Project 2 Write-Up

Michael Oles

Mbo10@pitt.edu

1 Program 1

1.1 Procedure

- I knew that this program must have a main if it was written in C, so I placed the breakpoint there and disassembled it. Looking at the call instructions, I saw the chomp seemed to be the only notable function besides the get and put and print functions. I noticed the low byte of EDX and EAX being compared but was unsuccessful in finding out how they were set so I looked at chomp. Inside I tried using "I r" to track the variables throughout chomp but didn't see the password stored. Then I ran "strings mbo10_1" and looked through that until I found familiar strings saying correct and incorrect password. Looking around that I noticed a long string which seemed out of place and tried that in the code

1.2 Solution

- That string "UgnkkRKTkLFAUsrtTxfVsyShlMZ" was the correct password for program one.

1.3 Notes

- After solving it I found that the answer was stored in ESI and I should have been focusing on the Repnz command which did the comparison
- When tracing the file I also noticed that C contained the size of the user input

2 Program 2

2.1 Procedure

- I knew that this program must have a main if it was written in C, so I placed the breakpoint there and disassembled it again. I saw that the entire program ran inside of a function called d so I disassembled that. There were three more unknown functions c, r, and s. I noticed that C called S three times and R also called S once. I tried looking in these functions and found that S was looping and making a count which seemed to be relating to the size of the function. After spending hours on this with no luck, I decided to trace backwards instead. I saw that to get to what I assumed was the print statement for a success, EAX needed to not be zero. I found that EAX was set in R. For EAX to not be set to zero, it had to be equal to DL. Printing out DL, I found that it had the ASCII value of the last letter in the input this had to be equal to EAX which contained the first letter of the input so the last and first letters must be the same. I tried

inputting “h” and this got past that step but then failed in D because eax was not greater than 10. I then figured out that R checked the first and last letters multiple times moving inward and needed more than 10 letters to get EAX greater than 10.

2.2 Solution

- Knowing that the string had to be larger than 10 and going inward the first an last letters must be the same I tried “HHHHHHHHHHH” which worked. Knowing how it worked I was able to get other strings to work such as “abcdeffedcba” to work as long as they were symmetrical, even numbered, and longer than 10 characters.

2.3 Notes

- I learned a lot of new strategies for tracing a program
- I learned how jumps work by making a comparison or test before the jump, setting that value in a flag and jumping in the next statement which is different from jumps in MIPS

3 Program 3

3.1 Procedure

- The first problem with program 3 was locating the main. B main didn’t work so I tried an objdump. After that I tried setting a breakpoint at a few addresses that had familiar commands until finally finding the main by setting the breakpoint at __libc_start_main

3.2 Solution

-

3.3 Notes

-