thống nhân có khả năng mở rộng.
  - Điều này khiến cho việc đánh giá chính xác khó có kết quả.
- ❖ Mô hình khái quát của cơ chế bảo vệ của Windows cho phép mô tả các tổ hợp quyền nhưng lại không có bất cứ mục tiêu an toàn cụ thể được xác định trong hệ thống.
  - Vì vậy, không thể nói được liệu hệ thống có an toàn hay không.
- ❖ Do mô hình Windows phức tạp hơn Unix và có thể mở rộng tùy ý nên việc kiểm chứng tính an toàn nói chung khó khăn hơn.

## Các vấn đề / lỗ hổng bảo mật tiêu biểu

- ❖ Sổ đăng ký (Windows registry)
- ❖ Người dùng quản trị
- ❖ Các tính năng ngầm định.

## Windows registry

- ❖ Windows registry là CSDL toàn cục được tổ chức theo mô hình phân cấp (cây) để lưu các dữ liệu của Windows và toàn bộ các chương trình;
- ❖ Khi chương trình mới được nạp, nó có thể cập nhật registry theo yêu cầu cụ thể của chương trình gồm các thông tin nhạy cảm như đường dẫn, thư viện.
- ❖ Mỗi mục đăng ký có thể được gắn với các ngữ cảnh an ninh và hạn chế truy cập, song các hạn chế này không được sử dụng một cách hiệu quả.

## Windows registry

- ❖ Việc chỉ định rõ ràng tên các thư viện cần thiết cho ứng dụng không phải là cách mà các hãng phần mềm thường sử dụng một cách rộng rãi:
  - Người dùng không muốn phần mềm mà họ mới mua lại không thể chạy một cách đúng đắn vì nó không có được thư viện cần thiết nên thường chạy phần mềm với quyền tối đa (admin);
  - Hãng sản xuất phần mềm không biết được cách thức quản trị Windows của người dùng nên giải pháp tình thế là mở các quyền để chấn chấn phần mềm sẽ chạy bất kể người dùng quản trị thế nào.

## Người dùng quản trị

- ❖ Trong Windows, người dùng thường chạy dưới định danh quản trị hay với các đặc quyền được chấp thuận:
  - Điều này là vì người dùng cần quyền truy cập rộng hơn để có thể sử dụng các chức năng cần thiết cho phép hệ thống hoạt động.
  - Nếu người dùng tải về phần mềm trò chơi, người dùng có thể cần đặc quyền để cài đặt và chắc chắn cần các đặc quyền khi chạy phần mềm trò chơi sử dụng nhiều tài nguyên.
  - Cuối cùng, người dùng có thể muốn kiểm chứng lý do phần mềm trò chơi không chạy và cho phép tất cả các đặc quyền để xử lý vấn đề.
- ❖ Unix thường được dùng bởi người dùng có nhiều kinh nghiệm hơn và hiểu rõ sự khác biệt giữa cài đặt và các hoạt động thông thường của máy tính.
  - Như vậy, việc vận dụng đặc quyền hợp lý hơn người dùng Windows.

## Các tính năng ngầm định

- ❖ Nhiều tính năng, nội dung ngầm định (default) có sẵn trong Windows có thể cung cấp nội dung nhạy cảm hoặc “đòn bẩy” giúp tin tặc tấn công hệ thống:
  - Dịch vụ Windows Remote Registry cho phép kết nối và chỉnh sửa nội dung Registry từ xa;
  - Sử dụng người dùng quản trị (Administrator) ngầm định khi cài đặt và sử dụng hệ thống;
  - Nhiều tính năng nội dung ngầm định dành cho người quản trị trong các dịch vụ, ứng dụng, nhưng không được giới hạn truy cập đúng mức cần thiết:
    - Siêu thủ tục xp\_cmdshell trong MS-SQL server
    - Tính năng cho xem mã nguồn, hoặc quản trị từ xa trong MS-IIS....

## Unix và Linux

- ❖ Giới thiệu
- ❖ Hệ thống bảo vệ
- ❖ Hệ thống xác thực
- ❖ Đánh giá
- ❖ Các vấn đề / lỗ hổng tiêu biểu

## Unix - Giới thiệu

- ❖ Unix được viết bằng ngôn ngữ C khởi nguồn từ Bell Labs của AT&T vào những năm 70 của thế kỷ 20;
- ❖ Là HĐH đầu tiên có tính khả chuyển (có thể chạy được trên nhiều phần cứng khác nhau);
- ❖ Unix thu hút được cộng đồng phát triển đông đảo;
- ❖ Unix có giao diện chương trình (API) thuận tiện cho người phát triển.

## Unix - Giới thiệu

- ❖ Unix hướng đến một chương trình căn bản nhỏ gọn, gọi là nhân (kernel) với giao diện chuẩn để đơn giản hóa việc phát triển ứng dụng;
- ❖ Mục tiêu thiết kế của Unix là phát triển nền tảng chung có thể chia sẻ dễ dàng giữa các người dùng với nhau.
- ❖ Hệ thống Unix bao gồm nhân hệ điều hành và các tiến trình.

## Unix - Giới thiệu

- ❖ Như vậy, mục tiêu an toàn của Unix là bảo vệ dữ liệu người dùng khỏi các lỗi vô tình trong chương trình người dùng.
  - Tuy nhiên, việc bảo vệ này không đảm bảo yêu cầu về tính bí mật và toàn vẹn.
- ❖ Cơ chế an toàn Unix nhằm bảo vệ người dùng với nhau và hạ tầng tính toán tin cậy của hệ thống khỏi toàn bộ các người dùng.

## Unix - Giới thiệu

- ❖ Lớp bảo vệ trong Unix được sử dụng để ngăn cách phần nhân với các tiến trình khác.
  - Mỗi tiến trình có không gian địa chỉ riêng xác định các địa chỉ bộ nhớ được phép truy cập.
  - Các hệ thống Unix hiện đại coi các không gian địa chỉ như là các trang nhớ.
  - Unix sử dụng khái niệm file cho tất cả các đối tượng lưu trữ hệ thống như lưu trữ thứ cấp (ổ đĩa), thiết bị vào/ra, mạng và liên lạc giữa các tiến trình.
  - Mỗi tiến trình Unix được liên kết với một định danh căn cứ vào người dùng sử dụng tiến trình đó và việc truy cập tới file bị hạn chế bởi định danh tiến trình.

## Unix - Giới thiệu

- ❖ Mục tiêu an toàn của Unix là bảo vệ người dùng với nhau và bảo vệ các chương trình tin cậy của hệ thống với người dùng.
- ❖ Các chương trình tin cậy trong Unix là các chương trình chạy bằng định danh root hay superuser.
- ❖ Các tiến trình root hay tiến trình nhân có toàn quyền truy cập hệ thống.

## Linux - Giới thiệu

- ❖ Linux có thể coi là biến thể của Unix thu hút được sự chú ý trong thời gian vừa qua từ các thiết bị cỡ nhỏ như điện thoại di động đến các máy chủ hay siêu máy tính.
- ❖ Linux rất giống với bất kỳ hệ thống Unix nào khác.
- ❖ Thực tế, mục tiêu thiết kế quan trọng của Linux là tương thích với Unix.
- ❖ Các tính năng an toàn và cơ chế bảo vệ của Linux cùng sử dụng nguyên tắc như trong Unix.

## Unix / Linux - Cơ chế bảo vệ

- ❖ Hệ thống bảo vệ trong Unix/Linux sử dụng cơ chế kiểm soát truy cập tùy chọn (DAC) - tương tự Windows:
  - Hệ thống bảo vệ mô tả các trạng thái bảo vệ và các thao tác cho phép các tiến trình sửa đổi các trạng thái đó.
  - Điều này có nghĩa là, hệ thống bảo vệ cho phép tiến trình thay đổi giữa các miền bảo vệ (từ mức có đặc quyền sang mức không đặc quyền và ngược lại).
  - Các tính năng bảo vệ này không đủ thỏa mãn các yêu cầu chặt chẽ về hệ điều hành an toàn.

## Unix / Linux - Cơ chế bảo vệ

- ❖ Trạng thái bảo vệ cho biết các thao tác mà chủ thể của hệ thống có thể thực hiện lên các đối tượng.
- ❖ Các trạng thái bảo vệ Unix/Linux liên kết với định danh tiến trình (chủ thể) với các truy cập tới file (đối tượng).
- ❖ Mỗi tiến trình bao gồm định danh người dùng, nhóm, và các nhóm phụ.
- ❖ Tất cả các tài nguyên trong hệ thống được biểu diễn như các file.
  - Các trạng thái bảo vệ mà chủ thể có thể thực hiện là đọc, ghi và thực thi lên file.

## Unix / Linux - Cơ chế bảo vệ

- ❖ Các file cũng được gắn với định danh người dùng và nhóm của chủ sở hữu mà nó cho phép các đặc quyền của tiến trình khi truy cập vào file đó.
- ❖ Tiến trình với định danh chủ sở hữu có thể sửa đổi bất cứ khía cạnh nào của các trạng thái bảo vệ với file đó.
- ❖ Tập hạn chế các đối tượng và thao tác sử dụng danh sách kiểm soát truy cập gọi là bít chế độ.
  - Các bít chế độ xác định các quyền của 3 nhóm chủ thể:
    - File chủ sở hữu UID
    - File trong nhóm GID với chủ sở hữu, và
    - Nhóm còn lại.

## Cơ chế bảo vệ bằng bit chế độ

Name	Owner	Group	Mode Bits
foo	alice	faculty	rwxr--r--
bar	bob	students	rwx-rw-r--
baz	charlie	faculty	rwxrwxrwx

## Cơ chế bảo vệ bằng bit chế độ

### ❖ Với đối tượng “foo”:

- alice là chủ sở hữu và có mọi quyền (rwx : đọc - ghi - thực hiện)
- Nhóm faculty gồm những người dùng cùng nhóm với alice chỉ có quyền đọc (r--)
- Những người dùng khác (ngoài nhóm faculty) chỉ có quyền đọc.

## Unix / Linux - Cơ chế bảo vệ

- ❖ Do hệ thống bảo vệ dựa trên kiểm soát truy cập tùy chọn:
  - Các bít chế độ của người sở hữu, nhóm, và nhóm còn lại có thể bị thay đổi bởi bất cứ tiến trình nào đang chạy sử dụng định danh người dùng.
  - Điều này sẽ gây rủi ro khi chương trình người dùng sử dụng không đáng tin cậy như chương trình tải về từ Internet.
- ❖ Các bít chế độ cũng chứa đựng cách thức dịch chuyển miền bảo vệ được gọi là bit *setuid*.
  - Khi bít này được thiết lập trên 1 file, bất cứ tiến trình nào được thực thi file đó được chuyển một cách tự động thành chủ sở hữu file và nhóm sở hữu.

## Unix / Linux - Hệ thống xác thực

- ❖ Hệ thống xác thực kiểm soát việc truy cập của các tiến trình tới các file và thực hiện việc dịch chuyển miền bảo vệ cho phép người dùng thay đổi định danh.
- ❖ Hệ thống này nằm ở trong nhân song phụ thuộc vào các tiến trình bên ngoài (hệ thống hay người dùng) để xác định các yêu cầu xác thực và trạng thái bảo vệ.

## Unix / Linux - Hệ thống xác thực

❖ Quá trình xác thực diễn ra mỗi khi có yêu cầu truy cập file và thao tác được phép trên file đó sẽ được thẩm tra:

- Tiến trình yêu cầu cung cấp tên file và thao tác cần được thực hiện tại thời điểm lời gọi hệ thống được thực hiện.
- Nếu được chấp thuận, hệ thống tạo ra thẻ mô tả file biểu diễn truy cập được phép của tiến trình để thực hiện các thao tác trong tương lai lên file đó.
- Các thẻ mô tả file được lưu trong phần nhân và chỉ có chỉ mục được trả về cho tiến trình.
- Tiến trình người dùng cung cấp chỉ mục này cho nhân hệ thống mỗi khi cần truy cập file.

## Unix / Linux - Hệ thống xác thực

- ❖ Hệ thống xác thực kiểm soát các thao tác file bằng cách ngăn chặn thao tác mở file cho các quyền đọc, ghi và thực thi.
- ❖ Như vậy, việc sử dụng các quyền này không phải lúc nào cũng cho phép việc kiểm soát phù hợp và không phải đối tượng nào cũng có thể biểu diễn dưới dạng file như việc truy cập tới các dữ liệu meta hay các liên lạc mạng.

## Unix / Linux - Hệ thống xác thực

❖ Hệ thống xác thực dựa vào các dịch vụ xác thực mức người dùng để xác định định danh của tiến trình:

- Khi người dùng đăng nhập vào hệ thống, các tiến trình được gán định danh phù hợp của người dùng.
- Tất cả các tiến trình do người dùng kích hoạt đều thừa kế định danh đăng nhập của người dùng trừ phi có sự thay đổi miền bảo vệ.
- Những dịch vụ mức người dùng này cũng cần đặc quyền *root* để thay đổi định danh tiến trình cũng như thực thi một số công việc của chúng.
- Trong khi đó, phần nhân tin cậy lại phải bao gồm tất cả tiến trình *root*.
- Điều này tạo ra lỗ hổng cho các phần mềm an toàn quan trọng cũng như là nhân.

## Unix / Linux - Hệ thống xác thực

- ❖ Hệ thống có thể cung cấp người dùng đặc biệt *nobody* mà không sở hữu file nào cũng như thuộc về nhóm nào cả.
- ❖ Như vậy, các tiến trình có thể bị hạn chế bởi người dùng đặc biệt này.
- ❖ Tuy nhiên, chủ thể *nobody* vẫn có các quyền khác và các bất cần trong việc sử dụng đặc quyền như lệnh *chroot* đều có thể làm rò rỉ hay phát sinh thêm quyền cho chủ thể *nobody*.

## Unix / Linux - Đánh giá

### ❖ Cơ chế xác thực:

- Không đảm bảo ngăn chặn toàn bộ tới tất cả tài nguyên hệ thống.
- Với một số đối tượng như mạng, không có cơ chế xác thực nào được triển khai.
- Do giao tiếp bộ giám sát tham chiếu được đặt tại nơi mà các thao tác nhạy cảm với an ninh được thực hiện, rất khó có thể biết liệu toàn bộ các thao tác được xác định và toàn bộ các đường dẫn được ngăn chặn.
- Không có cách tiếp cận nào được sử dụng để thẩm tra việc này.

## Unix / Linux - Đánh giá

- ❖ Cho dù các cơ chế bảo vệ và bộ giám sát truy cập được đặt vào phần nhân nhưng điều này không đảm bảo chống lại việc xâm nhập.
  - Cơ chế bảo vệ là tùy chọn nên cơ chế này có thể bị xâm nhập bởi bất kỳ tiến trình nào.
  - Các tiến trình không tin cậy của người dùng có thể sửa đổi quyền truy cập đến dữ liệu của họ một cách tùy ý nên việc bắt tuân thủ các mục tiêu an toàn với dữ liệu người dùng là không thể.

## Unix / Linux - Đánh giá

- ❖ Mặc dù sử dụng lớp bảo vệ, song các nhân Unix thường sử dụng các thủ tục để kiểm tra các tham số của lời gọi hệ thống, những thủ tục như vậy có thể bị đặt không đúng chỗ:
  - Các tiến trình người dùng có nhiều kiểu giao tiếp để truy cập và sửa đổi nhân, hoặc truy cập trực tiếp vào bộ nhớ của nhân.
  - Việc đảm bảo các giao tiếp này chỉ được truy cập bởi các đoạn mã tin cậy là không khả thi.

## Unix / Linux - Đánh giá

❖ Bên cạnh phần nhân, cơ sở tính toán tinh cậy của Unix/Linux bao gồm:

- Toàn bộ các tiến trình chạy với định danh *root* kể cả những tiến trình người dùng được đăng nhập bằng người dùng *root*.
- Như vậy, các tiến trình với định danh *root* có thể chạy bất kỳ chương trình nào nên việc khẳng định chống xâm nhập của cơ sở tính toán tinh cậy là không thể.
- Ngoài ra, phần tính toán tinh cậy quá lớn và đối mặt với quá nhiều mối đe dọa để chống lại việc xâm nhập.
  - Nếu có một trong số các tiến trình bị phá vỡ, hệ thống Unix/Linux rốt cuộc bị phá hủy vì không biện pháp bảo vệ nào hiệu quả với các tiến trình *root*.

## Unix / Linux - Đánh giá chung

- ❖ Các cơ sở cho tính đầy đủ của hệ thống Unix/Linux là không chính xác.
- ❖ Kích cỡ thực sự của cơ sở tính toán tin cậy không cho phép thực hiện bất kỳ việc kiểm nghiệm chính xác hiệu quả.
- ❖ Hơn thế, kích cỡ và khả năng mở rộng của nhân càng làm cho việc kiểm chứng tính đúng đắn không khả thi.

## Unix / Linux - Các vấn đề / lỗ hổng tiêu biểu

- ❖ Rootkit
- ❖ Các biến môi trường
- ❖ Các tài nguyên chia sẻ
- ❖ Vấn đề thời điểm kiểm tra tới thời điểm sử dụng

## Rootkit

- ❖ Hệ thống Unix/Linux hiện đại cho phép mở rộng nhân qua các mô-đun có thể được nạp một cách linh hoạt vào nhân.
- ❖ Tuy nhiên, các mô-đun bị lỗi hay xấu có thể cho phép người tấn công thực thi mã bên trong nhân với đặc quyền đầy đủ của hệ thống.
  - Một loạt các gói phần mềm độc hại, gọi là *rootkit*, đã được tạo ra để tận dụng ưu thế của mô-đun nhân hay các giao tiếp khác tới các tiến trình *root*.
  - Các *rootkit* như vậy cho phép thực thi các hàm của người tấn công và cách thức che dấu việc phát hiện.
  - Cho dù có nhiều cố gắng phát hiện mã độc trong nhân song những *rootkit* này khó phát hiện.

## Các biến môi trường

❖ Các biến môi trường có sẵn cho các tiến trình để chuyển tải trạng thái qua các chương trình khác nhau:

- Như biến LIBPATH cho phép xác định trật tự tìm kiếm thư viện liên kết động.
- Lỗi hỏng phổ biến là người tấn công có thể sửa đổi LIBPATH để nạp các file do mình tạo ra như là một bộ phận của thư viện động.
- Do các biến này được thừa hưởng lại khi tiến trình con sinh ra nên một tiến trình không tin cậy có thể gọi được chương trình tin cậy.
- Nếu tiến trình tin cậy dựa vào biến môi trường LIBPATH để truy cập thư viện, nó có thể bị rủi ro chạy phải các đoạn mã độc.

## Các tài nguyên chia sẻ

- ❖ Tài nguyên chia sẻ giữa tiến trình tin cậy với chương trình không tin cậy có thể tạo ra lỗ hổng để tấn công:
  - Vấn đề thường thấy với thư mục tạm /tmp.
  - Vì mọi tiến trình đều có thể tạo file trong thư mục này, tiến trình không tin cậy có thể tạo file và cấp quyền truy cập cho các chương trình khác, đặc biệt tiến trình tin cậy, truy cập tới các file này.
  - Sau đó, chương trình không tin cậy có thể có quyền truy cập tới các file của các chương trình tin cậy nhờ các đoạn mã nhúng trong file do chương trình tin cậy thực thi.
  - Các tiến trình tin cậy cần phải thận trọng khi sử dụng bất kỳ tài nguyên được chia sẻ bởi các tiến trình không tin cậy.

## Vấn đề thời điểm kiểm tra tới thời điểm sử dụng

- ❖ Thời điểm kiểm tra tới thời điểm sử dụng TOCTTOU (Time-of-Check-to-Time-of-Use) là tình trạng các tiến trình không tin cậy có thể thay đổi trạng thái của hệ thống giữa thời điểm của một thao tác được cấp phép tới thời điểm mà thao tác đó được thực hiện.
- ❖ Nếu việc thay đổi như vậy cho phép tiến trình không tin cậy truy cập tới một file mà nó đáng lẽ không được cho phép thì việc này làm xuất hiện một lỗ hổng.

## Vấn đề thời điểm kiểm tra tới thời điểm sử dụng

### ❖ Ví dụ:

- Tiến trình *root* dùng lời gọi hệ thống *access* để xác định liệu người dùng chạy tiến trình có truy cập tới một file cụ thể như */tmp/X*.
- Tuy nhiên, sau khi lời gọi hệ thống *access* cho phép truy cập file và trước khi file được mở, người dùng có thể thay đổi việc gắn tên file và đối tượng file cụ thể được truy cập (*inode*).
- Việc này được hoàn tất bằng các thay đổi file */tmp/X* thành liên kết tới file đích như là */etc/shadow*.
  - Để khắc phục, Unix thêm một cờ để yêu cầu *open* có thể ngăn chặn việc duyệt nội dung thư mục qua các liên kết.
  - Tuy nhiên, hệ thống file vẫn còn tồn tại nguy cơ bị tấn công kiểu này do việc ánh xạ giữa tên file và đối tượng file (lưu trữ) có thể bị điều chỉnh bởi các tiến trình không an toàn.

## Mô-đun an toàn cho Linux

- ❖ Giới thiệu
- ❖ Triển khai
- ❖ Đánh giá

## Mô-đun an toàn cho Linux - Giới thiệu

- ❖ LSM (Linux Security Modules) - Mô-đun an toàn cho Linux được phát triển vào đầu những năm 2000 nhằm tăng cường an ninh cho các hệ điều hành dựa trên Linux chống lại các dạng mã độc và tấn công mạng;
- ❖ LSM về cơ bản là hệ thống giám sát tham chiếu cho nhân Linux và bao gồm hai phần:
  - Giao tiếp bộ giám sát tham chiếu được tích hợp vào nhân cơ bản của Linux và
  - Mô-đun giám sát tham chiếu thực hiện các chức năng của bộ giám sát như xác thực, kho chính sách phía sau giao tiếp LSM.

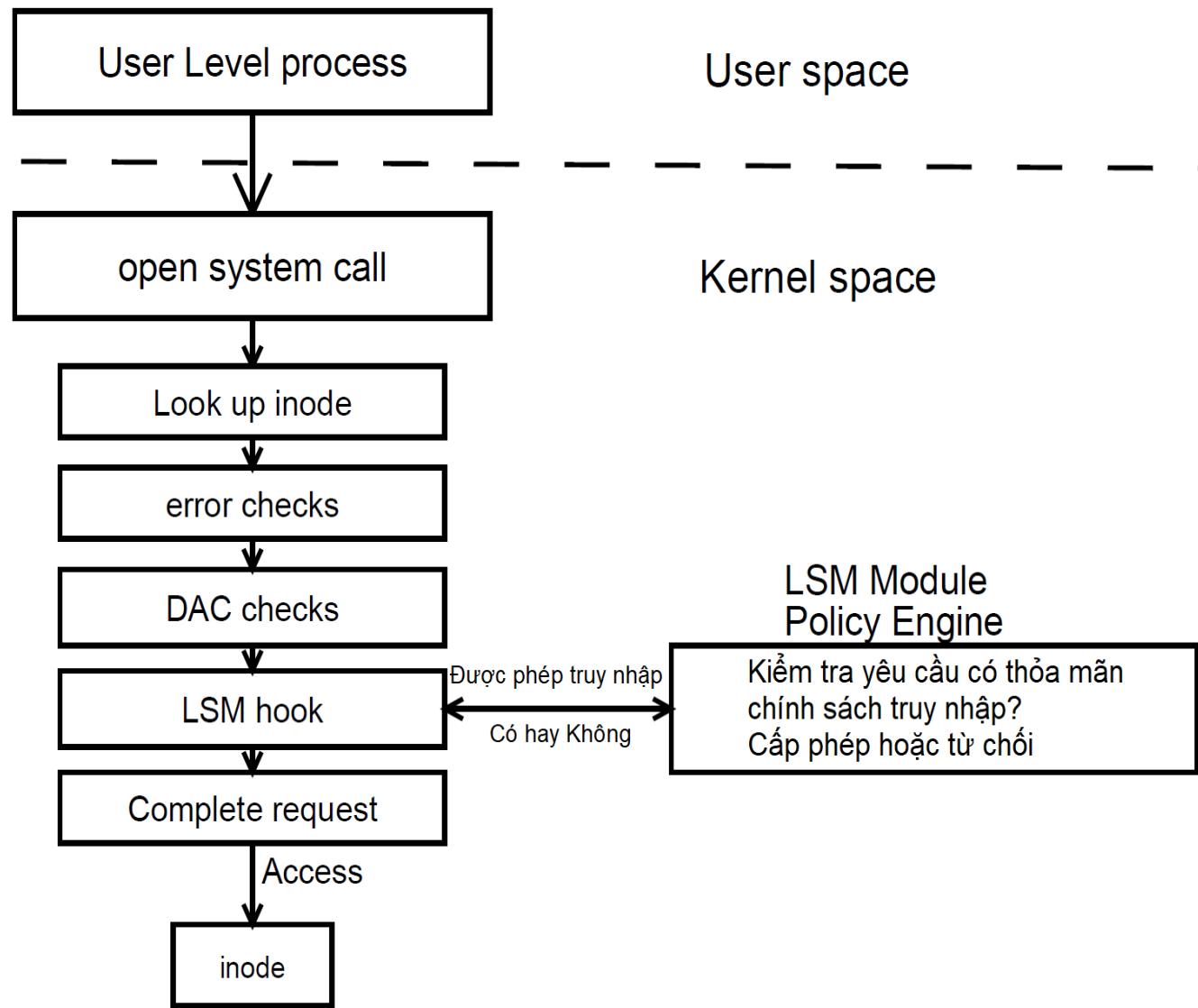
## Mô-đun an toàn cho Linux - Giới thiệu

- ❖ Mục tiêu thiết kế của khung LSM như sau:
  - Giao tiếp với bộ giám sát tham chiếu phải thực sự khái quát để cho việc sử dụng các mô hình an toàn khác nhau chỉ thuận túy là việc nạp các nhân khác nhau;
  - Giao tiếp cần phải đơn giản về khái niệm, tối giản và hiệu quả;
  - Cần hỗ trợ chuẩn POSIX.1e về kiểm soát truy cập dựa trên năng lực.
- ❖ LSM được chính thức thêm vào nhân Linux từ phiên bản 2.6 và biến đổi Linux cũng như Unix phần nào thành hệ thống thỏa mãn các yêu cầu của bộ giám sát tham chiếu.

## Mô-đun an toàn cho Linux - Triển khai

- ❖ Việc triển khai bộ khung LSM gồm ba phần:
  - Định nghĩa giao tiếp bộ giám sát tham chiếu;
  - Việc bố trí giao tiếp này; và
  - Việc triển khai cụ thể của bộ giám sát tham chiếu.

## Kiến trúc giao tiếp LSM



## Mô-đun an toàn cho Linux - Triển khai

- ❖ Định nghĩa giao tiếp LSM mô tả cách thức nhân Linux có thể gọi bộ giám sát tham chiếu LSM.
- ❖ Đây là tập con trỏ tới các hàm (function pointer) dùng để gọi các hàm khi LSM được nạp.
- ❖ Các con trỏ hàm được gọi các móc LSM (LSM hook).
  - Về cơ bản các móc LSM tương ứng với các truy vấn cấp phép, tuy nhiên giao tiếp LSM cũng chứa các móc cho các nhiệm vụ khác như gán nhãn an ninh, dịch chuyển hay duy trì các nhãn an ninh.
  - Có khoảng hơn 150 các móc LSM dùng cho các công việc an ninh.

## Mô-đun an toàn cho Linux - Triển khai

### ❖ Bố trí giao tiếp của bộ tham chiếu LSM:

- Thách thức chủ yếu trong thiết kế LSM là vị trí đặt các mốc LSM.
- Hầu hết các mốc LSM liên kết với lời gọi hệ thống cụ thể, như vậy với các mốc LSM này được đặt tại đầu vào của lời gọi hệ thống.
  - Tuy nhiên, một số mốc LSM không thể đặt tại vị trí như vậy. Ví dụ như trong hình về kiến trúc giao tiếp LSM, lời gọi hệ thống chuyển đường dẫn của file thành thẻ mô tả file mà cho phép truy cập tới file đó.
- Xác định vị trí file cụ thể được mô tả bởi đường dẫn cần cho phép truy cập tới các thư mục theo đường dẫn, bất kỳ các liên kết file, và cuối cùng là cho phép truy cập vào file với thao tác cụ thể.
- Do các thành phần được trích ra từ đường dẫn tạo các điểm khác nhau trong quá trình mở file *open*, vì vậy vị trí đặt các mốc LSM không đơn giản.

## Mô-đun an toàn cho Linux - Triển khai

- ❖ Trong khi có một số phép kiểm tra tùy chọn giúp cho việc xác định vị trí đặt các móc LSM cho các thao tác *open*, quá trình đặt các móc LSM nói chung mang tính tình thế.
  - Với những chương trình chưa từng thực hiện việc cấp phép truy cập tùy chọn thì phải thực hiện việc đặt các móc LSM thủ công.
  - Móc LSM được chèn vào đoạn mã theo kiểu trực tiếp (inline) và được liên kết tại thời điểm biên dịch. Việc này giúp cho các đoạn mã dễ theo dõi hơn.

## Mô-đun an toàn cho Linux - Triển khai

- ❖ Triển khai bộ giám sát tham chiếu: LSM bao gồm:
  - AppArmor (Application Armor);
  - Hệ thống phát hiện xâm nhập IDS (Intrusion detection system);
  - SELinux (Security-Enhanced Linux) và
  - Danh sách năng lực POSIX.
- ❖ Mỗi một mô-đun LSM cung cấp cách thức khác nhau với việc kiểm soát truy cập bắt buộc ngoại trừ danh sách năng lực POSIX là cơ chế kiểm soát tùy chọn đã có trong Linux.
  - Việc chuyển POSIX thành mô-đun nhằm cho phép phát triển độc lập với nhân cơ bản và vì một số mô-đun LSM triển khai việc kiểm soát năng lực theo cách khác.

## Mô-đun an toàn cho Linux - Triển khai

- ❖ AppArmor là hệ thống kiểm soát truy cập bắt buộc mà mô hình đe dọa tập trung vào môi trường Internet.
- ❖ Nếu hệ thống được cấu hình một cách đúng đắn thì Internet là cách thức duy nhất mà các đầu vào có mục đích xấu có thể tiếp cận đến hệ thống.
  - Một mối đe dọa chính đến từ các dịch vụ mạng, do chúng có nguy cơ nhận các dữ liệu đầu vào không đảm bảo như dữ liệu tràn bộ đệm.
  - AppArmor sử dụng các chính sách hạn chế cho các dịch vụ mạng như vậy để ngăn chặn các dịch vụ bị xâm hại làm vô hiệu cả hệ thống.

## Mô-đun an toàn cho Linux - Triển khai

- ❖ Hệ thống phát hiện xâm nhập IDS hướng tới việc ngăn chặn xâm nhập dưới dạng hệ thống kiểm soát truy cập.
- ❖ Hệ thống này thực hiện việc quản lý truy cập dựa vào các chính sách mô tả file nào mà chương trình có thể được truy cập.

## Mô-đun an toàn cho Linux - Triển khai

- ❖ SELinux triển khai kiểm soát truy cập bắt buộc dựa trên kiến trúc kiểm soát truy cập linh hoạt Flask bao gồm *kho chính sách* và *phần nhân* cho việc giám sát truy cập.
- ❖ SELinux hỗ trợ mô hình an toàn dựa trên vai trò RBAC (Role-Based Access Control) hay an toàn nhiều mức.
  - SELinux có thể hạn chế các chương trình người dùng theo nguyên tắc ít đặc quyền tối thiểu, bảo vệ tính toàn vẹn, hay tính bí mật của chương trình và dữ liệu, cũng như các nhu cầu an ninh của chương trình.
- ❖ Tính khái quát và đầy đủ của SELinux góp phần thúc đẩy các yêu cầu đối với LSM.

## Mô-đun an toàn cho Linux - Triển khai

- ❖ Danh sách năng lực POSIX được triển khai trong nhân cơ bản của Linux, song LSM tách biệt rạch rời chức năng này vào trong mô-đun an ninh.
- ❖ Việc này cho phép người dùng không cần chức năng này có thể loại bỏ nó khỏi nhân và cho phép việc phát triển kiểm soát theo năng lực độc lập với phần nhân cơ bản.

## Mô-đun an toàn cho Linux - Đánh giá

### ❖ Về tính ngăn chặn đầy đủ:

- Giao tiếp với bộ giám sát tham chiếu của khung LSM được thiết kế để cấp phép truy cập tới đối tượng (tài nguyên) cụ thể được dùng bởi nhân trong các thao tác nhạy cảm với an ninh để ngăn cản các lỗ hổng.
- Ngoài ra, khung LSM đứng ngăn chặn các thao tác được xác định bởi cộng đồng phát triển LSM tới các thao tác nhạy cảm.
- Cơ chế trung gian (ngăn chặn) thực tế là kết hợp toàn bộ các mẫu giám sát tham chiếu được xây dựng.

## Mô-đun an toàn cho Linux - Đánh giá

- ❖ Do các giao tiếp LSM được thiết kế theo kiểu không chính tắc, việc kiểm chứng với việc ngăn chặn đầy đủ là cần thiết:
  - Các công cụ phân tích mã nguồn được xây dựng để kiểm chứng các cấu trúc dữ liệu nhân nhạy cảm với an ninh được ngăn chặn theo cách nhất quán: Các lỗi được phát hiện được sửa.
  - Tuy nhiên, các công cụ này chỉ là xấp xỉ gần đúng các yêu cầu về ngăn chặn đầy đủ và chúng không được sử dụng thường xuyên.
  - Tuy nhiên, không có lỗi nào trong phần bố trí giao tiếp của bộ giám sát truy cập được phát hiện thêm kể từ khi được triển khai.

## Mô-đun an toàn cho Linux - Đánh giá

- ❖ Nói chung, việc kiểm chứng tính đúng đắn của các biện pháp thực thi an ninh là việc khó khăn nhất.
- ❖ Với mã nguồn lớn và được viết bằng ngôn ngữ không an toàn và bởi nhiều người, việc kiểm chứng là không thể hoàn thành trên thực tế.
- ❖ Linux đã được mức đánh giá theo tiêu chuẩn phổ quát EAL4.
- ❖ Mức này đòi hỏi việc lập tài liệu thiết kế mức thấp của nhân.
- ❖ Dù vậy việc kiểm chứng các đoạn mã tuân thủ các mô hình thiết kế có lẽ là việc không thực tế.

## Mô-đun an toàn cho Linux - Đánh giá

- ❖ Các chính sách SELinux định nghĩa các yêu cầu bắt buộc và chính xác các thao tác được phép trong hệ thống:
  - Hoàn toàn có thể xây dựng luồng thông tin từ các chính sách này và thậm chí cả các trạng thái dịch chuyển.
  - Các chính sách an ninh nhiều mức đảm bảo các luồng thông tin thỏa mãn các thuộc tính an toàn cơ bản và nâng cao.
  - Cách tiếp cận của SELinux cho phép hệ thống được an toàn, song người phát triển hệ thống cần phải quản lý việc sử dụng đoạn mã tin cậy một cách cẩn thận để chắc chắn các mục tiêu an toàn thực sự đạt được.

## Nhân SCOMP

- ❖ Giới thiệu
- ❖ Phần cứng
- ❖ Nền tảng tính toán an toàn
- ❖ Đánh giá

## Nhân SCOMP - Giới thiệu

- ❖ SCOMP (Secure communication Processor) là hệ thống dựa trên nhân an toàn được thiết kế để triển khai các yêu cầu về an ninh theo nhiều mức.
- ❖ Với yêu cầu về hiệu năng và vấn đề an toàn, người thiết kế SCOMP không chỉ xây dựng nhân an toàn mà còn xây dựng các cơ chế phần cứng và giao tiếp ứng dụng mới để lập trình với nhân.

## Nhân SCOMP - Giới thiệu

- ❖ Phần tính toán tin cậy của SCOMP bao gồm 3 bộ phận hoạt động ở lớp 0, 1 và 2:
  - Hệ điều hành tin cậy bao gồm nhân an toàn hoạt động ở lớp 0 và các phần mềm tin cậy ở lớp 1.
  - Các chức năng của gói giao tiếp nhân hoạt động ở lớp 2.
- ❖ Các ứng dụng truy cập các tài nguyên được bảo vệ bởi nền tảng tính toán tin cậy sử dụng bộ thư viện giao tiếp hoạt động trong không gian địa chỉ người dùng.

## Nhân SCOMP - Giới thiệu

Ứng dụng	Gói giao tiếp nhân SCOMP (Thư viện)
Nền tảng tính toán tin cậy SCOMP	Gói giao tiếp nhân SCOMP (Các hàm tin cậy)
	Hệ điều hành tin cậy SCOMP (Phần mềm tin cậy SCOMP)
	Hệ điều hành tin cậy SCOMP (Nhân an toàn)
	Phần cứng SCOMP

Lớp 3 (không tin cậy)

Lớp 2 (tin cậy)

Lớp 1 (tin cậy)

Lớp 0 (tin cậy)

## Nhân SCOMP - Giới thiệu

- ❖ Nhân an toàn của SCOMP ngăn chặn toàn bộ các truy cập tới các tài nguyên sử dụng chính sách an toàn nhiều lớp.
- ❖ Khi ứng dụng cần sử dụng các tài nguyên được bảo vệ như bộ nhớ hay thiết bị vào/ra nó cần phải yêu cầu nhân an toàn cấp các thẻ mô tả phần cứng (hardware descriptor) phù hợp để sử dụng tài nguyên đó.

## Nhân SCOMP - Giới thiệu

- ❖ Khi nhân an toàn chấp thuận yêu cầu của ứng dụng, nhân sẽ tạo ra thẻ mô tả phần cứng cho thao tác truy cập này và lưu thẻ trong bộ nhớ;
  - Tham chiếu tới thẻ mô tả phần cứng được trả lại cho ứng dụng cho các lần sử dụng sau.
  - Thẻ mô tả phần cứng bao gồm tham chiếu tới đối tượng và các quyền truy cập cho tiến trình.

## Nhân SCOMP - Giới thiệu

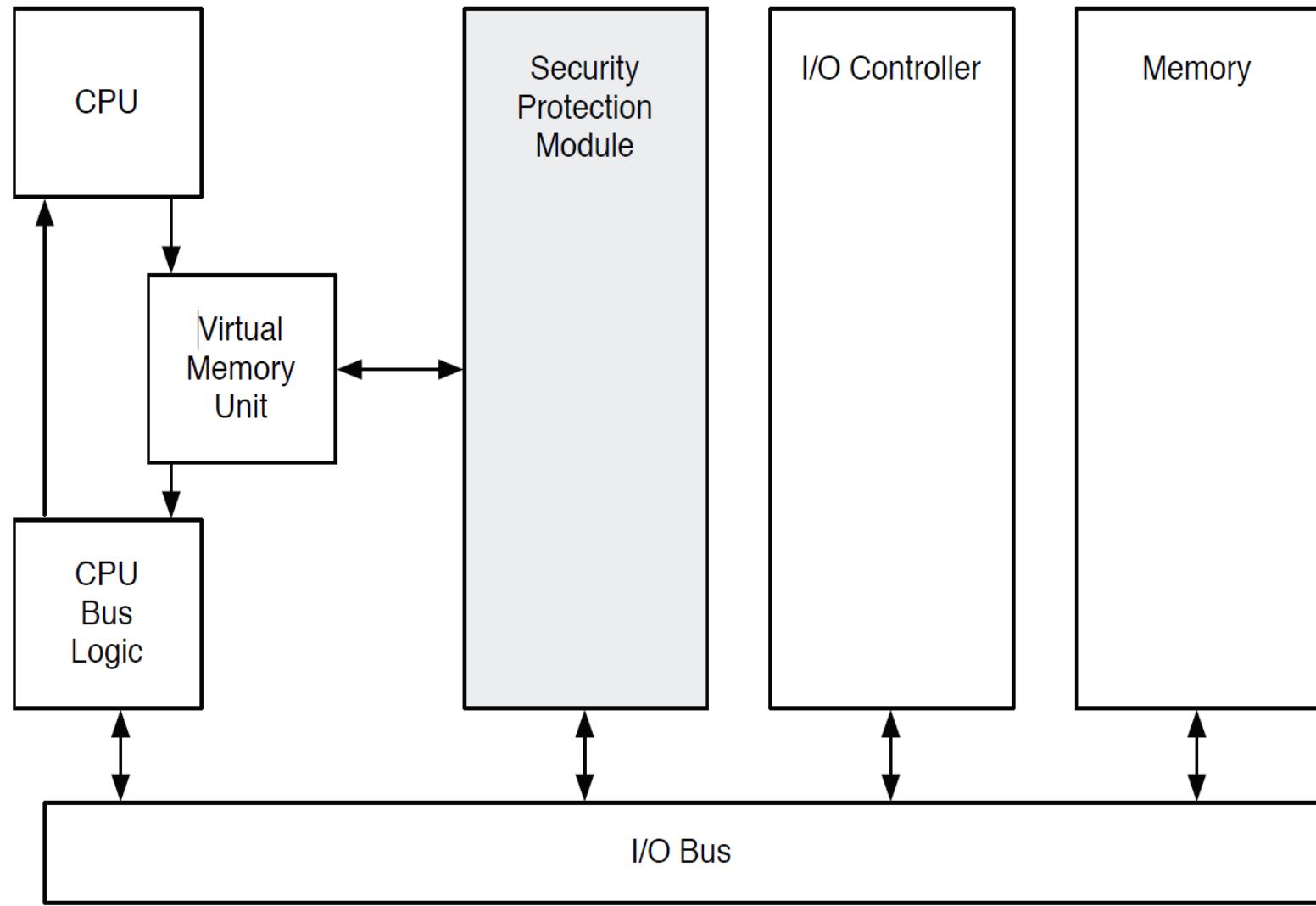
- ❖ Cách ly, chống xâm nhập được triển khai sử dụng cơ chế bảo vệ theo lớp:
  - Cơ chế lớp truy cập kiểm soát việc đoạn mã ở lớp này được chấp thuận để yêu cầu dịch vụ từ lớp khác.
  - Các lớp và dịch chuyển lớp được thực thi nhờ phần cứng.
- ❖ Trong quá trình xây dựng HĐH, việc kiểm chứng mô hình an toàn SCOMP và việc triển khai các chính sách an toàn được đặt ra hàng đầu cũng như các phần mềm tin cậy của hệ thống đều được thẩm tra.

## Nhân SCOMP - Phần cứng

### ❖ SCOMP sử dụng cơ chế bảo vệ gồm 4 lớp:

- Nhân an toàn chạy ở lớp có đặc quyền cao nhất lớp 0 và ứng dụng người dùng ở lớp có đặc quyền ít nhất.
- Bên cạnh đó, SCOMP sử dụng cơ chế gọi hàm bằng cách sử dụng địa chỉ các tham số bên gọi cung cấp, việc này cho phép ngăn cản các đoạn mã truy cập ra ngoài không gian nhớ này.

## Nhân SCOMP - Cấu trúc phần cứng



## Nhân SCOMP - Phần cứng

- ❖ Mô-đun bảo vệ an toàn SPM (Secure protection module) chịu trách nhiệm ngăn chặn toàn bộ các truy cập bộ nhớ và vào/ra.
  - Mỗi chương trình có địa chỉ cơ sở gốc trả tới bộ nhớ và các thẻ vào/ra.
  - Mỗi truy cập bộ nhớ đều phải đi qua SPM và phải qua phép kiểm tra truy cập.
- ❖ Do phần nhân cung cấp các thẻ và việc kiểm tra được thực hiện bằng phần cứng nên có thể cho phép chạy các câu lệnh vào/ra ở mức không cần đặc quyền.

## Nhân SCOMP - Nền tảng tính toán an toàn

- ❖ Nhân an toàn hoạt động ở lớp 0 cung cấp các chức năng cơ bản của hệ thống như quản lý bộ nhớ, điều độ tiến trình, quản lý ngắt, kiểm toán và giám sát tham chiếu.
- ❖ Các chức năng của nhân được giữ tối thiểu và chỉ chứa khoảng 10K dòng lệnh.
- ❖ Các đối tượng trong nhân gồm có các tiến trình, các đoạn nhớ và các thiết bị được định danh bằng các thẻ 64bit.
- ❖ Nhân duy trì các thông tin kiểm soát truy cập và trạng thái cho mỗi đối tượng.

## Nhân SCOMP - Nền tảng tính toán an toàn

- ❖ Mô hình kiểm soát truy cập sử dụng mô hình Bell-La Padula mở rộng với phần phân loại, cách chính sách lớp bảo vệ và chính sách tùy chọn.
- ❖ Nhân an toàn định nghĩa 38 cổng để các tiến trình nằm ngoài nhân có thể gọi các dịch vụ nhân.
- ❖ Cổng tương tự như lời gọi hệ thống cung cấp chức năng tạo đối tượng, ánh xạ các đoạn nhớ và gắn bộ nhớ vật lý với bộ nhớ ảo.

## Nhân SCOMP - Nền tảng tính toán an toàn

- ❖ Phần mềm tin cậy chạy các dịch vụ không cần mức 0 nhưng cung cấp các chức năng cần sự tin cậy để thực thi việc kiểm soát các ứng dụng người dùng một cách thích đáng.
- ❖ Có hai loại phần mềm tin cậy:
  - Loại thứ nhất được tin cậy không vi phạm tính bí mật và toàn vẹn của hệ thống như phần mềm nạp tiến trình.
  - Loại thứ 2 được tin cậy để duy trì các chính sách an toàn một cách đúng đắn như các dịch vụ sửa đổi dữ liệu xác thực người dùng. Phần này có 23 tiến trình chứa 11k dòng lệnh viết bằng ngôn ngữ C.

## Nhân SCOMP - Nền tảng tính toán an toàn

- ❖ Các phần mềm tin cậy được gọi thông qua các kênh thông tin tin cậy từ người dùng.
- ❖ Việc sử dụng các kênh này ngăn chặn các phần mềm xấu giả mạo người dùng hợp lệ.
- ❖ Người dùng biết việc tương tác trực tiếp với phần mềm an toàn khi kênh an toàn được kích hoạt.
- ❖ Chỉ nhân có thể nhận được tín hiệu ngắn, như vậy người dùng có thể chắc chắn các đáp ứng bắt nguồn từ phần mềm tin cậy.

## Nhân SCOMP - Đánh giá

- ❖ SCOMP thực hiện việc ngăn chặn ở mức phần cứng, như vậy đảm bảo bộ giám sát truy cập kiểm soát được tất cả các yêu cầu của hệ thống tới các tài nguyên của hệ thống.
- ❖ Các tài nguyên của hệ thống như các đoạn bộ nhớ, bộ nhớ và vào/ra và tất cả các lệnh truy cập vào các đoạn, phần ngăn chặn dựa trên phần cứng bẫy toàn bộ các truy cập nhạy cảm.

## Nhân SCOMP - Đánh giá

- ❖ Hệ thống file SCOMP ở lớp 2 kiểm soát việc truy cập tới các file mức cao, nơi các yêu cầu (chính sách truy cập) được soạn thảo.
- ❖ Tuy nhiên, các truy cập ban đầu tới file dữ liệu tùy thuộc vào truy cập tới thiết bị vào/ra.
- ❖ Hệ thống file phải được tin cậy để ngăn cản các truy cập trái phép tới dữ liệu của người dùng bởi chương trình khác.
- ❖ Mặt khác, cơ chế ngăn chặn của SCOMP được kiểm tra và đánh giá dựa trên phần cứng.

## Nhân SCOMP - Đánh giá

- ❖ SCOMP áp dụng lớp bảo vệ để bảo vệ nhân an ninh khỏi các sửa đổi trái phép:
  - Nhân an ninh hoạt động ở lớp 0 và chỉ có 38 cổng cho phép truy cập tới nhân từ các lớp bảo vệ khác.
  - SCOMP sử dụng mô hình kiểm soát toàn vẹn phức tạp để biểu diễn truy cập tới lớp 0 (lớp có nhiều đặc quyền nhất).
- ❖ Do nhân có thể cần phải cập nhật như hệ thống file, có một số các đối tượng (tài nguyên) và tiến trình có thể sửa đổi nhân bao gồm cả cơ chế bảo vệ và bộ giám sát tham chiếu.

## Nhân SCOMP - Đánh giá

- ❖ SCOMP cũng sử dụng lớp bảo vệ và mô hình truy cập để bảo vệ tính toàn vẹn của các phần còn lại trong cơ sở tính toán tin cậy.
- ❖ Cơ sở tính toán tin cậy hoạt động ở lớp 0,1, và 2.
  - Các mã không tin cậy không được hoạt động ở những lớp này.
  - Các tiến trình không tin cậy ở lớp 3 có thể gọi các đoạn mã trong phần tính toán tin cậy.
  - Khi đó các giao tiếp và các cổng được triển khai để bảo vệ cơ sở tính toán tin cậy không rõ ràng.

## Nhân SCOMP - Đánh giá chung

- ❖ Tính đúng đắn của phần cơ sở tính toán tin cậy giữa phần thiết kế và triển khai được kiểm chứng bằng công cụ phân tích chính tắc.
- ❖ Thêm vào đó các mục tiêu an toàn được tuân thủ một cách bắt buộc sử dụng chính sách an ninh nhiều lớp kiểu bắt buộc.
- ❖ Các hệ thống phát triển kế tiếp được kiểm chứng về tính đúng đắn của thiết kế các chính sách hệ thống.

## Nhân SCOMP - Đánh giá chung

- ❖ SCOMP đáp ứng một cách thuyết phục nhất cho câu hỏi về tính an toàn cũng như tính đúng đắn của các biện pháp đảm bảo an ninh và an toàn cho hệ thống.
- ❖ Mặc dù vẫn còn một số nhỏ các điểm nguy hiểm như độ phức tạp trong giao tiếp với phần tính toán tin cậy cũng như tính không đầy đủ trong việc kiểm chứng hệ thống, SCOMP và các nhân an toàn khác tiến gần nhất đến hệ điều hành an toàn.