Lab3:AttackLab

Junhao Yu

April 25, 2022

1 Problems I Met and How I Solved Them

1.1 Level 1

Since function touch1 does not have any parameters, I just need to put the touch1 address at the return address, and the getbuf will call touch1.

1.2 Level 2

Since function touch2 has a parameter which has to match with cookie, I first write assembly code which moves cookie to %rdi and pushes the touch2 address on the top of the stack. Also, I put the move operation's address at the return address so that getbuf function will first call move operation and then call touch2.

1.3 Level 3

Function touch3 resembles touch2, hence the solution is basically the same. But there are two differences: first, since function hexmatch uses the stack, thus I have to put the cookie higher, where it won't be overwritten; second, the parameter should be cookie's address.

1.4 Level 4

The main problem of this target is that the stack is inexecutable, thus I have to use the gadget to achieve the move operation. With hints in the handout, I carefully searched for pop and move operation since "pop" can move the information in stack to register and "move" can move cookie to %rdi.

1.5 Level 5

To attack this target, I fisrt carefully listed all the "move" operation available and drew the stack frame so that I could figure out where the cookie should be stored. Then all I have to do is transfer the cookie's address via a roundabout route to %rdi. Also, since this time the stack was randomized, I had to use address shift to make sure the right address were obtained.

2 What I Have Learned

1. Get a better knowledge of how the function call is executed and stack operations such as pop, push, ret.
2. Get a better knowledge of how to use simple buffer overflow and complex gadget to execute the code that I want to utilize.