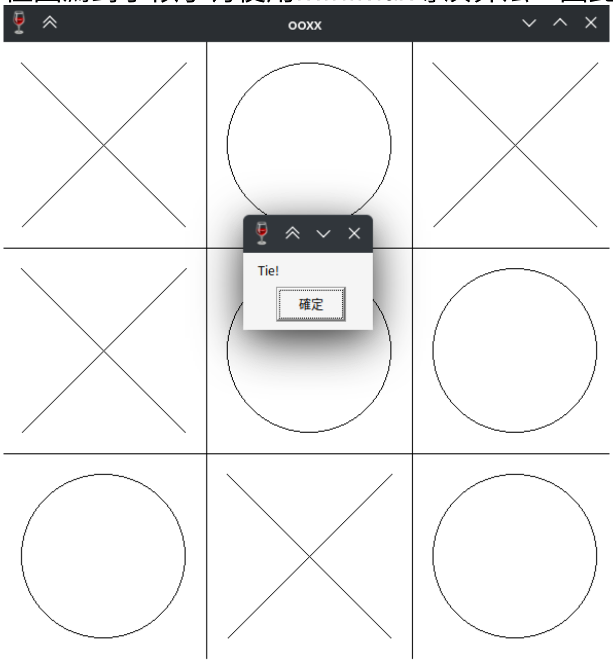
ooxx

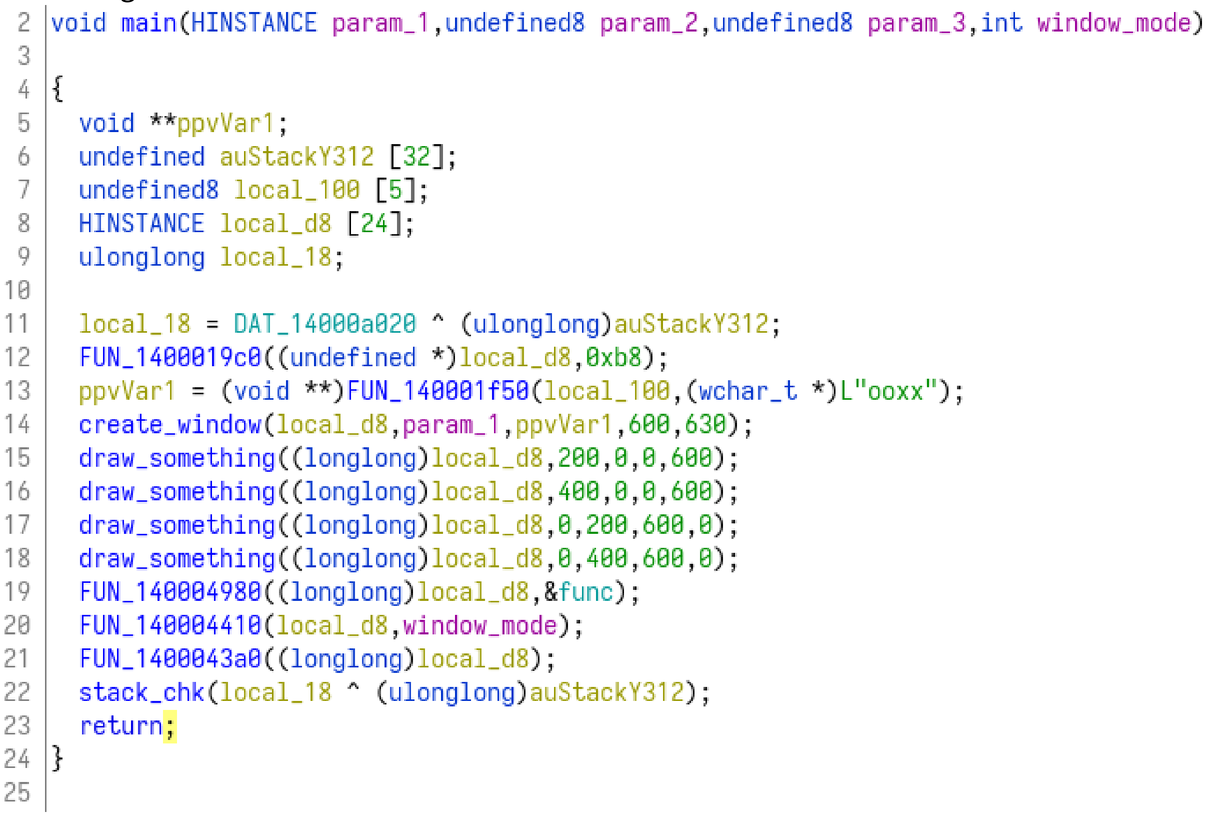
這個題目點開會是一個ooxx的遊戲，看起來只要獲勝就可以得到flag，
但因爲對手似乎有使用minimax等演算法，因此不可能獲勝，頂多平手。



總共觀察出以下幾點:
- 玩家爲O - 玩家先行 - 無法獲勝 - 程式視窗的標題是ooxx - 每回合判斷勝負

reverse

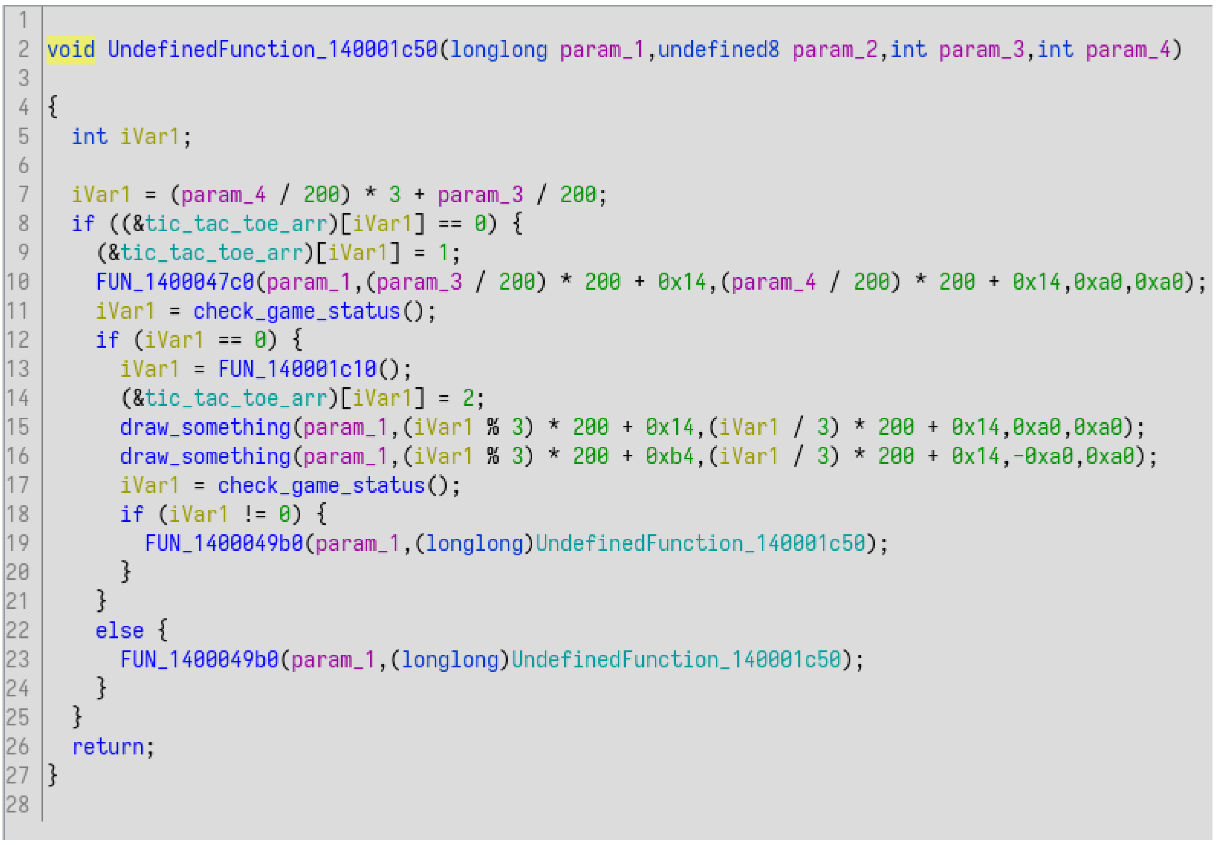
這次使用ghidra，首先看到反編譯出的main function:



```
void main(HINSTANCE param_1,undefined8 param_2,undefined8 param_3,int window_mode)

{
  void **ppvVar1;
  undefined auStackY312 [32];
  undefined8 local_100 [5];
  HINSTANCE local_d8 [24];
  ulonglong local_18;

  local_18 = DAT_14000a020 ^ (ulonglong)auStackY312;
  FUN_1400019c0((undefined *)local_d8,0xb8);
  ppvVar1 = (void **)FUN_140001f50(local_100,(wchar_t *)L"ooxx");
  create_window(local_d8,param_1,ppvVar1,600,630);
  draw_something((longlong)local_d8,200,0,0,600);
  draw_something((longlong)local_d8,400,0,0,600);
  draw_something((longlong)local_d8,0,200,600,0);
  draw_something((longlong)local_d8,0,400,600,0);
  FUN_140004980((longlong)local_d8,&func);
  FUN_140004410(local_d8,window_mode);
  FUN_1400043a0((longlong)local_d8);
  stack_chk(local_18 ^ (ulonglong)auStackY312);
  return;
}

```

根據程式執行的行爲，對一些function進行了分析與重新命名

其中，找到了一個似乎是程式邏輯的function:



```
void UndefinedFunction_140001c50(longlong param_1,undefined8 param_2,int param_3,int param_4)

{
  int iVar1;

  iVar1 = (param_4 / 200) * 3 + param_3 / 200;
  if ((&tic_tac_toe_arr)[iVar1] == 0) {
    (&tic_tac_toe_arr)[iVar1] = 1;
    FUN_1400047c0(param_1,(param_3 / 200) * 200 + 0x14,(param_4 / 200) * 200 + 0x14,0xa0,0xa0);
    iVar1 = check_game_status();
    if (iVar1 == 0) {
      iVar1 = FUN_140001c10();
      (&tic_tac_toe_arr)[iVar1] = 2;
      draw_something(param_1,(iVar1 % 3) * 200 + 0x14,(iVar1 / 3) * 200 + 0x14,0xa0,0xa0);
      draw_something(param_1,(iVar1 % 3) * 200 + 0xb4,(iVar1 / 3) * 200 + 0x14,-0xa0,0xa0);
      iVar1 = check_game_status();
      if (iVar1 != 0) {
        FUN_1400049b0(param_1,(longlong)UndefinedFunction_140001c50);
      }
    }
    else {
      FUN_1400049b0(param_1,(longlong)UndefinedFunction_140001c50);
    }
  }
  return;
}
```

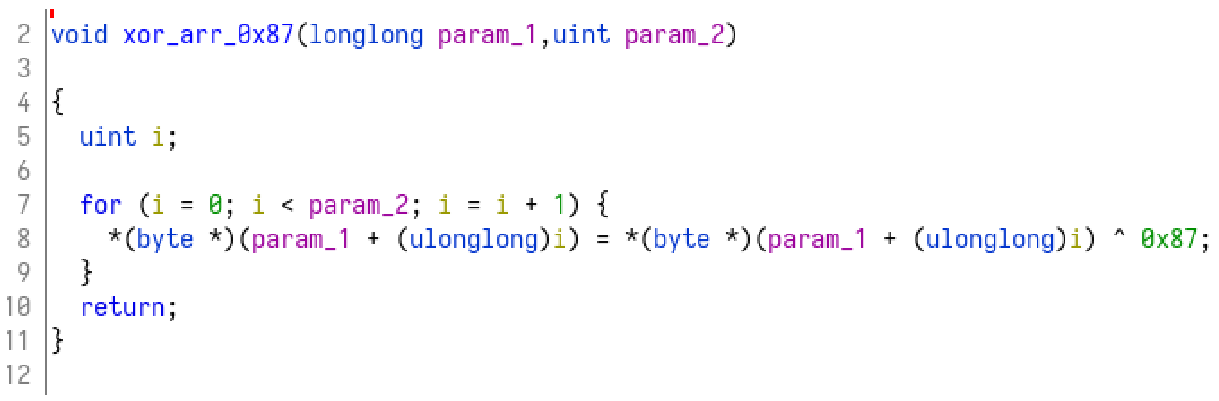
深入檢查，可以找到一個function，就是在做獲勝與否，遊戲結束與否的判斷。

function check status

反編譯這個重要的function後，可以獲得下面的pseudo code:

```
void check_game_status(void)

{
  undefined8 result;
  undefined auStack184 [32];
  undefined4 local_98;
  undefined8 *local_90;
  void **local_88;
  undefined8 local_80 [4];
  undefined local_60;
  undefined local_5f;
  undefined local_5e;
  CHAR local_5c;
  undefined local_5b;
  undefined local_5a;
  undefined local_59;
  undefined local_58;
  CHAR local_54;
  undefined local_53;
  undefined local_52;
  undefined local_51;
  undefined local_50;
  undefined local_4f;
  undefined local_4e;
  CHAR local_4c;
  undefined local_4b;
  undefined local_4a;
  undefined local_49;
  undefined local_48;
  undefined local_47;
  undefined local_46;
  longlong local_40 [5];
  ulonglong local_18;

  local_18 = DAT_14000a020 ^ (ulonglong)auStack184;
  result = FUN_1400014d0();
  if ((int)result == 0) {
    result = FUN_140001640();
    if ((int)result == 0) {
      result = is_all_place_filled();
      if ((int)result != 0) {
                    // "Tie!"
        local_5c = -0x2d;
        local_5b = 0xee;
        local_5a = 0xe2;
        local_59 = 0xa6;
        local_58 = 0x87;
        xor_arr_0x87((longlong)&local_5c,5);
        MessageBoxA((HWND)0x0,&local_5c,"Result",0);
      }
    }
    else {
                    // "O win!"
      local_4c = -0x38;
      local_4b = 0xa7;
      local_4a = 0xd0;
      local_49 = 0xee;
      local_48 = 0xe9;
      local_47 = 0xa6;
      local_46 = 0x87;
      xor_arr_0x87((longlong)&local_4c,7);
      MessageBoxA((HWND)0x0,&local_4c,"Result",0);
      FUN_1400019c0((undefined *)local_40,0x28);
      FUN_1400031e0(local_40,(longlong)&DAT_14000a040);
      local_60 = 0xe6;
      local_5f = 0xe5;
      local_5e = 0x87;
      xor_arr_0x87((longlong)&local_60,3);
      local_90 = local_80;
      local_88 = (void **)FUN_140001fd0(local_90,&local_60);
      FUN_140002480(local_40,local_88,MessageBoxA_exref);
      local_98 = 1;
      nop();
    }
  }
  else {
                    // "X win!"
    local_54 = -0x21;
    local_53 = 0xa7;
    local_52 = 0xd0;
    local_51 = 0xee;
    local_50 = 0xe9;
    local_4f = 0xa6;
    local_4e = 0x87;
    xor_arr_0x87((longlong)&local_54,7);
    MessageBoxA((HWND)0x0,&local_54,"Result",0);
  }
  stack_chk(local_18 ^ (ulonglong)auStack184);
  return;
}
```

其中可以看出來應該有一些變數其實是array，但他沒有辨識出來。
並且會將這些array賦予值之後進入一個function，點進去看這個function做的事情:



```
void xor_arr_0x87(longlong param_1,uint param_2)

{
  uint i;

  for (i = 0; i < param_2; i = i + 1) {
    *(byte *)(param_1 + (ulonglong)i) = *(byte *)(param_1 + (ulonglong)i) ^ 0x87;
  }
  return;
}
```

可以看出來這應該是會將傳入的array每個元素都做xor 0x87，那就可以根據這個寫出還原的程式碼:

```
s = [0xd3, 0xee, 0xe2, 0xa6, 0x87]
s = [chr(i ^ 0x87) for i in s]
print("".join(s))

s = [0xc8, 0xa7, 0xd0, 0xee, 0xe9, 0xa6, 0x87]
s = [chr(i ^ 0x87) for i in s]
print("".join(s))

s = [0xdf, 0xa7, 0xd0, 0xee, 0xe9, 0xa6, 0x87]
s = [chr(i ^ 0x87) for i in s]
print("".join(s))
```

執行結果:

```
Tie!
O Win!
X Win!
```

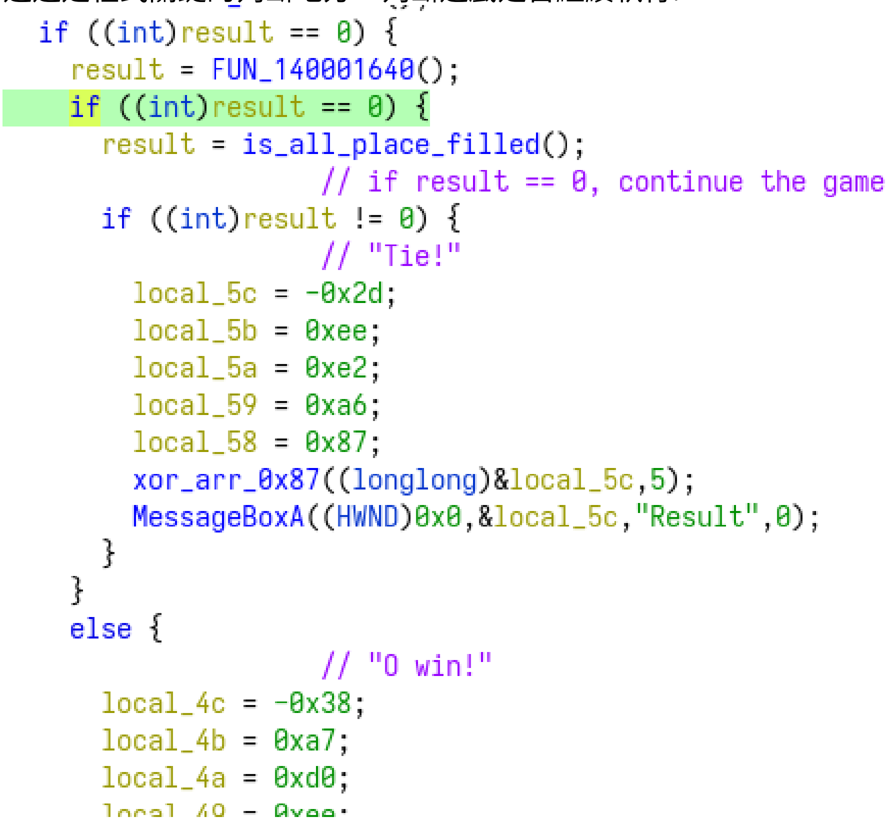
至此可以完全確定這邊就是判斷勝負，以及是否繼續進行遊戲的地方。

cheat

既然已經知道關鍵位置，那只需要繼續觀察 O Win! 的部分，應該就可以成功的解出flag
但稍微看了一下後面的程式碼，似乎又呼叫了許多function，有點不容易trace，因此決定使用作弊的方式。

patch instruction

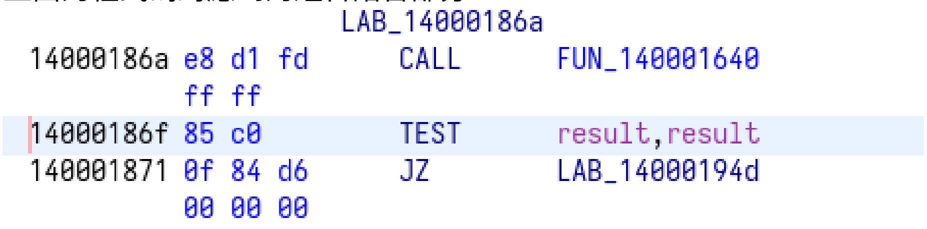
這邊是程式關鍵的判斷地方，判斷遊戲是否繼續執行:



```
  if ((int)result == 0) {
    result = FUN_140001640();
    if ((int)result == 0) {
      result = is_all_place_filled();
                    // if result == 0, continue the game
      if ((int)result != 0) {
                    // "Tie!"
        local_5c = -0x2d;
        local_5b = 0xee;
        local_5a = 0xe2;
        local_59 = 0xa6;
        local_58 = 0x87;
        xor_arr_0x87((longlong)&local_5c,5);
        MessageBoxA((HWND)0x0,&local_5c,"Result",0);
      }
    }
    else {
                    // "O win!"
      local_4c = -0x38;
      local_4b = 0xa7;
      local_4a = 0xd0;
      local_49 = 0xee;
```

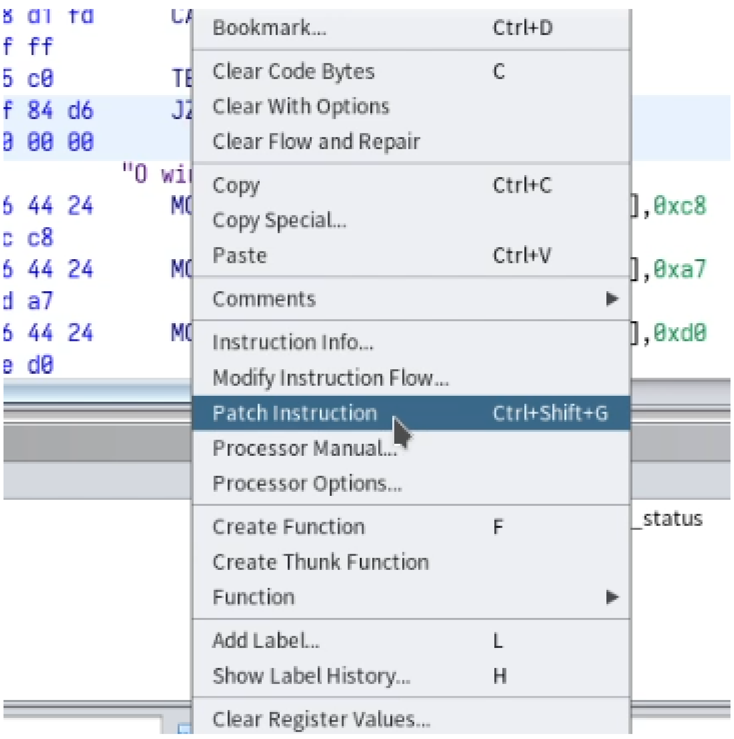
理論上只要能夠更改上圖中綠色的判斷條件，應該可以直接獲勝，免去後面的步驟。

上面的程式碼對應到的組合語言部分:

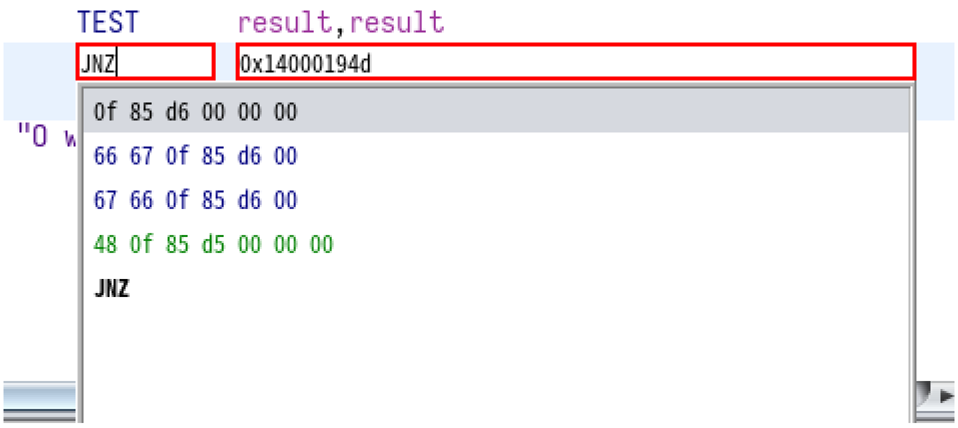


```
                      LAB_14000186a
 14000186a e8 d1 fd        CALL       FUN_140001640
           ff ff
 14000186f 85 c0           TEST       result,result
 140001871 0f 84 d6        JZ         LAB_14000194d
           00 00 00
```

按下右鍵，選 patch instruction:



並修改跳躍條件:

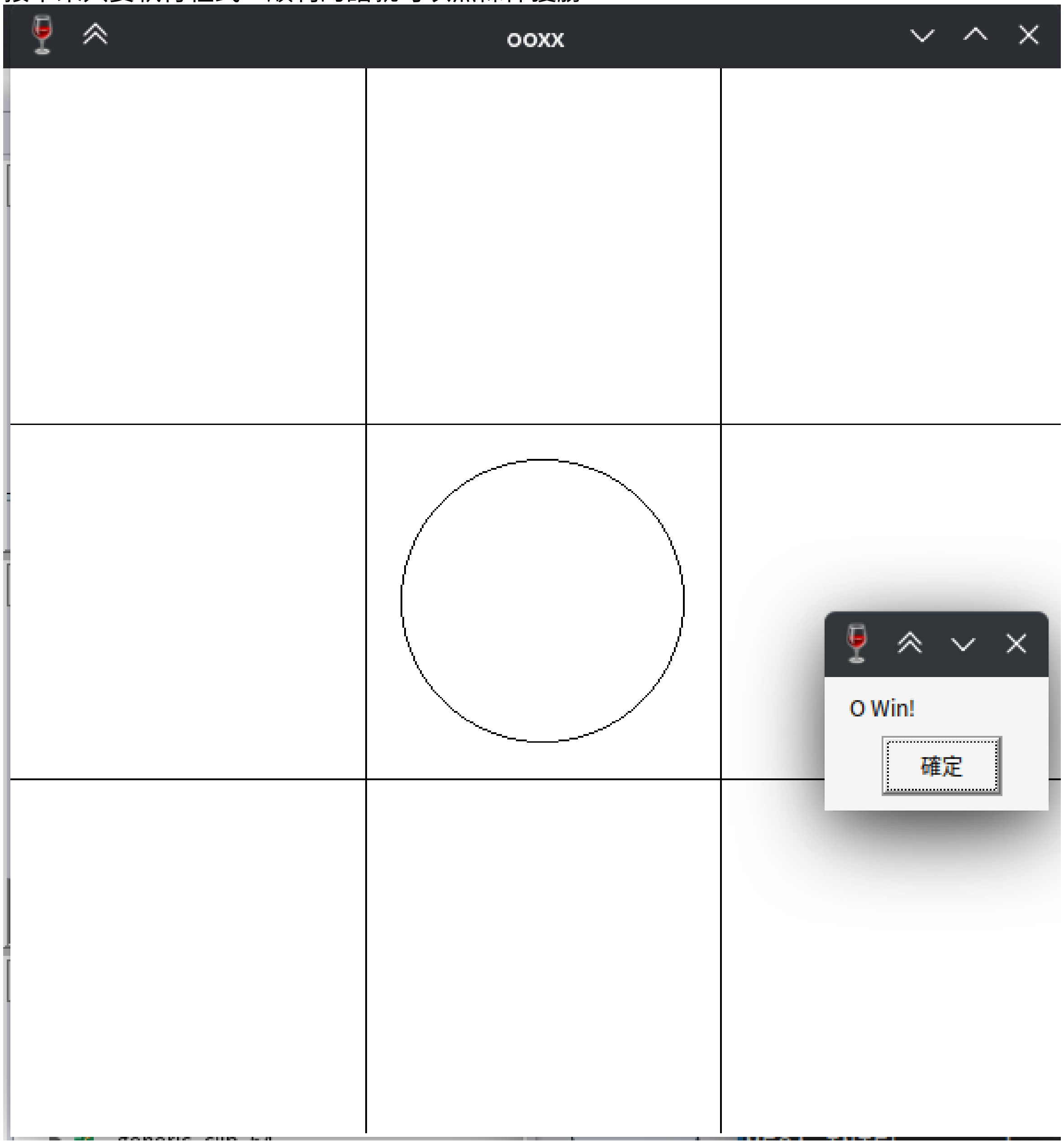


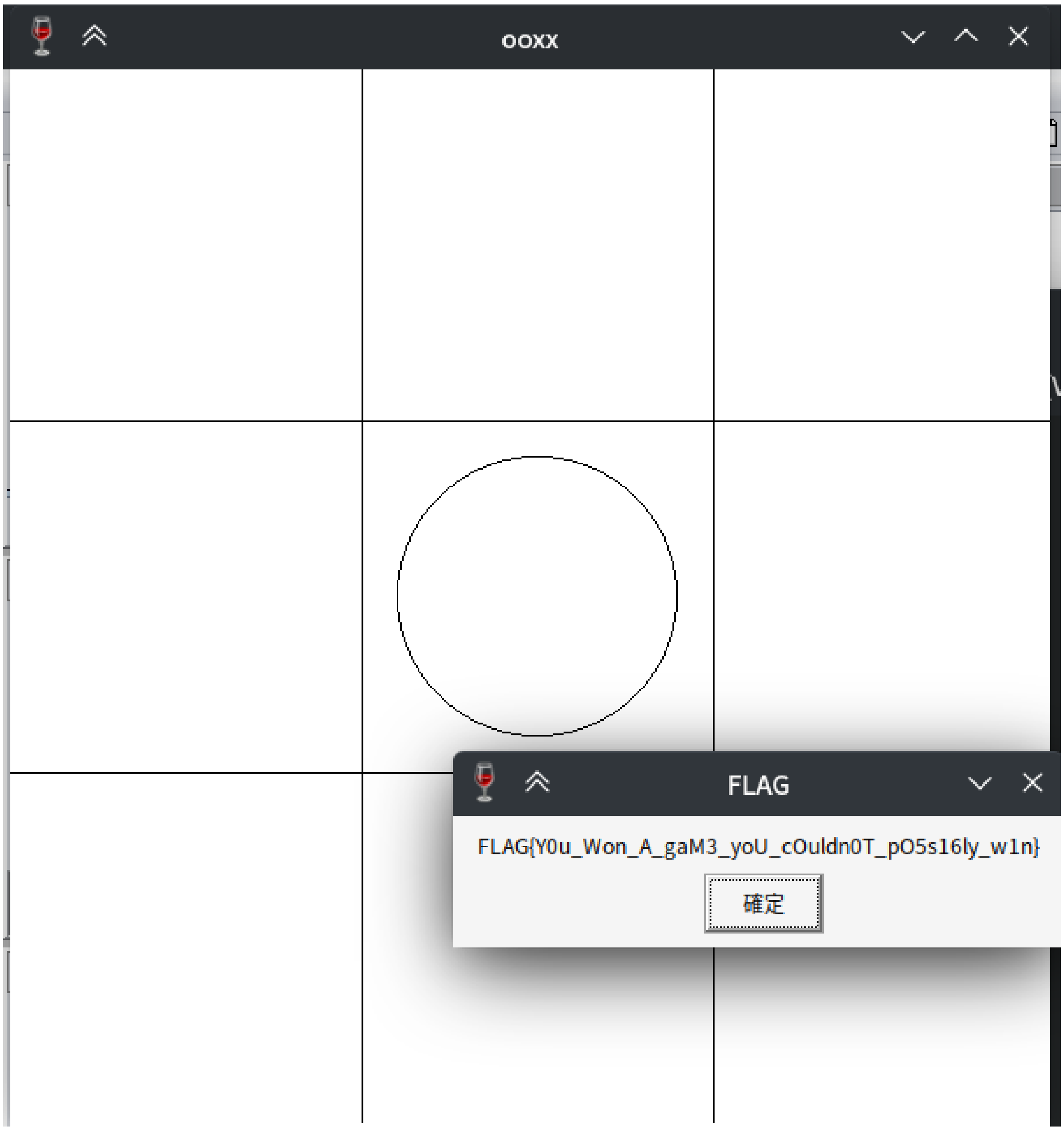
```
TEST        result,result
JNZ         0x14000194d
0f 85 d6 00 00 00
66 67 0f 85 d6 00
67 66 0f 85 d6 00
48 0f 85 d5 00 00 00
JNZ
```

修改完成後，將程式以PE格式匯出，就得到了一個patch過的程式。

solve

接下來只要執行程式，順利的話就可以無條件獲勝。





```
FLAG{Y0u_Won_A_gaM3_yoU_cOuldn0T_pO5s16ly_w1n}
```

成功解開。