CS 3844 Computer Organization - Lab #05 Name/abc123: Ariel Guerrero aes604

Now you should be at the hand-written assembly.

1. What is the value of eax after executing the xor instruction?

   xor eax,eax => eax = 0

   (xor anything by itself and its 0)

2. Step down to the je NOT_FOUND instruction (jmp if equal to zero). Google the test and je instructions and see what they do. Then apply that to test esi,esi, je NOT_FOUND.

[TEST] computes the bit-wise logical AND of first operand (source 1 operand) and the second operand (source 2 operand) and sets the SF, ZF, and PF status flags according to the result. The result is then discarded.

Je jumps if zf is set

   a. What is the value of searchWord? (may vary from machine to machine)

      address: 0x008CFE74 value: computer

   b. What is the offset from ebp of searchWord? (This will ALWAYS be the same. The offset is in the machine code, BUT from the video, you can tell what it is without looking at the machine code.)

      ebp+12

   c. What value do you expect the ZERO flag to be after the test instruction? Why?

      0 because if you and something with itself all the bits align. Therefore the zero flag is not set

   d. Will the je be taken? Why or why not? ("taken" means it will jump to the code at NOT_FOUND rather than execute the next instruction)

      no because the zero flag is not set

   e. In a memory window, type in &searchWord.
      i. What is the address of searchWord? (may vary)

         address of &searchWord: 0x0018FE48

      ii. What is the value at that address? (may vary)

         value @ &searchWord: 008CFE74

      iii. Subtract ebp from the address of searchWord. The result will be the same for everyone and should equal your answer to #b. above. What is the result?

0018FE48
- 0018FE3C

C → **12**

f. The value of searchWord (#ii. above) is an address, so type "searchWord" in a memory window. What is in memory at that location?

address of searchWord: 0x008CFE74
value @ searchword: computer

3. Single step to the next je instruction. It is a repeat of the prior set of instructions except this one is checking string2Search rather than searchWord.
   a. What is the value of the offset from EBP for string2Search? (Again, it's in the machine code but you should know based on the function call.)

      value of offset for ebp is ebp+8

   b. Based on the test instruction and the value of edi, will the jump be taken?

      No

   c. In a memory window, type in string2Search. What is in memory at that location?

      address string2Search: 0x004AE000

   d. Make a high level comment to describe what this series of instructions does. Consider that the function take two addresses as arguments. This functionality can be described in on brief sentence.

      Checks to see if the strings are empty

4. Neither of the jumps (je == jump if equal) are taken. Single step to the move instruction. Now figure out what you need to do to answer these questions:
   a. What value do you expect to be in al after the move?

      Fisrt character in searchWord, for me it was 'c'

   b. What is the test al,al instruction doing? Consider that searchWord/searchString are C style strings.

      Checking if they are empty

5. Single-step to the first instruction in MAIN_LOOP:.
   a. What do you expect al to equal after the "`mov al,byte ptr [edi]`" instruction?

      The character 'A' from string2Search (remember we moved it into edi previously)

b. Based on the value of al and the test al,al instruction, will the je be taken?

No

6. Single-step to the "`cmp al,bl`" instruction.
   a. What are the values of al and bl?

      AL: c
      BL: A

   b. The cmp instruction will do this: al – bl and set flags. The difference is not stored. (If you want to store the difference, use sub al, bl.) Based on the values of al and bl, which flags will be set/cleared?

      97 – 65 = 32 = 20h

      ZF not set
      OF not set
      SF not set
      CF not set
      PF not set

   c. Will the jne NEXT_CHAR be taken?

      Yes 'c' does not equal 'A'

7. Set a BP on the "`call CHECK_THE_WORD`" instruction. Press "continue" (the little green debugging arrow) and execution will stop at that call.
   a. How many parameters are pushed?
      Return address, edi (word to search) and esi (search word)

      in my case edi gets incremented once before jumping to this label and now starts with the word 'computer' instead of A
      edi = 004AE002

   b. What is the return address that will be pushed to the stack once the call is made?

      00434C43

   c. Step into the call. Show the top of the stack as it appears in memory?

      43 4C 43 00 34 FF 18 00

      Return address, and 0018FF34

      couldnt figure out what the 0018FF34 was but I know that its by &searchString &string2Search and &wordLength in memory

8. Comment the code. You don't need to comment each line, but you can group in blocks since several instructions may perform a single task. I've grouped it for you this time.

```
CHECK_THE_WORD:
0042E72B 57               push        edi
0042E72C 56               push        esi
```

storing our variables here stored in edi and esi
edi contains string2Search esi contains searchWord
onto the stack

```
0042E72D 66 8B 4D 10      mov         cx,word ptr [wordLength]
0042E731 0F B7 C9         movzx       ecx,cx
```

we store the word length in cx
then store that value in ecx zero extended since its unsigned

---

```
0042E734 B0 01            mov         al,1
```

what ever al is its now 01

```
0042E736 49               dec         ecx
0042E737 74 13            je          EXIT_CHECK_THE_WORD (42E74Ch)
```

decrement ecx by 1 and if it is 0 jump to the label
EXIT_CHECK_THE_WORD

Under what condition will the above "je" be taken? This ties into why we set al == 1 (true)
if the words are only length 1

```
WORD_CHECK:
0042E739 46               inc         esi
0042E73A 47               inc         edi
```

going to the next character of both strings we are checking

```
0042E73B 8A 07            mov         al,byte ptr [edi]
0042E73D 8A 1E            mov         bl,byte ptr [esi]
```

al has a char in string2Search and bl has a character from
search string

```
0042E73F 3A C3            cmp         al,bl
0042E741 75 07            jne         NO_MATCH (42E74Ah)
```

compares the 2 characters to see if they are the same and if they are
not then the code jumps to the NO_MATCH label

```
0042E743 49               dec         ecx
0042E744 75 F3            jne         WORD_CHECK (42E739h)
```

decrement ecx since we need to compare the next characters
and jump back to the top of the label unless ecx = 0

```
0042E746 B0 01            mov         al,1
0042E748 EB 02            jmp         EXIT_CHECK_THE_WORD (42E74Ch)
```

if we are here then exc =0 now we jump to EXIT_CHECK_THE_WORD

```
NO_MATCH:
0042E74A 32 C0            xor         al,al
```

```
al = 0
```

```
EXIT_CHECK_THE_WORD:
0042E74C 5E                 pop         esi
0042E74D 5F                 pop         edi

reset the stack popping off esi and edi to restore them

0042E74E C3                 ret

return to the callers next instruction
```

9. Put a BP on the "return result" right after the label "DONE:" Continue until you hit that BP. What is the return value? (i.e. the value of "result") NOTE: this is NOT the return address. The return value is, by convention, in al, ax, or eax depending on the size of the return value.

   True = 1 = al

10. Figure out where you can put a break point to see the return address before the function returns. What is that return address? (may vary)

    0018ff34