The "Cybersecurity" Specialization

Learn More

×

Feedback — VM BOF quiz

Help

You submitted this homework on **Sun 9 Nov 2014 12:01 PM PST**. You got a score of **31.00** out of **31.00**.

Complete this quiz when you have completed project 1. The questions for the quiz were presented in the description of the project, so you should just have to enter your answers here. To make sure your answers match, avoid spurious whitespace.

Question 1

There is a stack-based overflow in the program. What is the name of the stack-allocated variable that contains the overflowed buffer?

You entered:

wis

| Your Answer | | Score | Explanation |
|----------------|----------|--------|--|
| wis | ~ | 3.00 | The wis variable is allocated on the stack and can get overflowed by the call to gets. |
| Total | | 3.00 / | |
| | | 3.00 | |

Question 2

Consider the buffer you just identified: Running what line of code will overflow the buffer? (We want

the line number, not the code itself.)

You entered:

62

Your Answer Score Explanation

62 ✓ 3.00 Due to the gets overrunning the target buffer

Total 3.00 / 3.00

Question 3

There is another vulnerability, *not dependent at all on the first*, involving a *non*-stack-allocated buffer that can be indexed outside its bounds (which, broadly construed, is a kind of buffer overflow). What variable contains this buffer?

You entered:

ptrs

| Your Answer | | Score | Explanation |
|----------------|----------|--------|--|
| ptrs | ~ | 3.00 | Note that I->data can be overflowed, but only if wis is overflowed, so it would be dependent on the first. |
| Total | | 3.00 / | |
| | | 3.00 | |

Question 4

Consider the buffer you just identified: Running what line of code overflows the buffer? (We want the number here, not the code itself.)

Your Score Explanation 101 Your 3.00 This overflow happens by allowing the index variable to be too large. Properly, this is at line 102, but you can make arguments that earlier lines set this up to happen. Total 3.00 / 3.00

Question 5

What is the address of buf (the local variable in the main function)? Enter the answer in either hexadecimal format (a 0x followed by 8 "digits" 0–9 or a-f, like 0xbfff0014) or decimal format.

Note here that we want the address of buf, not its contents.

You entered:

0xbffff130

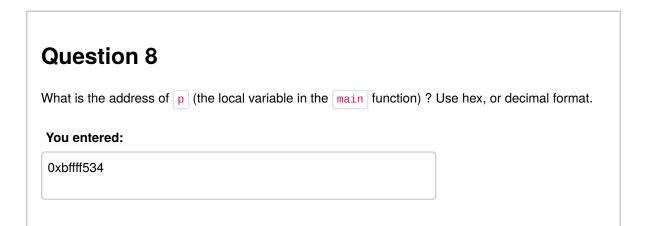
| Your Answer | | Score | Explanation |
|-------------|---|-------------|--|
| 0xbffff130 | ~ | 1.00 | break at wisdom-alt.c:100 and print &buf |
| Total | | 1.00 / 1.00 | |

Question 6

What is the address of ptrs (the global variable) ? As with the previous question, use hex or decimal format.

Your Score Explanation Answer Ox804a0d4 ✓ 1.00 Again, at the first breakpoint you can print &ptrs to get the answer Total 1.00 / 1.00

Question 7 What is the address of write_secret (the function) ? Use hex or decimal. You entered: 0x8048534 Your Answer Score Explanation 0x8048534 ✓ 1.00 Easy: Print & write_secret from gdb Total 1.00 / 1.00



| 0xbffff534 ✓ 1.00 Same drill as the earlier questions |
|---|
| |
| Total 1.00 / 1.00 |

Question 9

What input do you provide to the program so that ptrs[s"] reads (and then tries to execute) the contents of stack variable p instead of a function pointer stored in the buffer pointed to by ptrs?

As a hint, you can determine the answer by performing a little arithmetic on the addresses you have already gathered. If successful, you will end up executing the pat_on_back function. Provide the smallest positive integer.

You entered:

771675416

| Your Answer | | Score | Explanation |
|----------------|---|----------------|--|
| 771675416 | ~ | 4.00 | This is the result of doing (unsigned int)((int *)&p - (int*)&ptrs) in gdb. Note that doing (unsigned int)&p - (unsigned int)&ptrs) won't work because the difference will be in bytes, not pointer-sized words. We need the difference to be in words so using s in ptrs[s] does the right thing. |
| Total | | 4.00 / 4.00 | |

Question 10

What do you enter so that ptrs[s"] reads (and then tries to execute) starting from the 65th byte in buf, i.e., the location at buf[64]? Enter your answer as an (unsigned) integer.

Your Score Explanation 771675175 ✓ 2.00 (unsigned int)((int *)&buf[64] - (int *)&ptrs) in gdb. Same process as the previous question, but now you are using a different starting point. Total 2.00 / 2.00

Question 11 What do you replace \xEE\xEE\xEE with in the following input to the program (which due to the overflow will be filling in the 65th-68th bytes of buf) so that the ptrs[s] operation executes the write_secret function, thus dumping the secret? (Hint: Be sure to take endianness into \xEE\xEE You entered: \x34\x85\x04\x08 Your **Score Explanation Answer** 4.00 \x34\x85 This is the address of write_secret, which is 0x08048534, but \x04\x08 entered in hex bytes and accounting for little endianness 4.00 / Total

6 of 8 11/09/2014 03:01 PM

4.00

Question 12

Suppose you wanted to overflow the wis variable to perform a stack smashing attack. You could do this by entering 2 to call put_wisdom, and then enter enough bytes to overwrite the return address of that function, replacing it with the address of write_secret. How many bytes do you need to enter prior to the address of write_secret?

You entered:

148

| Your Answer | | Score | Explanation |
|----------------|----------|----------------|--|
| 148 | ~ | 5.00 | This number comes from the following calculation: 128 bytes for the buffer; 12 bytes for the three local variables (r, l, and v); 4 bytes for saved EBP; 4 bytes for saved EDI; And finally the return address, to be overflowed |
| Total | | 5.00 / 5.00 | |

https://class.coursera.org/softwaresec-001/qu...