Trường Đại học Công nghệ Thông tin
Khoa Mạng máy tính và truyền thông

# MỘT SỐ CÂU HỎI MẪU ÔN TẬP THI CUỐI KÌ

Môn học: **Lập trình an toàn và khai thác lỗ hổng phần mềm – NT521**

Năm học: **2024-2025**

Email liên hệ: **duypt@uit.edu.vn**

***Lưu ý: *Các câu hỏi ở bên dưới đây là ví dụ mẫu không bao gồm đáp án, và không phải là đề thi mẫu, hay toàn bộ các dạng câu hỏi trong kỳ thi cuối kỳ. Sinh viên sử dụng tài liệu này như một tài liệu tham khảo để ôn tập một số vấn đề quan trọng trong nội dung môn học.*

**Câu 1:** Trong qui trình phát triển phần mềm an toàn, nhận định "Giải quyết vấn đề liên quan tới bảo mật càng trễ, chi phí bỏ ra càng lớn" là _____.

**A.** Nguyên tắc viết kịch bản kiểm thử an toàn (Pentest)
**B.** Nguyên tắc Dịch trái (Pushing Left)
**C.** Công việc của kỹ sư DevOps
**D.** Nguyên tắc lập trình an toàn (Secure Programming)

**Câu 2:** Xét đoạn code C sau đây, hãy chọn payload nào có khả năng cao nhất để khai thác lỗ hổng Format String thành công trong hàm printf(buffer):

```
#include <stdio.h>

int main() {
    char buffer[100];
    printf("Enter a string: ");
    fgets(buffer, sizeof(buffer), stdin);
    printf(buffer);  // Note: Intentionally using printf without a format string
    return 0;
}
```

format string

A. "Hello World" - một chuỗi bình thường không chứa ký tự đặc biệt.

B. "%s %s %s" - một chuỗi định dạng có thể làm lộ ra nội dung của một số vùng nhớ liền kề.

C. "123456" - một chuỗi số đơn giản.

D. "%x %x %x" - một chuỗi định dạng hiển thị giá trị hex của các địa chỉ trên stack.

**Câu 3:** Cho đoạn mã như bên dưới, xác định độ phủ kiểm thử mức độ câu lệnh (phủ câu lệnh) với test case là (a=3, b=9).

| | |
|---|---|
| 1 | Prints (int a, int b) { |

```
2     int result = a + b;
3     If (result > 0)
4       Print ("Positive", result)
5     Else
6       Print ("Negative", result)
7   }
```

**A.** 71% **B.** 80% **C.** 81% **D.** 100%

**Câu 4:** Chọn một phát biểu ĐÚNG về vai trò của độ phủ trong kiểm thử phần mềm.

A. Độ đo bao phủ càng nhỏ thì độ tin cậy của bộ kiểm thử càng cao

B. Độ đo bao phủ càng lớn thì độ tin cậy của bộ kiểm thử càng cao

C. Độ đo bao phủ càng lớn thì độ tin cậy của bộ kiểm thử càng thấp

D. Độ đo bao phủ không ảnh hưởng đến độ tin cậy của bộ kiểm thử

**Câu 5:** Kết quả đầu ra của giai đoạn Cài đặt và bảo trì (Deployment & Maintenance) trong SDLC là gì?

**A.** Tài liệu yêu cầu khách hàng, tài liệu đề xuất giải pháp, tài liệu quản lý dự án
**B.** Sản phẩm (mã nguồn)
**C.** Hồ sơ thiết kế, Flowchart, diagram
**D.** Tài liệu đặc tả hệ thống, Tài liệu hướng dẫn sử dụng, Tài liệu cấu hình và cài đặt

**Câu 6:** Cho đoạn mã như bên dưới, xác định độ phủ kiểm thử mức độ điều kiện con (phủ điều kiện quyết định con – condition coverage) với 02 test case là (a=4, b=8) và (a=7, b=2).

```
1   Prints (int a, int b) {
2     int result = a + b;
3     int c = 3 *a;
4     If (result > 0 && result%2)
5       Print ("Awesome, waiting…", result)
6     If (result == c)
7       Print ("You are lucky", result)
8     Else
9       Print (“Try more”, result)
10  }
```

**A.** 25% **B.** 66,6% **C.** 83% **D.** 100%

**Câu 7:** Để thực thi được các gadget theo trình tự g1 → g2 → g3 → g4, trong đó addr g2 < addr g3 < addr g1 < addr g4, ta cần đặt địa chỉ của chúng liền nhau theo thứ tự nào từ ô nhớ có địa chỉ thấp đến cao?

**A.** g4 → g3 → g2 → g1
**B.** g1 → g2 → g3 → g4
**C.** g2 → g3 → g1 → g4
**D.** g4 → g1 → g3 → g2

**Câu 8:** Chuỗi định dạng (Format string) nào cho phép ghi dữ liệu vào bộ nhớ?

**A.** %n
**B.** %d
**C.** %x
**D.** %s

**Câu 9:** Chọn phát biểu **SAI** trong số các phát biểu biểu bên dưới.

**A.** Lỗi Off-by-one (dư một) là trường hợp riêng của lỗi tràn bộ đệm trong đó chỉ một ký tự bị tràn
**B.** Trong ngôn ngữ C, chuỗi định dạng là tham số thứ hai được truyền vào hàm *printf*, nó chứa các yêu cầu định dạng để xác định cách thức dữ liệu sẽ được hiển thị bởi hàm *printf*.
**C.** Lỗi chuỗi định dạng có thể giúp thăm dò và xác định các giá trị trên ngăn xếp (Stack). Nếu dữ liệu nhập vào của ta cũng nằm trên ngăn xếp thì sẽ có trường hợp ta gặp lại chính dữ liệu nhập.
**D.** Chương trình có thể bị mắc lỗi bảo mật ở một nơi nhưng chỉ thật sự bị khai thác/tận dụng tại vị trí khác

**Câu 10:** Cho biến **short a = 0x7ff0** và 2 biến **short b, x**. Giả sử dùng code C để tính toán và in ra màn hình kết quả **x = a + b** với printf("a + %hd = %hd", b, x). Với các giá trị b khác nhau, lệnh printf này có thể in ra được bao nhiêu dòng dưới đây?

(1) a + 1 = 32753
(2) a + 16 = 32768
(3) a + 16 = 0
(4) a + 17 = -32767
(5) a + 10 = -32760

**A.** 1
**B.** 2
**C.** 3
**D.** 4

**Câu 11:** Xem xét đoạn code C sau và chỉ ra đâu là lỗ hổng Off-by-one có thể xuất hiện.

```
#include <stdio.h>
#include <string.h>

void copy_input(char *input) {
    char buffer[100];
    strncpy(buffer, input, sizeof(buffer));
    buffer[99] = '\0';
    printf("Input: %s\n", buffer);
}

int main() {
```

```
    char input[150];
    fgets(input, sizeof(input), stdin);
    copy_input(input);
    return 0;
}
```

A. Hàm fgets nhận đầu vào có thể dài hơn kích thước của mảng input.

B. Gán buffer[99] = '\0' có thể không đủ để ngăn chặn tràn bộ nhớ nếu đầu vào dài hơn 99 ký tự.

C. Sử dụng strncpy với sizeof(buffer) là không an toàn và có thể gây tràn nếu đầu vào quá dài.

D. In ra buffer bằng printf không gây ra lỗi Off-by-one nhưng có thể dẫn đến lộ thông tin nếu buffer chứa dữ liệu nhạy cảm.

**Câu 12:** Kiểm thử hệ thống (System Testing) được thực hiện ngay sau khi:

A. Kiểm thử đơn vị

B. Kiểm thử chấp nhận

C. Kiểm thử tích hợp

D. Kiểm thử Module

**Câu 13:** Điều nào sau đây không phải là kỹ thuật kiểm thử hộp đen?

A. Phân tích giá trị biên
B. Kiểm tra cú pháp
C. Bảng quyết định
D. Phân vùng tương đương

**Câu 14:** Xem xét đoạn code C sau và chỉ ra đâu là lỗ hổng Format String có thể xuất hiện.

```
#include <stdio.h>


int main() {
    char buffer[100];
    printf("Enter a string: ");
    fgets(buffer, sizeof(buffer), stdin);
    printf(buffer);  // Note: Intentionally using printf without a format string
    return 0;
```

```
}
```

A. Sử dụng fgets để đọc đầu vào không gây ra lỗi Format String.

B. Dùng printf(buffer) mà không có chuỗi định dạng cố định là nguyên nhân gây ra lỗ hổng Format String.

C. Khai báo mảng buffer[100] không đủ lớn để chứa đầu vào, có thể gây tràn bộ nhớ đệm.

D. In ra chuỗi đầu vào không gây ra lỗ hổng an ninh nếu chuỗi đó không chứa ký tự đặc biệt.

**Câu 15:** Kiểm thử hộp trắng còn gọi là gì?

A. Kiểm thử cấu trúc
B. Kiểm thử đoán lỗi
C. Kiểm thử dựa vào thiết kế
D. Kiểm thử dựa vào kỹ thuật

**Câu 16:** Trong đoạn code C dưới đây, hãy chỉ ra đâu là điểm yếu có thể gây ra lỗ hổng Heap Overflow.

```
#include <stdio.h>
#include <stdlib.h>


int main() {
    char *buff = (char *)malloc(10);
    printf("Enter some text: ");
    gets(buff);
    printf("You entered: %s\n", buff);
    free(buff);
    return 0;
}
```

A. Sử dụng hàm malloc(10) không đủ an toàn vì không kiểm tra giá trị trả về.

B. Sử dụng hàm printf có thể gây rò rỉ thông tin.

C. Hàm gets(buff) không giới hạn số lượng ký tự đầu vào, có thể gây tràn bộ nhớ đệm.

D. Giải phóng bộ nhớ bằng free(buff) là không cần thiết và có thể gây ra lỗi.

**Câu 17:** Một input nhận 5 giá trị năm sinh trong [1000, 20000]. Các giá trị biên để kiểm thử cho dữ liệu input (đầu vào) của trường này là?

A. 0, 1900, 1990, 2000

B. 999, 1000, 2000, 2001, 2002

C. 1000, 1001, 1090, 1999

D. 1000, 2000

**Câu 18:** Cho đoạn mã dưới đây, số test case tối thiểu để phủ 100% branch coverage là bao nhiêu?

```
If (x>y) x+=1;

Else y+=1;

While (x>y)

Y=x*y; x=x+1;
```

A. 1
B. 2
C. 3
D. 4

***Chúc các bạn ôn tập tốt và có một kỳ thi như ý!***