Android 移动应用检测研究

刘 玮[1] 李蜀瑜[2]

[1](重庆师范大学涉外商贸学院数学与计算机学院 重庆 410520)

[2](陕西师范大学计算机科学与技术学院 陕西 西安 710062)

摘 要 随着 Android 操作系统在智能设备上的广泛应用,Android 应用的安全性检测成为了当前关注的重点。为了从 Android 应用程序中检测出恶意软件,研究 Android 应用静态分析技术、动态分析技术及基于机器学习的 Android 应用检测技术。提出一个通用的恶意软件检测框架。该框架通过逆向工程从 Android 应用中提取(安全应用、受感染应用)特征信息并建立关键信息特征库。通过机器学习建立检测模型,采用分类检测技术完成检测。通过该检测框架,可在软件安装前执行应用安全评估,其检测正确率高,并具有良好的扩展性,为 Android 应用的安全性检测提供参考。

关键词 安卓 安全性检测 静态分析 动态分析 机器学习

中图分类号 TP314 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2019.06.058

ANDROID MOBILE APPLICATION DETECTION

Liu Wei[1] Li Shuyu[2]

[1](*College of Mathematics and Computer Science, Chongqing Normal University Foreign Trade and Business College, Chongqing 410520, China*)

[2](*College of Computer Science and Technology, Shaanxi Normal University, Xi'an 710062, Shaanxi, China*)

Abstract With the wide application of the Android operating system in smart devices, the security detection of Android applications has become the focus of current attention. In order to detect malware from Android applications, this paper studied Android application static analysis, dynamic analysis and Android application detection technology based on machine learning, and proposed a general framework for malware detection. The framework extracted feature information from Android applications(secure applications, infected applications) by reverse engineering and established key information feature database. We established the detection model through machine learning, and completed the detection through classification detection. The framework can perform application security assessment before software installation. It has high detection accuracy and good expansibility. It provides a reference for the security detection of Android application.

Keywords Android Security detection Static analysis Dynamic analysis Machine learning

0 引 言

随着互联网与智能手机技术的飞速发展,各类手机应用应运而生,深刻地影响着人们的生活。安卓系统由于其开放、自由的特性,得到了开放手机联盟的大力支持,已成为全球智能移动设备中市场占有率最高的操作系统。

Android 是基于 Linux 内核的操作系统,安卓平台的开放性降低了安卓应用开发的难度,任何对安卓开发感兴趣的人员都可通过简单的注册成为安卓市场的一员,并自由发布自己的应用,这导致安卓应用的安全性大大降低。近年来,关于 Android 应用的安全性问题层出不穷,受到人们的广泛关注。许多学者分别从应用程序静态代码检测、应用程序执行流程及应用机器学习的自动化检测等方法对此问题进行了研究[1]。

收稿日期:2018-11-12。重庆师范大学涉外商贸学院校级教改项目(JG2016015)。**刘玮**,讲师,主研领域:嵌入式软件设计。**李蜀瑜**,副教授。

本文基于这三种方法,提出了一种基于机器学习的 Android 检测方法框架,可为 Android 应用的安全性检测提供参考。

1　Android 恶意应用静态检测方法研究

静态分析是在不执行 Android 应用的情况下检测出应用程序中恶意特征的方法。现有的静态分析方法主要集中在对不同类型恶意软件进行分类和检测上。文献[2]提出了一种基于特征树的静态分析方法,该方法通过将 Android 应用调用 API 信息构造成特征树,并设计一种比较算法对树的相似度进行分析,以此实现对 Android 应用的分类并检测恶意应用。文献[3]提出了一种基于贝叶斯分类的恶意软件检测方法,该方法采用逆向工程对 Android 应用进行静态分析。文献[4]提出一种执行静态污染分析的方法,通过解析移动应用程序并构造控制流图(CFG)分析源自敏感源的路径,应用数据流分析器检查从源程序发送到远程服务器的任何敏感数据,而不通知用户。文献[5]使用反编译程序从 Android 应用程序的安装程序生成 Java 源代码,并应用静态代码分析套件评估恢复的源代码,但此方法仅适用于使用相对较少数量的权限和 API 调用的应用程序。由于开发人员在应用市场上传应用时,需要签署证书,并分配给其序列号,因此可通过搜索序列号来追踪应用的包装和修改。这是最简单的一种 Android 应用静态检测方法。

综上所述,静态分析是一种简单、快速的恶意应用检测方法。但大多数静态检测技术很难检测代码执行过程中的恶意行为,例如,应用程序在运行后的代码自动修改及由移动僵尸网络或病毒引起的入侵。

2　Android 恶意应用动态检测方法研究

动态分析是通过执行应用程序来检测漏洞的。动态分析检测技术的核心是通过监视应用程序的动态行为,对污染源或系统调用进行跟踪和观察。与静态分析相比,动态分析更为复杂。

文献[6]提出了应用程序动态污点跟踪的方法,即应用程序调用 Dalvik 虚拟机执行四种类别污点粒度的污染传播:变量、方法、消息和文件,但这种方法只关注数据流,很难保证网络访问的安全。

文献[7]提出了一种基于前向执行能力动态跟踪分析系统 DroidTrace,但该系统需要将智能终端与计算机连接起来,并使用 Linux 命令来收集恶意检测数据,操作较复杂。

文献[8]提出了一种增强的物理架构 CARDODL,用于执行在线恶意软件检测,这种方法是一种硬件解决方案,但检测服务的部署实施缺乏灵活性。

文献[9]提出了从 API 调用、本机代码动态执行和系统调用提取敏感行为特征的检测框架,但由于行为数据的监测和收集要消耗大量的计算能力,所以这种方法会增加额外开销。

静态或动态分析方法检测应用的安全性各有优缺点,静态检测方法花费时间较长,且无法检测应用的交互行为和网络通信安全,而动态检测方法通常需消耗较多的计算资源,可能导致应用运行变慢,因此需要平衡系统检测开销与安全检测性能。大多数学者更倾向于使用二者相混合的移动应用恶意软件检测方法。即首先通过静态分析来检测应用,进一步通过执行应用来进行动态分析,混合分析方法可以实现更高的检测敏感度,如文献[10]提出的移动沙箱 MobileSandbox 法。

尽管应用静态与动态相结合的恶意软件检测方法可以抵抗恶意软件的反检测技术,并且此类方法已经大大改善了 Android 应用检测的效率和准确性,但依然缺乏恶意应用的自动分析和识别能力。如何根据已有的 Android 应用特征信息高效可靠地识别恶意应用是目前 Android 检测急需解决的问题。

3　基于机器学习的恶意软件检测方法

机器学习的概念是让算法自己从数据中学习所需的参数,以便做出最好的预测。目前已有多种机器学习方法用于数据安全性和恶意应用的检测。文献[11]提出了一种新的机器学习框架,该方法首先从应用中提取一些特征,将这些特征在离线模式下进行训练,进而用于恶意应用的检测。Andromaly[12] 是一种基于行为学习的恶意软件检测方法。该方法监视应用软件在移动设备上运行时的特征和事件,然后应用分类器来检测恶意应用程序。Drebin[13] 通过大量收集 Android 应用的特征,并且将这些特征存储用于模式识别,然后基于这些模式对合法应用和非法恶意应用进行判定。文献[14]提出了基于权限的恶意软件检测方法,该方法首先收集来自应用程序的各种权限和动作,然后将分类器应用于所有收集的特征。PMDS(Permission based Malware Detection System)[15] 是另一种基于机器学习的恶意软件检测方法,通过从 Android 应用程序中挖掘权限,基于权限特征进行机器学习,分类识别未知的恶意软件,这种方法的识别准确率已经达到 94%。

由于 Android 是一个开源的、可扩展的平台,因此尽可能多地提取 Android 应用特征,将会极大地丰富检测功能。且基于机器学习的恶意软件检测方法不仅算法稳定、具有可扩展性,还可大大降低检测过程中的资源消耗。

4 基于机器学习的 Android 应用检测框架

4.1 基于机器学习的检测模型框架

基于机器学习的 Android 恶意应用检测方法关键在于获取应用特征信息,并建立机器学习模型,根据模型判定应用的安全性。本文提出一种新的基于机器学习的 Android 应用检测框架,该框架混合了静态检测技术和动态检测技术的优点,可将所有的恶意应用在安装之前检测出来。该框架如图 1 所示,首先,从 Android 应用的 APK 文件以及清单文件(manifest. xml)中提取特征信息(如字符串常量、意图、权限以及进程信息等),将提取的特征信息和恶意应用进行相似性比较,利用机器学习模型将其分类为合法应用或恶意应用,最后对合法或恶意应用进行标记,输出结果。

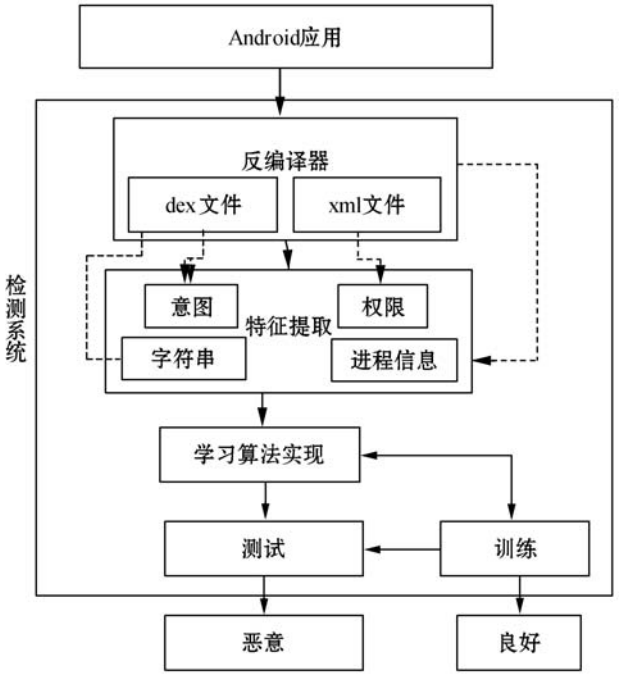

图 1 基于机器学习的 Android 应用检测模型框架

4.2 特征信息提取

为建立有效的 Android 恶意软件检测模型,需要收集应用程序具有代表性的特征:如字符串常量、API 调用、系统调用、权限、意图、过滤器、广播接收器、内容提供者、进程名等。这些特征信息可通过逆向工程反汇编应用程序的 APK 文件,解析应用的二进制文件及清单文件(manifest. xml)进行提取。利用这些特征信息创建(合法应用和恶意应用)特征信息数据库。

字符串常量以及清单文件中的关键字提取后,可将这些特征信息和恶意程序或者合法程序进行对比,以此对应用的安全性进行检测。如果 Android 应用信息被修改并重新包装或者非法用户越权访问那么此应用就是危险的。

为获得准确的比较结果,在特征信息数据库的基础上构建用于检测恶意样本的数据结构-分类器,以权限和意图为例,表 1 – 表 2 分别给出了 Android 应用最常用的 10 种权限特征与意图特征。将提取的关键字信息与合法或恶意应用的特征信息进行对比,通过调整阈值或应用不同的算法计算恶意应用的估分,以此评估恶意软件的危险性。

表 1 Android 应用常用权限特征

安全应用		危险应用	
权限	频率	权限	频率
INTERNET	98%	INTERNET	98%
ACCESS_NETWORK_STATE	89%	READ_PHONE_STATE	89%
WRITE_EXTERNAL_STORAGE	83%	WRITE_EXTERNAL_STORAGE	67%
WAKE_LOCK	53%	SEND_SMS	54%
READ_PHONE_STATE	53%	RECEIVE_SMS	38%
ACCESS_WIFI_STATE	48%	WAKE_LOCK	38%
GET_ACCOUNT	42%	READ_SMS	37%
VIBRATE	41%	ACCESS_COARSE_LOCATION	32%
BILLING	39%	ACCESS_FINE_LOCATION	30%
ACCESS_COARSE_LOCATION	24%	READ_CONTACTS	23%

表 2 Android 应用常用意图特征

安全应用		危险应用	
意图	频率	意图	频率
SEND_MULTIPLE	45%	BOOT_COMPLETED	56%
SCREEN_OFF	23%	SENDTO	45%
USER_PRESENT	18%	DIAL	42%
SEARCH	17%	SCREENOFF	37%
PICK	10%	TEXT	28%
DIAL	9.5%	SEND	27%
GET_CONTENT	9%	USER_PRESENT	22%
EDIT	8.7%	PACKAGE_ADDED	21%
MIDEA_MOUNTED	8%	SCREEN_ON	18%
BATTERY_CHANGE	7%	CALL	10%

本文仅建立了部分用于检测 Android 应用安全性的权限特征与意图特征,为了提高检测的准确性,需获取尽可能多的权限与意图特征用于模型建立。同时,

为了提高恶意软件判别的准确度,还可对特征信息的范围进行扩展,如加入字符串常量、进程名、广播发送者及接受者等信息,通过对特征信息的机器学习,提高检测准确度。

4.3 Android 应用安全性检测

通过机器学习将所有提取的特征融合并组成单个特征向量,应用数理统计算法对恶意软件检测模型进行完善。如使用 K-means 聚类算法按照功能类型对应用程序分组,然后对属于同一功能类型的所有应用程序提取其权限特征,并以权限特征为研究对象,使用 KNN 算法进行 Android 恶意软件的分类检测[16];通过支持向量机对恶意软件基因进行学习[17];应用统计学方法训练数据集,结合聚类算法建立模型以进行应用的安全性检测[18]。

此外,还可加入自动采样强化训练,并进行交叉验证及重复性测试等。通过算子设置自动调整阈值,得出适合检测模型的阈值。在此基础上进行样本训练,然后测试所获得的检测模型。一般地,学习的数据特征越多,基于算法评断的精确度越高。

5 结　语

本文研究了基于静态和动态的恶意软件检测方法,提出了一种新的基于机器学习的 Android 恶意应用检测框架。此方法可以弥补静态分析和动态分析技术的不足,具有较高的精确性。

该框架首先从各种 Android 应用(合法和恶意应用)提取特征信息(如用户权限、意图、应用程序敏感字符串等),将特征信息作为机器学习的分类器输入,然后利用应用程序清单文件的关键字等特征来检测恶意软件,最后将所有提取的特征融合并组成单个特征向量,以提高检测的准确性。

该机器学习检测框架具有较高的检测精度和运行效率,同时该框架可根据恶意关键字设计不同检测计算模型,具有扩展性。

参 考 文 献

[1] 卿斯汗. Android 安全研究进展[J],软件学报,2016,27(1):45-71.

[2] Li Q, Li X. Android Malware Detection Based on Static Analysis of Characteristic Tree[C]// Proceedings of the 2015 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery(CyberC). IEEE, 2015:84-91.

[3] Yerima S Y, Sezer S, Mcwilliams G, et al. A New Android Malware Detection Approach Using Bayesian Classification [C]// 2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA). IEEE, 2013.

[4] Egele M, Kruegel C, Kirda E, et al. PiOS: Detecting privacy leaks in iOS applications[C]// Proceedings of the Network and Distributed System Security Symposium, 2011.

[5] Enck W, Octeau D, Mcdaniel P, et al. A study of Android application security[C]// Proceedings of the 20th USENIX conference on Security. USENIX Association, 2011.

[6] Enck W, Gilbert P, Chun B G, et al. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones[J]. Communications of the ACM, 2014, 57(3): 99-106.

[7] Zheng M, Sun M, Lui J C S. DroidTrace: A Ptrace Based Android Dynamic Analysis System with Forward Execution Capability[C]// Wireless Communications & Mobile Computing Conference. IEEE, 2014: 128-133.

[8] Das S, Liu Y, Zhang W, et al. Semantics-Based Online Malware Detection: Towards Efficient Real-Time Protection Against Malware[J]. IEEE Transactions on Information Forensics & Security, 2017, 11(2):289-302.

[9] Quan D, Zhai L, Yang F, et al. Detection of Android Malicious Apps Based on the Sensitive Behaviors[C]// 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). IEEE Computer Society, 2014.

[10] Spreitzenbarth M, Freiling F, Echtler F, et al. Mobile-Sandbox: Having a Deeper Look into Android Applications [C]// Proceedings of the 28th Annual ACM Symposium on Applied Computing. ACM, 2013:1808-1815.

[11] Sahs J, Khan L. A Machine Learning Approach to Android Malware Detection[C]// Proceedings of the 2012 European Intelligence and Security Informatics Conference. IEEE, 2012: 141-147.

[12] Shabtai A, Kanonov U, Elovici Y, et al. "Andromaly": a behavioral malware detection framework for android devices [J]. Journal of Intelligent Information Systems, 2012, 38(1):161-190.

[13] Aung Z, Zaw W. Permission-Based Android Malware Detection[J]. International Journal of Scientific & Technology Research, 2013, 2(3):228-234.

[14] Damshenas M, Dehghantanha A, Choo K K, et al. M0droid: an android behavioral-based malware detection model[J]. Journal of Information Privacy & Security, 2015, 11(3):141-157.

[15] Rovelli P, Vigfusson Y. PMDS: Permission-based Malware Detection System [M]// Information Systems Security. Springer International Publishing, 2014:338-357.

[16] 李江华,邱晨. 一种基于元信息的 Android 恶意软件检测方法[J/OL]. 计算机应用研究,2019, 36(11). [2018 - 08 - 10]. http://www. arocmag. com/article/02 - 2019 - 11 - 037. html.

[17] 韩金,单征,赵炳麟,等. 基于软件基因的 Android 恶意软件检测与分类[J/OL]. 计算机应用研究,2019,36(6). [2018 - 03 - 16]. http://www. arocmag. com/article/02 - 2019 - 06 - 039. html.

[18] 冷波,李建彬. 基于统计学特征的 Android 恶意应用检测方法[J/OL]. 计算机应用于研究, 2019, 36(9). [2018 - 05 - 24]. http://www. arocmag. com/article/02 - 2019 - 09 - 041. html.

(上接第 295 页)

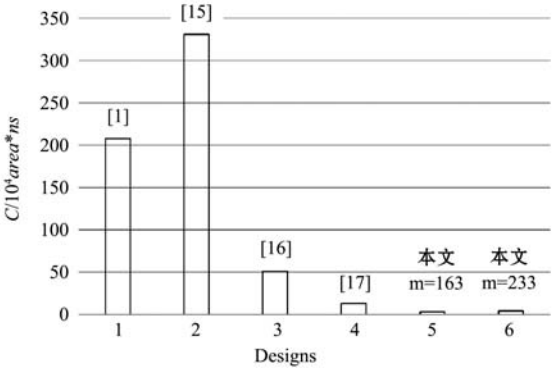

图 4 各 Montgomery 模乘器性能比较图

由图 4 不难看出,虽然文献[16]的频率最大,但是 C 参数的取值却并不理想。文献[18]中根据芯片的资源情况采用全局串行局部并行的结构设计乘法器,可以看出该模乘器的效率是相对不错的。而本文的 Montgomery 模乘方案综合考虑芯片资源与延时问题,依据算法设计中在单位时钟周期内所进行的不定次循环迭代,对应到硬件实现中的串并混合结构,最终取得速度与面积的最佳折衷。所以本设计的 Montgomery 模乘器的效率在 $GF(2^{233})$ 域和 $GF(2^{163})$ 域都有很大提升。

5 结 语

为了提高椭圆曲线密码体制中模乘运算的效率,改进后的 Montgomery 模乘算法将原算法中的逐位扫描运算改为在单位时钟内进行 k 步移位来计算模乘,大大减少了时钟周期数。同时考虑资源占用情况,针对不同的操作数长度选取特定步数 k,以取得速度与资源的最佳折衷。然后根据硬件实现的特点,设计一种基于 FPGA 的串并混合结构的 Montgomery 模乘器。通过与相关文献的对比分析,本方案的模乘器不仅在速度与资源上有平衡的优势,而且更适合硬件实现,具有较高的灵活性与可移植性。

参 考 文 献

[1] 李嘉敏,戴紫彬,王益伟. 可编程可伸缩的双域模乘加器研究与设计[J]. 电子技术应用,2018,44(1):28 - 32,36.

[2] Cash D, Kiltz E, Shoup V. The Twin Diffie - Hellman Problem and Applications[M]. Springer-Verlag New York, Inc. 2009.

[3] 李超,张强,曲英杰. 域椭圆曲线点乘的 VLSI 实现方法研究[J]. 计算机测量与控制,2017,25(12):232 - 236.

[4] Hankerson D. 椭圆曲线密码学导论[M]. 张焕国,译. 北京:电子工业出版社,2005.

[5] 韩炼冰. 椭圆曲线密码算法的 FPGA 设计与实现[D]. 成都:电子科技大学,2018.

[6] 袁宁,吴卫华. 公开密钥算法芯片的设计与实现[J]. 计算机应用与软件,2009,26(6):99 - 101.

[7] 胡诗玮,徐和根. 有限域 $GF(2^{256})$ 上椭圆曲线密码算法的硬件设计[J]. 机电一体化, 2015(2):71 - 75.

[8] 赖忠喜,张占军,陶东娅. 椭圆曲线底层域快速算法的研究[J]. 计算机工程与应用,2014,50(3):67 - 70.

[9] 车文洁,董秀则,高献伟,等. Montgomery 模乘法器的实现与优化[J]. 计算机应用与软件,2017,34(3):312 - 315,333.

[10] 胡进,何德彪,陈建华. 基于高基阵列乘法器的高速模乘单元设计与实现[J]. 计算机工程与设计,2010,31(6):1202 - 1204.

[11] 邬贵明,王淼,谢向辉. 一种基于 FPGA 的素域椭圆曲线标量乘结构[J]. 计算机工程与科学,2018,40(5):793 - 797.

[12] Khan S, Javeed K, Shah Y A. High-Speed FPGA Implementation of Full-Word Montgomery Multiplier for ECC Applications[J]. Microprocessors and Microsystems, 2018.

[13] 杜清. 基于 FPGA 的二进制域双基可重构 ECC 系统的设计[D]. 南京:东南大学,2016.

[14] 胡振宇. 椭圆曲线密码算法的嵌入式软件实现——基于 ARM 指令集[D]. 上海:复旦大学,2006.

[15] Deschamps J P, Imana J L, Sutter G. Hardware Implementation of Finite-Field Arithmetic[M]. New York: McGraw-Hill Companies, 2009.

[16] 杨晓辉,王雪瑞,秦帆,等. 基于 FIOS 类型的 Montgomery 双域模乘器设计[J]. 电子技术应用,2011(10):144 - 148.

[17] 郭晓,蒋安平,宗宇. SM2 高速双域 Montgomery 模乘的硬件设计[J]. 微电子学与计算机,2013,30(9):17 - 21.

[18] 卫学陶,戴紫彬,陈韬. $GF(2^m)$ 域上通用可配置乘法器的设计与实现[J]. 计算机工程与应用,2007(12):91 - 93.