

Keyboard

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     char v4; // [esp+0h] [ebp-2F0h]
4     char v5; // [esp+0h] [ebp-2F0h]
5     HANDLE hObject; // [esp+8h] [ebp-2E8h]
6     int i; // [esp+1Ch] [ebp-2D4h]
7     struct _PROCESS_INFORMATION ProcessInformation; // [esp+24h] [ebp-2CCh] BYREF
8     struct _STARTUPINFOA StartupInfo; // [esp+34h] [ebp-2BCh] BYREF
9     char v10[256]; // [esp+78h] [ebp-278h] BYREF
10    CHAR CommandLine[104]; // [esp+178h] [ebp-178h] BYREF
11    char Buf1[260]; // [esp+1E0h] [ebp-110h] BYREF
12    char v13[8]; // [esp+2E4h] [ebp-Ch] BYREF
13
14    memset(v10, 0, sizeof(v10));
15    strcpy(v13, "D0g3");
16    memset(Buf1, 0, sizeof(Buf1));
17    strcpy(
18        CommandLine,
19        "cmd /c start https://www.bilibili.com/video/BV1uT4y1P7CX/?spm_id_from=333.788.recommend_more_video.-1");
20    memset(&StartupInfo, 0, sizeof(StartupInfo));
21    memset(&ProcessInformation, 0, sizeof(ProcessInformation));
22    hObject = (HANDLE)sub_401430(v10, v13, &v13[strlen(v13) + 1] - &v13[1]);
23    sub_401840("please input your flag:", v4);
24    sub_401880("%s", (char)Buf1);
25    for (i = 0; i < 25; ++i)
26        Buf1[i] = (i ^ Buf1[i]) + 12;
27    sub_401340(v10, Buf1, 25);
28    if ( !memcmp(Buf1, &unk_404640, 0x19u) )
29        CreateProcessA(0, CommandLine, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation);
30    else
31        sub_401840("Sorry you are wrong!!!", v5);
32    CloseHandle(ProcessInformation.hProcess);
33    CloseHandle(ProcessInformation.hThread);
34    CloseHandle(hObject);
35    system("pause");
36    return 0;
37 }
```

打开长这样，看那个BV号



所以这个RC4解出来的flag也是假的

查看fn函数的汇编窗口，有花指令，

```
.text:004011FF test     eax, eax
.text:00401201 jz       short near ptr loc_401205+1
.text:00401203 jnz      short near ptr loc_401205+1
.text:00401205
.text:00401205 loc_401205: CODE XREF: .text:00401201
```

nop掉即可

然后进入该函数，发现是个迷宫题，40x40的。

```

6   v2 = *a1;
7   strcpy(v3, "Success!!!");
8   switch ( v2 )
9   {
10    case 'A':
11        --LINE;
12        break;
13    case 'D':
14        ++LINE;
15        break;
16    case 'S':
17        ++ROW;
18        break;
19    case 'W':
20        --ROW;
21        break;
22    default:
23        break;
24  }
25  if ( ROW < 0 || ROW >= 40 || LINE < 0 || LINE >= 40 || MAZE[40 * ROW + LINE] == 1 )
26  {
27      Sleep(0xAu);
28      exit(0);
29  }
30  ++COUNT;
31  if ( MAZE[40 * ROW + LINE] == 2 && COUNT == 106 )
32      printf("\n%s", v3);
33  return 1;
34 }

```

然后注意TLS函数

```

3   int result; // eax
4
5   ROW = 2;
6   LINE = 2;
7   MAZE[985] = 0;
8   result = NtCurrentTeb()->ProcessEnvironmentBlock;
9   if ( *(result + 2) )
10  {
11      ROW = 0;
12      LINE = 0;
13      result = 23;
14      MAZE[823] = 1;
15  }
16  return result;
17 }

```

里面有几个把迷宫的部分值替换的操作,以及起始坐标。

写脚本得出最原始的迷宫

[illegible]

[illegible]

```

0x01, 0x00, 0x00, 0x01, 0x01, 0x01, 0x00, 0x00, 0x00, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x00, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x00, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x00,
0x02]
for i in range(0,1600,40):
    for j in range(i,i+40):
        print((a[j]),end='')
    print()

```

根据TLS里的替换相应位置的数字即可。

然后就是WASD走迷宫了，得出来的路径md5一下就行

SDSDDDWDDSSSSSAASSDDDDDDWDDSSSSDDDDSSSDSSSSSSDSSSDSSSASSDDDDWWWWW
 WWDDDDSSDSSSDSSSSAAASSSDDDDDWDDSSSSD

flag:D0g3{1bf4c14e20c7f8559f0c72ad4605c8d5}

签到

```

17 sub_181740(v15);
18 v16 = 0;
19 printf(std::cout, "input:");
20 scanf(std::cin, v15);
21 if ( Concurrency::details::_CancellationTokenRegistration::_GetToken(v15) == 16 )
22 {
23     for ( i = 0; i < Concurrency::details::_CancellationTokenRegistration::_GetToken(v15); ++i )
24     {
25         v7[i] = *sub_181790(i);
26         key = byte_18609B + (byte_18609A << 8) + (byte_186099 << 16) + (byte_186098 << 24);
27         Token = Concurrency::details::_CancellationTokenRegistration::_GetToken(v15);
28         xxtea(v7, Token);
29         tea_changed(v7, &key);
30         tea_changed(v8, &key);
31         tea_changed(v9, &key);
32         tea_changed(v10, &key);
33         tea_changed(v11, &key);
34         tea_changed(v12, &key);
35         tea_changed(v13, &key);
36         tea_changed(v14, &key);
37         for ( j = 0; j < 16; ++j )
38         {
39             if ( v7[j] != compare_data[j] )
40             {
41                 printf(std::cout, "answer error!");
42                 v16 = -1;
43                 sub_181770(v15);
44                 return 0;
45             }
46         }
47         printf(std::cout, "Good!");
48         v16 = -1;
49         sub_181770(v15);
50         return 0;
51     }
52 }
000009F9 main:29 (1815F9)

```

main函数如上所示

先将输入用原版xxtea加密，在用改了一点的tea加密，

```

key = *a2;
v3 = a2[1];
v4 = a2[2];
result = a2;
v5 = a2[3];
v6 = 0;
for ( i = 0; i < 0x20; ++i )
{
    v6 -= 1640531527;
    *a1 += (v3 + (a1[1] >> 5)) ^ (v6 + a1[1]) ^ (key + 16 * a1[1]) ^ 0x10;
    result = a1[1] + ((v5 + (*a1 >> 5)) ^ (v6 + *a1) ^ (v4 + 16 * *a1) ^ 0x10);
    a1[1] = result;
}
return result;
}

```

可以看到tea加密只是在原本基础上多异或一个0x10

```

#include <stdio.h>
#include <stdint.h>

void decrypt (uint32_t* v, uint32_t* k) {
    uint32_t v0=v[0], v1=v[1], sum=0xC6EF3720, i;
    uint32_t delta=0x9e3779b9;
    uint32_t k0=k[0], k1=k[1], k2=k[2], k3=k[3];
    for (i=0; i<32; i++) {
        v1 -= (((v0<<4) + k2) ^ (v0 + sum) ^ ((v0>>5) + k3))^0x10;
        v0 -= (((v1<<4) + k0) ^ (v1 + sum) ^ ((v1>>5) + k1))^0x10;
        sum -= delta;
    }
    v[0]=v0; v[1]=v1;
}

int main()
{
    uint32_t v0[2] = {0xC36683EF, 0xFE447C91};
    uint32_t v1[2] = {0x7C7DBEFD, 0x5BE93F01};
    uint32_t v2[2] = {0x639B9622, 0xB7912A32};
    uint32_t v3[2] = {0x43F9232A, 0x041BE559};
    uint32_t v4[2] = {0x6CFF2066, 0x6DD8E9AA};
    uint32_t v5[2] = {0x707E69F0, 0xE03A64FF};
    uint32_t v6[2] = {0x95755750, 0x8BED264B};
    uint32_t v7[2] = {0xE95C2ED1, 0x40883962};

    uint32_t k[4]={0x44306733,0,0,0};
    decrypt(v0, k);
    decrypt(v1, k);
    decrypt(v2, k);
    decrypt(v3, k);
    decrypt(v4, k);
    decrypt(v5, k);
    decrypt(v6, k);
    decrypt(v7, k);
    printf("解密后的数据: 0x%x, 0x%x, ", v0[0], v0[1]);
    printf("0x%x, 0x%x, ", v1[0], v1[1]);
    printf("0x%x, 0x%x, ", v2[0], v2[1]);
}

```

```

printf("0x%x, 0x%x, ", v3[0], v3[1]);
printf("0x%x, 0x%x, ", v4[0], v4[1]);
printf("0x%x, 0x%x, ", v5[0], v5[1]);
printf(" 0x%x, 0x%x, ", v6[0], v6[1]);
printf("0x%x, 0x%x\n", v7[0], v7[1]);

return 0;

}

```

TEA解密的脚本如上，将得到的数据再用xxtea解密

```

#include <stdio.h>
#include <stdlib.h>
#define DELTA 0x9e3779b9
int main()
{
    unsigned int v[] = {0xadc30805, 0xa645cd26, 0xe719ccf0, 0x1c139bf7,
0xd588143, 0x5965483d, 0x675b1074, 0x8641a979, 0x1e51109a, 0x9ecbc2c,
0x7b6b9e0b, 0x4811094c, 0x84ea65be, 0x56017206, 0xc39297b7, 0x727dba8b};
    unsigned int key[] = {0x44306733,0,0,0};
    unsigned int sum = 0;
    unsigned int y,z,p,rounds,e;
    int n = 16; //
    int i = 0;
    rounds = 6 + 52/n;
    y = v[0];
    sum = (rounds*DELTA);
    do
    {
        e = sum >> 2 & 3;
        for(p=n-1;p>0;p--)
        {
            z = v[p-1];
            v[p] = (v[p] - (((z>>5)^(y<<2))+((y>>3)^(z<<4))) ^
((key[(p^e)&3]^z)+(y ^ sum)))) ;
            y = v[p];
        }
        z = v[n-1];
        v[0] = (v[0] - (((key[(p^e)&3]^z)+(y ^ sum)) ^ ((y<<2)^(z>>5))+
((z<<4)^(y>>3))))) ;
        y = v[0];
        sum = (sum-DELTA);
    }while(--rounds);
    for(i=0;i<n;i++)
    {
        printf("%c",v[i]);
    }
    return 0;
}

```

flag:D0g3{l_h3ve@Tea}

