基于同源策略的移动应用细粒度隐私保护技术

卢文雄，王浩宇
北京邮电大学计算机学院，北京 100876

摘要
Android等移动平台基于权限的访问控制机制是作用在应用粒度上的。应用中除了包含应用开发者本身的代码以外，还包含第三方库代码，导致应用权限滥用情况严重。引入类似浏览器同源策略的细粒度控制机制，打破了应用之间的界限，将粒度细化到代码来源。将控制机制实现到Android系统层，并提供了一套插桩工具对应用进行修改。实验结果表明，系统能够起到允许或禁止特定开发者执行特定敏感行为的作用。

关键词
隐私保护；第三方库；访问控制；移动应用

中图分类号：TP309.2 文献标识码：A doi: 10.11959/j.issn.2096-0271.2020003

Same origin based fine-grained privacy protection for mobile applications

LU Wenxiong, WANG Haoyu
School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract
Mobile systems, such as Android, use permission-based access control mechanism, which is at the granularity of each application. Apart from the code from developers themselves, applications also contain code from third-party libraries, which has led to serious overuse of application permissions. A novel origin-based (similar to browsers) and fine-grained control mechanism was introduced, which broke the boundary between applications in terms of access control and fine-grained the granularity to the level of code source. The mechanism was implemented onto Android framework, and a set of tools to modify applications were also offered. Experiment results suggest that system can allow (or limit) certain developers to execute certain sensitive behaviors.

Key words
privacy protection, third-party library, access control, mobile application

1 引言

近年来随首移动智能终端的快速发展，智能手机已经融入人们的日常生活，多样的移动应用带来了丰富的功能和友好的用户体验。目前各个主流移动应用市场中均有数百万的移动应用[1]，这些应用及移动终端中的海量用户信息构成了移动应用大数据生态系统。

在移动智能终端和多样的移动应用给用户带来便利的同时，移动应用大数据生态系统中各种新的安全和隐私问题也日益凸显。一方面，移动平台的恶意软件增长迅速，这些恶意软件会在用户不知情的情况下，恶意扣费、破坏系统、窃取用户隐私等，给用户带来经济损失和隐私泄露问题；另一方面，移动应用隐私滥用情况严重，如很多应用会获取用户的联系人信息和地理位置信息，用于定制化广告服务、第三方分析或者其他与应用功能相关的服务等。智能手机上存储着用户的各种隐私信息（如联系人、通话记录、照片、地理位置信息等），这些信息很容易被应用获取并泄露。因此，移动应用大数据生态系统中的安全隐私问题十分严峻，成为用户及研究者关注的焦点问题。

当前主流移动平台（如Android和iOS）均使用权限模型来控制移动应用对隐私信息的访问。目前，Android和iOS移动系统均使用运行时权限，即在应用第一次访问隐私信息时，系统会弹出一个提示框让用户决定是否允许该应用访问相应隐私信息。如果用户同意，则应用就会获得相应权限，并且后续仍具有访问该隐私信息的权限。然而，当前的移动平台上广泛存在权限滥用问题。很多应用经常申请不必要的敏感权限，这给用户隐私信息带来了被泄露的风险。很多应用会在用户不知情的情况下获取并泄露用户的隐私信息。用户很多时候不了解应用是否需要使用权限以及使用权限的原因，因此很难对应用的权限进行管理。

当前移动平台上基于权限的访问控制通常采用“全有或者全无（all-or-nothing）”的方案，即要么允许应用使用某个权限的所有行为，要么禁止应用使用该权限，而不能根据应用的行为选择性地赋予其权限。移动应用的一个特点是广泛使用第三方库，且第三方库与宿主应用本身共享相同权限，因此很多第三方库存在侵犯隐私或者越权行为，这就带来了很多安全隐私问题[2-3]。然而，用户无法了解应用会如何使用隐私信息，更不能根据用户隐私偏好对隐私信息使用细粒度控制（如只允许使用位置信息进行地图搜索，而不能使用位置信息进行广告推送和第三方分析）。

针对这一问题，本文提出了基于同源策略的移动应用细粒度隐私保护技术，将细粒度具体到开发者，可以允许或禁止特定开发者执行特定敏感行为。

2 研究背景及相关工作

2.1 研究背景

2.1.1 Android访问控制机制

Android系统采用权限模型对敏感信息的访问进行控制。如**图1**所示，在早期Android版本（6.0版本以下）中，用户根据应用声明的权限决定是否安装，安装即允许应用使用相应权限。自Android 6.0版本起，系统增加了运行时的权限检查机制，在目标应用程序接口（application

programming interface，API）的版本号大于或等于23的应用中，对于部分敏感权限，应用需要在首次使用时向用户发出请求，获得允许后才能使用相关权限。

在Android系统中，普通应用采用的权限级别如下[1]。

- 正常权限：如蓝牙、访问网络状态。
- 危险权限：如获取联系人、获取位置信息。此外，还有一些事关系统安全的特殊权限，其保护级别要求更高，如“修改系统设置”。
- 危险以上级别的权限：由于存在隐私泄露或系统安全风险，需要经用户允许才可使用。

Android 6.0及以上系统将危险权限分为9个权限组，包括日历、位置、联系人、电话等，用户可以根据应用功能，以权限组为单位选择是否允许使用权限。

2.1.2 第三方库引入的权限风险

Android应用中大量使用第三方库，研究表明，移动应用中大约有60%的代码属于第三方库[4]。这些第三方库包括广告库（如Admob）、社交网络库（如Facebook）、第三方分析库（如Google Analytic）等。第三方库拥有与宿主应用相同的权限，并且不能与应用核心代码进行权限分离。应用所用的第三方库代码和应用本身的核心代码均在同一个应用的字节码中，在权限模型中，一个应用实际执行时，其权限声明及设置的作用范围是整个应用，包括核心代码及第三方库。如果一个应用被用户授予与某种隐私相关的权限，则第三方库也能访问与之相关的隐私信息。在实际的使用过程中，这种授权方式存在越权访问风险[3]。如导航应用的开发者为了实现功能，声明了与位置相关的危险权限，用户为了确定自己所在的位置，授予导航应用获取位置的权限。而现有权限机制下，同一个导航应用中的广告库、分析库等也可以访问用户的位置，用户的位置信息通过上述越权访问方式，在用户及开发者对第三方库均缺乏了解的情况下被泄露到非预期的域。

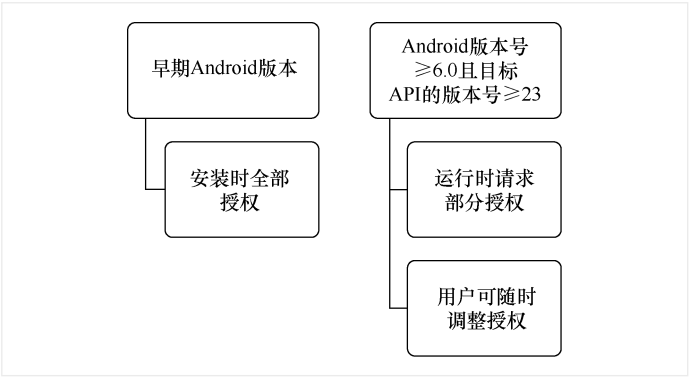


图1 Android 访问控制机制示意

2.2 相关工作

2.2.1 面向第三方库的安全分析和访问控制

关于第三方库的越权访问风险，Ryan S等人[3]曾进行相关研究，对比了广告库用到的权限与文档说明的权限，发现很多广告库会使用文档说明的权限之外的权限，并指出在应用中使用广告库，实际上允许了广告库具有与宿主应用相同的权限。当广告库动态检测到所在应用具有相应权限时，它们就会使用这些权限（其中很多是敏感权限），进而使广告提供者从用户设备中提取相关信息。该研究还指出，不同于浏览器，Android应用一旦被授予某种权限，第三方库就会自动获得该权限，导致敏感信息泄露到应用核心以外的域。

针对第三方库引入的安全隐私问题，学术界提出了一些细粒度访问控制机制。例如，Liu B等人[5]提出了一个Android应用细粒度访问控制系统，主要对第三方库中广告库的安全风险进行访问控制。该研究中访问控制的机制是利用机器学习区分应

用核心和广告库，以应用为基准，允许/禁止应用核心或广告库访问特定资源，但不会影响其他第三方库。系统对该应用的所有广告库的同一类访问采用相同的设置，即都禁止或都允许。例如，针对某款社区生活服务应用，可以设置允许应用核心访问精确位置，但禁止所有广告库访问精确位置，以此将应用核心和广告库能访问的信息进行区分。AdSplit[6]对Android系统进行了扩展，使得应用与其使用的广告库运行在不同的进程中，从而对广告库的行为进行控制。AdDroid[7]引入针对广告库的API和权限，使得广告库的功能能够从应用中分离出来。在AdDroid提供的开发框架下，应用不再需要集成广告库，而是通过使用扩展的广告API来集成广告功能。应用可通过广告API配置一些信息（如要使用的广告平台以及广告的上下文信息等），广告API会从广告平台服务器获取广告，并处理用户界面（user interface，UI）事件。Roesner F等人[8]提出通过修改Android系统来允许广告的UI嵌入应用界面，但并不赋予广告库敏感数据和权限。Perman[9-10]通过动态代码插桩技术对隐私信息的使用进行细粒度控制，允许用户指定策略来管理不同模块（包括第三方库和应用本身）的权限。胡冰惠[11]提出了FineDroid系统，用于追踪分析隐私获取、传播、泄露及返回流接收、传播的全过程，解决第三方库隐私泄露问题。Diamantaris M等人[12]在不需要修改系统或者应用的前提下，提出了一套动态分析框架，能够在运行时准确地判断隐私信息的使用者是应用本身还是嵌入的第三方库，在此基础上能够对第三方库实现细粒度控制。

2.2.2 移动应用权限的细粒度控制

此外，学术界有大量的工作关注移动应用权限的细粒度访问控制。例如，Wang H Y等人[13-14]提出了基于权限使用意图的访问控制，即首先分析隐私信息使用的意图（如使用位置信息进行地图搜索或者广告推送），然后允许用户指定基于意图的访问控制策略，在运行时进行实时控制。学术界提出了类似FlowDroid[15]的系统追踪隐私信息流，并尝试对信息流进行分析，以此区分正常和异常的隐私信息使用[16]。另外，其他工作包括对权限进行细化或者根据上下文进行细粒度的访问控制[17-18]、针对最小权限原则对应用进行分析和优化[19-20]等。

总体来看，虽然学术界有大量工作关注于细粒度访问控制，然而目前的研究工作仅仅是针对单个应用本身进行细粒度控制的，并没有考虑应用中代码的来源，即开发者。对于不同的开发者，用户有不同的信任级别，而且同一个开发者的多个应用中存在着隐私信息共享的情况，因此只需要对不同的开发者设置权限，而不是针对每个应用单独进行权限管理。

3 系统设计

3.1 总体设计

本文提出了一个基于同源策略的Android应用访问控制原型系统，与之前的工作相比，区别在于引入了类似同源策略的访问控制机制。本文提出的方法打破了应用之间的界限，并将粒度细化到开发者，从单个应用角度看，进行了比原有Android权限模型更加细粒度的访问控制，对单一应用的同一敏感信息的访问控制可以服从不同的规则，但对同一开发者的第三方库的行为控制则服从相同的规则。此外，同一个开发者可能会开发多个第三方库，很多流行的第三方

库往往会被多个应用使用。如果用户可以根据开发者选择是否允许访问指定隐私信息，将有助于减少特定开发者在多个应用（或多个第三方库）中给用户隐私带来的潜在风险。

系统的整体设计思路如下：该系统需要用户自行制定访问控制策略，允许或者禁止特定开发者的代码访问特定资源。在敏感信息访问处，需要动态检查访问行为是否被用户的策略设置为“允许”，应用根据检查结果，执行相应的访问控制措施。

为实现以上访问控制措施，在应用插桩阶段，该系统需要对应用进行如下处理：对应用中使用特定API获取敏感信息的字节码进行静态识别、分析，插桩用于修改或清除相关敏感信息的代码段，并在应用中增加设置检查步骤。访问控制的实现方式如**图2**所示，即在请求访问敏感信息之前，利用同源策略进行授权检查。

为此，该系统需分块实现以下功能。

- 针对Android应用敏感信息使用行为进行识别及代码插桩，生成修改后的应用。
- 允许用户通过UI向系统中添加基于开发者的访问控制设置。
- 对系统设置项进行管理，并实现设置项与系统设置存储的互通。
- 针对识别出的隐私信息使用行为，在运行时检查用户设置，以进行访问控制。
- 在访问控制过程中，针对具体行为对隐私信息进行修改或访问截断。

该系统在结构上由以下几个模块构成。

- 一个静态插桩工具，用于应用的分析及修改。该工具基于Soot静态分析工具实现。
- 一个允许用户手动添加、查看、修改、删除相关设置的系统应用。
- 一个后台服务，用于监听来自应用的添加相关设置的请求，弹出窗口，允许用户做出选择。

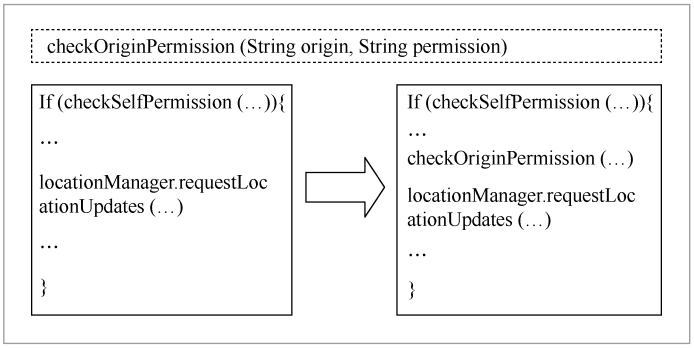


图 2 访问控制实现方式示意

- 一个用于管理用户相关设置及进行运行时检查的工具，对应用提供一个可供调用的、用于检查设置的API。
- 一个系统层面的隐私信息修改和拦截功能模块。例如，针对位置信息，需要根据已知位置生成随机偏差，得到模糊化处理后的位置。

该系统的整体结构如**图3**所示。

系统工作流程如下。

（1）应用修改

将Android应用程序包（Android application package，APK）文件及隐私信息修改/截断工具类统一导入Soot框架，并对给定应用进行特定敏感行为分析及代码插桩，以改变应用行为。

（2）将修改后的应用重新签名，并安装到Android手机上。

（3）用户设置

用户有两条途径对访问控制进行设置：一个是直接打开系统设置应用，在界面上手动添加或修改相关选项，允许（或者禁止）特定开发者执行指定的行为；另一个是在被修改的应用运行时，检查相关选项，如果未设置，通知后台服务，该服务随后会弹出一个对话框，内容为相关代码的开发者及敏感行为，提示用户设置相关选项，选择允许或禁止。

（4）运行时访问控制

在修改版的应用运行时，敏感行为代

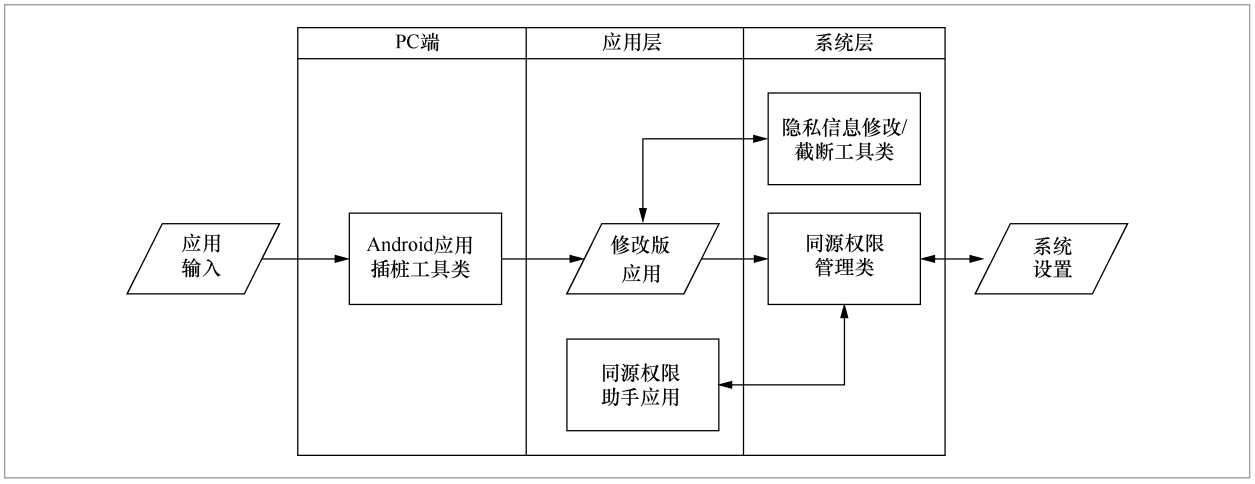


图 3 细粒度访问控制系统整体结构

码附近被插入的代码段会进行以下动作：调用被插桩的静态API，检查用户的相关设置，返回结果（允许/不允许），若未设置，则先请求设置；若结果为“不允许”，则调用相关工具类，对隐私信息做出模糊或清除处理，保护用户隐私。

3.2 应用分析及插桩

Soot是一个用于Java程序分析、优化及修改的开源框架，接受各类字节码、源码等多种输入，使用Jimple中间代码作为统一表示。Android应用APK文件中代码的存在形式是Dalvik字节码。利用Dexpler转换工具，Soot可以将输入的应用APK文件反编译成Jimple中间代码，并将分析/修改后的中间代码回编译为Dalvik字节码，输出一个新的APK文件。

本系统利用Soot分析框架，对Android应用APK文件进行静态分析，找出获取敏感信息（精确位置、联系人）的特定代码段，并在相应位置插桩与本系统功能相关的代码段，重新打包生成新的APK文件。利用Soot框架的Jimple分析功能，可以识别出应用中每个类及其中方法的具体实现，对于方法内部的语句（赋值、调用等）也能做到逐条识别。本系统中，重点关注了敏感API调用的方法名、方法传入的参数、返回值等信息以及方法所在包名（package name）中包含的开发者信息。此外，Soot框架还可以在Jimple代码层对应用进行修改，包括添加/删除语句、表达式，修改参数等。在本系统中表现为对应用执行行为的修改。

根据Soot框架的应用分析原理，本原型系统的应用插桩过程如**图4**所示。

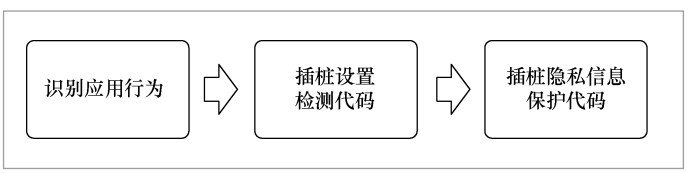


图 4 应用插桩过程

3.3 与访问控制相关的用户设置项

目前，该系统针对Android应用的两类敏感行为（获取位置信息、获取联系人）进行访问控制。该系统的用户设置项可以对开发者的不同访问行为，分别设置允许或禁止。用户设置项由数量不定的规则条目构

成，每个规则条目对应唯一一个开发者。

现阶段每条访问控制规则都由一个三元组定义：T=(DEV, OP1, OP2)。DEV代表开发者，是条目的查询关键字，采用字符串形式表示；“OPx”用整数表示，取值0,-1,-2，分别代表允许、不允许、该选项未设置。在工程实现上，依托Android系统的全局系统设置表（System.Settings.Global），以格式化字符串的形式存储全部条目，并用Android系统设置相关API，实现设置读写。

3.4 用户设置工具应用及系统服务

为方便用户查看、设置访问控制相关选项，本系统实现了一个用户设置工具。为了在某个开发者的访问控制选项未设置时，弹窗提醒用户进行设置，本系统在该用户设置工具上实现了后台服务。

用户通过管理应用，可以查看、添加、修改、删除各个条目。当修改后的应用检索不到某个开发者的访问控制选项时，启动后台服务。此时正在使用的应用会弹出提示窗口，询问用户是否允许开发者执行此类行为。用户只需设置一次，系统设置项中就会有相应记录。

3.5 设置检查与访问控制

在与敏感信息相关的代码段执行前，插入动态检查用户设置的代码段，并进行用户设置检查。其中，开发者信息是利用应用分析中获取的敏感API方法所在类的包名，在插桩时以常量字符串的形式插入的。针对位置信息敏感数据，通过设备位置改变时应用的回调方法将位置参数进行模糊处理。针对返回值，通过获取已知位置的方法进行模糊处理。针对联系人信息敏感数据，插入动态检查参数的代码段及如下条件语句：若确为联系人的统一资源标识符（uniform resource identifier, URI），则执行截断操作。

4 系统实现及实验结果

4.1 系统实现

在移动端，为方便应用调用以及用户设置，对Android 框架层进行修改，重新编译、安装，并将本系统的设置应用以系统应用的形式安装在测试用机上。在PC端，部署了Soot应用分析框架运行所需的环境，并实现了应用插桩相关工具类，可以对应用的APK文件进行插桩和重签名操作，在Linux系统或者cygwin环境下均可启动运行。将插桩处理并重签名的应用安装到测试机上，即可实现对该应用特定敏感行为的访问控制。

4.2 实验结果

测试阶段分两部分进行：第一部分验证具体的隐私信息处理措施及应用内的细粒度访问控制是否起到了作用；第二部分验证以开发者为基准的系统设计会对访问控制过程及结果起到何种作用。

在测试全过程中，为避免Android系统已有访问控制机制对测试结果的影响，对于插桩处理后的应用，启动前在“权限设置”里将其对“位置”“联系人”的访问权限全部设置为“允许”。在部署相应模块的开源Android系统上，对应用进行插桩、测试，结论如下。

- 通过对敏感API的判定、代码插桩，可以改变应用行为，实现访问控制。
- 通过用户策略的设置与运行时检查，可以根据敏感API调用代码的开发者，并选择性地进行访问控制，改变应用的相关行为。

4.2.1 细粒度访问控制功能测试

以与位置信息相关的应用为例，说明对位置信息修改的测试。利用开发的插桩工具插桩该位置应用，在插桩时设置位置信息偏移距离范围2 km<d<5 km，将插桩后的应用安装到设备。

清空本系统相关设置条目，启动应用，选择“我的位置（mylocation）”选项，屏幕上会出现Google地图，地图上会显示通过设备硬件（在测试环境下是通过Wi-Fi网络）的定位结果以及该位置应用根据该位置进行的标注。如果应用是未修改的版本，二者应该是同一位置。由于系统中未设置任何条目，已修改的应用在运行过程中先后弹出了2个窗口，如**图5**所示，分别要求用户对第三方开发者（com.google）和应用核心功能开发者（com.prime）的位置相关行为进行设置。

(a) 第三方开发者请求设置界面

(b) 应用核心功能请求设置界面

图 5 弹窗请求设置界面

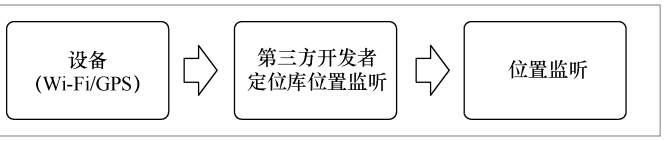


图 6 位置信息流动示意图

通过对该应用的原始版本进行反编译分析得知，在地图标注过程中，设备位置信息的流动如**图6**所示。

由以上分析可知，本应用中com.prime的地图标注点位置是通过监听com.google定位库的定位结果间接获得的。故在测试时，在实验组中以上两个开发者的“位置”设置项有且只有一个被设为“不允许”，以体现位置信息修改对应用行为的影响。

修改针对两个开发者的设置项，观察标注点情况，结果记录见**表1**。

表1中的“偏移量符合预期”，是指标注点相对于设备位置的偏移距离大于2 km且小于5 km。

为了与细粒度控制的实验组1、实验组2相比，对照组2将地图标记过程中所有涉及位置监听的开发者都设置为“不允许监听精确位置”，即进行较粗粒度的访问控制。标记结果偏移量可能超出预期（不小于5 km或不大于2 km），结合应用反编译分析结果，推测可能的原因是在地图标记过程中的2个监听行为获取到的位置信息均被加偏，且在位置信息传递过程中，两次偏移量被叠加，导致总的偏移量不在预期范围内，位置信息流动示意图如**图6**所示。

由以上测试可知，通过比较实验组1、实验组2和对照组1（无控制），系统中实现了“位置修改”动作，对该应用中监听器

表1 位置信息修改及细粒度控制测试结果

测试组别	行为类别	第三方开发者	应用核心功能	现象
实验组1	位置	允许	不允许	标注点与设备定位点不重合，偏移量符合预期
实验组2		不允许	允许	标注点与设备定位点不重合，偏移量符合预期
对照组1		允许	允许	标注点与设备定位点重合
对照组2		不允许	不允许	标注点与设备定位点不重合，有些标注点偏移量超出预期

监听到的位置信息进行了修改，并且对不同开发者的代码行为采用不同的访问控制设置。通过比较实验组1、实验组2与对照组2，结合反编译分析结果，初步验证了该系统可以对应用内部不同开发者的代码行为分别进行控制，即实现应用内部的细粒度控制。对com.freshideas.airindex、com.tweakersoft.aroundme等应用进行测试，验证了对位置信息的修改。

对联系人信息截断的测试，以与联系人相关的应用为例进行说明。利用插桩工具插桩该应用，并安装到设备。由于该应用在启动时会读取联系人列表，并显示在主界面，故测试的预设条件为：系统中已有至少一个联系人条目。进行2次测试，测试条件及结果见**表2**，测试界面如**图7**所示。

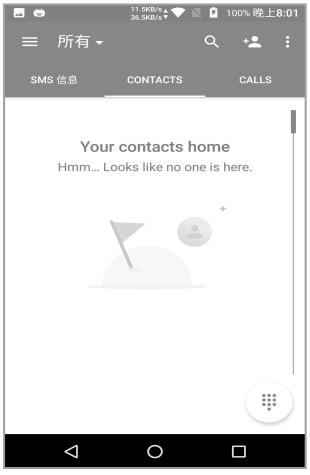
由此可知，插桩的截断代码阻止了应用通过调用查询函数获取联系人。

4.2.2 访问控制验证——跨应用测试

由于不同应用可能会使用相同开发者的第三方库执行相似行为，因此在设计上，只要首次执行时用户选择允许/禁止某开发者的代码执行某类行为，其后该开发者的代码在其他应用程序中再执行同样行为时，也会被相应允许/禁止，这是本文的系统与现有的访问控制系统最主要的区别，即其他相关的访问控制系统无法做到跨应用的访问控制。下面针对这一点进行测试，以验证本文的系统对同一开发者同类敏感

表2 联系人信息截断测试结果

测试次数	行为类别	联系人应用	现象
1	联系人	不允许	启动时不显示联系人
2		允许	启动时显示联系人



(a) 测试1结果

(b) 测试2结果

图7 联系人信息截断测试截图

行为的访问控制。

某社区信息应用以及某地图应用都是与位置信息相关的应用，在定位时都需要用到“com.google”开发的定位库。测试时，先清空系统中相关设置，将两个应用分别进行插桩，插桩时位置信息偏移距离范围均设置为2 km<d<5 km，并安装到设备，开启它们的位置访问权限，在本文设计的同源权限助手应用的设置中分别添加“允许”两个应用核心的开发者使用与位置相关的敏感行为的设置。

此时启动该社区信息应用，系统中后台服务弹出窗口，询问是否允许com.google访问精确位置，如**图8**所示。用户选择信任com.google的位置使用行为，点击了“是”。从运行结果看，该应用正确地返回了设备位置周边的公交车站等设施信息。

关闭该应用，再启动地图应用，选择“位置”选项，结果显示该应用的地图标注功能在地图上正确标注了设备位置，说明通过检测系统中已存在的用户相关设置，该应用中com.google的定位库使用了准确的位置信息。

回到同源权限助手应用，与用户的初始设置相反，此时将“com.google”设置为“不允许位置相关行为”，在两款应用中均观察到了标注或搜索偏差。

由此可见，在本文提出的访问控制系统中，基于开发者的访问控制机制可以使用户以同一设置项控制不同应用中同一开发者的某一类访问行为，信任粒度为开发者。这种控制机制的应用意义在于，当多个应用中包含同一开发者的第三方库（如广告库）时，用户可以通过单次设置来控制这些第三方库的敏感行为，免于对每个应用单独设置。事实上，很多第三方库会被多种应用使用。在第三方库访问控制中，基于开发者的访问控制设置能起到打破应用边界、扩展控制范围的作用。

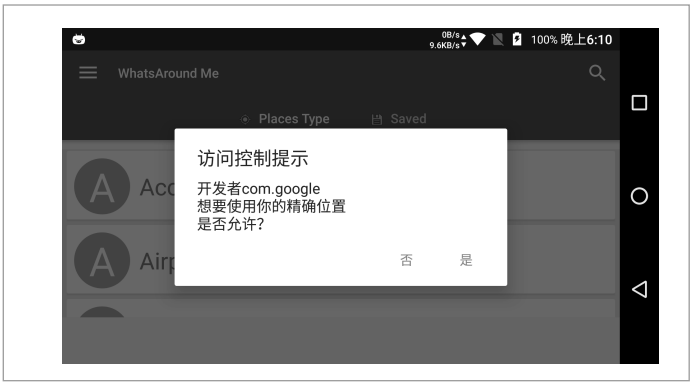


图 8 后台服务向用户请求设置

5 系统的限制

由于需要对应用进行插桩，因此本文提出的访问控制系统存在一些限制。首先，由于目前移动应用生态中重打包现象严重，一些应用出于安全考虑会检查代码是否被修改，因此本文提出的自动化插桩会受限。其次，应用的加壳和混淆会影响对敏感API的分析和插桩。此外，在对应用插桩时，需要准确识别第三方库代码，尽管可以利用学术界提供的工具来做，但是仍然可能存在检测不准确或者不全的情况。

6 结束语

本文借鉴同源策略，提出并设计了一个以应用来源（开发者）为控制粒度的Android应用细粒度访问控制原型系统。提出的方法打破了应用之间的界限，并将粒度细化到开发者，从单个应用角度看，进行了比原有Android权限模型更加细粒度的访问控制。本文对主流Android应用进行插桩及测试，结果表明本系统能够起到允许或限制特定开发者执行特定敏

感行为的作用，这也验证了对Android应用基于代码来源的细粒度访问控制方式的可行性。

参考文献:

[1] WANG H Y, LIU Z, LIANG J Y, et al. Beyond Google play: a large-scale comparative study of Chinese Android App markets[C]// 2018 ACM Internet Measurement Conference, October 31-November 2, Boston, USA. New York: ACM Press, 2018: 293-307.

[2] MICHAEL C G, ZHOU W, JIANG X X, et al. Unsafe exposure analysis of mobile in-App advertisements[C]// The 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks, April 16-18, 2012, Tucson, USA. New York: ACM Press, 2012: 101-112.

[3] RYAN S, CLINT G, JON C, et al. Investigating user privacy in Android ad libraries[C]// The 3rd Workshop on Mobile Security Technologies, May 24, 2012, San Francisco, USA. [S.l:s.n.], 2012.

[4] WANG H Y, GUO Y, MA Z, et al. WuKong: a scalable and accurate two-phase approach to android App clone detection[C]// 2015 International Symposium on Software Testing and Analysis, July 14-17, 2015, Baltimore, USA. New York: ACM Press, 2015: 71-78.

[5] LIU B, JIN H X, RAMESH G. Efficient privilege de-escalation for ad libraries in mobile apps[C]// The 13th Annual International Conference on Mobile Systems, Applications, and Services, May 19-22, 2015, Florence, Italy. New York: ACM Press, 2015: 89-103.

[6] SHASHI S, MICHAEL D, WALLACH D S. AdSplit: separating smartphone advertising from applications[C]// The 21st USENIX Conference on Security Symposium, August 8-10, 2012, Bellevue, USA. Berkeley: USENIX Association, 2012: 28.

[7] PEARCE P, FELT A P, NUNEZ G, et al. AdDroid: privilege separation for applications and advertisers in Android[C]// The 7th ACM Symposium on Information, Computer and Communications Security. May 2-4, 2012, Seoul, Korea. New York: ACM Press, 2012: 71-72.

[8] ROESNER F, KOHNO T. Securing embedded user interfaces: Android and beyond[C]// The 22nd Security Symposium, August 14-16, 2013, Washington, USA. Berkeley: USENIX Association, 2013: 97-112.

[9] FU J J, ZHOU Y F, LIU H, et al. Perman: fine-grained permission management for Android applications[C]// The 28th International Symposium on Software Reliability Engineering, October 23-27, 2017, Toulouse, France. Piscataway: IEEE Press, 2017: 250-259.

[10] FU J J, ZHOU Y F, WANG X. Component-based permission management of Android applications[J]. Software: Practice and Experience, 2019, 49(3).

[11] 胡冰惠. 基于细粒度动态分析的Android平台第三方库隐私泄露分析[D]. 北京: 北京交通大学, 2018.
HU B H. Privacy disclosure analysis of third-party library on Android platform based on free grained dynamic analysis[D]. Beijing: Beijing Jiaotong University, 2018.

[12] DIAMANTARIS M, PAPADOPOULOS E P, MARKATOS E P, et al. REAPER: real-time App analysis for augmenting the Android permission system[C]// The 9th ACM Conference on Data and Application Security and Privacy, March 19-21, 2018, Tempe, USA. New York: ACM Press, 2018: 37-48.

[13] WANG H Y, HONG J, GUO Y. Using text mining to infer the purpose of permission use in mobile Apps[C]// The

2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, September 7-11, 2015, Osaka, Japan. New York: ACM Press, 2015: 1107-1118.

[14] WANG H Y, LI Y C, GUO Y, et al. Understanding the purpose of permission use in mobile Apps[J]. ACM Transactions on Information Systems, 2017, 35(4): 1-40.

[15] ARZT S, RASTHOFER S, FRITZ C, et al. FlowDroid: precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for Android Apps[J]. ACM SIGPLAN Notices, 2014, 49(6): 259-269.

[16] YANG W, XIAO X, ANDOW B, et al. Appcontext: differentiating malicious and benign mobile App behaviors using context[C]// The 37th International Conference on Software Engineering, May 16-24, 2015, Florence, Italy. Piscataway: IEEE Press, 2015: 303-313.

[17] YUVRAJ A, MALCOLM H. Protect my privacy: detecting and mitigating privacy leaks on iOS devices using crowdsourcing[C]// The 11th Annual International Conference on Mobile Systems, Applications, and Services, June 25-28, 2013, Taipei, China. New York: ACM Press, 2013: 97-110.

[18] ENCK W, ONGTANG M, MCDANIEL P. On lightweight mobile phone application certification[C]// The 16th ACM Conference on Computer and Communications Security, November 9-13, 2009, Chicago, USA. New York: ACM Press, 2009: 235-245.

[19] HAMMAD M, BAGHERI, H, MALEK S. Determination and enforcement of least-privilege architecture in Android[C]// 2017 IEEE International Conference on Software Architecture, April 3-7, 2017, Gothenburg, Sweden. Amsterdam: Elsevier, 2019: 83-100,149.

[20] WANG H, GUO Y, TANG Z, et al. Reevaluating Android permission gaps with static and dynamic analysis[C]// 2015 IEEE Global Communications Conference, December 6-10, 2015, San Diego, USA. Piscataway: IEEE Press, 2015: 1-6.

作者简介

卢文雄(1992-),男,北京邮电大学计算机学院硕士生,主要研究方向为移动计算。

王浩宇(1991-),男,北京邮电大学计算机学院副教授,主要研究方向为软件安全和程序分析。

收稿日期: 2019-11-14

基金项目: 国家自然科学基金资助项目(No.61702045)

Foundation Item: The National Natural Science Foundation of China (No.61702045)