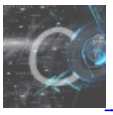
i春秋-网络与信息安全专项赛部分题目write up

原创

Whale_XM 于 2019-08-16 10:19:07 发布 771 收藏 2

分类专栏： 网安 文章标签： write up



逆向

flat

进入IDA分析，找到主函数，F5查看伪代码

```
{
  unsigned int v3; // eax
  char v4; // al
  signed int v5; // ecx
  char v6; // al
  signed int v7; // ecx
  char v8; // al
  unsigned int v9; // ecx
  char v10; // al
  unsigned int v11; // ecx
  char v12; // al
  unsigned int v13; // ecx
  __int64 v15; // [rsp+0h] [rbp-1A0h]
  signed int v16; // [rsp+4Ch] [rbp-154h]
  char dest; // [rsp+50h] [rbp-150h]
  int len; // [rsp+E8h] [rbp-B8h]
  int v19; // [rsp+ECh] [rbp-B4h]
  char v20[64]; // [rsp+F0h] [rbp-B0h]
  char s[108]; // [rsp+130h] [rbp-70h]
  int v22; // [rsp+19Ch] [rbp-4h]

  v22 = 0;
  printf("please input string:\n", argv, envp);
  gets(s);
  len = strlen(s);
  v19 = 5;
  v16 = -1046111848;
  while ( 1 )
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( v16 == -2012804730 )
        {
```


里面存在好几个while循环和if判断，而且数值特别大。

再往下看，下面还会输出“what a shame!!!”,意思是好可惜啊，可以判断，如果输出这句话，就说明我们错了，所以，通过上面的条件将这句话跳过去。

```
{
  while ( v16 == -2012804730 )
  {
    memcpy(&dest, "J", 144uLL);            // dest
    v4 = fun_check1(s);                    // v4=1
    v5 = 0x916CBF8C;
    if ( v4 & 1 )                          // 结果不为0，循环
      v5 = 0x70B25450;
    v16 = v5;
  }
  if ( v16 != 0x916CBF8C )
    break;
  v16 = 0xBB37AF4F;
  LODWORD(v15) = printf("what a shame !!!\n", v15);
}
```


可以看出来，当if（v4&1）成立时，v5会重新赋值，v16也会变，不满足下面的判断，执行“break”，从而跳过这句话。所以说，v4=1。

```
    v16 = 0x3000D7A7;
    break;
  case (int)0xF465F43B:
    v16 = 0xBB37AF4F;                      // 4
    HIDWORD(v15) = printf("you got it !\n", v15);
    break;
  case 0x3000D7A7:
```

接下来会调到一个switch语句中去，而且他通过不断地判断来更改v16的值，从而进入不同的case中。
经过分析，他一改先进入第七个case，通过if判断，进入第六个case，再通过if判断，来到switch上方的if语句出，从而跳过if的break；接着过if判断，进入第二个case，经过if，进入第四个判断，然后输出“you get it !”,成功。
所以他们需要满足fun_check1到5返回值均为1。
主要函数有4个（1没太大作用），其中2，3，4比较简单，作用如下

```
    break;
  case 0x48DC1E73:
    v8 = fun_check3(s);                    // 判断flag长度42（a[0]-a[41])且最后一位（即a[41])为“}”
    v9 = 0x916CBF8C;                       // 6
    if ( v8 & 1 )
      v9 = 0x9FDEA8EC;
    v16 = v9;
    break;
  case 0x70B25450:
    v6 = fun_check2(s);                    // 7     输出1
    v7 = 0x916CBF8C;                       // 判断前五位“flag{”
    if ( v6 & 1 )
      v7 = 0x48DC1E73;
    v16 = v7;
    break;
}
```



```
    }
    if ( v16 != 0x9FDEA8EC )
      break;
    v10 = fun_check4(s);                   // 过    说明a[28],a[13],a[23],a[18]是连接线“-”
    v11 = 0x916CBF8C;
```

我们进入函数5，

```
s = a1;
  seven = a2;
  v12 = 0;
  v11 = -624695604;
  while ( 1 )
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( 1 )
        {
          while ( 1 )
          {
            while ( 1 )
            {
              while ( 1 )
              {
                while ( 1 )
                {
                  while ( 1 )
                  {
                    while ( 1 )
                    {
                      while ( 1 )
                      {
                        while ( 1 )
                        {
                          while ( 1 )
                          {
                            while ( 1 )
                            {
                              while ( 1 )
                              {
                                while ( 1 )
                                {
                                  while ( v11 == -1771681815 )
                                  {
                                    v6 = 1740029224;
                                    if ( s[v12] >= 'a' )
                                      v6 = -1207418117;
                                    v11 = v6;
                                  }
                                  if ( v11 != -1490231676 )
                                    break;
                                  ++v12;
                                  v11 = -768723158;
                                }
                                if ( v11 != -1407902233 )
                                  break;
                                v4 = -1188300396;
                                if ( s[v12] <= 57 )
                                  v4 = -478229440;
                                v11 = v4;
                              }
                              if ( v11 != -1207418117 )
                                break;
                              v7 = 1740029224;
                              if ( s[v12] <= 'z' )
                                v7 = 2096910144;
```

```
                v7 = 2090910144;
                v11 = v7;
              }
              if ( v11 != -1188300396 )
                break;
              v5 = -1771681815;   // 插个眼  在此判断字母与-
              if ( s[v12] == 45 )
                v5 = -1167333891;
              v11 = v5;
            }
            if ( v11 != -1167333891 )
              break;
            v13[v12] = s[v12];    // 如果s里是“-”就不做处理
            v11 = -118846692;
          }
          if ( v11 != -995934932 )
            break;
          v3 = -1188300396;
          if ( s[v12] >= '0' )    // 第一次 s[0]
            v3 = -1407902233;     // 第二次来  s【1】
          v11 = v3;               // 第三次来 s[2]
        }
        if ( v11 != -991718889 )
          break;
        v11 = -1490231676;
      }
      if ( v11 != -768723158 )
        break;
      v8 = 1681851953;
      if ( v12 < 36 )
        v8 = 434013166;
      v11 = v8;
    }
    if ( v11 != -624695604 )
      break;
    v2 = 659899916;
    if ( v12 < 36 )
      v2 = -995934932;
    v11 = v2;
  }
  if ( v11 != -478229440 )
    break;
  v13[v12] = s[v12] + 17;         // 如果s里面有数字，则加17变为大写字母
  v11 = 1926387427;
}
if ( v11 != -451717645 )
  break;
++v12;
v11 = -624695604;
}
if ( v11 != -118846692 )
  break;
v11 = 1926387427;
}
if ( v11 != 329160926 )
  break;
v16 = 0;
v11 = 1269730414;
}
if ( v11 != 434013166 )
```

```
          break;
        v9 = -991718889;
        if ( v13[v12] != seven[v12] )
          v9 = 329160926;
        v11 = v9;
      }
      if ( v11 != 659899916 )
        break;
      v12 = 0;
      v11 = -768723158;
    }
    if ( v11 == 1269730414 )
      break;
    switch ( v11 )
    {
      case 1681851953:
        v16 = 1;
        v11 = 1269730414;
        break;
      case 1740029224:
        v11 = -118846692;
        break;
      case 1926387427:
        v11 = -451717645;
        break;
      case 2096910144:
        v13[v12] = s[v12] - 1347911315 + 1347911267;// 小写字母-48 变成数字
        v11 = 1740029224;
        break;
    }
  }
  return v16 & 1;
}
```

经过多次循环，可以看出，该函数主要有三点功能，

1. 当输入为小写字母时，将其减48，变为数字
2. 当输入为数字是，加17，变为大写字母
3. 输入为‘-’时，不变
   变化后的字符串，与主函数里的dest字符串比较，可以直接写脚本了

```
s="J2261C63-3I2I-EGE4-IBCC-IE41A5I5F4HB"
flag=""
for k in s:
 i=ord(k)
 if i==45:
  flag=flag+chr(i)
 elif i>=48 and i<=57:
  flag=flag+chr(i+48)
 elif i>=65 and i<=90:
  flag=flag+chr(i-17)
print(flag)
```

解得flag：flag{9bbfa2fc-c8b8-464d-8122-84da0e8e5d71}

Misc

签到题

坑爹的签到题，以前没遇见过这种类型的，

```
invalid file (bad magic number): Exec format error
root@kali:~/22/_11.zip.extracted# dig TXT gamectf.com

; <<>> DiG 9.11.4-P2-3-Debian <<>> TXT gamectf.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28636
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;gamectf.com.                   IN      TXT

;; ANSWER SECTION:
gamectf.com.            5       IN      TXT     "flag{welcome_TXT}"

;; Query time: 118 msec
;; SERVER: 192.168.152.2#53(192.168.152.2)
;; WHEN: 四 8月 15 06:48:04 EDT 2019
;; MSG SIZE  rcvd: 59
```


得到flag

24word

下载压缩包，解压缩，有一张图片，上书二十四真言，是社会主义核心价值观，作为社会主义新青年，我们知道，这是社会主义核心价值观加密。

网上找在线解密，不知道为什么，全都输进去解不出来，只能几个几个的试，经过无数次尝试，解出密码，CodeValues。这不是flag，继续。
一张png图片，binwalk一下，里面有东西，分析出来，又是一个压缩包，还有密码。

找伪加密，不是。将上步解出的密码输入，对了。
又是一张图片，还是二十四字真言，

中间的东西看着就像二维码，扫一下，没反应，ps编辑，放大，在扫，flag出来了。