XCTF reverse-box

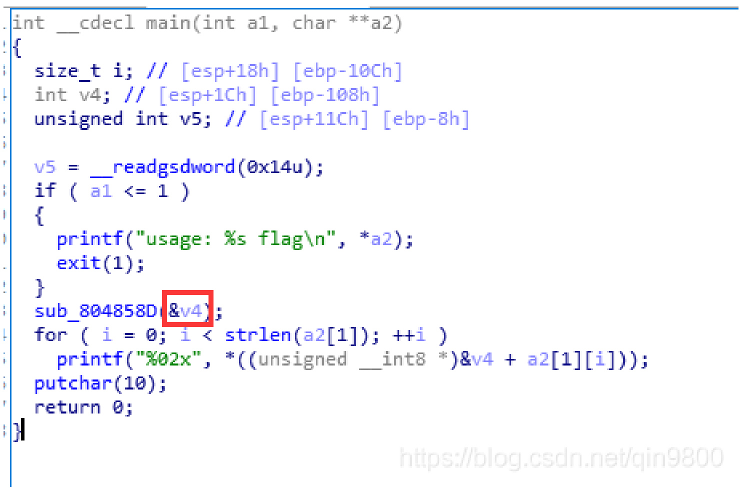
原创

夏了茶糜 于 2020-03-11 20:31:21 发布 185 收藏 1
分类专栏： CTF-REVERSE 文章标签： 安全

CTF-REVERSE 专栏收录该内容
18 篇文章 0 订阅
订阅专栏
题目要求：
输入flag让程序输出以下字符串，flag以TWCTF开头，xctf的题目提示被漏掉了

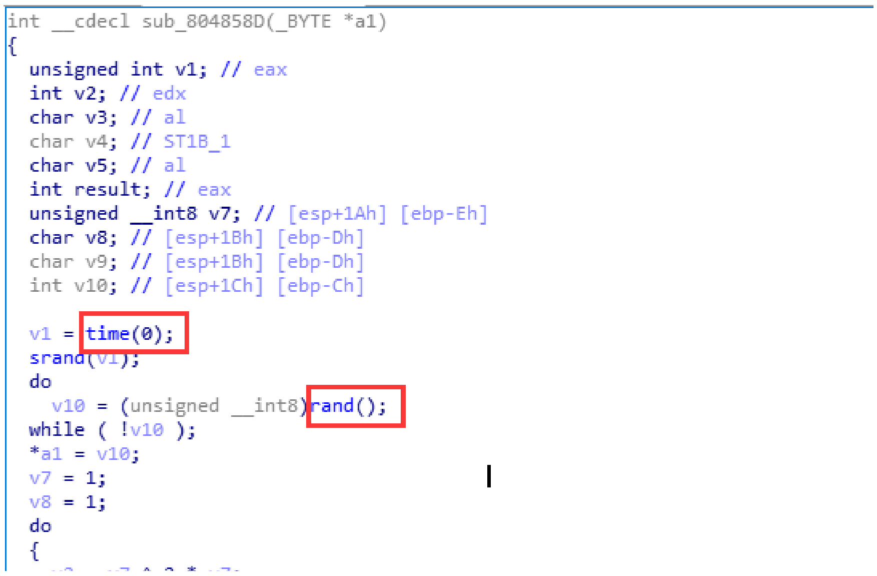
```
95eeaf95ef94234999582f722f492f72b19a7aaf72e6e776b57aee722fe77ab5ad9aaeb156729676ae7a236d99b1df4a
```

程序放入IDA中分析

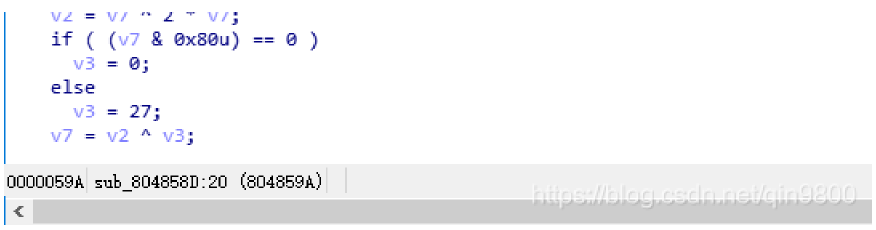


```
int __cdecl main(int a1, char **a2)
{
  size_t i; // [esp+18h] [ebp-10Ch]
  int v4; // [esp+1Ch] [ebp-108h]
  unsigned int v5; // [esp+11Ch] [ebp-8h]

  v5 = __readgsdword(0x14u);
  if ( a1 <= 1 )
  {
    printf("usage: %s flag\n", *a2);
    exit(1);
  }
  sub_804858D(&v4);
  for ( i = 0; i < strlen(a2[1]); ++i )
    printf("%02x", *((unsigned __int8 *)&v4 + a2[1][i]));
  putchar(10);
  return 0;
}
```

这题的关键点就是要获取到v4的值，v4是由sub_804858D生成的



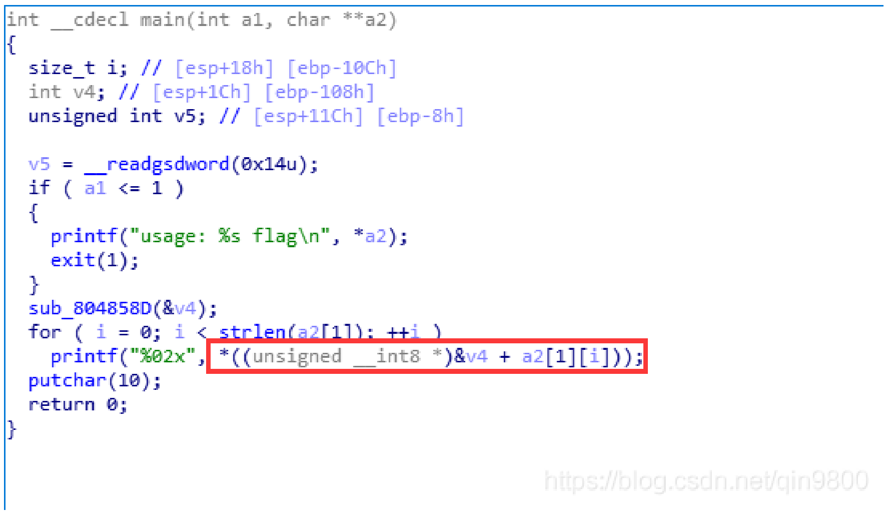
```
int __cdecl sub_804858D(_BYTE *a1)
{
  unsigned int v1; // eax
  int v2; // edx
  char v3; // al
  char v4; // ST1B_1
  char v5; // al
  int result; // eax
  unsigned __int8 v7; // [esp+1Ah] [ebp-Eh]
  char v8; // [esp+1Bh] [ebp-Dh]
  char v9; // [esp+1Bh] [ebp-Dh]
  int v10; // [esp+1Ch] [ebp-Ch]

  v1 = time(0);
  srand(v1);
  do
    v10 = (unsigned __int8)rand();
  while ( !v10 );
  *a1 = v10;
  v7 = 1;
  v8 = 1;
  do
  {
```



```
    v2 = v7 ^ 2 * v7;
    if ( (v7 & 0x80u) == 0 )
      v3 = 0;
    else
      v3 = 27;
    v7 = v2 ^ v3;
```

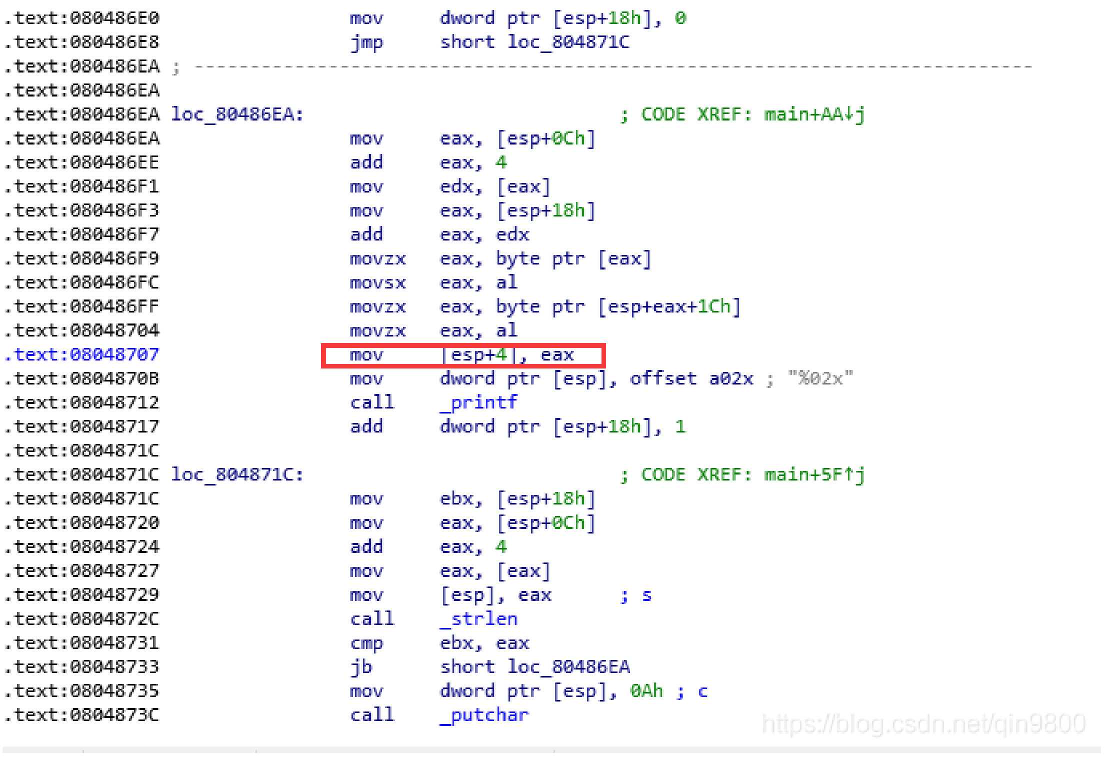
```
0000059A sub_804858D:20 (804859A)
```

但是它的值是通过时间产生的随机数，但是在赋值给v10的时候只保留了一个字节，所以只有0xff种可能



```
int __cdecl main(int a1, char **a2)
{
  size_t i; // [esp+18h] [ebp-10Ch]
  int v4; // [esp+1Ch] [ebp-108h]
  unsigned int v5; // [esp+11Ch] [ebp-8h]

  v5 = __readgsdword(0x14u);
  if ( a1 <= 1 )
  {
    printf("usage: %s flag\n", *a2);
    exit(1);
  }
  sub_804858D(&v4);
  for ( i = 0; i < strlen(a2[1]); ++i )
    printf("%02x", *((unsigned __int8 *)&v4 + a2[1][i]));
  putchar(10);
  return 0;
}
```

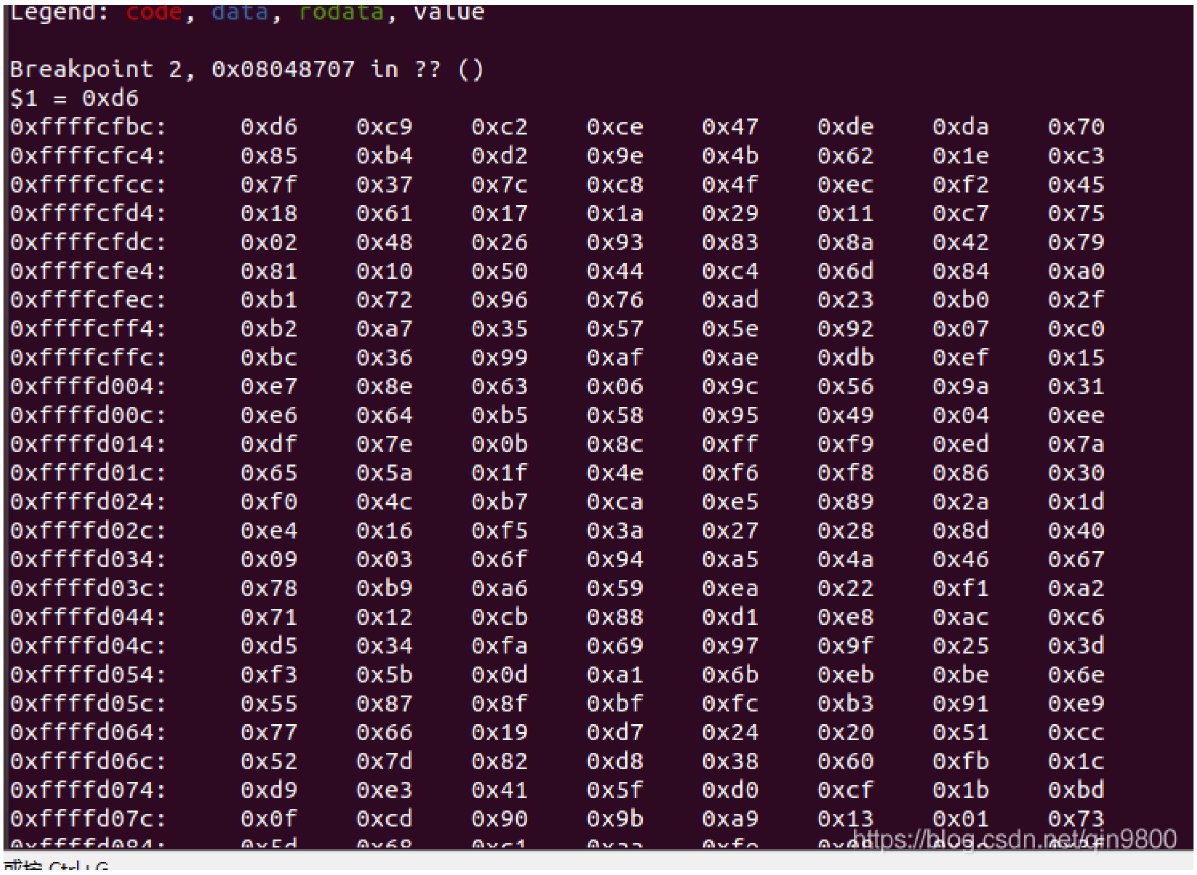
由于我们已经知道flag的前几个字符串，所以可以判断这里是最后输出的值，第一个值是否等于0x95



```
.text:080486E0                 mov     dword ptr [esp+18h], 0
.text:080486E8                 jmp     short loc_804871C
.text:080486EA ; ---------------------------------------------------------------------------
.text:080486EA
.text:080486EA loc_80486EA:                            ; CODE XREF: main+AA↓j
.text:080486EA                 mov     eax, [esp+0Ch]
.text:080486EE                 add     eax, 4
.text:080486F1                 mov     edx, [eax]
.text:080486F3                 mov     eax, [esp+18h]
.text:080486F7                 add     eax, edx
.text:080486F9                 movzx   eax, byte ptr [eax]
.text:080486FC                 movsx   eax, al
.text:080486FF                 movzx   eax, byte ptr [esp+eax+1Ch]
.text:08048704                 movzx   eax, al
.text:08048707                 mov     [esp+4], eax
.text:0804870B                 mov     dword ptr [esp], offset a02x ; "%02x"
.text:08048712                 call    _printf
.text:08048717                 add     dword ptr [esp+18h], 1
.text:0804871C
.text:0804871C loc_804871C:                            ; CODE XREF: main+5F↑j
.text:0804871C                 mov     ebx, [esp+18h]
.text:08048720                 mov     eax, [esp+0Ch]
.text:08048724                 add     eax, 4
.text:08048727                 mov     eax, [eax]
.text:08048729                 mov     [esp], eax      ; s
.text:0804872C                 call    _strlen
.text:08048731                 cmp     ebx, eax
.text:08048733                 jb      short loc_80486EA
.text:08048735                 mov     dword ptr [esp], 0Ah ; c
.text:0804873C                 call    _putchar
```

这个值在EAX中，写GDB脚本爆破v4的值

```
set $i=0
set $total=256
while($i<$total)
  b *0x80485b4
  b *0x8048707
  run TWCTF
  set $i=$i+1
  set *(char*)($ebp-0xc)=$i
  continue
  if ($eax==0x95)
    print $i, $i
    x/256xb $esp+0x1c
    set $i=256
  end
  stop
end
```



```
Legend: code, data, rodata, value

Breakpoint 2, 0x08048707 in ?? ()
$1 = 0xd6
0xffffcfbc:     0xd6    0xc9    0xc2    0xce    0x47    0xde    0xda    0x70
0xffffcfc4:     0x85    0xb4    0xd2    0x9e    0x4b    0x62    0x1e    0xc3
0xffffcfcc:     0x7f    0x37    0x7c    0xc8    0x4f    0xec    0xf2    0x45
0xffffcfd4:     0x18    0x61    0x17    0x1a    0x29    0x11    0xc7    0x75
0xffffcfdc:     0x02    0x48    0x26    0x93    0x83    0x8a    0x42    0x79
0xffffcfe4:     0x81    0x10    0x50    0x44    0xc4    0x6d    0x84    0xa0
0xffffcfec:     0xb1    0x72    0x96    0x76    0xad    0x23    0xb0    0x2f
0xffffcff4:     0xb2    0xa7    0x35    0x57    0x5e    0x92    0x07    0xc0
0xffffcffc:     0xbc    0x36    0x99    0xaf    0xae    0xdb    0xef    0x15
0xffffd004:     0xe7    0x8e    0x63    0x06    0x9c    0x56    0x9a    0x31
0xffffd00c:     0xe6    0x64    0xb5    0x58    0x95    0x49    0x04    0xee
0xffffd014:     0xdf    0x7e    0x0b    0x8c    0xff    0xf9    0xed    0x7a
0xffffd01c:     0x65    0x5a    0x1f    0x4e    0xf6    0xf8    0x86    0x30
0xffffd024:     0xf0    0x4c    0xb7    0xca    0xe5    0x89    0x2a    0x1d
0xffffd02c:     0xe4    0x16    0xf5    0x3a    0x27    0x28    0x8d    0x40
0xffffd034:     0x09    0x03    0x6f    0x94    0xa5    0x4a    0x46    0x67
0xffffd03c:     0x78    0xb9    0xa6    0x59    0xea    0x22    0xf1    0xa2
0xffffd044:     0x71    0x12    0xcb    0x88    0xd1    0xe8    0xac    0xc6
0xffffd04c:     0xd5    0x34    0xfa    0x69    0x97    0x9f    0x25    0x3d
0xffffd054:     0xf3    0x5b    0x0d    0xa1    0x6b    0xeb    0xbe    0x6e
0xffffd05c:     0x55    0x87    0x8f    0xbf    0xfc    0xb3    0x91    0xe9
0xffffd064:     0x77    0x66    0x19    0xd7    0x24    0x20    0x51    0xcc
0xffffd06c:     0x52    0x7d    0x82    0xd8    0x38    0x60    0xfb    0x1c
0xffffd074:     0xd9    0xe3    0x41    0x5f    0xd0    0xcf    0x1b    0xbd
0xffffd07c:     0x0f    0xcd    0x90    0x9b    0xa9    0x13    0x01    0x73
```

把数据复制出来编写解密脚本

```python
import re
a = "95eeaf95ef94234999582f722f492f72b19a7aaf72e6e776b57aee722fe77ab5ad9aaeb156729676ae7a236d99b1df4a"
box = '''0xffffcfbc: 0xd6 0xc9 0xc2 0xce 0x47 0xde 0xda 0x70
0xffffcfc4: 0x85 0xb4 0xd2 0x9e 0x4b 0x62 0x1e 0xc3
0xffffcfcc: 0x7f 0x37 0x7c 0xc8 0x4f 0xec 0xf2 0x45
0xffffcfd4: 0x18 0x61 0x17 0x1a 0x29 0x11 0xc7 0x75
0xffffcfdc: 0x02 0x48 0x26 0x93 0x83 0x8a 0x42 0x79
0xffffcfe4: 0x81 0x10 0x50 0x44 0xc4 0x6d 0x84 0xa0
0xffffcfec: 0xb1 0x72 0x96 0x76 0xad 0x23 0xb0 0x2f
0xffffcff4: 0xb2 0xa7 0x35 0x57 0x5e 0x92 0x07 0xc0
0xffffcffc: 0xbc 0x36 0x99 0xaf 0xae 0xdb 0xef 0x15
0xffffd004: 0xe7 0x8e 0x63 0x06 0x9c 0x56 0x9a 0x31
0xffffd00c: 0xe6 0x64 0xb5 0x58 0x95 0x49 0x04 0xee
0xffffd014: 0xdf 0x7e 0x0b 0x8c 0xff 0xf9 0xed 0x7a
0xffffd01c: 0x65 0x5a 0x1f 0x4e 0xf6 0xf8 0x86 0x30
0xffffd024: 0xf0 0x4c 0xb7 0xca 0xe5 0x89 0x2a 0x1d
0xffffd02c: 0xe4 0x16 0xf5 0x3a 0x27 0x28 0x8d 0x40
0xffffd034: 0x09 0x03 0x6f 0x94 0xa5 0x4a 0x46 0x67
0xffffd03c: 0x78 0xb9 0xa6 0x59 0xea 0x22 0xf1 0xa2
0xffffd044: 0x71 0x12 0xcb 0x88 0xd1 0xe8 0xac 0xc6
0xffffd04c: 0xd5 0x34 0xfa 0x69 0x97 0x9f 0x25 0x3d
0xffffd054: 0xf3 0x5b 0x0d 0xa1 0x6b 0xeb 0xbe 0x6e
0xffffd05c: 0x55 0x87 0x8f 0xbf 0xfc 0xb3 0x91 0xe9
0xffffd064: 0x77 0x66 0x19 0xd7 0x24 0x20 0x51 0xcc
0xffffd06c: 0x52 0x7d 0x82 0xd8 0x38 0x60 0xfb 0x1c
0xffffd074: 0xd9 0xe3 0x41 0x5f 0xd0 0xcf 0x1b 0xbd
0xffffd07c: 0x0f 0xcd 0x90 0x9b 0xa9 0x13 0x01 0x73
0xffffd084: 0x5d 0x68 0xc1 0xaa 0xfe 0x08 0x3e 0x3f
0xffffd08c: 0xc5 0x8b 0x00 0xd3 0xfd 0xb6 0x43 0xbb
0xffffd094: 0xd4 0x80 0xe2 0x0c 0x33 0x74 0xa8 0x2b
0xffffd09c: 0x54 0x4d 0x2d 0xa4 0xdc 0x6c 0x3b 0x21
0xffffd0a4: 0x2e 0xab 0x32 0x5c 0x7b 0xe0 0x9d 0x6a
0xffffd0ac: 0x39 0x14 0x3c 0xb8 0x0a 0x53 0xf7 0xdd
0xffffd0b4: 0xf4 0x2c 0x98 0xba 0x05 0xe1 0x0e 0xa3
'''
box = re.sub(r'(0xffff[a-z0-9][a-z0-9][a-z0-9][a-z0-9])',"",box)
p = re.compile(r'(0x[a-z0-9][a-z0-9])')
tmp = p.findall(box)
tmp = [int(i,16) for i in tmp]
print(tmp)
for i in range(len(a)//2):
 print(chr(tmp.index(int(a[i*2:i*2+2],16))),end="")
```

得到flag

```
TWCTF{5UBS717U710N_C1PH3R_W17H_R4ND0M123D_5-B0X}
```