overflowing put-wisdom's local var wis that takes the wisdom entry question

You are subscribed. Unsubscribe UNRESOLVED

project1 × + Add Tag Sort replies by: Oldest first Newest first Most popular

Karen West . 2 hours ago

Using the GDB command: x/48xw $esp, after entering the put_wisdom function, I noticed that after the local variable wis that takes the wisdom input (128 bytes in size allocated on stack), that the return address to main()'s line 102 (tmp() - the line that called put_wisdom() when we entered 2 into buf), that return address was at word 48, and the wis local variable within the stack allocation ended 10 words before the return address of tmp() in main. I noticed before wis was allocated on the stack, that $esp pointed to this address, but after wis was allocated, it was located 10 words beyond wis. I noticed the local variable to put_wisdom, r, is located just past the end of the 128 bytes of wis. So my entry to wis to overwrite the return address of tmp() was done where originally I saw that address (0x804880d) and I replaced it with write_secret's address written little endian style (0x8048534) so after 128 bytes of wis, I counted 10 words ahead where the return address of tmp() was on stack and entered it by: \x34\x85 \x04\x08, but this caused a seg fault. Any help appreciated, that does not violate the honor code, since it's Sat. at noon time, and it's due Mon. by 8am (the extended due date!) I got started just this week, since I almost dropped the class when I could not get the VM to work on my machine, and then, last minute, someone helped, and I was able to attempt the project and stick with the course. Thanks!

0 · flag

Carsten Hansen . 2 hours ago

Does your analysis agree with what I posted here https://class.coursera.org/softwaresec-001/forum/thread?thread_id=118#comment-846.
If you enter only 771675175\x00, can you in gdb manually overwrite the return address to e.g. 0x8048534 and get the program to print the secret?

0 · flag

 Karen West · an hour ago

Carsten - isn't that the previous question? (the one where we over write bytes 65-68 of buf with write_secret?) This one was posted when we select option 2 to enter wisdom, and then overwrite the wis local variable (a 128 byte buf) so that the return address is write_secret when it attempts to go back to main that called it. Weren't we instructed in the previous question to write the 771675175\x00(56 A's)\x34\x85\x04\x08? I thought in this question we were overwriting the return address for put_wisdom, and I"m confused as to how the entry to buf you wrote above is related to this question.

I did see the stack you printed though in the link, and saw that you were saying that after wis[128]'s allocation on the stack, that the return address follows 6 words later, not 10 words? It did not appear that way when I printed the stack within put_wisdom when I was running gdb, since I saw the tmp() function from main 10 words ahead of where wis[128] was allocated upon entry to put_wisdom.

↑ 0 ↓ · flag

+ Comment

 Karen West · just now

I got this one to correctly print the secret key and you were correct, it did seg fault, but did print the secret key anyway.
I had counted the number of bytes from where wis ended on the stack to where you write the return address. Thank you for your help.

↑ 0 ↓ · flag

+ Comment

New post

To ensure a positive and productive discussion, please read our forum posting policies before posting.

B I Link <code> Pic Math Edit: Rich Preview

Resolve thread

This thread is marked as unresolved. If the problem is fixed, please check the above box and make a post to let staff know that they no longer need to monitor this thread.

Make this post anonymous to other students

Subscribe to this thread at the same time

Add post