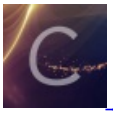
【CTF刷题之旅】XCTF嘉年华体验赛逆向题re2的writeup

原创

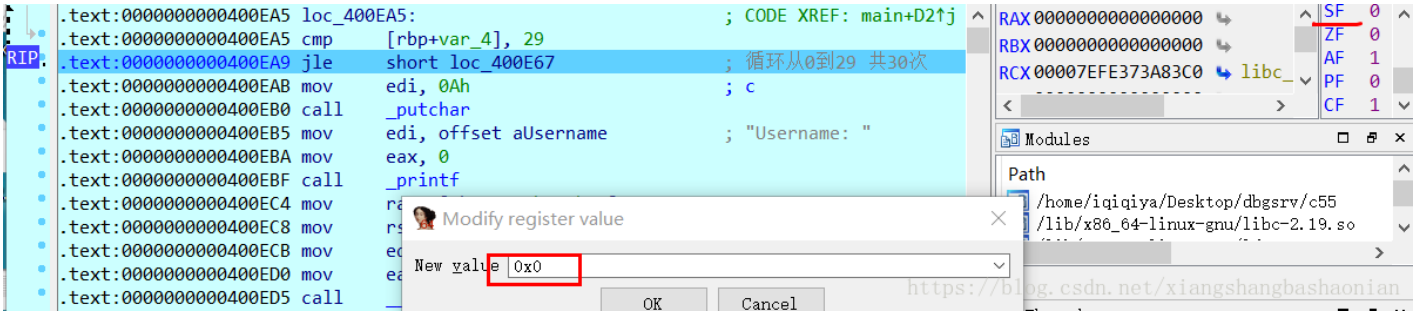
iqiqiya 于 2018-10-22 23:56:11 发布 987 收藏
分类专栏： 我的逆向之路 我的CTF之路 ------XCTF嘉年华体验赛 我的CTF进阶之路 文章标签： 【CTF刷题之旅】XCTF嘉年华体验赛逆向题re1的writ re1的writeup XCTF嘉年华体验赛逆向题 reverse



这道题采用动静结合的方法尝试了一下

动态调试的时候想加快进度 跳过sleep() 遇到那两个jle跳转 直接修改SF标志位为0来修改执行流程 可以看到 左侧箭头会变成虚线

对了 第三个jle在sub_400C9A()中



main()函数如下

```
__int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
  unsigned int v3; // eax
  int v4; // eax
  int v5; // eax
  void *my_mima; // [rsp+8h] [rbp-18h]
  unsigned int *my_zhanghao; // [rsp+10h] [rbp-10h]
  signed int j; // [rsp+18h] [rbp-8h]
  signed int i; // [rsp+1Ch] [rbp-4h]

  my_zhanghao = malloc(0x3E8uLL);    #申请分配内存空间给我们的输入
  my_mima = malloc(0x3E8uLL);
  v3 = time(0LL);
  srand(v3);    #srand()用来设置rand()产生随机数时的随机数种子，参数seed必须是整数，通常可以用time(0)的返回值作为see
  puts(a31m);    #红条出现
  puts(a33m);    #黄条出现 下面以此类推
  puts(a32m);
  puts(a36m);
  puts(a34m);
  puts(a35m);
  puts(a34m_0);
  puts(a36m);
  puts(a32m_0);
  puts(a33m_0);
  puts(a31m_0);
  puts("\x1B[0mWelcome to Catalyst systems");
  printf("Loading", a2);
  fflush(stdout);
  for ( i = 0; i <= 29; ++i )        #这里就是等待 输出30个"."
  {
    v4 = rand();
    sleep(v4 % (3 * i + 1));
    putchar('.');
    fflush(stdout);
  }
  putchar(10);
  printf("Username: ");
  __isoc99_scanf("%s", my_zhanghao);    #输入账号
  printf("Password: ", my_zhanghao);
  __isoc99_scanf("%s", my_mima);    #输入密码
  printf("Logging in", my_mima);
  fflush(stdout);
  for ( j = 0; j <= 29; ++j )    #又是等待30个"."
  {
    v5 = rand();
    sleep(v5 % (j + 1));
    putchar('.');
    fflush(stdout);
  }
  putchar(10);    #下面这五个函数是得到我们账号密码的关键
  sub_400C9A(my_zhanghao);
  sub_400CDD(my_zhanghao);
  sub_4008F7(my_zhanghao);
  sub_400977(my_zhanghao, my_mima);
  sub_400876(my_zhanghao, my_mima);
  return 0LL;
}
```

那我们先看下sub_400C9A()这个函数是来干嘛的 分析可知关键在于sub_400C41()这个函数

```
__int64 __fastcall sub_400C9A(__int64 a1)
{
  int my_zhanghao; // [rsp+1Ch] [rbp-4h]

  for ( my_zhanghao = 0; my_zhanghao <= 49 && *(my_zhanghao + a1); ++my_zhanghao )
    ;    #这里for循环后来看了师傅的wp才知道是用来限定输入的账号如果字符数大于50字节 就按50字节进行计算     作为参数传入s
  return sub_400C41(my_zhanghao);
}
```

那我们来分析吧

```
__int64 __fastcall sub_400C41(int i)
{
  __int64 result; // rax

  if ( 4 * (i >> 2) != i || 4 * (i >> 4) == i >> 2 || (result = (i >> 3), !result) || i >> 4 )    #这个条件看
  {
    puts("invalid username or password");
    exit(0);
  }
  return result;
}
```

```
for i in range(1,20):
        if 4 * (i >> 2)== i and 4 * (i >> 4) != i >> 2:
                if i << 4:
                        if i << 3 != 0:
                                print i
#4,8,12
#这里我解出来三个值  师傅wp是没有4的  还好这个不影响结果  这个等补坑吧
#20181026来补坑         4是不对的
#i必须是4的整数倍，但必须大于4，不能是16的整数倍,i必须小于16
#所以只有8,12符合
#具体可以看这个帖子：https://bbs.pediy.com/thread-247423-1.htm#1566256
```

下面sub_400CDD()这个函数用来计算正确的账号 也就是正确的用户名

```
signed __int64 __fastcall sub_400CDD(unsigned int *my_zhanghao)
{
  signed __int64 result; // rax
  __int64 v2; // [rsp+10h] [rbp-20h]
  __int64 v3; // [rsp+18h] [rbp-18h]
  __int64 v4; // [rsp+20h] [rbp-10h]

  v4 = *my_zhanghao;
  v3 = my_zhanghao[1];
  v2 = my_zhanghao[2];
  if ( v4 - v3 + v2 != 1550207830
    || v3 + 3 * (v2 + v4) != 12465522610LL
    || (result = 3651346623716053780LL, v2 * v3 != 3651346623716053780LL) )
  {
    puts("invalid username or password");
    exit(0);
  }
  return result;
}
```

这里求解用z3再好不过啦　求出十进制

```
from z3 import *
v4 = Int('v4')
v3 = Int('v3')
v2 = Int('v2')
s = Solver()
s.add(v4 - v3 + v2 == 1550207830)
s.add(v3 + 3 * (v2 + v4) == 12465522610)
s.add(v2 * v3 == 3651346623716053780)
if s.check() != sat:
 print 'unsat'
else:
 m = s.model()
 print m
#s2=[1868915551,1953724780,1635017059]
```

转成十六进制

```
s2=[1868915551,1953724780,1635017059]
for i in s2:
 print hex(i)
#s3 = [0x61746163,0x7473796c,0x6f65635f]
```

因为是小端序 手动调整下顺序 最后得到账号**catalyst_ceo**

```
import binascii
from libnum import n2s,s2n
print n2s(0x63617461) + n2s(0x6c797374) + n2s(0x5f63656f)
#catalyst_ceo
```

这时你就会惊奇的发现前面那个求账号长度的sub_400C9A()这个函数根本用不上嘛。。

对下面的sub_4008F7()顿时也没啥兴趣了 这个是用来限定字符范围 对输入的用户名进行遍历 每个字符十六进制范围是

(0x60-0x7A) 并且有一个字符必须是"_"

```
__int64 __fastcall sub_4008F7(__int64 a1)
{
  __int64 result; // rax
  int i; // [rsp+1Ch] [rbp-4h]

  for ( i = 0; ; ++i )
  {
    result = *(i + a1);
    if ( !result )
      break;
    if ( (*(i + a1) <= 0x60 || *(i + a1) > 'z') && *(i + a1) != '_' )
    {
      puts("invalid username or password");
      exit(0);
    }
  }
  return result;
}
```

直接看sub_400977()吧

```
__int64 __fastcall sub_400977(_DWORD *my_zhanghao, _DWORD *my_mima)
{
  int v2; // ebx
  int v3; // ebx
  int v4; // ebx
  int v5; // ebx
  int v6; // ebx
  int v7; // ebx
  int v8; // ebx
  int v9; // ebx
  int v10; // ebx
  int v11; // ebx
  unsigned int v12; // ebx
  __int64 result; // rax
  int i; // [rsp+2Ch] [rbp-14h]

  for ( i = 0; *(my_mima + i); ++i )    #这个for循环用来限定密码字符范围
  {
    if ( (*(my_mima + i) <= 96 || *(my_mima + i) > 122)
      && (*(my_mima + i) <= 64 || *(my_mima + i) > 90)
      && (*(my_mima + i) <= 47 || *(my_mima + i) > 57) )
    {
      puts("invalid username or password");
      exit(0);
    }
  }
  srand(my_zhanghao[1] + *my_zhanghao + my_zhanghao[2]);    #srand()用来设置rand()产生随机数时的随机数种子
  v2 = *my_mima;
  if ( v2 - rand() != 1441465642 )
  {
    puts("invalid username or password");
```

```
    exit(0);
  }
  v3 = my_mima[1];
  if ( v3 - rand() != 251096121 )
  {
    puts("invalid username or password");
    exit(0);
  }
  v4 = my_mima[2];
  if ( v4 - rand() != -870437532 )
  {
    puts("invalid username or password");
    exit(0);
  }
  v5 = my_mima[3];
  if ( v5 - rand() != -944322827 )
  {
    puts("invalid username or password");
    exit(0);
  }
  v6 = my_mima[4];
  if ( v6 - rand() != 647240698 )
  {
    puts("invalid username or password");
    exit(0);
  }
  v7 = my_mima[5];
  if ( v7 - rand() != 638382323 )
  {
    puts("invalid username or password");
    exit(0);
  }
  v8 = my_mima[6];
  if ( v8 - rand() != 282381039 )
  {
    puts("invalid username or password");
    exit(0);
  }
  v9 = my_mima[7];
  if ( v9 - rand() != -966334428 )
  {
    puts("invalid username or password");
    exit(0);
  }
  v10 = my_mima[8];
  if ( v10 - rand() != -58112612 )
  {
    puts("invalid username or password");
    exit(0);
  }
  v11 = my_mima[9];
  v12 = v11 - rand();
  result = v12;
  if ( v12 != 605226810 )
  {
    puts("invalid username or password");
    exit(0);
  }
  return result;
}
```

和前面说的一样

srand()用来设置rand()产生随机数时的随机数种子，参数seed必须是整数，通常可以用time(0)的返回值作为seed

但是如果每次seed都设置相同的值，rand()产生的随机数值每次也都一样

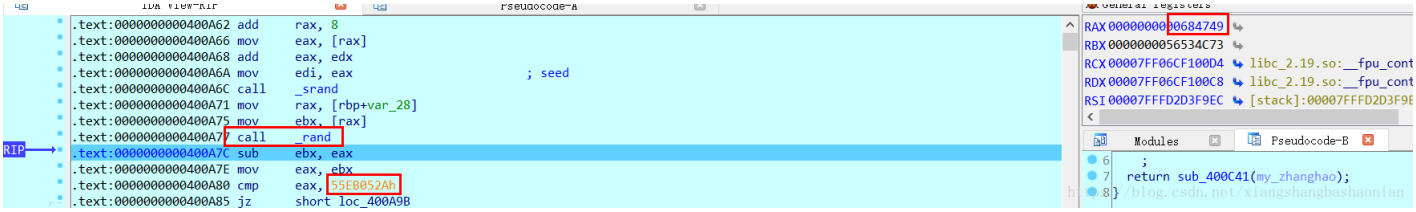
那么我们就可以直接在每个if语句那里下上断点 动态调试即可获取rand()产生的值啦

```
28  v2 = *my_mima;
29  if ( v2 - rand() != 1441465642 )
30  {
31    puts("invalid username or password");
32    exit(0);
33  }
34  v3 = my_mima[1];
35  if ( v3 - rand() != 251096121 )
36  {
37    puts("invalid username or password");
38    exit(0);
39  }
40  v4 = my_mima[2];
41  if ( v4 - rand() != -870437532 )
42  {
43    puts("invalid username or password");
44    exit(0);
```


F8步过call _rand 此时eax = 0x684749

也就是rand() = 0x684749

v2 = 0x00684749 + 0x55EB052A



接着得到v3 v4.....

```
v2 = 00684749 + 55EB052A
v3 = 673CE537 + 0EF76C39
v4 = 7B4505E7 + CC1E2D64
v5 = 70A0B262 + C7B6C6F5
v6 = 33D5253C + 26941BFA
v7 = 515A7675 + 260CF0F3
v8 = 596D7D5D + 10D4CAEF
v9 = 7CD29049 + C666E824
v10 = 59E72DB6 + FC89459C
v11 = 4654600D + 2413073A
```

python转成字符串就是密码

要注意进的高位1直接舍弃 还有注意小端序

```
import binascii
from libnum import n2s,s2n
v2 = 0x00684749 + 0x55EB052A
v3 = 0x673CE537 + 0x0EF76C39
v4 = 0x7B4505E7 + 0xCC1E2D64
v5 = 0x70A0B262 + 0xC7B6C6F5
v6 = 0x33D5253C + 0x26941BFA
v7 = 0x515A7675 + 0x260CF0F3
v8 = 0x596D7D5D + 0x10D4CAEF
v9 = 0x7CD29049 + 0xC666E824
v10 = 0x59E72DB6 + 0xFC89459C
v11 = 0x4654600D + 0x2413073A
mima = [v2,v3,v4,v5,v6,v7,v8,v9,v10,v11]
for i in mima:
 print hex(i)
'''0x56534c73
0x76345170
0x14763334b
0x138577957
0x5a694136
0x77676768
0x6a42484c
0x14339786d
0x156707352
0x6a676747'''
print n2s(0x734c5356)+n2s(0x70513476)+n2s(0x4b336347)+n2s(0x57795738)+n2s(0x3641695a)+n2s(0x68676777)+n2s(0
#sLSVpQ4vK3cGWyW86AiZhggwLHBjmx9CRspVGggj
```

最后那个函数改天再研究 直接输入正确的账号密码 即可得到flag

ALEXCTF{1_t41d_y0u_y0u_ar3__gr34t__reverser__s33}

```
Welcome to Catalyst systems
Loading..
Username: catalyst_ceo
Password: sLSVpQ4vK3cGWyW86AiZhggwLHBjmx9CRspVGggj
Logging in.
your flag is: ALEXCTF{1_t41d_y0u_y0u_ar3__gr34t__reverser__s33}
Looking for GNU DWARF file at "/lib/x86_64-linux-gnu/libc-2.19.so"... fou
```

参考链接：https://blog.csdn.net/weixin_43090100/article/details/82180752