# 两道花指令题目

原创

pipixia233333 于 2019-09-29 18:16:07 发布 782 收藏
分类专栏： 逆向之旅

本文链接：https://blog.csdn.net/qq_41071646/article/details/94872655



逆向之旅 专栏收录该内容
128 篇文章 2 订阅
订阅专栏

XCTF 3rd-ZCTF-2017 QExtend-300

这个题目 还是比较难的 一个就是花指令的恢复 一个就是题目的分析

这个有两个修复的地方 一个就是 pop/call 直接把pop call nop 就行

第二点就是 switch 每个函数的返回地址增加了 第一个增加一 第二个增加二 所以。。 需要把 函数地址下面的东西给nop了

然后nop玩 程序就很明了了

```
19  scanf("%s", Str1);
20  v0 = strlen(Str1);
21  if ( v0 < 0x20
22    && (*(&v2 + v0) = 0, Str1[4] = 0, !stricmp(Str1, "ZCTF"))
23    && sub_4026D0(printf, &Str1[5])
24    && sub_402800(printf, &Str1[5]) )
25  {
26    printf("Congratulations! Key Correct!\n");
27    result = 0;
28  }
29  else
30  {
31    printf("Sorry! Key Err!\n");
```

IDA View-A | Pseudocode-A | Hex View-1 | Structures | Enums | Imports | Exports

```
28        sub_4025E0(bytes);                    // 4-9 0-4
29        goto LABEL_9;
30      case 3u:
31        sub_402590(bytes);                    // 4-9 10-14
32        goto LABEL_9;
33      case 4u:
34        sub_402540(bytes);                    // 10-14 4-9
35        goto LABEL_9;
36      case 5u:
37        sub_4024F0(bytes);                    // 10-14 0-4
38LABEL_9:
39        if ( sub_402490(bytes) )              // 条件就是a[i]<a[i+1] 不过条件就是 5-9 是这样就行了
40          continue;
41        result = 0;
42        break;
43      default:
44        result = bytes[5] && bytes[6] && bytes[7] && bytes[8] && bytes[9];// 这里的值必须为真
45        break;
46    }
47    return result;
48  }
```



```
2{
3  _DWORD *v2; // eax
4  const char *v3; // eax
5  char v5; // [esp+24h] [ebp-34h]
6  bool v6; // [esp+43h] [ebp-15h]
7  char v7; // [esp+44h] [ebp-14h]
8  char *Str1; // [esp+54h] [ebp-4h]
9
10  Str1 = a1;
11  sub_401760(&v7);
12  md5(strlen(input), &v7, input);                // md5
13  v2 = strhex(&v7, &v5);
14  v3 = sub_401030(v2);
15  v6 = stricmp(Str1, v3) == 0;
16  sub_401000(&v5);
17  return v6 != 0;
18}
```

```
BOOL result; // eax
int v4; // [esp+1Ch] [ebp-4h]

v4 = 0;                                 // 00 00 00 AA DD
                                        // 00 00 00 00 BB
                                        // 00 00 00 CC EE
                                        // ->
                                        // 00 00 00 00 00
                                        // AA BB CC DD EE
                                        // 00 00 00 00 00
                                        //
                                        //
while ( 2 )
{
  v2 = input[v4++];
  switch ( ((v2 % 16) - 1) )
  {
    case 0u:
      sub_402680(bytes);                // 0-4   5-9
      goto LABEL_9;
    case 1u:
      sub_402630(bytes);                // 0-4  10-14
```



本来求 这个东西的时候想用 搜索解决 然后一直 没有解决 只能手看 然后下面是搜索代码 我大概知道了哪里错了 就是 状态没有判断是不是已经走过了 但是确实 这个状态不是很好判断 每次开一个数组 去维护这个 状态表 感觉会有瑕疵

然后求解的时候想用z3 发现md5这里 怎么也过不去 z3 好像不能md5 求值把

```
#include<stdio.h>
#include<string.h>
#include<algorithm>
#include<vector>
#include<iostream>
#include<map>
#include<time.h>
#include<queue>
#include "windows.h"
using namespace std;
struct node
{
    int s[15]= {0,0,0,1,4,0,0,0,0,2,0,0,0,3,5}; //状态
    int value[100]= {0}; //然后是向那个方向移动 保持记录
    int sum=0;
};
int k[15];
bool judge()//判断搜索是否违反了边界
{
    for(int i=0; i<4; i++)
    {
        if(k[i]>k[i+1])
        {
            return 0;
        }
    }
    for(int i=5; i<9; i++)
    {
        if(k[i]>k[i+1])
        {
```

```
            return 0;
        }
    }
    for(int i=10; i<14; i++)
    {
        if(k[i]>k[i+1])
        {
            return 0;
        }
    }
    return 1;
}
bool issuccess(int *xss)//判断是否成功
{
    for(int i=0; i<15; i++)
    {
        if(i>=5&&i<=9)
        {
            if(xss[i]!=i-4)
                return 0;
        }
        if(xss[i]!=0)
            return 0;
    }
    return 1;
}
bool mymove(int l1,int r1,int l2,int r2)//移动东西
{
    int v2; // esi
    int v3; // eax
    /*for(int i=0;i<15;i++)
    {
        printf("%d",k[i]);
    }*/
   // printf("\n");
    if (k[r1])
    {
        v2 = r1;
        v3 = r2;
        do
        {
            if ( !k[v2] )
                break;
            --v2;
        }
        while ( v2 >= l1 );
        do
        {
            if ( !k[v3] )
                break;
            --v3;
        }
        while ( v3 >= l2 );
        k[v3] = k[v2 + 1];
        k[v2 + 1] = 0;
        if(!judge())
        {
            /* for(int ii=0;ii<15;ii++)
             {
                 if(ii%5==0)
```

```
                    printf("\n");
                printf("%d",k[ii]);

            }
            printf("\n**************************\n");
*/
            return 0;
        }
        //printf("1\n");
        return 1;
    }
    return 0;
}
void slove()
{
    queue<node>Mynode;
    int i;
    int j;
    node ss;
    Mynode.push(ss);
    while(!Mynode.empty())
    {
        /*
        0 1->2
        1 1->3
        2 2->1
        3 2->3
        4 3->2
        5 3->1
        */
        ss=Mynode.front();
        Mynode.pop();
        if(issuccess(ss.s))
        {
            for(i=0; i<ss.sum; i++)
            {
                printf("%d",ss.value[i]);
            }
            return;
        }
        for(i=0; i<6; i++)
        {
            for(j=0; j<15; j++)
            {
                k[j]=ss.s[j];
            }
            switch(i)
            {
            case 0:
                if(mymove(0,4,5,9))
                {
                    for(j=0; j<15; j++)
                    {
                        ss.s[j]=k[j];
                    }
                    ss.value[ss.sum]=i;
                    ss.sum++;
                    Mynode.push(ss);
                    ss.sum--;
```

```
            }
            break;
        case 1:
            if(mymove(0,4,10,14))
            {
                for(j=0; j<15; j++)
                {
                    ss.s[j]=k[j];
                }
                ss.value[ss.sum]=i;
                ss.sum++;
                Mynode.push(ss);
                ss.sum--;
            }
            break;
        case 2:

            if( mymove(4,9,0,4))
            {
                for(j=0; j<15; j++)
                {
                    ss.s[j]=k[j];
                }
                ss.value[ss.sum]=i;
                ss.sum++;
                Mynode.push(ss);
                ss.sum--;
            }
            break;
        case 3:


            if(mymove(4,9,10,14))
            {
                for(j=0; j<15; j++)
                {
                    ss.s[j]=k[j];
                }
                ss.value[ss.sum]=i;
                ss.sum++;
                Mynode.push(ss);
                ss.sum--;
            }
            break;
        case 4:

            if(mymove(10,14,4,9))
            {
                for(j=0; j<15; j++)
                {
                    ss.s[j]=k[j];
                }
                ss.value[ss.sum]=i;
                ss.sum++;
                Mynode.push(ss);
                ss.sum--;
            }
            break;
        case 5:
```

```
                if( mymove(10,14,0,4))
                {
                    for(j=0; j<15; j++)
                    {
                        ss.s[j]=k[j];
                    }
                    ss.value[ss.sum]=i;
                    ss.sum++;
                    Mynode.push(ss);
                    ss.sum--;
                }
            }
        }

    }
}
int main()
{
    slove();
    printf("程序运行成功!\n");
    return 0;
}
```

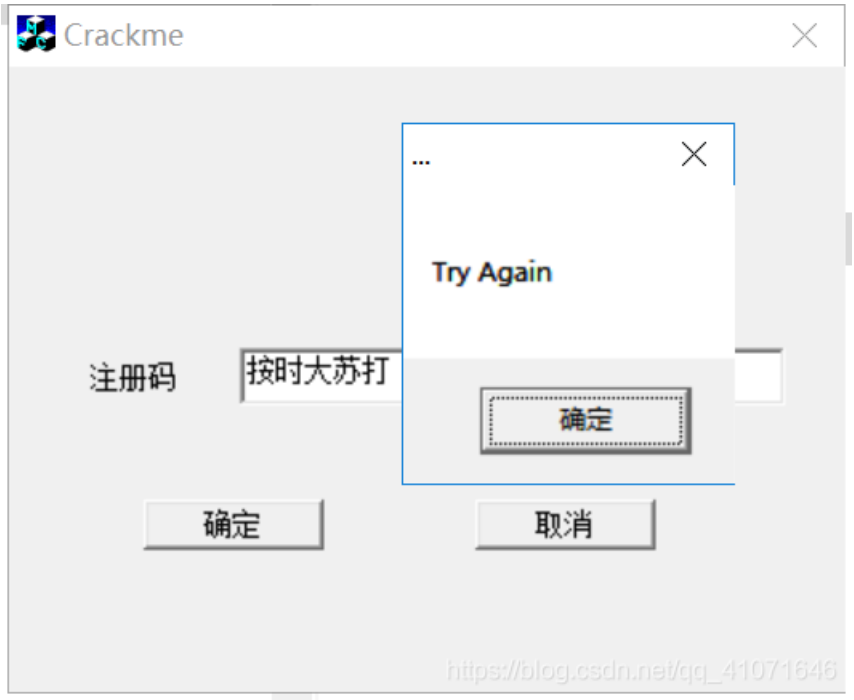
然后下面就是求flag

```
#!/usr/bin/env pytho3
# -*- coding:utf-8 -*-

#from z3 import*
from hashlib import md5
'''
由于我们知道了低位
那么我们就可以直接爆破高位
'''
#053254104123104524104
#164365215234215635215
if __name__ =="__main__":
    high_number=[0x20,0x30,0x40,0x50,0x60,0x70]
    md5_str = '0f2e7e447593ec9af3463e9c8745b892'
    high=[0,0,0,0,0,0]
    low_number=[1,2,3,4,5,6]
    number=[0,0,0,0,0,0,0,0]
    for high[0] in high_number:
        for high[1] in high_number:
            for high[2] in high_number:
                for high[3] in high_number:
                    for high[4] in high_number:
                        for high[5] in high_number:
                            for i in range(6):
                                number[i]=chr(high[i]+low_number[i])
                            str_md5= number[0]+number[5]+number[3]+number[2]+number[5]+number[4]+number[1]+
                            number[0]+number[4]+number[1]+number[2]+number[3]+number[1]+number[0]+\
                            number[4]+number[5]+number[2]+number[4]+number[1]+number[0]+number[4]
                            if md5(str_md5.encode("utf8")).hexdigest() == md5_str :
                                print('ZCTF{%s}' %str_md5)
```

另一道题目不知道出处是哪里的 是一个朋友发的==

如果 有老哥知道出处 还请说一下 我加上出处

ok 先看看题目的样子




大概就是这个样子

然后ida 找到按钮事件准备f5的时候 出了错===

这个花指令 比上面的多了几个地方

```
2BFA                jb      short near ptr loc_402C00+3
2BFC                jb      short loc_402C38
2BFE                jnb     short loc_402C38
2C00
2C00 loc_402C00:                            ; CODE XREF: .text:loc_402BFA↑j
2C00                call    near ptr 3B52C78h
2C05                jz      short near ptr loc_402C07+1
2C07
2C07 loc_402C07:                            ; CODE XREF: .text:00402C05↑j
2C07                call    near ptr 0E8F97D61h
2C0C                add     eax, [eax]
2C0E                add     [esi-42h], dl
2C11                pusha
2C12                push    eax
2C13                inc     eax
2C14                add     [esi+59h], bl
2C17                jb      short loc_402BD1
2C19                jb      short near ptr loc_402C3C+4
2C1B                jnb     short near ptr loc_402C3C+4
2C1D
2C1D loc_402C1D:                            ; CODE XREF: .text:00402C46↓j
2C1D                jmp     near ptr 4607DD95h
2C22 ; ---------------------------------------------------------------------------
2C22
2C22 loc_402C22:                            ; CODE XREF: .text:00402BD5↑j
```

发现了很多的红色点 ===

可以变成data

```
.text:00402C01                db  73h ; s
.text:00402C02                db    0
.text:00402C03 ; ---------------------------------------------------------------------------
.text:00402C03
.text:00402C03 loc_402C03:                            ; CODE XREF: .text:loc_402BF
.text:00402C03                jnz     short loc_402C08
.text:00402C05                jz      short loc_402C08
.text:00402C05 ; ---------------------------------------------------------------------------
.text:00402C07                db 0E8h
.text:00402C08 ; ---------------------------------------------------------------------------
.text:00402C08
.text:00402C08 loc_402C08:                            ; CODE XREF: .text:loc_402C0
.text:00402C08                                        ; .text:00402C05↑j
.text:00402C08                push    ebp
.text:00402C09                push    ecx
```



然后 在 把圈到的地方 变成代码就好了 然后 再把他们nop了 就ok了

还算是有不少地方的==

DA View-A | Pseudocode-A | Hex View-1 | Structures | Enums | Imports | Exports

```
4           __debugbreak();
5       }
6       *(&v52 + j) = (&(&v54)[9 * j])[i];
7     }
8   }
9   v52 = 0;
0   *v53 = 0;
1   v53[4] = 0;
2   for ( i = 0; i < 3; ++i )
3   {
4     for ( j = 0; j < 3; ++j )
5     {
6       v52 = 0;
7       *v53 = 0;
8       v53[4] = 0;
9       for ( l = 0; l < 3; ++l )
0       {
1         for ( m = 0; m < 3; ++m )
2         {
3           for ( k = 0; k < m + 3 * l; ++k )
4           {
5             if ( *(&v52 + k) == (&(&v54)[9 * (l + 3 * i)])[m + 3 * j] )
```



00002A5B sub_402100:341 (402A5B)

发现都是不让相等的 而且都是 9*9的数据　不让一列的数据相等 不让 一行的数据相等 不让3*3里面相等 那不就是 数独么

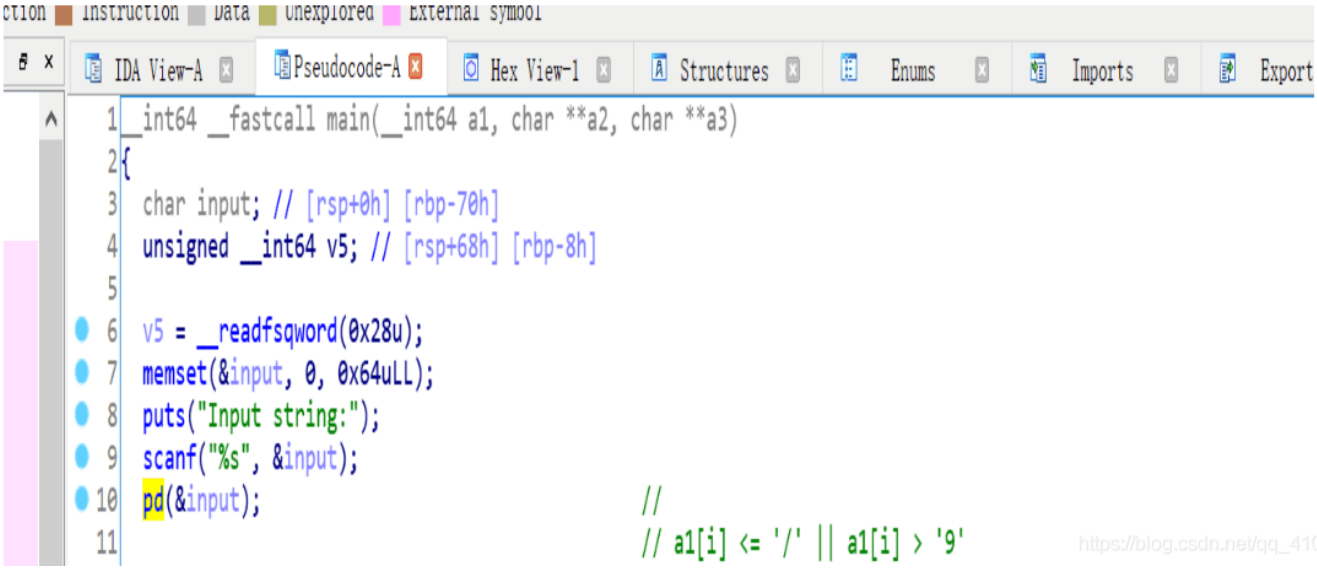
关于数独 南邮有一个题目 可以去看看

https://blog.csdn.net/qq_41071646/article/details/88554159

Single 这道题 有点搞 实话实说 真的有点搞 一开始 都被这个题 搞懵了

这 判断条件都是什么啊

看懂之后 其实就会感觉很简单

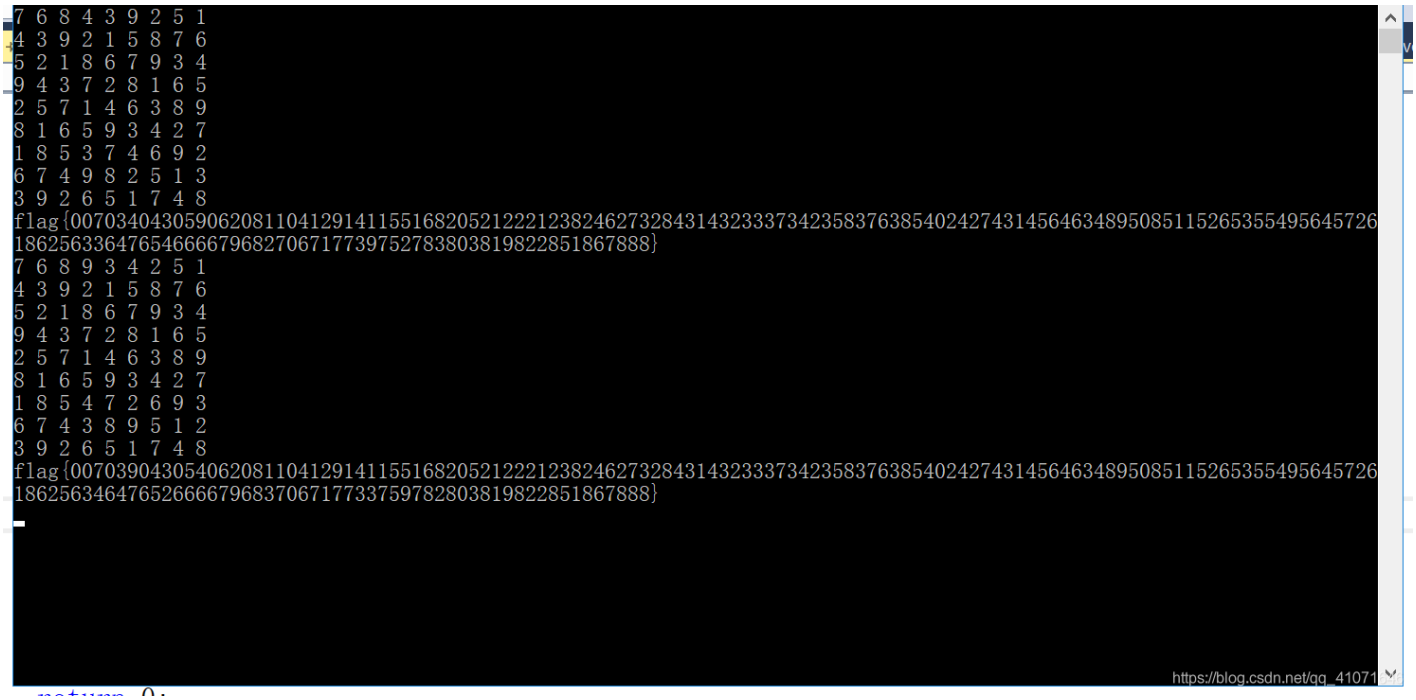



然后根据脚本改一下 直接跑就行 =

```
// re_exp.cpp : 定义控制台应用程序的入口点。
//

#include "stdafx.h"
#include<stdio.h>
#include<string.h>
#include<algorithm>
#include<vector>
#include<iostream>
#include<map>
#include<time.h>
#include<queue>
#include "windows.h"
using namespace std;
int s[9][9] = { 0, 6, 8, 0, 0, 0, 0, 5, 0, 0, 3,
 0, 2, 0, 0, 0, 7, 6, 0, 0, 0, 0, 0, 7, 9, 0, 0,
 9, 0, 0, 0, 0, 0, 1, 0, 0, 0, 5, 0, 0, 4, 0, 0,
 8, 0, 0, 0, 0, 0, 0, 3, 0, 0, 7, 1, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 4, 0, 8, 0, 5, 1, 0, 0, 0, 0, 6, 5, 0, 0, 4, 0
};
int po[9][9] = { 0, 6, 8, 0, 0, 0, 0, 5, 0, 0, 3,
0, 2, 0, 0, 0, 7, 6, 0, 0, 0, 0, 0, 7, 9, 0, 0,
9, 0, 0, 0, 0, 0, 1, 0, 0, 0, 5, 0, 0, 4, 0, 0,
8, 0, 0, 0, 0, 0, 0, 3, 0, 0, 7, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 4, 0, 8, 0, 5, 1, 0, 0, 0, 0, 6, 5, 0, 0, 4, 0
};
struct ppx
{
 int x;
 int y;
} ss[100];
```

```
} ss[100];
int sum = 0;
bool pd(int len, int t)
{
 for (int i = 0; i<9; i++)
 {
  if (s[i][ss[len].y] == t || s[ss[len].x][i] == t)
  {
   return 0;
  }
 }
 int x = (ss[len].x / 3) * 3;
 int y = (ss[len].y / 3) * 3;
 for (int i = x; i<x + 3; i++)
 {
  for (int j = y; j<y + 3; j++)
  {
   if (s[i][j] == t)
   {
    return 0;
   }
  }
 }
 return 1;
}
void dfs(int len)
{
 if (len == sum)
 {
  for (int i = 0; i<9; i++)
  {
   for (int j = 0; j<9; j++)
   {
    printf("%d ", s[i][j]);
   }
   printf("\n");
  }
  printf("flag{");
  for (int i = 0; i<9; i++)
  {
   for (int j = 0; j<9; j++)
   {
    if (po[i][j] != 0)
    {
     //printf("0");
     continue;
    }
    printf("%d%d%d",i,j,s[i][j]);
   }

  }
  printf("}\n");
  return;
 }
 for (int i = 1; i<10; i++)
 {
  if (pd(len, i))
  {
   s[ss[len].x][ss[len].y] = i;
   dfs(len + 1);
```

```
   s[ss[len].x][ss[len].y] = 0;
  }
 }
 return;
}
int main()
{
 for (int i = 0; i<9; i++)
 {
  for (int j = 0; j<9; j++)
  {
   if (s[i][j] == 0)
   {
    ss[sum].x = i;
    ss[sum].y = j;
    sum++;
   }
  }
 }
 dfs(0);

 getchar();
 system("pause");
 return 0;
}
```




发现两个答案都对===

flag{007034043059062081104129141155168205212221238246273284314323337342358376385402427431⁴

flag{007039043054062081104129141155168205212221238246273284314323337342358376385402427431⁴



链接：https://pan.baidu.com/s/1eFddEUqW9zp0_JIJWXBvvQ
提取码：k2rm
复制这段内容后打开百度网盘手机App，操作更方便哦