doi :10.3969/j.issn.1671-1122.2016.09.043

Android 移动恶意代码检测的研究概述与展望

蔡林[1]，陈铁明[2]
（1. 浙江省公安厅网络安全保卫总队，浙江杭州 310012；2. 浙江工业大学计算机科学与技术学院，浙江杭州 310023）

摘　要：随着 Android 系统在移动智能终端的应用越来越广，Android 系统的信息安全问题也日趋严重。尽管 Android 操作系统的进程采用了独立的虚拟内存空间保障其程序内核的可靠性，但由于应用程序各种事件之间的调用和关联，导致隐私数据泄露、程序越权操作、电池耗尽攻击、恶意进程交互等手机安全事件频繁涌现。因此 Android 恶意代码检测技术成为移动应用安全防护的一个研究热点。文章从 Android 恶意代码检测的应用需求和背景出发，概述了动态检测和静态检测方法、基于机器学习的智能检测方法、基于形式化的软件工程方法等各个方面的研究进展，最后提出了融合机器学习和软件工程方法的综合静态检测方法的研究方向，并分析了技术难点，可为学术研究和产品开发提供有价值的参考。

关键词：移动恶意代码；动态检测；静态检测；机器学习；模型检测
中图分类号：TP309　　**文献标识码：**A　　**文章编号：**1671-1122（2016）09-0218-05

中文引用格式：蔡林，陈铁明 . Android 移动恶意代码检测的研究概述与展望 [J]. 信息网络安全，2016(9)：218-222.

英文引用格式：CAI Lin, CHEN Tieming. Research Review and Outlook on Android Mobile Malware Detection [J]. Netinfo Security, 2016 (9) : 218-222.

Research Review and Outlook on Android Mobile Malware Detection

CAI Lin[1], CHEN Tieming[2]
(1. *Cyber Security Center*, *Department of Public Security Zhejiang Province*, *Hangzhou Zhejiang* 310012, *China*; 2. *College of Computer Science and Technology*, *Zhejiang University of Technology*, *Hangzhou Zhejiang* 310023, *China*)

Abstract: With the wide spread of Android-based mobile applications, the problem of information security in Android system is increasingly serious . Although Android operating system adopted independent virtual memory space to guarantee the reliability of its kernel , because of calls and association between various events in application , it will lead to private data leakage , unauthorized operation procedures, attacks to run out the battery , malicious processes interact and other mobile security events. Therefore , Android malware detection techniques become a hot topic in the domain of mobile application security. In this paper, the application requirements and environments for Android malware detection are firstly described, and then the diversity malware detection methods are surveyed which include dynamic and static methods, machine learning-based schemes, formal method-based software engineering techniques. Finally, the research direction to initiate a comprehensive static detection framework by integrating machine learning and software engineering is proposed, with some key challenges concomitantly analyzed, which can be valuable reference both for academic communities and industrial products.

Key words: mobile malware; dynamic detection; static detection; machine learning; model checking

收稿日期：2016-07-25
基金项目：国家自然科学基金 [U1509214]；浙江省自然科学基金 [LY16F020035]；浙江省信息安全重点实验室开放课题 [KF201603]
作者简介：蔡林（1963—），男，浙江，高级工程师，主要研究方向为网络安全；陈铁明（1978—），男，浙江，教授，博士，主要研究方向为网络与信息安全。
通信作者：陈铁明 tiemingchen@foxmail.com

0 引言

IDC 最新发布的数据显示，2015 年 Android 和苹果 iOS 的总市场份额超过 95%，而 Android 独占 80% 以上，全年出货数量超过 10 亿部，成为目前应用最广的智能手机操作系统，仅 Google Play Market 商店已发布超过 140 万 Android 应用程序。随着 Android 智能手机市场的极速发展，越来越多手机厂商及应用提供商开始花大量的精力专注于 Android 应用的研发，包括小米、华为等国有品牌。因而作为保障手机应用质量的自动化测试面临极大的挑战，成为移动互联网软件工程领域凸现的新问题。软件测试自动化同时可提高测试质量并降低测试成本，但当前因缺少有效工具还无法面向手机应用软件全面实现自动化测试。

更重要的是，尽管 Android 操作系统的进程采用了独立的虚拟内存空间保障其程序内核的可靠性，但由于应用程序各种事件之间的调用和关联，导致隐私数据泄露、程序越权操作、电池耗尽攻击、恶意进程交互等手机安全事件频繁涌现，截至 2015 年 Android 恶意代码的数量已超过 120 万。Google Play Store 发布的 APP 应用 60% 为免费，仅通过注册身份认证机制保障应用程序的合法性。即开发者首先进行注册，支付费用后发布应用，但是没有对发布的程序进行二次认证，若开发者发布了恶意应用，虽在发现后会被撤销发布，但已下载使用的手机用户则已承受攻击。由于开发者能够通过服务器控制客户端恶意行为的发作时间，可在应用通过审核发布后再激活恶意行为，因此即使通过审核也难以完全确保应用程序的安全。例如，Android 系统级漏洞被频频曝光，Android 系统签名漏洞可让黑客在不破坏数字签名的情况下，篡改任何正常手机应用，如植入偷账号、窃隐私或恶意扣费等各种恶意代码。这些恶意代码通过未经安全检测的应用商店可直接装入用户手机使用户蒙受经济损失。另外，随着物联网和智慧城市的应用推进，在基于移动 APP 的智能家居、智能工厂等应用平台上，黑客可能远程控制家居设备或工厂控制等系统造成后果不堪设想的巨大灾难。据 TrustGo 公司的分析报告，在国内知名的 91 应用市场上隐含恶意代码威胁的应用比例将近 20%[1]。由于中国用户无法直接从 Google Play 上下载应用，存在大量管理混乱的第三方移动应用市场，对 Android 终端安全造成非常严重的威胁。

尽管中国移动互联网基地在 2011 年开始联合国内众多终端厂商、软件开发商等，成立了首个手机应用“绿色安全诚信联盟”，成立了专业的手机应用监控测试中心，形成了一套从开发、测试到上线的规范流程和机制，但主要还是依赖人工的内容安全审核机制，并利用工具对隐藏的病毒和恶意插件进行全面扫描。基于人工和软件工具的审核机制，无论在检测效率和可靠性等方面均面临极大挑战。

下文将在概括几类 Android 恶意代码检测技术的基础上，提出一套融合多类方法的综合检测机制，并分析其面临的关键难题，为相关研究开发人员提供参考。

1 静态检测与动态检测方法

手机平台恶意软件与传统恶意软件相比具有特殊性，例如不需要像病毒或蠕虫一样大规模自我复制和分发，而只需放置到应用商店，即可得到大规模下载。此外，手机恶意软件的特征签名比较隐蔽，绝大部分恶意软件的特征代码都不尽相同，且无法及时检测到新的恶意软件，因此基于特征签名匹配的检测技术不能很好的检测 Android 手机恶意代码，如 2013 年学术界实现的著名开源工具 Androguard[2] 也是一种基于开发者签名匹配的解决方案，通过建立庞大、有效的数据库可快速有效地发现已知恶意应用，但无法判定新的恶意代码。

当前另一种较流行的方法是基于行为的检测技术，即通过监控 Android 代码的行为（例如通过动态拦截或静态分析的方法获取程序的系统调用序列等），再结合已知的恶意行为模式（恶意软件的系统调用序列），精准地检测手机恶意软件行为 [3]。行为检测技术的最大优势在于其特征库较小，且不需要频繁更新，还可检测行为模式相似的未知恶意软件，另外还能够抵抗代码的混淆加密，因为代码加密不会更改程序行为模式。事实上，基于行为的检测技术还可分为动态和静态两种，动态行为检测是在程序运行的过程中执行，而静态行为检测则在程序运行之前对其代码进行分析 [4]。动态检测由于在程序运行时执行，自动化程度高，但实时性要求较高，且必须确保在恶意程序对系统产生损害前检测出威胁，耗费相对大，通常的解决方法是利用沙盒来封闭运行并监控程序。同时，由于恶意行为触发的条件多变，动态行为检测的漏报率也较高，代表性

的 Android 恶意代码动态检测方法包括基于内核行为和运行环境等行为分析 [5,6]。静态检测方法则在程序运行之前分析恶意行为，一般通过逆向工程抽取程序的特征，如二进制序列、操作码序列、函数调用序列等模型。与动态行为检测相比，静态行为检测的耗费更低（无需沙盒、虚拟机等程序动态运行机制）、风险更小、实时性要求更低（只要在程序执行前完成检测即可），但自动化程度则相对低，代表性的 Android 恶意代码静态检测方法主要包括抽取程序依赖图和调用序列图等进行建模分析 [7,8]。

鉴于两类方法的优缺点，目前已有不少研究提出将动态检测和静态检测相混合的方法 [9,10]，一般是通过两种方法的混合使用，利用基于特征分析的静态检测技术快速准确检测出已知恶意代码，采用基于行为分析的动态检测技术高效直观地检测未知恶意代码，但该类方法仍然面临动态检测过程的执行效率问题 [11]。

2 基于机器学习的检测方法

基于机器学习的数据挖掘方法是近两年来 Android 恶意代码检测技术发展的研究热点 [12]。就机器学习算法而言，主要包括 k-NN、SVM、神经网络、贝叶斯分类、聚类等常用方法的应用；就特征处理模式而言，主要包括源代码特征提取、移动应用配置文件分析、API 调用分析、应用权限监控、行为异常分析等多层面多维度的方法 [13]。事实上，从是否需要运行应用程序的角度，仍可将基于机器学习的移动恶意代码检测技术分为静态和动态两大类，其主要技术特点体现如下：1）针对基于非运行态特征匹配的静态方法，利用机器学习可以提升对未知恶意代码的检测能力，建立具有不断演进和泛化学习的智能模型 [15]；2）针对基于运行态行为特征的动态方法，利用机器学习可以提高恶意代码行为的检测精度，但也使动态检测的效率面临更大挑战 [16,17]。从执行效率和应用前景看，基于机器学习的静态检测模型更受关注，且随着云计算和大数据处理技术的发展，可对移动应用建立更高维特征、更全样本的检测模型，不断提高检测精度 [18,19]。

机器学习从人工智能研究的角度出发，以实现自动化智能识别为目标，但从恶意代码检测的精准度来看，主要面临两个关键问题：1）无法完全消除恶意代码识别的误差；2）无法精准分析定位恶意代码的行为。

3 基于软件工程的检测方法

因此在软件工程研究领域，为实现恶意代码的精准识别，软件形式化建模与分析技术也开始被研究应用于恶意代码检测，主要包括形式语义 [20]、程序分析 [21]、模型检验 [22] 和符号执行 [23] 等方法。2010 年提出基于程序污点分析的 TaintDroid 检测方法掀起了利用程序静态分析检测 Android 恶意代码的研究热潮 [24]，基于程序分析的形式化方法主要聚焦 Android 应用消息泄露 [25]、权限调用异常 [26] 等个别安全漏洞的检测；为能检测更一般化的恶意代码，最近有研究提出了支持谓词语义的静态程序分析方法 [27] 和支持完整逻辑推演模型的安全分析器 [28] 等方法，但都需要大量的人工建模难以实现自动化；针对恶意代码的自动验证，已有研究提出了基于 LTL 模型检验 [29] 和安全类型检验 [30] 等自动验证方法，但 Android 程序模型抽象过程仍将耗费较多的手工建模，恶意行为的 LTL 描述过程也往往过于复杂 [31]，不过最近提出的基于下推系统（Pushdown Systems）的恶意代码验证方法可有效提高执行效率 [32]；在恶意代码自动化测试方面，虽已有研究提出了基于 JPF 符号执行的 Android 安全测试方法 [33]，但主要采用基于 GUI 测试用例的路径遍历方法，尚未出现基于移动恶意代码行为特征的执行路径自动测试的有效方法。

不同于机器学习方法的高效性和智能性，形式化方法通常需要复杂的模型抽象过程，虽然模型建立后可实现自动推理验证，但因其自身的执行性能不高而不适用于对实时性要求较高的恶意代码动态检测 [34]。然而，静态的形式化方法可有效弥补机器学习静态检测的两大不足：1）模型检验本质上属于证伪技术，可实现对移动恶意代码的确定性验证，消除机器误报；2）符号执行本质上属于测试技术，可实现对代码程序执行的自动测试，挖掘程序漏洞。另外，将模型检验和符号执行相结合的代码分析方法 [35]，可将模型检验产生的反例路径用于构建符号执行的测试用例，有效提高分析执行效率，可为研究 Android 恶意代码的自动测试提供约束求解依据。

4 研究展望及难点分析

综上所述，为保障移动应用安全，Android 恶意代码检测成为当前研究热点，虽已提出机器学习、模型检验、

符号执行等多种创新方法，但整体研究及关键技术尚处于起步阶段，还具备较大的研究空间[36]：1）从安全监管的角度，如何为杂乱的移动应用市场建立一套以人工专业服务为本的软件评测与安全管理制度；2）从应用需求的角度，如何实现可部署可操作的移动恶意代码检测平台，保障APP应用检测的高效性、及时性、准确性、扩展性等；3）从框架研究的角度，如何结合现有多种移动应用软件检测技术的优势，进一步研究设计一套综合的解决方案；4）从技术突破的角度，针对移动恶意代码精准检测问题，如何实现快速高效的恶意代码智能识别、恶意行为自动验证、代码执行自动测试。

综上所述，围绕Android恶意代码检测，可以结合市场亟需的安全监管和技术应用需求，研究一套以专家分析为核心，融合机器学习、模型检验、符号执行等方法的恶意代码精准识别、验证、测试的综合完整的静态检测机制。

针对上述研究计划，为有效实现静态自动化精准检测，面临的主要技术研究难点包括：

1）在机器学习方面：研究移动应用和恶意代码的类别与家族特征，及其与机器学习特征处理的关联问题；研究移动恶意代码识别误报/漏报率的可控性，及其与检测模型参数的关联问题；研究移动恶意代码特征提取模型的稳定性，及其对代码混淆加密的处理方法等。

2）在模型检验方面：根据移动应用类别及恶意代码家族特性等，研究自动生成程序的形式化模型；根据移动应用类别及恶意代码家族特性等，研究自动简化恶意行为的逻辑表达式；研究新型高效的移动恶意代码模型快速验证算法，及其反例路径自动生成方法等。

3）在符号执行方面：研究Java代码的符号执行自动分析，以及Dalvik和Java虚拟机的自动转换方法；研究Android系统的事件驱动特性，以及GUI用例生成与路径分歧消除等方法；研究恶意代码行为特征与模型检验反例路径的模式分析，及其符号执行测试方法等。

5 结束语

针对移动APP应用现状，快速、高效、自动的代码安全检测技术研究亟待突破。系统地、有效地开展Android恶意代码的检测方法研究，既是可信软件领域的一项衍生课题，也是移动应用软件开发安全保障的一项基础性课题，具备极大的学术研究价值和市场应用前景，将有效推动移动互联网应用的健康持续发展。（责编 李鹏）

参考文献：

[1]TRUSTGO. BSides Las Vegas: Your Droid Has No Clothes[EB/OL]. http://blog.trustlook.com. 2014-01-08.

[2]Google Project Hosting. Andoguard[EB/OL]. https://code.google.com/p/androguard/, 2013-12-09.

[3]LIU Wu, REN Ping, LIU Ke, et a1.Behavior-based Malware Analysis and Detection[C]//IEEE. 2011 First International Workshop on Complexity and Data Mining(IWCDM), September 24-28, 2011, Nanjing, Jiangsu, China, NJ: IEEE, 2011 : 39-42.

[4]ZHOU Y, JIANG X. Dissecting Android Malware:Characterization and Evolution. [C]//IEEE. of the 2012 IEEE Symposium on Security and Privacy, May 20-23, 2012. San Francisco Bay Area, California, IEEE Computer Society Washington, DC,USA, 2012 : 95-109.

[5]ZOU S, ZHANG J, LIN X. An Effective Behavior-based Android Malware Detection System[J]. Security and Communication Networks, 2015, 8 (12) : 2079-2089.

[6]KUBOTA A. Kernel-based Behavior Analysis for Android Malware Detection[C]//IEEE. 2011 Seventh International Conference on Computational Intelligence and Security,CIS 2011,December 3-4,2011, Sanya ,Hainan. NJ: IEEE, 2011 : 1011-1015.

[7]WU D, MAO C, Wei T. DroidMat: Android Malware Detection through Manifest and API Calls Tracing[C]//IEEE. 2012 Seventh Asia Joint Conference on Information Security,August 9-10, 2012, Tokyo, Japan. NJ: IEEE, 2012: 62-69.

[8]GASCON H, YAMAGUCHI F, ARP D, et al. Structural Detection of Android Malware Using Embedded Call Graphs[C]//ACM. of the 2013 ACM Workshop on Artificial Intelligence and Security, November. 4-8 Berlin, Germany. NY: ACM, 2013: 45-54.

[9]AVG. Malware Detection Methods[EB/OL]. http://www.avg.com/us-en/avg-software-technology, 2016-1-15.

[10]Android and Security [EB/OL]. http://googlemobile.blogspot.com/2012/02/android-and-security.html, 2016-1-15.

[11]SPREITZENBARTH M, SCHRECK T, ECHTLER F, et al. Mobile-Sandbox: Combining Static and Dynamic Analysis with Machine-learning Techniques[J]. International Journal of Information Security, 2014:1-13.

[12]ALLIX K, BISSYANDE T, KLEIN J. Machine Learning-Based Malware Detection for Android Applications: History Matters! University of Luxembourg [EB/OL], http://hdl.handle.net/10993/17251, 2016-1-15.

[13] 王蕊，冯登国，杨轶，等．基于语义的恶意代码行为特征提取及检测方法 [J]. 软件学报，2012，23（2）：378-393.

[14] 任伟，柳坤，周金．AnDa：恶意代码动态分析系统 [J]. 信息网络安全，2014（8）：28-33.

[15]AMOS B, TURNER H, White J. Applying Machine Learning Classifiers to Dynamic Android Malware Detection at Scale[C]//IEEE. Wireless Communications and Mobile Computing Conference (IWCMC), July 1-5,2013, Sardinia, Italy. NJ: IEEE, 2013:1666-1671.

[16]NARUDIN F A, FEIZOLLAH A, ANUAR N B, et al. Evaluation

of Machine Learning Classifiers for Mobile Malware Detection[J]. Soft Computing, 2014, 20(1):1−15.

[17] 李桂芝，韩臻，周启惠，等 . 基于 Binder 信息流的 Android 恶意行为检测系统 [J]. 信息网络安全，2016（2）：54−59.

[18]PREDA M D, al. E. A semantics−based Approach to Malware Detection[J]. ACM Transactions on Programming Languages and Systems, 2007, 42 (25) : 377−388.

[19] 贾同彬，蔡阳，王跃武，等 . 一种面向普通用户的 Android APP 安全性动态分析方法研究 [J]. 信息网络安全，2015（9）：1−5.

[20]SONG D, BRUMLEY D, YIN H, et al. BitBlaze: A New Approach to Computer Security via Binary Analysis[M]. Springer Berlin Heidelberg Information Systems Security, 2008.

[21]JOHANNES K, STEFAN K, CHRISTIAN S, et al. Detecting Malicious Code by Model Checking[J]. Lecture Notes in Computer Science Volume 3548, 2005 : 174−187.

[22]ENCK W, GILBERT P, CHUN B, et al. TaintDroid: An Information−Flow Tracking System for Realtime Privacy Monitoring on Smartphones.[J]. ACM Transactions on Computer Systems, 2010, 57(3):99−106.

[23]MOSER A, KRUEGEL C, KIRDA E. Exploring Multiple Execution Paths for Malware Analysis[J]. Security and Privacy, Sp 07, IEEE Symposium on, 2007:231 − 245.

[24]XIAO X, TILLMANN N, FAHNDRICH M, et al. User−aware Privacy Control via Extended Static−information−flow Analysis[J]. Automated Software Engineering, 2014, 7304(4):80 − 89.

[25]FANG Z, HAN W, LI Y. Permission Based Android Security: Issues and Countermeasures[J]. Computers & Security, 2014, 43(6):205−218.

[26]LIANG S, MIGHT M, VAN H D. Anadroid: Malware analysis of Android with User−supplied Predicates[J]. Electronic Notes in Theoretical Computer Science, 2013 (311) : 3−14.

[27]ALESSANDRO A, GABRIELE C, ALESSIO M. Formal Modeling and Reasoning about the Android Security Framework [J]. Lecture Notes in Computer Science, 2012: 64−81.

[28]MICHELE B, STEFANO C, ALVISE S. Lintent: Towards Security Type−Checking of Android Applications [J]. Lecture Notes in Computer Science, 2013 (7892) : 289−304.

[29]FU S, TAYSSIR T. LTL Model−Checking for Malware Detection [J]. Lecture Notes in Computer Science , Springer Berlin Heidelberg, 2013 (7795) : 416−431.

[30]FU S, TAYSSIR T. Efficient Malware Detection Using Model−Checking [J]. Lecture Notes in Computer Science , Springer Berlin Heidelberg, 2012 (7436): 418−433.

[31]SONG F, TOUILI T. Pushdown Model Checking for Malware Detection[J]. International Journal on Software Tools for Technology Transfer, 2014, 16(2): 147−173.

[32]RASTOGI V, CHEN Y, ENCK W. AppsPlayground: automatic security analysis of smartphone applications[C] //ACM third ACM conference on Data and Application Security and Privacy. February 18−20, 2013, San Antonio, Texas, USA. NY: ACM, 2013: 209−220.

[33]PETSAS T, VOYATZIS G, ATHANASOPOULOS E, et al. Rage Against the Virtual Machine: Hindering Dynamic Analysis of Android Malware[C]//ACM Seventh European Workshop on System Security. April 13−16, 2014, Amsterdam, Netherlands. NY: ACM, 2014: 5−15.

[34]VIJAYENDRA G, VIJAY K, SANJAY R, et al. Category Based Malware Detection for Android. [C]// SSCC. Second International Symposium, September 24−27, Delhi, India. SSCC 2014,2014: 239−249.

[35]PASAREANU C S, VISSER W, BUSHNELL D, et al. Symbolic PathFinder: Integrating Symbolic Execution with Model Checking for Java Bytecode Analysis[J]. Automated Software Engineering, 2013, 20(3):391−425.

[36]SABA A, MUNAM A, ABID K, et al. Android Malware Detection & Protection: A Survey[J]. International Journal of Advanced Computer Science and Applications, 2016, 7(2): 463−475.