全国大学生信息安全竞赛writeup--careful(pwn150)

原创

Anciety 于 2016-07-10 21:40:32 发布 2862 收藏 2

分类专栏： ctf C/C++ linux基础 文章标签： 信息安全 汇编

描述

其实没有描述。。就是一个文件，careful.
打开发现输入index和value，10个之后会打出来一个数组。

思路

先逆向，ida看看。
main函数很简单，return一个init_array。
到init_array里边就比较好玩了，不想看汇编直接f5，大致函数如下

```
int initarray()
{
    int result;
    char s[20];
    int v2;
    int v3;
    int i;
    memset(s, 0, 0x14u);
    for (i = 0;i <= 9;i++)
    {
        printf("input index:");
        fflush(stdout);
        scanf("%d", &v3);
        printf("input value:");
        fflush(stdout);
        scanf("%d", &v2);
        result = v3;
        s[v3] = v2;
        if (i > 10)
            return result;
    }
    result = printf("Your Array:");
    for (i = 0;i <= 9;i++)
    {
        printf("%d ", s[i]);
        result = fflush(stdout);
    }
    return result;
}
```

(手打的，因为在虚拟机里边复制不出来，可能跟原文不太一样)
算法比较简单，就是读入index和value，把value存入相应的index对应的位置。
存入的汇编:

```
mov eax, [ebp+var_10];index
mov edx, [ebp+var_14];value
mov [ebp+eax+s], dl;这里的s即-0x28,是存s的地方
```

到这里基本上就可以看出来了，先读入index和value，然后存入s数组，但是由于没有下标判断，所以造成任意位置写，但是问题来了，写可以改变返回地址，改变运行过程，可是改变到哪儿呢。gdb打开之后checksec发现开了NX(堆栈不可执行)，所以写入shellcode直接执行的想法肯定是不行了，不过需要注意观察，有一个函数是没有使用的。

通过观察，发现位于0x08048615处有一个add函数，没有使用到，而且输出一个字符串后使用system调用调用date。即system(“date”)。 既然有了system，剩下的就好办许多了，通过一个工具，即rop-tool(本来是用来写ROP的，不过这些时候也可以用一用)，可以查找字符串等，rop-tool search –string sh PROG ，可以查找到程序里的sh。

```
R-X 0x0804828e -> sh
 1 strings found.
 0 strings found.
```

于是我们找到了sh字符串，不过还需要把他放到esp的地址处作为system调用的参数，esp地址处是函数调用的参数的地址，所以需要把sh的地址写到调用system时的esp处。

剩下的工作:

1. 改变esp指向的值，指向sh
2. 改变保存的eip，指向system调用

通过在gdb下加断点，断到initarray，可以看到ebp的值为0xffffcf28，在0xffffcf2c处记录了返回地址，覆盖掉返回地址，用一个字节即可，通过计算0xffffcf2c和0xffffcf28-0x28的偏移，得到index应该为44，覆盖为46（2e）,因为返回值本来保存的是0x08040640,而system位于0x0804062e。再通过单步执行一次，到system时发现esp的值为0xffffcf30，于是用4个字节覆盖掉0xffffcf30的值，覆盖为sh的地址，就成功打开了shell。