基于多类特征的Android应用恶意行为检测系统

杨 欢[1)] 张玉清[1),2)] 胡予濮[1)] 刘奇旭[2)]

[1)](西安电子科技大学综合业务网理论及关键技术国家重点实验室 西安 710071)

[2)](中国科学院大学国家计算机网络入侵防范中心 北京 100190)

摘 要 目前针对未知的Android恶意应用可以采用数据挖掘算法进行检测,但使用单一数据挖掘算法无法充分发挥Android应用的多类行为特征在恶意代码检测上所起的不同作用.文中首次提出了一种综合考虑Android多类行为特征的三层混合系综算法THEA(Triple Hybrid Ensemble Algorithm)用于检测Android未知恶意应用.首先,采用动静态结合的方法提取可以反映Android应用恶意行为的组件、函数调用以及系统调用类特征;然后,针对上述3类特征设计了三层混合系综算法THEA,该算法通过构建适合3类特征的最优分类器来综合评判Android应用的恶意行为;最后,基于THEA实现了Android应用恶意行为检测工具Androdect,并对现实中的1126个恶意应用和2000个非恶意应用进行检测.实验结果表明,Androdect能够利用Android应用的多类行为特征有效检测Android未知恶意应用.并且与其它相关工作对比,Androdect在检测准确率和执行效率上表现更优.

关键词 系综算法;Android应用;多类特征;恶意代码检测;行为分析;数据挖掘;智能手机;网络行为

中图法分类号 TP393 **DOI号** 10.3724/SP.J.1016.2014.00015

A Malware Behavior Detection System of Android Applications Based on Multi-Class Features

YANG Huan[1)] ZHANG Yu-Qing[1),2)] HU Yu-Pu[1)] LIU Qi-Xu[2)]

[1)](*State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an* 710071)

[2)](*National Computer Network Intrusion Protection Center, University of Chinese Academy of Sciences, Beijing* 100190)

Abstract At present, data mining algorithm is always used to detect unknown malicious applications of Android. While single data mining algorithm could not play the role of multi-class Android features in malware detection. For this problem, a Triple Hybrid Ensemble Algorithm (THEA) was first proposed, which considered multi-class Android features. First, we combined the static and dynamic methods to extract three classes of Android features, which could reflect malicious behavior effectively, such as components, function calls and system calls. Second, we designed THEA to build optimal classifier by handling three classes of features and then made a comprehensive judgment of unknown Android application. Finally, we implemented an automated tool named Androdect to detect 1126 malicious and 2000 non-malicious apps in real. The experimental results show that Androdect plays the role of multi-class Android features in unknown malware detection and it performs better than other related works on the availability, efficiency and accuracy.

Keywords ensemble algorithm; Android application; multi-class features; malicious code detection; behavior analysis; data mining; smartphone; network behavior

收稿日期:2013-04-13;**最终修改稿收到日期**:2013-11-19.**本课题得到国家自然科学基金**(61272481)、**中国博士后科学基金资助项目**(2011M500416,2012T50152)、**北京市自然科学基金**(4122089)、**国家发改委信息安全专项**(**发改办高技**[2012]1424**号**)**资助**.**杨 欢**,**女**,1984**年生**,**博士**,**助理工程师**,**主要研究方向为信息安全、漏洞挖掘和恶意代码检测**.** E-mail: yangh@nipc.org.cn.**张玉清**,**男**,1966**年生**,**博士**,**教授**,**中国计算机学会**(CCF)**会员**,**主要研究领域为网络与信息系统安全**.**胡予濮**,**男**,1955**年生**,**博士**,**教授**,**主要研究领域为序列密码与分组密码、网络安全协议的设计与分析**.**刘奇旭**,**男**,1984**年生**,**博士**,**主要研究方向为信息安全、漏洞挖掘和漏洞等级评估**.**

1 引 言

近年来,智能手机普及率迅速增加,智能手机应用功能激增.全球最具权威的 IT 研究与顾问咨询公司 Gartner 的数据显示①,预期到 2016 年年底,将有 23 亿部电脑、平板电脑以及智能手机使用 Android 系统,而微软 Windows 设备的数量为 22.8 亿.Android 近年来的快速发展使其成为了一个占主导地位的智能手机平台,拥有整个市场约三分之二的份额.由于 Android 系统的开放性,它也成为了众多恶意代码开发者的活跃地盘.

Android 用户可以从 Google Play② 和第三方 Android 市场(例如:Amazon、hiapk、gfan 等)下载应用程序.应用程序开发者可以上传自己的应用程序到第三方 Android 市场.Google Play(前名为 Android Market),是一个由 Google 为 Android 设备开发的在线应用程序商店,可以让用户浏览、下载及购买 Google Play 上的第三方应用程序③.Google Play 提供了 Google's Bouncer 检测恶意程序,但是实时性不够,恶意程序在被检测出来之前已经被用户大量下载[1].

根据独立安全机构 Av-test 在 2013 年 1 月使用 ESET、Lookout:Antivirus & Security、Kaspersky、Qihoo360 等 22 个手机杀毒软件对 869 个已知 Android 恶意应用的检测报告显示④,各大手机杀毒软件的平均查杀率为 94%.而对于未知恶意应用的检测,各大厂商没有公开杀毒软件的技术细节,本文无法做详细分析.由于恶意代码的数量和种类越来越多,加上代码混淆、加密等技术的兴起,使得恶意代码检测变得越来越困难.

目前,典型的恶意代码检测技术包括基于签名、行为的检测方法.传统的基于签名的检测技术被普遍使用,但是它必须拥有一类恶意应用的签名库后才能检测该类恶意应用,因此无法有效检测未知的恶意应用.

基于行为的恶意代码检测目前主要采用动态和静态两种方法.动态方法是在系统运行过程中收集应用程序的一些行为信息.优点是绕过了静态方法遇到的代码混淆和加密等方面的问题,缺点是动态测试代码覆盖率低,并且有些恶意程序可以防止自身在模拟器下运行,当在模拟器下运行时会自动崩溃;静态方法主要研究使用反汇编反编译技术或者在 smali 中间代码上进行控制流和数据流分析技术来进行恶意代码检测.优点是代码覆盖率高,缺点是要解决静态方法无法检测的代码混淆、加密以及在动态执行中才解码恶意代码的问题.

由于数据挖掘技术可以从大量数据中挖掘出有意义的信息[2-4],有些研究者利用该技术挖掘出恶意应用的行为特征,以此来检测未知恶意应用.但是,目前的研究方法存在 3 方面的局限性:(1)不同的数据挖掘算法针对同一类特征的检测效果不同,无法预知哪个算法效果最优;(2)同一种算法对不同类型的特征检测效果不一定都是最优;(3)使用单一的算法不能充分发挥每类特征在 Android 恶意应用检测时所起的不同作用.

本文针对以上方法的局限性,首次提出一种基于三层混合系综算法的多类特征 Android 应用恶意行为检测系统,自动检测 Android 平台的恶意应用程序.该系统采用动静态结合的行为特征提取方法,提取 Android 应用的多类行为特征,再采用三层混合系综算法建立检测模型,并对大量现实中的应用程序进行检测.具体方法介绍如下:

首先,本文创建恶意应用库和非恶意应用库.恶意应用库的创建参考文献[5]中所使用的部分恶意应用.非恶意应用库的创建是通过修改 Akdeniz⑤ 的爬虫程序从 Google Play 上批量下载应用程序,并使用多种已有检测软件、杀毒软件和人工进行检测,确保数据库的准确性.

其次,在特征提取阶段,本文采用动静态结合的分析技术.静态分析主要对 AndroidManifest.xml 文件进行自动化分析.一是提取 packet 名称和 activity 名称为动态分析提供参数;二是对 permission、activity、service、receiver 和 provider 组件进行特征提取;三是对使用 native 代码的应用程序中的 lib 库文件进行分析,提取应用程序的函数调用序列.动态分析主要采用沙盒技术,自动完成启动模拟器、安装应用程序、模拟用户行为和提取系统调用序列等工作.

而后,在特征描述阶段,对提取的多类特征进行

① Gartner: Android usage within four years beyond Windows [EB/OL]. http://tech.163.com/12/1025/06/8EL1H4IQ000915BD.html. 2012-10-25
② Google play [EB/OL]. https://play.google.com/store. 2012-04-01
③ Google Play [EB/OL]. http://en.wikipedia.org/wiki/Google_Play. 2013-03-02
④ Test: 22 Security Apps for Android [EB/OL]. http://www.av-test.org/en/tests/mobile-devices/android/jan2013/. 2013-04-01
⑤ Google-play-crawler[EB/OL]. https://github.com/Akdeniz/google-play-crawler. 2012-12-01

格式化处理，构建各类特征的集合. 一是静态获取的各个组件使用个数的统计数据；二是静态获取的应用程序 lib 文件所使用的重要函数调用的统计数据；三是动态获取的各个系统调用使用次数的统计数据.

最后，使用三层混合系综算法建立检测模型. 第一层使用多个基础分类器对 3 种不同类型特征进行训练，给出适合不同类型特征的最优算法；第二层将系综算法与最优分类算法结合再次进行训练建立模型，进一步提高检测准确率；第三层对待检测的应用程序使用建立的模型进行分类并使用判决算法给出检测结果.

本方法通过实验对 1126 个恶意应用和 2000 个非恶意应用进行了检测. 实验结果表明，本文提出的方法在有效性、准确性和执行效率上表现良好，准确率为 94.24%，优于其它 Android 恶意应用检测系统.

本文的主要创新点与贡献如下：

(1) 采用动静态结合的行为特征提取方法，动态方法弥补了静态方法不能检测运行中释放的恶意代码的缺点，也绕过了静态方法遇到的代码加密和混淆问题；静态方法为动态方法提供了运行参数，可以分析动态方法无法处理的应用程序.

(2) 本文设计的检测系统提取了应用程序的多类行为特征，充分反映了应用程序行为，并且该系统具有可扩展性，可以加入更多类型特征进行检测.

(3) 首次提出了一种三层混合系综算法 THEA (Triple Hybrid Ensemble Algorithm)，对各类特征分别选取最优的基础分类算法从而给出综合判决结果. 并首次将其应用到 Android 应用的恶意代码检测，实验证明了该算法的有效性.

(4) 将提出的原型系统实现了自动化的检测工具 Androdect，对大量现实中的恶意应用和非恶意应用进行检测.

本文第 2 节介绍相关工作；第 3 节总体介绍检测系统；第 4 节阐述系统的关键技术和算法；第 5 节给出实验设计与结果分析；第 6 节讨论本文提出系统的局限性和下一步工作；最后在第 7 节对本文进行总结.

2　相关工作

随着智能手机的广泛应用，发布手机应用程序的官方市场和第三方市场上都出现了含有恶意代码的应用程序，使得智能手机应用程序的恶意代码检测成为了研究热点. 由于 Android 系统占有大量的市场份额以及其平台的开放性，手机平台恶意代码检测研究更多地集中在 Android 系统平台. 目前，国内外典型的 Android 平台恶意代码检测技术主要有基于签名和基于行为的检测方法.

首先出现的是基于签名的方法. 文献[6]重点介绍了基于特征码的恶意代码检测方法. 国外著名的 Android 恶意代码检测工具 Androguard① 也是基于签名的方法，它的主要缺点是无法检测未知恶意应用.

随后，由于基于签名的方法无法检测未知恶意应用，研究者开始研究基于行为的恶意代码检测方法. 文献[7]使用动态方法监视手机应用程序的网络行为、短信和隐私等信息. Bläsing 等人[8]使用动态方法分析 Android 应用的系统调用、网络访问、文件和内存等方面的信息. Enck 等人[9]提出了使用动态污染分析变量层、方法层、消息层和文件层的信息，用于检测隐私泄露方面的漏洞. 路程[7]、国外研究者 Shabtai 等人[10]以及 Padriya 等人[11]研究了 Android 源代码的静态分析技术.

近年来，也有研究者将数据挖掘技术应用到手机平台恶意代码检测. 2009 年，Schmidt 等人[12]使用静态方法获取函数调用特征，对 Symbian 系统和 Android 系统[13]进行基于分类算法的恶意代码检测，由于研究时间比较早，还没有大量真实的恶意样本用来验证结果. 2011 年，Iker 等人[14]使用动态方法提取系统调用特征，应用 K-means 聚类算法区分基于同一应用程序改写的恶意代码和非恶意代码，该方法只限于检测同一应用程序的不同变种是否包含恶意代码. 2012 年，Shabtai 等人[15]使用动态人工方法提取 API 特征并使用分类算法对自己编写的 Android 恶意代码进行检测，但没有使用现实中的大量应用程序进行验证.

上述工作从基于签名的检测方法过渡到基于行为的检测方法，从检测已知恶意应用过渡到检测未知恶意应用，在基于数据挖掘算法的研究中大部分是基于处理单一类型特征的数据挖掘算法，然而，实际中需要考虑恶意应用的多类行为特征，目前还没有一种数据挖掘算法可以充分发挥不同类型特征在 Android 恶意代码检测中所起的不同作用.

① Androguard [EB/OL]. http://code.google.com/p/androguard/2013-3-4

3 Androdect 检测系统

本文提出了一种基于三层混合系综算法的多类特征 Android 恶意行为检测系统，并实现了工具 Androdect，可以自动化地检测未知恶意应用. 该系统采用动静态结合的分析技术提取 Android 应用的多类行为特征，并设计了三层混合系综算法 THEA 建立检测模型.

Androdect 检测系统框图如图 1 所示，该系统由 6 个部分组成：Apps 数据库、预处理模块、特征提取模块、特征集合、格式化处理模块和三层混合系综算法检测模块，具体介绍如下：

(1) Apps 数据库. 本文创建了恶意应用程序库和非恶意应用程序库. 恶意应用程序库使用文献[5]中提供的部分恶意应用. 而对于非恶意应用库，由于 Google Play 将应用程序下载并直接安装在手机上，这对实验造成不便，因此，本文使用 Java 语言改写了 Akdeniz 的爬虫程序，增加了异常处理、去重、迭代等功能，可以实现从 Google Play 上批量不间断下载应用程序到 PC 端.

(2) 预处理模块. 文献[16]指出 Google Play 上 52 208 个应用程序中只有 2 个为恶意应用，所以可以认为 Google Play 上的应用程序基本为非恶意应用. 但是，为了使结果更加精确，本文对从 Google Play 上批量下载的应用程序使用其它检测工具、杀毒软件和人工进行检测，确保非恶意应用程序库中的应用程序均为非恶意应用，不包含可疑的恶意应用.

(3) 特征提取模块. 本文结合动静态行为分析技术. 其中，静态分析主要对应用程序进行自动化分析，提取各个组件特征和函数调用序列；动态分析采用沙盒技术，自动提取各个应用程序的系统调用序列特征.

(4) 特征集合. 在特征描述阶段，构建 3 种不同类型特征的集合. ① 静态获取各个组件使用个数的统计数据；② 静态获取应用程序 lib 文件中所使用的重要函数调用的统计数据；③ 动态获取各个系统调用使用次数的统计数据.

(5) 格式化处理模块. 本文使用 Python 语言编写代码对提取的特征集合进行格式化处理，统一处理为 CSV(Comma Separated Values)格式，即逗号分隔值格式. 每一行为一个应用程序测试样本.

(6) 三层混合系综算法检测模块. 使用三层混合系综算法对数据进行处理，创建最优的检测模型，对待检测应用程序进行检测.

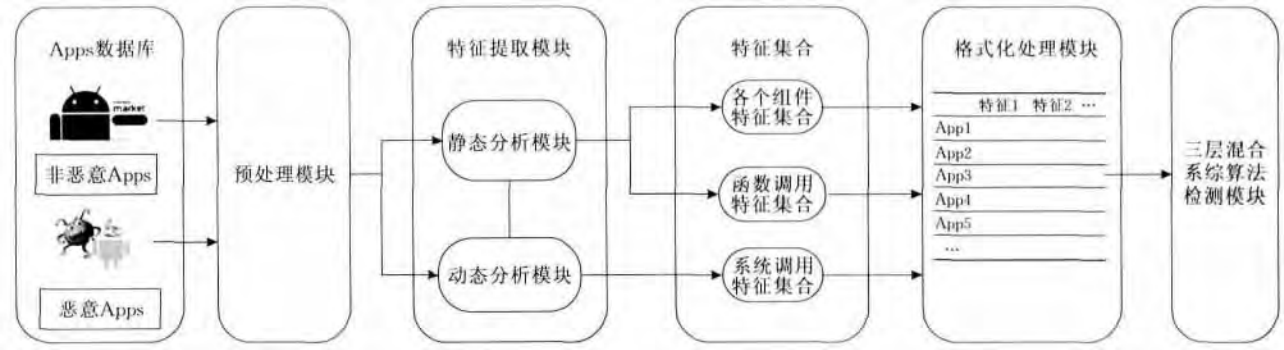

图 1 Androdect 检测系统框图

4 关键技术与算法

4.1 特征提取技术

本文结合了动静态行为分析技术，提取 Android 应用的 3 种不同类型特征，下面分别作详细介绍.

4.1.1 静态分析技术

系统中的静态分析模块采用的是静态分析技术，使用 Android SDK(Software Development Kit，软件开发工具包)中自带的 AAPT(Android Asset Packaging Tool，Android 资产打包工具)对每个 Android 应用(即 APK)进行解压缩，提取 AndroidManifest. xml 文件和 lib 库文件(.so 文件). 每个应用程序都包含一个 AndroidManifest. xml 文件，位于根目录下，它定义了应用程序的内容和行为. 静态分析模块主要完成以下 3 个工作，如图 2 所示.

(1) 对 Android 应用进行自动化分析，提取其中的 package name 和 launchable activity 为动态分析提供参数.

(2) 对 AndroidManifest.xml 文件进行自动化分析，提取各个组件使用个数情况，包括 permission、

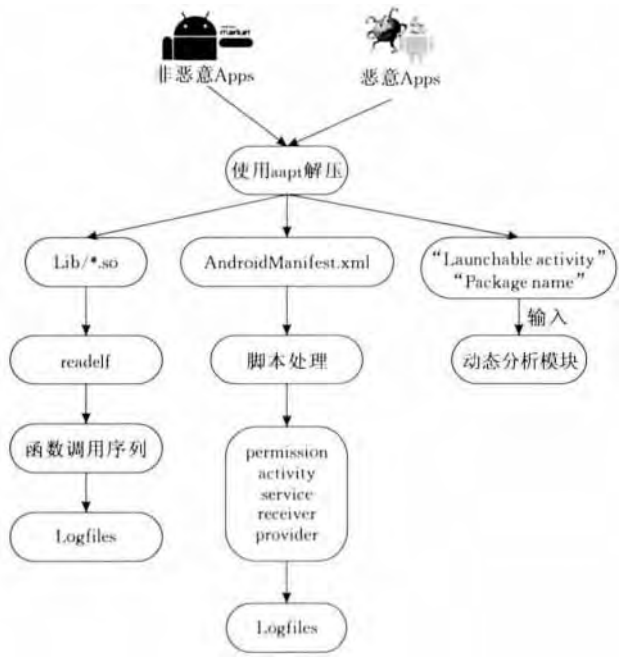

图 2　静态分析流程图

activity、service、receiver 和 provider[①]. Permission 描述了该应用程序请求的权限，为了保护设备上的资源和功能. Activity 通常表示应用程序的一个用户界面. Service 类似于 Windows 平台的服务，它是可能长时间运行的后台进程. Receiver 使得应用程序可以接收系统或者其它应用程序的 Intent 消息. 内容 provider 定义了一种标准机制来共享数据，允许应用程序使用其它应用程序的数据. 例如：随机挑选几个 APK 进行静态分析，结果如表 1 所示，cn.cntv.apk 应用程序声明了 10 个 permission、20 个 activity 和 2 个 service 组件，但是没有使用 receiver 和 provider 组件.

表 1　APK 中各个组件使用统计数据

	permission	activity	service	receiver	provider
cn.cntv.apk	10 个	20 个	2 个	0 个	0 个
bubei.tingshu.apk	14 个	59 个	4 个	1 个	0 个
com.meitu.kankan.apk	16 个	32 个	1 个	0 个	0 个
…	…	…	…	…	…

(3) 使用 Android NDK(Native Development Kit，原生软件开发工具包)中的 arm-linux-androideabi-readelf.exe 提取 native 代码编译链接后生成的 ELF 文件的函数调用序列. 可执行链接格式 ELF (Executable and Linking Format)目标文件有 3 种类型：可重定位文件、可执行文件和共享目标文件. 共享目标文件即 lib 文件夹下的.so 文件，它是 Android native 代码所涉及的文件，用于和其它共享目标文件或者可重定位文件一起生成 ELF 目标文件或者和执行文件一起创建进程映像. 本文对每个应用程序的多个.so 文件进行合并后使用 readelf 工具提取".dynsym"section 中保存的动态符号表"Symbol Table". 并且所抽取的信息选取"FUNC"类型，不能选取"OBJECT"等其它类型，因为函数调用必须是从外部调用、共享库或者系统调用的函数. 对数据库中的应用程序进行统计，如表 2 所示，可以得出恶意应用中含 native 代码的应用程序占 47%，非恶意应用中含 native 代码的应用程序占 33%.

表 2　APK 中包含 native 代码的统计数据

	总的个数/个	含 native 代码的个数/个	百分比/%
恶意应用程序库	1126	531	47
非恶意应用程序库	2000	651	33

4.1.2　动态分析技术

系统中动态分析模块使用动态分析技术，本文主要采用沙盒技术，流程图如图 3 所示.

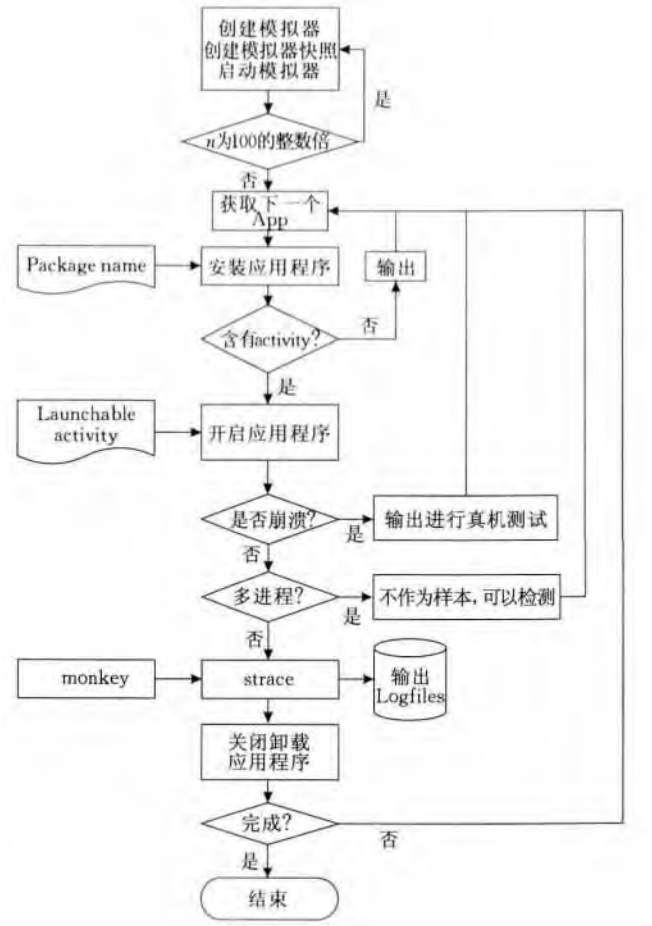

图 3　动态分析流程图

① The AndroidManifest. xml File [EB/OL]. http://developer. android. com/guide/topics/manifest/manifest-intro. html. 2013-03-19

步骤 1. 创建模拟器快照并从快照启动.

步骤 2. 使用静态分析获得的 package name 参数安装应用程序.

步骤 3. 使用静态分析获得的 launchable activity 参数开启应用程序.

步骤 4. 使用 SDK 自带的工具 monkey 模拟用户行为.

步骤 5. 使用 SDK 自带的工具 strace 提取系统调用序列.

步骤 6. 关闭并卸载应用程序.

以前研究者也做过此类工作,但都没有给出具体实现方案和在现实环境下的测试结果.本文根据实际工作提出此过程中的一些具体问题和相应的解决方案,具体介绍如下:

(1)模拟器快照.为了加快系统运行速度,在创建模拟器后创建模拟器快照,每运行整百个应用程序后重新启动一次模拟器,以免重复安装卸载应用程序造成垃圾文件过多影响运行速度,也避免每处理一个应用程序重启模拟器一次影响运行速度.

(2)是否使用 activity 组件.判断应用程序是否使用 activity 组件,因为没有 activity 组件无法使用 monkey 工具提取系统调用特征.因此,本文构建的系统会自动提取出没有使用 activity 组件的应用程序,而后使用静态分析技术来弥补该缺点.

(3)模拟器下无法运行.开启应用程序后要判断其是否崩溃,部分开发者为了防止应用程序在模拟器中运行,会使应用程序开启后自动崩溃.本文提出的系统在运行过程中会输出这些应用程序,后面在真机上依次进行同样的系统调用特征提取.

(4)多进程.有些应用程序含有多个进程,一个进程为正常行为,另一个进程为恶意行为,由于无法预先判断某应用程序中哪个进程为恶意行为,哪个进程为正常行为.因此,本系统不把此类应用程序作为训练样本,通过对每个进程分别提取系统调用特征可以检测此类应用程序,只要含有恶意行为特征就判定为恶意应用.

4.2 三层混合系综算法

表 3 列出了之前研究者使用分类聚类算法进行恶意代码检测所选取的特征和算法.

表 3 相关工作中选取特征及使用的算法对比

检测工具	选取特征	使用的算法
DroidMat[17]	权限、Intent、API	K-means/EM、kNN/NB
Andromaly[15]	API	NB、BN、DTJ48、Histogram、K-means、Logistic
Crowdroid[14]	系统调用	K-means
文献[12]中工具	函数调用	Centroid Machine、NB、SVM
文献[13]中工具	函数调用	PART、Prism、kNN

DroidMat[17]提取权限、Intent、API 特征,使用聚类算法进行聚类而后使用分类算法检测恶意应用,对比了两种聚类算法 K-means 和 Expectation Maximization(EM)以及两种分类算法 k-Nearest Neighbor(kNN)和 NaïveBayes(NB)的检测结果.Andromaly[15]提取了 API 特征,对比了 6 种数据挖掘算法 NB、Bayesian Networks(BN)、Decision Tree(DT)J48、Histogram、K-means 和 Logistic 的检测结果.Crowdroid[14]针对系统调用特征进行提取,使用 K-means 聚类算法进行检测.文献[12]针对函数调用特征进行提取,对比了 Centroid Machine、NB、Support Vector Machine(SVM) 3 种分类算法的检测结果.文献[13]针对函数调用特征进行提取,对比了 PART、Prism、kNN 3 种分类算法的检测结果.

由上面的分析结果可以提出以下 3 个问题:

(1)针对同一类型特征选用不同的算法,检测效果不同,那么,哪一种算法最优呢?

(2)针对不同类型的特征,使用同一算法检测,会不会有一种算法总表现为最优呢?

(3)如何充分发挥不同类型 Android 特征在恶意代码检测上所起的不同作用呢?

以上相关研究的检测结果说明使用传统分类聚类算法无法预先知道哪个算法对该特征检测效果最优;也没有一种固定的算法对不同类型的特征表现效果总是最优;并且单一算法不能充分发挥不同类型 Android 特征在恶意代码检测上所起的不同作用.

因此,针对以上提出的问题,本文设计了一种针对多类特征基于多种基础分类器的三层混合系综算法 THEA(Triple Hybrid Ensemble Algorithm).

4.2.1 相关概念

定义 1. 分类器系综[18].分类器系综是一种从根本上提升分类精度的方法,是将 k 个学习得到的模型系列 $M_1, M_2, \cdots, M_k$ 组合起来,创建一个改进的复合模型 M^*.传统的系综方法有装袋(Bagging)和提升(Adaboost).装袋是对每个模型给出等权重预测;提升是初始时赋予每个元组同样的权重,每得到分类器 M_i 之后,更新权重,使得后面的分类器 M_{i+1} 更关注 M_i 误分类的训练元组.

定义 2. k 折交叉确认(k-fold cross-validation)[18].k 折交叉确认就是将初始数据随机划分为 k 个互不相交的子集,也叫"折" $D_1, D_2, \cdots, D_k$,每个折的大小大致相等.训练和检验进行 k 次.在第 i 次

迭代，划分 D_i 用作检验集，其余的划分一起用来训练模型.

4.2.2　THEA

图 4 为本文设计的三层混合系综算法 THEA 框图. 具体思想为，因为无法预先知道哪个算法对特定类型特征表现效果最优，为了不盲目地随机选择一个算法进行检测，本算法在第 1 层设计采用多个基础分类器分别进行检测，选出最优分类器. 并且针对不同类型的特征选择效果最优的分类器，不是盲目地使用一种分类器对所有类型特征进行检测. 在第 2 层为了使检测效果更优，本算法使用了系综算法分别对选取的 3 个最优分类器的分类效果进行提升. 最后，在第 3 层对一个应用程序的 3 种不同类型特征进行一个综合评判，使用计算加权平均值的判决算法给出检测结果.

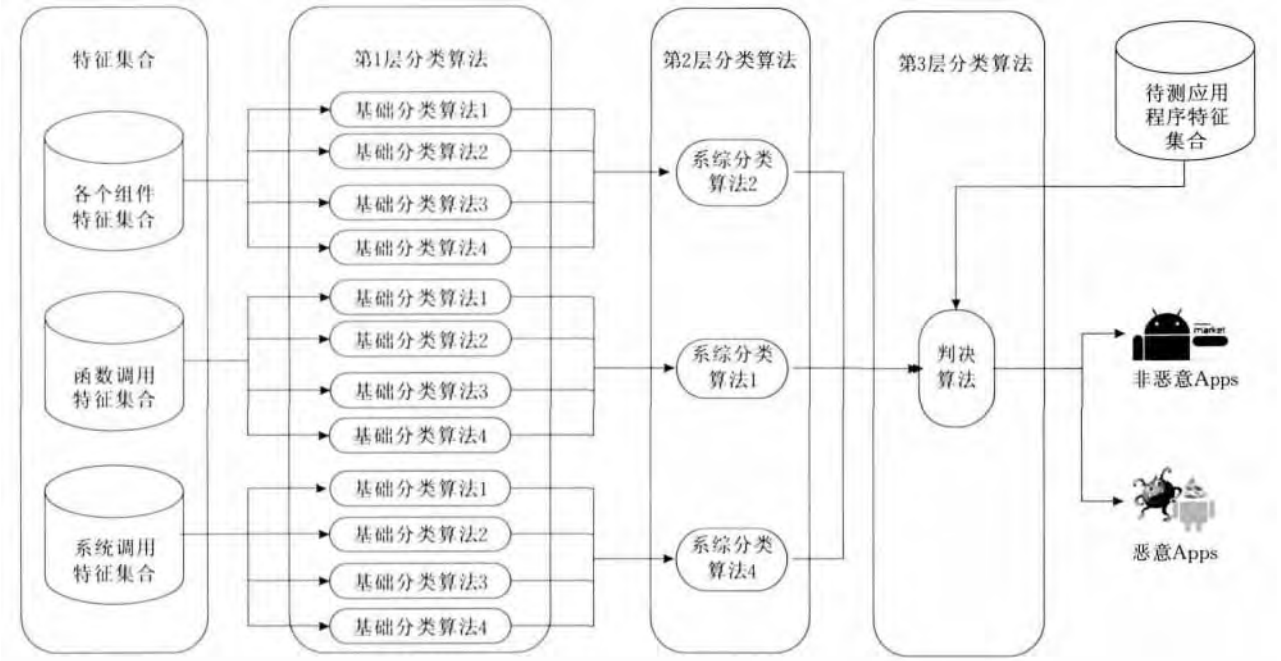

图 4　三层混合系综算法 THEA

下面对算法 1 的流程做详细介绍：

第 1 层对 3 个不同类型特征库 $D_i(i=1,2,3)$ 使用多个基础分类算法 $M_m(m=1,2,3,4)$ 依次采用 k 折交叉确认方法进行训练，此处 m 选取 4，k 选取 10. 在一种特征集合中，每一个基础分类算法 $M_m(m=1,2,3,4)$ 进行 10 折交叉确认，生成的分类器可以用矩阵描述为

$$\boldsymbol{A}=\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{k1} & c_{k2} & \cdots & c_{km} \end{bmatrix},$$

每列取 k 次检测的平均值作为最后的分类器. 对 3 个不同的特征集合 $D_i(i=1,2,3)$ 依次迭代 $M_m(m=1,2,3,4)$ 基础分类算法，生成的分类器可以用矩阵描述为

$$\boldsymbol{B}=\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1i} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mi} \end{bmatrix},$$

在每一列中选取最优的分类器，共选取 i 个分类器.

最后，相当于针对每种类型特征集合输出一个最优分类器 $C_i(i=1,2,3)$. 本文选取的基础分类算法为 NB、kNN、PART、J48[2]，由于系统具有可扩展性，还可以加入更多基础分类算法. 关键是基础分类算法要选取差异大的，这样有利于分类器的效果.

第 2 层使用 RotationForest 算法提升第 1 层输出的 3 个最优分类器的分类效果. RotationForest 算法[19]是利用特征提取来构造基分类器差异性的集成学习方法，该算法的基分类器只能选定一种算法，但是经过变换产生多个基分类器，它们不仅精度高而且差异大，该算法的设计者也证实其特性比传统的 Bagging、Adaboost 系综算法分类效果更优. 因此，本文选用 RotationForest 算法在第 2 层输出 3 个改进的最优分类器.

第 3 层对待检测应用程序提取 3 种类型的特征，使用第 2 层输出的 3 个最优分类器进行分类，最后使用 Voting 函数求加权平均值给出判决结果. 本文选取 0 代表非恶意应用，1 代表恶意应用，每个分

类器给出的结果 $Result$ 为 0 到 1 之间某个值(例如 0.8,表示将该应用程序判定为恶意应用的概率为 80%,判定为非恶意应用程序的概率为 20%),该值体现了该类特征判定结果的权重,越接近 1 则表示根据该类特征将待检测的未知应用程序判定为恶意应用的权重就越大.然后将每种特征的结果求加权平均值得出最后判断结果.Voting 函数公式定义如下:

$$\text{加权平均值}=\frac{x\times Result_1+y\times Result_2+z\times Result_3}{x+y+z},$$

$x,y,z\in(0,1)$; $0\leq Result_1,Result_2,Result_3\leq 1$,

其中,x,y,z 表示该应用程序是否含有该类特征,1 代表有,0 代表无;$Result$ 表示根据各类特征的判定结果.

算法 1. THEA.

输入:

D_i: 3 种类型训练样本的集合,$i=1,2,3$

D_{ij}: 每种训练样本集合被等分为 k 个元组,$j=1,2,\cdots,k$

k: 轮数(每轮产生一个分类器)

M_m: 基础分类算法,$m=1,2,3,4$

C: 生成的分类器

Cal: 计算 $w=Accuracy$ 的值

RF: RotationForest 算法

输出: C_{THEA} 三层混合系综分类器,$Result$

过程 1. 生成 THEA 分类器并给出分类结果

//第 1 层分类算法

FOR each D_i in D DO

  FOR each M_m in M DO

    Divide D_i to k parts

    FOR $j=1$ to k DO

      $Train(D_{i1},D_{i2},\cdots,D_{i(j-1)},D_{i(j+1)},\cdots,D_{ik},M_m)$

      $Emit\ C_{jm}$

      $Test(D_{ij},C_{jm})$

    Output C_m with Mean value

    $Cal(w_m)$

  Set $w_i=\text{Max}(w_1,w_2,w_3,w_4)$

  Output C_i with w_i

Output C_1,C_2,C_3

//第 2 层分类算法

FOR each D_i in D DO

  Divide D_i to k parts

  FOR $j=1$ to k DO

    $Train(D_{i1},D_{i2},\cdots,D_{i(j-1)},D_{i(j+1)},\cdots,D_{ik},RF,C_i)$

    $Emit\ RF(C_i)_j$

    $Test(D_{ij},RF(C_{ij}))$

  Output $RF(C_i)_i$ with Mean value

Output $RF(C_1)_1,RF(C_2)_2,RF(C_3)_3$

//第 3 层分类算法

Read new test D_n

FOR each $RF(C_i)_i$ in $RF(C)$ DO

  $Test(D_n,RF(C_i)_i)$

  Output $Result_i$

Voting ($Result_i,Result_2,Result_3$)

Output ($C_{\text{THEA}},Result$)

5 实验设计与结果分析

本文通过实验来评估 Androdect 系统的有效性,与之前的相关工作进行对比,并重点与国外著名的 Android 恶意代码检测工具 Androguard 使用相同的测试集进行对比分析.

5.1 实验环境

本文使用 Java 和 Python 语言实现了自动化检测工具 Androdect.其中,动静态行为特征提取部分主要由 Python 语言实现,APK 捕获和三层混合系综算法实现主要由 Java 语言实现.使用 Androdect 对现实中 1126 个恶意程序和 2000 个非恶意程序进行检测.所有实验在内存为 4 GB,处理器为 Intel (R) Core(TM)2 Quad 2.67 GHz 的机器上完成.Androguard 测试实验在 1 GB 内存的 Virtual Box 虚拟机上完成.为了对比,Androdect 的测试实验同样在 1 GB 内存的虚拟机上完成.

5.2 实验结果

在介绍本实验结果之前,先介绍几个用来衡量分类器性能的评估参数[18].

定义 3. 混淆矩阵(confusion matrix).混淆矩阵是分析分类器识别不同类元组情况的一种有用工具,如表 4 所示.本文将非恶意应用定义为正元组,恶意应用定义为负元组.True positives(t_pos)指分类器将非恶意应用正确识别为非恶意应用的元组;True negatives(t_neg)指分类器将恶意应用正确识别为恶意应用的元组;False positives(f_pos)指分类器将恶意应用错误识别为非恶意应用的元组;False negatives(f_neg)指分类器将非恶意应用错误识别为恶意应用的元组.

表 4 混淆矩阵

预测为 ↓	非恶意应用	恶意应用
非恶意应用	t_pos	f_pos
恶意应用	f_neg	t_neg

由混淆矩阵计算出相应的分类器评估参数的公式为

$$Accuracy=\frac{t\_pos+t\_neg}{t\_pos+t\_neg+f\_pos+f\_neg},$$

$$TP\ Rate=Recall=\frac{t\_pos}{t\_pos+f\_neg},$$

$$FP\ Rate=\frac{f\_pos}{f\_pos+t\_neg},$$

$$Precision=\frac{t\_pos}{t\_pos+f\_pos},$$

$$F\text{-measure}=\frac{2\times recall\times precision}{recall+precision}.$$

定义 4.　*ROC* 曲线(Receiver Operating Characteristic 接收者运行特征). *ROC* 曲线显示了给定模型的 True positives 概率与 False positives 概率之间的比较评定.

定义 5.　*AUC*(Area Under Curve). *ROC* 曲线下面的面积是模型准确率的度量 *AUC*. 为了评估模型的准确率,可以测量曲线下方的面积,面积越接近 0.5,对应模型的准确率越低,完全准确的模型的面积为 1.

第 1 层分类使用选定的基础分类器 PART、NB、kNN、J48 对 3 种类型特征采用 10 折交叉检验测试,结果如表 5～表 7 所示. 由测试结果可以看出,针对各个组件特征,*k*NN 算法表现最优;针对函数调用特征,PART 算法的 TP Rate 值最高,*k*NN 算法的 *AUC* 值最高. 由于函数调用特征数量较多,本文采用特征选择算法对原有特征进行筛选,目的是选择对分类器影响较大的 255 个特征,同时降低维度减少计算时间. 具体使用 InfoGainAttributeEval 评价策略方法(根据与分类有关的每一个属性的信息增益进行评估)和 Ranker 搜索策略方法(属性判据值排序);针对系统调用特征,J48 算法的 *TP Rate* 值最高,PART 算法的 *AUC* 值最高. 由于 *Accuracy* 指标综合考虑了混淆矩阵的 4 个分量,本文使用 *Accuracy* 评估分类器性能,表 8 给出了各种特征集合各个基础分类器 *Accuracy* 值的结果. 由此得出,*k*NN 算法为针对各个组件特征的最优分类器,PART 算法为针对函数调用特征的最优分类器,J48 算法为针对系统调用特征的最优分类器.

表 5　各个组件特征分类结果

	TP Rate	*FP Rate*	*Precision*	*Recall*	*F*-Measure	*AUC*
PART	0.872	0.136	0.874	0.872	0.873	0.936
NB	0.777	0.188	0.808	0.777	0.781	0.870
*k*NN	0.906	0.093	0.908	0.906	0.906	0.938
J48	0.898	0.117	0.898	0.898	0.898	0.923

表 6　函数调用特征分类结果

	TP Rate	*FP Rate*	*Precision*	*Recall*	*F*-Measure	*AUC*
PART	0.975	0.026	0.975	0.975	0.975	0.977
NB	0.91	0.092	0.91	0.91	0.91	0.969
*k*NN	0.972	0.028	0.972	0.972	0.972	0.985
J48	0.97	0.033	0.97	0.97	0.969	0.979

表 7　系统调用特征分类结果

	TP Rate	*FP Rate*	*Precision*	*Recall*	*F*-Measure	*AUC*
PART	0.812	0.25	0.809	0.812	0.808	0.854
NB	0.649	0.486	0.628	0.649	0.628	0.603
*k*NN	0.755	0.285	0.755	0.755	0.755	0.715
J48	0.818	0.216	0.817	0.818	0.817	0.814

表 8　基础分类器的 *Accuracy* 值

	各个组件特征	函数调用特征	系统调用特征
PART	87.17	97.46	81.16
NB	77.67	91.02	64.88
*k*NN	90.56	97.21	75.46
J48	89.83	96.95	81.77
最优算法	*k*NN	PART	J48

第 2 层分类使用 RotationForest 算法对第 1 层选取的 3 个最优分类器使用 10 折交叉确认重新训练,得出表 9. 由此看出,经过 RotationForest 算法后,分类器效果得到了明显的改善.

表 9　第 2 层系综算法分类结果

	TP Rate	*FP Rate*	*Precision*	*Recall*	*F*-Measure	*AUC*
各个组件特征	0.912	0.09	0.914	0.912	0.913	0.948
函数调用特征	0975	0.027	0.976	0.975	0.975	0.988
系统调用特征	0.874	0.164	0.873	0.874	0.873	0.945

第 3 层分类使用第 2 层输出的 3 个最优分类器模型并结合判决算法对新的未知应用程序进行检测. 依旧采用 10 折交叉确认对 3126 个应用程序进行测试,得到的混淆矩阵为表 10,准确率为 94.24%. 测试结果充分证明了 Androdect 系统的有效性.

表 10　Androdect 的混淆矩阵

预测为 ↓	非恶意应用/个	恶意应用/个
非恶意应用	1940	120
恶意应用	60	1006

5.3　结果比较

为了评估 Androdect 系统的有效性,本文将提出的方法与近年来的相关工作进行对比,并与国外著名的 Androguard 工具使用相同的测试样本进行对比分析.

5.3.1　与近年来相关工作的对比

表 11 列出了本文提出的方法与近年来的相关

工作的对比情况. DroidMat[17] 提取权限、Intent、API 等特征,使用聚类算法进行聚类而后使用分类算法检测恶意应用,对比了两种聚类算法 K-means 和 Expectation Maximization(EM)以及两种分类算法 k-Nearest Neighbor(kNN)和 NaïveBayes(NB)的检测结果. 该方法将多类特征放在一个集合中,使用单一种类分类算法进行检测. Andromaly[15],动态提取 API 特征,对比使用 3 种特征选择算法 ChiSquare、FisherScore、InfoGain 和 6 种数据挖掘算法 NB、BN、DTJ48、Histogram、K-means、Logistic 的测试结果,缺点一是没有使用真实的恶意应用进行实验验证,仅使用自己编写的 4 个恶意应用进行检测;二是动态提取整个运行环境的 API 特征,没有针对运行中的 APK 进行特征提取;三是测试过程多人工参与. Crowdroid[14] 使用动态方法对系统调用进行特征提取,基于 K-means 聚类算法进行检测,该工具只能区分同一应用程序的各个变种是否为恶意应用. AASandbox[8] 提出了动态提取系统调用特征的总体框架和思想,没有应用数据挖掘算法进行检测,也没有给出验证结果. 文献[12]提出的工具采用静态方法提取函数调用特征,对比了 3 种分类算法 Centroid Machine、NB、SVM 的检测结果,但只针对 Symbian 系统进行了验证. 文献[13]静态提取了函数调用特征,对比了 3 种分类算法 PART、Prism、kNN 在 Android 系统上的检测结果,对 240 个应用进行验证.

表 11 相关工作对比

检测工具	选取特征	动态/静态	使用的算法	系统	测试对象
DroidMat[17]	权限、API、Intent 等特征	静态	K-means/EM 、kNN/NB	Android	1738 个 APK
Andromaly[15]	API 特征	动态	NB、BN、DTJ48、Histogram、K-means、Logistic	Android	4 个自己编写 APK
Crowdroid[14]	系统调用特征	动态	K-means	Android	两类 APK
AASandbox[8]	系统调用特征	动静结合	无	Android	150 个 APK
文献[12]中工具	函数调用特征	静态	Centroid Machine、NB、SVM	Symbian	1000 个 APK
文献[13]中工具	函数调用特征	静态	PART、Prism、kNN	Android	240 个 APK
Androdect	组件、系统调用、函数调用等特征	动静结合	三层混合系综算法	Android	3126 个 APK

综上所述,DroidMat 和 Androdect 系统提取了 Android 应用的多类行为特征,充分反映了 Android 应用恶意行为特征;AASandbox 和 Androdect 系统采用了动静态结合的特征提取技术,弥补了动静态分析技术各自的缺点;除了 Androdect 系统,其它工具都采用单一分类或者聚类算法进行检测;本文设计的实验收集了更多现实中的应用程序进行检测来验证系统的有效性和准确性.

5.3.2 与 Androguard 对比

由于在网上可以很容易获得 Androguard 工具,本文使用同样的测试样本对 Androguard 以及 Androdect 进行了测试. 使用 Androguard 官方网站提供的 Ubuntu 系统镜像 ARE(Android Reverse Engineering),ARE 默认安装的 Androguard 版本过低,作者没有更新,本文将其进行更新,并下载相关依赖库重新编译,更新至 2013 年 4 月最新版本 Androguard-1.9. 但是最新版本 Androguard-1.9 以及 Androguard-1.6 对于 APK 的解析存在问题,有很多不能处理的应用程序,对恶意应用的检测效果很差,反而较低版本更加稳定. 并且发现更新版本的 Androguard 签名库并没有发生变化,只是加入了更多的程序分析功能,对本实验没有影响. 因此本文还是与 ARE 默认安装的 Androguard 版本进行对比.

对比发现,Androguard 处理大量应用程序时无法完全自动化,并有部分应用程序不能正确处理,进程可能被意外杀死,报错退出,或者停止执行. 而且运行时间慢,平均一个应用程序大约需要运行 2.5 min,程序意外中止后需人工发现重启程序,排除运行中的间隔时间,运行 3126 个应用程序需要耗费 5.5 天时间. 具体的对比数据如表 12 所示,测试结果使用混淆矩阵说明,如表 13、表 10 所示. 由此得出,Androdect 在准确率上高于 Androguard,并且运行时间短,可以处理几乎所有的应用程序.

表 12 Androdect 与 Androguard 测试数据对比

	准确率/%	运行时间/天	不能处理的 APK/个	总 APK/个
Androguard	90.74	大于 5.5	166	3126
Androdect	94.24	2.1	0	3126

表 13 Androguard 的混淆矩阵

预测为 ↓	非恶意应用/个	恶意应用/个
非恶意应用	1834	274
恶意应用	0	852

5.3.3 结 论

本文提出的系统结合了上述研究方法的优点,改进了它们的缺点,主要优点有以下几个方面:

(1)以往的研究方法大多提取单一种类特征,

本文提出的系统提取 3 种不同类型特征，充分反映了应用程序行为，并且该系统具有可扩展性，可以加入更多类型特征进行检测.

(2) 采用动静态结合的特征提取方法，动态方法弥补了静态方法不能检测运行中释放恶意代码的缺点，绕过了代码加密混淆问题；静态方法为动态方法提供了运行参数，解决了动态方法无法处理的应用程序.

(3) 首次设计了三层混合系综算法 THEA，可以充分考虑不同类型特征在 Android 恶意代码检测中所起的不同作用，对各类特征分别选取最优基础分类器并给出综合判决结果，实验证明了该算法的有效性.

(4) 对大量现实中的应用程序进行检测，可以处理几乎所有的应用程序，并且实现了完全自动化.

6　讨　论

第 5 节介绍了 Androdect 系统与其它相关工作比较的主要优点，但是该系统仍然存在一些局限性. 下面给出该系统的局限性和下一步工作方向：

(1) 本文提出的 Android 恶意应用检测系统具有一定的扩展性，主要表现在两个方面. 一是特征类型方面，可以加入更多类型的特征，不仅局限于本文提出的 3 类特征. 例如，可以加入具体的权限使用情况、上层 API 使用情况等特征；二是基础分类器的选择，本文强调了基础分类器的选择应选取差异较大的不同类型的分类器，也可以加入 SVM 等精度更高的分类器作为基础分类器，本文只选取了之前研究者通常使用的四个分类器，下一步工作中将加入分类精度更高的分类器.

(2) 在 Android 恶意应用的检测研究中本文提出分别针对不同特征做分类还出于以下几个原因：① 不是所有的 Android 应用程序都包含本文提取的全部 3 种类型特征，部分 Android 应用程序不存在本文提取的函数调用特征，部分 Android 应用程序没有使用组件，所以无法将所有特征混在一起考虑. ② 一个 Android 应用程序的组件特征只有 5 个，而系统调用特征可能使用 100 多个，函数调用特征可能使用上千个，如果将所有这些特征融合在一起考虑，那么组件类的 5 个特征可能会被淹没. 并且对于不同类型的特征，本文衡量的数据不同，针对组件特征本文考虑的是每个应用程序使用每个组件的个数；针对系统调用特征本文考虑的是每个应用程序使用每个函数调用的次数；针对函数调用特征本文考虑的是每个应用程序使用了哪些函数调用. ③ 本文提出的检测系统的时间消耗主要在各类特征提取和格式化处理部分，而分类器的训练和检测上执行时间很短，通常在几秒或几分钟内完成，并不会牺牲很多时间.

(3) 本文中的动态行为特征提取模块使用的是 Monkey 工具[①]，它是 Android 中的一个命令行工具，可以在模拟器或实际设备中运行. 它向系统发送伪随机的用户事件流(如按键输入、触摸屏输入、手势输入等)，模拟人类使用应用程序的动作. 后面的工作可以参考文献[20]所做的研究，在动态模拟事件时基于 UI(User Interface)智能地有规律地触发事件. 但是，该文献也指出，此研究没有在大量应用程序上做测试，因为基于 UI 触发恶意行为的应用程序占少数.

(4) 针对多类特征的三层混合系综算法可以结合云计算平台进行更快速的处理，进一步研究实现在线实时数据检测，预防现有的 update 攻击. Update 攻击就是在应用程序提示更新后，用户确认并下载安装含有恶意代码的更新程序.

(5) 本文实验中的恶意样本已经是广为流传的恶意应用，各大杀毒软件已经在各自的病毒库中包含了这些病毒，所以检测率会比较高. 手机上的杀毒软件对已知病毒的查杀率也比较高. 根据独立安全机构 Av-test 在 2013 年 1 月使用 ESET、Lookout：Antivirus&Security、Kaspersky、Qihoo360 等 22 个手机杀毒软件对 869 个已知 Android 恶意应用的检测报告显示，各大手机杀毒软件的平均查杀率为 94%，与本文工具的准确率 94.24%相平. 而本文主要研究的是对未知恶意应用的检测，在此方面，各大厂商没有公开自己的杀毒软件的技术细节，本文无法做详细分析.

7　总　结

本文采用动静态结合的技术提取 Android 应用的多类行为特征，通过设计三层混合系综算法 THEA 建立检测模型，并实现检测工具 Androdect 进行恶意代码检测. 一方面弥补了动态分析和静态分析各自的缺点，另一方面充分考虑了不同类型特

① UI/Application Exerciser Monkey [EB/OL]，http://developer.android.com/tools/help/monkey.html. 2013-04-01

征在 Android 恶意行为检测中所起的不同作用，对各类特征分别选取最优算法并给出综合判决结果. 使用 Androdect 对现实中 1126 个恶意应用和 2000 个非恶意应用进行测试. 实验结果表明，Androdect 在准确率和执行效率上表现良好，优于其它著名的恶意代码检测工具. 下一步工作将研究加入更多类型行为特征以及精度更高的基础分类算法，在动态阶段模拟更精确的用户触发行为，并对三层混合系综算法结合云计算实现更快速的处理，进一步完善系统.

致 谢 感谢我的导师、师兄以及我们的团队给予我的所有帮助，还要感谢给予此论文意见和帮助的各位审稿专家和编辑，最后感谢我的家人给予我的大力支持！

参 考 文 献

[1] Peng Hao, Gates Chris, Sarma Bhaskar, et al. Using probabilistic generative models for ranking risks of Android apps//Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS'12). Raleigh, NC, USA, 2012: 241-252

[2] Witten Ian H. Data Mining: Practical Machine Learning Tools and Techniques. 3rd Edition. Beijing: China Machine Press, 2012

[3] Li Hai-Feng, Zhang Ning, Zhu Jian-Ming, Cao Huai-Hu. Frequent itemset mining over time-sensitive streams. Chinese Journal of Computers, 2012, 35(11): 2283-2293(in Chinese)
(李海峰，章宁，朱建明，曹怀虎. 时间敏感数据流上的频繁项集挖掘算法. 计算机学报，2012，35(11)：2283-2293)

[4] Zheng Li-Ming, Zou Peng, Jia Yan, Han Wei-Hong. How to extract and train the classifier in traffic anomaly detection system. Chinese Journal of Computers, 2012, 35(4): 719-730(in Chinese)
(郑黎明，邹鹏，贾焰，韩伟红. 网络流量异常检测中分类器的提取与训练方法研究. 计算机学报，2012，35(4)：719-730)

[5] Zhou Ya-Jin, Jiang Xu-Xian. Dissecting Android malware: Characterization and evolution//Proceedings of the 33rd IEEE Symposium on Security and Privacy. Oakland, USA, 2012: 95-109

[6] Wang Fei-Fei. Study on detection and protection techniques of mobile phone malicious code under the Android platform [M. S. dissertation]. Beijing Jiaotong University, Beijing, 2012(in Chinese)
(王菲飞. 基于 Android 平台的手机恶意代码检测与防护技术研究[硕士学位论文]. 北京交通大学，北京，2012)

[7] Lu Cheng. Design and implementation of malwares detection system on Android [M. S. dissertation]. Beijing University of Posts and Telecommunications, Beijing, 2012(in Chinese)
(路程. Android 平台恶意软件检测系统的设计与实现[硕士学位论文]. 北京邮电大学，北京，2012)

[8] Bläsing Thoma, Batyuk Leonid, Schmidt Aubrey-Derrick. An Android application sandbox system for suspicious software detection//Proceedings of the 5th International Conference on Malicious and Unwanted Software (Malware 2010). Nancy, Lorraine, France, 2010: 55-62

[9] Enck Willian, Gilbert Peter, Chun Byung-Gon, et al. TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones//Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10). Vancouver, Canada, 2010: 1-6

[10] Shabtai Asaf, Fledel Yuval, Elovici Yuval. Automated static code analysis for classifying Android applications using machine learning//Proceedings of the 2010 International Conference on Computational Intelligence and Security (CIS'10). Nanning, China, 2010: 329-333

[11] Padriya Nitin, Mistry Nilay. Review of behavior malware analysis for Android. International Journal of Engineering and Innovative Technology (IJEIT), 2013, 2(7): 230-232

[12] Schmidt Aubrey-Derrick, Clausen Jan Hendrik, Camtepe Ahmet, Albayrak Sahin. Detection Symbian os malware through static function call analysis//Proceedings of the 4th IEEE International Conference on Malicious and Unwanted Software (Malware 2009). Montréal, Canada, 2009: 15-22

[13] Schmidt Aubrey-Derrick, Bye Rainer, Schmidt Hans-Gunther, et al. Static analysis of executables for collaborative malware detection on Android//Proceedings of the 2009 IEEE international conference on Communications (ICC'09). Dresden, Germany, 2009: 631-635

[14] Iker Burguera, Urko Zurutuza, Simin Nadjm-Tehrani. Crowdroid: Behavior-based malware detection system for Android//Proceedings of the ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM'11). Chicago, USA, 2011: 15-26

[15] Shabtai Asaf, Kanonov Uri, Elovici Yuval, et al. Andromaly: A behavioral malware detection framework for Android devices. Journal of Intelligent Information Systems, 2012, 38(1): 161-190

[16] Grace Michael, Zhou Yajin, Zhang Qiang, et al. RiskRanker: Scalable and accurate zero-day Android malware detection//Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys'12). Low Wood Bay, Lake District, United Kingdom, 2012: 281-294

[17] Wu Dong-Jie, Mao Ching-Hao, Wei Te-En, et al. DroidMat: Android malware detection through manifest and API calls tracing//Proceedings of the Seventh Asia Joint Conference on Information Security (AsiaJCIS 2012). Tokyo, Japan, 2012: 62-69

[18] Han Jia-Wei, Kan Bo write, Fan Ming, Meng Xiao-Feng translation. Data Mining Concepts and Techniques. 2nd Edition. Beijing: China Machine Press, 2007(in Chinese)
(韩家炜，堪博著，范明，孟小峰译. 数据挖掘概念与技术. 第 2 版. 北京：机械工业出版社，2007)

[19] Rodriguez Juan J, Kuncheva Ludmila I, Alonso Carlos J. Rotation forest: A new classifier ensemble method. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006, 28(10): 1619-1630

[20] Zheng Cong, Zhu Shi-Xiong, Dai Shuai-Fu, et al. SmartDroid: An automatic system for revealing UI-based trigger conditions in Android applications//Proceedings of the 2nd ACM Workshop on Security and Privacy in Smartphones and Mobile Devices(SPSM'12). Raleigh, USA, 2012: 93-104

YANG Huan, born in 1984, Ph. D., assistant engineer. Her research interests include information security, vulnerability discovery and malicious code detection.

ZHANG Yu-Qing, born in 1966, Ph. D., professor. His research interests include network and information system security.

HU Yu-Pu, born in 1955, Ph. D., professor. His research interests include stream cipher and the block cipher, the network security protocol design and analysis.

LIU Qi-Xu, born in 1984, Ph. D. His research interests include information security, vulnerability discovery and vulnerability evaluation.

Background

This paper mainly researches on applying the data mining algorithm to the malicious code detection of Android application, which belongs to the security of the mobile intelligent terminals. By now, there are some malicious code detection techniques, such as signature-based detection technology, behavior-based detection technology. Traditional signature-based detection technique is commonly used, but it should have a class of signature database of malicious applications before to detect such kind of malicious applications. While behavior-based detection technology can detect unknown malicious applications, which becomes a hot topic in recent years, and mainly used methods are dynamic and static analysis. In addition, data mining technology can extract meaningful information from large amounts of data, so malicious code detection technology based on data mining algorithms has gradually transferred from PC platforms to mobile platforms. First it uses static analysis or dynamic analysis to extract apps' behavior features and then uses traditional data mining algorithm to classify app to determine whether it contains malicious code. But static analysis and dynamic analysis have their own shortcomings and difficulties, and traditional data mining algorithms have different performance characteristics.

Therefore, this paper combines the static and dynamic methods to extract three classes of Android behavioral features, designs an ensemble algorithm named THEA (Triple Hybrid Ensemble Algorithm), which can handle many classes of features, and realizes a detection tool named Androdect to detect malicious code of 1126 malicious and 2000 non-malicious applications in real. The experimental results show that the approach performs better then the other malicious code detection tools of Android applications on the availability, efficiency and accuracy.

This paper is mainly supported by projects of the National Natural Science Foundation of China (No. 61272481), China Postdoctoral Science Foundation (No. 2011M500416, No. 2012T50152), the Beijing Natural Science Foundation (No. 4122089) and National Development and Reform Commission (NDRC) Special Information Security (High-Tech Development and Reform Commission No. [2012]1424). The National Natural Science Foundation of China (No. 61272481) mainly researches on the security of the mobile intelligent terminals. This paper is dedicated to detect malicious code of Android apps, which is a very important part of that project.