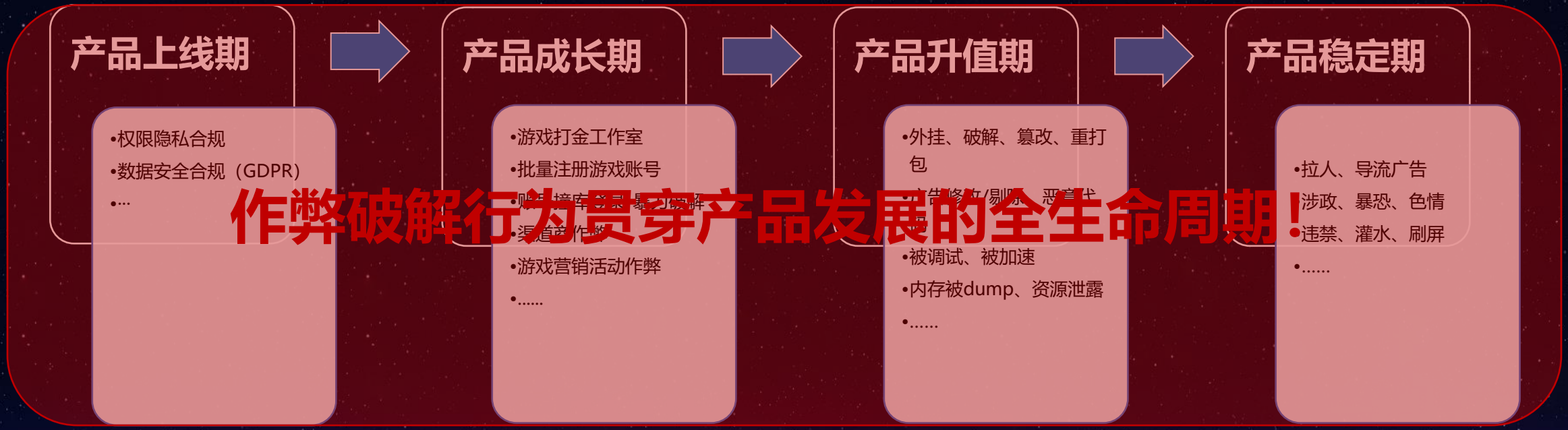
网易易盾

# 基于Unity引擎iOS游戏的安全风控实践

网易易盾　　王桂林

目录

# 游戏安全风险与痛点分析

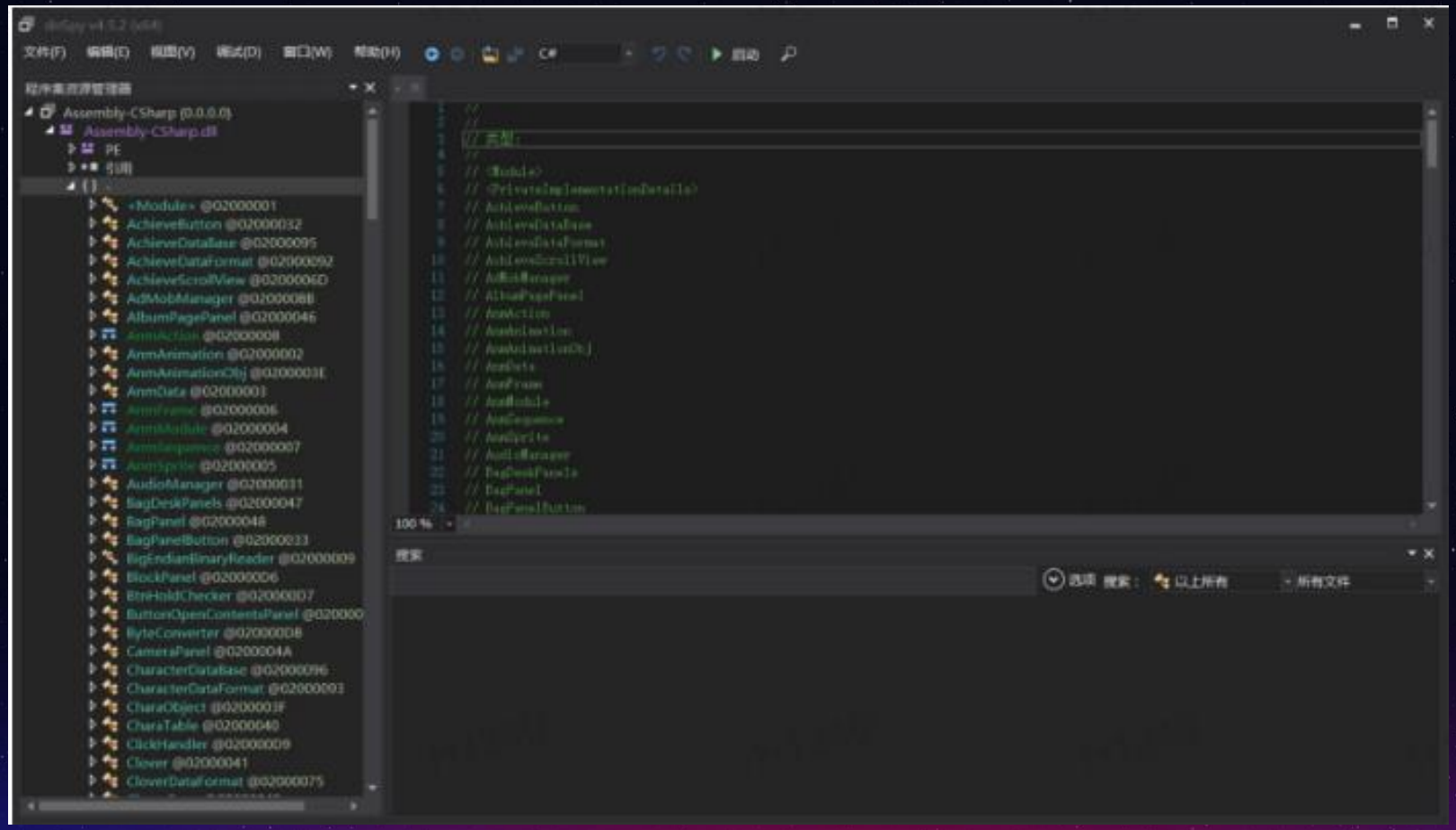



**作弊破解行为贯穿产品发展的全生命周期！**

# 某爆款悠闲游戏破解

Android版本的该休闲游戏是u3d，C#脚本没有加密，而且是mono模式

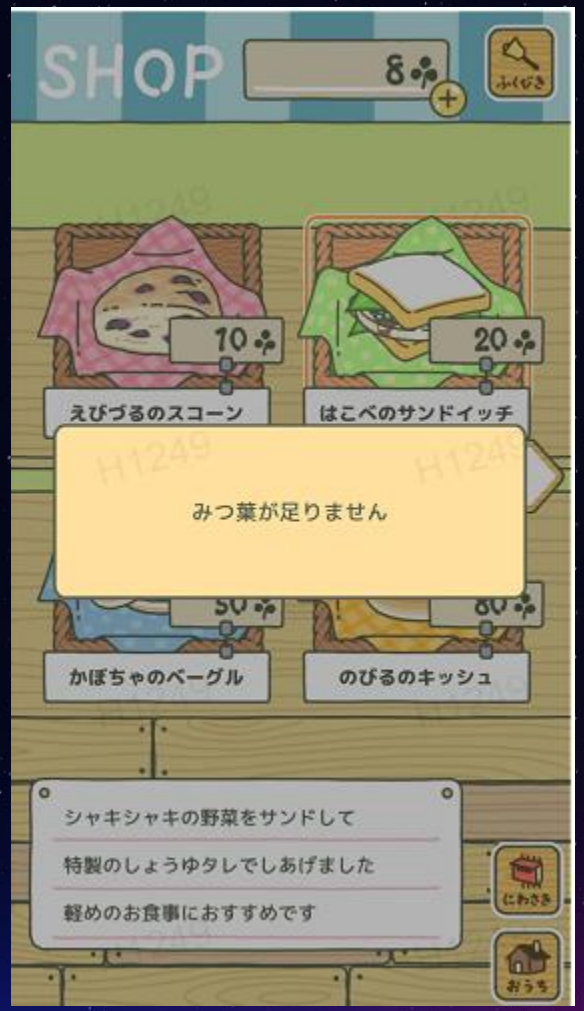
首先将Android apk安装包解压，查看assets/bin/Data/Managed文件夹下面的dll文件，其中Assembly-Csharp.dll便是游戏中的C#脚本，使用Reflector或者dnSpy进行反编译都可以



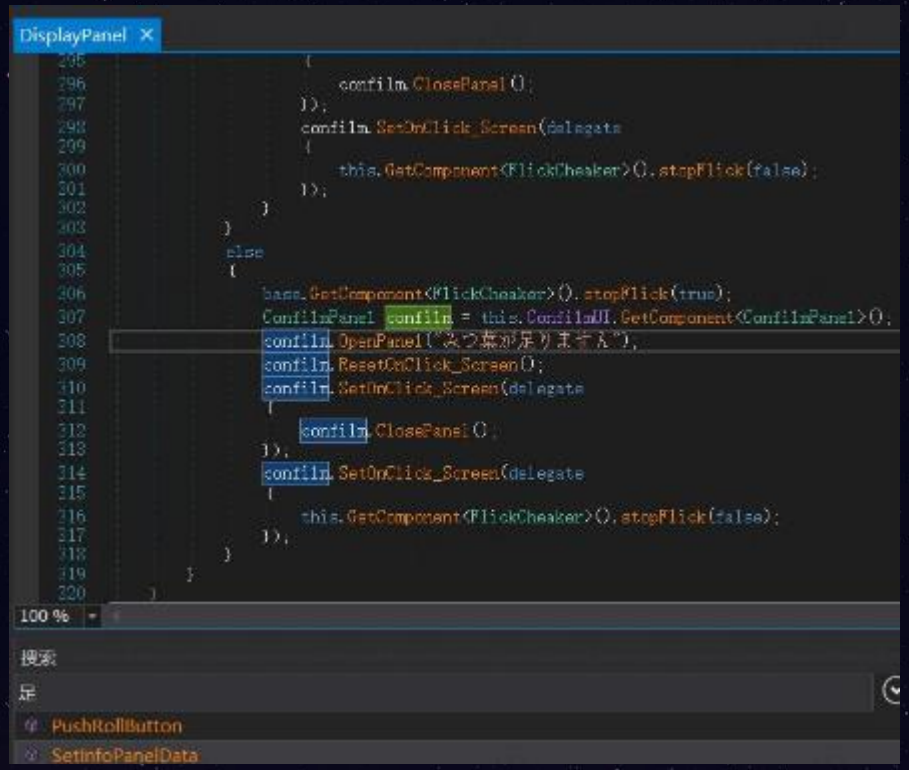
# 某爆款悠闲游戏破解

## ➢ 修改三叶草数目

1. 要修改三叶草数可以从购买的时候入手，比如提示三叶草不足的时候




2. 直接搜索，找到目标代码




3. 当进行购买时，都会通过 SuperGameMaster.CloverPointStock()函数，saveData.CloverPoint的值来判断三叶草的数目

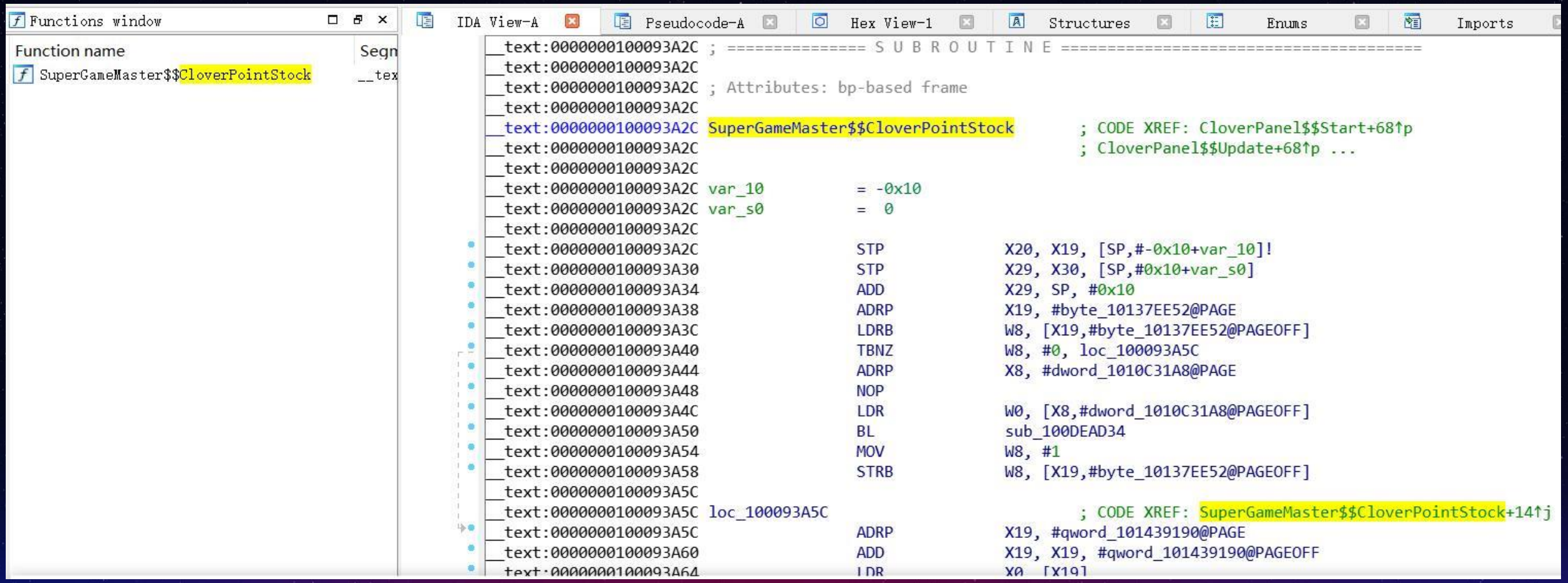
```
public static int CloverPointStock()
{
    return SuperGameMaster.saveData.CloverPoint;
}
```

```
public static int CloverPointStock()
{
    return 9999;
}
```

# 某爆款悠闲游戏破解

如果Android的脚本进行了加密，或者也使用IL2CPP模式，那又该怎么办呢?

iOS使用IL2CPP模式，C#脚本转成C/C++代码，使用Il2CppDumper还原符号
然后经过调试分析，找到每个功能对应的函数，进行HOOK



# 某爆款悠闲游戏破解

- HOOK函数

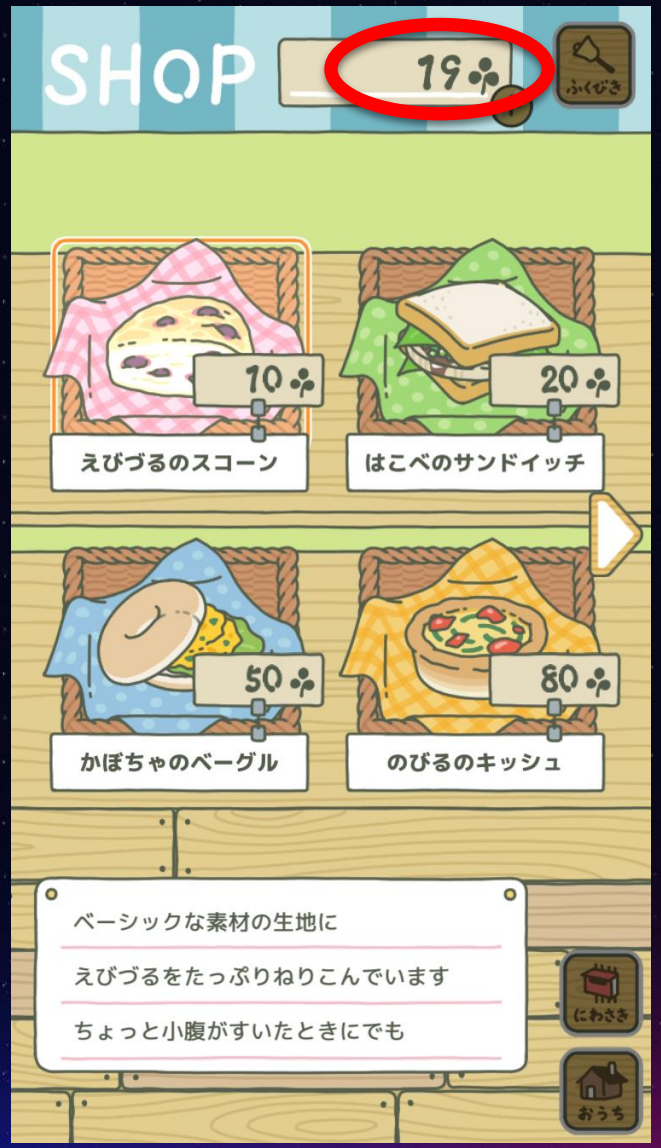
```
int (*old_clover_point_stock)(void);

int new_clover_point_stock(void)
{
	return 9999;
}

%ctor
{
	@autoreleasepool{
		unsigned long clover_point_stock = _dyld_get_image_vmaddr_slide(0) + 0x100093A2C;
		MSHookFunction((void *)clover_point_stock, (void *)&new_clover_point_stock, (void
**)&old_clover_point_stock);
	}
}
```

# 某爆款悠闲游戏破解

- 破解效果

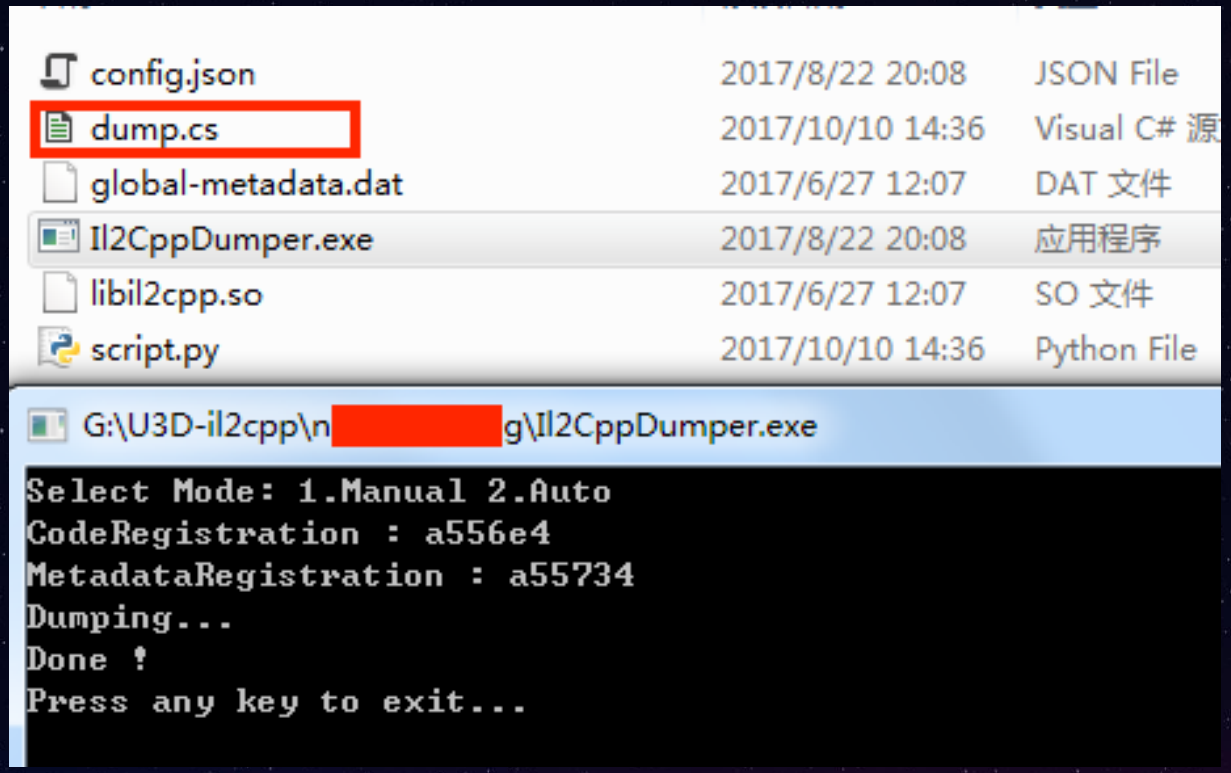



修改三叶草数目

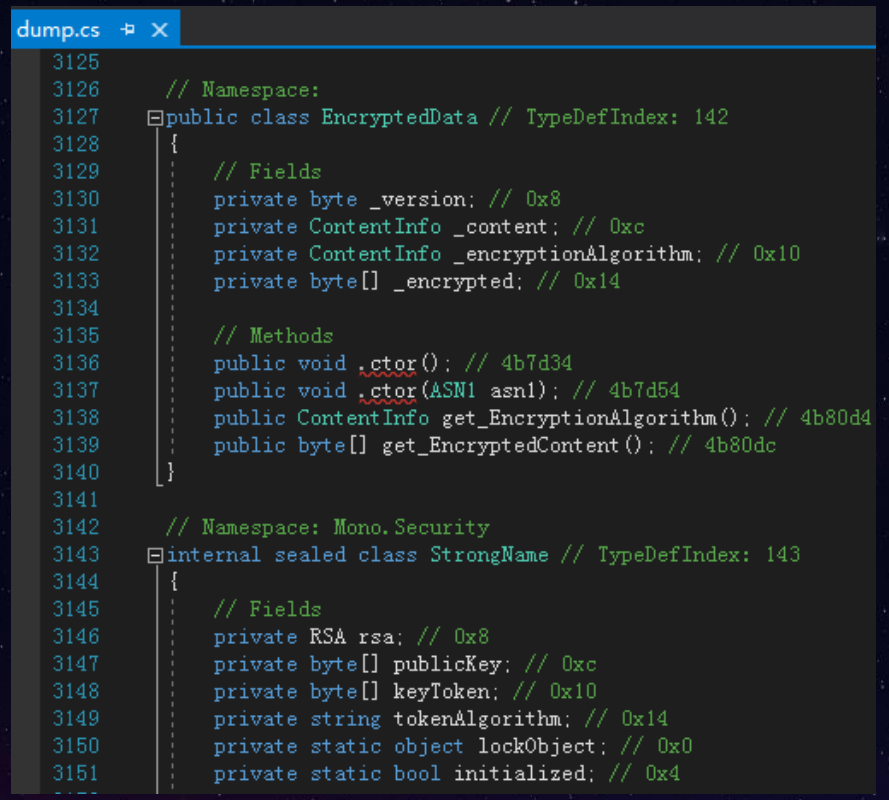
目录

# global-metadata.dat安全

unity项目，在生成App包后，C#脚本里的符号和字符串都保存在global-metadata.dat文件中，在逆向破解时，只要Dump这个文件，可以得到清晰的代码逻辑



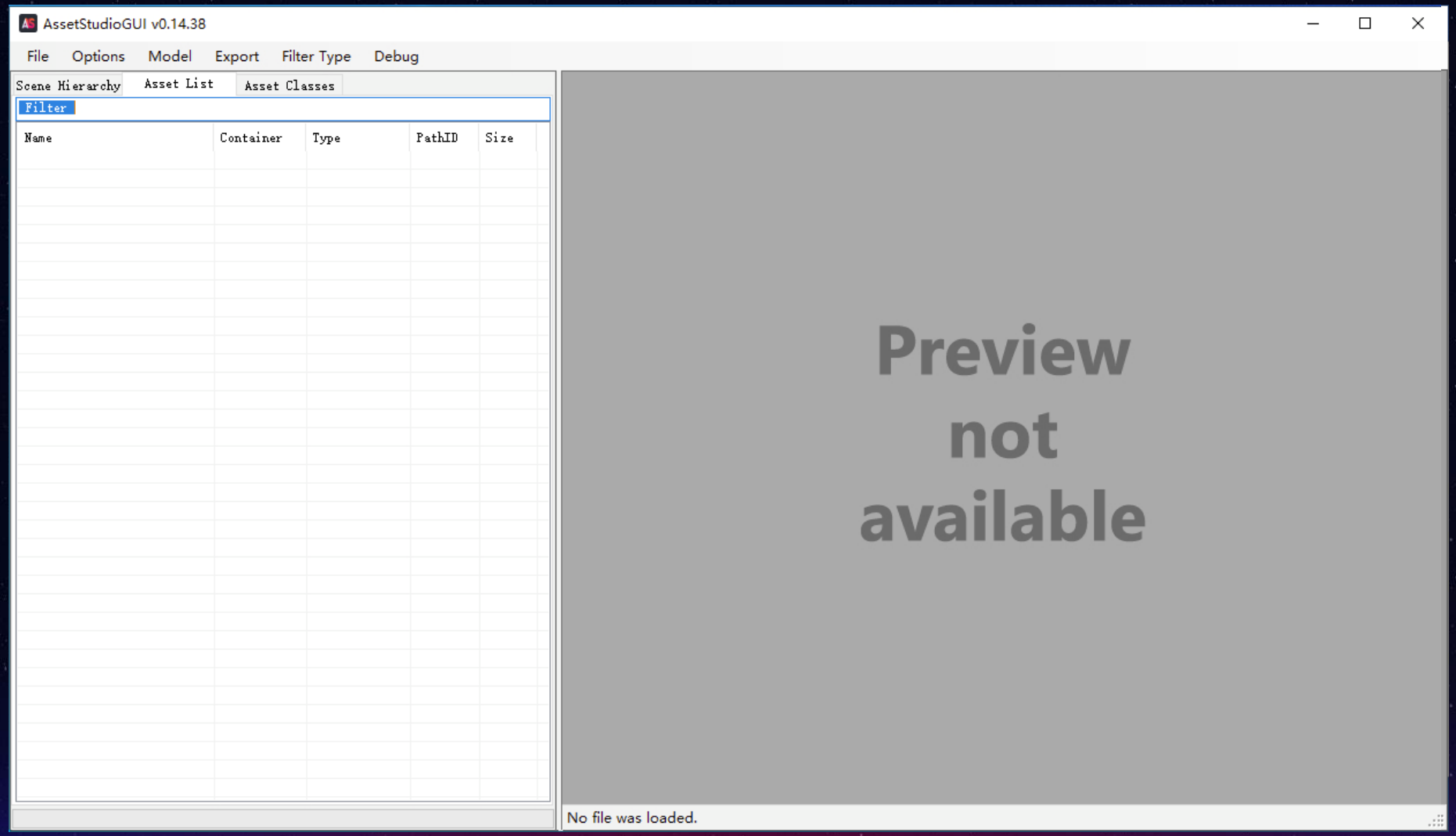

以global-metadata.dat作为输入，使用Il2CppDumper即可进行解析




解析出来的效果：类名、函数名以及对应的偏移都被解析出来了

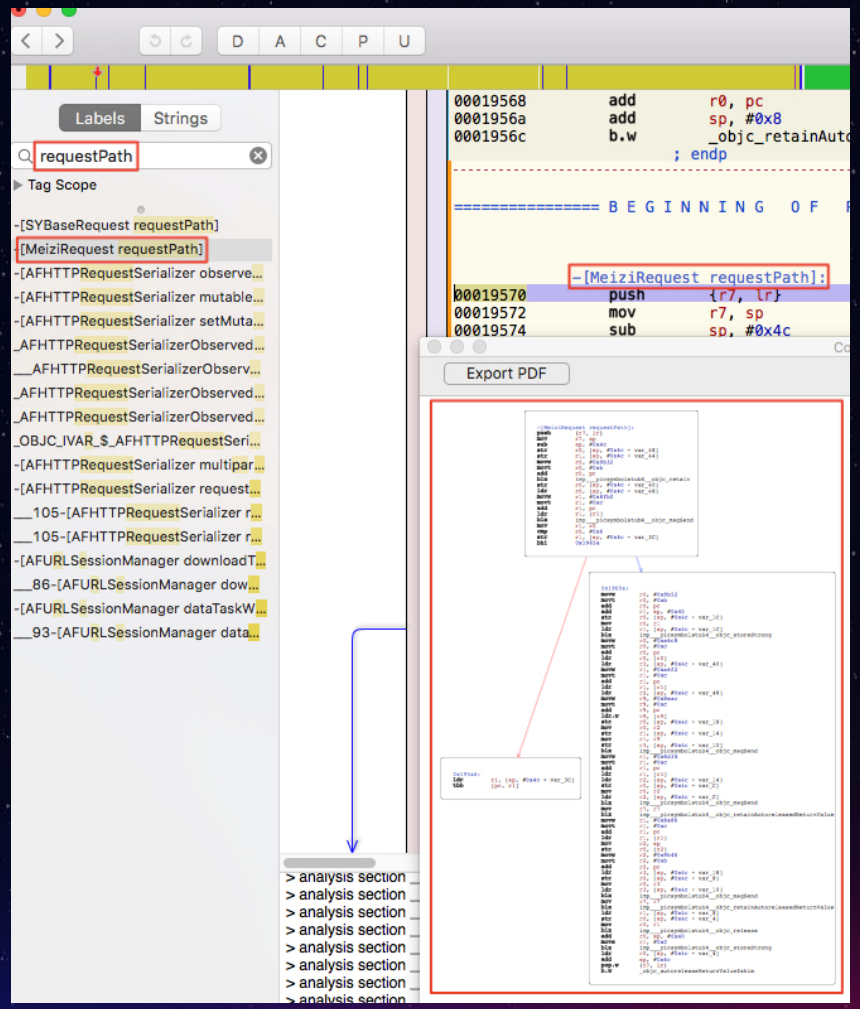
# Assetbundle资源保护

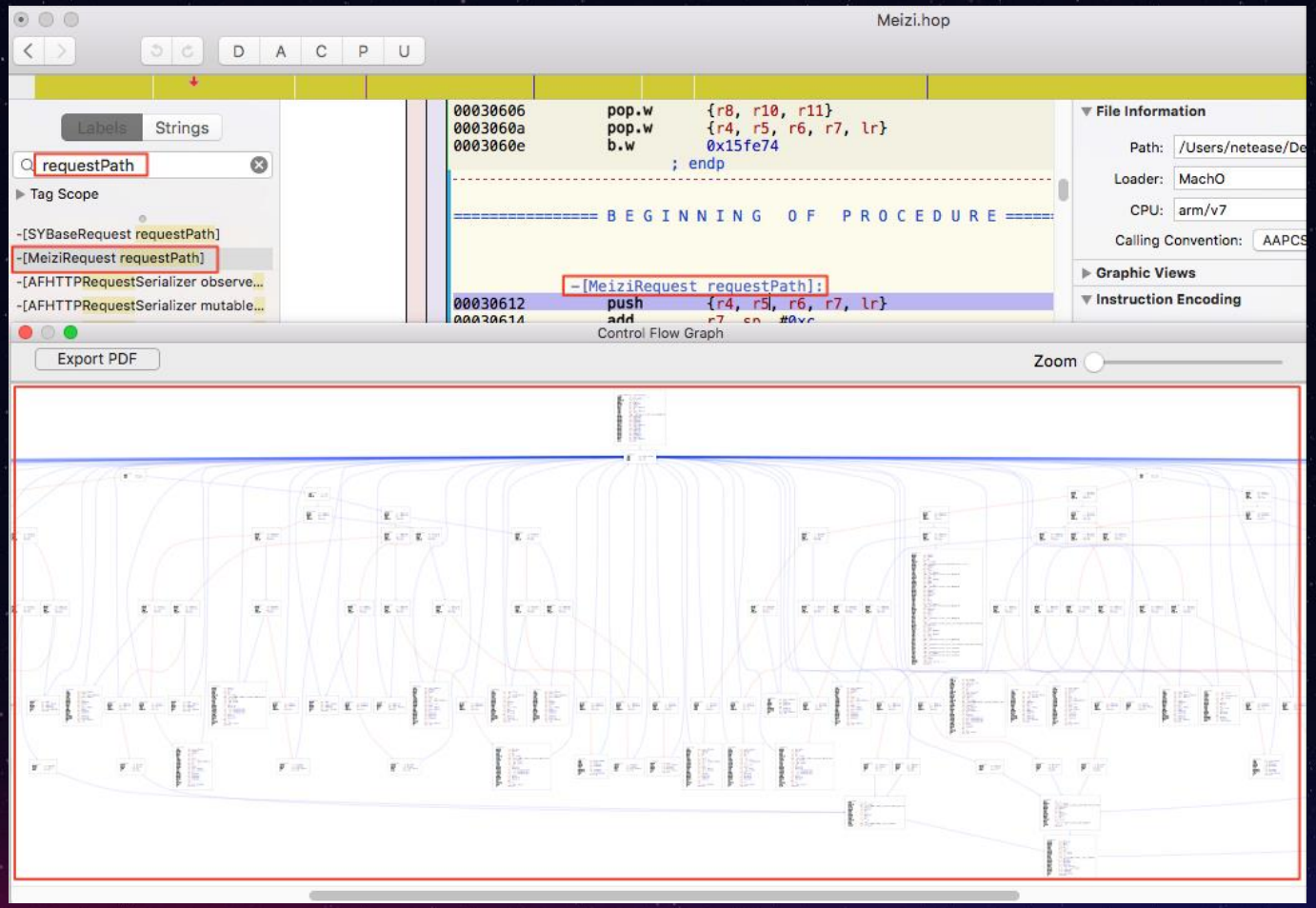
使用一些工具（AssetStudicGUI工具）或者方法，可以直接查看、解析出原始的资源文件



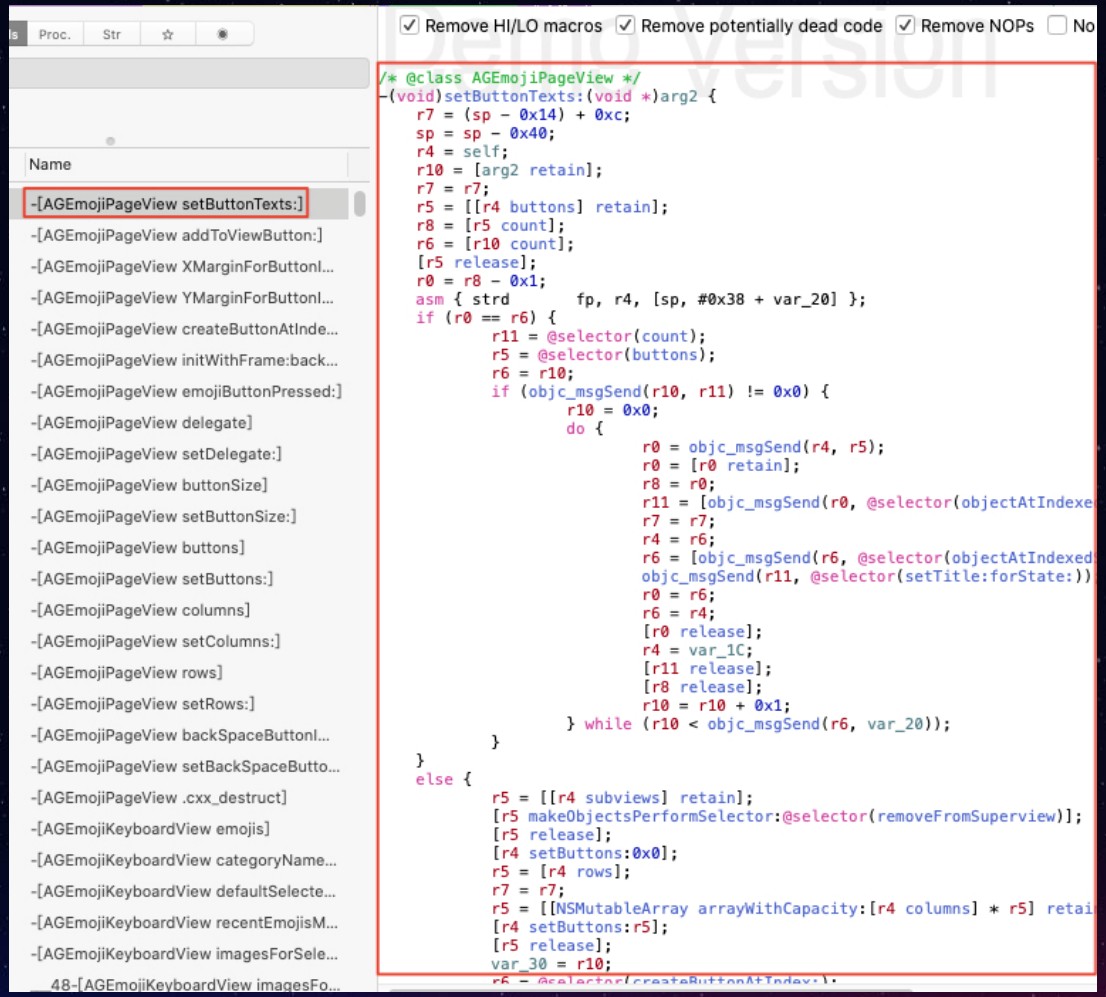
# 代码逻辑分析

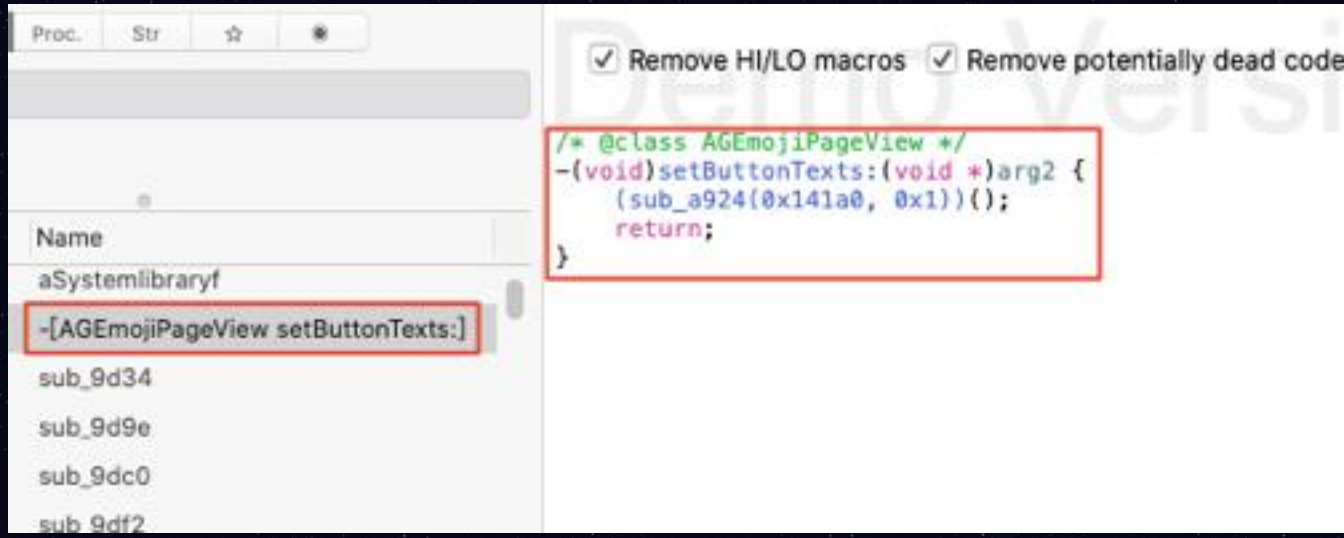
对代码逻辑进行混淆，提高代码的复杂度和逆向分析难度，从而保护程序不被轻易破解



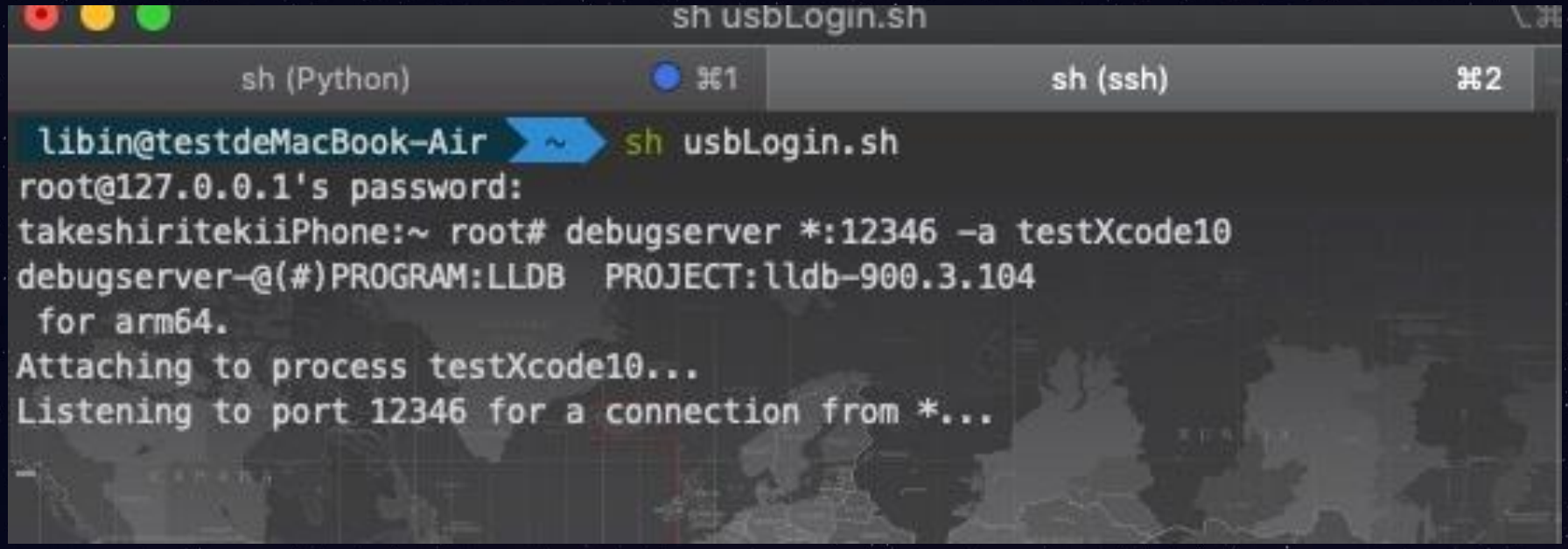
# 防逆向分析


Proc.
Str
Name
-[AGEmojiPageView setButtonTexts:]
-[AGEmojiPageView addToViewButton:]
-[AGEmojiPageView XMarginForButtonI...
-[AGEmojiPageView YMarginForButtonI...
-[AGEmojiPageView createButtonAtInde...
-[AGEmojiPageView initWithFrame:back...
-[AGEmojiPageView emojiButtonPressed:]
-[AGEmojiPageView delegate]
-[AGEmojiPageView setDelegate:]
-[AGEmojiPageView buttonSize]
-[AGEmojiPageView setButtonSize:]
-[AGEmojiPageView buttons]
-[AGEmojiPageView setButtons:]
-[AGEmojiPageView columns]
-[AGEmojiPageView setColumns:]
-[AGEmojiPageView rows]
-[AGEmojiPageView setRows:]
-[AGEmojiPageView backSpaceButtonI...
-[AGEmojiPageView setBackSpaceButto...
-[AGEmojiPageView .cxx_destruct]
-[AGEmojiKeyboardView emojis]
-[AGEmojiKeyboardView categoryName...
-[AGEmojiKeyboardView defaultSelecte...
-[AGEmojiKeyboardView recentEmojisM...
-[AGEmojiKeyboardView imagesForSele...
48-[AGEmojiKeyboardView imagesFo...
Remove HI/LO macros
Remove potentially dead code
Remove NOPs
No
/* @class AGEmojiPageView */
-(void)setButtonTexts:(void *)arg2 {
r7 = (sp - 0x14) + 0xc;
sp = sp - 0x40;
r4 = self;
r10 = [arg2 retain];
r7 = r7;
r5 = [[r4 buttons] retain];
r8 = [r5 count];
r6 = [r10 count];
[r5 release];
r0 = r8 - 0x1;
asm { strd fp, r4, [sp, #0x38 + var_20] };
if (r0 == r6) {
r11 = @selector(count);
r5 = @selector(buttons);
r6 = r10;
if (objc_msgSend(r10, r11) != 0x0) {
r10 = 0x0;
do {
r0 = objc_msgSend(r4, r5);
r0 = [r0 retain];
r8 = r0;
r11 = [objc_msgSend(r0, @selector(objectAtIndexe
r7 = r7;
r4 = r6;
r6 = [objc_msgSend(r6, @selector(objectAtIndexed
objc_msgSend(r11, @selector(setTitle:forState:))
r0 = r6;
r6 = r4;
[r0 release];
r4 = var_1C;
[r11 release];
[r8 release];
r10 = r10 + 0x1;
} while (r10 < objc_msgSend(r6, var_20));
}
}
else {
r5 = [[r4 subviews] retain];
[r5 makeObjectsPerformSelector:@selector(removeFromSuperview)];
[r5 release];
[r4 setButtons:0x0];
r5 = [r4 rows];
r7 = r7;
r5 = [[NSMutableArray arrayWithCapacity:[r4 columns] * r5] retai
[r4 setButtons:r5];
[r5 release];
var_30 = r10;


Proc.
Str
Name
aSystemlibraryf
-[AGEmojiPageView setButtonTexts:]
sub_9d34
sub_9d9e
sub_9dc0
sub_9df2
Remove HI/LO macros
Remove potentially dead code
/* @class AGEmojiPageView */
-(void)setButtonTexts:(void *)arg2 {
(sub_a924(0x141a0, 0x1))();
return;
}

# 防调试

对做游戏外挂的来说，一般都会动态调试分析程序的，所以防调试的用处是非常大的。使用反调试技术,使应用无法被调试，提高被逆向分析的风险



# 防调试

检测是否是调试状态

sysctl();

syscall(202,...);

```
atile("mov x1, %[mib_ptr]\n"
      "mov x2, #4\n"
      "mov x3, %[info_ptr]\n"
      "mov x4, %[size_ptr]\n"
      "mov x5, #0\n"
      "mov x6, #0\n"
      "mov x0, #202\n"
      "mov w16, #0\n"
      "svc #0x0"
```

禁止调试器附加

sysptrace(31);

syscall(26,31);

```
latile(
      "mov w16,#0\n"
      "mov x0,#26\n"
      "mov x1,#31\n"
      "svc #0\n"
      "mov w16,#26\n"
      "mov x0,#31\n"
      "svc #0\n"
```

# 防调试

破解一、遍历 Text 段，找到所有的svc #80 | svc #0 直接替换为 nop指令

```
uint32_t svc_x80_byte = 0xd4000001; // svc #0
while (curr_addr < text_end_addr) {
        svc_x80_addr = (unsigned long)find_svc((void *)curr_addr, (void
*)text_end_addr, (unsigned char *)&svc_x80_byte, 4);
        if (svc_x80_addr) {
            NSLog(@"patch svc #0x80 with 'nop' at %p with aslr (%p without
aslr)",
                (void *)svc_x80_addr, (void *)(svc_x80_addr -
                slide));
            unsigned long nop_bytes = 0xD503201F;
            patch_svc(svc_x80_addr, (void *)&nop_bytes, 4);
            curr_addr = svc_x80_addr + 4;
        } else {
          break;
        }
    }
```

解决方法：判断全局 svc 是否被替换

# 防调试

破解二、遍历Text段，替换ptrace，sysctl ,syscall 的 svc； hook svc 指令

解决方法：通过遍历 Text 段指令，确保自己的内联汇编代码没有被篡改。如右图：

```
    const u_int32_t magic0 = 0x14000000;
    const u_int32_t magicbl = 0x94000000;
    const u_int32_t magic1 = 0xfc000000;
    const u_int32_t mov_w16_byte = 0x52800010; //0x52800010
    const u_int32_t mov_x0_byte = 0xd2800340;
    const u_int32_t mov_x1_byte = 0xd28003e1;
    const u_int32_t nop_byte = 0xd503201f;



    while (end_addr > curr_addr) {
        if (!memcmp(curr_addr, &mov_x1_byte, data_len)) {
            void *pre_addr = (void *)((unsigned long)curr_addr - data_len);
            void *next_addr = (void *)((unsigned long)curr_addr + data_len);
            void *pre_addr2 = (void *)((unsigned long)curr_addr - data_len*2);
            if (!memcmp(pre_addr2, &mov_w16_byte, data_len) && !
memcmp(pre_addr, &mov_x0_byte, data_len)) {
                ///  B  || nop
                if (((*(uint32_t *)next_addr)&magic1) == magic0 || !
memcmp(next_addr, &nop_byte, data_len)) {
                    return curr_addr;
                }
            }
        }
        curr_addr = (void *)((unsigned long)curr_addr + data_len);
    }
```

# 盗版包


2020.01.30更新
修改版iOS无限绿钞
作者 狂野_疾风, 12月 1, 2020
2021.04.04更新不拉回
上帝视角 iOS 一键全图透视 视距透视更新不拉回
作者 狂野_疾风, 11月 14, 2018
上帝视角
制 透视 自瞄 方框 射线 除草 iOS
必须越狱
2021.09.12更新
修改版 Shadow Fight 3 iOS辅助
作者 狂野_疾风, 12月 1, 2020
有效期1年 任意APP签名iOS安装 无需信任
作者 狂野_疾风, 2月 28
2021.02.01更新
全图透视iOS无需越狱
作者 狂野_疾风, 12月 12, 2018
地图透视

目录

# 修改器


**常用修改器:**

- ✓ 八门神器
- ✓ IGG
- ✓ GameGem修改器
- ✓ DLGMemor修改器
- ✓ 其他

# 加速器


使用加速器的效果，游戏进度会随加速倍数加大而不断提高。当倍数提高到10倍后,每个任务几乎是瞬间完成。

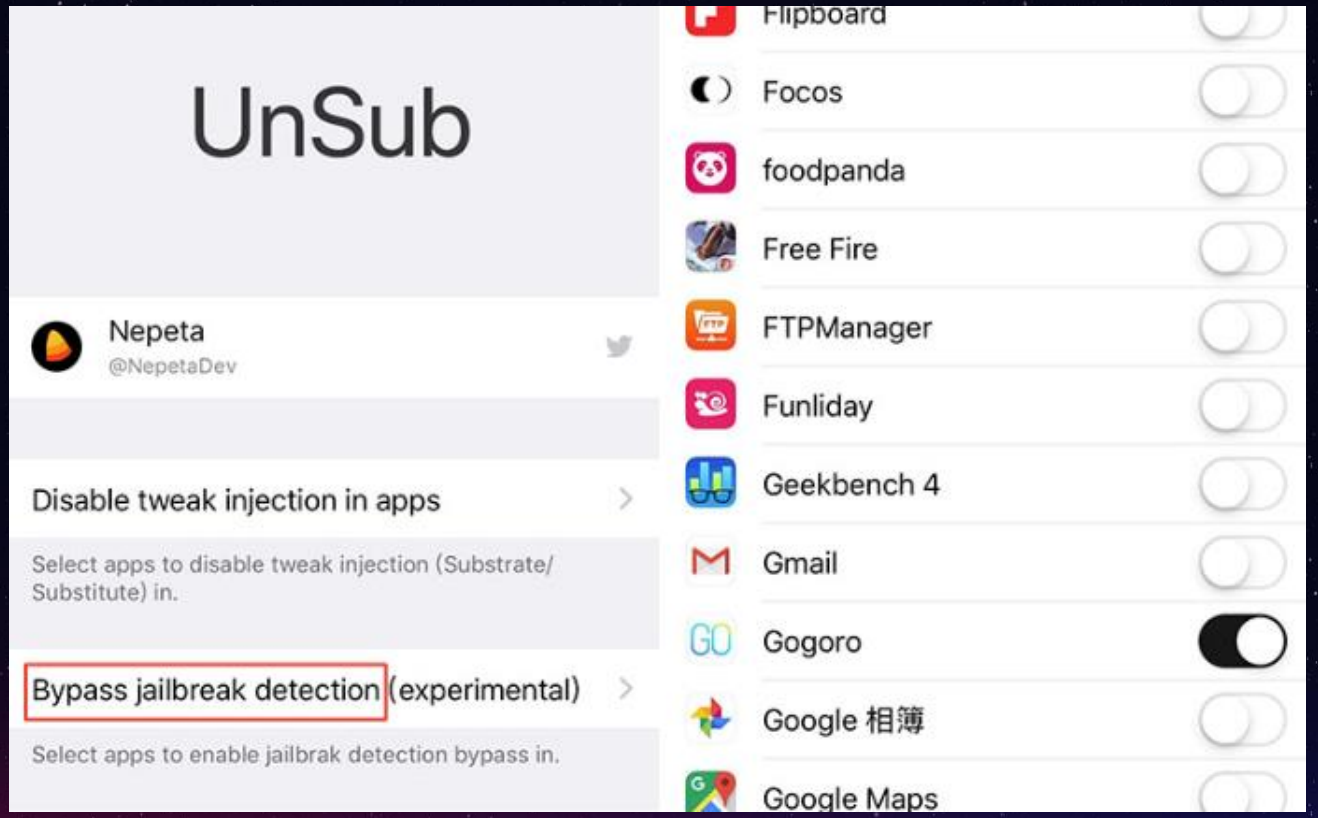
加速器除了可以加速，还可以减速。

# 反越狱插件

用户使用反越狱插件,以致检测不出越狱设备的正确状态，破解游戏的某些功能。

Liberty反越狱


UnSub反越狱



# 云真机

**云真机危害**

1.免越狱

2.自动挂机

3.无限多开

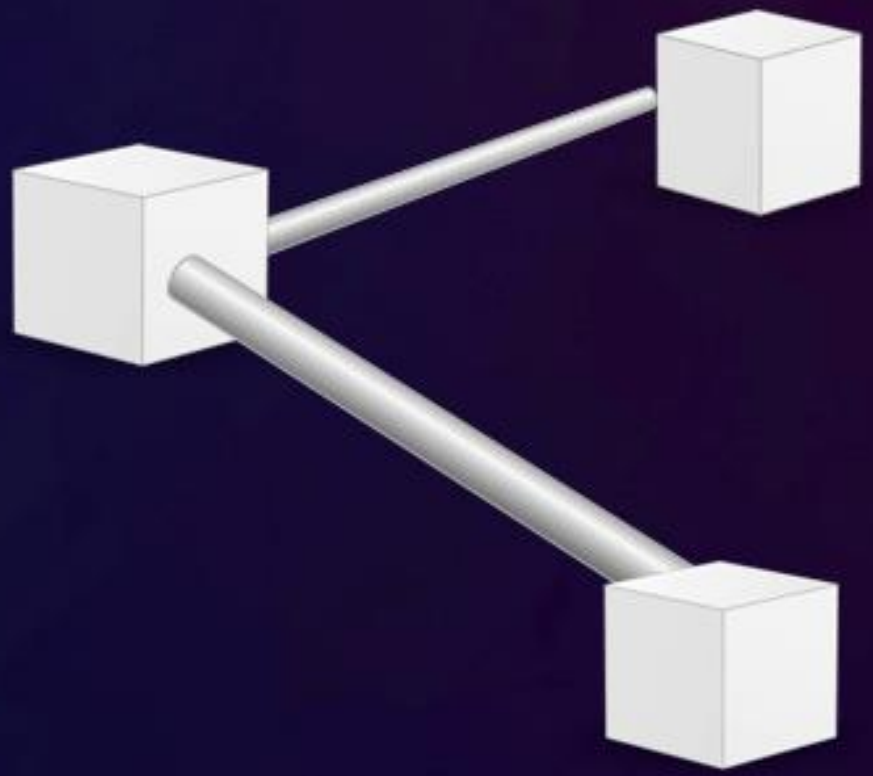
4.自动任务

目录

# 游戏坏账

## ✓ 汇率：

- AppStore中的价格体系和现实世界的货币汇率并不绝对相等。
- 代充店就可以在RMB汇率高时，购买美国的充值卡，然后以较低的价格，给国内的玩家充值。特别是汇率剧变的时候，他们就可以从中赚取差价。



## ✓ 退款：

- 代充店抓住规则的漏洞，有组织大规模地频繁退款，形成了今天淘宝上“职业差评师”的行业。
- 在退款风波最高涨的时候，有些游戏开发商从苹果分得的收入能从70%缩水到不足50%。

## ✓ 黑卡：

- 所谓“黑卡”，即来源不明的信用卡，这里特指与iTunes账户绑定的非法信用卡
- 有些游戏在ranking上排名靠前，流水不错，但是从苹果拿分成的时候，大部分收入变成了无法兑现的坏账，答案就是黑卡。

# 解决方案

## 设备模型

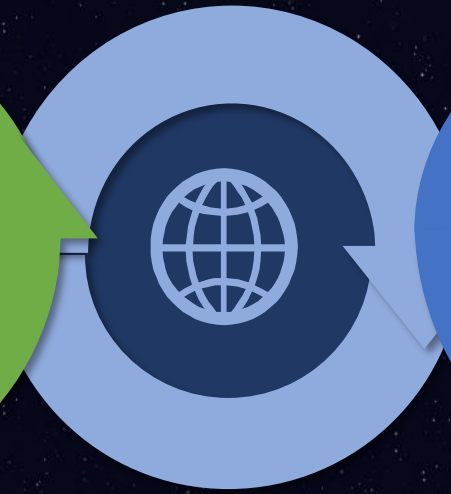
### 唯一标识设备

- ✓ 设备指纹
- ✓ 高稳定性
- ✓ 抗篡改

### 识别恶意设备

- ✓ 异常设备模型
- ✓ 运行环境探测

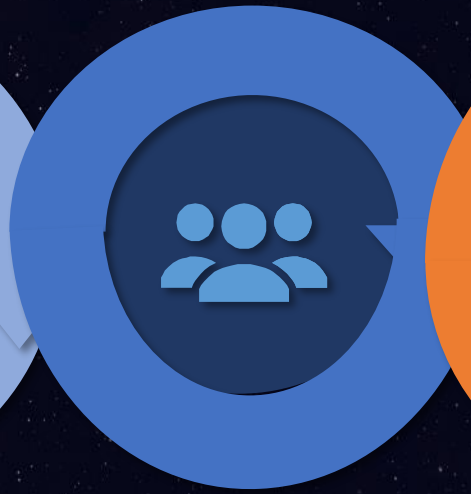
## 行为模型



### 场景化行为模型

- ✓ 正常充值
- ✓ 作弊充值

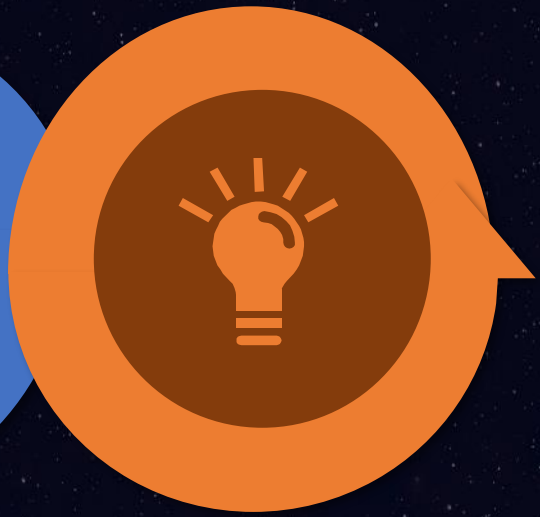
## 数据模型



### 综合数据分析

- ✓ 基于外挂
- ✓ 基于风险
- ✓ 基于画像
- ✓ 基于名单
- ✓ .....

## 规则系统



### 建立综合分析规则

- ✓ 综合上述所有维度
- ✓ 量身定制规则

# 核心能力


**稳定唯一的设备ID**

- 对代充工作室做到持续打击

**丰富的黑名单数据**

- 沉淀了网易十多年的风险名单数据
- 沉淀的支付安全工作室黑灰产库，通用于所有游戏

**综合数据分析**

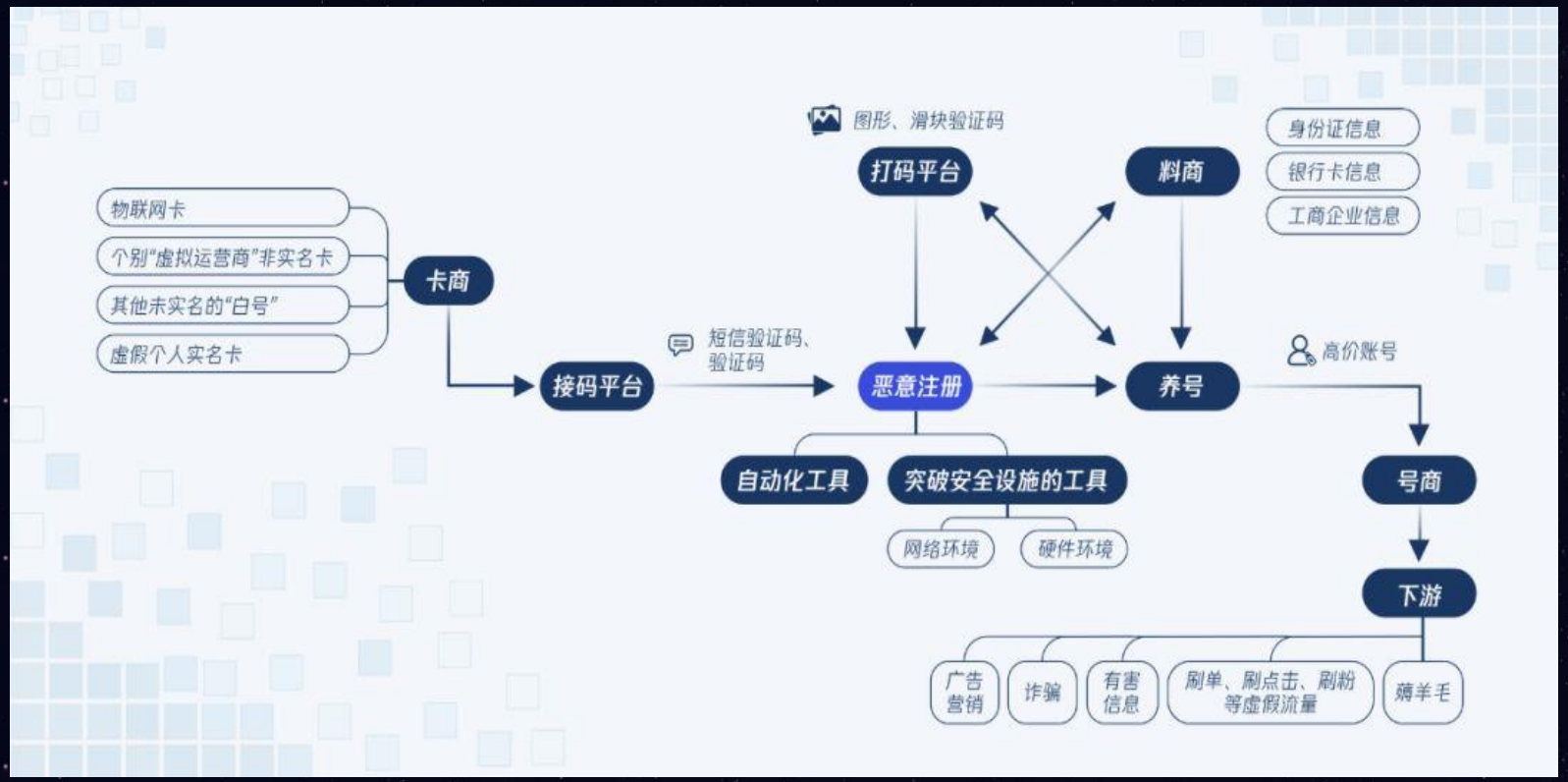
- 正常充值和作弊充值的差异
- 正常用户环境和作弊用户环境
- 用户的作弊行为

**成熟的外挂检测能力**

- 更有效确认代充工作室

目录

# 黑灰产之工作室



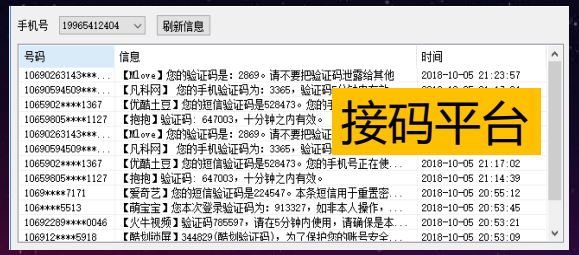


群控设备

刷机工具


卡商


猫池



接码平台

打码平台

# 游戏打金工作室防御

## 工作室危害：

- 批量注册：降低用户质量
- 售卖装备：影响游戏收益
- ...

## 工作室形势：

- 手机墙
- 群控脚本
- 云挂机
- ...


## 防御切入点：

**设备**

- 信息篡改
- 模拟器
- Root/越狱
- 多开器
- 云真机
- ...

**用户**

- 账号
- IP
- 手机号
- 邮箱号
- 注册事件
- 昵称
- ...

**行为**

- 异常时间登陆
- 地理位置异常
- 高频异常
- 批量注册
- 暴力破解
- 撞库
- ...

**业务**

- 地域限制
- 活动次数限制
- 完成活动任务
- 时间限制
- 信息认证
- 活体检测
- ...

# 游戏整体解决方案


业务流程
游戏开发
游戏测试
游戏下载
单人训练
组队对战
技能使用
装备使用
公屏聊天
社区交流
拉新促活
奖池类活动
基础风控
外挂风控
内容风控
运营风控
安全防御
隐私权限合规
权限检测
行为检测
游戏反外挂
iOS
反修改器
反加速器
反反越狱插件
反模拟器
反手游辅助
反云真机
防改机工具
...
内容安全检测
涉黄/涉暴/涉政/违禁/谩骂/
打金工作室
人机识别
小号识别
脚本识别
行为分析
高频检测
游戏加固
防破解
防篡改
防调试
防二次打包
防拉人
引流广告/公会拉人/竞品拉人
代充工作室
黑卡
代充
退款

# 谢谢！

扫码关注网易易盾

携手共建健康绿色游戏环境