# 安卓 APP 安全风险分析与渗透测试

葛彦平

南京中新赛克科技有限责任公司，江苏南京，211153；

**摘要：**安卓系统结构由内核层、动态库层、运行时层、应用程序框架层与应用层组成，各层次均存在不同的安全风险。文章从安卓系统各层级的安全特性出发，分析了内核漏洞、动态库篡改、运行时钩取以及应用层存储与通信缺陷等常见安全隐患，并指出代码逆向、数据加密不足、权限验证松散等问题对 APP 安全构成威胁。在此基础上，文章探讨了移动应用安全风险与等保合规要求下的防护要点，提出通过渗透测试手段如逆向分析、环境模拟、动态拦截与组件审计等方法，识别并定位潜在漏洞，强化应用运行环境与安全机制，以提高安卓 APP 的整体安全性。

**关键词：**安卓系统；安全风险；渗透测试

**DOI：**10.69979/3041-0673.24.8.004

## 1 安卓结构介绍

安卓系统的整体架构由多个层级构成，每一层级均承担特定功能，并在安全性方面存在各自的挑战与风险，系统框架见图 1。

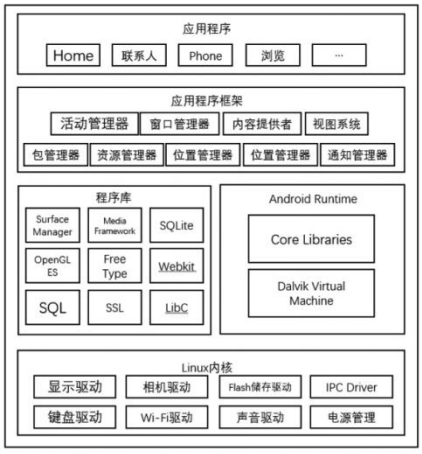


图 1 安卓系统框架图

### 1.1 内核层

安卓系统的内核层基于 Linux 内核构建，它为上层组件提供进程管理、内存分配、网络协议栈以及硬件驱动等基础功能。在内核态中，安全策略多依赖底层权限划分与访问控制机制实现，例如使用 SELinux 策略来限定进程间资源操作范围、采用 cgroups 控制资源分配，从而构建出相对受控的运行环境。对于安全研究而言，内核层漏洞一旦出现，可能导致底层权限突破，使攻击者利用提权手段掌控系统资源，进而间接威胁应用程序数据的保密与完整。研究此层结构可为后续渗透测试提供技术参考，如识别驱动层安全缺陷、探测特权提升通道，以期在攻击链中实现更深层级的突破。

### 1.2 动态库层

动态库层涵盖了系统 C/C++标准库、图形与多媒体处理库、数据库驱动以及网络访问相关库文件。这些库通过 JNI 接口为上层 Java 应用提供关键功能。当攻击者对动态库函数进行拦截、篡改或注入恶意代码时，会对应用中图像渲染、数据处理、网络通信产生影响。对该层的安全研究包括分析库版本兼容性、探查库函数调用栈、审计安全相关函数调用点，以期在特定组件中锁定潜在漏洞点。通过深入理解动态库层的结构与内在关联，可在实际渗透测试中评估加密模块是否易受中间人攻击、数字签名校验能否被绕过、甚至是否存在利用过期或存在已知漏洞的第三方库的隐患。

### 1.3 运行时层

运行时层以 Dalvik 或 ART 虚拟机为核心，承载字节码执行、内存分配及垃圾回收。该层在安全层面涉及对字节码逆向、内存布局分析与运行时钩子技术的防护。例如：攻击者可能利用插桩技术在虚拟机中注入自定义逻辑，从而获取 APP 中敏感数据，或通过改变字节码结构实现加密算法函数的调用路径截断。在运行时层进行深入分析，有助于在渗透测试中使用 Hook 工具如 Xposed、Frida 等框架对目标函数进行动态截获，解析特定功能逻辑，理解数据流与控制流，并挖掘潜藏在中间层的安全缺陷。

### 1.4 应用程序框架层

应用程序框架层由一组核心服务构成，如 Activity Manager、Package Manager、Content Provider、Telephony Manager 等，它们为上层应用提供组件生命周期管理、数据存取接口、系统消息分发等功能。在该层面，安全关注点包括权限管控策略、消息分发机制、防止组件间不当交互等。攻击者可能通过构造恶意 Intent、调用未授权的 Provider 接口、或绕过权限验证逻辑实现数据窃取或资源滥用。在实际测试中，分析框架层的关键步骤在于拆解应用组件间的通信路径、审查敏感信息访问路径，并对相关权限声明与调用栈进行溯源，从而锁定潜在的安全缝隙。

### 1.5 应用层

应用层为用户直观可见的 APP 提供运行空间，具体呈现于 UI 界面、数据处理与业务逻辑中。此层安全问题常体现为存储明文密码、硬编码密钥、网络请求未使用安全协议、逻辑漏洞以及本地数据文件缺乏安全保护。如果应用层存在页面跳转漏洞、组件泄漏或无证书校验的网络请求，则攻击者可轻易利用。对于渗透测试人员而言，应用层是重点关注对象，通过分析 APK 结构、提取 dex 文件、逆向 Java 代码、抓取网络请求报文、检查安全策略实现状况，来定位前端输入校验缺失、数据加密方式不当及接口认证机制松散等潜在风险点。

## 2 安卓 APP 安全风险分析

### 2.1 APP 应用安全风险

在移动应用场景中，APP 所暴露的安全风险类型广泛。例如，在代码层面，攻击者可以通过逆向工具（JD-GUI、JEB、Baksmali）还原敏感逻辑，进而解析密钥算法、收集接口参数；在数据层面，若本地存储缺乏有效加密或访问控制，敏感用户信息如账号、交易记录可能被不法分子直接读取或修改。同时 APP 中若缺乏严谨的业务逻辑判断，一旦存在订单校验不严格、支付环节未加固或权限控制过于宽松，就有可能被利用进行恶意操作，带来财产损失和数据泄露。在网络层面，APP 数据传输若未使用 HTTPS 或未对证书进行严格校验，将为中间人攻击敞开大门，攻击者可拦截和篡改通信数据，甚至植入鱼目混珠的返回结果。

一些 APP 为图方便而在代码中留下调试信息、日志输出或开发测试接口，这些冗余信息为攻击者剖析系统内部实现、寻找弱点提供线索。而应用内若存在对第三方 SDK 调用不慎，则可能引入已知安全隐患的外部组件，使整套应用安全基础被侵蚀。

### 2.2 等保合规下的安全风险

在我国网络安全等级保护制度框架下，移动应用需按照分级策略实施全链路安全防护，保障各环节的安全合规性。在移动终端接入方面，要求无线网络设备开启接入认证功能，支持使用认证服务器或国家批准的密码模块进行鉴权。若未启用认证功能，任何设备均可无授权接入，可能导致网络资源被滥用，并对敏感数据的安全性构成威胁。未受控的接入权限将扩大安全攻击面，带来不必要的风险隐患。

入侵防范层面强调对无线设备及移动终端的严格管控，要求精准定位并阻断未授权的接入行为。如用户私自搭建非法 WiFi 或恶意钓鱼网络，将严重威胁其他用户的信息安全，导致数据泄露与滥用。通过全面的设备授权管理机制，可以有效限制未授权无线设备接入，避免网络安全边界被随意突破，保障系统稳定运行。

移动应用管控和软件开发环节需强化安全措施，确保各环节可控可管。移动终端应用应通过白名单机制限制安装范围，并使用合法签名证书验证应用的完整性，防止恶意篡改或未授权应用运行。此外，开发者资质、技术水平及签名证书的合法性需严格审查，避免因开发者能力不足或恶意行为导致安全漏洞。通过加强应用全生命周期的安全控制，可以有效降低系统风险，提升移动网络的整体防护能力。

## 3 安卓 APP 渗透测试

### 3.1 程序漏洞

渗透测试旨在通过模拟攻击者视角评估 APP 存在的程序性漏洞。其中常用手段包括逆向工程与调试分析：利用反编译工具从 APK 中提取 DEX 文件，将其转化为可读 Java 代码，查找敏感函数、硬编码密钥及无加密传输接口。测试者可通过修改 Smali 代码插桩、强行改变逻辑分支，观察应用在异常输入下的响应，以便发现潜在逻辑漏洞。例如，支付流程若无严谨校验，测试者可能通过绕过订单检查强行发起支付请求；若用户认证逻辑存在硬编码 Token 或对设备指纹检测薄弱，则可模拟合法用户行为，获取敏感资源。

除了逆向分析，测试人员还会借助调试器、Hook

框架以及内存注入技术，对应用运行过程中的关键函数进行截获。例如，通过 Frida 脚本对特定函数进行动态监控，实时查看函数入参与返回值，从而找到敏感数据泄露点或识别关键验证步骤的薄弱环节。若 APP 采用混淆和加固手段，渗透人员仍可借助脱壳、动态内存 Dump 及 Patch 技术对关键组件进行全面挖掘。此类测试活动不仅能识别明显的漏洞点，也可深入挖掘深层嵌套的安全软肋。

### 3.2 运行安全

运行安全测试关乎应用对外部恶意环境的抵抗程度。攻击者可在非官方环境下运行 APP，如使用 Root 设备、模拟器对应用进行逆向和分析；若 APP 缺乏有效的环境检测机制，则很可能被恶意修改。在渗透测试中，可通过搭建动态分析环境，对 APP 在不同运行条件下的行为进行观察：

①模拟 Root 环境：通过暴露 su 命令、修改系统属性观察 APP 是否进行 Root 检测与防护。若无相应策略，测试人员可轻易拦截应用数据流量或修改内存参数。

②抓包与中间人攻击：使用代理工具对 APP 网络通信进行拦截，若无法对证书链条进行严格认证，那么数据流将被窜改或窃取。

③Hook 与虚假环境注入：通过对 Dalvik/ART 层注入自定义逻辑，渗透人员可查看函数调用栈、修改返回值，实现对应用关键流程的干预。若应用无检测调试器或 Hook 框架存在的措施，则可能被攻击者轻易分析内部逻辑。

④动态权限与组件访问：对测试者而言，通过故意构造异常请求来访问未授权组件，可评估应用在运行态下的访问控制策略强度。

在运行安全的测试中，分析 APP 对环境多样性的适应程度以及对运行时异常条件的应对策略是重点。如果应用的安全防护机制过于单薄，如仅简单校验 Build 信息或将签名校验流程置于易修改的逻辑代码中，那么渗透者可轻松伪造满足条件的运行环境，从而取得安全防线的突破。

## 4 结语

安卓 APP 安全问题涉及多个层级，每一层都可能成为潜在的攻击入口。通过深入剖析系统结构与常见安全风险，有助于理解攻击者的思维路径，进而制定有效的防御策略。渗透测试作为安全评估的重要手段，不仅能够揭示代码漏洞和权限缺陷，还能识别运行环境中的薄弱环节。结合等保合规要求，提升系统防护能力，强化应用全生命周期的安全管理，是应对复杂安全威胁的有效途径。随着攻击手段的不断演变，持续完善测试方法和安全机制，将成为保障移动应用安全的有力支撑。

## 参考文献

[1]南姿.安卓 App 配置安全性检查方法研究[D].西安科技大学,2022.

[2]宋恺,邓佑军,王浩仟,等.安卓应用软件代码签名的风险挑战与应对措施[J].信息安全与通信保密,2023,(09):36-44.

[3]张威楠,孟昭逸,熊焰,等.基于异质信息网络的安卓虚拟化程序检测方法[J].计算机应用研究,2023,40(06):1764-1770.