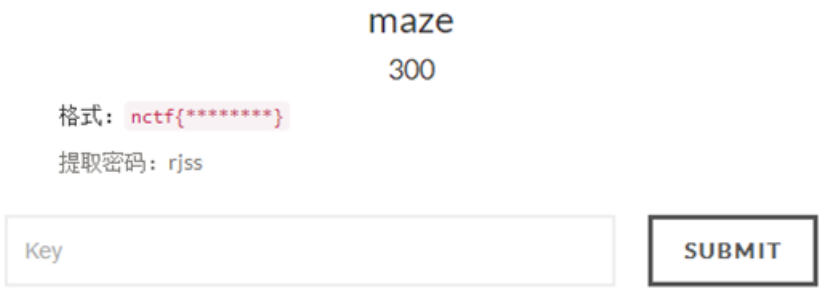
南邮CTF逆向题第五道maze解题思路

原创

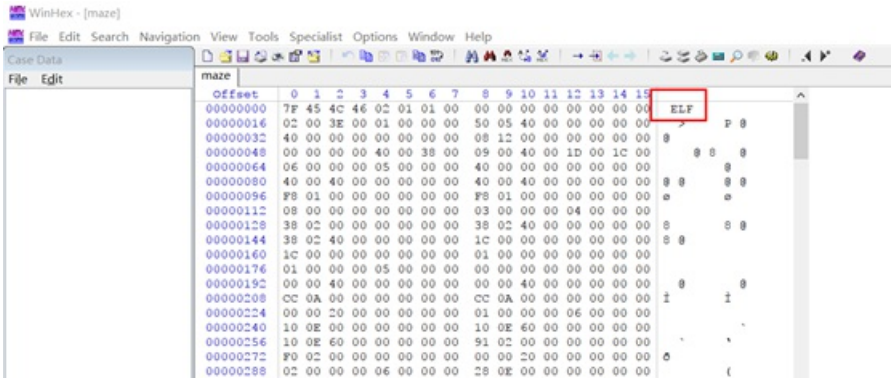
iqiqiya 于 2017-12-24 09:48:12 发布
分类专栏： ------南邮CTF 我的CTF进阶之路 文章标签： maze 逆向 南邮CTF writeup CTF

如题



先百度一下名字 万一有收获呢 猜测可能考到迷宫算法

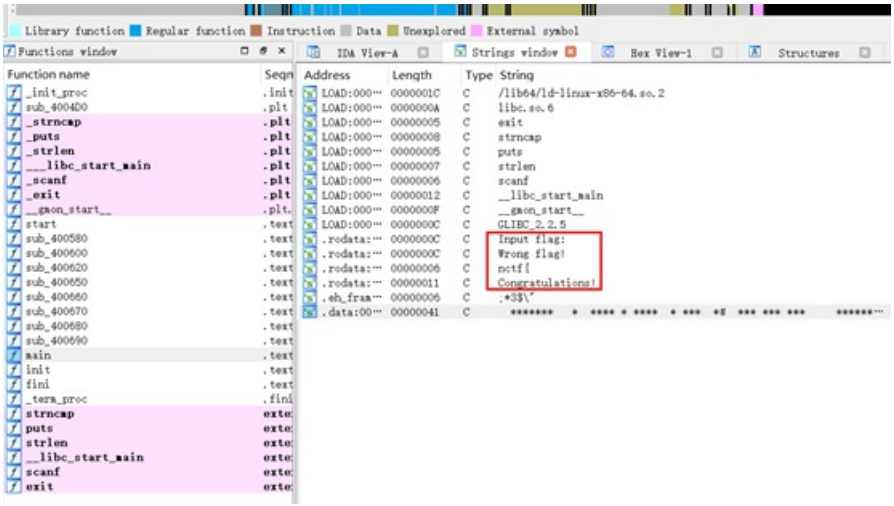
maze_百度翻译



依旧是ELF

直接载入IDA

我一般习惯先shift+f12看下字符串



再看看main

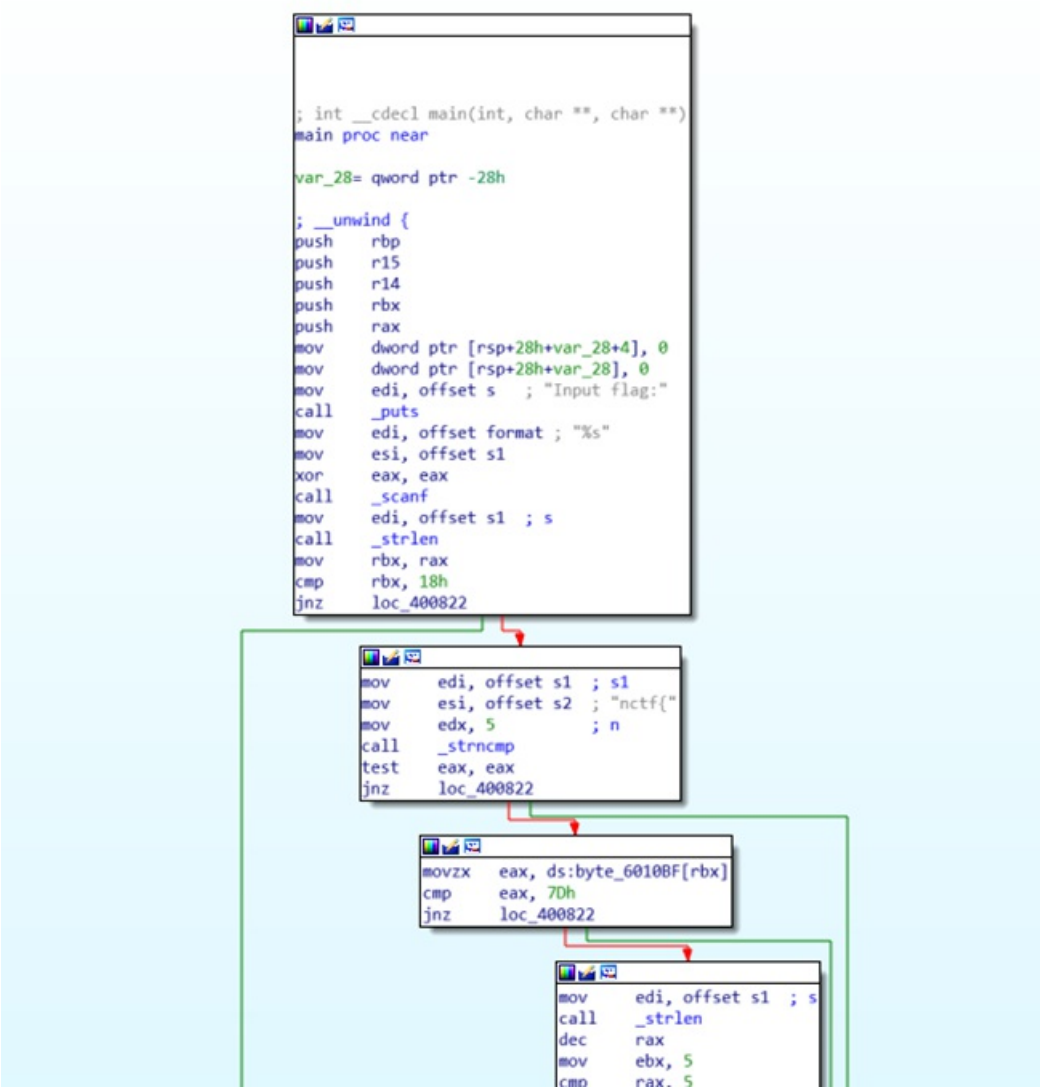
```
__int64 __fastcall main(__int64 a1,char**a2,char**a3)

{

constchar*v3;// rsi

signed __int64 v4;// rbx

signedint v5;// eax

char v6;// bp

char v7;// al

constchar*v8;// rdi

__int64 v10;// [rsp+0h] [rbp-28h]



v10 =0LL;

puts("Input flag:");

scanf("%s",&s1,0LL);//限制获取输入字符串长度为24且必须开头5个字符为"nctf{" 最后一个字符位"}"

if( strlen(&s1)!=24||(v3 ="nctf{", strncmp(&s1,"nctf{",5uLL))||*(&byte_6010BF +24)!=125)

{

LABEL_22:

puts("Wrong flag!");

exit(-1);

}
```

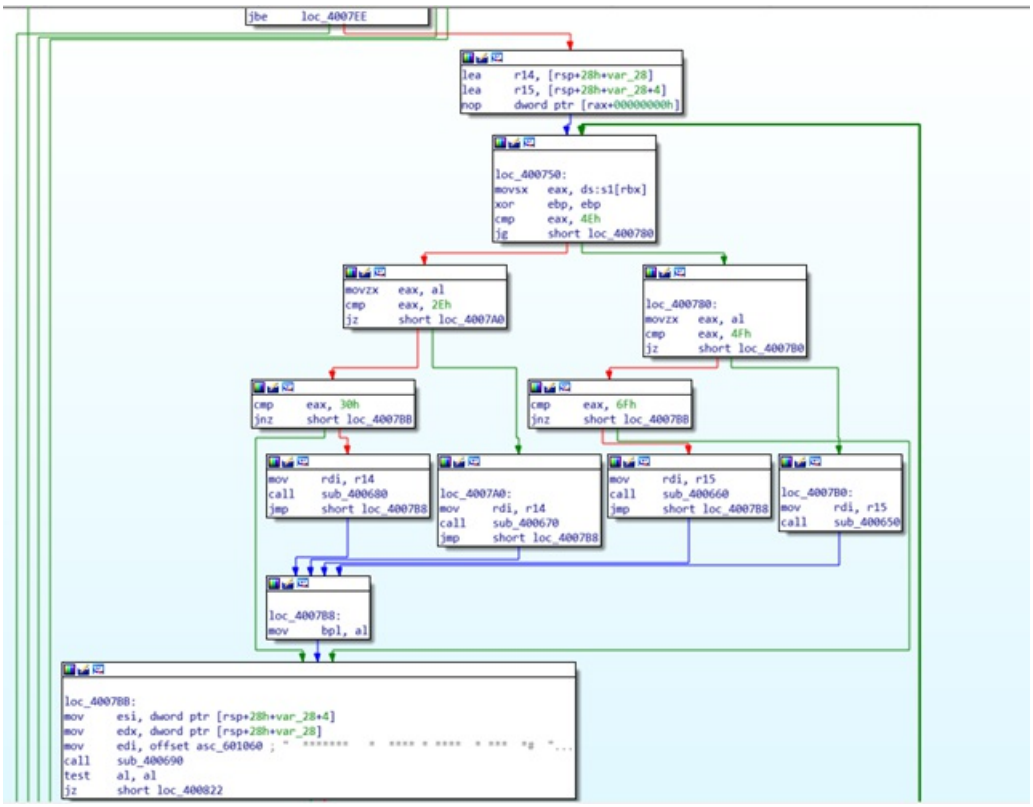
```
v4 =5LL;
if( strlen(&s1)-1>5)
{
while(1)
{
v5 =*(&s1 + v4);
v6 =0;
if( v5 >78)
{
v5 =(unsigned __int8)v5;
if((unsigned __int8)v5 ==79)
{
v7 = sub_400650((char*)&v10 +4, v3);
goto LABEL_14;
}
if( v5 ==111)
{
v7 = sub_400660((char*)&v10 +4, v3);
goto LABEL_14;
}
}
else
{
v5 =(unsigned __int8)v5;
if((unsigned __int8)v5 ==46)
{
v7 = sub_400670(&v10, v3);
goto LABEL_14;
}
if( v5 ==48)
```

```
{
v7 = sub_400680(&v10, v3);
LABEL_14:
v6 = v7;
goto LABEL_15;
}
}
LABEL_15:
v3 =(constchar*)HIDWORD(v10);
if(!(unsigned __int8)sub_400690(asc_601060, HIDWORD(v10),(unsignedint)v10))
goto LABEL_22;
if(++v4 >= strlen(&s1)-1)
{
if( v6 )
break;
LABEL_20:
v8 ="Wrong flag!";
goto LABEL_21;
}
}
}
if( asc_601060[8*(signedint)v10 + SHIDWORD(v10)]!=35)
goto LABEL_20;
v8 ="Congratulations!";
LABEL_21:
puts(v8);
return0LL;
}
```

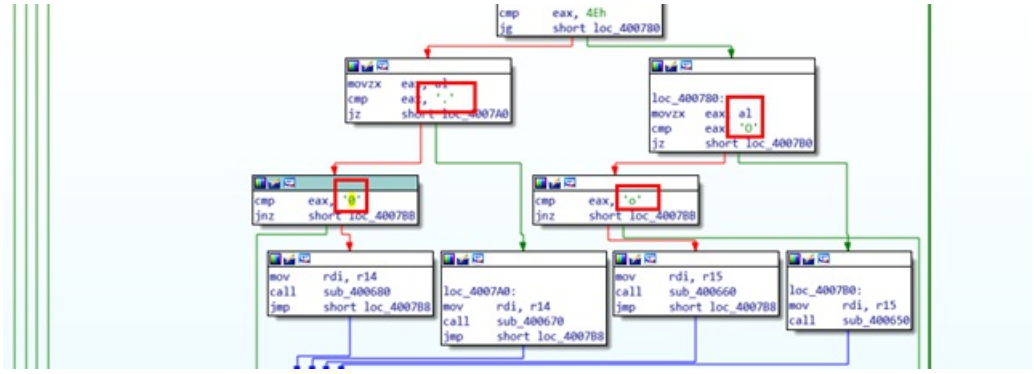
```
12   puts("Input flag:");
13   scanf("%s", &s1, 0LL);//限制获取输入字符串长度为24且必须开头5个字符为"nctf{" 最后一个字符位"}"
14   if ( strlen(&s1) != 24 || (v3 = "nctf{", strncmp(&s1, "nctf{", 5uLL)) || *(&byte_6010BF + 24) != 125 )
15   {
16 LABEL_22:
17     puts("Wrong flag!");
18     exit(-1);
19   }
```

下边全是goto语句 我们直接将视图切为图表（Graph view）

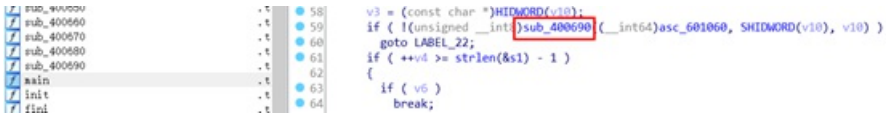




按r可以发现

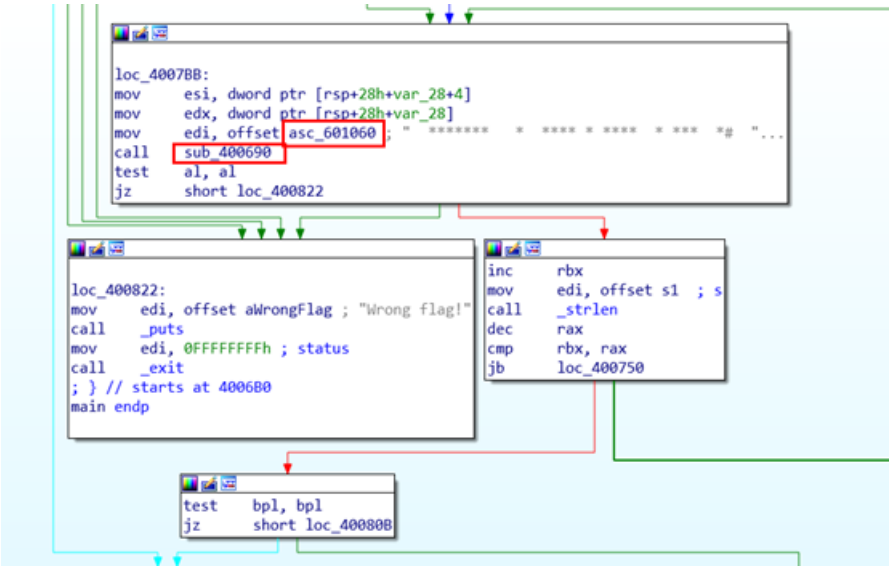


根据这四个字符"**.**" ,"**0**","**o**" , "**O**"分别跳到不同的位置进行操作



我们再看下400690在搞什么鬼

```
sub_400690 proc near
; __unwind {
movsxd  rax, esi
add     rax, rdi
movsxd  rcx, edx
movzx   eax, byte ptr [rax+rcx*8]
cmp     eax, 20h
setz    cl
cmp     eax, 23h
setz    al
or      al, cl
retn
; } // starts at 400690
sub_400690 endp
```



想到是和601060有关

猜测是判断上面601060数组的第edi个值是否等于20h或23h，如果不等于就跳到

400822输出"wrong flag"

那么我们现在要做的就是看下601060这个地址 直接切到HEX界面

```
0000000000601050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000601060  20 20 2A 2A 2A 2A 2A 2A  2A 20 20 20 2A 20 20 2A  ..*******...*..*
0000000000601070  2A 2A 2A 20 2A 20 2A 2A  2A 2A 2A 20 2A 20 2A 2A  ***.*.****..*.**
0000000000601080  2A 20 20 2A 23 20 20 2A  2A 2A 20 2A 2A 2A 20 2A  *..*#..***.***.*
0000000000601090  2A 2A 20 20 20 20 20 2A  2A 2A 2A 2A 2A 2A 2A 2A  **.....*********
00000000006010A0  00 ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ??  .???????????????
00000000006010B0  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ??  ???????????????
```

发现恰好为一个八阶方阵，而且数值只有3个，分别为20h，2Ah以及23h

那么函数400690的意思不就是判断当前的位置是否是从（0，0）走到（4，4），最初的四个跳转应该就是对应着上下左右四个方向

20 20 2A 2A 2A 2A 2A 2A

2A 20 20 20 2A 20 20 2A

2A 2A 2A 20 2A 20 2A 2A

2A 2A 20 20 2A 20 2A 2A

2A 20 20 2A 23 20 20 2A

2A 2A 20 2A 2A 2A 20 2A

2A 2A 20 20 20 20 20 2A

2A 2A 2A 2A 2A 2A 2A 2A

那么上下左右分别对应字符"." "**0**" "**O**" "**o**"

路径为：右下右右下下左下下下右右右右上上左左

o0oo00O000oooo..OO

还没完，记得要加上nctf{}

我们得到flag为nctf{o0oo00O000oooo..OO}