HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

# CƠ SỞ AN TOÀN THÔNG TIN

## NỘI DUNG 2 – LỖ HỔNG BẢO MẬT VÀ ĐIỂM YẾU HỆ THỐNG

**Giảng viên:** TS. Đinh Trường Duy

**Điện thoại/E-mail:** duydt@ptit.edu.vn

**Khoa:** An toàn thông tin

*Nội dung bài giảng dựa trên bài giảng và giáo trình Cơ sở an toàn thông tin của PGS.TS. Hoàng Xuân Dậu*

# NỘI DUNG 2

1. Tổng quan về lỗ hổng bảo mật và các điểm yếu hệ thống
2. Các dạng lỗ hổng trong hệ điều hành và phần mềm ứng dụng
3. Quản lý, khắc phục các lỗ hổng bảo mật và tăng cường khả năng đề kháng cho hệ thống
4. Giới thiệu một số công cụ rà quét lỗ hổng bảo mật

## 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

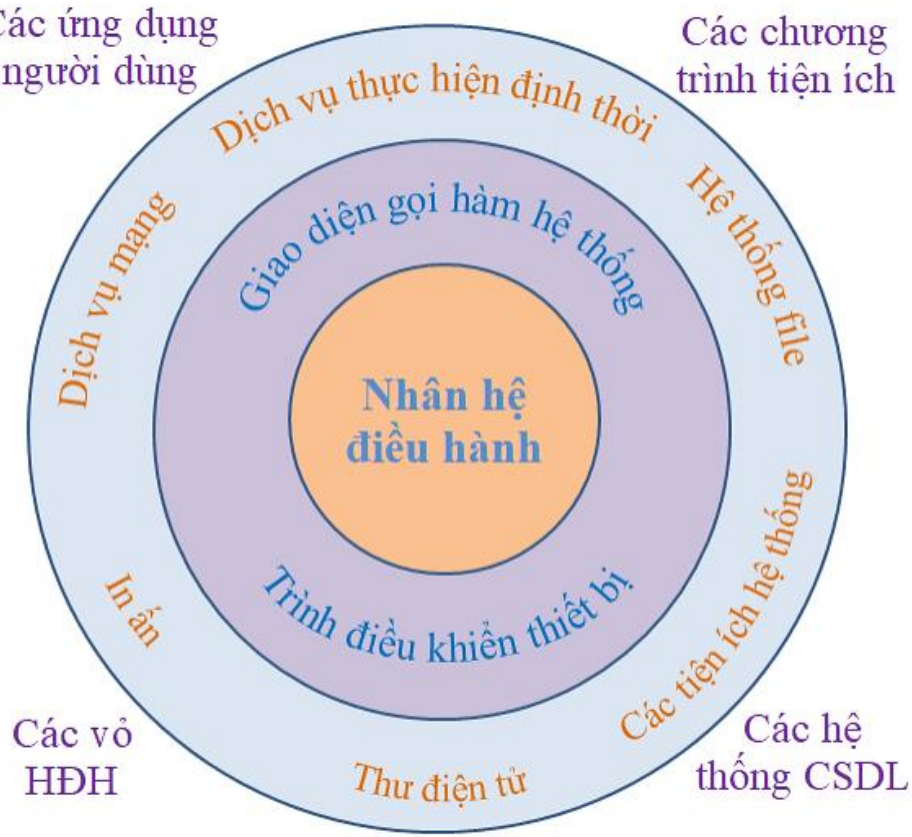
- Các thành phần của hệ thống máy tính
- Khái niệm điểm yếu hệ thống và các lỗ hổng bảo mật
- Phân bố các lỗ hổng bảo mật:
  - Phần cứng / phần mềm
  - Các hệ điều hành phổ biến
  - Các ứng dụng phổ biến

# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

- Các thành phần của hệ thống máy tính:
  - Hệ thống phần cứng
    - CPU, ROM, RAM, Bus,...
    - Các giao diện ghép nối và các thiết bị ngoại vi.
  - Hệ thống phần mềm
    - Hệ điều hành
      - Nhân hệ điều hành, các trình điều khiển thiết bị
      - Các trình cung cấp dịch vụ, tiện ích,…
    - Các phần mềm ứng dụng
      - Các dịch vụ (máy chủ web, CSDL, DNS,...)
      - Trình duyệt web, các ứng dụng giao tiếp,…
      - Các bộ ứng dụng văn phòng, lập trình.

# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

- Mô hình hệ điều hành Unix/Linux, các dịch vụ và các ứng dụng



## 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

- Các điểm yếu hệ thống (system weaknesses) là các lỗi hay các khiếm khuyết (thiết kế, cài đặt, phần cứng hoặc phần mềm) tồn tại trong hệ thống.
  - Có điểm yếu đã biết và đã được khắc phục;
  - Có điểm yếu đã biết và chưa được khắc phục;
  - Có điểm yếu chưa biết/chưa được phát hiện

## 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

- Lỗ hổng bảo mật (Security vulnerability) là một điểm yếu trong một hệ thống cho phép kẻ tấn công khai thác gây tổn hại đến các thuộc tính an ninh, an toàn của hệ thống đó:
  - Toàn vẹn (integrity)
  - Bí mật (confidentiality)
  - Sẵn dùng (availability).

## 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

- Toàn vẹn (integrity):
  - Mọi sửa đổi đến thông tin/hệ thống chỉ được thực hiện bởi các bên có đủ thẩm quyền;
  - Kẻ tấn công có thể lợi dụng điểm yếu an ninh để lặng lẽ sửa đổi thông tin/hệ thống → phá vỡ tính toàn vẹn;
  - Ví dụ:
    - Thông thường trong hệ thống kiểm soát truy nhập, chỉ người quản trị có quyền thay đổi quyền truy nhập đến mọi file;
    - Một điểm yếu trong hệ thống có thể cho phép một người dùng bình thường thay đổi quyền truy nhập đến mọi file tương tự người quản trị.

# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

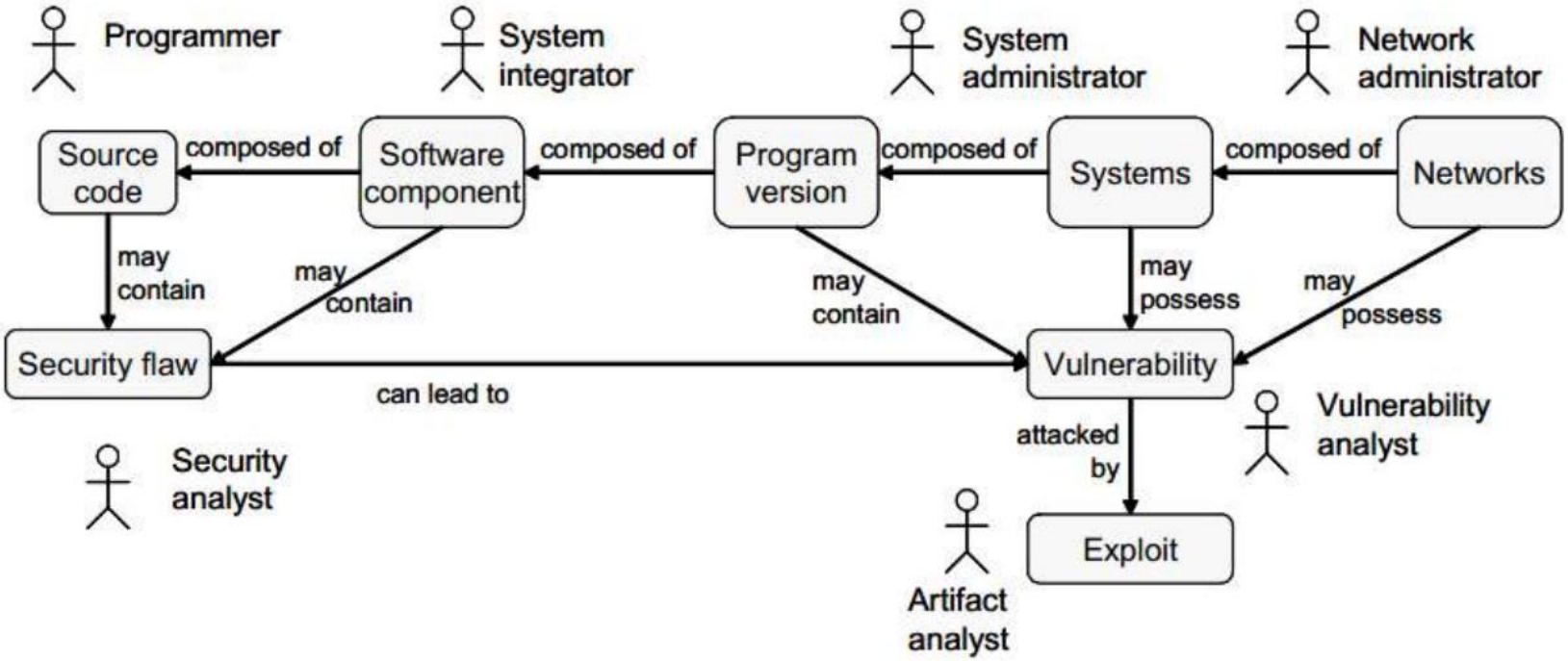
- Bí mật (confidentiality):
  - Chỉ những người có thẩm được phép truy nhập đến thông tin/hệ thống;
  - Kẻ tấn công có thể lợi dụng điểm yếu an ninh để truy nhập trái phép → phá vỡ tính bí mật;
  - Ví dụ:
    - Một điểm yếu an ninh cho phép người dùng web thông thường đọc được nội dung một file mà lẽ ra người đó không được quyền đọc;
    - Một điểm yếu trong hệ thống kiểm soát truy nhập cho phép một nhân viên bình thường đọc được các báo cáo “mật” của công ty mà chỉ Ban Giám đốc được phép đọc

# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

- Sẵn dùng (availability):
  - Đảm bảo khả năng truy nhập đến thông tin/hệ thống cho người dụng hợp pháp;
  - Kẻ tấn công có thể lợi dụng điểm yếu an ninh để ngăn chặn hoặc gây khó khăn cho người dụng hợp pháp truy nhập vào thông tin/hệ thống;
  - Ví dụ:
    - Một điểm yếu an ninh có thể cho phép kẻ tấn công làm máy chủ ngừng hoạt động → không thể cung cấp dịch vụ cho người dùng hợp pháp →phá vỡ tính sẵn dùng;
    - Kẻ tấn công cũng có thể gửi một lượng lớn yêu cầu giả mạo đến máy chủ gây cạn kiệt tài nguyên hoặc tắc nghẽn đường truyền → người dùng hợp pháp không thể truy cập → phá vỡ tính sẵn dùng

# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

Mô hình các quan hệ giữa các đối tượng và vai trò trong hệ thống



# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

- Source code: mã nguồn
- Software component: thành phần phần mềm
- Program version: phiên bản chương trình
- Systems: các hệ thống
- Networks: các mạng
- Security flaw: khiếm khuyết an ninh
- Vulnerability: lỗ hổng an ninh
- Exploit: khai thác lỗ hổng an ninh
- Programmer: lập trình viên
- System integrator: nhân viên tích hợp hệ thống
- System administrator: nhân viên quản trị hệ thống
- Network administrator: nhân viên quản trị mạng
- Security analyst: nhân viên phân tích an ninh
- Vulnerability analyst: nhân viên phân tích lỗ hổng an ninh
- Artifact analyst: nhân viên phân tích hiện vật.

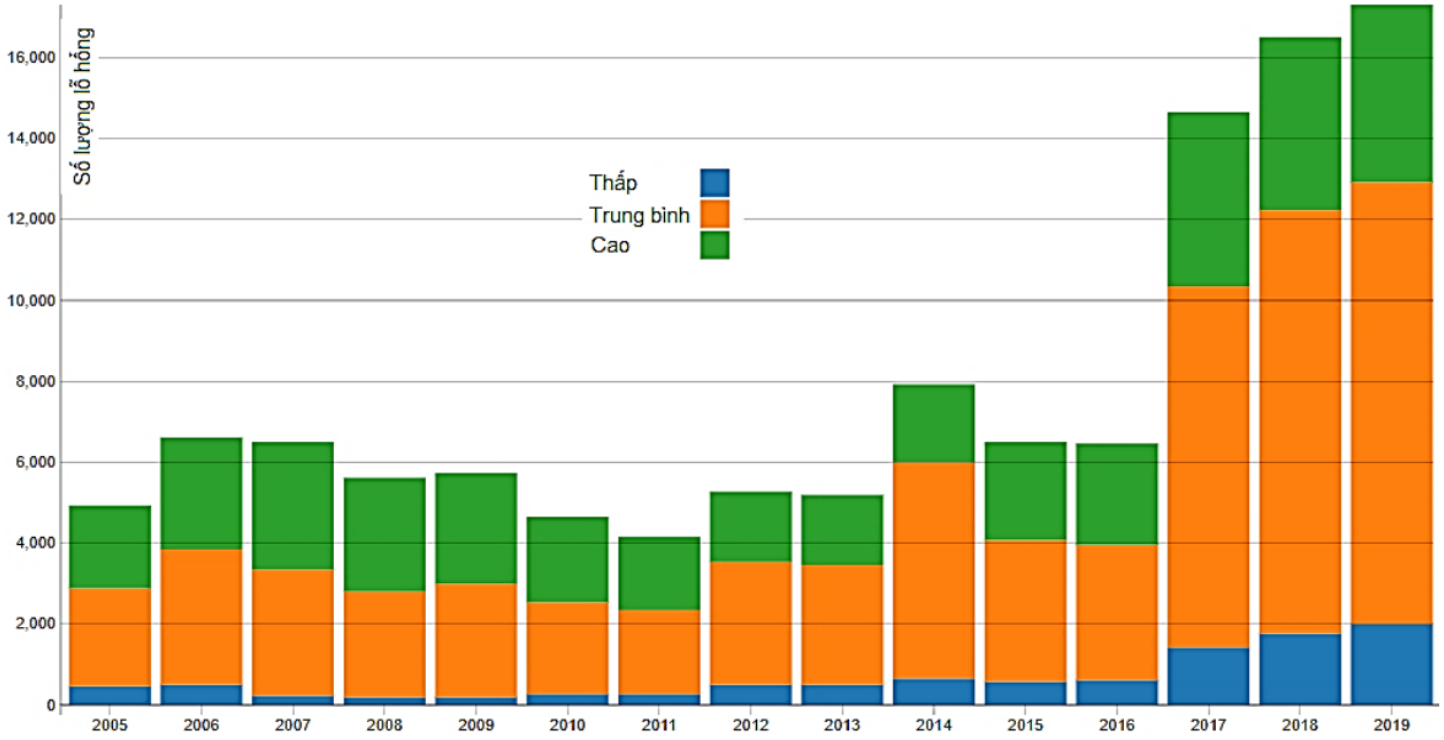
## 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

❖ Mức độ nghiêm trọng của lỗ hổng bảo mật:

- 4 mức độ nghiêm trọng theo Microsoft:
  - Nguy hiểm (Critical)
  - Quan trọng (Important)
  - Trung bình (Moderate)
  - Thấp (Low).
- 3 mức độ nghiêm trọng theo một số tổ chức khác:
  - Cao (High)
  - Trung bình (Medium)
  - Thấp (Low)

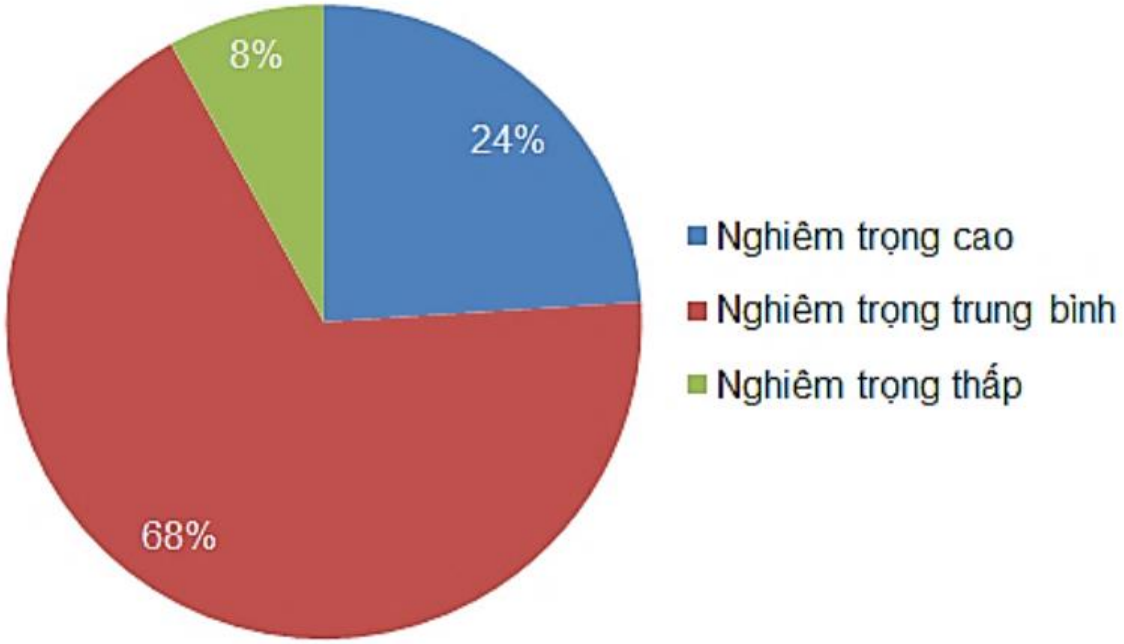
# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

Số lượng các lỗ hổng bảo mật được phát hiện trong giai đoạn 2005-2019 (US National Vulnerability Database)



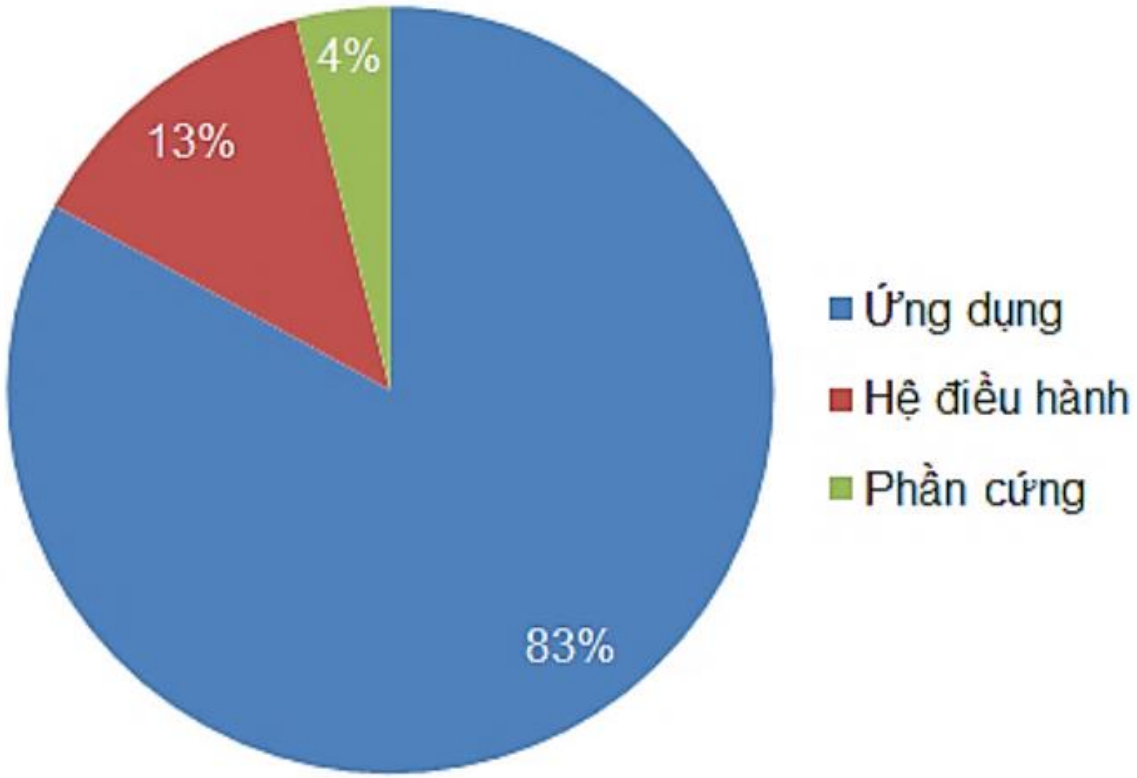
# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

Phân bố các lỗ hổng bảo mật phát hiện trong năm 2014 theo mức độ nghiêm trọng



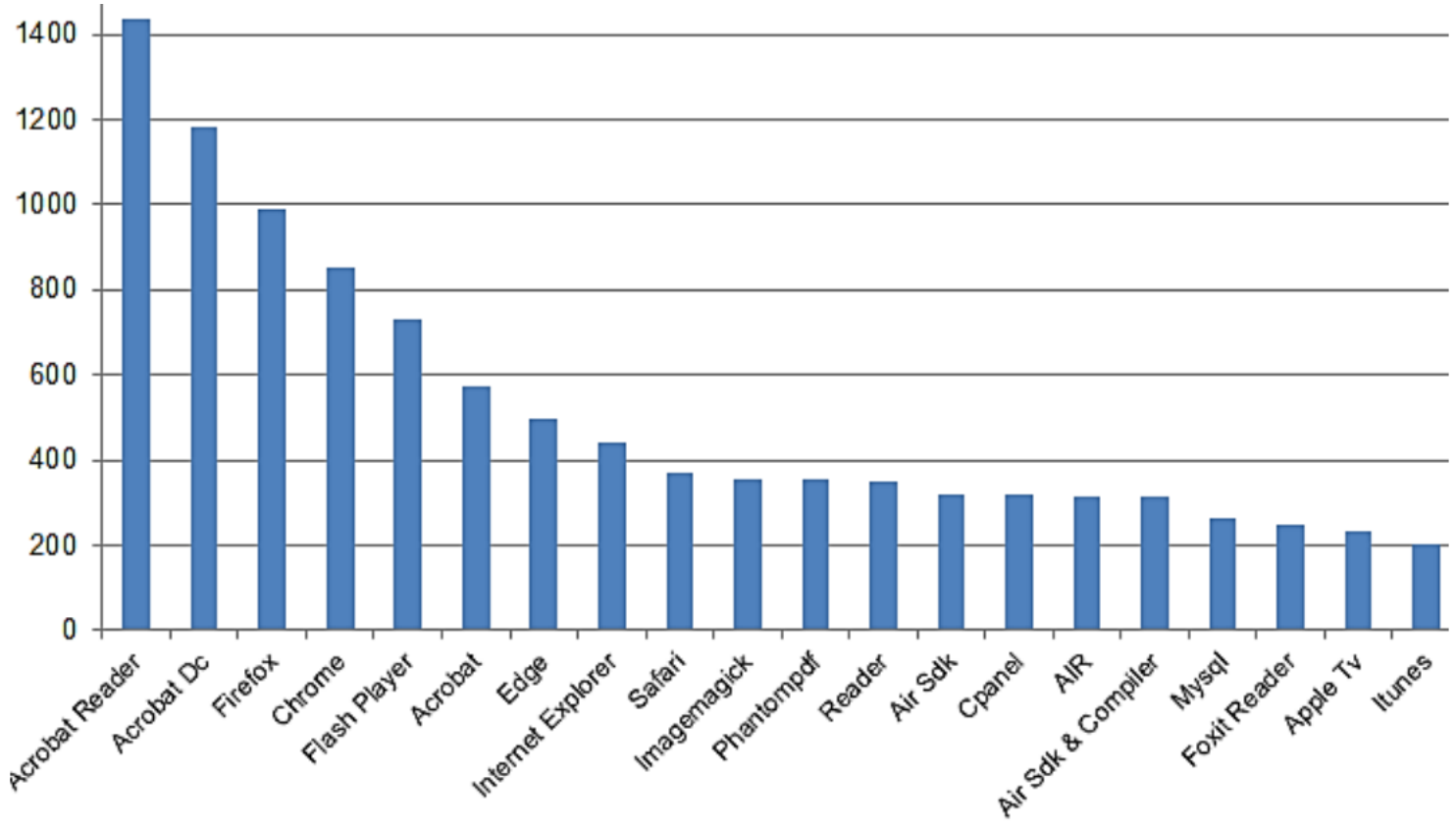
# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

Phân bố lỗ hổng bảo mật trong các thành phần của hệ thống năm 2014



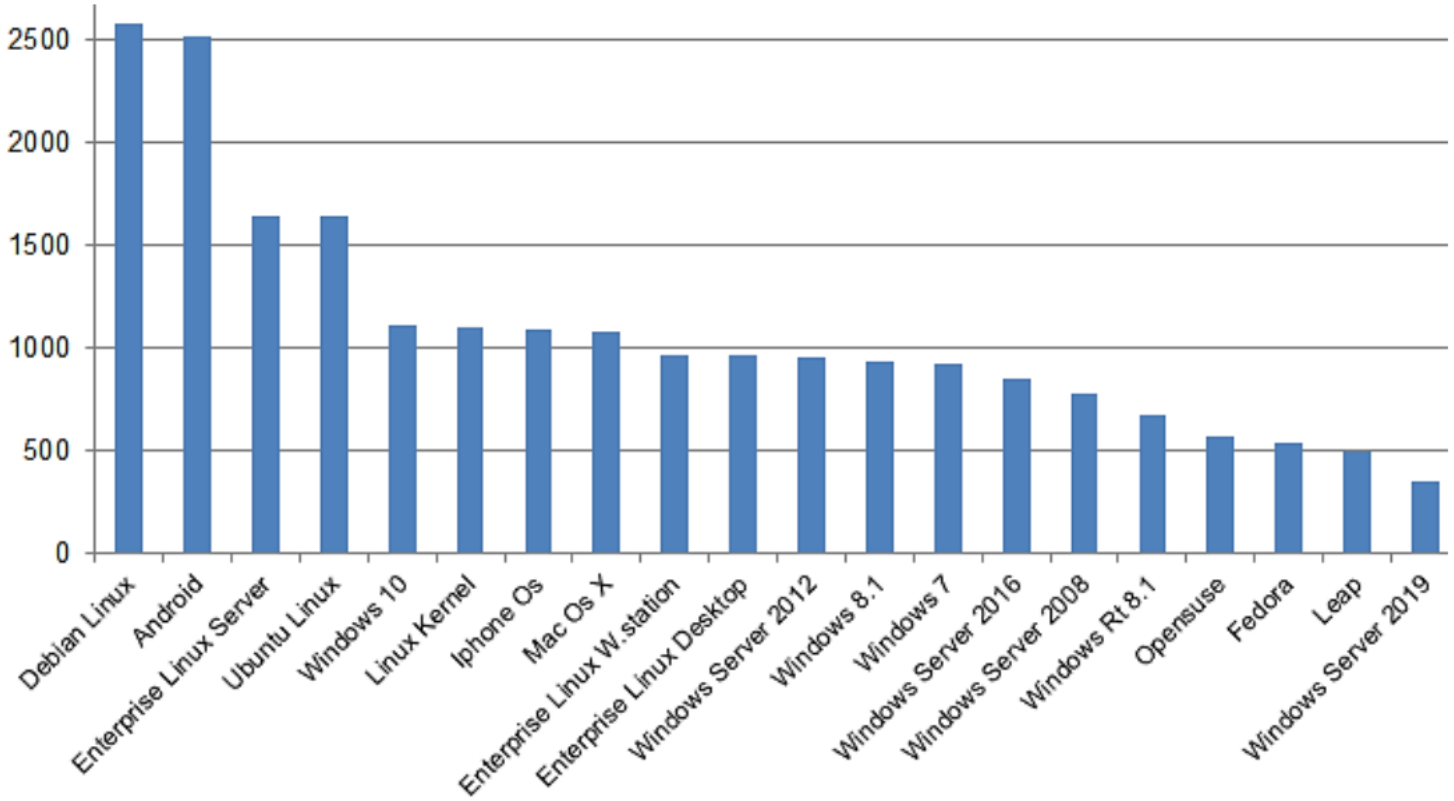
# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

Top 20 ứng dụng có lỗ hổng bảo mật phát hiện từ 2015 đến 2019



# 2.1 Tổng quan về lỗ hổng bảo mật và điểm yếu hệ thống

Top 20 hệ điều hành có lỗ hổng bảo mật phát hiện từ 2015 đến 2019

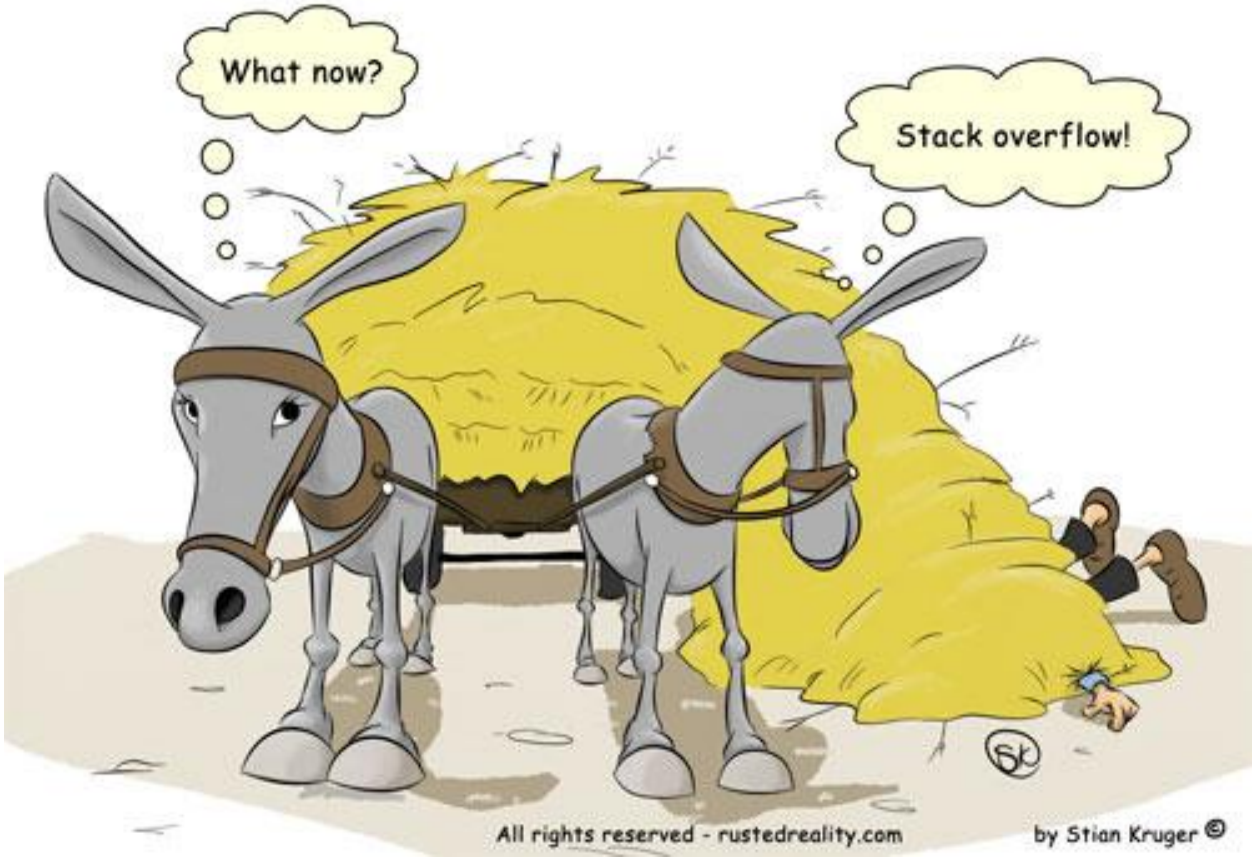


## 2.2 Các dạng lỗ hổng trong HĐH và phần mềm ứng dụng

- Các dạng lỗ hổng bảo mật thường gặp trong hệ điều hành và các phần mềm ứng dụng:
  - Lỗi tràn bộ đệm (buffer overflows)
  - Không kiểm tra đầu vào (unvalidated input)
  - Các vấn đề với điều khiển truy cập (access-control problems)
  - Các điểm yếu trong xác thực, trao quyền hoặc các hệ mật mã (weaknesses in authentication, authorization, or cryptographic practices)
  - Các lỗ hổng bảo mật khác.

# 2.2.1 Các dạng lỗ hổng - lỗi tràn bộ đệm

- Lỗi tràn bộ đệm xảy ra khi một ứng dụng cố gắng ghi dữ liệu vượt khỏi phạm vi bộ đệm (giới hạn cuối hoặc cả giới hạn đầu của bộ đệm);



# 2.2.1 Các dạng lỗ hổng - lỗi tràn bộ đệm

- Lỗi tràn bộ đệm có thể khiến ứng dụng ngừng hoạt động, gây mất dữ liệu hoặc thậm chí giúp kẻ tấn công kiểm soát hệ thống;
- Lỗi tràn bộ đệm chiếm một tỷ lệ lớn cho số các lỗi gây lỗ hổng bảo mật;
- Không phải tất cả các lỗi tràn bộ đệm có thể bị khai thác bởi kẻ tấn công.

# 2.2.1 Các dạng lỗ hổng - lỗi tràn bộ đệm

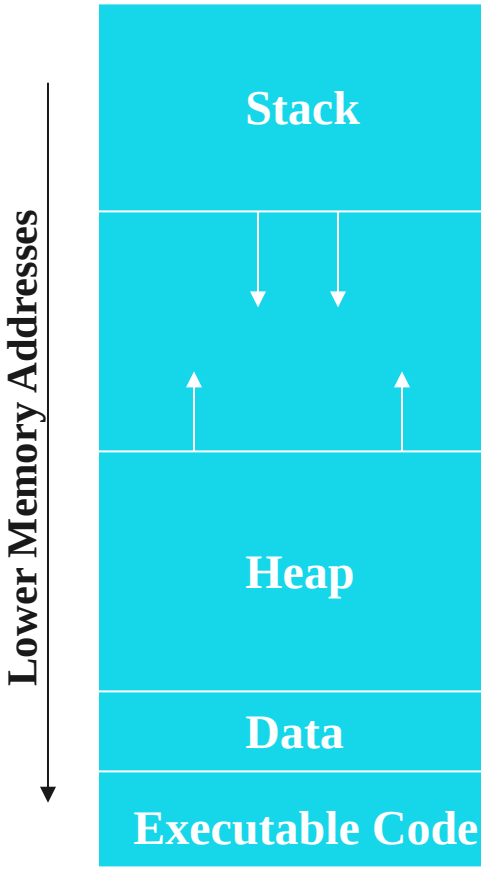
- Các vùng nhớ chứa bộ đệm của ứng dụng:
  - Ngăn xếp (Stack): vùng nhớ lưu các tham số gọi hàm, phương thức và dữ liệu cục bộ của chúng;
    - Các biến cục bộ được cấp phát tĩnh.
  - Vùng nhớ heap: là vùng nhớ chung lưu dữ liệu cho ứng dụng
    - Bộ nhớ heap thường được cấp phát động theo yêu cầu.

# 2.2.1 Các dạng lỗ hổng - lỗi tràn bộ đệm

- Giải thích cơ chế lỗi tràn bộ đệm trên bộ nhớ Stack và khả năng khai thác lỗ hổng:
  - Bài trình bày “Smashing the Stack” của tác giả Mark Shaneck, 2003.
    - Cơ chế hoạt động của Stack
    - Minh họa lỗi tràn bộ đệm trong Stack
    - Giải thích khả năng khai thác lỗi
    - Giải thích cơ chế hoạt động của sâu SQL Slammer và MS Blast – khai thác lỗi tràn bộ đệm.

# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

- Các vùng nhớ chứa bộ đệm của ứng dụng:
  - Ngăn xếp (Stack): vùng nhớ lưu các tham số gọi hàm, phương thức và dữ liệu cục bộ của chúng;
    - Các biến cục bộ được cấp phát tĩnh.
  - Vùng nhớ heap: là vùng nhớ chung lưu dữ liệu cho ứng dụng
    - Bộ nhớ heap thường được cấp phát động theo yêu cầu.



# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

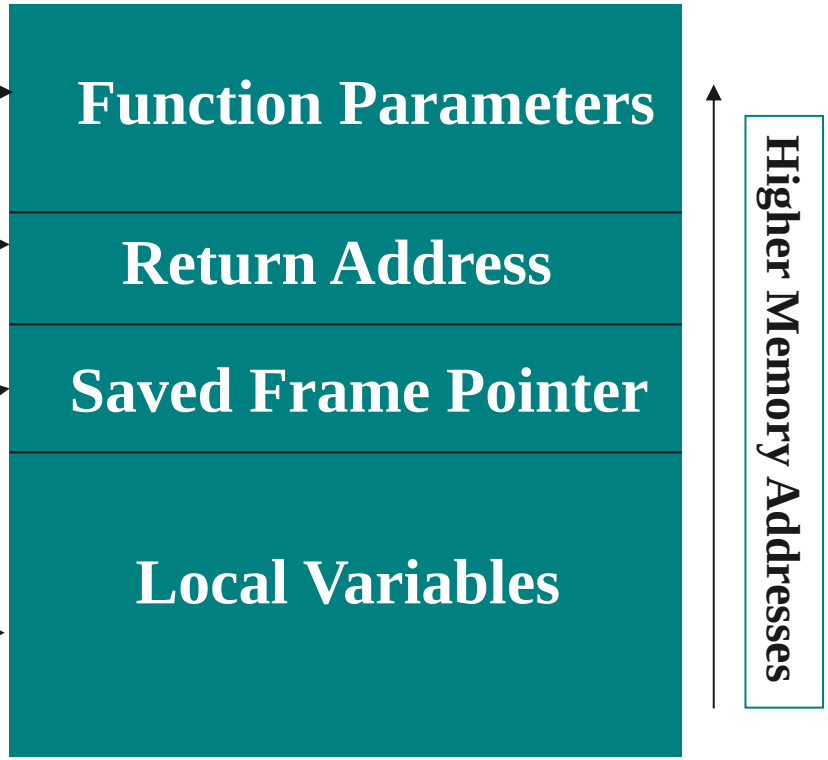
❖ **Xét chương trình sau để thấy được bộ nhớ ngăn xếp:**

```
void function(int a, int b, int c){
     char buffer1[5];
     char buffer2[10];
}

int main(){
     function(1,2,3);
}
```

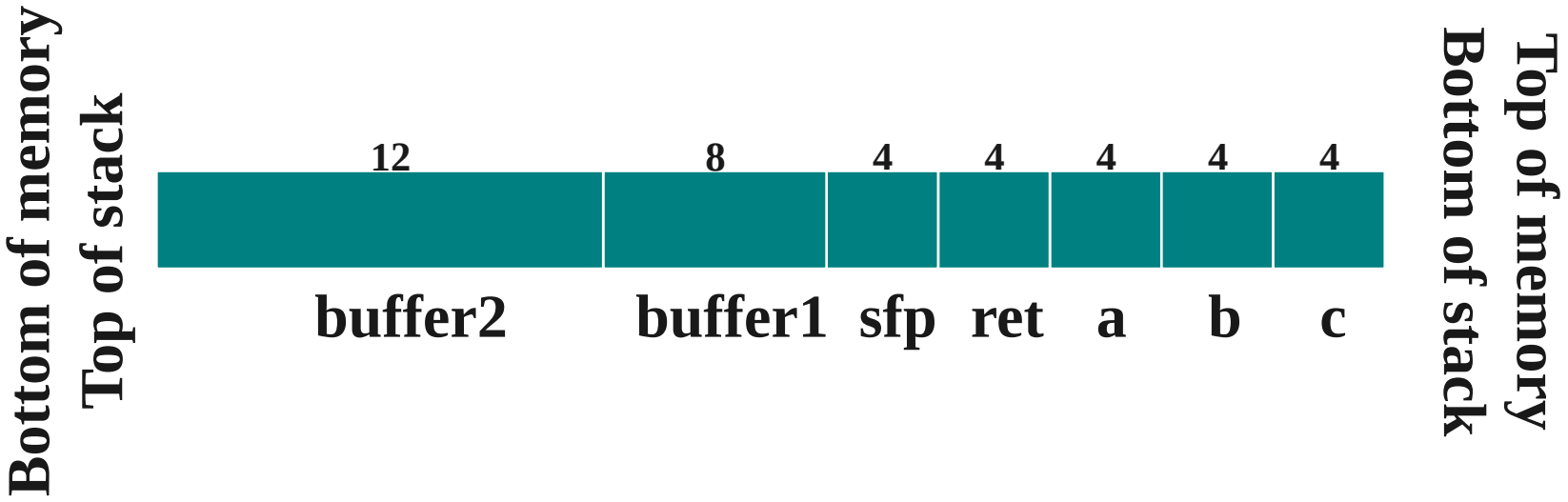
# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

```
pushl   $3
pushl   $2
pushl   $1
call function

function prolog
pushl   %ebp
movl    %esp, %ebp
subl    $20, %esp
```

Allocates space for local variables



# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm



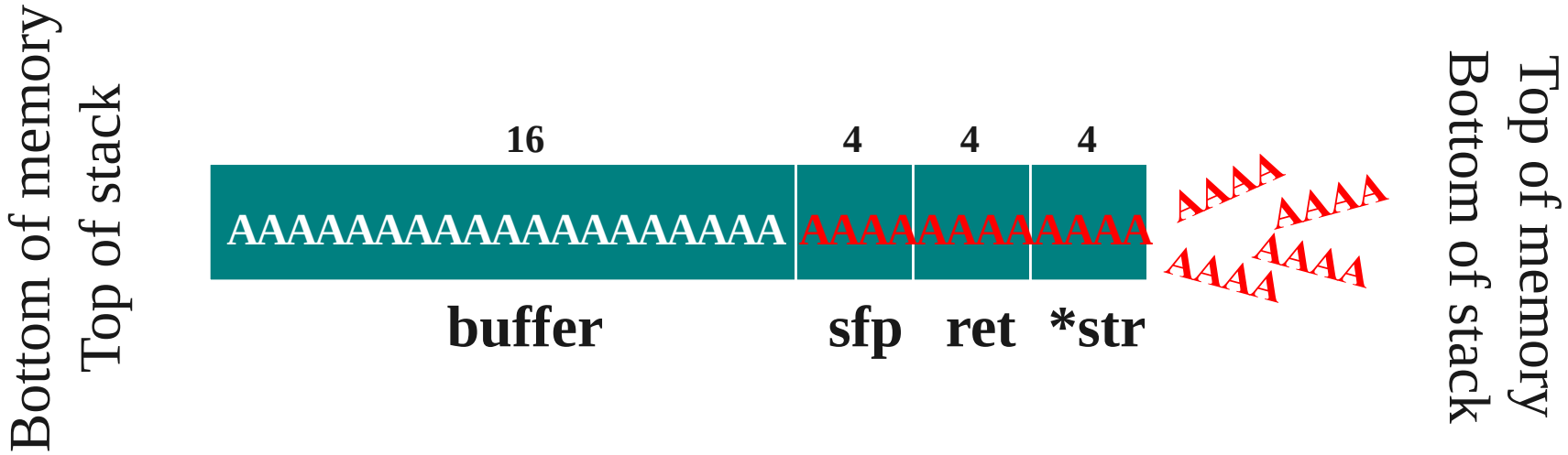
# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

- **Xét chương trình có lỗi tràn bộ đệm do không kiểm tra kích thước dữ liệu đầu vào:**

```
void function(char *str){
      char buffer[16];
      strcpy(buffer, str);
}

int main(){
      char large_string[256];
      int i;
      for (i = 0; i < 255; i++){
            large_string[i] = 'A';
      }
      function(large_string);
}
```

# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

- Khi chương trình chạy sẽ gây ra lỗi




- Giá trị trả về "ret" bị ghi đề bởi "AAAA" (0x41414141)
- Sau khi thực hiện xong hàm hiện tại, chương trình sẽ thực thi lệnh tại địa chỉ 0x41414141

# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

- Khi một ứng dụng chứa lỗ hổng tràn bộ đệm, tin tặc có thể khai thác bằng cách gửi mã độc dưới dạng dữ liệu đến ứng dụng nhằm ghi đè, thay thế địa chỉ trở về với mục đích tái định hướng chương trình đến thực hiện đoạn mã độc mà tin tặc gửi đến.
- Đoạn mã độc tin tặc xây dựng là mã máy có thể thực hiện được và thường được gọi là shellcode
- Mã shellcode có thể được viết bằng hợp ngữ, C, hoặc các ngôn ngữ lập trình khác, sau đó được chuyển thành mã máy, rồi chuyển định dạng thành một chuỗi dữ liệu và cuối cùng được gửi đến ứng dụng.

# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

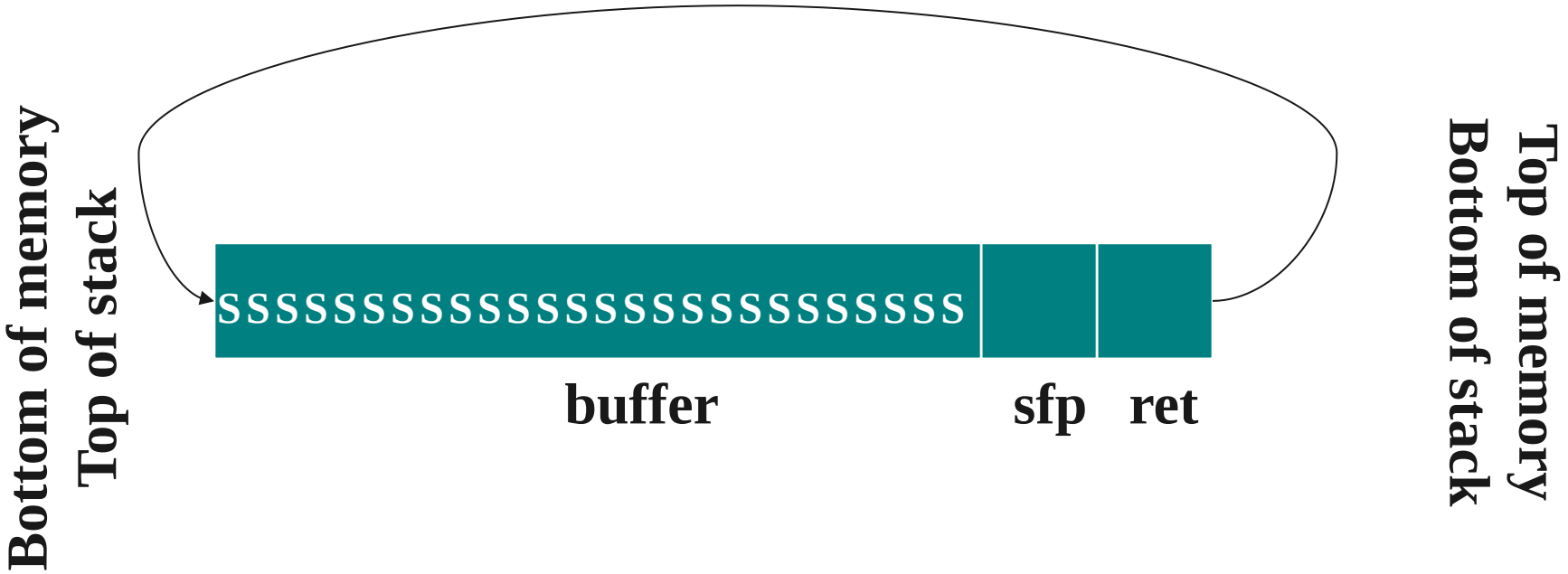
- Một shellcode viết bằng hợp ngữ và chuyển thành chuỗi tấn công

```
jmp     0x1F
popl    %esi
movl    %esi, 0x8(%esi)
xorl    %eax, %eax
movb    %eax, 0x7(%esi)
movl    %eax, 0xC(%esi)
movb    $0xB, %al
movl    %esi, %ebx
leal    0x8(%esi), %ecx
leal    0xC(%esi), %edx
int     $0x80
xorl    %ebx, %ebx
movl    %ebx, %eax
inc     %eax
int     $0x80
call    -0x24
.string "/bin/sh"
```

```
char shellcode[] =
"\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89"
"\x46\x0c\xb0\x0b\x89\xf3\x8d\x4e\x08\x8d\x56\x0c"
"\xcd\x80\x31\xdb\x89\xd8\x40\xcd\x80\xe8\xdc\xff"
"\xff\xff/bin/sh";
```

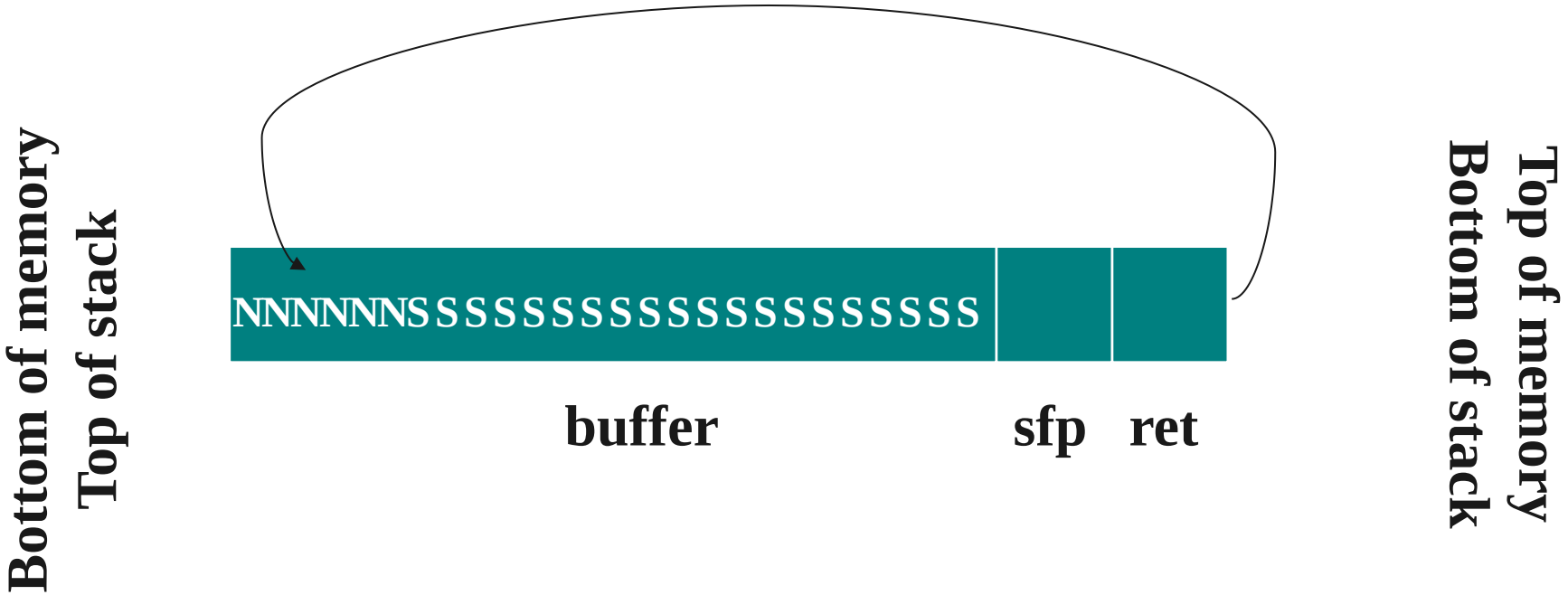
# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

- Shellcode được chèn, ghi đè lên ô nhớ chứa địa chỉ trở về ret, tái định hướng việc trở về từ chương trình con, chuyển đến thực hiện mã shellcode được chèn vào



# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

- Trên thực tế, để tăng khả năng đoạn mã shellcode được thực hiện, người ta thường chèn một số lệnh NOP (N) – No Operation (không thực hiện tác vụ nào) vào phần đầu shellcode để phòng khả năng địa chỉ ret mới không trỏ chính xác đến địa chỉ bắt đầu shellcode.

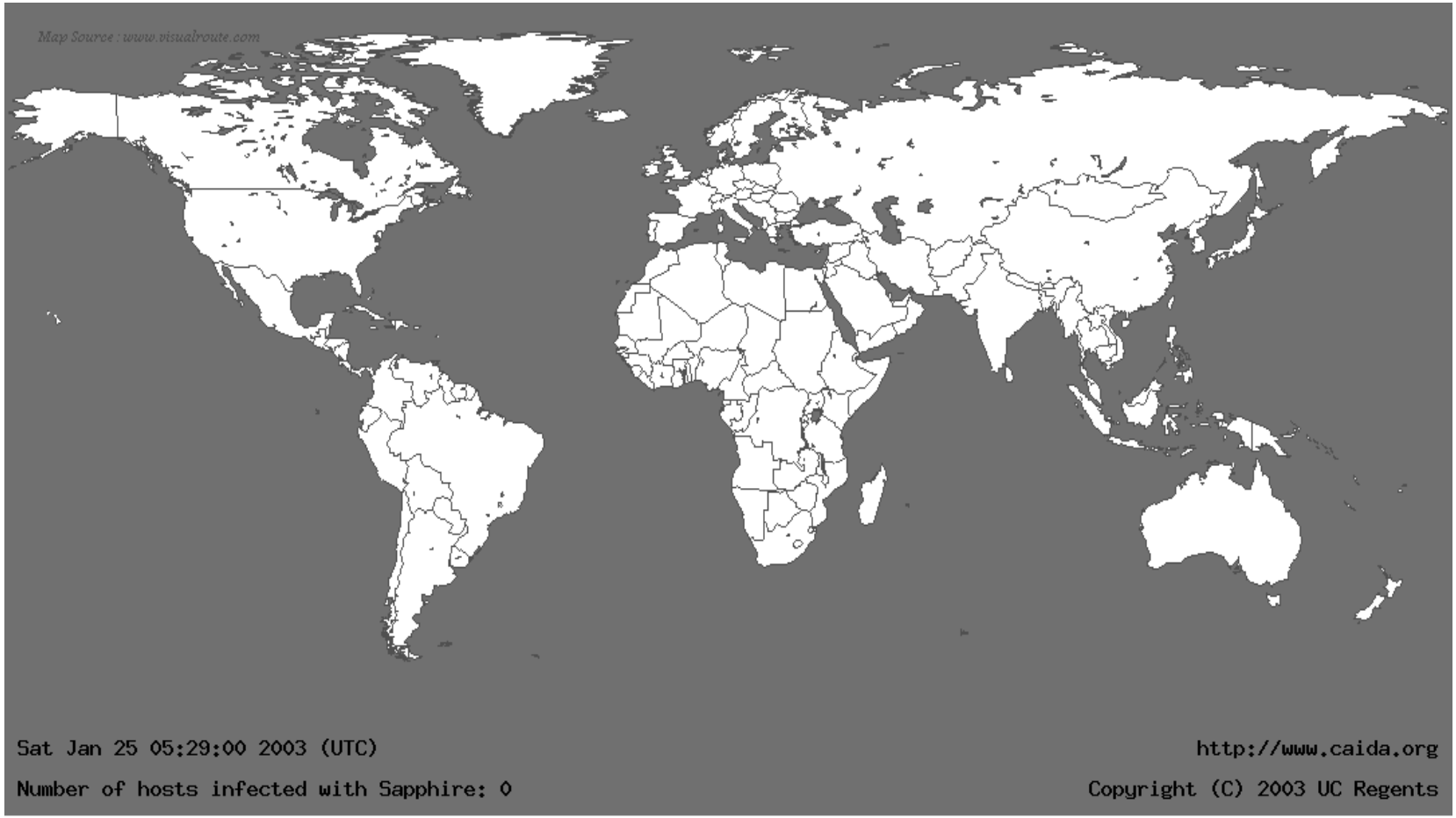


# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

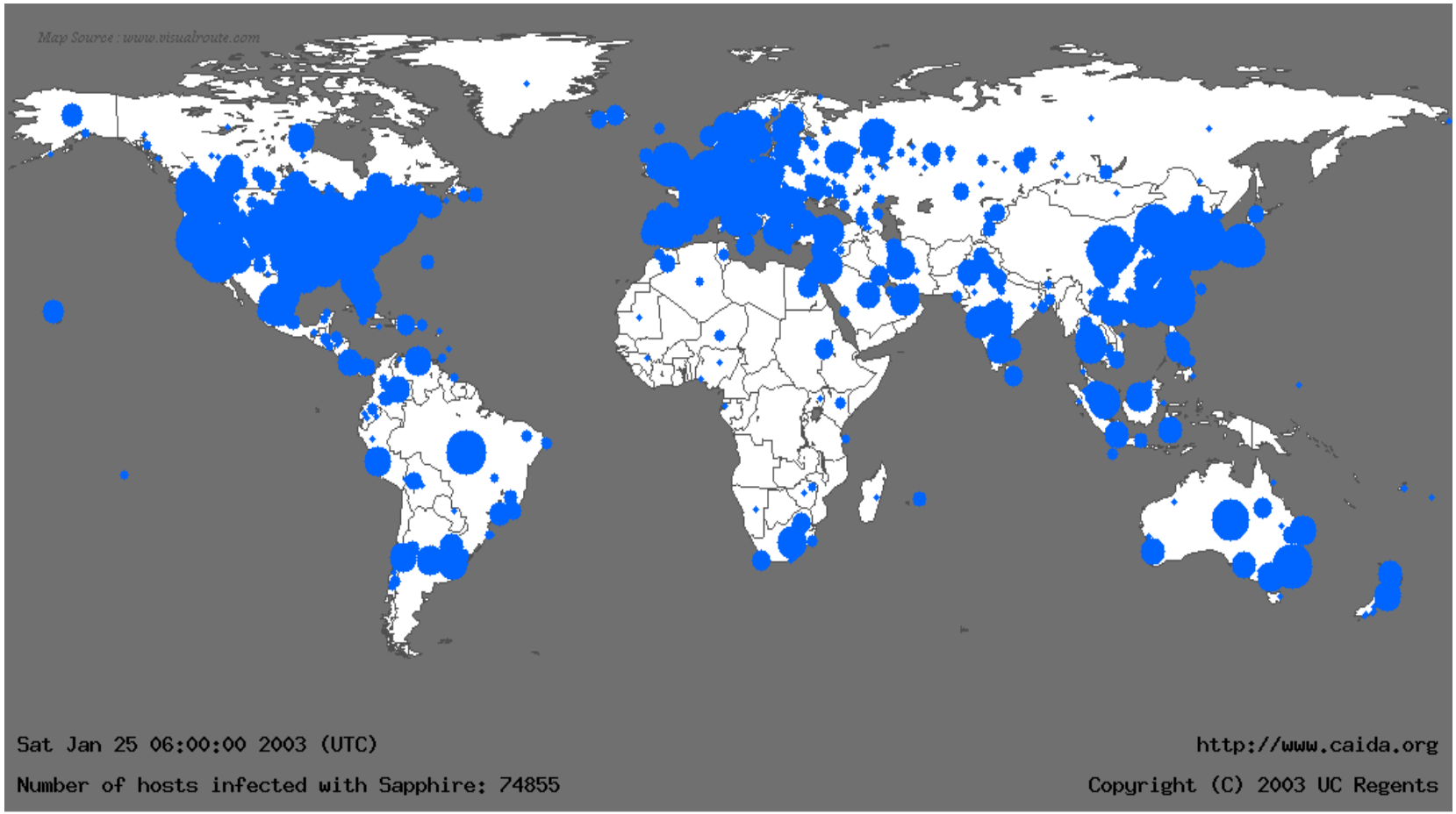
Ví dụ về khai thác lỗi tràn bộ đệm

- Sâu SQL Slammer (một số tài liệu gọi là sâu Sapphire) được phát hiện ngày 25/1/2003 lúc 5h30 (UTC) là sâu có tốc độ lây lan nhanh nhất lúc bấy giờ: nó lây nhiễm ra khoảng 75.000 máy chủ chỉ trong khoảng 30 phút
- Sâu Slammer khai thác lỗi tràn bộ đệm trong thành phần Microsoft SQL Server Resolution Service của hệ quản trị cơ sở dữ liệu Microsoft SQL Server 2000.
- Sâu sử dụng giao thức UDP với kích thước gói tin 376 byte và vòng lặp chính của sâu chỉ gồm 22 lệnh hợp ngữ. Chu trình hoạt động của sâu SQL Slammer gồm:
  - Sinh tự động địa chỉ IP;
  - Quét tìm các máy có lỗi với IP tự sinh trên cổng dịch vụ 1434;
  - Nếu tìm được, gửi một bản sao của sâu đến máy có lỗi;
  - Mã của sâu gây tràn bộ đệm, thực thi mã của sâu và quá trình lặp lại

# Slammer Worm - Eye Candy



# Slammer Worm - Eye Candy



# 2.3 Các dạng tấn công - Tấn công bằng mã độc: Tấn công lợi dụng lỗi tràn bộ đệm

- SQL Slammer là sâu "lành tính" vì nó không can thiệp vào hệ thống file, không thực hiện việc phá hoại hay đánh cắp thông tin ở hệ thống bị lây nhiễm. Tuy nhiên, sâu tạo ra lưu lượng mạng khổng lồ trong quá trình lây nhiễm, gây tê liệt đường truyền mạng Internet trên nhiều vùng của thế giới.
- Do mã của SQL Slammer chỉ được lưu trong bộ nhớ nó gây tràn mà không được lưu vào hệ thống file, nên chỉ cần khởi động lại máy là có thể tạm thời xóa được sâu khỏi hệ thống. Tuy nhiên, hệ thống chứa lỗ hổng có thể bị lây nhiễm lại nếu nó ở gần một máy khác bị nhiễm sâu.
- Các biện pháp phòng chống triệt để khác là cập nhật bản vá cho bộ phần mềm Microsoft SQL Server 2000.

# 2.2.1 Các dạng lỗ hổng - lỗi tràn bộ đệm

- Các biện pháp phòng chống lỗi tràn bộ đệm:
  - Kiểm tra mã nguồn bằng tay để tìm và vá các điểm có khả năng xảy ra lỗi tràn bộ đệm;
  - Sử dụng các công cụ phân tích mã tự động tìm các điểm có khả năng xảy ra lỗi tràn bộ đệm;
  - Đặt cơ chế không cho phép thực hiện mã trong Stack;
  - Sử dụng các cơ chế bảo vệ Stack:
    - Thêm một số ngẫu nhiên (canary) phía trước địa chỉ trở về;
    - Kiểm tra số ngẫu nhiên này trước khi trở về chương trình gọi để xác định khả năng bị thay đổi địa chỉ trở về.
  - Sử dụng các thư viện an toàn hoặc các ngôn ngữ không tràn, như Java, nền tảng .net.

# 2.2.2 Các dạng lỗ hổng – không kiểm tra đầu vào

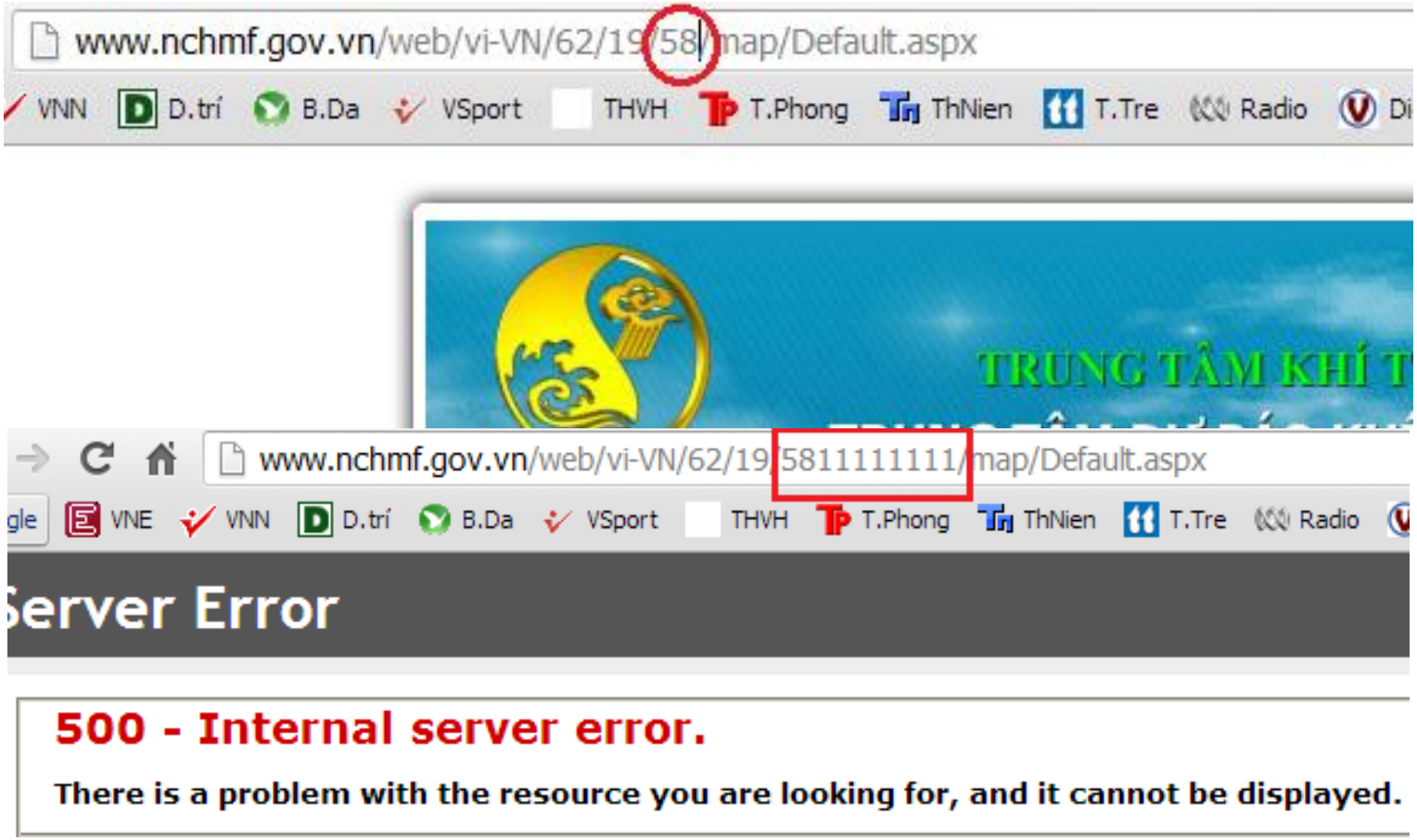
- Các dữ liệu đầu vào (input data) cần được kiểm tra để đảm bảo đạt các yêu cầu về định dạng và kích thước;
- Các dạng dữ liệu nhập điển hình cần kiểm tra:
  - Các trường dữ liệu text
  - Các lệnh được truyền qua URL để kích hoạt chương trình
  - Các file âm thanh, hình ảnh, hoặc đồ họa do người dùng hoặc các tiến trình khác cung cấp
  - Các đối số đầu vào trong dòng lệnh
  - Các dữ liệu từ mạng hoặc các nguồn không tin cậy
- Kẻ tấn công có thể kiểm tra các dữ liệu đầu vào và thử tất cả các khả năng để khai thác.

# 2.2.2 Các dạng lỗ hổng – không kiểm tra đầu vào

- Một số dạng tấn công lợi dụng lỗi không kiểm tra đầu vào:
  - Cố tình nhập dữ liệu quá lớn hoặc sai định dạng gây lỗi cho ứng dụng
    - Gây lỗi ứng dụng/dịch vụ, có thể làm ứng dụng ngừng hoạt động
  - Chèn mã khai thác vào dữ liệu đầu vào để thực hiện trên hệ thống của nạn nhân, nhằm đánh cắp dữ liệu nhạy cảm hoặc thực hiện các hành vi phá hoại

# 2.2.2 Các dạng lỗ hổng – không kiểm tra đầu vào

Trang web bị lỗi do không kiểm tra dữ liệu đầu vào từ URL



# 2.2.2 Các dạng lỗ hổng – không kiểm tra đầu vào

❖ Chèn mã SQL để đăng nhập mà không cần tài khoản và mật khẩu:

```
'Mã asp + SQL server
Dim username, password, sqlString
username = "dauhoang"
password = "abc123"
sqlString = "SELECT * FROM tbl_users WHERE username = '" & username &
"' AND password = '" & password & "'"
==> SELECT * FROM tbl_users WHERE username = 'dauhoang' AND password =
'abc123'
username = "aa' OR 1=1--"
password = "aaaa"
sqlString = "SELECT * FROM tbl_users WHERE username = " & username & "
AND password = " & password & ""
==> SELECT * FROM tbl_users WHERE username = 'aa' OR 1=1--' AND
password = 'aaaa'
```

# 2.2.2 Các dạng lỗ hổng – không kiểm tra đầu vào

❖ Chèn mã độc SQL vào trường text:

```
‘Mã asp + SQL server
Dim searchString, sqlString
searchString = "Lumia 640XL"
sqlString = "SELECT * FROM tbl_products WHERE product_name =
'" & searchString & “’”
==> SELECT * FROM tbl_products WHERE product_name = 'Lumia
640XL’
searchString = "Lumia 640XL';DELETE FROM tbl_products;--”
sqlString = "SELECT * FROM tbl_products WHERE product_name =
'" & searchString & “’”
==> SELECT * FROM tbl_products WHERE product_name = 'Lumia
640XL'; DELETE FROM tbl_products; --’
```

# 2.2.2 Các dạng lỗ hổng – không kiểm tra đầu vào

- Các biện pháp phòng chống dựa trên kiểm tra và lọc dữ liệu đầu vào:
  - Kiểm tra tất cả các dữ liệu đầu vào, đặc biệt dữ liệu nhập từ người dùng và từ các nguồn không tin cậy;
  - Kiểm tra định dạng và kích thước dữ liệu đầu vào;
  - Tạo các bộ lọc để lọc bỏ các ký tự đặc biệt và các từ khóa của các ngôn ngữ trong các trường hợp cần thiết mà kẻ tấn công có thể sử dụng:
    - Các ký tự đặc biệt: *, ‘, =, --
    - Các từ khóa:  SELECT, INSERT, UPDATE, DELETE, DROP,....

# 2.2.3 Các dạng lỗ hổng – Các v.đề với điều khiển truy nhập

- Điều khiển truy nhập (Access control) liên quan đến việc điều khiển ai (chủ thể) được truy cập đến cái gì (đối tượng)?
- Điều khiển truy nhập có thể được thiết lập bởi hệ điều hành hoặc mỗi ứng dụng, thường gồm 2 bước:
  - Xác thực (Authentication): xác thực thông tin nhận dạng của chủ thể;
  - Trao quyền (Authorization): cấp quyền truy nhập cho chủ thể sau khi thông tin nhận dạng được xác thực.
- Các chủ thể được cấp quyền truy nhập vào hệ thống theo các cấp độ khác nhau dựa trên chính sách an ninh của tổ chức

## 2.2.3 Các dạng lỗ hổng – Các v.đề với điều khiển truy nhập

- Nếu kiểm soát truy nhập bị lỗi, một người dùng bình thường có thể đoạt quyền của người quản trị và toàn quyền truy nhập vào hệ thống;
- Một kẻ tấn công có thể lợi dụng lỗ hổng bảo mật của hệ thống kiểm soát truy nhập để truy nhập vào các file trong hệ thống.
- Một ứng dụng chạy trên user quản trị có toàn quyền truy nhập vào hệ thống:
  - Nếu một kẻ tấn công chiếm được quyền điều khiển chương trình sẽ có toàn quyền truy nhập vào hệ thống.

## 2.2.3 Các dạng lỗ hổng – Các v.đề với điều khiển truy nhập

- Phương pháp phòng chống:
  - Không dùng user quản trị (root hoặc admin) để chạy các chương trình ứng dụng;
  - Luôn chạy các chương trình ứng dụng với quyền tối thiểu – vừa đủ để thực thi các tác vụ;
  - Kiểm soát chặt chẽ người dùng, xóa bỏ hoặc cấm truy nhập với những người dùng ngầm định kiểu everyone;
  - Thực thi chính sách mật khẩu an toàn;
  - Cấp quyền vừa đủ cho người dùng thực thi nhiệm vụ

# 2.3.4 Các dạng lỗ hổng - Các vấn đề với xác thực, trao quyền và mật mã

- Xác thực:
  - Mật khẩu được lưu dưới dạng rõ (plain text) → nguy cơ bị lộ mật khẩu rất cao trong quá truyền thông tin xác thực;
  - Sử dụng mật khẩu đơn giản, dễ đoán, hoặc dùng mật khẩu trong thời gian dài;
  - Sử dụng cơ chế xác thực không đủ mạnh: ví dụ các cơ chế xác thực của giao thức HTTP.
- Trao quyền:
  - Cơ chế thực hiện trao quyền không đủ mạnh, dễ bị vượt qua;
  - Ví dụ: một trang web chỉ thực hiện ẩn các links đến các trang web mà người dùng không được truy nhập mà không thực sự kiểm tra quyền truy nhập trên từng trang. Nếu người dùng tự gõ URL của trang thì vẫn có thể truy nhập

# 2.3.4 Các dạng lỗ hổng - Các vấn đề với xác thực, trao quyền và mật mã

❖ Các vấn đề với các hệ mật mã:

- Sử dụng giải thuật mã hóa/giải mã, hàm băm yếu, lạc hậu, hoặc có lỗ hổng (DES, MD4, MD5,...);
- Sử dụng khóa mã hóa/giải mã yếu;
  - Khóa có chiều dài ngắn;
  - Khóa dễ đoán.
- Các vấn đề trao đổi khóa bí mật;
- Các vấn đề xác thực người gửi/người nhận;
- Chi phí tính toán lớn (đặc biệt đối với các hệ mã hóa khóa công khai)

## 2.3.5 Các dạng lỗ hổng - Các điểm yếu bảo mật khác

- Các thao tác không an toàn với files:
  - Thực hiện đọc/ghi file lưu ở những nơi mà các người dùng khác cũng có thể ghi file đó;
  - Không kiểm tra chính xác loại file, định danh thiết bị, các links hoặc các thuộc tính khác của file trước khi sử dụng;
  - Không kiểm tra mã trả về sau mỗi thao tác với file;
  - Giả thiết một file có đường dẫn cục bộ là file cục bộ và bỏ qua các thủ tục kiểm tra:
    - File ở xa có thể được ánh xạ vào hệ thống file cục bộ  có đường dẫn cục bộ

## 2.3.5 Các dạng lỗ hổng - Các điểm yếu bảo mật khác

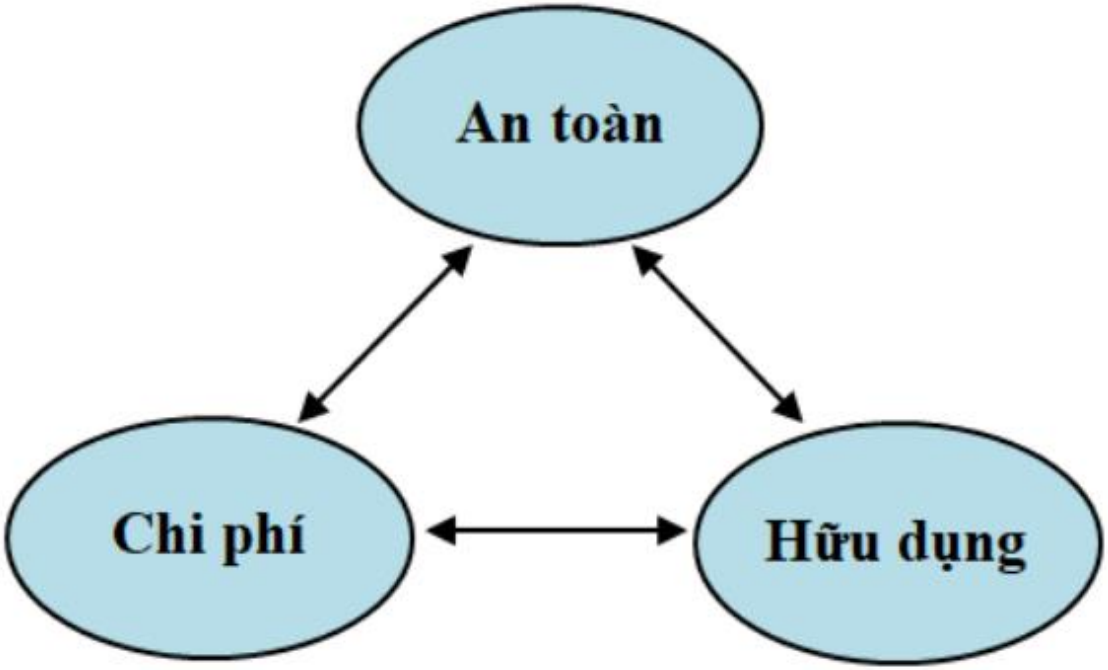
- Các điều kiện đua tranh (Race conditions):
  - Một điều kiện đua tranh tồn tại khi có sự thay đổi trật tự của 2 hay một số sự kiện gây ra sự thay đổi hành vi của hệ thống;
  - Đây là một dạng lỗi nếu chương trình chỉ có thể thực hiện đúng chức năng nếu các sự kiện phải xảy ra theo đúng trật tự;
  - Kẻ tấn công có thể lợi dụng khoảng thời gian giữa 2 sự kiện để chèn mã độc, đổi tên file hoặc can thiệp vào quá trình hoạt động bình thường của hệ thống.

# 2.3.5 Các dạng lỗ hổng - Các điểm yếu bảo mật khác

- Các điều kiện đua tranh – Ví dụ:
  - Các file tạm thời (Temporary files) thường được lưu ở một thư mục chung quản lý bởi HDH. Tiến trình của nhiều người dùng cùng có thể đọc/ghi file tạm thời;
  - Nếu 1 tiến trình tạo các cặp khóa bí mật và công khai, và lưu chúng vào một file tạm thời và sau đó đọc lại để lưu vào CSDL;
  - Kẻ tấn công có thể tạo một race condition trong khoảng thời gian tiến trình chuyển từ ghi sang đọc các cặp khóa:
    - Thay file tạm của tiến trình bằng file chứa khóa của hắn → tiến trình sẽ đọc và lưu khóa của kẻ tấn công → mọi thông điệp mã hóa sử dụng khóa trên có thể được giải mã;
    - Đọc file tạm của tiến trình để đánh cắp các cặp khóa.

# 2.4 Quản lý, khắc phục các lỗ hổng bảo mật và tăng cường khả năng đề kháng cho hệ thống

- Nguyên tắc: cân bằng giữa An toàn (Secure), Chi phí (Cost) và Hữu dụng (Usable)



# 2.4 Quản lý, khắc phục các lỗ hổng bảo mật và tăng cường khả năng đề kháng cho hệ thống

❖ Một số biện pháp cụ thể:

- Thường xuyên cập nhật thông tin về các điểm yếu, lỗ hổng bảo mật từ các trang web chính thức:
  - http://cve.mitre.org/ (CVE - Common Vulnerabilities and Exposures)
  - http://www.cvedetails.com/ (CVE Details)
  - http://web.nvd.nist.gov (National Vulnerability Database)
  - https://www.owasp.org/index.php/Category:Vulnerability
- Định kỳ cập nhật các bản vá, nâng cấp hệ điều hành và các phần mềm ứng dụng;
  - Sử dụng các hệ thống quản lý các bản vá và tự động cập nhật định kỳ
    - Microsoft Windows Updates
    - Tiện ích Update trên Linux
    - Tính năng tự động Update của các ứng dụng (Như Google Update service)
  - Với các lỗ hổng nghiêm trọng, cần cập nhật tức thời

# 2.4 Quản lý, khắc phục các lỗ hổng bảo mật và tăng cường khả năng đề kháng cho hệ thống

❖ Một số biện pháp cụ thể:

- Cần có chính sách quản trị người dùng, mật khẩu và quyền truy nhập chặt chẽ ở mức hệ điều hành và mức ứng dụng:
  - Người dùng chỉ được cấp quyền truy nhập vừa đủ để thực hiện công việc được giao.
  - Nếu được cấp nhiều quyền hơn mức cần thiết → lạm dụng.
- Sử dụng các biện pháp phòng vệ ở lớp ngoài như tường lửa, proxies:
  - Chặn các dịch vụ/cổng không thực sự cần thiết;
  - Ghi logs các hoạt động truy nhập mạng.
- Sử dụng các phần mềm rà quét lỗ hổng, rà quét các phần mềm độc hại:
  - Có thể giảm thiểu nguy cơ bị lợi dụng, khai thác lỗ hổng bảo mật

# 2.4 Quản lý, khắc phục các lỗ hổng bảo mật và tăng cường khả năng đề kháng cho hệ thống

- Công cụ quét cổng dịch vụ:
  - Nmap
  - Angry IP Scanner
  - SuperScan
- Công cụ rà quét lỗ hổng bảo mật hệ thống
  - Microsoft Baseline Security Analyser
  - Nessus Vulnerability Scanner
- Công cụ rà quét lỗ hổng ứng dụng web
  - Acunetix Web Vulnerability Scanner
  - IBM AppScan
  - Beyond Security AVDS
  - SQLMap