

基于 Hook 技术的 Android 平台隐私保护系统

朱晓妍¹, 章辉¹, 马建峰²

(1. 西安电子科技大学通信工程学院, 陕西 西安 710071;
2. 西安电子科技大学网络与信息安全学院, 陕西 西安 710071)

摘要: 针对 Android 中恶意应用窃取用户隐私的问题, 设计并实现了一个基于 Hook 技术的隐私保护系统。该系统包括应用分类模块、隐私信息管理模块和共谋攻击管理模块。应用分类模块利用权限特征向量集构建分类模型, 对手机上的应用进行分类; 隐私信息管理模块利用 Hook 技术对所有涉及隐私信息的 API 进行挂钩, 通过返回虚假值或者空值, 拦截恶意应用获取隐私信息; 共谋攻击管理模块同样使用 Hook 技术, 对提供应用间通信的 API 进行挂钩, 当通信双方满足共谋攻击的权限规则, 通过返回空值实现拦截双方的通信内容。最后, 通过实验验证了该系统的可行性和有效性, 并且该系统不需要对 Android 系统和应用进行修改, 占用的内存较小, 能够实时拦截窃取隐私的行为。

关键词: 隐私保护; Android 平台; Hook 技术; 共谋攻击; 权限管理

中图分类号: TN91

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2018033

Privacy protection system based on Hook for Android

ZHU Xiaoyan¹, ZHANG Hui¹, MA Jianfeng²

1. School of Telecommunications Engineering, Xidian University, Xi'an 710071, China
2. School of Cyber Engineering, Xidian University, Xi'an 710071, China

Abstract: A privacy protection system based on Hook technology was designed and implemented to address the problem of user privacy leakage caused by malicious applications in Android. The system consists of three modules: application classification module, privacy information management module and collusion attack management module. The application classification module constructs classification model by using permission feature vectors and then classifies mobile applications into malicious applications and normal applications. The privacy information management module hooks all privacy-related application programming interfaces (API) based on Hook technology and then intercepts malicious applications by returning false data or empty data. The collusion attack management module hooks all inter-application communication channels and then intercepts corresponding communication contents by using the permission rules of collusion attack. Finally, the proposed privacy protection scheme on Android virtual machine was implemented to prove and its feasibility and effectiveness were proved. In addition, the system can intercept application behaviors of breaching privacy in real time with less memory, which does not need to modify Android system and applications.

Key words: privacy protection, Android platform, Hook technology, collusion attack, permission management

收稿日期: 2018-02-07; 修回日期: 2018-04-02

通信作者: 朱晓妍, xyzhu@mail.xidian.edu.cn

基金项目: 国家自然科学基金资助项目 (No.61772406); NSFC-通用技术基础研究联合基金资助项目 (No.U1636209); 中央高校基本科研业务费基金资助项目 (No.JB180110)

Foundation Items: The National Nature Science Foundation of China (No.61772406), NSFC-General Technical Fundamental Research Joint Project (No.U1636209), Fundamental Research Funds for Central Universities (No.JB180110)

1 引言

近年来,随着移动互联网的迅速发展,智能手机几乎成为人们生活及工作中的必需品。根据美国市场研究公司 Gartner^[1]发布的 2017 年全球智能手机系统市场份额报告显示,全球手机销量超过 15 亿台,其中,搭载 Android 系统的手机在全球手机市场的占有率达到 85.9%,以绝对的优势占据着主导地位。与此同时,Android 各类应用软件激增,为用户带来方便的同时也带来了危险,因为其中不乏有泄露隐私信息的恶意应用,对用户的隐私数据造成了严重的威胁。

针对 Android 智能手机隐私泄露的问题,国内外经典的恶意应用检测技术主要是基于行为的检测技术。

首先出现的是静态分析方法,采用静态数据流分析技术对应用中的敏感数据流向进行分析,如 Gibler 等^[2]提出的 AndroidLeaks 系统、Yang 等^[3]提出的 Leakminer 系统、Arzt 等^[4]提出的 Flowdroid 系统。静态分析方法虽然具有运行速度快、代码覆盖率高的特点,但无法解决代码混淆、加固的问题,因此出现了动态分析方法。动态分析方法根据应用程序运行时敏感信息的传输特征,动态监控应用泄露隐私的行为,其中,最著名的是 Enck 等^[5]设计的 TaintDroid 系统,该系统修改了 Android 内核代码,使用动态污点分析技术对隐私数据进行标记,当被标记的隐私数据通过网络或者其他途径离开手机系统时,TaintDroid 会记录隐私数据的去向,并警告用户。之后研究者对 TaintDroid 进行改进,设计出其他类似的系统,如 Hornyack 等^[6]设计的 AppFence 系统和 Lantz 等^[7]设计的 DroidBox 系统。

近年来,基于动态污点分析技术的思想,研究者设计出不需要对 Android 系统进行修改的动态分析系统。Rastogi 等^[8]提出的 Uranine 通过对应用进行修改,实现对应用的动态监控:首先对应用进行反编译,然后注入污染代码,最后将应用重新打包。You 等^[9]提出的 TaintMan 同样采取注入污染代码的方式对应用进行修改,同时对库文件进行导出,然后对库文件注入污染代码,最后将库文件重新导入,实现对应用更全面的动态监控。

虽然动态分析方法解决了静态分析不能解决的问题,但是也存在代码覆盖率不足、运行慢等问题,因此研究者结合 2 种分析方法的优点,提出了动静态结合分析方法。Yang 等^[10]提出的 AppIntent 首先利用静态分析方法获取应用中敏感数据传播的候选集,然后通过动态符号化执行验证候选敏感数据流是否会发生。Xia 等^[11]提出的 AppAudit 首先使用静态方法找出可疑函数,然后通过近似执行的动态分析方法执行部分应用代码,有效猜测未知变量,保证分析路径尽可能完全。

近年来,数据挖掘技术也被应用到检测恶意应用中^[12]。杨欢等^[13]提出的 Androject 系统首先利用静态和动态方法获取应用尽可能多的特征,然后分别对不同特征应用不同的分类器,最后通过判决算法对应用进行分类。Talha 等^[14]提出的 APKAuditor 系统首先使用静态方法获取应用权限特征,利用概率论方法对每个权限进行评分,然后用 Logistic 回归方法确定恶意应用分数的临界值。

由于针对单个应用的检测已经有很多方法,近几年出现了隐蔽性更高的共谋攻击。Bhandari 等^[15]针对内部组件通信而导致的共谋攻击提出了一种自动化框架,首先利用静态分析技术获取 2 个应用的通信流向,然后通过自定义权限策略判断是否存在共谋攻击。Bugiel 等^[16,17]设计了 XManDroid 框架,该框架能够分别从 Android 中间层、内核层监控应用间通信,同样通过自定义权限策略判断是否存在共谋攻击。Blasco 等^[18]针对共谋攻击开发出 ACE 系统,该系统能够根据配置文件自动生成共谋攻击应用对,为共谋攻击实验提供样本集。

基于以上提出技术的优点以及存在的问题,本文提出一种数据挖掘分类、Hook 技术监控和拦截的用户隐私信息保护系统。该系统提取基于权限的静态特征作为训练集,对智能手机上的应用进行分类判断,同时使用 Hook 技术对用户隐私信息的获取进行监控及拦截,针对隐蔽性较高的共谋攻击,采用 Hook 技术监控应用间通信渠道,对满足共谋攻击规则的应用进行拦截,实现更加全面的用户隐私信息保护。本文设计了多个实验

对 VirusShare 网站上的恶意应用和应用市场上的非恶意应用进行检测,证明本文提出方法的有效性,相比于基于 DTA 技术的检测系统,该系统不需要修改 Android 系统和应用本身,占用的内存不会随着系统的运行成倍增加,相比于文献[15]和文献[17]的权限策略,本文制订权限规则降低了误报率,并且该系统能够实时拦截窃取隐私的行为。

2 系统框架

本文提出了一种基于数据挖掘技术和 Hook 技术的 Android 用户隐私信息保护系统,能够检测与拦截恶意应用,保护用户的隐私信息。该系统将权限信息作为分类特征,对手机上的应用进行分类,同时使用 Hook 技术对应用获取隐私信息的行为进行监控与拦截;针对共谋攻击,同样使用 Hook 技术监控应用间通信,对于满足共谋攻击规则的应用行为进行拦截。

系统结构框架如图 1 所示,该系统由 3 部分组成:应用分类模块、隐私信息管理模块和共谋攻击管理模块,具体介绍如下。

1) 应用分类模块

提取从网络上收集的正常应用与恶意应用的权限特征,将各自的权限信息组合成权限向量,所有权限向量组合成的向量集作为数据挖掘分类算法的训练集,生成分类模型,内置在系统中;对于手机上的应用,通过 Android 系统提供的 PackageManager 类获取权限信息,组合成特征向量,根据分类模型进行分类判断,输出分类结果。

2) 隐私信息管理模块

使用 Hook 技术对提供涉及用户隐私信息的接口进行 Hook,用户隐私信息包括联系人信息、通话记录、短信信息、电话标识、地理位置、浏览历史、日历信息、多媒体文件等,当分类为恶意的应用调用涉及用户隐私信息的接口时,将空值或者随机生成的虚假值替换这些接口的返回

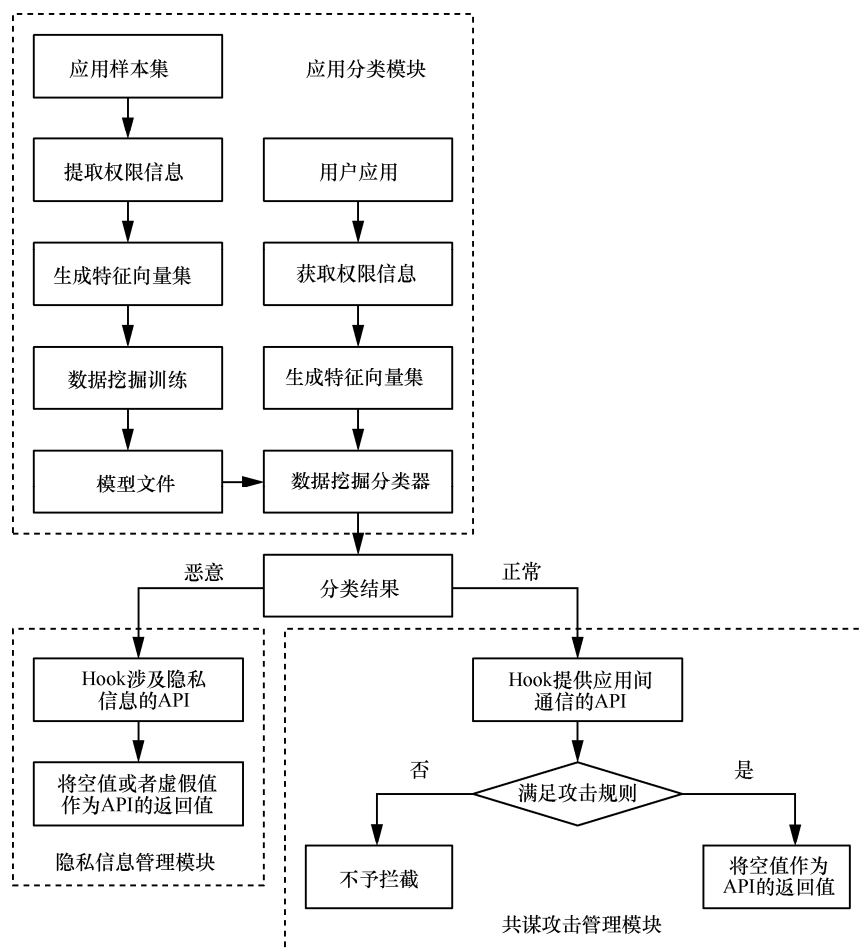


图1 系统结构框架

值,实现对用户隐私信息的拦截,对于分类为正常的应用不予拦截。

3) 共谋攻击管理模块

使用 Hook 技术对提供应用间通信的接口进行 Hook,通信渠道包括 Intent、ContentProvider、SharedPreferences 以及 External Storage,当 2 个分类为正常的应用调用提供通信的接口并访问同一个位置时,对双方权限信息进行共谋攻击权限规则判断,若满足规则,将空值替换通信渠道接口的返回值,实现对通信内容的拦截,并对双方的隐私信息获取和外部访问能力进行限制,否则不予拦截。

3 关键技术及实现

3.1 特征构建

系统中的特征构建采用的是静态分析技术,图 2 为特征构建的流程,具体步骤如下。

1) 使用反编译软件对应用 apk 文件进行批量反编译操作,获得反编译文件夹中的 AndroidManifest.xml 文件。

2) 使用脚本对每个 xml 文件进行解析,提取其中的权限特征信息,各自放到一个集合中。

3) 自定义权限向量矩阵,并根据权限信息集对权限向量矩阵进行赋值,将权限向量矩阵作为特征向量集。

上述步骤除了反编译操作使用软件进行,其余步骤都是通过 Python 脚本自动执行。首先通过 ElementTree 解析 AndroidManifest.xml 文件,获取所有<uses-permission>节点的内容,并放进一个列表中。然后将 Android 提供的所有权限信息按照字母顺序生成一个权限向量,向量元素只包含 1 和 0,1 表示应用具有元素对应的权限,0 表示应用不具有元素对应的权限,再在这个向量中添加一个元素表示应用类型,1 表示恶意应用,0 表示正常应用,形成特征向量。将这些特征向量放进一个 CSV 文件中,形成一个训练集。

3.2 Hook 技术

系统中隐私信息管理模块和共谋攻击管理模块的监控和拦截都是基于 Hook 技术,其中,隐私信息管理模块主要对涉及用户隐私信息的接口进行 Hook,而共谋攻击管理模块主要对应用间的通信渠道进行 Hook,下面分别介绍 2 个模块实现 Hook 的细节。

3.2.1 隐私信息 Hook

Android 框架层中提供了很多管理系统服务的接口,可以通过这些接口获取所需的系统资源,首先需要通过 getSystemService 函数获取相关管理类,其中,LocationManager 类和 TelephonyManager 类分别提高了位置信息和电话信息的获取,因此这 2 个类中的一些函数是 Hook 的目标。

Android 四大组件之一的 ContentProvider 可以用来实现应用程序之间共享数据,Android 系统通过默认的 URL 路径为开发者提供常见的数据获取方法,其中,联系人、通话记录、短信、浏览记录、日历信息、多媒体文件等就是通过这种方式获取的。本方案调用 getContentResolver 函数得到对象来获取 query 函数。因为 query 函数根据传入的 URL 值返回对应的数据,本方案把它作为 Hook 的目标。

表 1 展示了部分被 Hook 的提供用户隐私信息的接口结构。

3.2.2 应用间通信渠道 Hook

Intent 是 Android 组件之间相互通信的纽带,封装了不同组件之间通信的条件,根据这些条件启动或者唤醒想要通信的组件,而这些组件可以位于不同应用中。2 个组件通信时,发送方可以通过 putExtra 等函数携带数据,接收方通过 getExtra 等函数获得数据,因此,Intent 类中传递数据和获取数据的函数是 hook 的目标。

ContentProvider 组件不仅可以获取系统资源,也可以通过自定义 URL 路径实现自己的数据共享。ContentProvider 提供了类似于数据库的操

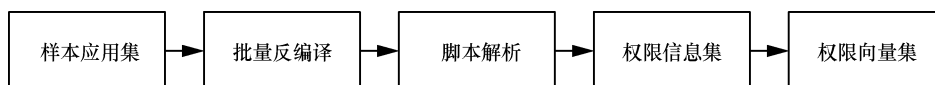


图 2 特征构建流程

表 1 提供用户隐私信息的接口结构

获取指定类的接口	指定类	类提供的接口（参数）
getSystemService	LocationManager	getLastKnownLocation requestLocationUpdates
	TelephonyManager	getDeviceId getLine1Number
getContentResolver	ContentResolver	query (content://call_log、content://browser.....)

作，通过 insert、delete、update 和 query 函数实现增删改查，因此，ContentResolver 类中插入、修改和查询的函数是 Hook 的目标。

Android 文件系统提供了 SharedPreferences 和外部存储对数据进行存放和读取，其中 SharedPreferences 实际上是一个 xml 文件，而外部存储的文件可以是任何格式。不管使用哪种方式，最终都会调用 IoBridge 类的 open 函数，因此 open 函数是 Hook 的目标。

表 2 展示了部分被 Hook 的应用间通信渠道接口结构。

表 2 应用间通信渠道的接口结构

类	接口
Intent	putExtra、putExtras
	getExtras、getStringExtra.....
ContentResolver	inert、update、query
IoBridge	open

3.2.3 Xposed 模块实现

确定了 Hook 目标，就可以进行 Hook 操作。著名的开源项目 Xposed 是一个基于 Java 层 Hook 的框架服务，其 Hook 函数的原理就是修改 Java 虚拟机调用函数的方式，将指向原函数的指针指向自己实现的函数，然后在这个函数中调用原来的函数，并在调用前后插入钩子，这样就达到 Hook 的目的。

本文基于 Xposed 框架实现了一个 Hook 模块，针对以上提到的接口进行 Hook，添加的操作主要位于上述接口之前。

针对涉及用户隐私信息的各个接口的处理流程大致相同，具体过程如下。

1) 对于 LocationManager 类和 Telephony Manager 类，获取类的所有接口名称，对于

ContentResolver 类，获取 query 方法的参数信息。

2) 根据接口的名称或者接口的参数判断通过该接口能否获取隐私信息。

3) 若不能获取隐私信息，结束处理流程；若可以，查询应用分类模块中的恶意应用数据库是否存在调用该接口的应用的名称。

4) 若不存在，结束处理流程；若存在，生成与该接口返回数据类型相同的虚假数据，作为该接口的返回值。

针对所提供应用间通信的各个接口的处理流程，可以根据处理数据的不同分为两类，具体过程如下。

存储数据的接口的处理流程如下。

1) 通过接口的参数获取存储位置，并通过 Binder 类的接口获取调用该接口的应用的名称。

2) 查询应用分类模块的恶意应用数据库中是否存在该应用的名称。

3) 若存在，结束处理流程，若不存在，将该应用的名称、存储数据的位置和通信渠道类型写入共谋攻击数据库中。

读取数据的接口的处理流程如下。

1) 通过接口的参数获取读取位置，并通过 Binder 类的接口获取调用该接口的应用的名称。

2) 查询应用分类模块的恶意应用数据库中是否存在该应用的名称。

3) 若存在，结束处理流程；若不存在，查询共谋攻击数据库中是否存在通信位置和通信类型相同但是应用名称不同的记录。

4) 若不存在，结束处理流程；若存在，获取两个应用的权限信息，并判断是否符合权限规则。

5) 若不符合规则，结束处理流程；若符合，将 2 个应用的名称写入恶意应用数据库中，并将空值作为接口的返回值。

3.3 共谋攻击规则

为了隐蔽自己的行为,一些恶意应用的作者将获取隐私信息的行为和发送隐私信息的行为分割到2个应用执行,减少了每个应用申请的权限,而一般的检测工具都是针对单个应用的,通常很难检测到这种攻击。系统针对数据挖掘分类不能很好检测出共谋攻击的问题,设计了一种针对共谋攻击的检测规则。

应用获取隐私信息和访问外界都需要申请相应的权限,因此可以针对应用的权限进行分析。文献[15]中提出若一个申请了某种隐私权限的应用通过 Intent 传递了信息,而接收 Intent 的应用申请了某种外部访问权限,那么它们之间可能存在共谋攻击。而实际上有些正常应用会相互调用,如在支付时,可能会调用支付宝或者微信,而双方可能满足了这种权限规则,这种情况下进行拦截就会造成交易异常。因此,本文在这种规则的基础上,添加了一些限制,从而减小误判的概率。

定义集合 S_p 为所有隐私权限的集合,集合 S_T 为所有外部访问权限的集合,对于2个应用 A 和 B,定义它们的权限集合为 S_A 和 S_B ,若2个应用同时满足以下式子,则认为双方存在共谋攻击。

$$S_A \cap S_p \neq \emptyset \quad (1)$$

$$S_A \cap S_T \neq \emptyset \quad (2)$$

$$S_B \cap S_T \neq \emptyset \quad (3)$$

$$S_B \cap S_p \neq \emptyset \quad (4)$$

其中,式(1)表示应用 A 申请了隐私权限,式(3)应用 B 申请了外部访问权限,式(2)和式(4)是针对共谋攻击的分割性,同时为了降低误判而添加的条件,式(2)表示应用 A 没有申请外部访问的权限,式(4)表示应用 B 没有申请隐私权限。

4 实验分析

4.1 实验工具及环境

本文所有实验在内存为 6 GB,处理器为 Intel(R) Core i5-3210M 2.50 GHz 的 Windows7 系统上完成,Android 虚拟机系统版本为 4.4。特征构建部分和样本收集部分使用 Python 语言,在 Eclipse 平台上完成,其他部分使用 Java 语言,在

Android Studio 平台上完成。

4.2 实验组成

1) 应用分类模块测试:包括对收集的样本应用集进行特征构建和分类测试。恶意应用来自 VirusShare 网站,正常应用使用 Python 脚本爬取应用商店不同类别的应用,经过筛选,剔除部分无效应用,最终选取了 688 个恶意应用和 594 个正常应用,测试分类的正确率。

2) 隐私信息管理模块测试:包括对应用获取隐私信息的监控以及对恶意应用进行拦截的测试,拦截以日志的形式存储并展示。

3) 共谋攻击管理模块测试:包括对应用间通信的监控以及对存在共谋攻击的应用进行拦截的测试,拦截以日志的形式存储并展示。

4.3 实验评估

4.3.1 应用分类模块

本实验采用混淆矩阵对应用分类性能进行分析,它是分析分类器识别不同类元组情况的一种工具。本文将恶意应用定义为正元组,将正常应用定义为负元组,如表 3 所示。

表 3 混淆矩阵

预测结果	恶意应用	正常应用
恶意应用	TP(true positive)	FP(false positive)
正常应用	FN(false negative)	TN(true negative)

TP 表示分类器将恶意应用正确识别为恶意应用的数量,FP 表示分类器将正常应用错误识别为恶意应用的数量,FN 表示分类器将恶意应用错误识别为正常应用的数量,TN 表示分类器将正常应用正确识别为正常应用的数量。

由于随机森林算法的思想简单,实现容易,计算开销小,本文采用随机森林算法通过十折交叉确认对 1 282 个应用进行测试,得到的混淆矩阵如表 4 所示,并根据混淆矩阵可以计算出分类效果的评定指标——准确率、精确率、找回率以及精确率和召回率的调和平均值 F -Measure。

表 4 应用分类模块的混淆矩阵

预测结果	恶意应用/个	正常应用/个
恶意应用	634	65
正常应用	54	529

$$\text{准确率} = \frac{TP + TN}{TP + FP + FN + TN} = 90.7\%$$

$$\text{精确率} = \frac{TP}{TP + FP} = 90.7\%$$

$$\text{召回率} = \frac{TP}{TP + FN} = 92.2\%$$

$$F\text{-Measure} = \frac{2TP}{2TP + FP + FN} = 91.4\%$$

根据对机器学习的性能评定, 精确率和召回率的结果越趋近于1, 说明查准率和查全率越高。而根据计算结果显示, 分类的准确率达到90.7%, 精确率达到90.7%, 召回率达到92.2%, F -Measure达到91.4%, 证明了随机森林算法对应用进行分类的准确性。

为了进一步说明随机森林算法对应用分类的分类效果, 本文同时采用 Native Bayes 算法、SVM 算法、J48 算法和 Bagging 算法对1282个应用进行分类测试, 图3为各个算法分类的准确率对比。

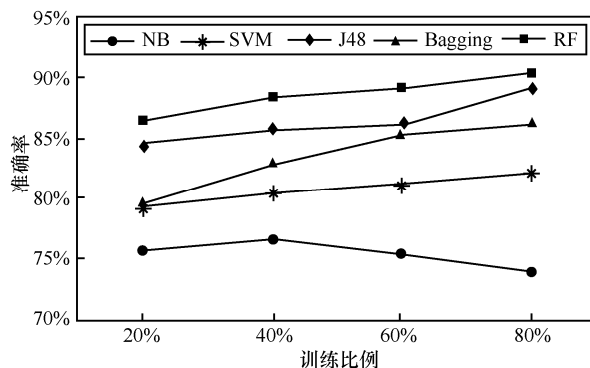


图3 各个算法分类的准确率

从图3可以看出, 相比于单一分类算法, 随机森林算法的准确率要高得多, 而相比于同为集成学习算法的 Bagging, 随机森林算法的准确率也较高, 充分证明了随机森林算法对应用分类模块的有效性。

4.3.2 隐私信息管理模块

本实验通过在 Android 虚拟机上运行隐私信息管理模块, 来实时监控恶意应用以及拦截它们获取隐私信息的行为, 并将具体行为记录下来, 通过界面进行展示, 验证该模块的有效性, 部分结果如图4所示。



图4 隐私信息管理模块拦截日志

图4展示了2个恶意应用试图获取的隐私信息, uid为10033是一个会计从业资格考的应用, uid为10031是一个常用汉字识字卡的应用, 它们都是在提供一些正常功能的情况下额外获取用户的隐私信息。以常用汉字识字卡这个应用为例, 当单击图标进入该应用主界面时, 该应用会尝试获取位置信息和电话标识, 并且可以看出来, 它试图通过调用不同API来获取位置信息, 但都被拦截了下来。

通过隐私信息管理模块的拦截日志, 用户可以清楚知道恶意应用获取的隐私信息种类、获取时间、获取方式和获取次数, 证明了隐私信息管理模块的有效性。

4.3.3 共谋攻击管理模块

本实验首先验证权限规则的准确性, 然后通过Android虚拟机上运行共谋攻击管理模块, 来实时监控存在共谋攻击的应用以及拦截它们之间的通信行为, 并将具体行为记录下来, 通过界面进行展示, 验证该模块的可靠性。

文献[18]中设计了一个共谋攻击应用生成系统, 它能够根据配置文件自动生成相对应的应用对, 模拟恶意应用进行共谋攻击, 因此本实验选取文献[18]中设计的系统生成的应用对作为实验

样本。该实验样本中，将 Intent 作为通信渠道的应用对共有 81 对，将 SharedPreferences 作为通信渠道的应用对共有 80 对，将 External Storage 作为通信渠道的应用对共有 80 对，共计 241 对应用对。

首先，为了验证权限规则的准确性，本实验通过 Python 脚本将每对应用的权限信息提取出来，对这 2 个应用进行权限规则判断，图 5 展示了获取各类隐私信息的应用个数以及检测出来的应用个数。

由图 5 可以看出，所有的共谋攻击应用对都被检测了出来，即检测率为 100%，证明了权限规则的准确性。

为了与机器学习算法的分类效果做比较，本文将这 241 对应用的权限信息提取并转换成特征向量，使用应用分类模块的训练模型对它们进行分类，表 5 展示了每对应用的检测结果以及检测个数。

表 5 每对应用的检测结果及检测个数

检测结果	检测个数
检测出 0 个应用	62
检测出 1 个应用	126
检测出 2 个应用	53

由表 5 可以计算出 482 个恶意应用的检测率。

$$\text{检测率(应用)} = \frac{\text{检测出1个应用} + 2 \times \text{检测出2个应用}}{\text{应用个数}} = 48.1\%$$

考虑到共谋攻击中，一旦其中一个恶意应用被检测出来，就无法完成隐私信息的窃取，因此可以计算出 241 对恶意应用的检测率。

$$\text{检测率(应用对)} = \frac{\text{检测出1个应用} + \text{检测出2个应用}}{\text{应用对数}} = 74.3\%$$

由以上 2 个检测率的结果可以看出，使用机器学习算法对共谋攻击应用的检测率比使用权限规则低很多，充分证明了使用权限规则检测共谋攻击的有效性。

然后，为了证明本文设计的权限规则降低了误报率，选取了 5 个存在互相调用的正常应用，触发它们之间调用的原因都是信息分享，表 6 展示了各个应用是否存在互相调用的关系。

表 6 各个应用互相调用的关系

应用	是否调用新浪微博	是否调用微信
QQ 音乐	是	是
新浪新闻	是	否
腾讯新闻	是	是

笔者使用 QQ 音乐、新浪新闻以及腾讯新闻的信息分享功能时，触发了新浪微博和微信的唤起，唤起过程中应用运行正常，并且没有触发模块的拦截提示。而若采用文献[17]的策略，唤起过程中会触发策略 8 和策略 11（如图 6 所示），导致应用运行异常，证明了本文设计的权限规则降低了误报率。

最后，为了验证该模块拦截的有效性，分别

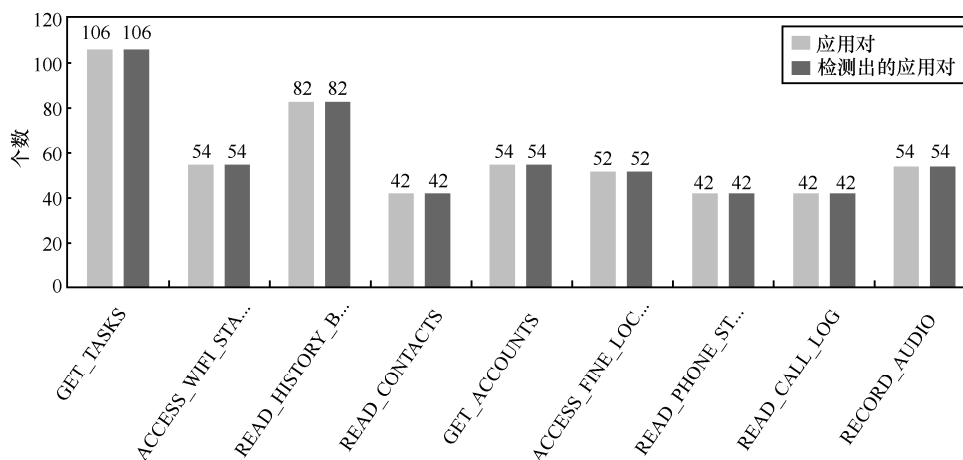


图 5 各类应用个数以及检测出的应用个数

Advanced rules	
(8)	A third party application with permission ACCESS_FINE.LOCATION must not communicate to a third party application that has permission INTERNET
(9)	A third party application with permission READ_CONTACTS must not communicate to a third party application that has permission INTERNET
(10)	A third party application with permission READ_SMS must not communicate to a third party application that has permission INTERNET
Strong rules	
(11)	A third party application with permission RECORD_AUDIO and PHONE.STATE or PROCESS_OUTGOING_CALLS must not communicate to a third party application that has permission INTERNET

图6 部分策略

选取了样本中通过3种不同渠道窃取隐私的应用对,运行于Android系统中,拦截的日志记录如图7所示。

由图7可以看出,共谋攻击管理模块对3种通信渠道都实现了监控并拦截了恶意应用,同时获取了通信双方以及通信渠道的路径或者标识,能够一目了然地知道恶意应用的具体行为,证明了该模块拦截恶意应用的有效性。



图7 共谋攻击拦截日志

5 结束语

本文提出一种基于Android平台的隐私信息保护系统,该系统结合数据挖掘分类技术和hook技术,在不需要修改Android系统和应用自身、占用内存较小的情况下实现了动态监控和拦截恶意应用获取隐私信息的行为,同时针对隐蔽性较高的共谋攻击,设计了基于权限的规则检测,在保证检测共谋攻击的情况下减少了误报率。实验结果表明,该系统能够准确拦截隐私信息的获取以及应用间通信内容,对保护隐私信息能够起到较好的作用。下一步工作将继续完善系统,使用更精确的分类算法和权限规则,扩展系统监控的API范围,更全面地保护Android系统中的隐私数据。

参考文献:

- [1] Gartner. Market share: final PCs, ultramobiles and mobile phones, all countries, 4Q17[R]. Connecticut: Gartner, 2018.
- [2] GIBLER C, CRUSSELL J, ERICKSON J, et al. AndroidLeaks: automatically detecting potential privacy leaks in android applications on a large scale[J]. Trust, 2012, 12: 291-307.
- [3] YANG Z, YANG M. Leakminer: detect information leakage on android with static taint analysis[C]//2012 Third World Congress on Software Engineering (WCSE). 2012: 101-104.
- [4] 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.
- [5] ENCK W, GILBERT P, CHUN B G, et al. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones[C]//The 9th USENIX Conference on Operating Systems Design and Implementation. USENIX Association, 2010.
- [6] HORNYACK P, HAN S, JUNG J, et al. These aren't the droids you're looking for: Retrofitting Android to protect data from imperious applications[C]//The 18th ACM Conference on Computer and Communications Security. ACM, 2011: 639-652.
- [7] Droidbox[EB/OL]. <http://code.google.com/p/droidbox/>.
- [8] RASTOGI V, QU Z, MCCLURG J, et al. Uranine: real-time privacy leakage monitoring without system modification for Android[J]. Security and Privacy in Communication Networks, 2015: 256-276.
- [9] YOU W, LIANG B, SHI W, et al. TaintMan: an ART-compatible dynamic taint analysis framework on unmodified and non-rooted Android devices[J]. IEEE Transactions on Dependable and Secure Computing, 2017.
- [10] YANG Z, YANG M, ZHANG Y, et al. Appintent: Analyzing sensitive data transmission in Android for privacy leakage detection[C]//2013 ACM SIGSAC Conference on Computer & Communications Security. ACM, 2013: 1043-1054.
- [11] XIA M, GONG L, LYU Y, et al. Effective real-time android application auditing[C]//2015 IEEE Symposium on Security and Privacy (SP). 2015: 899-914.

- [12] 孙博文, 黄炎裔, 温俏琨, 等. 基于静态多特征融合的恶意软件分类方法[J]. 网络与信息安全学报, 2017, 3(11): 68-76.
SUN B W, HUANG Y Y, WEN Q K, et al. Malware classification method based on static multiple-feature fusion[J]. Chinese Journal of Network and Information Security, 2017, 3(11): 68-76.
- [13] 杨欢, 张玉清, 胡予濮, 等. 基于多类特征的 Android 应用恶意行为检测系统[J]. 计算机学报, 2014, 37(1): 15-27.
YANG H, ZHANG Y Q, HU Y P, et al. A malware behavior detection system of Android applications based on multi-class features[J]. Chinese Journal of Computers, 2014, 37(1): 15-27.
- [14] TALHA K A, ALPER D I, AYDIN C. APK auditor: permission-based Android malware detection system[J]. Digital Investigation, 2015, 13: 1-14.
- [15] BHANDARI S, LAXMI V, ZEMMARI A, et al. Intersection automata based model for android application collusion[C]//2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA). 2016: 901-908.
- [16] BUGIEL S, DAVI L, DMITRIENKO A, et al. Xmandroid: a new android evolution to mitigate privilege escalation attacks[R]. Technical Report, 2011.
- [17] BUGIEL S, DAVI L, DMITRIENKO A, et al. Towards taming privilege-escalation attacks on Android[C]//NDSS. 2012: 19.
- [18] BLASCO J, CHEN T M. Automated generation of colluding Apps for experimental research[J]. Journal of Computer Virology and Hacking Techniques, 2017: 1-12.

[作者简介]



朱晓妍 (1979-), 女, 陕西西安人, 博士, 西安电子科技大学通信工程学院副教授, 主要研究方向为大数据环境下移动互联网隐私保护与风险管控、通信与计算机网络安全、应用密码学、大数据应用。



章辉 (1993-), 男, 福建三明人, 西安电子科技大学硕士生, 主要研究方向为移动互联网隐私保护。



马建峰 (1963-), 男, 陕西西安人, 博士, 西安电子科技大学教授、博士生导师, 主要研究方向为应用密码学、无线网络安全、数据安全、移动智能系统安全。