jarvis oj DebugMe writeup

原创

charlie_heng 于 2017-11-27 14:04:11 发布 443 收藏

分类专栏： 二进制-逆向工程

本文链接：https://blog.csdn.net/charlie_heng/article/details/78644424

话说这题真的是玄学做出来的。。。

首先看一波main函数

```
if ( sub_A30(2u, (int)argv) )
{
  for ( i = 0; i < j_strlen(v12); ++i )
    byte_4008[i] = v12[i] ^ i;
}
```

sub_A30这里是fork了一个子进制，然后调用ptrace post了一些东西到子进程那里

然后这里就是异或了一波输入的东西

接着在下面看到有这个东西，也是输入异或了一波

```
if ( !j_strstr(v20, &v15) )
{
  for ( j = 0; j < j_strlen(v12); ++j )
    byte_4008[j] = v12[j] ^ 1;
}
```

这里的判断条件有点看得不是很懂，这里也不多说。。。免得误导

```
if ( !sub_B1C() )
  __breakpoint(0);
if ( sub_C14(byte_4008, 7) )
  v10 = "you win!\nFlag is your password!";
else
  v10 = "The password you input is wrong!";
```

这里插了一个breakpoint，目测就是跟上面子进程那里有关

在sub_c14里面就是主要的判断逻辑的地方

```
switch ( v3 )
{
  case 0:
    if ( *v2 == 105 )
      goto LABEL_26;
    return 0;
  case 1:
    if ( *v2 != 101 )
      return 0;
    goto LABEL_26;
  case 3:
    if ( *v2 != 110 )
      return 0;
    goto LABEL_26;
  case 4:
    if ( *v2 != 100 )
      return 0;
    goto LABEL_26;
  case 5:
    if ( *v2 != 97 )
      return 0;
    goto LABEL_26;
  case 6:
    if ( *v2 != 103 )
      return 0;
    goto LABEL_26;
  case 7:
    if ( *v2 != 115 )
      return 0;
    goto LABEL_26;
  case 9:
    if ( *v2 == 114 )
    {
EL_26:
      sub_EAC(7 * (v3 + 1), 11);
      ++v2;
      v3 = v10;
      continue;
    }
```


v3就是传进来的第二个参数，就是7

然后根据v3的顺序来判断输入的字符串是否正确，根据题目所说，有8个字符，这里应该所有判断条件都要用上

在case9那个label 那里就是更新v3的地方，这里的反编译成c语言的结果有点问题

看了下arm的汇编，准确的是

v3=7 * (v3 + 1) - 11*sub_E14(7 * (v3 + 1), 11)

然后编个程序，看下验证的顺序，发现是 7 1 3 6 5 9 4 2 只验证了7个字母，这里感觉就有点奇怪了

于是想动态调试一波

把所有可疑的反调试的全部nop掉

然后开了安卓虚拟机，用gdbserver 调了一下，发现的确是这个顺序。。。感觉就更奇怪了。。。

于是想了想，是不是输入的第二个参数不是7，而是其他呢？

又跑了下，发现输入为0的时候，完整的验证了8个字母 ,然后这个时候就能猜测出上面那个可疑的breakpoint和fork的程序到底干了啥，目测就是停在这个breakpoint，然后父进程把输入的参数改成0，再继续运行验证

然后关键上面有两个异或的操作，到底真正执行了哪一个呢？

这个时候爆破一波就可以了，总共的可能性也不多

代码如下，s4就是flag

```python
a=[105,115,101,110,103,97,114,100]
b=[i^1 for i in a]
c=[b[i]^i for i in range(len(b))]
d=[a[i]^i for i in range(len(a))]
s1=''
s2=''
s3=''
s4=''
for i in range(len(a)):
    s1+=chr(a[i])
    s2+=chr(b[i])
    s3+=chr(c[i])
    s4+=chr(d[i])
print(s1)
print(s2)
print(s3)
print(s4)
```