BÁO CÁO THỰC HÀNH

**Môn học: Lập trình an toàn & Khai thác lỗ hổng phần mềm**

**Lab 5: Integer overflow và ROP**

*GVHD: Nguyễn Hữu Quyền*

1. **THÔNG TIN CHUNG:**

*(Liệt kê tất cả các thành viên trong nhóm)*

Lớp: NT521.O12.ATCL - Nhóm 3

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Email** |
| 1 | Bùi Hoàng Trúc Anh | 21521817 | [21521817@gm.uit.edu.vn](mailto:21521817@gm.uit.edu.vn) |
| 2 | Lê Hoàng Oanh | 21521253 | 21521253@gm.uit.edu.vn |
| 3 | Nguyễn Ngọc Trà My | 21520353 | [21520353@gm.uit.edu.vn](mailto:21520353@gm.uit.edu.vn) |
| 4 | Huỳnh Minh Tân Tiến | 21521520 | 21521520@gm.uit.edu.vn |

1. **NỘI DUNG THỰC HIỆN:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Công việc** | **Kết quả tự đánh giá** |
| 1 | **Yêu cầu 1** | 100% |
| 2 | **Yêu cầu 2** | 100% |
| 3 | **Yêu cầu 3** | 100% |
| 4 | **Yêu cầu 4** | 100% |
| 5 | **Yêu cầu 5** | 100% |
| 6 | **Yêu cầu 6** | 100% |
| 7 | **Yêu cầu 7** | 100% |

**Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.**

BÁO CÁO CHI TIẾT

**Yêu cầu 1. Sinh viên giải thích kết quả thực hiện, vì sao ta có được những kết quả như hình trên? Khi nào xảy ra tràn trên?**

Khi thực hiện phép cộng với các giá trị a = 0x7fff và b = 0xffff, bạn đang làm việc với các giá trị short (2 byte). Dưới đây là giải thích kết quả của các phép cộng và lý do xảy ra tràn trên:

1. Với trường hợp có dấu (signed):

- Giá trị a = 0x7fff là giá trị tối đa cho một số nguyên có dấu (signed short). Khi thực hiện phép cộng a + 1, giá trị sẽ trở thành 0x8000. Tuy nhiên, đây là giá trị tối thiểu cho một số nguyên có dấu 2 byte, nên nó sẽ trở thành số âm. Kết quả sẽ là -32768 (được hiểu là giảm giá trị tối đa đi một đơn vị).

- Định dạng in %hd sẽ hiển thị giá trị short có dấu, và kết quả sẽ là -32768.

2. Với trường hợp không dấu (unsigned):

- Giá trị b = 0xffff là giá trị tối đa cho một số nguyên không dấu (unsigned short). Khi thực hiện phép cộng b + 1, giá trị sẽ trở thành 0x10000. Tuy nhiên, do giới hạn của 2 byte, nó sẽ trở thành 0.

- Định dạng in %hu sẽ hiển thị giá trị short không dấu, và kết quả sẽ là 0.

3. Về lý do xảy ra tràn trên:

- Tràn trên xảy ra khi một phép toán tạo ra một giá trị lớn hơn giá trị tối đa có thể biểu diễn trong một kiểu dữ liệu nhất định. Trong trường hợp có dấu, việc thêm 1 vào giá trị tối đa dẫn đến việc nó trở thành giá trị tối thiểu với dấu, nơi tràn trên xảy ra.

- Trong trường hợp không dấu, việc thêm 1 vào giá trị tối đa sẽ đưa giá trị về 0, không tạo ra tràn trên.

**Yêu cầu 2. Sinh viên giải thích kết quả thực hiện, vì sao ta có được những kết quả như hình trên? Khi nào xảy ra tràn dưới?**

Khi thực hiện phép trừ với các giá trị a = 0x8000 và b = 0x0000, ta đang làm việc với các giá trị short (2 byte). Dưới đây là giải thích kết quả của các phép trừ và lý do xảy ra tràn dưới:

1. Với trường hợp có dấu (signed):

- Giá trị a = 0x8000 là giá trị tối thiểu cho một số nguyên có dấu (signed short). Khi thực hiện phép trừ a - 1, giá trị sẽ trở thành 0x7fff. Điều này là kết quả hợp lý vì đang giảm giá trị tối thiểu đi một đơn vị, không xảy ra tràn dưới.

- Định dạng in %hd sẽ hiển thị giá trị short có dấu, và kết quả sẽ là 32767.

2. Với trường hợp không dấu (unsigned):

- Giá trị b = 0x0000 là giá trị tối thiểu cho một số nguyên không dấu (unsigned short). Khi thực hiện phép trừ b - 1, giá trị sẽ trở thành 0xffff. Điều này là kết quả hợp lý vì đang giảm giá trị tối thiểu đi một đơn vị, không xảy ra tràn dưới.

- Định dạng in %hu sẽ hiển thị giá trị short không dấu, và kết quả sẽ là 65535.

3. Về lý do không xảy ra tràn dưới:

- Tràn dưới xảy ra khi một phép toán tạo ra một giá trị nhỏ hơn giá trị tối thiểu có thể biểu diễn trong một kiểu dữ liệu nhất định. Trong trường hợp có dấu, giả sử khi ta trừ 1 từ giá trị tối thiểu, nó chỉ tạo ra giá trị lớn hơn tối thiểu và không làm xảy ra tràn dưới.

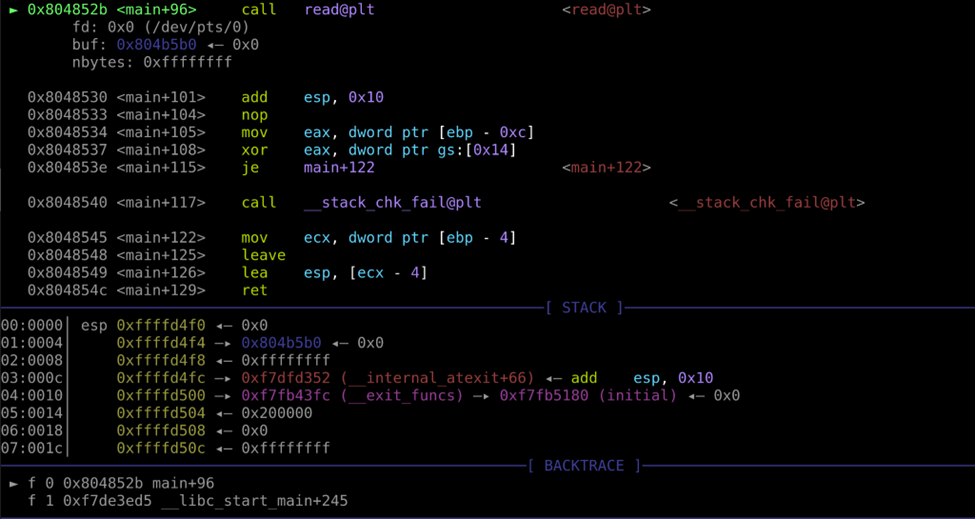
- Tương tự, trong trường hợp không dấu, giả sử khi ta trừ 1 từ giá trị tối thiểu, nó chỉ tạo ra giá trị lớn hơn tối thiểu và không làm xảy ra tràn dưới.

**Yêu cầu 3. Với data\_len nhập vào là -1, hàm malloc() sẽ nhận giá trị tham số bao nhiêu? Read sẽ đọc chuỗi có giới hạn là bao nhiêu byte? Giải thích các giá trị? Lưu ý: Báo cáo các giá trị sau khi đã chuyển sang hệ 10. Ví dụ: 0xB là 11.**

- Giá trị -1 được lưu thành 0xffffffff

- Hàm malloc nhận tham số có giá trị 0xf do khi 0xffffffff + 0x10 xảy ra tràn số

- Có thể thấy kết quả phép tính là 1 0000 000F vượt quá phạm vi biểu diễn 4 bytes (int len) do đó bị cắt bớt byte đầu là 1. Kết quả lưu lại thành 0000 000F (0xF)



-Read nhận tham số thứ 3 là 0xffffffff tuy nhiên hàm này nhận số nguyên không dấu do đó read sẽ hiểu thành 4,294,967,295 - ký tự tối đa read đọc

**Yêu cầu 4. Sinh viên thử tìm giá trị của a để chương trình có thể in ra thông báo “OK! Cast overflow done”? Giải thích?**

Trước hết, ta chạy thử chương trình để chọn số phù hợp, do biết được ngưỡng 4 byte của int nên ta có thể dự đoán số cần cần nhập là 2\*\*32 (32 bit). Giải thích việc xóa bit, việc tràn số sẽ khiến chương trình xóa những bit dư ở phía bên trái và trở thành một số với giá trị khác

0x00000000ffffffff -> 0xffffffff

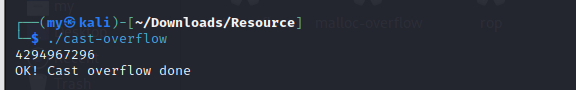
0x0000000100000000 -> 0x00000000

0x0000000100000001 -> 0x00000001

vậy số ta cần chọn là 2\*\*32 ở dạng dec: 4294967296

Từ đó ta có thể phát hiện quy luật bằng là với 1 con số bất kỳ có dạng 0xXXXXXXXX00000000 ta có thể thực hiện việc buffer overflow

vd 2\*\*33, 2\*\*34,...



**Yêu cầu 5. Sinh viên khai thác lỗ hổng stack overflow của file thực thi vulnerable, điều hướng chương trình thực thi hàm success. Báo cáo chi tiết các bước thực hiện.**

Địa chỉ hàm success: 0x0804846b

A screen shot of a computer

Description automatically generated

Return address của vulnerable()   
Cách 1 : disass main

A screen shot of a computer program

Description automatically generated\

=> return address của vulnerable là 0x080484c8   
Cách 2: kiểm tra frame của vulnerable()

A screen shot of a computer code

Description automatically generated

saved eip đóng vai trò là thanh ghi lưu return address

Debug check stackframe của vulnerabl():

A screen shot of a computer program

Description automatically generated  
A screenshot of a computer screen

Description automatically generated

Từ địa chỉ bắt đầu của stack 0xffffcf84 đến return address là 24byte , và ghi đè return address thành địa chỉ của success()

A computer screen with red and white text

Description automatically generated

Kết quả

A screen shot of a computer screen

Description automatically generated

**Yêu cầu 6. Sinh viên tự tìm hiểu và giải thích ngắn gọn về: procedure linkage table và Global Offset Table trong ELF Linux.**

- Procedure linkage table được sử dụng để gọi hàm hoặc thủ tục mà không cần biết địa chỉ trong thời gian link tới.

- Global Offset Table là một phần của chương trình máy tính(file thực thi hoặc thư viện chung) được dùng để giúp code chương trình biên dịch như một file ELF để chạy chính xác mà không cần phụ thuộc vào địa chỉ bộ nhớ của code hoặc dữ liệu được load lên lúc chạy.

**Yêu cầu 7. Sinh viên khai thác lỗ hổng stack overflow trong file rop để mở shell tương tác.**

Tìm gadget của eax : 0x080bb196

A screen shot of a computer code

Description automatically generated

Gadget của edx\_ecx\_ebx : 0x0806eb90

A screen shot of a computer code

Description automatically generated

Địa chỉ của/bin/sh : 0x080be408

A screenshot of a computer

Description automatically generated

Địa chỉ của int0x80: 0x08049421

A screen shot of a computer

Description automatically generated

Đặt break ở hàm get

A computer screen with white and orange text

Description automatically generated

Return address sau khi gọi hàm get là ebp + 4 : từ vị trí lưu stack đến ebp+4 : 0xffffcff8 +4 – 0xffffcf8c = 112

A computer screen shot of a program

Description automatically generated

Kết quả:

A screen shot of a computer screen

Description automatically generated

***Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này***

# **YÊU CẦU CHUNG**

* Sinh viên tìm hiểu và thực hiện bài tập theo yêu cầu, hướng dẫn.
* Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
* Sinh viên nộp bài theo thời gian quy định trên course.

**Báo cáo:**

* File **.PDF**. Tập trung vào nội dung, không mô tả lý thuyết.
* Đặt tên theo định dạng: **[Mã lớp]-AssignmentX\_NhomY** (trong đó X là Thứ tự Assignment, Y là số thứ tự nhóm đồ án theo danh sách đã đăng ký).

*Ví dụ: [NT521.O11.ATCL]-Assignment01\_Nhom03.pdf.*

* Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.

**Đánh giá**:

* Hoàn thành tốt yêu cầu được giao.
* Có nội dung mở rộng, ứng dụng.

*Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.*

**HẾT**