客户端安全测试说明

——测试中心

# 一、测试概述

## 1.1、客户端安全测试

从APP软、硬件结构着手，对其本身采取的针对一系列可能存在的被恶意利用的漏洞的全面检测，并对其应用在软件结构，运行环境中可能存在的漏洞和风险做合理的风险评估和风险防范

## 1.2、测试目标

检查被测应用在静态和动态模式中的程序安全、数据安全、业务逻辑安全、系统环境安全、集成插件安全等方面做到全方位的安全测试检测，以做好提前安全防护，为项目组，运营团队提供有效的客户端安全质量评估

## 1.3、测试周期

完整一次安全测试预估周期：5~7天/人

## 1.4、测试条件

### 研发期项目：

项目产品核心功能趋于稳定，且无重大BUG情况下提交测试

### 运营期项目：

重大版本更新，例如：底层更换，功能重构等情况发生时，可由项目衡量提测需求

*其他未提及情况，项目组可在具体情况下衡量后与测试中心沟通是否需要展开此类专项测试*

*以上专项介入时间安排，不涉及接口安全测试，若需增加接口安全，则需要在功能达到一定完善度后单独提交接口测试，不受以上条件约束*

*特别说明：安全测试原理是尽早介入越好，可在早起发现设计上的漏洞以减少后期维护成本*

# 二、测试策略

## 2.1、测试方法

安全测试涉及领域较多，针对各单一领域测试方法整体有所不同，主要核心集中在以下几点：

*外挂类工具的使用，检查其核心数据防御能力*

*资源及核心业务混淆能力，检查其APP包本身的防篡改和核心业务屏蔽能力*

*应用权限配置检查，防止权限被恶意利用造成隐私窃取*

*模块注入，SQL注入验证，防止第三方工具的恶意破坏和机密窃取*

## 2.2、测试范围

*外挂安全测试，如：内存篡改，锁定，游戏加速减速，游戏核心数据截取*

*应用安全测试，如：activity安全，receiver安全，service安全，inter安全，webviwe安全，模拟器安全*

*源码数据测试，如：代码混淆测试，Dex保护，SO保护，资源文件保护*

*接口安全测试，如：远程通信数据加密，日志信息安全，敏感数据加密，接口防挂校验，接口参数容错处理*

# 三、测试指标评定标准

## 3.1、游戏类标准

结合移动应用以及游戏软件自身特点综合执行和标准制定，具体执行手段和评定标准，详见附录1《安全测试项及结果评定标准》

# 四、测试终止标准

## 4.1、环境终止

当提交的版本出现影响测试的重大问题，将终止测试并重启版本提测申请。包含但不限于：服务器环境异常，游戏主流程阻塞，版本为非稳定专项测试版本

## 4.2、过程终止

测试过程中，版本出现突发事件处理需要更换版本的，终止当前版本测试

测试过程中，出现项目优先级变更等第三方原因，终止当前测试

## 4.3、结果终止

当前安全测试工作完成，测试终止

# 五、结果输出

## 5.1、测试文档

测试完成，由专项组针对被测产品，完成并输出以下测试文档：

***《xxx项目安全专项测试用例》***

***《xxx项目安全专项测试报告》***

## 5.2、结果定义

### *高风险：*

1. 致命问题≥1，严重/一般性质问题无要求
2. 不满足1条件下，严重问题≥10

### *中风险：*

1. 无致命问题，严重问题≥5，一般性质问题无要求
2. 不满足1条件情况下，一般问题≥10

### *低风险：*

1、无致命问题，严重问题＜5，一般问题＜10

# 六、附录

## 附录1：安全测试项及评定标准

|  |  |  |
| --- | --- | --- |
| **测试项** | **通过标准** | **可能出现的问题举例** |
| 内存修改 | 1. 无法通过工具静态/动态查找到相应游戏内核心数值并操作（操作包括：修改，锁定，被查找） 2. 客户端/服务端对异常修改数值有校验机制 | 1. 内存地址中反复搜索变化中的金币数，可将其锁定，修改 2. 检索某些存在次数限制的设置，可更改其本次限制 3. 本地战斗结算，可修改其战斗计算结果或战斗过程buff伤害 |
| 加速减速 | 1. 无法通过工具人为的提高游戏内帧率刷新的次数 2. 无法通过工具认为的破话游戏有时间和空间距离限定的规则，如果：CD，位移距离，有限制的动画播放 | 1. 个别倒计时防止本地计算，导致加速很破坏既定规则 2. 有距离限制的空间位移，将在加速情况下破坏既定规则 |
| 内购破译 | 1、Android/iOS无法通过内购破译软件跳过/模拟支付接口，服务器端/客户端有内购验证 | 1、iOS内购程序启动越狱应用后，可以在软件内部购买逻辑开始后欺骗购买成功，达到跳过支付接口完成不真实的支付消费结果 |
| 资源检查 | 1. 安装包或安装后的游戏资源存在应有保护 2. 无法替换游戏内资源进行二次打包并获取利益 3. 安装包中不存在和游戏核心业务相关的明文信息 | 1. 安装包可被反编译修改和重新打包、安装、运行成功 2. 安装包资源中存在和游戏核心业务相关的明文信息 3. 安装游戏成功后，重要信息未在安装路径中隐藏或者加密显示 |
| 代码混淆 | 1. 反编译代码，无法查看到核心业务逻辑代码 2. 无法在反编译后，修改代码逻辑，或借壳打包成功，客户端需要加壳保护或服务器对安装包进行签名校验 | 1. 反编译后可以查看到核心业务代码，例如服务器通讯地址，资源地址，重要业务逻辑原理 2. 修改或增加部分恶意代码后，可以重新打包并安装，运行成功 |
| 敏感信息 | 1、通过检查Logcat，log文件内容，不应该包含敏感信息，如用户资料，业务代码，服务器信息 | 1、可能存在log文件中打印出用户名和密码明文信息，此漏洞很容易被黑客利用 |
| 接口测试 | 1. 通过重发/修改接口协议请求，不能获取额外的利益 2. 不能通过修改，模拟接口协议请求，达到破坏游戏既定规则，造成负面影响，破坏游戏平衡 3. 不能通过修改/模拟/重发等操作接口协议请求，达到跳过游戏限制，恶意跳过客户端限制达到欺骗服务器获取额外利益 4. 服务器和客户端，在接口协议请求上有充分的逻辑判断，尽量保证有判断，可校验 | 1. 例如任务收益，可以重复领取奖励的请求，若服务器无奖励已领判断，或任务状态更变异常，此操作可获取额外收益 2. 例如副本当天挑战存在限制，若服务器无判断，可在挑战次数耗完后，通过接口协议再次进入副本挑战，刷取副本奖励，或者额外利益 |

## 附录2：风险元素级别定义表

|  |  |  |
| --- | --- | --- |
| 风险元素 | 级别 | 定义 |
| 影响面 | 大 | 影响游戏整体平衡（被波及的功能范围广）或因此造成利益 |
| 中 | 影响游戏核心玩法难度或独立区域影响，不影响游戏整体平衡或不会涉及超过2个功能受损 |
| 小 | 不影响游戏整体平衡，不影响游戏核心利益，仅在局部或者独立功能中出现 |
| 发现难度 | 高（容易） | 普通玩家经过常规的研究和少量时间就可以操作，易被发现形成风险 |
| 中（一般） | 普通玩家无需通过特殊方法或者复杂手段，只需要长时间或者一定步骤就可以操作生成风险 |
| 低（不易） | 需要专业技术手段才能被发现和操作（一般此操作较为复杂） |
| 可传播性 | 高 | 聊天，游戏内，工具即可快速达到传播效果并形成风险 |
| 中 | 通过教程，可快速传播并形成风险 |
| 低 | 较难传播，需要采取一定技术手段才能学习和使用 |
| 可识别性 | 高（不易操作） | 协议及其他服务端无此项检测识别，需要更新版本才能修复封堵 |
| 中（容易操作） | 客户端或服务端有相关检测，服务器可通过热更或其他技术手段实现漏洞封堵 |
| 低（策略完善） | 服务器已有策略，可以及时通过人为手段封堵该类问题 |

## 附录3：BUG级别定义

|  |  |  |
| --- | --- | --- |
| BUG级别 | 风险元素 | 举例 |
| 致命 | 任意2风险元素定义：高 | 1. 可以选择未开放的隐藏关卡战斗  2. 可以修改内存地址中存放的数据并因此获益 |
| 严重 | 任意3风险元素定义：中  任意1风险元素定义：高 | 1. 可以人为干预游戏逻辑，但不存在利益 2. 软件层级问题，如log敏感性字眼，权限问题 |
| 一般 | 无高、中风险低于3个元素 | 1. 部分未加密的明文，但不造成核心数据泄露 2. 数据可修改，但无法利用且有校验机制 |

*注：发现难度级别呈相反等级*

## 附录4：测试涉及工具及其说明

|  |  |  |
| --- | --- | --- |
| 工具名称 | 应用领域 | 涉及测试方法简述 |
| 烧饼修改器 | 变速检查、内存修改 | 内置加速功能，内存查找和修改，通过内存地址映射修改获取的内存值 |
| 葫芦侠修改器 | 变速检查、内存修改 | 内置加速功能，内存查找和修改，通过内存地址映射修改获取的内存值 |
| APKtool | 客户端资源检查、反编译 | 反编译APK，class文件，检查内部源码 |
| IDA | 代码混淆检查 | 通过对目标破译，查看被测目标的混淆程度和能力 |
| APKIDE | apk反编译，APK重编译 | 直接对目标APK做反编译破解，并支持修改文件内容后重新打包，强于VTS |
| jd-gui | 源码查看 | 可以直接对class，zip，jar做反编译查看 |
| smail2javaUI | 源码查看 | 可以直接对apk、jar做反编译查看 |
| drozer | 安全组件扫描，SQL注入检查等 | 需搭建drozer安全扫描环境，通过其自身命令逐一对被测app进行安全扫描 |