Tool Developer Qualification Course

Reverse Engineering project

Bomb

Joshua Abraham

Taught by by
Dr. Joseph Santmeyer

November 17, 2017

Contents

1 Summary **2**

2 Stage 1 Analysis **3**

3 Stage 2 Analysis **4**

1 Summary

The Fibonacci project has been a challenging experience. Lack of high-level loops, conditionals, etc. made implementing a program that calculates the Fibonacci sequence hard. Having to calculate up to the 500^{th} number (an absurdly large value) made it even harder. However, this project was a resounding success. All requirements were accomplished, with minimal C function calls. The program is fast, easy to read, and well documented. I learned a great deal during development; one of the most practical lessons was being able to translate high-level C into assembly almost line by line. If I had more time, I would spend it on removing my call to *printf* entirely and use 100% assembly.

1. Stage 1: swordfish
2. Stage 2: jabraham
3. Stage 3: 1872
4. Stage 4: 107 214 428 856 1713
5. Stage 5: 1172
6. Stage 6: 48 a 48
7. Stage 7: (press enter)
8. Stage 8: (run ./stag8.sh in the same directory as the binary, then press enter).

2 Stage 1 Analysis

Stage 1 is straightforward. Below is an abbreviated listing of the relevant disassembled code:

```
mov     esi, 0x4018f8    ; "swordfish"
mov     rdi, rax         ; user input
call    strcmp
test    eax, eax
setz    al
leave
```

This stage calls strcmp with user input and the string "swordfish". If the return value is zero (meaning the strings are identical), this stage returns non-zero and the program proceeds.

3 Stage 2 Analysis

Stage 2 starts with a call to getenv. This function takes one parameter and returns the value of a specified environment variable.

```
mov     edi, 0x401902     ; "USER"
call    getenv
mov     QWORD PTR [rip+0x201264], rax
```

The user name is stored in a global variable that is used throughout the binary at rip+0x201264. In my case the user name is "jabraham". This and the saved user input are passed to strcmp, similarly to Stage 1.

```
mov     rax, [rbp-0x8]
mov     rsi, rdx          ; "jabraham"
mov     rdi, rax          ; "jabraham"
call    strcmp
test    eax, eax
setz    al
leave
```

After the call to strcmp, the program checks the return value in eax and sets al if the zero flag is set.