移动平台用户隐私保护技术综述

杨哲慜　杨　珉

复旦大学

关键字：移动安全 安卓 用户隐私保护

引言

以谷歌公司的安卓系统和苹果公司的iOS系统为代表的移动操作系统正逐渐取代桌面操作系统，成为公众接入互联网的主要入口平台。短短数年内，移动智能终端出现爆发式增长，据权威市场研究机构IDC的数据显示[1]，2013年全球智能手机的出货量达4.3亿台，比2009年增长了十倍，到2015年第二季度更是单季度出货量达3.415亿台，其主要增长来自于亚太和中东亚地区。

移动操作系统生态链正从智能手机拓展到如平板电脑、智能家居、可穿戴设备、车载控制系统等多种衍生产品中，日益驱动着互联网的改变和发展。作为移动互联时代的突出标志之一，移动应用聚集大量高附加值的信息和资源。这些信息和资源不仅包含个人的手机信息、身份信息、地理位置信息，还包含诸多账号信息以及邮件、文件等信息。

移动应用利用用户赋予的个人信息为其提供便捷、即时、精确的定制服务。正是在这一背景下，移动平台得以取代传统PC平台成为人们接入互联网的首选方式。因此，基于移动平台的产业链成为近年来最具发展活力的时代宠儿。

然而，移动平台的高度互联性也是一把双刃剑，它同时使用户轻易暴露在攻击者的威胁之下。受政治、军事和商业利益驱使，针对移动终端系统的攻击行为层出不穷，其中针对个人隐私的窃取行为又是其中的重灾区。“棱镜门”以及多项外电资料披露，多个国家在资助移动操作系统安全防护方法的同时，也在研究利用移动操作系统的安全缺陷，通过恶意软件对我国重要人员以及公众实施高级持续性的大规模数据搜集。美国政府曾在过去的三年中资助英国政府通信总部（GCHQ）1亿英镑，专门用于研究手机操作系统的攻击方法。

在即将过去的2015年中，国内个人信息泄露事件频发。中国互联网协会2015年发布的《中国网民权益保护调查报告》[2]称，2015年中，中国网民因个人信息泄露、垃圾信息、诈骗信息等现象导致总体损失约805亿元。其中，78.2%的网民个人身份信息被泄露过，63.4%的网民个人网上活动信息被泄露过。11月份被批露的百度Wormhole漏洞，其影响力覆盖过亿安卓手机用户，造成大量用户隐私泄漏风险。

面对移动互联网用户隐私保护的严峻形势，隐私泄漏的检测和保护近年来颇受关注。本文结合我们团队的研究成果和正在进行的研究工作，揭示了移动应用隐私泄露检测和防护的总体进展，并在增强对用户隐私的识别和保护能力等方面提出值得关注的方向。

隐私泄漏检测和防护的主要技术发展历程

隐私泄漏问题本身已经不是个新问题，早在PC时代就有针对个人隐私的恶意攻击案例存在。但是，自从2008年第一部安卓智能手机面世以来，智能移动终端正在逐渐取代PC成为汇集个人隐私的核心媒介。这在赋予智能移动终端越来越强大功能的同时，也给个人隐私安全防护提出了必须面对的严峻挑战：面对海量的移动应用，种类众多的隐私数据，复杂的数据流动渠道，如何确保个人隐私不被恶意利用？针对这一问题，早期的隐私泄漏分析的关注点主要集中在敏感数据的传播上。研究人员通过用户的隐私数据是否离开移动设备来判断应用是否存在隐私泄露，并从隐私泄漏的检测和防护两个方面展开研究：

隐私泄漏检测

从核心原理来看，隐私泄露检测技术经历了静态分析、动态分析、动静态分析结合的

发展路线。

静态分析方法通常采用静态数据流分析的手段分析程序中的静态敏感数据流向，如加利福尼亚大学戴维斯分校的Androidleaks系统[3]、加利福尼亚大学伯克利分校的ComDroid系统[4]、我们之前提出的LeakMiner系统[5]、乔治亚理工的Chex系统[6]、和德国达姆施塔特工业大学的FlowDroid系统[7]。在隐私泄露检测上，静态分析方法具有运行速度快，代码覆盖率高等特点，但是由于静态方法无法反映恶意软件运行时的动态行为，使得这类检测方法的实际效果受到明显影响。在攻防持续对抗的过程中，攻击者如果采用代码混淆技术也可轻易规避此类方法的检测。

意识到静态分析的缺陷，动态分析方法根据应用程序的敏感信息传输特征动态监测安卓系统中的恶意隐私泄露应用，其中最有代表性的是宾夕法尼亚大学的William Enck等人提出的TaintDroid系统[8]，该系统通过修改安卓内核代码，利用动态指令插桩的方式实时监控程序中的数据传播过程，并在发生敏感数据传播时警告用户。剑桥大学的Rubin Xu等人提出的Aurasium系统[9]则通过另一种方式实现了类似的功能，他们通过对需要监控的应用程序进行重打包，以此给应用程序插入监控逻辑。相比静态分析方法，动态数据流跟踪具有分析精度高的特点，但是由于动态监测技术的局限，其具有代码覆盖率不足、检测结果滞后于数据传输发生等问题。

鉴于动、静态分析在隐私泄露检测过程中的不足和优点，如果可以结合两种分析方法，综合利用其优势，实现既具有较高覆盖率和检测速度，又具有高分析精度的检测方法就具有了相当大的现实意义。我们在2013年提出的AppIntent系统[10]对此作了一些尝试，AppIntent首先利用静态数据流分析获取应用程序中敏感数据传播的候选集，再通过动态符号化执行验证候选敏感数据流是否会真实发生。通过结合两种分析方法，可以在较短时间内，在不损失分析覆盖率的前提下，提高隐私泄露的检测精度。

隐私泄漏防护

在隐私泄漏的防范方面，移动终端系统主动抑制恶意软件对敏感资源的不合理使用是防范恶意软件的重要途径。微软雷蒙德研究院的Jaeyeon Jung研究员提出AppFence系统[11]在动态检测隐私泄漏风险的基础上实时阻止软件对隐私信息的收集行为。北卡罗来纳大学的William Enck教授所在团队提出的Kirin[12]和Saint[13]系统用于阻止一个应用软件同时申请多个可疑的敏感资源。此外，雪城大学的Wenliang Du教授所在团队提出的AFrame系统[14]、伯克利的David Wagner教授所在团队提出的AdDroid系统[15]、和莱斯大学的Shashi Shekhar 等人提出的AdSplit系统[16]均通过进程隔离的方式限制程序内部不可信组件对敏感资源的使用。此外，剑桥大学的Ross Anderson等人提出的Aurasium系统[9]提供给用户直接管理程序访问敏感资源行为的能力。

近年来隐私泄漏研究的新问题

随着隐私泄漏研究的深入展开，近年来研究者们逐渐意识到传统的隐私泄漏定义并不准确。而该定义的不精确和不完整限制了隐私泄漏检测技术的效率和效果。为了在隐私泄漏检测和防护过程中获得更准确更全面的结果，相关研究在以下两个方面对隐私泄漏定义进行了修正：

隐私泄漏的用户感知问题：

基于敏感数据传播的隐私泄漏检测技术认为：通过检测用户的隐私数据是否离开移动设备就可以判断应用是否存在隐私泄露。事实上，由于大量手机应用都采用云计算，许多非恶意应用均利用云服务器为终端用户提供定制化服务。这些应用通常需要收集一些敏感数据（比如位置，联系人信息等），并将其发送到自身服务器。而一些恶意应用在窃取用户敏感数据后也表现为类似的行为，也就是将隐私信息传送到自己的服务器。因此，仅仅依靠应用程序是否传输敏感数据不可以真正判别出应用软件是否存在隐私泄露。

意识到这一问题之后，安全研究人员通过不断改进隐私泄漏的判断标准，力图提高隐私泄漏检测的精度。美国杜克大学的Peter Gilbert等人提出的Vision系统[17]认为如果应用程序将自身敏感数据传输行为加入最终用户许可协议（EULA）并得到用户许可，或者应用程

序在敏感数据传输行为发生时明确的通知告知用户，则该数据传输可以被认为并非隐私泄露。与之类似的是，佐治亚理工学院的Wenke Lee教授所在团队提出的BLADE系统[18]通过识别应用程序是否得到了用户许可，来检测从网络上动态下载的恶意软件。然而，手机应用一般不提供最终用户许可协议（EULA），同时，即使在数据传输符合用户意图的时候（如转发短信），应用程序也很少向用户弹出显式的用户提示。伯克利的Dawn Song教授所在团队提出的Pegasus系统[19]以应用程序使用API和权限的顺序为特点来检测软件恶意行为。与我们的研究类似的是，它也检测那些与图形界面操作不相符的恶意行为。然而，隐私泄露无法通过简单的权限或者API的使用顺序来概括。因此应用程序的隐私泄露行为无法通过这种方法被检测出来。此外，Pegasus需要分析人员根据应用程序的行为手动指定用户程序行为验证的属性特征。在不了解应用程序代码的情况下，分析人员很难制定这些属性特征。这些方法通过归纳隐私泄漏行为的外部表征试图区分隐私泄漏和应用软件的正常敏感数据传输行为，对隐私泄漏的精确检测进行了有效尝试，但是这些方法仍然不能准确判断隐私泄漏。

我们团队于2013年发表在CCS会议上的论文[10]中提出了另一种判断隐私泄漏的方法。我们认为，一个更好的隐私泄露判别方法应该是敏感信息传输是否出于用户本身的意图：

- **符合用户意图的敏感数据传输**。为了使用应用软件的一些功能，用户经常需要容忍自己的隐私数据经过特定渠道被发送出手机。比如，当使用辅助管理短信的应用时，用户可以通过点击屏幕上的一些按钮将特定短信转发给其它用户；而在使用一些基于用户地理位置的服务时，用户为了得到一些其感兴趣的内容，往往要允许将自己的位置信息发送给应用服务器。由于这类功能在使用敏感数据时，用户已经知情，所以不应该将其归类为隐私泄露。
- **不符合用户意图的敏感数据传输**。恶意应用在用户不知情且与用户使用的功能无关的情况下非法传输敏感数据，我们称之为用户不知情的敏感数据传输，或者叫隐私泄露。在大部分情况下，恶意应用为了使得用户很难察觉到其恶意行为，通常都会秘密传输敏感数据。

综上所述，应用程序对敏感数据的传输是不是在泄露隐私，取决于它是否满足用户的意图。不幸的是，由于用户的意图非常繁杂而且不同的应用采用不同的实现方式，因此通过工具自动化检测用户意图基本不可能实现。相对而言，通过为人工分析人员提供敏感数据传输的上下文信息是一种较为可行的方法。通过分析数据传输对应的用户输入序列，分析人员可以更容易的判断数据传输是否出于用户意图。这促使我们设计并实现了AppIntent应用分析框架。

AppIntent从敏感数据的传播路径中进一步分析出导致这次数据传输的用户数据输入和交互输入。再利用这两部分用户输入控制应用程序的执行。在控制应用程序执行的过程中，AppIntent将对应敏感信息传输的图形界面操作的顺序呈现给软件分析人员。通过观察这些界面操作，软件分析人员可以快速做出判断。

	敏感信息源	设备号	手机号	地理位置	通讯录	短信	总数
恶意应用	AppIntent (静态分析/ 符号化执行/ 演示案例)	389/ 256/ 246	53/ 50/ 50	76/ 68/ 67	13/ 13/ 13	27/ 27/ 17	442/ 304/ 288
	不满足用户意图/ 满足用户意图的数据传输	198/0	50/0	46/4	1/10	16/3	219/ 17
Google Play 应用	AppIntent (静态分析/ 符号化执行/ 演示案例)	98/ 43/ 43	0/0/0	36/ 15/ 15	10/ 10/ 10	9/8/8	140/ 70/ 70

	不满足用户意图/ 满足用户意图的数据传输	24/0	0/0	0/13	1/9	1/7	26/29

表格 1. 检测到的传输敏感数据的应用。表格中的第一部分是750个恶意软件样本的数据，第二部分是1000个Google Play上热门应用样本的测试结果。对每个样本集，第一行的内容是检测到的敏感数据类型。第二行的数据是AppIntent各个阶段发现的敏感数据传播应用数，第三行是通过AppIntent最终确定的符合用户意图和不符合用户意图的应用数。

如表格 1所示，静态污点分析从这两个数据组中检测出582（442+140）个可能包含敏感数据传输的实例。经过分析，我们发现其中有164个为误报，在我们的AppIntent系统中，我们通过符合化执行去除了这些误报。随后，利用符号化执行提取的应用程序输入，AppIntent成功的对358（288+70）个应用程序生成了演示案例。其中，245（219+26）个应用中发现含有不符合用户意图的数据传输行为。我们注意到，即使是在热门免费应用中，用户隐私泄露的现象依旧存在，这一现象主要发生在一些社交网络服务或者含有内嵌广告模块的应用中。其中，泄露用户的短信和联系人信息的行为都发生在社交网络服务中。此外，我们也发现，由于恶意的数据泄露行为可以隐藏在正常的数据传输背后去欺骗软件安全检查，因此恶意应用程序中可能既包含符合用户意图的数据传输，也包含不符合用户意图的数据传输。例如，我们发现了一个将自己伪装成短信应用的程序，但在背地里，它偷偷的在未得到用户允许的情况下将用户的联系人信息从手机上传送出去。

隐私泄漏的敏感数据源问题：

早在研究人员把敏感数据传播作为隐私泄漏判断依据的同时，“何为敏感数据”这个问题就已经存在了。早期的移动平台隐私泄漏检测方法把固定格式的API函数作为定义敏感数据的标准，例如，在安卓隐私泄漏检测方面，研究人员通常将安卓系统管控的隐私数据（如IMEI，地理位置、短信等）作为敏感数据源。但是随着用户越来越依赖移动终端管理其隐私数据，敏感数据的边界正在被不断拓展，而传统的隐私泄漏检测技术中对敏感数据源的定义已经无法覆盖这些新的敏感数据。

德国达姆施塔特工业大学Siegfried Rasthofe等人在2014年时提出[20]，现有方法在确定敏感数据源时采用的API函数列表是不完整的，并提出通过机器学习获取更详细的敏感数据访问API，提高隐私泄漏检测的覆盖率。

我们提出的UIPicker系统[21]和普渡大学的SUPOR系统[22]认为，单纯检测系统定义的API是不足以覆盖所有的隐私泄露源的，其问题主要来源于由应用程序管控的用户输入的隐私数据，其中包括但不限于用户的个人资料信息如账户密码，用户输入的地理位置信息，以及银行卡等支付类信息。

尽管用户输入的隐私数据面临着重大的安全威胁，如何保护这类隐私数据却具有很大的挑战性。与系统管控的隐私数据不同，用户输入隐私数据无法通过某些特定的API函数进行标注，而是依赖于需要针对界面的语义信息以及上下文进行解析。在TaintDroid等系统中，将所有用户输入标记为隐私数据，这样明显覆盖太广，并且将很多常规数据纳入了不必要的保护范畴，增加了系统分析工作量以及性能。

为了保护用户输入的敏感数据免于各类情况的泄露，自动化地标识这类数据变得非常必要以及紧迫。然而由于该类数据缺乏固定的结构，使得识别工作非常具有挑战性。为了解决这个问题，UIPicker和SUPOR发现在应用当中，很多与隐私输入相关的界面元素均在界面自身，以及界面描述文件当中通过丰富的语义信息有着较为明确的表述。在这样一个观察基础之上，这两个系统通过结合自然语言处理，机器学习，以及程序分析等技术，利用界面语义信息识别由用户输入的敏感信息源。

隐私类别	系统管控隐私API (#应用个数)	含有敏感属性标签的界面元素 (#应用个数)
账户凭据信息&用户资料	4,900	5,330
地理位置信息	15,221	2,883

金融支付类信息	0	1,318
总计	15,632	6,179

表格2：敏感信息源数量。表中第二列揭示了包含系统管控隐私的应用个数，第三列为含有敏感属性标签的用户输入隐私个数。

如表格2所示，通过对Google Play上17425个应用的分析，我们发现其中有六千多个应用使用了用户输入的隐私数据。在传统使用系统管控隐私API的系统中，受限于隐私数据的识别能力，无法对此类隐私进行有效保护。

在经历了系统管控隐私数据、用户输入隐私数据之后，我们不禁会想，下一个敏感信息源是什么？事实上，能够推测用户隐私的敏感信息远不止上文提到的这些。在移动平台上，印第安纳大学的Xiaofeng Wang教授[23]曾经利用安卓应用的网络数据统计、应用程序声音开关等信息，利用旁道攻击的方式获取用户的身份、位置、身体状况等敏感信息。类似的工作还揭露出用户的其他敏感信息也可以被恶意工作所捕获。面对这一类型的攻击，目前尚没有成熟的方法可以进行有效检测和防御。

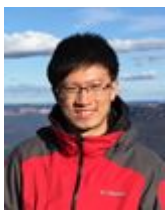
总结与展望

在移动互联网飞速发展的背景下，为增强对用户隐私信息的保护能力，还可以从两个角度开展研究工作：

第一，用户隐私的定义范围还可以做进一步扩展。在传统系统定义用户隐私和本文提到的用户输入用户隐私以外还存在众多用户数据可以被用于推测用户隐私。现有工作虽然揭示了相关问题的存在，但是并未达成对此类隐私的有效分析和检测。因此，需要进一步深化对用户隐私的分析和保护。

第二，对于隐私泄漏的用户感知问题，目前已经实现了辅助分析人员判断敏感数据传输的用户感知度，但是在大规模应用程序分析时，此类方法无法做到自动化判断。目前仍有大量应用收集各类用户隐私，其中大部分不经用户同意。因此，仍需要加强对应用商城运营商与程序开发商（者）的监督管理。同时从技术角度讲，也需要研究用户感知度评估的自动化方法。

本文部分内容引用课题组撰写的相关学术论文和技术报告。



杨哲慜
复旦大学软件学院讲师，主研领域：系统软件与系统安全。yangzhemin@fudan.edu.cn

杨 珉

国家973项目首席科学家，复旦大学软件学院副教授、博士生导师，主研领域：系统软件与系统安全。m_yang@fudan.edu.cn

参考文献

[1] Smartphone Vendor Market Share, 2015 Q2. http://www.idc.com/prodserv/smartphone-market-share.jsp
[2] 中国网民权益保护调查报告（2015）. http://www.cac.gov.cn/2015-07/22/c_1116002040.htm
[3] C. Gibler, J. Crussell, J. Erickson, and H. Chen. Androidleaks: automatically detecting potential privacy leaks in android applications on a large scale. In Proceedings of the 5th international conference on Trust and Trustworthy Computing, TRUST'12, pages 291–307, 2012
[4] Chin, E., Felt, A. P., Greenwood, K., and Wagner, D. Analyzing Inter-Application Communication in Android. In Proc. of the Annual International Conference on Mobile Systems, Applications, and Services (2011).
[5] Z. Yang and M. Yang. Leakminer: Detect information leakage on android with static taint analysis. In Software Engineering (WCSE), 2012 Third World Congress on, pages 101–104, 2012.
[6] L. Lu, Z. Li, Z. Wu, W. Lee, and G. Jiang. Chex: statically vetting android apps for component hijacking vulnerabilities. In CCS 2012, pages 229–240, 2012.
[7] C. Fritz. FlowDroid: A Precise and Scalable Data Flow Analysis for Android. Master's thesis, TU Darmstadt, July 2013.
[8] W. Enck, P. Gilbert, B. gon Chun, L. P. Cox, J. Jung, P. McDaniel, and A. Sheth. Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones. In R. H. Arpaci-Dusseau and B. Chen, editors, OSDI, pages 393–407. USENIX Association, 2010.
[9] R. Xu, H. Sa¨ıdi, and R. Anderson. Aurasium: practical policy enforcement for android applications. In USENIX Security 2012, Security'12, pages 27–27, Berkeley, CA, USA, 2012. USENIX Association.
[10] Z. Yang, M. Yang, Y. Zhang, G. Gu, P. Ning, and X. S. Wang. Appintent: Analyzing sensitive data transmission in android for privacy leakage detection. In Proc. of ACM CCS'13, 2013.
[11] P. Hornyack, et al., "These aren't the droids you're looking for: retrofitting android to protect data from imperious applications," In Proc. CCS, 2011.
[12] ENCK, W., ONGTANG, M., AND MCDANIEL, P. On Lightweight Mobile Phone Application Certification. In Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS) (November 2009).
[13] ONGTANG, M., MCLAUGHLIN, S., ENCK, W., AND MCDANIEL, P. Semantically Rich Application-Centric Security in Android. In Proceedings of the 25th Annual Computer Security Applications Conference (ACSAC) (2009).
[14] X. Zhang, A. Ahlawat, and W. Du. AFrame: Isolating advertisements from mobile applications in Android. In Proceedings of the 29th Annual Computer Security Applications Conference, 2013.
[15] P. Pearce, A. P. Felt, G. Nunez, and D. Wagner. Addroid: Privilege separation for applications and advertisers in android. In 7th ACM Symposium on Information,
[16] S. Shekhar, M. Dietz, and D. S. Wallach. AdSplit: Separating smartphone advertising from applications. In 21th Usenix Security Symposium, 2012.
[17] P. Gilbert, B.-G. Chun, L. P. Cox, and J. Jung. Vision: automated security validation of mobile apps at app markets. In Proceedings of the second international workshop on Mobile cloud computing and services, MCS '11, pages 21–26, New York, NY, USA, 2011. ACM.
[18] L. Lu, V. Yegneswaran, P. Porras, and W. Lee. Blade: an attack-agnostic approach for preventing drive-by malware infections. In Proc. CCS, pages 440–450, 2010.
[19] K. Z. Chen, N. Johnson, V. D'Silva, S. Dai, K. MacNamara, T. Magrino, E. X. Wu, M. Rinard, and D. Song. Contextual policy enforcement in android applications with permission event graphs. In

Proc. NDSS, 2013.
[20] S. Arzt, S. Rasthofer, and E. Bodden, “Susi: A tool for the fully automated classification and categorization of android sources and sinks,” 2013.
[21] Y Nan, M Yang, Z Yang, et al. Uipicker: User-input privacy identification in mobile applications[C]//USENIX Security. 2015: 993-1008.
[22] J Huang, Z Li, X Xiao, et al. SUPOR: precise and scalable sensitive user input detection for android apps[C]//Proceedings of the 24th USENIX Conference on Security Symposium. USENIX Association, 2015: 977-992.
[23] X Zhou, S Demetriou, D He, et al. Identity, location, disease and more: Inferring your secrets from android public resources[C]//Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013: 1017-1028.