BUUCTF

原创

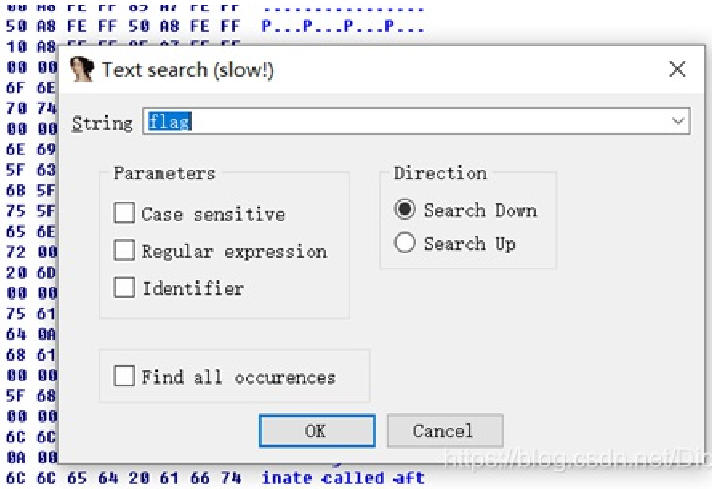
[Dicked](https://blog.csdn.net/Dicked) 于 2020-06-14 17:55:16 发布 151 收藏

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。
本文链接：https://blog.csdn.net/Dicked/article/details/106749274
版权

BUUCTF

easyre

拉入ida直接搜索flag



```
50 A8 FE FF 50 A8 FE FF  P...P...P...P...
10 A8
00 00
6F 6E
70 74
00 00
6E 69
5F 63
6B 5F
75 5F
65 6E
72 00
20 6D
00 00
75 61
64 0A
68 61
00 00
5F 68
00 00
6C 6C
0A 00
6C 6C 65 64 20 61 66 74  inate called aft
```

String: flag

Parameters

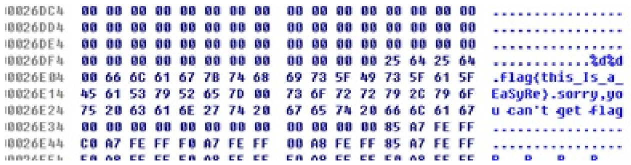
☐ Case sensitive
☐ Regular expression
☐ Identifier

Direction

● Search Down
○ Search Up

☐ Find all occurences

https://blog.csdn.net/Dicked

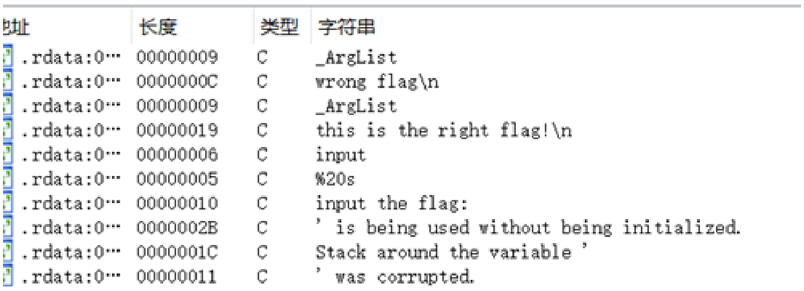


```
00026DC4  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00026DD4  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00026DE4  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00026DF4  00 00 00 00 00 00 00 00  00 00 00 00 25 64 25 64  ............%d%d
00026E04  00 66 6C 61 67 7B 74 68  69 73 5F 49 73 5F 61 5F  .flag{this_Is_a_
00026E14  45 61 53 79 52 65 7D 00  73 6F 72 72 79 2C 79 6F  EaSyRe}.sorry,yo
00026E24  75 20 63 61 6E 27 74 20  67 65 74 20 66 6C 61 67  u can't get flag
00026E34  00 00 00 00 00 00 00 00  00 00 00 00 85 A7 FE FF  ................
00026E44  C0 A7 FE FF F0 A7 FE FF  00 A8 FE FF 85 A7 FE FF  ................
```

得到
flag{this_Is_a_EaSyRe}

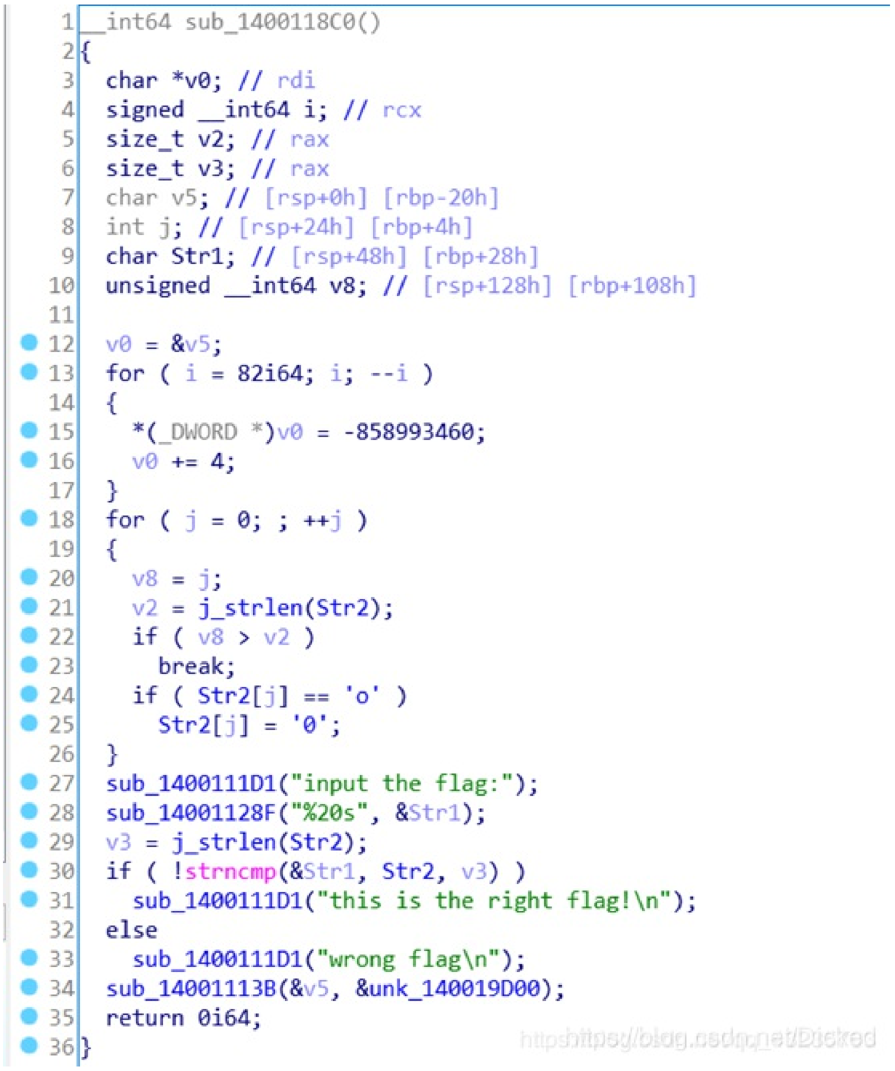
reverse

放入ida中，查看字符串



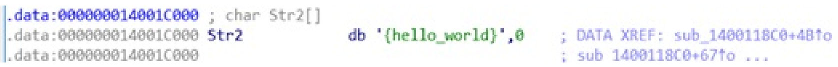
址	长度	类型	字符串
.rdata:0···	00000009	C	_ArgList
.rdata:0···	0000000C	C	wrong flag\n
.rdata:0···	00000009	C	_ArgList
.rdata:0···	00000019	C	this is the right flag!\n
.rdata:0···	00000006	C	input
.rdata:0···	00000005	C	%20s
.rdata:0···	00000010	C	input the flag:
.rdata:0···	0000002B	C	' is being used without being initialized.
.rdata:0···	0000001C	C	Stack around the variable '
.rdata:0···	00000011	C	' was corrupted.

看到 this is the right flag！双击进去查看伪代码



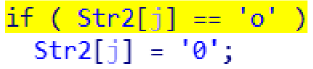
```
__int64 sub_1400118C0()
{
  char *v0; // rdi
  signed __int64 i; // rcx
  size_t v2; // rax
  size_t v3; // rax
  char v5; // [rsp+0h] [rbp-20h]
  int j; // [rsp+24h] [rbp+4h]
  char Str1; // [rsp+48h] [rbp+28h]
  unsigned __int64 v8; // [rsp+128h] [rbp+108h]

  v0 = &v5;
  for ( i = 82i64; i; --i )
  {
    *(_DWORD *)v0 = -858993460;
    v0 += 4;
  }
  for ( j = 0; ; ++j )
  {
    v8 = j;
    v2 = j_strlen(Str2);
    if ( v8 > v2 )
      break;
    if ( Str2[j] == 'o' )
      Str2[j] = '0';
  }
  sub_1400111D1("input the flag:");
  sub_14001128F("%20s", &Str1);
  v3 = j_strlen(Str2);
  if ( !strncmp(&Str1, Str2, v3) )
    sub_1400111D1("this is the right flag!\n");
  else
    sub_1400111D1("wrong flag\n");
  sub_14001113B(&v5, &unk_140019D00);
  return 0i64;
}
```

Str2可疑



```
.data:000000014001C000 ; char Str2[]
.data:000000014001C000 Str2            db '{hello_world}',0    ; DATA XREF: sub_1400118C0+4B↑o
.data:000000014001C000                                         ; sub 1400118C0+67↑o ...
```

再看伪代码多了一个if语句



```
if ( Str2[j] == 'o' )
  Str2[j] = '0';
```

根据这个吧o换成0得出flag

flag{hell0_w0rld}