Stack

Alföldi Mátyás

June 25, 2020

Contents

stack0

This challange is relative simple, the following code is given:

```
int main(int argc, char **argv)
{
  volatile int modified;
  char buffer[64];
  modified = 0;
  gets(buffer);

  if(modified != 0) {
    printf("you have changed the 'modified' variable\n");
  } else {
    printf("Try again?\n");
  }
}
```

I just gave the following input for this:
10xa 10xb 10xc 10xd 10xe 10xf 10xg (So basically something that is longer than 64)

stack1

Source code:
https://web.archive.org/web/20170419031559/https://exploit-exercises.com/protostar/stack1/
This time the argument has to be longer than 64, and 0x61626364 has to go into the variable, which is the ascii code for abcd, but because of the endianness has to be reversed.

stack2

Source code:
https://web.archive.org/web/20170419023252/https://exploit-exercises.com/protostar/stack2/
Answer:
Same as above but \n \r \n \r has to be used.

stack3

Source code:
https://web.archive.org/web/20170417130221/https://exploit-exercises.com/protostar/stack3/
For this one we use objdump -d to get the address of the win function, which seems to be 08048424.

Using /bin/echo -e -n '64 filler chars\x24\x84\x04\x08' | ./stack3 does the work

stack4

Source code:
https://web.archive.org/web/20170417130121/https://exploit-exercises.com/protostar/stack4/
During the call to main the location to where the ret should return is pushed into the stack, so after checking with objdump the address of the win function a long enough input has to be generated so that we overwrite that.
win is at: 080483f4
Looking at main the following instructions modify esp:

- push ebp
- and esp, 0xfffffff0
- sub esp, 0x50

And the following changes how long our filler string has to be:

- lea eax,[esp+0x10]
- mov DWORD PTR [esp],eax

This way esp+0x10 gets passed to the gets function, and this means that the buffer is 64 long.
The push ebp modifies esp by 4, and the and esp, 0xfffffff0 by a max of 15, so 64 + 15 + 4 should be the max len needed.
So tests should be done with filler string len 68-84
In the end 76 x a and 080484f4 reversed for the endiannes(in hex) was the amount needed to change where the ret brings us.