IDF实验室-简单的ELF逆向 writeup

转载

weixin_30339969 于 2016-12-02 19:35:00 发布 141 收藏
文章标签： python php
原文链接：http://www.cnblogs.com/zhengjim/p/6127104.html

题目：http://ctf.idf.cn/index.php?g=game&m=article&a=index&id=39

下载得到ElfCrackMe1文件，直接用IDA打开。

最早想到的是 function一路F5下去。可以看到关键的main函数

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int v3; // ebx@6
  const char **v4; // rdx@22
  __int64 v6; // [sp+0h] [bp-C0h]@1
  __int64 v7; // [sp+8h] [bp-B8h]@1
  __int64 v8; // [sp+10h] [bp-B0h]@1
  __int64 v9; // [sp+18h] [bp-A8h]@1
  __int64 v10; // [sp+20h] [bp-A0h]@1
  __int64 v11; // [sp+28h] [bp-98h]@1
  __int64 v12; // [sp+30h] [bp-90h]@1
  __int64 v13; // [sp+38h] [bp-88h]@1
  int v14; // [sp+40h] [bp-80h]@1
  char v15[17]; // [sp+80h] [bp-40h]@2
  char v16; // [sp+91h] [bp-2Fh]@14
  char v17; // [sp+92h] [bp-2Eh]@15
  char v18; // [sp+93h] [bp-2Dh]@16
  char v19; // [sp+94h] [bp-2Ch]@17
  char v20; // [sp+95h] [bp-2Bh]@18
  int v21; // [sp+A4h] [bp-1Ch]@1
  int v22; // [sp+A8h] [bp-18h]@7
  int i; // [sp+ACh] [bp-14h]@9

  v21 = 0;
  memset(&v6, 0, 0x58uLL);
  v6 = 854698492143LL;
  v7 = 880468295913LL;
  v8 = 597000454391LL;
  v9 = 605590388953LL;
  v10 = 932007903423LL;
  v11 = 760209211613LL;
  v12 = 579820585151LL;
  v13 = 940597838039LL;
  v14 = 191;
  printf(
    "plz enter the flag:",
    argv,
    11LL,
    854698492143LL,
    880468295913LL,
    597000454391LL,
    605590388953LL,
    932007903423LL,
```

```
    760209211613LL,
    579820585151LL,
    940597838039LL,
    *(_QWORD *)&v14);
  while ( 1 )
  {
    v3 = v21;
    v15[v3] = getch();
    if ( !v15[v3] || v15[v21] == 10 )
      break;
    if ( v15[v21] == 8 )
    {
      printf("\b\b", v6, v7, v8, v9, v10, v11, v12, v13, *(_QWORD *)&v14);
      --v21;
    }
    else
    {
      putchar(v15[v21++]);
    }
  }
  v22 = 0;
  if ( v21 != 22 )
    v22 = 1;
  for ( i = 0; i <= 16; ++i )
  {
    if ( v15[i] != (*((_DWORD *)&v6 + i) - 1) / 2 )
    {
      v22 = 1;
      argv = (const char **)((*((_DWORD *)&v6 + i) - 1) / 2);
      printf("%d", argv, v6, v7, v8, v9, v10, v11, v12, v13, *(_QWORD *)&v14);
    }
  }
  if ( v16 != 48 || v17 != 56 || v18 != 50 || v19 != 51 || v20 != 125 )
    v22 = 1;
  v15[v21] = 0;
  puts("\r");
  if ( v22 )
  {
    puts("u r wrong\r\n\r");
    main((unsigned __int64)"u r wrong\r\n\r", argv, v4);
  }
  else
  {
    puts("u r right!\r");
  }
  return 0;
}
```

看到最下面代码

```
if ( v22 )
{
  puts("u r wrong\r\n\r");
  main((unsigned __int64)"u r wrong\r\n\r", argv, v4);
}
else
{
  puts("u r right!\r");
}
```

当v22=1时 输出 u r wrong

所以就正确就得使v22为0

我们往上看

```
if ( v16 != 48 || v17 != 56 || v18 != 50 || v19 != 51 || v20 != 125 )
  v22 = 1;
v15[v21] = 0;
puts("\r");
```

当v16 != 48 || v17 != 56 || v18 != 50 || v19 != 51 || v20 != 125时

V22=1

所以就是v16=48 v17=56 v18 = 50 v19=51 v20= 125

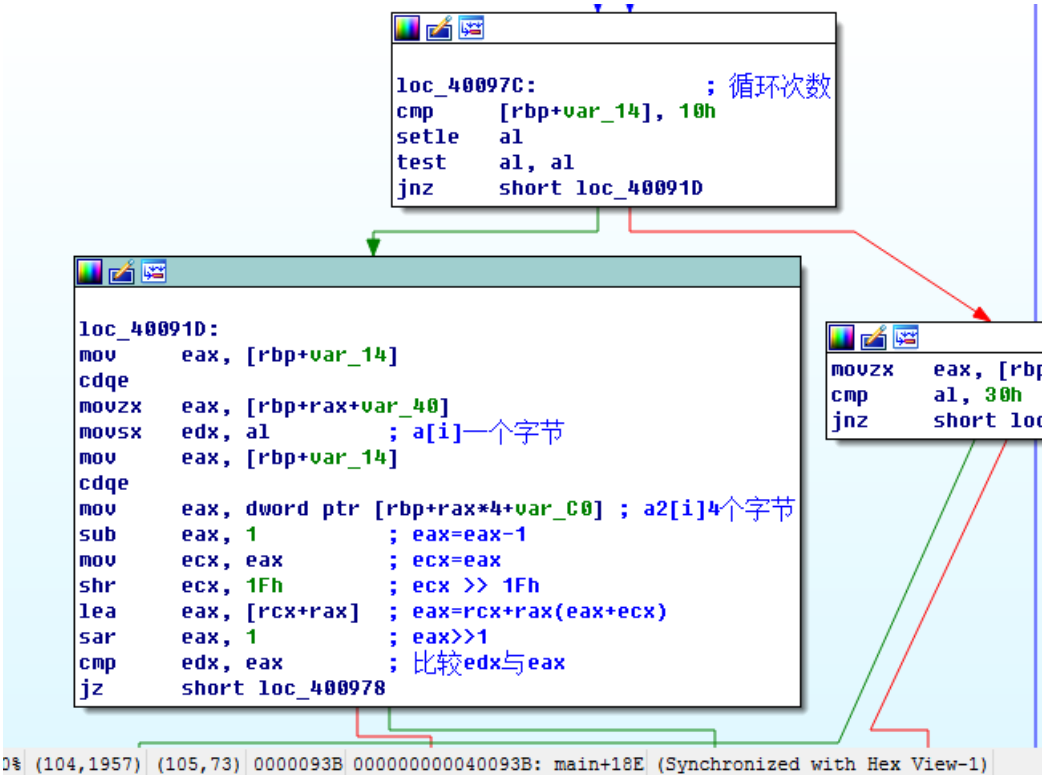
V22=0

用python的chr()转化下得到 0 8 2 3 }

可以猜测这个是flag的后半部分

继续往上看发现一串while代码，读起来很吃力。。。所以果断选择放弃，直接去看看汇编代码

观察视图 找到while循环的那一部分



shr指令与sar指令，查了下是逻辑右移和算术右移，在编程里都是<<

但选择用什么是编译器决定的……
所以编程时，最好只对unsigned类型做这种操作……

起初lea=eax,[rcx+rax]这指令没看懂，只知道eax=rcx+rax 。就不知道什么意思，

后来查阅质料才知道因为是64位，所以寄存器变化了

64位寄存器的变化
eax,ebx,ecx,edx,esi,edi,ebp,esp等变为
rax,rbx,rcx,rdx,rsi,rdi,rbp,rsp

所以得出 a[i]=((a2[i]-1)+((a2[i])-1)>>1Fh))>>1

要注意运算优先级，直接要括号就行了。

所以我们要找a2[i]的值，

跳转到a2[i]的定义，看到：

```
-00000000000000C0 ; D/A/*
-00000000000000C0 ; N
-00000000000000C0 ; U
-00000000000000C0 ; Use data
-00000000000000C0 ; Two spec
-00000000000000C0 ; Frame si
-00000000000000C0 ;
-00000000000000C0
-00000000000000C0 var_C0
-00000000000000B8 var_B8
-00000000000000B0 var_B0
-00000000000000A8 var_A8
-00000000000000A0 var_A0
-0000000000000098 var_98
-0000000000000090 var_90
-0000000000000088 var_88
-0000000000000080 var_80
```

在回代码，往上看到：

```
mov     dword ptr [rbp+var_C0], 0EFh
mov     dword ptr [rbp+var_C0+4], 0C7h
mov     dword ptr [rbp+var_B8], 0E9h
mov     dword ptr [rbp+var_B8+4], 0CDh
mov     dword ptr [rbp+var_B0], 0F7h
mov     dword ptr [rbp+var_B0+4], 8Bh
mov     dword ptr [rbp+var_A8], 0D9h
mov     dword ptr [rbp+var_A8+4], 8Dh
mov     dword ptr [rbp+var_A0], 0BFh
mov     dword ptr [rbp+var_A0+4], 0D9h
mov     dword ptr [rbp+var_98], 0DDh
mov     dword ptr [rbp+var_98+4], 0B1h
mov     dword ptr [rbp+var_90], 0BFh
mov     dword ptr [rbp+var_90+4], 87h
mov     dword ptr [rbp+var_88], 0D7h
mov     dword ptr [rbp+var_88+4], 0DBh
mov     [rbp+var_80], 0BFh
```

所以知道，a2[i]的值是[0x0EF, 0x0C7, 0x0E9, 0x0CD, 0x0F7, 0x8B, 0x0D9,0x8D, 0x0BF, 0x0D9, 0x0DD, 0x0B1, 0x0BF, 0x87,0x0D7, 0x0DB, 0x0BF]

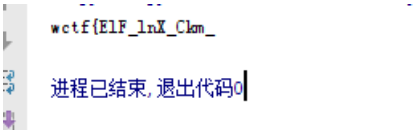
这样就可以直接写python脚本

```
#!usr/bin/env python
#!coding=utf-8

__author__ = 'zhengjim'


a2=[0x0EF, 0x0C7, 0x0E9, 0x0CD, 0x0F7, 0x8B, 0x0D9,
 0x8D, 0x0BF, 0x0D9, 0x0DD, 0x0B1, 0x0BF, 0x87,
 0x0D7, 0x0DB, 0x0BF]
L=[]
for i in range(17):
    flag =((a2[i] - 1) + ((a2[i] - 1) >> 0x1f))>>1
    aa = chr(flag)
    L.append(aa)
flag1 = ''.join(L)
print flag1
```

得到



在与前面的0 8 2 3 }连接 得到：

wctf{ElF_lnX_Ckm_0823}

转载于:https://www.cnblogs.com/zhengjim/p/6127104.html