# Frida快速入门

## 课前准备

1. 学员基础：
   1. 了解汇编语言
   2. 了解Python
   3. 熟悉TypeScript
   4. 了解JavaScript
   5. C/C++(高级部分)
2. 演示环境：
   1. VMware Workstation Pro
   2. Windows 10
   3. PyCharm
   4. VSCode
   5. Visual Studio 2022
   6. OD
   7. CE
   8. IDA
3. 演示目标：winmine.exe (windows xp系统附带的扫雷游戏)
   1. 修改时间：2001‎年‎8‎月‎31‎日，‏‎16:04:32
   2. **软件版本：5.1.2600.0**
   3. **软件大小：117 KB (119,808 字节)**
4. 课程源码：https://github.com/zmrbak/Frida
5. 参考资料：https://frida.re/
6. 课程类别：演示、实操教学

## Frida安装

https://frida.re/

<https://www.python.org/>

<https://www.jetbrains.com/>

<https://pc.weixin.qq.com/>

## Frida中的Python脚本与JavaScript脚本

## Frida中的TypeScript脚本与JavaScript脚本（1）

<https://frida.re/>

<https://nodejs.org/en/>

<https://github.com/oleavr/frida-agent-example>

## Frida中的TypeScript脚本与JavaScript脚本（2）

## 棋盘数据定位与读取

CE

C:\Program Files\Cheat Engine 7.4\languages

<https://cheatengine.org/downloads.php>

宽度

winmine.exe+5334 - 16 - push ss

高度

winmine.exe+5338 - 15 00000000 - adc eax,00000000 { 0 }

雷数

winmine.exe+5330 - 2A 00 - sub al,[eax]



棋盘头

winmine.exe+5340 - 10 10 - adc [eax],dl

<?xml version="1.0" encoding="utf-8"?>

<CheatTable>

<CheatEntries>

<CheatEntry>

<ID>4</ID>

<Description>"高度"</Description>

<LastState Value="10" RealAddress="01005338"/>

<ShowAsSigned>0</ShowAsSigned>

<VariableType>4 Bytes</VariableType>

<Address>winmine.exe+5338</Address>

</CheatEntry>

<CheatEntry>

<ID>3</ID>

<Description>"宽度"</Description>

<LastState Value="20" RealAddress="01005334"/>

<ShowAsSigned>0</ShowAsSigned>

<VariableType>4 Bytes</VariableType>

<Address>winmine.exe+5334</Address>

</CheatEntry>

<CheatEntry>

<ID>5</ID>

<Description>"雷数"</Description>

<LastState Value="30" RealAddress="01005330"/>

<ShowAsSigned>0</ShowAsSigned>

<VariableType>4 Bytes</VariableType>

<Address>winmine.exe+5330</Address>

</CheatEntry>

<CheatEntry>

<ID>2</ID>

<Description>"棋盘头"</Description>

<LastState Value="10101010" RealAddress="01005340"/>

<ShowAsHex>1</ShowAsHex>

<ShowAsSigned>0</ShowAsSigned>

<VariableType>4 Bytes</VariableType>

<Address>winmine.exe+5340</Address>

</CheatEntry>

</CheatEntries>

</CheatTable>

## Frida读取棋盘数据

## 分析和修改棋盘数据

====================== 2022-06-12T04:23:14.717Z ==========================

Frida.version 15.1.24

棋盘高度: 9

棋盘宽度: 9

地雷数量: 10

棋盘头: 0x1005340

10 10 10 10 10 10 10 10 10 10 10

10 0F 0F 0F 0F 8F 0F 0F 0F 0F 10

10 0F 0F 0F 8F 0F 0F 0F 0F 0F 10

10 0F 0F 0F 0F 0F 0F 8F 0F 8F 10

10 8F 0F 0F 0F 0F 0F 0F 0F 0F 10

10 0F 0F 0F 8F 8F 0F 0F 0F 0F 10

10 0F 0F 0F 0F 0F 8F 0F 0F 8F 10

10 0F 0F 0F 0F 0F 0F 0F 0F 0F 10

10 0F 0F 0F 0F 0F 8F 0F 0F 0F 10

10 0F 0F 0F 0F 0F 0F 0F 0F 0F 10

10 10 10 10 10 10 10 10 10 10 10

10：棋盘边框

**游戏开始之前：**

**8F：地雷**

**0F：不是雷（空的）**

**游戏开始之后：**

**8E：做了标记（对的）**

**0E：标错的地方（错的）**

**游戏结束**

**0F：不是雷（空的）**

**CC：爆掉的地雷**

**8A：未爆的地雷（游戏结束）**

**8E：做了标记**

**41:1**

**42:2**

**。。。**

**48:8**

## 让系统重新绘制指定窗口

**SpyXX.exe**

**C:\Program Files\Microsoft Visual Studio\2022\Preview\Common7\Tools**

1. **获取窗口句柄**

**FindWindow(不好用，最顶层符合条件的窗口)**

<?xml version="1.0" encoding="utf-8"?>

<CheatTable>

<CheatEntries>

<CheatEntry>

<ID>0</ID>

<Description>"窗口句柄"</Description>

<LastState Value="001904C0" RealAddress="01005B24"/>

<ShowAsHex>1</ShowAsHex>

<VariableType>4 Bytes</VariableType>

**<Address>winmine.exe+5B24</Address>**

</CheatEntry>

</CheatEntries>

</CheatTable>

1. 获取窗口客户区域

GetClientRect

3、将这一部分区域重新绘制

InvalidateRect

private GetClientRect!: NativePointer | null;

private InvalidateRect!: NativePointer | null;

this.GetClientRect = Module.findExportByName("User32.dll", "GetClientRect");

this.InvalidateRect = Module.findExportByName("User32.dll", "InvalidateRect");

// BOOL GetClientRect(

// [in] HWND hWnd,

// [out] LPRECT lpRect

// );

const lpRect = Memory.alloc(4 \* 4);

let GetClientRect = new NativeFunction(this.GetClientRect!, "bool", ["pointer", "pointer"]);

let result = GetClientRect(this.hWnd, lpRect);

console.log("result", result);

// BOOL InvalidateRect(

// [in] HWND hWnd,

// [in] const RECT \* lpRect,

// [in] BOOL bErase

// );

let InvalidateRect = new NativeFunction(this.InvalidateRect!, "bool", ["pointer", "pointer", 'bool']);

result = InvalidateRect(this.hWnd, lpRect, 1);

console.log("result", result);

## Frida中TypeScript导入模块的问题

$ git clone git://github.com/oleavr/frida-agent-example.git

$ cd frida-agent-example/

$ npm install

$ frida -U -f com.example.android --no-pause -l \_agent.js

$ npm run watch

import { WinApi } from "./win\_api";

export class WinApi {}

## TypeScript模块的封装