whaleCTF RE方向部分writeup

原创

43v3rY0unG 于 2020-01-02 21:44:21 发布 260 收藏
分类专栏： # RE 文章标签： CTF RE

本文链接：https://blog.csdn.net/weixin_43876357/article/details/103806304

0x00 PE格式

用PEtools打开，通过查找

块数目 0x0004
时间戳 0x591D5CCC
入口地址 0x0005B789
信息标识 0x0103

得出flag：BJWXB_CTF{0004h-591D5CCCh-0005B789h-0103h}

0x01 WarmUp

经过一系列操作，进入compare函数，然后进入分支语句。

```
.data:004140B0 aPause           db 'pause',0            ; DATA XREF: _main:loc_4011F3↑o
.data:004140B6                  align 4
.data:004140B8 aSorryTryAgain   db 'Sorry! Try again.',0
.data:004140B8                                          ; DATA XREF: _main:loc_40118F↑o
.data:004140CA                  align 4
.data:004140CC aContratulation  db 'Contratulations!',0 ; DATA XREF: _main+132↑o
.data:004140DD                  align 10h
.data:004140E0 aLdyvlqmzhuyCqQ  db 'LDYVLQMZHuY:|cQ[^Qyo|cQ{~QYO\CQ[^/s',0
.data:004140E0                                          ; DATA XREF: _main:loc_4010FC↑o
.data:00414104 unk_414104       db  50h ; P             ; DATA XREF: _main+1E↑o
.data:00414105                  db  6Ch ; l
```

回头看，若v17和上图中高亮部分相等，则进入congratulations语句，否则try again

```
v6 = 0;
v7 = strlen(&v17) + 1;
if ( (v7 - 1) > 0 )
{
  do
  {
    *(&v17 + v6) ^= 0xEu;
    ++v6;
  }
  while ( v6 < (v7 - 1) );
}
if ( !strcmp(&v17, aLdyvlqmzhuyCqQ) )
{
  v8 = sub_401700(&unk_417CF8, aContratulation);
  sub_401280(10);
  v9 = 0;
  v10 = *(*v8 + 4) + v8;
```


可见是一个简单的异或操作，写一个脚本（devc++）

```
string v17 = "LDYVLQMZHuY:|cQ[^Qyo|cQ{~QYO\CQ[^/s";
int v7 = v17.length() +1;
int v6 = 0;
if ( (v7 - 1) > 0 )
  {
    do
    {
      v17[v6] = (char)((int)v17[v6] ^ 0xEu);
      //cout<<*(&v17 + v6);
      ++v6;
    }
    while ( v6 < (v7 - 1) );
  }
cout<<v17<<endl;
}
```

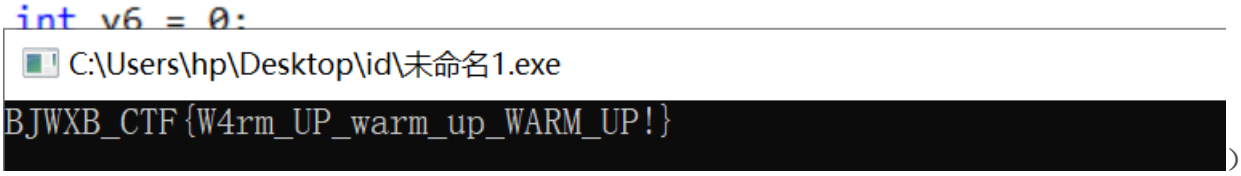

得到flag：BJWXB_CTF{W4rm_UP_warm_up_WARM_UP!}

（不过这中间出了一个小插曲，最后的WARM_UP中的R没有显示出来，猜测应该是被转义了，查了一下资料：

因为C/C++/C#都有转义符这个东西，所以字符串中的反斜杠“\”会被识别成别的东西，不能完整的打印。　比如：string id = USB\VID_";。使用cout << id << endl打印可能就是“USBVID_”。　为了完整打印，可以利用无视转义符操作。　C++：string id = R"(USB\VID_)";。添加R。　C#：string id = @"USB\VID_";。添加@。

但是在devc++上这样写会报错，没什么用，所以莫不如直接在反斜杠前再加一个反斜杠就好了.....

```
string v17 = "LDYVLQMZHuY:|cQ[^Qyo|cQ{~QYO\\CQ[^/s";
int v7 = v17.length() +1;
int v6 = 0;
```



)

0x02 app-release

拿到一个apk文件，首先就对apk文件进行反编译，这里的一系列操作就不说了，可以上网查看，然后丢到jd-gui里反编成Java代码，找到main主函数，发现也是异或，和上一题差不多

```
public byte[] a(byte[] paramArrayOfByte) {
  if (paramArrayOfByte == null)
    return null;
  int i = paramArrayOfByte.length;
  byte b = 0;
  while (true) {
    byte[] arrayOfByte = paramArrayOfByte;
    if (b < i) {
      paramArrayOfByte[b] = (byte)(paramArrayOfByte[b] ^ 0x12);
      b++;
      continue;
    }
    return arrayOfByte;
  }
}
```


找到a.class文件，看完就大致明白了，和0x01题一个思路

```
public void onClick(View paramView) {
  MainActivity.a(this.a, "PXEJPMQFTiS|v`\"#vMDw`KMA3_b~w3o");
  MainActivity.a(this.a, (EditText)this.a.findViewById(2131427412));
  MainActivity.b(this.a, MainActivity.a(this.a).getText().toString()
  String str = new String(this.a.a(MainActivity.b(this.a).getBytes()
  Log.i(str, MainActivity.b(this.a));
  t t = new t(this.a);
  if (str.equals(MainActivity.c(this.a)) == true) {
    t.a("提示");
    t.b("Contratulations! You Got Correct Flag.");
    t.c();
    return;
  }
  t.a("提示");
  t.b("Sorry! You Got Error Flag.");
  t.c();
}
```

然后就改一下上一题的脚本就好了，方便的很，运行出结果，得到flag：BJWXB_CTF{Andr01d_VerY_S!Mple!}

此题难点就在于apk的反编译，多做几个这种题熟悉熟悉，apk是一个安卓上运行的文件，你可以发到自己手机上验证一下，本人闲的无聊，在电脑上下载了一个手机模拟器，蛮好玩的，直接在电脑上就可以跑了。

0x03 r100(defcamp)

找到核心位置代码：

```
signed int i; // [rsp+14h] [rbp-24h]
const char *v3; // [rsp+18h] [rbp-20h]
const char *v4; // [rsp+20h] [rbp-18h]
const char *v5; // [rsp+28h] [rbp-10h]

v3 = "Dufhbmf";
v4 = "pG`imos";
v5 = "ewUglpt";
for ( i = 0; i <= 11; ++i )
{
  if ( (&v3)[i % 3][2 * (i / 3)] - *(i + a1) != 1 )
    return 1LL;
}
return 0LL;
```


蛮有意思的，二维数组这里有点没太看懂，&v3就是v3的长度7，然后最后的脚本是这样写的，可以参考一下：

```
string v3 = "DufhbmfpG`imosewUglpt";
 for ( int i = 0; i <= 11; ++i )
 {

   cout<< (char)(v3[7 * (i % 3) + 2 * (i / 3) ] -1 );
 }
}
```


因为要求的是s，也就是a1，所以直接输出就可以了，得到flag：Code_Talkers

搜了一下其他大佬的博客，这个说的很好，转载自原文链接：https://blog.csdn.net/xiangshangbashaonian/article/details/82860388

一个for循环 对三个串(以二维数组的形式)进行操作 就可以得到密码

在解之前 先把三个串转成ascii

```
b = [[68, 117, 102, 104, 98, 109, 102],
     [112, 71, 96, 105, 109, 111, 115],
     [101, 119, 85, 103, 108, 112, 116]]
flag = ''
for i in range(12):
    flag += chr(b[i % 3][2* (i / 3)] -1)
    print(flag)
```

运行就可以得到flag

0x04 逆向练习

进入主函数：

```
  v6 = getch();
  v32[v31] = v6;
  if ( !v6 || v32[v31] == 13 )
    break;
  if ( v32[v31] == 8 )
  {
    printf("\b\b");
    --v31;
  }
  else
  {
    printf("%c", v32[v31++]);
  }
}
v8 = 0;
for ( i = 0; i < 17; ++i )
{
  if ( v32[i] != byte_415768[*(&v9 + i)] )
    v8 = 1;
}
if ( v33 != '1' || v34 != '0' || v35 != '2' || v36 != '4' || v37 != '}' )
  v8 = 1;
v32[v31] = 0;
printf("\r\n");
if ( v8 )
{
  printf("u r wrong\r\n\r\n");
  main(v3, v4, v5);
}
```

高亮这里还挺有趣的，v9的内存地址和v10,v11....连在一起，所以就是这个字符串的第v9/v10/v11...依此类推个数了，v33-v37在v32后面，所以可以认为他们是一起的，然后在加上1024}，下图写个简单的脚本（c++）：

```
-00000028 var_28          db 17 dup(?)
-00000017 var_17          db ?
-00000016 var_16          db ?
-00000015 var_15          db ?
-00000014 var_14          db ?
-00000013 var_13          db ?
-00000012                 db ? ; undefined
```

```
int s[17] = {1,4,14,10,5,36,23,42,13,19,28,13,27,39,48,41,42};
string a = "KfxEeft}f{gyrYgthtyhifsjei53UUrrr_t2cdsef66246087138\0087138";
for(int i=0;i<17;i++)
{
    cout<<a[s[i]-1];
}
cout<<"1024}";
}
```


得到flag：KEY{e2s6ry3r5s8f61024}