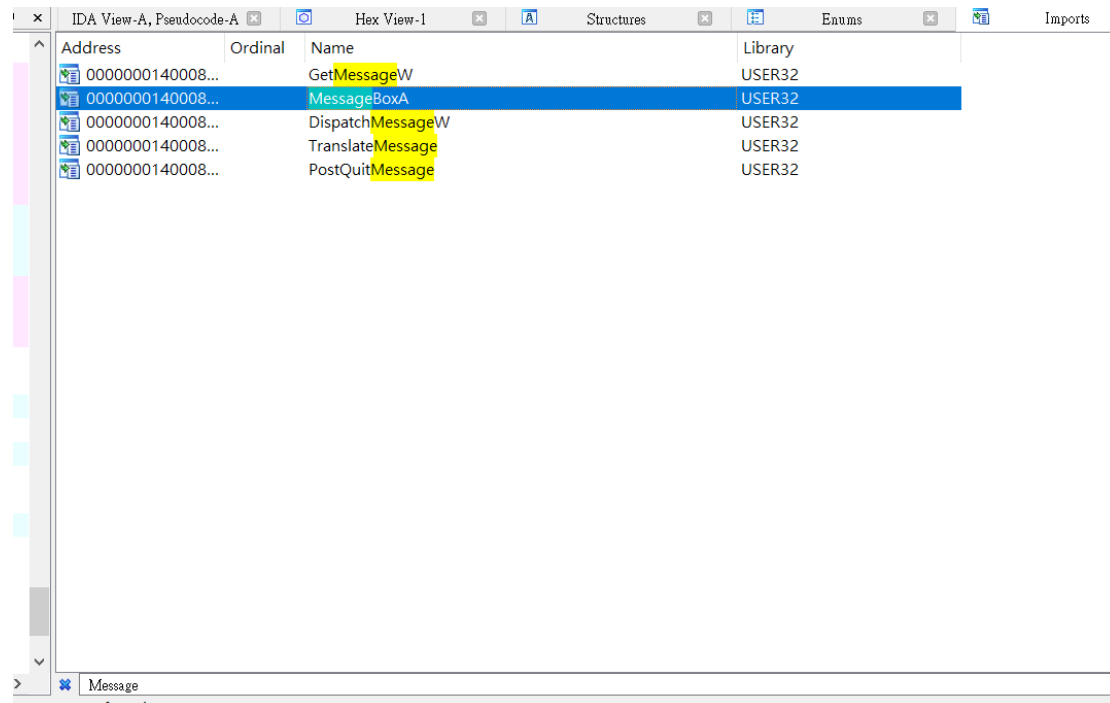


OOXX

這個程式發現是一個圈圈叉叉的遊戲，玩起來營不了，所以用 ida pro 開啟，發現每次遊戲結束後，都會有一個跳出的訊息視窗，查了後發現該功能很可能透過 Windows api 提供的 MessageBoxA

有關所以試試看在 import 的地方是否找得到這個 api，發現這支程式果然有用



之後再查詢其在 main 程式中使用到的地方，發現主要在一些 if else 中判斷，很有可能就是根據玩出不同結果，而給對應不同的訊息視窗邏輯。

```
if ( (unsigned int)sub_1400014D0() )
{
    Text[0] = -33;
    Text[1] = -89;
    Text[2] = -48;
    Text[3] = -18;
    Text[4] = -23;
    Text[5] = -90;
    Text[6] = -121;
    sub_140001480(Text, 7i64);
    MessageBoxA(0i64, Text, "Result", 0);
    result = 1i64;
}
else if ( sub_140001640() )
{
    v6[0] = -56;
    v6[1] = -89;
    v6[2] = -48;
    v6[3] = -18;
    v6[4] = -23;
    v6[5] = -90;
    v6[6] = -121;
    sub_140001480(v6, 7i64);
    MessageBoxA(0i64, v6, "Result", 0);
    sub_1400019C0(v7, 40i64);
    sub_1400031E0(v7, &unk_14000A040);
    v3[0] = -26;
    v3[1] = -27;
```

在仔細看懂這部分邏輯後發現，140001871 的這步是關鍵，因為我們不可能贏過，程式敵人所以上面圖片 sub_140001640()判斷贏的函式永遠都是 false，導致 1871 這步 jz 永遠都會跳到 loc_14000194D 的地方，

```
• .text:000000014000186F      test     eax, eax
• .text:0000000140001871      jz       loc_14000194D
• .text:0000000140001877      mov     [ebp+0B8h+var_4C], 0C8h
```

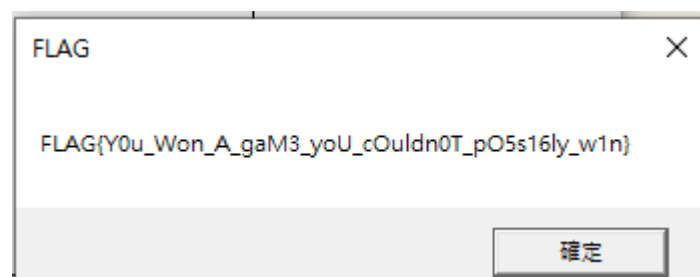
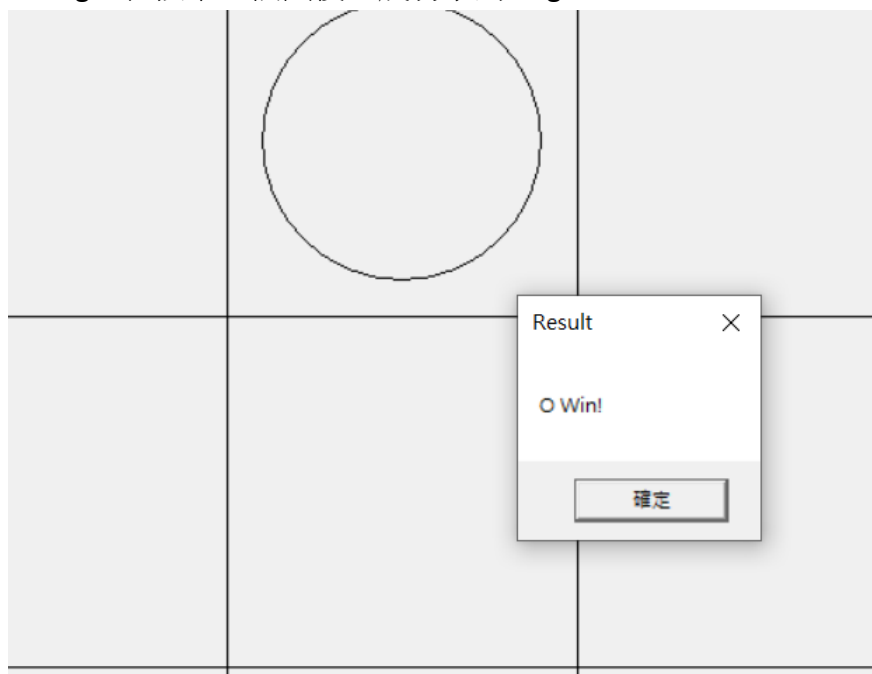
而在它下面接續的 CODE 其實很可能就是取得 FLAG 的 MessageBox。所以我使用 x64dbg 找到 1871 的位置

00007FF60522186F	85C0	test eax, eax
00007FF605221871	0F84 D6000000	je oox_f5c123f4e157e53d.7FF60522194D
00007FF605221877	C64424 6C C8	mov byte ptr ss:[rsp+6C], C8

將其 patch 成 nop，讓它不要跳過它之後的 code

00007FF60522186A	E8 D1FDFFFF	call oox_f5c123f4e157e53d.7FF605221640
00007FF60522186F	85C0	test eax, eax
00007FF605221871	90	nop
00007FF605221872	90	nop
00007FF605221873	90	nop
00007FF605221874	90	nop
00007FF605221875	90	nop
00007FF605221876	90	nop
00007FF605221877	C64424 6C C8	mov byte ptr ss:[rsp+6C], C8
00007FF60522187C	C64424 6D A7	mov byte ptr ss:[rsp+6D], A7

之後在 x64dbg，在按下一個圈後，成功拿到 flag。



SSTrojan

從 main funtion 裡面的 7E30 可以知道這個程式會跟 localhost:19832 建立連線。

```
1 int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
2 {
3     __int64 v4; // rax
4     char v6[40]; // [rsp+28h] [rbp-90h] BYREF
5     char v7[80]; // [rsp+50h] [rbp-68h] BYREF
6
7     sub_1400017E0(v7, 72i64);
8     v4 = sub_140001830(v6, "127.0.0.1");
9     sub_140007C90(v7, v4, 19832i64);
10    sub_140007D50(v7, sub_140001560);
11    sub_140007E30(v7);
12    while ( 1 )
13    {
14        sub_140002FA0(&unk_14000EB48);
15        Sleep(0xDBBA00u);
16    }
17 }
```

在 1560 裡面發現 unk_14000EB48 為一 PNG 且經過加密後傳到本機。

```
if ( recv(a1, buf, 512, 0) > 0 )
{
    v4 = sub_140001830(v5, "cDqr0hUUz1");
    v2 = sub_140001870(v4, buf);
    sub_140001800(v5);
    if ( !v2 )
    {
        v3 = (char *)sub_140003180(&unk_14000EB48, v6);
        if ( send(a1, v6, 4, 0) != -1 )
        {
            sub_1400014B0(v3, *(unsigned int *)v6);
            if ( send(a1, v3, *(int *)v6, 0) != -1 )
                j_j_free(v3);
        }
    }
}
```

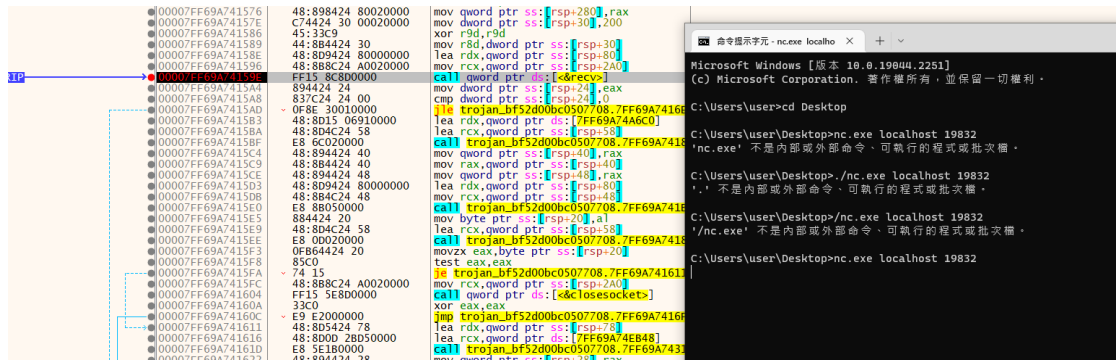
```
int64 __fastcall sub_1400014B0(int64 a1, unsigned int a2)
{
    int64 result; // rax
    int i; // [rsp+0h] [rbp-48h]
    char v4[24]; // [rsp+10h] [rbp-38h] BYREF

    strcpy(v4, "0vCh8RrvqkrbxN9Q7Ydx");
    for ( i = 0; ; ++i )
    {
        result = a2;
        if ( i >= (int)a2 )
            break;
        *(_BYTE *)(a1 + i) ^= v4[i % 0x15ui64];
    }
    return result;
}
```

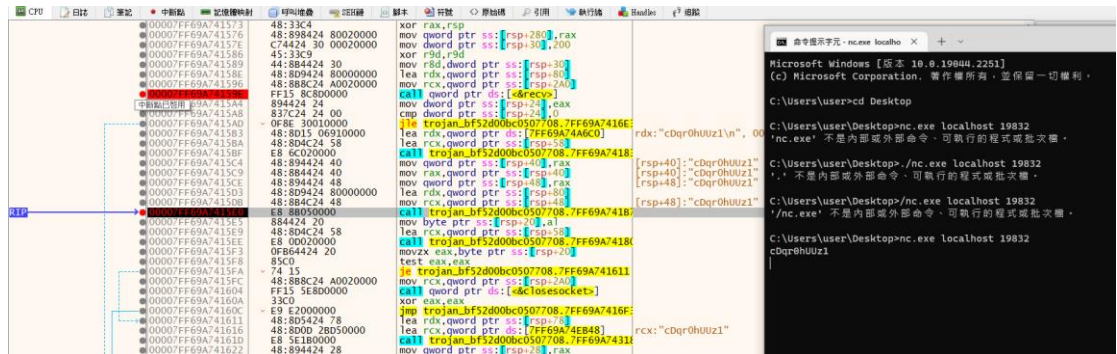
對圖片進行加密

再用 x64dbg 先連上並設斷點在

recv



傳入比較相符的值後



將je的判斷是改成jmp 這樣就不會被 closesocket 並且永遠執行下面 send 的指令，且可以看到都是加密過後的亂碼。



```

1  int64 __fastcall sub_140002FA0(int64 a1)
2  {
3      int64 result; // rax
4      int64 v2; // rax
5      int cy; // [rsp+50h] [rbp-78h]
6      int v4; // [rsp+54h] [rbp-74h]
7      HDC v5; // [rsp+60h] [rbp-68h]
8      HDC hdc; // [rsp+68h] [rbp-60h]
9      HBITMAP h; // [rsp+78h] [rbp-50h]
10     HGDIOBJ v8; // [rsp+80h] [rbp-48h]
11     char v9[24]; // [rsp+88h] [rbp-40h] BYREF
12     char v10[16]; // [rsp+A0h] [rbp-28h] BYREF
13
14     v4 = *(_DWORD*)(a1 + 40);
15     result = *(unsigned int*)(a1 + 44);
16     cy = *(_DWORD*)(a1 + 44);
17     if ( v4 > 0 && cy > 0 )
18     {
19         hdc = GetDC(0i64);
20         v5 = CreateCompatibleDC(hdc);
21         h = CreateCompatibleBitmap(hdc, v4, cy);
22         v8 = SelectObject(v5, h);
23         BitBlt(v5, 0, 0, v4, cy, hdc, 0, 0, 0xCC0020u);
24         sub_1400017E0(v9, 24i64);
25         sub_1400025A0(v9, h, 0i64);
26         sub_140002E70(L"image/png", v10);
27         v2 = sub_140004C00(a1 + 8);
28         sub_1400024E0(v9, v2, v10, 0i64);
29         SelectObject(v5, v8);
30         DeleteDC(v5);
31         ReleaseDC(0i64, hdc);
32         DeleteObject(h);
33         result = sub_1400026F0(v9);
34     }
35     return result;
36 }

```

在 main function 的 2FA0 裡在進行螢幕截圖的程式，所以就得知 PNG 檔為螢幕截圖。

在 24E0 中知道會存在我們的電腦裡

```

1  int64 __fastcall sub_1400024E0(int64 a1, int64 a2, int64 a3,
2  {
3      unsigned int v4; // eax
4
5      v4 = GdipSaveImageToFile(*(_QWORD*)(a1 + 8), a2, a3, a4);
6      return sub_140002370(a1, v4);
7  }

```

並用 x64dbg 到 GdipSaveImageToFile 設斷點可以知道他把圖片存在右邊的那個位址，然後點開後即為我們的螢幕截圖。

00007FF69A7424F4	48:83EC 28	sub rsp,28	
00007FF69A7424F8	4C:884C24 48	mov r9,qword ptr ss:[rsp+48]	
00007FF69A7424FD	4C:884424 40	mov r8,qword ptr ss:[rsp+40]	
00007FF69A742502	48:885424 38	mov rdx,qword ptr ss:[rsp+38]	
00007FF69A742507	48:884424 30	mov rax,qword ptr ss:[rsp+30]	
00007FF69A74250C	48:8848 08	mov rcx,qword ptr ds:[rax+8]	
00007FF69A742510	FF35 12800000	call qword ptr [edx]	[rsp+38]:L"C:\\Users\\user\\AppData\\Local\\Temp_____.png"
00007FF69A742516	8BD0	mov edx,edx	
00007FF69A742518	48:884C24 30	mov rcx,qword ptr ss:[rsp+30]	
00007FF69A74251D	E8 4EFFFFFF	call x64dbg.bt52d00bc0507708.7FF69A742370	
00007FF69A742522	48:83C4 28	add rsp,28	

最後我用題目給的 Pcap 檔裡傳輸的 Data 紀錄去作解密

8	127.0.0.1	127.0.0.1	TCP	65539
9	127.0.0.1	127.0.0.1	TCP	65539
10	127.0.0.1	127.0.0.1	TCP	23038

```

b"\x36\x9a\xd9\x65\xd8\xa6\x4e\xcd\xa3\x10\xf2\x9
\xe7\x6e\xc6\x96\xe6\x6b\x54\x02\xc9\x86\x1e\x9
b"\xce\xbd\x66\x58\x47\x3d\x30\x76\x43\x68\x71\x
b"\x12\xe0"

v4="0vCh8RrvqkrbxN9Q7Ydx"+"0"
newpng=b""
✓ for i in range(0x25980):
    tmp =png[i]^ord(v4[i % 21])
    newpng+=tmp.to_bytes(1, 'big')

✓ with open("flag.png", "wb") as binary_file:
    binary_file.write(newpng)

```

得到 flag



(好想要 apex cheater.....)