## 附加题

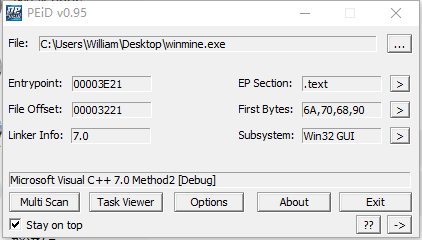
一、环境

VS2017 MFC

Windows 10

二、.分析过程

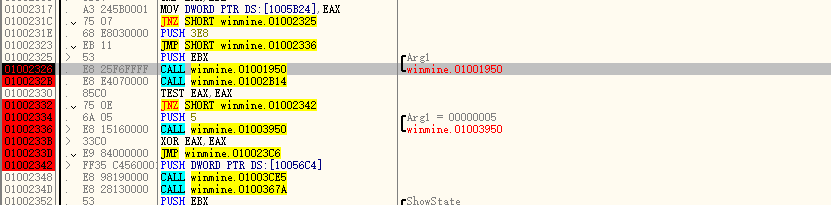
1. 加壳扫描



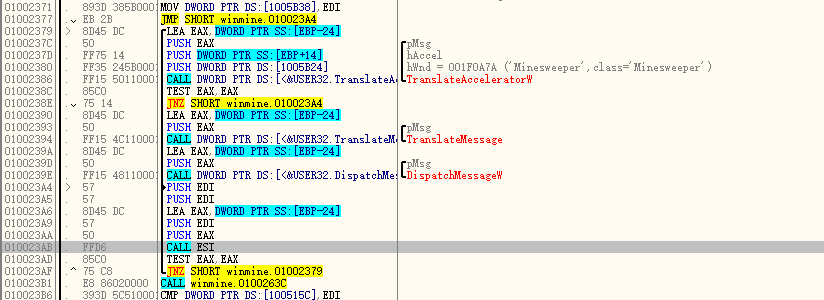
并没有加壳，万岁。

2.、用ollydbg进行分析

经过一系列艰难的调试，找到了它生成雷区的部分

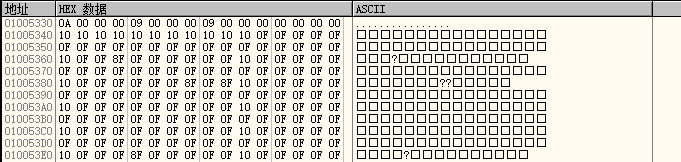


3、然后继续执行，发现他在此处循环等待鼠标点击



此时界面已经完全显示出来了。

4. 在底下的内存信息中也已经能够找到雷区的信息。于是

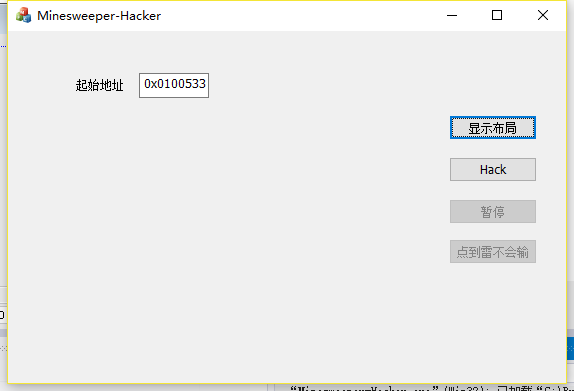


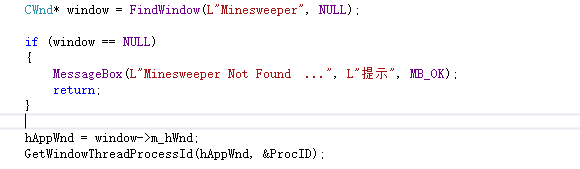
0x01005330 处的数据0x0A 表示有10个雷，宽和高分别为9。

0x01005340 处所示的0x10 表示边界。有雷是0x8F;没有雷是0x0F。

5. 着手编写代码

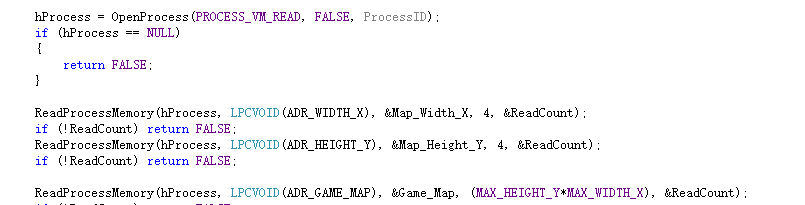
①因为在内存中可以读取到 扫雷游戏的布局，因此我们需要调用Windows的接口去读取这块内存。





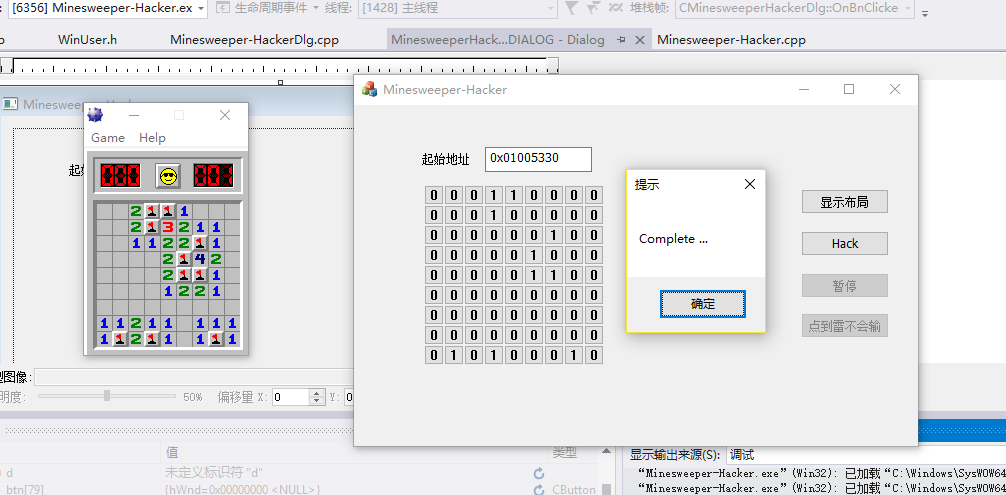
② 使用FindWindow(L"Minesweeper", NULL);找到扫雷游戏的窗口句柄

③ GetWindowThreadProcessId(hAppWnd, &ProcID);来获取进程的ID。



④紧接着利用ReadProcessMemory读取相应的内存部分。

⑤经过一些额外的处理，点击界面上的【显示布局】后即可即可出现相应的雷区，数字1表示雷，数字0表示正常区域。



⑥因为我们已经可以确定雷区的位置了，因此我们可以模拟鼠标去点击那些没有雷的地方，最方便的方法是调用PostMessage(hMineAppWnd, WM\_LBUTTONDOWN, 0x1, MAKELPARAM(pt.x, pt.y));发送鼠标点击事件。于是如上图所示，我们实现了自动完成游戏的功能。

⑦尝试暂停时间

方案一：