BÁO CÁO THỰC HÀNH

**Môn: Lập trình hệ thống**

**Buổi báo cáo:** Lab 04

**Lớp:** NT209.O22.ANTT.2

**THÔNG TIN CHUNG**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Họ và Tên | MSSV | Lớp |
| 1 | Trần Tuấn Anh | 22520080 | ATTT2022.1 |
| 2 | Nguyễn Khắc Hậu | 22520410 | ATTT2022.1 |

**Báo Cáo Chi Tiết**

Câu 1

Ảnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động



Đây là đoạn code của hàm xử lý hardcode của câu 1.

Input đầu vào được lưu vào địa chỉ [ebp+s1].

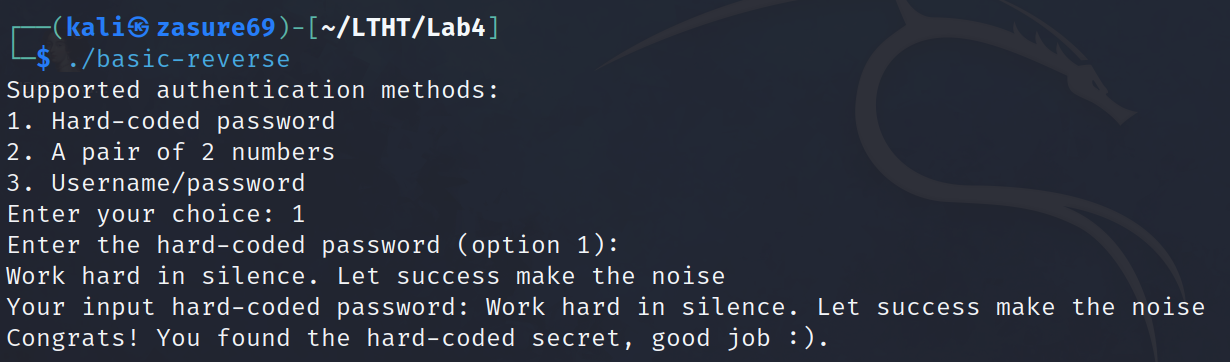
Sau đó câu lệnh push offset s2 đẩy chuỗi s2 vào stack.

Chuỗi input được đẩy vào stack và gọi hàm strcmp để so sánh.

Hàm success\_1 sẽ được gọi nếu ZF = 1 (được gán bằng câu lệnh test eax, eax), ngược lại nếu ZF = 0, lệnh jnz sẽ nhảy đến nhãn loc\_80486C9 và gọi hàm failed.

Để ZF = 1 🡪 strcmp trả về 0 🡪 2 chuỗi so sánh phải bằng nhau 🡪 Chuỗi cần tìm là s2 = “Work hard in silence. Let success make the noise”.

Kết quả thực thi:



Câu 2

Hàm xử lý của câu 2 là otherhardCode

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình

Mô tả được tạo tự động

2 số input đầu vào sẽ được lưu ở 2 ô nhớ [ebp+var\_14] và [ebp+var\_10]

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động



Ta thấy ô nhớ [ebp+var\_C] được gán giá trị 3, và thanh ghi eax được gán giá trị của input thứ 1.

Lệnh cmp eax, [ebp+var\_C] so sánh 2 giá trị input 1 và 3 nếu không bằng sẽ gọi hàm failed 🡪 **input 1 sẽ là 3**.

Hàm funny\_func được gọi và truyền 2 tham số là input 1 và ds:funny\_seq[3\*4] (eax được gán giá trị input 1)

Ảnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động

Phần tử funny\_seq[3\*4] = 8

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Đây là nội dung hàm funny\_func sẽ trả về kết quả biểu thức

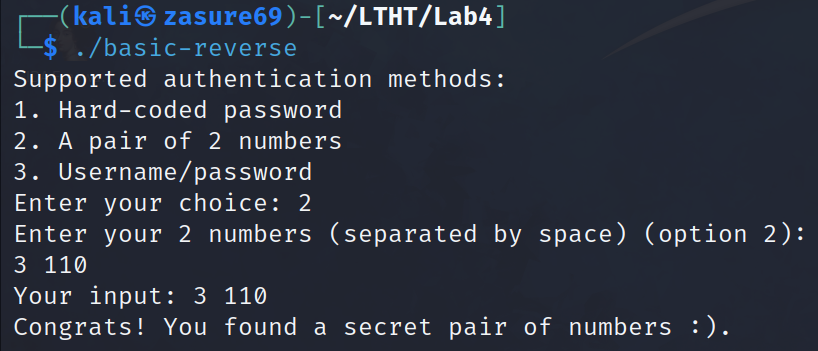
(arg\_0 + arg\_4 - 1) \* (arg\_0 + arg\_4) mà arg\_0 và arg\_4 lần lượt là 8 và 3 🡪hàm trả về 110.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

Sau khi gọi hàm funny\_func sẽ so sánh giá trị trả về từ hàm funny\_func với input 2 nếu không bằng nhau sẽ gọi hàm failed 🡪 **input 2 = kết quả trả về của hàm funny\_func = 110**.

Kết quả thực thi:



Câu 3:

Hàm xử lý của câu 3 là hàm userpass

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

2 input username, password sẽ được lưu lần lượt vào 2 ô nhớ [ebp+s] và [ebp+var\_25].

Và username = 0080-0410.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

Đoạn lệnh này kiểm tra độ dài của username và password có độ dài bằng 9 không nếu không sẽ gọi hàm failed nếu không sẽ nhảy đến nhãn loc\_804882B 🡪 password có độ dài bằng 9.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

Gán 0 vào ô nhớ [ebp+var\_C] 🡪 nhảy đến nhãn loc\_804888E.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

Kiểm tra giá trị ô nhớ [ebp+var\_C] <= 8 🡪 nhảy đến nhãn loc\_804883B

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Kiểm tra giá trị ô nhớ [ebp+var\_C] < 1🡪gán kí tự thứ 3 (kí tự ‘8’) của username vào ô nhớ [ebp+var\_2E] ([ebp+var\_C] = 0) 🡪nhảy đến nhãn loc\_804888A.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

Tăng giá trị ô nhớ [ebp+var\_C] lên 1 và tiếp tục vòng lặp từ nhãn loc\_804888E.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Sau khi chạy đoạn lệnh này ô nhớ [ebp+var\_2E +1] ([ebp+var\_C] = 1) được gán giá trị bằng kí tự thứ 4 của username (kí tự ‘0’)

Ở lần lặp tiếp theo, [ebp+var\_C] = 2 > 1 🡪 nhảy đến nhãn loc\_8048858.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Sau khi thực thi đoạn lệnh này khi [ebp+var\_C] = 2 và [ebp+var\_C] = 3, ô nhớ [ebp+var\_2E+2] và [ebp+var\_2E+3] sẽ được gán bằng ký tự thứ 8 và thứ 9 của chuỗi username (kí tự ‘1’ và ‘0’).

Ở lần lặp tiếp theo, [ebp+var\_C] = 4 > 3 🡪 nhảy đến nhãn loc\_8048875.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Sau khi thực hiện lệnh ô nhớ [ebp+var\_2E+4] sẽ được gán bằng giá trị trong ô nhớ [ebp+var\_11] ([ebp+var\_C]-4 = 0). Giá trị trong ô nhớ [ebp+var\_11] bằng 0x42 hay kí tự ‘B’.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình

Mô tả được tạo tự động

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, chữ viết tay

Mô tả được tạo tự động

Ở các vòng lặp tiếp theo, khi [ebp+var\_C] = 5, 6, 7, 8 các ô nhớ [ebp+var\_2E+5], [ebp+var\_2E+6] , [ebp+var\_2E+7], [ebp+var\_2E+8] sẽ được gán các giá trị trong các ô nhớ [ebp+var\_11+1] hay [ebp+var\_10], [ebp+var\_11+2] hay [ebp+var\_F], [ebp+var\_11+3] hay [ebp+var\_E], [ebp+var\_11+4] hay [ebp+var\_E], [ebp+var\_11+5] hay [ebp+var\_D] với các kí tự ‘/’, ‘C’, ‘F’, ‘Y’.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

Sau khi kết thúc vòng lặp [ebp+var\_C] > 8, chuỗi được lưu trong bộ nhớ (tạm gọi s) là “8010B/CFY”. [ebp+var\_C] được gán bằng 0 và nhảy đến nhãn loc\_80488DC.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Kiểm tra [ebp+var\_C] = 0 < 9 🡪 nhảy đến nhãn loc\_804889D.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

Đoạn lệnh này lấy 2 kí tự thứ [ebp+var\_C] ở 2 chuỗi username và chuỗi được lưu trong bộ nhớ s = “8010B/CFY” cộng lại và shift phải 1 bit. Sau đó so sánh với kí tự thứ [ebp+var\_C] trong chuỗi password nếu không bằng nhau thì hàm failed sẽ được gọi như đoạn lệnh bên dưới.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

🡪 password[i] = (username[i] + s[i]) >> 1

hay (username[i] + s[i]) / 2.

password[0] = (‘0’ + ‘8’) / 2 = ‘4’

password[1] = (‘0’ + ‘0’) / 2 = ‘0’

password[2] = (‘8’ + ‘1’) / 2 = ‘4’

password[3] = (‘0’ + ‘0’) / 2 = ‘0’

password[4] = (‘-’ + ‘B’) / 2 = ‘7’

password[5] = (‘0’ + ‘/’) / 2 = ‘/’

password[6] = (‘4’ + ‘C’) / 2 = ‘;’

password[7] = (‘1’ + ‘F’) / 2 = ‘;’

password[8] = (‘0’ + ‘Y’) / 2 = ‘D’

Sau khi áp dụng công thức ta tính ra được password = “40407/;;D”

Kết quả thực thi:

