Android 权限滥用检测系统①

徐 尉[1,2], 吴敬征[1]

[1](中国科学院软件研究所 基础软件国家工程研究中心, 北京 100190)
[2](中国科学院大学, 北京 100190)

摘 要: 凭借开源策略及精准的市场定位, Android 系统占据了智能移动终端操作系统 84.2%的市场份额. 然而, 其开放的权限机制带来更多使用者和开发者的同时, 也带来了相应的安全问题. 中国互联网络信息中心调查数据显示, 仅有 44.4%的用户在下载安装 Android 应用的过程中会仔细查看授权说明, 而大部分人存在着盲目授权的行为. 对于应用开发者来说, 由于缺乏安全开发监管, 缺乏权限申请相关代码规范, 权限滥用问题在 Android 应用开发中普遍存在, 严重影响了代码的规范和质量. 其次, 用户的盲目授权和软件开发者的权限申请滥用也是用户信息泄露的主要原因, 存在严重的安全风险. 针对以上问题, 本文在现有的权限检测方案基础上, 设计和实现了一套新的权限滥用检测系统 PACS(Pemission Abuse Checking System). PACS 针对 1077 个应用进行分析, 发现 812 个应用存在权限滥用问题, 约占全部应用的 75.4%, 同时对实验结果进行抽样验证, 证明了 PACS 的权限检测结果的准确性和有效性.

关键词: Android; 权限滥用检测; 频繁项集挖掘; 分类

Abused Detection System of Android

XU Wei[1,2], WU Jing-Zheng[1]

[1](Institute of Software, Chinese Academy of Sciences, Beijing 100190, China)
[2](University of Chinese Academy of Science, Beijing 100190, China)

Abstract: Android is the mobile operating system which has the highest market share of 84.2%. Its open access mechanism not only brings more users and developers, but also a lot of security issues. According to the survey by China Internet Network Information Center, only 44.4% of users will view the authorization instructions carefully in the process of downloading and installing an application. Most people have the risk of blind authorization. For software developers, due to the lack of strong supervision and proper permission specifications, application authority abuse in the procedure of Android application development has been widespread, which seriously affected the code and quality specifications. Besides, the user's blind authorization and the software developer's permission application abuse has become the main reason for the users' information leakage. In this paper, based on the existing permission detection scheme, it designs and implements an abuse of authority detection system named PACS. Test results show that 812 applications abuses permission among 1077 applications, which account for about 75.4%. Meanwhile the sampling of test results proves the accuracy and validity of the PACS's results.

Key words: Android; abused permission detection; frequent itemsets mining; classify

1 引言

随着移动互联网时代的到来, 智能终端的普及率不断提高. 越来越多的移动应用给用户带来便利. 截止到 2015 年 11 月, Google Play Store 上有超过 180 万个应用, 而与之相关的安全问题也引起了相关研究者的关注.

在智能终端中, Android 系统占有了 84.2%的市场份额. Android 系统权限限制机制要求, 如果应用需要

① 基金项目:国家自然科学基金(61432001,91218302,61303057)
收稿时间:2016-01-24;收到修改稿时间:2016-03-08 [doi:10.15888/j.cnki.csa.005362]

访问某个系统 API, 就需要拥有相应的权限(Permission), 拥有相应的权限才能够访问相应的 API. 所有应用都在其配置文件 AndroidManifest.xml 中进行权限声明. 当应用程序安装时, Android 系统会提醒用户应用所申请的权限, 由用户来选择是否安装该应用. Android 系统把权限审核的决定权交给用户, 是一种相对灵活的安全防护策略. 分析安卓应用的配置文件 AndroidManifest.xml 中申明的权限可以发现, 很多不同类别、不同功能的应用申请了读取短信、通讯录、相册等涉及用户个人隐私信息的权限. 中国互联网络信息中心调查数据显示, 仅有 44.4%的用户在下载安装 Android 应用的过程中会仔细查看授权说明, 大部分用户不会仔细查看应用的授权说明, 导致盲目授权的安全风险.

Android 权限滥用主要会导致两个方面的问题. 一方面, 权限滥用会造成安全隐患. 针对 Android 平台的攻击主要针对用户的隐私信息. 由应用程序权限滥用构造合谋攻击导致的隐私泄露案例越来越多. Soundcomber[1]就是一个典型的攻击案例. 合谋权限攻击的核心思想是, 程序 A 有某个特定的执行权限 P, 另一个程序 B 未申请 P 权限, 但是 B 可以通过一系列操作借用 A 的动作执行 P 权限, 从而达到隐私泄露、攻击的目的. 设定上述中的程序 A 是一个权限申请滥用的正常软件, 而作为恶意软件的 B 则无需在自己的代码中申请任何的权限, 大大减小了 B 被杀毒软件检测出来的可能性, 同时又利用 A 的权限达到了恶意行为, 如上传用户隐私、发送短信等.

另一方面, 权限滥用违反了应用开发权限最小化原则, 严重影响应用的代码质量.

针对 Android 权限滥用的问题, 本文设计和实现了一套基于评论和简介数据的频繁项集挖掘的权限滥用检测系统 PACS. PACS 将 Android 市场上应用软件的简介和评论信息作为新增的检测条件, 利用支持向量机算法对应用进行分类. 同时使用 Apriori 算法[2]挖掘应用类别内部的权限之间关联性的权限频繁模式, 构造基于类别的权限关系特征库, 最终来检测未知的应用的权限滥用情况. 试验结果表明, 针对 1077 个应用进行分析, 发现 812 个应用存在权限滥用问题.

本文的主要贡献和创新点如下:

(1) 提出了一种基于评论和简介数据的 apk 自动归类方案. 分类系统使得 PACS 能够在确定的类别下对权限滥用的进行分析, 提高了权限滥用检测的准确度.

(2) 将 Apriori 算法应用到挖掘同类别应用的权限特征库, 有效地挖掘出各个类别下的极大频繁权限项集, 从而检测未知应用的权限滥用问题.

2 相关工作

Android 系统给开发者提供了大量系统 API, 开发者向系统申请权限后可以调用相应 API 进行开发. Google 在官方的开发文档中给开发者提供了系统 API 使用说明. 例如, 开发者需要调用 sendTextMessage() 这样的系统 API 来完成短信发送功能开发, 则需要申请权限 android.permission.SEND_SMS. 但是, 开发文档对于大多数 API 所需要的权限没有给出明确的解释, 这也是导致权限滥用情况发生的一个主要原因. 因此, 很多研究人员通过研究应用权限和 API 的映射关系, 以是否调用了该权限相对应的正确的 API 作为依据, 来判断应用是否存在权限滥用的问题.

AP Felt 等人[3]提出了一种通过权限和 API 的映射关系, 检测应用权限滥用的方案, 并实现了工具 Stowaway. 他们对 Android 2.2.1 版本中的 API 进行分析, 统计出需要申请权限才能调用的 API 有 1259 个. 然而, 在对应版本的 Android 文档中只定义了其中 78 个 API 的权限对应情况. 他们从 940 个应用中随机挑选 40 个应用在 Stowaway 上运行, Stowaway 将其中 18 个应用定义为权限滥用, 对结果进行了进一步的人工分析, 认为缺乏 API 到权限的映射是造成 Stowaway 误判的主要原因. 最后, 指出权限滥用问题的根源在于 Android API 文档的一些错误和不完整.

KWY Au 等人[4]对 Android 2.2 到 4.0 四个版本的源代码进行分析, 给出了较为完整的 API 和权限之间的映射, 并提出了分析权限滥用问题的 PScout 静态分析工具. 他们使用 PScout 对 1260 个应用进行了权限分析, 发现其中 543 个应用有权限滥用问题.

Moonsamy V 等人[5]解析出应用在配置文件 AndroidManifest.xml 中声明的权限, 采用聚类的方法, 对正常应用和恶意应用的权限模式进行挖掘, 为正常应用集和恶意应用集分别建立权限特征库.

虽然大部分的研究工作都在研究 Android API 和权限的映射关系, 但是目前的 Android API 和权限映射仍然不够完善. 例如, PScout 中所得到的 API 权限映射关系是目前最为完整的, PScout 对 Android 4.1.1 中的

API 权限映射进行分析, 共找到了 101 个权限与 API 有映射关系, 但 Android 4.1.1 系统中的所有权限个数为 180 个. 而 Android 系统仍有 92 个权限在 PScout 中没有得到对应, 权限覆盖率相对较低.

其次, 这些研究都基于一种假设, 即只要在代码中出现了相关 API 的调用, 则该权限的申请是正常的. 然而, 分析一些软件的源码可以看到, 由于目前开发人员水平的参差不齐, 在代码中应用了不该使用或者没有必要使用的 API, 同时申请了该 API 对应的权限, 也造成了权限滥用问题.

本文提出的基于权限项集的挖掘的权限滥用检测系统 PACS, 避免了使用 API 和权限的映射, 从同类应用的权限申请标准出发, 挖掘该类别下的应用的权限申请模式, 检测应用的权限滥用问题.

3 系统概述

PACS 针对 Android 权限滥用问题, 实现了对未知应用的权限信息的检测, 能够检测出应用是否存在权限滥用的问题, 并给出相关判断说明. PACS 系统根据功能总共可以分为训练和测试两大模块. PACS 系统架构图如图 1 所示.

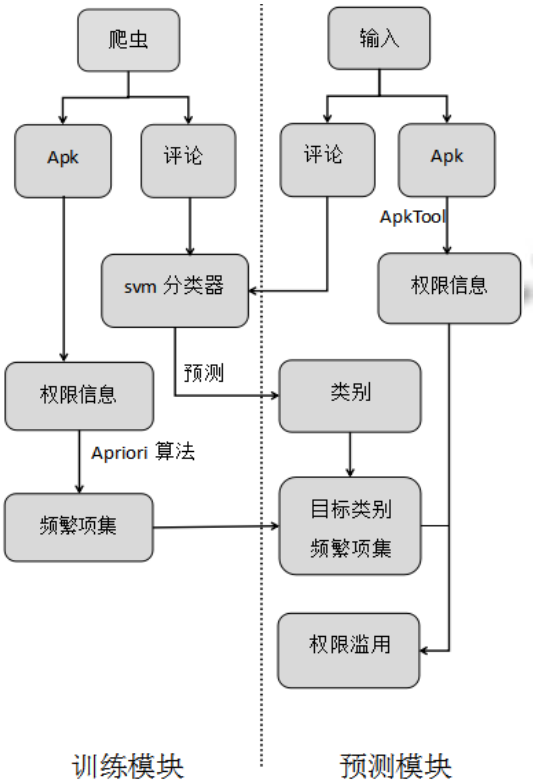

图 1 PACS 系统架构图

训练模块: 包含爬虫模块、权限频繁项集挖掘模块和语料分类器模块. 爬虫模块是所有数据的来源, 从coolapk官网[6]抓取应用包和该应用的简介和评论数据. 频繁项集模块是通过解析 apk 包获取该应用的权限信息, 然后将同一类别下的所有应用的权限集合输入 Apriori 算法进行迭代, 计算得出极大频繁权限项集. svm 分类器模块是通过对应用的简介和评论数据进行处理, 建立文本分类器, 将数据输入分类器进行训练, 调节参数, 得到最优模型.

预测模块: 包含分类模块和权限检测模块. 分类模块是依据未知软件的简介和评论数据对软件进行分类预测, 输出类别标签. 权限检测模块是结合分类模块输出的类别标签, 读取未知应用目标类别下的极大频繁权限项集. 同时, 将应用的 apk 文件进行反编译, 获取 AndroidManifest.xml 中的权限信息, 与极大频繁项集比较, 判断是否有权限滥用现象.

4 各模块的算法设计与实现

4.1 爬虫模块

系统的爬虫进程首先获取到初始化的 URL 队列, 即 coolapk 官网的各个类别下的 URL 链接集合, 然后对该队列进行遍历, 对每个 URL 链接对应的页面进行解析, 从中抽取到符合条件的 URL, 把新的 URL 放入到队列中, 直到满足抓取的停止条件. 对于具体应用的网页, 下载应用的 apk 包, 并抽取其中应用的评分、名称、评论、简介等内容. 爬虫模块处理流程如图 2 所示.

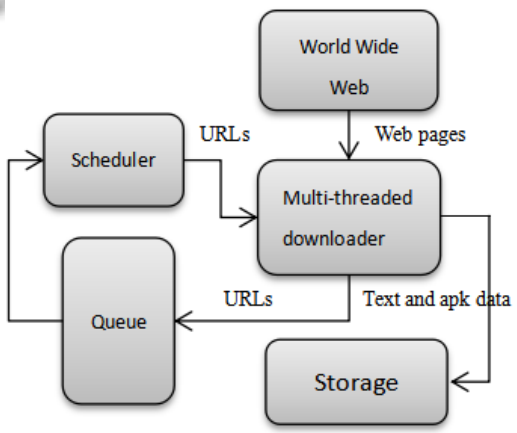

图 2 PACS 爬虫模块流程图

PACS 爬虫模块研发基于 Python 的 BeautifulSoup

开源包[7]. BeautifulSoup 是一个 Python 写的 HTML 解析库，它不依赖于其它的 Python 库文件. BeautifulSoup 具有小巧，快速的优点，它能超高速解析 HTML 从而能快速便捷的获取网页中的文本内容.

4.1.1 apk 数据

爬虫模块抓取 10771 个 apk 应用样本，依据应用的功能用途被分成了 16 个类别. 样本来源为 coolapk 市场. Apk 数据分类类别如表 1 所示.

表 1　apk 数据分类信息

类别 id	应用分类	应用个数
001	系统工具	1288
002	桌面插件	554
003	主题美化	1853
004	社交聊天	535
005	资讯阅读	847
006	通讯网络	791
007	影音娱乐	1091
008	摄影拍照	464
009	生活服务	851
010	实用工具	1356
011	文档商务	245
012	金融财经	239
013	运动健康	124
014	学习教育	258
015	旅行交通	145
016	购物	130

4.1.2 应用详情数据

爬虫模块同时抓取 10771 个样本的应用详情. 包含软件名称、APK 名称、最新版本、支持 ROM 版本、界面语言、软件大小、更新日期、开发者等信息. 应用详情示例数据如表 2 所示.

表 2　样本的应用详情示例

软件名称	WeLoop
APK 名称	com.yf.smart.weloopx.dist
最新版本	3.8.4
支持 ROM	4.3 及更高版本
界面语言	简体中文
软件大小	10.81M
更新日期	20151228
开发者	WeLoo p

4.1.3 应用简介和评论数据

爬虫模块完成每个应用的简介和评论数据的抓取，同时依据类别名称和类别 ID 的映射关系，记录保存每个应用评论数据的类别 ID，存入数据库. 数据库中的简介和评论示例数据如表 3 所示.

表 3　应用的简介和评论数据示例

类别	名称	简介	评论
005	BuzzFeed	SmartNews 是一款新闻分类阅读软件，可以根据分类自由滑动切换.	-切换地区有日文新闻 -滑动动画流畅 -一款有情怀的新闻阅读软件

4.2 反编译 apk 包

安卓应用程序通常为 apk 格式，主要包括五部分:

(1) META-INF 目录，用来存放签名信息，验证 apk 是否完整;

(2) res 目录，用来存放资源文件;

(3) AndroidManifest.xml 文件，描述应用的名字、权限等信息;

(4) classes.dex 文件，是 Java 源码编译后生成的 Java 字节码文件，可以根据这个文件对 apk 进行反编译;

(5) resources.arsc 文件，是编译后的二进制资源文件.

由于 apk 文件中 AndroidManifest.xml 是经过编译的，需要对其进行反编译才能进行分析. 目前常用的反编译工具是 ApkTool，能将一个 apk 文件分解出 smali 代码和里面的 xml 文件. 反编译后的 AndroidManifest.xml 文件部分内容如图 3 所示.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.READ_LOGS" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.VIBRATE" />
```

图 3　AndroidManifest.xml 部分内容

对反编译后的 AndroidManifest.xml 文件进行分析，可以发现其主要包括以下几个部分:

(1) <uses-permission> 标签，请求安全授权; <application> 标签，表示每一个应用程序的组件及其属性;

(2) <activity> 标签，用于用户交互机制;

(3) <receiver> 标签，接收数据变化等; <service> 标签，在后台可以随时运行的组件.

PACS 主要对<uses-permission> 标签包含的参数进行分析，依据规则抽取出该 apk 包在 AndroidManifest.xml 中声明的权限信息. 表 4 列举了

部分权限名称及其解释.

表 4 部分权限名称及其解释

(权限名称均省略了前缀 android.permission.)

名称	功能
ACCESS_NETWORK_STATE	获取网络状态
ACCESS_WIFI_STATE	获取 WiFi 状态
INTERNET	访问网络
BLUETOOTH	使用蓝牙
READ_CONTACTS	读取联系人
CALL_PHONE	拨打电话

4.3 频繁项集挖掘

频繁模式, 指挖掘频繁出现在数据集中的模式. 频繁模式 Apriori 算法[2]是 AgrawalR 等人提出的为布尔关联规则挖掘频繁项集的原创性算法. 本文基于频繁模式 Apriori 算法, 挖掘同类应用申请的权限之间的关联, 分别构建权限关系特征库, 进行未知应用权限检测.

以下构造一个实例说明具体的算法过程. 假设某一类别下有四个应用程序, 每个应用程序列出了所申请的权限数据, 如表 5 中所示. 支持度 support=P(AB), 指的是事件 A 和事件 B 同时发生的概率. 为了简化计算, 本例中直接假设最小支持度 min_sup 为 50%. 在每次迭代中算法都将产生一些候选集, 统计计算每个候选集 I 出现的频率, 如果项集 I 的相对支持度满足预定义的最小支持度阈值, 则 I 是频繁项集.

表 5 某类别下应用部分权限数据库

(权限名称均省略了前缀 android.permission.)

TID	权限项集
Apk.01	CALL_PHONE,READ_CONTACTS, INTE- RNET, WAKE_LOCK
Apk.02	VIBRATE, READ_CONTACTS, WAKE_LOCK
Apk.03	INTERNET, WAKE_LOCK, CALL_PHONE
Apk.04	INTERNET,WAKE_LOCK, READ_CONTAC- TS, CALL_PHONE

表 6 中描述了算法所有迭代步骤. 第一次迭代中, 有 4 个相对支持度不小于 50%的项集, 这 4 个项集即为 1-项集中的频繁项集. 在第二次迭代中, 由 1-项频繁项集两两组合, 组成了 6 个支持度大于 min_sup 的 2-项频繁项集. 第三次迭代有三个支持度不小于 min_sup 的 3-项集, 都选取为频繁项集. 第四次迭代只有 1 组项集满足最小支持度阈值, 这一组权限项集即是 4-项频繁项集, 且不存在频繁项集 Y, 使得 {CALL_PHONE, READ_CONTACTS, INTERNET, WAKE_LOCK}包含于 Y, 则其为极大频繁权限项集.

表 6 某类别下应用部分权限数据库

(权限名称均省略了前缀 android.permission.)

迭代	权限项集	计数	支持度(%)	频繁集
1	CALL_PHONE	3	75	是
	READ_CONTACTS	3	75	是
	VIBRATE	1	25	否
	INTERNET	3	75	是
	WAKE_LOCK	4	100	是
2	CALL_PHONE, READ- _CONTACTS	2	50	是
	CALL_PHONE, INTER- NET	1	25	否
	CALL_PHONE, WAKE- _LOCK	3	75	是
	READ_CONTACTS, IN- TERNET	2	50	是
	READ_CONTACTS,WA- KE_LOCK	2	50	是
	INTERNET, WAKE_LO- CK	3	75	是
3	CALL_PHONE,READ_C- ONTACTS,INTERNET	2	50	是
	CALL_PHONE,READ_C- ONTACTS,WAKE_LOCK	2	50	是
4	CALL_PHONE,READ_C- ONTACTS,INTERNET, WAKE_LOCK	2	50	是

使用 Apriori 算法迭代计算权限的频繁模式, 需要保证训练集的可靠性. 由于训练集中的 apk 中的申请的权限规范与否一定程度上影响了频繁权限项集的准确度, 所以需要依据 apk 的评分数据做了一个训练集的筛选工作. Coolapk 网站上对软件的评分等级为 1-5, 根据对训练集和测试集大小的预期, 在选择训练集时选择了评分达到 4 及 4 以上的 apk.

4.4 应用分类模块

不同类别下的应用所申请的权限往往相差很大, 而同一类别下的应用所需权限差异性较小. 如在类别“摄影拍照”下, 通常应用都会申请的权限包含读取外部存储、拍照权限、录音、使用振动、获取网络状态等, 而访问网络、读取电话状态、拨打电话、接收短信、读取短信内容等权限的申请多是集中在“通讯网

络”类别的应用. 由此可见, 在检测应用是否出现了滥用权限问题时, 对未知应用进行分类, 依据该类别下建立的权限特征库来进行校验, 是一个科学可行的方法.

本文采用 tfidf 的词频统计方法, 将文本用空间向量表示, 词频作为权重信息, 输入支持向量机模型(Libsvm 工具), 训练求取最优参数.

5 系统实现与验证

5.1 系统实现

PACS 系统全部使用 Python 语言设计和实现. 其中爬虫模块用到了 BeautifulSoup 包对 HTML 进行解析. apk 反编译模块用到了开源项目 aapt 对应用进行反编译, 提取权限信息. PACS 系统界面使用 Python 的 Tkinter 开发, 如图 4 所示.

图 4　PACS 系统界面

5.2 实验分析

本文数据均抓取自 coolapk 网站. 包含 10771 个应用的 apk 包, 应用详情、得分、简介和评论数据. 10771 个应用共分为了 16 个类别, 从每个类别中随机抽取出 1/10 的数据, 作为测试数据, 来验证系统的实验效果. 共随机抽取了 1077 个实验样本, 输入 PACS 系统进行权限滥用检测. 检测结果表明, 共 812 个应用存在权限滥用的现象, 约占全部应用的 75.4%.

2011 年, AP Felt 等人[3]使用 Stowaway 工具分析 940 个应用, 检测出约有 34.4%的应用有权限滥用的情况. 2012 年, KWY Au 等人[4]使用 PScout 对 1260 个应用进行分析, 发现 543 个应用存在权限滥用情况, 约占 43.1%. 2014 年, Bai Xiao-long 等人[8]使用 PTailor 对采集到的 1246 个应用进行剪裁, 发现其中有 811 个应用存在权限滥用的现象, 约占 65.1%. A Bartel 等人[9]通过分析不同版本的应用程序, 给出了权限滥用的应用的比例会随着时间不断的增加的结论. 通过图 5 中的数据比较可以看出, PACS 检测的权限滥用的应用的比例比起之前的工具的结果来说呈现明显上升趋势, 这也符合 A Bartel 等人[9] 的结论.

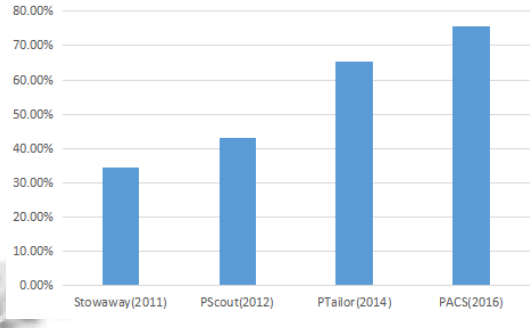

图 5　权限滥用检测结果比较

5.3 实验验证

随机选取检测出了权限滥用的应用, 反编译到 java 代码, 查找 API 和权限的映射关系, 分析代码和需求, 人工确认申请的权限是否属于滥用行为.

从 PACS 输出的权限滥用的应用集合中, 随机挑选 5 个 apk, 如表 7 所示.

表 7　部分 apk 检测结果

apk 名称	权限滥用个数
dopool.player.apk	5
cn.ersansan.kichikumoji.apk	4
cn.am321.android.am321.apk	6
cn.etouch.ecalendar.apk	20
com.aikan.apk	6

以 dopool.player.apk 为例, 列出人工检测过程及检测结论.

5.3.1 反编译生成 java 源码

Apk 文件是一个应用发布包, 对其进行反编译获得 java 源码.

首先, 使用 ApkTool 反编译 apk 包. 在反编译后的文件夹中, 可以找到其中的 class.dex 文件, 这个文件即是 java 文件经过编译再通过 dx 工具打包而成. 这里引入开源工具 dex2jar 对 class.dex 进行反编译, 生成 class-dex2jar.jar. 然后需要将 class 文件反编译成 java 源代码, 这边使用开源工具 jd-gui. 使用 jd-gui 打开 class-dex2jar.jar 可得到 java 源代码. 如图 6 所示.

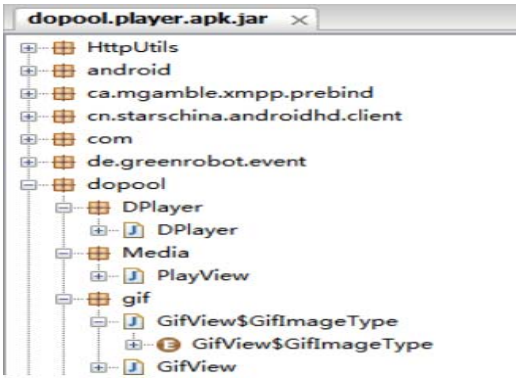

图 6 反编译出的 dopool 的源代码

5.3.2 PACS 系统进行权限滥用检测

运行 PACS 系统对 apk 进行检测. 检测结果如表 8 所示.

表 8 dopool 在 PACS 中的运行结果
(权限均省略了前缀 android.permission.)

检测结果	滥用权限列表
滥用	SEND_SMS RECEIVE_USER_PRESENT READ_CALENDAR CALL_PHONE WRITE_CALENDAR

5.3.3 PACS 运行结果分析

首先, 找出滥用权限列表中的权限和其对应的 API 的映射关系. 表 9 中列出了 dopool 的部分滥用权限列表中的权限和 API 的映射关系.

表 9 部分权限和 API 的映射关系

权限名称	Android API
SEND_SMS	sendTextMessage
	endMultipartTextMessages
	sendDataMessage
CALL_PHONE	enforceCallPermission
	endCall
	call
	dialRecipient

在 dopool 源代码中查找, 没有发现表 9 中所示的任何一个 API 调用. 由此可以判断, SEND_SMS 和 CALL_PHONE 这两个权限确实属于权限滥用问题.

另外两个权限 WRITE_CALENDA 和 READ_CALENDAR, 可以看到在源码中确实使用了这两个权限所对应的 API, 但是由于其他的同类视频软件中, 很少出现这两个权限, 所以 PACS 也将这两个权限的申请判定为了权限滥用. 根据 google 提供的安卓官网的开发文档可以了解到, 正常情况下, 要想读取或写入日历数据, 应用的清单文件必须包括用户权限中所述的适当权限. 为简化常见操作的执行, 日历提供程序提供了一组 Intent 对象, 日历 Intent 对象中对这些 Intent 做了说明. 这些 Intent 对象会将用户转到日历应用, 执行插入事件、查看事件和编辑事件操作. 用户与日历应用交互, 然后返回原来的应用. 因此, 应用不需要请求权限, 也不需要提供用于查看事件或创建事件的用户界面. 由此可以看出, 作为一个视频应用, 由于开发的并不是完备的日历应用或同步适配器, 是不需要申 WRITE_CALENDA 和 READ_CALENDAR 这两个权限的. 因此, 这两个权限也可以判定属于权限滥用问题.

6 结语

针对 Android 应用的权限滥用问题, 本文设计和实现了一套基于频繁项集挖掘的权限滥用检测系统 PACS. 本文的主要贡献在于提出了一种基于评论和简介数据的 apk 自动归类方案. 分类系统使得 PACS 能够在确定的类别下对权限滥用的进行分析. 同时, 将 Apriori 算法应用到挖掘同类别应用的权限特征库, 有效地挖掘出各个类别下的极大频繁权限项集, 提高检测准确率.

参考文献

1 Schlegel R, Zhang K, Zhou X, et al. Soundcomber: A stealthy and context-aware sound trojan for smartphones. NDSS. 2011, 11: 17–33.

2 Agrawal R, Mannila H, Srikant R, et al. Fast discovery of association rules. Advances in knowledge discovery and data mining, 1996, 12(1): 307–328.

3 Felt AP, Chin E, Hanna S, et al. Android permissions demystified. Proc. of the 18th ACM Conference on Computer and Communications Security. ACM. 2011. 627–638.

4 Au KWY, Zhou YF, Huang Z, et al. Pscout: Analyzing the Android permission specification. Proc. of the 2012 ACM Conference on Computer and Communications Security. ACM. 2012. 217–228.

5 Moonsamy V, Rong J, Liu S. Mining permission patterns for contrasting clean and malicious android applications. Future

Generation Computer Systems, 2014, 36: 122–132.

6 酷安市场. http://www.coolapk.com/.

7 BeautifulSoup. http://www.crummy.com/software/BeautifulSoup/.

8 白小龙. Android 应用程序权限自动裁剪系统.计算机工程与科学,2014,36(11):2074–2086.

9 Bartel A, Klein J, Le Traon Y, et al. Automatically securing permission-based software by reducing the attack surface: An application to Android. Proc. of the 27th IEEE/ACM International Conference on Automated Software Engineering. ACM. 2012. 274–277.

10 杨欢,张玉清,胡予濮,等.基于权限频繁模式挖掘算法的 Android 恶意应用检测方法.通信学报,2013,34(1): 106–115.

11 Nauman M, Khan S, Zhang X. Apex: Extending Android permission model and enforcement with user-defined runtime constraints. Proc. of the 5th ACM Symposium on Information, Computer and Communications Security. ACM. 2010. 328–332.

12 Wei X, Gomez L, Neamtiu I, et al. Permission evolution in the Android ecosystem. Proc. of the 28th Annual Computer Security Applications Conference. ACM. 2012. 31–40.

13 李寅,范明钰,王光卫.基于反编译的 Android 平台恶意代码静态分析.计算机系统应用,2012,(2111):187–189.

14 Nauman M, Khan S, Zhang X. Apex: Extending Android permission model and enforcement with user-defined runtime constraints. Proc. of the 5th ACM Symposium on Information, Computer and Communications Security. ACM. 2010. 328–332.

15 Shin W, Kiyomoto S, Fukushima K, et al. A formal model to analyze the permission authorization and enforcement in the Android framework. 2010 IEEE Second International Conference on Social Computing (SocialCom). IEEE. 2010. 944–951.

16 Zhou Y, Zhang X, Jiang X, et al. Taming information-stealing smartphone applications (on Android), Trust and Trustworthy Computing. Springer Berlin Heidelberg, 2011: 93–107.