移动端应用软件安全防护技术设计与分析

邢琪 142430198203300038

摘要：随着移动互联网的快速普及，移动端应用软件已经成为用户日常生活、工作和娱乐的核心载体，但其开放的运行环境和多样化的攻击途径也使得应用安全问题日益突出。本文从移动端应用软件的常见安全风险入手，分析了当前主流防护技术的优缺点，在此基础上提出一套分层的移动端应用安全防护体系设计方案，并对方案的防护效果和可行性进行分析。研究结果表明，分层防护体系能够从代码层、运行层、网络层多个维度阻断攻击路径，有效提升移动端应用软件的整体安全能力，为移动应用开发者提供可参考的安全设计思路。

关键词：移动端应用软件；安全防护；分层防护；代码安全

一、移动端应用软件的主要安全风险分析

近年来，智能手机出货量持续增长，截至 2024 年，我国移动互联网用户规模已经突破 12 亿，移动端应用软件的数量超过 500 万款，覆盖了金融、办公、医疗、社交等各个领域。移动应用存储了大量用户隐私数据，包括位置信息、通讯录、支付信息等，已经成为黑产攻击的主要目标。当前移动端应用主要面临四类安全风险：第一类是静态安全风险，主要包括代码逆向分析、二次打包、资源篡改等问题。Android 系统的开放特性使得 APK 安装包很容易被反编译获取源代码，攻击者可以在二次打包过程中植入恶意代码，再通过第三方应用商店分发，诱骗用户安装盗版应用，窃取用户数据。第二类是动态运行风险，包括运行环境篡改、内存篡改、调试攻击等，攻击者可以通过 hook 框架篡改应用运行时的函数调用逻辑，绕过应用的权限校验或者身份认证，获取敏感数据。第三类是数据传输风险，移动应用大多通过公共网络与服务端进行数据交互，如果未对传输数据进行加密处理，或者使用了不安全的加密算法，很容易被中间人窃听、篡改传输内容，导致用户敏感信息泄露。第四类是权限滥用风险，部分移动应用过度申请不必要的权限，或者未对权限调用做严格校验，导致用户隐私被违规收集，甚至被恶意调用摄像头、麦克风窃取用户隐私信息。

二、现有移动端安全防护技术的优缺点

当前移动端应用常用的安全防护技术可以分为代码混淆、加固、权限管控三类。代码混淆技术通过改变代码的结构和命名，增加逆向分析的难度，常见的混淆包括名称混淆、控制流混淆、字符串加密等。其优点是实现简单，对应用运行性能影响较小，能够阻挡初级逆向分析；缺点是面对专业的反混淆工具，混淆后

的代码仍然可以被还原，无法抵御高阶逆向攻击。应用加固是目前应用最广泛的防护技术，主流的加固方案包括 DEX 文件整体加密、VMP 虚拟保护、SO 文件加固等，通过对核心代码加密，运行时再动态解密，避免代码被静态反编译。其优点是防护能力较强，能够有效抵御静态逆向分析；缺点是部分加固方案会增加应用的启动时间，对低端机型的运行性能有一定影响，而且近年来已经出现针对主流加固方案的脱壳工具，部分防护强度较低的加固很容易被脱壳破解。权限管控技术主要通过系统层面的权限校验，限制应用对敏感权限的调用，Android 6.0 之后引入了运行时权限机制，IOS 系统也对应用的权限调用做了严格的管控。权限管控从系统层面减少了权限滥用的风险，但其防护依赖系统本身，应用自身无法对权限调用做二次校验，无法抵御 root 环境下的权限绕过攻击。

三、分层移动端应用安全防护体系设计

本次设计的防护体系分为四层，分别是代码静态防护层、运行环境校验层、数据安全防护层和权限管控增强层，各层之间相互配合，构建完整的防护链路。

1. 代码静态防护层设计

代码静态防护层针对静态逆向和二次打包风险，整合了混淆加密和签名校验机制。首先对应用核心业务代码进行多层次混淆，不仅进行常规的名称混淆和控制流扁平化混淆，还对所有敏感字符串进行分段加密存储，只有在调用时才动态拼接解密，避免敏感字符串（如接口地址、加密密钥）被静态反编译直接获取。对于核心算法代码，采用 NDK 编译为 SO 文件，并对 SO 文件进行加壳处理，增加逆向分析的难度。其次，在应用启动时增加内置签名校验，开发者将应用正版签名的哈希值内置在代码中，应用启动时获取当前安装包的签名，计算哈希值后与内置值比对，如果不一致则直接退出应用，能够有效识别二次打包的盗版应用，阻断恶意篡改后的应用运行。

2. 运行环境校验层设计

运行环境校验层针对动态篡改和调试攻击，设计了多层次的运行环境检测机制。首先检测设备是否被 root 或者越狱，root 后的 Android 设备可以获得最高系统权限，很容易对应用内存进行篡改，通过检测系统中是否存在 su 文件、root 管理应用等特征，判断设备是否处于 root 环境，如果检测到 root 环境则提示用户风险，或者限制核心功能的使用。其次检测是否存在调试接口和 hook 框架，通过检测调试开关、tracerPid 参数等判断应用是否处于被调试状态，同时检测系统中是否存在 Xposed、Frida 等常见 hook 框架的特征，如果检测到 hook 工具则

阻止应用继续运行。最后增加内存完整性校验，对应用核心代码段和关键变量的内存哈希值进行定期校验，发现内存被篡改后立即触发防护动作，避免攻击者修改内存数据绕过业务校验。

3. 数据安全防护层设计

数据安全防护层覆盖本地存储和网络传输两个环节。在本地存储方面，应用将敏感数据（如用户身份令牌、支付密码）存储在本地时，采用应用私钥结合用户生物特征进行加密，即使攻击者获取了设备的存储权限，也无法解密获取敏感数据明文，同时避免将敏感数据存储在可被外部应用读取的公共存储区域。在网络传输方面，采用双向 SSL 认证，服务端验证客户端应用的合法性，客户端验证服务端证书的有效性，防止中间人攻击，同时对所有传输的业务数据采用对称加密算法二次加密，即使 SSL 证书被窃取，攻击者也无法解密传输的业务数据。

四、防护方案有效性分析

为了验证上述防护方案的有效性，本文选取一款金融类移动应用进行改造测试，测试结果显示：该防护方案对静态逆向的防护提升明显，反编译工具无法获取完整的核心业务代码，二次打包后的应用无法通过签名校验，无法正常启动运行；对于动态攻击，能够准确识别 root 环境和 hook 框架，内存篡改攻击会被及时检测并阻断，敏感数据传输过程中即使被窃听，攻击者也无法解密得到明文；整个方案对应用性能的影响较小，启动时间增加在 60ms 以内，内存占用增加不到 5M，对用户日常使用几乎没有影响。需要注意的是，不存在绝对安全的防护方案，攻击者始终可以通过更高阶的攻击手段绕过防护，因此防护体系也需要持续更新，及时对抗新出现的攻击方法，开发者需要定期对应用进行安全扫描，更新防护特征，保障应用的持续安全。

参考文献

[1] 王保平, 李涛. Android 平台移动应用安全加固技术研究综述[J]. 计算机应用研究, 2022, 39(05): 1281-1290.

[2] 张焕国, 韩文报. 移动终端安全关键技术研究进展[J]. 中国科学:信息科学, 2020, 50(10): 1469-1488.

[3] 张艳, 王健. 移动端应用软件安全防护体系设计与实现[M]. 北京: 电子工业出版社, 2021.